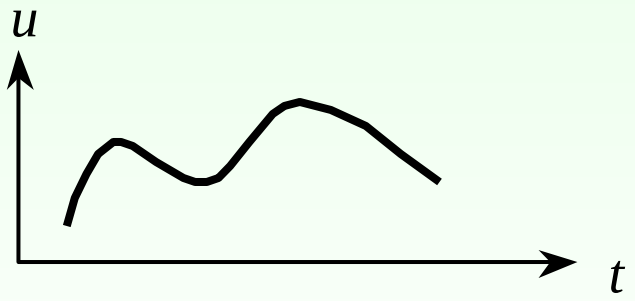
第11章　组合逻辑电路

数字电路概述

数字信号与数字电路

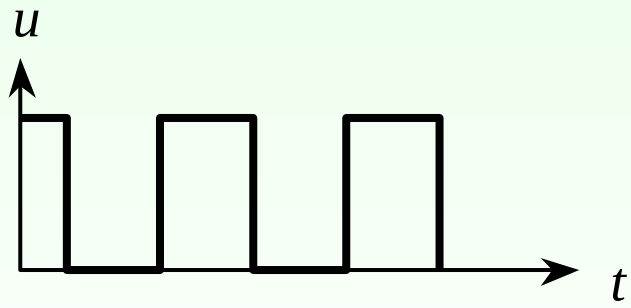
模拟信号：在时间上和数值上连续的信号。



模拟信号波形

对模拟信号进行传输、处理的电子线路称为模拟电路。

数字信号：在时间上和数值上不连续的（即离散的）信号。



数字信号波形

对数字信号进行传输、处理的电子线路称为数字电路。

数字电路的特点

（1）工作信号是二进制的数字信号，在时间上和数值上是离散的（不连续），反映在电路上就是低电平和高电平两种状态（即0和1两个逻辑值）。

（2）在数字电路中，研究的主要问题是电路的逻辑功能，即输入信号的状态和输出信号的状态之间的逻辑关系。

（3）对组成数字电路的元器件的精度要求不高，只要在工作时能够可靠地区分0和1两种状态即可。

数制与编码

1、数制

（1）进位制：表示数时，仅用一位数码往往不够用，必须用进位计数的方法组成多位数码。多位数码每一位的构成以及从低位到高位的进位规则称为进位计数制，简称进位制。

（2）基 数：进位制的基数，就是在该进位制中可能用到的数码个数。

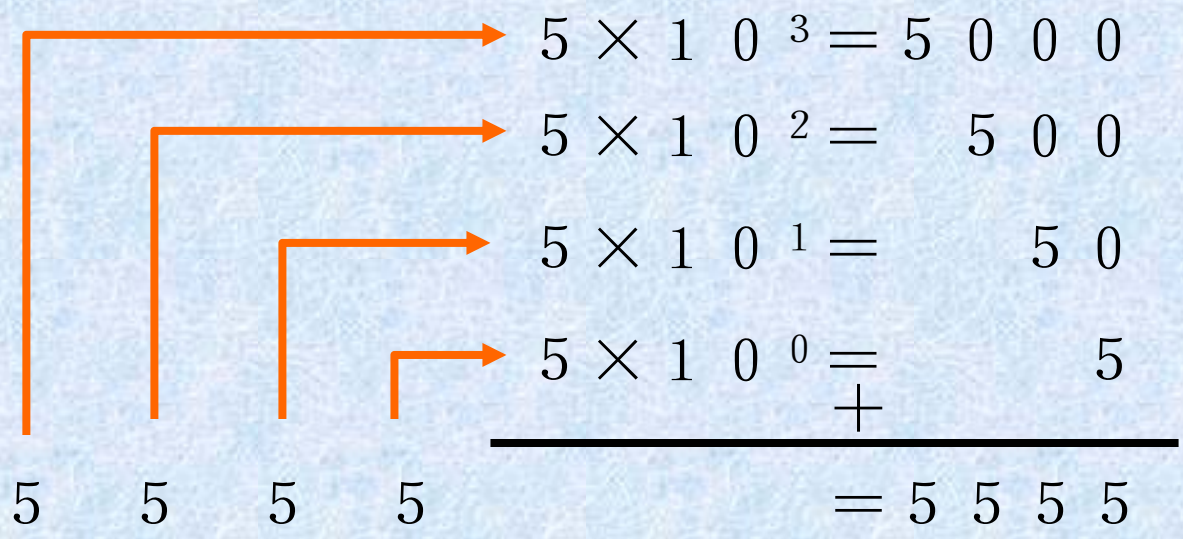
（3） 位 权（位的权数）：在某一进位制的数中，每一位的大小都对应着该位上的数码乘上一个固定的数，这个固定的数就是这一位的权数。权数是一个幂。

(1)、十进制

数码为：0～9；基数是10。

运算规律：逢十进一，即：9＋1＝10。

十进制数的权展开式：



10^3、10^2、10^1、10^0称为十进制的权。各数位的权是10的幂。

任意一个十进制数都可以表示为各个数位上的数码与其对应的权的乘积之和，称权展开式。

同样的数码在不同的数位上代表的数值不同。

即：$(5555)_{10}=5\times10^3+5\times10^2+5\times10^1+5\times10^0$

又如：$(209.04)_{10}=2\times10^2+0\times10^1+9\times10^0+0\times10^{-1}+4\times10^{-2}$

(2)、二进制

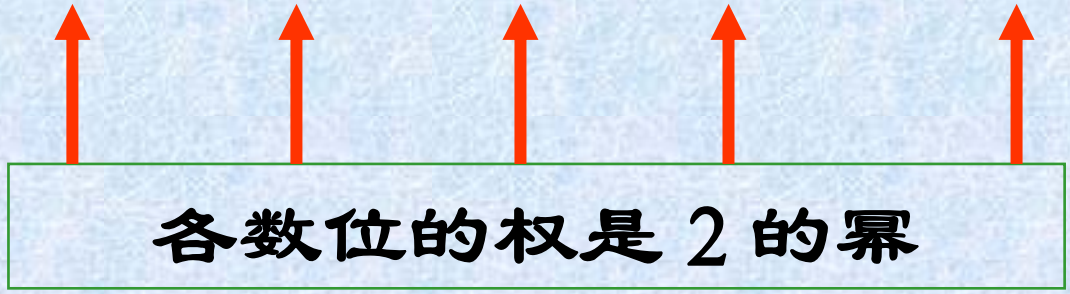
数码为：0、1；基数是2。

运算规律：逢二进一，即：1＋1＝10。

二进制数的权展开式：

如：$(101.01)_2 = 1\times2^2 + 0\times2^1 + 1\times2^0 + 0\times2^{-1} + 1\times2^{-2}$

$= (5.25)_{10}$



二进制数只有0和1两个数码，它的每一位都可以用电子元件来实现，且运算规则简单，相应的运算电路也容易实现。

运算规则

加法规则：0+0=0，0+1=1，1+0=1，1+1=10

乘法规则：0.0=0， 0.1=0 ， 1.0=0，1.1=1

(3)、八进制

数码为：0～7；基数是8。

运算规律：逢八进一，即：7＋1＝10。

八进制数的权展开式：

如：$(207.04)_{10}= 2\times8^{2}+0\times8^{1}+7\times8^{0}+0\times8^{-1}+4\times8^{-2}$

$=(135.0625)_{10}$

各数位的权是8的幂

(4)、十六进制

数码为：0～9、A～F；基数是16。

运算规律：逢十六进一，即：F＋1＝10。

十六进制数的权展开式：

如：$(D8.A)_{2}= 13\times16^{1}+8\times16^{0}+10\times16^{-1}=(216.625)_{10}$

各数位的权是16的幂

结论

①一般地，N进制需要用到N个数码，基数是N；运算规律为逢N进一。

②如果一个N进制数M包含 n 位整数和m位小数，即

$(a_{n-1}\,a_{n-2}\ldots a_1\,a_0\,.\,a_{-1}\,a_{-2}\ldots a_{-m})_2$

则该数的权展开式为：

$(M)_2 = a_{n-1}\times N^{n-1} + a_{n-2}\times N^{n-2} + \ldots + a_1\times N^1 + a_0\times N^0 + a_{-1}\times N^{-1} + a_{-2}\times N^{-2} + \ldots + a_{-m}\times N^{-m}$

③由权展开式很容易将一个N进制数转换为十进制数。

几种进制数之间的对应关系

十进制数	二进制数	八进制数	十六进制数
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

数制转换

将N进制数按权展开，即可以转换为十进制数。

(1)、二进制数与八进制数的相互转换

（1）二进制数转换为八进制数：将二进制数由小数点开始，整数部分向左，小数部分向右，每3位分成一组，不够3位补零，则每组二进制数便是一位八进制数。

0 0 1 | 1 0 1 | 0 1 0 . 0 1 0 ＝ $(152.2)_8$

（2）八进制数转换为二进制数：将每位八进制数用3位二进制数表示。

$(374.26)_8$ = 011 111 100 . 010 110

(2)、二进制数与十六进制数的相互转换

二进制数与十六进制数的相互转换，按照每4位二进制数对应于一位十六进制数进行转换。

0 0 0 1 | 1 1 0 1 | 0 1 0 0 . 0 1 1 0 = $(1D4.6)_{16}$

$(AF4.76)_{16}$ = 1010 1111 0100 . 0111 0110

(3)、十进制数转换为二进制数

采用的方法 — 基数连除、连乘法

原理：将整数部分和小数部分分别进行转换。

整数部分采用基数连除法，小数部分采用基数连乘法。转换后再合并。

整数部分采用基数连除法，先得到的余数为低位，后得到的余数为高位

2	44	余数	低位
2	22	····· $0=K_0$	
2	11	····· $0=K_1$	
2	5	····· $1=K_2$	
2	2	····· $1=K_3$	
2	1	····· $0=K_4$	
	0	····· $1=K_5$	高位

小数部分采用基数连乘法，先得到的整数为高位，后得到的整数为低位

0.375			
× 2		整数	高位
0.750	·····	$0=K_{-1}$	
0.750			
× 2			
1.500	·····	$1=K_{-2}$	
0.500			
× 2			
1.000	·····	$1=K_{-3}$	低位

所以：$(44.375)_{10}=(101100.011)_2$

采用基数连除、连乘法，可将十进制数转换为任意的N进制数。

2、编码

数字系统只能识别0和1，怎样才能表示更多的数码、符号、字母呢？用编码可以解决此问题。

用一定位数的二进制数来表示十进制数码、字母、符号等信息称为编码。

用以表示十进制数码、字母、符号等信息的一定位数的二进制数称为代码。

二-十进制代码：用4位二进制数$b_3b_2b_1b_0$来表示十进制数中的 0 ~ 9 十个数码。简称BCD码。

用四位自然二进制码中的前十个码字来表示十进制数码，因各位的权值依次为8、4、2、1，故称8421 BCD码。

2421码的权值依次为2、4、2、1；余3码由8421码加0011得到；格雷码是一种循环码，其特点是任何相邻的两个码字，仅有一位代码不同，其它位相同。

常用BCD码

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

逻辑门电路

逻辑门电路：用以实现基本和常用逻辑运算的电子电路。简称门电路。

基本和常用门电路有与门、或门、非门（反相器）、与非门、或非门、与或非门和异或门等。

逻辑0和1： 电子电路中用高、低电平来表示。

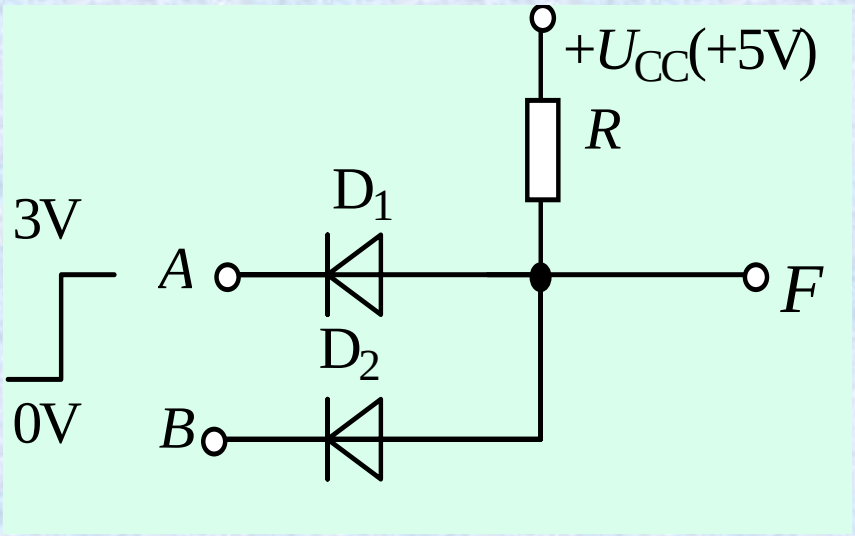
获得高、低电平的基本方法：利用半导体开关元件的导通、截止（即开、关）两种工作状态。

基本逻辑关系及其门电路

1、与逻辑和与门电路

当决定某事件的全部条件同时具备时，结果才会发生，这种因果关系叫做与逻辑。

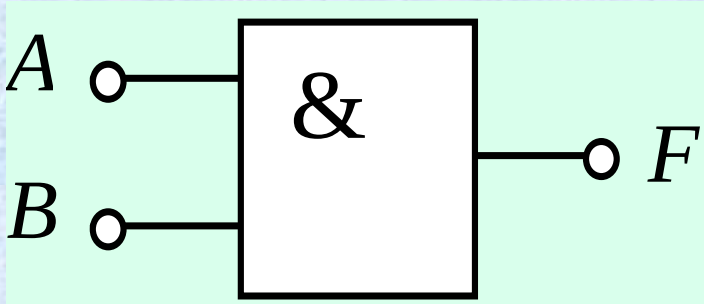
实现与逻辑关系的电路称为与门。



A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

u_A	u_B	u_F	D_1	D_2
0V	0V	0V	导通	导通
0V	3V	0V	导通	截止
3V	0V	0V	截止	导通
3V	3V	3V	截止	截止

F=AB



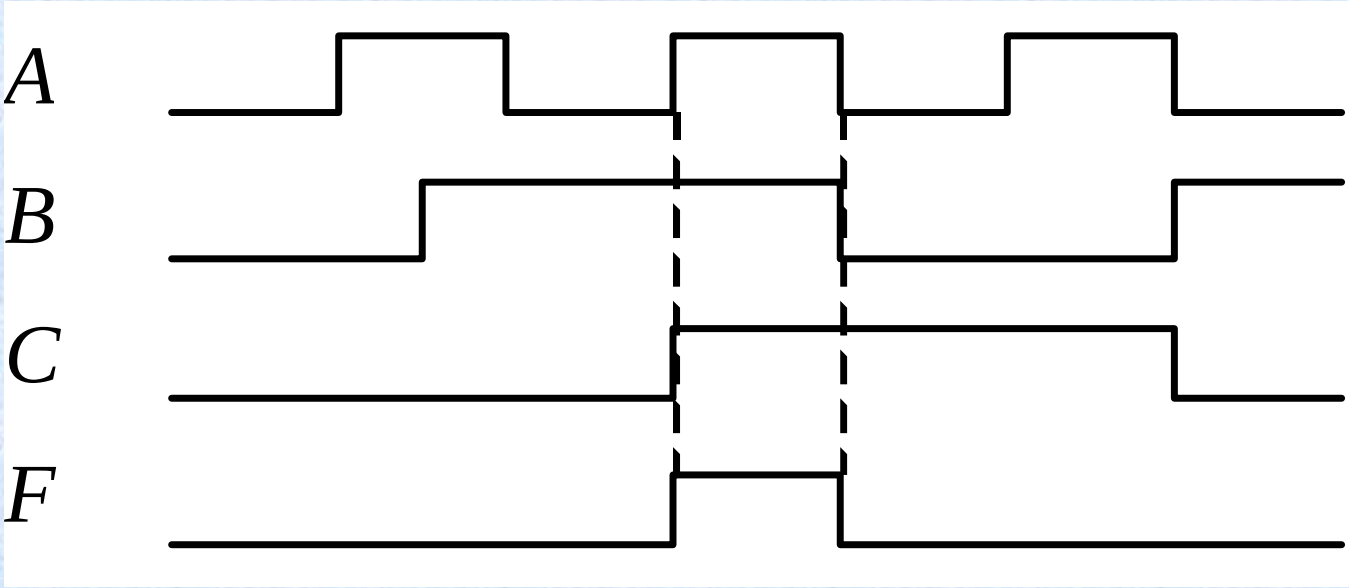
与门的逻辑功能可概括为：输入有0，输出为0；输入全1，输出为1。

F=AB

逻辑与（逻辑乘）的运算规则为

$0\cdot0=0$　$0\cdot1=0$　$1\cdot0=0$　$1\cdot1=1$

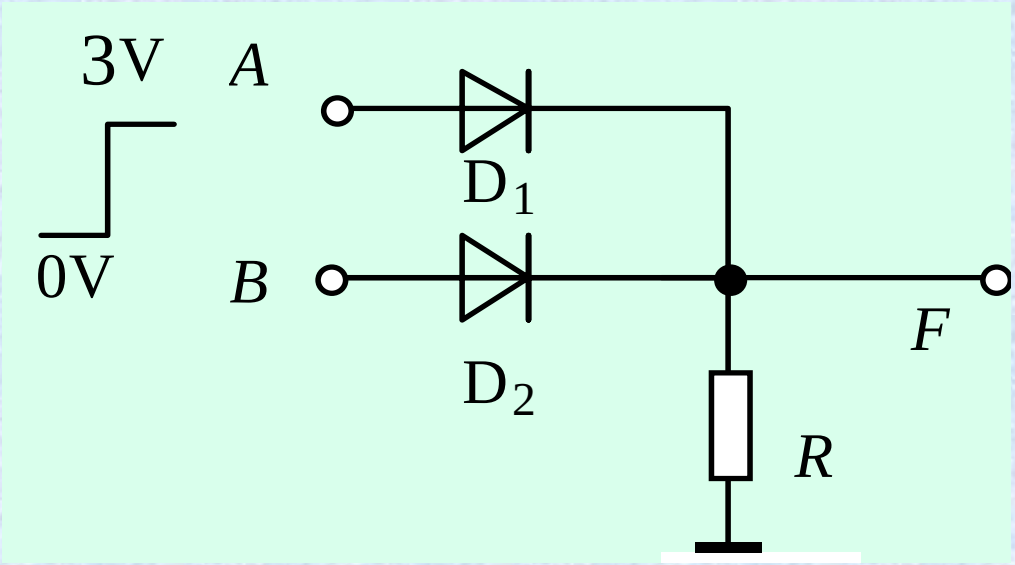
与门的输入端可以有多个。下图为一个三输入与门电路的输入信号A、B、C和输出信号F的波形图。



