2.4 常用MSI组合逻辑器件

2.4.1 译码器和编码器

- 译码器和编码器的一般结构、二进制译码器、MSI译码器及其级联与应用
- 编码器、优先级编码器、MSI编码器及其应用

2.4.2 数据分配器和多路选择器

- 数据分配器原理
- 多路选择器、MSI多路选择器及其扩展和应用

2.4.3 三态门

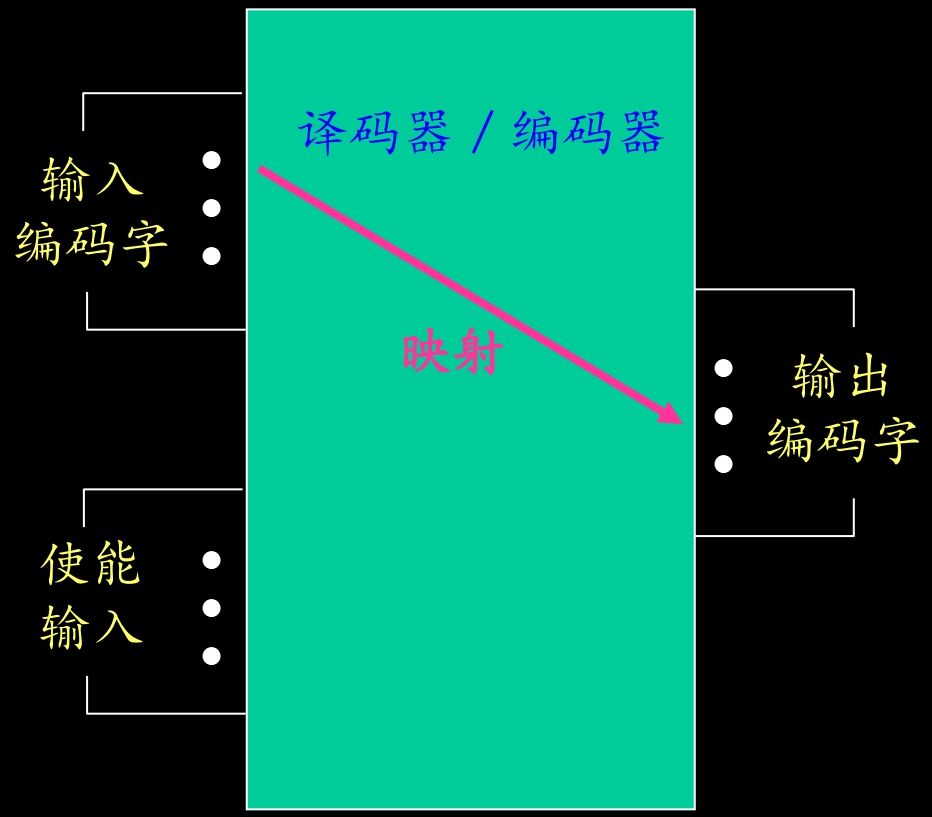
- 三态门、多路收发器和双向收发器

2.4.4 加法器和比较器

- 加法器
- 比较器

2.4.1 译码器与编码器

译码器与编码器的的一般结构如图所示：



- **编码**：将需要处理的任何**信息**（包括数和字符等）转换成符合一定规则的二进制**代码**。
- **译码**：是编码的逆过程，将每一组**代码**所包含的**信息**“翻译”出来。

- 译码器**输入端数n**小于**输出端数m**；反之，**输入端数n**大于**输出端数m**为编码器。

1. 二进制译码器

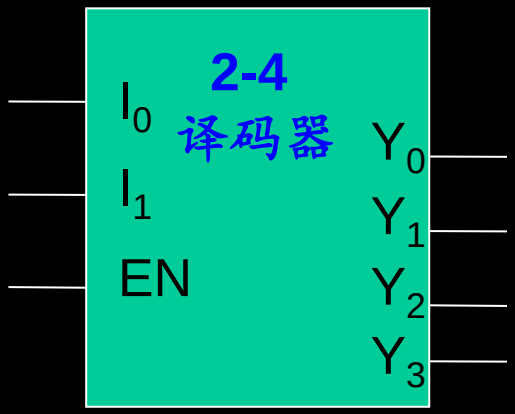
最常用的译码器是二进制译码器。又称为n - 2^n译码器。其中：输入编码为n位二进制数；输出编码为2^n取1码。换句话说，译码器输出为2^n个最小项，所以有时称为最小项发生器。

例 2-4 译码器

输入代码字：I_1、I_0 ；输入使能：EN

输出代码字：Y_3 Y_2 Y_1 Y_0

功能描述：当 EN = 1，输入代码字是i的二进制表示，则输出Y_i（i为十进制数）位为1，其他位均为0。



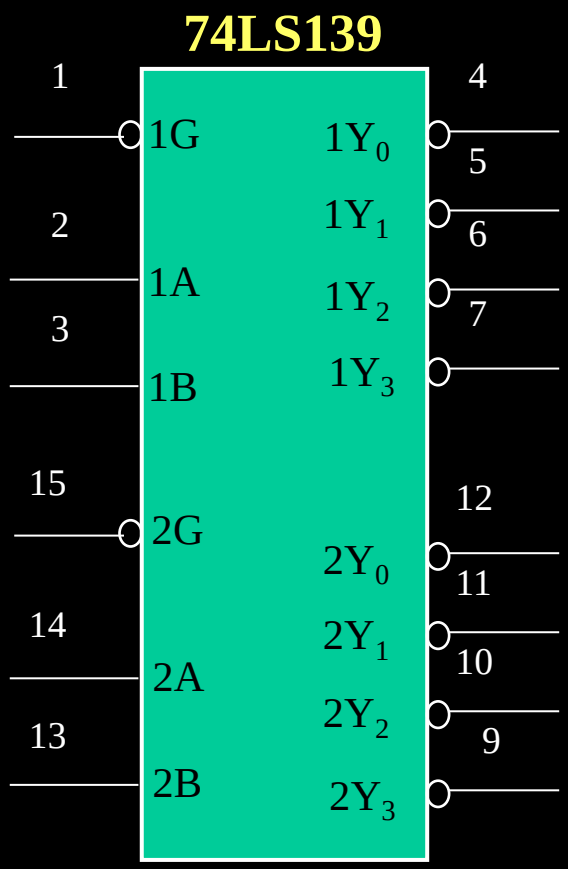
逻辑框图

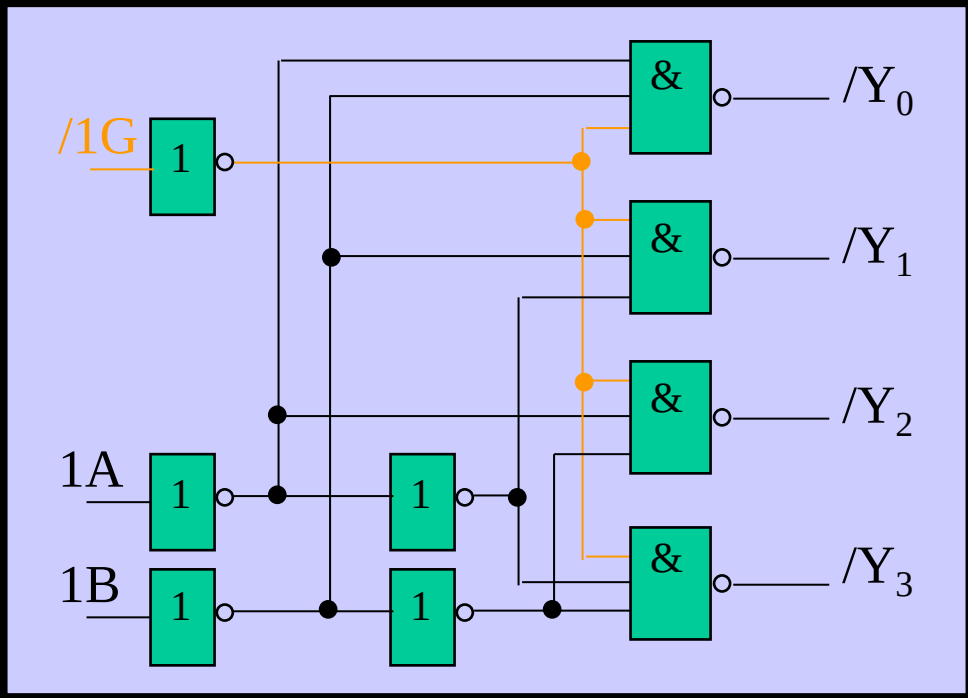
真值表

输 入			输 出			
EN	I_1	I_0	Y_3	Y_2	Y_1	Y_0
0	d	d	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

双2-4译码器74LS139

输 入			输 出			
/G	B	A	$/Y_3$	$/Y_2$	$/Y_1$	$/Y_0$
1	d	d	1	1	1	1
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1

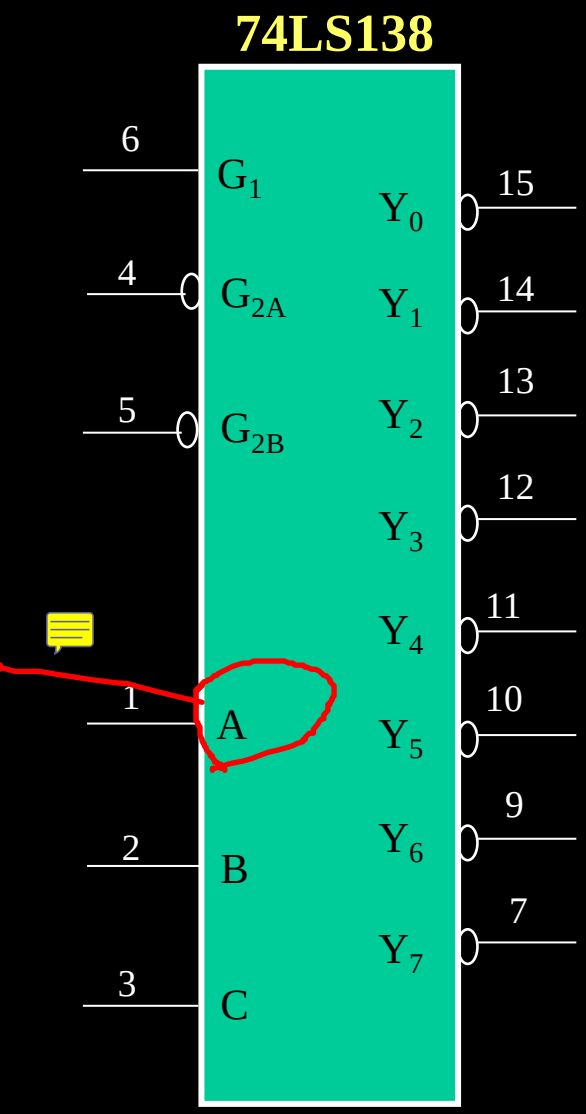


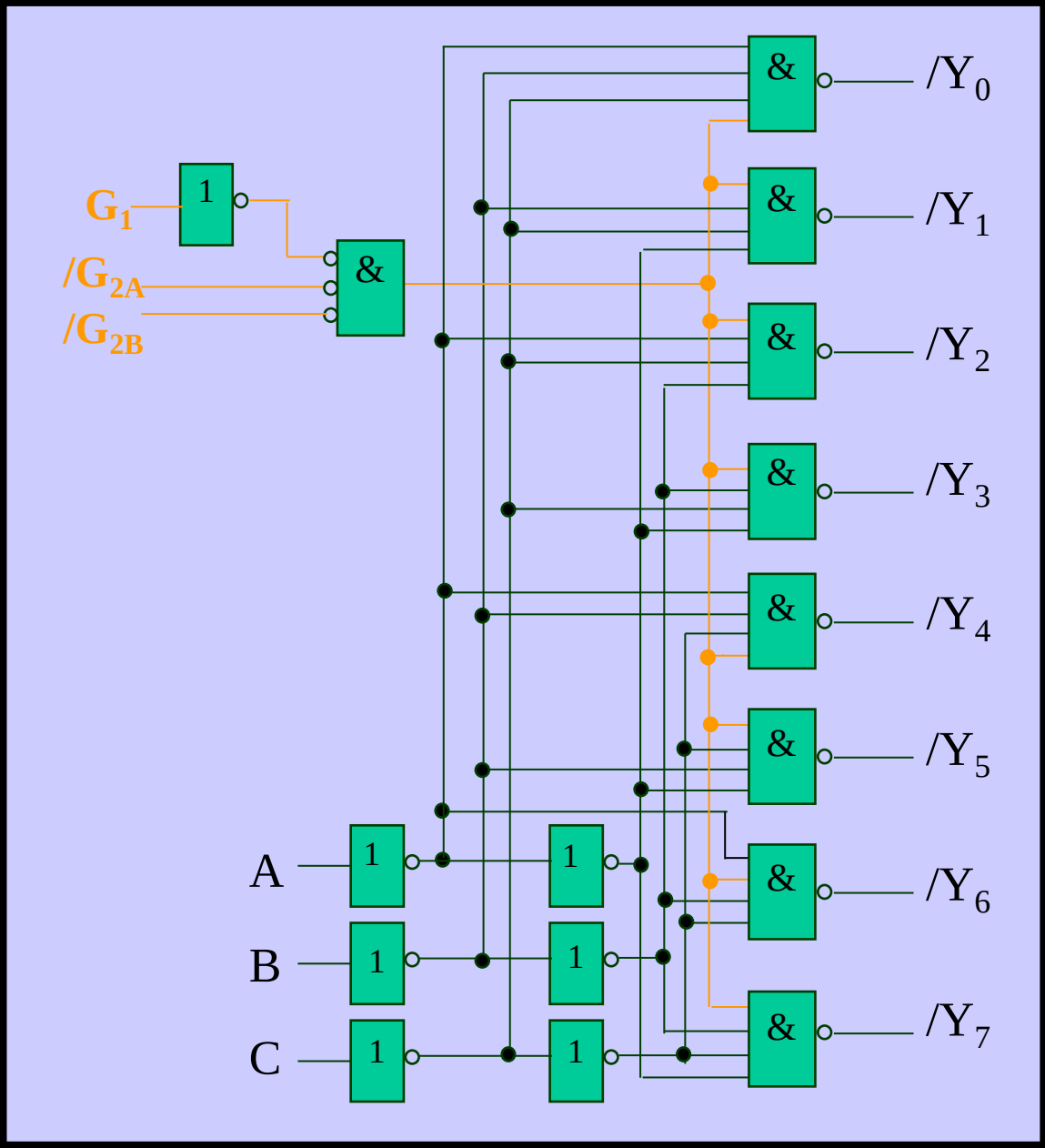


1） 3-8 译码器74LS138

74LS138真值表

输入						输出							
G_1	$/G_{2A}$	$/G_{2B}$	C	B	A	$/Y_7$	$/Y_6$	$/Y_5$	$/Y_4$	$/Y_3$	$/Y_2$	$/Y_1$	$/Y_0$
0	d	d	d	d	d	1	1	1	1	1	1	1	1
d	1	d	d	d	d	1	1	1	1	1	1	1	1
d	d	1	d	d	d	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1	0
1	0	0	0	0	1	1	1	1	1	1	1	0	1
1	0	0	0	1	0	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	0	1	1	1
1	0	0	1	0	0	1	1	1	0	1	1	1	1
1	0	0	1	0	1	1	1	0	1	1	1	1	1
1	0	0	1	1	0	1	0	1	1	1	1	1	1
1	0	0	1	1	1	0	1	1	1	1	1	1	1


74LS138
6
G1
4
G2A
5
G2B
1
A
2
B
3
C
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
15
14
13
12
11
10
9
7


G1
/G2A
/G2B
1
&
/Y0
/Y1
/Y2
/Y3
/Y4
/Y5
/Y6
/Y7
A
B
C

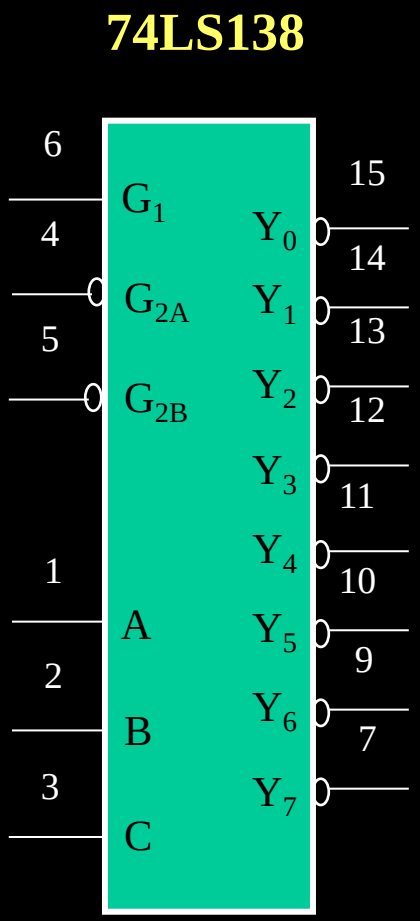
译码器74LS138的使用要点

- 74LS138的输出信号为低有效，它有三个使能输入端（G_1、$/G_{2A}$、$/G_{2B}$），只有在三个使能输入全部有效时，才能有正确的有效输出。
- 74LS138的内部功能可用逻辑表达式描述如下：$Y_i = G_1 \cdot G_{2A} \cdot G_{2B} \cdot m_i$

  其中，Y_i为内部输出编码字的第i位，

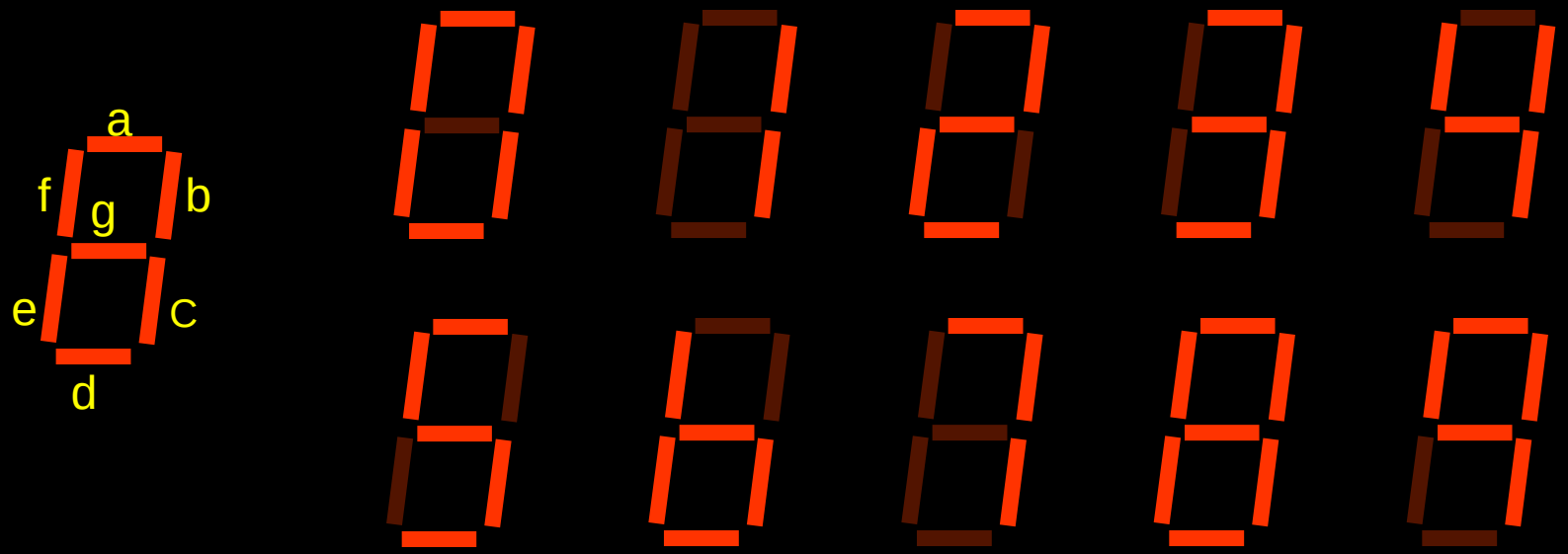
  m_i为输入变量C、B、A的最小项。
- 74LS138 外部信号之间的关系为：

$$/Y_i = \overline{G_1 \cdot /G_{2A} \cdot /G_{2B} \cdot m_i}$$



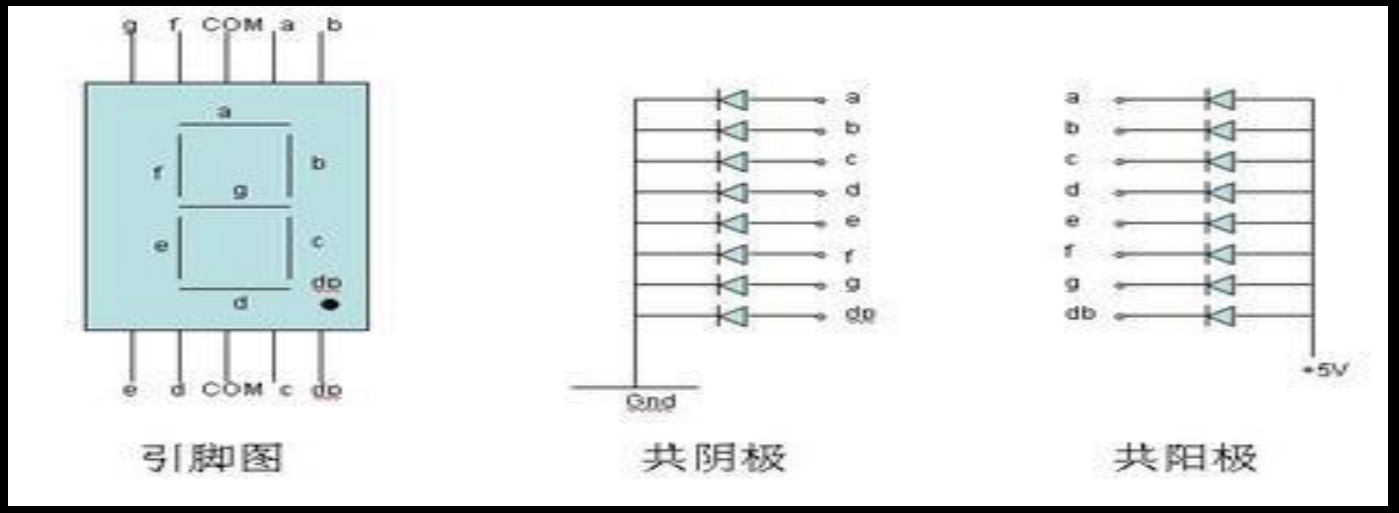
2）BCD译码器74LS49

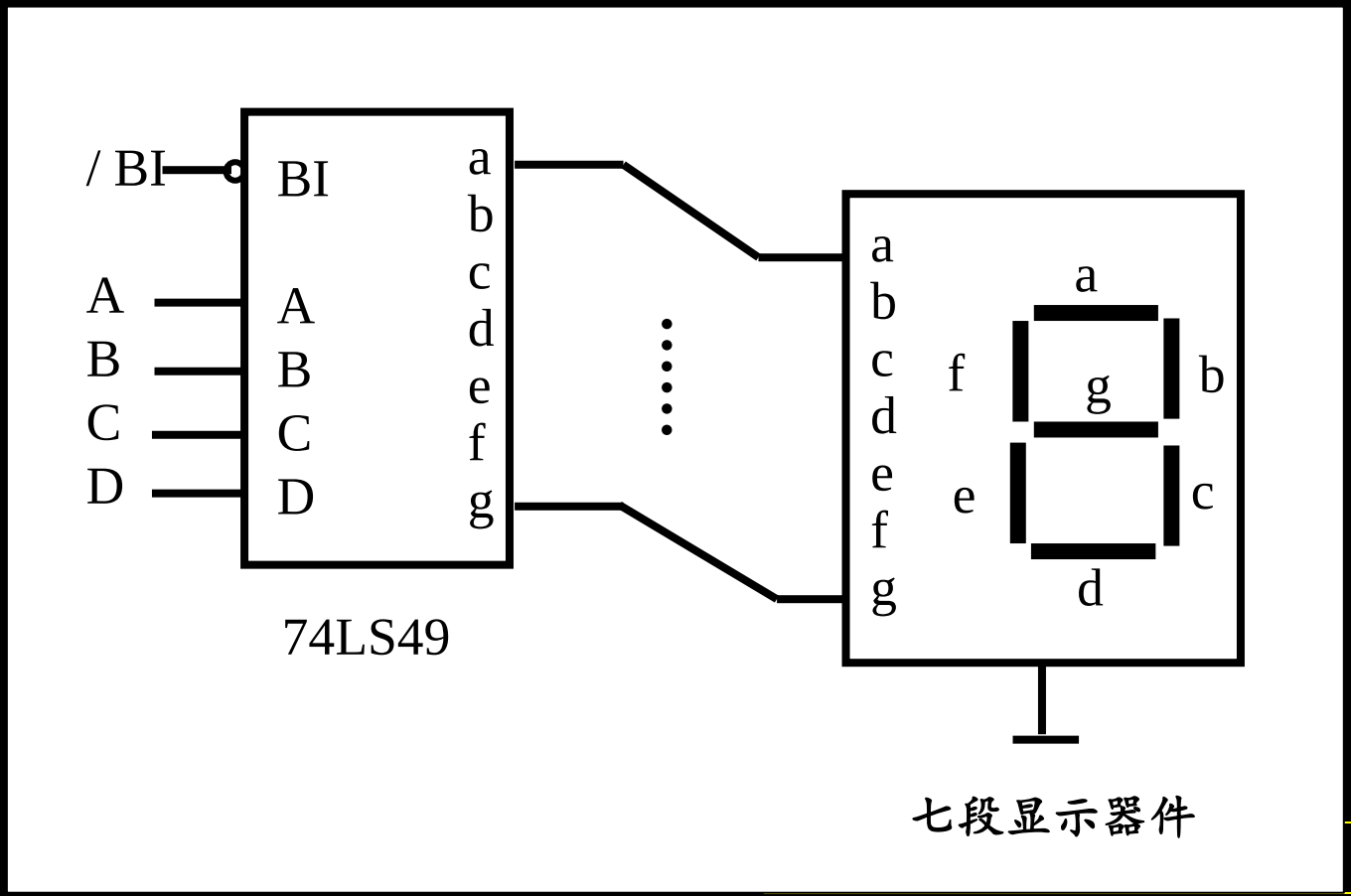
74LS49是常用的一种BCD码MSI器件，它的输入编码为**4位的BCD码**，输出为**7位编码字**。



七段显示器件结构

七段数码管分为共阳极和共阴极两种：





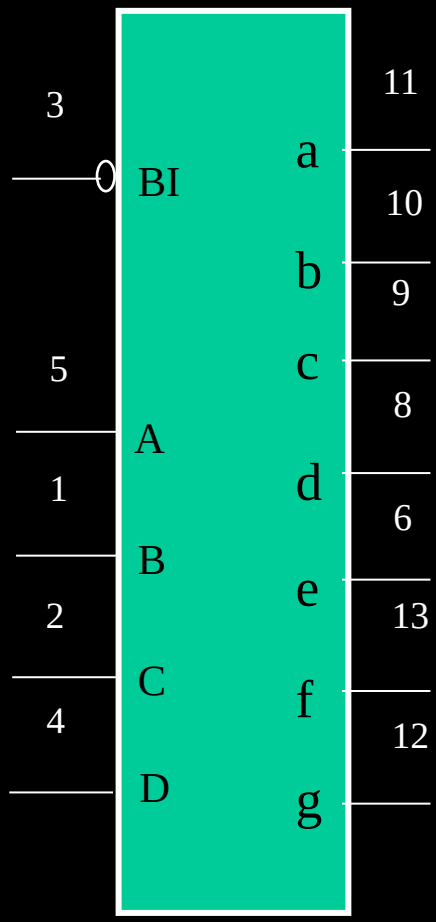
74LS49真值表

（/BI：禁止显示控制端）

输入					输出						
/BI	D	C	B	A	a	b	c	d	e	f	g
0	d	d	d	d	0	0	0	0	0	0	0
1	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
1	0	0	1	0	1	1	0	1	1	0	1
1	0	0	1	1	1	1	1	1	0	0	1
1	0	1	0	0	0	1	1	0	0	1	1
1	0	1	0	1	1	0	1	1	0	1	1
1	0	1	1	0	0	0	1	1	1	1	1
1	0	1	1	1	1	1	1	0	0	0	0
1	1	0	0	0	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1	0	0	1	1
1	1	0	1	0	0	0	0	1	1	0	1
1	1	0	1	1	0	0	1	1	0	0	1
1	1	1	0	0	0	1	0	0	0	1	1
1	1	1	0	1	1	0	0	1	0	1	1
1	1	1	1	0	0	0	0	1	1	1	1
1	1	1	1	1	0	0	0	0	0	0	0

逻辑符号：

74LS49



思考：一、如何设计74LS49？

阅读

设：非十进制数的输入组合为无关项。

例1 设计BCD译码器。

A_2A_1 \ A_8A_4				
			d	
			d	
			d	d
			d	d

思考：一、如何设计74LS49？

阅读

设：非十进制数的输入组合为无关项。

例1 设计BCD译码器。

$Y_0 = m_0 = \overline{A}_8\overline{A}_4\overline{A}_2\overline{A}_1$

$Y_1 = m_1 = \overline{A}_8\overline{A}_4\overline{A}_2A_1$

$Y_2 = m_2 + d_{10} = \overline{A}_4A_2\overline{A}_1$

$Y_3 = m_3 + d_{11} = \overline{A}_4A_2A_1$

$Y_4 = m_4 + d_{12} = A_4\overline{A}_2\overline{A}_1$

$Y_5 = m_5 + d_{13} = A_4\overline{A}_2A_1$

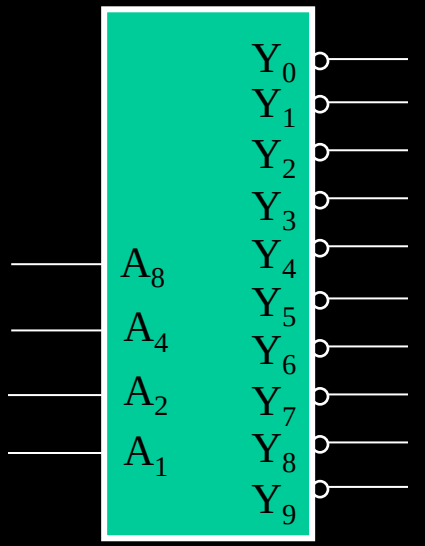
$Y_6 = m_6 + d_{14} = A_4A_2\overline{A}_1$

$Y_7 = m_7 + d_{15} = A_4A_2A_1$

$Y_8 = m_8 + d_{12} + d_{10} + d_{14} = A_8\overline{A}_1$

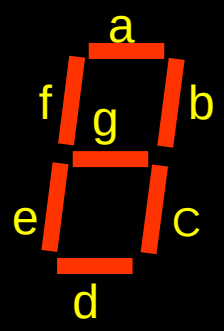
$Y_9 = m_9 + d_{12} + d_{11} + d_{15} = A_8A_1$

A_2A_1 \ A_8A_4				
	A_8A_4			
A_2A_1	Y_0	Y_4	d	Y_8
	Y_1	Y_5	d	Y_9
	Y_3	Y_7	d	d
	Y_2	Y_6	d	d



例2 设计BCD七段数码显示译码器

阅读



输入信号：BCD码DCBA

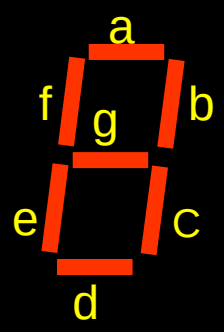
输出信号：控制数码管发光的信号

a、b、c、d、e、f、g

真值表如下所示：

阅读

输 入						输 出						
	/BI	D	C	B	A	a	b	c	d	e	f	g
	0	d	d	d	d	0	0	0	0	0	0	0
0	1	0	0	0	0	1	1	1	1	1	1	0
1	1	0	0	0	1	0	1	1	0	0	0	0
2	1	0	0	1	0	1	1	0	1	1	0	1
3	1	0	0	1	1	1	1	1	1	0	0	1
4	1	0	1	0	0	0	1	1	0	0	1	1
5	1	0	1	0	1	1	0	1	1	0	1	1
6	1	0	1	1	0	0	0	1	1	1	1	1
7	1	0	1	1	1	1	1	1	0	0	0	0
8	1	1	0	0	0	1	1	1	1	1	1	1
9	1	1	0	0	1	1	1	1	0	0	1	1
	1	1	0	1	0	d	d	d	d	d	d	d
	1	1	0	1	1	d	d	d	d	d	d	d
	1	1	1	0	0	d	d	d	d	d	d	d
	1	1	1	0	1	d	d	d	d	d	d	d
	1	1	1	1	0	d	d	d	d	d	d	d
	1	1	1	1	1	d	d	d	d	d	d	d



由真值表填出卡诺图　①填入无关项

阅读

		d	
		d	
		d	d
		d	d

a

		d	
		d	
		d	d
		d	d

b

		d	
		d	
		d	d
		d	d

c

		d	
		d	
		d	d
		d	d

d

		d	
		d	
		d	d
		d	d

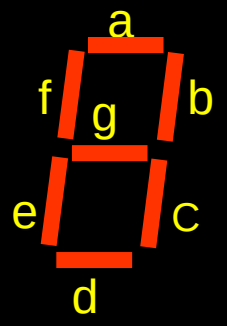
e

		d	
		d	
		d	d
		d	d

f

		d	
		d	
		d	d
		d	d

g



输入	输出
D C B A	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
1 0 1 0	d d d d d d d
1 0 1 1	d d d d d d d
1 1 0 0	d d d d d d d
1 1 0 1	d d d d d d d
1 1 1 0	d d d d d d d
1 1 1 1	d d d d d d d

由真值表填出卡诺图　①填入无关项

②分别填“1”

