

# 逻辑电路

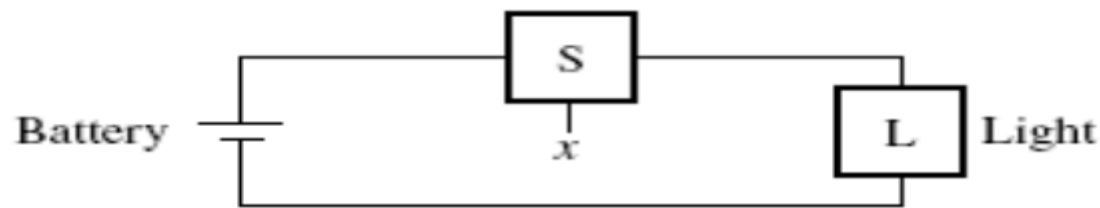
任何一个电路中，信号只能取有限个分离值，这种电路称为逻辑电路。

在二进制逻辑电路中只用两个值，0 和 1。

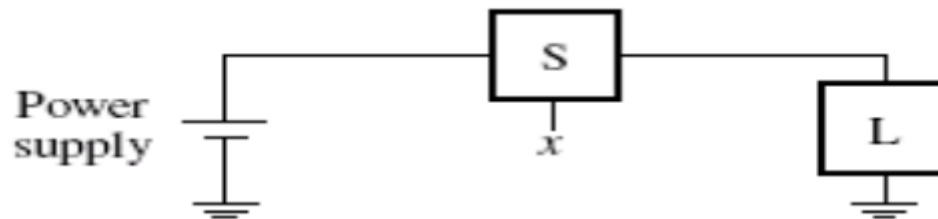
在十进制的逻辑电路中用十个值，从 0 到 9。因为人们很自然地用数字来表示每个信号值，所以这类逻辑电路也被称作数字电路。

## 小灯泡的开启和关闭

- 某给定开关是由输入变量  $x$  控制的，则我们说当  $x = 0$  时开关  $S$  断开，当  $x = 1$  时开关  $S$  闭合。引起电路行为变化的输入变量是控制开关（断/合）的变量  $x$ 。



(a) Simple connection to a battery



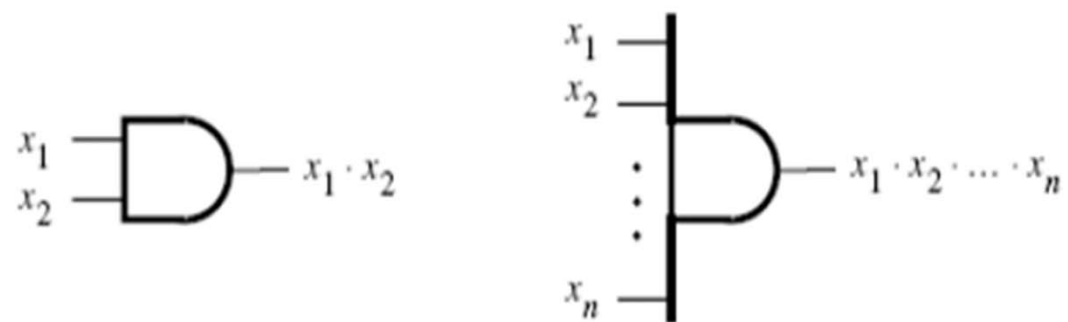
- 输出被定义为灯泡 L 的状态。若灯开时, 则  $L = 1$ 。若灯关时, 则  $L = 0$ 。根据这个约定, 我们可以将灯泡 L 的状态描述成输入变量 x 的函数。当  $x = 1$  时,  $L = 1$ , 当  $x = 0$  时,  $L = 0$ , 则我们说:

$$L(x) = x$$

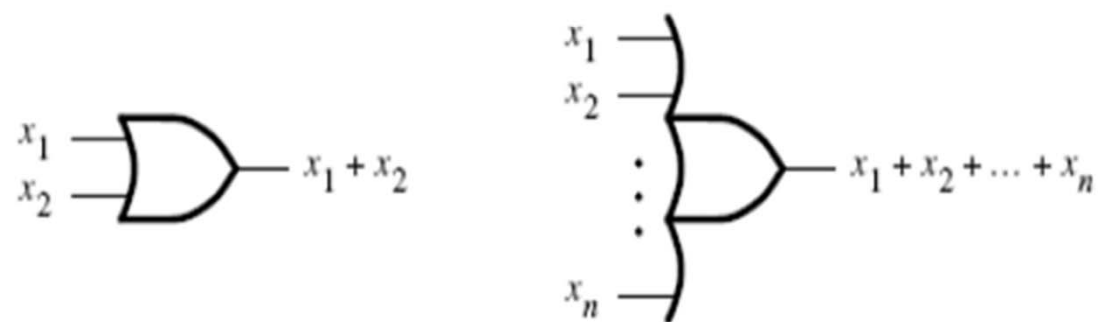
- 简单的逻辑表达式将输出描述成输入的函数。
- 我们说  $L(x) = x$  是个逻辑函数, 而 x 是输入变量。

# 逻辑门和网络

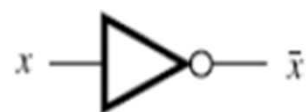
- 基本逻辑操作有三种      与      或      非
- 三种基本逻辑操作可以实现任意复杂的逻辑函数。
- 复杂逻辑函数可能需要许多这样的基本操才能实现。
- 每个逻辑操作都能用晶体管来实现，实现逻辑操作的电路元件叫做逻辑门。
- 逻辑门有一个或若干个输入，有一个输出，输出表示为输入的函数。用画电路图的方法来描述逻辑电路通常是很方便的，电路图由表示逻辑门的图形符号组成。



(a) AND gates



(b) OR gates



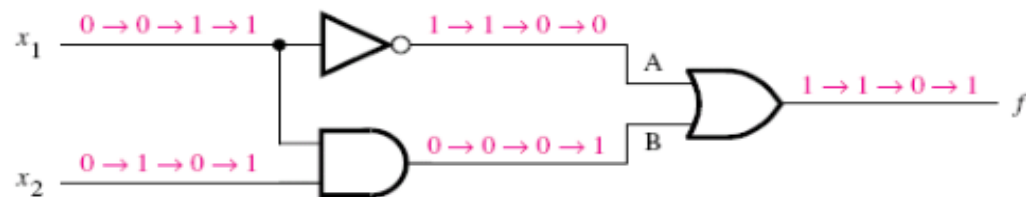
# 逻辑门和网络

- 大规模电路由逻辑门组成的网络实现。给定网络的复杂程度直接影响电路的制造成本。因为我们总是想要降低制造产品的成本，所以找到成本尽可能低的逻辑电路实现方案就非常重要。

# 逻辑网络的分析

- 数字系统的设计者会遇到两个基本问题：
- 第一个问题：对已存在的逻辑网络，一定能够确定所实现的函数，这种任务称为分析过程。
- 第二个问题：设计一个实现所需函数功能的新网络，这个过程称为综合过程。
- 分析过程直观易懂，比综合过程简单得多。

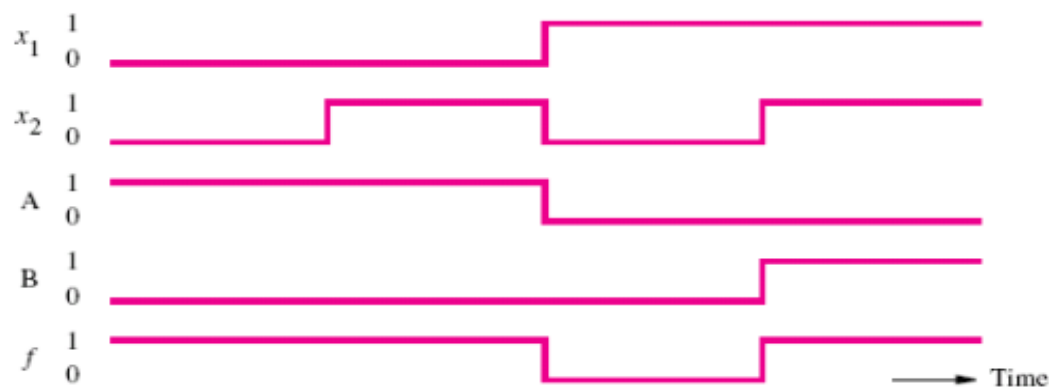
## 功能等价的网络



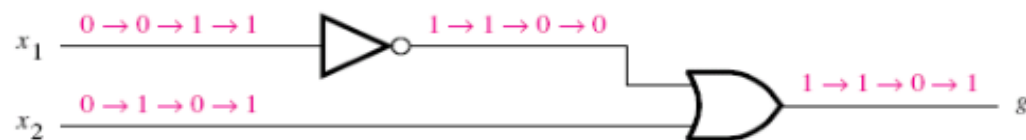
(a) Network that implements  $f = \bar{x}_1 + x_1 \cdot x_2$

$x_1$	$x_2$	$f(x_1, x_2)$
0	0	1
0	1	1
1	0	0
1	1	1

(b) Truth table for  $f$



(c) Timing diagram



(d) Network that implements  $g = \bar{x}_1 + x_2$



# 功能等价的网络

- 用同样的分析步骤，可以发现输出  $g$  和 输出  $f$  完全相同。因此，

$$g(x_1, x_2) = f(x_1, x_2)$$

这也就是说这两个电路等价；既然这两个网络实现相同的功能，那么选用简单的那一个就很有意义了，因为简单的网络实现成本低。

- 一般情况下，一个逻辑函数可以用许多种不同的网络来实现，实现电路的成本也可能有所不同。这就产生了一个重要的问题，怎样为给定的逻辑函数寻找最优的实现电路呢？

逻辑函数的数学处理方法---- 布尔代数

# 逻辑代数基础

1

逻辑代数的基本概念与运算

2

逻辑函数的公式化简法

3

逻辑函数的卡诺图化简法

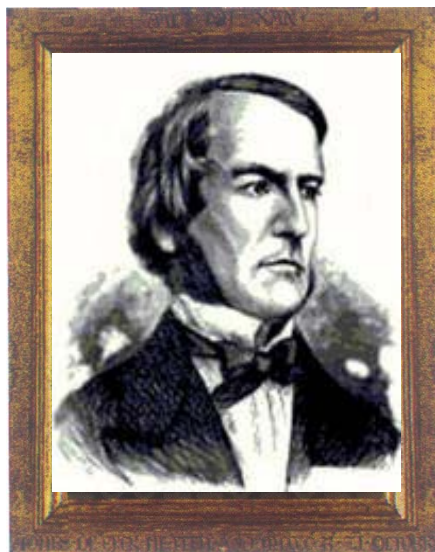
4

具有无关项的逻辑函数及其化简

## 1、

## 逻辑代数的基本概念和运算规则

逻辑代数是英国数学家George Boole于1847年提出的，所以又称为布尔代数或开关代数，它是分析和设计逻辑电路的重要数学工具。



- ◆ 英国数学家 *George Boole* 于1815年11月生于英格兰的林肯。
- ◆ 1847年，发表了著作 **《The Mathematical Analysis of Logic》**。
- ◆ 1849年，他被任命位于爱尔兰科克的皇后学院的数学教授。
- ◆ 1854年，他出版了 **《The Laws of Thought》**。
- ◆ 布尔撰写了**微分方程**和**差分方程**的课本。
- ◆ 1864年，布尔死于肺炎。

## 1. 逻辑变量与逻辑函数

在逻辑代数中的变量称为逻辑变量，通常用字母A、B、C等表示。逻辑变量的取值只有两种：真（“1”）和假（“0”）。这里的“1”和“0”并不表示数量的大小，而是表示完全对立的两种状态。

若以逻辑变量作为输入，以运算结果作为输出，那么当输入变量的取值确定之后，输出的取值便随之而定。因此，输出与输入之间乃是一种函数关系。这种函数关系称为逻辑函数，写作  $Y=F(A, B, C\dots)$ 。

