第三章

组合逻辑电路

宗　汝
西安电子科技大学电子工程学院

注：标*号的部分仅做参考了解，不要求掌握。

本章要点

- 组合逻辑电路的定义
- 组合逻辑电路的分析
- 组合逻辑电路的设计
- 常见组合逻辑功能电路

本章学习目标

- 掌握组合逻辑电路的分析方法和步骤。
- 掌握组合逻辑电路的设计方法和步骤，根据器件选型要求设计最简逻辑电路。
- 掌握主要的组合逻辑电路功能，掌握译码器、数据选择器、加法器、减法器、比较器等。
- 了解组合逻辑电路的延时特点

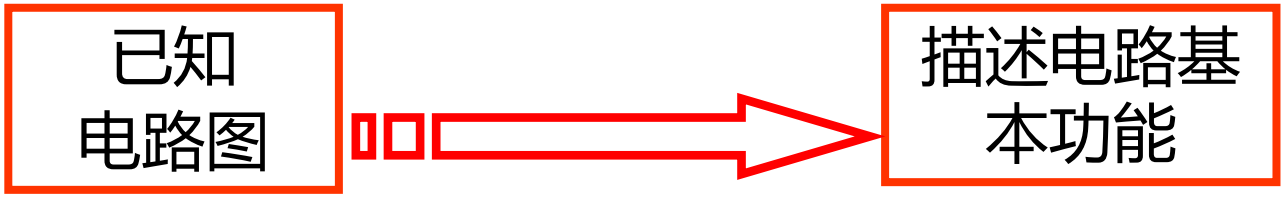
组合逻辑电路的定义

逻辑电路

- 组合逻辑电路：现时的输出仅取决于现时的输入
- 时序逻辑电路：除与现时输入有关外还与电路的原状态有关

组合逻辑电路的分析

基本思想：



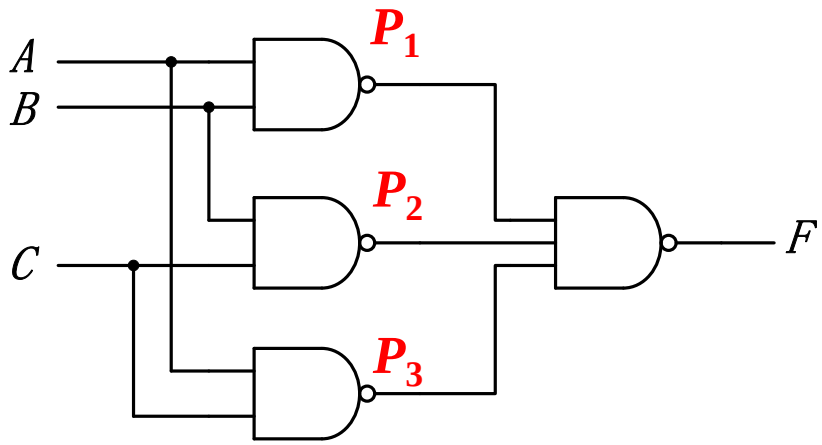
分析方法和步骤：

1. 由给定的逻辑图写出逻辑关系表达式。
2. 对逻辑表达式进行必要的化简，列出输入输出真值表。
3. 概括出电路功能的结论。

示例

例：分析图示组合逻辑电路的逻辑功能。

解：



1.逐级推导

$$P_1 = \overline{AB},$$

$$P_2 = \overline{BC},$$

$$P_3 = \overline{AC}$$

2.写出逻辑表达式

$$F = \overline{P_1 \cdot P_2 \cdot P_3} = \overline{\overline{AB} \cdot \overline{BC} \cdot \overline{AC}} = AB + BC + AC$$

示例

$$F = AB + BC + AC$$

3.列出真值表

A	B	C	F
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

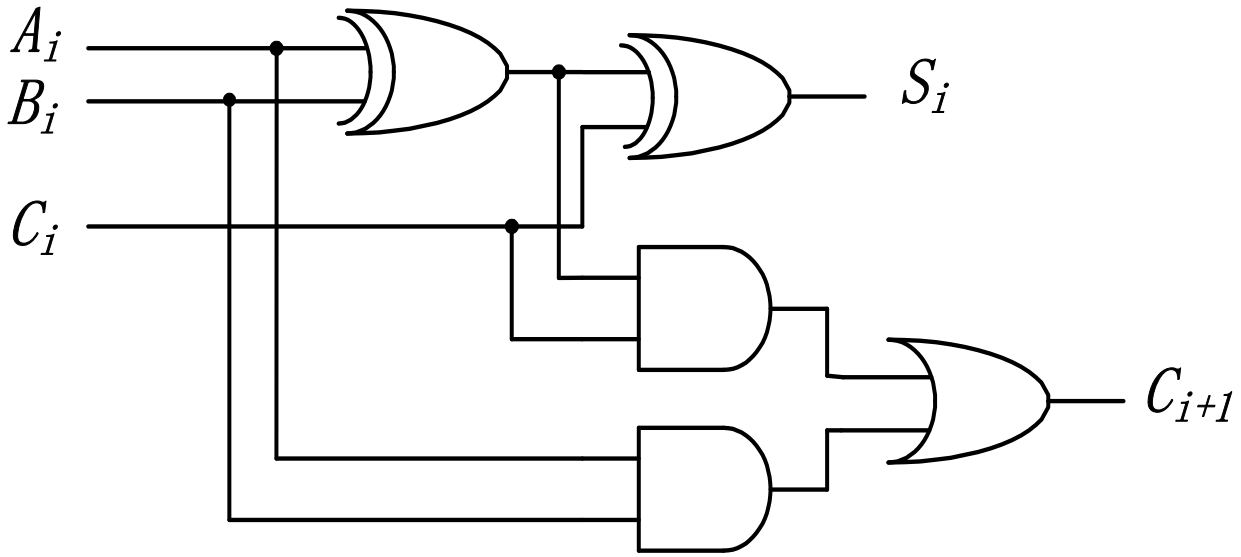
4.分析功能

通过观察分析真值表，电路表示了一种“少数服从多数”的逻辑关系。

因此可以将该电路概括为：三变量多数表决器。

示例

例2：分析下图的逻辑功能。



① 写出函数表达式

$$S_i = A_i \oplus B_i \oplus C_i$$

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

示例

代入整理后，两输出为：

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

$$C_{i+1} = (A_i \oplus B_i)C_i + A_iB_i$$
$$= \overline{A}BC_i + A\overline{B}C_i + A_iB_i$$

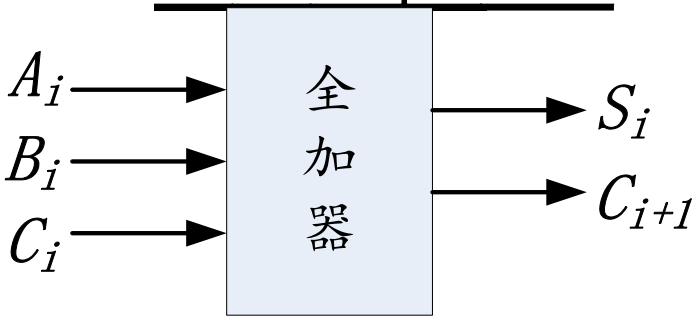
② 列真值表

A	B	C_i	S_i	C_{i+1}
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

③ 分析功能

功能： S_i为A、B、C_i之和，
C_{i+1}为三个数之和产生的进位。

一位全加器



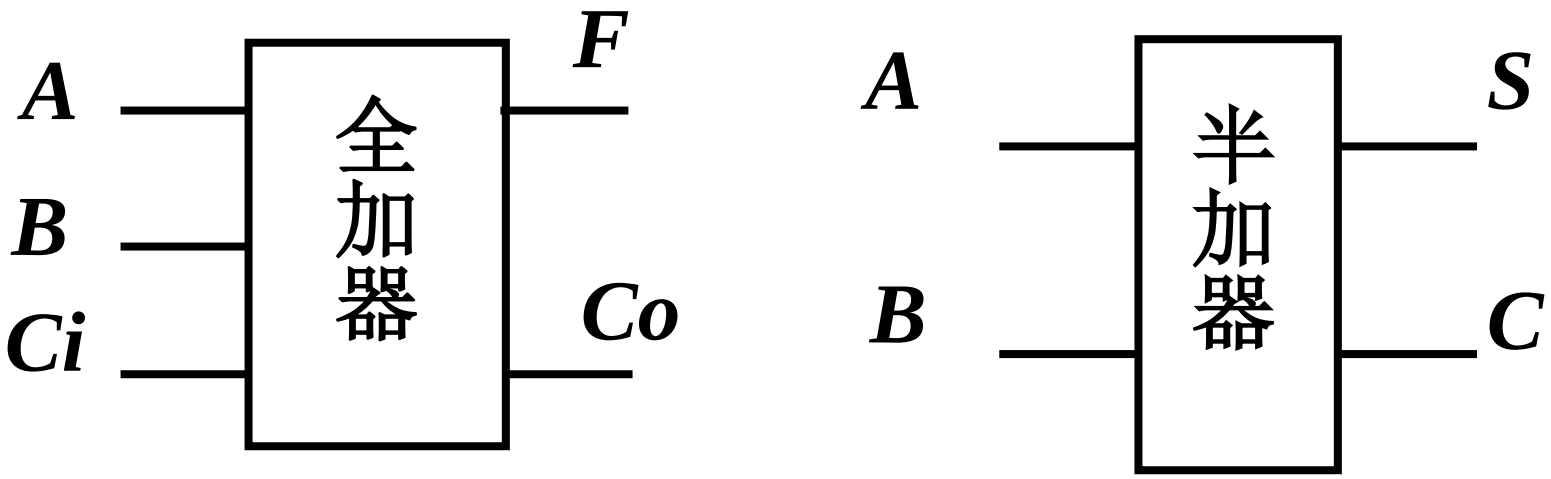
组合逻辑电路分析小结

1. 电路→逻辑表达式;

2. 化简，列出真值表;

3. 概括结论。

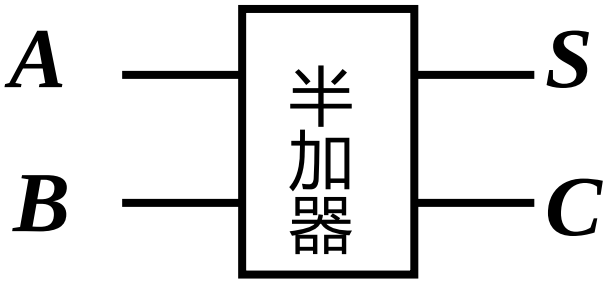
加法器



半加运算不考虑从低位来的进位。

半加器

A---加数；*B*---被加数；*S*---本位和；*C*---进位。

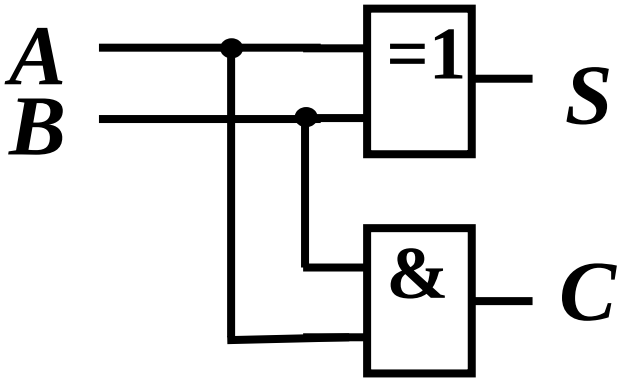


集成逻辑符号

逻辑函数

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

$$C = AB$$



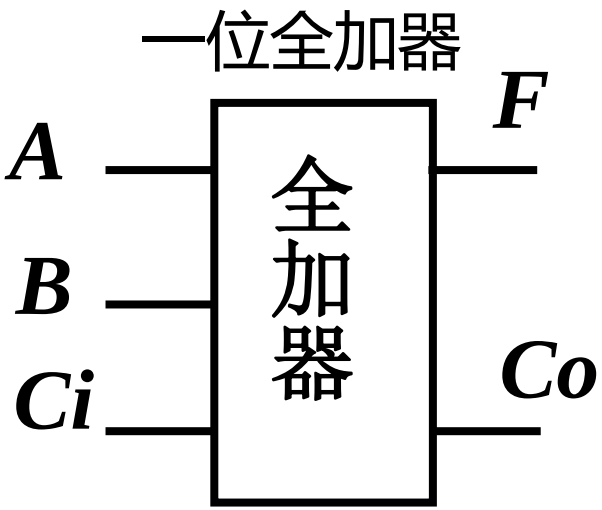
逻辑图

真值表

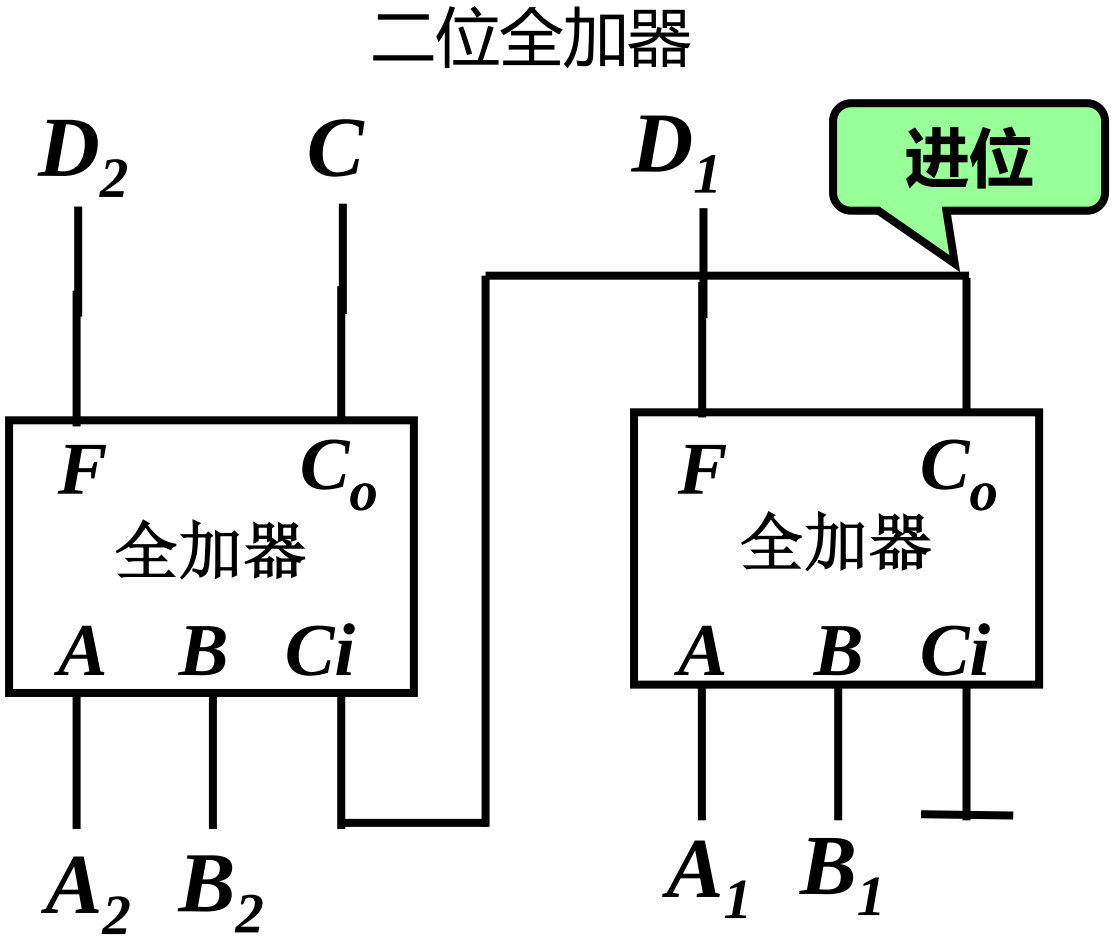
A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

