

ShanghaiTech University

**EE 115B: Digital Circuits**

Fall 2022

Lecture 5

Hengzhao Yang  
September 22, 2022

# EE115 Analog and *Digital Circuits*

## Topic 5: Boolean Algebra and Logic Simplification

Prof. Yajun Ha

ShanghaiTech University  
Shanghai, China



上海科技大学  
ShanghaiTech University

Part of this set of slides is by © Pearson Education

# Boolean Algebra and Logic Simplification

---

## Boolean Algebra

### ■ Logic Simplification

- Karnaugh Map
- Quine-McCluskey Method



# Boolean Addition

---

In Boolean algebra, a **variable** is a symbol used to represent an action, a condition, or data. A single variable can only have a value of 1 or 0.

The **complement** represents the inverse of a variable and is indicated with an overbar. Thus, the complement of  $A$  is  $\bar{A}$ .

A **literal** is a variable or its complement.

Addition is equivalent to the OR operation. The sum term is 1 if one or more of the literals are 1. The sum term is zero only if each literal is 0.

**Example** Determine the values of  $A$ ,  $B$ , and  $C$  that make the sum term of the expression  $\bar{A} + B + \bar{C} = 0$ ?

**Solution** Each literal must = 0; therefore  $A = 1$ ,  $B = 0$  and  $C = 1$ .



# Boolean Multiplication

---

In Boolean algebra, multiplication is equivalent to the AND operation. The product of literals forms a product term. The product term will be 1 only if all of the literals are 1.

**Example** What are the values of the  $A$ ,  $B$  and  $C$  if the product term of  $A \cdot \overline{B} \cdot \overline{C} = 1$ ?

**Solution** Each literal must = 1; therefore  $A = 1$ ,  $B = 0$  and  $C = 0$ .



# Commutative Laws

---

The **commutative laws** are applied to addition and multiplication. For addition, the commutative law states

**In terms of the result, the order in which variables are ORed makes no difference.**

$$A + B = B + A$$

For multiplication, the commutative law states

**In terms of the result, the order in which variables are ANDed makes no difference.**

$$AB = BA$$



# Associative Laws

---

The **associative laws** are also applied to addition and multiplication. For addition, the associative law states

**When ORing more than two variables, the result is the same regardless of the grouping of the variables.**

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

For multiplication, the associative law states

**When ANDing more than two variables, the result is the same regardless of the grouping of the variables.**

$$A(BC) = (AB)C$$

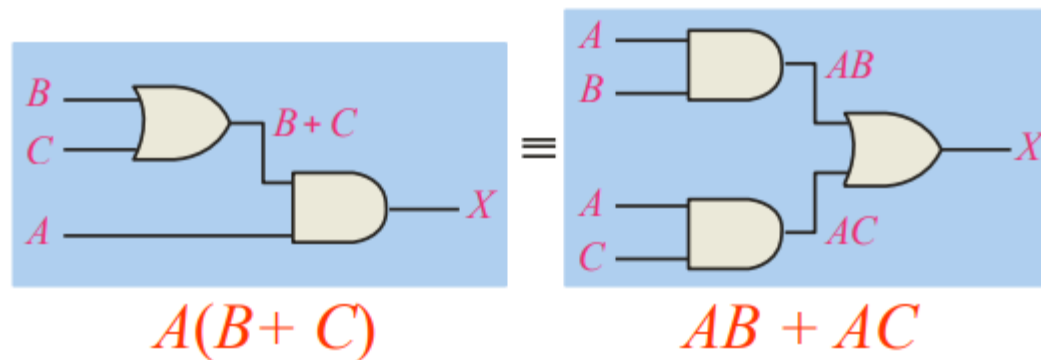


# Distributive Law

The **distributive law** is the factoring law. A common variable can be factored from an expression just as in ordinary algebra. That is

$$AB + AC = A(B + C)$$

The distributive law can be illustrated with equivalent circuits:





# Rules of Boolean Algebra (1)

---

$$1. A + 0 = A$$

$$2. A + 1 = 1$$

$$3. A \cdot 0 = 0$$

$$4. A \cdot 1 = 1 \quad A \& 1 = A$$

$$5. A + A = A$$

$$6. A + \bar{A} = 1$$

$$7. A \cdot A = A$$

$$8. A \cdot \bar{A} = 0$$

$$9. \bar{\bar{A}} = A$$

$$10. A + AB = A$$

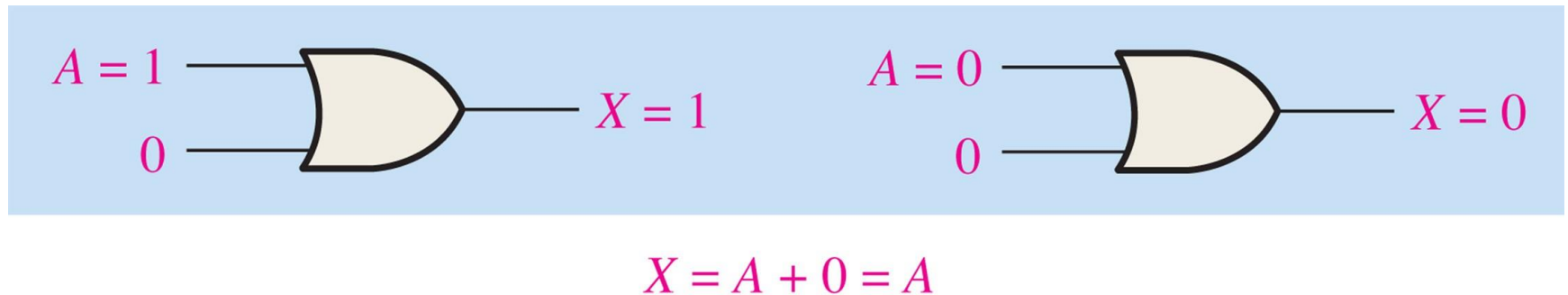
$$11. A + \bar{A}B = A + B$$

$$12. (A + B)(A + C) = A + BC$$



**FIGURE 4-8**

**Rule 1:  $A + 0 = A$**  A variable ORed with 0 is always equal to the variable. If the input variable  $A$  is 1, the output variable  $X$  is 1, which is equal to  $A$ . If  $A$  is 0, the output is 0, which is also equal to  $A$ . This rule is illustrated in Figure 4–8, where the lower input is fixed at 0.



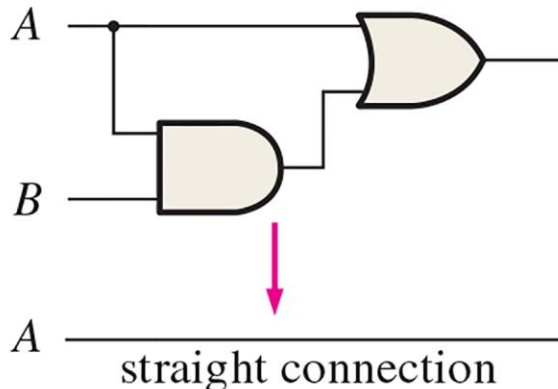
**Rule 10:  $A + AB = A$**  This rule can be proved by applying the distributive law, rule 2, and rule 4 as follows:

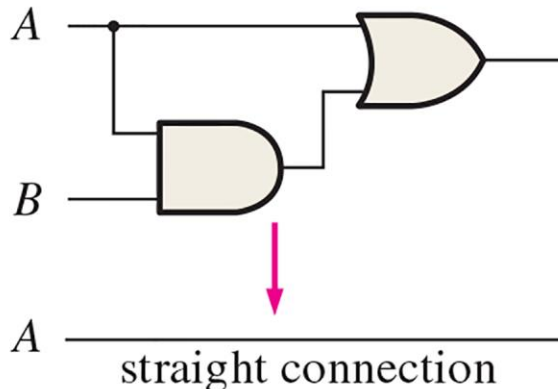
$$\begin{aligned}
 A + AB &= A \cdot 1 + AB = A(1 + B) && \text{Factoring (distributive law)} \\
 &= A \cdot 1 && \text{Rule 2: } (1 + B) = 1 \\
 &= A && \text{Rule 4: } A \cdot 1 = A
 \end{aligned}$$

**TABLE 4-2**

Rule 10:  $A + AB = A$ . Open file T04-02 to verify.

$A$	$B$	$AB$	$A + AB$
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1





equal

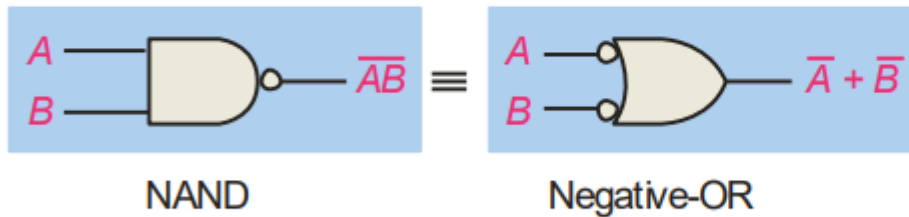
# DeMorgan's Theorem (1)