阅读

1		d	1
	1	d	1
1	1	d	d
1		d	d

a

1	1	d	1
1		d	1
1	1	d	d
1		d	d

b

1	1	d	1
1	1	d	1
1	1	d	d
	1	d	d

c

1		d	1
	1	d	
1		d	d
1	1	d	d

d

1		d	1
		d	
		d	d
1	1	d	d

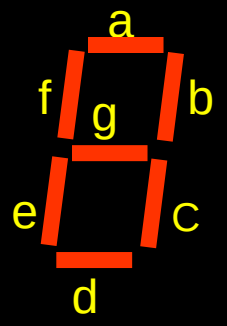
e

1	1	d	1
	1	d	1
		d	d
	1	d	d

f

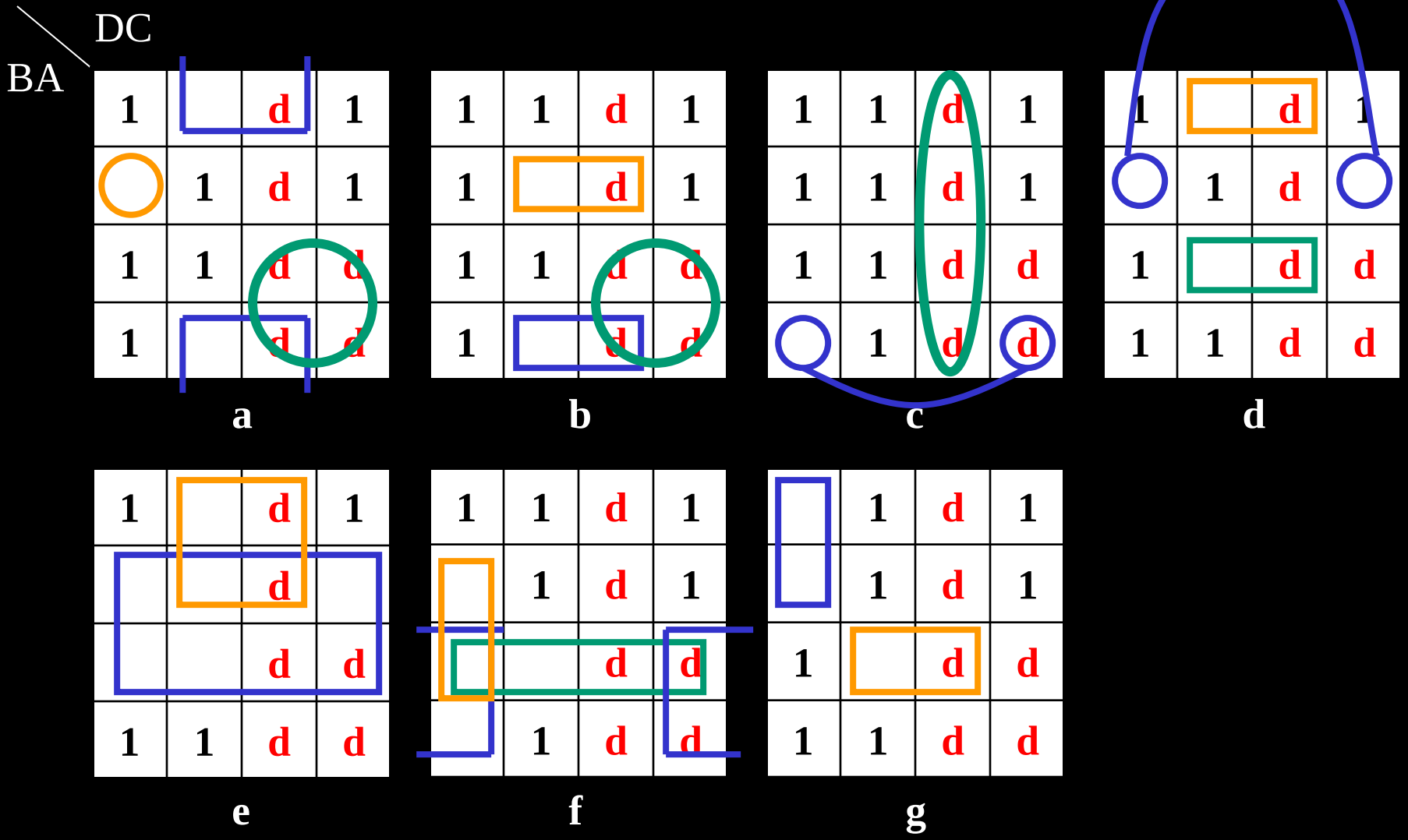
	1	d	1
	1	d	1
1		d	d
1	1	d	d

g

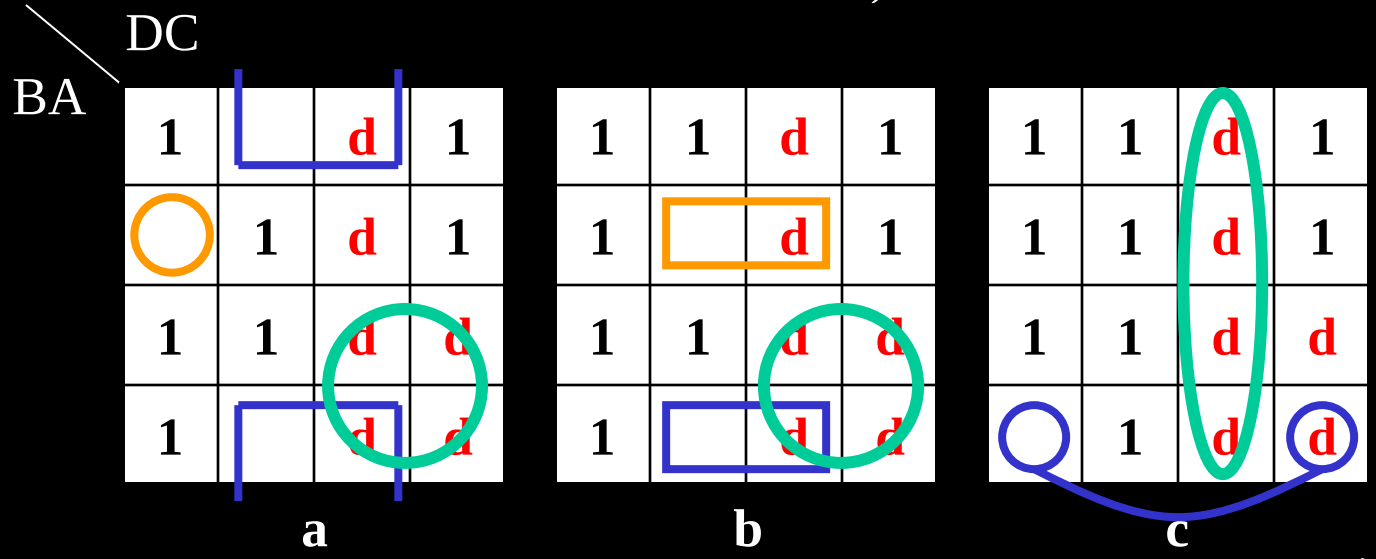


输入	输出
D C B A	a b c d e f g
0 0 0 0	1 1 1 1 1 1 0
0 0 0 1	0 1 1 0 0 0 0
0 0 1 0	1 1 0 1 1 0 1
0 0 1 1	1 1 1 1 0 0 1
0 1 0 0	0 1 1 0 0 1 1
0 1 0 1	1 0 1 1 0 1 1
0 1 1 0	0 0 1 1 1 1 1
0 1 1 1	1 1 1 0 0 0 0
1 0 0 0	1 1 1 1 1 1 1
1 0 0 1	1 1 1 0 0 1 1
1 0 1 0	d d d d d d d
1 0 1 1	d d d d d d d
1 1 0 0	d d d d d d d
1 1 0 1	d d d d d d d
1 1 1 0	d d d d d d d
1 1 1 1	d d d d d d d

③圈“0”，找出反函数的最小覆盖(为得到与电路对应的“或与”式) 阅读



阅读 由卡诺图写出表达式，如下：



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

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

$$\overline{b} = C\overline{B}A + CB\overline{A} + DB$$

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

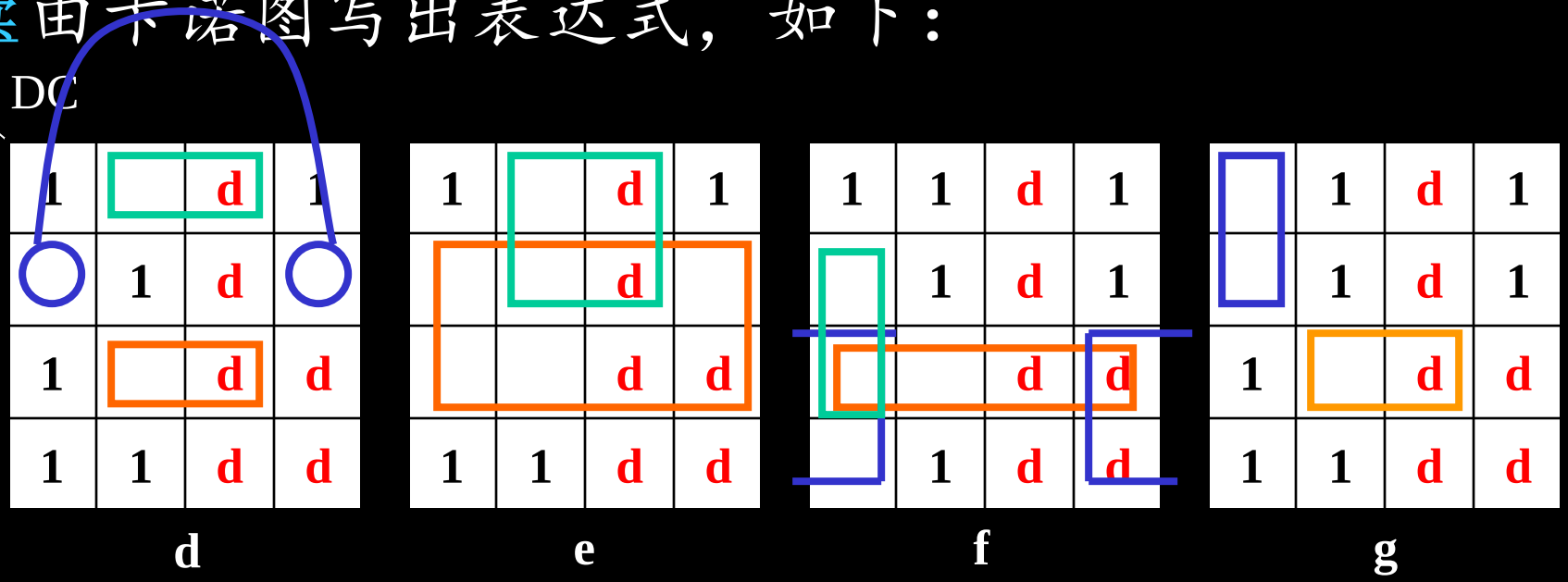
$$\overline{c} = \overline{C}B\overline{A} + DC$$

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

三个表达式中各包含有一个由“d”构成的多余项(绿色)，为何？

当禁止显示输入信号/BI= 0时，通过它们封锁a、b和c 的输出。

阅读 由卡诺图写出表达式，如下：



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

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

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

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

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

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

$$\overline{g} = CBA + \overline{D}\overline{C}\overline{B}$$

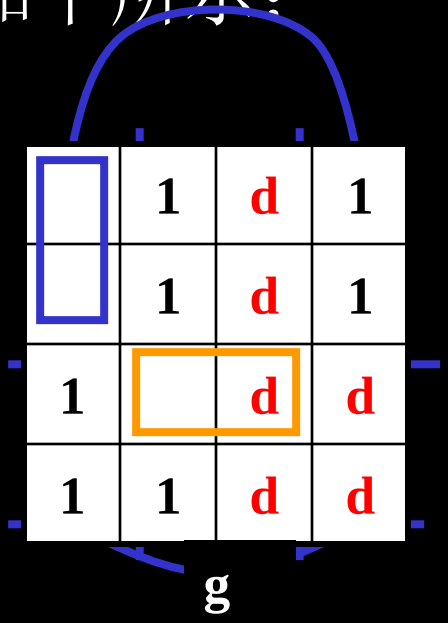
$$g = (\overline{C} + \overline{B} + \overline{A})(D + C + B)$$

思考：二、当给74LS49输入1010—1111组合时，显示的数字是什么？

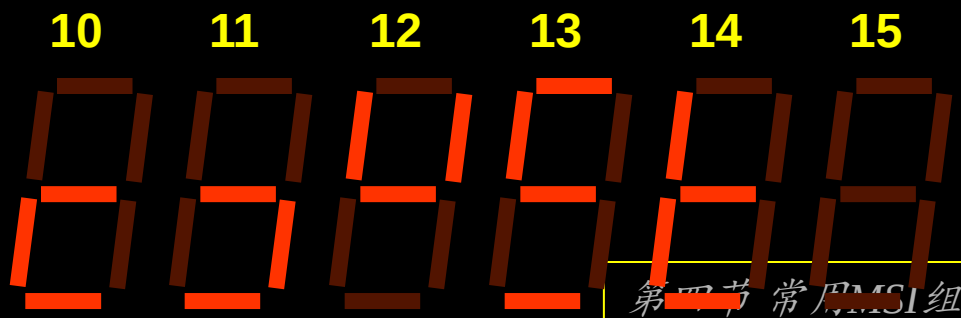
阅读

由卡诺图填出真值表的无关项部分，如下所示：

输 入					输 出						
/BI	D	C	B	A	a	b	c	d	e	f	g
10 1	1	0	1	0	0	0	0	1	1	0	1
11 1	1	0	1	1	0	0	1	1	0	0	1
12 1	1	1	0	0	0	1	0	0	0	1	1
13 1	1	1	0	1	1	0	0	1	0	1	1
14 1	1	1	1	0	0	0	0	1	1	1	1
15 1	1	1	1	1	0	0	0	0	0	0	0

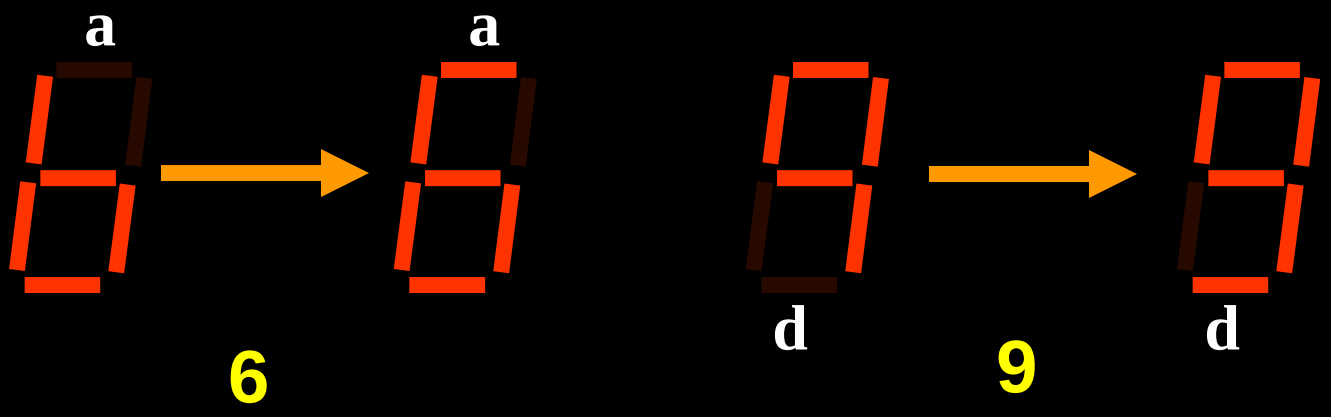


显示的数字分别是



思考：阅读

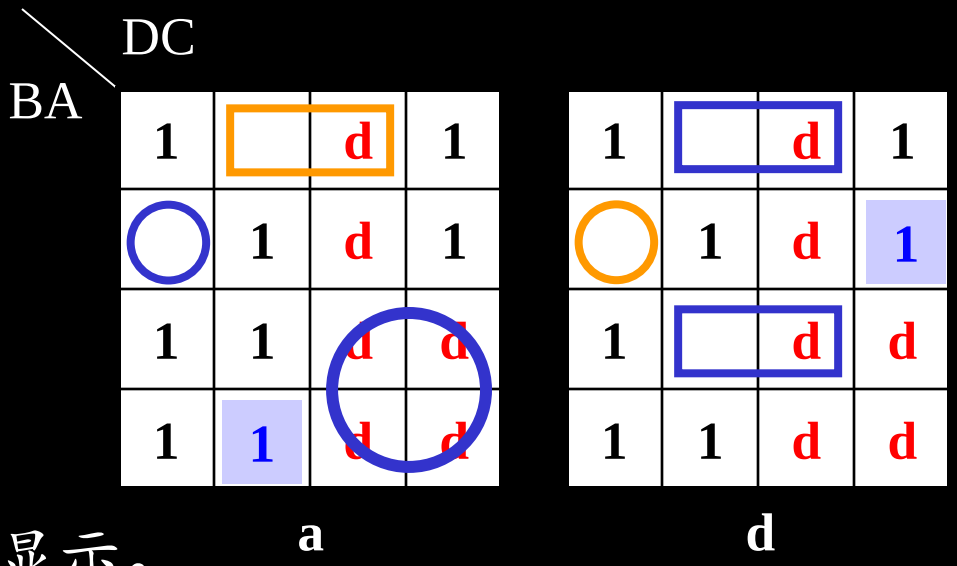
三、重新设计74LS49，使数字6和9带头尾，如图所示。并判断此设计影响非十进制1010——1111输入的显示吗？



设计只需改变输出 **a** 和 **d** 的表达式即可。分析其相应的卡诺图，如下：

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

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



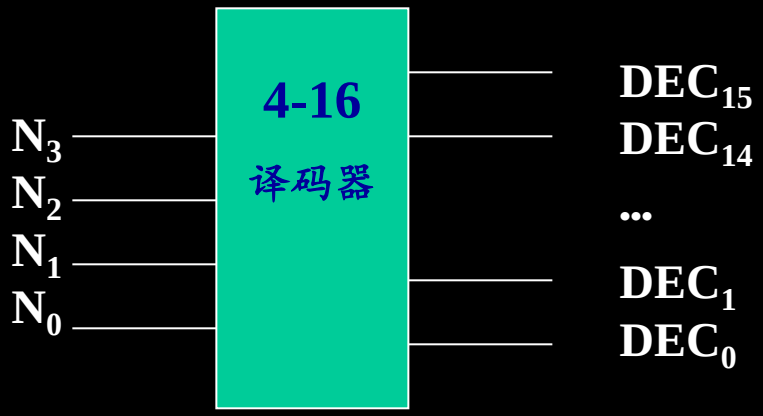
这种设计**不影响**非十进制数的显示。

3）二进制译码器的级联

Cascading Binary Decoders

当输入变量数 n大于器件的输入变量数时，可以用多个二进制译码器的级联来实现。

例1 用两个 3-8 译码器组成 4-16 译码器。



N_3	N_2	N_1	N_0	DEC_0	…	DEC_7	DEC_8	…	DEC_{15}
0	0	0	0	1	…	0	0	…	0
	…	…	…	…	…	…	…	…	…
	1	1	1	0	…	1	0	…	0
1	0	0	0	0	…	0	1	…	0
	…	…	…	…	…	…	…	…	…
	1	1	1	0	…	0	0	…	1

用两片74LS138 U_1和U_2级联起来,见图示。

①将输入的最高位N_3分别接到U_1./G_{2A}及U_2.G_1;

②整个级联电路的使能输入/EN分别接到U_1./ G_{2B}和U_2./ G_{2A}。

当 /EN = 0（有效）时

①若$N_3 = 0$，则U_2的输出无效(输出1), U_1的输出为按$N_2N_1N_0$译码:

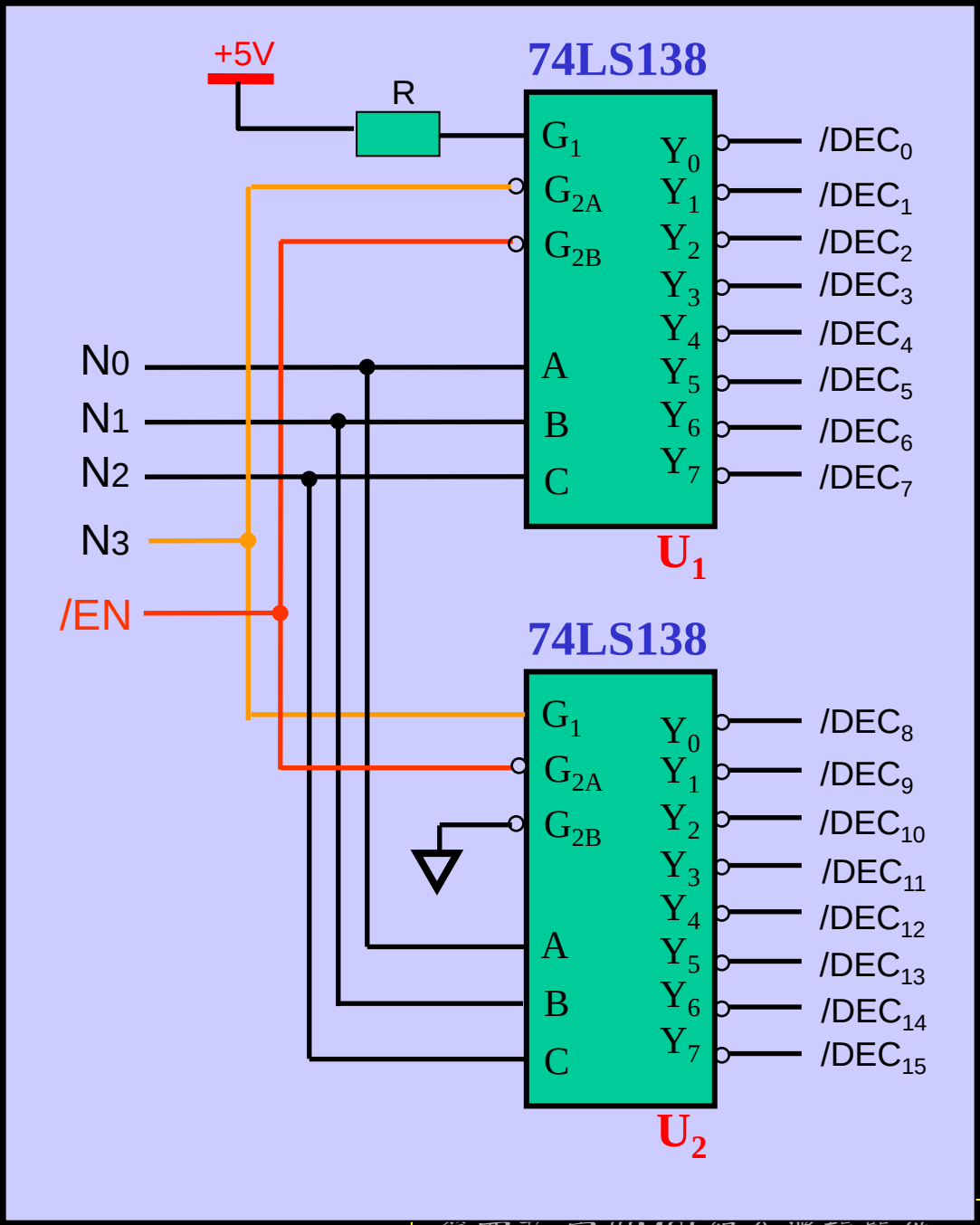
$/DEC_i = \overline{m}_i$ （i = 0 ~ 7）

②若$N_3 = 1$，则U_1的输出无效(输出1), U_2的输出为按$N_2N_1N_0$译码:

$/DEC_i = \overline{m}_i$ （i = 8 ~ 15）

总的级联译码器的输出逻辑表达式为： $/DEC_i = /EN + \overline{m}_i$， i = 0 ~ 15

式中：m_i 为 $N_3N_2N_1N_0$ 的对应最小项。



例2 设计一个 5-32 二进制译码器。

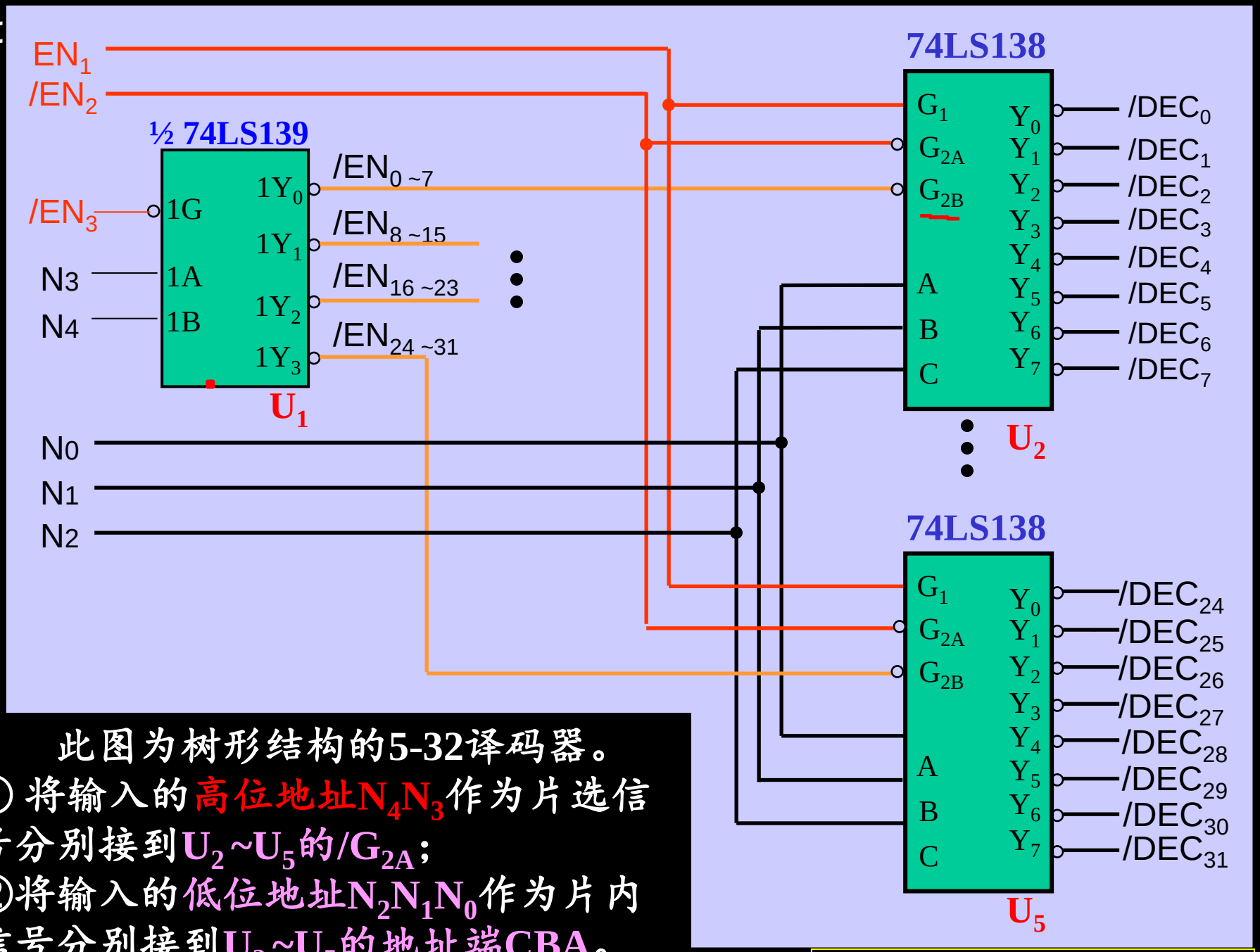
采用四片 74LS138 和一片74LS139组成一个二级译码的级联译码器。

4-16译码

N_3	N_2	N_1	N_0
0	0	0	0
	⋮	⋮	⋮
	1	1	1
1	0	0	0
	⋮	⋮	⋮
	1	1	1

5-32译码

N_4	N_3	N_2	N_1	N_0
0	0	0	0	0
		⋮	⋮	⋮
		1	1	1
0	1	0	0	0
		⋮	⋮	⋮
		1	1	1
1	0	0	0	0
		⋮	⋮	⋮
		1	1	1
1	1	0	0	0
		⋮	⋮	⋮
		1	1	1



此图为树形结构的5-32译码器。

① 将输入的高位地址N_4N_3作为片选信号分别接到U_2~U_5的/G_{2A}；

②将输入的低位地址$N_2N_1N_0$作为片内信号分别接到U_2~U_5的地址端CBA。

总的级联译码器的输出逻辑表达式为

$$/DEC_i = \overline{EN_1} + /EN_2 + /EN_3 + \bar{m}_i \qquad i = 0 \sim 31$$

式中 m_i 为 $N_4N_3N_2N_1N_0$ 的对应最小项。

此例中的树形结构又称为二级译码，速度较前例 4-16 译码器的一级译码要慢。

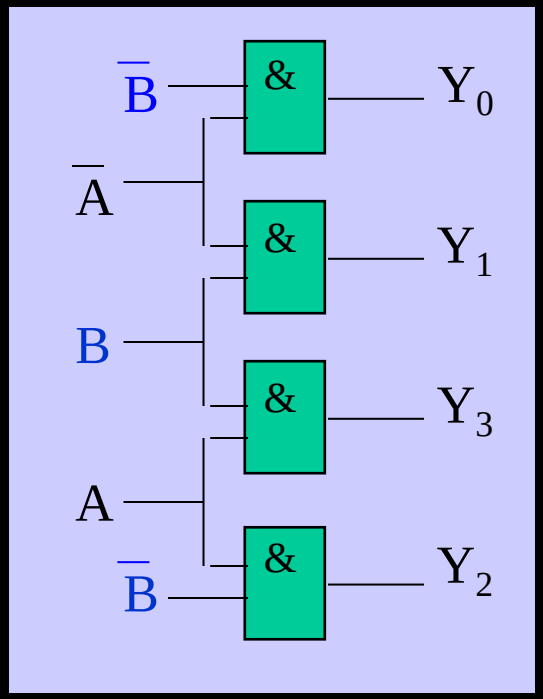
当 n数更大时，可以采用树形结构的多级译码方案。

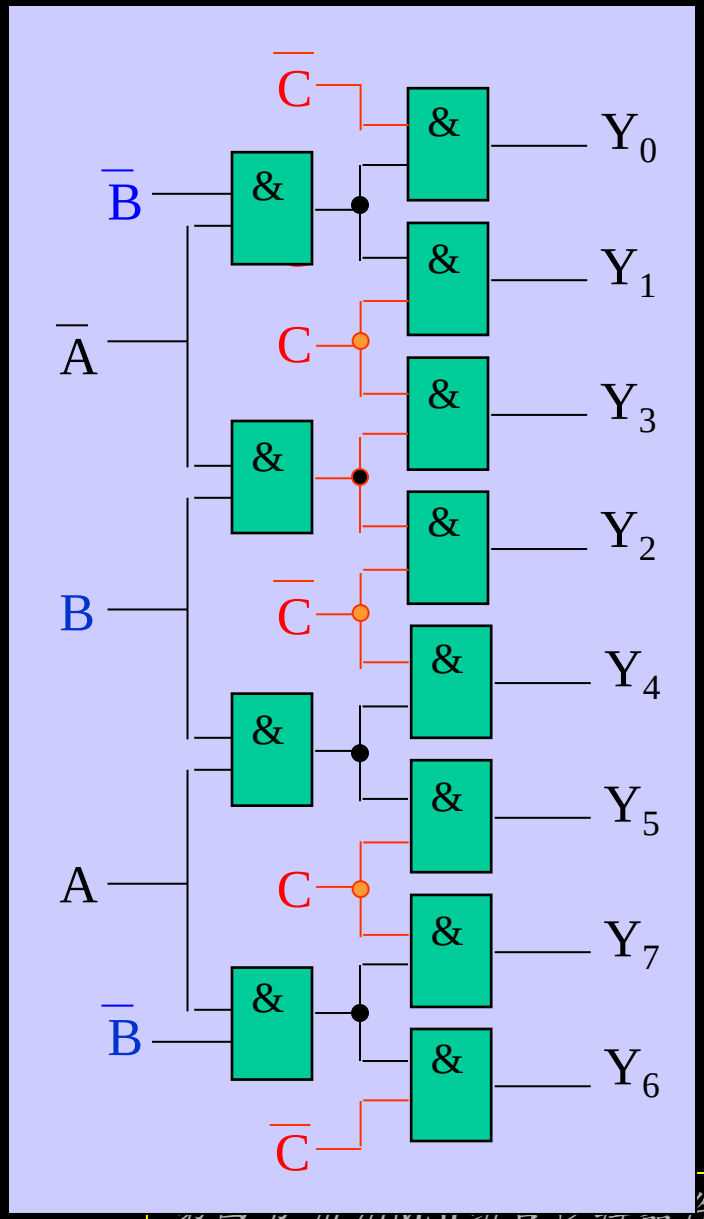
例 设计一个 9-512 二进制译码器。

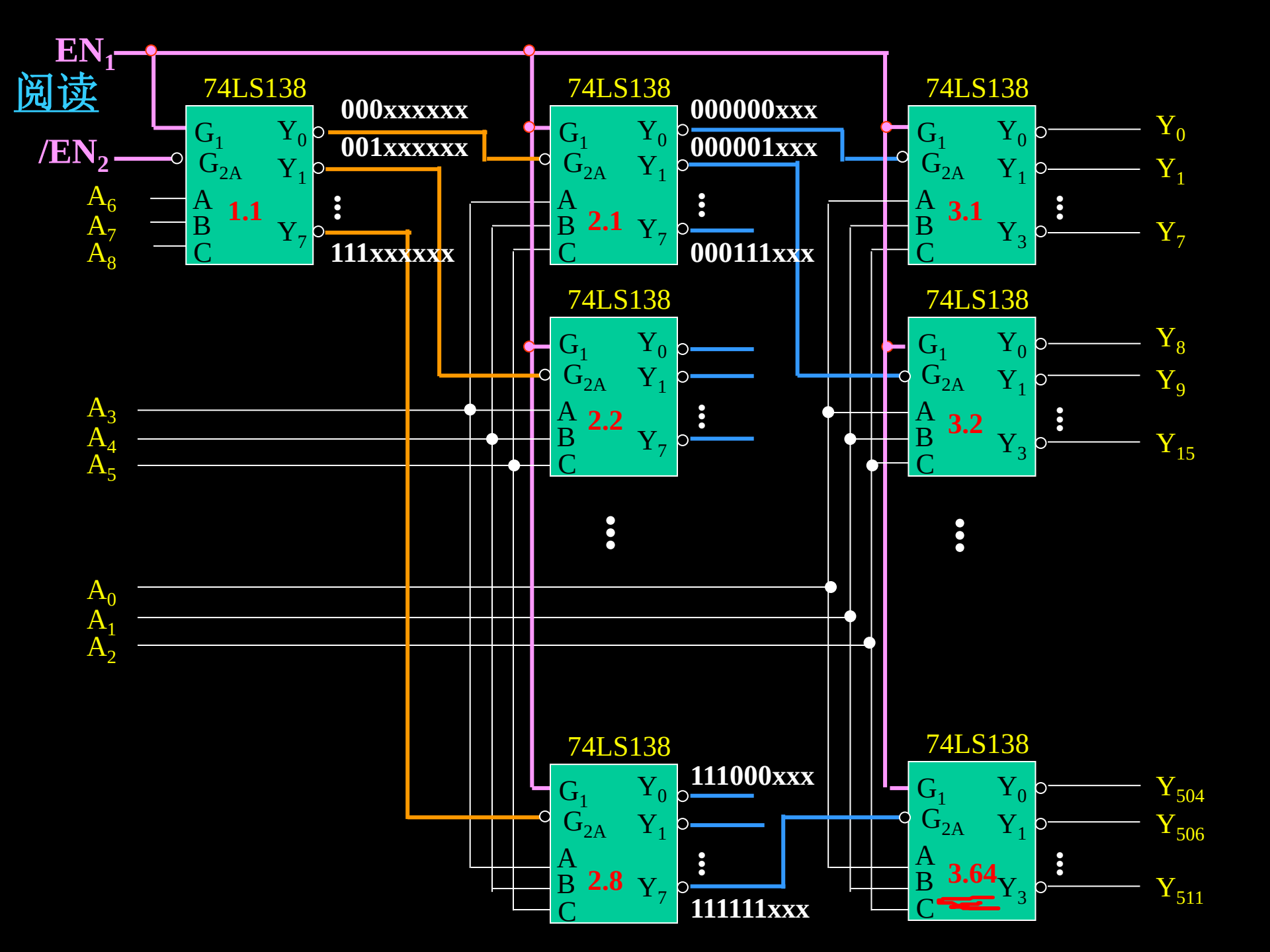
例 设计一个 9-512 二进制译码器。

阅读

所谓树形结构，如下图所示：





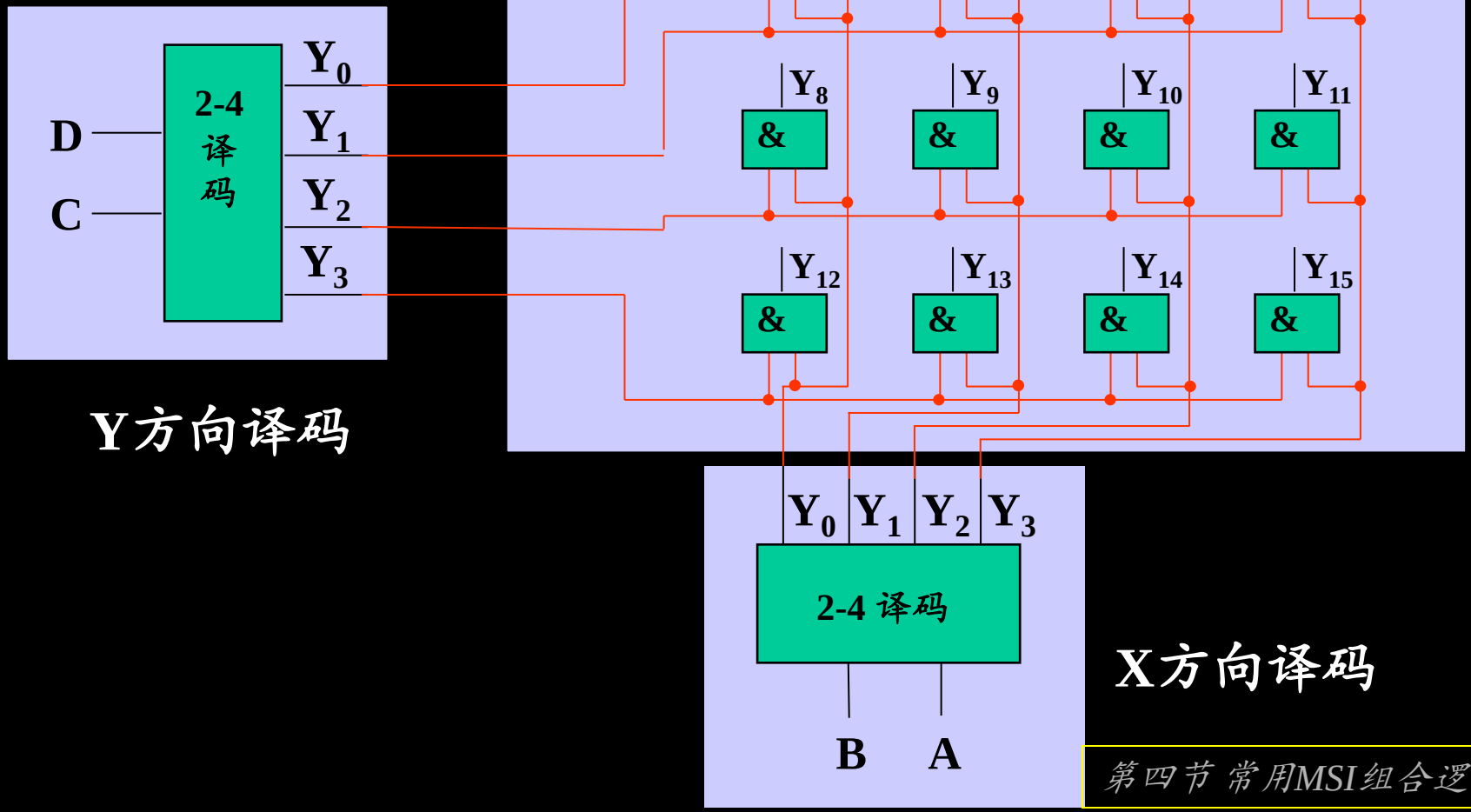

EN1
阅读
/EN2
A6
A7
A8
A3
A4
A5
A0
A1
A2
74LS138
G1
G2A
A
B
C
Y0
Y1
Y7
Y3
1.1
2.1
2.2
2.8
3.1
3.2
3.64
000xxxxxx
001xxxxxx
111xxxxxx
000000xxx
000001xxx
000111xxx
111000xxx
111111xxx
Y0
Y1
Y7
Y8
Y9
Y15
Y504
Y506
Y511

