ECE 2050 Digital Logic and Systems

Chapter 5 : Combinational Logic Design

Instructor: Yue ZHENG, Ph.D.

香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

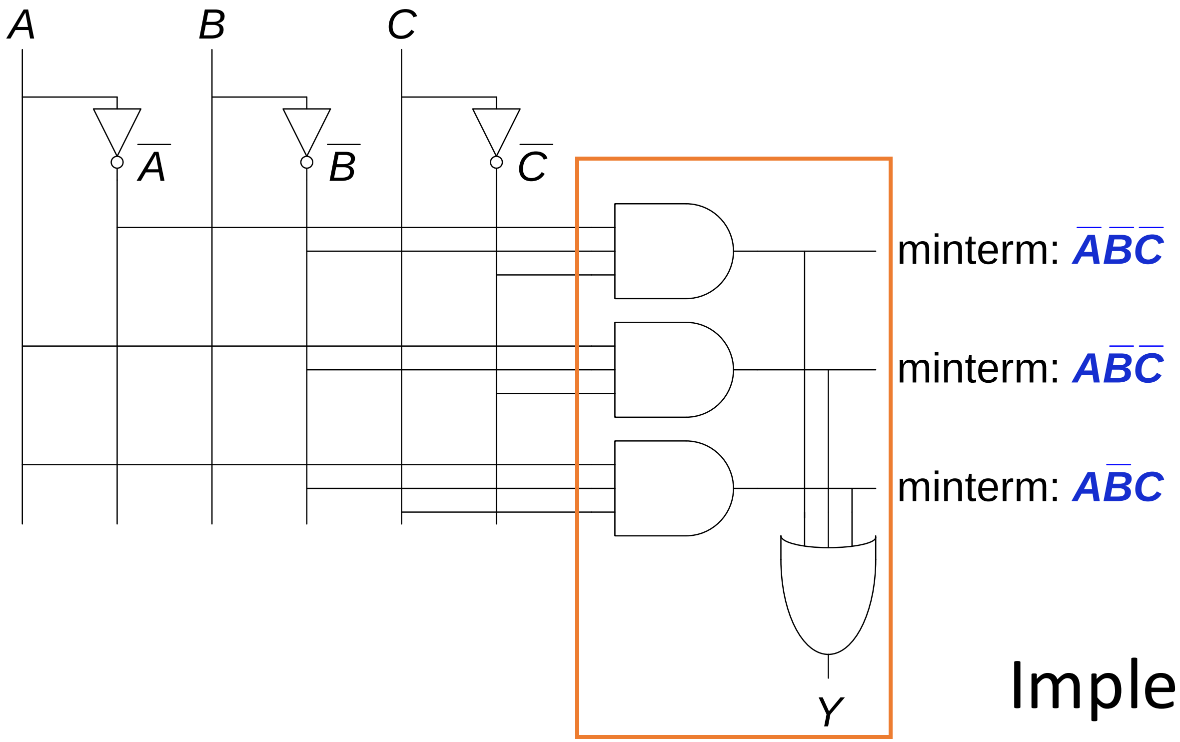
Last Week

- Combinational Circuits
- Boolean Equations
- Axioms & Theorems
  - Commutative laws
  - Associative laws
  - Distributive law
  - Rules of Boolean Algebra
  - DeMorgan's Theorems
- Simplifying Equations
  - SOP and POS
- Karnaugh Maps
  - Don't Cares

Basic Combinational Logic Circuits

AND-OR Logic

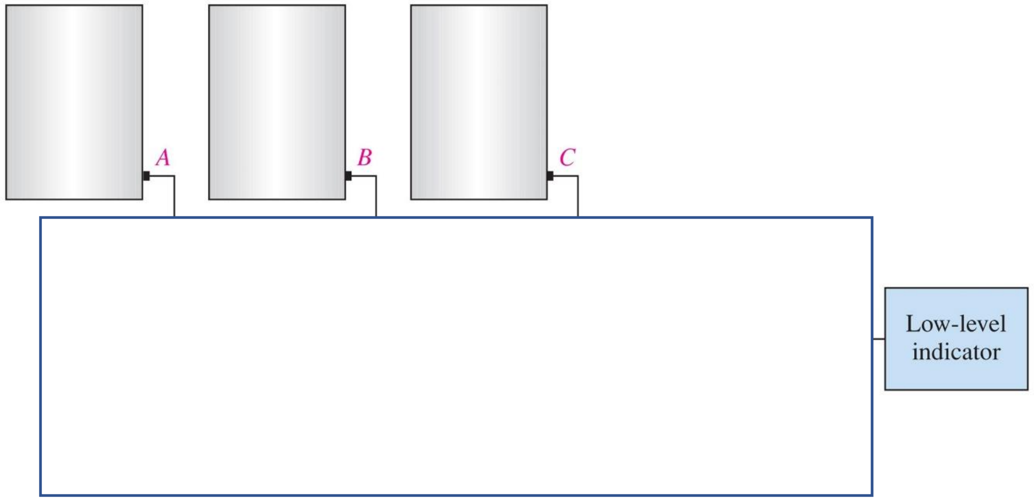
- Two-level logic: **ANDs** followed by **ORs**
- Example: $Y = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + A\overline{B}C$



