(T)EE2026
Digital Fundamentals
Gate-level Design & Minimization

Prof. Massimo Alioto

Dept of Electrical and Computer Engineering

Email: ***massimo.alioto@nus.edu.sg***

Outline

- Gate-level logic design
- Karnaugh map
- Boolean function simplification using K-Map
- Gate-level implementation

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)

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

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

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 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”.

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 function on K-map:

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

A \ B	0	1
0	0	1
1	1	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”)

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

Boolean function in K-map (cont.)

Represent the following function on K-map:

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

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

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

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

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

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 function in K-map (cont.)

What about Boolean function in non-canonical form?

Example-1:

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

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

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

Or $\bar{A}B$ → 01, $C = 0 \text{ or } 1$

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

Example-2:

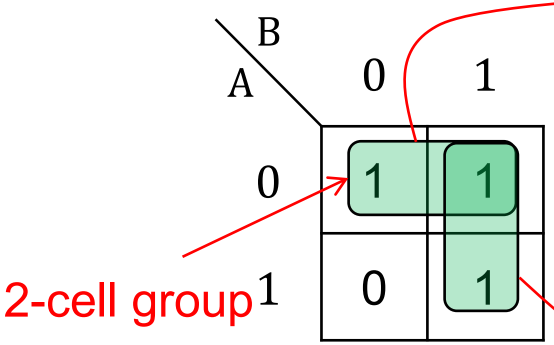
$F = A + \bar{A}\bar{B}CD + B\bar{C}\bar{D}$

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

Boolean function simplification using K-map

Boolean function (SOP) simplification using K-map

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



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

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

MSOP:

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

Alternatively,

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

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

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

Three-variables:

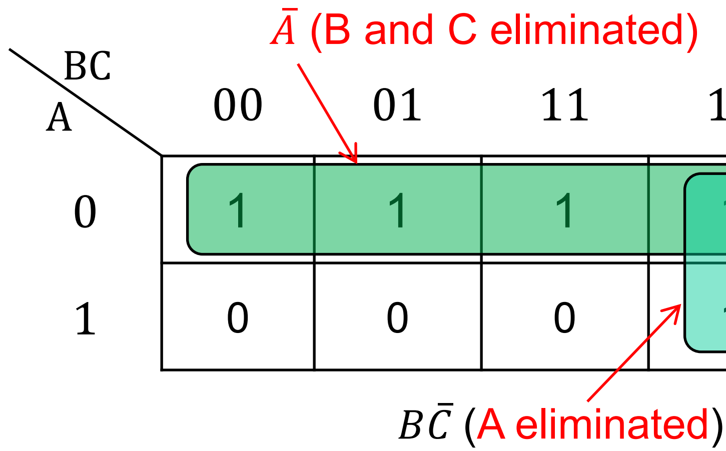
$$F = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}BC + \bar{A}B\bar{C} + AB\bar{C}$$

$$\downarrow$$

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

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

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



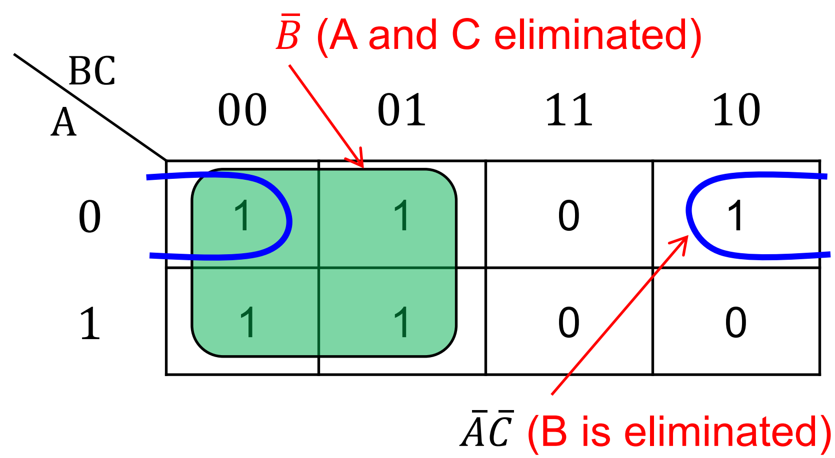
$$F = \bar{A}\bar{B}C + A\bar{B}\bar{C} + A\bar{B}C + \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C}$$

