§1 数字逻辑概论

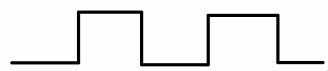
① 数字信号描述方法

电信号 $\begin{cases} \text{模拟电信号：连续} \\ \text{数字电信号：离散} \end{cases}$

数字信号的描述方法

(1) 0，1 数码 $\begin{cases} \text{表示数量时称二进制数} \\ \text{表示事物状态时称二值逻辑} \end{cases}$

(2) 逻辑电平 $\begin{cases} \text{高电平：电压 } V_{H(min)} \sim +V_{DD} \quad 1 \\ \text{低电平：电压 } 0 \sim V_{L(max)} \quad 0 \end{cases}$

(3) 数字波形 

一般采用二值数字逻辑

实际脉冲波形及主要参数

① 幅值

② 上升时间 t_r (ns) $10\% V_m - 90\% V_m$ 所经历的时间

③ 下降时间 t_f (ns)

④ 周期 T 频率 $f = \frac{1}{T}$

⑤ 脉冲宽度 t_w 上升 $50\% V_m$ － 下降 $50\% V_m$

⑥ 占空比 $q = \frac{t_w}{T} \times 100\%$

② 数制

(1) 数制：计数规则（构成方法＋进位规则）

十进制：采用 0~9 数码，"逢10进1"

$55.316 = 5\times10^1 + 5\times10^0 + 3\times10^{-1} + 1\times10^{-2} + 6\times10^{-3}$

推广

一般表达式 $(N)_D = \sum_{i=-\infty}^{\infty} (\underbrace{K_i}_{\text{系数}} \times \underbrace{10^i}_{\text{位权}})$

R进制：以R为基数的计数体制

采用0~R-1数码，"逢R进1"

$$N_R = \sum_{i=-\infty}^{n} (k_i \times R^i)$$

二进制：多位二进制中的每一个数码称为1位 (1 bit)

Binary　8位二进制数称为一个字节 (Byte)

e.g.

	7	6	5	4	3	2	1	0
	1	1	0	1	1	1	0	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

MSB (Most Significant Bit)　　LSB

$$(N)_B = \sum_{i=-\infty}^{n} (k_i \times 2^i)$$

Octal **八进制**　$(N)_D = \sum_{i=-\infty}^{n} (k_i \times 8^i)$

十六进制　0 1 2 3 4 5 6 7 8 9 A B C D E F

Hexadecimal　$(N)_H = \sum_{i=-\infty}^{n} (k_i \times 16^i)$

(2) **二-十进制数转换**

二 ⟶ 十　$\sum_{i=-\infty}^{n} (k_i \times 2^i)$

十 ⟶ 二

① 整数部分

加权求和法

确定一组二进制权使它们的和等于已知的十进制数

在计算机中　2^{10} - 1K　2^{20} - 1M

2^{30} - 1G　2^{40} - 1T

e.g. $(9)_D = 8+1 = 1\times 2^3 + 1\times 2^0 = 1001$

重复除以2取余法

e.g.

		余
2	45	1
2	22	0
2	11	1
2	5	1
2	2	0
2	1	1
	0	

$(45)_D = (101101)_B$

② 小数部分

加权求和法

e.g. $(0.625)_{10} = 0.5 + 0.125 = 2^{-1} + 2^{-3} = (0.101)_B$

重复乘以2取整法　　$(0.3125)_D = (0.0101)_B$

e.g.

$0.3125 \times 2 = 0.625$	取	0
$0.625 \times 2 = 1.25$		1
$0.25 \times 2 = 0.5$		0
$0.5 \times 2 = 1.0$		1

e.g. 要求转换误差小于1%

即 $2^{-m} \le 1\% \Rightarrow m \ge \frac{2}{\lg 2} = 6.64$

即转换到小数点后第7位

(3) 其他不同数制之间的转换

二 → 十六：将4位二进制看作一个整体

以二进制的小数点为基准，

将小数点左边的整数从右到左每4个分为一组；

… 右 … 从左到右 …

将每组数以1个十六位数代替

e.g.						
二	1101	1001	1011	0011	.	0100
十六	D	9	B	3	.	4

十 → 十六

方法①：十 → 二 → 十六

方法②：仿照"十→二"的方法

e.g.