2、或逻辑和或门电路

在决定某事件的条件中，只要任一条件具备，事件就会发生，这种因果关系叫做或逻辑。

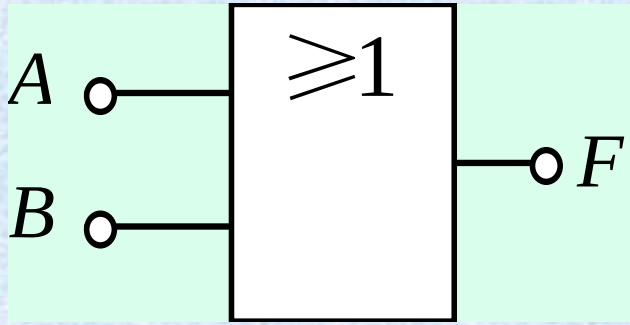
实现或逻辑关系的电路称为或门。



A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

u_A	u_B	u_F	D_1	D_2
0V	0V	0V	截止	截止
0V	3V	3V	截止	导通
3V	0V	3V	导通	截止
3V	3V	3V	导通	导通

F=A+B



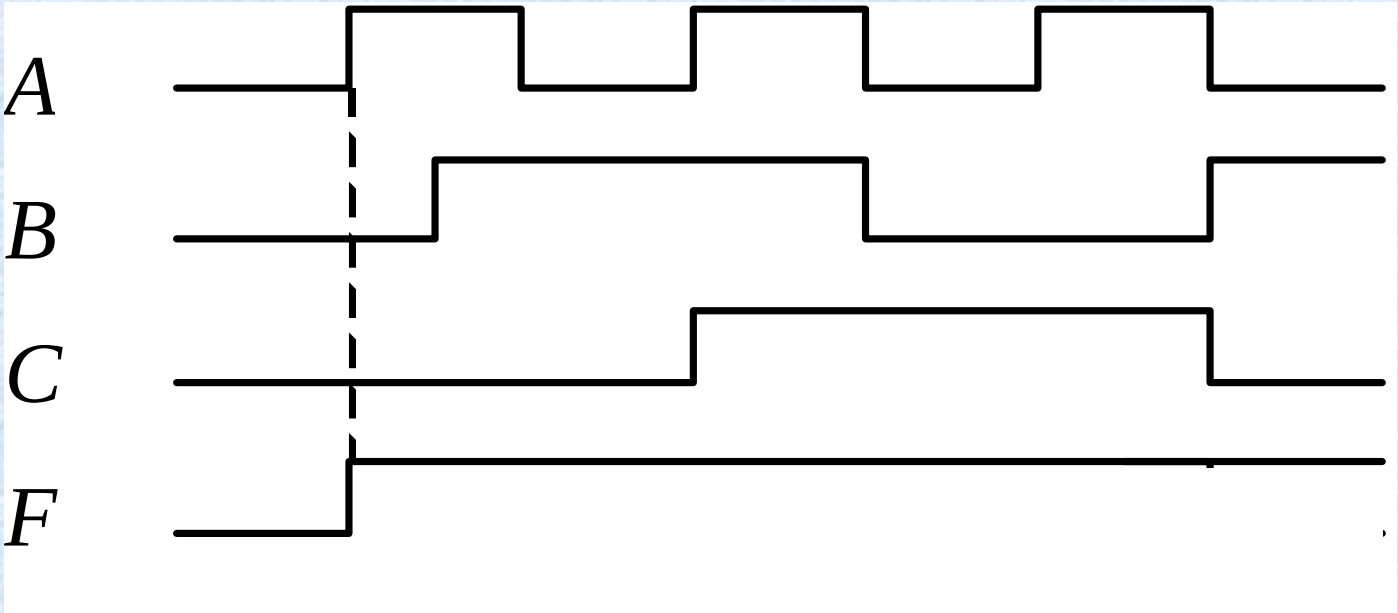
或门的逻辑功能可概括为：输入有1，输出为1；输入全0，输出为0。

F=A+B

逻辑或（逻辑加）的运算规则为

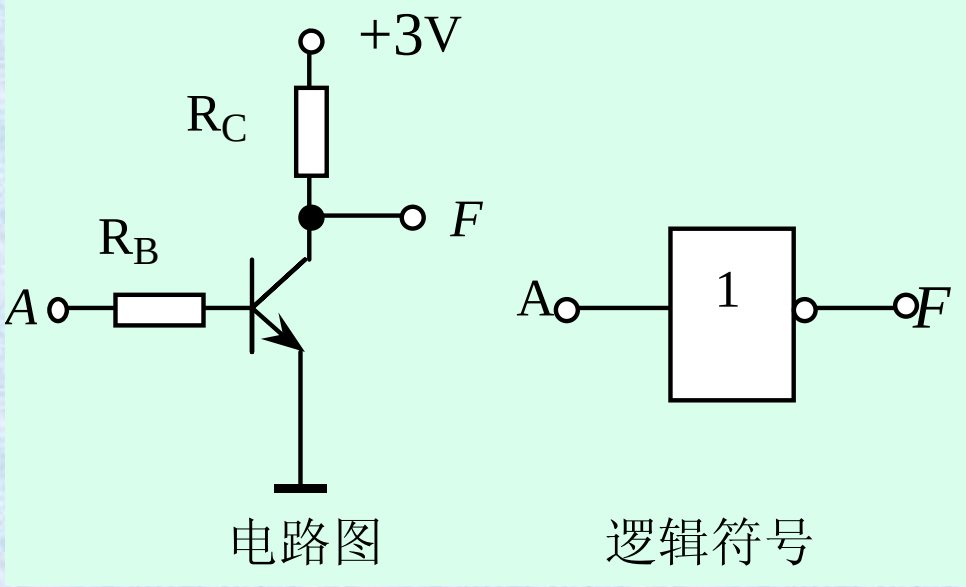
$0+0=0$　　$0+1=0$　　$1+0=0$　　$1+1=1$

或门的输入端也可以有多个。下图为一个三输入或门电路的输入信号A、B、C和输出信号F的波形图。



3、非逻辑和非门电路

决定某事件的条件只有一个，当条件出现时事件不发生，而条件不出现时，事件发生，这种因果关系叫做非逻辑。
实现非逻辑关系的电路称为非门，也称反相器。



A	F
0	1
1	0

$$F = \overline{A}$$

输入A为高电平1(3V)时，三极管饱和导通，输出F为低电平0（0V)；输入A为低电平0(0V)时，三极管截止，输出F为高电平1（3V)。

逻辑非（逻辑反）的运算规则为：

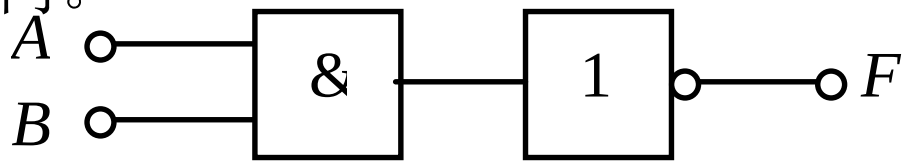
$$\overline{0} = 1 \qquad \overline{1} = 0$$