$$\downarrow$$

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

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

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



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

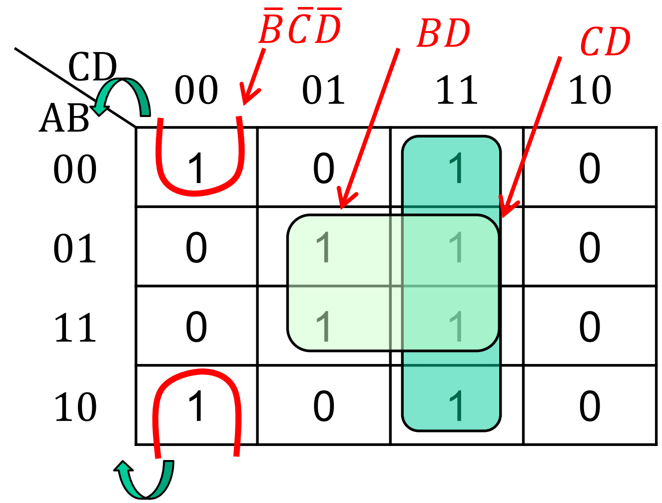
Minimization (SOP) using K-Map (cont.)

Four-variables:

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

$$\downarrow$$

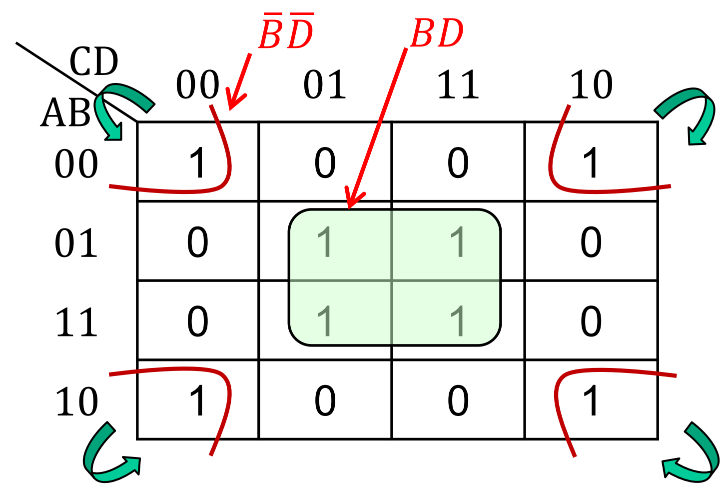
$$F = \bar{B}\bar{C}\bar{D} + BD + CD$$



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

$$\downarrow$$

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



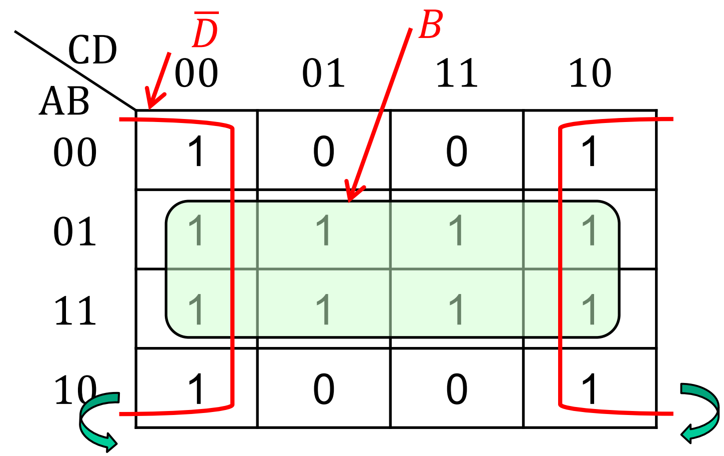
Minimization (SOP) using K-Map (cont.)

Four-variables:

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

$$\downarrow$$

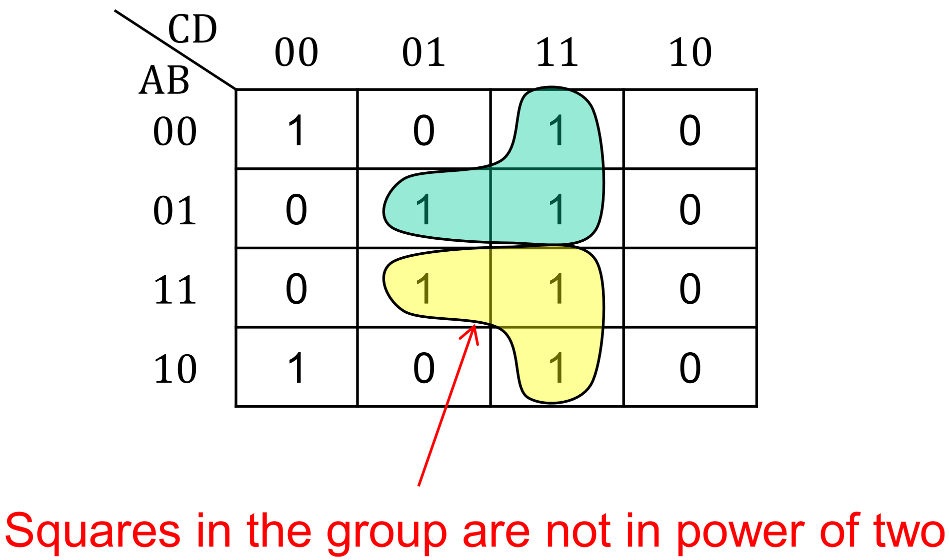
$$F = B + \bar{D}$$

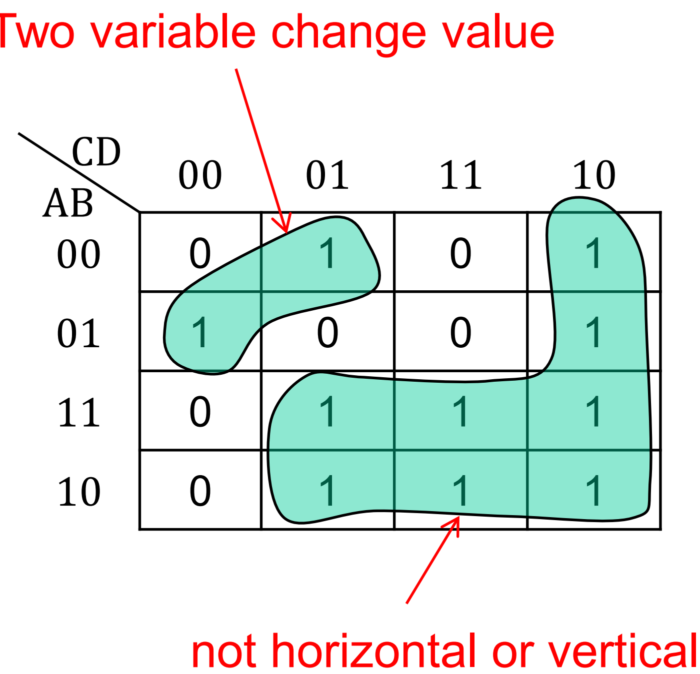


Grouping rules:

- 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





Don’t-care condition

- So far we assume that all combination of the input variables of a Boolean function are valid (for example, 3-variable Boolean function has 8 different input combinations that makes the function equal to 0 or 1)
- There are applications in which some variable combinations never appear.
- One of such examples is the BCD code
  - 4-bit BCD code can have 16 values
  - However, 1010 – 1111 are never used, or $A\bar{B}C\bar{D}$, $A\bar{B}CD$, $AB\bar{C}\bar{D}$, $AB\bar{C}D$, $ABC\bar{D}$, and ABCD never occur
- 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”.

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

Minimization with don't-care conditions

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

$$\downarrow$$

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

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

*Treat X = 1 and group the squares as usual

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

Boolean function in POS:

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

$$\downarrow$$

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

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

Minimal SOP (MSOP)

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?

Minimal SOP (MSOP) – Prime implicant

- ***Prime implicant***
  - An implicant that cannot be combined with another term to eliminate a variable

Example-1:

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

$$F = AB + ABC + BC$$

Prime implicants

Example-2:

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

$\bar{A}B\bar{D}$ $\bar{A}B$

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

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

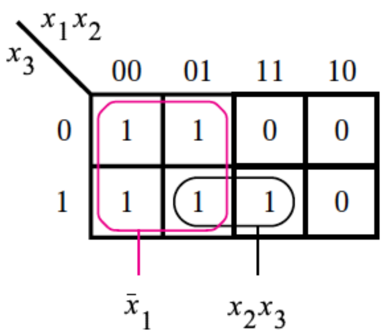
Identifying prime implicants

- A single “1” on a K-map is a prime implicant if is not adjacent to any other 1 of the function.
- Two adjacent “1”s represent a prime implicant, provided that they are not within a rectangle of 4 or more squares containing “1”s.
- Four “1”s (that are an implicant) are a prime implicant if they are not within a group of 8 squares containing “1”s

Basically, implicant is prime if it cannot be enclosed within a larger square/rectangle (as per K-map rules)

