

Digital Systems Design and Laboratory

[7. Multi-Level Gate Circuits]

Chung-Wei Lin

cwlin@csie.ntu.edu.tw

CSIE Department

National Taiwan University

We would like to thank Prof. Hui-Ru Jiang (NTU) for contributing the lecture material

Recap: Logic Design

❑ Steps

- Translate the word description into a switching function (Unit 4)
 - Truth table
 - Boolean expression
 - SOP/POS derived from minterm or maxterm expansion (Unit 4)
- Simplify the function
 - Boolean algebra (Units 2 and 3)
 - Karnaugh map (Unit 5)
 - Quine-McCluskey (Unit 6)
 - Other methods
- Realize it using available logic gates
 - AND-OR, OR-AND (Unit 2)
 - NAND-NAND, NOR-NOR (Unit 7)
 - Others

Outline

❑ **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

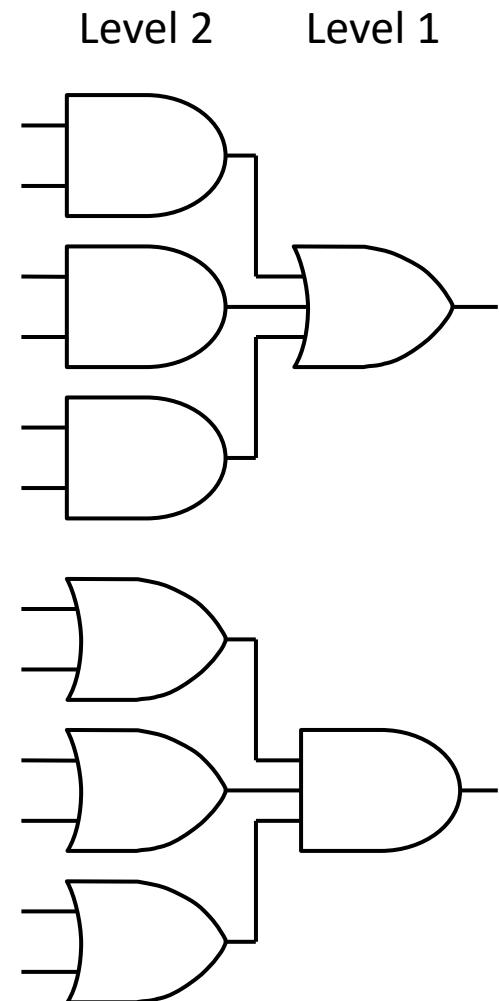
AND-OR & OR-AND Circuits

- The # of levels in a circuit =
the maximum # of gates cascaded
in series between an input & the output

- Count from the output
- Do NOT count inverters at inputs

- Two-level circuits learned so far

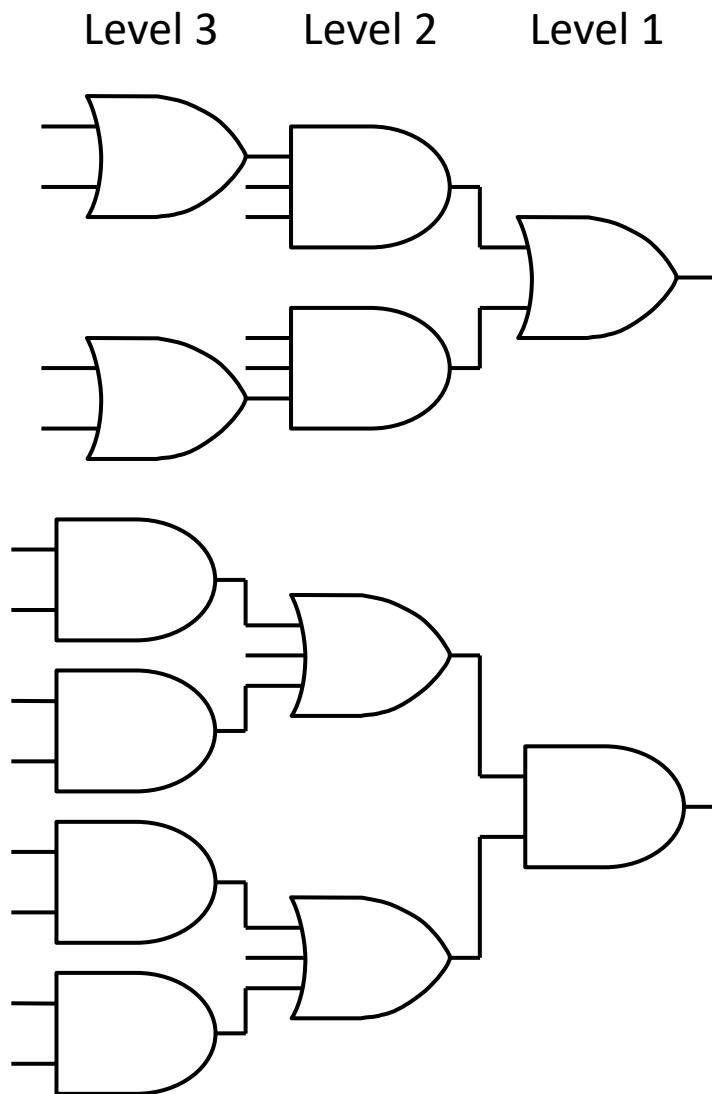
- AND-OR circuit = SOP form
 - One level of AND gates + one OR gate
- OR-AND circuit = POS form
 - One level of OR gates + one AND gate



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

❑ Three-level circuits

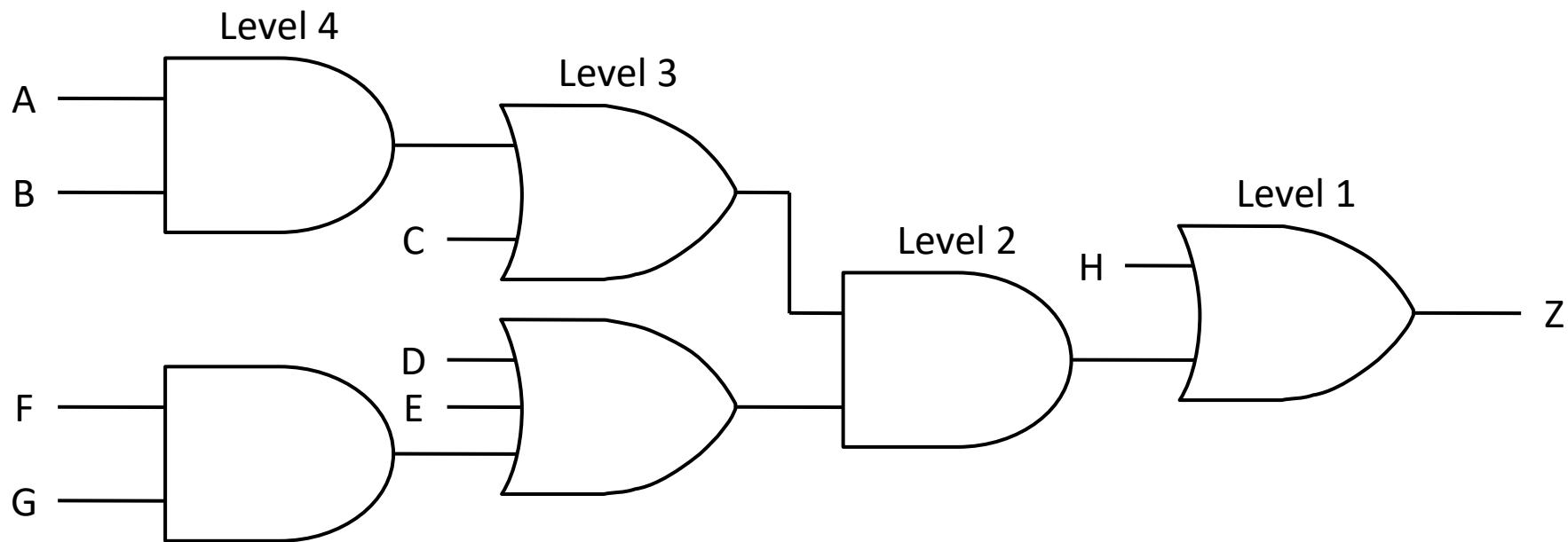
- OR-AND-OR circuit
- AND-OR-AND circuit



One More Example (1/2)

❑ How to count the # of levels?