Implements functions in **SOP form**

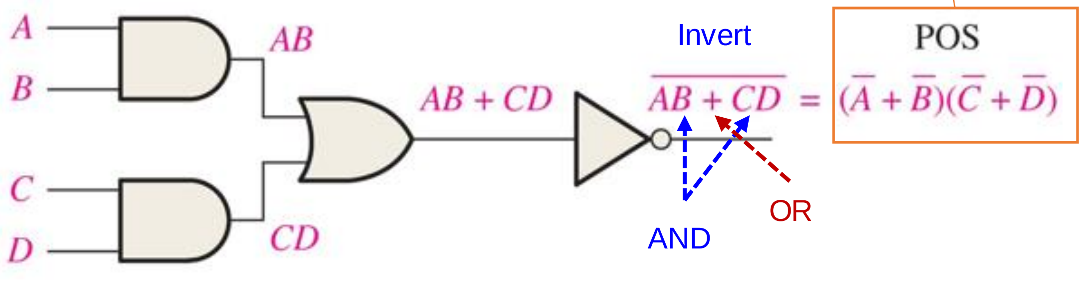
AND-OR Logic: Example

A level sensor in each tank produces a HIGH voltage when the level of chemical in the tank drops below a specified point. Design a circuit that monitors the chemical level in each tank and indicates when the level in **any two of the tanks** drops below the specified point.



AND-OR-Invert Logic

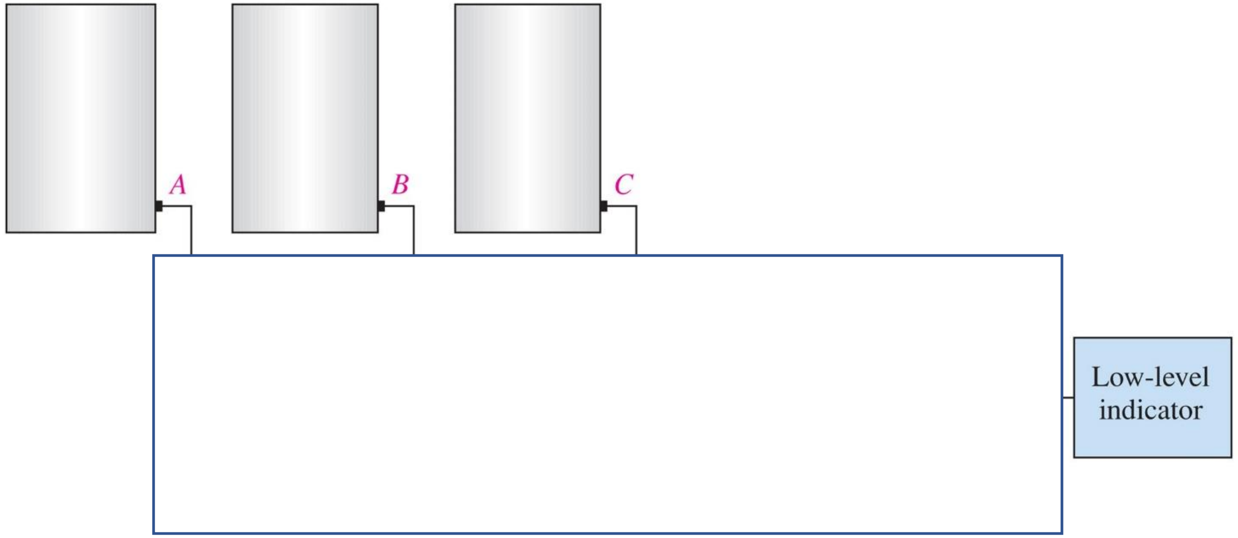
- Two-level logic: **ORs** followed by **ANDs**
- Example: $Y = \overline{AB+CD} = (\overline{A}+\overline{B})(\overline{C}+\overline{D})$



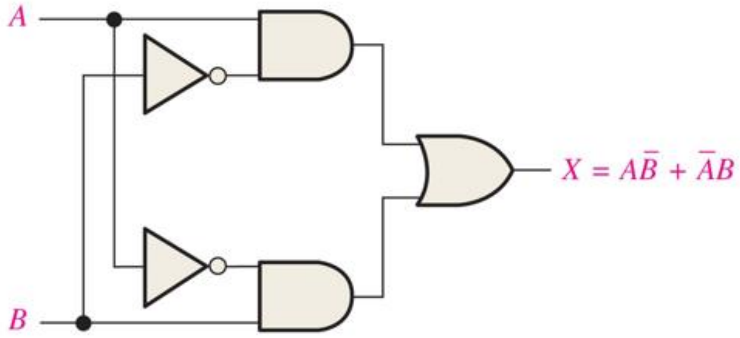
Implements functions in **POS form**

AND-OR-Inverter Logic: Example

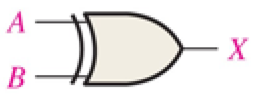
A level sensor in each tank produces a **LOW** voltage when the level of chemical in the tank drops below a specified point. Design a circuit that monitors the chemical level in each tank and indicates when the level in any two of the tanks drops below the specified point.