译码器还可以采用矩形结构。

阅读

所谓矩形结构，

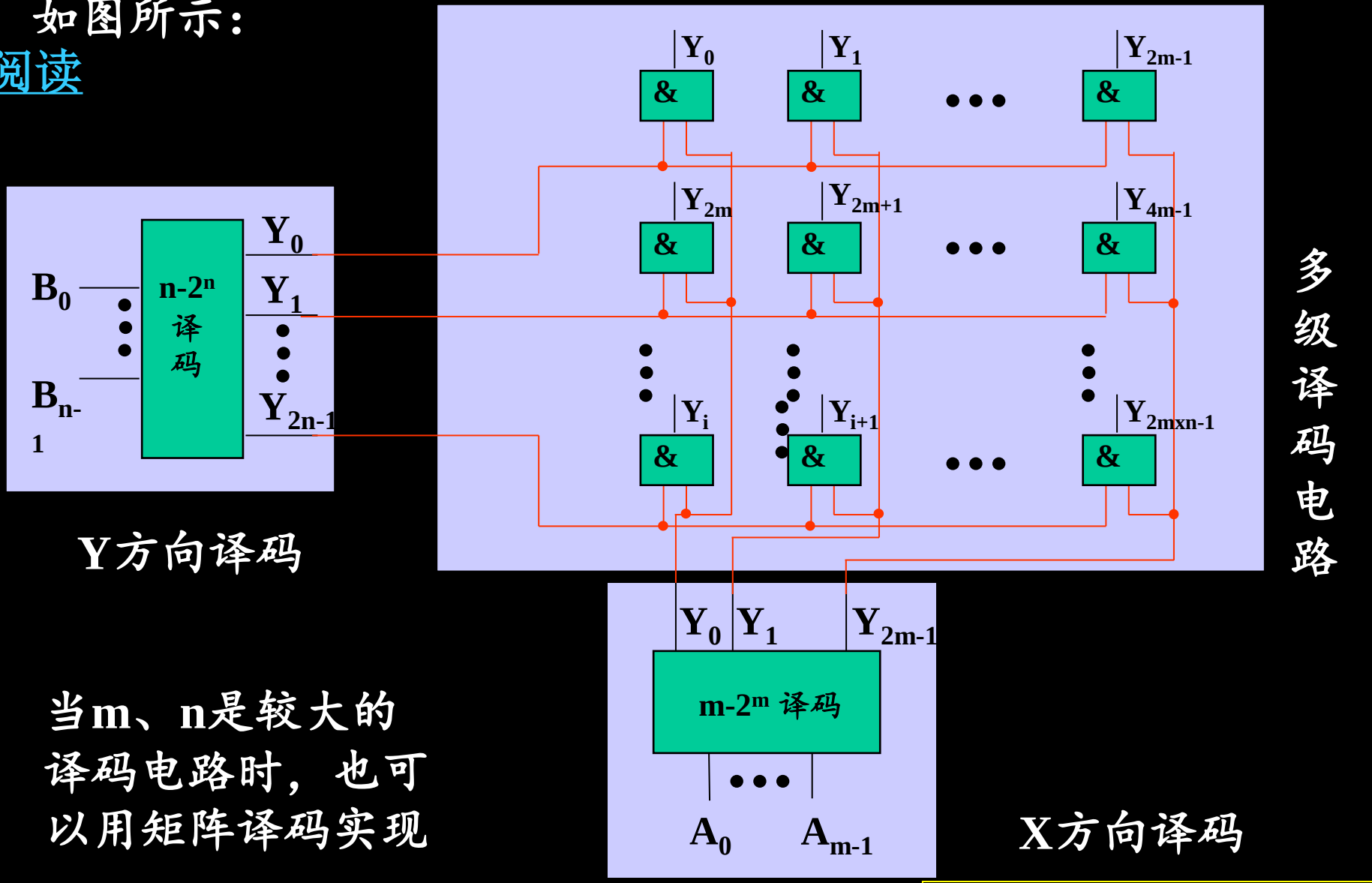
如图所示：



译码器采用m×n矩形结构时， m、n越接近，电路越简单。

如图所示：

阅读



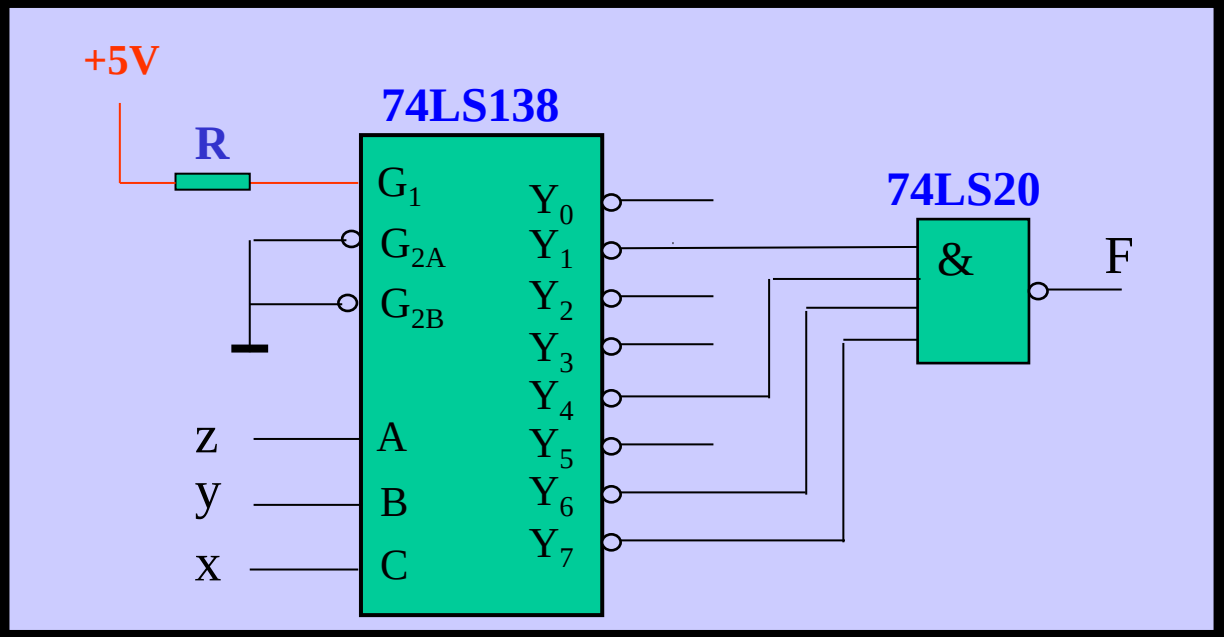
当m、n是较大的译码电路时，也可以用矩阵译码实现。

4）用MSI译码器实现组合逻辑函数

因为 n-2^n 二进制译码器的输出对应于 n 变量函数的2^n个最小项，所以可以借用此器件来实现任何组合逻辑函数。

例1 用译码器74LS138实现 $F(x,y,z)=\sum m(1,4,6,7)$

逻辑图如下所示：



例2 设计一个一位全加器。

设：输入端分别为：被加数输入x_i、加数输入y_i、
低位向本位的进位输入C_{i-1}

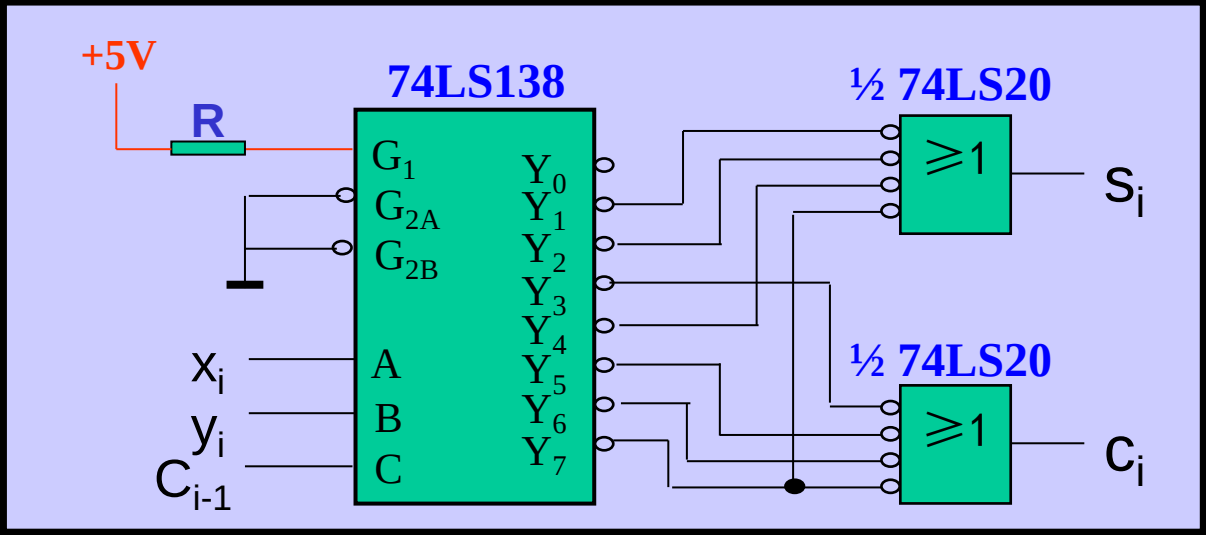
输出端分别为：本位的和输出S_i
本位向高位的进位输出C_i

则一位全加器的真值表如下所示。

C_{i-1} x_i y_i	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

由真值表得到 $S_i = \sum m^3(1,2,4,7)$

$C_i = \sum m^3(3,5,6,7)$



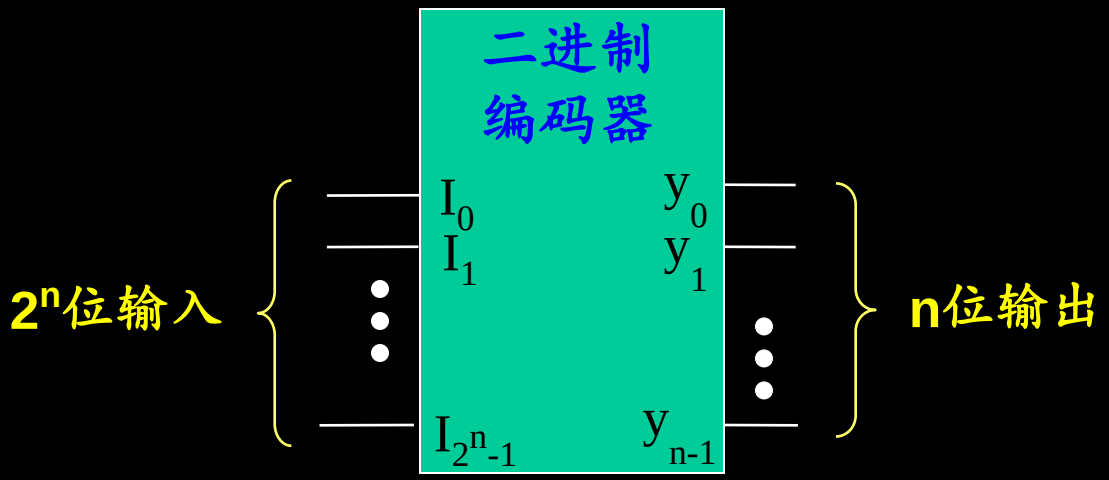
一位全加器逻辑图

2. 二进制编码器

当译码器的输出编码位数少于输入编码位数时，这种器件称为编码器。

2^n-n二进制编码器通用结构如图所示，其中输入端为2^n个，输出为n位二进制数，因此它的输入输出关系正好与译码器的相反。

约束条件：同一时刻只能有一个输入端有效。



二进制编码器框图

例 设计一个操作码形成器。

当按下+、–、×各操作码时，要求产生加法、减法、乘法的操作码01、10、11。

① 真值表

A B C	Y_1 Y_0	操作
0 0 0	0 0	空操作
0 0 1	0 1	加法
0 1 0	1 0	减法
0 1 1	d d	
1 0 0	1 1	乘法
1 0 1	d d	
1 1 0	d d	
1 1 1	d d	

当且仅当输入代码中的仅一位为 1

② 卡诺图

	1	d	1
	d	d	d

Y_1

		d	1
1	d	d	d

Y_0

③ 输出函数表达式

$Y_1 = A + B$

$Y_0 = A + C$

函数仅包括那些使其为1的输入组合。

例 一个8位输入、3位输出的编码器如图所示。

这是一个8-3二进制编码器。输入：$I_0 \sim I_7$，输出：$Y_0 \sim Y_2$

① 简化真值表

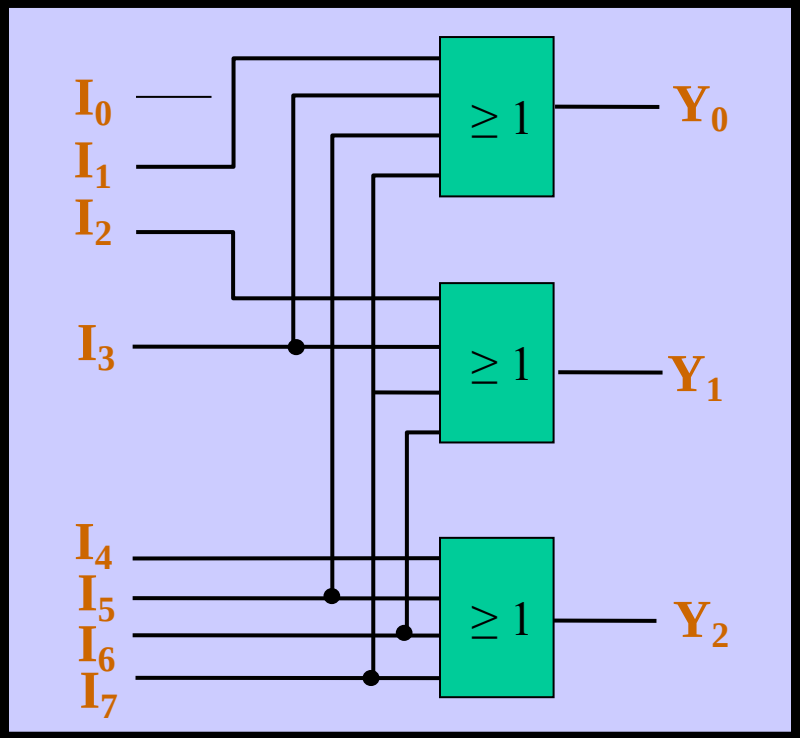
I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

② 输出函数表达式

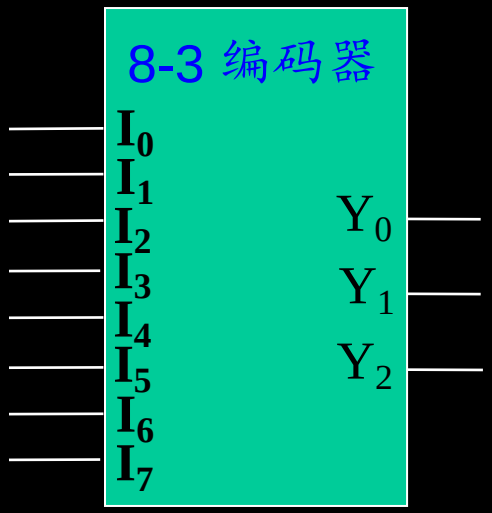
$$Y_0 = I_1 + I_3 + I_5 + I_7$$
$$Y_1 = I_2 + I_3 + I_6 + I_7$$
$$Y_2 = I_4 + I_5 + I_6 + I_7$$

当且仅当输入代码中的一位为1，输出编码不可能重复。

③电路图



④ 逻辑符号



I_i与Y_j之间的关系：使Y_j为 1 的是那些I_i，其下标 i 的二进制数的第 j 位均为1。

例 $Y_1 = I_2+I_3+I_6+I_7$

即 $Y_1 = I_{010}+I_{011}+I_{110}+I_{111}$

阅读　根据前述的输出与输入下标的关系，可以直接写出4-2 编码器的输出函数表达式，如下：

$$Y_0 = I_{01} + I_{11} = I_1 + I_3$$

$$Y_1 = I_{10} + I_{11} = I_2 + I_3$$

根据前述的输出与输入下标的关系，可以直接写出16-4 编码器的输出函数表达式，如下：

$$Y_0 = I_1 + I_3 + I_5 + I_7 + I_9 + I_{11} + I_{13} + I_{15}$$

$$Y_1 = I_2 + I_3 + I_6 + I_7 + I_{10} + I_{11} + I_{14} + I_{15}$$

$$Y_2 = I_4 + I_5 + I_6 + I_7 + I_{12} + I_{13} + I_{14} + I_{15}$$

$$Y_3 = I_8 + I_9 + I_{10} + I_{11} + I_{12} + I_{13} + I_{14} + I_{15}$$

优先权编码器 Priority Encoders

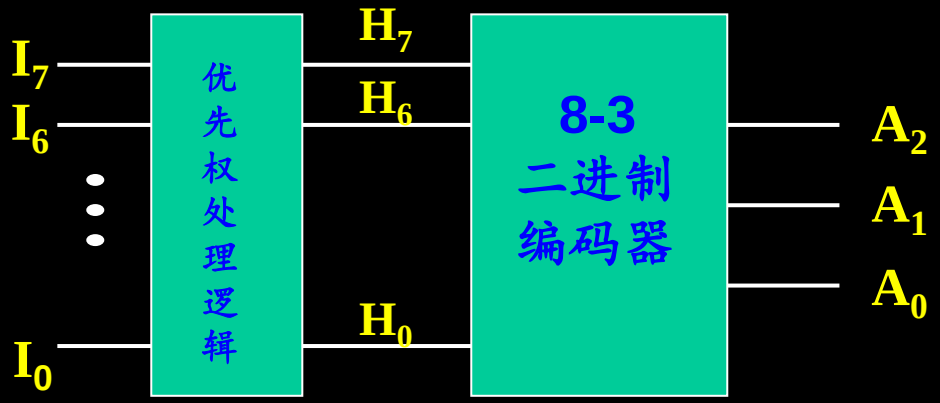
如果在任一时刻，允许 2^n 个部件中有多个器件同时提出请求，则 2^n - n 二进制编码器产生的 n 位编码必定有重复，而不能与输入请求的条件一一对应了。为此，应对输入端进行优先权分配，使编码器仅响应请求中优先权最高的有效输入端，并产生相应的输出编码。这种具有指定输入端优先权顺序的编码器。称为优先权编码器。

如：某工控系统中的优先级列表：

火警：最高优先级、主电源故障：次高、系统安全连锁：系统存在危险因素，自动报警、系统连锁：存在较小的系统险情、机器状态1操作连锁：流水线存在问题，如瓶颈现象、机器状态2操作连锁、机器状态3操作连锁、机器状态4操作连锁。

8-3 优先权编码器的结构框图如下所示：

设优先权为：I_7(最高)→ I_6→ I_5 → I_4 → I_3 → I_2 → I_1 → I_0



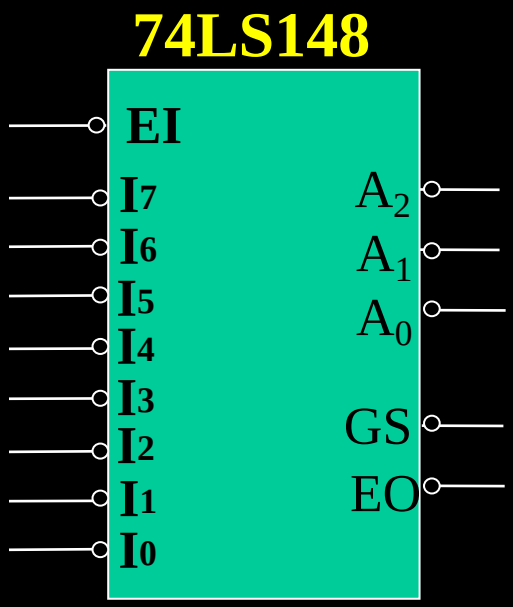
$$A_2 = H_4 + H_5 + H_6 + H_7$$
$$A_1 = H_2 + H_3 + H_6 + H_7$$
$$A_0 = H_1 + H_3 + H_5 + H_7$$

1） MSI优先权编码器 74LS148

② 真值表

输 入									输 出				
/EI	$/I_0$	$/I_1$	$/I_2$	$/I_3$	$/I_4$	$/I_5$	$/I_6$	$/I_7$	$/A_2$	$/A_1$	$/A_0$	/GS	/EO
1	d	d	d	d	d	d	d	d	1	1	1	1	1
0	d	d	d	d	d	d	d	0	0	0	0	0	1
0	d	d	d	d	d	d	0	1	0	0	1	0	1
0	d	d	d	d	d	0	1	1	0	1	0	0	1
0	d	d	d	d	0	1	1	1	0	1	1	0	1
0	d	d	d	0	1	1	1	1	1	0	0	0	1
0	d	d	0	1	1	1	1	1	1	0	1	0	1
0	d	0	1	1	1	1	1	1	1	1	0	0	1
0	0	1	1	1	1	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0

① 逻辑符号



③ 优先权处理逻辑

设 优先权为：

I_7(最高) $\rightarrow I_6 \rightarrow I_5 \rightarrow I_4 \rightarrow I_3 \rightarrow I_2 \rightarrow I_1 \rightarrow I_0$

H_i 与I_i的关系是：当 I_i 是最高优先权且为 1 时，H_i 才为 1。

即：

$$H_7 = I_7$$
$$H_6 = \overline{I_7}\, I_6$$
$$H_5 = \overline{I_7}\,\overline{I_6}\, I_5$$
$$H_4 = \overline{I_7}\,\overline{I_6}\,\overline{I_5}\, I_4$$
$$H_3 = \overline{I_7}\,\overline{I_6}\,\overline{I_5}\,\overline{I_4}\, I_3$$
$$H_2 = \overline{I_7}\,\overline{I_6}\,\overline{I_5}\,\overline{I_4}\,\overline{I_3}\, I_2$$
$$H_1 = \overline{I_7}\,\overline{I_6}\,\overline{I_5}\,\overline{I_4}\,\overline{I_3}\,\overline{I_2}\, I_1$$
$$H_0 = \overline{I_7}\,\overline{I_6}\,\overline{I_5}\,\overline{I_4}\,\overline{I_3}\,\overline{I_2}\,\overline{I_1}\, I_0$$

④ 输出编码为：

$$/A_2 = \overline{H_4 + H_5 + H_6 + H_7}$$
$$/A_1 = \overline{H_2 + H_3 + H_6 + H_7}$$
$$/A_0 = \overline{H_1 + H_3 + H_5 + H_7}$$

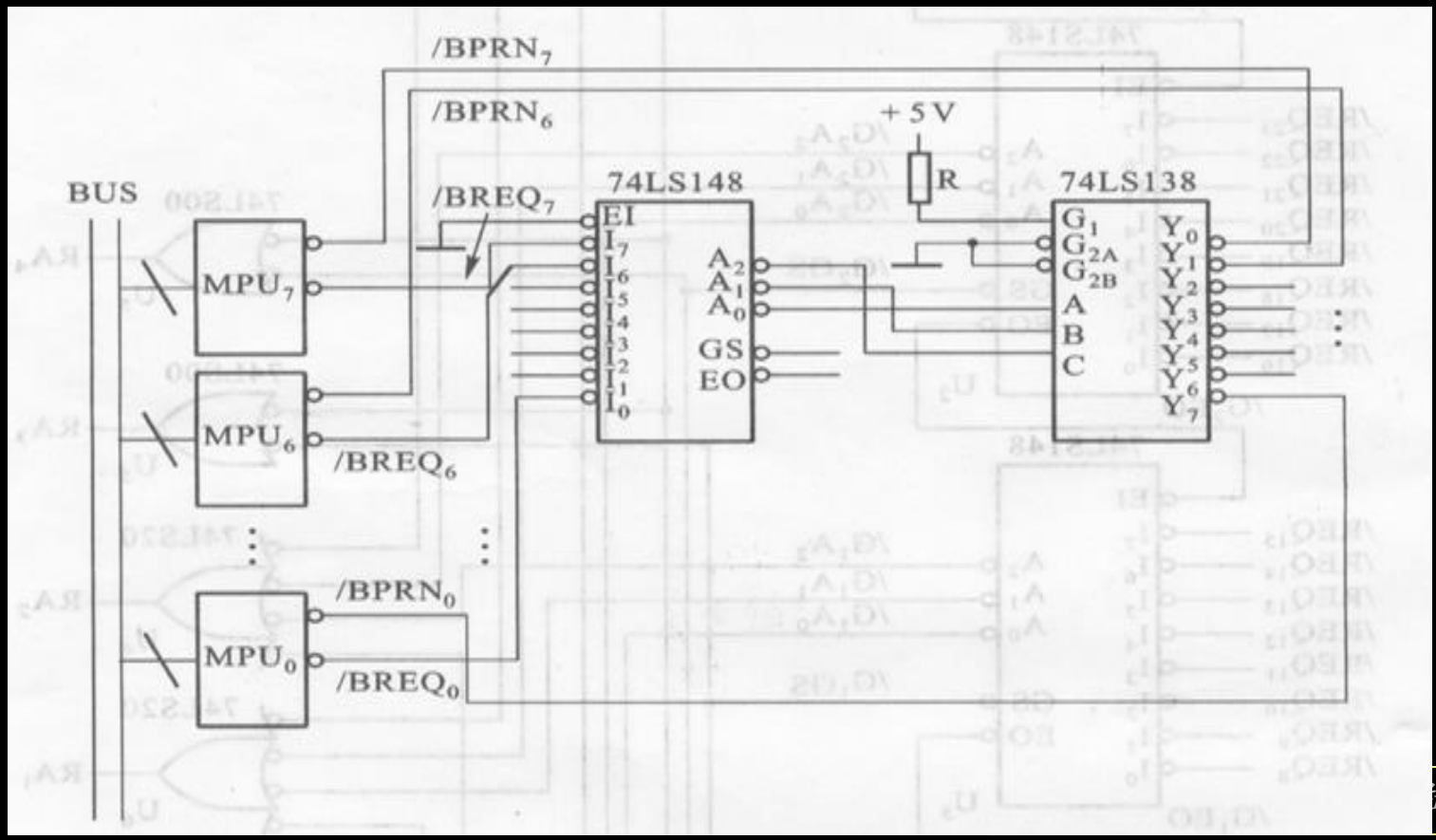
输出使能为：

$$EO = /GS = \overline{I_0 + I_1 + \ldots + I_7}$$

$$/EO = GS = /I_0\, /I_1 \ldots /I_7$$

2）编码器应用举例

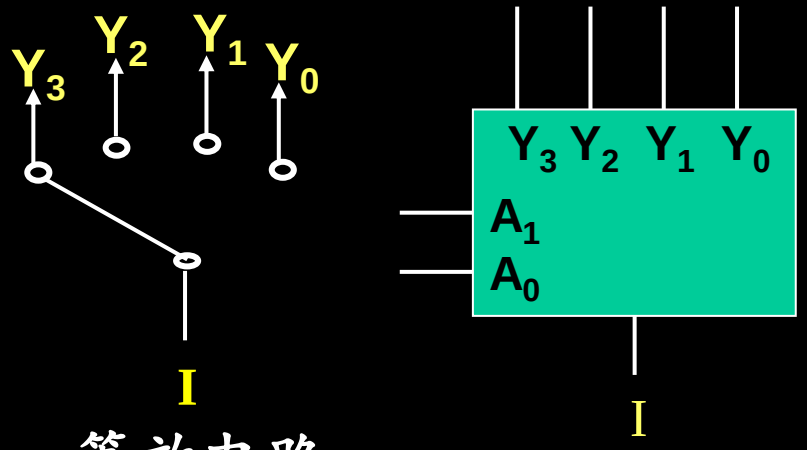
在多处理器系统中，需对各处理器争用总线作出仲裁。为提高仲裁速度，通常采用并行优先权仲裁方式。在争用总线的各处理器进行优先权分配后，通过优先权编码器和译码器进行裁决。逻辑电路图参见书P70图2.45。



2.4.2. 数据分配器和多路选择器

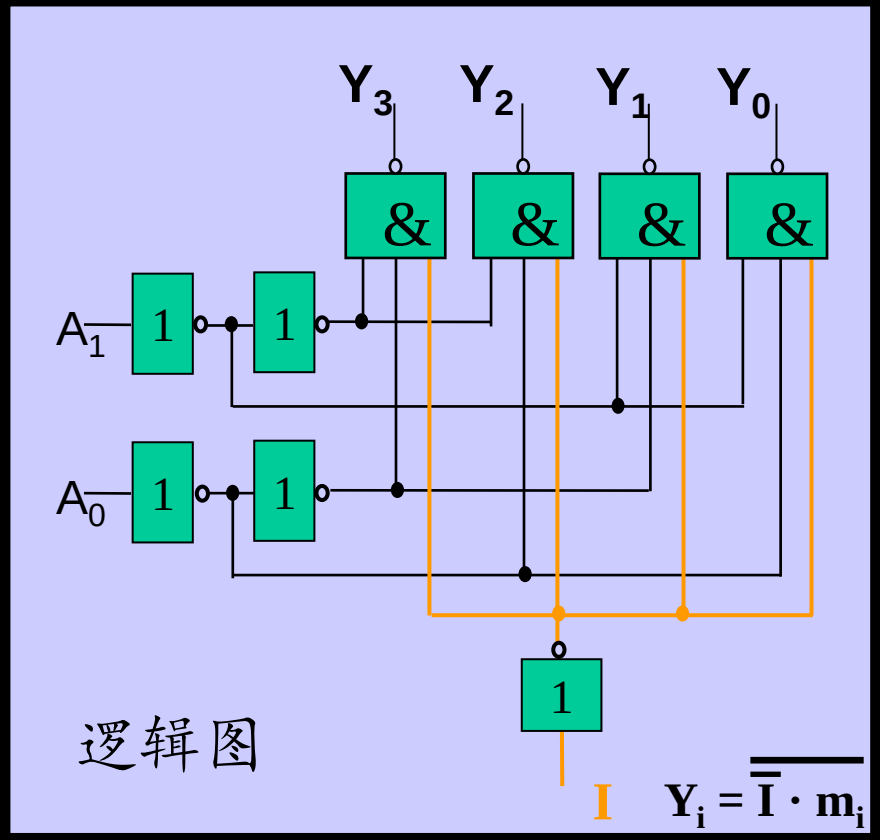
1. 数据分配器

如图四路数据分配器的等效电路和框图。



等效电路　　框图

单刀多掷开关



图中 I:传送数据输入端;

A_1,A_0:地址码输入端;