## DeMorgan's 1<sup>st</sup> Theorem

**The complement of a product of variables is equal to the sum of the complemented variables.**

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

Applying DeMorgan's first theorem to gates:



Inputs		Output	
A	B	$\overline{AB}$	$\overline{A} + \overline{B}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

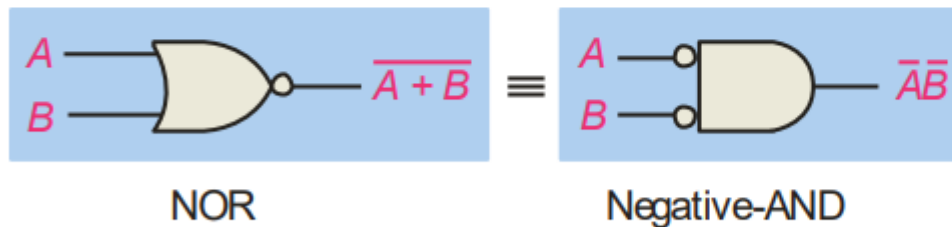
# DeMorgan's Theorem (2)

## DeMorgan's 2<sup>nd</sup> Theorem

**The complement of a sum of variables is equal to the product of the complemented variables.**

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

Applying DeMorgan's second theorem to gates:



Inputs		Output	
A	B	$\overline{A + B}$	$\overline{A} \overline{B}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0



# DeMorgan's Theorem (3)

---

## Example

Apply DeMorgan's theorem to remove the overbar covering both terms from the expression  $X = \overline{\overline{C} + D}$ .

## Solution

To apply DeMorgan's theorem to the expression, you can break the overbar covering both terms and change the sign between the terms. This results in  $X = \overline{\overline{C}} \cdot \overline{D}$ . Deleting the double bar gives  $X = C \cdot \overline{D}$ .



# Example

Apply DeMorgan's theorems.

$$\overline{AB + CD + EF}$$

Solution:

Let  $\overline{AB} = X$ ,  $\overline{CD} = Y$ , and  $\overline{EF} = Z$ . The expression  $\overline{AB + CD + EF}$  is of the form  $\overline{X + Y + Z} = \overline{XYZ}$  and can be rewritten as

$$\overline{AB + CD + EF} = (\overline{AB})(\overline{CD})(\overline{EF})$$

Next, apply DeMorgan's theorem to each of the terms  $\overline{AB}$ ,  $\overline{CD}$ , and  $\overline{EF}$ .

$$(\overline{AB})(\overline{CD})(\overline{EF}) = (\overline{A} + \overline{B})(\overline{C} + \overline{D})(\overline{E} + \overline{F})$$



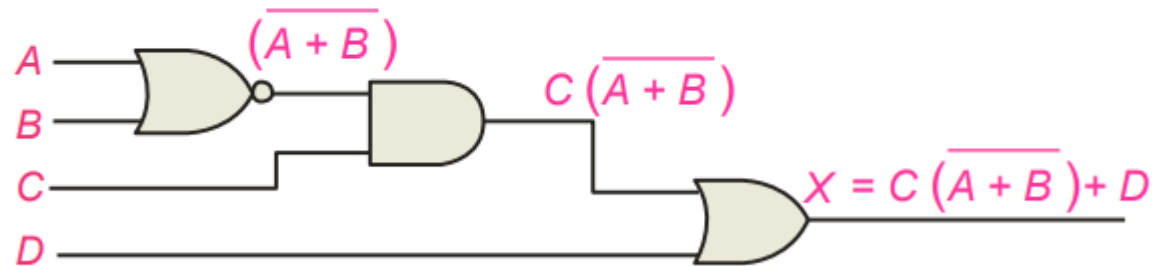
# Boolean Analysis of Logic Circuits

Combinational logic circuits can be analyzed by writing the expression for each gate and combining the expressions according to the rules for Boolean algebra.

## Example Solution

Apply Boolean algebra to derive the expression for  $X$ .