- By inspection
- Example: $Z = (AB + C)(D + E + FG) + H$
 - 4 levels
 - 6 gates
 - 13 gate inputs

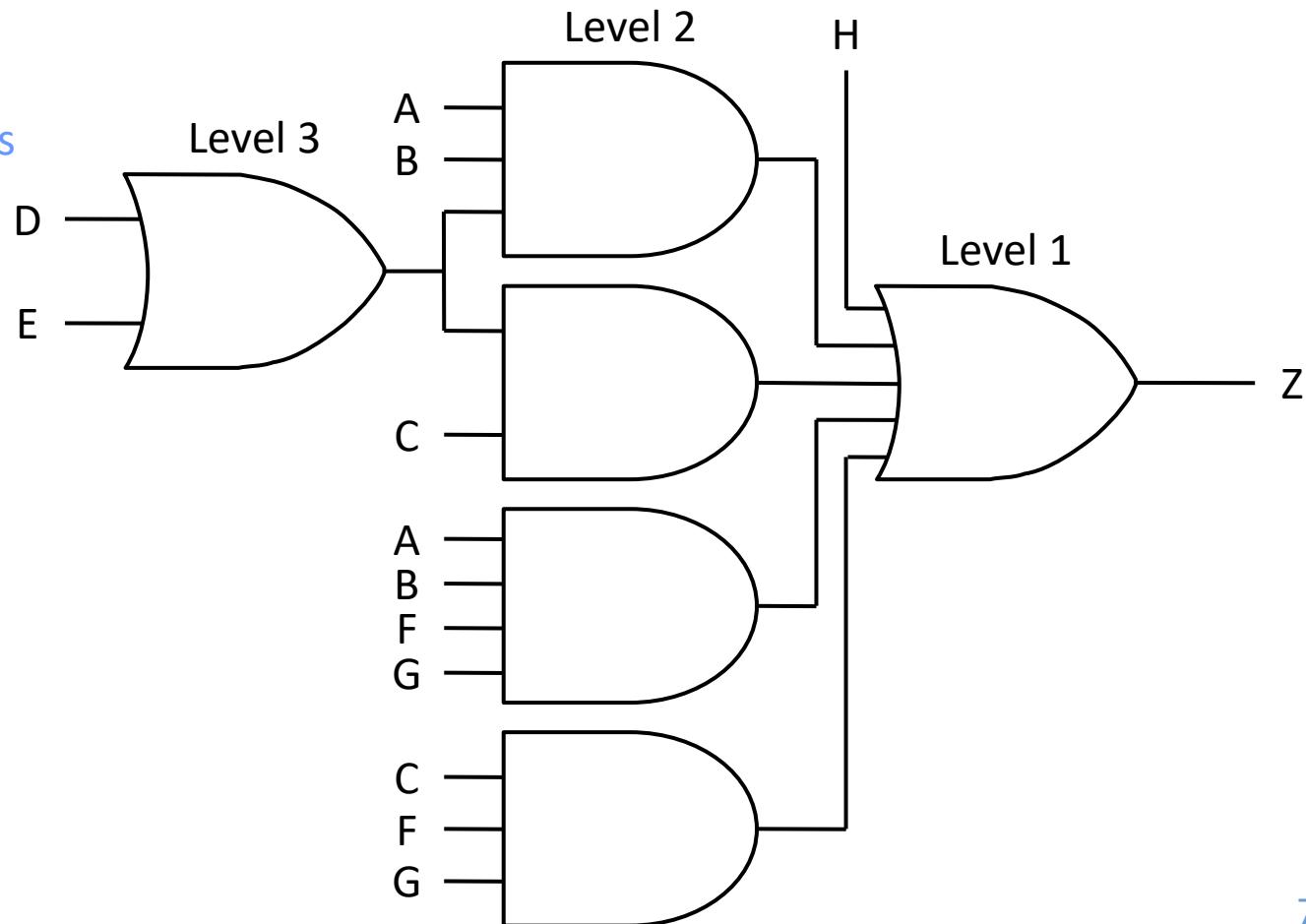


One More Example (2/2)

□ Same function in another realization

➤ $Z = AB(D + E) + C(D + E) + ABFG + CFG + H$

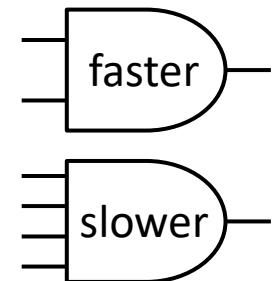
- 3 levels
- 6 gates
- 19 gate inputs



Multi-Level Gate Circuits

- ❑ Q1: Why does the # of levels matter?
- ❑ A1: The # of levels affects cost and delay
 - Cost depends on # of gates & # of gate inputs
 - Delay depends on # of levels & # of inputs for a single gate
 - # of levels \uparrow : cost \downarrow (maybe) and circuit delay \uparrow (maybe)
- ❑ Q2: Any application for the level of each gate?
- ❑ A2: Evaluation according to the level in nonincreasing order

- ❑ Q3: How to change the # of levels?
- ❑ A3: Factoring or multiplying out



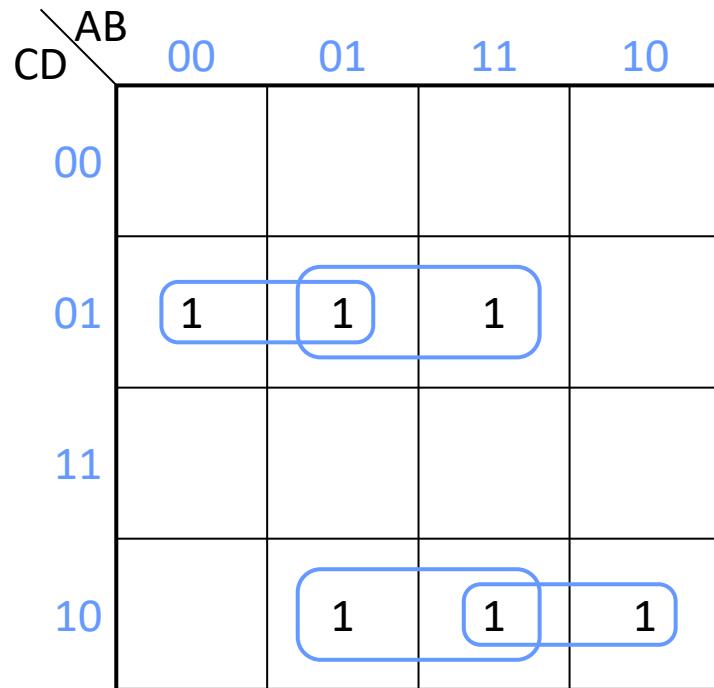
Multi-Level Design with AND & OR (1/4)

- Find a circuit of AND and OR gates to realize

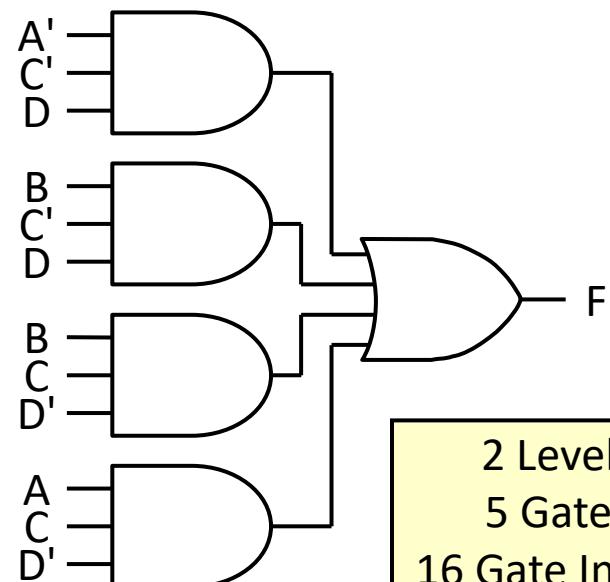
- $F(A,B,C,D) = \sum m(1, 5, 6, 10, 13, 14)$

- Solution 1: try two-level AND-OR circuit first

- Simplify F by K-map (circle 1's)
- Realize it



$$F = A'C'D + BC'D + BCD' + ACD'$$



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

❑ From Solution 1

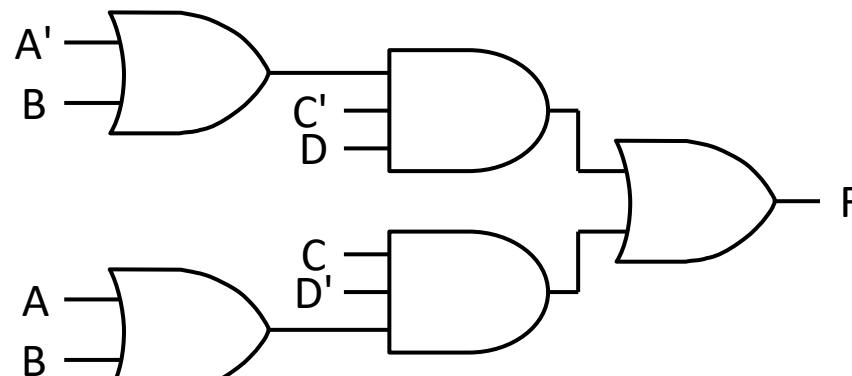
➤ $F = A'C'D + BC'D + BCD' + ACD'$

❑ Solution 2

➤ Factoring it yields $F = (A' + B)C'D + (B + A)CD'$

➤ Lead to a three-level OR-AND-OR circuit

- # of gate inputs \downarrow , cost \downarrow (maybe), delay \uparrow (maybe)



3 Levels
5 Gates
12 Gate Inputs

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

- Find a circuit of AND and OR gates to realize

- $F(A,B,C,D) = \sum m(1, 5, 6, 10, 13, 14)$

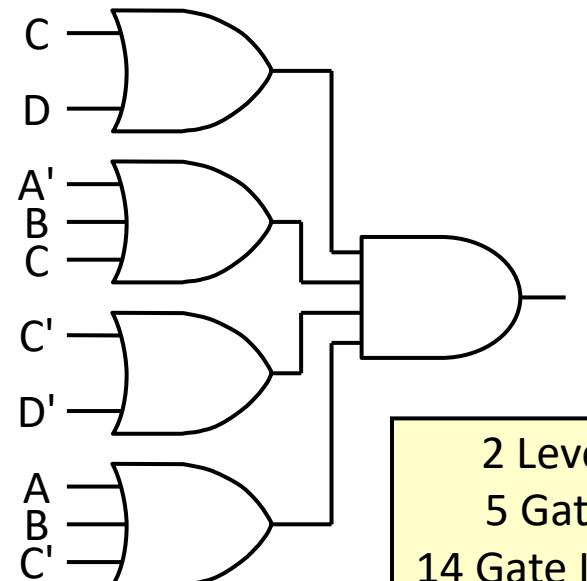
- Solution 3: try two-level OR-AND circuit first