4、复合门电路

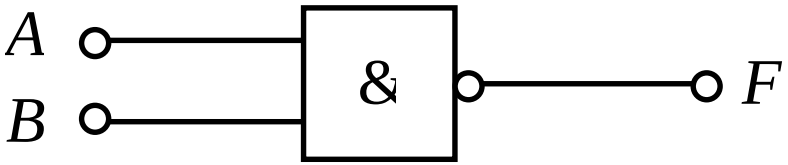
将与门、或门、非门组合起来，可以构成多种复合门电路。

(1) 与非门

由与门和非门构成与非门。



(a)　与非门的构成



(b)　逻辑符号

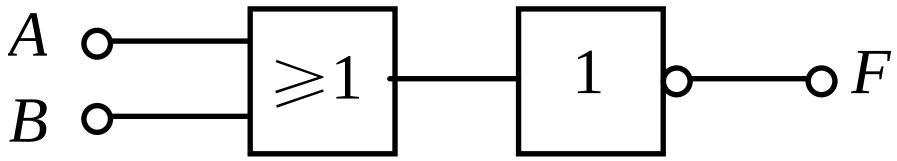
A	B	F
0	0	1
0	1	1
1	0	1
1	1	0

$$F = \overline{AB}$$

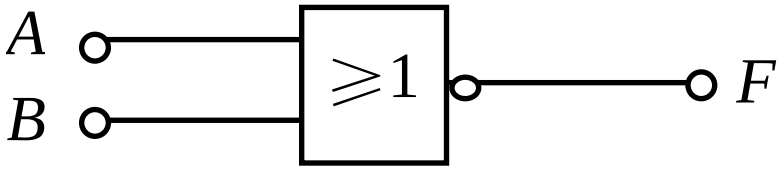
与非门的逻辑功能可概括为：输入有0，输出为1；输入全1，输出为0。

(2) 或非门

由或门和非门构成或非门。



(a) 或非门的构成



(b) 逻辑符号

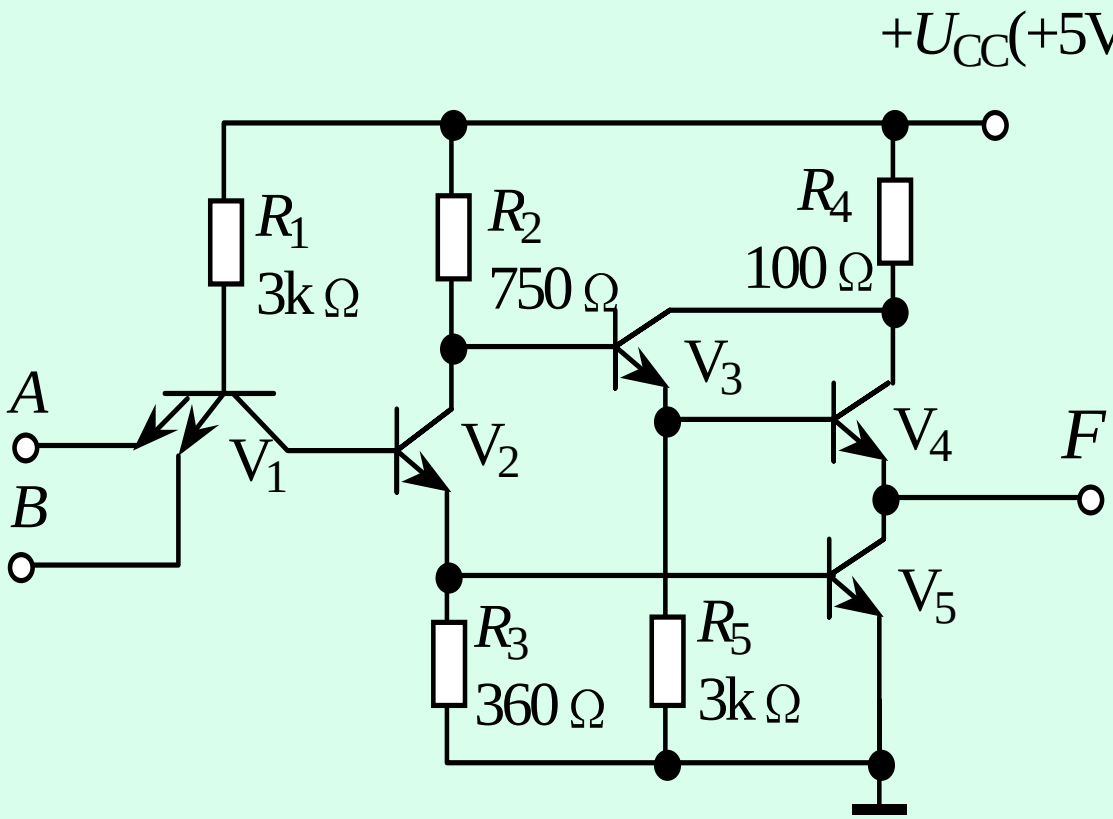
A	B	F
0	0	1
0	1	0
1	0	0
1	1	0

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

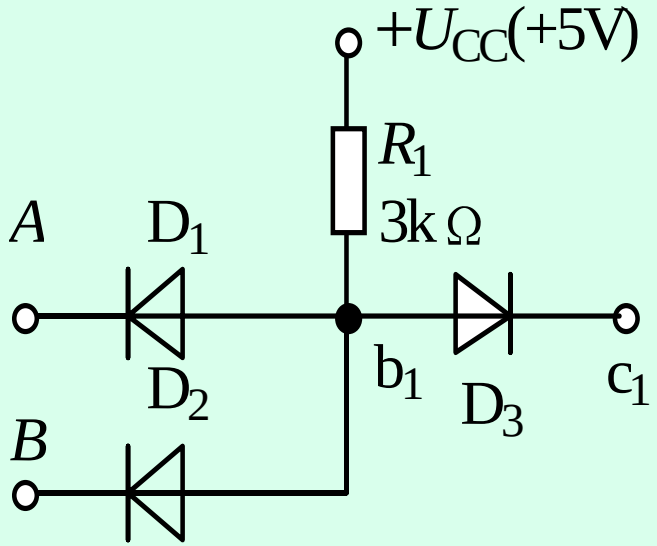
或非门的逻辑功能可概括为：输入有1，输出为0；输入全0，输出为1。

集成门电路

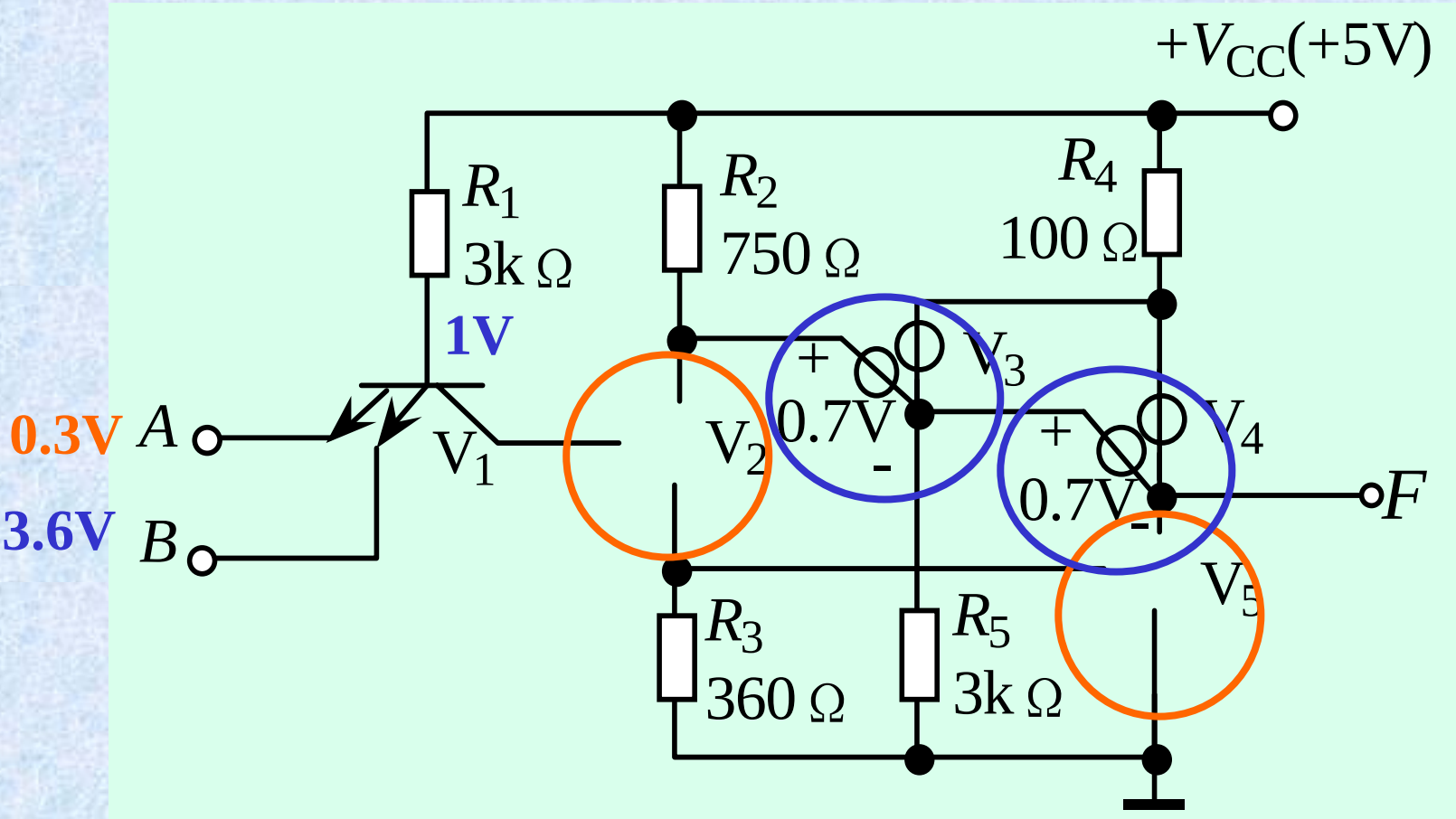
1、TTL与非门



TTL 与非门电路



V_1 的等效电路



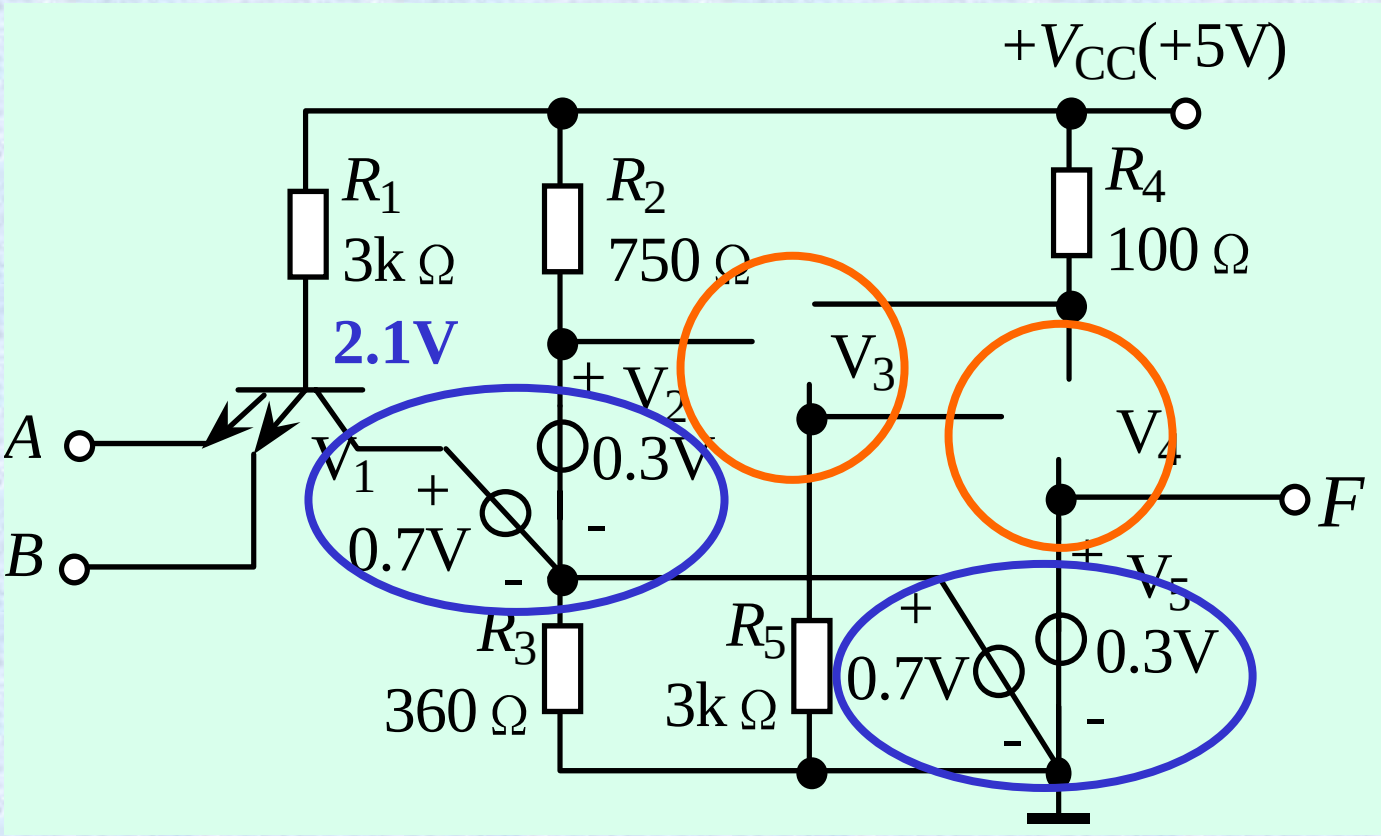
①输入信号不全为1：如u_A=0.3V， u_B=3.6V

则u_{B1}=0.3+0.7=1V，V_2、V_5截止，V_3、V_4导通

忽略i_{B3}，输出端的电位为： $u_F \approx 5-0.7-0.7=3.6V$

输出F为高电平

1



②输入信号全为1：如$u_A=u_B=3.6V$

则$u_{B1}=2.1V$，V_2、V_5导通，V_3、V_4截止

输出端的电位为：　$u_F=U_{CES}=0.3V$

输出F为低电平0。

功能表

u_A	u_B	u_F
0.3V	0.3V	3.6V
0.3V	3.6V	3.6V
3.6V	0.3V	3.6V
3.6V	3.6V	0.3V

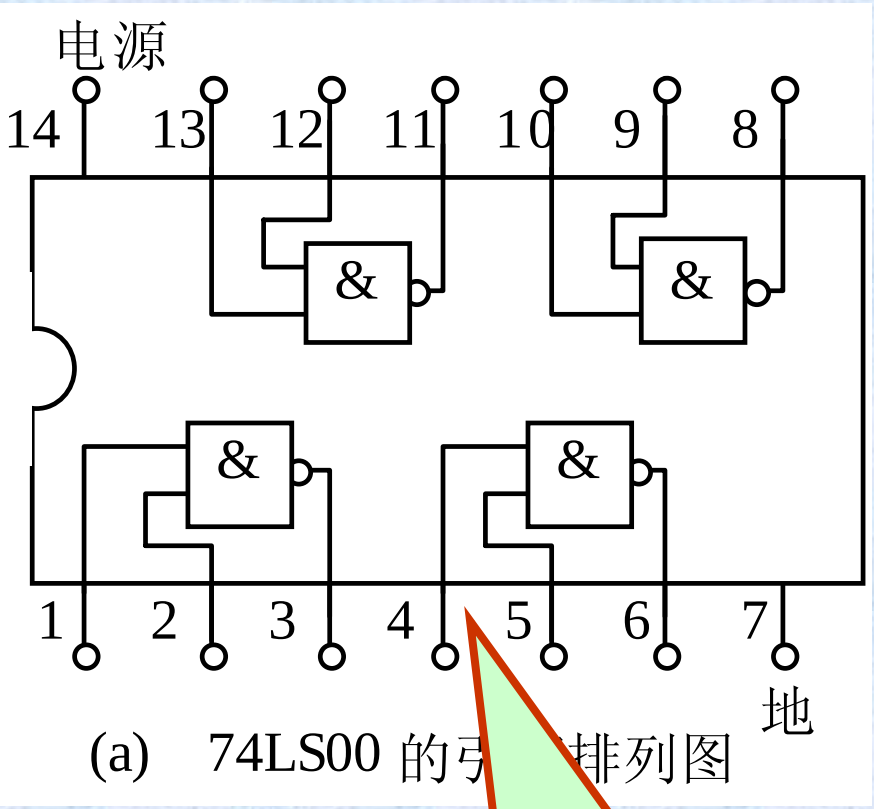
真值表

A	B	F
0	0	1
0	1	1
1	0	1
1	1	0

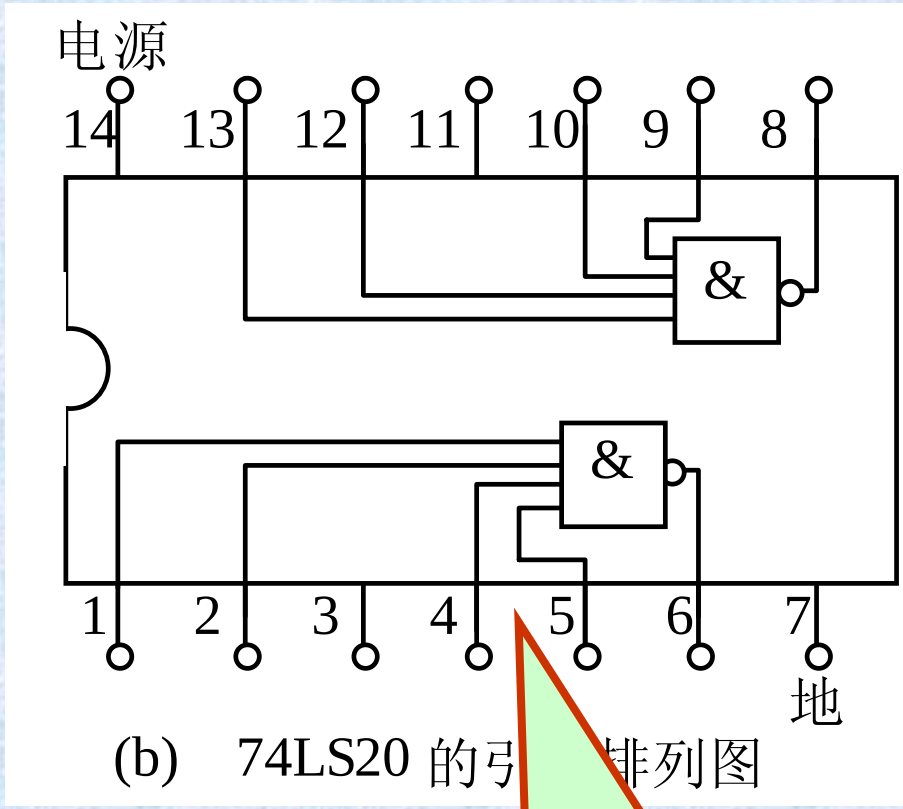
输入有0，输出为1；输入全1，输出为0。

逻辑表达式：

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



(a) 74LS00 的引 排列图



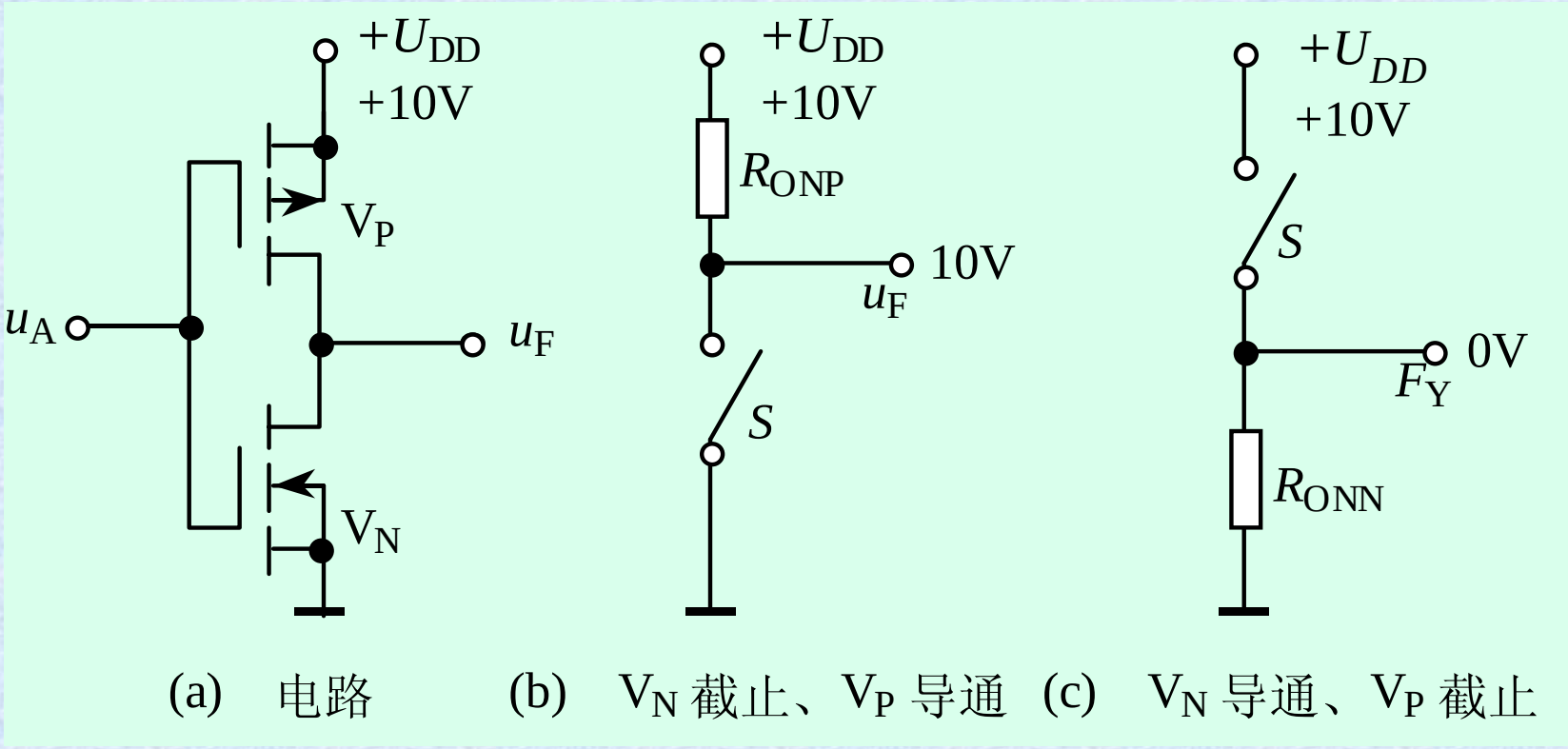
(b) 74LS20 的引 排列图

内含4个两输入端的与非门，电源线及地线公用。

内含两个4输入端的与非门，电源线及地线公用。

2、CMOS门电路

CMOS非门



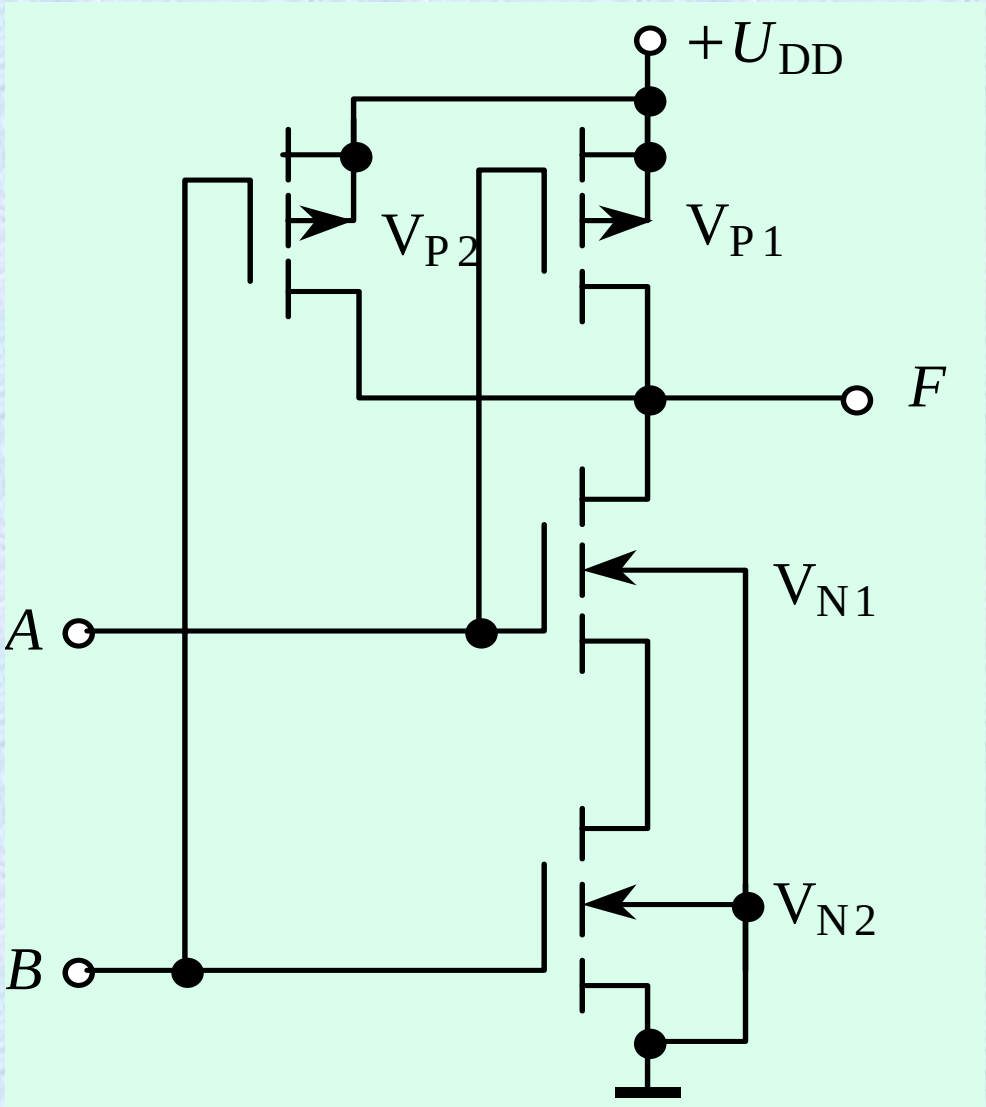
(a) 电路　(b) V_N 截止、V_P 导通　(c) V_N 导通、V_P 截止