全加器

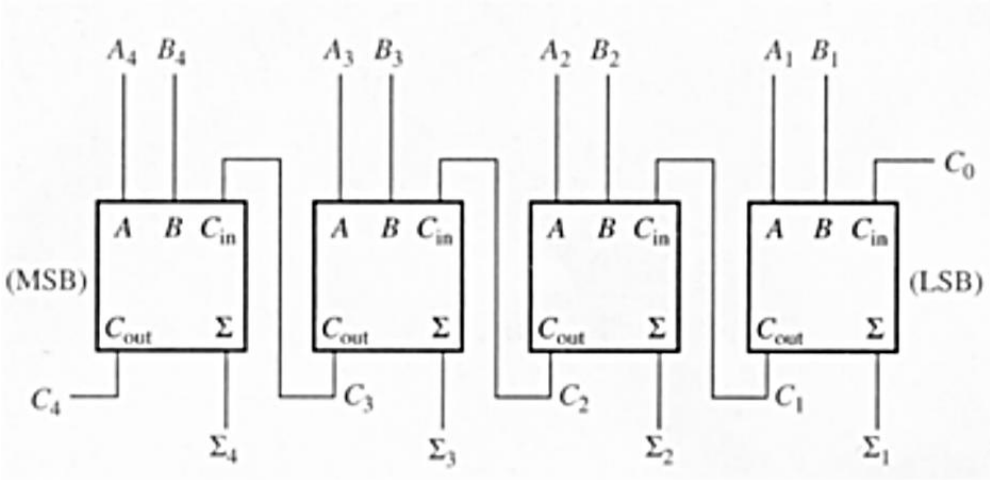
相加过程中，既考虑加数、被加数又考虑低位的进位位。

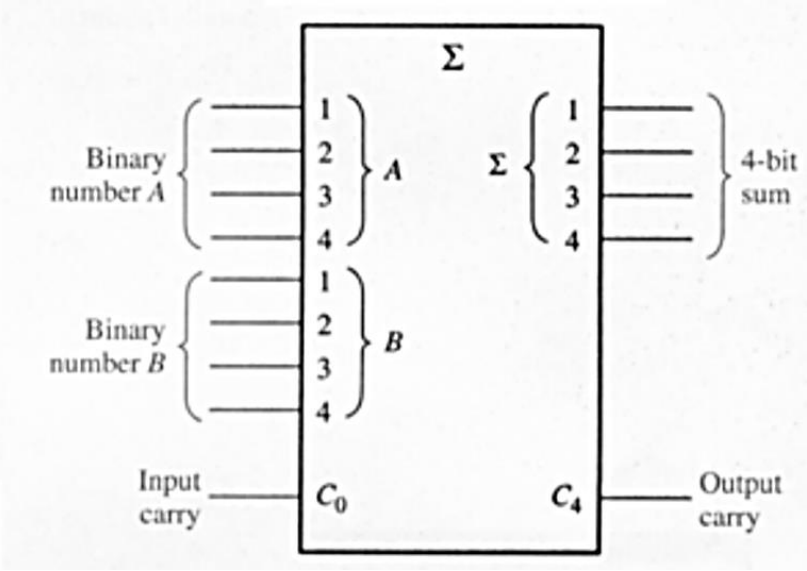


$$\begin{array}{r} A_2\ A_1 \\ +\ B_2 C_1 B_1 \\ \hline C\ D_2\ D_1 \end{array}$$



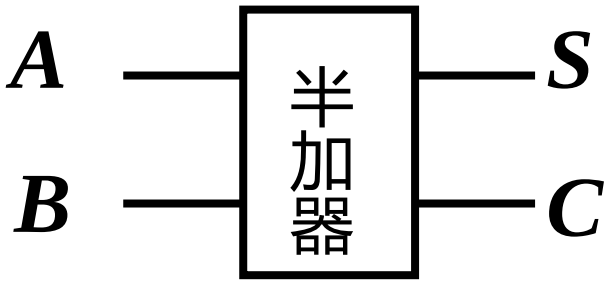
4位二进制全加器

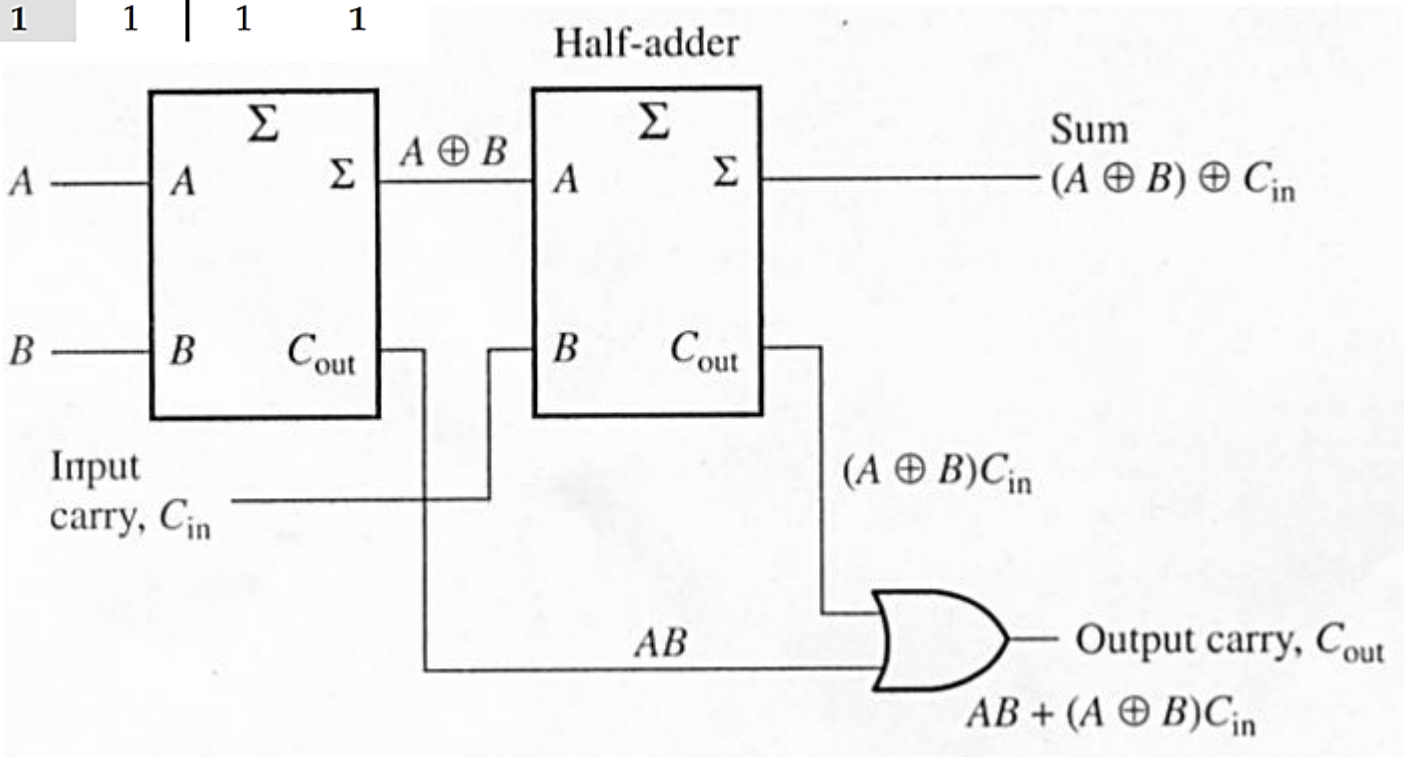

A_4 B_4
A_3 B_3
A_2 B_2
A_1 B_1
C_0
A B C_{in}
C_{out} Σ
(MSB)
(LSB)
C_4
C_3
C_2
C_1
Σ_4
Σ_3
Σ_2
Σ_1


Σ
Binary number A
Binary number B
1
2
3
4
A
B
Σ
4-bit sum
Input carry
C_0
C_4
Output carry

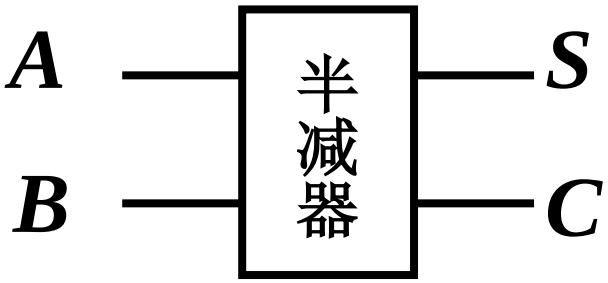
半加器构成全加器

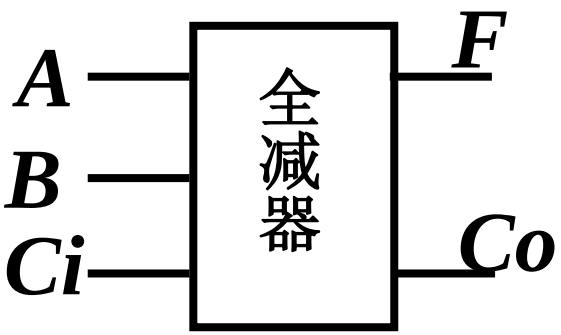
A	B	HA_s	HA_c	C_i	S	C_o
0	0	0	0	0	0	0
0	0	0	0	1	1	0
0	1	1	0	0	1	0
0	1	1	0	1	0	1
1	0	1	0	0	1	0
1	0	1	0	1	0	1
1	1	0	1	0	0	1
1	1	0	1	1	1	1