- Simplify F by K-map (circle 0's)

- Realize it

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

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



2 Levels
5 Gates
14 Gate Inputs

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

❑ From Solution 3

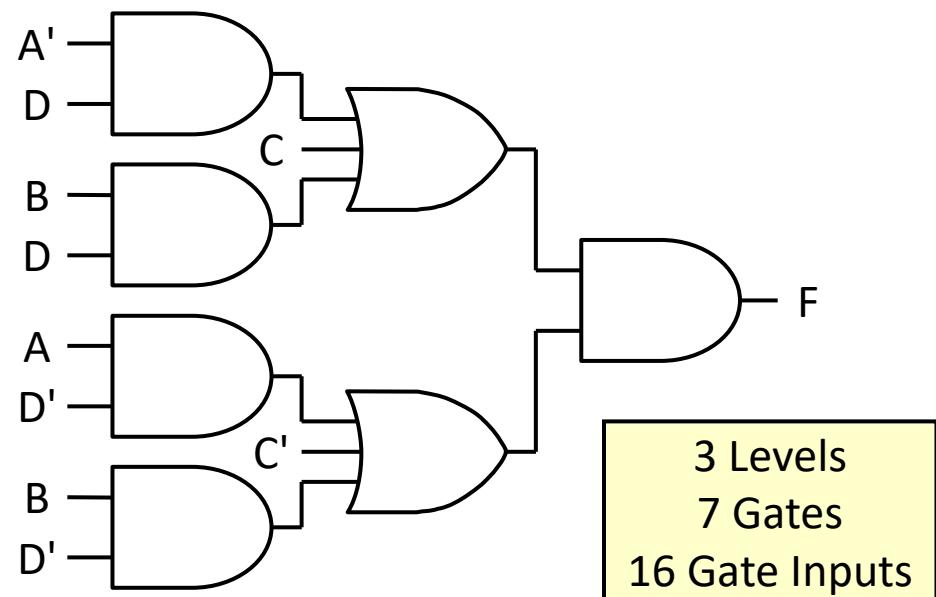
$$\rhd F = (C + D)(A' + B + C)(C' + D')(A + B + C')$$

❑ Solution 4

- Multiplying it out yields $F = [C + D(A' + B)][C' + D'(A + B)]$ (4-level)
- Partially multiplying out yields $F = (C + A'D + BD)(C' + AD' + BD')$
- Lead to a three-level AND-OR-AND circuit

❑ Best solutions for F

- Two-level: OR-AND
 - [\(Solution 3\)](#)
- Three-level: OR-AND-OR
 - [\(Solution 2\)](#)



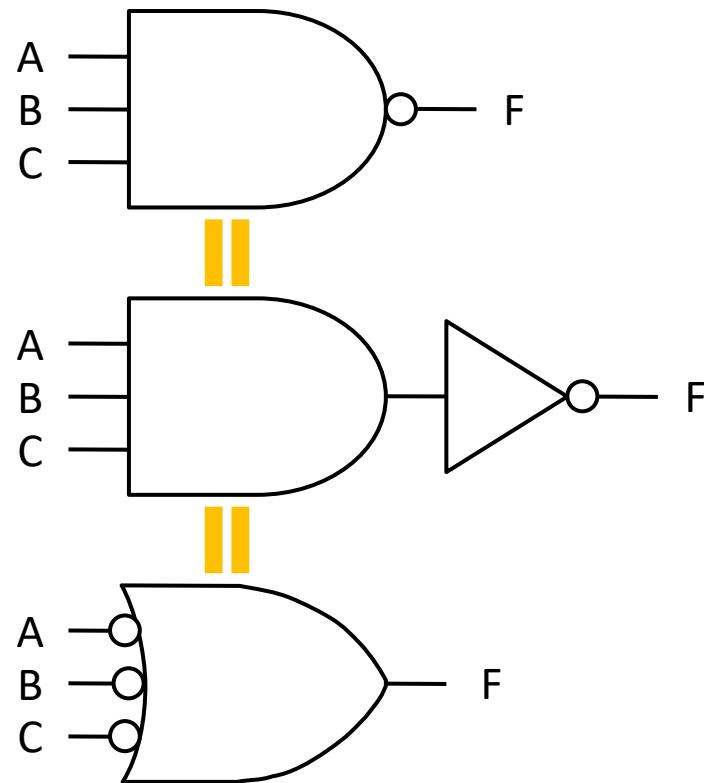
Outline

- ❑ 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

NAND Gate

□ NAND \equiv AND-NOT

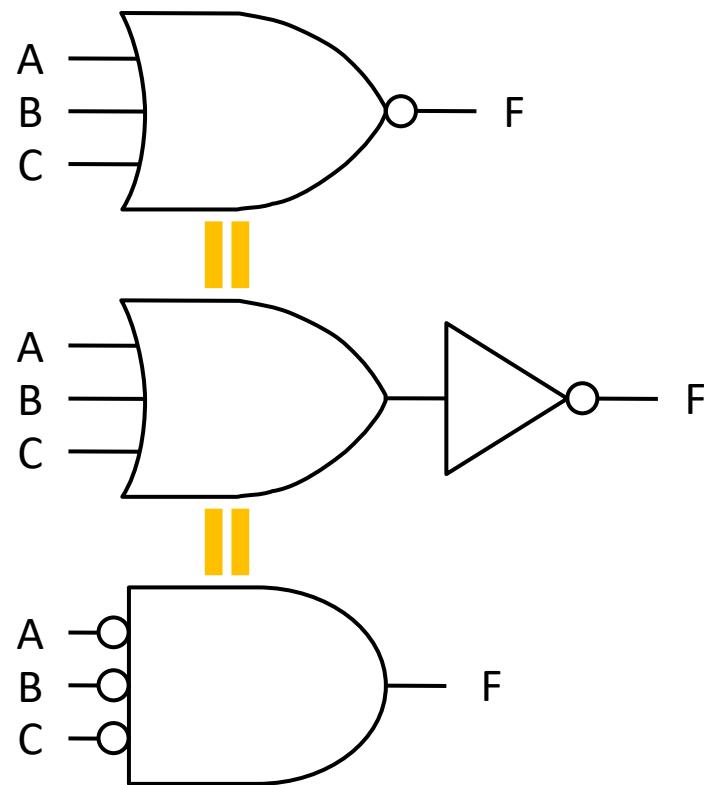
- DeMorgan's law: $F = (ABC)' = A' + B' + C'$
- General: $F = (X_1 X_2 \dots X_n)' = X_1' + X_2' + \dots + X_n'$
- Symbol: Bubble \equiv NOT



NOR Gate

□ NOR \equiv OR-NOT

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



NAND & NOR Gates

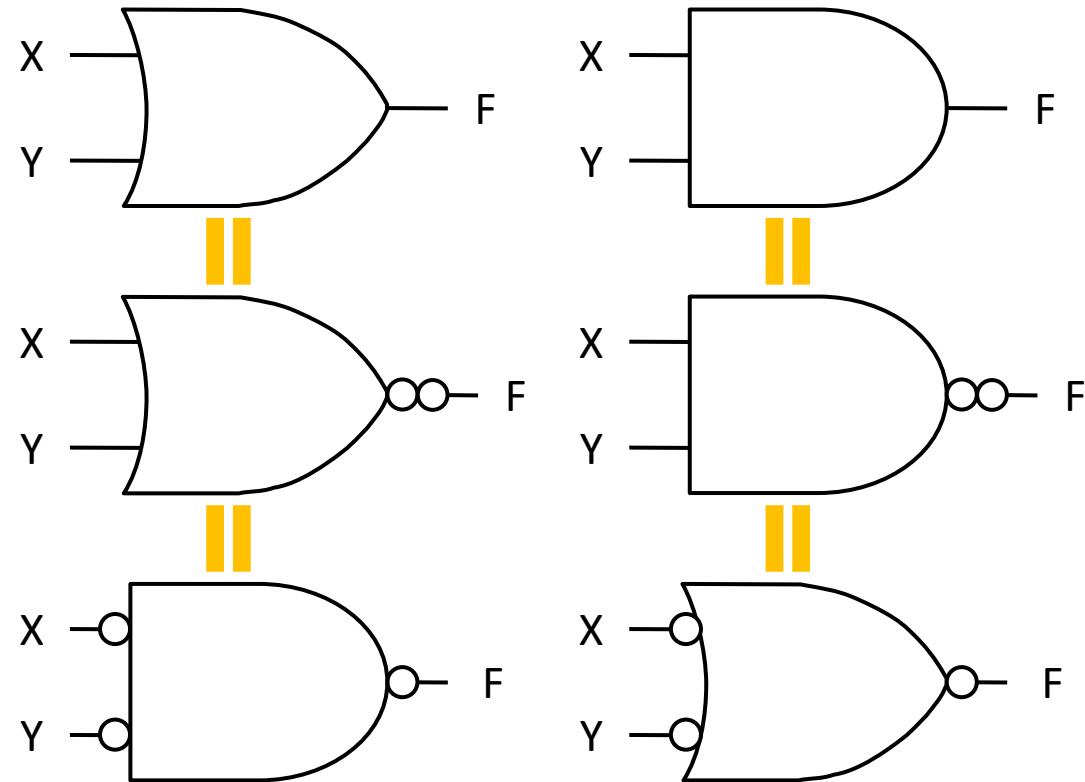
- ❑ Actually, NAND & NOR are implemented by fewer switch devices and operate faster than AND & OR gates