（1）u_A＝0V时，V_N截止，V_P导通。输出电压u_F＝V_{DD}＝10V。

（2）u_A＝10V时，V_N导通，V_P截止。输出电压u_F＝0V。

$$F = \overline{A}$$

CMOS与非门

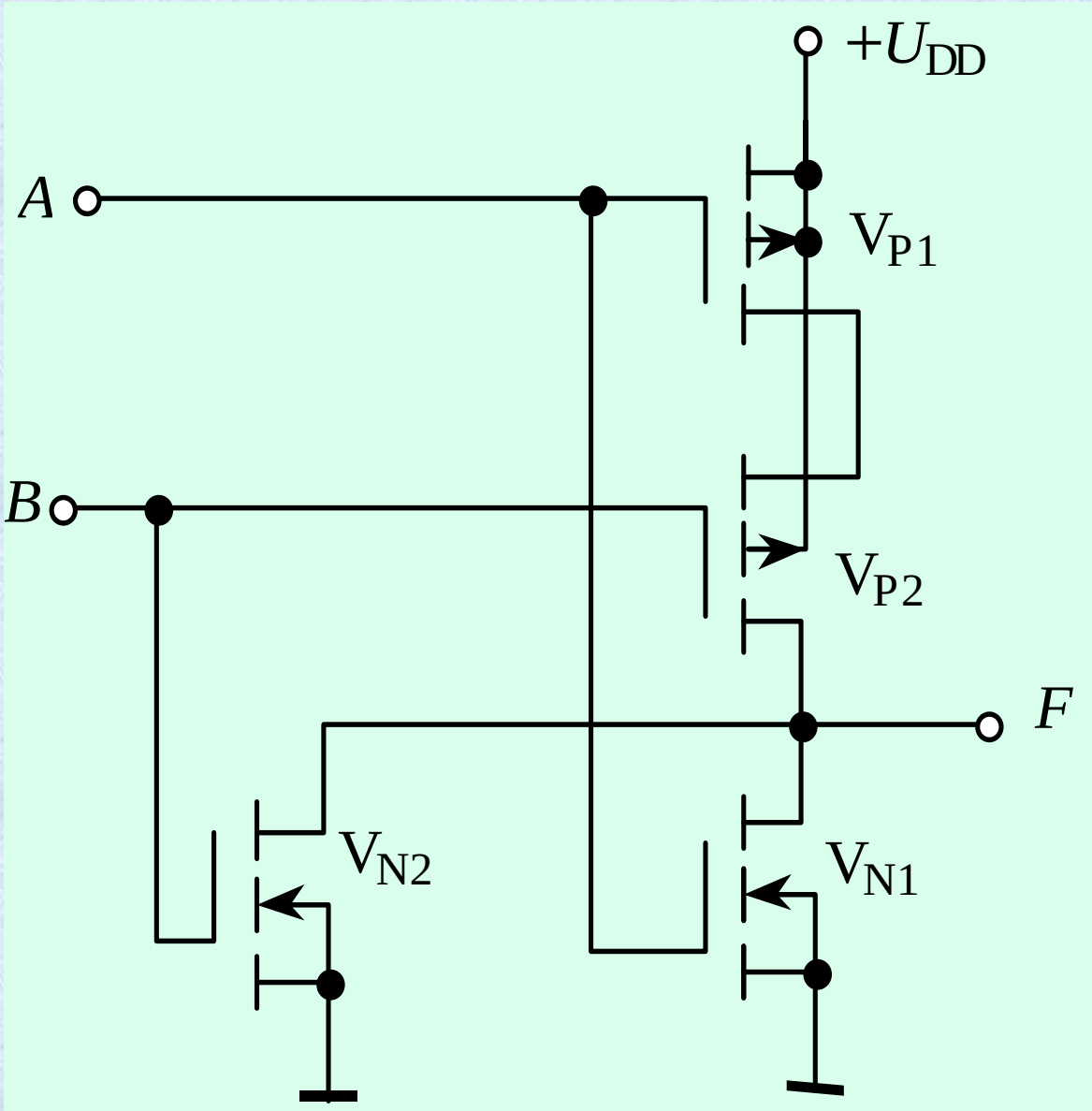


①A、B当中有一个或全为低电平0时，V_{N1}、V_{N2}中有一个或全部截止，V_{P1}、V_{P2}中有一个或全部导通，输出F为高电平1。

②只有当输入A、B全为高电平1时，V_{N1}和V_{N2}才会都导通，V_{P1}和V_{P2}才会都截止，输出F才会为低电平0。

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

CMOS或非门



①只要输入A、B当中有一个或全为高电平1，V_{P1}、V_{P2}中有一个或全部截止，V_{N1}、V_{N2}中有一个或全部导通，输出F为低电平0。

②只有当A、B全为低电平0时，V_{P1}和V_{P2}才会都导通，V_{N1}和V_{N2}才会都截止，输出F才会为高电平1。

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

逻辑函数及其化简

将门电路按照一定的规律连接起来，可以组成具有各种逻辑功能的逻辑电路。分析和设计逻辑电路的数学工具是逻辑代数（又叫布尔代数或开关代数）。逻辑代数具有3种基本运算：与运算（逻辑乘）、或运算（逻辑加）和非运算（逻辑非）。

逻辑代数的公式和定理

（1）常量之间的关系

与运算：$0 \cdot 0 = 0$　$0 \cdot 1 = 0$　$1 \cdot 0 = 0$　$1 \cdot 1 = 1$

或运算：$0 + 0 = 0$　$0 + 1 = 1$　$1 + 0 = 1$　$1 + 1 = 1$

非运算：$\overline{1} = 0$　$\overline{0} = 1$

（2）基本运算

与运算：$A \cdot 0 = 0$　$A \cdot 1 = A$　$A \cdot A = A$　$A \cdot \overline{A} = 0$

或运算：$A + 0 = A$　$A + 1 = 1$　$A + A = A$　$A + \overline{A} = 1$

非运算：$\overline{\overline{A}} = A$

分别令A=0及A=1代入这些公式，即可证明它们的正确性。

（3）基本定理

利用真值表很容易证明这些公式的正确性。如证明

$A \cdot B = B \cdot A$

交换律：
$$\begin{cases} A \cdot B = B \cdot A \\ A + B = B + A \end{cases}$$

结合律：
$$\begin{cases} (A \cdot B) \cdot C = A \cdot (B \cdot C) \\ (A + B) + C = A + (B + C) \end{cases}$$

分配律：
$$\begin{cases} A \cdot (B + C) = A \cdot B + A \cdot C \\ A + B \cdot C = (A + B) \cdot (A + C) \end{cases}$$

反演律（摩根定律）：
$$\begin{cases} \overline{A \cdot .B} = \overline{A} + \overline{B} \\ \overline{A + B} = \overline{A} \cdot \overline{B} \end{cases}$$

A	B	A.B	B.A
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	1

证明分配率：A+BA=(A+B)(A+C)

证明：

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

$=A+AB+AC+BC$

$=A(1+B+C)+BC$

$=A+BC$

分配率
$A(B+C)=AB+AC$

$AA=A$

分配率
$A(B+C)=AB+AC$

$A+1=1$

吸收律：

$$\begin{cases} A \cdot B + A \cdot \overline{B} = A \\ (A + B) \cdot (A + \overline{B}) = A \end{cases}$$

$$\begin{cases} A + A \cdot B = A \\ A \cdot (A + B) = A \end{cases} \qquad \begin{cases} A \cdot (\overline{A} + B) = A \cdot B \\ A + \overline{A} \cdot B = A + B \end{cases}$$

证明：$A+\overline{A}B=(A+\overline{A})(A+B)$

$=1\cdot(A+B)$

$=A+B$

分配率

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

$A+\overline{A}=1$

$A\cdot 1=1$

逻辑函数的表示方法

逻辑函数有5种表示形式：真值表、逻辑表达式、卡诺图、逻辑图和波形图。只要知道其中一种表示形式，就可转换为其它几种表示形式。

1、真值表

真值表：是由变量的所有可能取值组合及其对应的函数值所构成的表格。

真值表列写方法：每一个变量均有0、1两种取值，n个变量共有2^i种不同的取值，将这2^i种不同的取值按顺序（一般按二进制递增规律）排列起来，同时在相应位置上填入函数的值，便可得到逻辑函数的真值表。

例如：当A、B取值相同时，函数值为0；否则，函数取值为1。

A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

2、逻辑表达式

逻辑表达式：是由逻辑变量和与、或、非3种运算符连接起来所构成的式子。

表达式列写方法：将那些使函数值为1的各个状态表示成全部变量（值为1的表示成原变量，值为0的表示成反变量）的与项（例如A=0、B=1时函数F的值为1，则对应的与项为$\overline{A}B$）以后相加，即得到函数的与或表达式。

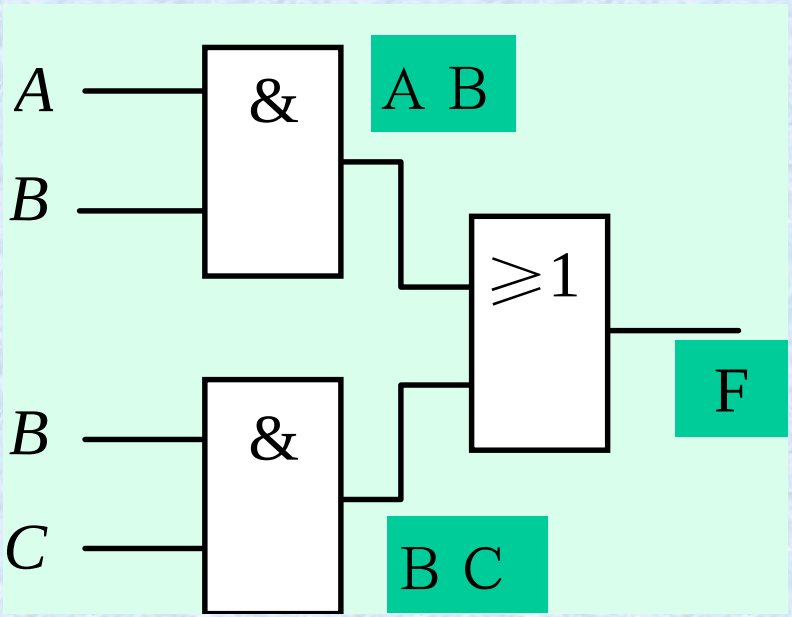
A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

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

3、逻辑图

逻辑图：是由表示逻辑运算的逻辑符号所构成的图形。

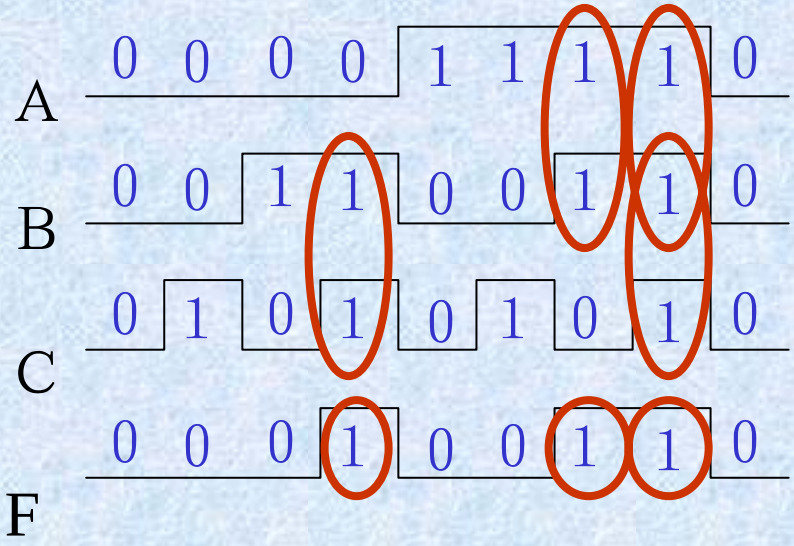
$F = A\,B + B\,C$



4、波形图

波形图：是由输入变量的所有可能取值组合的高、低电平及其对应的输出函数值的高、低电平所构成的图形。

$F = A\,B + B\,C$



逻辑函数的化简

逻辑函数化简的意义：逻辑表达式越简单，实现它的电路越简单，电路工作越稳定可靠。

利用公式 $A+\overline{A}=1$，将两项合并为一项，并消去一个变量。

运用分配律

$$Y_1 = ABC + \overline{A}BC + B\overline{C} = (A+\overline{A})BC + B\overline{C}$$

$$= BC + B\overline{C} = B(C+\overline{C}) = B$$

运用分配律

$$Y_2 = ABC + A\overline{B} + A\overline{C} = ABC + A(\overline{B}+\overline{C})$$

$$= ABC + A\overline{BC} = A(BC + \overline{BC}) = A$$

运用摩根定律

若两个乘积项中分别包含同一个因子的原变量和反变量，而其他因子都相同时，则这两项可以合并成一项，并消去互为反变量的因子。

利用公式 A＋A B＝A，消去多余的项。

$$Y_1 = \overline{A}B + \overline{A}BCD(E+F) = \overline{A}B$$

运用摩根定律

$$Y_2 = A + \overline{\overline{B} + \overline{CD}} + \overline{\overline{AD}\overline{B}} = A + BCD + AD + B$$

$$= (A + AD) + (B + BCD) = A + B$$

如果乘积项是另外一个乘积项的因子，则这另外一个乘积项是多余的。

利用公式 A＋$\overline{\text{A}}$ B＝A＋B，消去多余的变量。

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

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

$$= AB + \overline{AB}C$$

$$= AB + C$$

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

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

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

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

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

如果一个乘积项的反是另一个乘积项的因子，则这个因子是多余的。

利用公式A＝A（B＋$\overline{B}$），为某一项配上其所缺的变量，以便用其它方法进行化简。

$$
\begin{aligned}
Y &= A\overline{B} + B\overline{C} + \overline{B}C + \overline{A}B \\
&= A\overline{B} + B\overline{C} + (A + \overline{A})\overline{B}C + \overline{A}B(C + \overline{C}) \\
&= A\overline{B} + B\overline{C} + A\overline{B}C + \overline{A}\overline{B}C + \overline{A}BC + \overline{A}B\overline{C} \\
&= A\overline{B}(1 + C) + B\overline{C}(1 + \overline{A}) + \overline{A}C(\overline{B} + B) \\
&= A\overline{B} + B\overline{C} + \overline{A}C
\end{aligned}
$$

利用公式A＋A＝A，为某项配上其所能合并的项。

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

组合逻辑电路的分析与设计

组合逻辑电路：输出仅由输入决定，与电路当前状态无关；电路结构中无反馈环路（无记忆）。

组合逻辑电路的分析

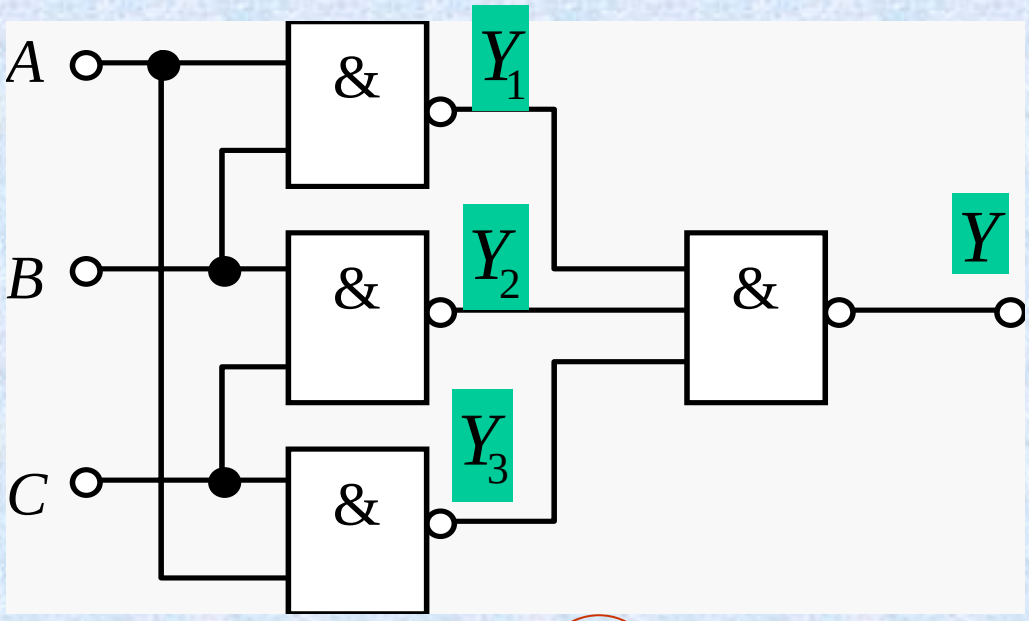
逻辑图

1 从输入到输出逐级写出

逻辑表达式

化简 2

最简与或表达式



1

$$Y_1 = \overline{AB}$$

$$Y_2 = \overline{BC}$$

$$Y_3 = \overline{CA}$$

$$Y = \overline{Y_1 Y_2 Y_3} = \overline{\overline{AB}\ \overline{BC}\ \overline{AC}}$$

2

$$Y = AB + BC + CA$$

最简与或表达式

3

真值表

4

电路的逻辑功能

$$Y = AB + BC + CA$$

3

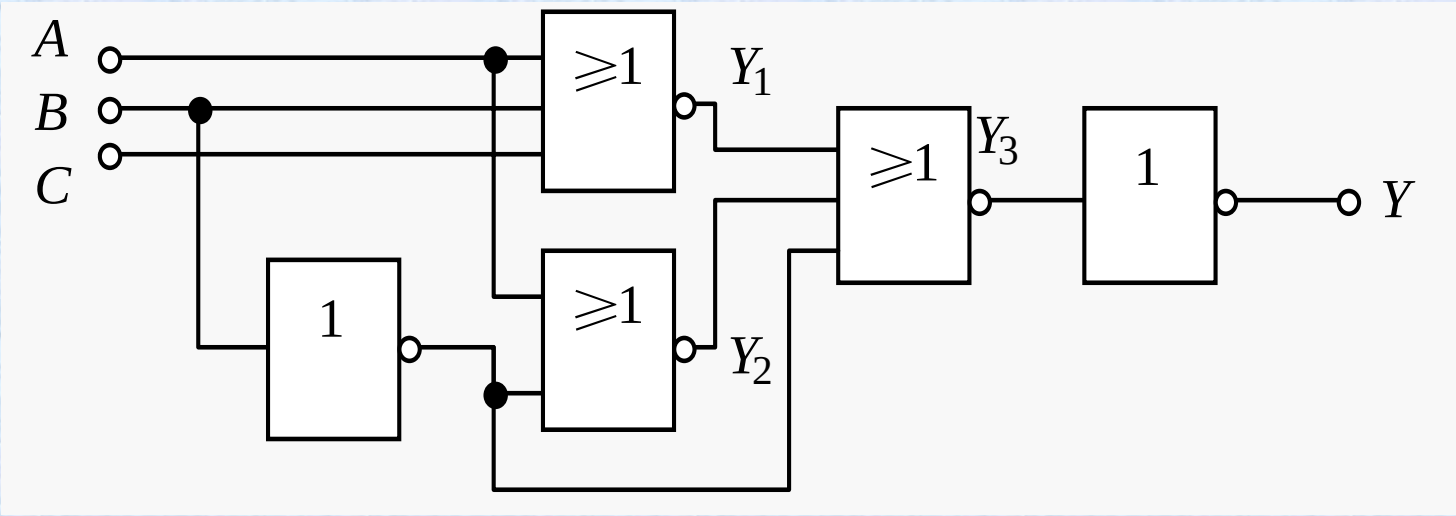
A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