Write the expression for each gate:



Applying DeMorgan's theorem and the distribution law:

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

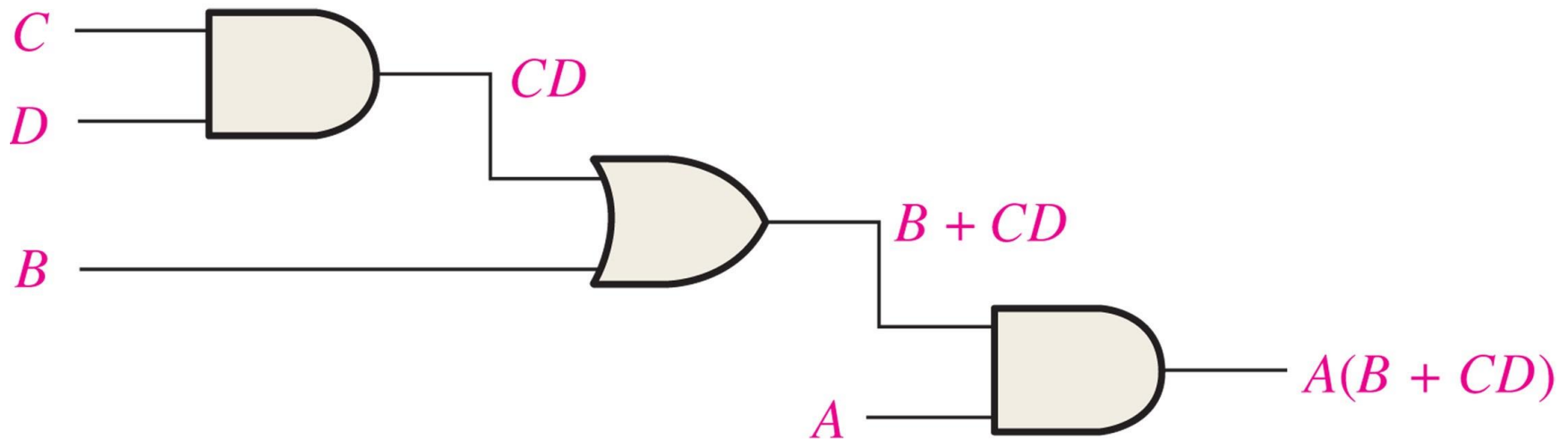




## Example: Boolean expression for a logic circuit

---

**FIGURE 4-18** A combinational logic circuit showing the development of the Boolean expression for the output.



# Example: Constructing a truth table for a logic circuit

**TABLE 4-5**

Truth table for the logic circuit in Figure 4-18.

Inputs				Output
<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	$A(B + CD)$
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

# SOP and POS Forms

---

Boolean expressions can be written in the **sum-of-products** form (**SOP**) or in the **product-of-sums** form (**POS**). These forms can simplify the implementation of combinational logic, particularly with PLDs. In both forms, an overbar cannot extend over more than one variable.

An expression is in SOP form when two or more product terms are summed as in the following examples:

$$\overline{A} \overline{B} \overline{C} + A B$$

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

$$C D + \overline{E}$$

An expression is in POS form when two or more sum terms are multiplied as in the following examples:

$$(A + B)(\overline{A} + C)$$

$$(A + B + \overline{C})(B + D)$$

$$(\overline{A} + B)C$$



# SOP Standard Form

In **SOP standard form**, every variable in the domain must appear in each term. This form is useful for constructing truth tables or for implementing logic in PLDs.

You can expand a nonstandard term to standard form by multiplying the term by a term consisting of the sum of the missing variable and its complement.

## Example Solution

Convert  $X = \bar{A} \bar{B} + A B C$  to standard form.

The first term does not include the variable  $C$ . Therefore, multiply it by the  $(C + \bar{C})$ , which = 1:

$$\begin{aligned} X &= \bar{A} \bar{B} (C + \bar{C}) + A B C \\ &= \bar{A} \bar{B} C + \bar{A} \bar{B} \bar{C} + A B C \end{aligned}$$



## 2.5.3 逻辑函数的两种标准形式

**最小项之和**

**最大项之积**

**最小项  $m$ :**

- $m$ 是乘积项
- 包含 $n$ 个因子
- $n$ 个变量均以原变量和反变量的形式在 $m$ 中出现一次

对于 $n$ 变量函数  
有 $2^n$ 个最小项

## 最小项举例：

- 两变量 $A, B$ 的最小项

$$A'B', A'B, AB', AB \quad (2^2 = 4 \text{个})$$

- 三变量 $A, B, C$ 的最小项

$$A'B'C', A'B'C, A'BC', A'BC$$

$$AB'C', AB'C, ABC', ABC \quad (2^3 = 8 \text{个})$$

# 最小项的编号：

最小项	取值	对应	编号
	$A \ B \ C$	十进制数	
$A'B'C'$	0 0 0	0	$m_0$
$A'B'C$	0 0 1	1	$m_1$
$A'BC'$	0 1 0	2	$m_2$
$A'BC$	0 1 1	3	$m_3$
$AB'C'$	1 0 0	4	$m_4$
$AB'C$	1 0 1	5	$m_5$
$ABC'$	1 1 0	6	$m_6$
$ABC$	1 1 1	7	$m_7$