16	650	余 A
16	40	8
16	2	2
	0	

$(650)_D = (28A)_H$

3 二进制数的算术运算

(1) 二进制数的算术运算

① 加法运算　　和十进制类似　　"逢二进一"

② 减法运算

③ 乘法运算

$0\times0=0$　　$0\times1=0$　　$1\times0=0$　　$1\times1=1$

④ 除法运算

$0\div1=0$　　$1\div1=1$

(2) 有符号数的表示

计算机无法识别(+)/(-) ⇒ 将 +/- 也用 0/1 表示

机器数/机器码：将数的符号和数值部分均用0、1进行编码所表示出来的二进制数

真值：用+/-表示出来的十进制数或二进制数.

机器数
- 无符号数：所有二进制位均用来表示数值
- 有符号数：数的符号和编码均用二进制编码表示

正0
负1

- 原码：最高位符号，其余相同
- 反码：最高位符号，正数与原码相同，负数按位取反.
- 补码："模"指一个系统的量程

eg. 时钟模为12，4位二进制数模为16

补码的补码⇒原码　　8位二进制数模为256

减去某数可用加上其补码代替

最高位符号，正数与原码相同，负数按位取反后，在最低位+1

(2) 补码的加减运算

$[X]_{补} + [Y]_{补} = [X+Y]_{补}$

$[X]_{补} - [Y]_{补} = [X]_{补} + [-Y]_{补} = [X-Y]_{补}$

注意：

① 参与运算的操作均为补码

② 符号位和数值位按同样的规则参加运算，结果的符号位由运算得出.

③ 补码总是对确定的模而言，如果运算结果超过了模，则去掉模（即进位）才能得到正确结果.

"溢出" ⇒ 位扩展

△ n位有符号的二进制数的原码、反码和补码的数值范围

原码： $-(2^{n-1}-1) \sim +(2^{n-1}-1)$

反码： $-(2^{n-1}-1) \sim +(2^{n-1}-1)$

补码： $-2^{n-1} \sim +(2^{n-1}-1)$

4 二进制代码

码制：编制代码所要遵循的规则

二进制代码：将若干个二进制数码（0和1）按一定规律排列起来表示某种特定的信息，称为二进制代码.

编码：用n位二制数表示2^n个不同的信息，给每个信息规定一个具体的二进制数编码，这个过程为编码.

(1) 二一十进制码 (Binary - Coded - Decimal)

用二进制编码表示的十进制数，简称BCD码

$2^3 < 10 < 2^4$ ⇒ 需要4位二进制数

16种组合 ⇒ 选择10种表示 ⇒ 每种方案表示一种BCD码

e.g. 8421BCD码（最常用）

十进制数 ⟺ BCD码

$(97)_D$ ⟺ $(1001\ 0111)_{BCD}$

注意："十进制与二进制转换" 与 "用BCD码表示十进制数" 概念不同!!

0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

几种常用的BCD代码

无权码

十进制数码	"权" 8421码	2421 码	5421 码	余3码	余3循环码
0	0000	0000	0000	0011	0010
1	0001	0001	0001	0100	0110
2	0010	0010	0010	0101	0111
3	0011	0011	0011	0110	0101
4	0100	0100	0100	0111	0100
5	0101	1011	1000	1000	1100
6	0110	1100	1001	1001	1101
7	0111	1101	1010	1010	1111
8	1000	1110	1011	1011	1110
9	1001	1111	1100	1100	1010

(2) 格雷码

无权码、循环码、反射码

特点：① 两个相邻代码（包括首和尾）之间仅有一位取值不同.

② 最高位的0和1只改变一次；若以最高位的0和1交界为轴，其他位的代码是上下对称的.