4

当输入A、B、C中有2个或3个为1时，输出Y为1，否则输出Y为0。所以这个电路实际上是一种3人表决用的组合电路：只要有2票或3票同意，表决就通过。

例：

逻辑图



逻辑表达式

$$\left.\begin{aligned} Y_1 &= \overline{A+B+C} \\ Y_2 &= \overline{\overline{A}+\overline{B}} \\ Y_3 &= \overline{X+Y+\overline{B}} \end{aligned}\right\} Y=\overline{Y}_3=Y_1+Y_2+\overline{B}=\overline{A+B+C}+\overline{\overline{A}+\overline{B}}+\overline{B}$$

最简与或表达式

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

真值表

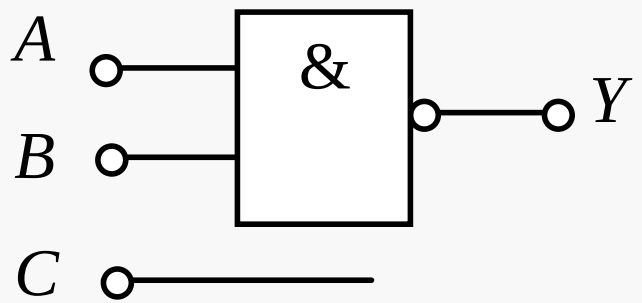
A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

电路的逻辑功能

电路的输出Y只与输入A、B有关，而与输入C无关。Y和A、B的逻辑关系为：A、B中只要一个为0，Y=1；A、B全为1时，Y=0。所以Y和A、B的逻辑关系为与非运算的关系。

用与非门实现

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



组合逻辑电路的设计

电路功能描述

穷举法

1

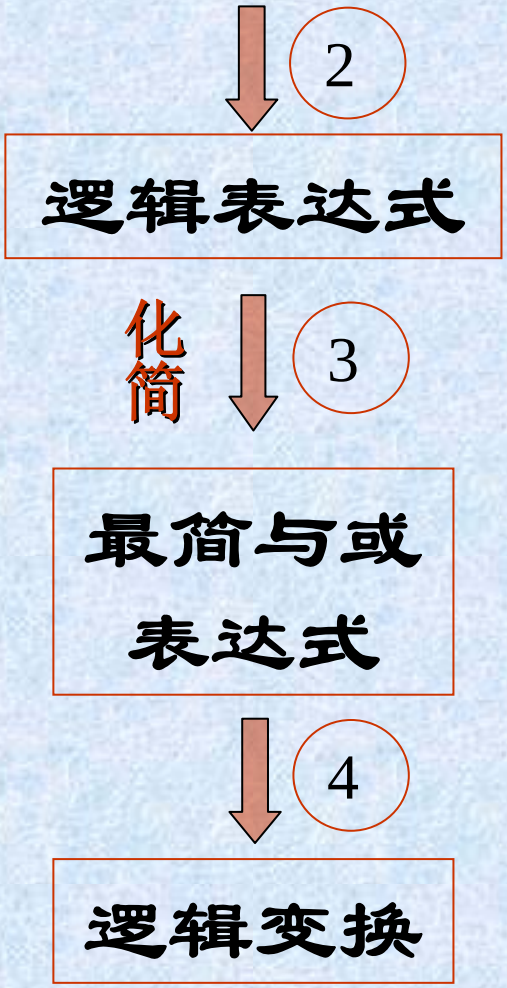
真值表

例：用与非门设计一个交通报警控制电路。交通信号灯有红、绿、黄3种，3种灯分别单独工作或黄、绿灯同时工作时属正常情况，其他情况均属故障，出现故障时输出报警信号。

1

设红、绿、黄灯分别用*A*、*B*、*C*表示，灯亮时其值为1，灯灭时其值为0；输出报警信号用*F*表示，灯正常工作时其值为0，灯出现故障时其值为1。根据逻辑要求列出真值表。

A	*B*	*C*	*F*	*A*	*B*	*C*	*F*
0	0	0	1	1	0	0	0
0	0	1	0	1	0	1	1
0	1	0	0	1	1	0	1
0	1	1	0	1	1	1	1



2

$$F = \overline{A}\overline{B}\overline{C} + A\overline{B}C + AB\overline{C} + ABC$$

3

$$\begin{aligned} F &= \overline{A}\overline{B}\overline{C} + ABC + AB\overline{C} + ABC + A\overline{B}C \\ &= \overline{A}\overline{B}\overline{C} + AB(C + \overline{C}) + AC(B + \overline{B}) \\ &= \overline{A}\overline{B}\overline{C} + AB + AC \end{aligned}$$

4

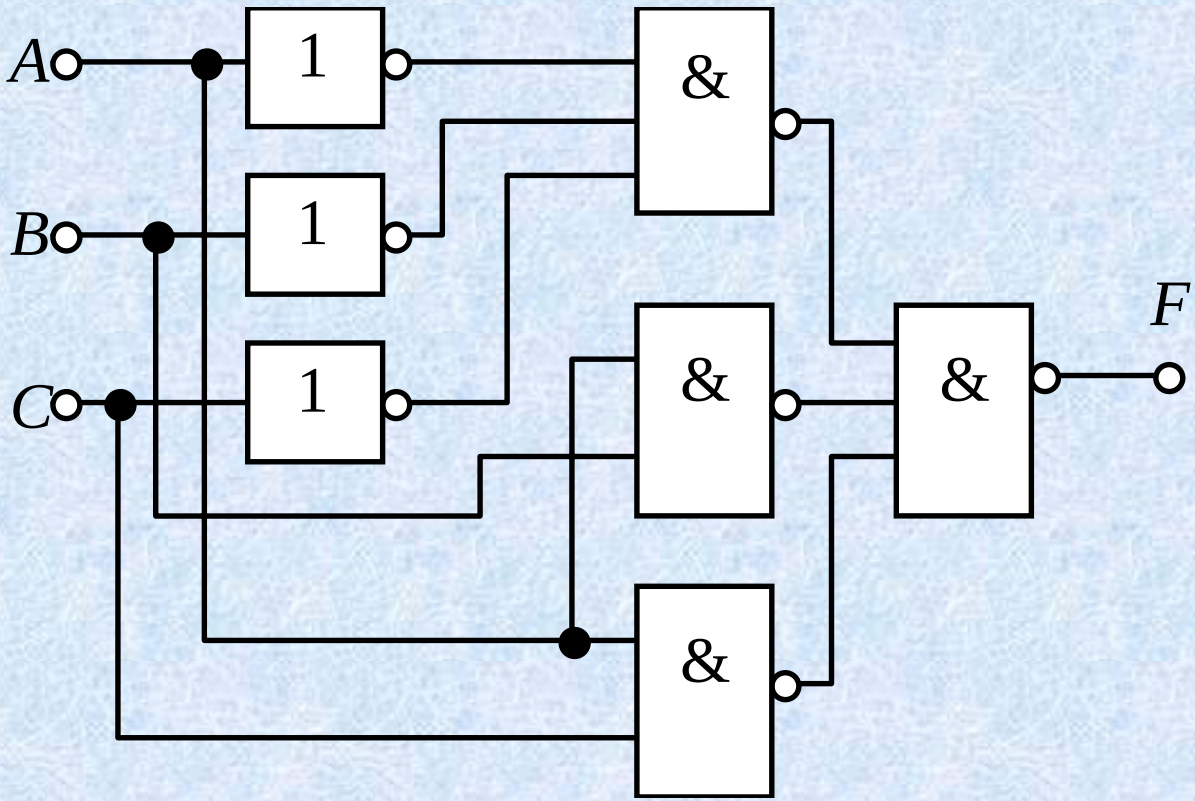
$$F = \overline{\overline{A}\overline{B}\overline{C}\;\overline{AB}\;\overline{AC}}$$

5

逻辑电路图

$$F = \overline{\overline{A}\overline{B}C\,AB\overline{A}C}$$

5



电路功能描述

穷举法

1

真值表

2

逻辑表达式

例： 用与非门设计一个举重裁判表决电路。设举重比赛有3个裁判，一个主裁判和两个副裁判。杠铃完全举上的裁决由每一个裁判按一下自己面前的按钮来确定。只有当两个或两个以上裁判判明成功，并且其中有一个为主裁判时，表明成功的灯才亮。

1

设主裁判为变量A，副裁判分别为B和C；表示成功与否的灯为Y，根据逻辑要求列出真值表。

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

2

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

化简 3

最简与或表达式

4

逻辑变换

5

逻辑电路图

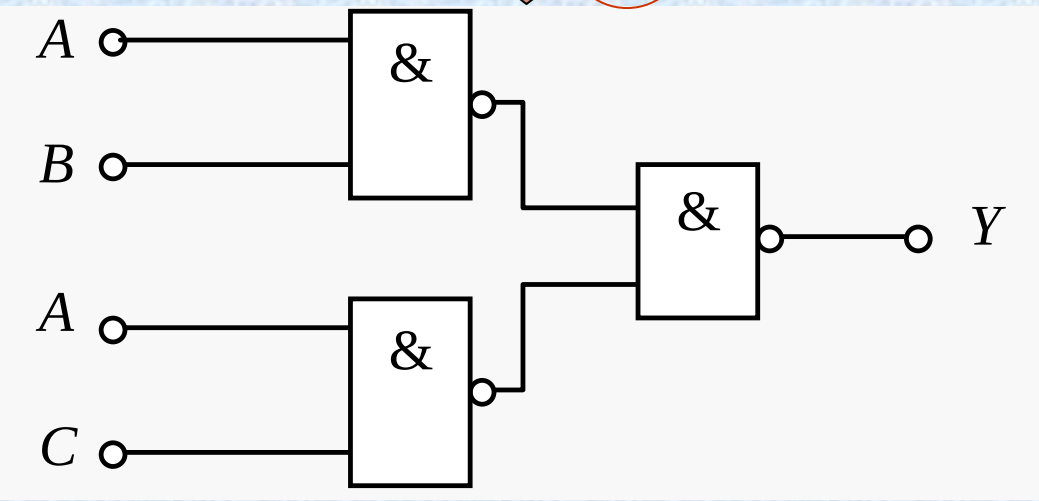
化简 3

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

4

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

5



组合逻辑电路部件

组合逻辑部件是指具有某种逻辑功能的中规模集成组合逻辑电路芯片。常用的组合逻辑部件有加法器、数值比较器、编码器、译码器、数据选择器和数据分配器等。

加法器

1、半加器

能对两个1位二进制数进行相加而求得和及进位的逻辑电路称为半加器。

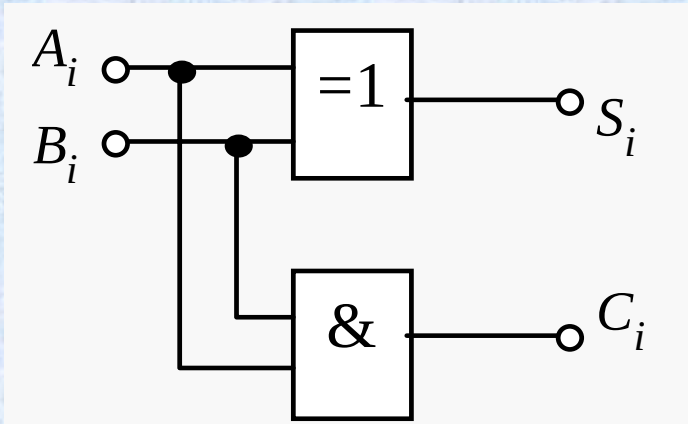
半加器真值表

A_i	B_i	S_i	C_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

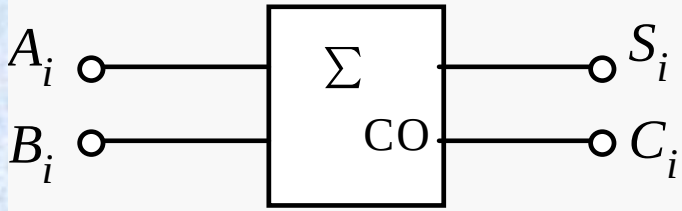
加数　本位的和　向高位的进位

$$S_i = \overline{A}_i B_i + A_i \overline{B}_i = A_i \oplus B_i$$

$$C_i = A_i B_i$$



半加器电路图



半加器符号

2、全加器

能对两个1位二进制数进行相加并考虑低位来的进位，即相当于3个1位二进制数相加，求得和及进位的逻辑电路称为全加

A_i	B_i	C_{i-1}	S_i	C_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$
\begin{aligned}
S_i &= \bar{A}_i\bar{B}_iC_{i-1} + \bar{A}_iB_i\bar{C}_{i-1} + A_i\bar{B}_i\bar{C}_{i-1} + A_iB_iC_{i-1} \\
&= \bar{A}_i(\bar{B}_iC_{i-1} + B_i\bar{C}_{i-1}) + A_i(\bar{B}_i\bar{C}_{i-1} + B_iC_{i-1}) \\
&= \bar{A}_i(B_i \oplus C_{i-1}) + A_i\overline{(B_i \oplus C_{i-1})} \\
&= A_i \oplus B_i \oplus C_{i-1}
\end{aligned}
$$

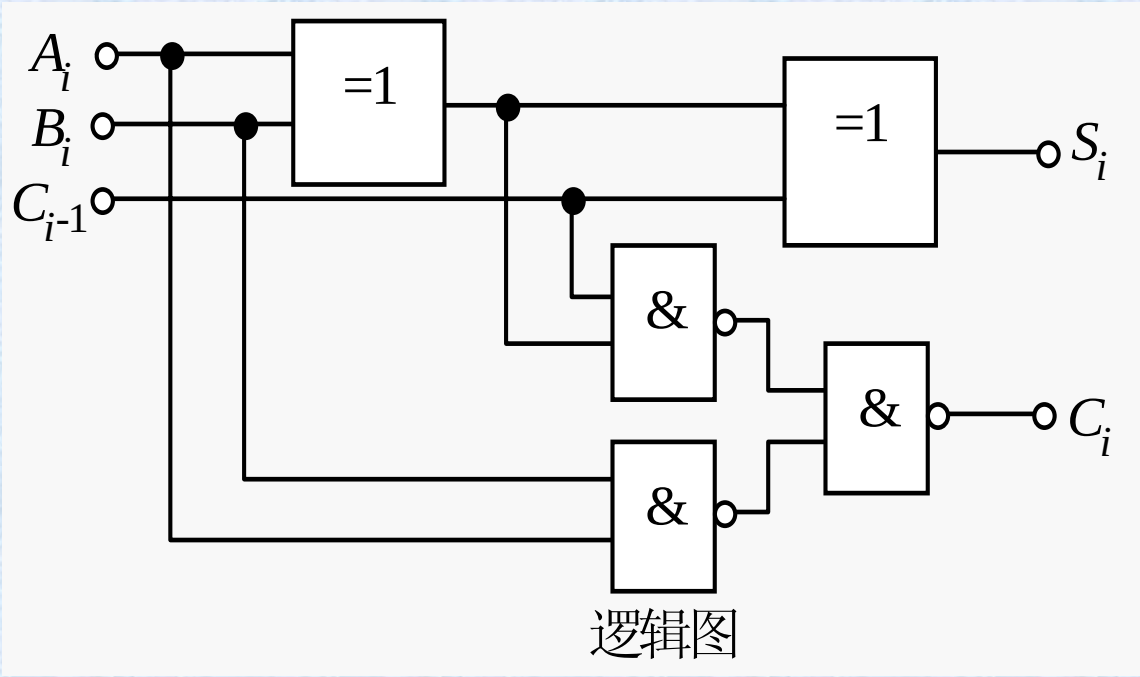
$$
\begin{aligned}
C_i &= \bar{A}_iB_iC_{i-1} + A_i\bar{B}_iC_{i-1} + A_iB_i \\
&= (\bar{A}_iB_i + A_i\bar{B}_i)C_{i-1} + A_iB \\
&= (A_i \oplus B_i)C_{i-1} + A_iB_i
\end{aligned}
$$