- Lower cost
- Smaller delay

Functionally Complete Set (1/2)

- ❑ A set of logic operations is functionally complete if any Boolean function can be expressed by this set
- ❑ Examples

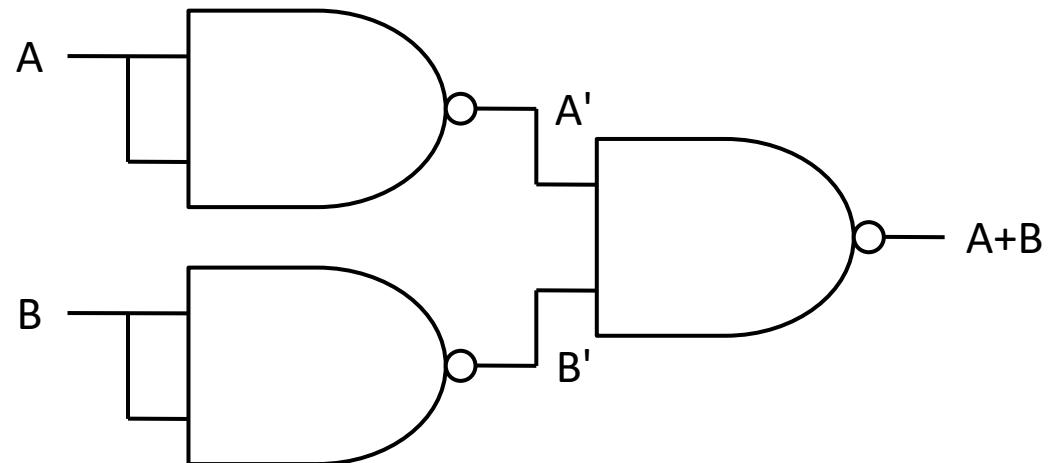
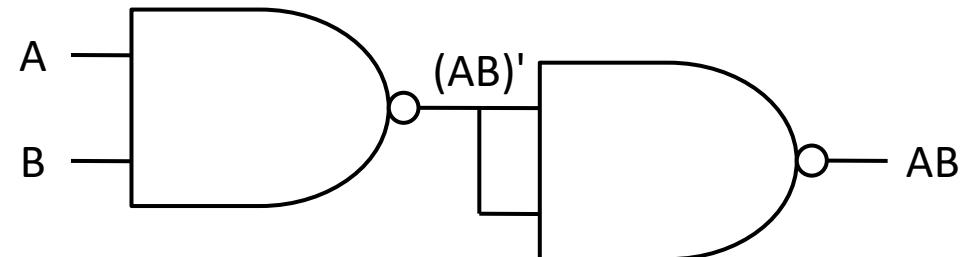
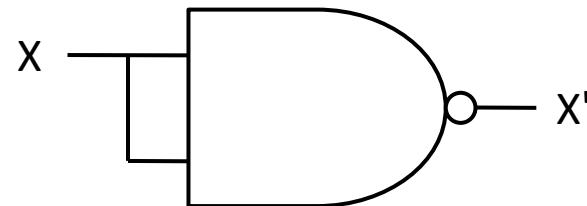
- Q1: {AND, OR, NOT}
- A1: Of course, yes!
- Q2: {AND, NOT}
- A2: Yes (check figure)
 - $X + Y = (X + Y)'' = (X'Y')'$
- Q3: {OR, NOT}
- A3: Yes (check figure)
 - $XY = (XY)'' = (X' + Y')'$
- Q4: {AND}
- Q5: {OR}



Functionally Complete Set (2/2)

❑ Examples

- Q6: {NAND}
- A6: Yes (check figure)
- Q7: {NOR}
- A7: Yes
- Q8: {AND, OR}



Outline

- ❑ 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

Recap: DeMorgan's Law

□ General

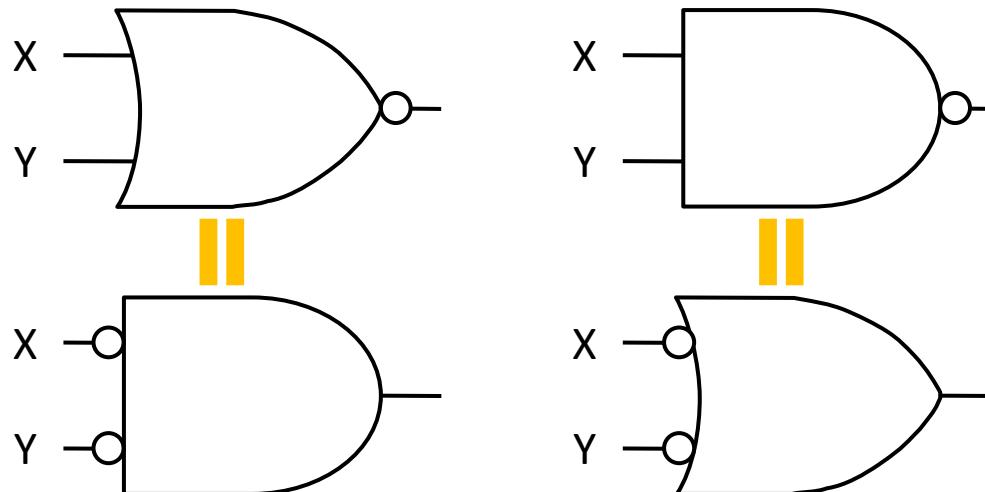
- $(X_1 + X_2 + \dots + X_n)' = X_1'X_2'\dots X_n'$
- $(X_1X_2\dots X_n)' = X_1' + X_2' + \dots + X_n'$

□ One-step rule

- $[F(X_1, X_2, \dots, X_n, 0, 1, +, \bullet)]' = F(X_1', X_2', \dots, X_n', 1, 0, \bullet, +)$
 - $X \leftrightarrow X'; + \leftrightarrow \bullet; 0 \leftrightarrow 1$

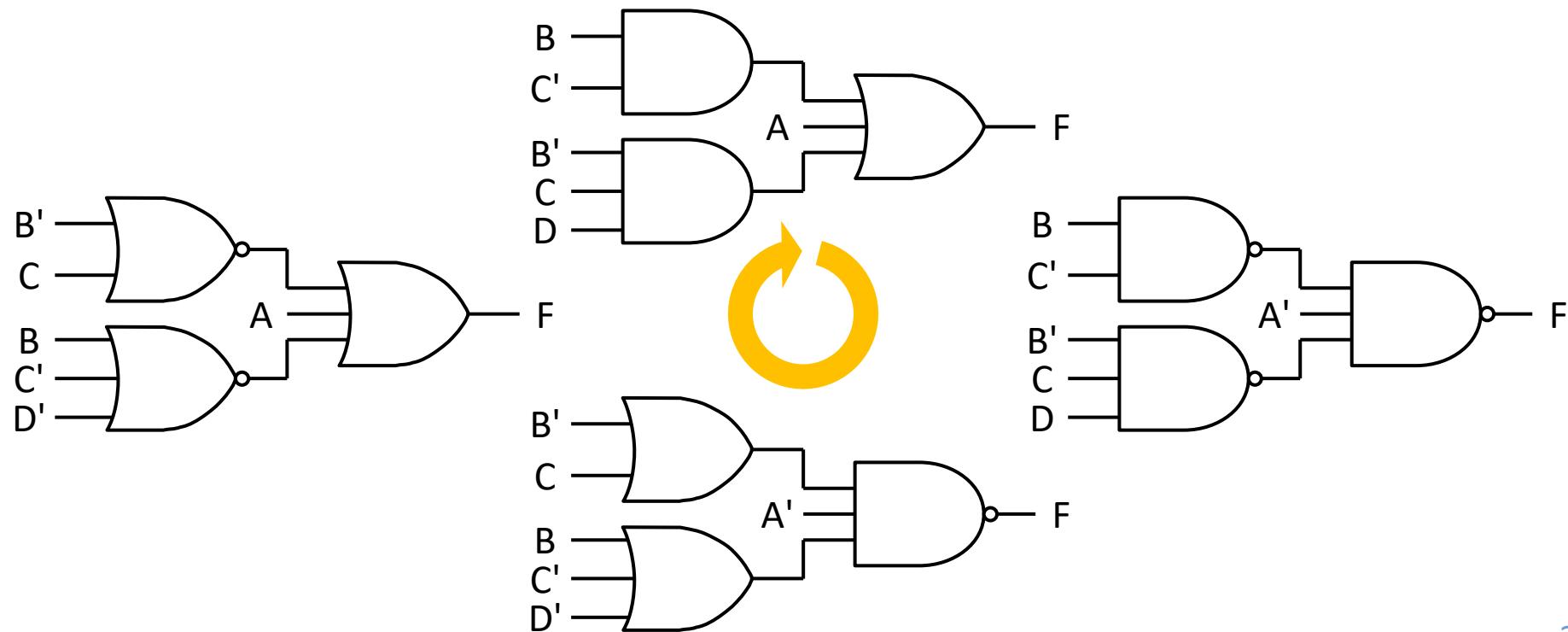
□ Idea

- Move "bubbles" and exchange AND gates and or gates



8 Basic Forms for 2-Level Circuits (1/2)

- ❑ AND-OR \leftrightarrow AND-(NOT-NAND) \leftrightarrow NAND-NAND
- ❑ NAND-NAND \leftrightarrow (NOT-OR)-NAND \leftrightarrow OR-NAND
- ❑ OR-NAND \leftrightarrow OR-(NOT-OR) \leftrightarrow NOR-OR
- ❑ NOR-OR \leftrightarrow (NOT-AND)-OR \leftrightarrow AND-OR