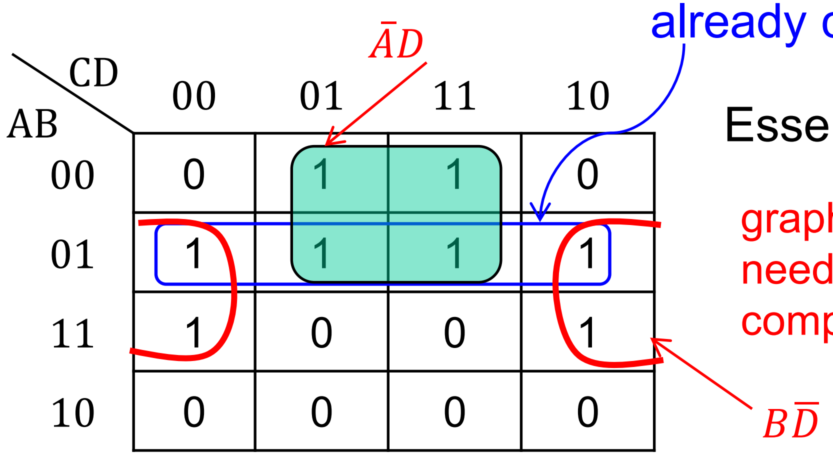
Essential prime implicant

- ***Essential prime implicant***
  - A 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)

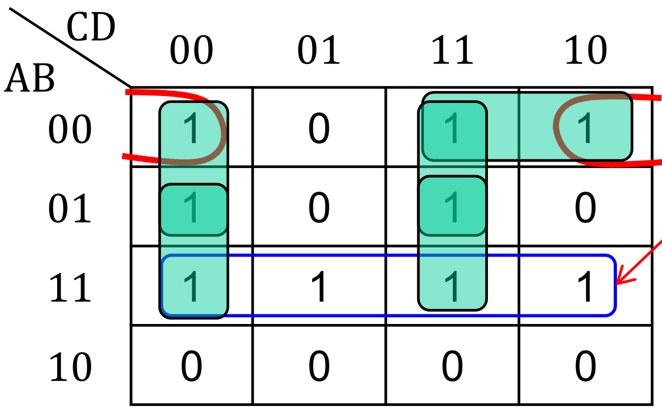
Minimal SOP Expression (MSOP)

- What is MSOP?
  - It contains a minimal number of literals and terms
  - All essential prime implicants must be included in MSOP

- Determination of MSOP
  - Finding all of the *prime implicants* of the function
  - Select essential prime implicants (those with “1”s that have only been grouped once
  - Finding a minimal subset of these prime implicants that covers all of the *minterms* of the function

Obtaining MSOP - examples

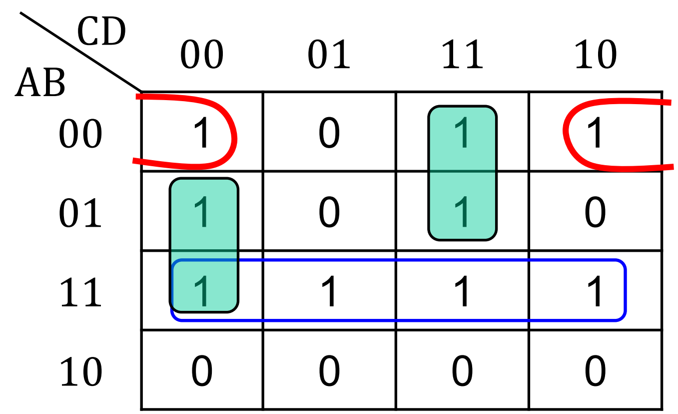
Example – 1:



Essential
Prime implicant

All implicants including
one essential prime implicant

Select the essential prime implicant
with minimum set of prime implicants



Obtaining MSOP - examples (cont.)

Example – 2:

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

Obtaining MSOP - examples (cont.)

Example – 3:

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

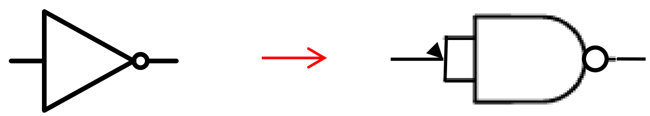
AB \ CD	00	01	11	10
00	1	0	0	1
01	1	1	1	1
11	1	0	1	0
10	1	0	1	1

Gate-level implementation

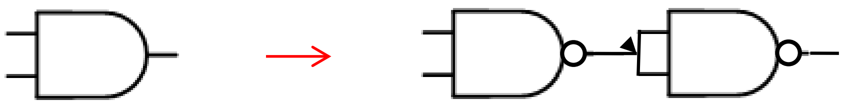
- NAND only implementation
- NOR only implementation

