

UNIT 7

MULTI-LEVEL GATE CIRCUITS



Spring 2011

Multi-Level Gate Circuits

- **Contents**
 - Multi-level gate circuits
 - NAND and NOR gates
 - Two-level NAND- and NOR-gate circuits
 - Multi-level NAND- and NOR-gate circuits
 - Circuit conversion using alternative gate symbols
 - Design of two-level, multiple-output circuits
 - Multiple-output NAND and NOR circuits
- **Reading**
 - Unit 7

Objectives

□ **Recap logic design steps:**

1. Translate the word description into a switching function (Unit 4)
2. Simplify the function
 - Boolean algebra (Units 2&3)
 - **Karnaugh map (Unit 5)**
 - Quine-McCluskey (Unit 6)
 - ... etc

3. Realize it using available logic gates
 - AND-OR, OR-AND (Unit 2)
 - **NAND-NAND, NOR-NOR (Unit 7)**
 - ... etc

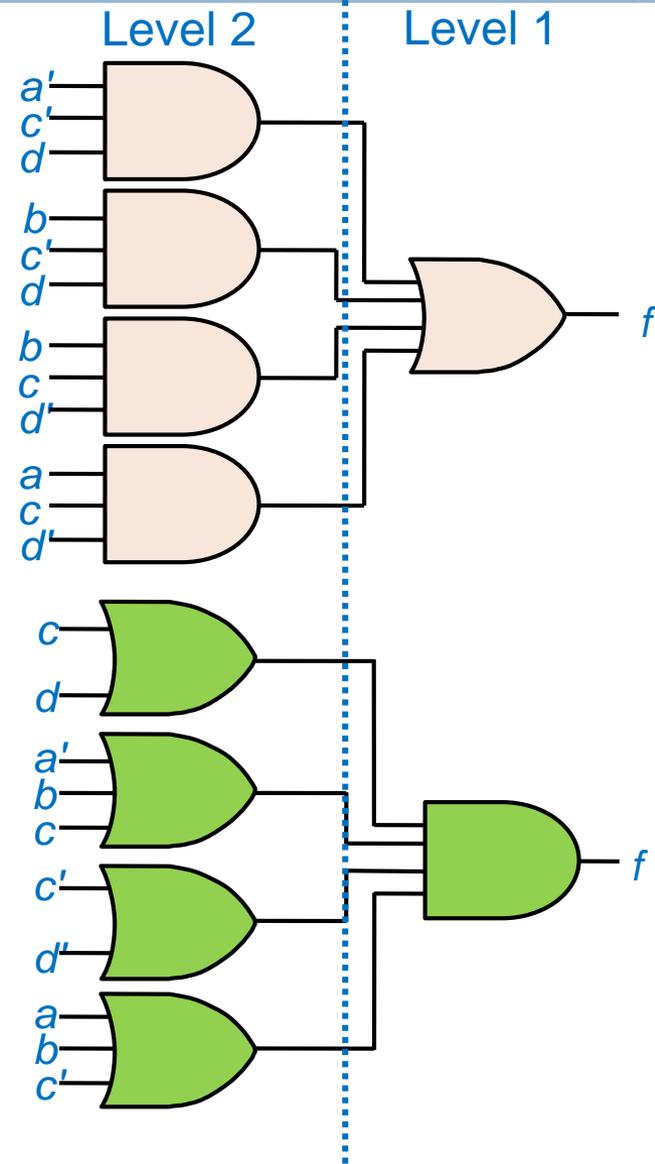
□ **Design/convert two-/multi-level single-/multiple-output circuits using Karnaugh maps and NAND/NOR/AND/OR gates**

AND-OR & OR-AND Circuits

4

© Iris H.-R. Jiang

- The # of **levels** of gates = the **maximum** # of gates cascaded in series between an **input** & the **output**
 - Do **NOT** count inverters at inputs
 - **Count from the output**
- **Two-level** circuits learned so far:
 - **AND-OR** circuit = **SOP** form
 - One level of AND gates + one OR gate
 - **OR-AND** circuit = **POS** form
 - One level of OR gates + one AND gate



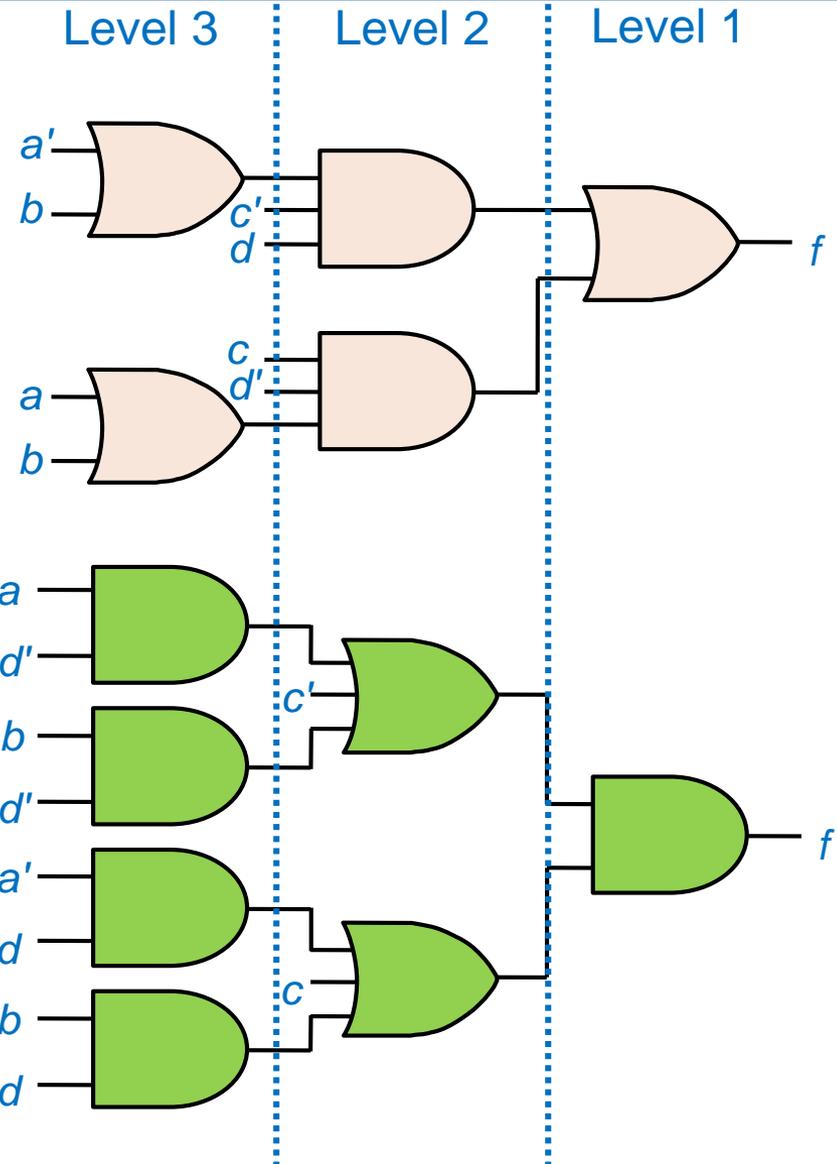
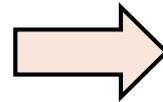
OR-AND-OR & AND-OR-AND Circuits

5

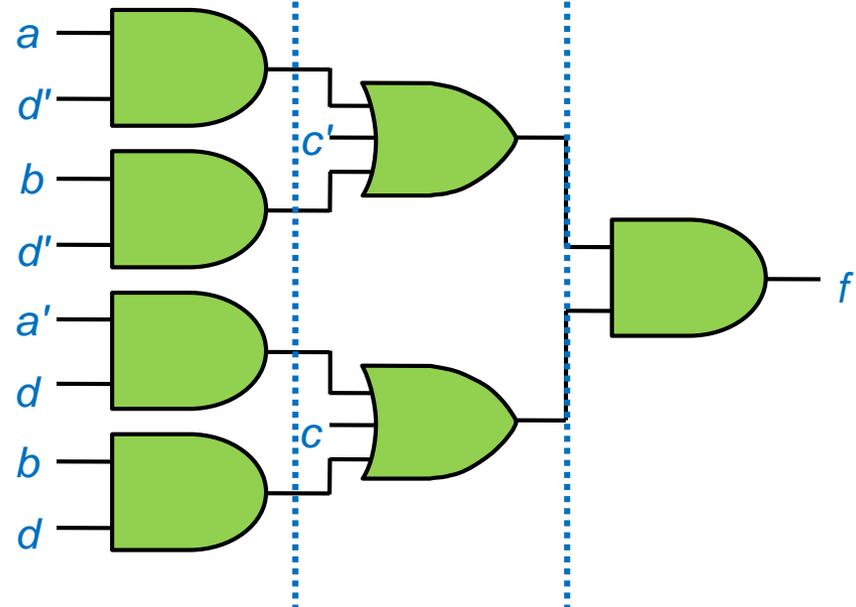
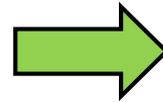
© Iris H.-R. Jiang

Three-level circuits:

OR-AND-OR circuit



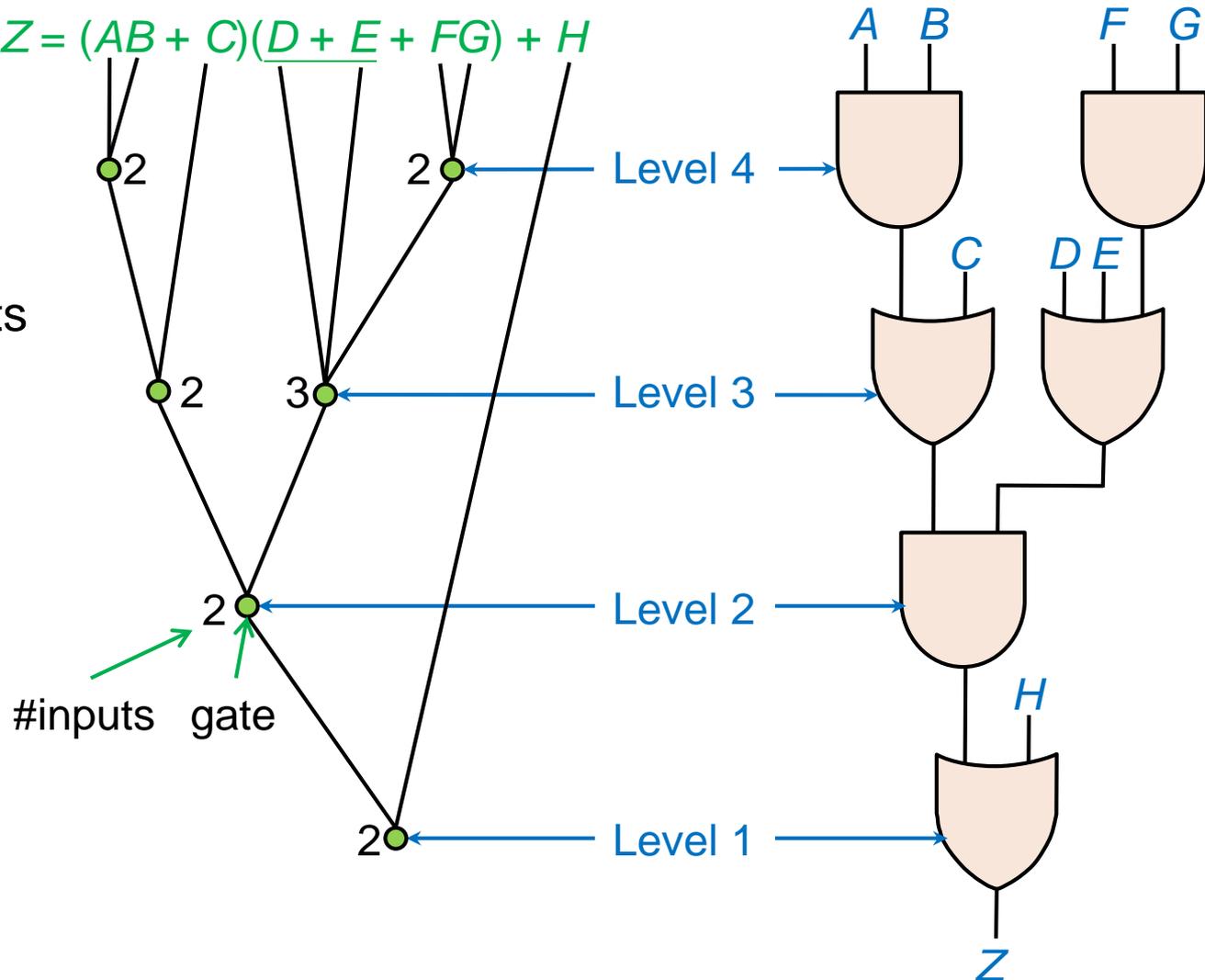
AND-OR-AND circuit



One More Example (1/2)