8 Basic Forms for 2-Level Circuits (2/2)

□ OR-AND \leftrightarrow

OR-(NOT-NOR) \leftrightarrow

NOR-NOR \leftrightarrow

(NOT-AND)-NOR \leftrightarrow

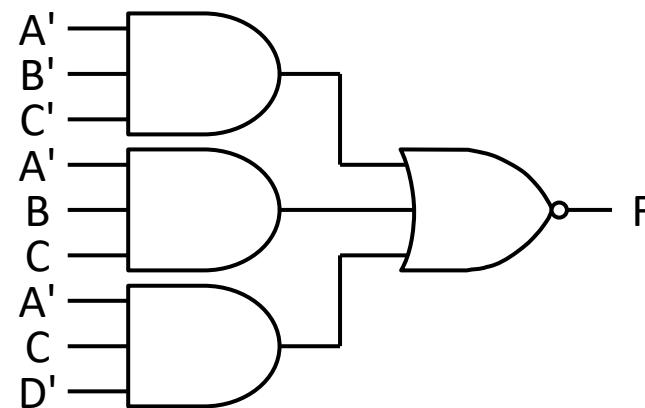
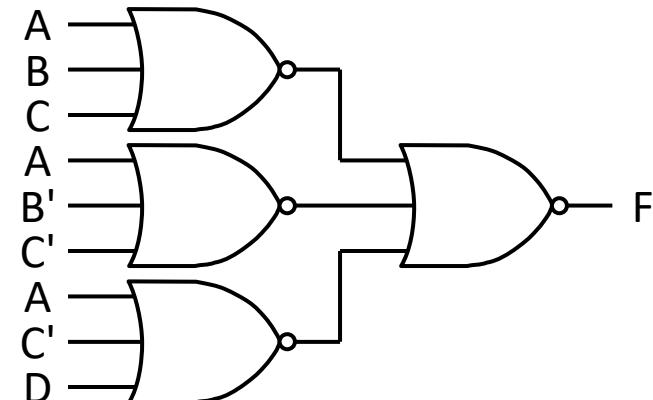
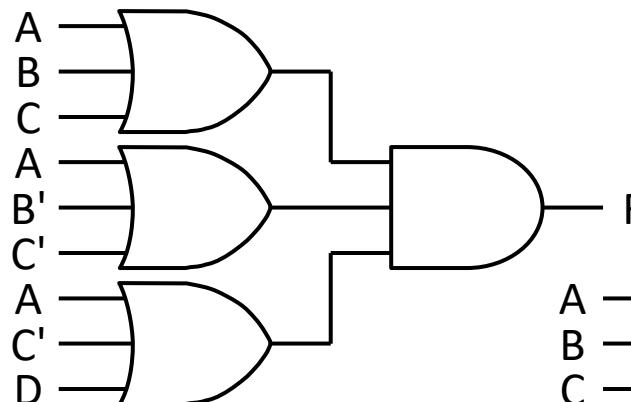
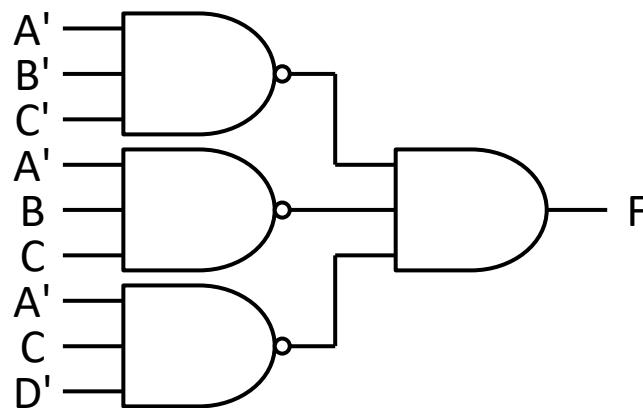
AND-NOR \leftrightarrow

AND-(NOT-AND) \leftrightarrow

NAND-AND \leftrightarrow

(NOT-OR)-AND \leftrightarrow

OR-AND



Other Forms

- ❑ Other 8 forms are degenerated, i.e., cannot realize all switching functions
 - AND-AND, AND-NAND
 - OR-OR, OR-NOR
 - NAND-OR, NAND-NOR
 - NOR-AND, NOR-NAND
- ❑ Example
 - NAND-NOR \equiv AND-AND...

Example: NAND-NAND & NOR-NOR

❑ Procedure for designing a min 2-level NAND-NAND circuit

- Find a minimum SOP
- Draw AND-OR (2-level) circuit
- Convert into NAND-NAND circuit
 - Introduce 2 bubbles into each intermediate line or the output

❑ Procedure for designing a min 2-level NOR-NOR circuit

- Find a minimum POS
- Draw OR-AND (2-level) circuit
- Convert into NOR-NOR circuit
 - Introduce 2 bubbles into each intermediate line or the output

❑ Examples

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

Summary: 2-Level Logic Minimization

- ❑ 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)
 - $\# \text{ of gates} \leq \# \text{ of terms} + 1$
 - $\# \text{ of gate inputs} \leq \# \text{ of literals} + \# \text{ of terms}$
 - " $<$ " holds if a term is degenerated to a single variable
- ❑ Convert the minimum SOP (POS) to the target form
 - 8 basic forms can convert each other
 - DeMorgan's law
 - Use bubbles
- ❑ NAND/NOR gates: low cost & high speed

Outline

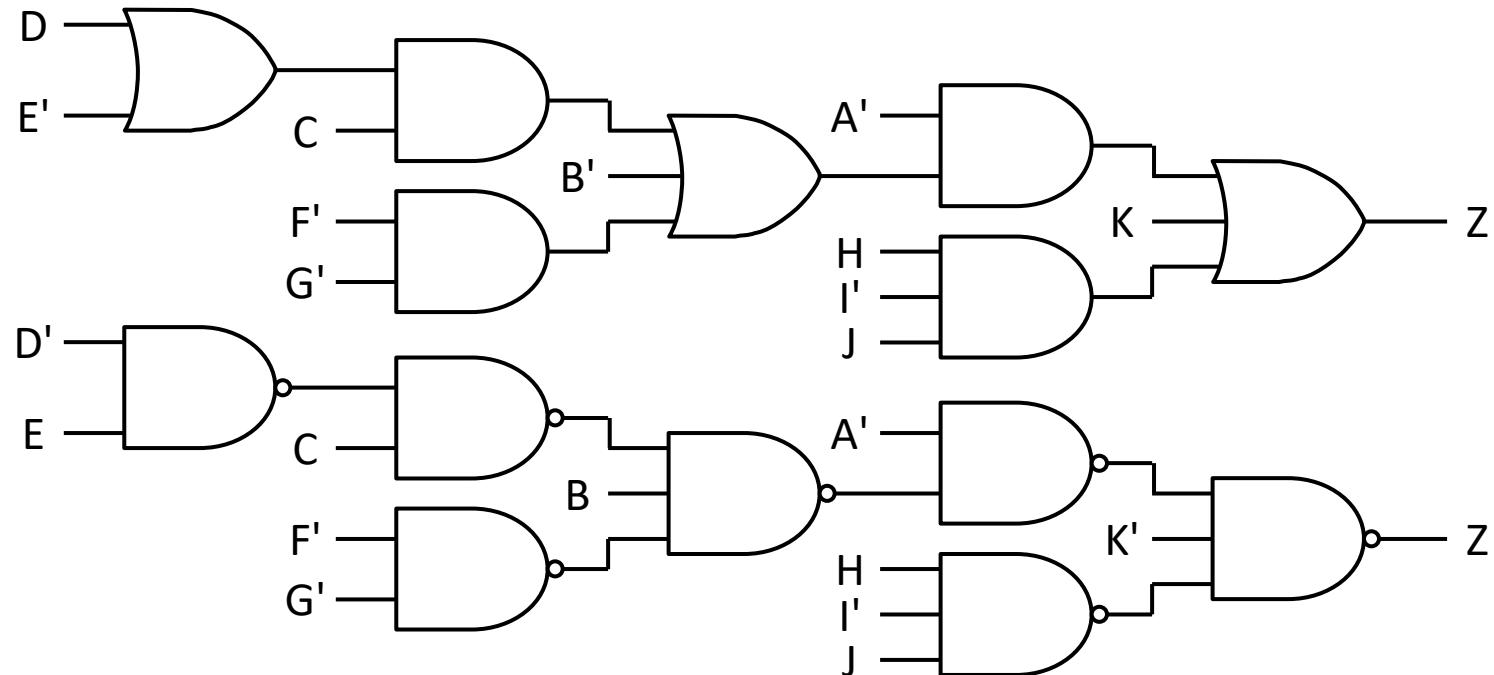
- ❑ Multi-Level Gate Circuits
- ❑ NAND and NOR Gates
- ❑ Two-Level NAND- and NOR-Gate Circuits
- ❑ Multi-Level NAND- and NOR-Gate Circuits**
- ❑ Circuit Conversion Using Alternative Gate Symbols
- ❑ Design of Two-Level, Multiple-Output Circuits
- ❑ Multiple-Output NAND and NOR Circuits

Multi-Level NAND-Gate Circuits

❑ Procedure for designing a multi-level NAND circuit

- Simplify the switching function
- Draw its multi-level AND-OR circuit
- Replace all gates with NANDs and number all levels (from the output)
- Invert inputs to levels 1, 3, 5, ...