用途：主要用于角度编码 （不能直接进行算术运算）

二进制码转换为格雷码

① 格雷码的最高位与二进制码的最高位相同

② 从左到右，逐一将二进制码相邻二位相加（舍去进位），

作为格雷码的下一位.

e.g. 二进制码 1 0 1 1 0 1

		$1+0$	$0+1$	$1+1$	$1+0$	$0+1$
二进制码	1	0	1	1	0	1
格雷码	1	1	1	0	1	1

(3) ASCII码和奇偶校验码

ASCII码：七位二进制编码 共128个

奇偶校验码：有效信息(k位) + 检验位(1位)

- 奇检验码：信息位和检验位中"1"个数之和为奇数
- 偶检验码：信息位和检验位中"1"个数之和为偶数

$\boxed{\text{发送端}} \longrightarrow \boxed{\text{接收端}}$

由编码器根据信息位编码产生奇偶检验位　　　通过检测器检查含"1"个数的奇偶

注意. ① 只有检错能力，无纠错能力

② 只能发现单个错误，不能发现双错

§2 逻辑代数

① 逻辑代数简介

George Boole

↓

Claude E. Shannon

逻辑代数 / 布尔代数

按一定逻辑规律进行运算的代数

布尔代数是分析继电器的有效方法

"二值逻辑"