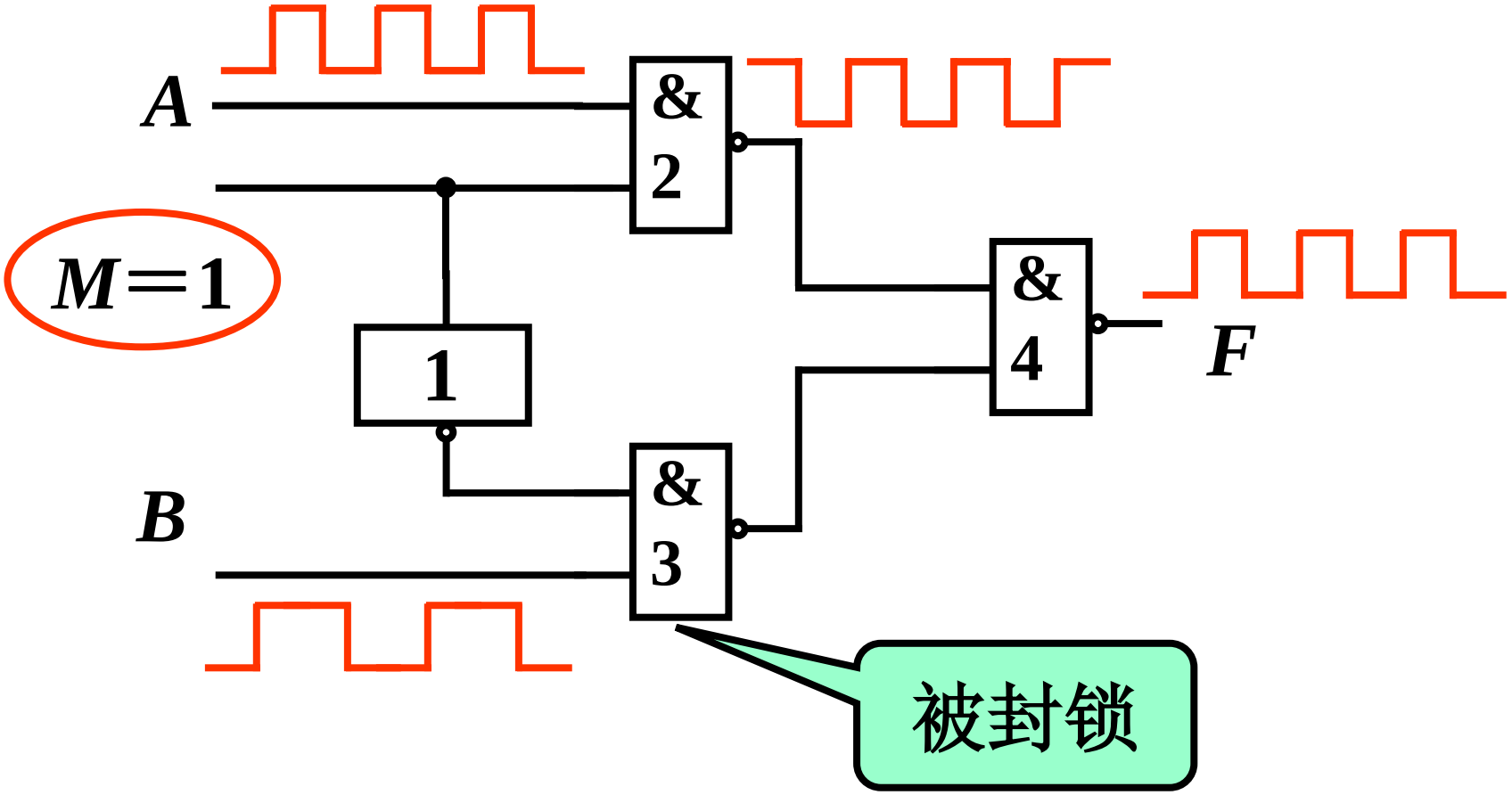
一位半减器与全减器



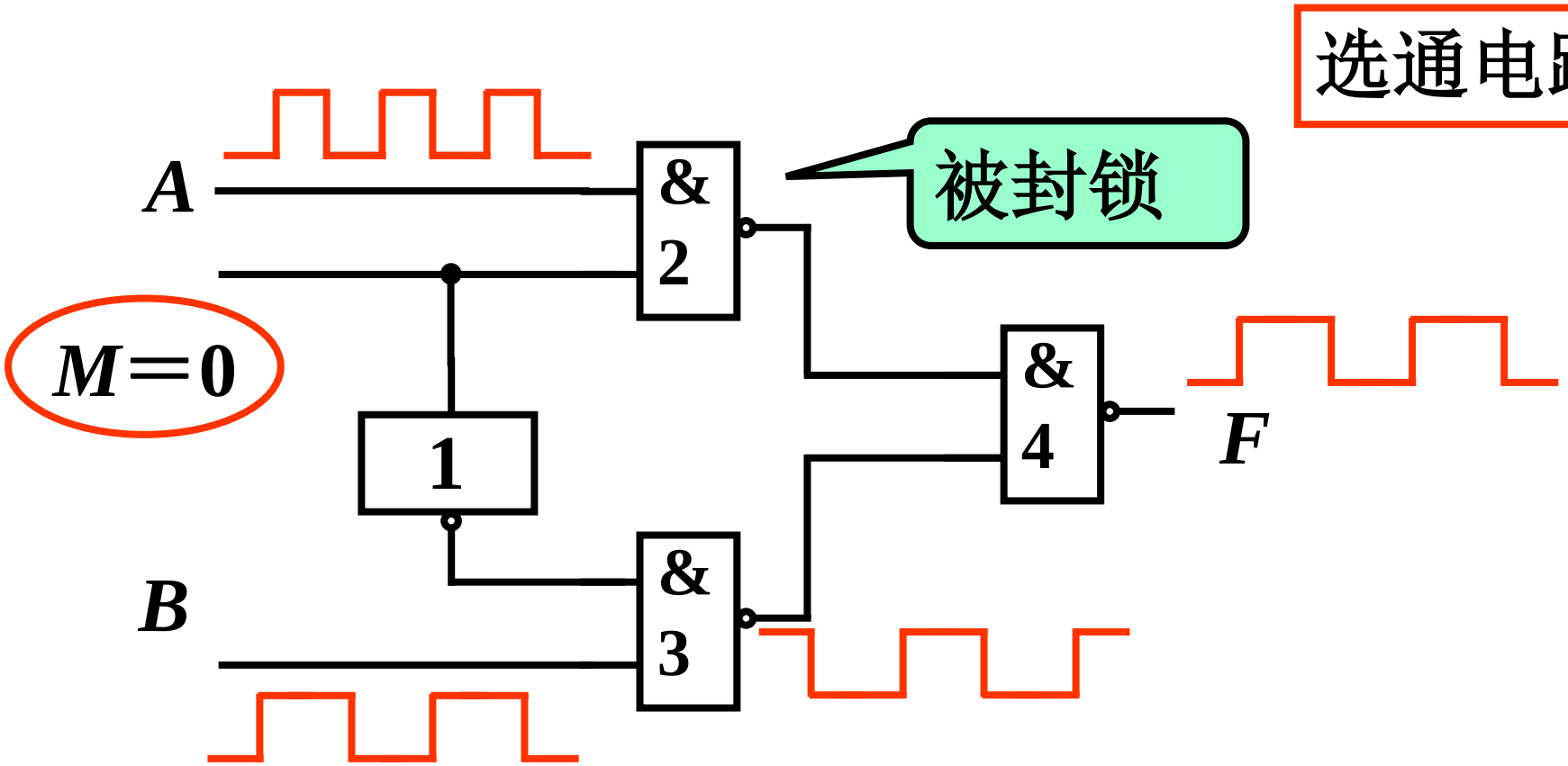


*示例

例3：分析下图的逻辑功能。



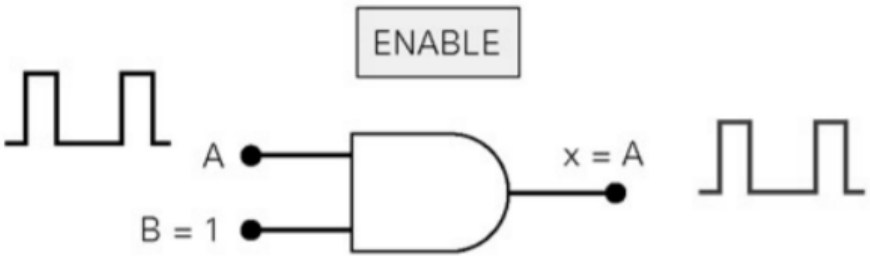
*示例

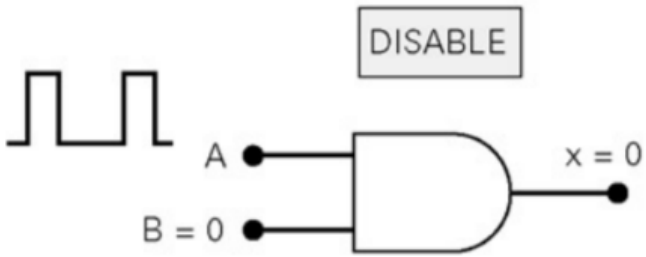


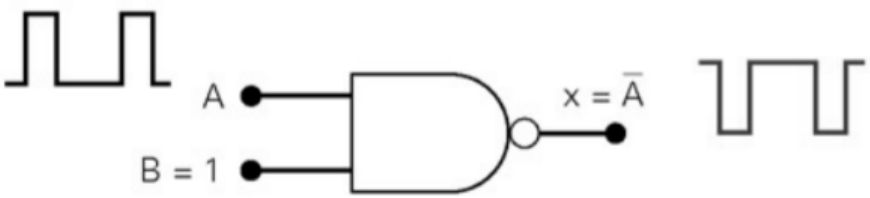
使能电路：

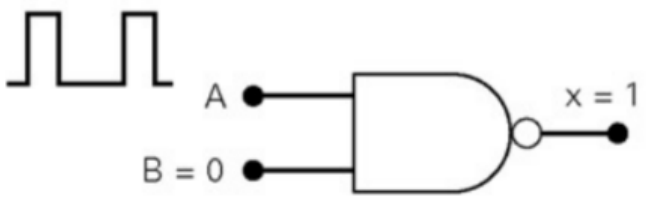
1. 利用与非门（或其他逻辑门）设计选通电路。

*简单门结构选通电路



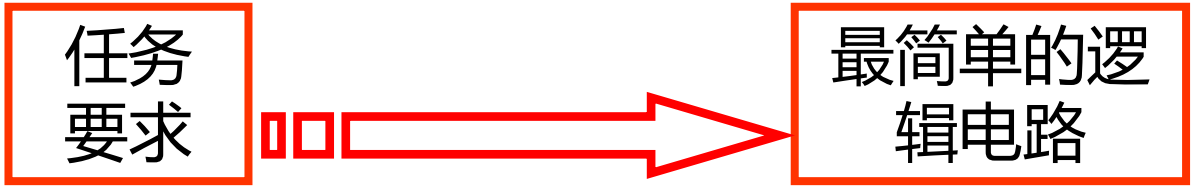






组合逻辑电路的设计

基本思想：



设计原则：

最佳设计主要考虑问题有：

① 所用的逻辑器件数目最少，器件的种类最少，且器件之间的连线最简单。这样的电路称“最小化”电路。

② 满足速度要求，应使级数尽量少，以减少门电路的延迟。

③ 功耗小，工作稳定可靠。

设计步骤

1. 逻辑抽象：

   指定实际问题的逻辑符号与含义，列出真值表。

2. 选择器件类型：

   根据任务和现有条件选择采用合适的器件。

3. 写出逻辑表达式：

   用逻辑代数或卡诺图对逻辑表达式进行化简，并变换成与器件对应的最简表达式。

4. 画出逻辑电路图。

示例

例3：设计一个一位二进制全减器。

① 逻辑抽象。

全减器有三个输入变量：被减数A_n、减数B_n、低位向本位的借位C_n；有两个输出变量：本位差D_n、本位向高位的借位C_{n+1}。

列真值表：

A_n	B_n	C_n	C_{n+1}	D_n
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

② 选器件。

这里分别选用异或门、与或非门来实现（以比较不同的实现电路）

③ 写逻辑表达式。

画出C_{n+1}和D_n的K图，根据选用的器件将C_{n+1}、D_n分别化简为相应的函数式。

A_n	B_n	C_n	C_{n+1}	D_n
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

C_n \ A_nB_n	00	01	11	10
0	0	1	0	0
1	1	1	1	0

C_{n+1}

C_n \ A_nB_n	00	01	11	10
0	0	1	0	1
1	1	0	1	0

D_n

1.当用“与或非”门实现电路时，写出相应的函数式为:

$$D_n = \overline{\overline{A}_n \overline{B}_n \overline{C}_n + \overline{A}_n B_n C_n + A_n B_n \overline{C}_n + A_n \overline{B}_n C_n}$$

$$C_{n+1} = \overline{\overline{B}_n \overline{C}_n + A_n \overline{C}_n + A_n \overline{B}_n}$$

③ 写逻辑表达式。

2.当用异或门实现电路时，
写出相应的函数式为:

$$D_n = A_n \oplus B_n \oplus C_n$$

$$\begin{aligned} C_{n+1} &= \overline{A}_n \overline{B}_n C_n + \overline{A}_n B_n \overline{C}_n + B_n C_n \\ &= \overline{A}_n (B_n \oplus C_n) + B_n C_n \\ &= \overline{\overline{A}_n (B_n \oplus C_n)} \cdot \overline{B_n C_n} \end{aligned}$$

C_n \ A_nB_n	00	01	11	10
0	0	1	0	0
1	1	1	1	0

C_{n+1}

C_n \ A_nB_n	00	01	11	10
0	0	1	0	1
1	1	0	1	0

D_n

④ 画出逻辑电路图

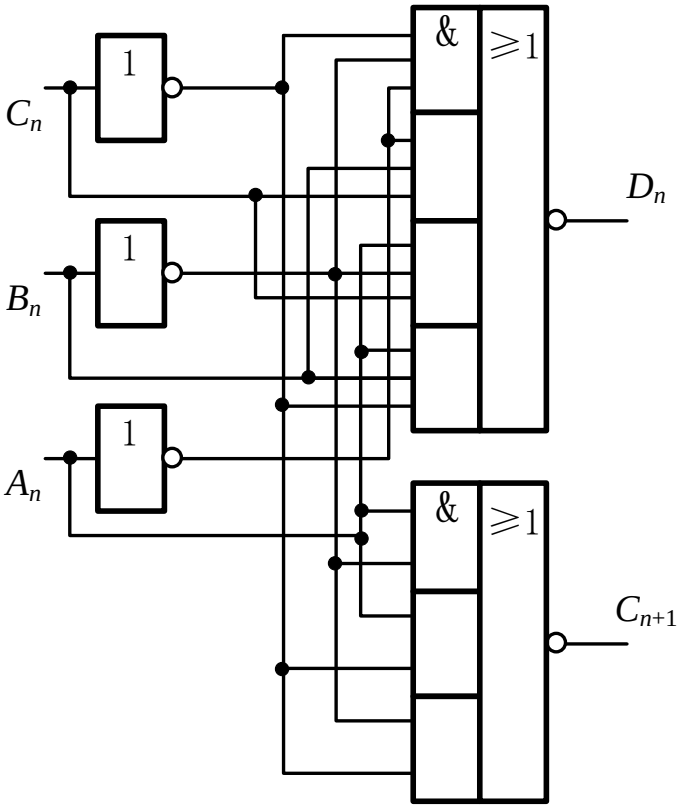
(转下页)

$$D_n = \overline{\overline{A}_n \overline{B}_n \overline{C}_n + \overline{A}_n B_n C_n + A_n B_n \overline{C}_n + A_n \overline{B}_n C_n}$$

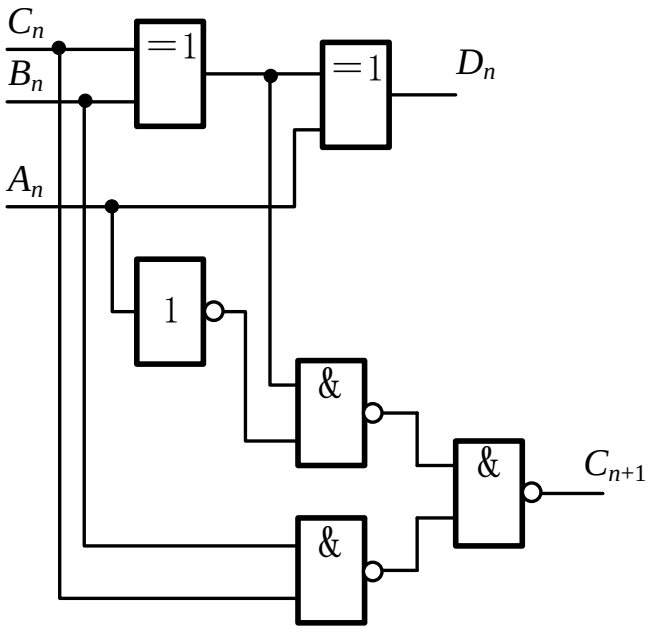
$$C_{n+1} = \overline{\overline{B}_n \overline{C}_n + A_n \overline{C}_n + A_n \overline{B}_n}$$

$$D_n = A_n \oplus B_n \oplus C_n$$

$$\begin{aligned} C_{n+1} &= \overline{A}_n \overline{B}_n C_n + \overline{A}_n B_n \overline{C}_n + B_n C_n \\ &= \overline{A}_n (B_n \oplus C_n) + B_n C_n \\ &= \overline{\overline{A}_n (B_n \oplus C_n) \cdot \overline{B_n C_n}} \end{aligned}$$



(a)



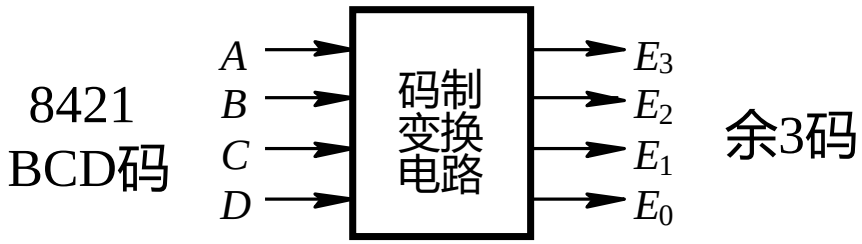
(b)