- How to count the # of level?
- By inspection $Z = (AB + C)(D + E + FG) + H$

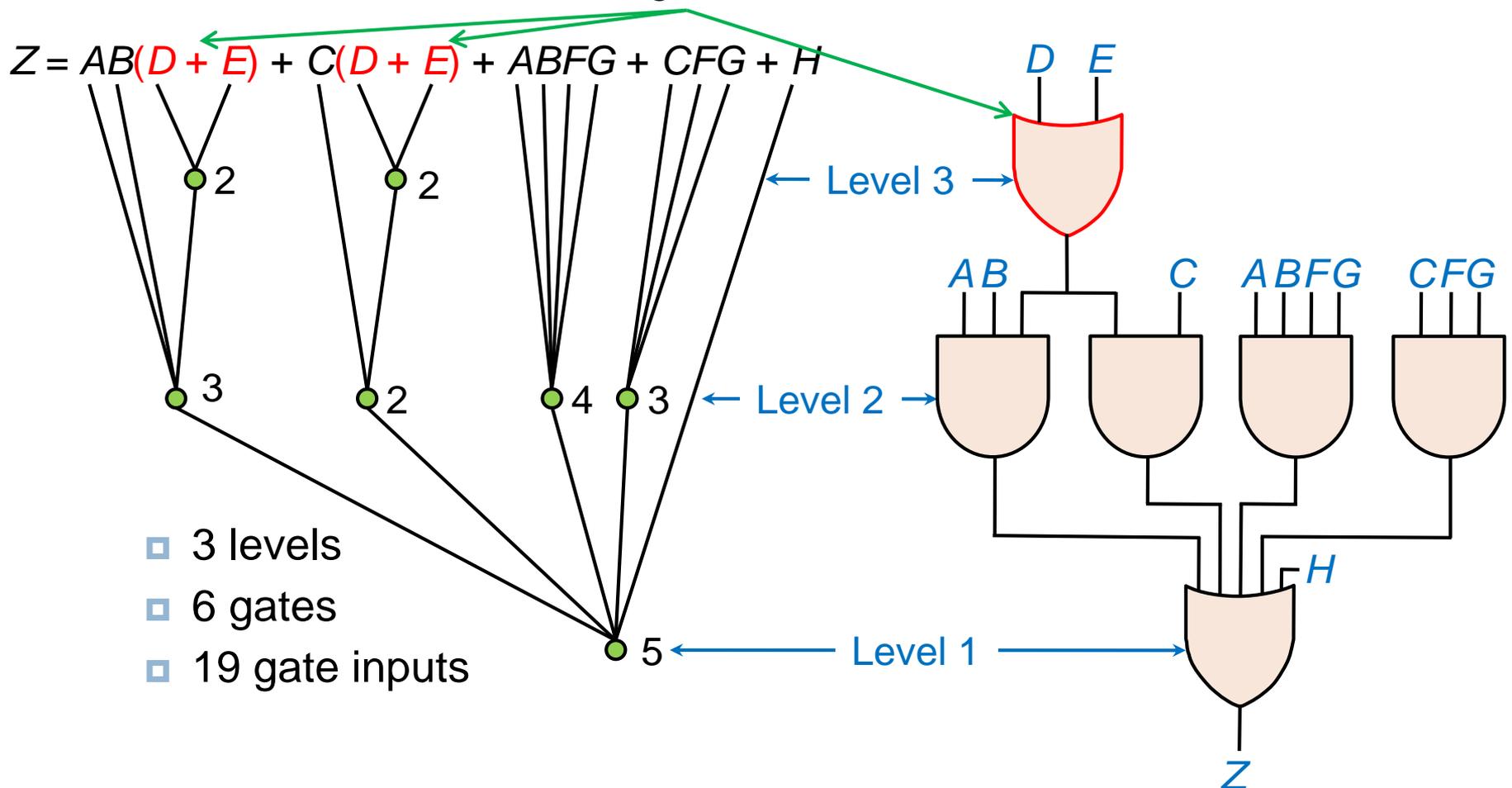
- e.g.,
 - 4 levels
 - 6 gates
 - 13 gate inputs



One More Example (2/2)

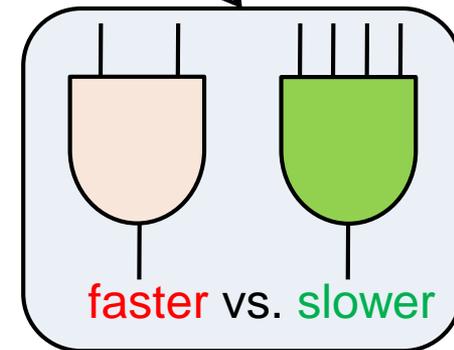
Same function in another realization

Use one gate for both



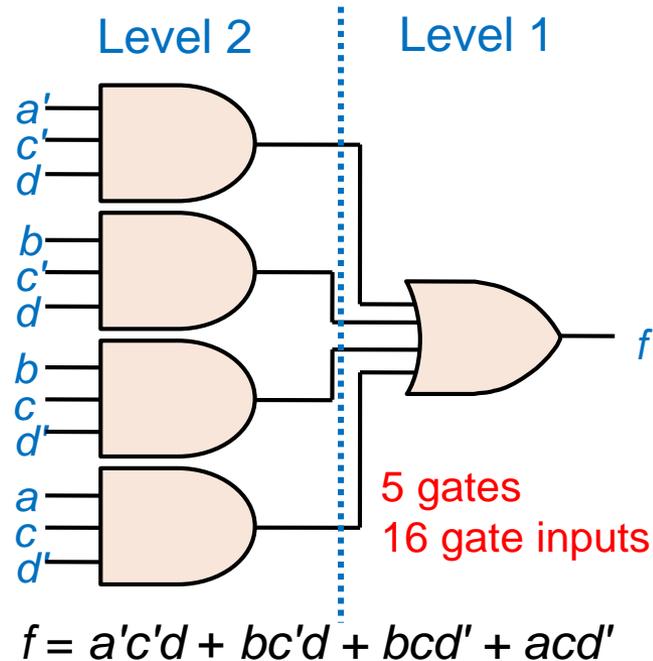
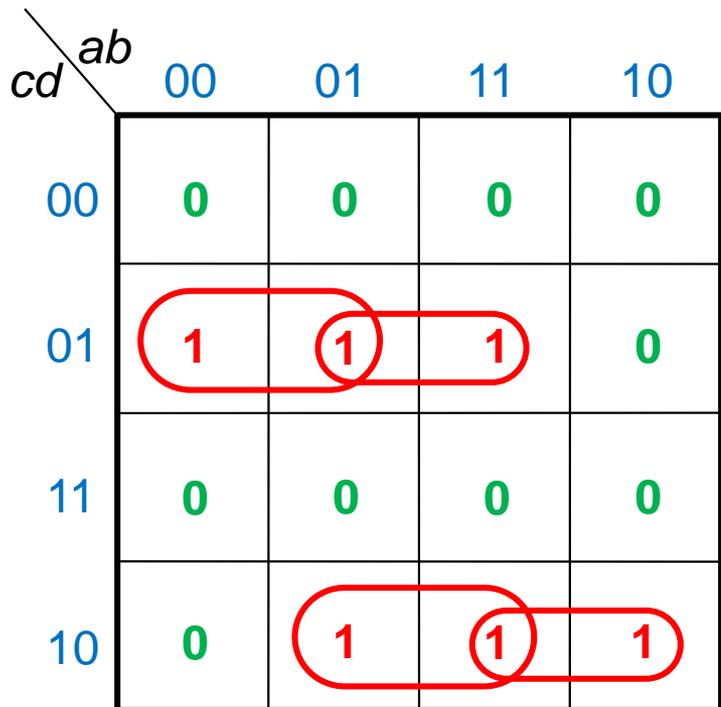
Multi-Level Gate Circuits

- **Q1: Why does the # of level matter?**
- **A1: The # of level affects cost and delay (Trade-off!)**
 - ▣ Cost depends on # of gates & # of gate inputs
 - ▣ Delay depends on # of level & # of inputs for a single gate
 - ▣ # of level $\uparrow \Rightarrow$ cost \downarrow (maybe), circuit delay \uparrow (maybe)
 - ▣ e.g.,
 - $Z = (AB + C)(D + E + FG) + H$
4 levels/6 gates/13 gate inputs
 - $Z = AB(D + E) + C(D + E) + ABFG + CFG + H$
3 levels/6 gates/19 gate inputs
- **Q2: How to change the # of levels?**
- **A2: Factoring or multiplying out (see the next example)**



Multi-Level Design using AND & OR (1/4)

- e.g., find a circuit of AND and OR gates to realize $f(a, b, c, d) = \sum m(1, 5, 6, 10, 13, 14)$
- Sol 1: try two-level **AND-OR** circuit first!
 1. Simplify f by K-map (circle 1's)
 2. Realize it

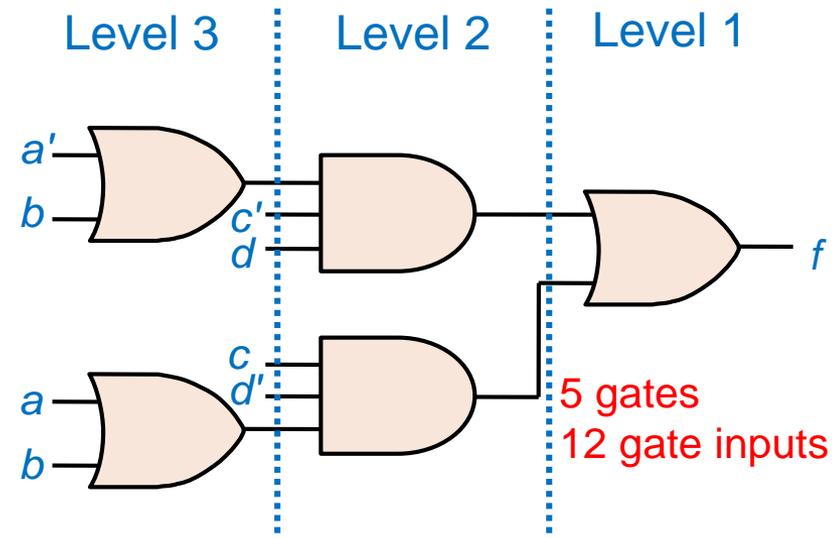


Multi-Level Design using AND & OR (2/4)

10

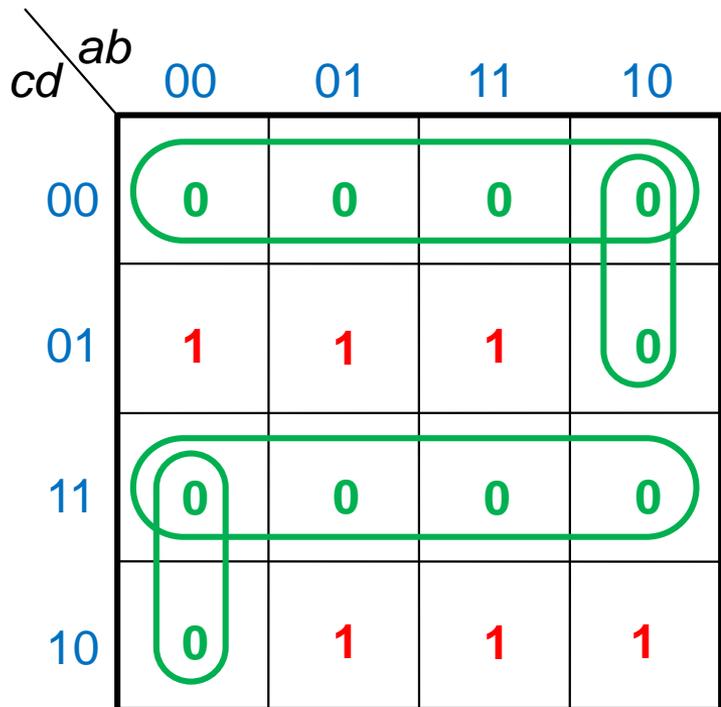
© Iris H.-R. Jiang

- From Sol 1, $f(a, b, c, d) = a'c'd + bc'd + bcd' + acd'$
- Sol 2:
 1. Factoring it yields $f(a, b, c, d) = (a' + b)c'd + (b + a)cd'$
 - Sharing!
 2. Lead to a three-level **OR-AND-OR** circuit
 - Gate inputs ↓; cost ↓; delay ↑

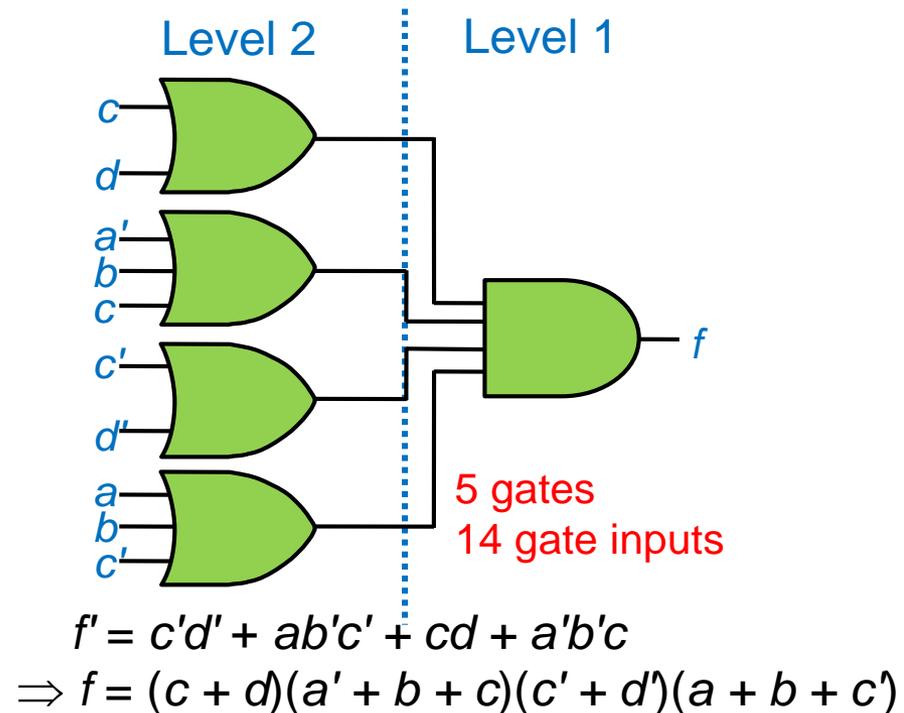


Multi-Level Design using AND & OR (3/4)

- e.g., find a circuit of AND and OR gates to realize $f(a, b, c, d) = \sum m(1, 5, 6, 10, 13, 14)$
- Sol 3: try two-level **OR-AND** circuit
 1. Simplify f by K-map (circle 0's)
 2. Realize it



Multi-level gate circuits



Multi-Level Design using AND & OR (4/4)

□ From Sol 3, $f(a, b, c, d) = (c + d)(a' + b + c)(c' + d')(a + b + c')$

□ Sol 4:

1. Multiplying it out yields

$$f = [c + d(a' + b)][c' + d'(a + b)] \text{ (4-level)}$$

2. Partially multiplying out $d(a' + b)$ and $d'(a + b)$

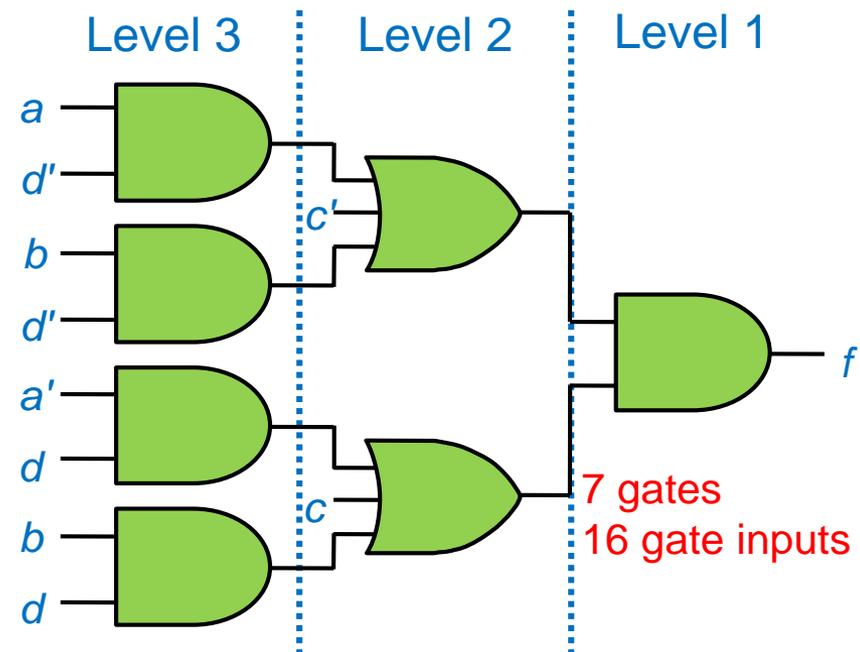
$$f = (c + a'd + bd)(c' + ad' + bd')$$

3. Lead to a three-level **AND-OR-AND** circuit

□ For this case, best solutions:

▣ Two-level: **OR-AND**

▣ Three-level: **OR-AND-OR**



DeMorgan's law

Double inversions

NAND Gate

Double inversions
= 2 cascaded bubbles
= do nothing, $(X')' = X$

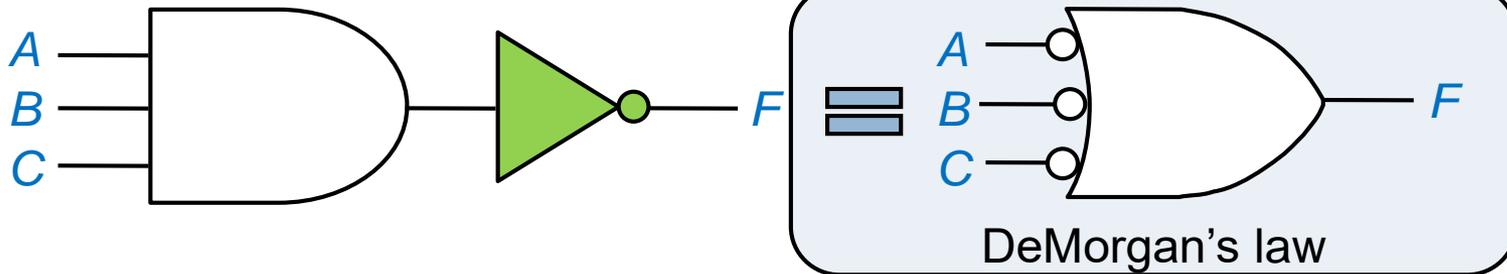
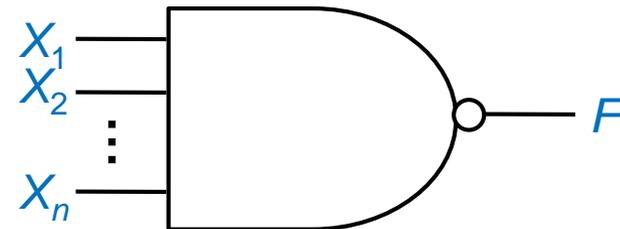
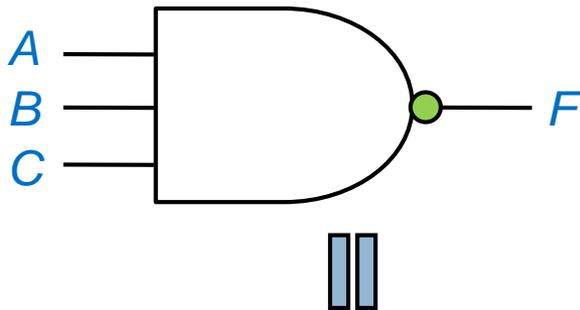
14

© Iris H.-R. Jiang

□ NAND ≡ AND-NOT

- ▣ $F = (ABC)' = A' + B' + C'$ (DeMorgan's law)
- ▣ General: $F = (X_1 X_2 \dots X_n)' = X_1' + X_2' + \dots + X_n'$
- ▣ Symbol

■ Bubble ≡ NOT



NOR Gate

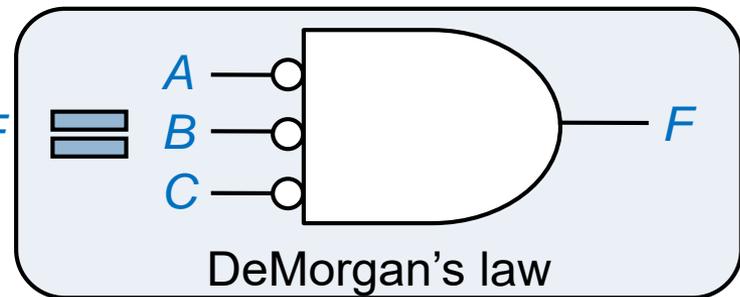
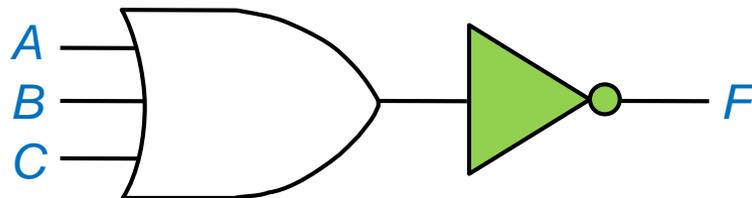
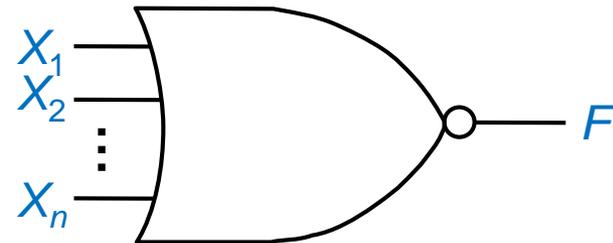
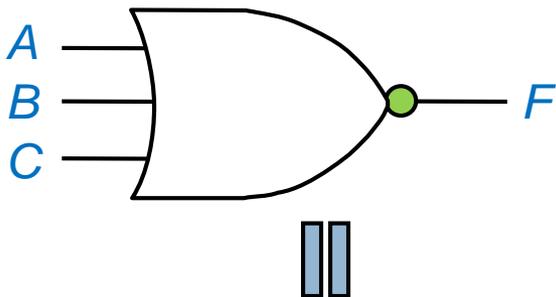
Actually, NAND & NOR are implemented by fewer switch devices and operate faster than AND & OR gates \Rightarrow Low cost, small delay

15

© Iris H.-R. Jiang

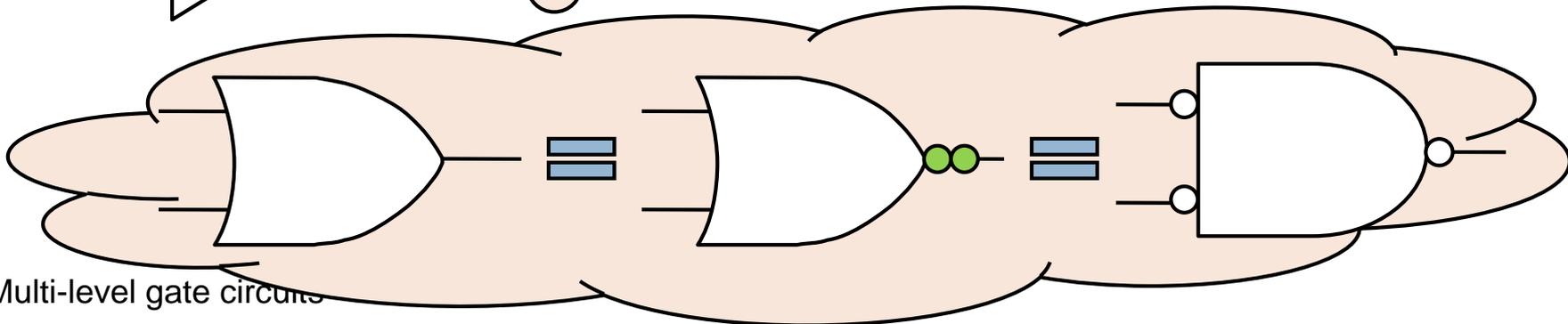
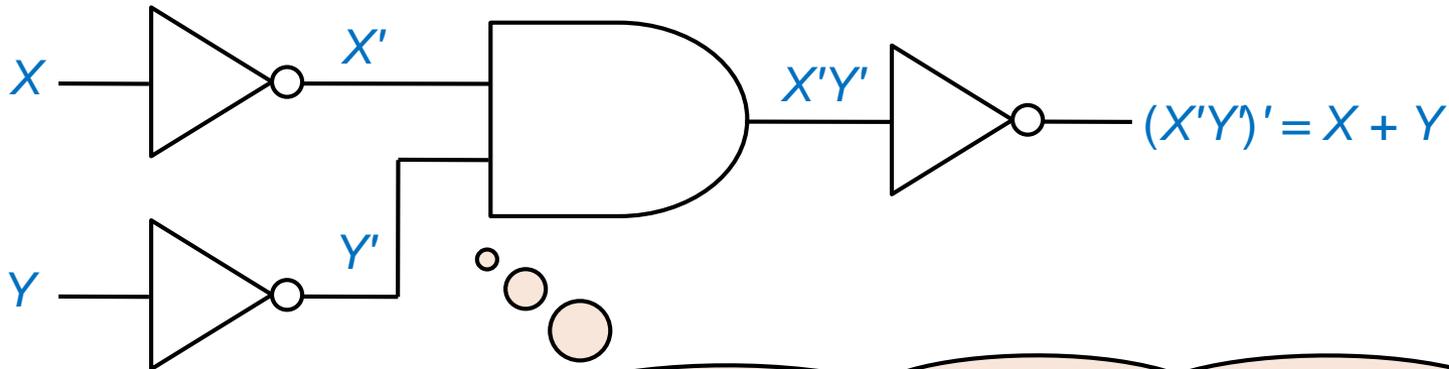
□ NOR \equiv OR-NOT

- $F = (A + B + C)' = A'B'C'$ (DeMorgan's law)
- General: $F = (X_1 + X_2 + \dots + X_n)' = X_1'X_2'\dots X_n'$
- Symbol
 - Bubble \equiv NOT



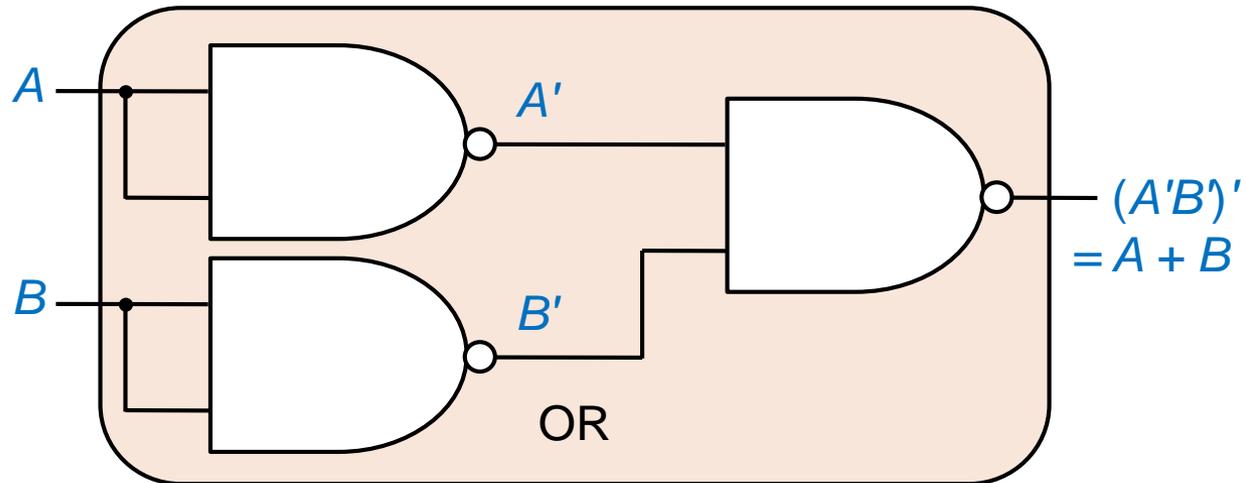
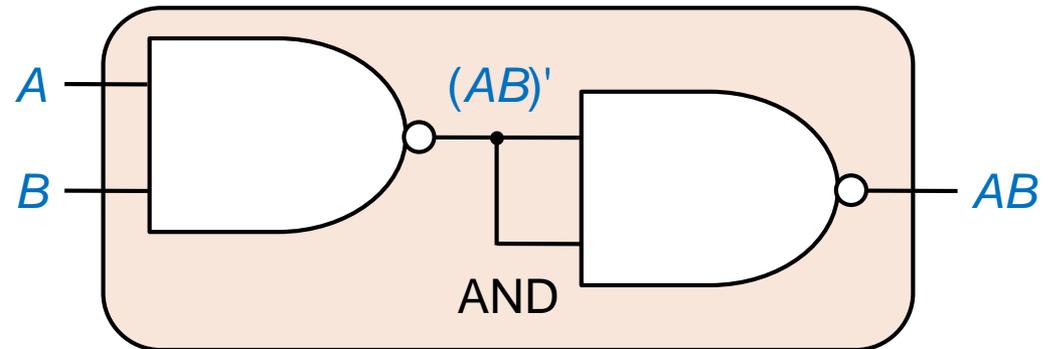
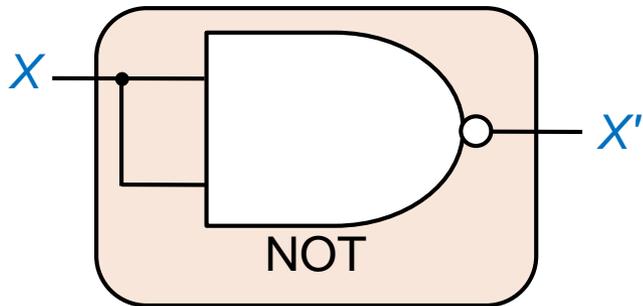
Functionally Complete Set (1/2)

- A set of logic operations is **functionally complete** if **any Boolean function** can be expressed by this set
- e.g., $f = a'b + b'c + c'a$
 - ▣ Q: {AND, OR, NOT} A: Of course, yes!
 - ▣ Q: {AND, NOT}? OR? A: Yes! (**Why?**)
 - $(X + Y) = [X' \cdot Y']' = X + Y$ (DeMorgan's law)



Functionally Complete Set (2/2)

- Q: {NAND}?