在数字电路中
- 条件 → 输入信号
- 结果 → 输出信号

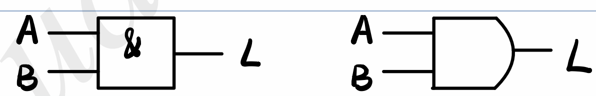
条件和结果的状态用逻辑"1"和"0"表示

② 逻辑运算和集成逻辑门简介

(1) 基本逻辑运算

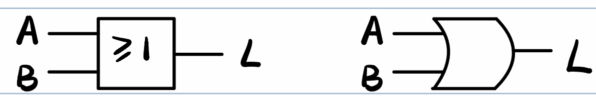
与　与门电路

$L = A \cdot B$



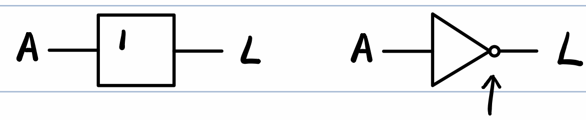
或　或门电路

$L = A + B$



非　非门电路

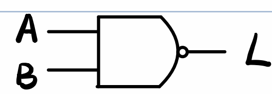
$L = \overline{A}$



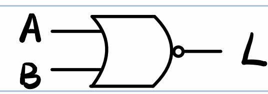
表示反相

(2) 复合逻辑运算

与非逻辑运算　$L = \overline{A \cdot B}$



或非逻辑运算　$L = \overline{A + B}$

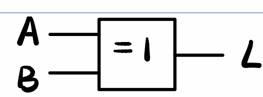


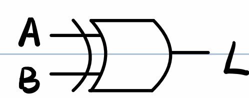
异或逻辑运算

A、B相同时，输出 $L = 0$

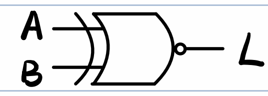
A、B不同时，输出 $L = 1$

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





同或逻辑运算

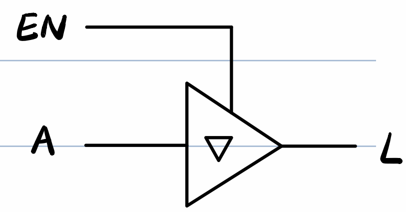


A、B相同时，输出 $L=1$

A、B不同时，输出 $L=0$

$L = A \odot B = AB + \bar{A}\bar{B}$

(3) 三态门



有三种可能的输出：0、1、Z

Z 指输出为高阻态

Z 意味着输入与输出之间是断开的

真值表

使能EN	输入A	输出L
1	0	0
1	1	1
0	X	Z

(4) 集成电路简介

集成电路 { 双极型、单极型、混合型

传输延迟：输出滞后于输入

高→低 延迟 t_{pHL}

低→高 延迟 t_{pLH}

平均传输延迟时间 $t_{pd} = \frac{1}{2}(t_{pHL} + t_{pLH})$

3 逻辑代数的基本定理和规则

(1) 逻辑代数基本定律

0、1律 $A \cdot 0 = 0$ $A + 0 = A$

$A \cdot 1 = A$ $A + 1 = 1$

重叠律 $A\cdot A=A$ $A+A=A$

互补律 $A\cdot\bar{A}=0$ $A+\bar{A}=1$

还原律 $\bar{\bar{A}}=A$

交换律 $A\cdot B=B\cdot A$ $A+B=B+A$

结合律 $(A\cdot B)\cdot C=A\cdot(B\cdot C)$

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

可以以任何方式对变量分组

分配律 $A\cdot(B+C)=A\cdot B+A\cdot C$

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

反演律 / 摩根定理 $\overline{AB}=\bar{A}+\bar{B}$

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

可用完全归纳法证明

(2) 逻辑代数常用公式

吸收律 $A+A\cdot B=A$

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

补吸收律 $A+\bar{A}\cdot B=A+B$

重要公式 $AB+\bar{A}C+BC=AB+\bar{A}C$

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

(3) 逻辑代数基本规则

代入规则 在任何一个包含A的式子，用另一个逻辑式代替A，等式仍然成立.

反演规则 对于任一逻辑表达式L，将其中与(·)和或(+)交换、原变量和反变量交换、1和0交换，得到的表达式是原函数的反函数.

注意. ① 保持原有优先级

② 对反变量以外的非号保持不变

e.g. $L = \bar{A}\cdot\bar{B} + C\cdot D$

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

对偶规则　"或"、"与"互换，0、1互换

得到L的对偶式 L'

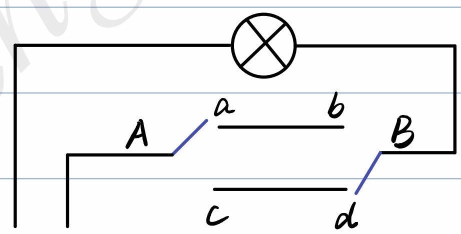
eg. $L = (A+\bar{B})(A+C)$

$L' = A\cdot\bar{B} + A\cdot C$

4 逻辑函数及其表示方法

输入（条件） $\xrightarrow[\text{逻辑函数(Logic Fuction)}]{}$ 输出（结果）

example.



(1) 逻辑真值表

开关 A、B　上(1)　下(0)

灯 L　灭(0)　亮(1)

A	B	L
0	0	1
0	1	0
1	0	0
1	1	1

$\bar{A}\bar{B}$

AB

(2) 逻辑函数表达式

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

步骤：

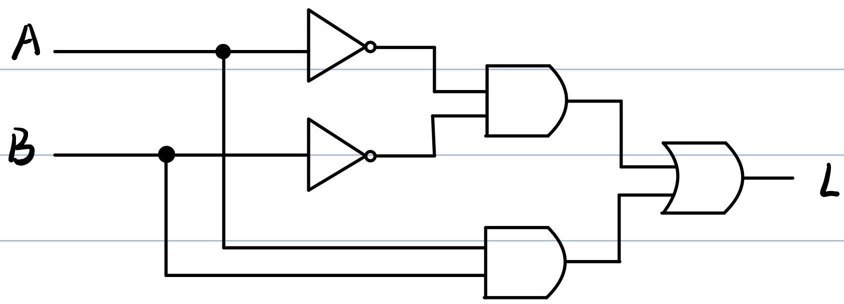
① 把每个输出为1的一组输入变量组合写成乘积项

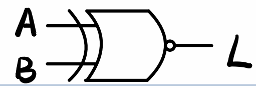
② 乘积项中，逻辑值1用原变量，0用反变量表示

③ 将乘积项进行逻辑或

注意：① 输入变量间是"与"
输出状态间组合是"或"
② 对A、B、L，取值1用原变量，0用反变量表示

(3) 逻辑图





(4) 波形图

(5) 卡诺图

(6) 硬件描述语言 (HDL)