例：一个举重裁判电路

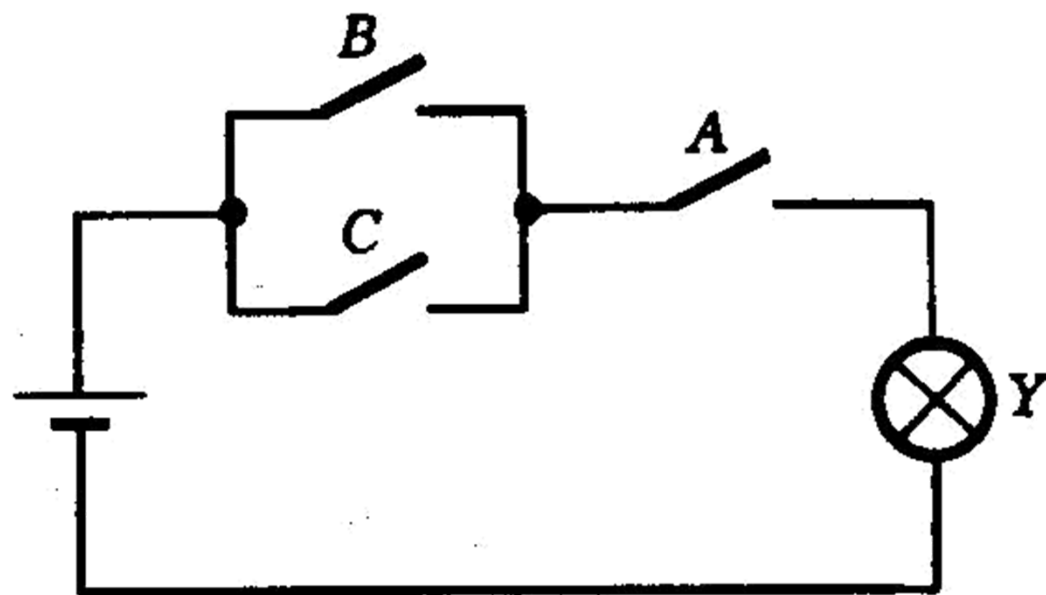


图 举重裁判电路

## 2. 逻辑代数中的三种基本运算

逻辑代数的基本运算有与(AND)、或(OR)、非(NOT)三种。它们各自的含义如图中 (a)、(b)、(c) 所示。

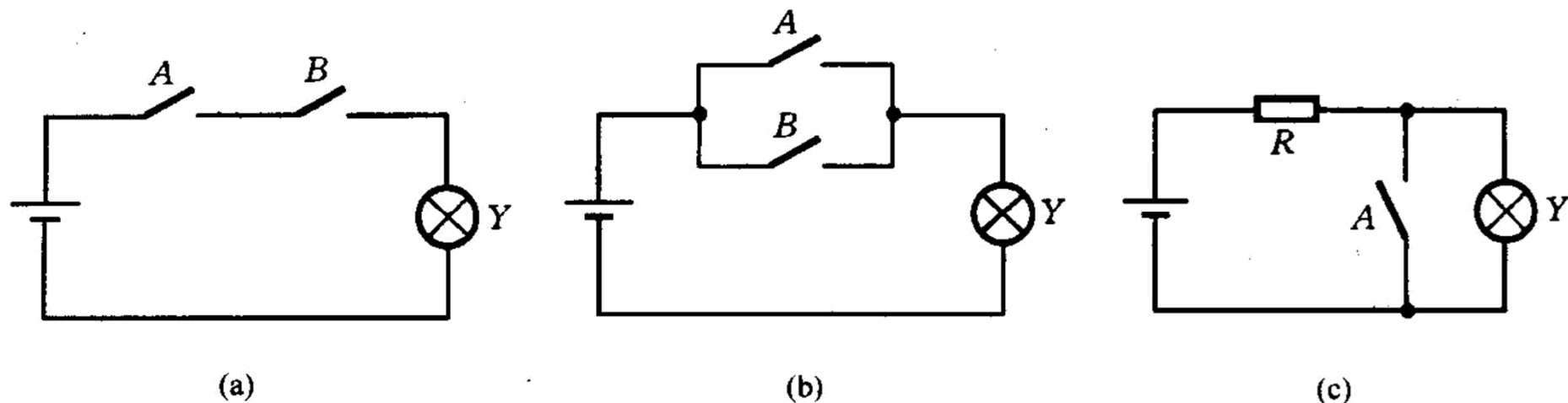
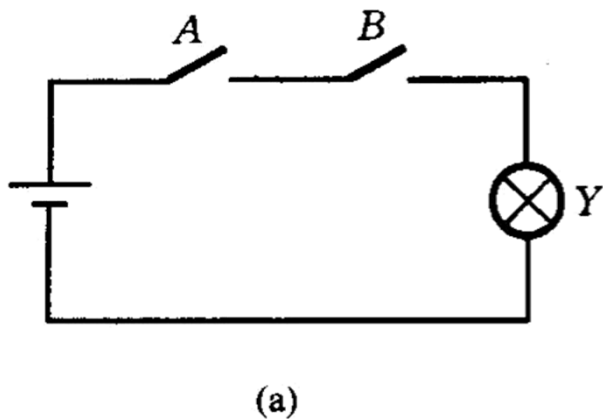


图 与、或、非说明电路

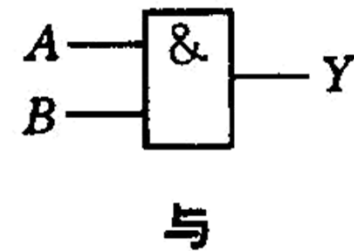
若把开关闭合作为条件，把灯亮作为结果，那么图中的三个电路代表了三种不同的因果关系：

(a) 逻辑与，也叫逻辑相乘：表示只有决定事物结果的全部条件同时具备时，结果才会发生。记作： $Y=A \text{ AND } B$ 或 $Y=A \cdot B$ 或 $Y=AB$ 。其逻辑真值表如表。



与逻辑真值表

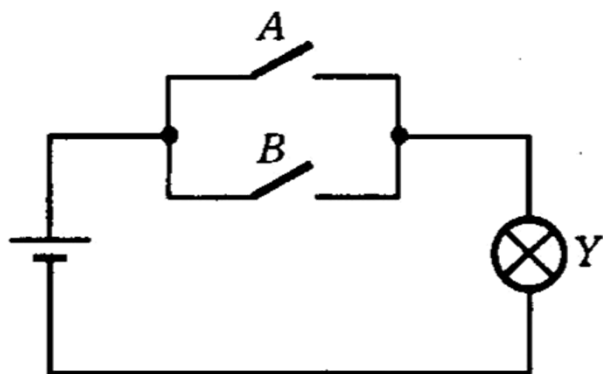
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



“与门”的图形符号



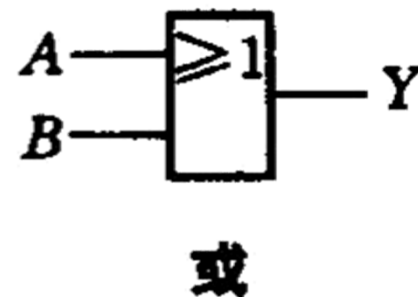
(b) 逻辑或，也叫逻辑相加：表示决定事物结果的条件中只要有任何一个满足，结果就会发生。记作： $Y=A \text{ OR } B$  或  $Y=A+B$ 。其逻辑真值表如表。



(b)

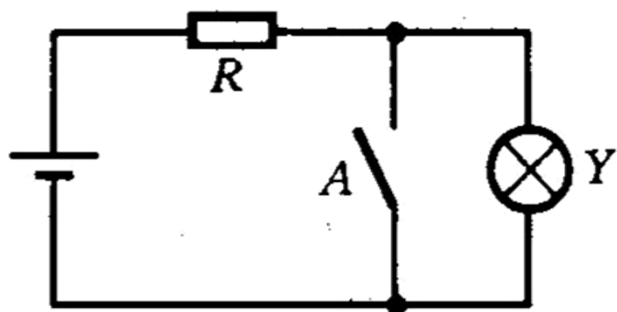
或逻辑真值表

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1



“或门”的图形符号

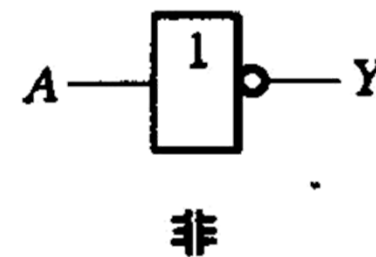
(c) 逻辑非，也叫逻辑求反：表示只要条件具备了，结果就不会发生，否则结果一定发生。记做：或  $Y = \overline{A}$   
 $\text{NOT } A$ 。其逻辑真值表如表。



(c)

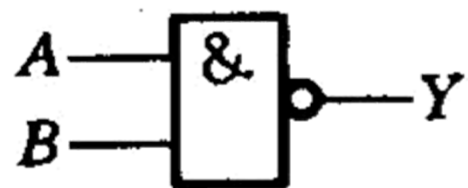
非逻辑真值表

A	Y
0	1
1	0



“非门”（或反相器）的图形符号

## 最常见的复合逻辑运算——“与非”（NAND）



与非



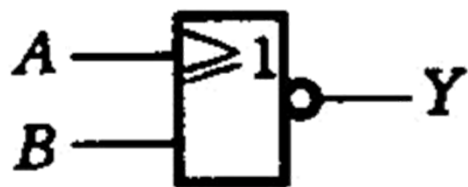
$$Y = \overline{A \cdot B}$$

NAND Truth Table

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

图 “与非” 复合逻辑的图形符号和运算符号

最常见的复合逻辑运算——“或非”（NOR）



或非



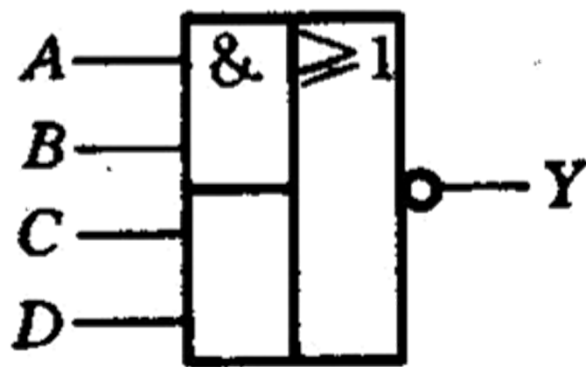
$$Y = \overline{A + B}$$

NOR Truth Table

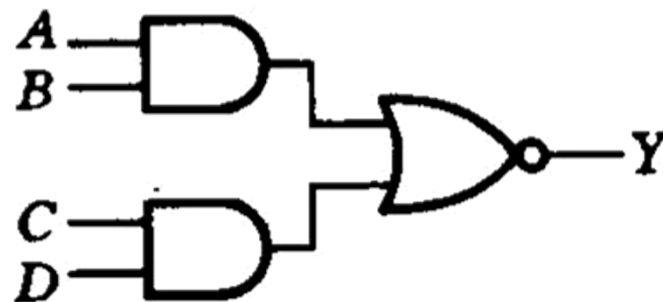
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

图 “或非” 复合逻辑的图形符号和运算符号

最常见的复合逻辑运算——“与或非”（AND-NOR）



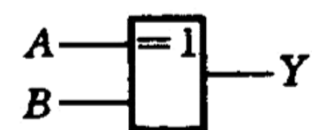
与或非



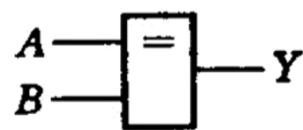
$$Y = \overline{A \cdot B + C \cdot D}$$