Y_3,Y_2,Y_1,Y_0:输出的数据通道。

这种分配器被称为“1~4多路分配器”。

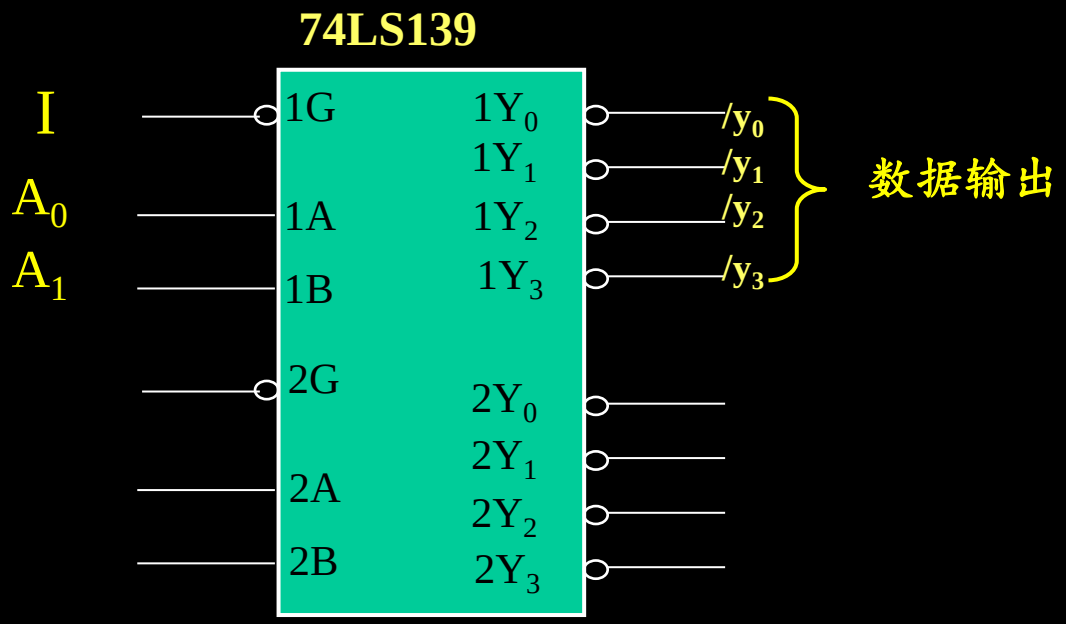
一般表达式为：$Y_i = I$ 其中 i为地址码$A_{n-1}\cdots\cdots A_0$的十进制值

➢ 用二进制译码器作为数据分配器

例1 用1/2 74LS139作为四输出数据分配器。

- 将使能端G 作为数据输入端，即 I 接至G端；
- 将数据输入端作为地址输入端，即A端接地址A_0位；B端接地址A_1位。

则：$/y_i = \overline{/G \cdot m_i} = \overline{\overline{I} \cdot m_i} = I$　　i为地址码A_1A_0的十进制数

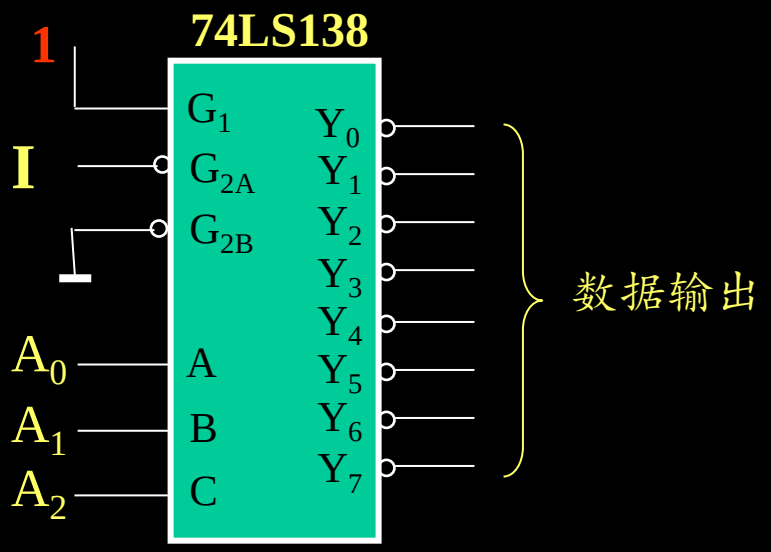


输　入			输　出			
/G	B	A	$/Y_3$	$/Y_2$	$/Y_1$	$/Y_0$
1	d	d	1	1	1	1
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1

例2 用74LS138作为八输出数据分配器

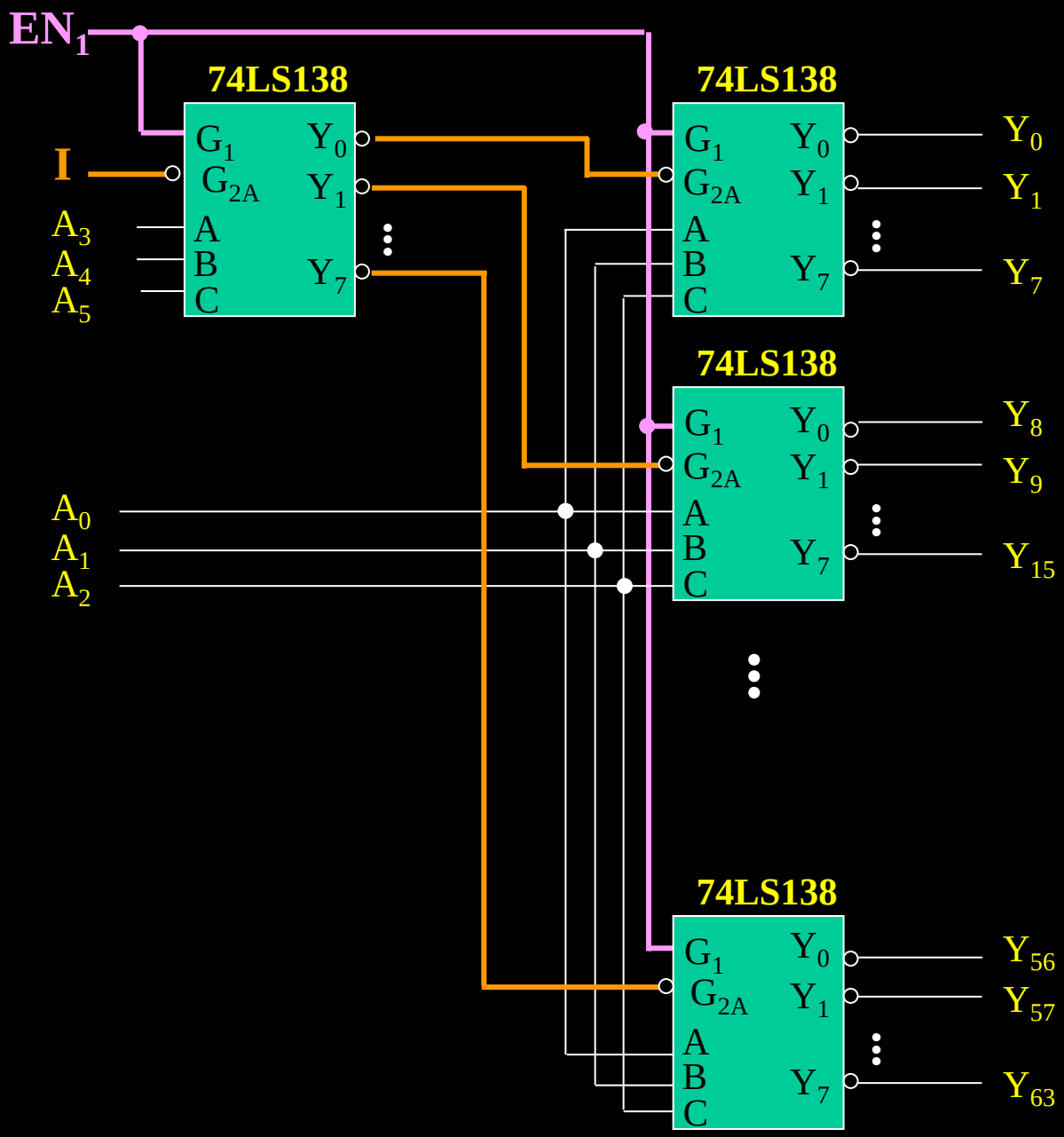
- 将/G_{2A}作为数据输入端，G_1接高电平、/G_{2B}端接地；
- 将数据端CBA分别接地址码$A_2A_1A_0$位。
- $/Y_i = G_1 \cdot /G_{2A} \cdot /G_{2B} \cdot m_i = 1 \cdot \overline{I} \cdot \overline{0} \cdot m_i$

  则： $/Y_i = I$　　i为地址码$A_2A_1A_0$的十进制数



输　入						输　出							
G_1	$/G_{2A}$	$/G_{2B}$	C	B	A	$/Y_7$	$/Y_6$	$/Y_5$	$/Y_4$	$/Y_3$	$/Y_2$	$/Y_1$	$/Y_0$
0	d	d	d	d	d	1	1	1	1	1	1	1	1
d	1	d	d	d	d	1	1	1	1	1	1	1	1
d	d	1	d	d	d	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1	0
1	0	0	0	0	1	1	1	1	1	1	1	0	1
1	0	0	0	1	0	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	0	1	1	1
1	0	0	1	0	0	1	1	1	0	1	1	1	1
1	0	0	1	0	1	1	1	0	1	1	1	1	1
1	0	0	1	1	0	1	0	1	1	1	1	1	1
1	0	0	1	1	1	0	1	1	1	1	1	1	1

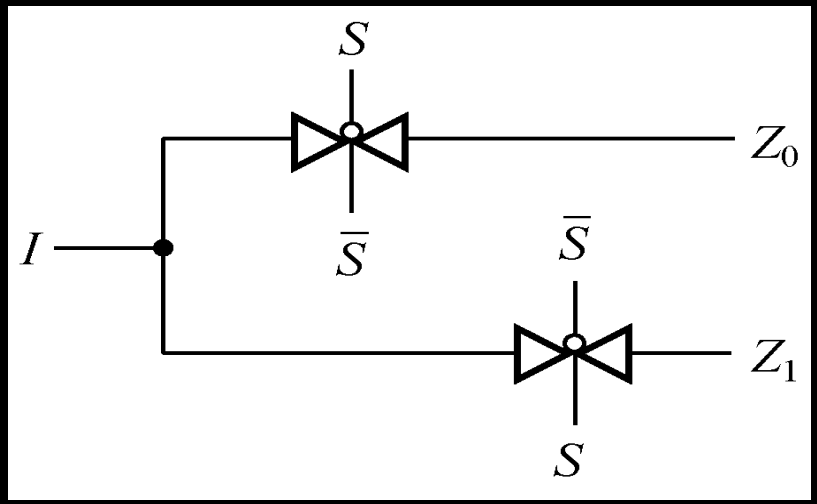
阅读 例3 用74LS138作为64路数据分配器



➢ 使用CMOS传输门实现的数据分配器的结构

阅读　CMOS传输门由一对并联的由互补控制信号控制的常开开关NMOS管和常关开关PMOS管构成。当外部控制信号有效时，传输门传送“0”和“1”都非常好。

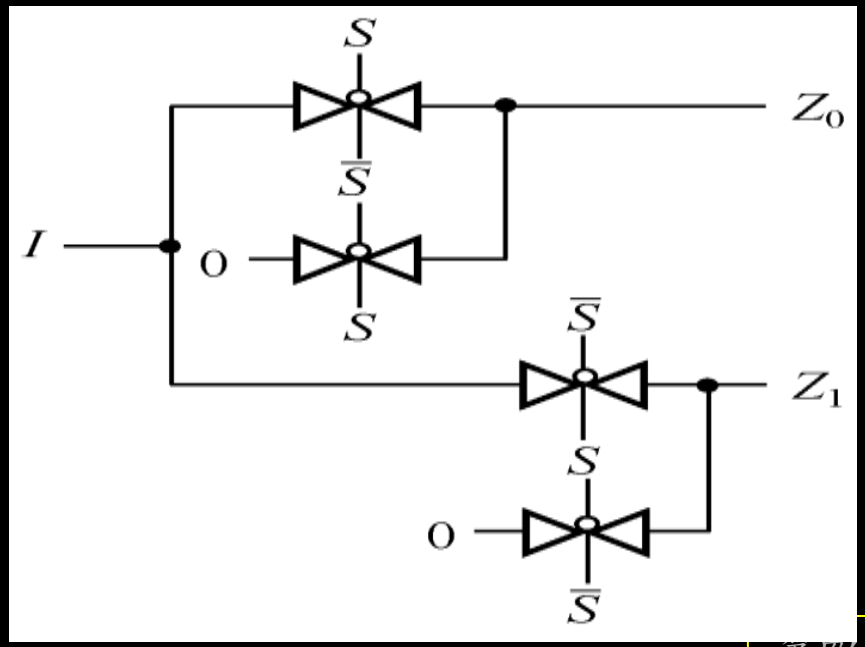
下图给出了控制逻辑一个例子，即用控制逻辑实现的数据分配器。如果S是有效的，则这一网络将把唯一的输入信号引导到Z_1端；如果S是失效的，则将被引导至Z_0端。这也被称为多路输出选择器，因为它刚好完成了与多路选择器相反的功能。



➢ 使用CMOS传输门实现的数据分配器的结构

阅读　但是，该电路存在一个问题：当S有效时，I被导向Z_1，此时输出端Z_0的值又是什么呢?遗憾的是，输出端Z_0既不是“0”，也不是“1”，而是悬空的。当S失效时，Z_1端也会有同样的问题。它违反了作为一个合适的功能网络所需的条件之一。

下图给出了一种有效的解决方法。当输入信号引导至Z_1时，一个额外的传输门将“0”引导至Z_0。当输入信号引导至Z_0时，对输出端Z_1也做同样的处理。

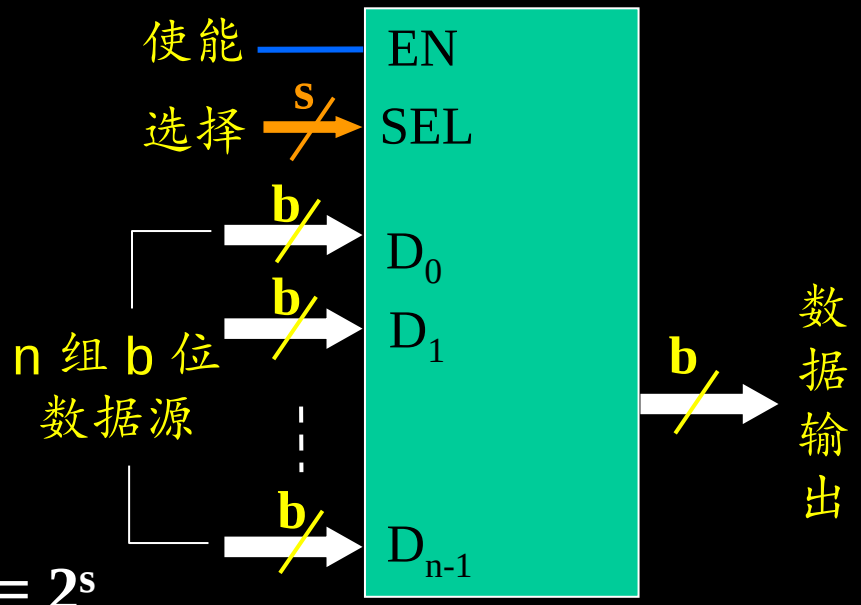


2. 多路选择器

多路选择器又称数据选择器(MUX)，是一个数字开关，可以从 n 路源数据中选择一路送至输出端。

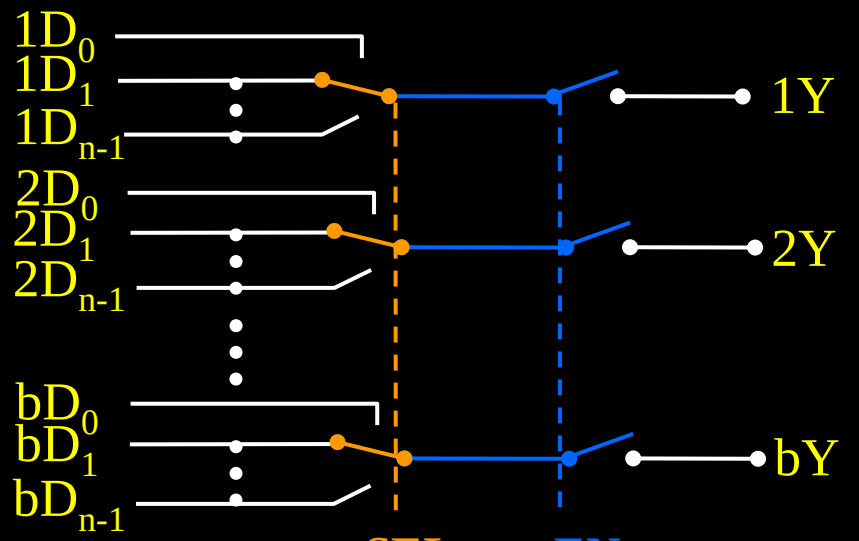
假设有 n 组输入数据源，每组数据源的宽度为 b 位二进制数，则反映多路选择器输出关系的框图和等效的电路如下图所示。其中高有效使能端 EN 的功能为：当EN = 0 时，所有的输出为 0。

① 多路选择器的结构框图



$n = 2^s$

② 多路选择器的等效功能



$KY = \sum EN \cdot m_i \cdot KD_i \quad K = 1, 2, \ldots, b$

③ 多路选择器输出逻辑表达式

从n组数据源中选择哪一组源数据传送到输出端，由选择输入端的输入值 S 决定。

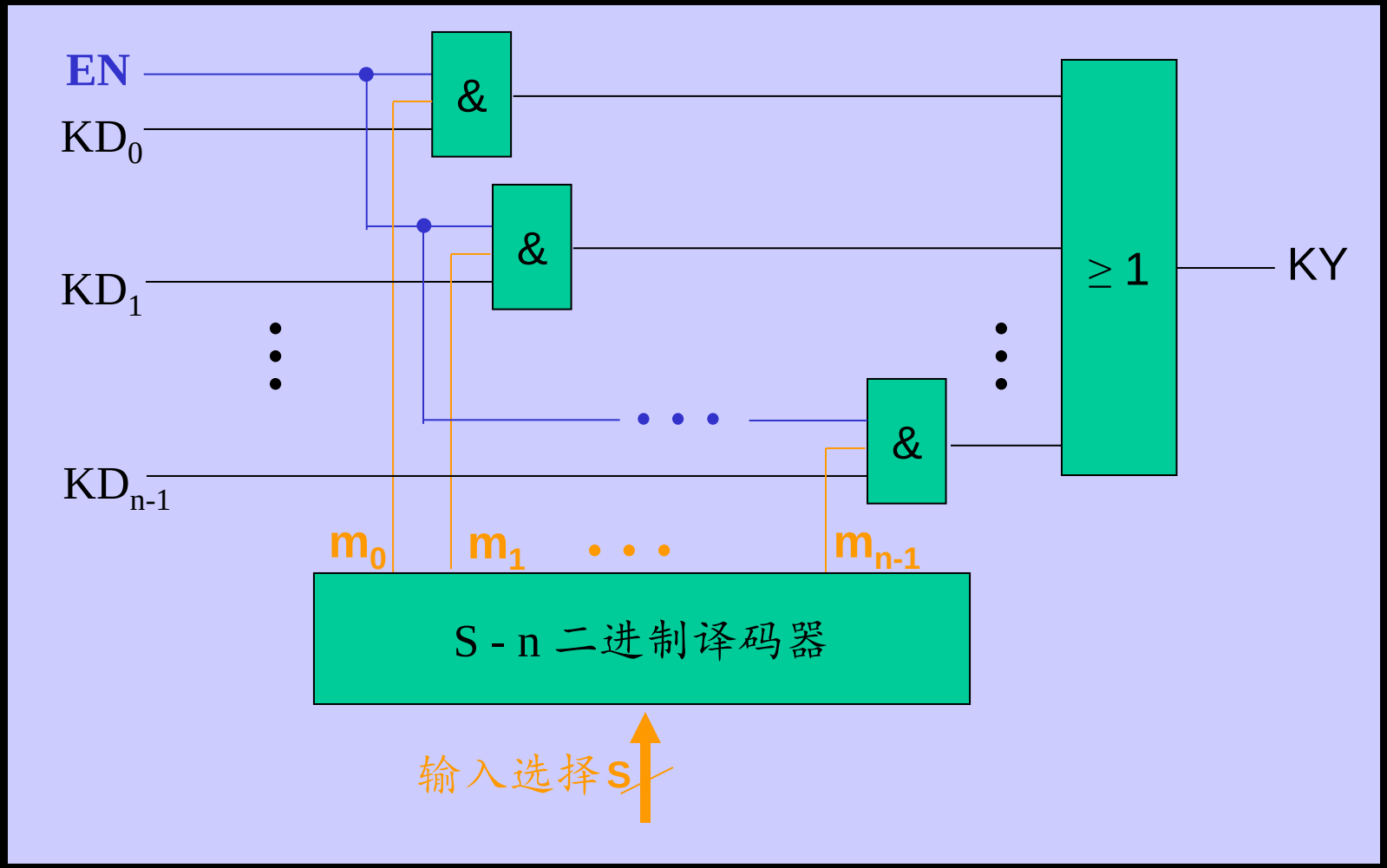
S与 n 的关系为：$n = 2^s$　　(或 $S = \log_2 n$)

S 位选择信号有 2^s 种组合(即最小项)。每一种组合对应选择 n ($= 2^s$)组输入源数据中的一组。逻辑表达式为：

$$KY = \sum_{i=0}^{n-1} EN \cdot m_i \cdot KD_i \qquad K = 1，2，\ldots，b$$

式中：KY为输出位，KD_i是第 i 组输入源数据的第K位，m_i 是 S 位选择输入变量的最小项。

④ 多路选择器的原理图（某位输出多路选择器）



1）MSI 多路选择器

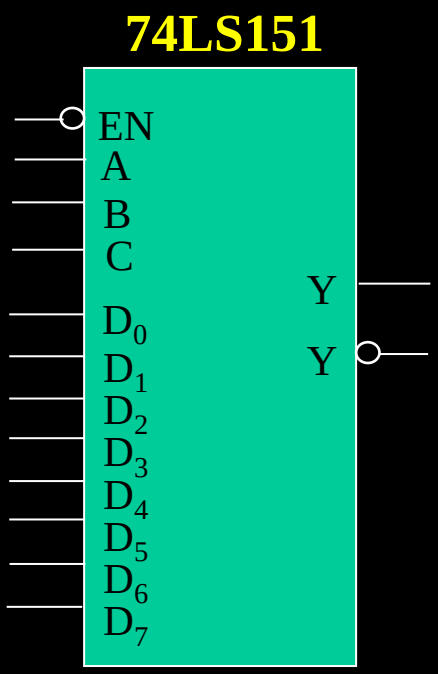
➢ 八输入 1 位输出多路选择器74LS151

- 一个低有效使能输入端/EN
- 三个选择输入端C、B、A
- 8 个数据输入端 D_7～D_0
- 2 个互反输出 Y、/Y

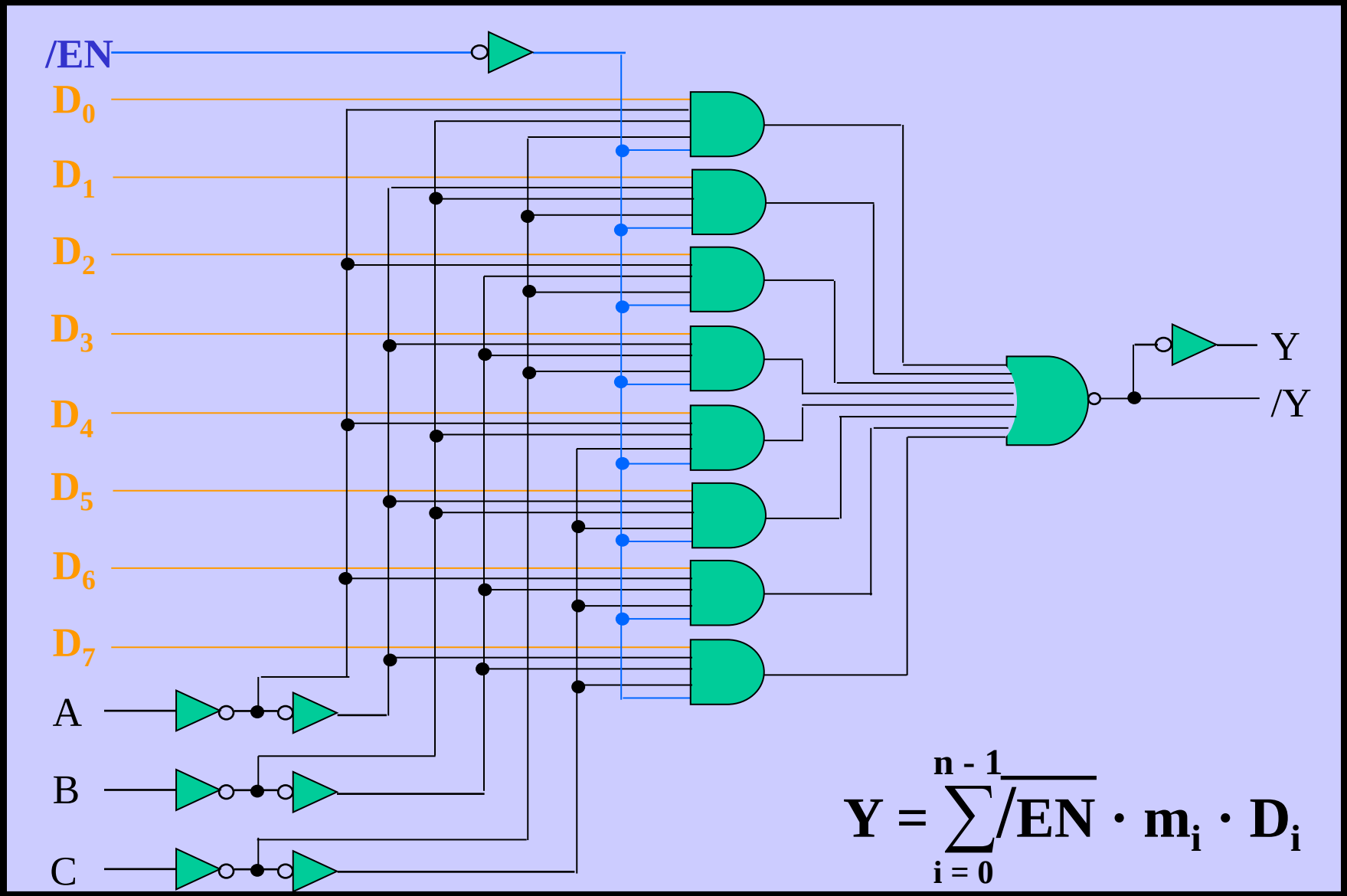
① 简化真值表

输 入				输 出	
/EN	C	B	A	Y	/Y
1	d	d	d	0	1
0	0	0	0	D_0	$\overline{D}_0$
0	0	0	1	D_1	$\overline{D}_1$
0	0	1	0	D_2	$\overline{D}_2$
0	0	1	1	D_3	$\overline{D}_3$
0	1	0	0	D_4	$\overline{D}_4$
0	1	0	1	D_5	$\overline{D}_5$
0	1	1	0	D_6	$\overline{D}_6$
0	1	1	1	D_7	$\overline{D}_7$

② 逻辑符号



③逻辑电路图

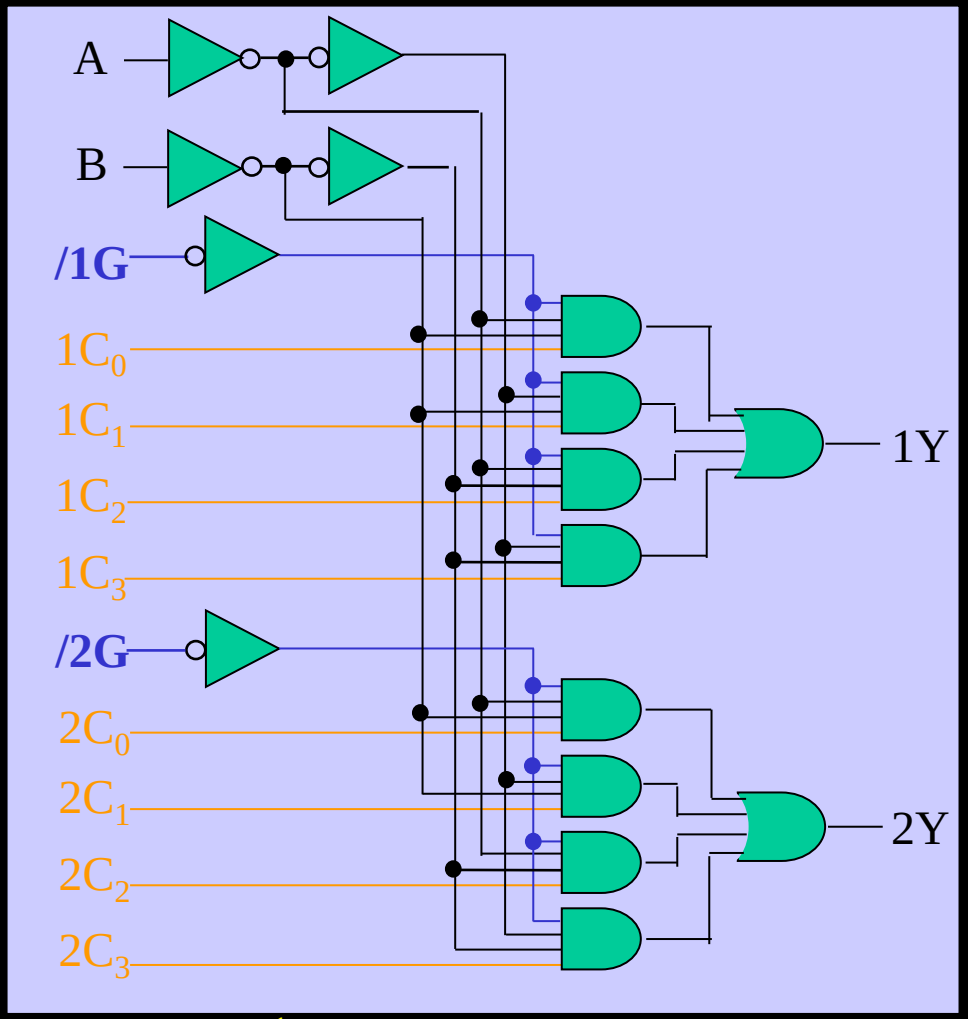

/EN
D0
D1
D2
D3
D4
D5
D6
D7
A
B
C
Y
/Y
$Y = \sum_{i=0}^{n-1} \overline{/EN \cdot m_i \cdot D_i}$

➢ 四输入 2 位多路选择器 74LS153

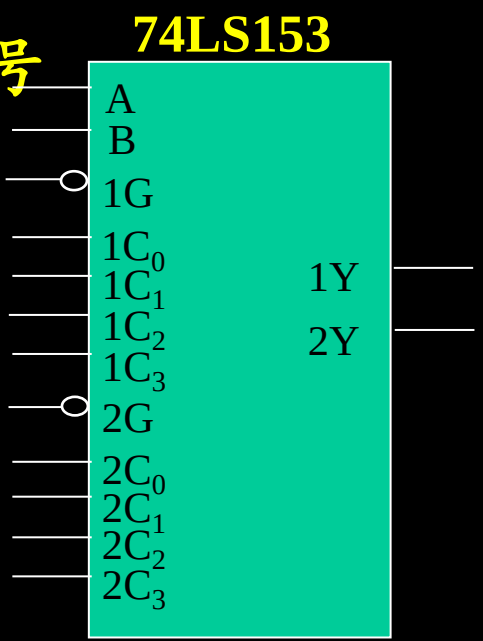
① 简化真值表

输入			输出	
/1G	/2G	B A	1Y	2Y
1		d d	0	0
0		0 0	$1C_0$	$2C_0$
0		0 1	$1C_1$	$2C_1$
0		1 0	$1C_2$	$2C_2$
0		1 1	$1C_3$	$2C_3$

② 逻辑电路图



③ 逻辑符号



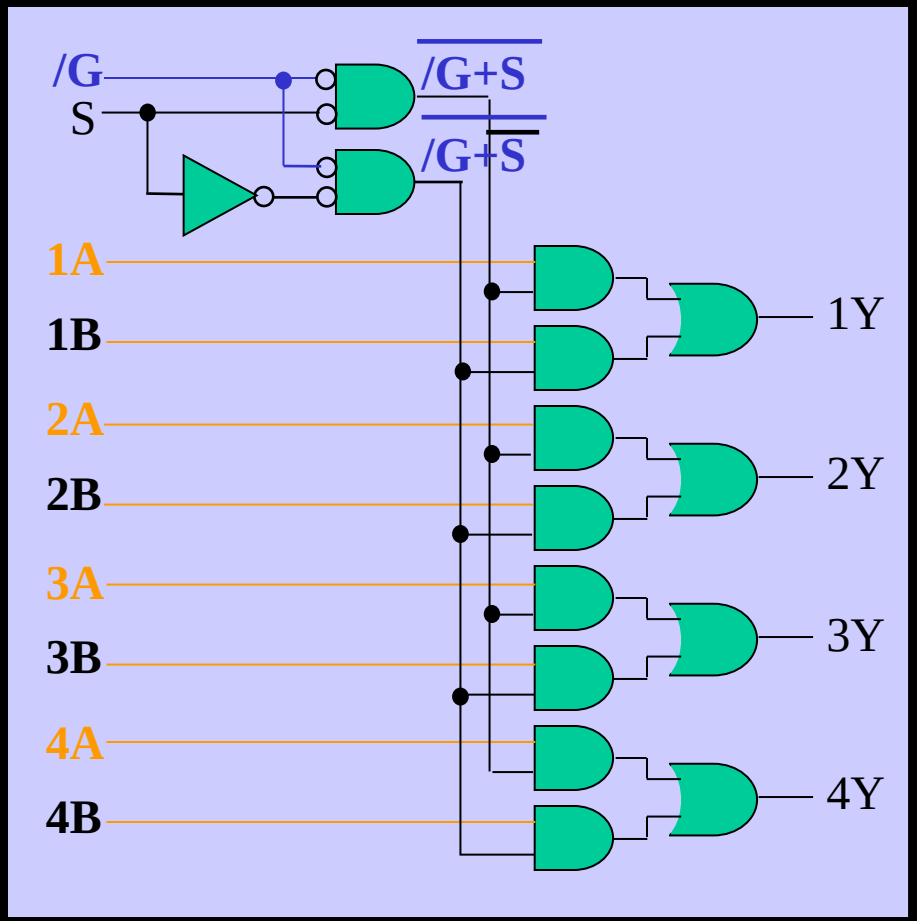
$$1Y = \sum_{i=0}^{n-1} \overline{/1G} \cdot m_i \cdot 1C_i \quad i = 0,1,2,3$$