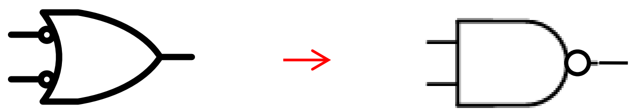
NAND only implementation

Logic operation: | **NAND implementation:**



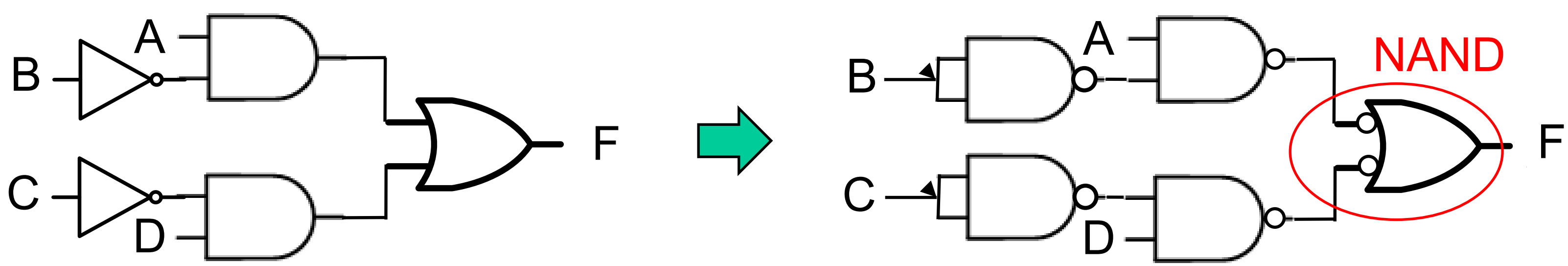
$$\overline{x \cdot x} = \overline{x}$$





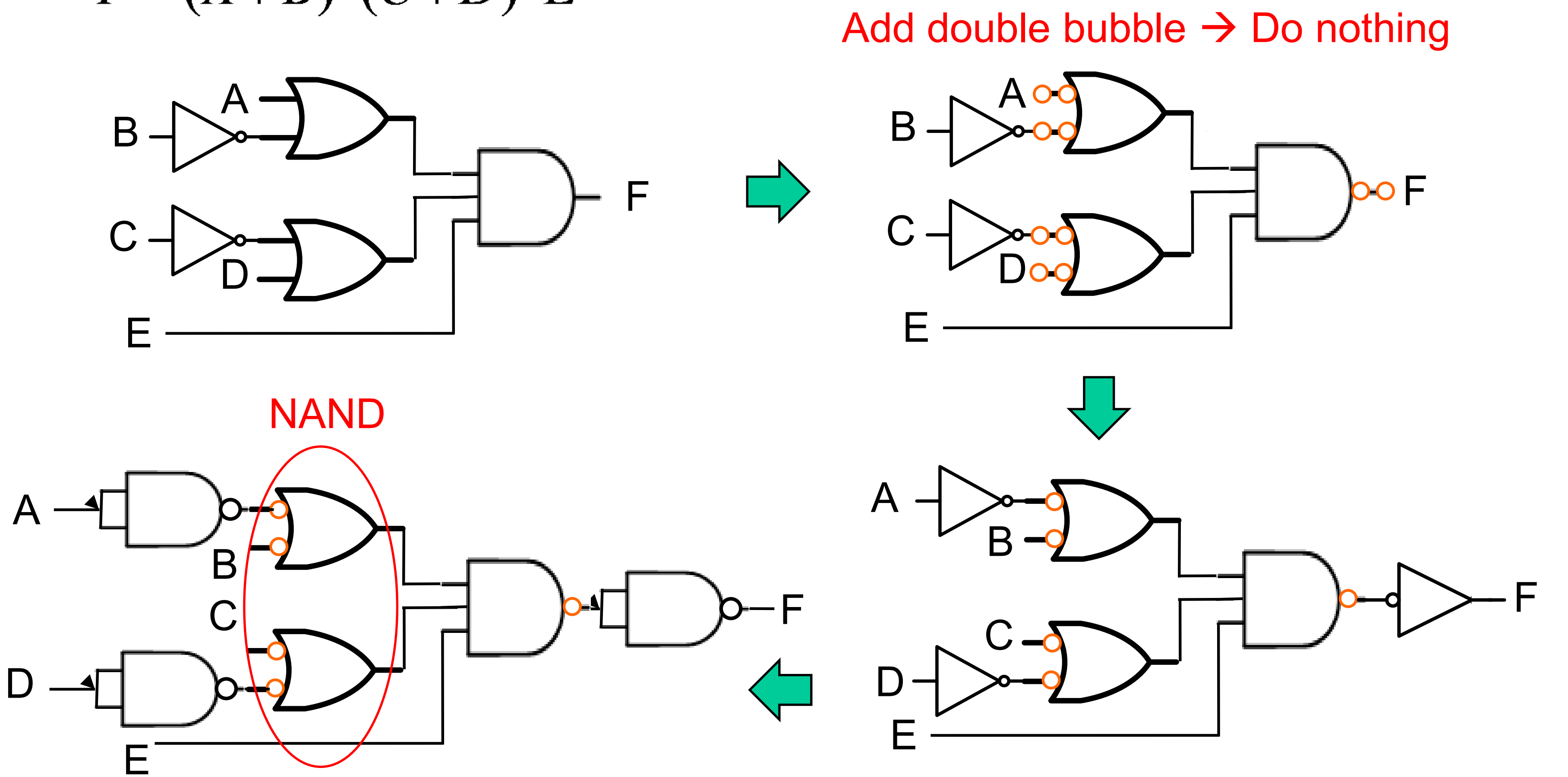
$$F = A\overline{B} + \overline{C}D$$

- Replace the OR gate with NAND gate and balance the bubble
- Replace the inverter with NAND gate



NAND only implementation – cont.

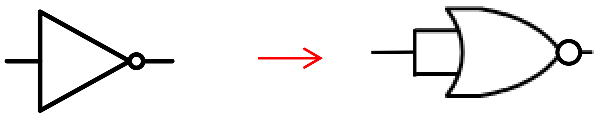
$$F = (A + \overline{B}) \cdot (\overline{C} + D) \cdot E$$



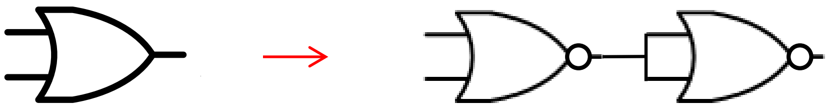
Replace inverters with NAND gates,
push bubbles