图 “与或非” 复合逻辑的图形符号和运算符号

最常见的复合逻辑运算——“异或”（EXCLUSIVE-OR）  
和“同或”（EXCLUSIVE-NOR）



异或



同或



$$Y = A \oplus B$$



$$Y = A \odot B$$

EX-OR		
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

EX-NOR		
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

$$\overline{A \oplus B} = A \odot B$$

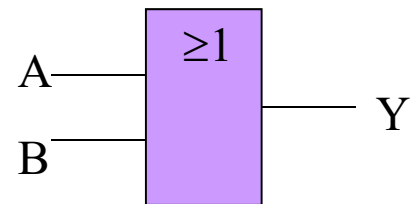
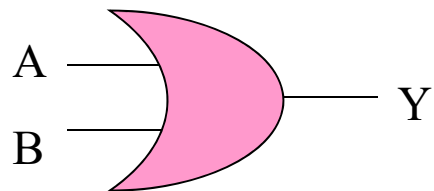
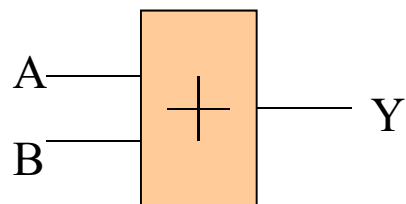
图 “异或” “同或” 复合逻辑的图形符号和运算符号

(a) 常用符号

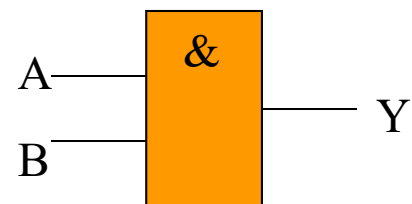
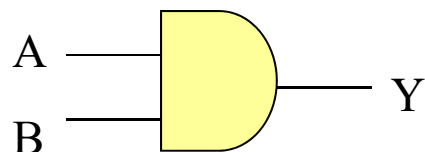
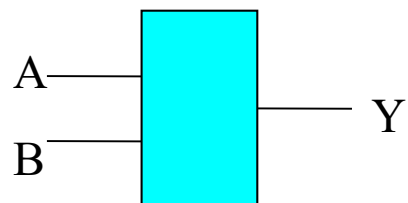
(b) 美、日常用符号

(c) 国标符号

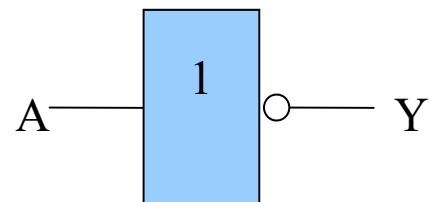
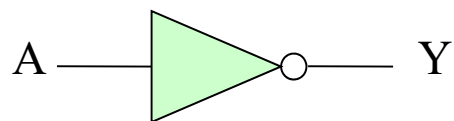
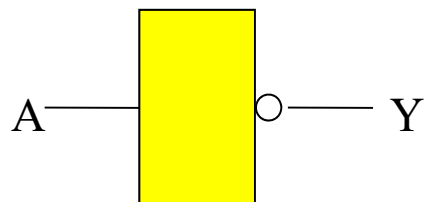
或门



与门



非门

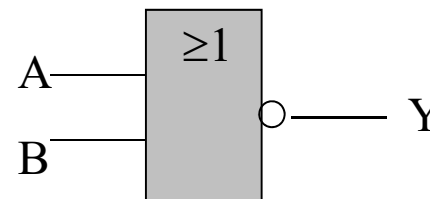
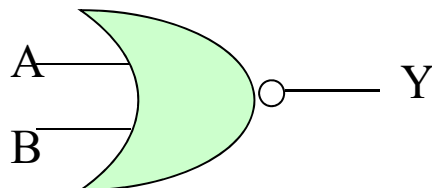
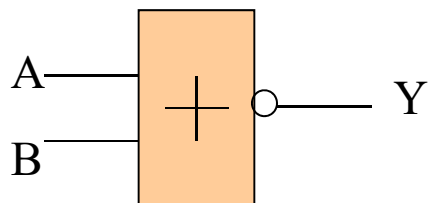


(a) 常用符号

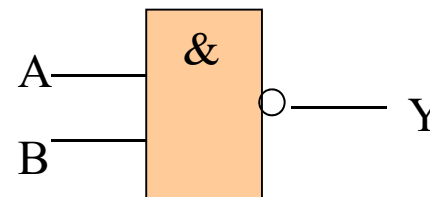
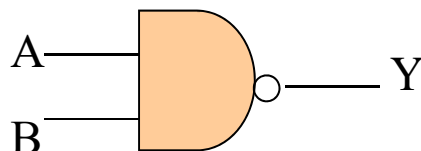
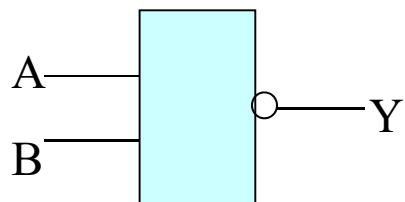
(b) 美、日常用符号

(c) 国标符号

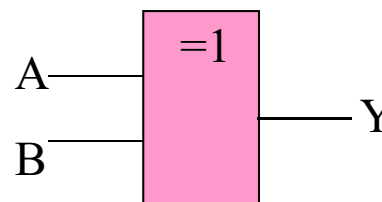
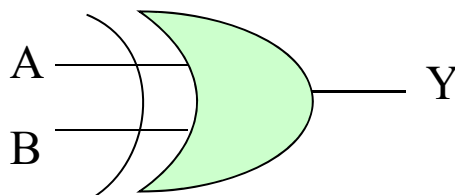
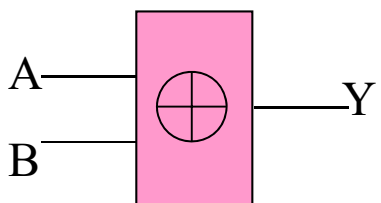
或非门



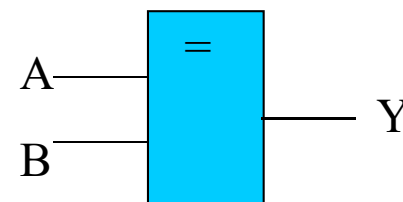
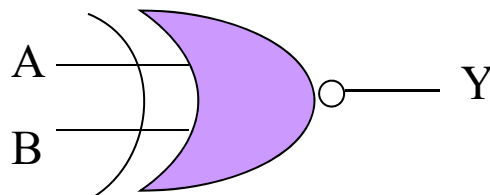
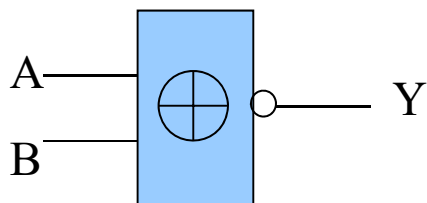
与非门



异或门



异或非门





### 3 逻辑函数的描述（即表示方法）

描述逻辑函数的方法有以下六种：

#### 一、逻辑表达式（logic function）

用与、或、非等逻辑运算表示逻辑关系的代数式叫逻辑函数表达式或简称函数式。

例：  $Y = AB + \overline{A}CD$

#### 二、真值表（truth table）

将输入变量所有的取值对应的输出值找出来，列成表格，即可得真值表。列真值表时，需注意以下几点：

（1）所有的输入的组合不可遗漏，也不可重复；输入组合最好按二进制数递增的顺序排列（完整性）。

（2）同一逻辑函数的真值表具有唯一性。

例：请列出举重裁判电路  $Y = A \cdot (B + C)$  的真值表。

**例：**已知某逻辑函数的真值表如下所示，试写出其逻辑函数式。

A	B	C	Y
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

从真值表写出**逻辑函数**的一般方法：

- 1、找出真值表中使逻辑函数 **$Y=1$** 的那些输入变量取值的组合；
- 2、每组输入变量取值的组合对应一个乘积项，其中取值为**1**的写入**原变量**，取值为**0**的写入**反变量**；
- 3、将这些乘积项相**或（加）**，即可得逻辑函数式。

**（3）**真值表还可作为**判断两函数是否相等**的依据。

### 三、逻辑电路图(logic diagram)

用代表逻辑运算的逻辑门符号所构成的逻辑关系图形，叫逻辑电路图，简称逻辑图。

**例：**请画出  $Y = A \cdot B + \bar{A} \cdot C$  的逻辑电路图。

### 四、卡诺图(Karnaugh Map)

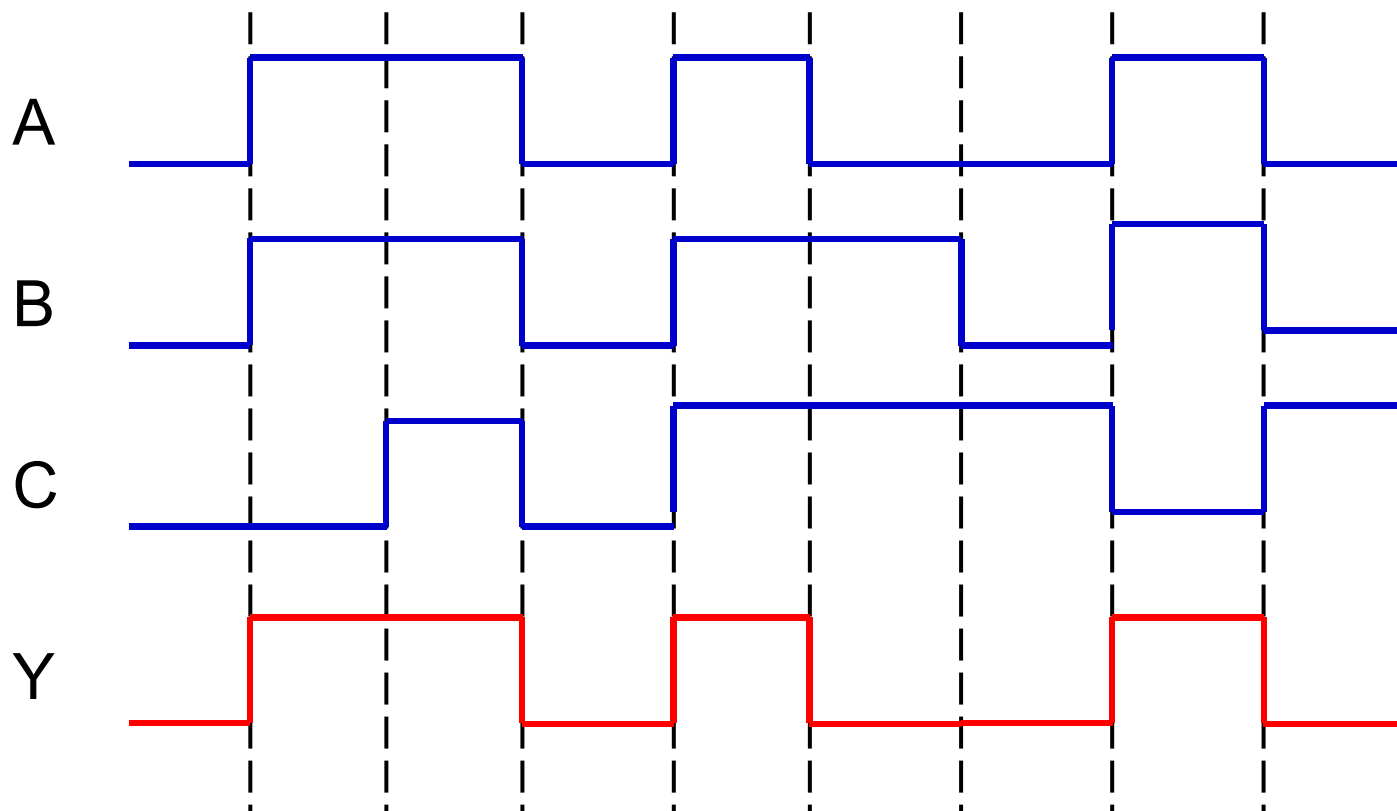
后述。

### 五、波形图（时序图）(waveform or timing diagram)

指各个逻辑变量的逻辑值随时间变化的规律图。

**例：**试画出举重裁判电路的波形图。

例：试画出举重裁判电路的波形图。



## 六、文字描述

逻辑函数的各种表示方法可以相互转换。

## 4 逻辑代数的基本公式和常用公式

### 一、逻辑代数的公理（基本假设）

$$(1) \overline{1} = 0; \overline{0} = 1$$

$$(2) 1 \cdot 1 = 1; 0 + 0 = 0$$

$$(3) 1 \cdot 0 = 0 \cdot 1 = 0; 0 + 1 = 1 + 0 = 1$$

$$(4) 0 \cdot 0 = 0; 1 + 1 = 1$$

$$(5) \text{若 } A \neq 0 \text{ 则 } A = 1; \text{ 若 } A \neq 1 \text{ 则 } A = 0$$