➢ 二输入4位多路选择器 74LS157

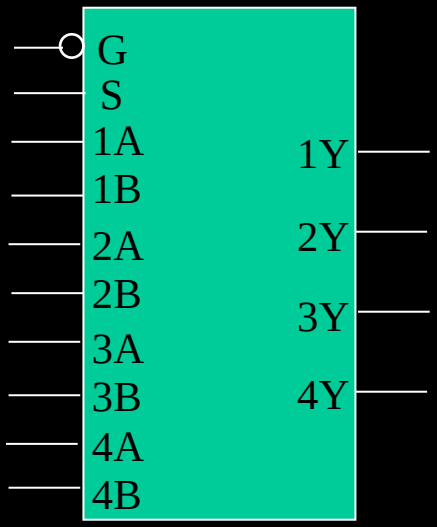
① 简化真值表

输 入		输 出			
/G	S	1Y	2Y	3Y	4Y
1	d	0	0	0	0
0	0	1A	2A	3A	4A
0	1	1B	2B	3B	4B

② 逻辑电路图



③ 逻辑符号



2）多路选择器的扩展

在实际应用中不一定能找到完全适用的多路选择器，这可能有 3 种情况：

- 输入组数 n 满足要求，但位数 b 不够；
- 输入组数 n 不够，而位数 b 满足要求；
- 组数 n 和位数 b 都不满足要求。

➢ MUX的输入组数 n 不够，而位数 b 满足

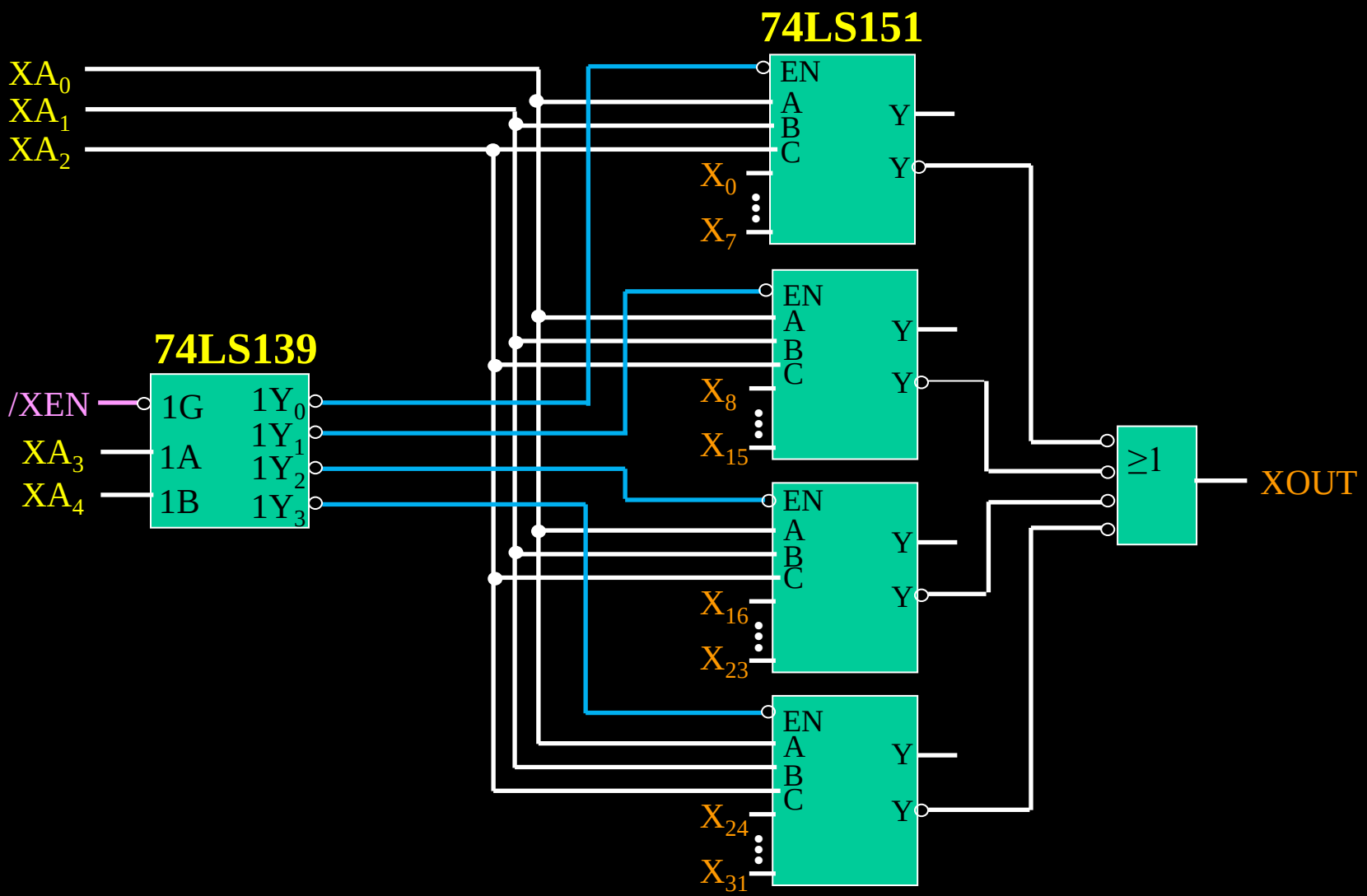
➢ 使用无三态输出的多路选择器及译码器

例：设计一个32输入 1 位多路选择器。

- 5个选择输入：XA_4～XA_0　32路输入：X_{31}～X_0
- 采用 4 个74LS151，每个器件可处理8个输入，这样将输入分为4组，每组由一个74LS151处理
- 选择输入的低三位XA_2～XA_0连接到 4 个74LS151的C、B、A端，决定组内选择
- 选择输入的高二位XA_4、XA_3通过一级2-4译码器1/2 74LS139产生 4 个输出，每个输出连接到一个74LS151的使能输入端

XA_4 XA_3	XA_2 XA_1 XA_0	Y
0 0	0 0 0	X_0
	…….	…
	1 1 1	X_7
0 1	0 0 0	X_8
	…….	…
	1 1 1	X_{15}
1 0	0 0 0	X_{16}
	…….	…
	1 1 1	X_{23}
1 1	0 0 0	X_{24}
	…….	…
	1 1 1	X_{31}

用74LS151组成的 32输入 1 位多路选择器

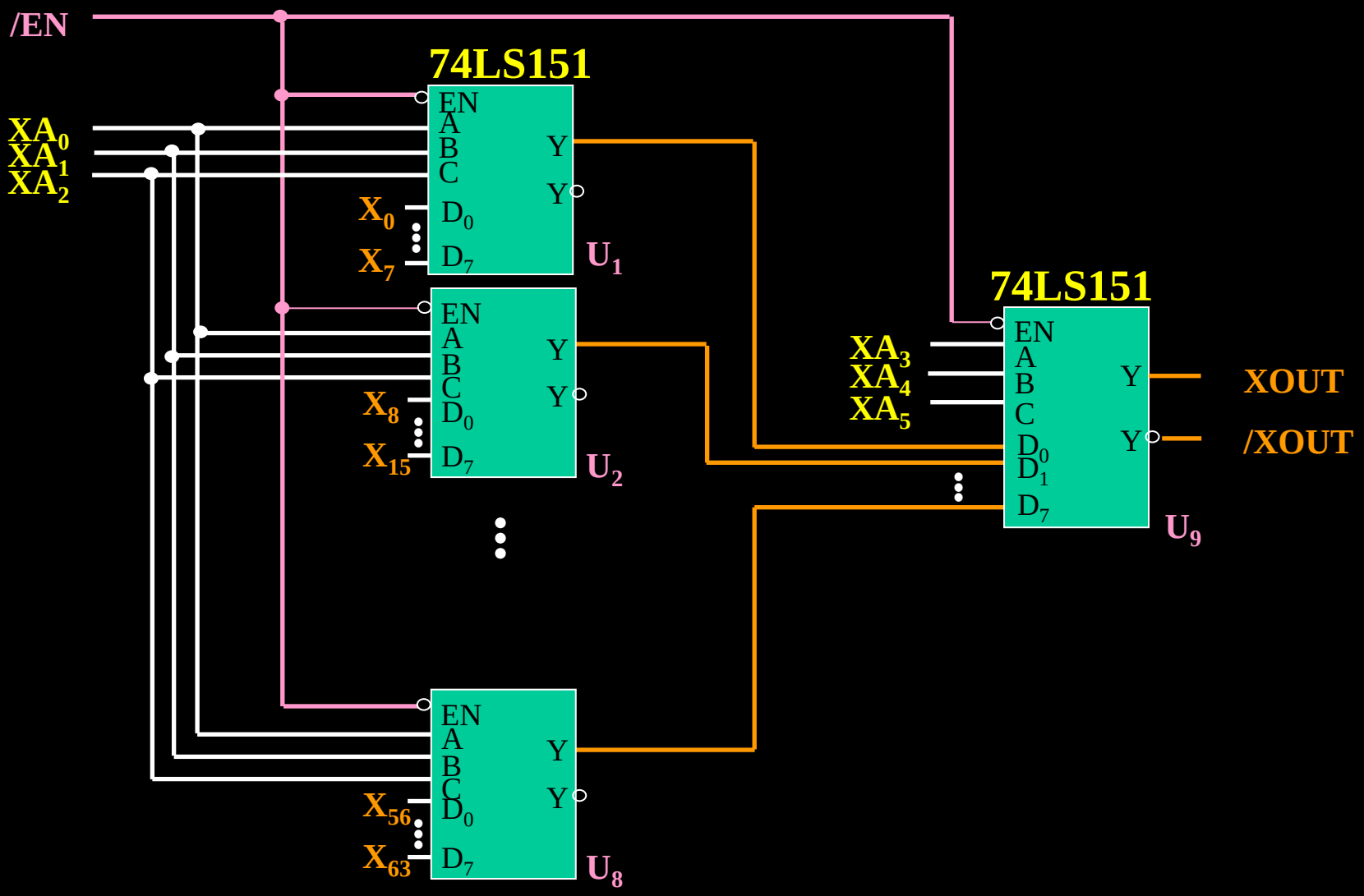


➢ 采用多级MUX的树形结构

- 将多路选择器MUX分级连接，低一级(前一级) MUX的输出作为其高一级(后一级) MUX的数据输入
- 用选择输入信号的低位控制低一级MUX，高位控制高一级MUX
- 各级的使能输入可以同一控制

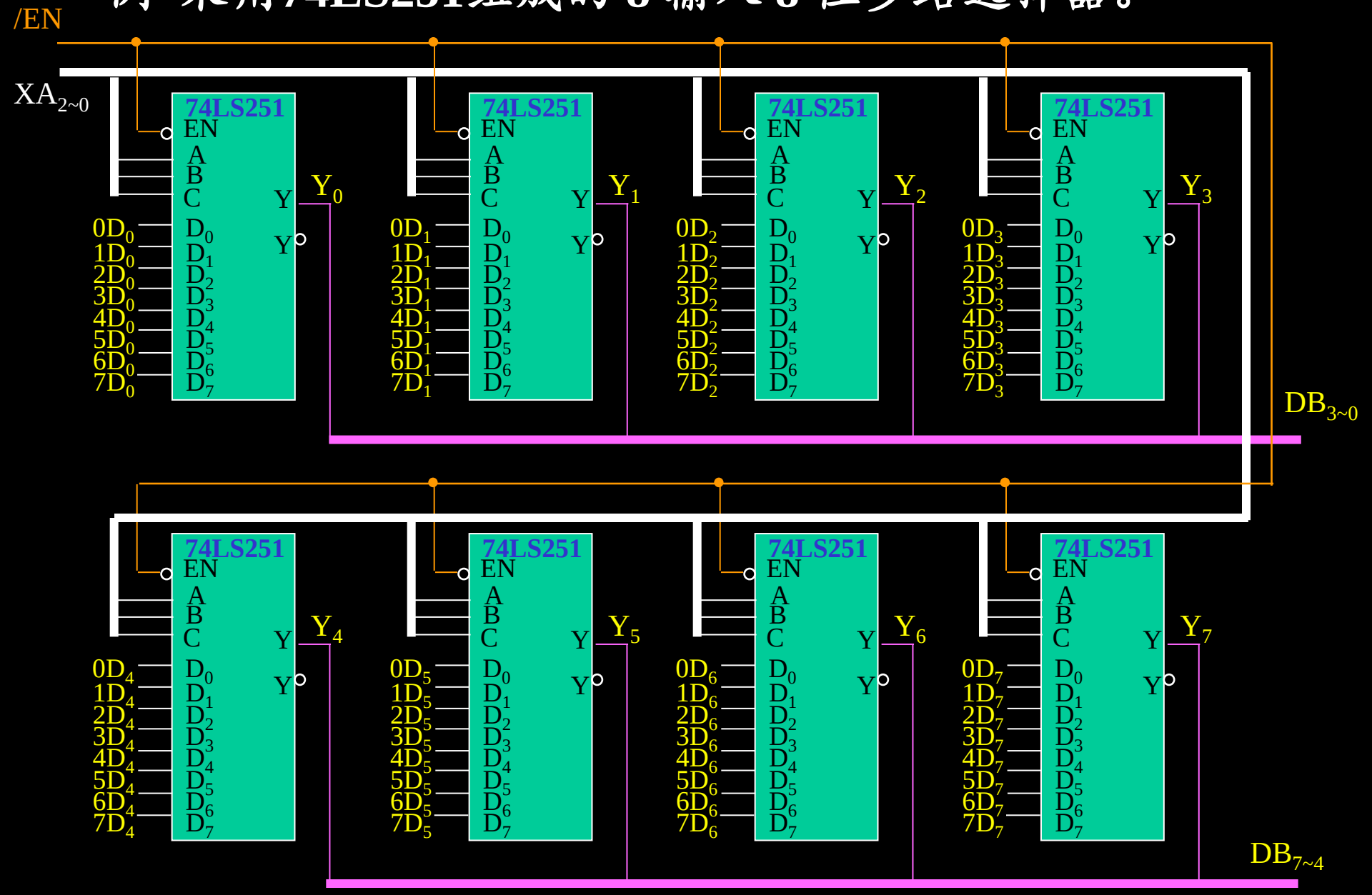
例 采用多级树形结构组成64输入 1 位多路选择器

64输入1位多路选择器



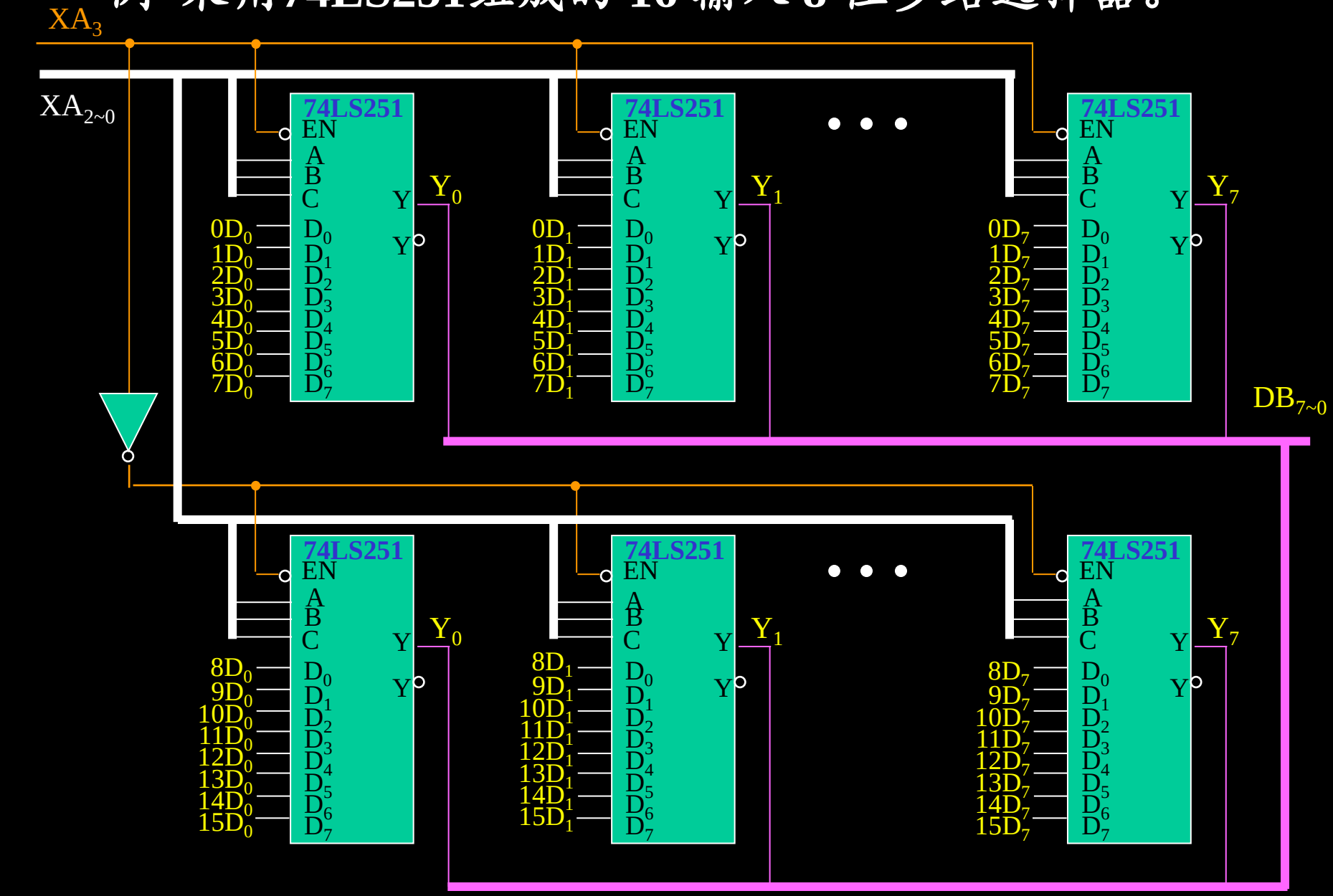
阅读➢ **MUX的输入组数 n 满足，位数 b 不满足**

例 采用74LS251组成的 8 输入 8 位多路选择器。



阅读➢ **MUX的输入组数 n 、位数 b 均不满足**

例 采用74LS251组成的 16 输入 8 位多路选择器。



3）用多路选择器实现任意组合逻辑函数

例1　$F(x,y,z) = \sum m^3(1,2,6,7) = m_1 + m_2 + m_6 + m_7$

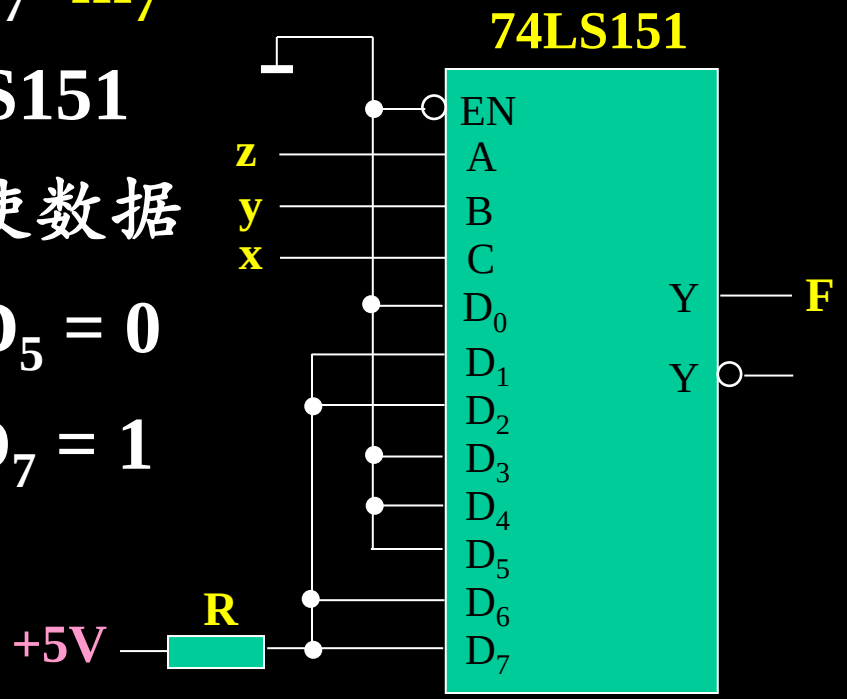
① 选择 S = 3 的MUX 74LS151，则：

$$F = D_0 \cdot m_0 + D_1 \cdot m_1 + D_2 \cdot m_2 + D_3 \cdot m_3 + D_4 \cdot m_4 + D_5 \cdot m_5 + D_6 \cdot m_6 + D_7 \cdot m_7$$

把 x、y、z 分别连到 74LS151 的 C、B、A 选择端，并使数据输入端为：$D_0 = D_3 = D_4 = D_5 = 0$

$D_1 = D_2 = D_6 = D_7 = 1$

则输出端Y的输出即为F。



$$Y = \sum_{i=0}^{n-1} EN \cdot m_i \cdot D_i, \quad K = 1$$

②用“四选1”多路选择器实现该三变量逻辑函数

将函数 F 改写成变量表达式： $F(x,y,z) = \sum m^3(1,2,6,7)$

$$= \bar{x}\,\bar{y}\,z + \bar{x}\,y\,\bar{z} + x\,y\,\bar{z} + x\,y\,z$$

$$= \bar{x}\,\bar{y}\,z + \bar{x}\,y\,\bar{z} + x\,y$$

$$= (\bar{x}\,\bar{y})\cdot z + (\bar{x}\,y)\cdot\bar{z} + (x\,\bar{y})\cdot 0 + (x\,y)\cdot 1$$

x、y作为地址选择变量，z、$\bar{z}$、0、1

作为MUX的源数据输入D，则有：

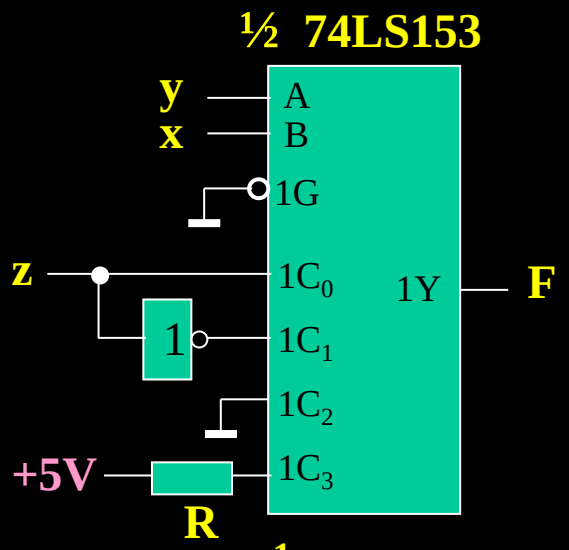
$$F = \sum_{i=1}^{3} m_i D_i$$

$$= m_0 D_0 + m_1 D_1 + m_2 D_2 + m_3 D_3$$

式中 m_i 为x、y的最小项， D_i为：

$D_0 = z$， $D_1 = \bar{z}$， $D_2 = 0$， $D_3 = 1$

电路逻辑图如图所示。



$$1Y = \sum_{i=0}^{n-1} /1G \cdot m_i \cdot 1C_i$$

例2 $F(x,y,z) = \sum m^3(1,2,3,6)$

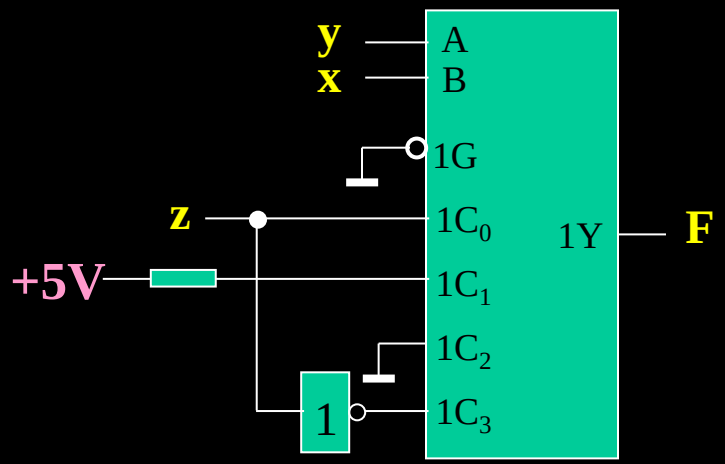
选用 4 输入 1 位多路选择器74LS153。

① 列出函数F的真值表

$$1Y = \sum_{i=0}^{n-1} /\overline{1G} \cdot m_i \cdot 1C_i$$

m_i	x y	z	D_i
m_1	0 0	1	$C_0 = z$
m_2	0 1	0	$C_1 = 1$
m_3	0 1	1	
m_6	1 1	0	$C_3 = \bar{z}$

½ 74LS153



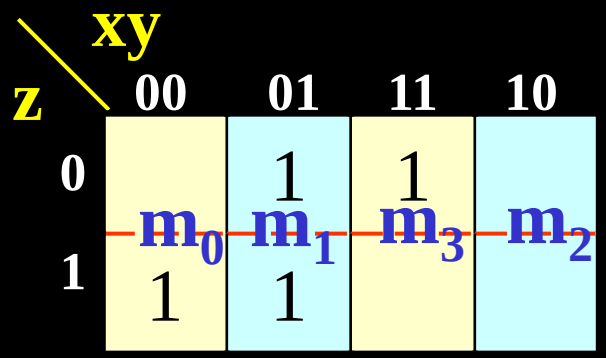
A \ B	0	1
0	C_0	C_2
1	C_1	C_3

y \ x	0	1
0	z	0
1	1	$\bar{z}$

$$F = \bar{x}\,\bar{y}\,z + \bar{x}\,y\,\bar{z} + \bar{x}\,y\,z + x\,y\,\bar{z}$$
$$= (\bar{x}\,\bar{y}) \cdot z + (\bar{x}\,y) \cdot 1 + (x\,y) \cdot \bar{z}$$

例2 $F(x,y,z) = \sum m^3(1,2,3,6)$

② 列出函数F的卡诺图



$D_0 = z$

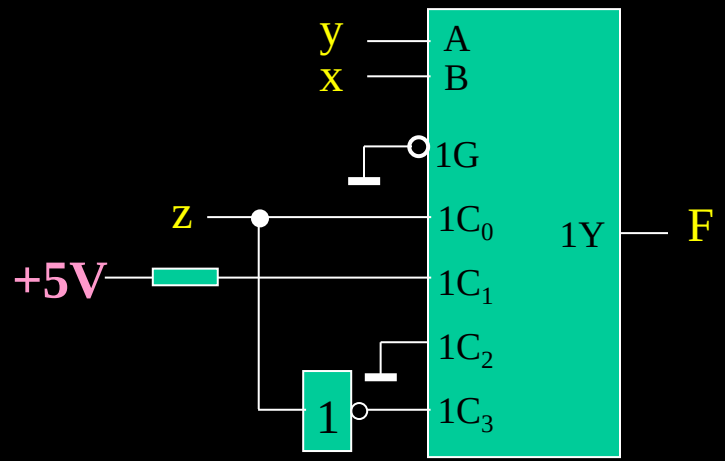
$D_1 = 1$

$D_2 = 0$

$D_3 = \overline{z}$

$$1Y = \sum_{i=0}^{n-1} \overline{/1G} \cdot m_i \cdot 1C_i$$

½ 74LS153



$$F = \overline{x}\,\overline{y}\,z + \overline{x}\,y\,\overline{z} + \overline{x}\,y\,z + x\,y\,\overline{z}$$

$$= (\overline{x}\,\overline{y}) \cdot z + (\overline{x}\,y) \cdot 1 + (x\,y) \cdot \overline{z}$$

例3 $F(w,x,y,z) = \sum m^4 (3,4,5,6,7,9,10,12,14,15)$

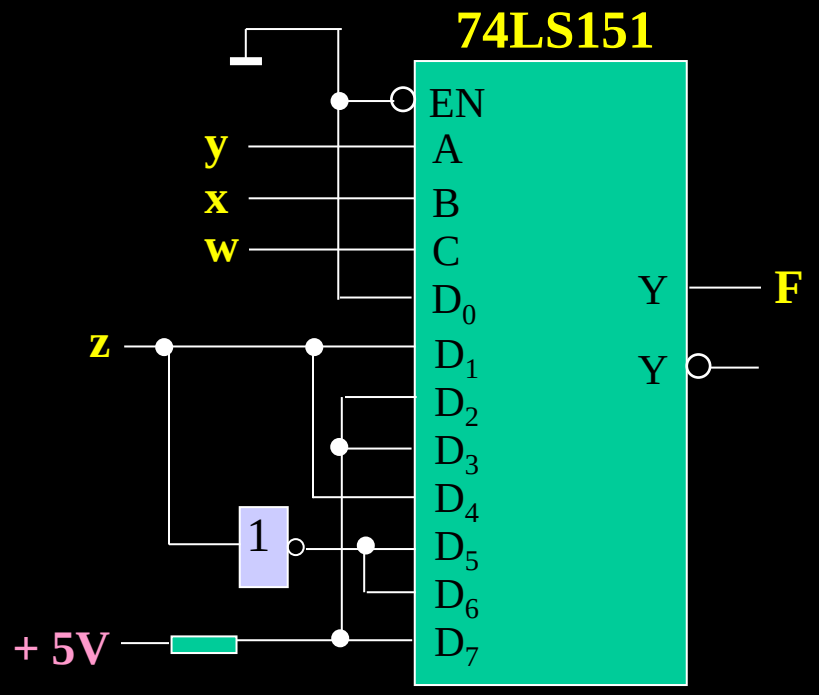
①选择有三个输入选择变量的 8 输入 1 位多路选择器 74LS151。将w、x、y分别接入地址端，z 接入数据端。

yz \ wx	00	01	11	10
00	m_0	1 m_2	1 m_6	m_4
01		1		1
11	1 m_1	1 m_3	1 m_7	m_5
10		1	1	1

$D_0 = 0$ $D_1 = z$ $D_2 = 1$

$D_3 = 1$ $D_4 = z$ $D_5 = \overline{z}$

$D_6 = \overline{z}$ $D_7 = 1$



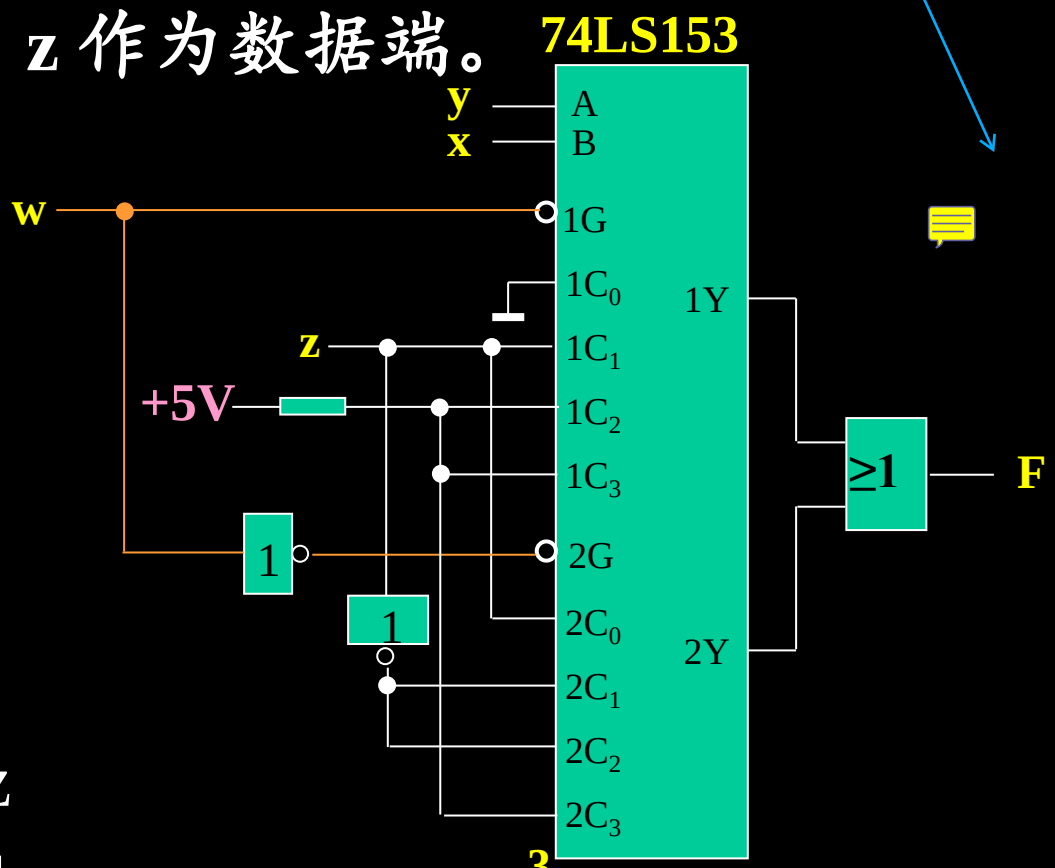
$$Y = \overline{\sum_{i=0}^{n-1} EN \cdot m_i \cdot D_i}$$

例3 $F(w,x,y,z) = \sum m^4(3,4,5,6,7,9,10,12,14,15)$

②选择4输入2位多路选择器74LS153。

将w、x、y作为地址端，z作为数据端。

yz \ wx	00	01	11	10
00	m_0	1 m_2 1	1 m_2	m_0 1
01				
11	1 m_1	1 m_3 1	1 m_3 1	m_1 1
10				



当w=0时：$D_0 = 0$ $D_1 = z$ $D_2 = 1$ $D_3 = 1$

当w=1时：$D_4 = z$ $D_5 = \overline{z}$ $D_6 = \overline{z}$ $D_7 = 1$

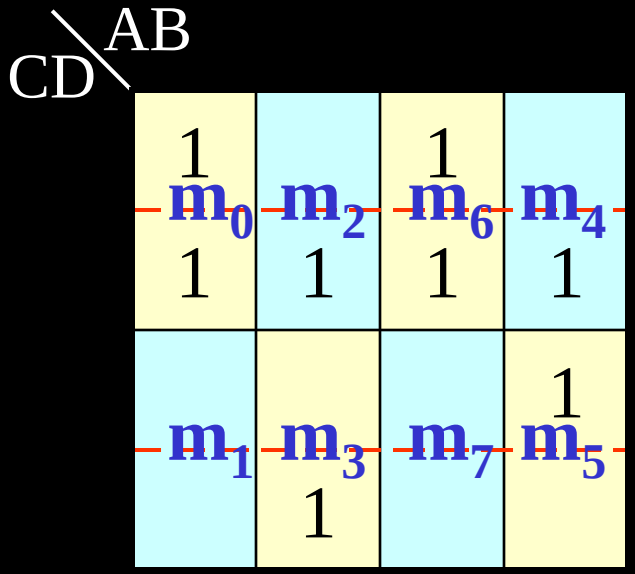
$$1Y = \sum_{i=0}^{3} \overline{w} \cdot m_i \cdot 1C_i$$