- Q: {OR, NOT}?
- Q: {NOR}?
- Q: {AND, OR}?

- NOT is MUST!**

18

Two-Level NAND-/NOR-Gate Circuits

DeMorgan's Law Is the Key of Conversion

□ General:

$$\square (X_1 + X_2 + \dots + X_n)' = X_1'X_2'\dots X_n'$$

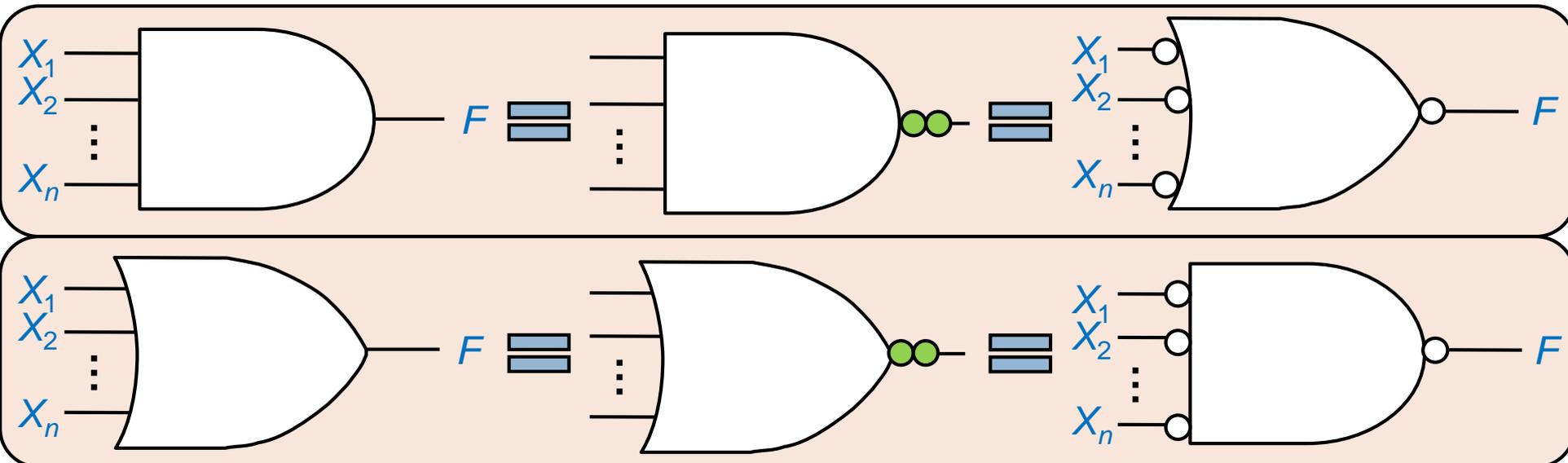
$$\square (X_1X_2\dots X_n)' = X_1' + X_2' + \dots + X_n'$$

□ One-step rule:

$$\square [f(x_1, x_2, \dots, x_n, 0, 1, +, \bullet)]' = f(x_1', x_2', \dots, x_n', 1, 0, \bullet, +)$$

$$\blacksquare x \leftrightarrow x'; + \leftrightarrow \bullet; 0 \leftrightarrow 1$$

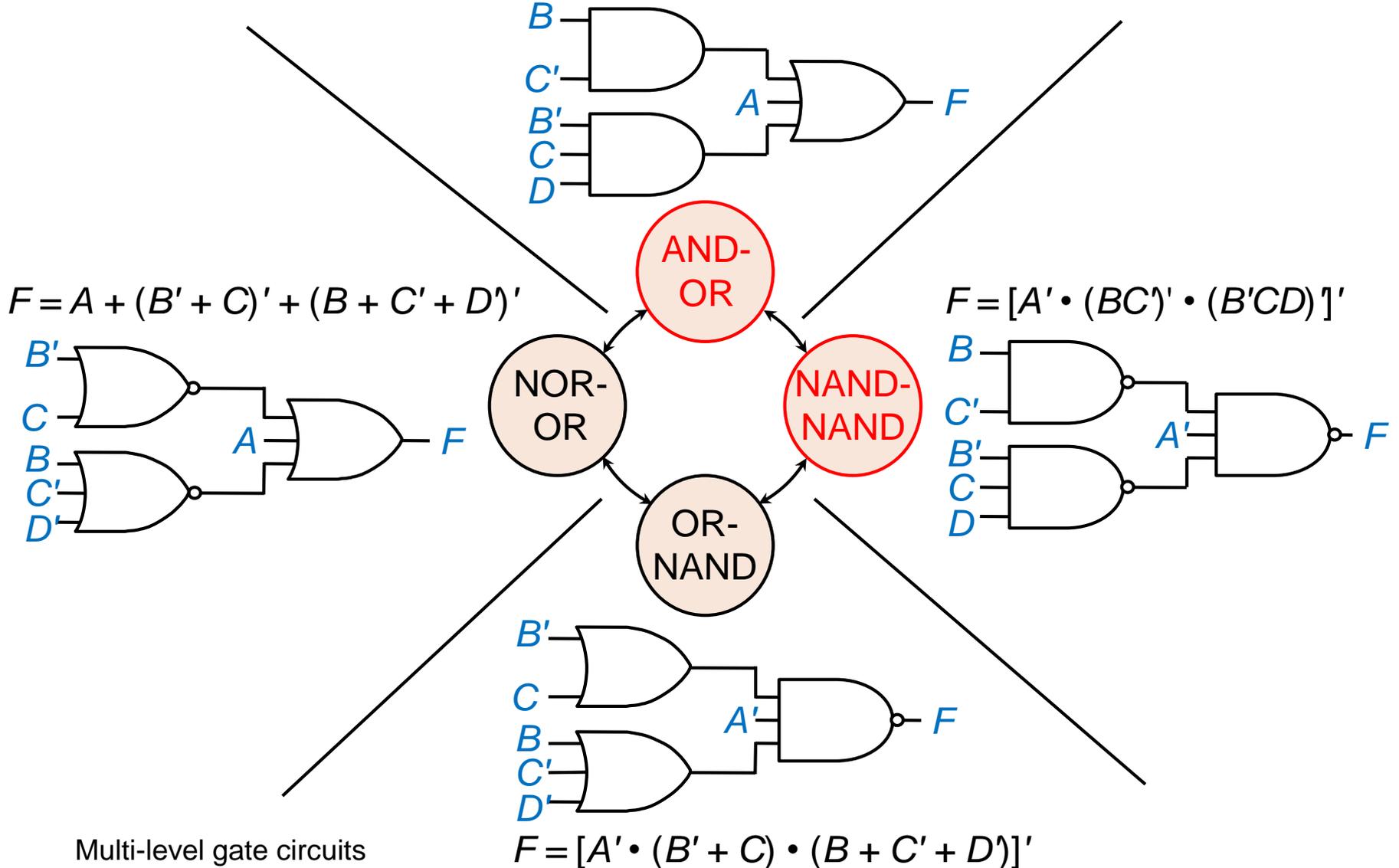
□ Idea: introduce 2 cascaded bubbles into an intermediate line



Multi-level gate circuits

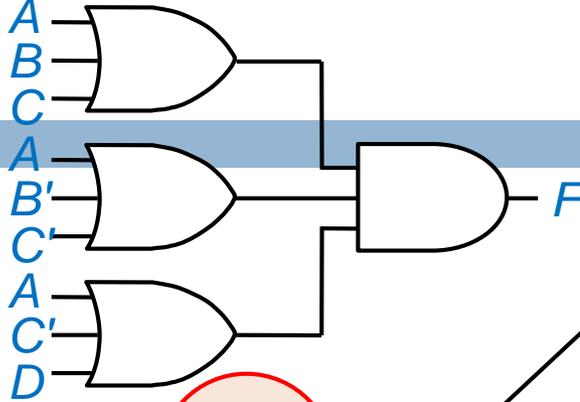
Eight Basic Forms for Two-Level CKTs (1/2)

$$F = A + BC' + B'CD$$

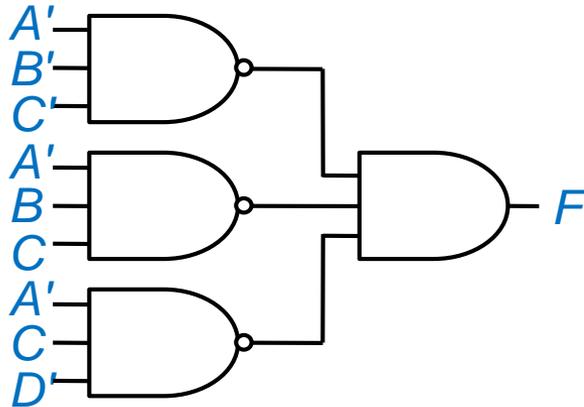


8 Forms (2/2)

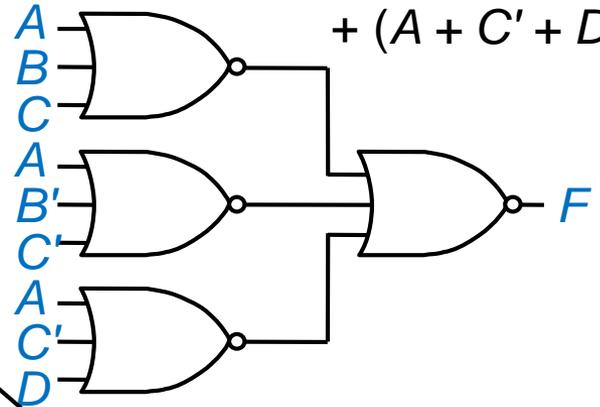
$$F = (A + B + C) (A + B' + C') (A + C' + D)$$