## 二、逻辑代数的基本公式（性质）

(1) 交换律:  $A \cdot B = B \cdot A$ ;  $A + B = B + A$

(2) 结合律:  $A \cdot (B \cdot C) = (A \cdot B) \cdot C$ ;

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

(3) 分配律:  $A \cdot (B + C) = A \cdot B + A \cdot C$ ;

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

(4) 01定律:  $1 \cdot A = A$ ;  $0 + A = A$

$$0 \cdot A = 0; \quad 1 + A = 1$$

(5) 互补律:  $A \cdot \bar{A} = 0$ ;  $A + \bar{A} = 1$

(6) 重叠律:  $A \cdot A = A$ ;  $A + A = A$

(7) 反演律 (De. Morgan定理):

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

(8) 还原律:  $\overline{\overline{A}} = A$

注: 1、若两个逻辑函数具有完全相同的真值表, 则这两个逻辑函数相等。证明以上定律的基本方法均采用真值表法。

2、逻辑代数与普通代数是不同的。

例:  $\overline{A}B + A\overline{B} + AB = A + B + AB,$

但  $\overline{A}B + A\overline{B} \neq A + B$

### 三、逻辑代数的常用公式（吸收律）

$$(1) A + AB = A; \quad A(A + B) = A$$

$$(2) AB + A\bar{B} = A; \quad (A + B)(A + \bar{B}) = A$$

$$(3) A + \bar{A}B = A + B; \quad A(\bar{A} + B) = AB$$

$$(4) AB + \bar{A}C + BC = AB + \bar{A}C$$

$$\text{推论: } AB + \bar{A}C + BCDE \dots = AB + \bar{A}C$$

$$(5) A \cdot \overline{A \cdot B} = A \cdot \bar{B}; \quad \bar{A} \cdot \overline{\bar{A} \cdot B} = \bar{A}$$



## 四、异或运算

1、异或定义：

$A \oplus B = \overline{A}B + A\overline{B} \Rightarrow A、B$ 相异为1，相同为0；

$A \odot B = \overline{\overline{A}B + A\overline{B}} \Rightarrow A、B$ 相同为1，相异为0；

可见： $A \odot B = \overline{A \oplus B}$

**思考：**  $A \odot B \odot C$  ?  $A \oplus B \oplus C$

2、异或运算的性质：

(1) 交换律： $A \oplus B = B \oplus A$

(2) 结合律： $(A \oplus B) \oplus C = A \oplus (B \oplus C)$

(3) 分配律： $A (B \oplus C) = AB \oplus AC$

(4) 常量与变量:  $A \oplus 1 = \overline{A}$ ;  $A \oplus 0 = A$ ;

$$A \oplus A = 0; A \oplus \overline{A} = 1。$$

(5) 因果互换关系: 若  $A \oplus B = C$ , 则有

$$A \oplus C = B, B \oplus C = A。若 A \oplus B \oplus C \oplus D = 0, 则有 0 \oplus A \oplus B \oplus C = D, A \oplus B \oplus D \oplus 0 = C等。$$

(6) 多变量异或运算:

在多变量异或运算中, 若变量为1的个数为奇数, 异或运算结果为1, 若变量为1的个数为偶数, 异或运算结果为0, 与变量为0的个数无关。即:

$$\underbrace{A \oplus A \oplus A \oplus \cdots \oplus A}_{\text{偶数个}A} = 0$$

$$\underbrace{A \oplus A \oplus A \oplus \cdots \oplus A}_{\text{奇数个}A} = A$$

## 5 逻辑代数的三个基本定律

### (一) 代入规则

将逻辑等式两边的某一变量均用同一个逻辑函数替代，等式仍然成立。

★  $A + \bar{A}B = A + B$

$A$ 均用 $\bar{A}$ 代替  $\Rightarrow \bar{A} + \bar{A}B = \bar{A} + B$

$A$ 均用 $C \oplus D$ 代替  $\Rightarrow C \oplus D + \overline{C \oplus D}B = C \oplus D + B$

$B$ 均用 $C$ 代替  $\Rightarrow A + \bar{A}C = A + C$

利用代入规则能扩展基本定律的应用。

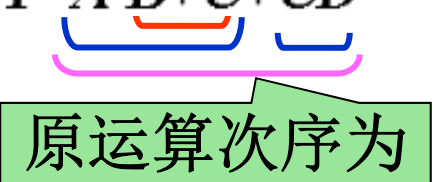
## (二) 反演规则

对任一个逻辑函数式  $Y$ ，将“ $\cdot$ ”换成“ $+$ ”，“ $+$ ”换成“ $\cdot$ ”，“ $0$ ”换成“ $1$ ”，“ $1$ ”换成“ $0$ ”，原变量换成反变量，反变量换成原变量，则得到原逻辑函数的反函数  $\bar{Y}$ 。

变换时注意：

- (1) 不能改变原来的运算顺序。非、与、或
- (2) 反变量换成原变量只对单个变量有效，而长非号保持不变。

$$\star \quad Y = A \cdot \overline{B + C} + CD \qquad \bar{Y} = (\bar{A} + \overline{\bar{B} \cdot \bar{C}}) \cdot (\bar{C} + \bar{D})$$



例：已知  $Y = \overline{\overline{AB} + C + D + C}$ ，求  $\bar{Y}$ 。

可见，求逻辑函数的反函数有两种方法：利用反演规则或摩根定律。

### (三) 对偶规则

对任一个逻辑函数式  $Y$ ，将 “ $\cdot$ ” 换成 “ $+$ ”， “ $+$ ” 换成 “ $\cdot$ ”， “ $0$ ” 换成 “ $1$ ”， “ $1$ ” 换成 “ $0$ ”，则得到原逻辑函数式的对偶式  $Y'$ 。

对偶规则：两个函数式相等，则它们的对偶式也相等。

变换时注意：(1) 变量不改变  
(2) 不能改变原来的运算顺序

$$\star \quad A + AB = A \Rightarrow A \cdot (A + B) = A$$

$$\star \quad AB + \bar{A}C + BC = AB + \bar{A}C$$



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

应用对偶规则可将基本公式和定律扩展。

证明两个逻辑式相等，有时可通过证明它们的对偶式相等来完成，因为有些情况下证明其对偶式相等更加容易。

例：

$$\text{已知 } AB + A\overline{B}C = AB + AC,$$

$$\text{试求 } (A + B) (A + \overline{B} + C) = ?$$

**对偶性**意味着每个逻辑函数至少可以用**两种不同**的布尔代数表达式表示。其中一种表达式比另一种表达式的物理实现更简单，因而更被认可。

# 3个规则

1. **代入规则**：对任何一个含有原变量 $A$ 的逻辑等式，如果在所有出现原变量 $A$ 的位置处都代入一个逻辑函数 $F$ ，则原逻辑等式仍然成立。
2. **反演规则**：在一个逻辑函数中，若将函数中所有原来的“与”运算变成“或”运算，所有原来的“或”运算变成“与”运算，原变量变成反变量，反变量变成原变量，同时函数中的所有“1”变成“0”，且“0”变成“1”，这样所得到的一个新函数就称为原函数的反函数。
3. **对偶规则**：在一个逻辑函数中，若将函数中所有原来的“与”运算变成“或”运算，所有原来的“或”运算变成“与”运算，函数中的所有“1”变成“0”，且“0”变成“1”，但逻辑变量保持不变，这样所得到的一个新函数就称为原函数的对偶函数，用 $F$ 表示。

## 逻辑代数基本定律总结表

定律名称	公 式	
0—1律	$A \cdot 0 = 0$	$A + 1 = 1$
自等律	$A \cdot 1 = A$	$A + 0 = A$
重叠律	$A \cdot A = A$	$A + A = A$
互补律	$A \cdot \bar{A} = 0$	$A + \bar{A} = 1$
交换律	$A \cdot B = B \cdot A$	$A + B = B + A$
结合律	$A \cdot (B \cdot C) = (A \cdot B) \cdot C$	$A + (B + C) = (A + B) + C$
分配律	$A \cdot (B + C) = AB + AC$	$A + BC = (A + B)(A + C)$
还原律	$\overline{\overline{A}} = A$	$(A^*)^* = A$
反演律	$\overline{A \cdot B} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A} \cdot \bar{B}$
吸收律（一）	$AB + A\bar{B} = A$	$(A + B)(A + \bar{B}) = A$
吸收率（二）	$A + AB = A$	$A \cdot (A + B) = A$
吸收率（三）	$A + \bar{A}B = A + B$	$A \cdot (\bar{A} + B) = AB$
吸收率（四）	$AB + \bar{A}C + BC = AB + \bar{A}C$	$(A + B)(\bar{A} + C)(B + C) = (A + B)(\bar{A} + C)$



## 逻辑函数式化简的意义与标准

化  
简  
意  
义

使逻辑式最简，以便设计出最简的逻辑电路，从而节省元器件、优化生产工艺、降低成本和提高系统可靠性。

不同形式逻辑式有不同的最简式，一般先求取最简与-或式，然后通过变换得到所需最简式。

### 最简与-或式标准

- (1) 乘积项(即与项)的个数最少
- (2) 每个乘积项中的变量数最少

用与门个数最少  
与门的输入端数最少

### 最简与非式标准

- (1) 非号个数最少
- (2) 每个非号中的变量数最少

用与非门个数最少  
与非门的输入端数最少

评价逻辑电路成本高低的一个重要指标是统计电路中逻辑门的总数加上各个门的输入端的总数。

## 逻辑函数式的几种常见形式和变换

例如  $Y = \overline{A}\overline{B} + \overline{B}\overline{C}$

与或表达式

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

与非 - 与非表达式

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

或与表达式

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

或非 - 或非表达式

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

与或非表达式

转换方法举例

与或式  $\longrightarrow$  与非式

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

$$= \overline{\overline{\overline{A}\overline{B}} \cdot \overline{\overline{B}\overline{C}}} \quad \text{用还原律}$$

$$= \overline{\overline{A}\overline{B} \cdot \overline{B}\overline{C}} \quad \text{用摩根定律}$$

或与式  $\longrightarrow$  或非式  $\longrightarrow$  与或非式

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

$$= \overline{\overline{(A + B)} + \overline{(\overline{B} + \overline{C})}} \quad \text{用还原律}$$

$$= \overline{\overline{A + B} + \overline{\overline{B}} + \overline{\overline{C}}} \quad \text{用摩根定律}$$

$$= \overline{\overline{A}\overline{B} + \overline{B}\overline{C}} \quad \text{用摩根定律}$$

## 二、常用的公式化简方法

运用逻辑代数的基本定律和常用公式对逻辑式进行化简，消去函数中多余的乘积项和乘积项中的多余变量，使逻辑函数成为最简与-或式。

- **并项法** 运用  $AB + A\bar{B} = A$   
将两项合并为一项，并消去一个互补的变量

$$\star Y = \overline{A}BC + A\overline{B}C + AB = A$$

$$\begin{aligned}\star Y &= A(\underline{BC} + \overline{BC}) + A(\underline{B\overline{C}} + \overline{B\overline{C}}) \\ &= \overline{A}B \oplus C + A(B \oplus C) = A\end{aligned}$$