Exclusive OR Logic



(a) Logic diagram



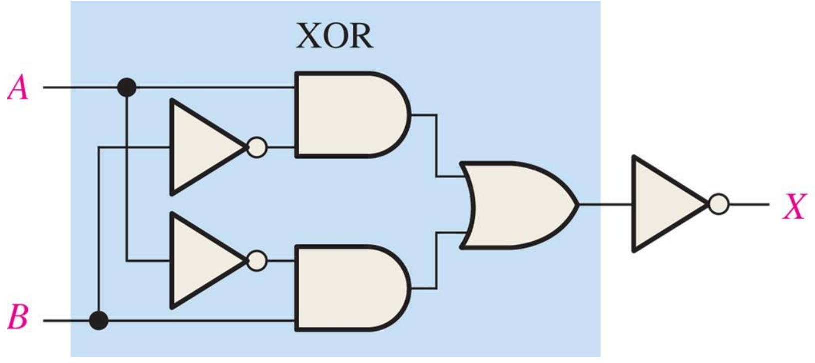
(b) ANSI distinctive shape symbol

$$X = A\bar{B} + \bar{A}B = A \oplus B$$

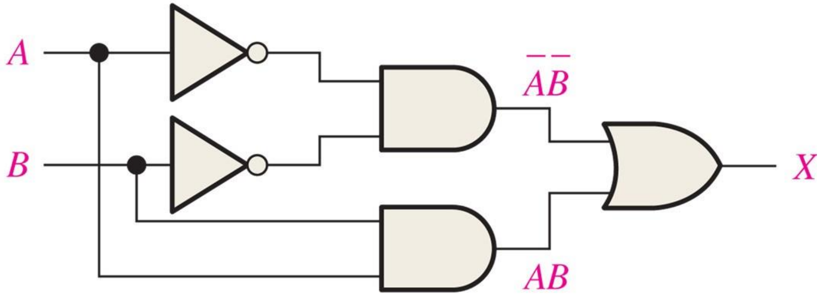
Truth Table for an Exclusive-OR

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Exclusive-NOR Logic



(a) $X = \overline{A\overline{B} + \overline{A}B}$



(b) $X = \overline{A}\,\overline{B} + AB$

Truth Table for an Exclusive-NOR

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

$$X = \overline{A\overline{B} + \overline{A}B} = \overline{(A\overline{B})}\;\overline{(\overline{A}B)}$$
$$= (\overline{A} + B)(A + \overline{B}) = \overline{A}\,\overline{B} + AB$$

Parity Codes

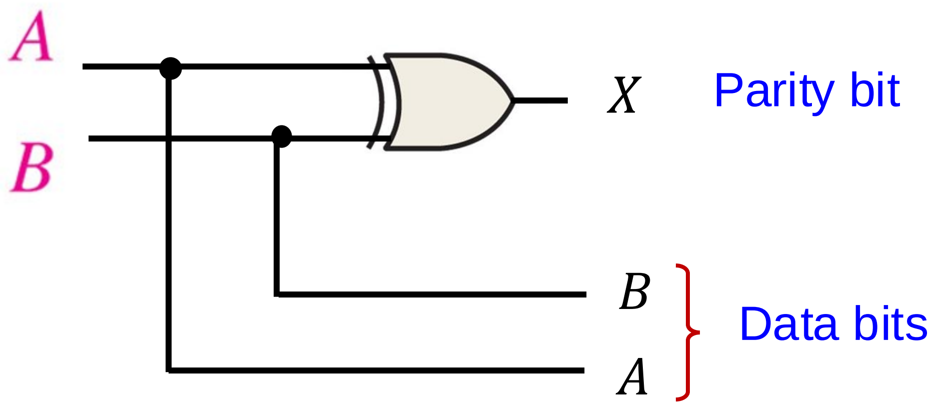
2-bit Even Parity Codes

Truth Table

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

A \ B	0	1
0	0	1
1	1	0

$$X = A\bar{B} + \bar{A}B = A \oplus B$$



3-Bit Even-Parity Codes

Use exclusive-OR gates to implement an even-parity code generator for an original 3-bit code.

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

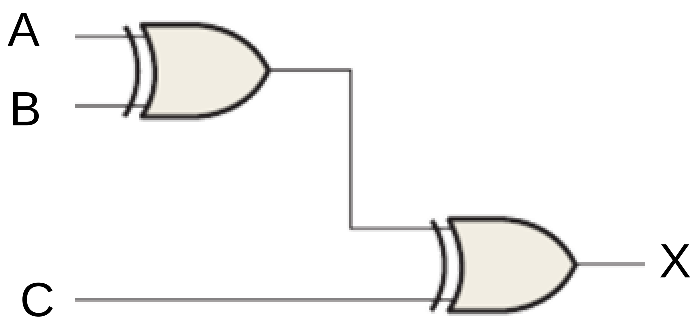
$$X = \underbrace{\bar{A}\bar{B}C + \bar{A}B\bar{C}} + \underbrace{A\bar{B}\bar{C} + ABC}$$

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

Recall $\bar{A}\bar{B} + AB = \overline{A \oplus B}$ XNOR

$A\bar{B} + \bar{A}B = A \oplus B$ XOR

$$= \bar{A}(\boxed{B \oplus C}) + A(\boxed{\overline{B \oplus C}}) \quad = A \oplus B \oplus C$$