❑ Example: $Z = A'[B' + C(D + E') + F'G'] + HI'J + K$

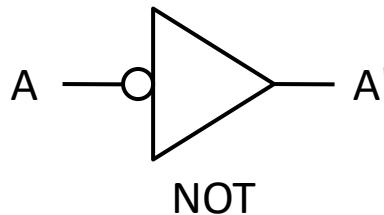


Outline

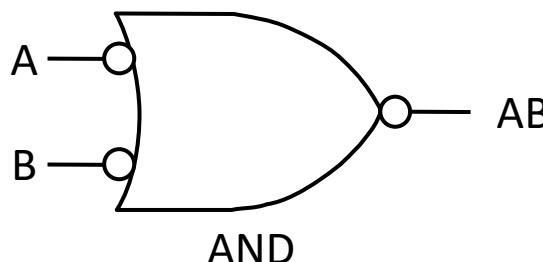
- ❑ Multi-Level Gate Circuits
- ❑ NAND and NOR Gates
- ❑ Two-Level NAND- and NOR-Gate Circuits
- ❑ Multi-Level NAND- and NOR-Gate Circuits
- ❑ Circuit Conversion Using Alternative Gate Symbols**
- ❑ Design of Two-Level, Multiple-Output Circuits
- ❑ Multiple-Output NAND and NOR Circuits

Alternative Gate Symbols

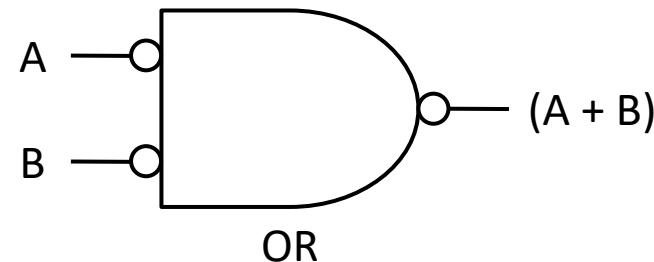
- ❑ Do NOT memorize these gate symbols
- ❑ DeMorgan's law is the key of conversion
 - Use bubbles adequately