## 吸收法

运用  $A+AB=A$  和  $AB + \overline{A}C + BC = AB + \overline{A}C$ ,  
消去多余的与项。

$$\star Y = \underline{AB} + \underline{AB}(E + F) = AB$$

$$\star Y = ABC + \underline{\overline{A}D} + \underline{\overline{C}D} + BD$$

$$= ABC + D(\overline{A} + \overline{C}) + BD$$

$$= \underline{ACB} + \underline{\overline{A}C} \cdot \underline{D} + \underline{BD}$$

$$= ACB + \underline{\overline{A}CD}$$

$$= ABC + \overline{A}D + \overline{C}D$$

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

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

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

$$A + A\bar{B} = A(1 + \bar{B}) = A$$

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

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

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

$$\begin{aligned} AB + \bar{A}C + BC &= AB + \bar{A}C + BC(A + \bar{A}) \\ &= AB + \bar{A}C + \underline{ABC} + \underline{\bar{A}BC} \\ &= AB(1 + C) + \bar{A}C(1 + B) \\ &= AB + \bar{A}C \end{aligned}$$

● **消去法** 运用吸收律  $A + \overline{A}B = A + B$ ，消去多余因子。

$$\begin{aligned}\star Y &= AB + \overline{A}C + \overline{B}C \\ &= AB + (\overline{A} + \overline{B})C = \underline{AB} + \underline{\overline{A}BC} = AB + C\end{aligned}$$

$$\begin{aligned}\star Y &= \overline{A}\overline{B} + \overline{A}B + \underline{ABCD} + \underline{\overline{A}\overline{B}CD} \\ &= \underline{\overline{A}\overline{B}} + \underline{\overline{A}B} + CD(\underline{AB} + \underline{\overline{A}\overline{B}}) \\ &= A \oplus B + CD \cdot \overline{A \oplus B} \\ &= A \oplus B + CD \\ &= \overline{A}\overline{B} + \overline{A}B + CD\end{aligned}$$

● **配项法** 通过乘  $A + \bar{A} = 1$  或加入零项  $A \cdot \bar{A} = 0$  进行配项，然后再化简。

$$\begin{aligned}\star Y &= AB + \bar{B}\bar{C} + \bar{A}\bar{C}D = AB + \bar{B}\bar{C} + \underline{\bar{A}\bar{C}D \cdot (B + \bar{B})} \\ &= \underline{AB} + \underline{\bar{B}\bar{C}} + \underline{\bar{A}BCD} + \underline{\bar{A}\bar{B}CD} \\ &= AB + \bar{B}\bar{C}\end{aligned}$$

$$\begin{aligned}\star Y &= \underline{ABC} + \overline{ABC} \cdot \overline{AB} \\ &= \underline{ABC} + \overline{ABC} \cdot \overline{AB} + \underline{AB \cdot \overline{AB}} \\ &= AB(\overline{AB} + \bar{C}) + \overline{ABC} \cdot \overline{AB} \\ &= AB \cdot \overline{ABC} + \overline{ABC} \cdot \overline{AB} \\ &= \overline{ABC} = \bar{A} + \bar{B} + \bar{C}\end{aligned}$$



## ● 综合灵活运用上述方法

[例] 化简逻辑式  $Y = \underline{AD} + \underline{A\bar{D}} + AB + \bar{A}C + \bar{C}D + \bar{A}BEF$

解:  $Y = \underline{A} + \underline{AB} + \bar{A}C + \bar{C}D + \underline{\bar{A}BEF}$

$$= \underline{A} + \bar{A}C + \bar{C}D \quad \boxed{\text{应用 } A + \bar{A}B = A + B}$$

$$= A + \underline{C} + \bar{C}D = A + C + D$$


[例] 化简逻辑式  $Y = \overline{A+B} \cdot \overline{ABC} \cdot \overline{\overline{AC}}$

解:  $\overline{Y} = \overline{\overline{A+B} \cdot \overline{ABC} \cdot \overline{\overline{AC}}}$  用摩根定律

$$= \underline{A} + B + \underline{ABC} + \overline{\overline{AC}}$$

$$= \underline{A} + B + \underline{\overline{AC}} \quad \text{应用 } A + \overline{AB} = A + B$$

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

  $Y = \overline{A+B+\overline{C}} = \overline{ABC}$

### 三、指定器件的逻辑函数化简（\*）

用门电路实现逻辑函数时，需要使用与门、或门、非门、与或非门等器件，究竟将函数式变换成什么形式，要视所用门电路的功能而定。

**例1：**将逻辑函数  $Y = \overline{\overline{AB} + C} + \overline{BC} + BD$  化为与非—与非形式。

**例2：**试用或非门画出函数  $Y = A + B\overline{C}$  的逻辑图。

**注：**将最简与—或式直接变换为其他类型的逻辑式时，得到的结果不一定是简的。

## 最小项的概念与性质

### 1. 最小项的定义和编号

$n$  个变量有  $2^n$  种组合，可对应写出  $2^n$  个乘积项，这些乘积项均具有下列特点：**包含全部变量，且每个变量在该乘积项中（以原变量或反变量）只出现一次。**这样的乘积项称为这  $n$  个变量的最小项，也称为  $n$  变量逻辑函数的最小项。

- 最小项定义

逻辑函数中，如果一个与项（乘积项）包含该逻辑函数的全部变量，且每个变量或以原变量或以反变量只出现一次，则该与项称为最小项。对于 $n$ 个变量的逻辑函数共有 $2^n$ 个最小项。

- 最小项的表示

最小项用 $m$ 表示，通常用十进制数作最小项编号。把最小项中的原变量当作1，反变量当作0，所得的二进制数所对应的十进制数即为最小项的编号。

- 若干最小项之和构成最小项表达式（也叫标准与-或表达式）：

例如

3 变量逻辑函数的最小项有  $2^3 = 8$  个

将输入变量取值为 1 的代以原变量，取值为 0 的代以反变量，则得相应最小项。

<i>A</i>	<i>B</i>	<i>C</i>	最小项	简记符号	输入组合对应的十进制数
0	0	0	$\overline{A}\overline{B}\overline{C}$	$m_0$	0
0	0	1	$\overline{A}\overline{B}C$	$m_1$	1
0	1	0	$\overline{A}B\overline{C}$	$m_2$	2
0	1	1	$\overline{A}BC$	$m_3$	3
1	0	0	$A\overline{B}\overline{C}$	$m_4$	4
1	0	1	$A\overline{B}C$	$m_5$	5
1	1	0	$AB\overline{C}$	$m_6$	6
1	1	1	$ABC$	$m_7$	7

例如  $\overline{A}BC \Rightarrow 101 \Rightarrow 5 \Rightarrow m_5$

$m_4 \Rightarrow 4 \Rightarrow 100 \Rightarrow A\overline{B}\overline{C}$

## 2. 最小项的基本性质

- (1) 对任意一最小项，只有一组变量取值使它的值为 1，而其余各种变量取值均使其值为 0。
- (2) 不同的最小项，使其值为 1 的那组变量取值也不同。
- (3) 对于变量的任一组取值，任意两个最小项的乘积为 0。
- (4) 对于变量的任一组取值，全体最小项的和为 1。

### 三变量最小项表

## 二、逻辑函数的“最小项之和”形式——标准“与或”表达式（**积之和**形式）（**SOP**）

利用基本公式  $A + \overline{A} = 1$ ，可将任何一个逻辑函数化为**最小项之和**的标准形式。这种标准形式在逻辑函数的化简以及计算机辅助分析和设计中得到了广泛的应用。

**例：**试将逻辑函数  $Y = \overline{A}\overline{B}\overline{C}\overline{D} + \overline{A}CD + AC$  展为**最小项之和**的形式。

任何函数  $f$  都可以用**真值表**中使  $f = 1$  的各行所对应的最小项的**逻辑和**来表示。

若逻辑表达式由乘积项（与）的逻辑和（或）的组成，则该逻辑表达式被称为**积之和**形式。若每个乘积项都为最小项，则该式被称为函数  $f$  的**正则积之和**表达式。



- 最大项定义

逻辑函数中，如果一个或项（和项）包含该逻辑函数的全部变量，且每个变量或以原变量或以反变量只出现一次，则该或项称为最大项。对于 $n$ 个变量的逻辑函数共有 $2^n$ 个最大项。

- 最大项的表示

最大项用 $M$ 表示，通常用十进制数作最大项编号。把最大项中的原变量当作0，反变量当作1，所得的二进制数所对应的十进制数即为最大项的编号。

- 若干最大项之积构成最大项表达式（也叫标准或-与表达式）：
- 变量数相同时，下标编号相同的最大项和最小项应为互补。

### 三变量最大项编号表

最大项	使最大项为 0 的变量取值			对应的十进制数	编 号
	A	B	C		
$A + B + C$	0	0	0	0	$M_0$
$A + B + \overline{C}$	0	0	1	1	$M_1$
$A + \overline{B} + C$	0	1	0	2	$M_2$
$A + \overline{B} + \overline{C}$	0	1	1	3	$M_3$
$\overline{A} + B + C$	1	0	0	4	$M_4$
$\overline{A} + B + \overline{C}$	1	0	1	5	$M_5$
$\overline{A} + \overline{B} + C$	1	1	0	6	$M_6$
$\overline{A} + \overline{B} + \overline{C}$	1	1	1	7	$M_7$

思考：最大项与最小项之间存在什么关系？

$$M_i = \overline{m_i}, m_i = \overline{M_i}$$

Row number	$x_1$	$x_2$	$x_3$	Minterm	Maxterm
0	0	0	0	$m_0 = \bar{x}_1\bar{x}_2\bar{x}_3$	$M_0 = x_1 + x_2 + x_3$
1	0	0	1	$m_1 = \bar{x}_1\bar{x}_2x_3$	$M_1 = x_1 + x_2 + \bar{x}_3$
2	0	1	0	$m_2 = \bar{x}_1x_2\bar{x}_3$	$M_2 = x_1 + \bar{x}_2 + x_3$
3	0	1	1	$m_3 = \bar{x}_1x_2x_3$	$M_3 = x_1 + \bar{x}_2 + \bar{x}_3$
4	1	0	0	$m_4 = x_1\bar{x}_2\bar{x}_3$	$M_4 = \bar{x}_1 + x_2 + x_3$
5	1	0	1	$m_5 = x_1\bar{x}_2x_3$	$M_5 = \bar{x}_1 + x_2 + \bar{x}_3$
6	1	1	0	$m_6 = x_1x_2\bar{x}_3$	$M_6 = \bar{x}_1 + \bar{x}_2 + x_3$
7	1	1	1	$m_7 = x_1x_2x_3$	$M_7 = \bar{x}_1 + \bar{x}_2 + \bar{x}_3$