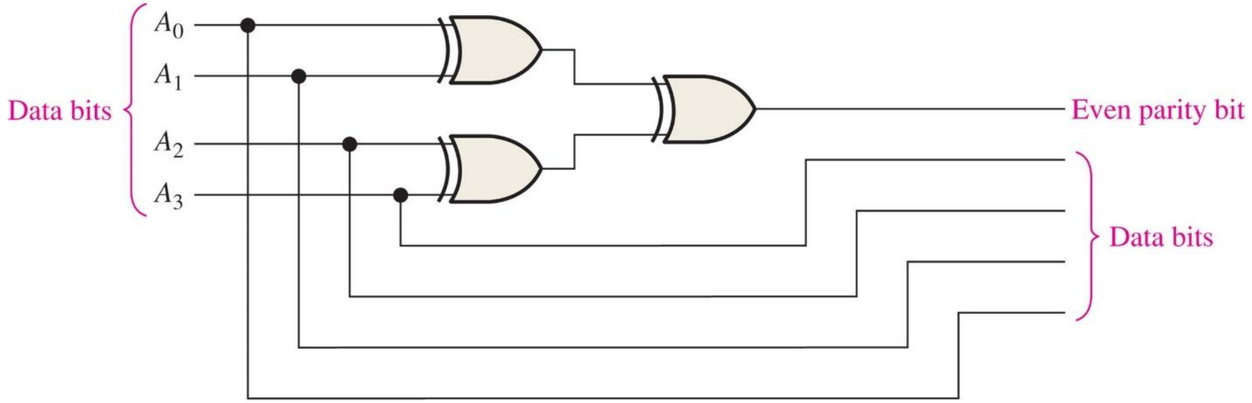


Truth Table

A	B	C	X
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

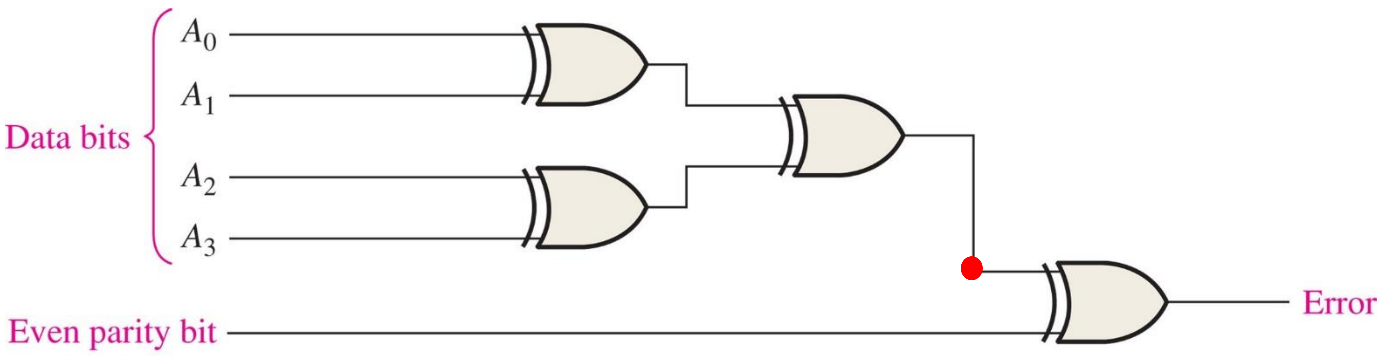
4-bit Even-Parity Code Generator

Use XOR gates to implement an even-parity code generator for an original 4-bit code.



5-bit Even-Parity Code Generator

Use XOR gates to implement an even-parity checker for the 5-bit code generated by the 4-bit even-parity code generator.



The circuit produces a 1 output when there is an error in the five-bitcode and a 0 when there is no error.

1 → odd, parity bit → 1, error → 0; 1 → odd, parity bit → 0, error → 1
0 → even, parity bit → 1, error → 1; 0 → even, parity bit → 0, error → 0