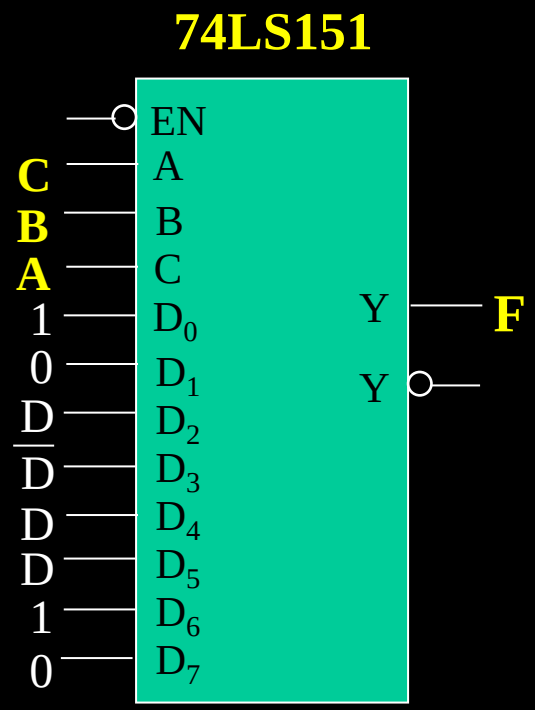
$$2Y = \sum_{i=0}^{3} w \cdot m_i \cdot 2C_i$$

例4 $F(A,B,C,D) = \sum m^4 (0,1,5,6,9,11,12,13)$

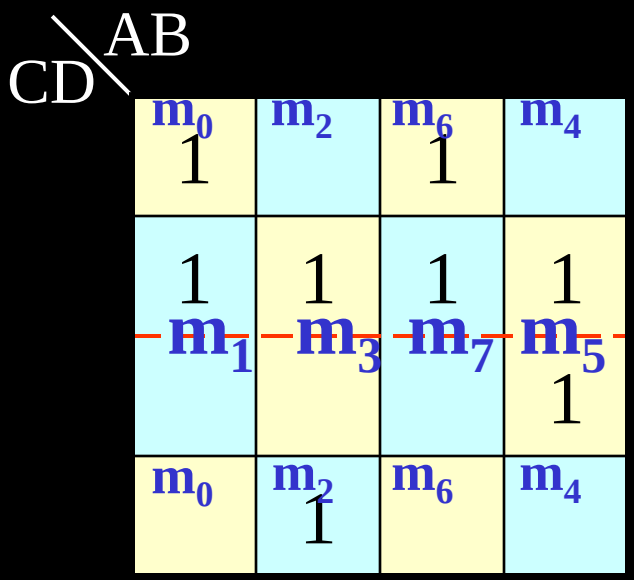
①选择A、B、C为地址端输入，

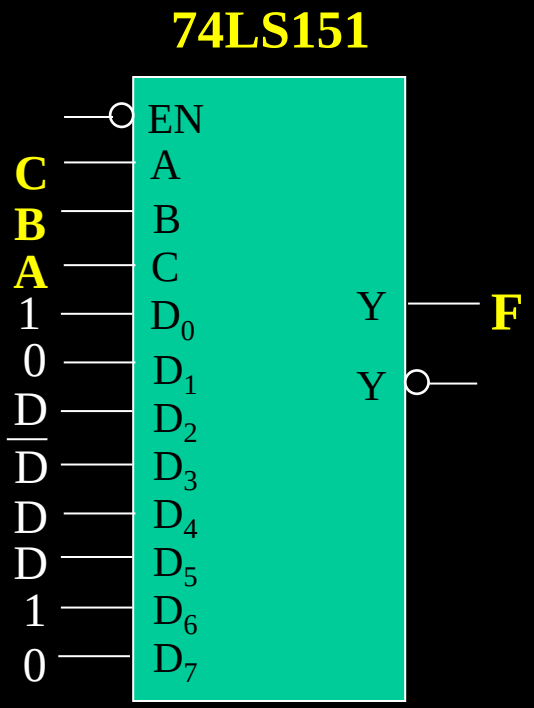
D为数据端输入。如图所示

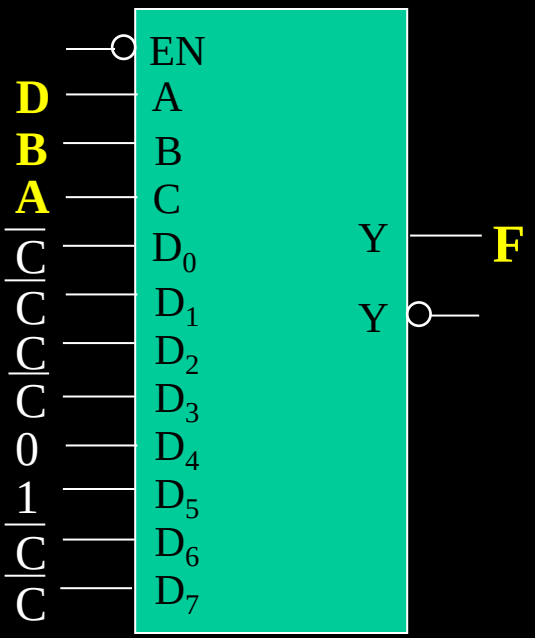




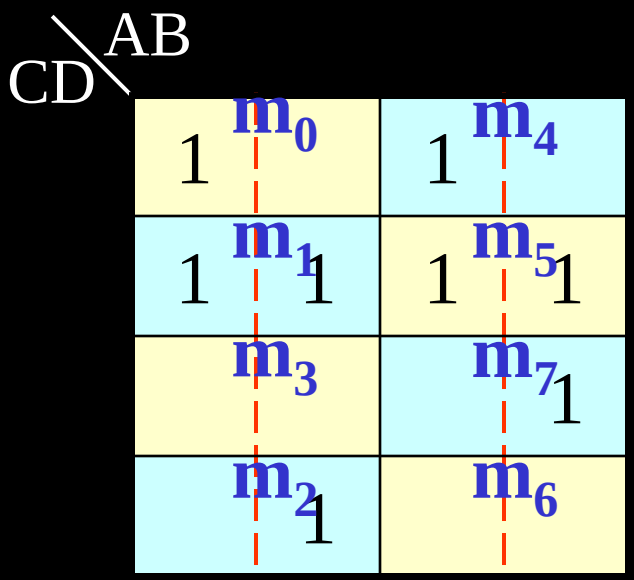
②选择A、B、D为地址端输入，C为数据端输入。

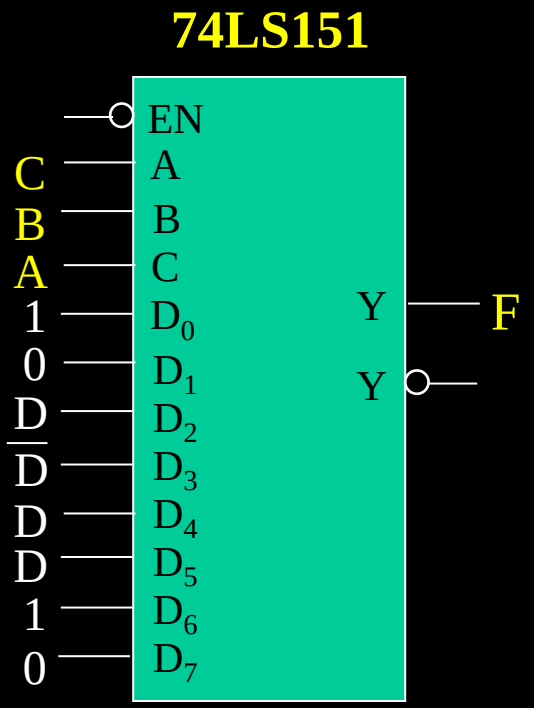


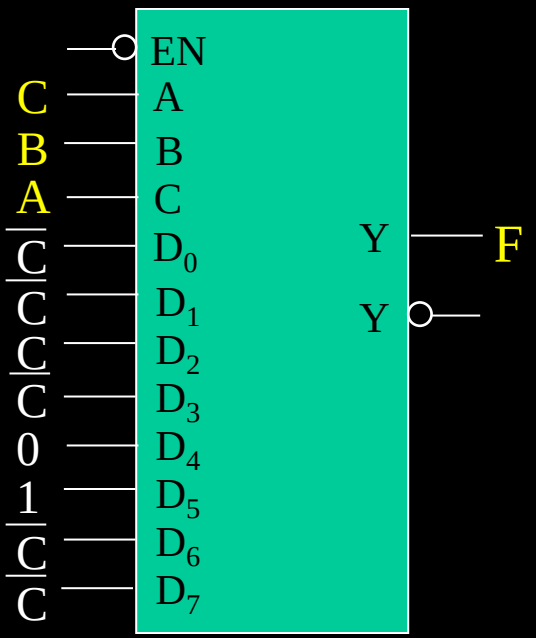


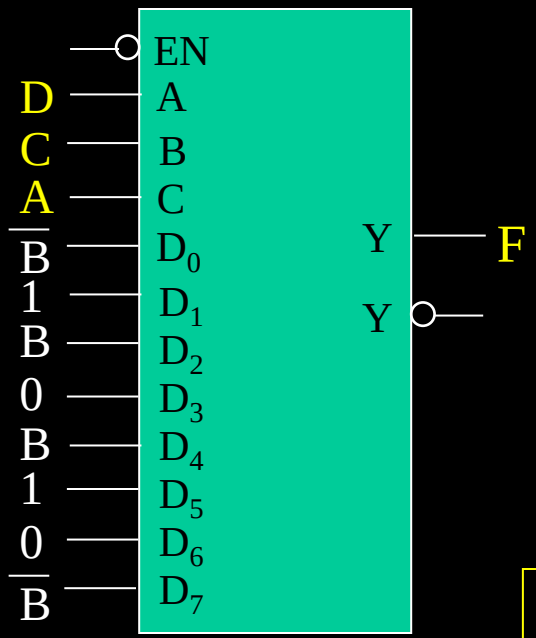


③选择A、C、D为地址端输入，B为数据端输入。

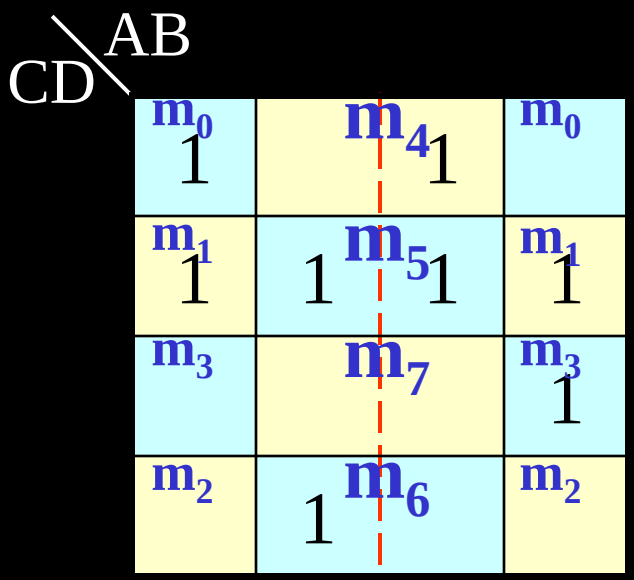




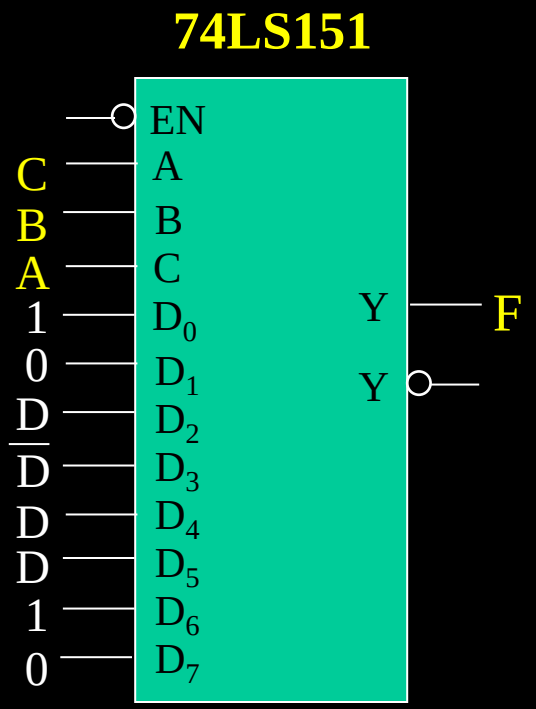


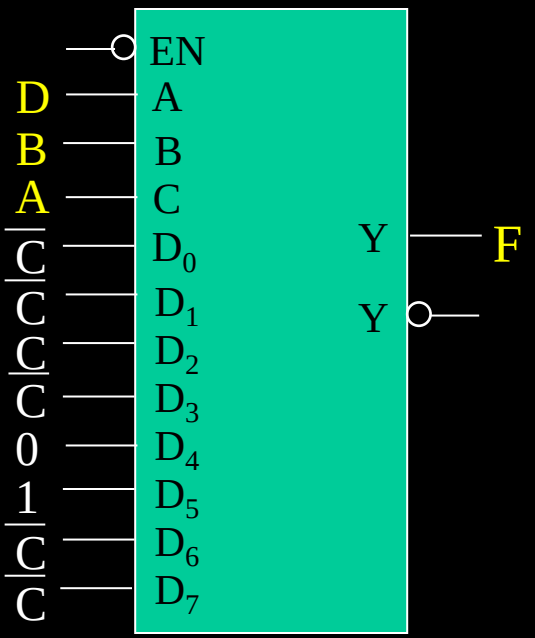


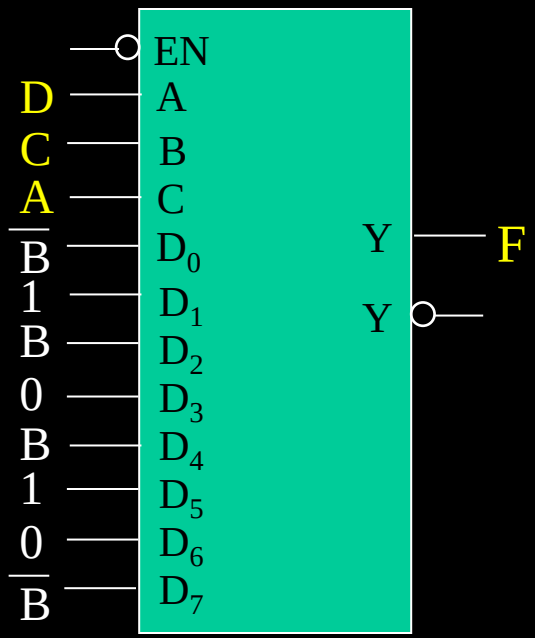
③选择B、C、D为地址端输入，
A为数据端输入。

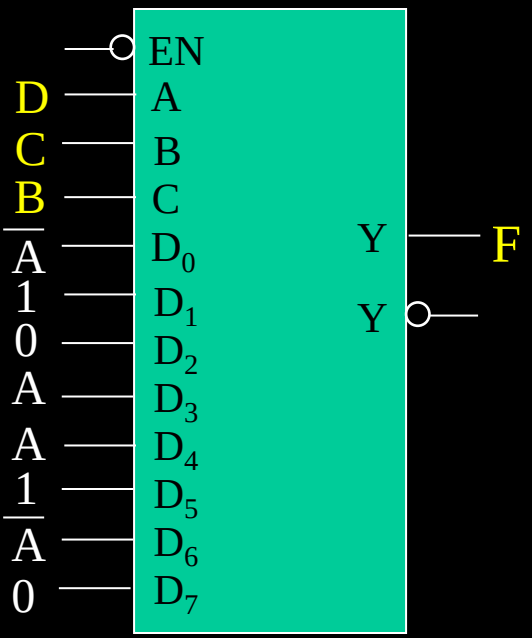


这是 24 种设计方案中的 4 种。





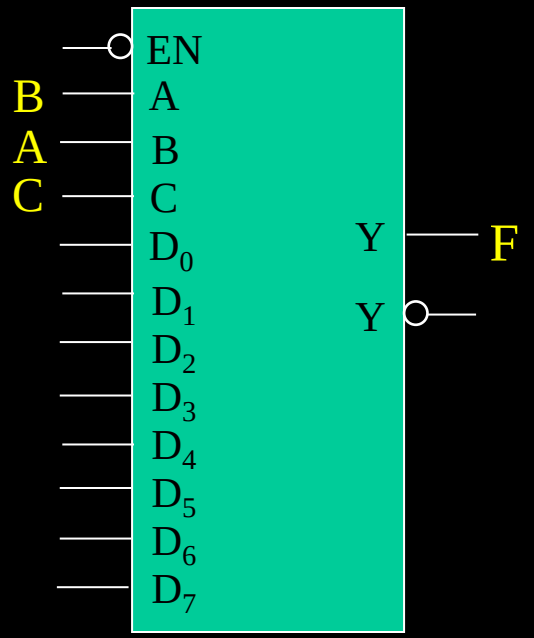




例5 $F(A,B,C,D) = \sum m^4(0,1,5,6,9,11,12,13)$

阅读

若选择C、A、B为地址端输入，D为数据端输入。由于改变了原有的变量的先后次序，如图所示，则称为“非常规顺序的设计”。



非常规顺序的设计要将函数的最小项按新的顺序重新进行编排，得到新的表达式和相应的卡诺图，然后进行设计。

$$F(A,B,C,D) = \sum m(0,1,5,6,9,11,12,13)$$

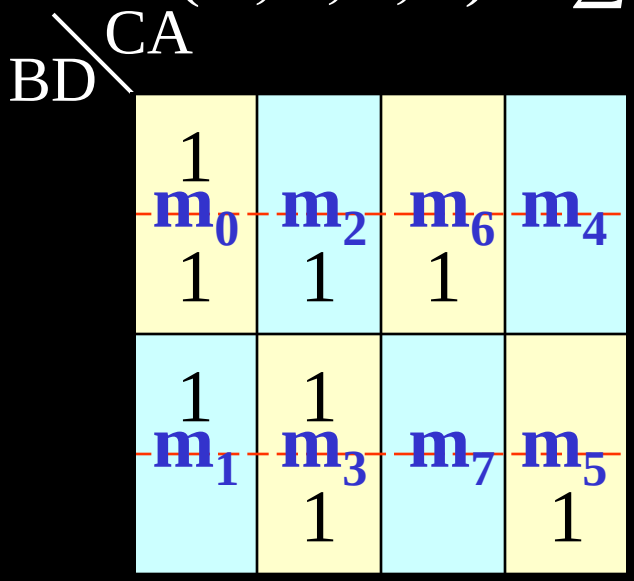
$$= \bar{A}\bar{B}\bar{C}\bar{D}+\bar{A}\bar{B}\bar{C}D+\bar{A}B\bar{C}\bar{D}+\bar{A}B\bar{C}D+A\bar{B}\bar{C}D+A\bar{B}CD+AB\bar{C}D+AB\bar{C}D$$

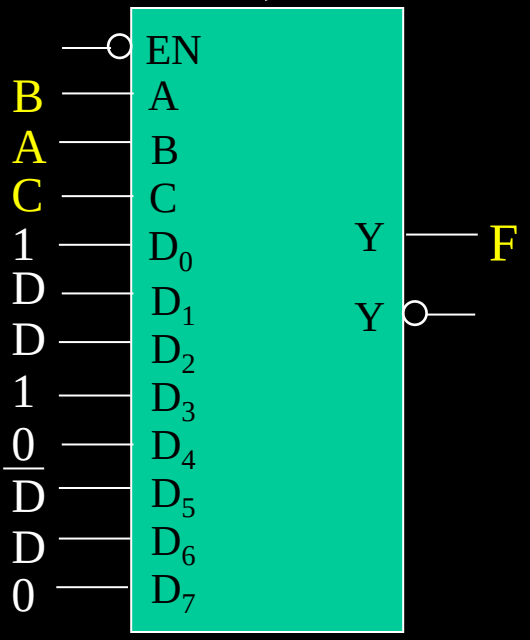
$$= \bar{C}\bar{A}\bar{B}\bar{D}+\bar{C}\bar{A}\bar{B}D+\bar{C}A\bar{B}D+C\bar{A}\bar{B}\bar{D}+\bar{C}A\bar{B}D+CA\bar{B}D+\bar{C}AB\bar{D}+\bar{C}ABD$$

得到： $F(C,A,B,D) = \sum m(0,1,3,5,6,7,10,13)$

阅读

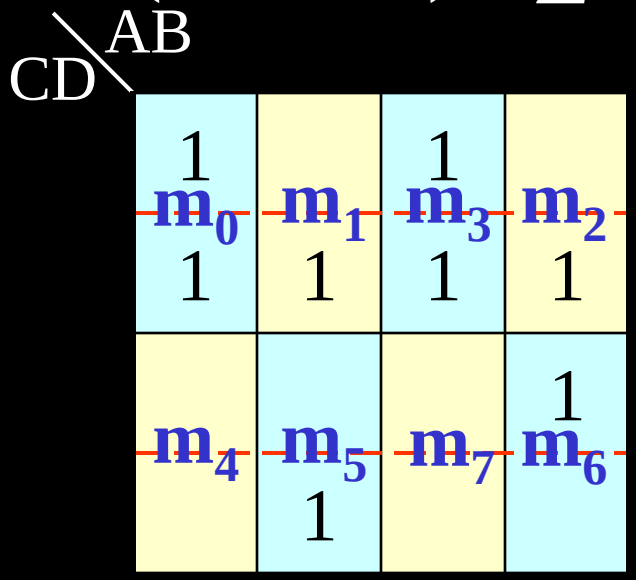
例5 $F(C,A,B,D)=\sum m(0,1,3,5,6,7,10,13)$

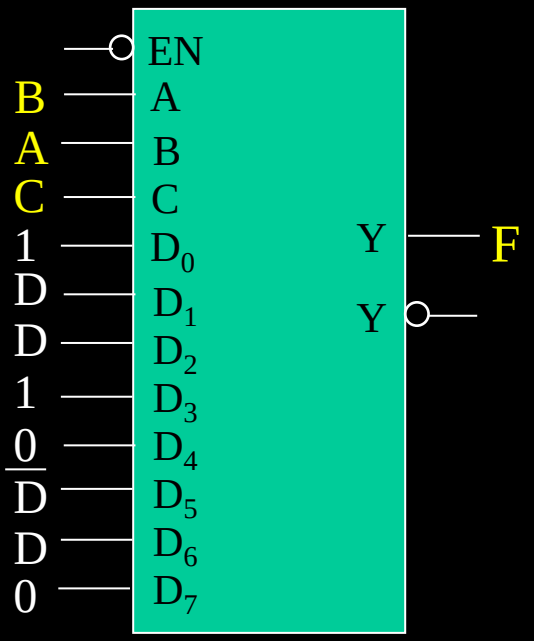




事实上，非常规顺序的设计也可以用原变量顺序的卡诺图进行分析，关键在于要能够按照新的顺序找出相应的最小项。

阅读 例5 $F(A,B,C,D) = \sum m^4 (0,1,5,6,9,11,12,13)$

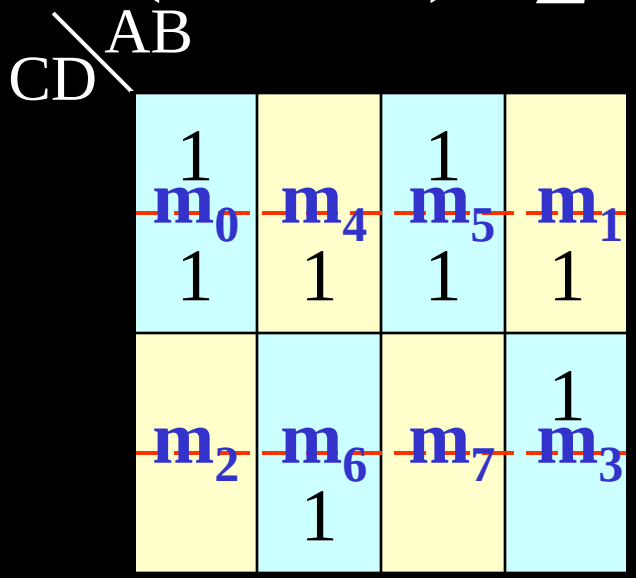


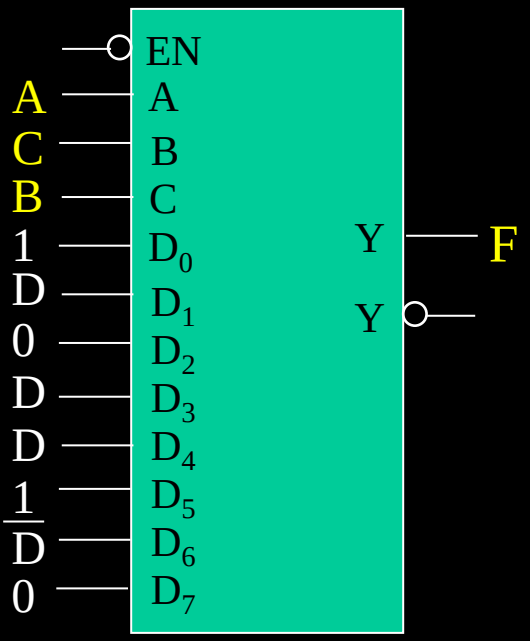


按照新的顺序 **CAB** 分别找出相应的 8 个最小项。

由此卡诺图得到的设计结果与前例是一致的。

阅读 例5 $F(A,B,C,D) = \sum m^4(0,1,5,6,9,11,12,13)$





按照新的顺序 **BCA** 分别找出相应的 8 个最小项。

2.4.3 三态缓冲器

三态是指器件的输出有三种状态：

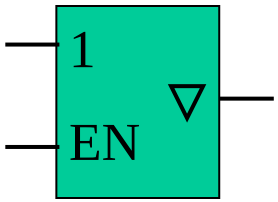
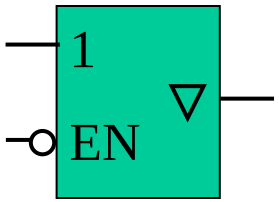
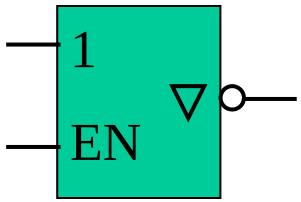
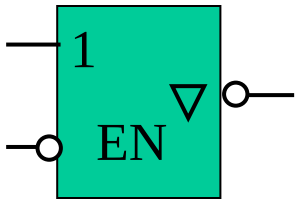
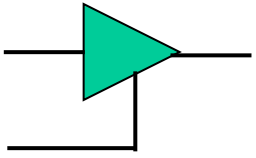
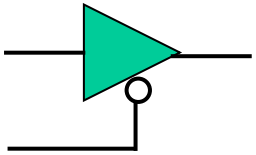
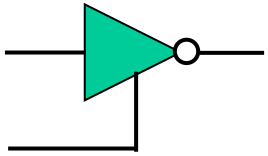
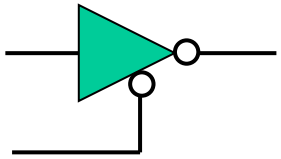
即**逻辑0**（L电平）、**逻辑1**（H电平）和**高阻抗**状态（或悬浮态）。

高阻态相当于隔断状态，即与所连的电路断开。高阻态时电阻无穷大，无电流流入或流出。

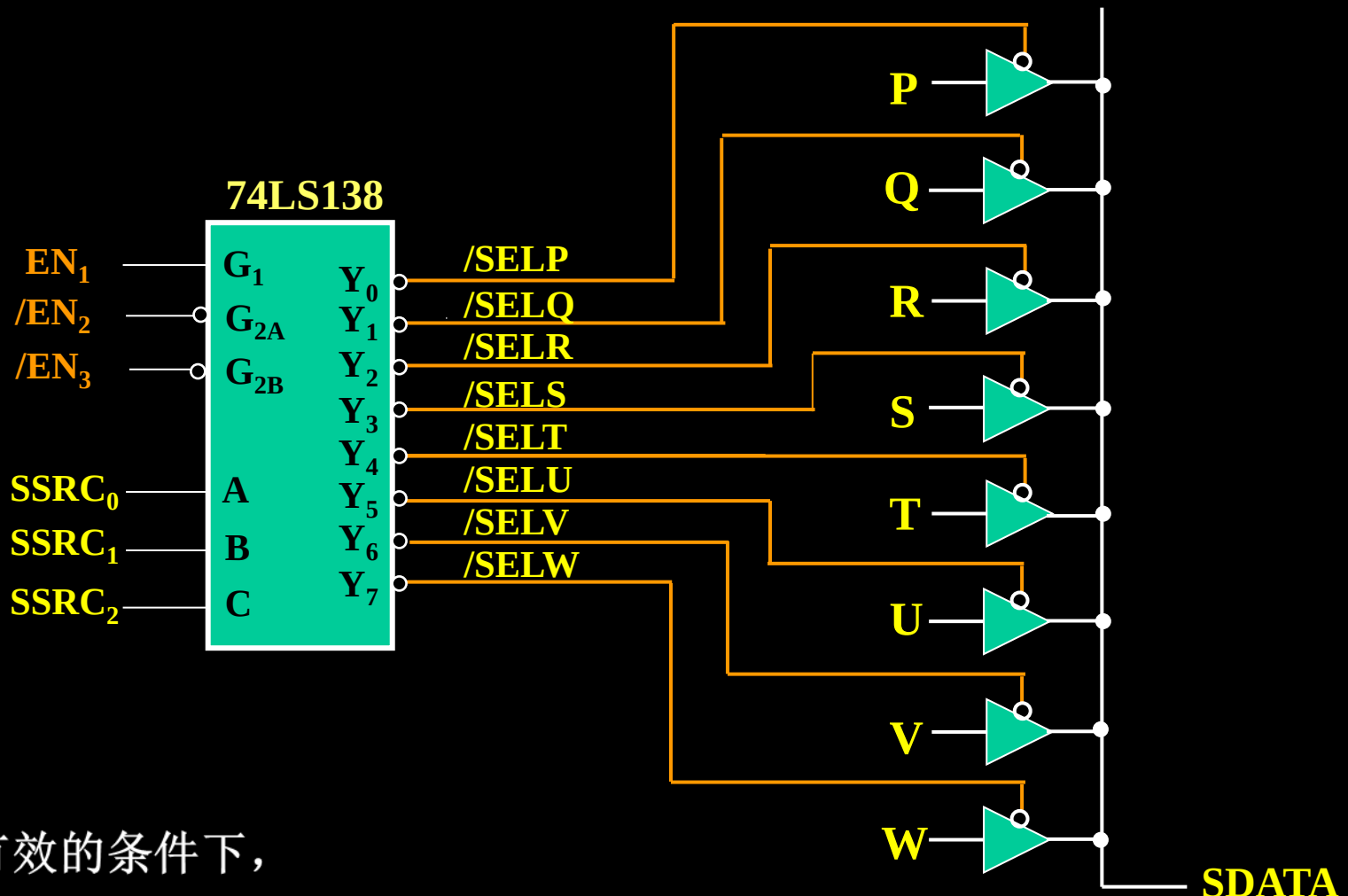
最基本的三态器件是**三态缓冲器**，又称为三态门或三态驱动器。使能输入有效时，器件有**正常逻辑状态**输出；当使能输入无效时输出就处于**高阻抗**状态。

三态缓冲器可使多个源数据**分时共享**一根公用线，为了避免多个源数据同时驱动共享线，则不能在使能一个源数据的同时使能另一个源数据。

三态缓冲器逻辑符号

矩形符号				
变形符号				
▽表示三态输出	原码输出 高有效使能	原码输出 低有效使能	反码输出 高有效使能	反码输出 低有效使能

8个数据源共享一根数据线



在使能有效的条件下，

相当于 $SDATA=\overline{/SELP}\cdot P+\overline{/SELQ}\cdot Q+\overline{/SELR}\cdot R+\overline{/SELS}\cdot S$

$+\overline{/SELT}\cdot T+\overline{/SELU}\cdot U+\overline{/SELV}\cdot V+\overline{/SELW}\cdot W$

1）标准的SSI三态缓冲器

74LS125 和 74LS126 都是四总线缓冲门，每一个缓冲门都有独立的使能端。在 74LS125 中是低有效使能，在 74LS126 中是高有效使能。

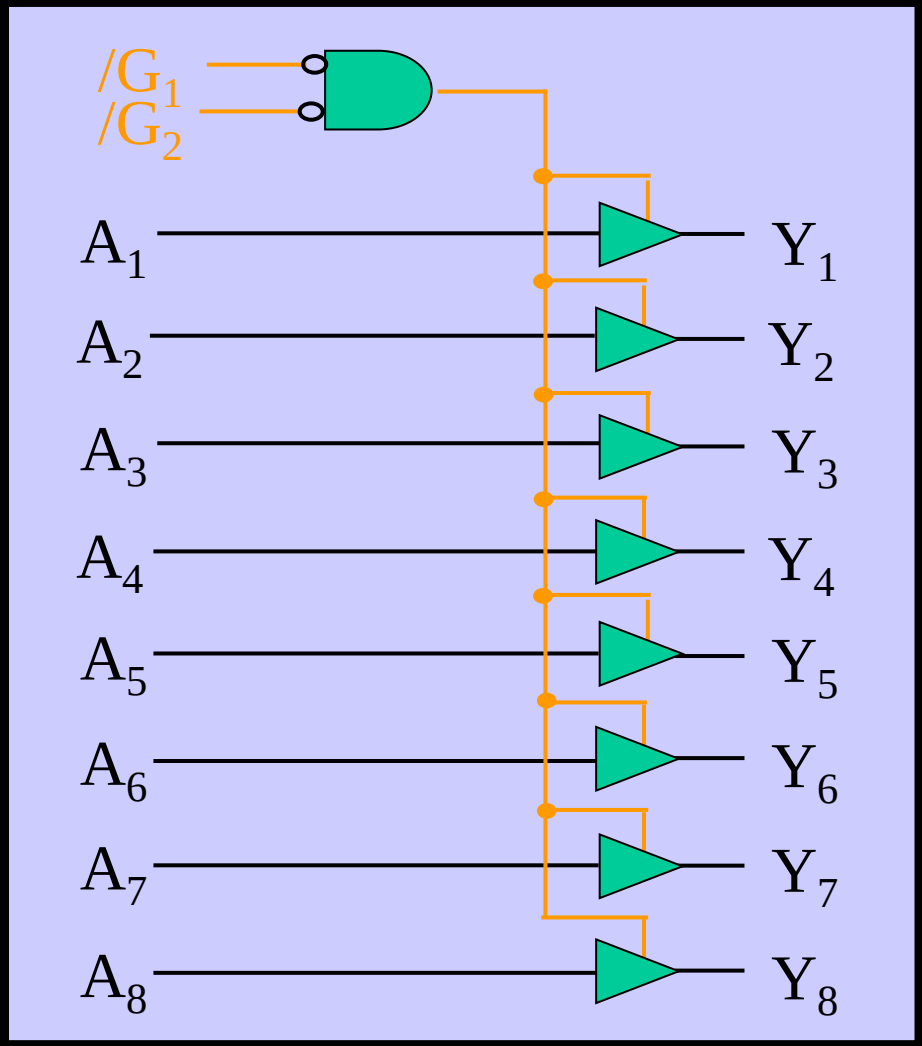
最常使用共享线的场合是**多位数据总线**。

例如： 在8位微处理机系统中，数据总线的宽度是8，并且外围器件通常**一次置8位数据到总线上**。这样外围器件都在同一时刻使能8个三态缓冲器，因此独立的使能输入端都多余了。