从最大项的定义出发可以证明它具有如下的重要性质：

(1) 在输入变量的任何取值下必有一个且仅有一个最大项的值为0；

(2) 全体最大项之积为0；

(3) 某一最大项若不包含在 $F$ 中，则必在 $\overline{F}$ 中；

(4) 任意两个最大项之和为1；

(5) 只有一个变量不同的两个最大项的乘积等于各相同变量之和。

### 三、逻辑函数的“**最大项之积**”形式——标准“或与”表达式（**和之积**形式**POS**）

**证明：**任何一个逻辑函数都可以化成最大项之积的标准形式。

**例：**试将逻辑函数  $Y = A\overline{B}\overline{C} + BC$

化为最大项之积的标准形式。

**对偶原理**指出：若考虑真值表中使  $f = 1$  的各行可以综合出一个函数  $f$ ，则考虑使  $f = 0$  的各行也可以综合出函数  $f$

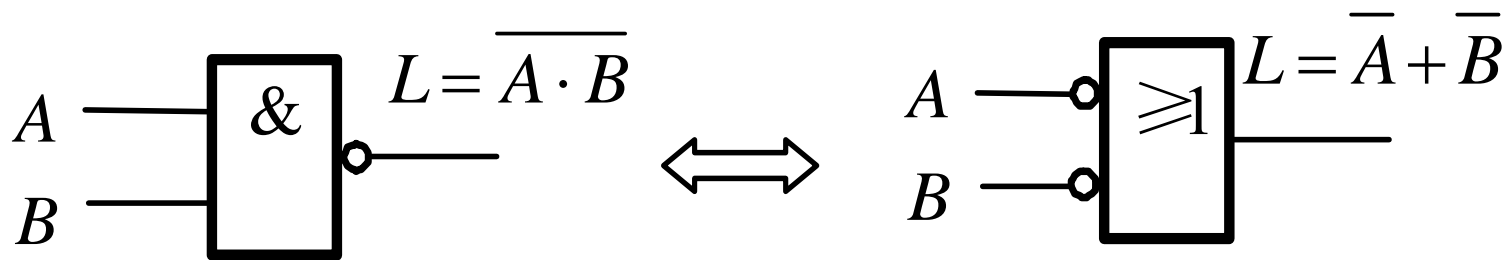
任何函数  $f$  可以通过找到它的**正则和之积**来实现综合，即从真值表中的每一行中取出使  $f = 0$  的最大项，把这些最大项组成乘积。

# 与非和或非逻辑网络

系统输入信号中，有的是高电平有效，有的是低电平有效。

低电平有效，输入端加小圆圈；高电平有效，输入端不加小圆圈。

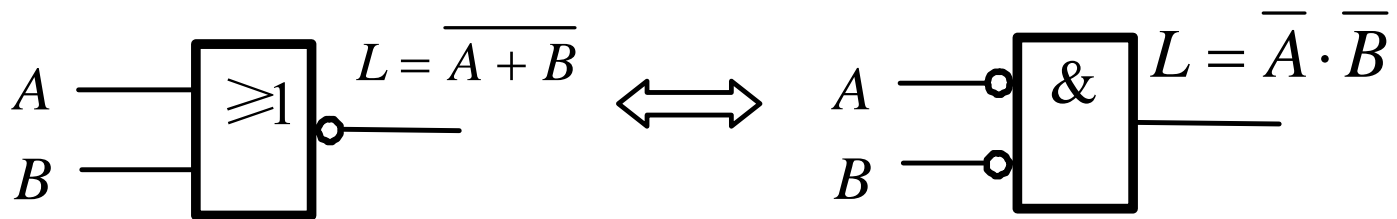
## 1、 基本逻辑门电路的等效符号 $L = \overline{A \cdot B} = \overline{A} + \overline{B}$



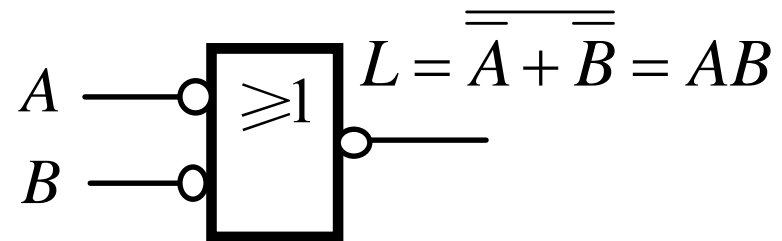
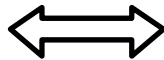
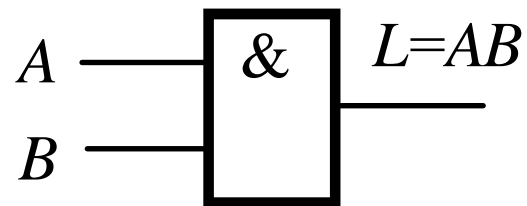
与非门及其等效符号

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

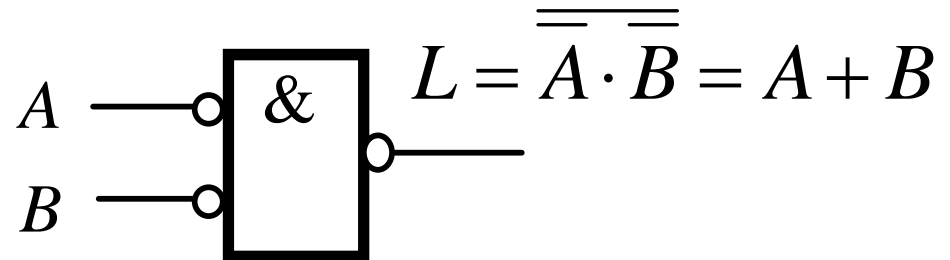
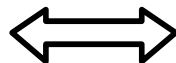
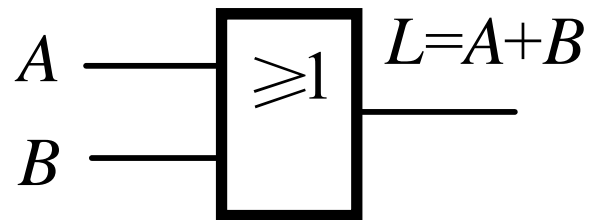
或非门及其等效符号



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



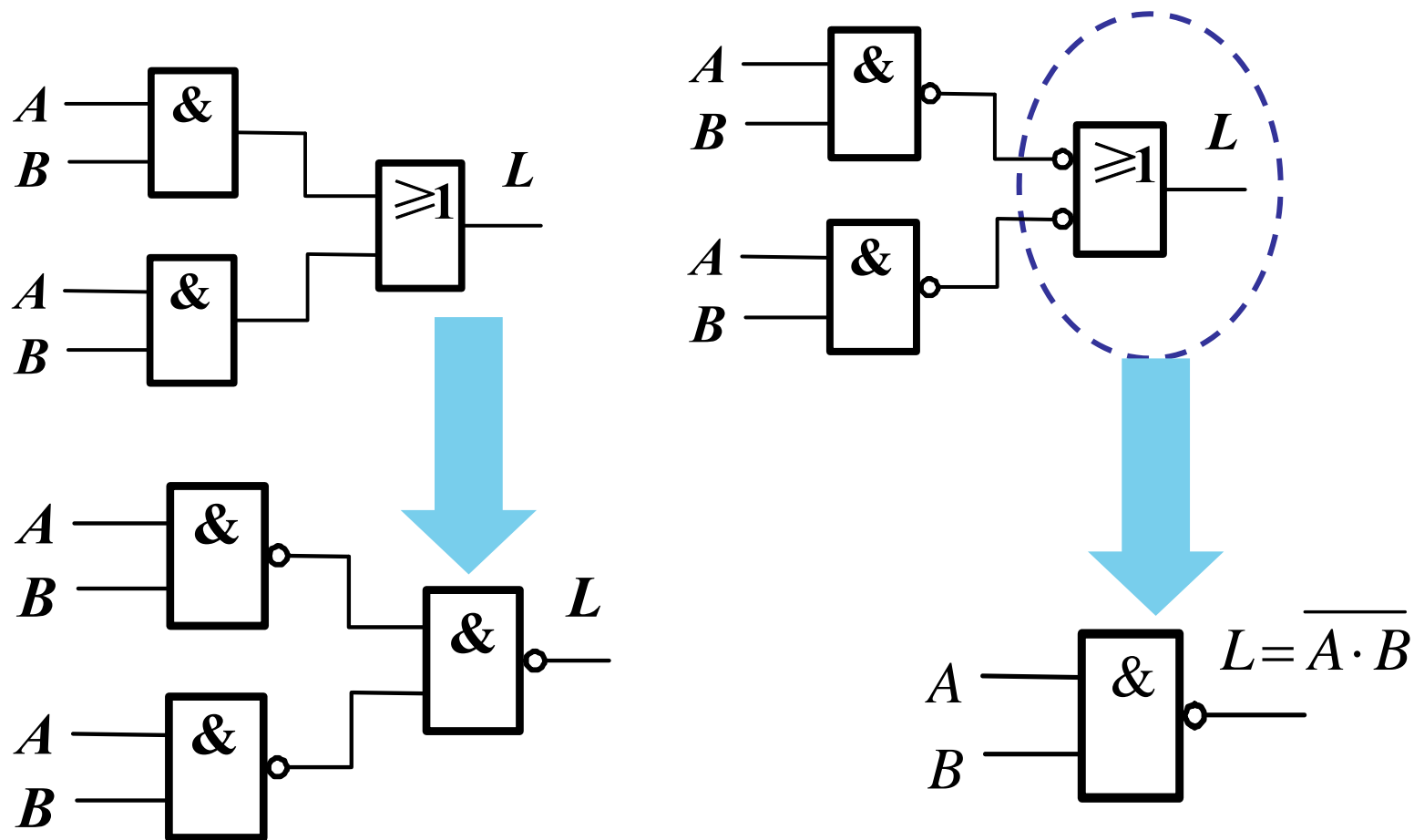
$$L = A + B = \overline{\overline{A} \cdot \overline{B}}$$



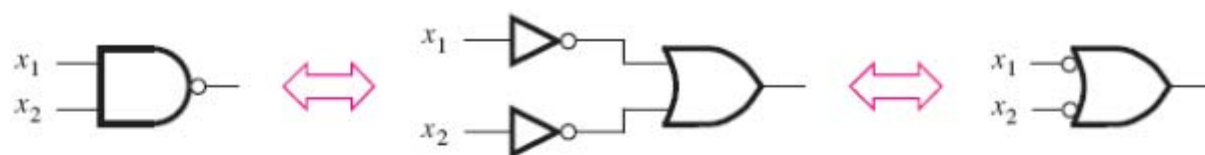


## 逻辑门等效符号的应用

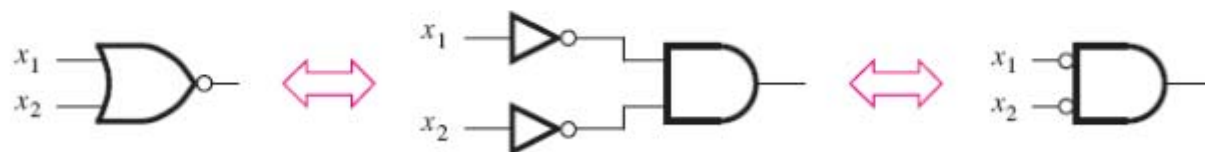
利用逻辑门等效符号，可实现对逻辑电路进行变换，以简化电路，能减少实现电路的门的种类。