示例

例4:用门电路设计一个将8421 BCD码转换为余3码的变换电路。

① 分析题意，逻辑抽象列真值表。

该电路输入为8421 BCD码，输出为余3码，因此它是一个四输入、四输出的码制变换电路。



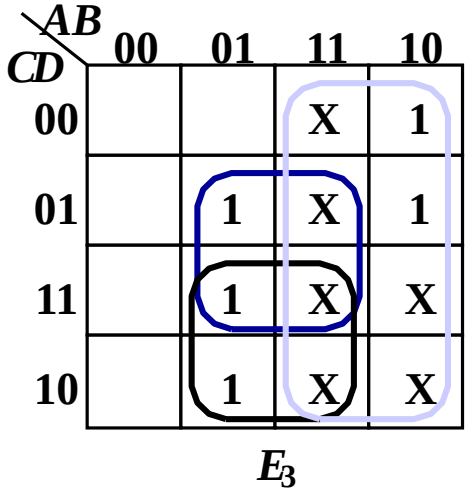
根据两种BCD码的编码关系，列出真值表。由于8421 BCD码不会出现1010~1111这六种状态， 因此把它视为无关项。

A	B	C	D	E_3	E_2	E_1	E_0
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	×	×	×	×
1	0	1	1	×	×	×	×
1	1	0	0	×	×	×	×
1	1	0	1	×	×	×	×
1	1	1	0	×	×	×	×
1	1	1	1	×	×	×	×

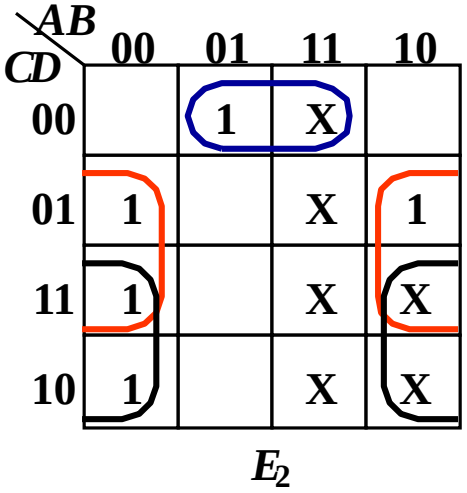
② 选择器件，写出输出函数表达式。

题目没有具体指定用哪一种门电路，因此可以从门电路的数量、种类、速度等方面综合折衷考虑，选择最佳方案。

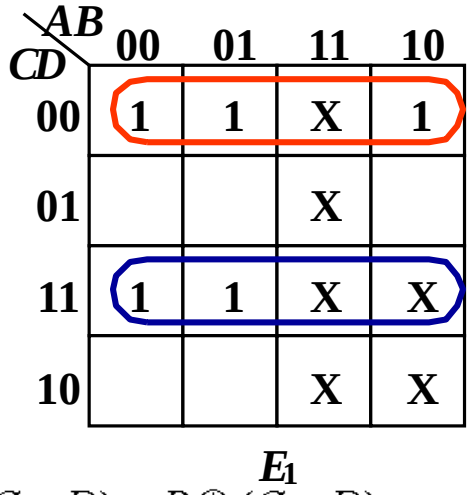
A	B	C	D	E_3	E_2	E_1	E_0
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	×	×	×	×
1	0	1	1	×	×	×	×
1	1	0	0	×	×	×	×
1	1	0	1	×	×	×	×
1	1	1	0	×	×	×	×
1	1	1	1	×	×	×	×



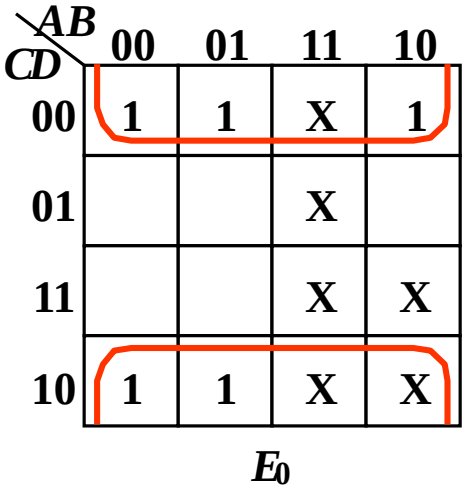
E_3



E_2



E_1



E_0

画K图、写表达式：

$$E_3 = A + BC + BD = \overline{\overline{A} \cdot \overline{BC} \cdot \overline{BD}}$$

$$E_2 = B\overline{C}\,\overline{D} + \overline{B}C + \overline{B}D = B(\overline{C+D}) + \overline{B}(C+D) = B \oplus (C+D)$$

$$E_1 = \overline{C}\,\overline{D} + CD = C \otimes D = C \oplus \overline{D}$$

$$E_0 = \overline{D}$$

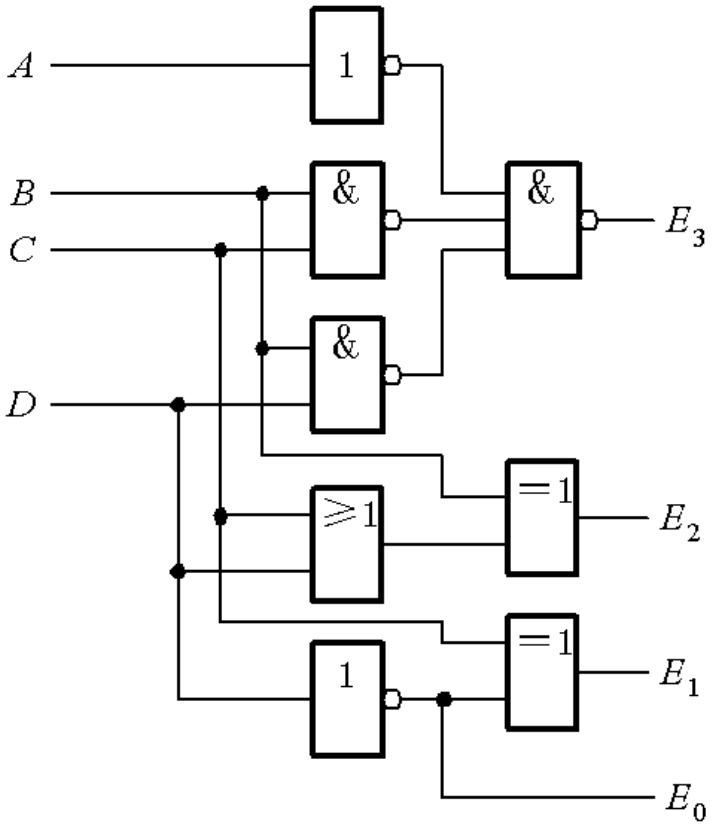
③ 画逻辑电路。

$$E_3 = A + BC + BD = \overline{\overline{A} \cdot \overline{BC} \cdot \overline{BD}}$$

$$E_2 = B\overline{C}\,\overline{D} + \overline{B}C + \overline{B}D = B\overline{(C + D)} + \overline{B}(C + D) = B \oplus (C + D)$$

$$E_1 = \overline{C}\,\overline{D} + CD = C \otimes D = C \oplus \overline{D}$$

$$E_0 = \overline{D}$$



8421 BCD码转换为余3码的电路

组合电路设计的小结

1）正确建立给定问题的逻辑描述是关键；

2）工程考量，指标兼顾：电路简单，器件多见门类少，级数少，功耗小等；

3）不同的逻辑表达式可能功能相同，如

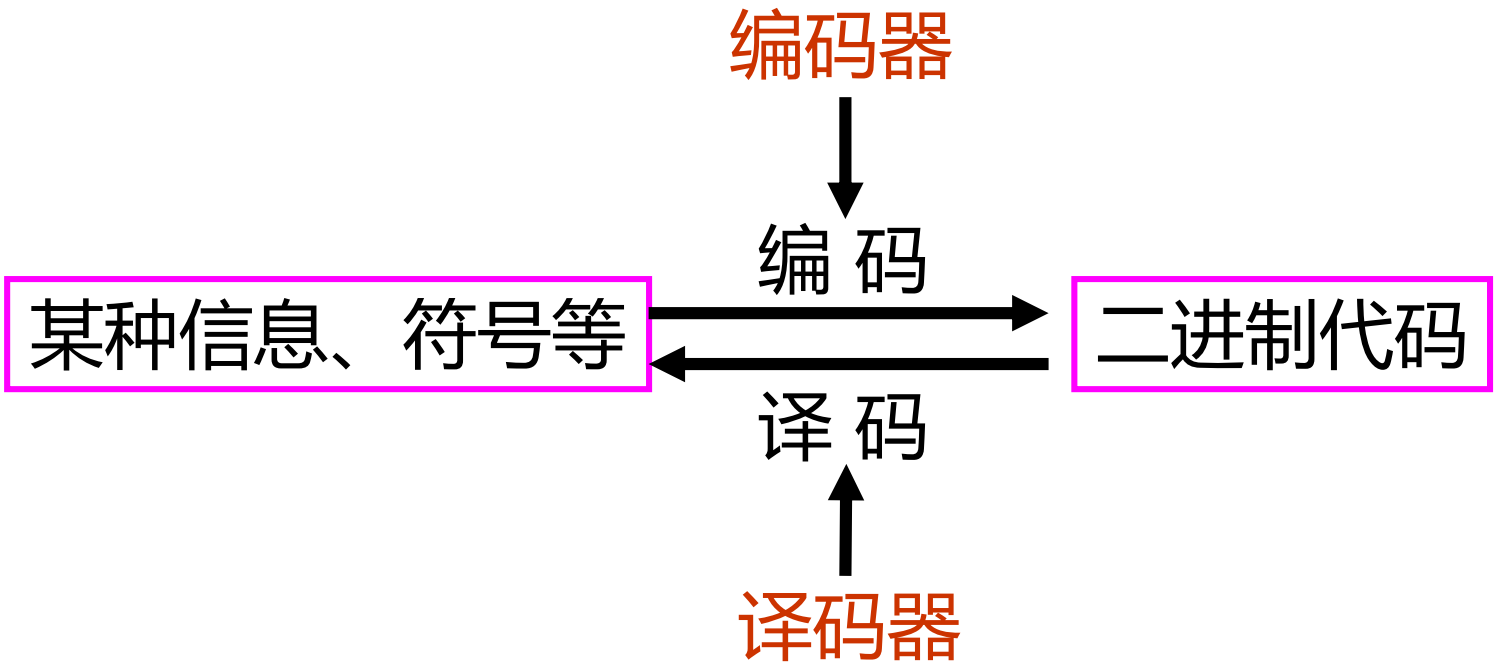
$$F(A,B,C,D)=B\overline{C}+\overline{A}\overline{B}C+\overline{A}BD$$

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

常见组合逻辑电路

- 加法器
  全加器、半加器；一位、多位预算
- 减法器
  全减器、半减器；一位、多位运算
- 译码器
- 编码器
- 数据选择器
- 数据分配器
- 数码比较器

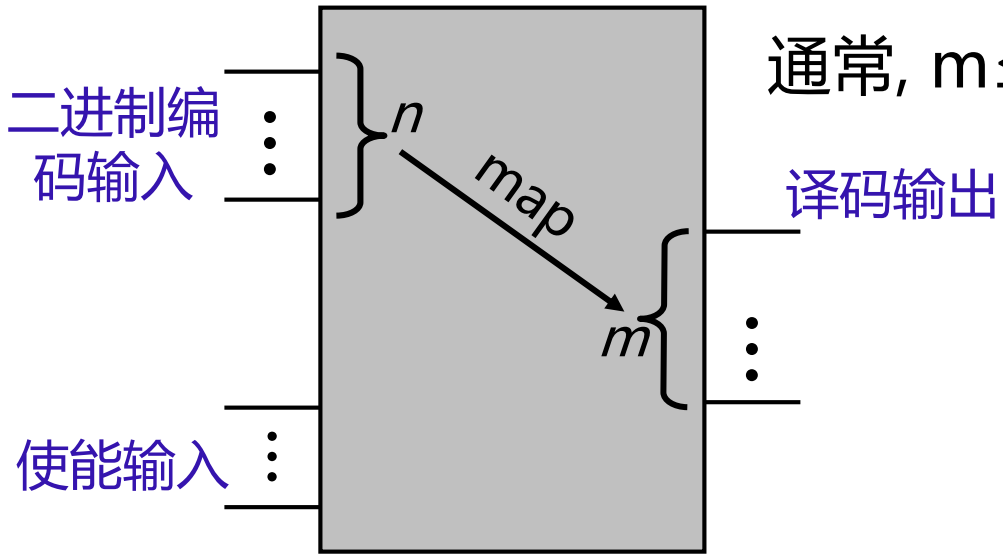
编码变换



全班有62名同学，需几位二进制代码才能表示?