A_i、B_i：加数，C_{i-1}：低位来的进位，S_i：本位的和，C_i：向高位的进位。

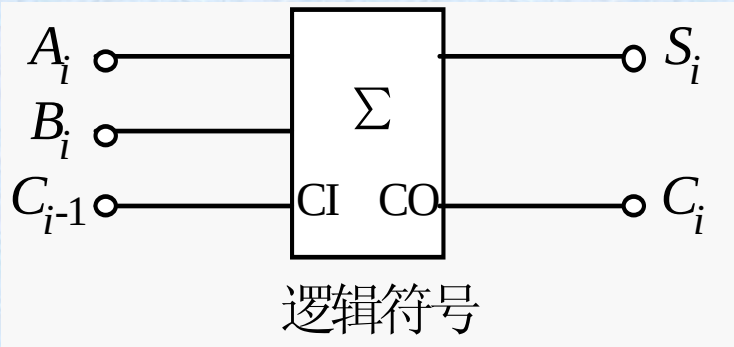
全加器的逻辑图和逻辑符号

$$S_i = A_i \oplus B_i \oplus C_{i-1}$$

$$C_i = (A_i \oplus B_i)C_{i-1} + A_iB_i$$



逻辑图

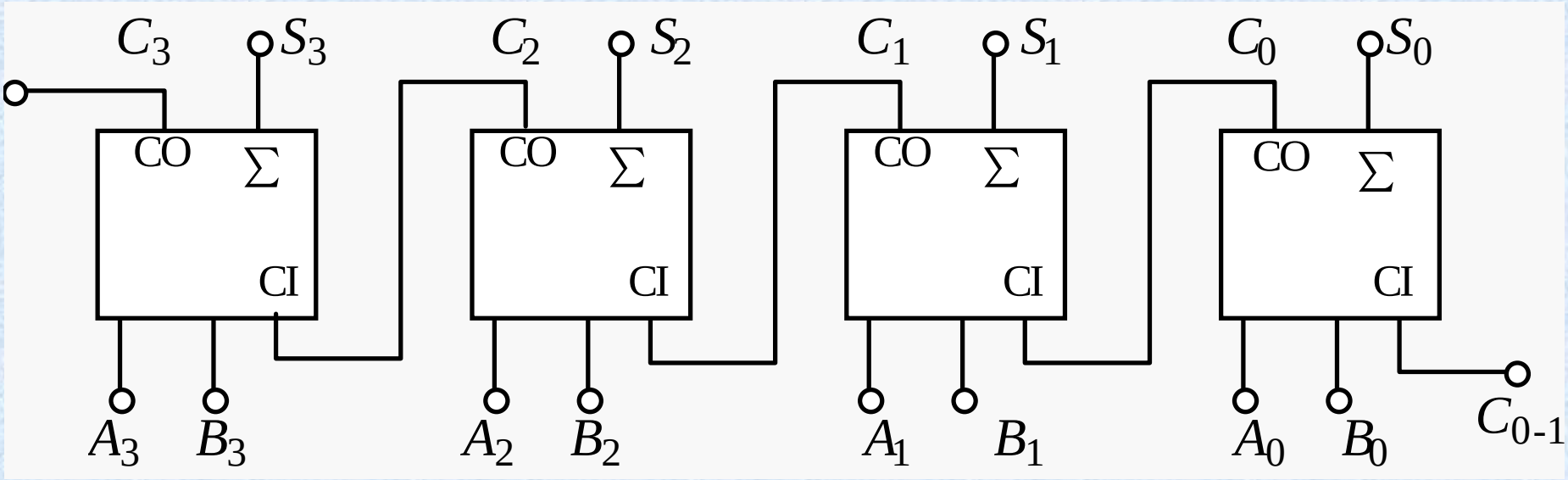


逻辑符号

实现多位二进制数相加的电路称为加法器。

串行进位加法器

构成：把n位全加器串联起来，低位全加器的进位输出连接到相邻的高位全加器的进位输入。



特点：进位信号是由低位向高位逐级传递的，速度不高。

为了提高运算速度，在逻辑设计上采用超前进位的方法，即每一位的进位根据各位的输入同时预先形成，而不需要等到低位的进位送来后才形成，这种结构的多位数加法器称为超前进位加法器。

数值比较器

用来完成两个二进制数的大小比较的逻辑电路称为数值比较器。

1位数值比较器

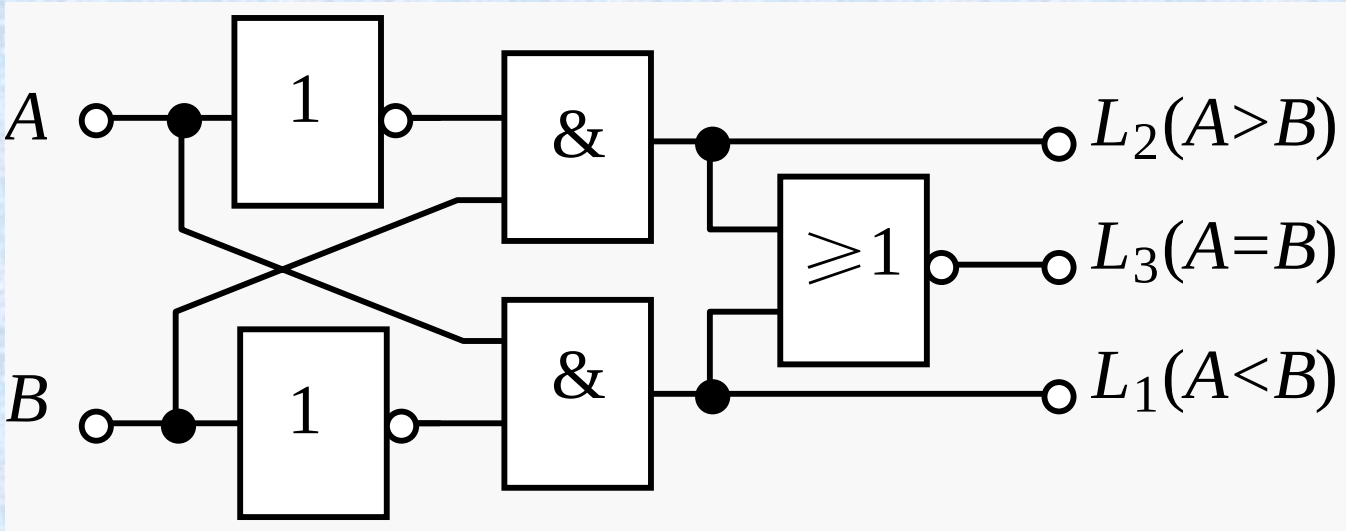
设$A>B$时$L_1=1$；$A<B$时$L_2=1$；$A=B$时$L_3=1$。得1位数值比较器的真值表。

A	B	$L_1(A>B)$	$L_2(A<B)$	$L_3(A=B)$
0	0	0	0	1
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1

逻辑表达式

$$\begin{cases} L_1 = A\overline{B} \\ L_2 = \overline{A}B \\ L_3 = \overline{A}\,\overline{B} + AB = \overline{\overline{A}B + A\overline{B}} \end{cases}$$

逻辑图



编码器

实现编码操作的电路称为编码器。

1、3位二进制编码器

输入	输出		
	Y_2	Y_1	Y_0
I_0	0	0	0
I_1	0	0	1
I_2	0	1	0
I_3	0	1	1
I_4	1	0	0
I_5	1	0	1
I_6	1	0	0
I_7	1	1	1

真值表

输入 8个互斥的信号

输出 3位二进制代码

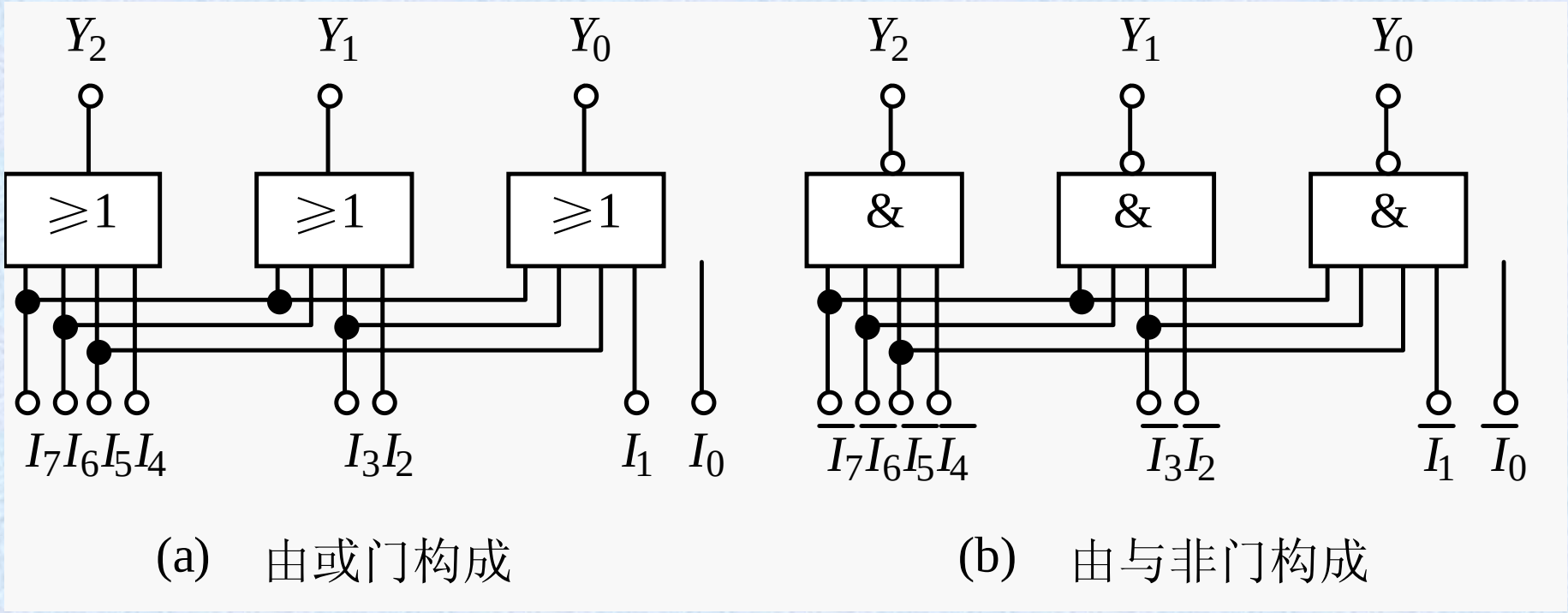
逻辑表达式

$$Y_2 = I_4 + I_5 + I_6 + I_7 = \overline{\overline{I}_4 \overline{I}_5 \overline{I}_6 \overline{I}_7}$$

$$Y_1 = I_2 + I_3 + I_6 + I_7 = \overline{\overline{I}_2 \overline{I}_3 \overline{I}_6 \overline{I}_7}$$

$$Y_0 = I_1 + I_3 + I_5 + I_7 = \overline{\overline{I}_1 \overline{I}_3 \overline{I}_5 \overline{I}_7}$$

逻辑图



2、8421 码编码器

真值表

输 入 I	输 出 Y_3	Y_2	Y_1	Y_0
$0(I_0)$	0	0	0	0
$1(I_1)$	0	0	0	1
$2(I_2)$	0	0	1	0
$3(I_3)$	0	0	1	1
$4(I_4)$	0	1	0	0
$5(I_5)$	0	1	0	1
$6(I_6)$	0	1	1	0
$7(I_7)$	0	1	1	1
$8(I_8)$	1	0	0	0
$9(I_9)$	1	0	0	1

输入 **10** 个互斥的数码

输出 **4** 位二进制代码

逻辑表达式

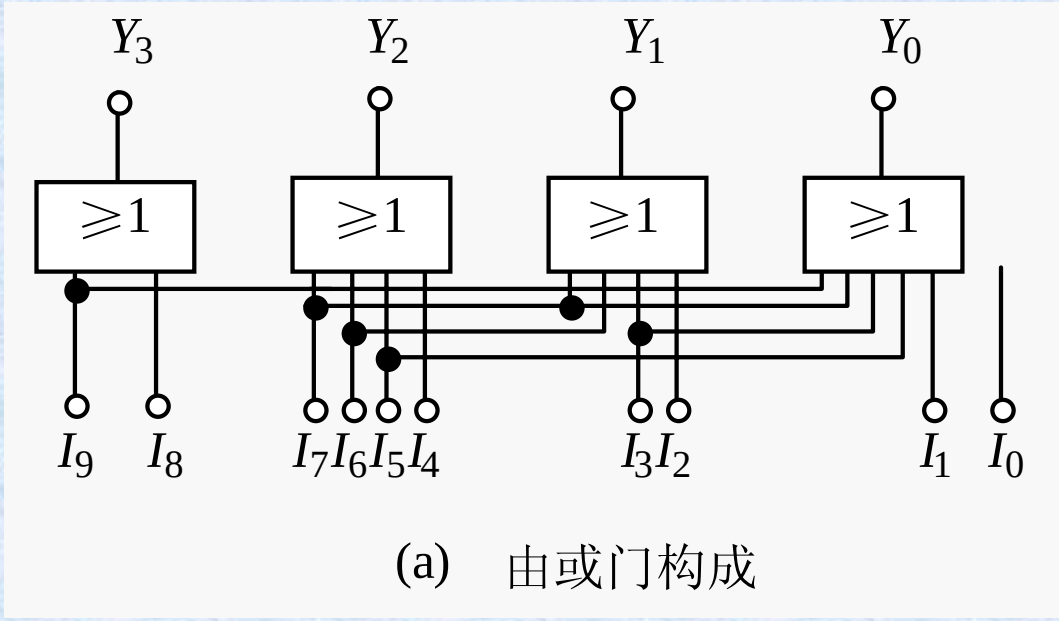
$$Y_3 = I_8 + I_9 = \overline{\overline{I_8}\,\overline{I_9}}$$

$$Y_2 = I_4 + I_5 + I_6 + I_7 = \overline{\overline{I_4}\,\overline{I_5}\,\overline{I_6}\,\overline{I_7}}$$

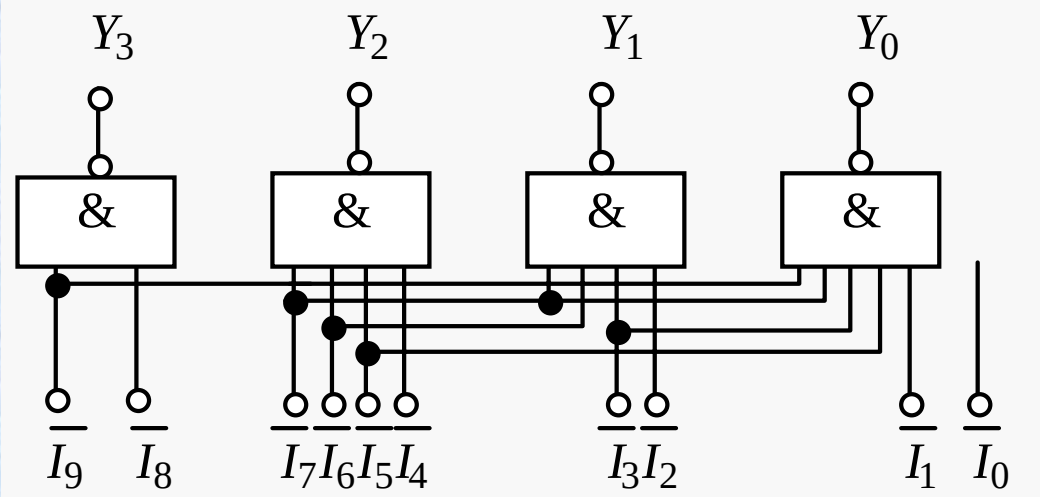
$$Y_1 = I_2 + I_3 + I_6 + I_7 = \overline{\overline{I_2}\,\overline{I_3}\,\overline{I_6}\,\overline{I_7}}$$

$$Y_0 = I_1 + I_3 + I_5 + I_7 + I_9 = \overline{\overline{I_1}\,\overline{I_3}\,\overline{I_5}\,\overline{I_7}\,\overline{I_9}}$$

逻辑图



(a) 由或门构成



(b) 由与非门构成

3、3位二进制优先编码器

在优先编码器中优先级别高的信号排斥级别低的，即具有单方面排斥的特性。设I_7的优先级别最高，I_6次之，依此类推，I_0最低。

真值表

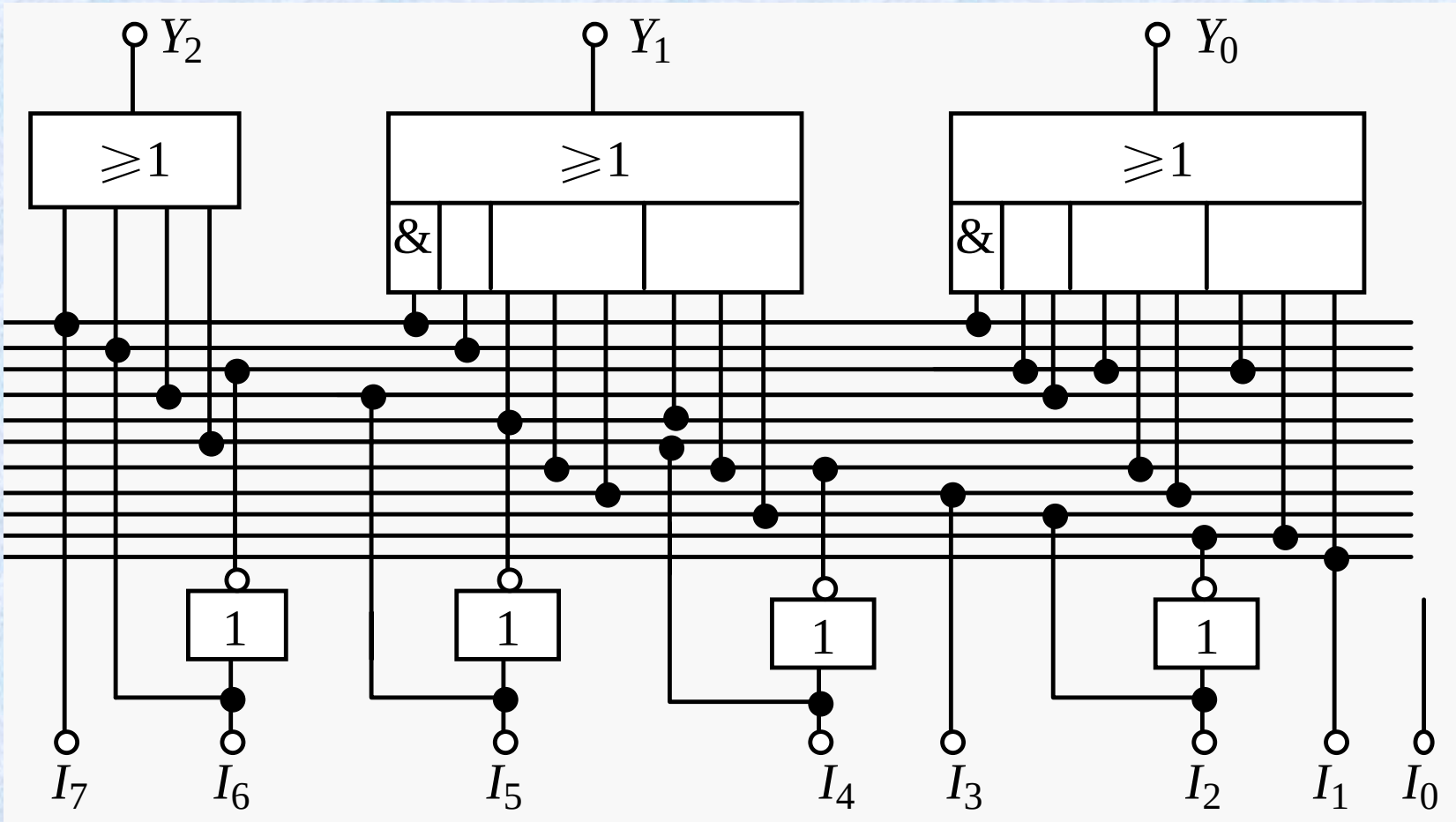
输入								输出		
I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0	Y_2	Y_1	Y_0
1	×	×	×	×	×	×	×	1	1	1
0	1	×	×	×	×	×	×	1	1	0
0	0	1	×	×	×	×	×	1	0	1
0	0	0	1	×	×	×	×	1	0	0
0	0	0	0	1	×	×	×	0	1	1
0	0	0	0	0	1	×	×	0	1	0
0	0	0	0	0	0	1	×	0	0	1
0	0	0	0	0	0	0	1	0	0	0