$$F = (A'B'C')' \cdot (A'BC)' \cdot (A'CD)'$$



$$F = [(A + B + C)' + (A + B' + C')' + (A + C' + D)]'$$

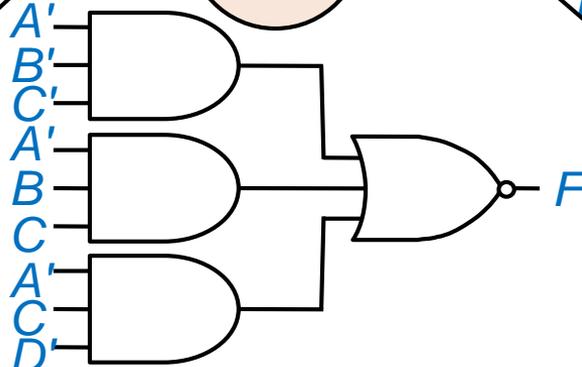


OR-AND

NAND-AND

NOR-NOR

AND-NOR

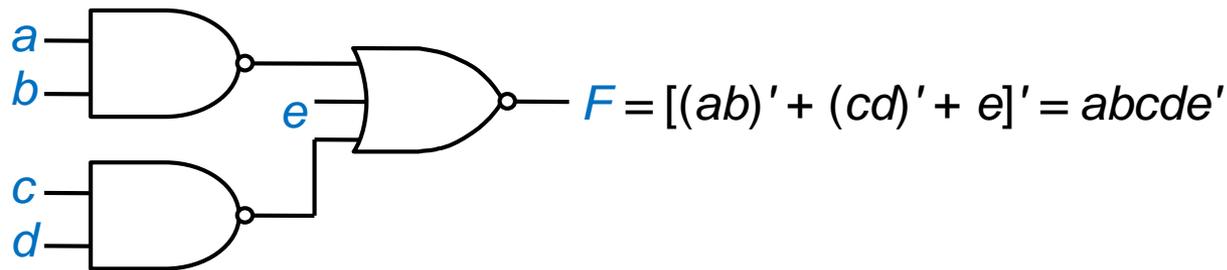


$$F = (A'B'C' + A'BC + A'CD)'$$

Multi-level gate circuits

Other Forms

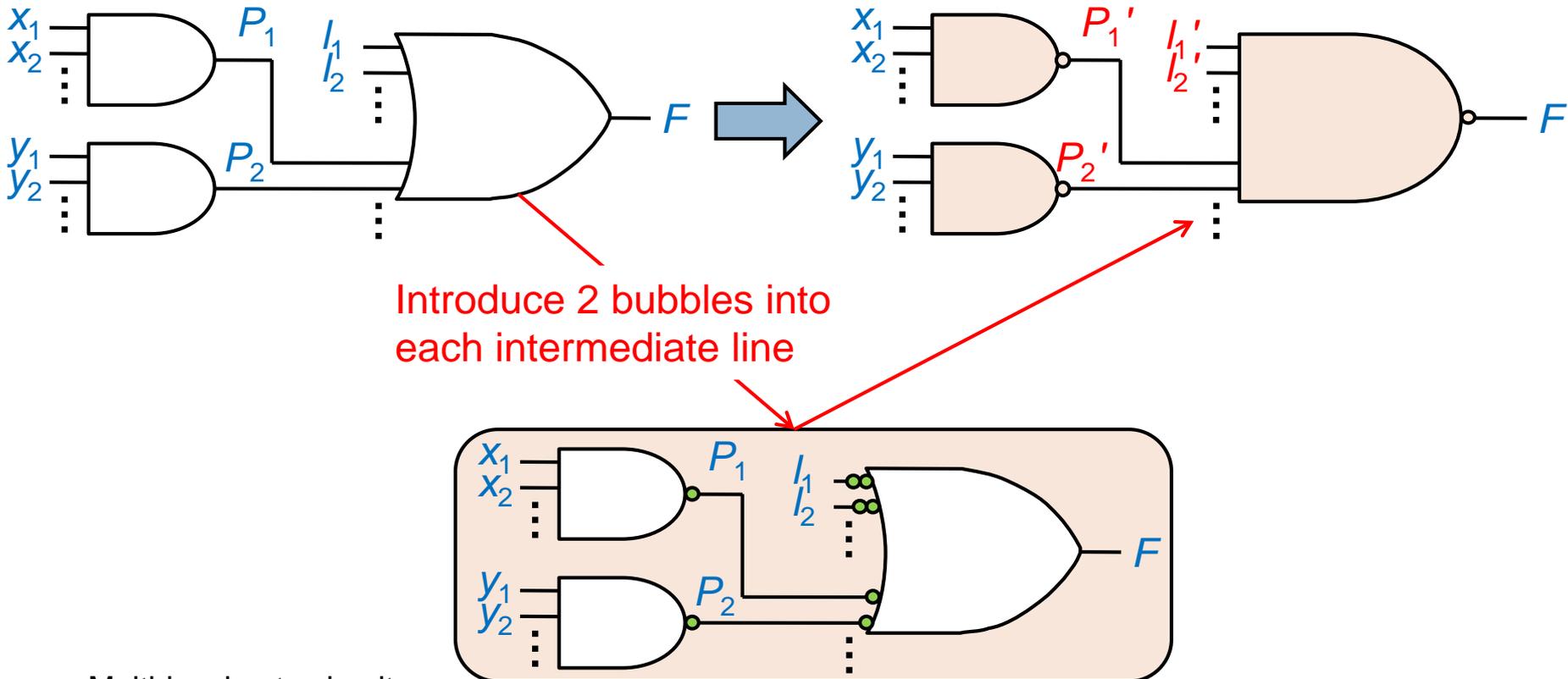
- **AND-AND, OR-OR, OR-NOR, AND-NAND, NAND-NOR, NOR-NAND, NAND-OR, NOR-AND are degenerated**
 - ▣ Cannot realize all switching functions
 - ▣ e.g., NAND-NOR can realize only a product of literals and not a SOP



Minimum 2-Level NAND-NAND Circuits

□ Procedure for designing a min 2-level NAND-NAND circuit:

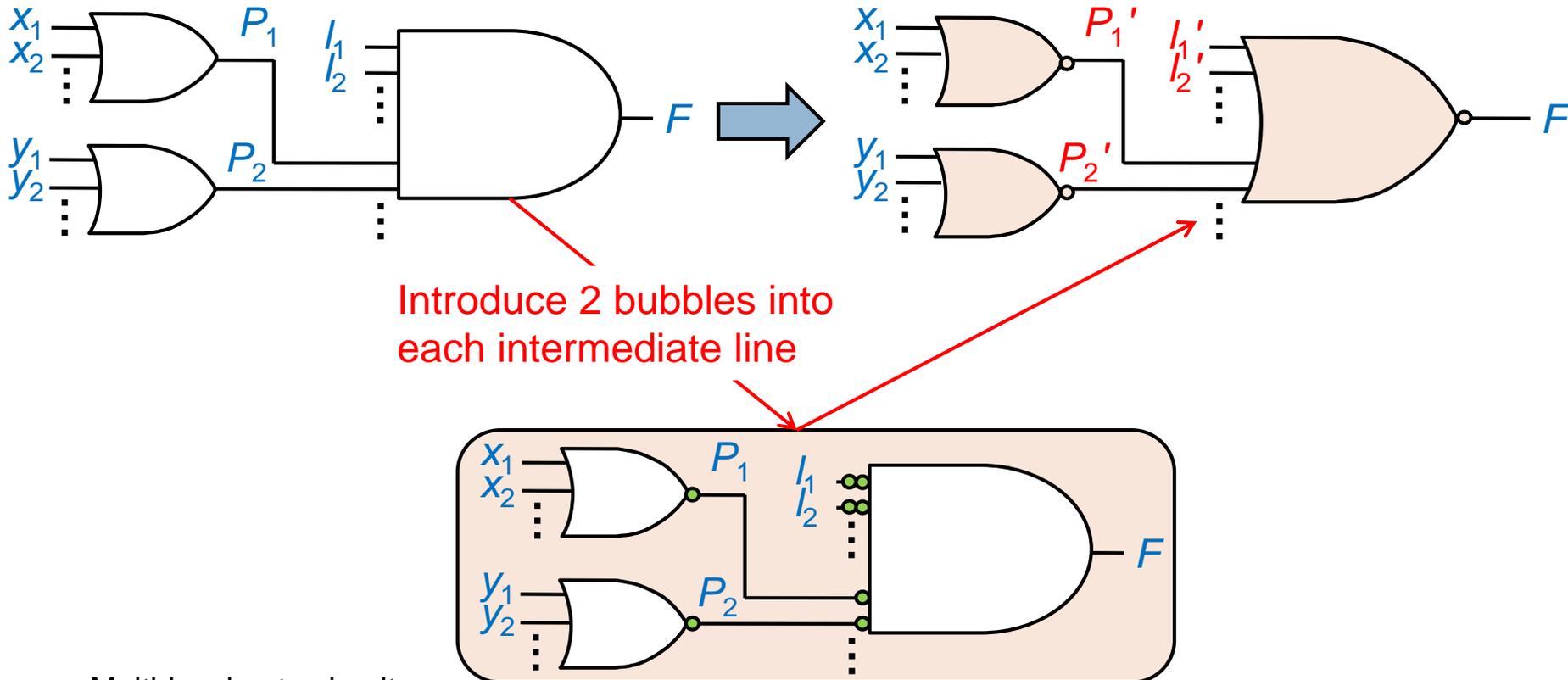
1. Find a minimum SOP
2. Draw AND-OR (2-level) circuit
3. Convert into NAND-NAND circuit



Minimum 2-Level NOR-NOR Circuits

□ Procedure for designing a min 2-level NOR-NOR circuit:

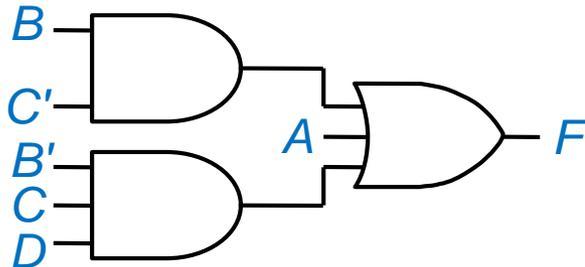
1. Find a minimum POS
2. Draw OR-AND (2-level) circuit
3. Convert into NOR-NOR circuit



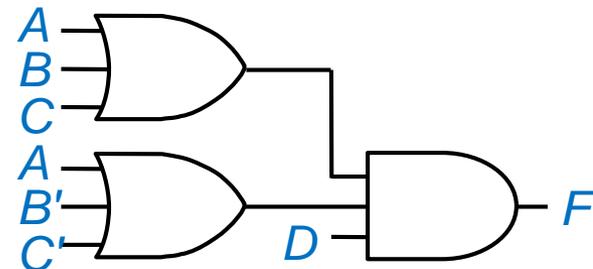
Example: NAND-NAND & NOR-NOR

□ DIY!

□ $F = A + BC' + B'CD$



□ $F = (A + B + C)(A + B' + C')D$



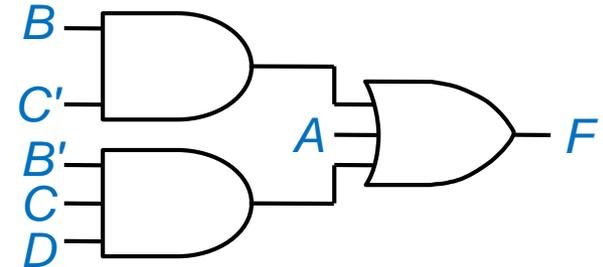
Summary: Two-Level Logic Minimization

26

© Iris H.-R. Jiang

- **Start with minimum SOP (AND-OR) or minimum POS (OR-AND)**
 - Simplified: **cost = (# of terms, # of literals)** (Units 2—5)
 - Real: **cost = (# of gates, # of gate inputs)**
 - # of gates \leq # of terms + 1
 - # of gate inputs \leq # of literals + # of terms
 - “<” holds if a term is degenerated to a single variable
- **8 basic forms can convert each other**
 - DeMorgan’s law
 - Use bubbles
- **NAND/NOR gates: low cost & high speed**

$$F = A + BC' + B'CD$$



27

Multi-Level NAND-/NOR-Gate Circuits

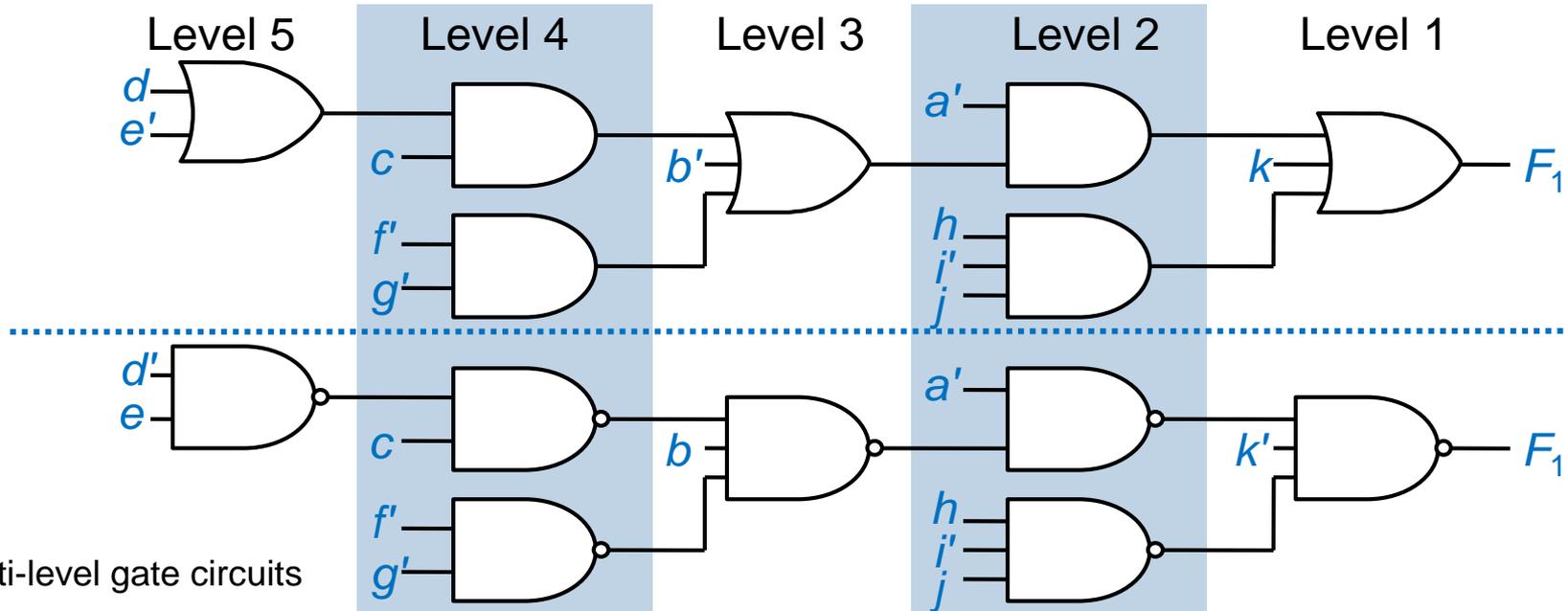
Multi-Level NAND-Gate Circuits

□ **Procedure for designing a multi-level NAND circuit:**

1. Simplify the switching function
2. Draw its multi-level AND-OR ckt
3. Replace all gates with NANDs and number all levels (starting with output gate as level 1)
4. **Invert inputs at levels 1, 3, 5, ...**

Add 2 bubbles into an intermediate line ...