译码器

译码是将某个二进制编码翻译成电路的某种状态，是将输入的某个二进制编码与电路输出的某种状态相对应。



通常, $m \leq 2^n$

MSI译码器的基本结构

分类：

- 二进制译码器
- BCD译码器
- 显示译码器

（1）二进制译码器（Binary decoder）

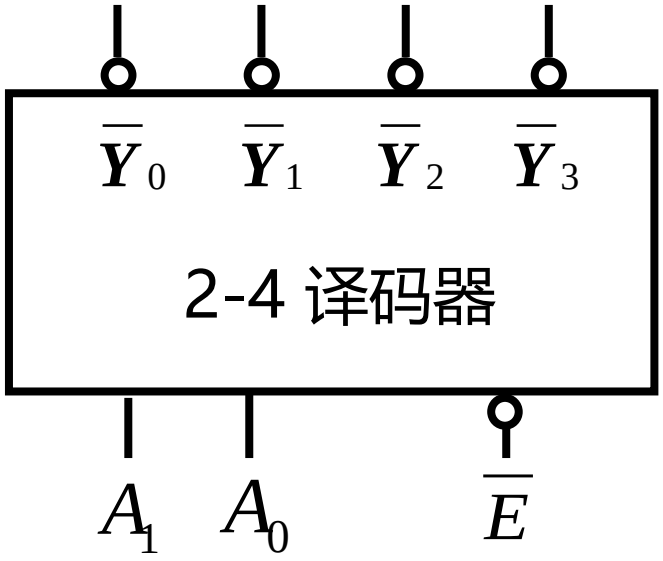
将n个输入的组合码译成2^n种电路状态。也叫n---2^n译码器

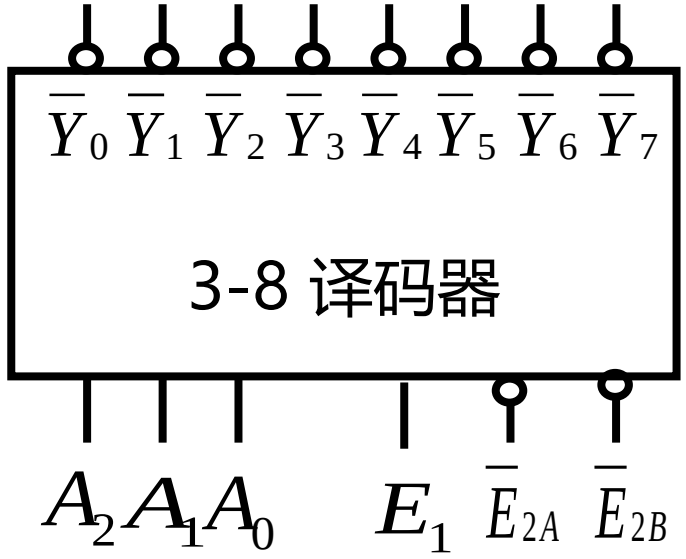
。

译码器的输入：一组二进制代码

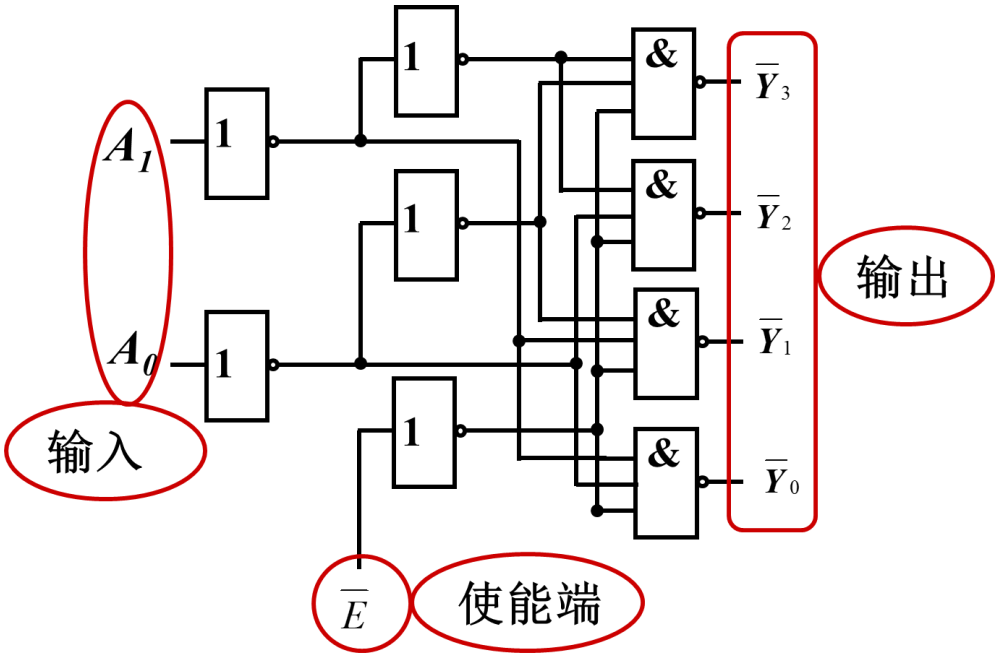
译码器的输出：一组高低电平信号

常用二进制译码器举例：





2-4译码器



当E=0时，2-4译码器的输出函数分别为:

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

$$Y_2 = \overline{A_1\overline{A_0}} \qquad Y_3 = \overline{A_1A_0}$$

$$Y_i = \overline{m_i}$$

考虑E，输出函数有:

E	Yi
0	$\overline{m_i}$
1	1

功能表

$\overline{E}$	A_1	A_0	$\overline{Y}_0$	$\overline{Y}_1$	$\overline{Y}_2$	$\overline{Y}_3$
1	×	×	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

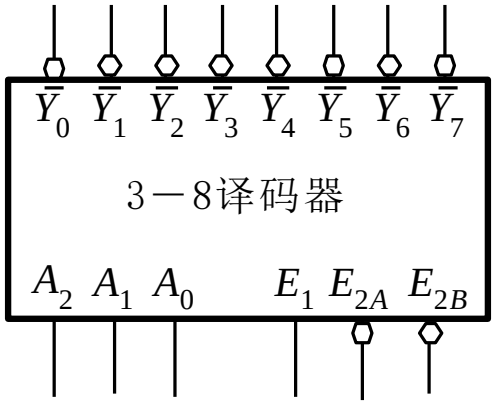
$$Y_i = \overline{E\,\overline{m_i}} + E = E + \overline{m_i} = \overline{\overline{E}m_i}$$

$$Y_i = \overline{\overline{E}m_i}(i = 0,1,2,3)$$

*3-8译码器74LS138

3-8译码器逻辑符号

译码输出，低电平有效



A_2 A_1 A_0

译码地址输入端

使能输入端

E_1 =1

E_{2A}=0

E_{2B}=0

功能表

E_1	$E_{2A}+E_{2B}$	A_2	A_1	A_0	$\overline{Y}_0$	$\overline{Y}_1$	$\overline{Y}_2$	$\overline{Y}_3$	$\overline{Y}_4$	$\overline{Y}_5$	$\overline{Y}_6$	$\overline{Y}_7$
0	×	×	×	×	1	1	1	1	1	1	1	1
×	1	×	×	×	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1	1	1
1	0	1	0	0	1	1	1	1	0	1	1	1
1	0	1	0	1	1	1	1	1	1	0	1	1
1	0	1	1	0	1	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	1	1	1	1	0

$Y_0=\overline{\overline{A}_2\overline{A}_1\overline{A}_0}$

$Y_1=\overline{\overline{A}_2\overline{A}_1A_0}$

$Y_2=\overline{\overline{A}_2A_1\overline{A}_0}$

$Y_3=\overline{\overline{A}_2A_1A_0}$

$Y_4=\overline{A_2\overline{A}_1\overline{A}_0}$

$Y_5=\overline{A_2\overline{A}_1A_0}$

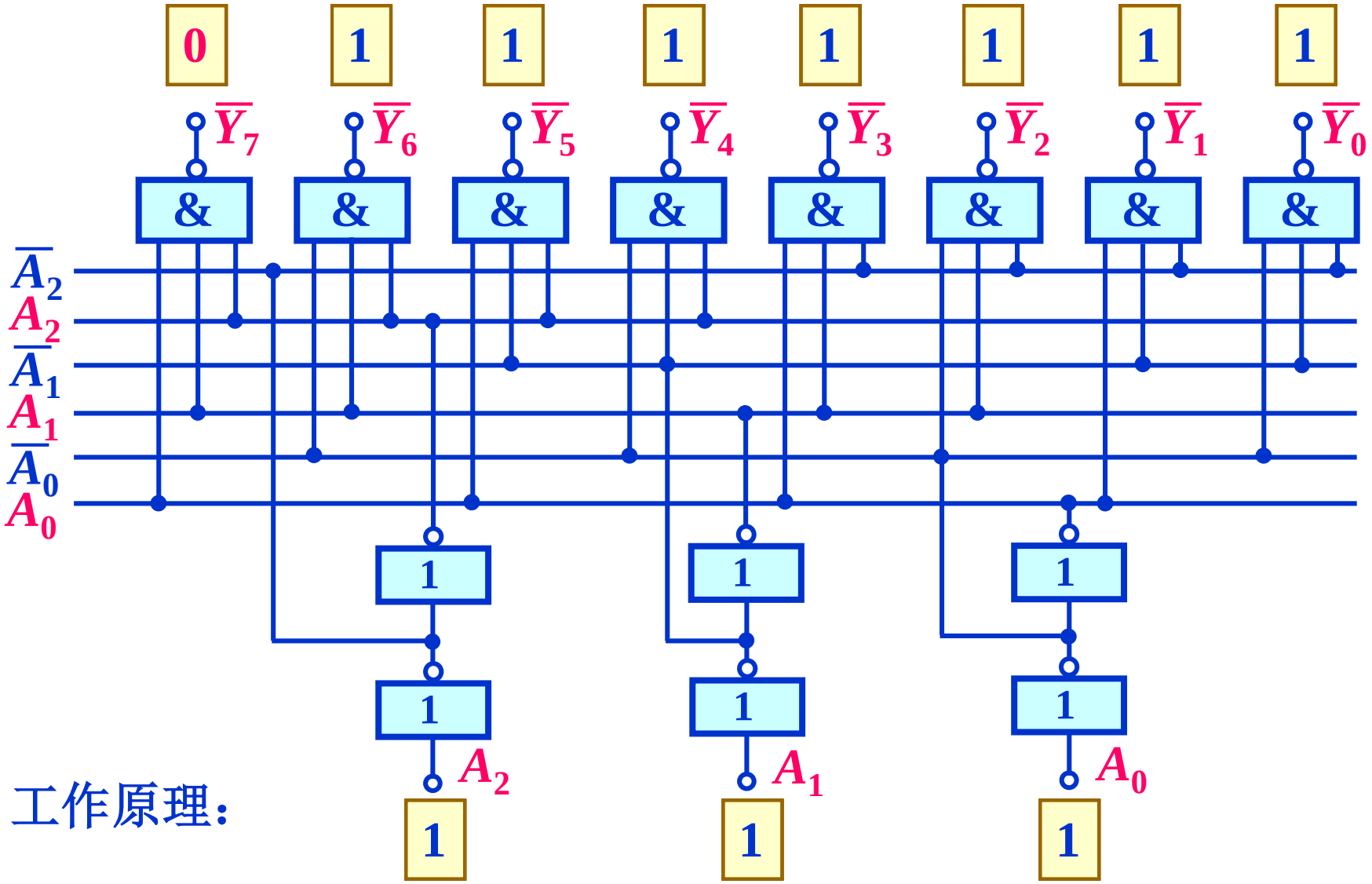
$Y_6=\overline{A_2A_1\overline{A}_0}$

$Y_7=\overline{A_2A_1A_0}$

$$Y_i=\overline{Em_i} \qquad (i=0\sim 7)$$

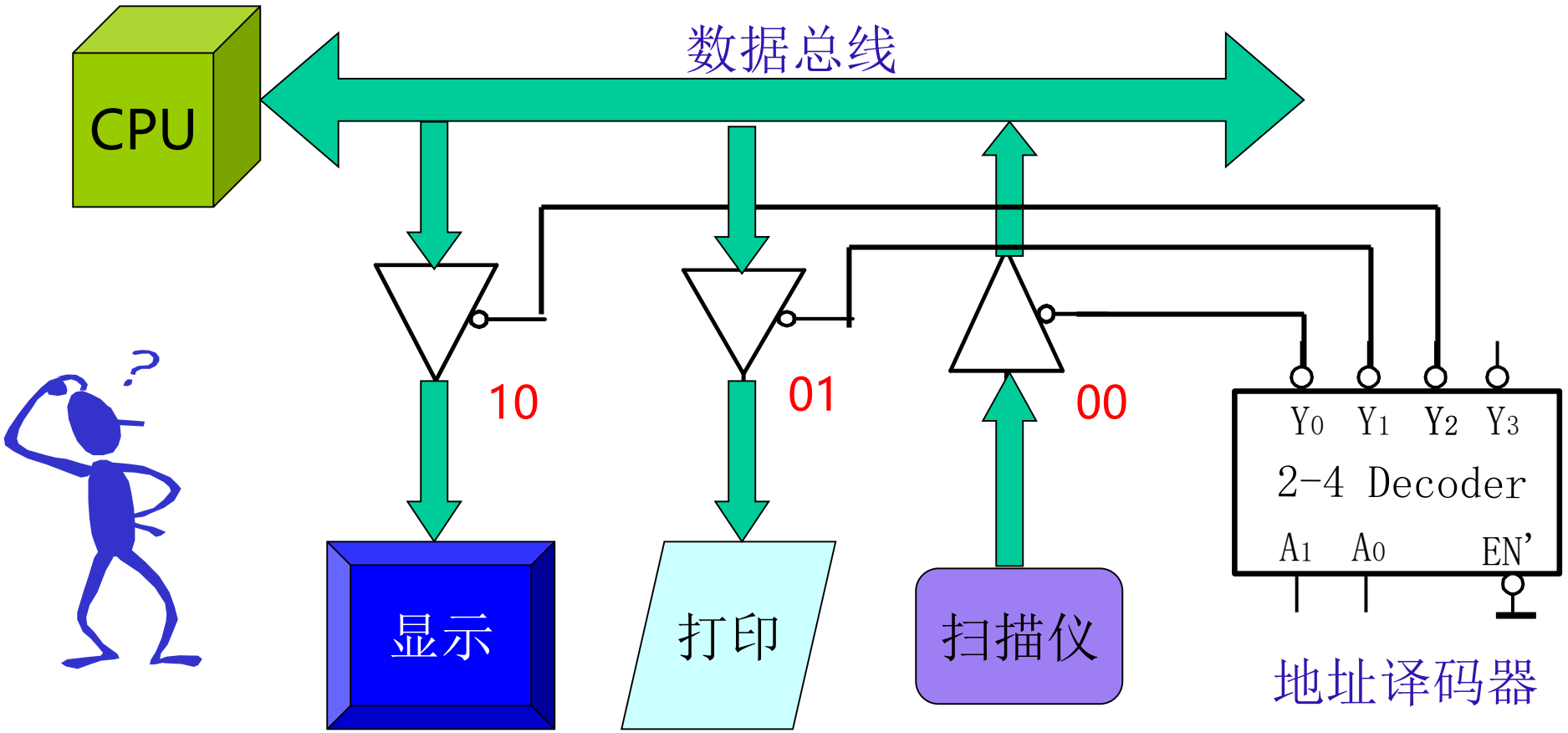
$$E=E_1\cdot\overline{E_{2A}+E_{2B}}=E_1\cdot\overline{E_{2A}}\cdot\overline{E_{2B}}$$