# 与非和或非逻辑网络



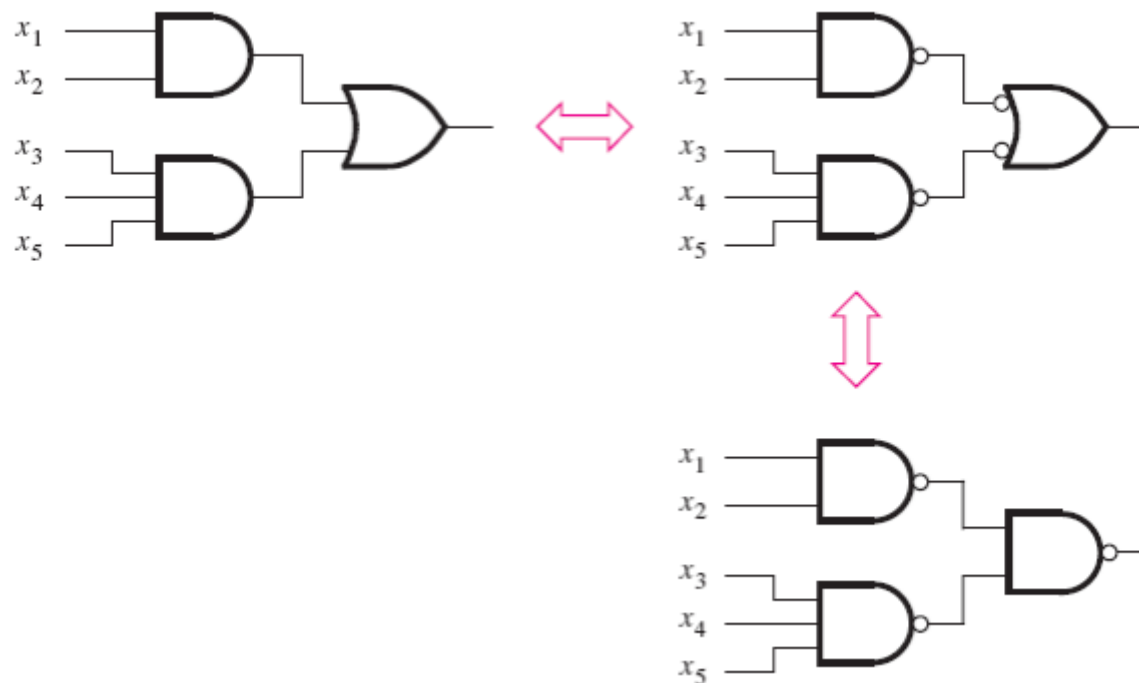
(a)  $\overline{x_1 x_2} = \bar{x}_1 + \bar{x}_2$



(b)  $\overline{\bar{x}_1 + \bar{x}_2} = x_1 x_2$

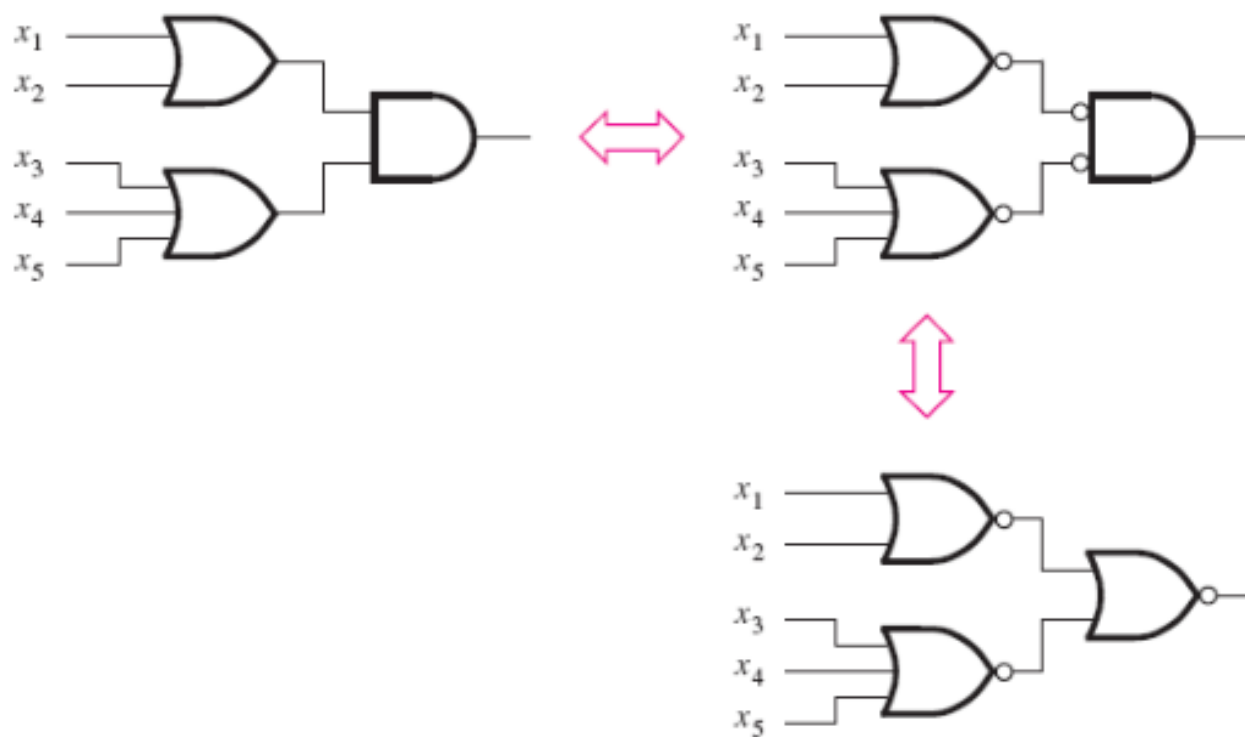
逻辑门项中的德摩根（ DeMorgan ）定理

# 使用与非门实现积之和形式



结论就是任何与-或网络都可以用同样拓扑结构的与非-与非网络实现。

# 使用或非门实现和之积形式



结论就是任何或 - 与网络都可以用同样拓扑结构的或非 - 或非网络实现。

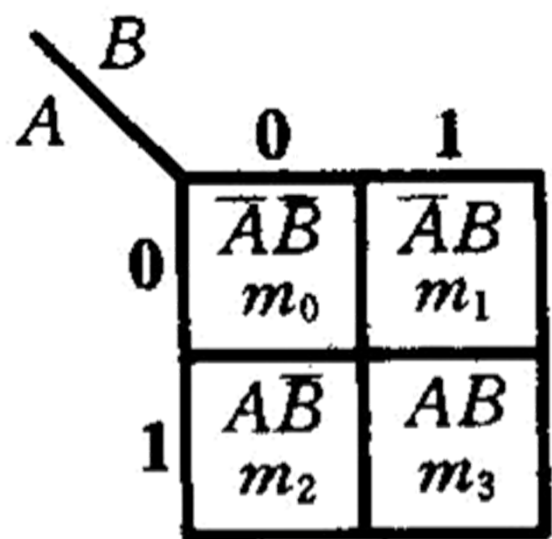
## 8 逻辑函数的卡诺图化简

### 一、逻辑函数的卡诺图表示法

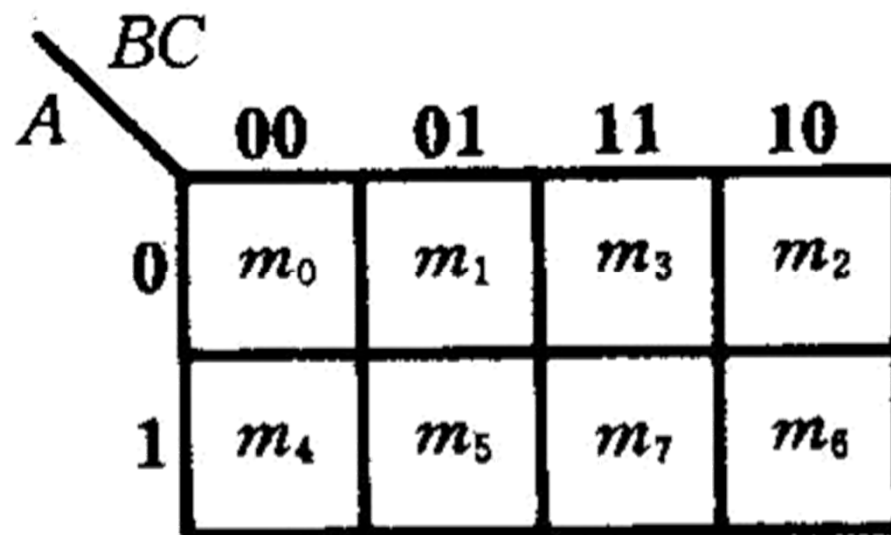
#### (一) 卡诺图 (Karnaugh Diagram)

卡诺图是由美国工程师卡诺首先提出的一种用来描述逻辑函数的特殊方格图。在这个方格图中，每个小方格代表逻辑函数的一个**最小项**，而且**几何相邻**的小方格具有**逻辑相邻性**，即两相邻小方格所代表的最小项只有一个变量取值不同。

二、三、四、五变量卡诺图为：



二变量卡诺图



三变量卡诺图

		<i>CD</i>			
		00	01	11	10
<i>AB</i>	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	$m_8$	$m_9$	$m_{11}$	$m_{10}$

四变量卡诺图

$AB \backslash CDE$									
		000	001	011	010	110	111	101	100
00		$m_0$	$m_1$	$m_3$	$m_2$	$m_6$	$m_7$	$m_5$	$m_4$
01		$m_8$	$m_9$	$m_{11}$	$m_{10}$	$m_{14}$	$m_{15}$	$m_{13}$	$m_{12}$
11		$m_{24}$	$m_{25}$	$m_{27}$	$m_{26}$	$m_{30}$	$m_{31}$	$m_{29}$	$m_{28}$
10		$m_{16}$	$m_{17}$	$m_{19}$	$m_{18}$	$m_{22}$	$m_{23}$	$m_{21}$	$m_{20}$

五变量卡诺图



## （二）卡诺图的特点：

1、卡诺图中的小方格数等于最小项总数，若逻辑函数的变量数为  $n$ ，则小方格数为  $2^n$  个。

2、卡诺图行列两侧标注的0和1表示使对应方格内最小项为1的变量取值。同时，这些0和1组成的二进制数大小就是对应最小项的编号。此外，在卡诺图中，几何相邻的最小项具有逻辑相邻性，因此，变量的取值不能按照二进制数的顺序排列，必须按循环码排列。

3、卡诺图是一个上下、左右闭合的图形，即不但紧挨着的方格是相邻的，而且上下、左右相对应的方格也是相邻的。

### (三) 用卡诺图表示逻辑函数

1、已知函数式  $\longrightarrow$  化成最小项之和形式  $\longrightarrow$   
卡诺图中对应最小项格填入“1”  $\longrightarrow$  得到卡诺图。

**例：**试用卡诺图表示逻辑函数

$$Y = \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}D + ACD + A\overline{B}$$

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

- 2、已知真值表  $\longrightarrow$  每组变量（即最小项）所对应的函数值  $\longrightarrow$  填入卡诺图中相应方格  $\longrightarrow$  化简得到函数式。
- 3、已知逻辑函数的卡诺图  $\longrightarrow$  该函数的真值表  $\longrightarrow$  写出该函数的逻辑函数式。