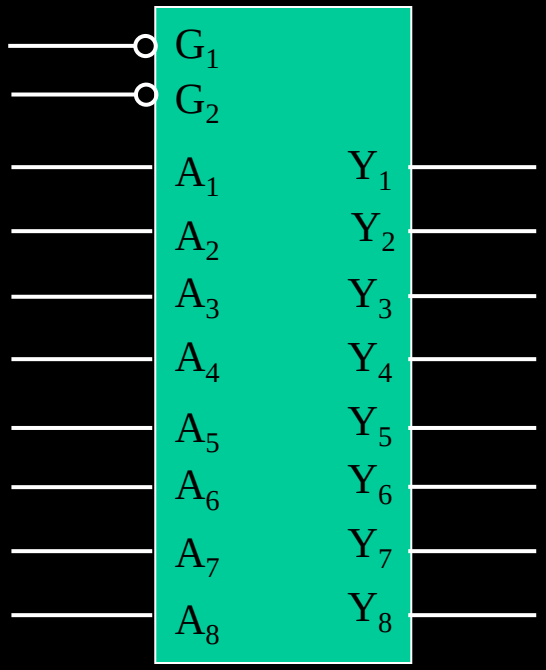
为减少总线应用中三态缓冲器的芯片数及连线，三态缓冲器中包含多个三态缓冲器时常常**共用使能输入**。

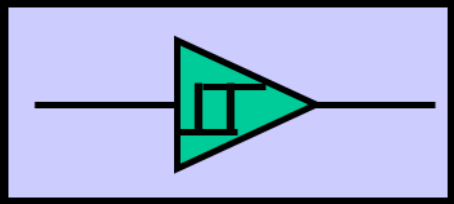
2） MSI 三态缓冲器---74LS541

多端口输入，MSI 74LS541为八三态缓冲器

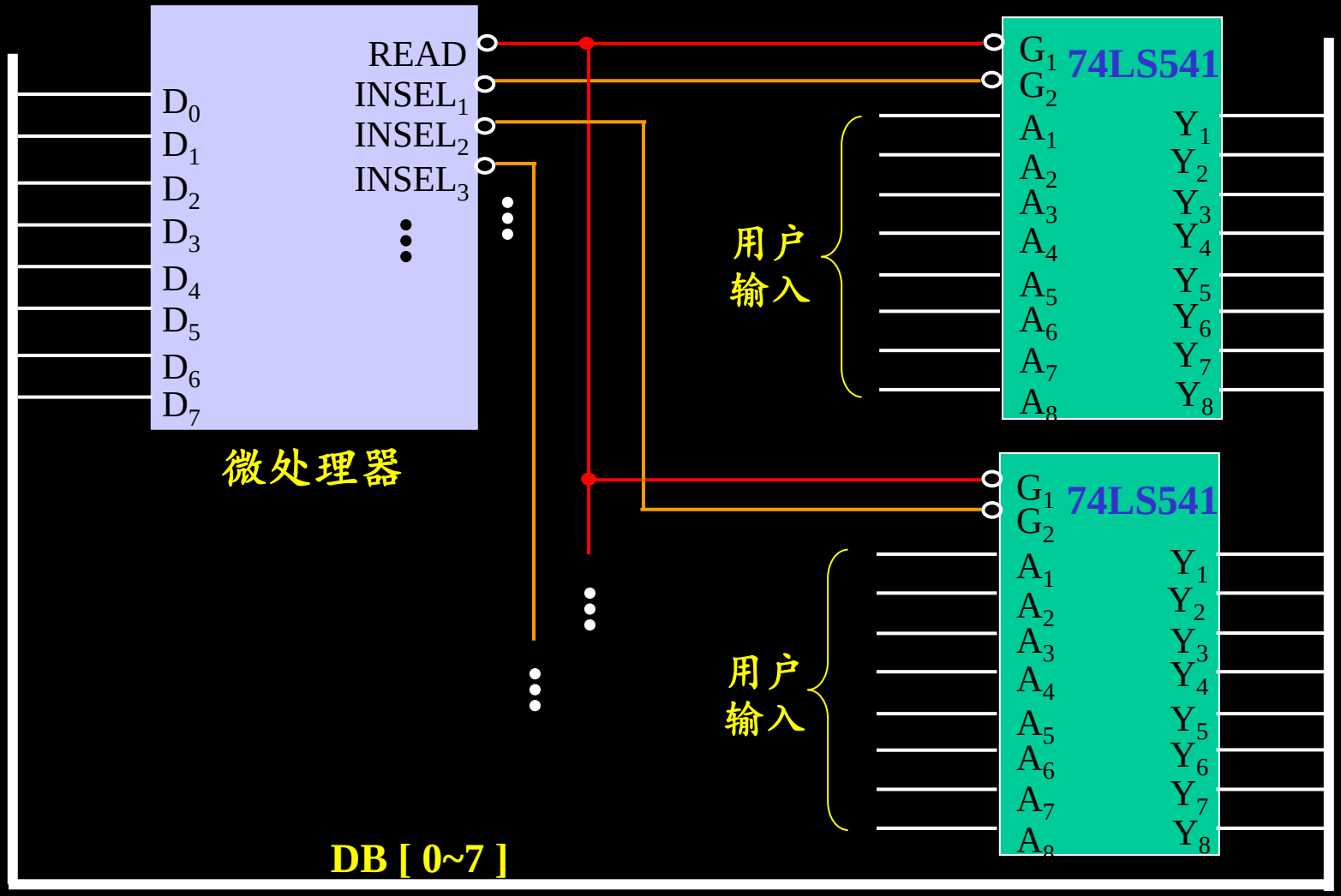


74LS541





例如：MSI 74LS541（八三态缓冲器）的应用

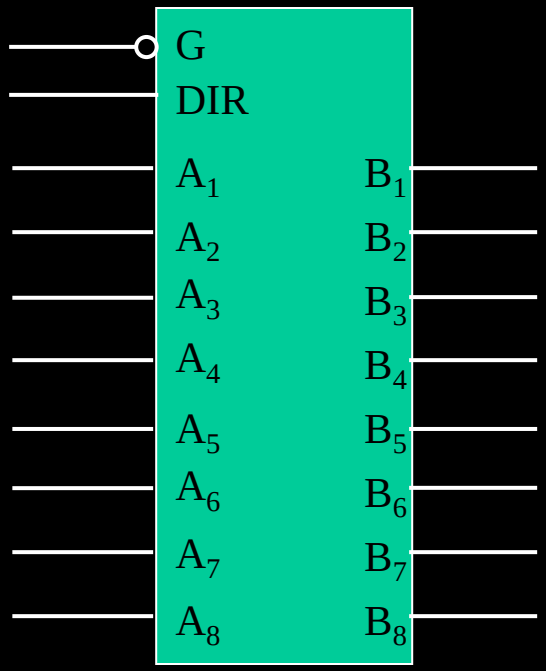


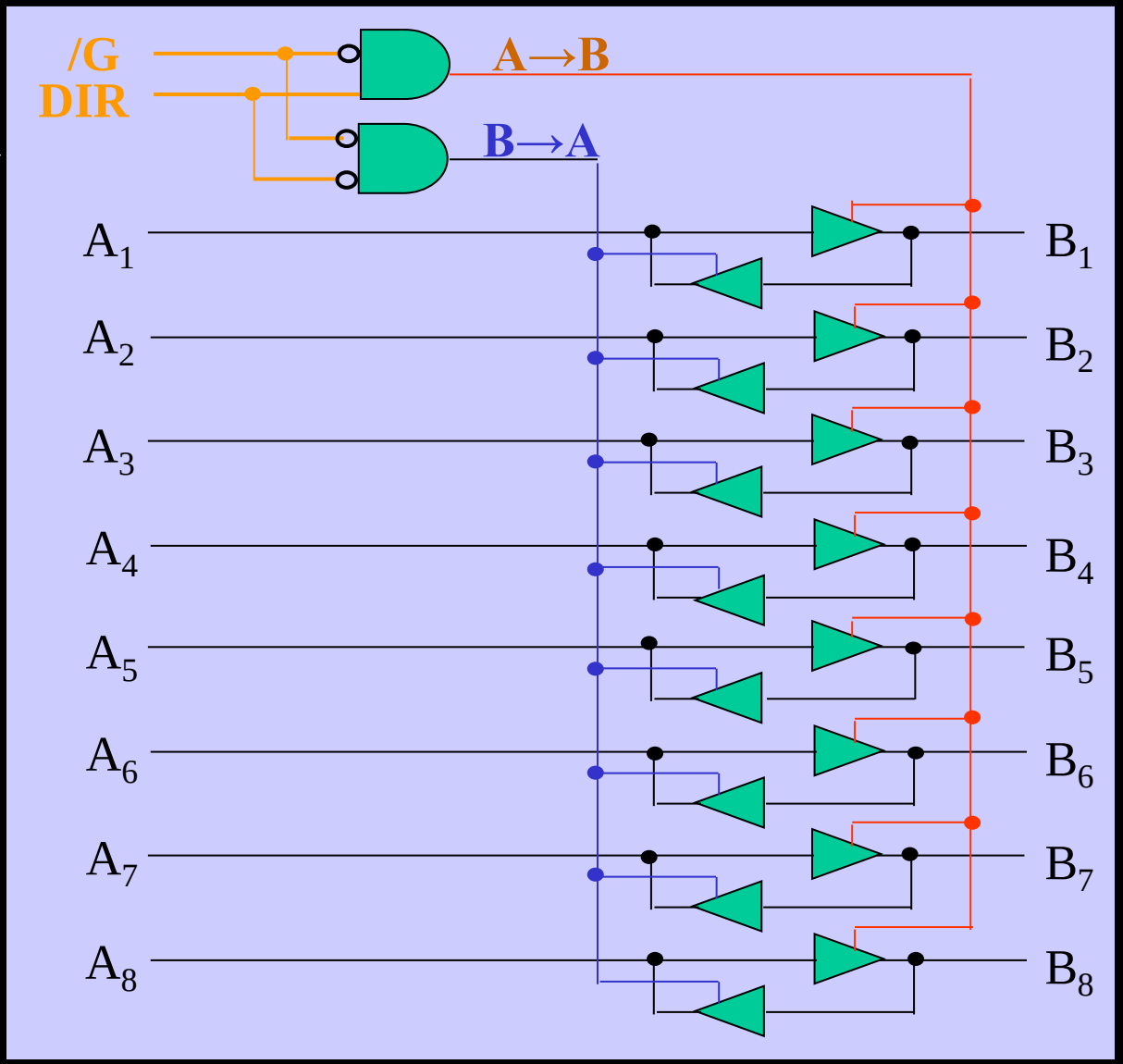
3)双向总线收发器

74LS245

八三态总线收发器

74LS245

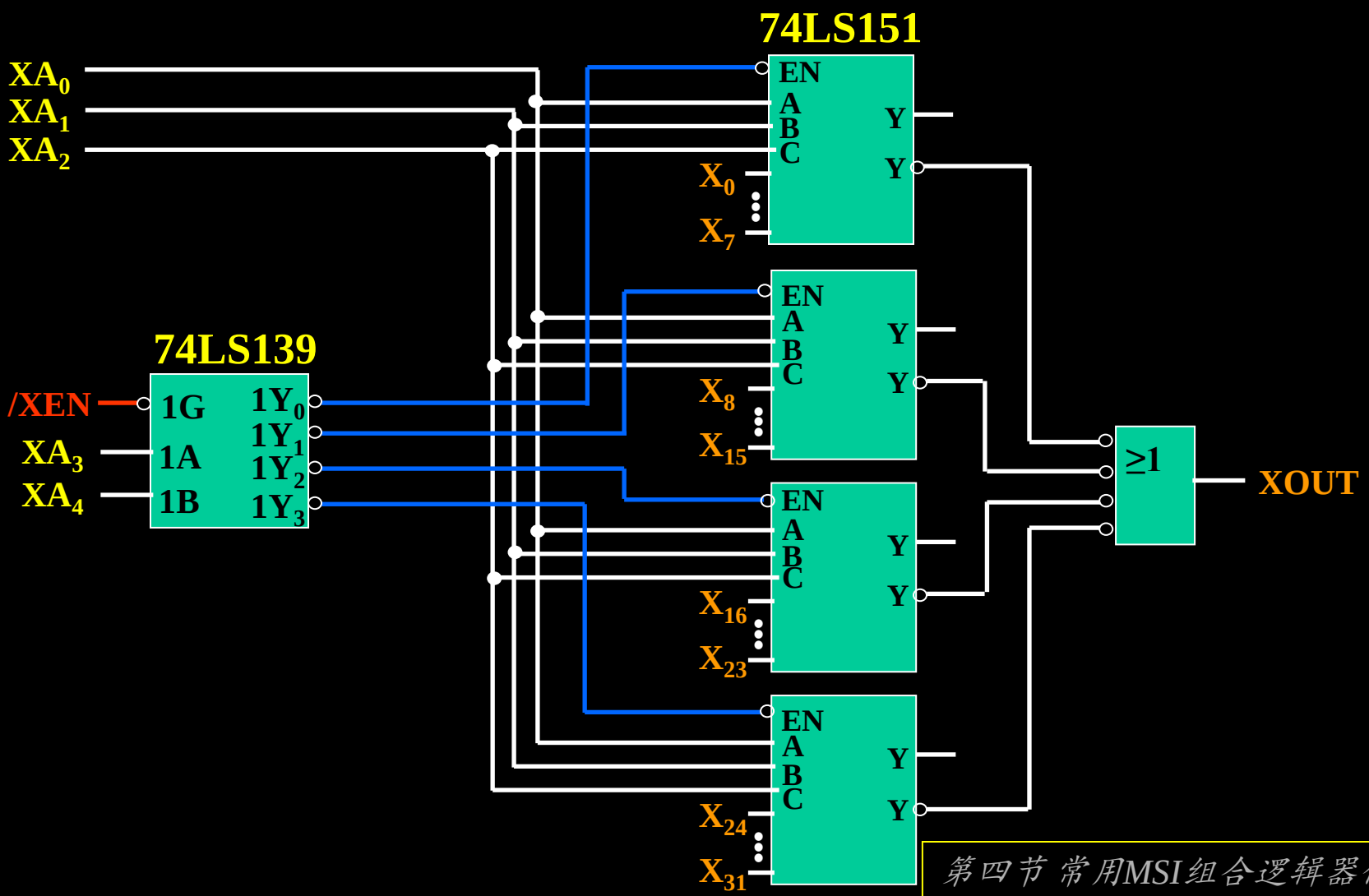




见书P81图2.65

4) 使用三态输出的多路选择器

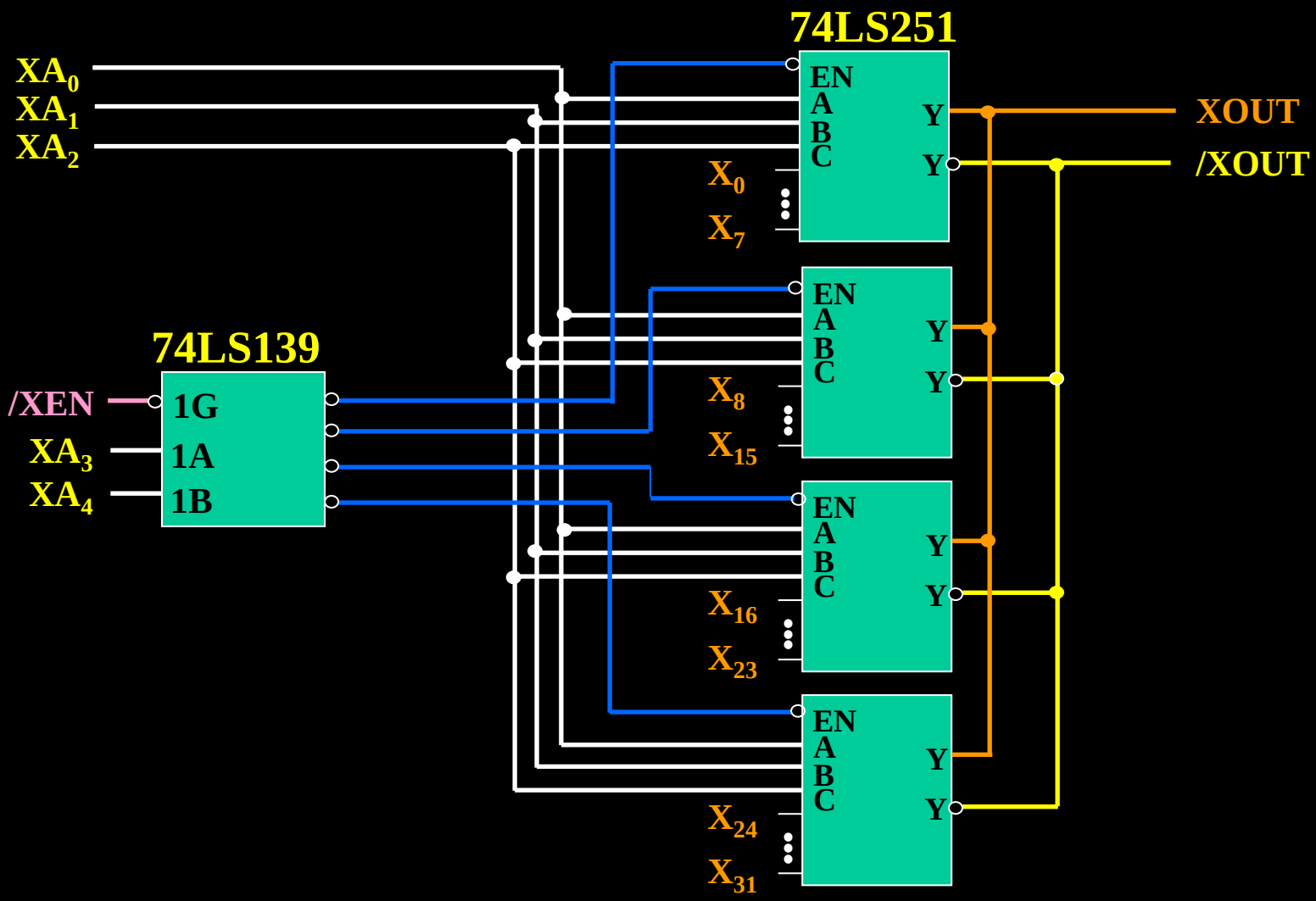
➢ 用74LS151组成的 32输入 1 位多路选择器



例：用74LS251设计一个32输入1位多路选择器。

- 当输出处于高阻态时，该输出线可以与其他输出线直接连接在一起，并且不影响其他输出线的高、低电平。
- 在任意时刻只能有一个74LS251被74LS139使能，此时输出线XOUT和/XOUT上的逻辑值就是该被使能的74LS251的输出值
- 当输入使能/XEN无效时，所有74LS251的输出为高阻态，输出线XOUT和/XOUT上的逻辑值不确定。

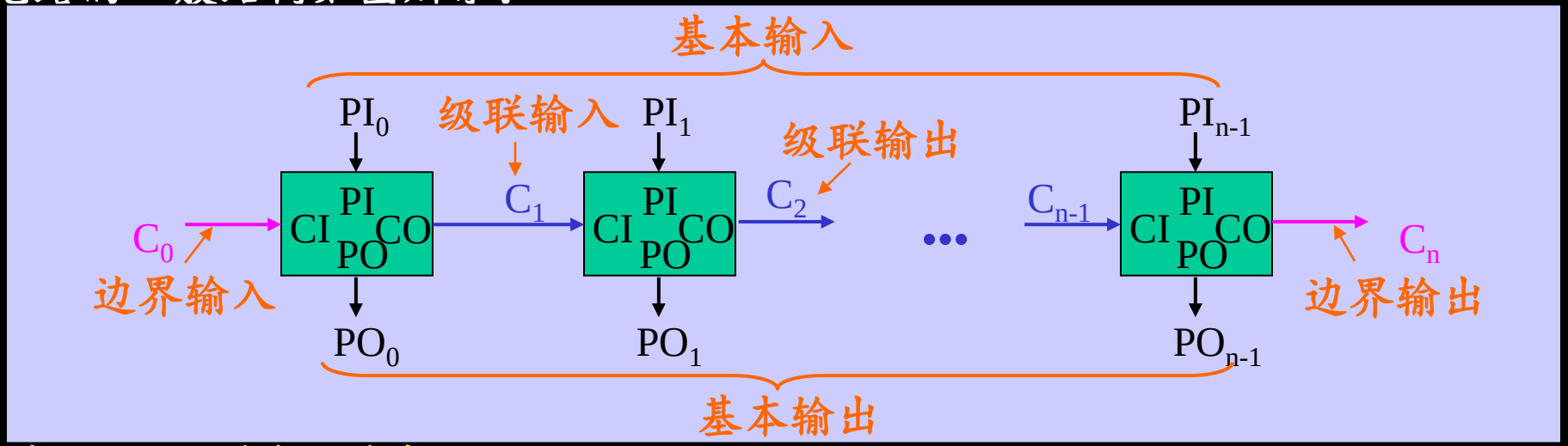
➢ 用74LS251组成的 32输入 1 位多路选择器



2.4.4 比较器和加法器

1）重复电路

串行重复电路是一种串行组合逻辑电路，它包含几个同样的模块，每个模块有四种类型的输入和输出：基本输入、基本输出、级联输入、级联输出。其中：最左边的级联输入称为边界输入，最右边的级联输出称为边界输出。电路的一般结构如图所示：



串行电路的重复操作步骤如下：

(1) C_0 置初值，并置 i 为0；

(2) 用C_i 和 PI_i 运算得到 PO_i和 C_{i+1}；

(3) $i+1 \rightarrow i$；

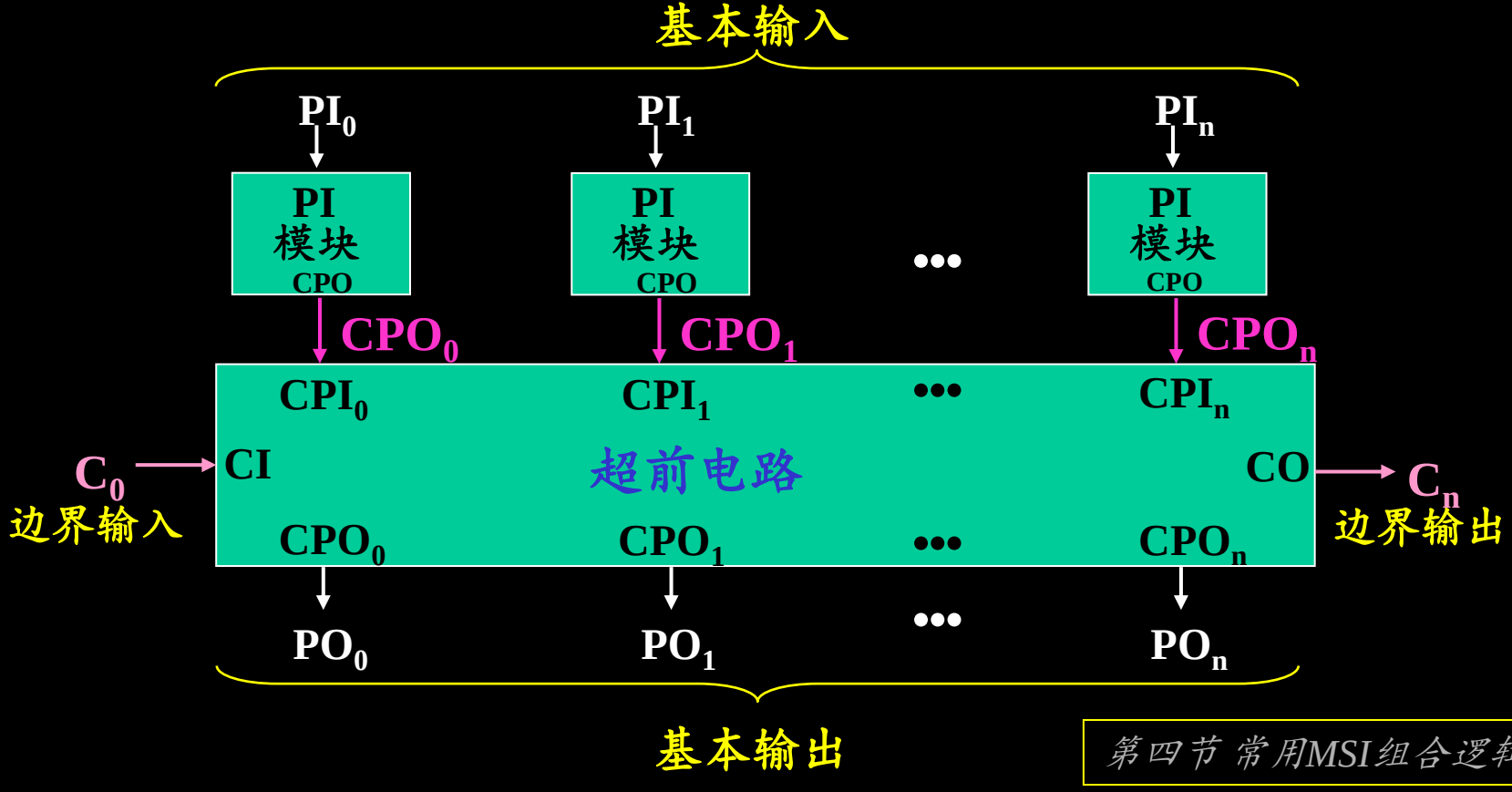
(4) 如果 i 小于 n，返回步骤(2)。

在串行重复电路中，通过提供给各个模块基本输入 $PI_0 \sim PI_n$，利用模块的串行级联完成步骤(2)～(4)的循环。

串行电路非常适用于简单、重复运算的逻辑问题，用串行重复电路可以组成串行比较器和串行加法器等。

2） 超前电路

在串行电路中，延迟时间随着位数 n 的增加而增大。为了提高速度，采用超前电路。即各个模块直接产生供超前电路进行运算的中间信号，由超前电路对这些信号同时进行处理，从而产生输出结果。超前电路框图如图所示。



超前电路与串行电路的区别：

- 超前电路的各个结构相同的输入处理模块没有级联信号，且输出是为了方便超前电路的实现而采用的中间变量。
- 超前电路对所有输入处理模块提供的中间信号同时进行处理，减少了级联传输，提高了速度。
- 超前电路也有边界输入和边界输出，这使得此电路可被串行级联成更大的功能模块。由于级联的级数较少，因此整个电路的速度仍然较高。

1. 比较电路

1）比较单元

比较器是对**两个位数相同**的二进制整数进行数值比较，并判断其大小关系的逻辑器件。

比较大小关系有三种：

大于(>)、等于(=)、小于(<)

相等的比较

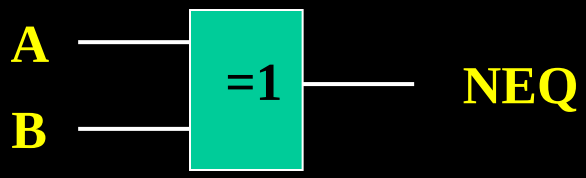
相等比较的过程总是**从高位开始比较**，只有当同位比较结果相等时，才进行低位比较。因此，两个一位数的比较是整个比较器操作的基础。

判断两个一位二进制数是否相等，可用异或门或异或非门实现之，其逻辑表达式为：

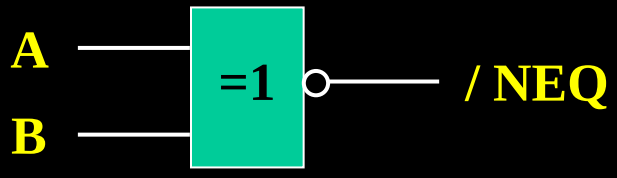
$$NEQ = A \oplus B \qquad /NEQ = \overline{A \oplus B} = EQ$$

对应的逻辑电路如图所示。

¼ 74LS86



¼ 74LS266



同理，异或门及异或非门也可以实现两个多位数的比较。

比如，两个四位二进制数的相等比较。

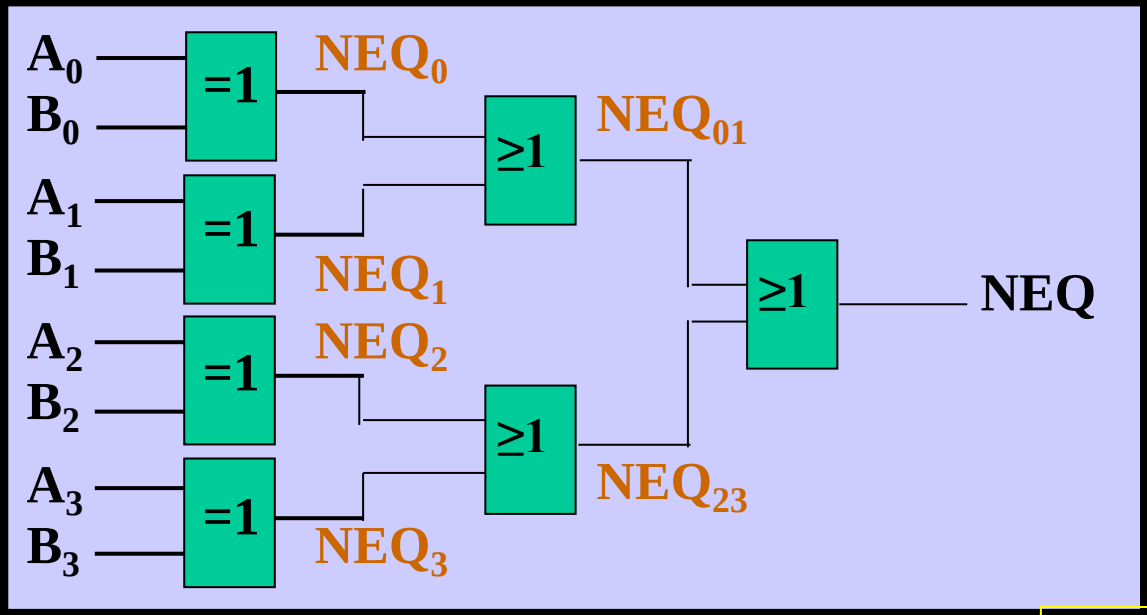
例 四位二进制数的相等比较器。

$$NEQ = (A_0 \oplus B_0) + (A_1 \oplus B_1) + (A_2 \oplus B_2) + (A_3 \oplus B_3)$$

$$/NEQ = \overline{(A_0 \oplus B_0) + (A_1 \oplus B_1) + (A_2 \oplus B_2) + (A_3 \oplus B_3)}$$

$$= \overline{(A_0 \oplus B_0)} \cdot \overline{(A_1 \oplus B_1)} \cdot \overline{(A_2 \oplus B_2)} \cdot \overline{(A_3 \oplus B_3)} = EQ$$

对应的逻辑电路如图所示。



2）串行比较电路

串行比较电路　　*Iterative Comparator Circuits*

两个 n 位数 x 和 y：x_i、y_i (i = 0,1, …,n-1)　相等比较，

其中：x_0、y_0 为最高位。

假设每步比较的结果为EQ_{i+1} (i = 0,1, …,n-1)。

当第 i 次比较结果相等时，$EQ_{i+1} = 1$，

则运算步骤为：

(1) EQ_0 置初值 1，并使 i = 0；

(2) 如果EQ_i 为 1，并且 x_i 和 y_i 相等，则 $EQ_{i+1} = 1$；

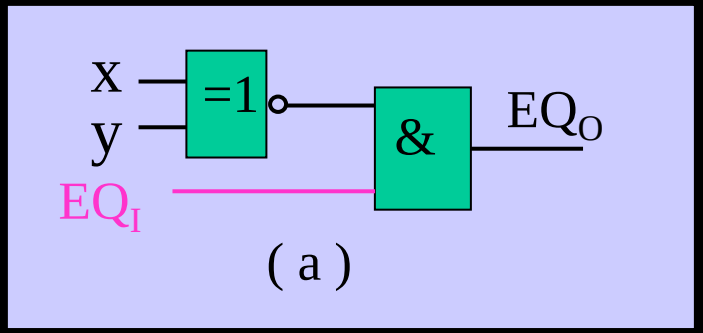
否则，置$EQ_{i+1} = 0$；

(3) $i + 1 \rightarrow i$；

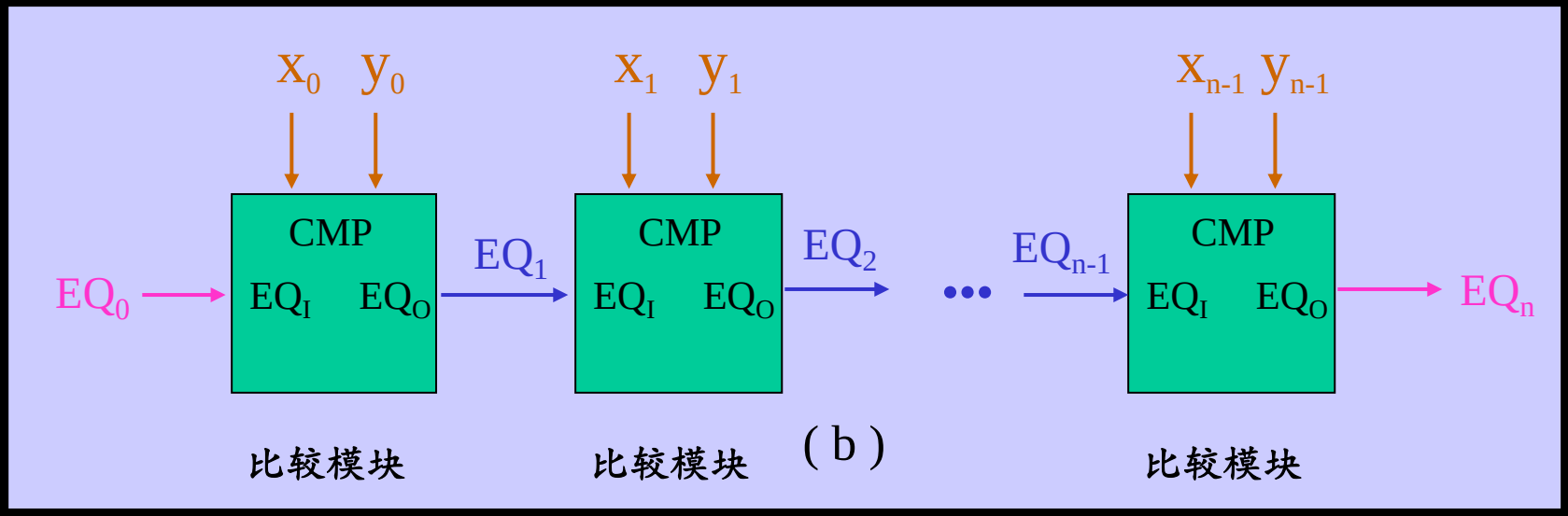
(4) 如果 $i < n$，重复步骤(2)。

判断二个 n 位数是否相等的串行比较电路，如图所示：

- 一位串行比较模块参见图(a)



- 完整的 n 位串行比较电路参见图(b)