□ e.g., $F = a[b' + c(d + e') + f'g] + hi'j + k$



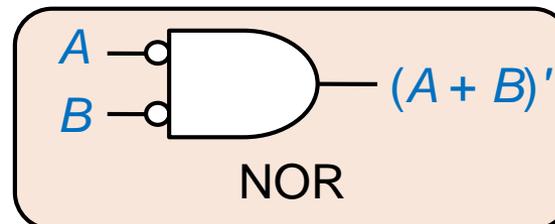
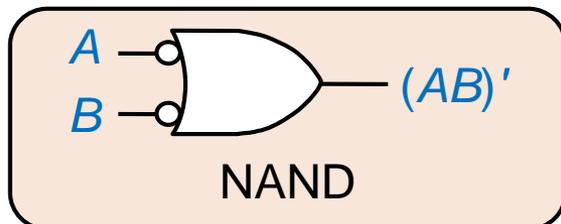
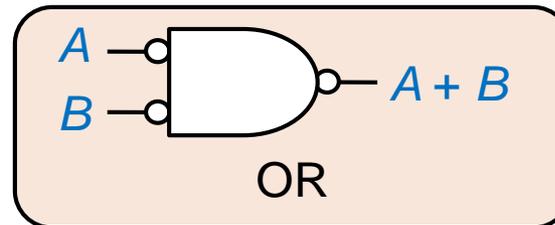
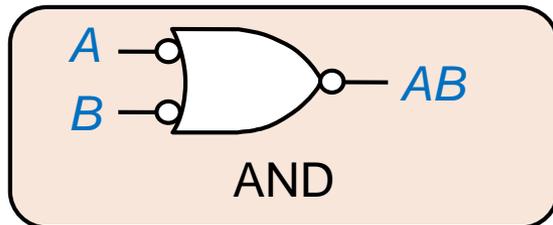
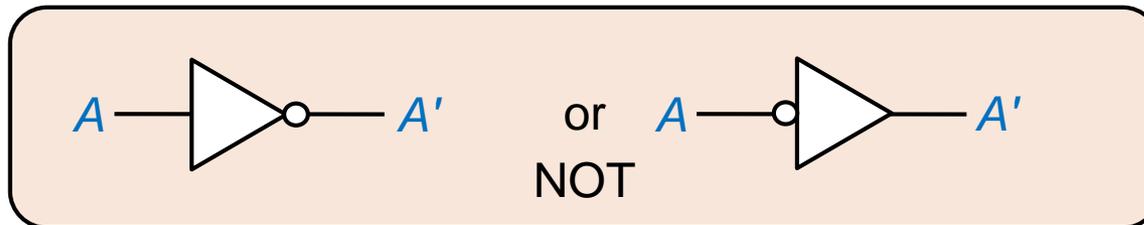
Multi-level gate circuits

29

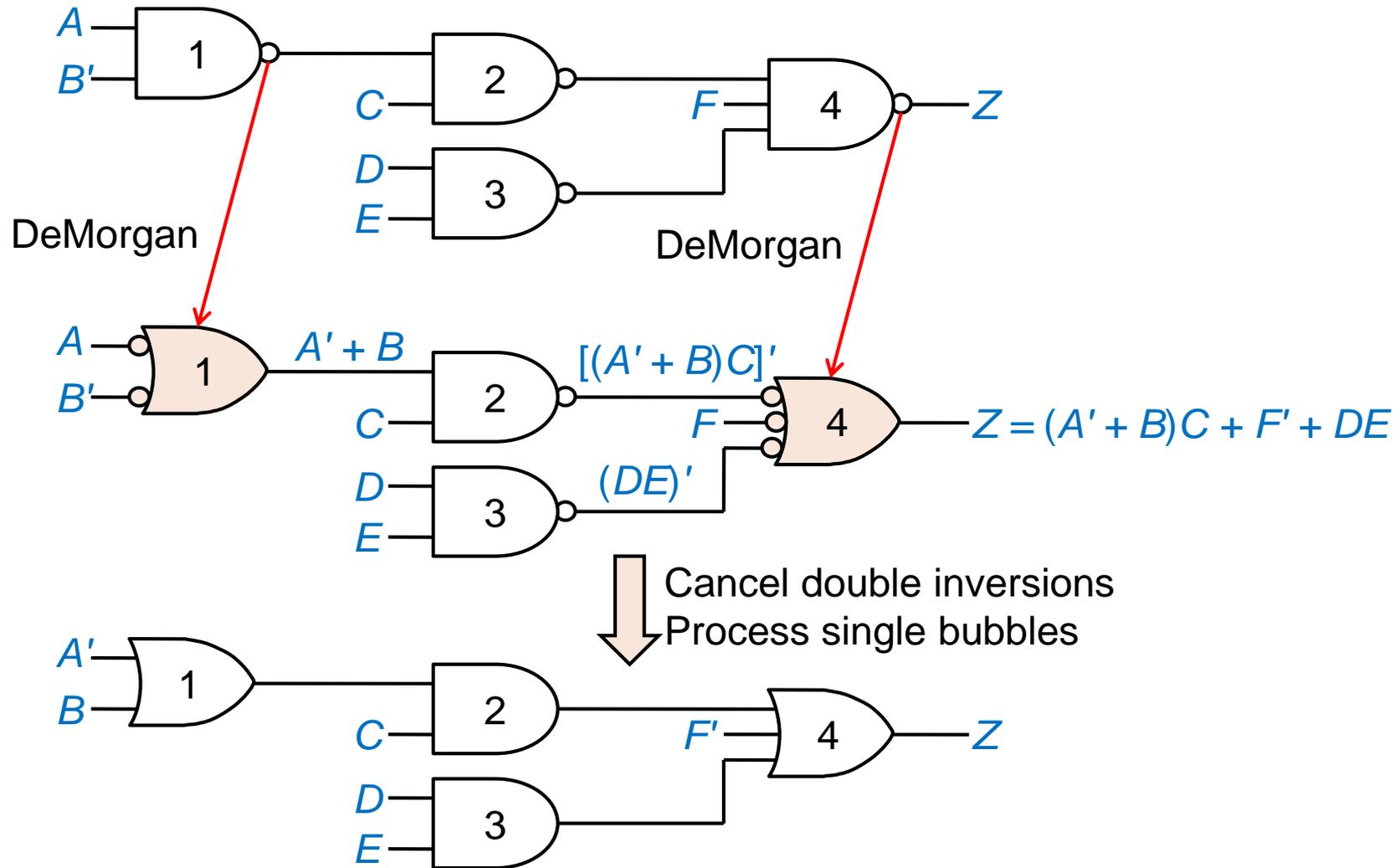
Circuit Conversion

Alternative Gate Symbols

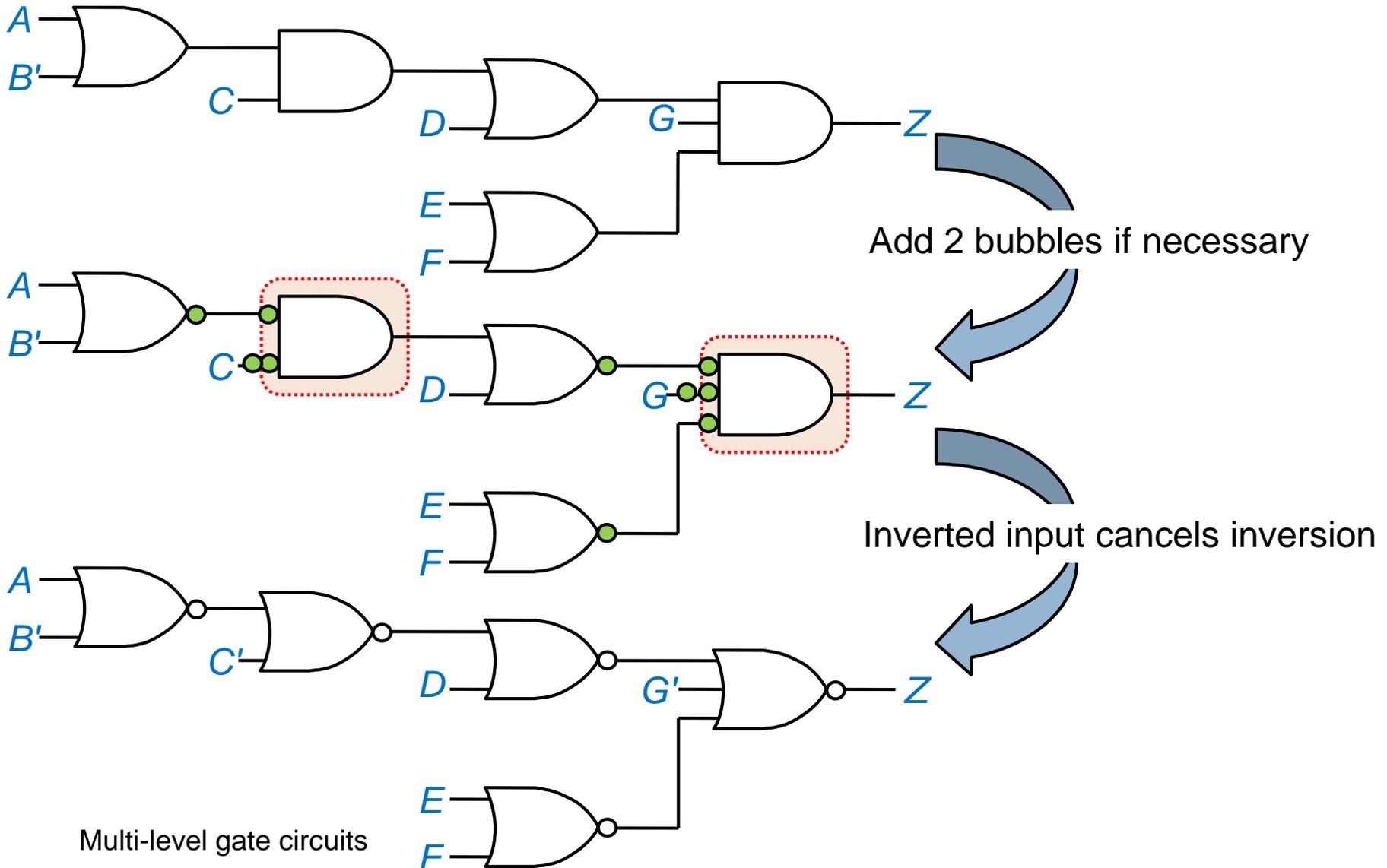
- Please do **NOT** memorize these gate symbols
- DeMorgan's law is the key of conversion
 - ▣ Use bubbles adequately



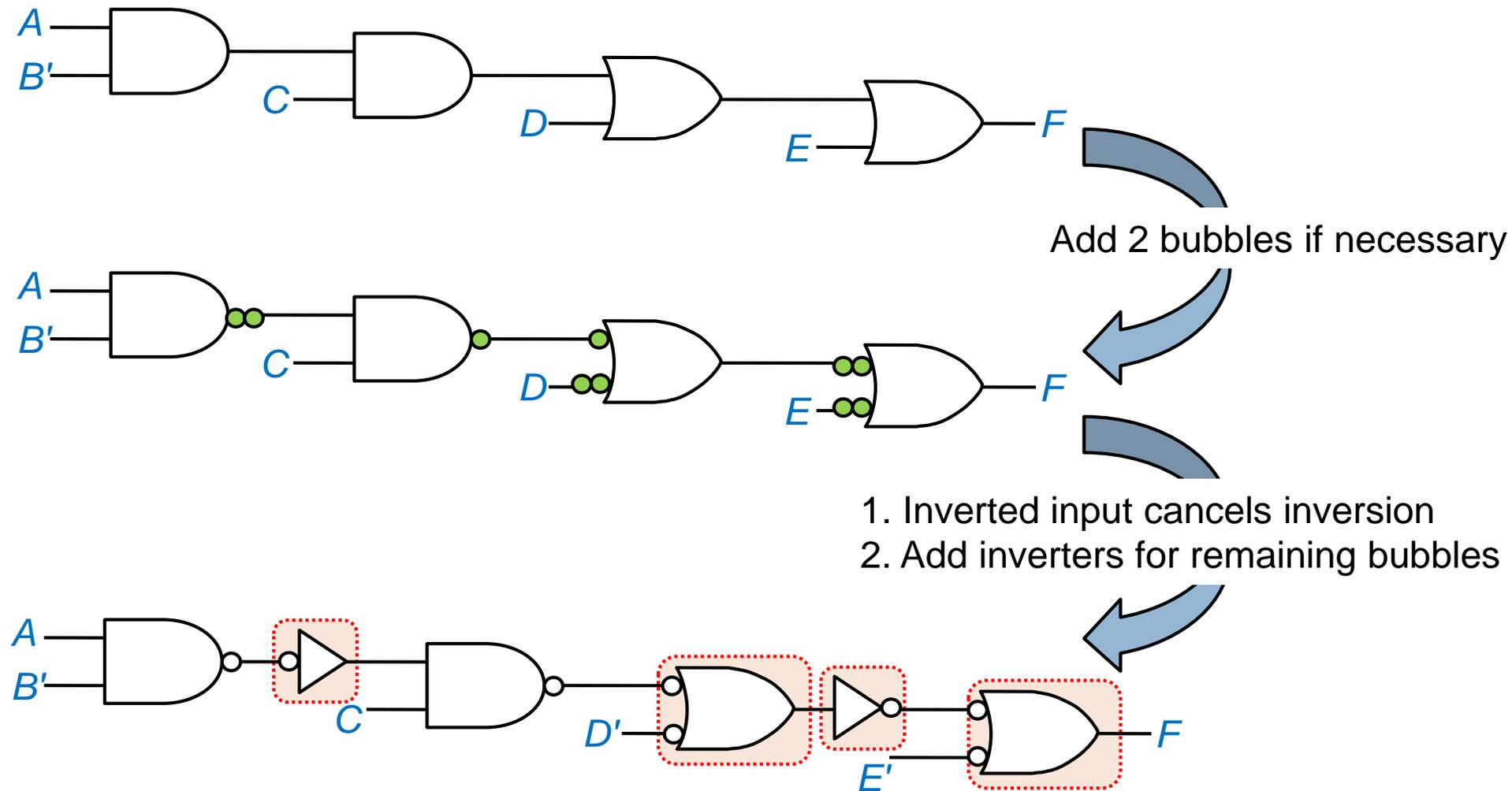
Conversion from NAND to AND-OR Gates



Conversion from OR-AND to NOR Gates



From a General AND-OR to NAND Network



Multi-level gate circuits

34

Two-Level Multiple-Output Circuits

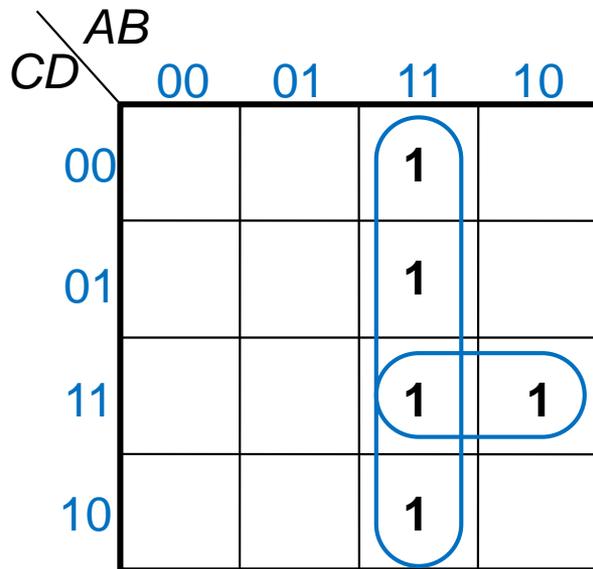
Case 1: 4 Inputs and 3 Outputs (1/2)