*3-to-8 译码器内部等效电路——低电平输出有效

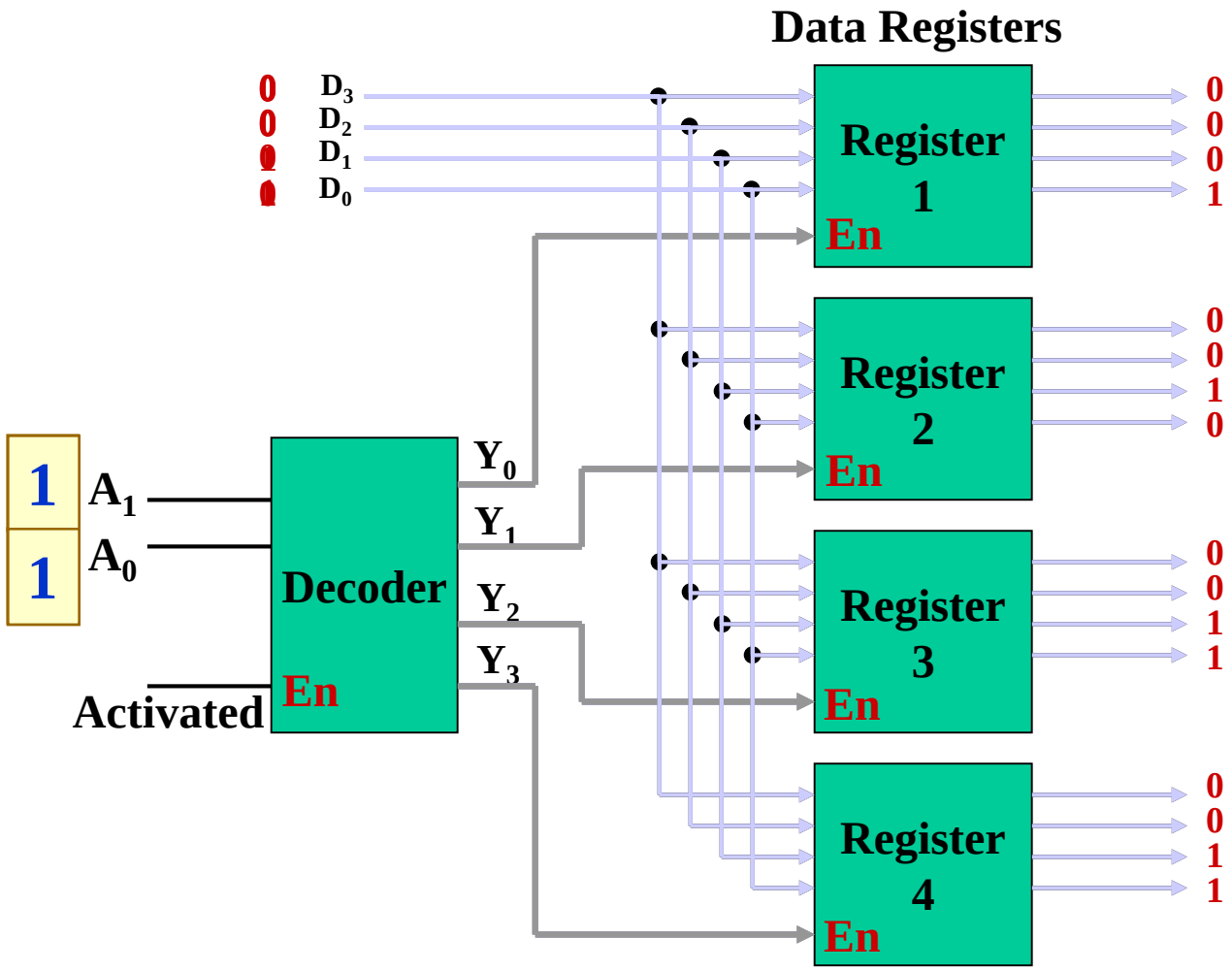


工作原理：

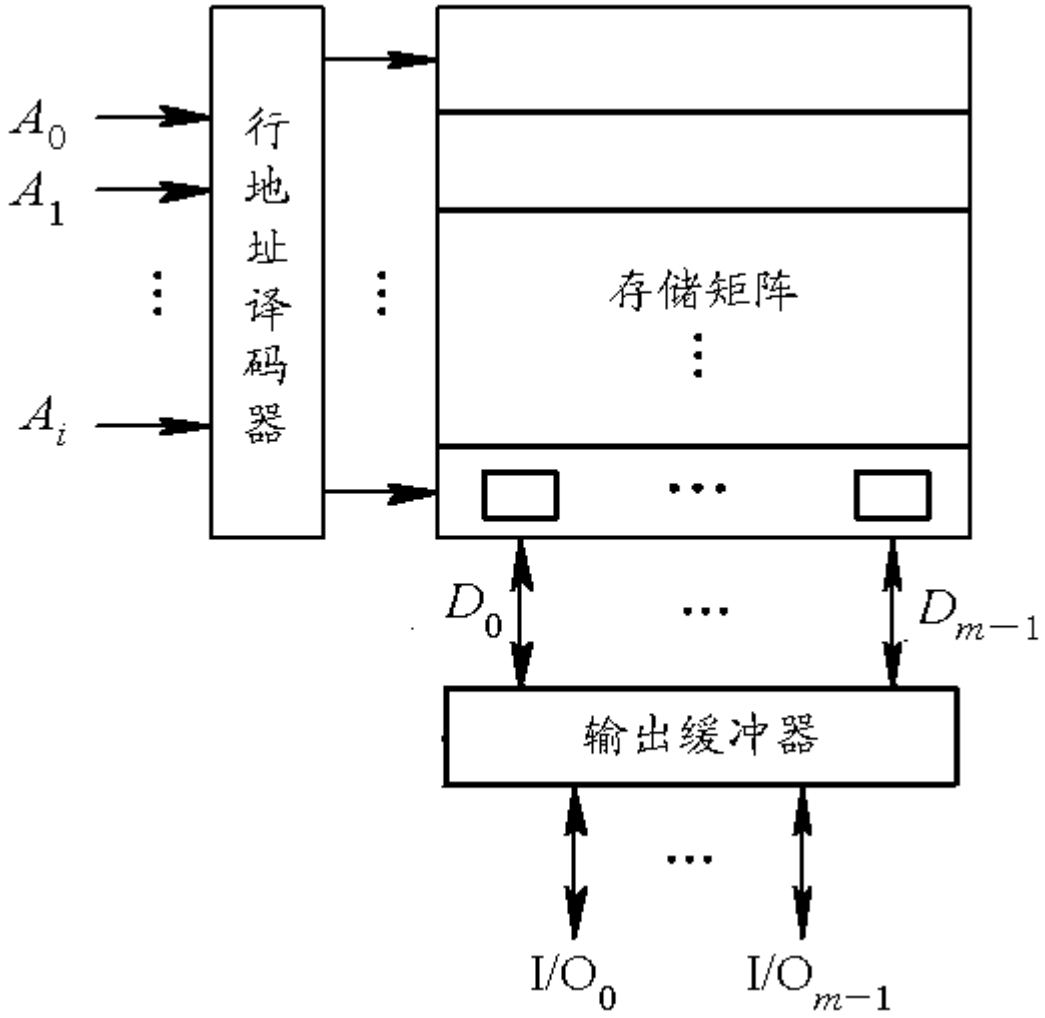
译码器应用： 计算机外设地址译码



译码器应用： 寄存器的地址译码

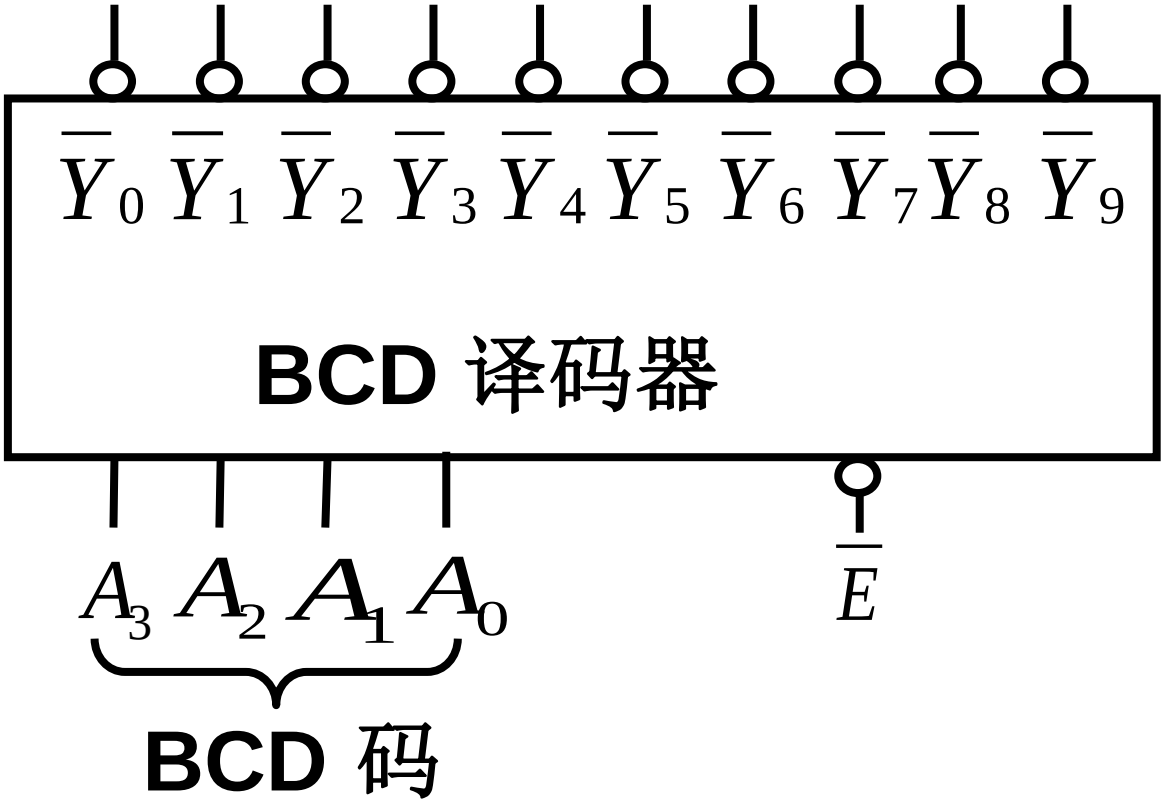


译码器应用： 存储空间地址译码



*(2) 二-十进制译码器（BCD译码器）

将输入的一位BCD码（四位二进制数）译成10种不同的电路状态。

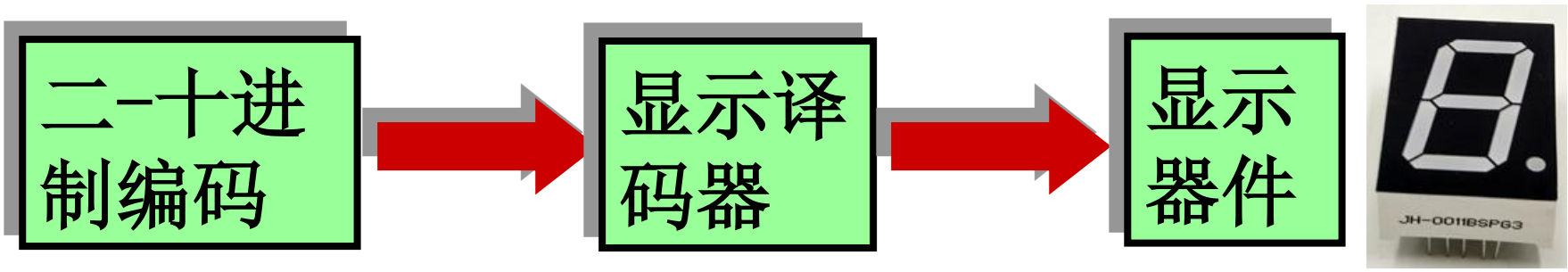


*（3）显示译码器

在数字系统中，常常需要将运算结果用人们习惯的十进制显示出来，这就要用到显示译码器。

显示译码器是用来驱动显示器件，以显示数字或字符的MSI部件。

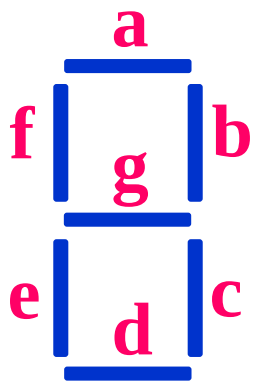
显示译码器随显示器件的类型而异，与辉光数码管相配的是BCD十进制译码器，而常用的发光二极管(LED)数码管、液晶数码管、荧光数码管等是由7个或8个字段构成字形的，因而与之相配的有BCD七段或BCD八段显示译码器。



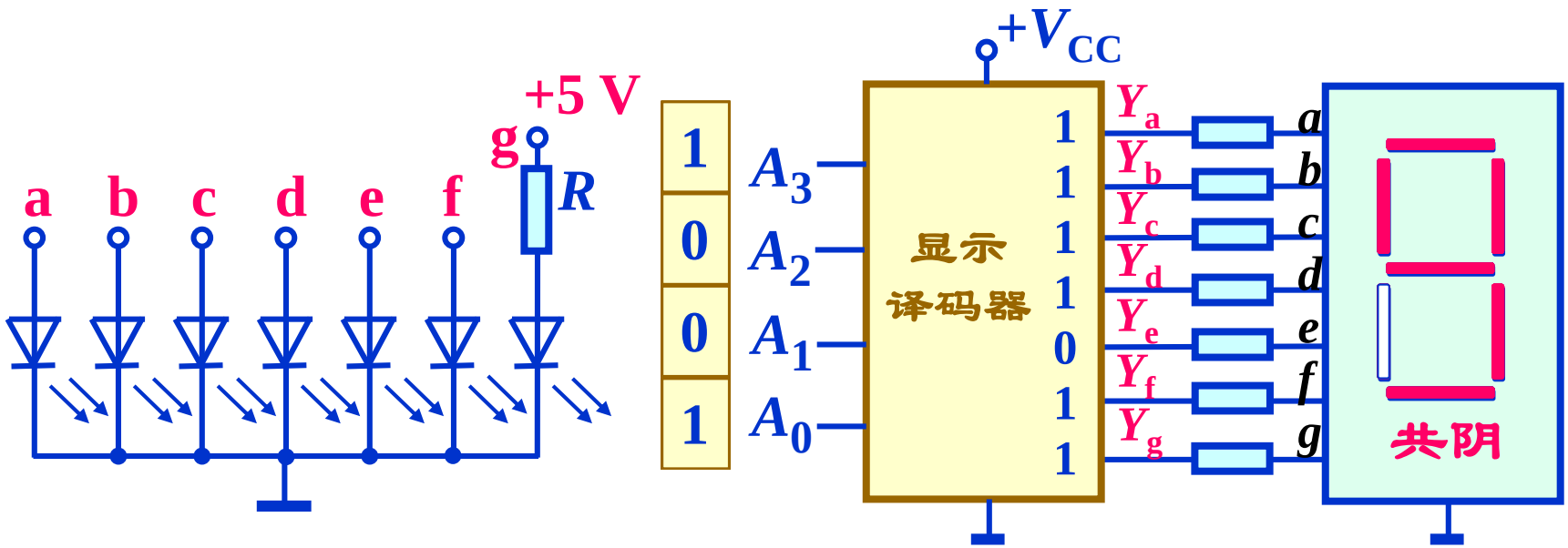
*（3）显示译码器

Display Decoder

——Seven-segment display



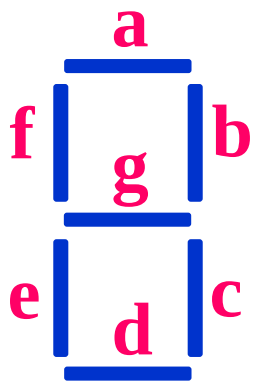
Common Cathode (共阴极)