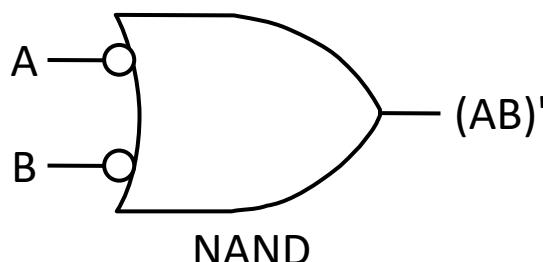
NOT



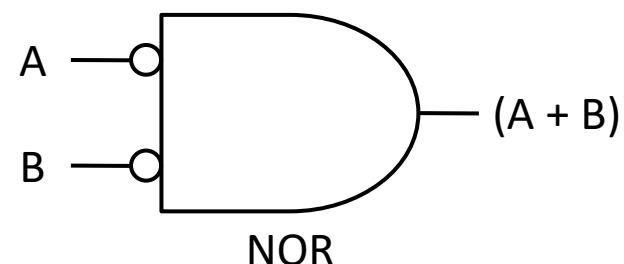
AND



OR

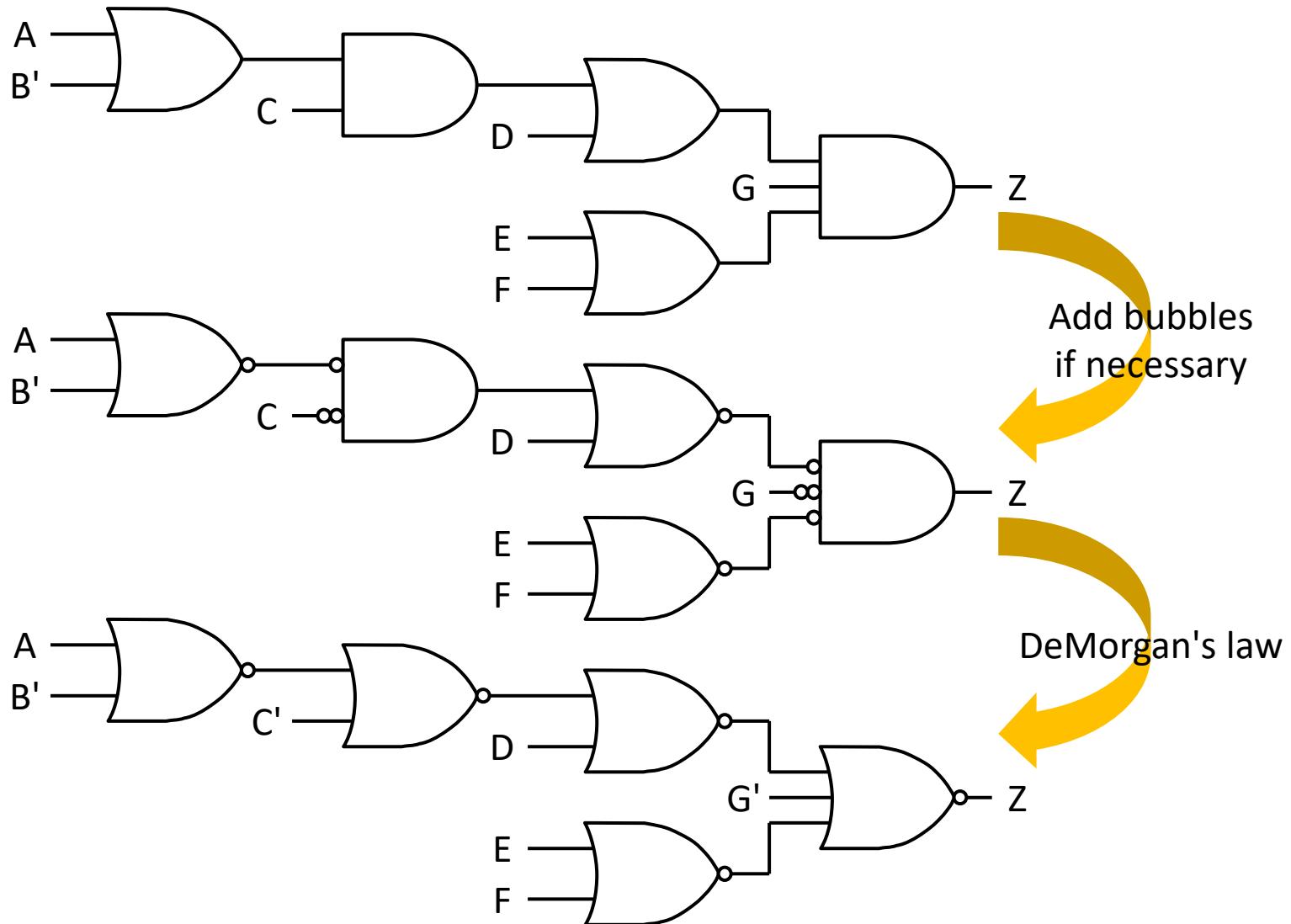


NAND

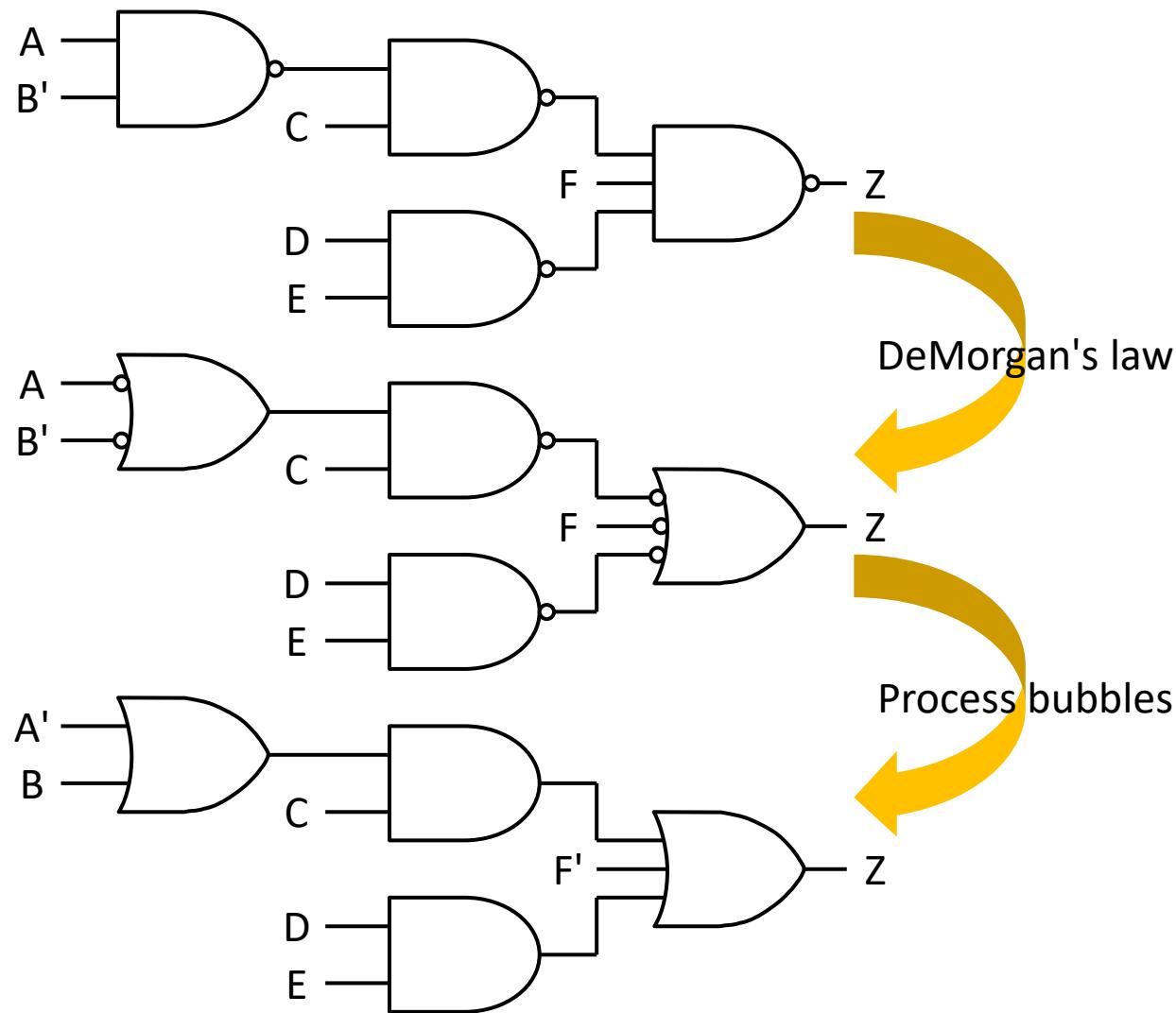


NOR

Example from OR-AND to NOR Gates



Example from NAND to AND-OR Gates



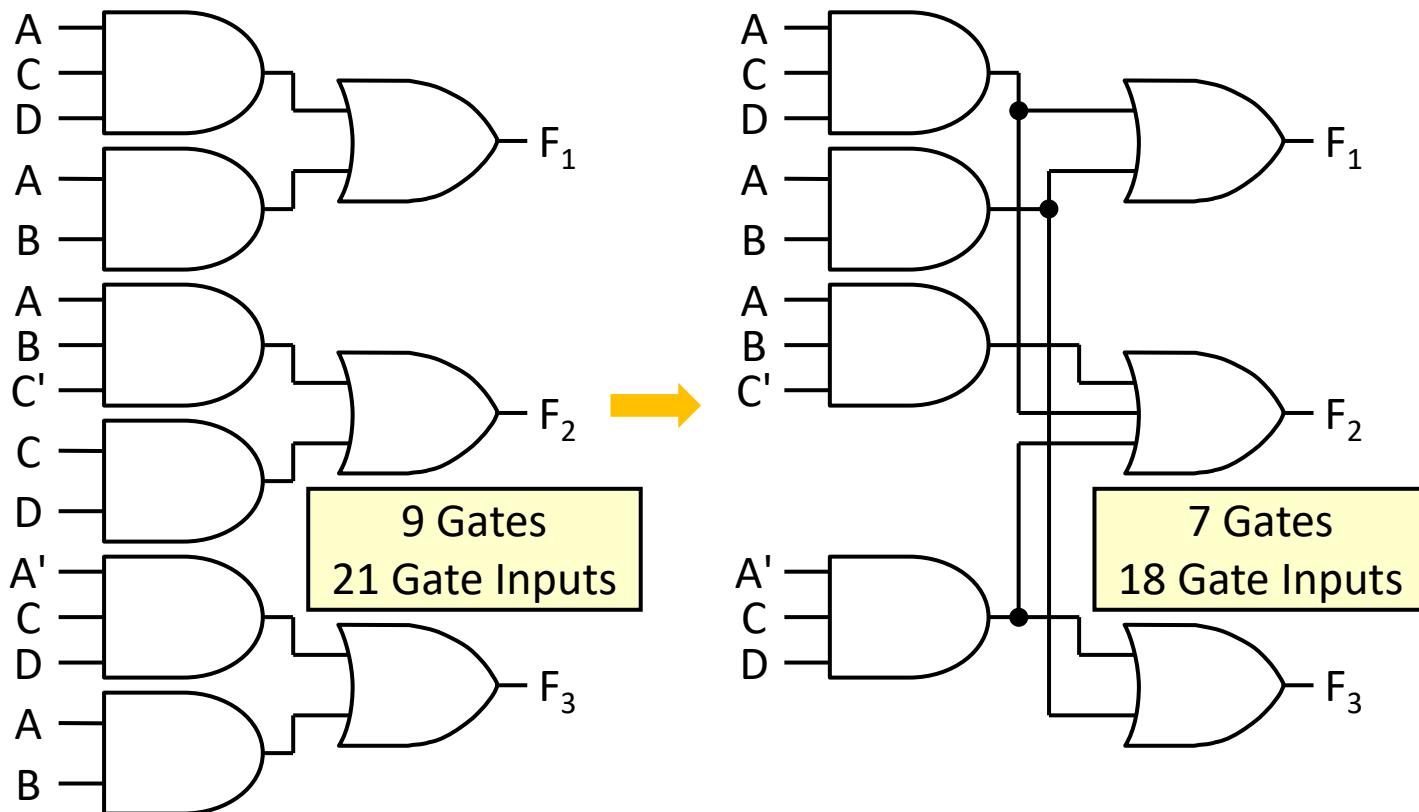
Outline

- ❑ 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

Case 1: 4 Inputs and 3 Outputs

□ Example

- $F_1(A,B,C,D) = \sum m(11, 12, 13, 14, 15) = ACD + AB$
- $F_2(A,B,C,D) = \sum m(3, 7, 11, 12, 13, 15) = ABC' + CD = ABC' + \underline{ACD} + \underline{A'CD}$
- $F_3(A,B,C,D) = \sum m(3, 7, 12, 13, 14, 15) = A'CD + AB$



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

□ Example

- $F_1(A,B,C,D) = \sum m(2, 3, 5, 7, 8, 9, 10, 11, 13, 15) = BD + B'C + AB'$
- $F_2(A,B,C,D) = \sum m(2, 3, 5, 6, 7, 10, 11, 14, 15) = C + A'BD$
- $F_3(A,B,C,D) = \sum m(6, 7, 8, 9, 13, 14, 15) = BC + AB'C' + ABD$ (or $AC'D$)

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

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

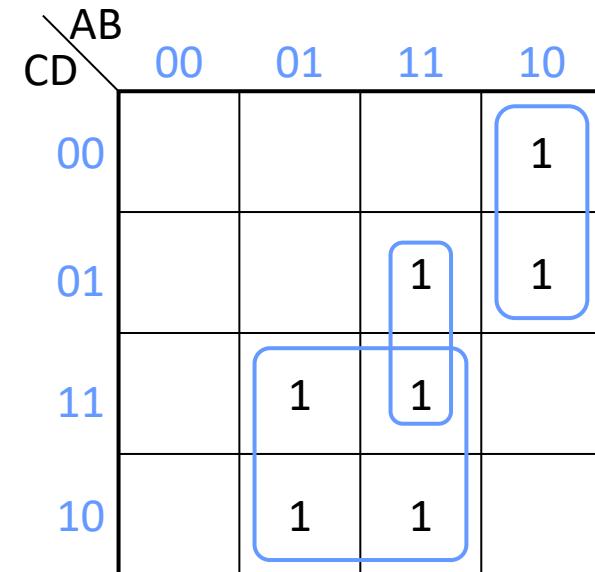
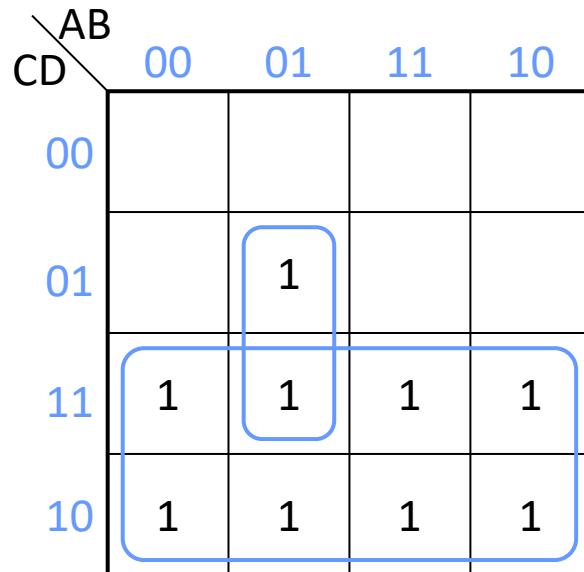
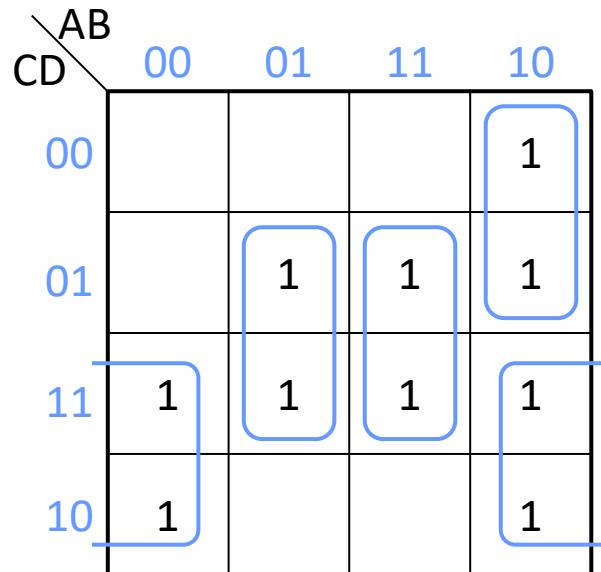
AB	CD	00	01	11	10
CD	00				
00					
01				1	
11		1	1		1
10		1	1	1	