5 逻辑函数的代数化简法

(1) 逻辑函数表达式的形式

与-或表达式　　积之和

或-与表达式　　和之积

(2) 逻辑函数的代数化简法

如何判断"与-或"最简？

① 包含"与"项的个数最少

② 每个"与"项中的变量数最少

化简的主要方法

① 公式法（代数法）　　② 图解法（卡诺图法）

（之后介绍）

- 并项法　　$A+\bar{A}=1$
- 吸收法　　$A+AB=A$
- 消去法　　$A+\bar{A}B=A+B$
- 配项法　　$A=A(B+\bar{B})$

(3) 逻辑函数表达式的变换

一种集成电路芯片内部通常只有一种类型的逻辑门.
为了减少门的种类，适应芯片的情况，需要
变换逻辑表达式

e.g. 与非

$$L = A(B+C) = AB + AC$$ ① 取非 2 次

$$= \overline{\overline{AB + AC}}$$ ② 摩根定理

$$= \overline{\overline{AB} \cdot \overline{AC}}$$

或非

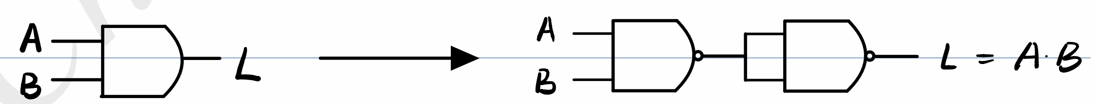
$$L = \bar{A}\bar{B}C + A\bar{B}\bar{C}$$

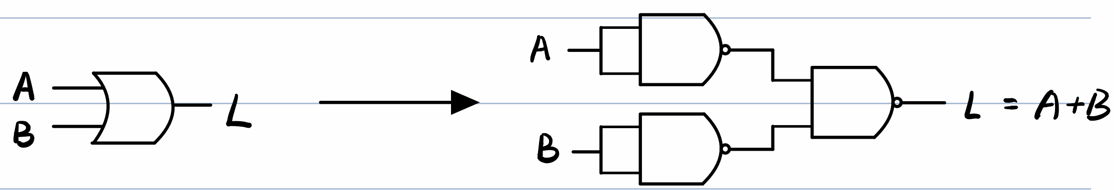
$$= \overline{\overline{\bar{A}\,\bar{B}\,C}} + \overline{\overline{A\bar{B}\bar{C}}}$$

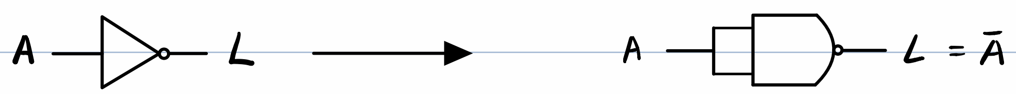
$$= \overline{A + B + \bar{C}} + \overline{\bar{A} + B + C}$$

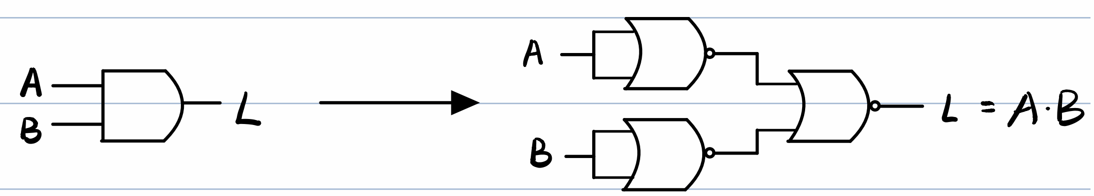
$$= \overline{\overline{\overline{A + B + \bar{C}} + \overline{\bar{A} + B + C}}}$$

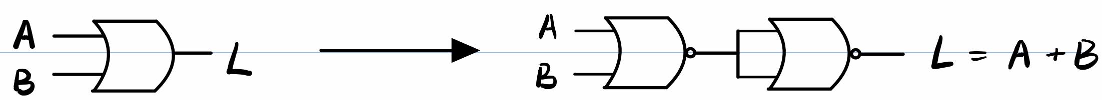
Note. ① 所有的逻辑电路可以只用与非门实现，
也可以只用或非门实现.