□ e.g.,

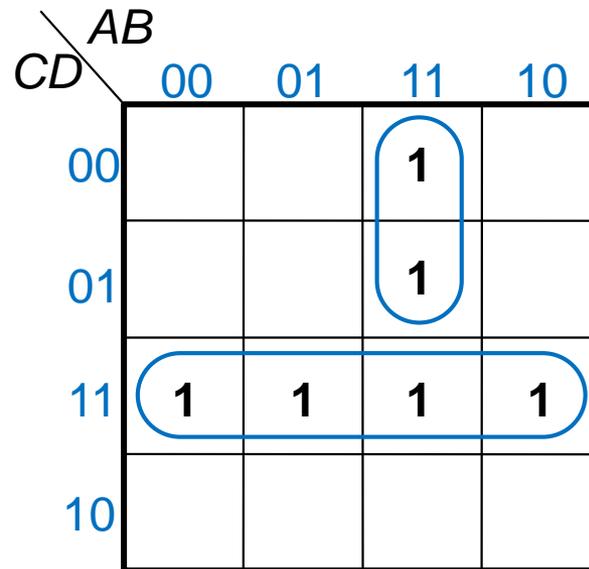
□ $F_1(A, B, C, D) = \Sigma m(11, 12, 13, 14, 15)$

□ $F_2(A, B, C, D) = \Sigma m(3, 7, 11, 12, 13, 15)$

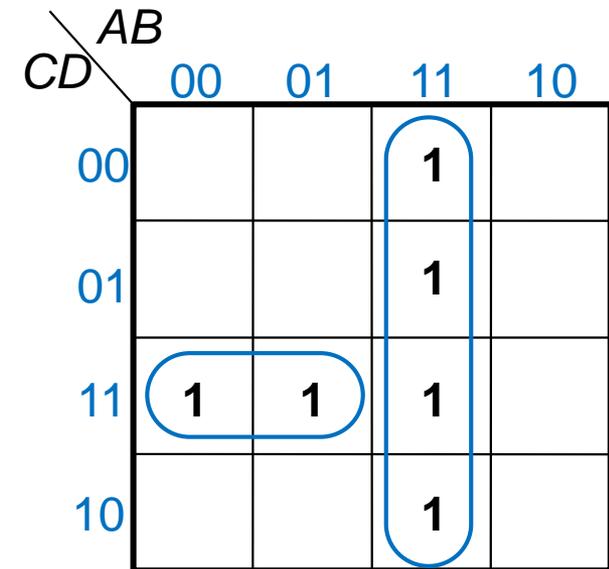
□ $F_3(A, B, C, D) = \Sigma m(3, 7, 12, 13, 14, 15)$



F_1
 $F_1 = AB + ACD$



F_2
 $F_2 = ABC' + CD$

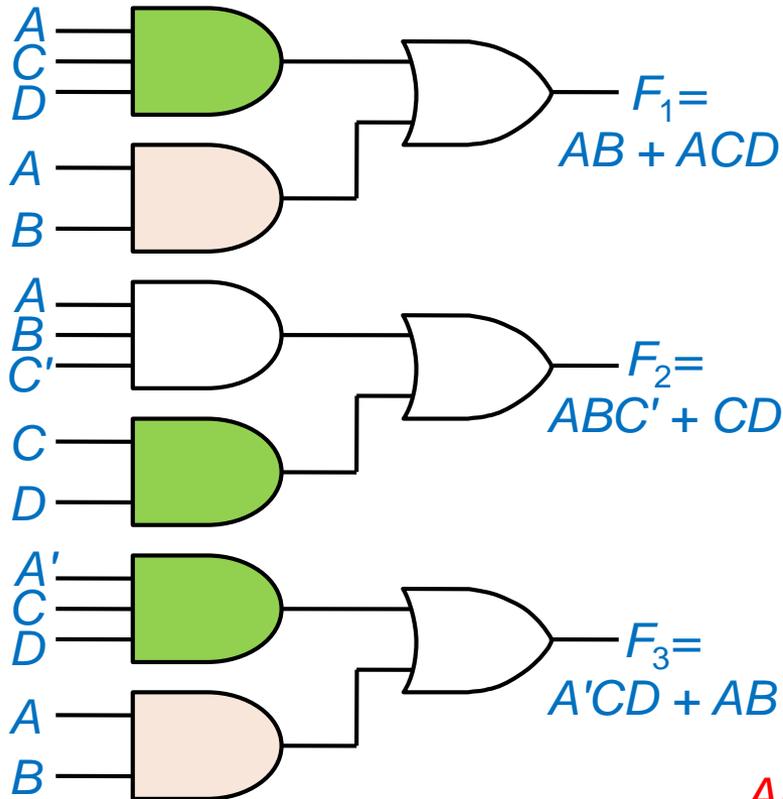


F_3
 $F_3 = A'CD + AB$

Case 1: 4 Inputs and 3 Outputs (2/2)

Direct implementation

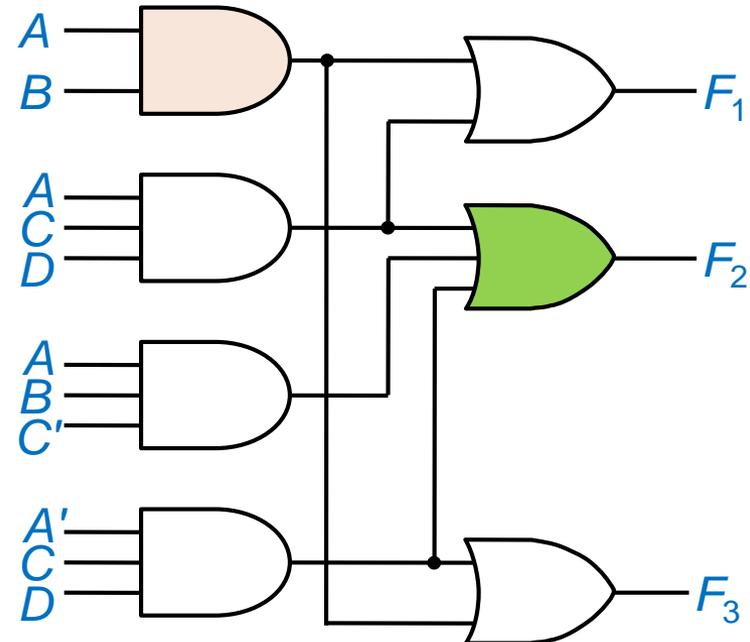
- 9 gates; 21 gate inputs



Multi-level gate circuits

After sharing

- 7 gates; 18 gate inputs

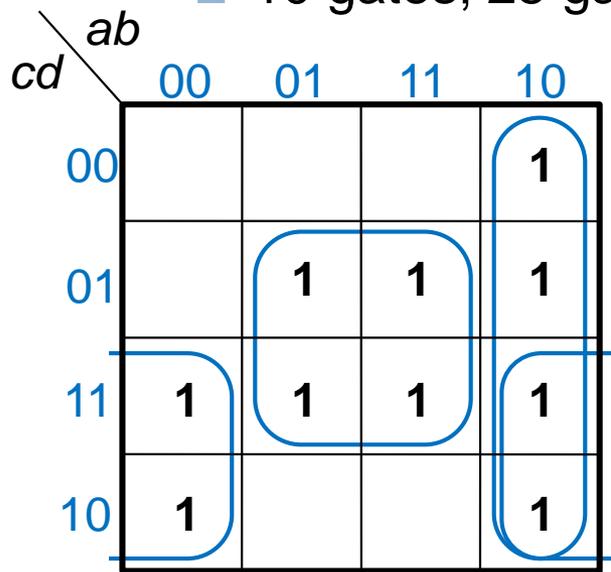


AB is shared (F_1 & F_3)

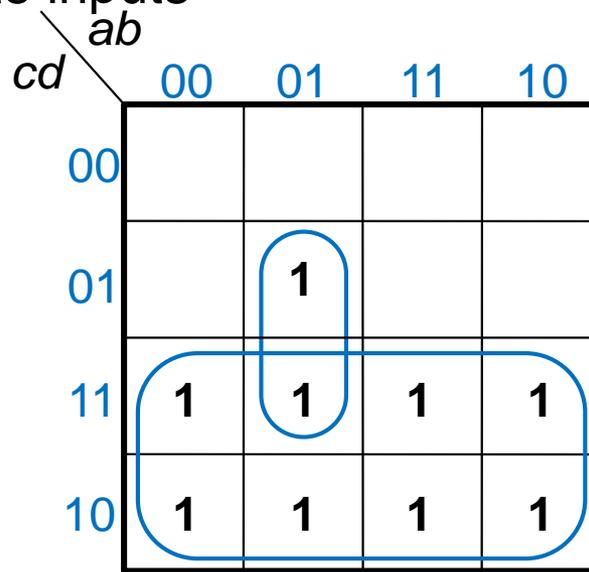
$CD = A'CD + ACD$ (F_1 & $F_3 \rightarrow F_2$)

Case 2: 4 Inputs and 3 Outputs (1/2)

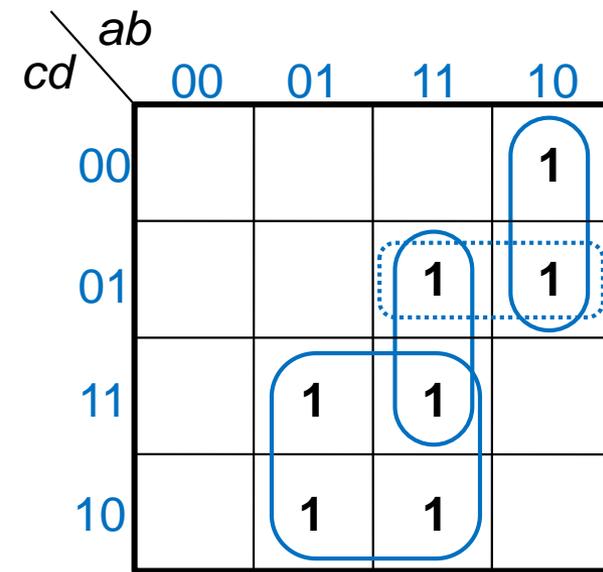
- e.g.,
 - $f_1(a, b, c, d) = \Sigma m(2, 3, 5, 7, 8, 9, 10, 11, 13, 15)$
 - $f_2(a, b, c, d) = \Sigma m(2, 3, 5, 6, 7, 10, 11, 14, 15)$
 - $f_3(a, b, c, d) = \Sigma m(6, 7, 8, 9, 13, 14, 15)$
- **If each function is minimized separately**
 - 10 gates; 25 gate inputs



$$f_1 = bd + b'c + ab'$$



$$f_2 = c + a'bd$$



$$f_3 = bc + abd + ab'c' \quad (ac'd)$$

Multi-level gate circuits

Case 2: 4 Inputs and 3 Outputs (2/2)

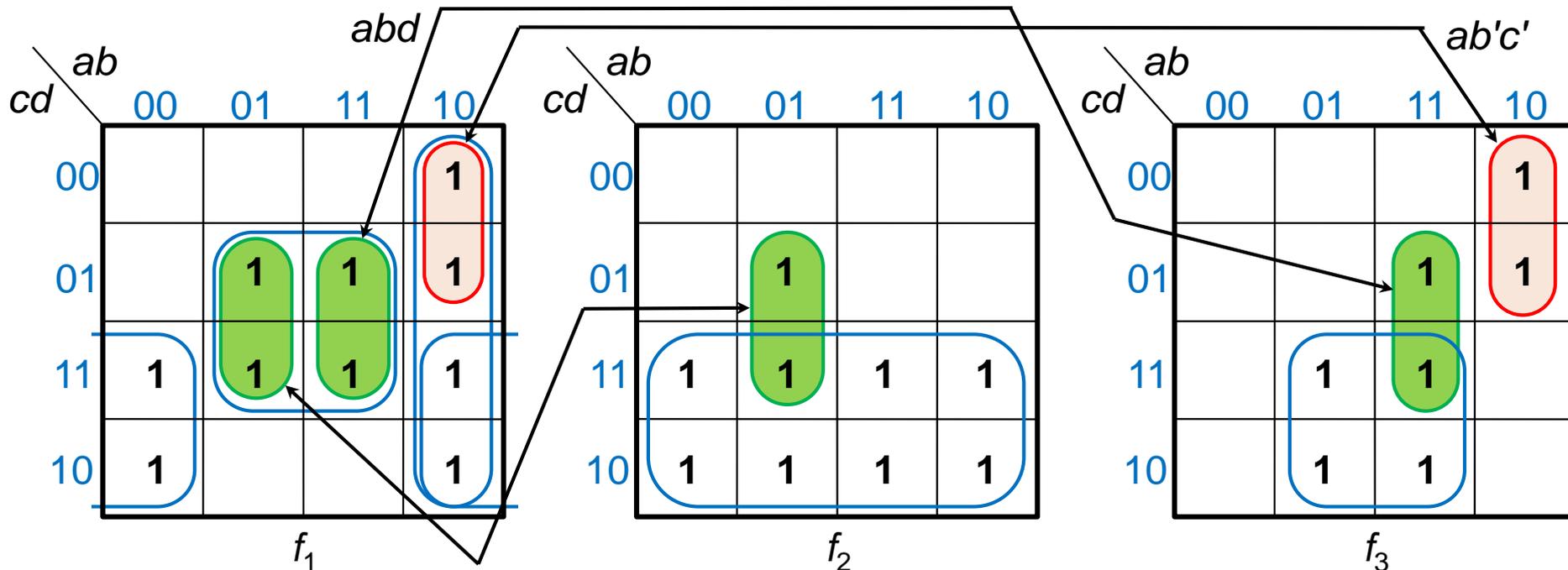
Find common minterms from K-map

$f_1 = abd + a'bd + b'c + ab'c'$

$f_2 = c + a'bd$

⇒ 8 gates; 22 gate inputs

$f_3 = bc + ab'c' + abd$



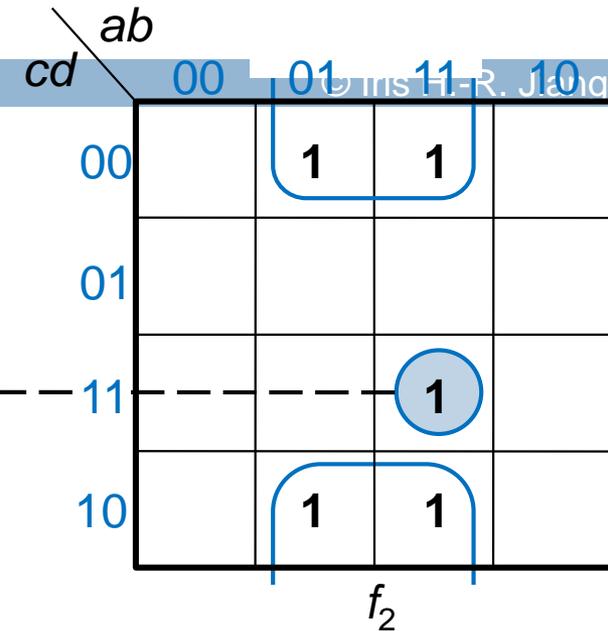
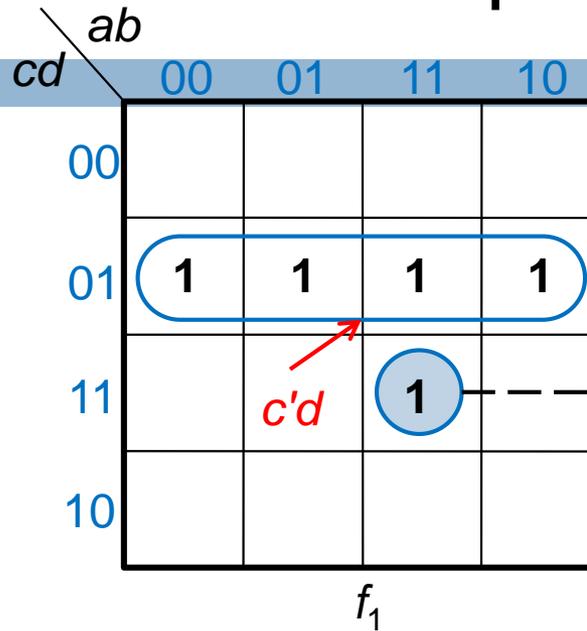
$bd (f_1) \rightarrow a'bd (f_2) + abd (f_3)$ (combine)