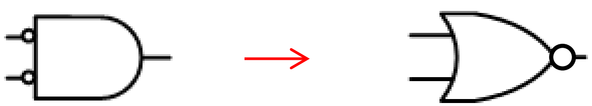
NOR only implementation

Logic operation: **NOR implementation:**



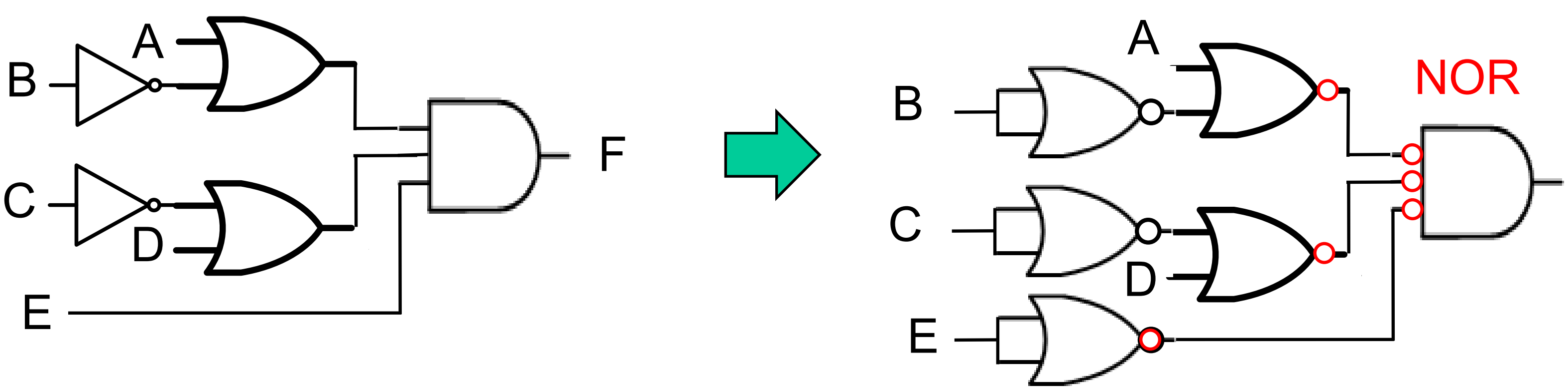
$\overline{x+x} = \overline{x}$





$F = (A+\overline{B})\cdot(\overline{C}+D)\cdot E$

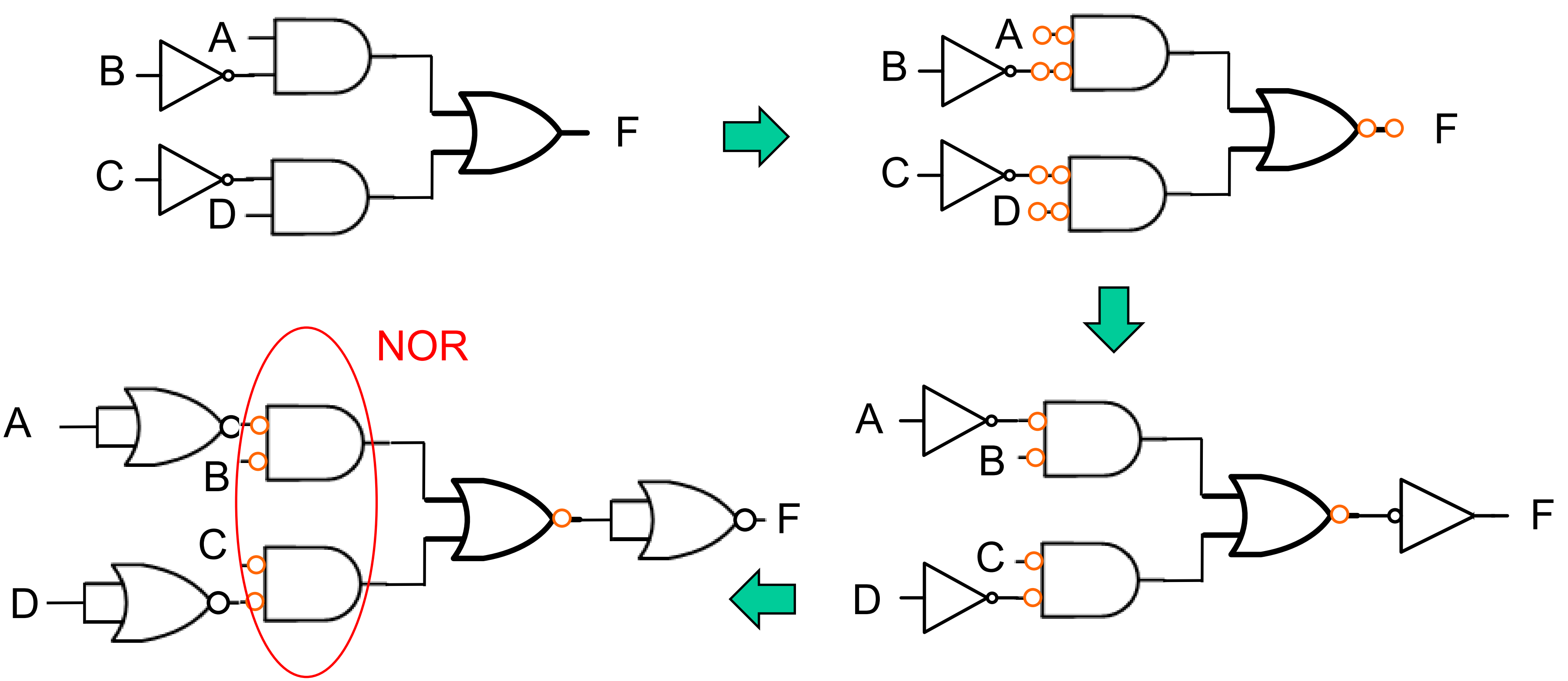
- Replace the AND gate with NOR gate and balance the bubble
- Replace the inverter with NOR gate



NOR only implementation – cont.