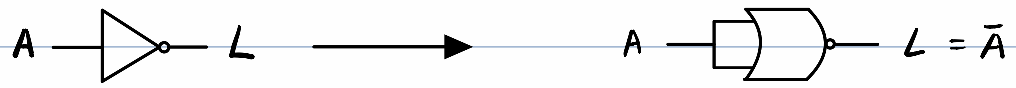
② 与


或


非


③ 与


或


非 

6 逻辑函数的卡诺图化简法

逻辑函数表达式的基本形式 $\begin{cases} \text{与一或表达式} \longrightarrow \text{最小项表达式} \\ \text{或一与表达式} \longrightarrow \text{最大项表达式} \end{cases}$

(1) 逻辑函数的最小项表达式

最小项：在n变量逻辑函数中，若一个乘积项包含了全部的n个变量，每个变量都以它的原变量或非变量的形式在乘积项中出现1次.

n个变量的最小项有2^n个

最小项的编号

e.g. $\bar{A}\bar{B}\bar{C}$ =进制 000 $\longrightarrow$ 十进制 0 $\xrightarrow{\text{记作}}$ m_0

$\bar{A}\bar{B}C$ =进制 001 $\longrightarrow$ 十进制 1 $\xrightarrow{\text{记作}}$ m_1

最小项的性质：

① ∀最小项，输入变量只有一组取值使其值为1，其它各组取值均使它为0.

② ∀取值，任意两个最小项乘积为0

③ ∀一组取值，全体最小项之和为1.

最小项表达式/标准与一或式

每个"与"项都是最小项.

e.g. $L(A,B,C) = AB + \bar{A}C$

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

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

$= m_7 + m_6 + m_3 + m_1$

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

逻辑函数变换为最小项表达式

① 变换为乘积项之和

② 用 $A+\bar{A}=1$ 配项

(2) 逻辑函数的最大项表达式 (与最小项表达式类似)

最大项：每一个或项都包含全部n个变量

最大项的编号 M_i

对于一个最大项，输入变量只有一组二进制数使其取值为0，与该组二进制数对应的十进制数是该最大项的下标编号.

e.g. $\bar{A}+\bar{B}+C$

$A=1 \quad B=1 \quad C=0 \longrightarrow \bar{A}+\bar{B}+C=0$

$110 \longrightarrow 6 \longrightarrow M_6$

最大项的性质

① ∀最大项，只有一组变量的取值使其值为0

② ∀二最大项之和为1

③ ∀一组变量取值，全体最大项之积为0

最大项表达式 / 标准或-与式

△ 最小项与最大项的关系 $M_i=\bar{m}_i$

e.g. 某电路真值表如下，试写出最小项和最大项表达式

① 最小项表达式：将L=1各最小项相加

$L(A,B,C) = m_3 + m_5 + m_6$

$= \sum m(3,5,6)$

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

② 最大项表达式：将L=0各最大项相乘

$L(A,B,C) = M_0 \cdot M_1 \cdot M_2 \cdot M_4 \cdot M_7$

A	B	C	L
0	0	0	0 M_0
0	0	1	0 M_1
0	1	0	0 M_2
0	1	1	1 → m_3
1	0	0	0 M_4
1	0	1	1 → m_5
1	1	0	1 → m_6
1	1	1	0 M_7

$$= \Pi M(0,1,2,4,7)$$

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

(3) 卡诺图的引出

① 相邻最小项：只有一个变量互为反变量

② 填写规则：逻辑相邻和几何位置相邻一致

③ "折叠展开"