Implementing Combinational Circuits

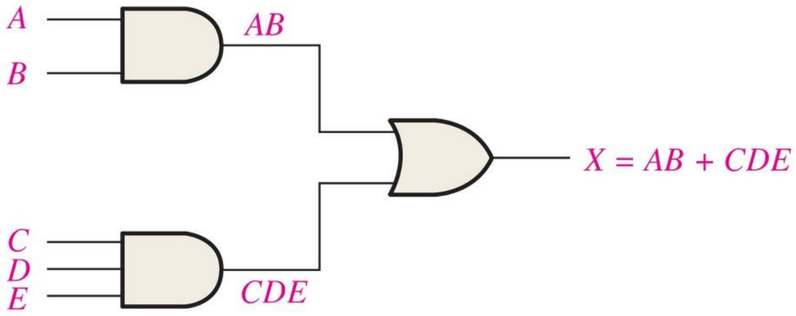
Implementing Combinational Logic

❑ From a Boolean Expression to a Logic Circuit

AND

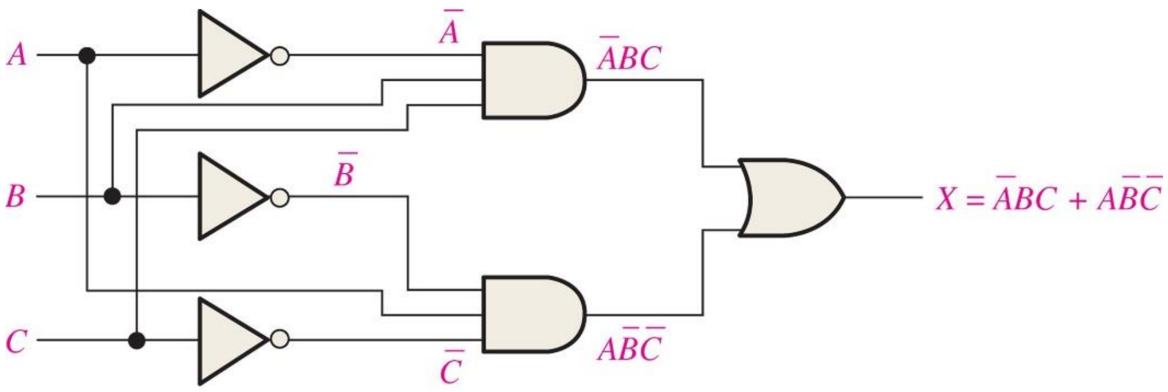
$$X = \boxed{AB} + \boxed{CDE}$$

OR



❑ From a Truth Table to a Logic Circuit

Inputs			Output	
A	B	C	X	Product Term
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\overline{A}BC$
1	0	0	1	$A\overline{B}\,\overline{C}$
1	0	1	0	
1	1	0	0	
1	1	1	0	

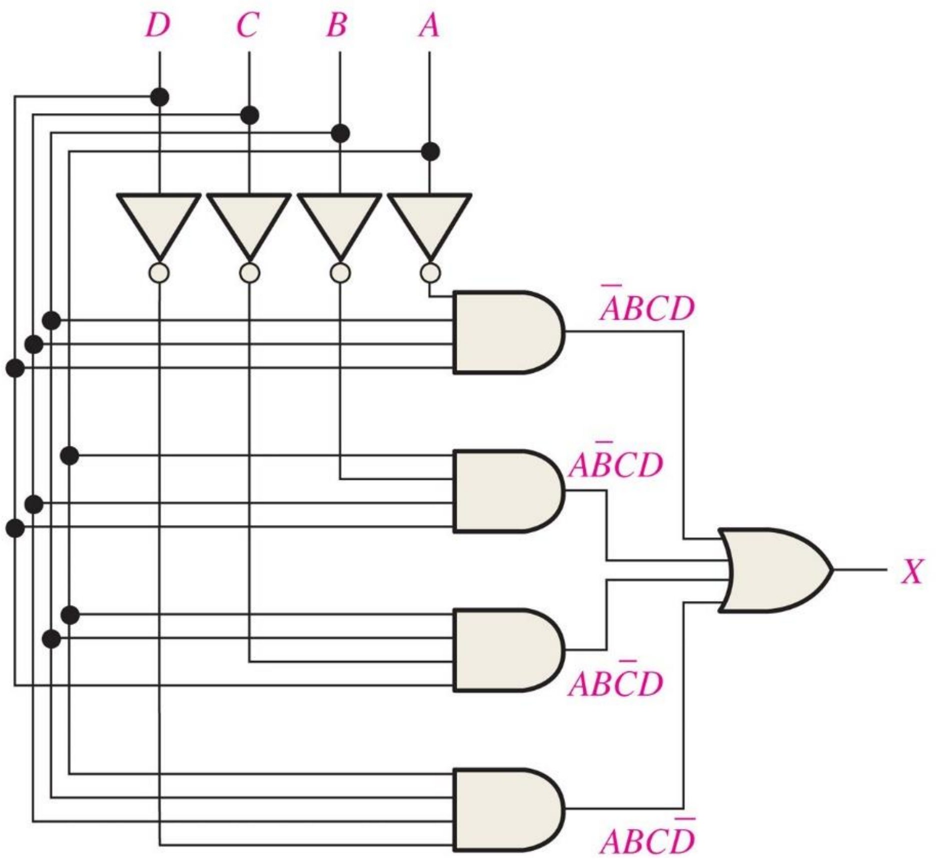


Example 1

Develop a logic circuit with four input variables that will only produce a 1 output when exactly three input variables are 1s.

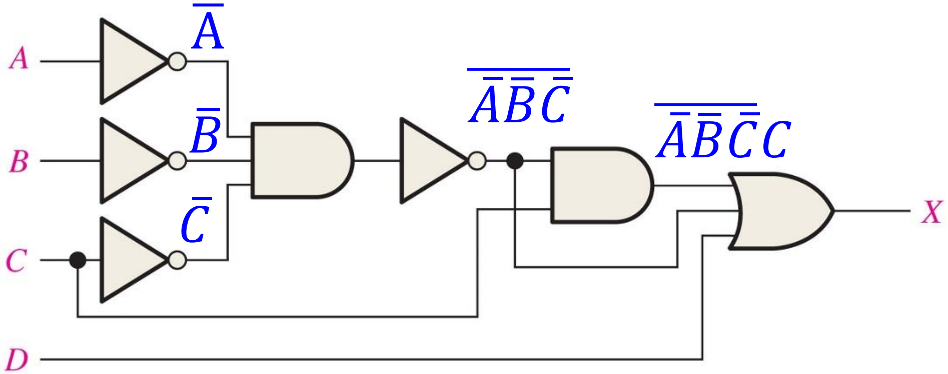
A	B	C	D	Product Term
0	1	1	1	$\bar{A}BCD$
1	0	1	1	$A\bar{B}CD$
1	1	0	1	$AB\bar{C}D$
1	1	1	0	$ABC\bar{D}$

$$X = \bar{A}BCD + A\bar{B}CD + AB\bar{C}D + ABC\bar{D}$$



Example 2

Reduce the combinational logic circuit to a minimum form.