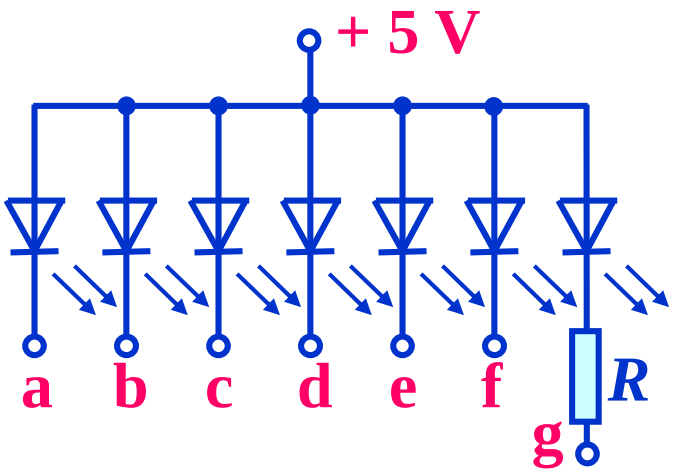
*（3）显示译码器

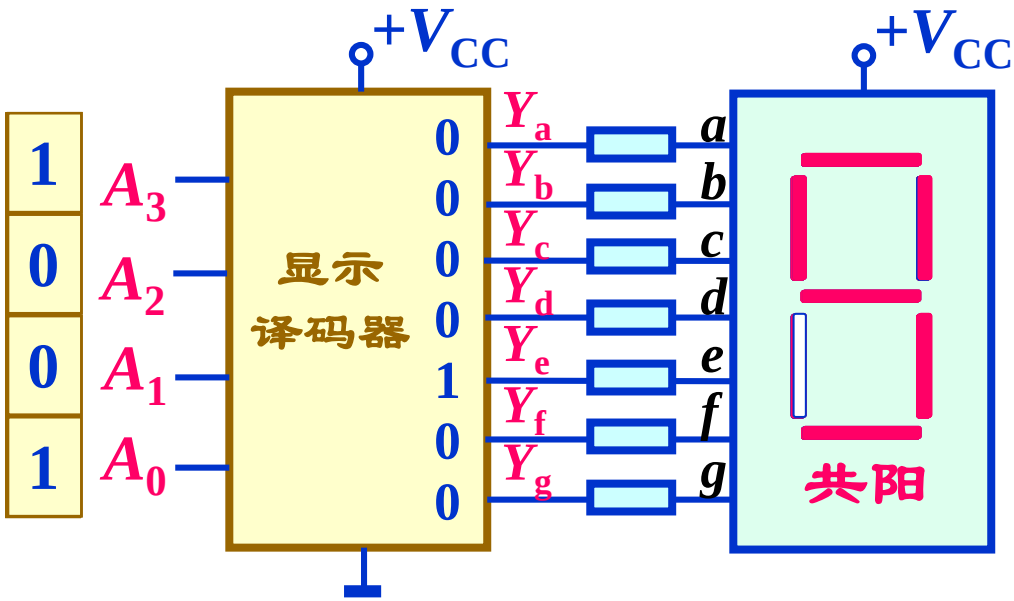
Display Decoder

——Seven-segment display



Common Anode (共阳极)





*共阴极8421BCD七段译码器真值表

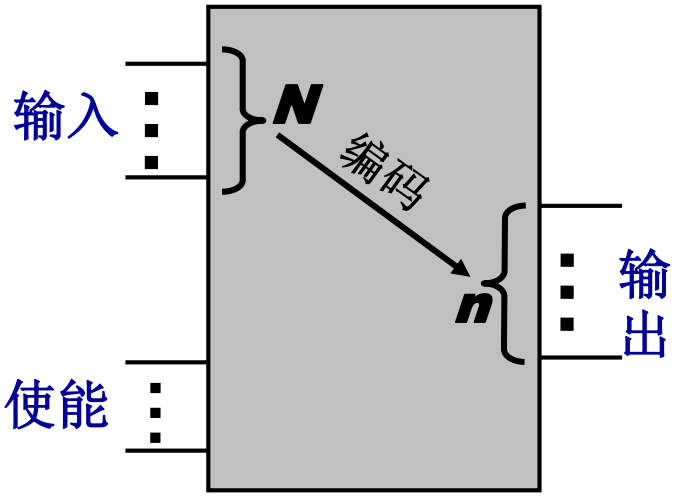
输入				输出							字形
D	C	B	A	F_a	F_b	F_c	F_d	F_e	F_f	F_g	
0	0	0	0	1	1	1	1	1	1	0	0
0	0	0	1	0	1	1	0	0	0	0	1
0	0	1	0	1	1	0	1	1	0	1	2
0	0	1	1	1	1	1	1	0	0	1	3
0	1	0	0	0	1	1	0	0	1	1	4
0	1	0	1	1	0	1	1	0	1	1	5
0	1	1	0	1	0	1	1	1	1	1	6
0	1	1	1	1	1	1	0	0	0	0	7
1	0	0	0	1	1	1	1	1	1	1	8
1	0	0	1	1	1	1	1	0	1	1	9

编码器

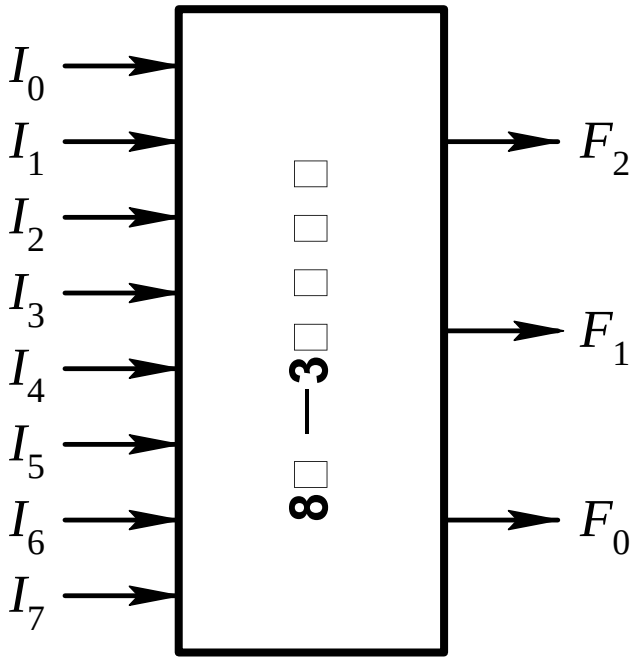
- 在数字电路中用二进制代码表示有关的信号称为二进制编码，实现编码操作的电路就是编码器。
  - 二进制编码器
  - **BCD编码器**
  - 优先编码器

(1)二进制编码器

- 用n位二进制代码对$N=2^n$个一般信号进行编码的电路，叫做二进制编码器。



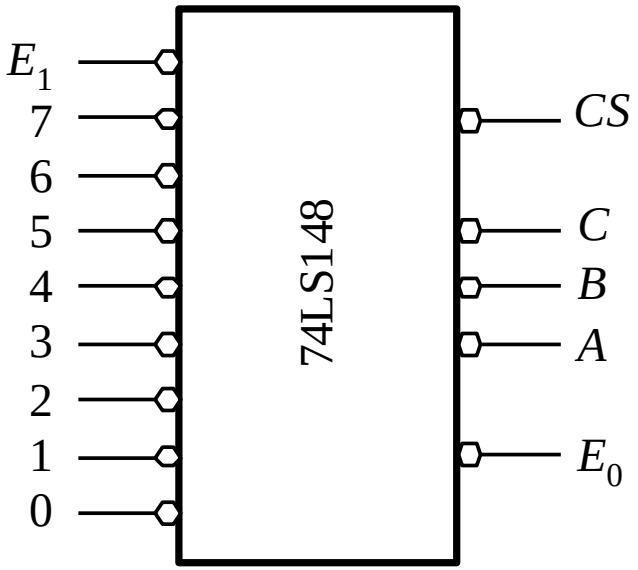
*8线—3线编码器



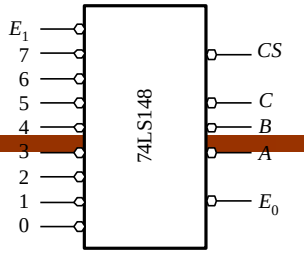
输入								输出		
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	F_2	F_1	F_0
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

*优先编码器

- 与普通编码器不同，优先编码器允许多个输入信号同时有效，但它只按其中优先级别最高的有效输入信号编码，对级别较低的输入信号不予理睬。
- 优先编码器常用于优先中断系统和键盘编码。
- 常用的优先编码器有10线-4线、8线-3线。



*8-3优先编码器

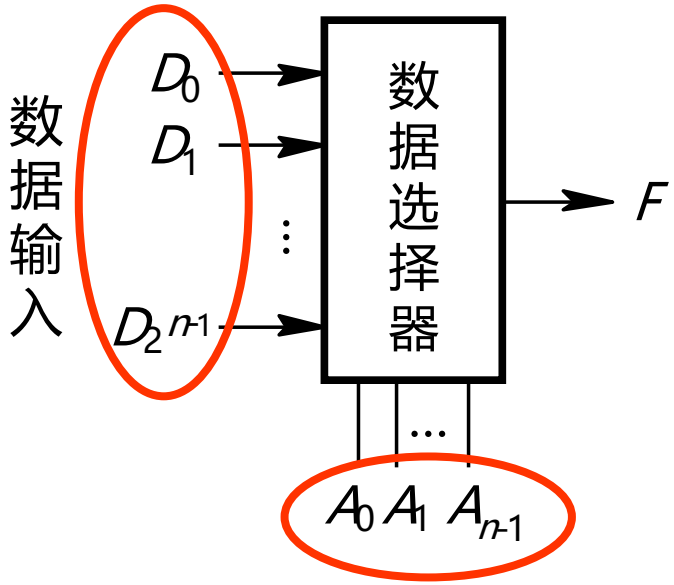


No	输入									输出			
	E_1	7	6	5	4	3	2	1	0	C	B	A	CS	E_0
1	1	×	×	×	×	×	×	×	×	1	1	1	1	1
2	0	1	1	1	1	1	1	1	1	1	1	1	1	0
3	0	0	×	×	×	×	×	×	×	0	0	0	0	1
4	0	1	0	×	×	×	×	×	×	0	0	1	0	1
5	0	1	1	0	×	×	×	×	×	0	1	0	0	1
6	0	1	1	1	0	×	×	×	×	0	1	1	0	1
7	0	1	1	1	1	0	×	×	×	1	0	0	0	1
8	0	1	1	1	1	1	0	×	×	1	0	1	0	1
9	0	1	1	1	1	1	1	0	×	1	1	0	0	1
10	0	1	1	1	1	1	1	1	0	1	1	1	0	1