逻辑表达式

$$
\begin{cases}
Y_2 = I_7 + \bar{I}_7 I_6 + \bar{I}_7 \bar{I}_6 I_5 + \bar{I}_7 \bar{I}_6 \bar{I}_5 I_4 \\
\quad = I_7 + I_6 + I_5 + I_4 \\
Y_1 = I_7 + \bar{I}_7 I_6 + \bar{I}_7 \bar{I}_6 \bar{I}_5 \bar{I}_4 I_3 + \bar{I}_7 \bar{I}_6 \bar{I}_5 \bar{I}_4 \bar{I}_3 I_2 \\
\quad = I_7 + I_6 + \bar{I}_5 \bar{I}_4 I_3 + \bar{I}_5 \bar{I}_4 I_2 \\
Y_0 = I_7 + \bar{I}_7 \bar{I}_6 I_5 + \bar{I}_7 \bar{I}_6 \bar{I}_5 \bar{I}_4 I_3 + \bar{I}_7 \bar{I}_6 \bar{I}_5 \bar{I}_4 \bar{I}_3 \bar{I}_2 I_1 \\
\quad = I_7 + \bar{I}_6 I_5 + \bar{I}_6 \bar{I}_4 I_3 + \bar{I}_6 \bar{I}_4 \bar{I}_2 I_1
\end{cases}
$$

逻辑图

8线-3线优先编码器



如果要求输出、输入均为反变量，则只要在图中的每一个输出端和输入端都加上反相器就可以了。

译码器

把代码状态的特定含义翻译出来的过程称为译码，实现译码操作的电路称为译码器。

译码器就是把一种代码转换为另一种代码的电路。

1、二进制译码器

设二进制译码器的输入端为n个，则输出端为2^n个，且对应于输入代码的每一种状态，2^n个输出中只有一个为1（或为0），其余全为0（或为1）。

二进制译码器可以译出输入变量的全部状态，故又称为变量译码器。

3位二进制译码器

真值表

A_2	A_1	A_0	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

输入：3位二进制代码

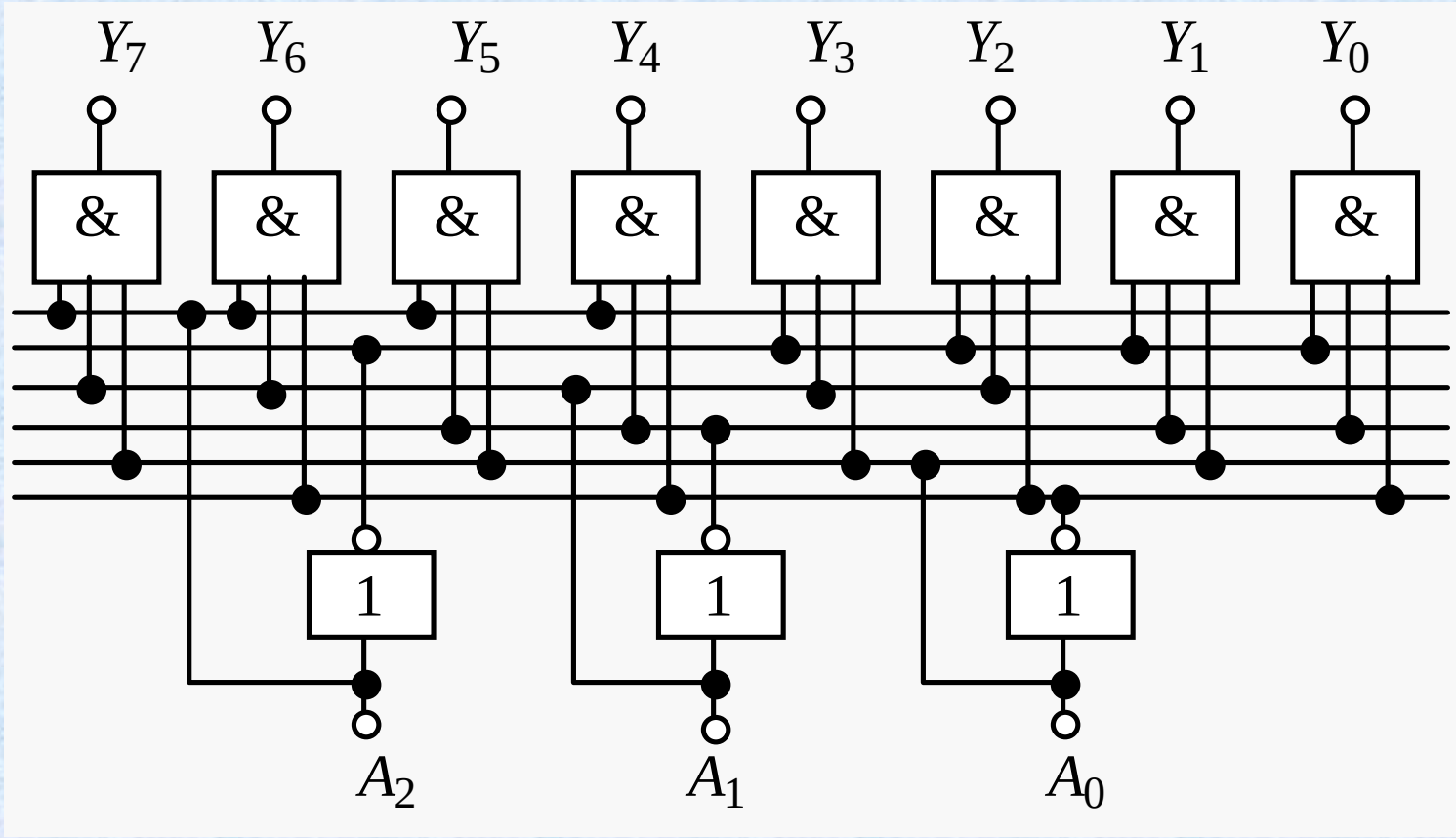
输出：8个互斥的信号

3 线-8 线译码器

逻辑表达式

$$\begin{cases} Y_0 = \overline{A}_2\overline{A}_1\overline{A}_0 \\ Y_1 = \overline{A}_2\overline{A}_1 A_0 \\ Y_2 = \overline{A}_2 A_1\overline{A}_0 \\ Y_3 = \overline{A}_2 A_1 A_0 \\ Y_4 = A_2\overline{A}_1\overline{A}_0 \\ Y_5 = A_2\overline{A}_1 A_0 \\ Y_6 = A_2 A_1\overline{A}_0 \\ Y_7 = A_2 A_1 A_0 \end{cases}$$

逻辑图



电路特点：与门组成的阵列

2、8421 码译码器

把二-十进制代码翻译成10个十进制数字信号的电路，称为二-十进制译码器。

二-十进制译码器的输入是十进制数的4位二进制编码（BCD码），分别用A_3、A_2、A_1、A_0表示；输出的是与10个十进制数字相对应的10个信号，用Y_9～Y_0表示。由于二-十进制译码器有4根输入线，10根输出线，所以又称为4线-10线译码器。

真值表

A_3	A_2	A_1	A_0	Y_9	Y_8	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0	0	0	1	0	0
0	0	1	1	0	0	0	0	0	0	1	0	0	0
0	1	0	0	0	0	0	0	0	1	0	0	0	0
0	1	0	1	0	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	0	1	0	0	0	0	0	0
0	1	1	1	0	0	1	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	0	0	0	0	0	0	0

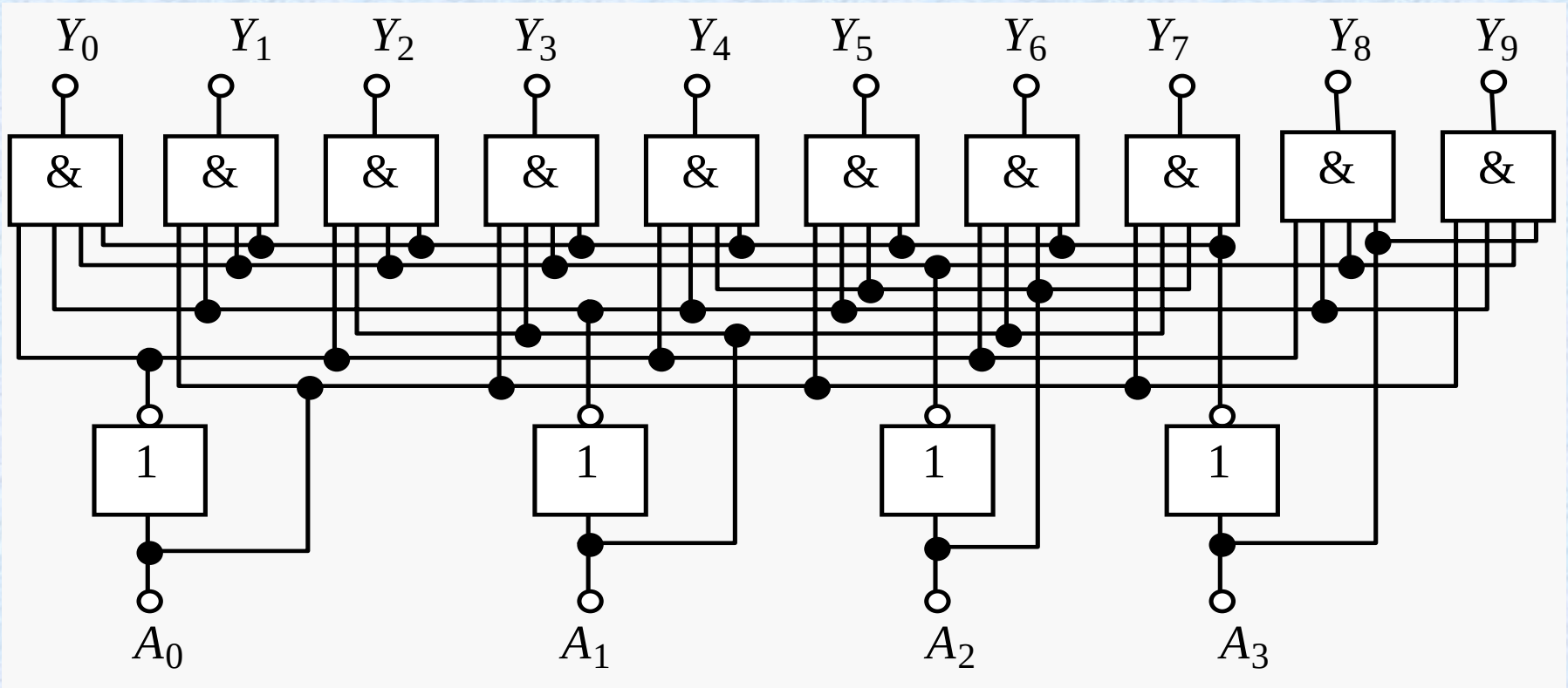
逻辑表达式

$$Y_0 = \overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0 \qquad Y_1 = \overline{A}_3\overline{A}_2\overline{A}_1 A_0 \qquad Y_2 = \overline{A}_3\overline{A}_2 A_1\overline{A}_0 \qquad Y_3 = \overline{A}_3\overline{A}_2 A_1 A_0$$

$$Y_4 = \overline{A}_3 A_2\overline{A}_1\overline{A}_0 \qquad Y_5 = \overline{A}_3 A_2\overline{A}_1 A_0 \qquad Y_6 = \overline{A}_3 A_2 A_1\overline{A}_0 \qquad Y_7 = \overline{A}_3 A_2 A_1 A_0$$

$$Y_8 = A_3\overline{A}_2\overline{A}_1\overline{A}_0 \qquad Y_9 = A_3\overline{A}_2\overline{A}_1 A_0$$