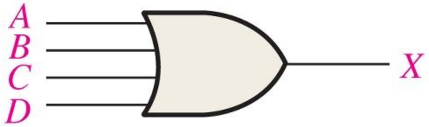


$$X = \overline{\bar{A}\bar{B}\bar{C}}C + \overline{\bar{A}\bar{B}\bar{C}} + D$$

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

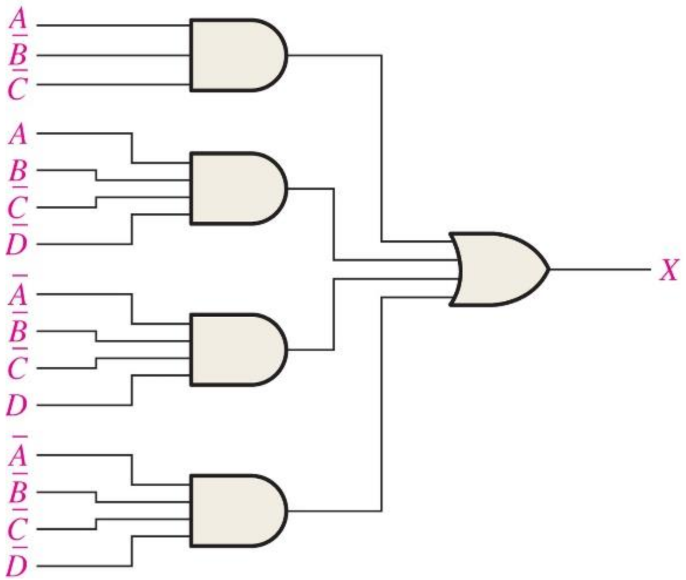
$$= (A+B+C) + D$$

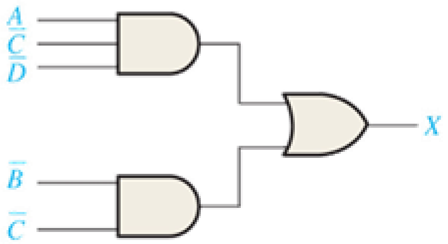
$$= A+B+C + D$$



Example 3

Reduce the combinational logic circuit to a minimum form.

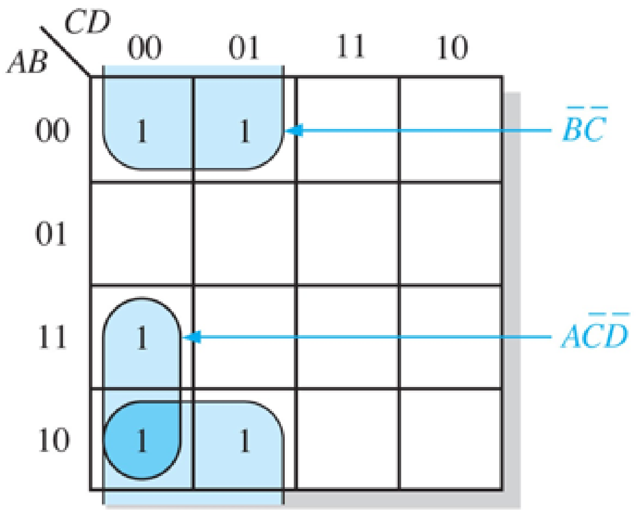




Basic rules of Boolean algebra.

1. $A + 0 = A$	**7.** $A \cdot A = A$
2. $A + 1 = 1$	**8.** $A \cdot \bar{A} = 0$
3. $A \cdot 0 = 0$	**9.** $\bar{\bar{A}} = A$
4. $A \cdot 1 = A$	**10.** $A + AB = A$
5. $A + A = A$	**11.** $A + \bar{A}B = A + B$
6. $A + \bar{A} = 1$	**12.** $(A + B)(A + C) = A + BC$

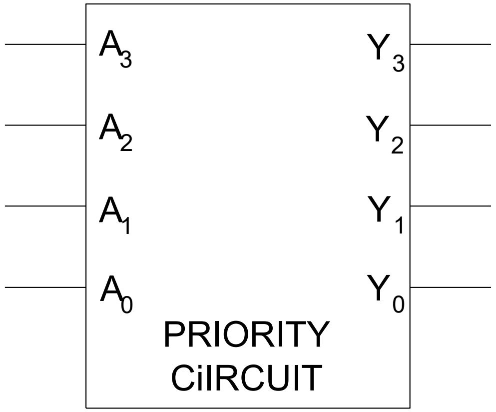
A, *B*, or *C* can represent a single variable or a combination of variables.



$$X = A\bar{B}\bar{C} + AB\bar{C}\bar{D} + \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}\bar{C}\bar{D} = A\bar{C}\bar{D} + \bar{B}\bar{C}$$
$$= A\bar{B}\bar{C}(\mathrm{D} + \bar{D}) + AB\bar{C}\bar{D} + \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}\bar{C}\bar{D}$$

Example 4: Priority Circuit

Output asserted corresponding to most significant TRUE input



A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1				

Priority Circuit Hardware

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0

Don't Cares

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0

$Y_3 = A_3$
$Y_2 = \overline{A_3}\, A_2$
$Y_1 = \overline{A_3}\, \overline{A_2}\, A_1$
$Y_0 = \overline{A_3}\, \overline{A_2}\, \overline{A_1}\, A_0$