$ab'c' (f_1) \rightarrow$ covered by $ab'c' (f_3)$ (share)

Case 3: Essential Prime Implicants

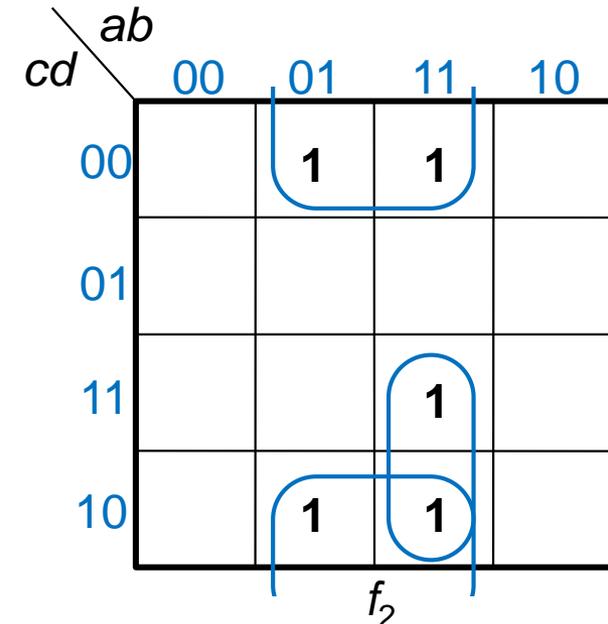
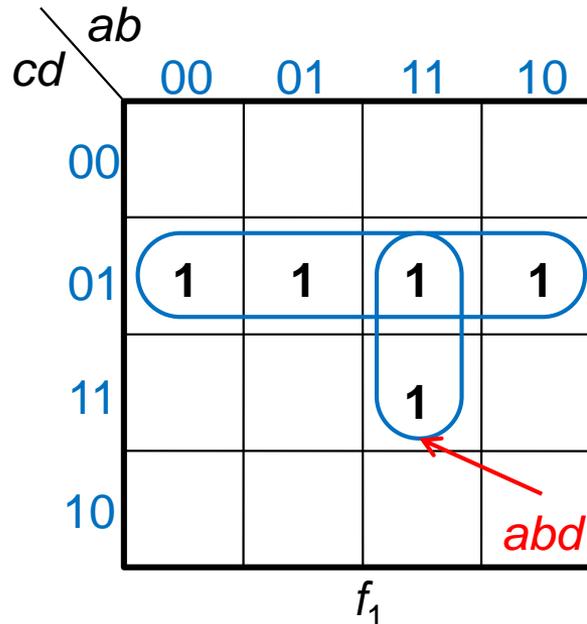
39

□ **Best**



□ **One more gate**

Essential for multi-output:
 Only check 1's which do not appear in other maps
 e.g,
 $c'd$ is essential
 abd is not



Multi-level gate circuits

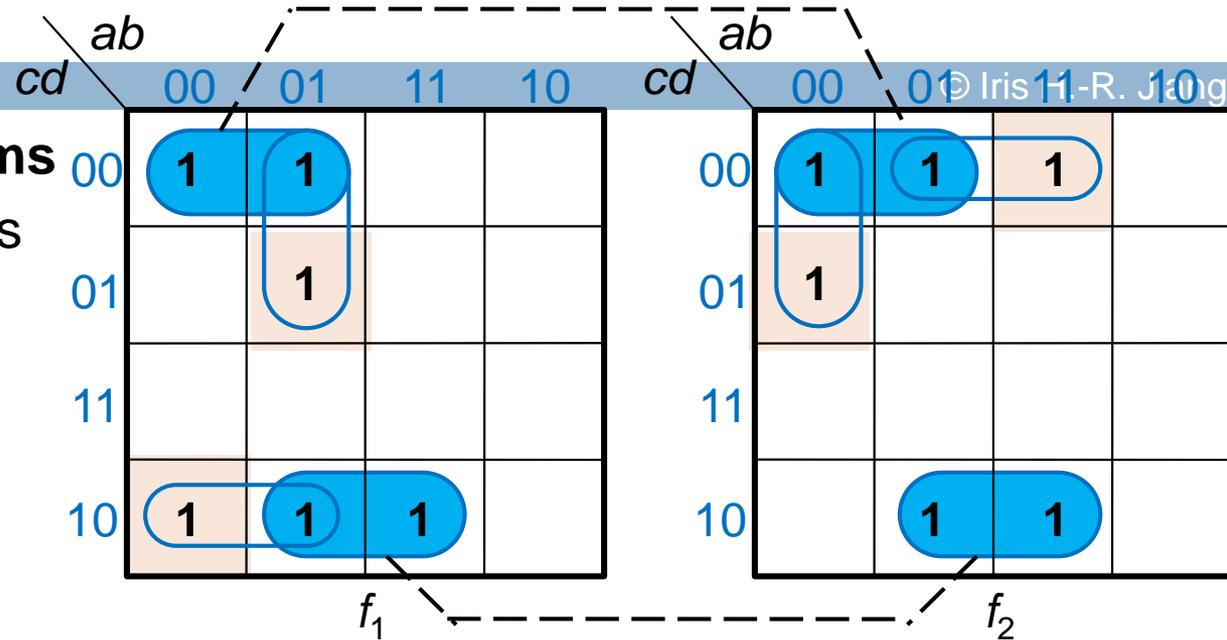
Case 4

40

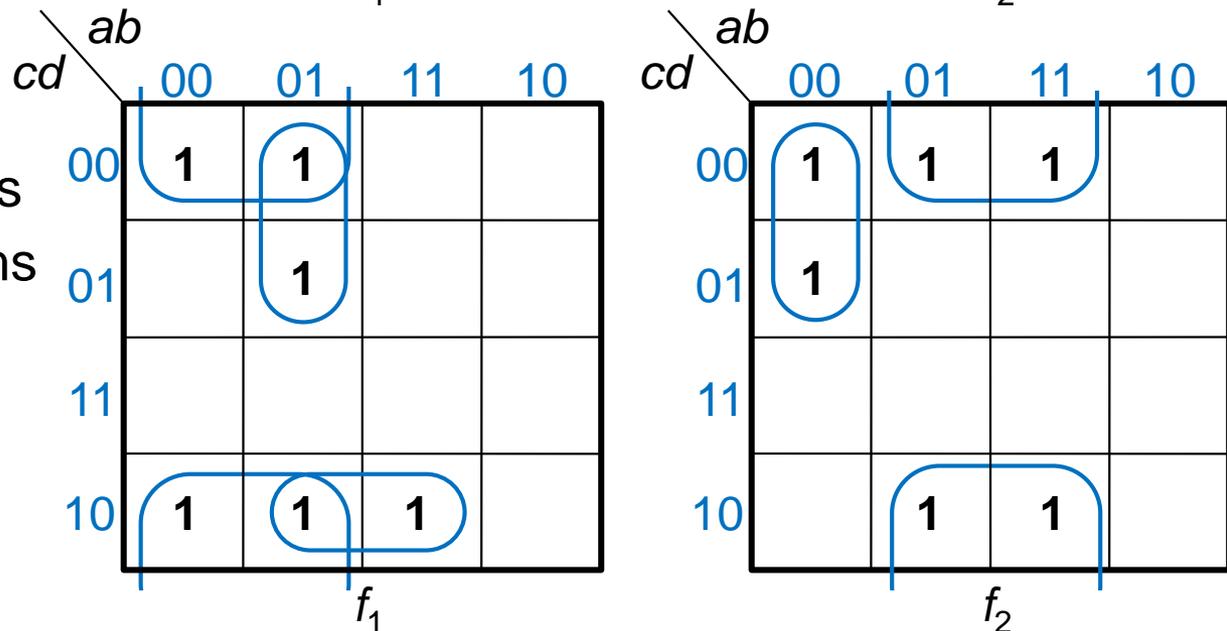
© Iris 11-R. J. J. g

- **Most common terms**
 - ▣ 8 gates; 26 inputs

Who is essential?



- **Best**
 - ▣ 7 gates; 18 inputs
 - ▣ No common terms



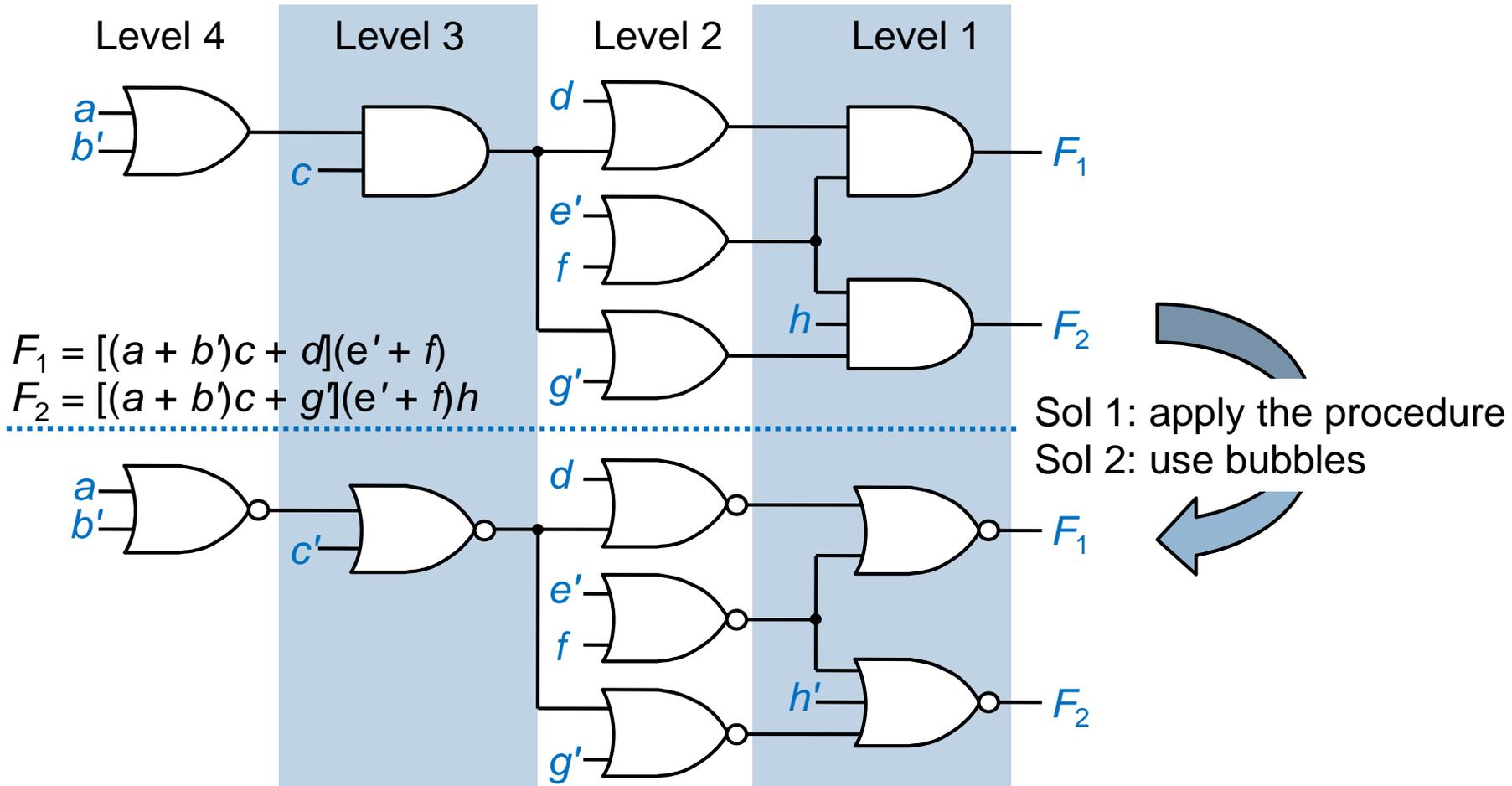
Summary: 2-Level **Multiple-Output** Circuits

- **Cost = (total # of gates, total # of gate inputs)**
- **Think in a global way**
- **A min SOP for each cannot guarantee min cost for all**
- **Common and essential** terms may help
 - ▣ Sometimes, we do not need to make circles that large
 - ▣ 'Essential' is defined in a different way
- **Very difficult to find global optimal solution**
 - ▣ **Boolean algebra is very useful in more sophisticated techniques**
(Units 2&3) (later courses)

42 Multiple-Output NAND & NOR Circuits

Multiple-Output NAND & NOR Circuits

- If all of the output gates are **OR/AND** gates, direct conversion to a **NAND-/NOR-gate** circuit is possible



Multi-level gate circuits