i) 一变量卡诺图

0	1

ii) 两变量卡诺图

0	1	3	2

iii) 三变量卡诺图

0	1	3	2
4	5	7	6

iv) 四变量卡诺图

AB \ CD	00	01	11	10
00				
01				
11				
10				

0	1	3	2
4	5	7	6
12	13	15	14
8	9	11	10

(4) 逻辑函数的卡诺图表示法

方法 ① 逻辑函数 ⟶ 最小项表达式

② 填写卡诺图

③ L中列出的最小项，填1

L中不存在的最小项，填0

(4) 逻辑函数的卡诺图化简法

化简依据 $\begin{cases} \text{相邻有一项互为相反项} \\ A+\bar{A}=1 \end{cases}$

化简步骤 ① 逻辑函数 ⟶ 最小项之和

② 填写卡诺图

③ 画包围圈，每个包围圈含 2^n 个方格

④ 写出每个包围圈的乘积项，

将所有包围圈的乘积项相加。

画包围圈的一般规则

① 圈内方格数 2^n 个，$n=0,1,2,\cdots$

② 相邻方格包括上下底、左右边、四个角

③ 一个方格可被重复包围

④ 圈内方格尽可能多，包围圈数目尽可能少。

(5) 具有无关项的逻辑函数化简

无关项

无关项的处理 化简时可以取0或1，使函数尽可能简化。用"d"，"x"表示

单输出逻辑函数的化简

方法 ① 列真值表

② 画卡诺图

③ 画包围圈

e.g. $L(A,B,C,D)=\sum m(5,6,7,8,9)+\sum d(10,11,12,13,14,15)$

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

2^n

1, 2, 4, 8

$L(A,B,C,D) = A + BC + BD$

多输出逻辑函数的化简

tips：找共同部分，使整体最简（成本降低）

e.g. $L_1(A,B,C)=\sum m(0,4,5,6,7)$

$L_2(A,B,C)=\sum m(0,6,7)$

① L_1

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

② L_2

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

考虑共享乘积项 $\bar{A}\bar{B}\bar{C}$

$L_1(A,B,C)=A+\bar{A}\bar{B}\bar{C}$

$L_2(A,B,C)=AB+\bar{A}\bar{B}\bar{C}$

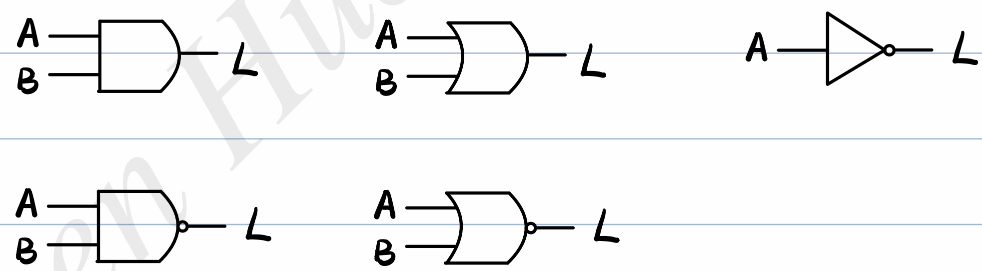
不考虑共享乘积项 $\bar{A}\bar{B}\bar{C}$

$L_1(A,B,C)=A+\bar{B}\bar{C}$

$L_2(A,B,C)=AB+\bar{A}\bar{B}\bar{C}$

7 逻辑门的替代符号

标准符号：

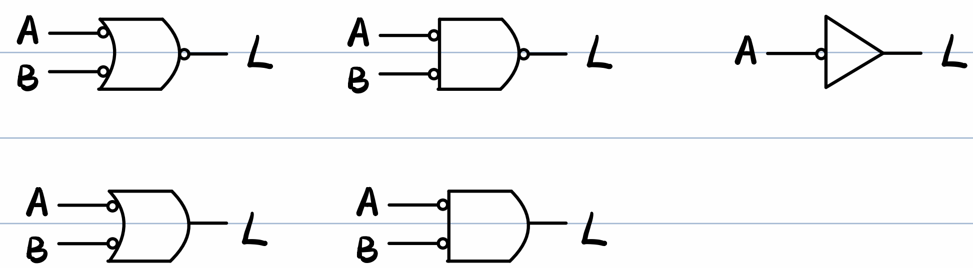


如何将标准符号变为替代符号

① 把标准符号的输出和每一个输入反相

② 运算符号"与" ⟷ "或"

替代符号



注意：① 等效逻辑符号可以推广到具有更多输入端的门电路.

② 标准符号的输入端无小圆圈，
所有替代符号的输入端都有小圆圈.
③ 每一种门的标准符号和替代符号都代表相同的实际电路.

有效逻辑电平

输入或输出线上无小圆圈 ⟶ 高电平有效
输入或输出线上有小圆圈 ⟶ 低电平有效