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0

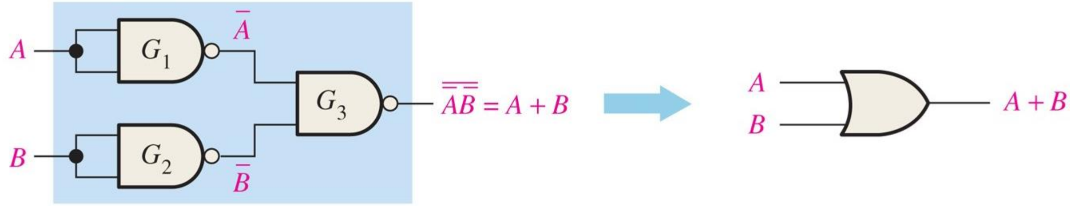
NAND, NOR: Universal Property

The Universal Property of NAND Gates

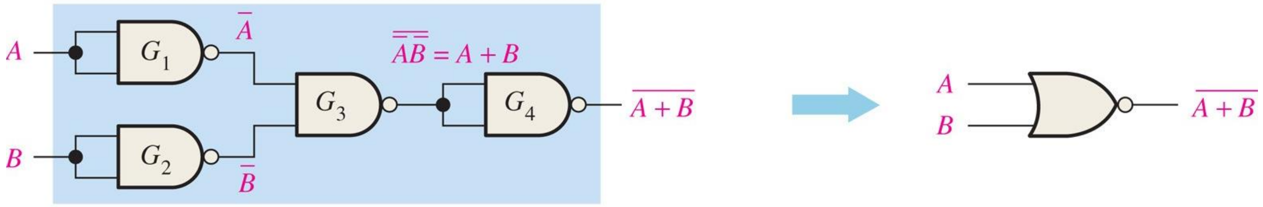
- Combinations of NAND gates can function as NOT, AND, OR, and NOR

(a) One NAND gate used as an inverter

(b) Two NAND gates used as an AND gate



(c) Three NAND gates used as an OR gate



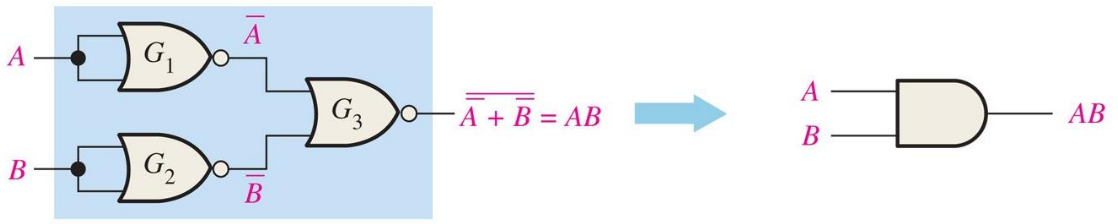
(d) Four NAND gates used as a NOR gate

The Universal Property of NOR Gates

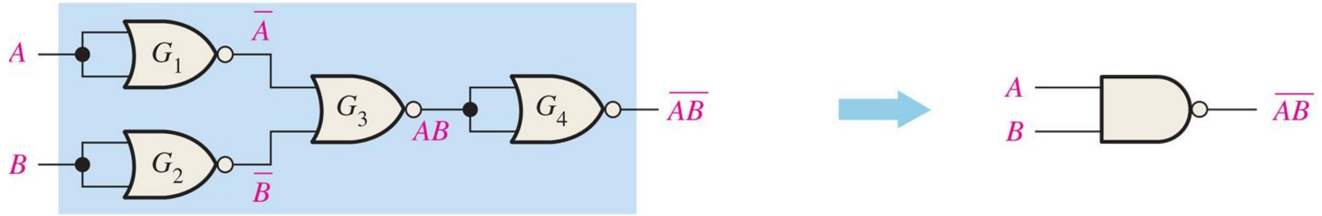
- Combinations of NOR gates can function as NOT, OR, AND, and NAND

(a) One NOR gate used as an inverter

(b) Two NOR gates used as an OR gate



(c) Three NOR gates used as an AND gate

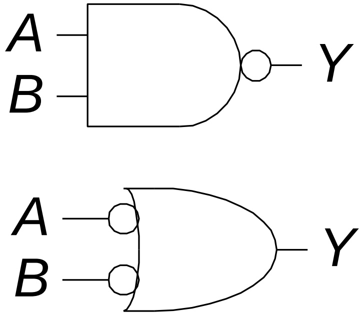


(d) Four NOR gates used as a NAND gate

Bubble Pushing

De Morgan's Theorem: Gates

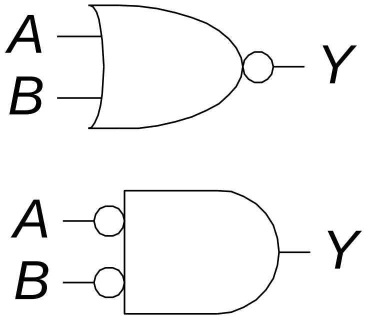
- $Y = \overline{AB} = \overline{A} + \overline{B}$