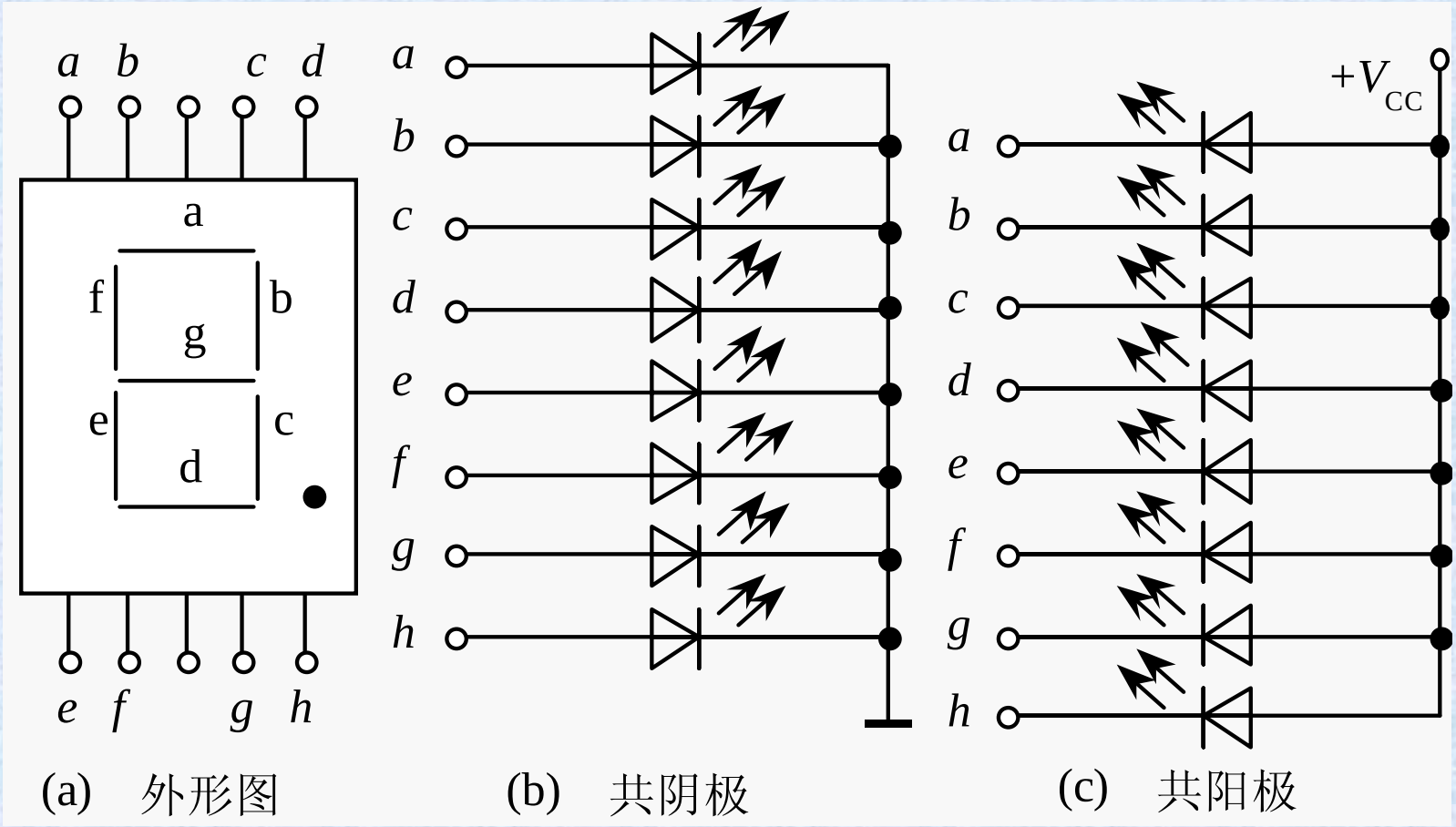
逻辑图



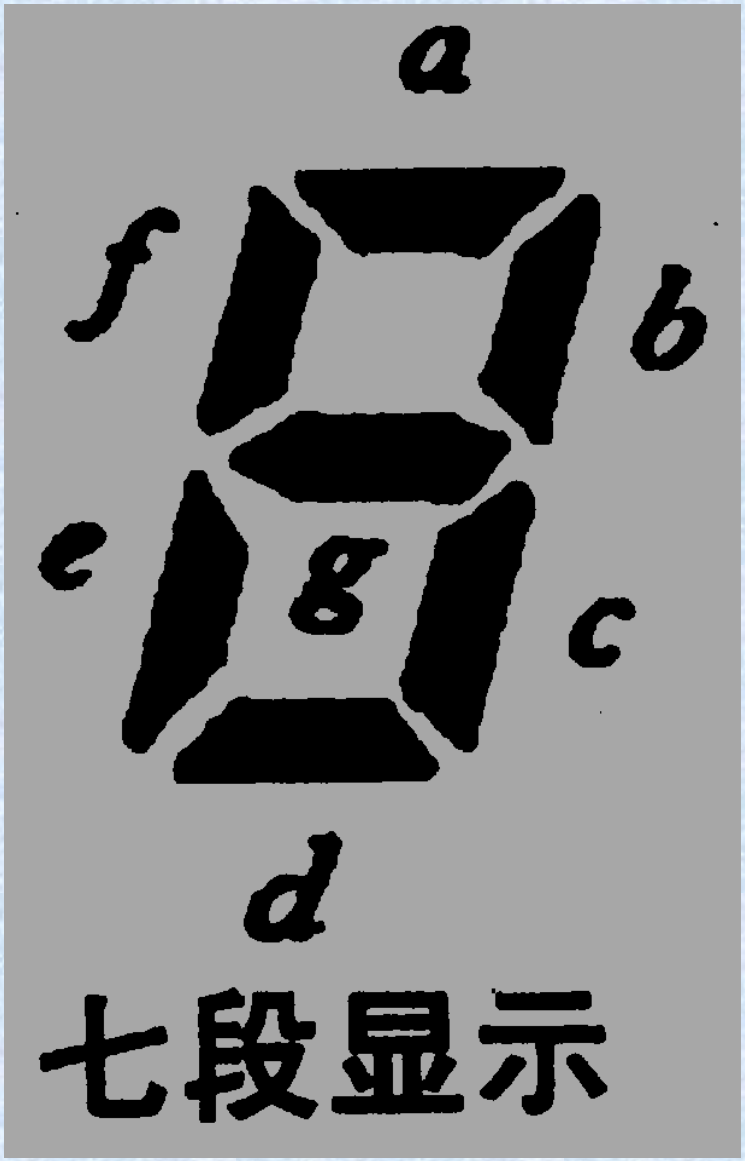
3、显示译码器

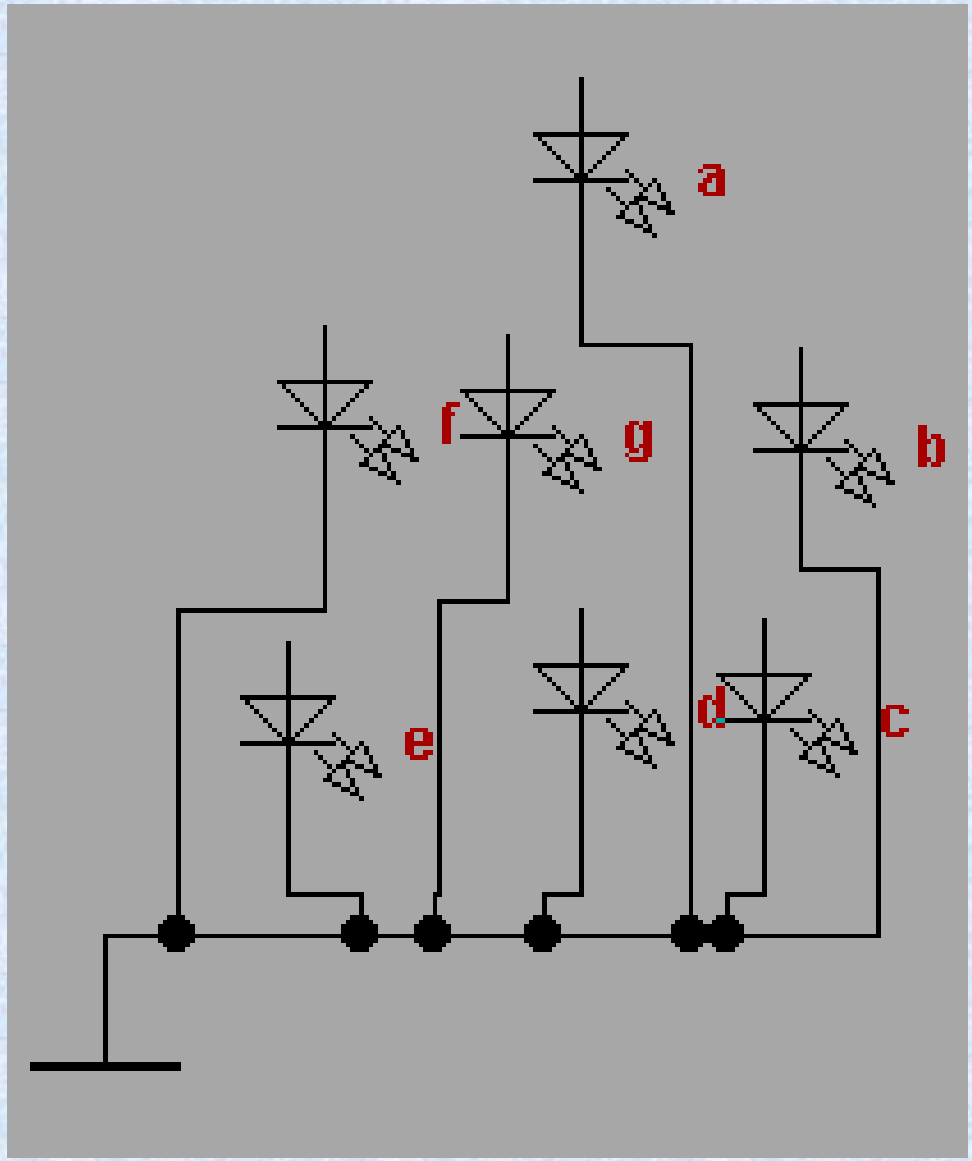
用来驱动各种显示器件，从而将用二进制代码表示的数字、文字、符号翻译成人们习惯的形式直观地显示出来的电路，称为显示译码器。

数码显示器

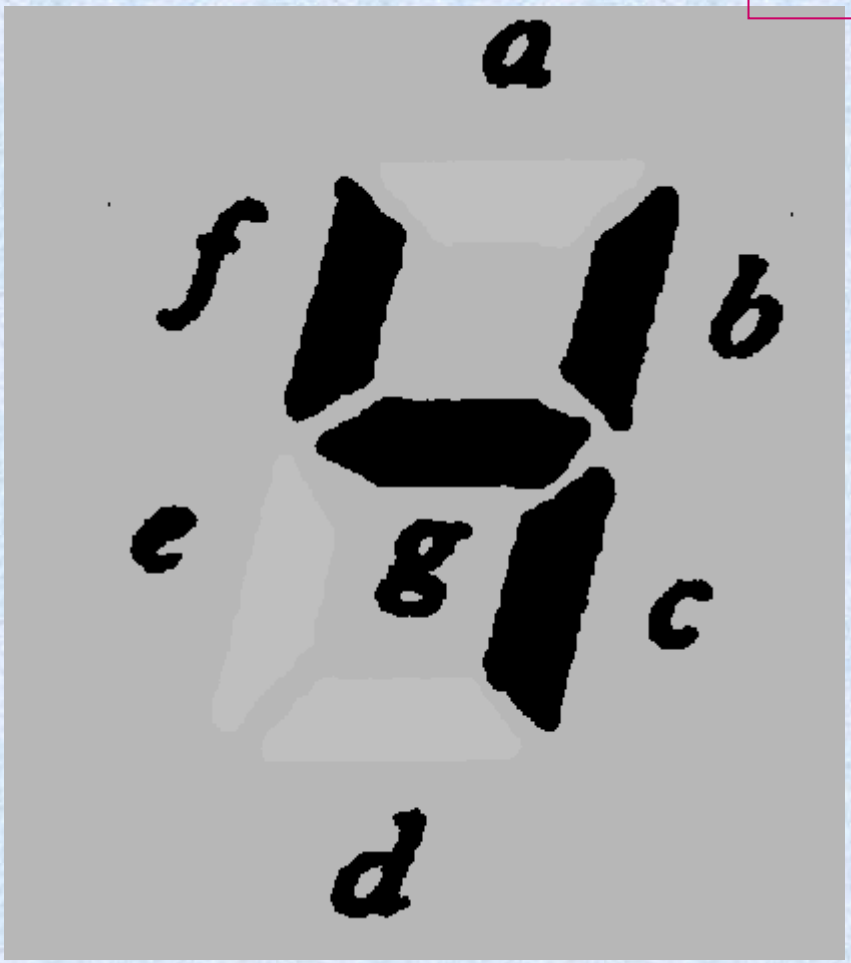


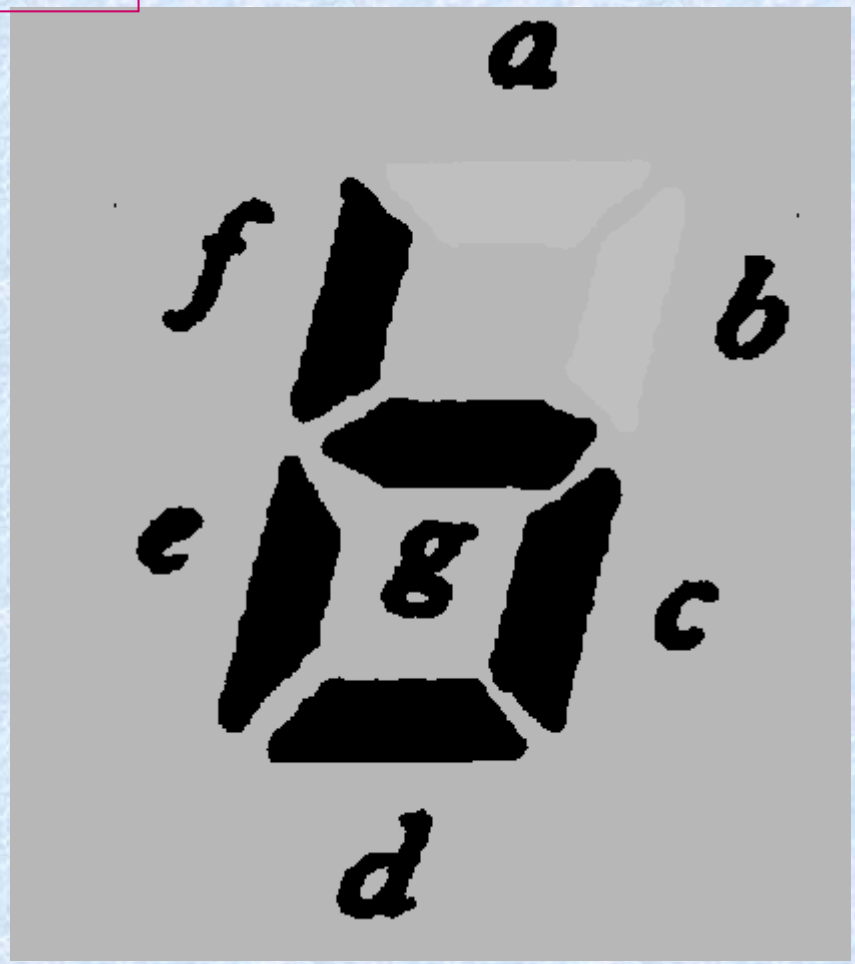
(a) 外形图　(b) 共阴极　(c) 共阳极


a
f
b
g
c
c
d
七段显示


a
f
g
b
e
d
c

共阴极





b=c=f=g=1,
a=d=e=0时

c=d=e=f=g=1,
a=b=0时

显示译码器真值表

输入				输出							显示字形
A_3	A_2	A_1	A_0	a	b	c	d	e	f	g	
0	0	0	0	1	1	1	1	1	1	0	
0	0	0	1	0	1	1	0	0	0	0	
0	0	1	0	1	1	0	1	1	0	1	
0	0	1	1	1	1	1	1	0	0	1	
0	1	0	0	0	1	1	0	0	1	1	
0	1	0	1	1	0	1	1	0	1	1	
0	1	1	0	0	0	1	1	1	1	1	
0	1	1	1	1	1	1	0	0	0	0	
1	0	0	0	1	1	1	1	1	1	1	
1	0	0	1	1	1	1	0	0	1	1	

真值表仅适用于共阴极LED

4选1数据选择器

输入数据

地址变量

输入			输出
D	A_1	A_0	Y
D_0	0	0	D_0
D_1	0	1	D_1
D_2	1	0	D_2
D_3	1	1	D_3

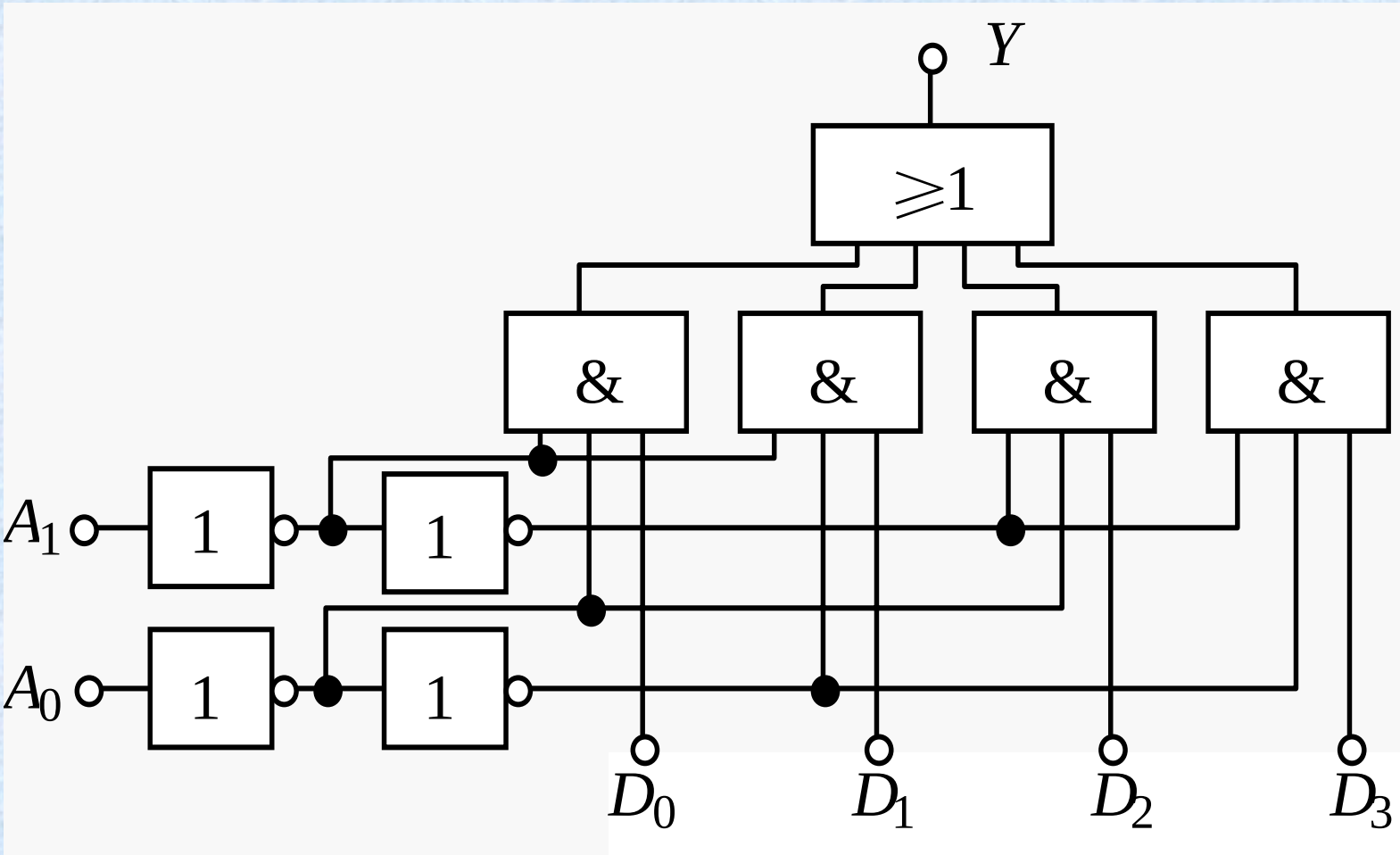
真值表

由地址码决定从 4 路输入中选择哪 1 路输出。

逻辑表达式

$$Y = D_0\overline{A}_1\overline{A}_0 + D_1\overline{A}_1A_0 + D_2A_1\overline{A}_0 + D_3A_1A_0$$

逻辑图



1路-4路数据分配器

输入数据

地址变量

输入		输出			
A_1	A_0	Y_0	Y_1	Y_2	Y_3
0	0	D	0	0	0
0	1	0	D	0	0
1	0	0	0	D	0
1	1	0	0	0	D

D

真值表

由地址码决定将输入数据 D 送给哪 1 路输出。

逻辑表达式

$$Y_0 = D\overline{A_1}\,\overline{A_0} \qquad Y_1 = D\overline{A_1}A_0$$

$$Y_2 = DA_1\overline{A_0} \qquad Y_3 = DA_1A_0$$

逻辑图

$$Y_0 = D\overline{A}_1\overline{A}_0 \qquad Y_1 = D\overline{A}_1 A_0$$

$$Y_2 = DA_1\overline{A}_0 \qquad Y_3 = DA_1 A_0$$