10 Gates
25 Gate Inputs

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

□ Find common minterms from K-map

- $F_1(A,B,C,D) = \underline{BD} + B'C + \underline{AB'} = \underline{\underline{ABD}} + \underline{A'BD} + B'C + \underline{\underline{AB'C'}}$
- $F_2(A,B,C,D) = C + \underline{A'BD}$
- $F_3(A,B,C,D) = BC + \underline{AB'C'} + \underline{ABD}$



8 Gates
22 Gate Inputs

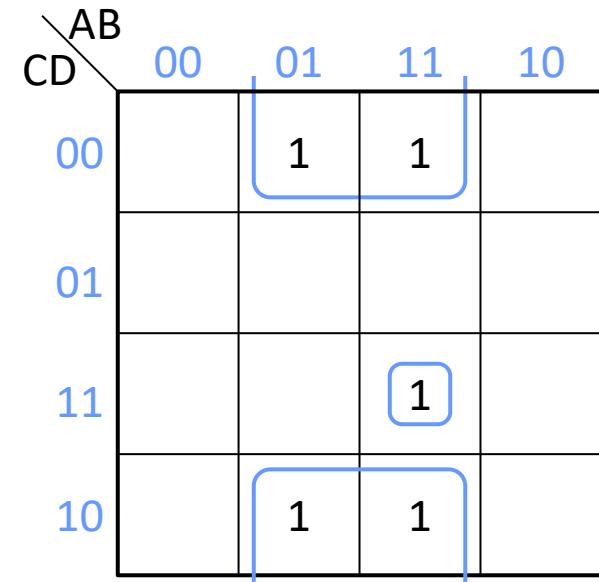
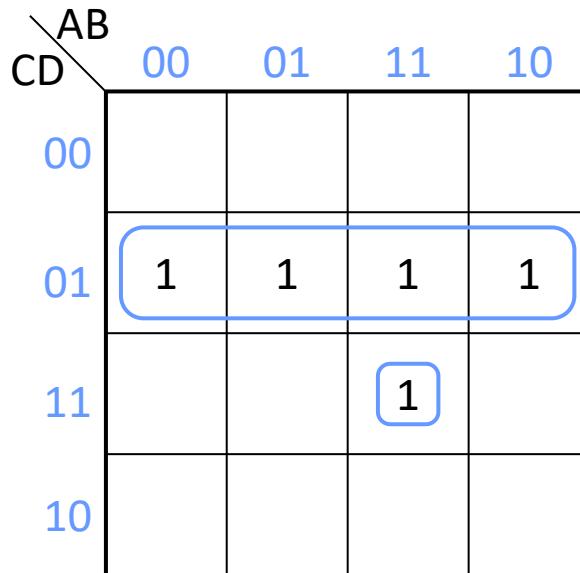
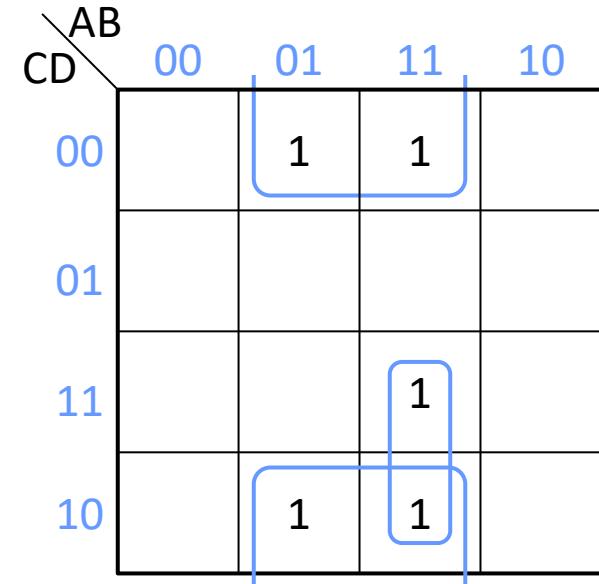
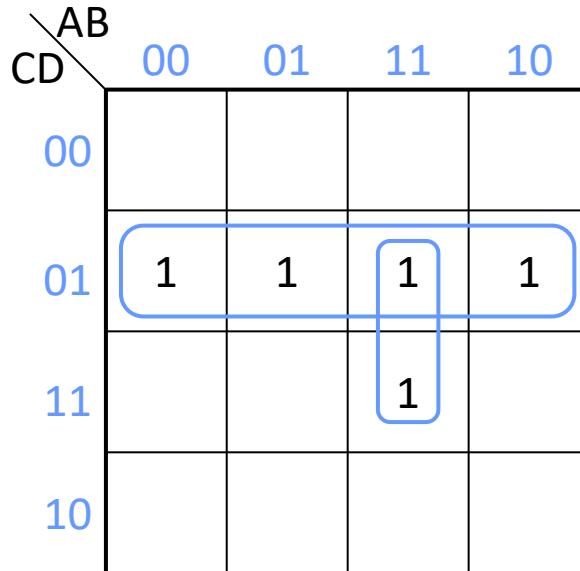
Case 3: Essential Prime Implicants

- ❑ Essential for multi-output

- Only check 1's which do not appear in other maps
- Example

- $C'D$ and BD' are essential
- ABD and ABC are not

Reduce one gate by sharing ABCD



Case 4

- ❑ Having more common terms is not always better

8 Gates
26 Gate Inputs

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

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

7 Gates
18 Gate Inputs

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

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

Summary: 2-Level Multiple-Output Circuits

❑ Cost depends on

- The total # of gates
- The total # of gate inputs

❑ Cost minimization

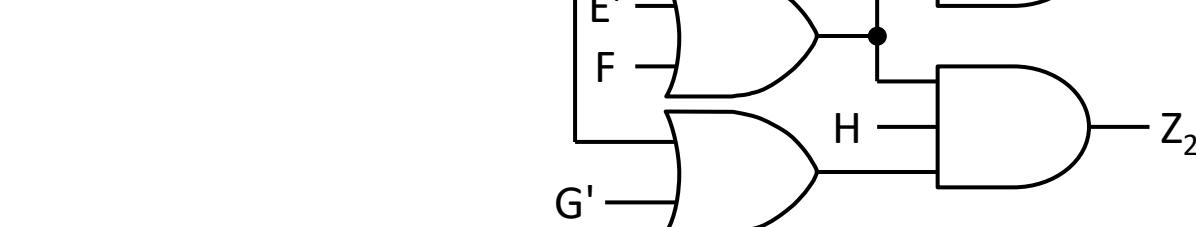
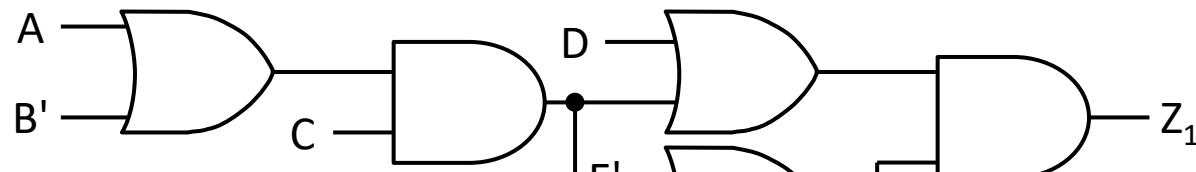
- A global view may help
 - A minimum SOP for each cannot guarantee minimum cost for all
- Common and essential terms may help
 - Sometimes, we do not need large circles
 - "Essential" is defined in a different way
- Very difficult to find global optimal solution
 - Boolean algebra is very useful in more sophisticated techniques (advanced courses)

Outline

- ❑ 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**

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
 - Example: $Z_1 = [(A + B')C + D](E' + F)$ and $Z_2 = [(A + B')C + G'](E' + F)H$



Q&A