NAND gate

two forms

- $Y = \overline{A + B} = \overline{A} \bullet \overline{B}$

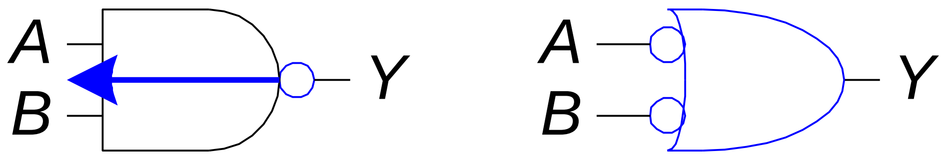


NOR gate

two forms

Bubble Pushing

- **Backward:**
  - Body changes
  - Adds bubbles to inputs

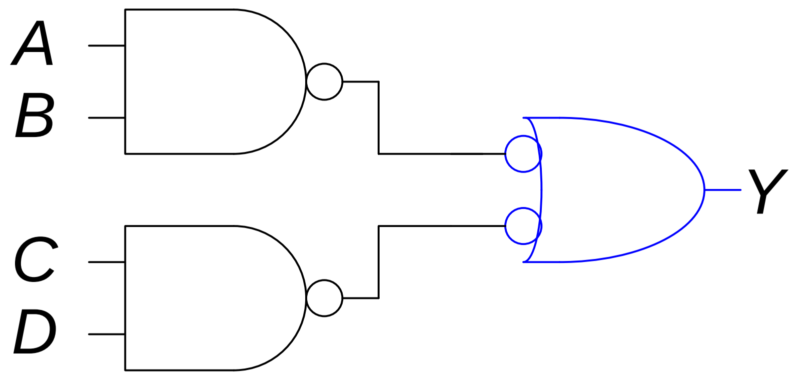


- **Forward:**
  - Body changes
  - Adds bubble to output

Bubble Pushing is a helpful way to redraw circuits so that the bubbles cancel out and the function can be more easily determined.

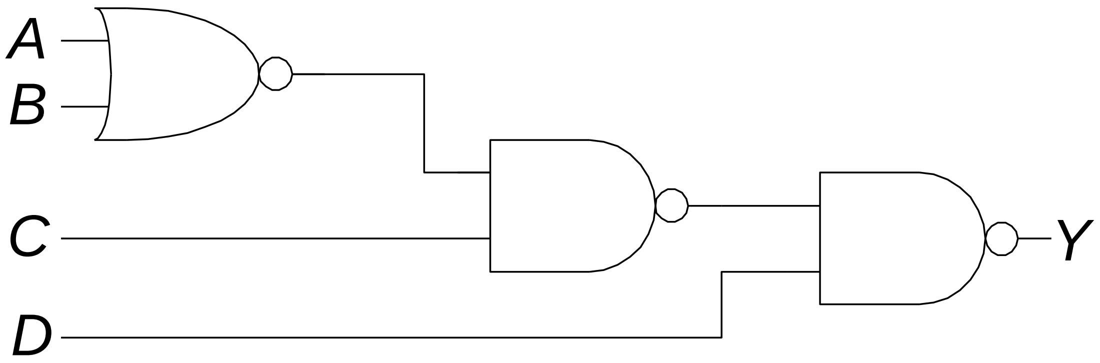
Bubble Pushing

- What is the Boolean expression for this circuit?

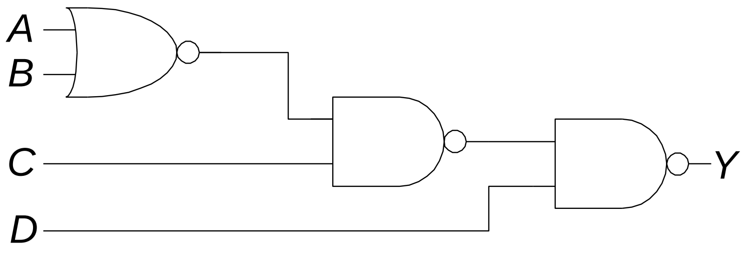


Bubble Pushing Rules

- Begin at output, then work toward inputs
- Push bubbles on final output back
- Draw gates in a form so bubbles cancel



Bubble Pushing Example



Chapter Review

- Basic Combinational Logic Circuits
  - AND-OR Logic
  - AND-OR-Invert Logic
  - Exclusive-OR Logic
  - Exclusive-NOR Logic
- The Universal Property of NAND Gates
  - Combinations of NAND gates can function as NOT, AND, OR, and NOR
  - Combinations of NOR gates can function as NOT, AND, OR, and NAND.
- Dual Symbols :

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

  - NAND Logic Diagrams Using Dual Symbols : NAND and Negative-OR;
  - NOR Logic Diagrams Using Dual Symbols : NOR and Negative-AND

True/False Quizs

- ✗ AND-OR logic can have only two 2-input AND gates.
- ✓ AOI is an acronym for AND-OR-Invert.
- ✓ If the inputs of an exclusive-OR gate are the same, the output is LOW (0).
- ✗ If the inputs of an exclusive-NOR gate are different, the output is HIGH (1).
- ✗ A parity generator cannot be implemented using exclusive-OR gates.
- ✓ NAND gates can be used to produce the AND functions.
- ✗ NOR gates cannot be used to produce the OR functions.
- ✓ Any SOP expression can be implemented using only NAND gates.
- ✗ The dual symbol for a NAND gate is a negative-AND symbol.
- ✓ Negative-OR is equivalent to NAND.