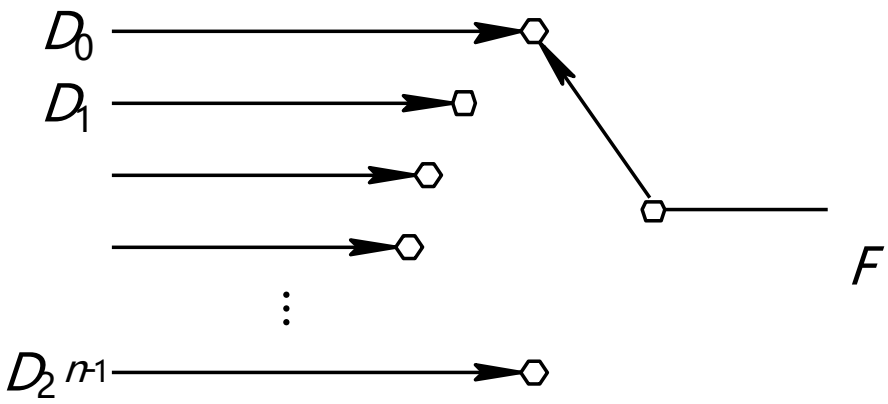
数据选择器（Multiplexer，简称MUX）

数据选择器又称多路选择器。它有n位地址输入、2n位数据输入、1位输出。

每次在地址输入的控制下，从多路输入数据中选择一路输出，其功能类似于一个单刀多掷开关。

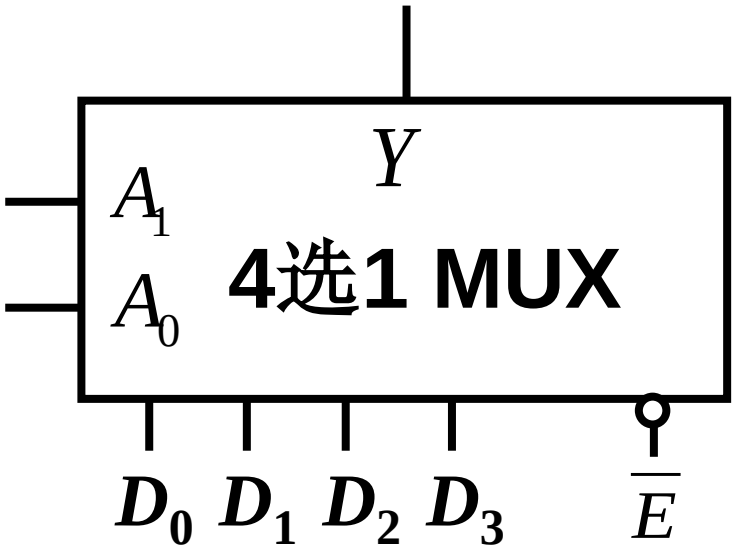


数据选择器框图

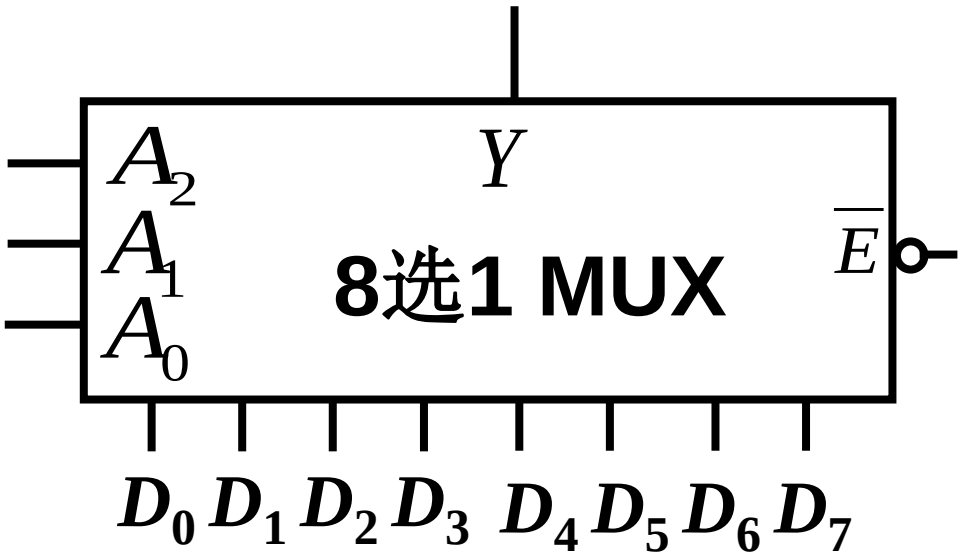


数据选择器等效开关

常用数据选择器举例



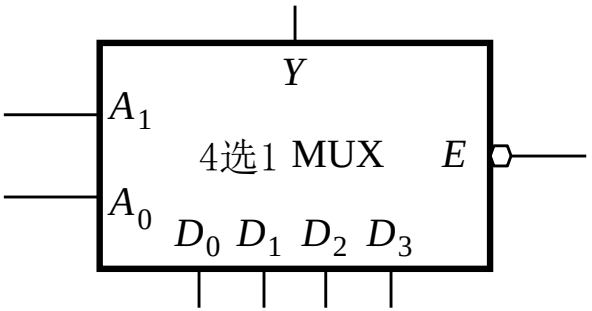
74LS153



74LS151

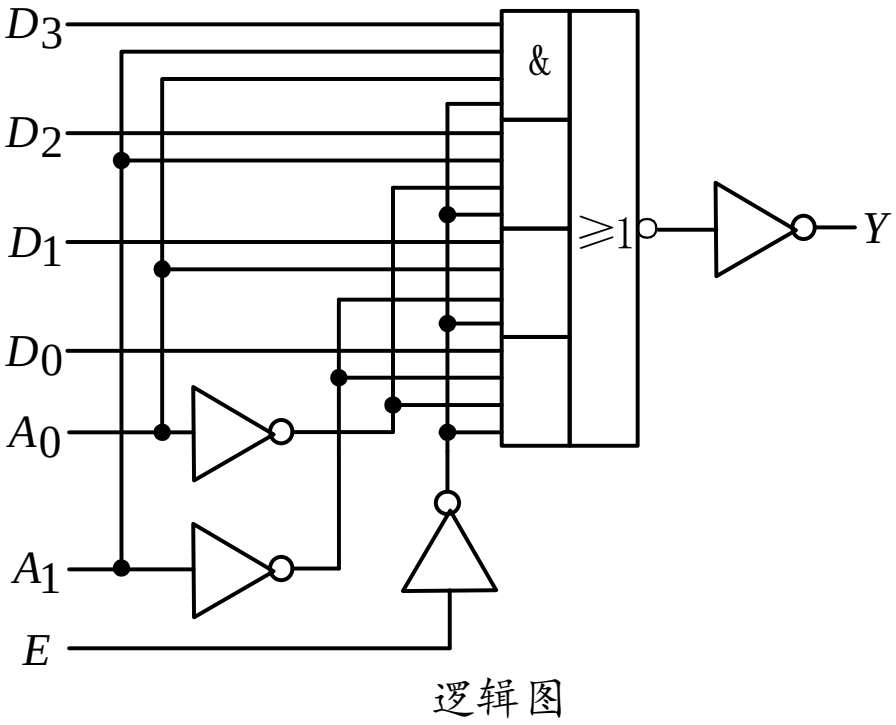
*4选1数据选择器

4选1 MUX的逻辑符号



4选1 MUX功能表

E	A_1	A_0	Y
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	×	×	0



逻辑图

当E=0时，4选1 MUX的逻辑功能还可以用表达式表示：

$$Y=\overline{A}_1\overline{A}_0D_0+\overline{A}_1A_0D_1+A_1\overline{A}_0D_2+A_1A_0D_3=\sum_{i=0}^{3}m_iD_i$$

数据选择器的应用

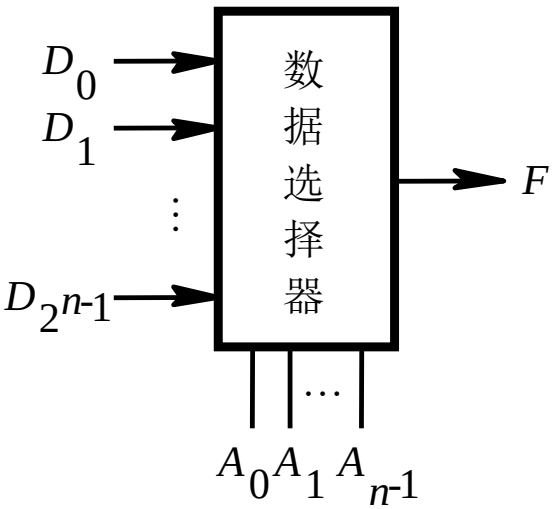
数据选择器的应用很广，典型应用有：

① 作数据选择，以实现多路信号分时传送。

② 实现组合逻辑函数。

③ 在数据传输时实现并—串转换。

④ 产生序列信号。



比较器

*（1）一位数值比较器

功能表

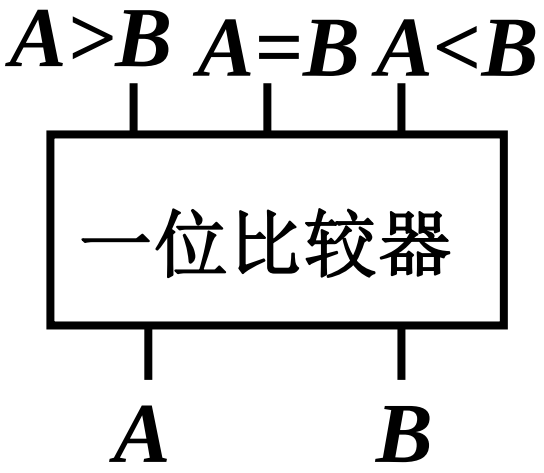
输入		输出		
A	B	$A>B$	$A=B$	$A<B$
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

$$“A>B”=A\overline{B}$$

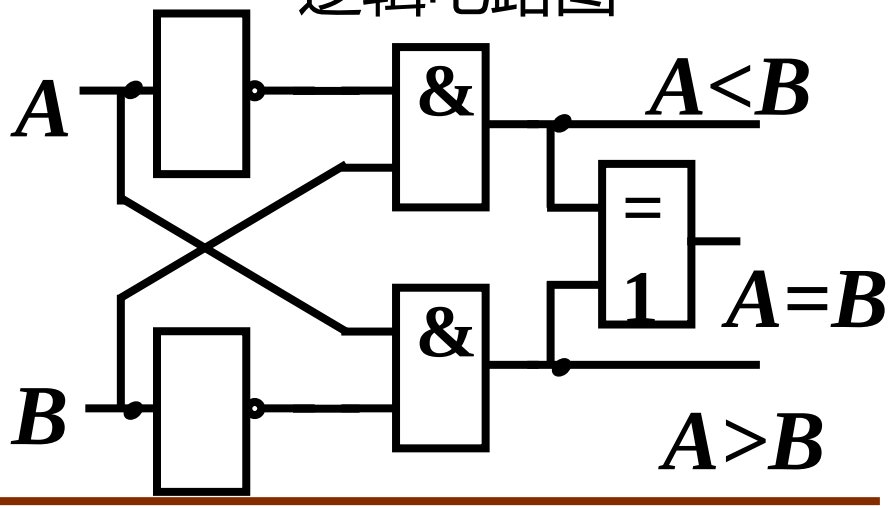
$$“A=B”=\overline{A}\,\overline{B}+AB$$

$$“A<B”=\overline{A}B$$

逻辑符号



*逻辑电路图



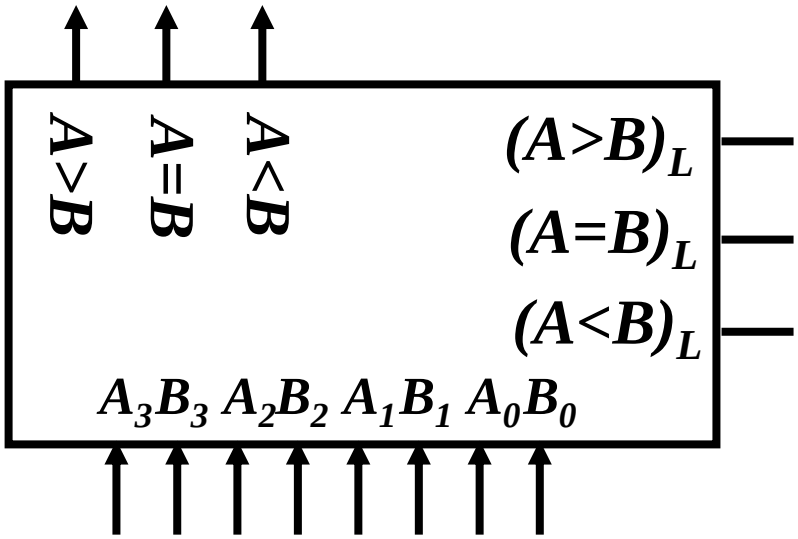
*比较器

* （2）四位数值比较器

比较原则：

A. 先从高位比起,高位大的数值一定大。

B. 若高位相等,则再比较低位数,最终结果由低位的比较结果决定。

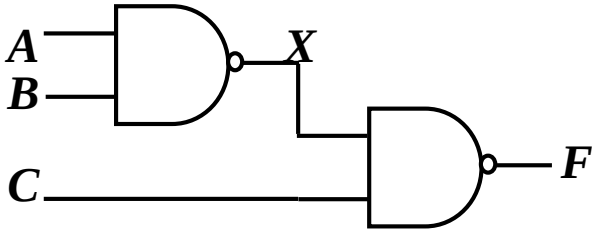


组合逻辑电路中的竞争与冒险

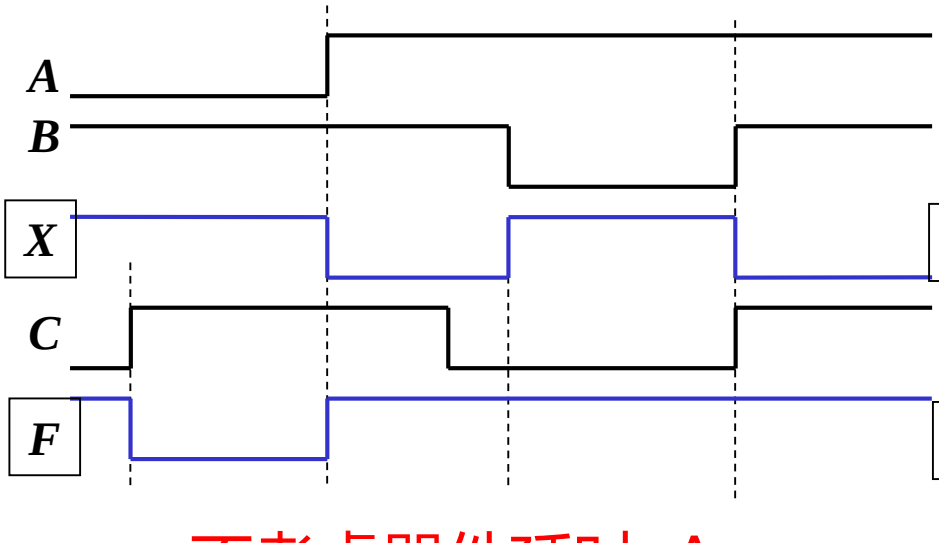
在组合电路中，某一输入变量经不同途径传输后，到达电路中某一会合点的时间有先有后，这种现象称为竞争。

由于竞争而使电路输出发生瞬时错误的现象称为冒险。

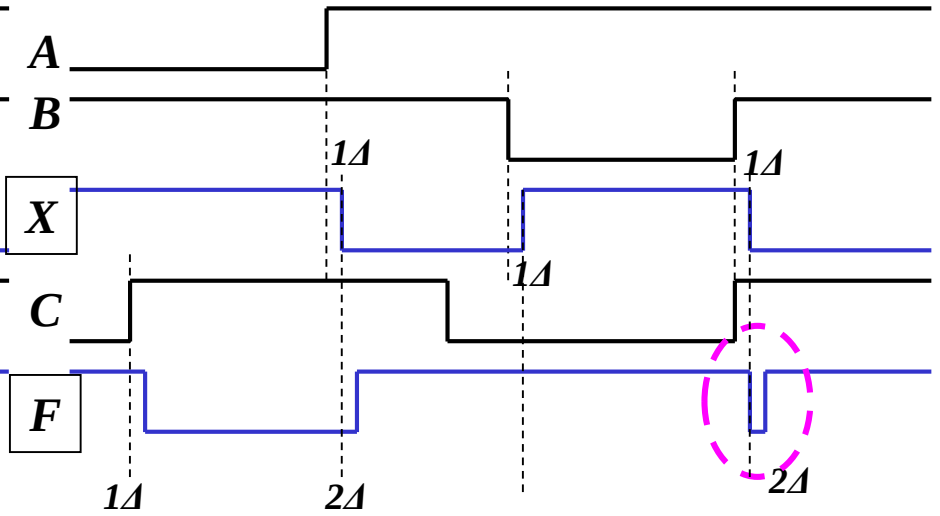
例：



$$F = \overline{\overline{A \bullet B} \bullet C}$$



不考虑器件延时，Δ.



考虑器件延时，Δ.

注意：竞争是经常发生的，但不一定都会产生毛刺。

作业

- 3-3
- *(3-4)
- 3-5（b）
- 3-7
- 3-9

本章完，谢谢大家！