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

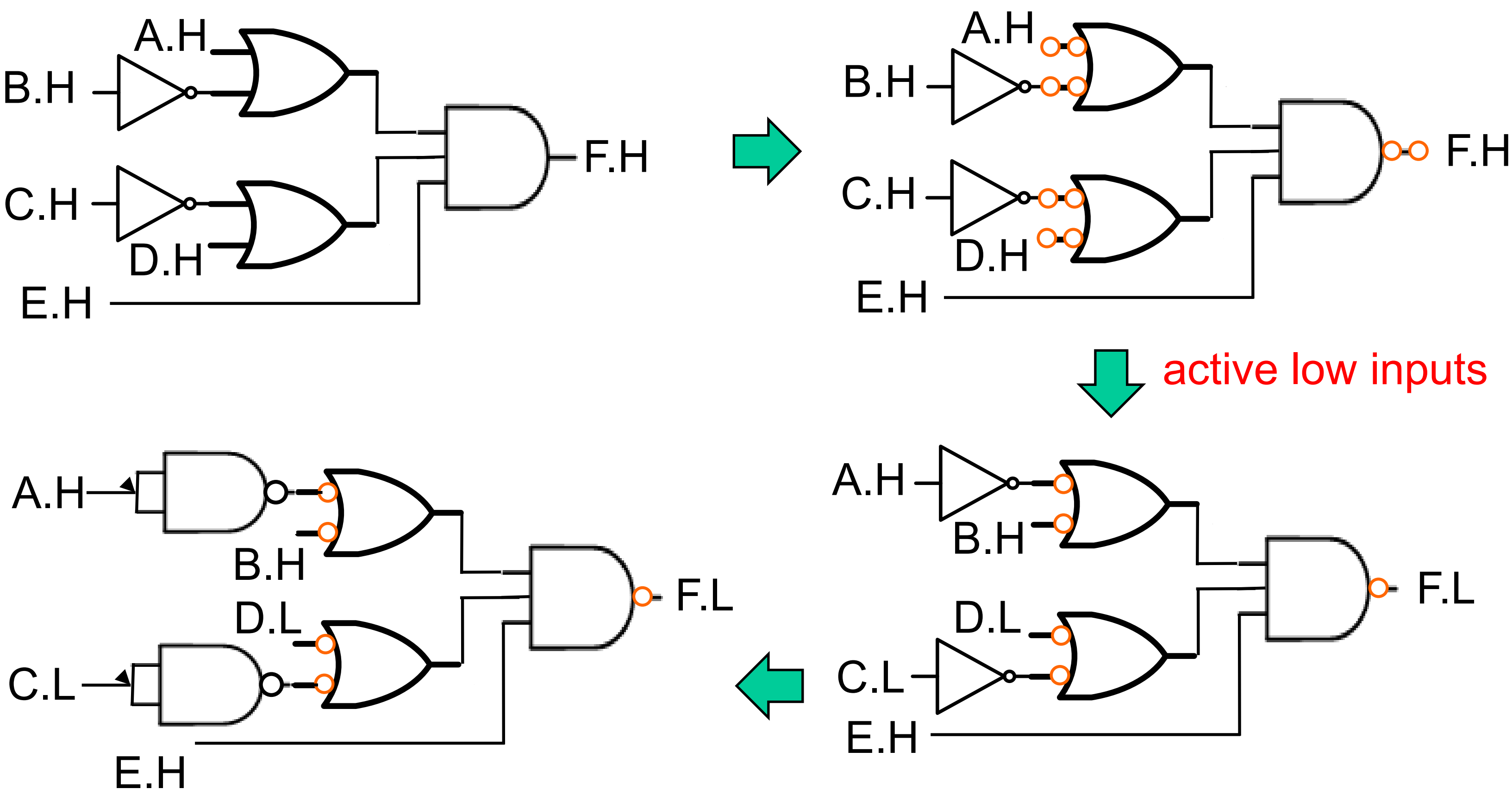
Add double bubble → Do nothing



Replace inverters with NOR gates,
push bubbles

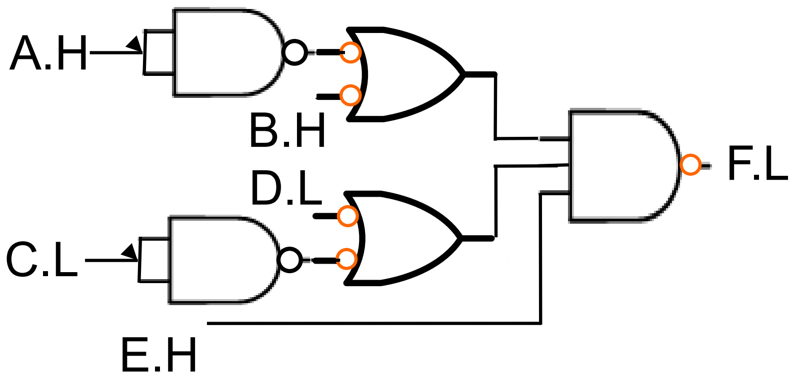
NAND only Implementation with Mixed Logic

$F = (A + \overline{B}) \cdot (\overline{C} + D) \cdot E$ (where C, D and F are active low)



NAND only implementation with Mixed Logic – cont.

Write the Boolean function implemented by the circuit below and express F in positive logic.



Summary

- Karnaugh map
- Boolean function simplification using K-map
  - SOP simplification
  - POS simplification
  - Don’t-care condition
  - Minimal SOP (MSOP) and POS (MPOS)
- Gate-level implementation
  - NAND only
  - NOR only

Suggestions for Self-Improvement

- In addition to the tutorials, you may want to practice with questions in chapter 2 of the textbook

  (see IVLE Workbin)

  – QUESTIONS (highlighted in yellow): harris_exercises_chapter2odd.pdf

  – ANSWERS: harris_solutions_chapter2odd.pdf