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

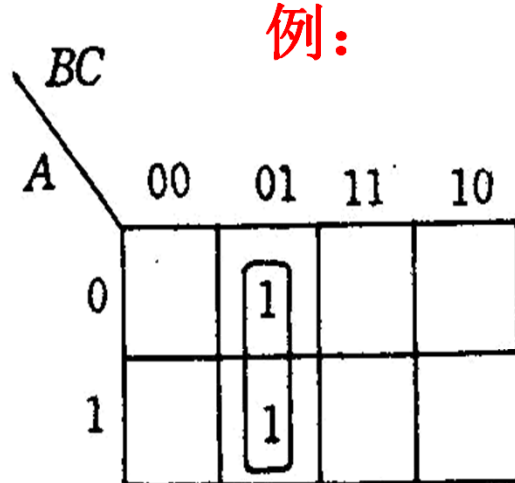
## 二、用卡诺图化简逻辑函数

### (一) 用卡诺图化简逻辑函数的依据（基本性质）

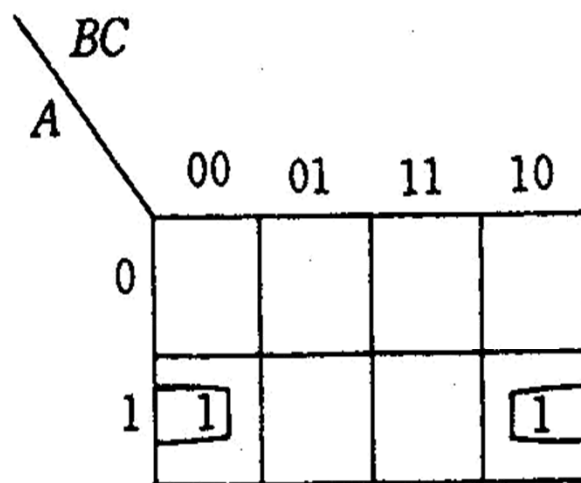
基本原理：各几何相邻方格的逻辑相邻性。

性质1：卡诺图中两个相邻“1”格的最小项可以合并成一个与项，并消去一个变量。

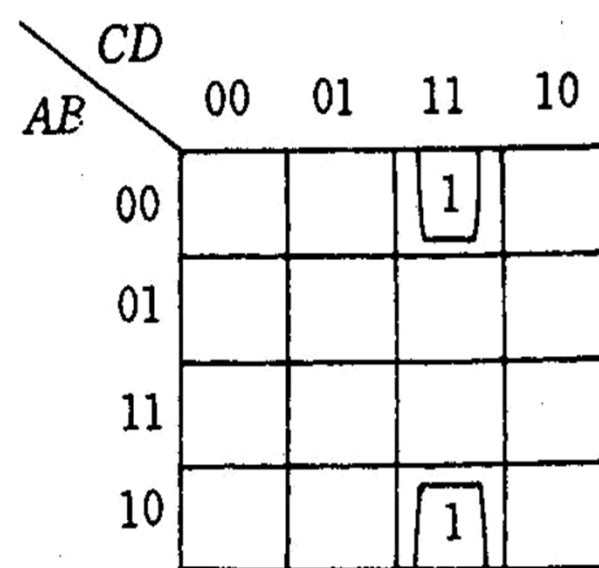
例：



(a)  $\bar{B}C$



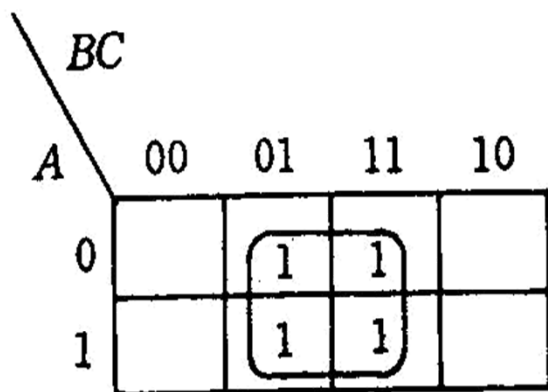
(b)  $A\bar{C}$



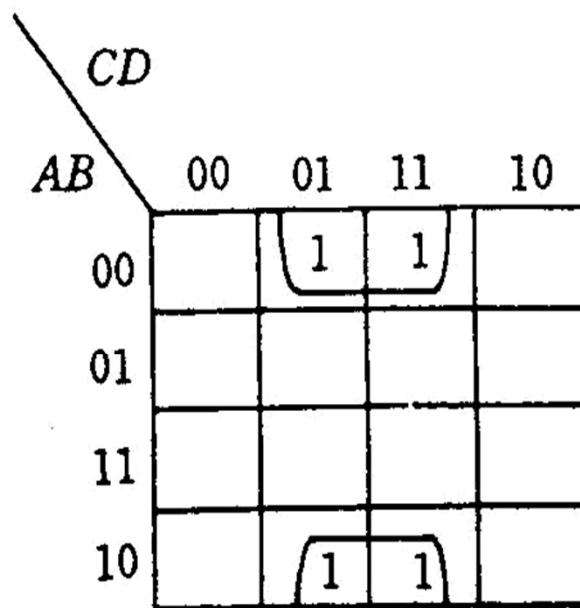
(c)  $\bar{B}CD$

**性质2:** 卡诺图中四个相邻“1”格的最小项可以合并成一个与项，并消去两个变量。

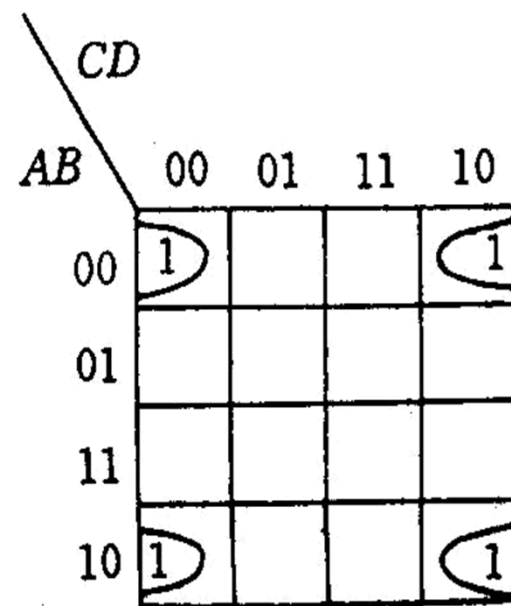
例:



(a)  $C$



(b)  $\bar{B}D$



(c)  $\bar{B}\bar{D}$

**性质3:** 卡诺图中八个相邻“1”格的最小项可以合并成一个与项，并消去三个变量。

例:

		<i>CD</i>			
		00	01	11	10
<i>AB</i>	00	1	1	1	1
	01				
	11				
	10	1	1	1	1

(a)  $\bar{B}$

		<i>CDE</i>							
		000	001	011	010	110	111	101	100
<i>AB</i>	00			1			1		
	01			1			1		
	11			1			1		
	10			1			1		

(b)  $DE$

**推论：**在 $n$ 个变量的卡诺图中，若有 $2^k$ 个“1”格相邻（ $k=0, 1, 2, 3, \dots, n$ ），它们可以圈在一起加以合并，合并时可以消去 $k$ 个不同的变量，简化为一个具有 $(n-k)$ 个变量的与项。若 $k=n$ ，则合并时可消去全部变量，结果为1。

## （二）用卡诺图求最简与或表达式

例1：用卡诺图化简函数

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

步骤：

1、得到函数的真值表或将函数化为最小项之和的标准形式；

2、画出函数的卡诺图；

3、合并最小项（即“画圈”）；

“画圈”时应注意的问题：

- ① “1”格一个也不能漏，否则表达式与函数不等；
- ② “1”格允许被一个以上的圈包围，因为 $A+A=A$ ；



- ③ 圈的个数应尽可能少，因为一个圈对应一个与项，即与项最少；

例：

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

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

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

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

- ④ 圈的面积越大越好，但必须为  $2^k$  个方格。这是因为圈越大，消去的变量就越多，与项中的变量数就越少。

例：

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

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

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

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

- ⑤ 每个圈至少应包含一个新的“1”格，否则这个圈是多余的，即增加了冗余项；

例：

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

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

总之，即“可以重画，不能漏画，圈数要少，圈面要大，每个圈必有一个新‘1’”。

4、写出最简“与-或”表达式。

**例2：**试用图形化简法求逻辑函数 $F(A, B, C, D) = \sum m(1, 2, 4, 9, 10, 11, 13, 15)$ 的最简与或表达式。

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

例3：用卡诺图化简函数

$$Y = \sum m(1, 3, 6, 7, 14)$$

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

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

注：卡诺图化简得到的最简与或式不一定唯一。

## 一、约束项、任意项和逻辑函数式中的无关项

**约束项**——在某些情况下，输入变量的取值不是任意的。当限制某些输入变量的取值不能出现时，可以用它们对应的最小项恒等于0来表示。这些恒等于0的最小项叫约束项。

**任意项**——有时输入变量的某些取值是1还是0皆可，并不影响电路的功能。在这些变量取值下，其值等于1的那些最小项称为任意项。

**无关项**——约束项和任意项统称为逻辑函数中的无关项。“无关”指是否将这些最小项写入逻辑函数式无关紧要，在卡诺图中用“×”表示无关项。在化简逻辑函数时，可认为它是1，也可认为它是0。

## 二、无关项在化简逻辑函数中的应用

化简具有无关项的逻辑函数时，如果能合理利用这些无关项，一般都可以得到更加简单的化简结果。

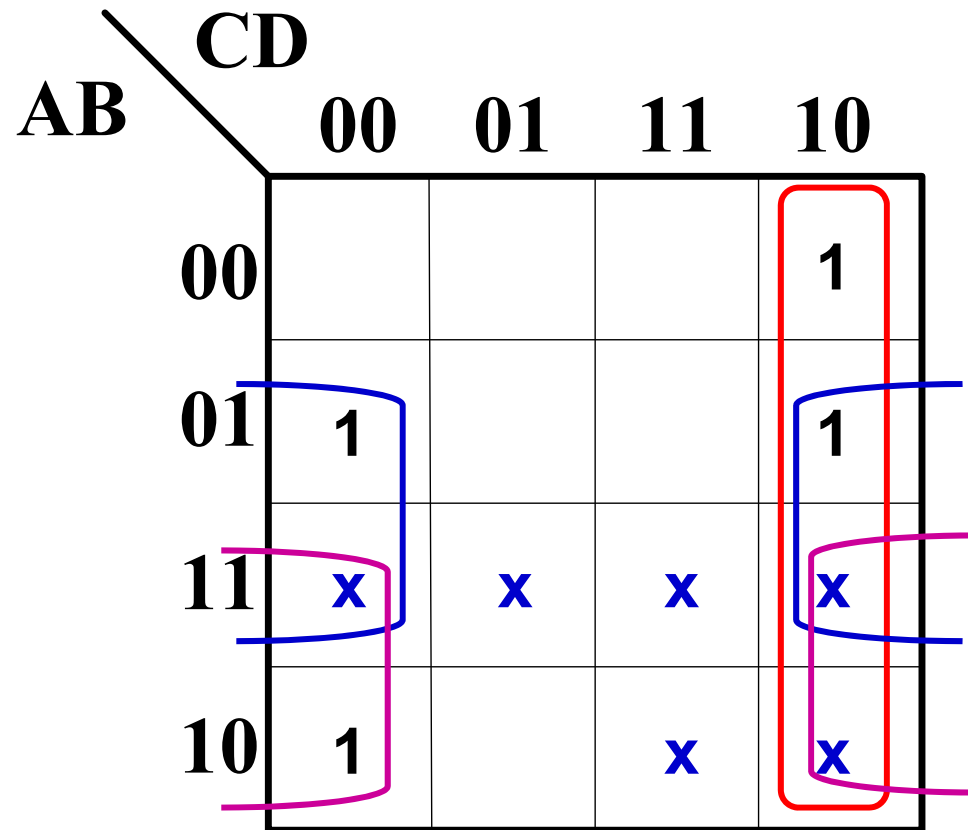
合并最小项时，究竟把卡诺图上的“×”作为1还是0，应以得到的相邻最小项矩形组合最大，而且矩形组合数目最小为原则。

**例：**试化简逻辑函数

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

已知约束条件为：

$$\overline{A}B\overline{C}\overline{D} + \overline{A}B\overline{C}D + \overline{A}B\overline{C}\overline{D} + \overline{A}B\overline{C}D + \overline{A}B\overline{C}\overline{D} + \overline{A}B\overline{C}D = 0$$





**例：**试用卡诺图化简逻辑函数

$$F(W, X, Y, Z) = \sum_m (1, 3, 7, 11, 15) + \sum_d (0, 2, 5)$$

		YZ			
		00	01	11	10
WX	00	×	1	1	×
	01		×	1	
	11			1	
	10			1	

思考：由上例可得出什么结论和启示？

解答：此例有两种解法，从原理而言，两种解法均正确，但就“最简”原则而言，只有一种解法最简单、最可取。因此，在考虑卡诺图化简不唯一性的同时，还应考虑“最简”原则。

**思考1：**公式化简和卡诺图化简各有何优缺？

化简法	优点	缺点
公式法	化简不受输入变量数目的影响。	化简过程没有固定的、通用的步骤可循，不适用于计算机辅助化简。
卡诺图法	直观、简单	输入变量数目较多时（例如>5），不再直观，且化简需凭设计者的经验，不便于利用计算机完成化简工作。