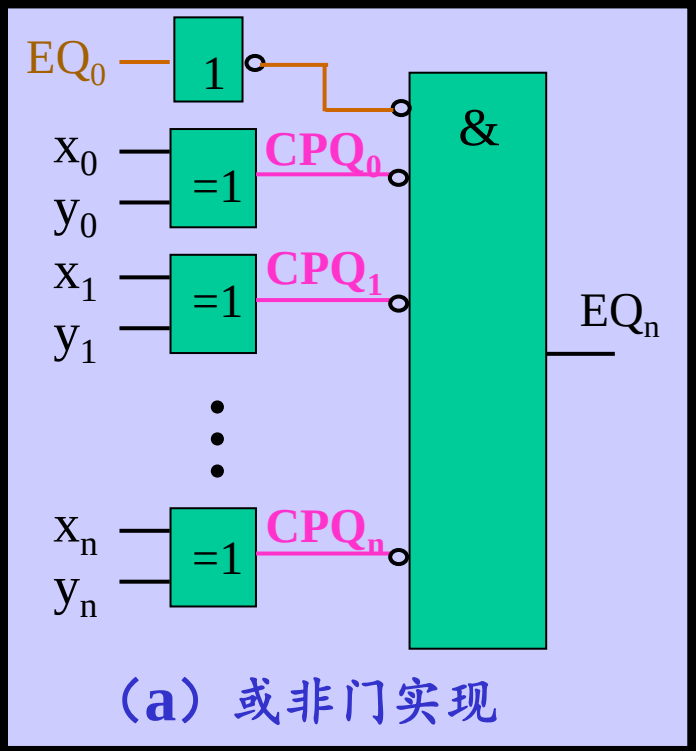
3）超前相等比较器

基本输入：x_i 和 y_i
边界输入：EQ_0
边界输出：EQ_n
中间变量：CPQ_i
$(i = 0,1, \ldots,n)$

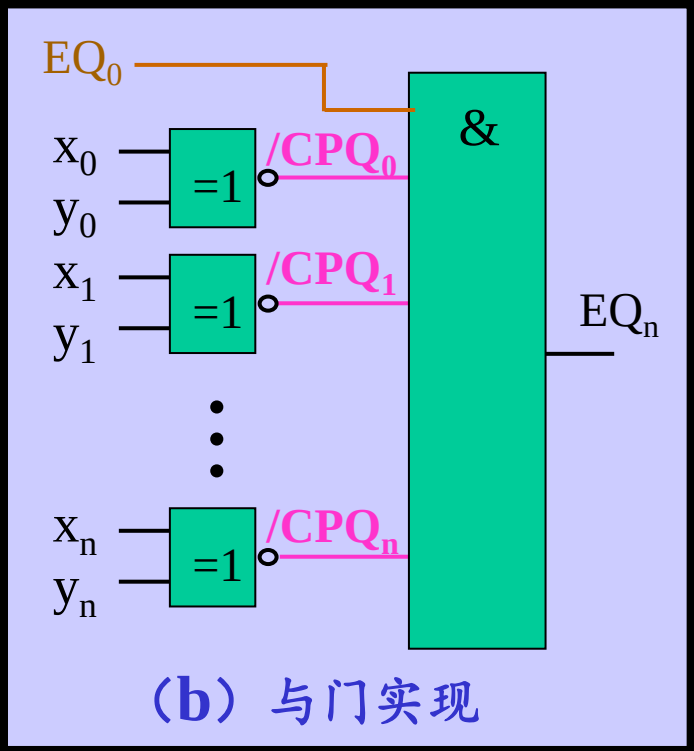
$$CPQ_i = x_i \oplus y_i \qquad i = 0,1, \ldots,n$$

$$EQ_n = EQ_0 \cdot \overline{CPQ_0} \cdot \overline{CPQ_1} \cdot \ldots \cdot \overline{CPQ_n} \quad (b)$$

$$= \overline{\overline{EQ_0} + CPQ_0 + CPQ_1 + \ldots + CPQ_n} \quad (a)$$



（a）或非门实现

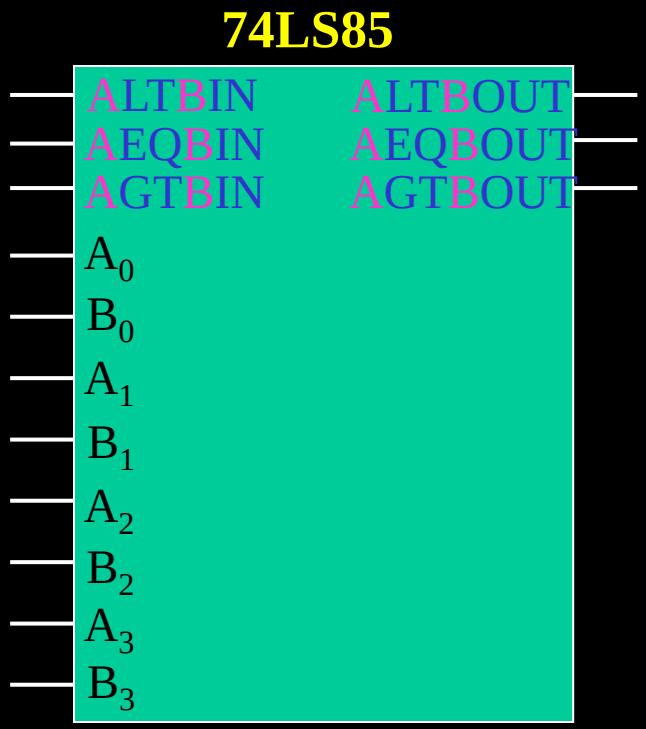


（b）与门实现

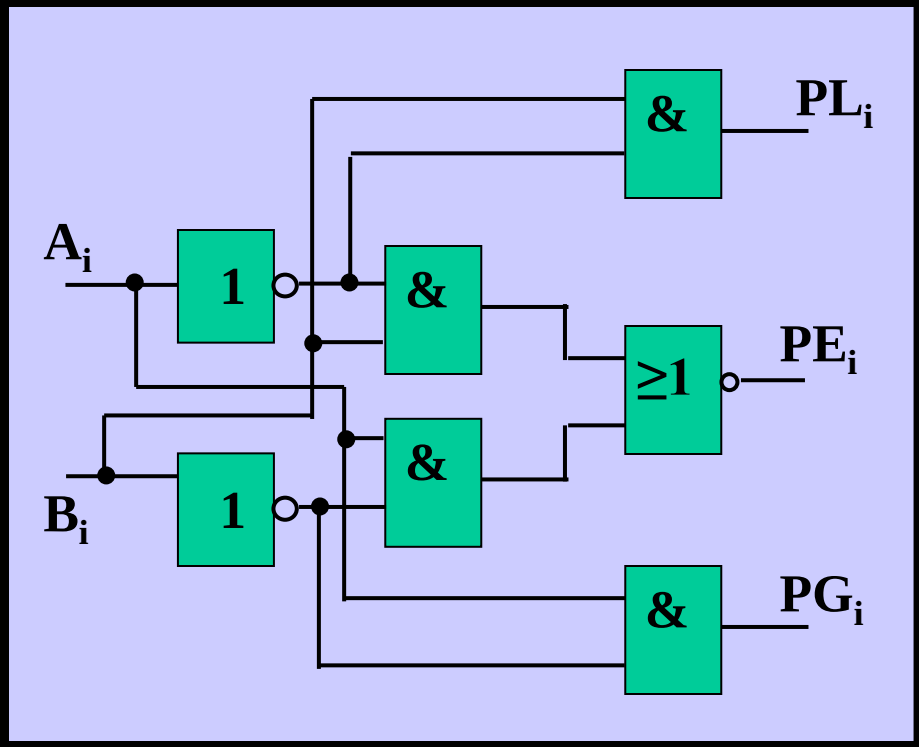
4）MSI比较器

➢ 四位比较器74LS85

① 电路的逻辑符号



② 一个输入处理模块的逻辑框图



比较器内部采用超前电路

③ 逻辑表达式

四位比较器有四个输入处理模块，产生了 12个**中间变量**，它们是：

$PG_0=A_0\overline{B_0}$；　$PE_0=\overline{A_0 \oplus B_0}$；　$PL_0=\overline{A_0}B_0$

$PG_1=A_1\overline{B_1}$；　$PE_1=\overline{A_1 \oplus B_1}$；　$PL_1=\overline{A_1}B_1$

$PG_2=A_2\overline{B_2}$；　$PE_2=\overline{A_2 \oplus B_2}$；　$PL_2=\overline{A_2}B_2$

$PG_3=A_3\overline{B_3}$；　$PE_3=\overline{A_3 \oplus B_3}$；　$PL_3=\overline{A_3}B_3$

比较次序：A_3 与 B_3，A_2 与 B_2，A_1 与 B_1，A_0 与 B_0，

最后是边界输入条件 **ALTBIN**、**AEQBIN**、**AGTBIN**。

④ 四位比较器的输出逻辑表达式

$$AGTBOUT = (A>B) + (A=B) \cdot AGTBIN$$

$$= (PG_3 + PE_3 \cdot PG_2 + PE_3 \cdot PE_2 \cdot PG_1 + PE_3 \cdot PE_2 \cdot PE_1 \cdot PG_0) + (PE_3 \cdot PE_2 \cdot PE_1 \cdot PE_0) \cdot AGTBIN$$

$$AEQBOUT = (A=B) \cdot AEQBIN$$

$$= (PE_3 \cdot PE_2 \cdot PE_1 \cdot PE_0) \cdot AEQBIN$$

$$ALTBOUT = (A<B) + (A=B) \cdot ALTBIN$$

$$= (PL_3 + PE_3 \cdot PL_2 + PE_3 \cdot PE_2 \cdot PL_1 + PE_3 \cdot PE_2 \cdot PE_1 \cdot PL_0) + (PE_3 \cdot PE_2 \cdot PE_1 \cdot PE_0) \cdot ALTBIN$$

⑤ 逻辑电路图

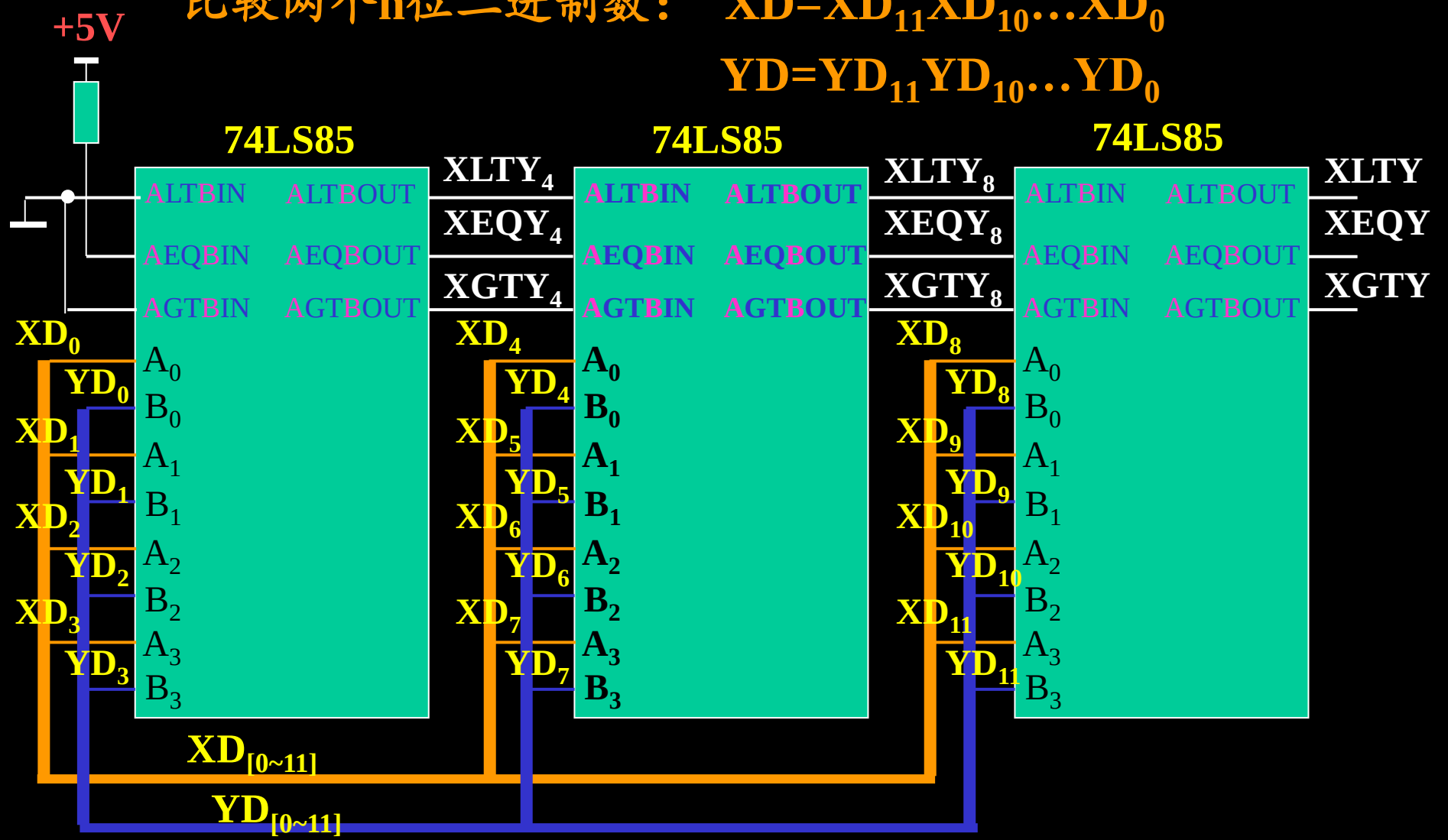
74LS85的逻辑符号，参见书P84图2.71

⑥ 74LS85比较器的级联

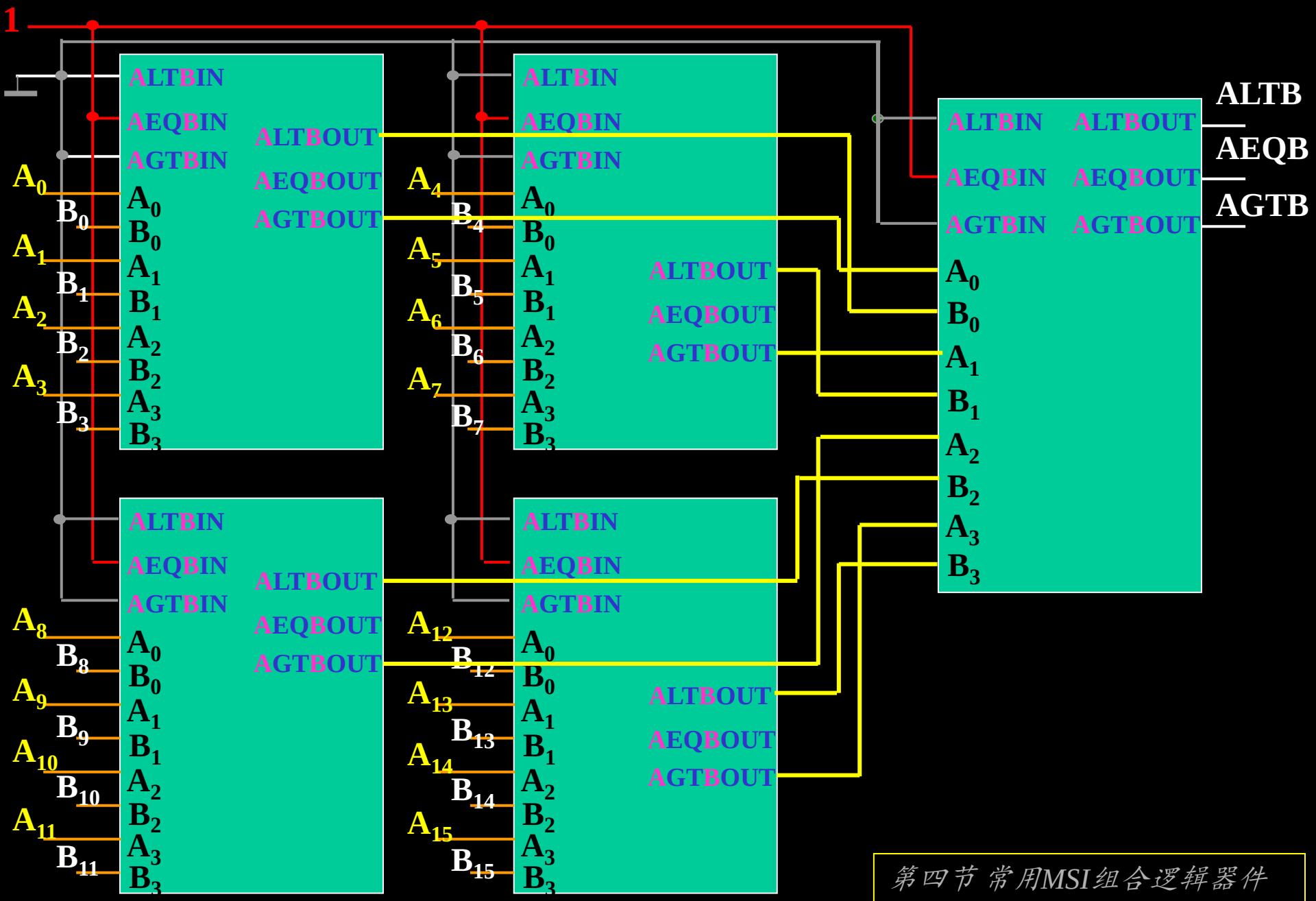
例 用三个74LS85级联构成 12 位比较器

比较两个n位二进制数： $XD=XD_{11}XD_{10}\cdots XD_0$

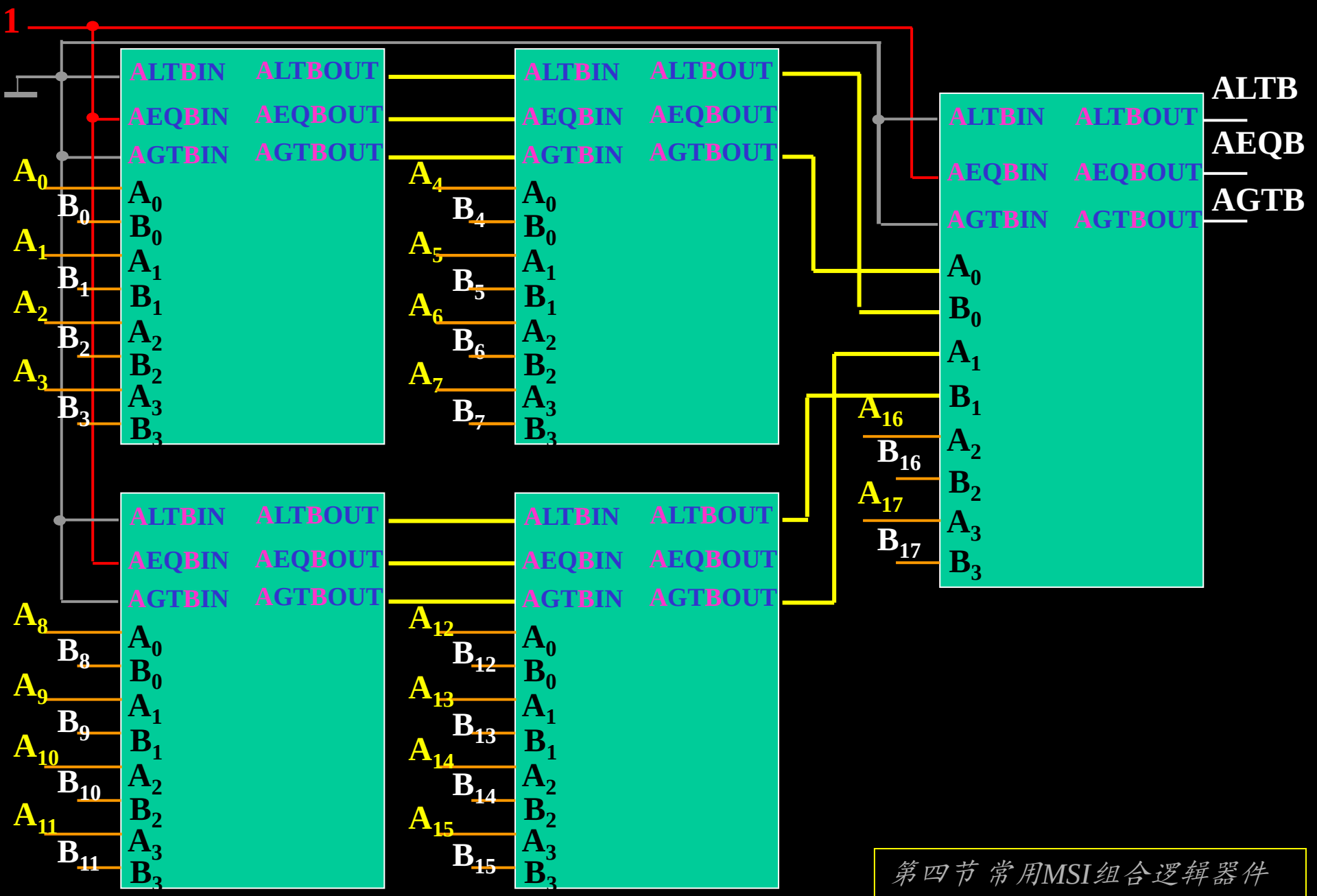
$YD=YD_{11}YD_{10}\cdots YD_0$



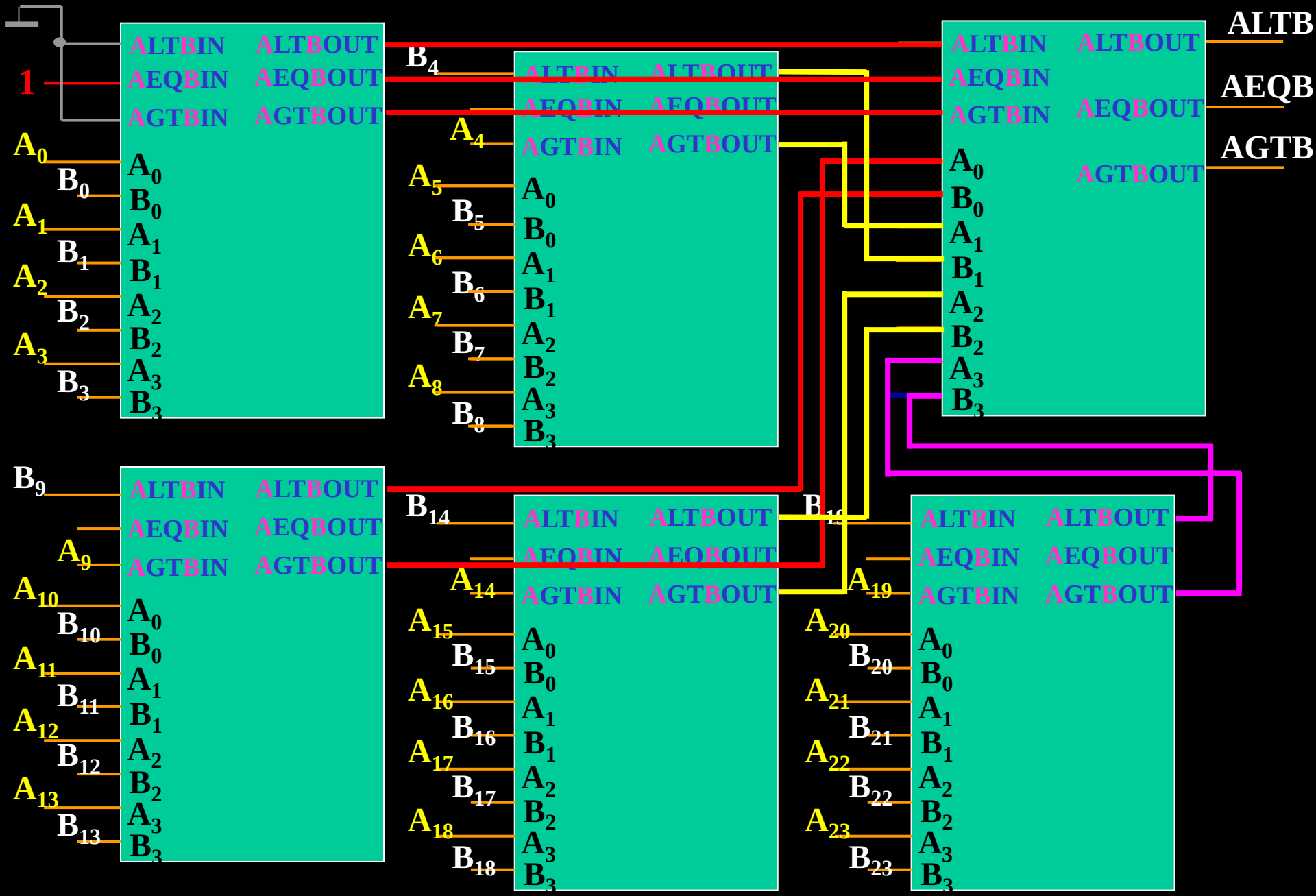
例　用五个74LS85构成16位两级串联树型级联比较器。



阅读例 用五个74LS85构成18位两级串联树型级联比较器。



阅读例 用五个74LS85构成24位两级串联树型级联比较器。



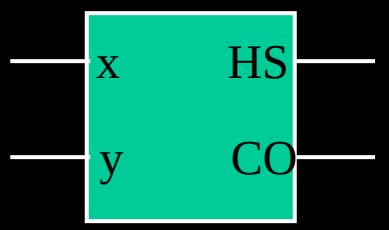
2. 加法器　1）全加器

➢ 半加器的 HS 和 CO 的逻辑表达式为：

$$HS = x \oplus y = \overline{x} \cdot y + x \cdot \overline{y}$$

$$CO = x \cdot y$$

逻辑符号如图所示。

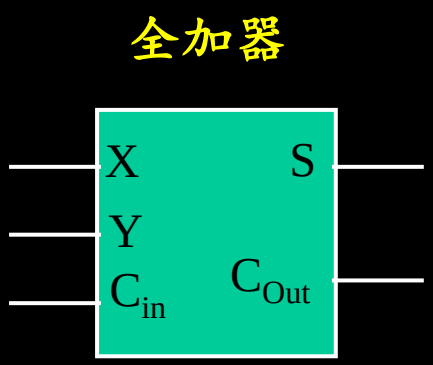


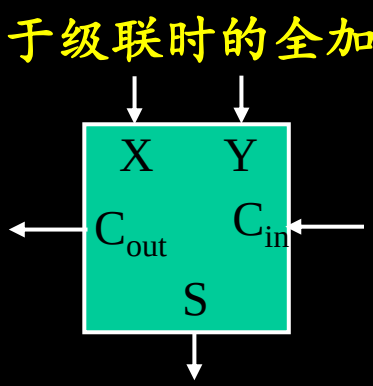
➢ 全加器的 S 和 C_{out} 的逻辑表达式为：

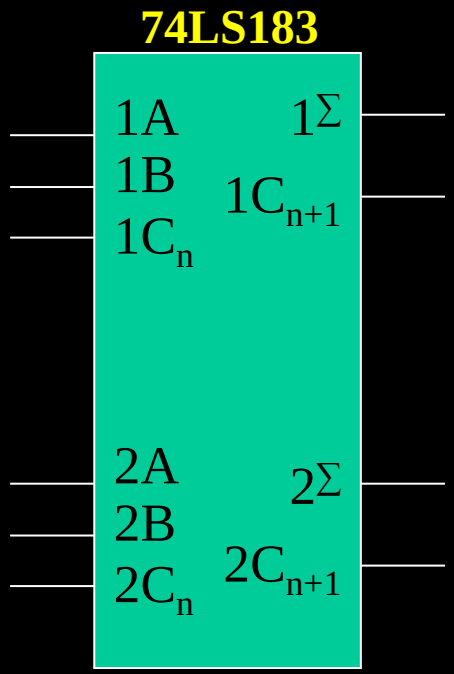
$$S = x \oplus y \oplus C_{in}$$

$$C_{out} = x \cdot y + x \cdot C_{in} + y \cdot C_{in}$$

逻辑符号如图所示。

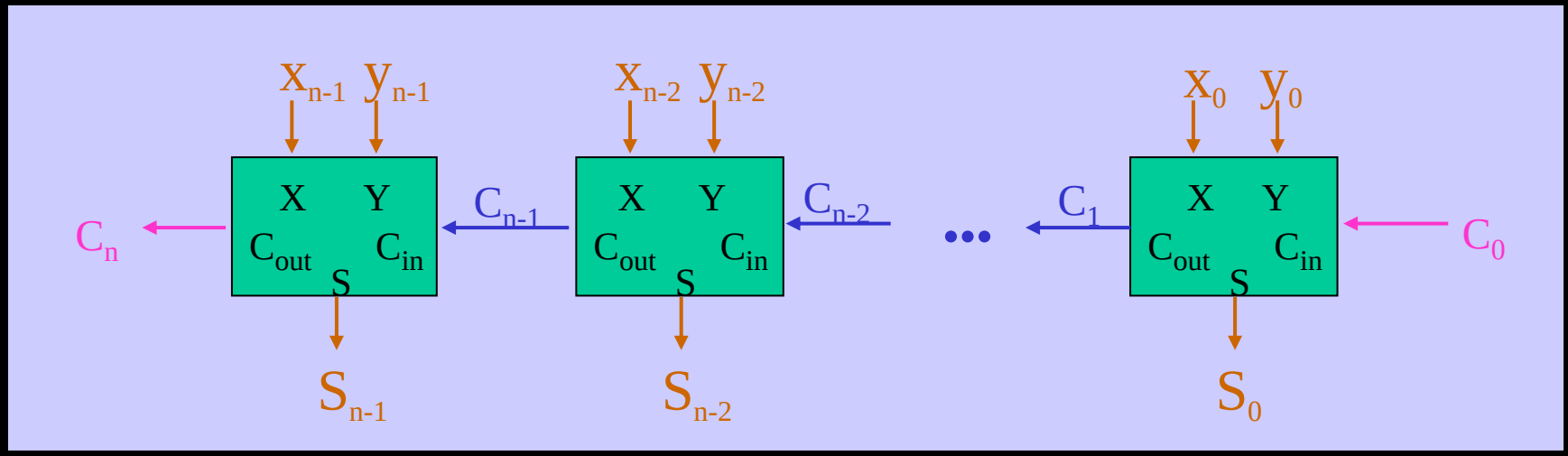






2）并行加法器（行波加法器）

n 个全加器级联，每个全加器处理两个一位二进制数，则可以构成两个 n 位二进制数相加的加法器。由于进位信号是一级一级地由低位向高位逐位产生，故又称为行波加法器。



由于进位信号逐位产生，这种加法器速度很低。行波加法器的最大运算时间为：

$$T_{ADD} = T_{XYCOUT} + (n-2) \cdot T_{CINCOUT} + T_{CINS}$$

其中：T_{XYCOUT} 是最低位全加器中由 x 和 y 产生进位 C_{out} 的延迟时间，$T_{CINCOUT}$ 是中间位全加器中由 C_{in} 产生 C_{out} 的延迟时间，T_{CINS} 是最高位全加器中由 C_{in} 产生 S 的延迟时间。

阅读 *全减器及减法器

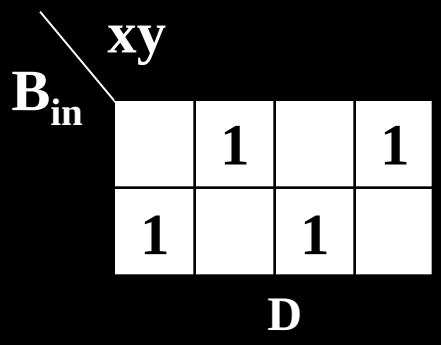
全减器是完成**一位二进制减法运算**的器件。

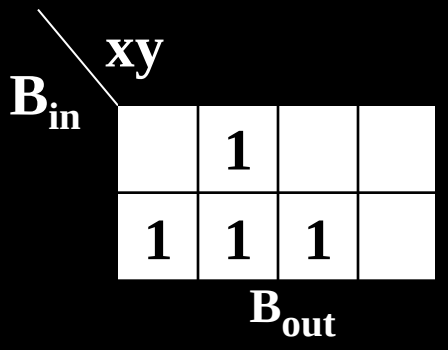
- 三个输入端：被减数 **x**、减数 **y**
  低位向本位的借位 $\mathbf{B_{in}}$
- 两个输出端：本位的差 **D**、本位向高位的借位 $\mathbf{B_{out}}$

① 真值表如下：

x	y	B_{in}	D	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

② 卡诺图如下：





阅读③ 逻辑表达式为：

$$D = x \oplus y \oplus B_{in}$$

$$B_{out} = \overline{x} \cdot y + \overline{x} \cdot B_{in} + y \cdot B_{in}$$

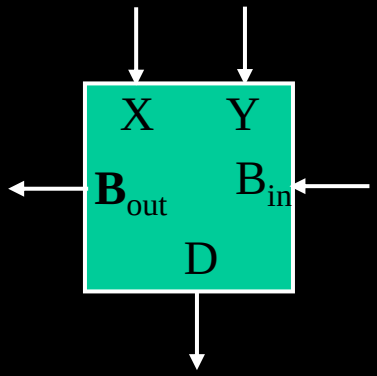
B_{in} \ xy	00	01	11	10
0		1		1
1	1		1	

D

B_{in} \ xy	00	01	11	10
0		1		
1	1	1	1	

B_{out}

④ 逻辑符号



阅读

⑤ 用加法器实现减法器的功能

在实际应用中，是将**全加器推演为全减器**，则全减器的逻辑表达式变换为：

$$B_{out} = \overline{x} \cdot y + \overline{x} \cdot B_{in} + y \cdot B_{in}$$

$$\overline{B_{out}} = (x + \overline{y}) \cdot (x + \overline{B_{in}}) \cdot (\overline{y} + \overline{B_{in}})$$

$$= x \cdot \overline{y} + x \cdot \overline{B_{in}} + \overline{y} \cdot \overline{B_{in}}$$

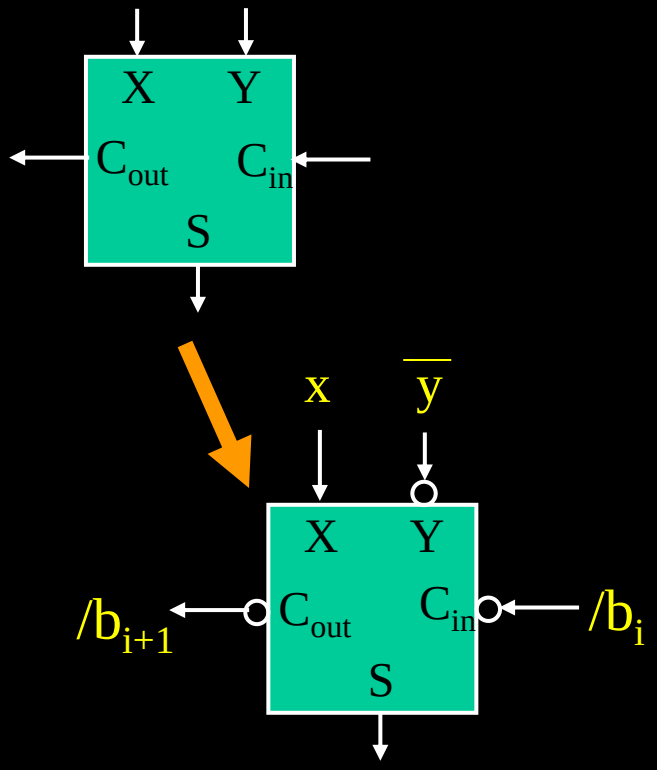
$$D = x \oplus y \oplus B_{in}$$

$$= x \oplus \overline{y} \oplus \overline{B_{in}}$$

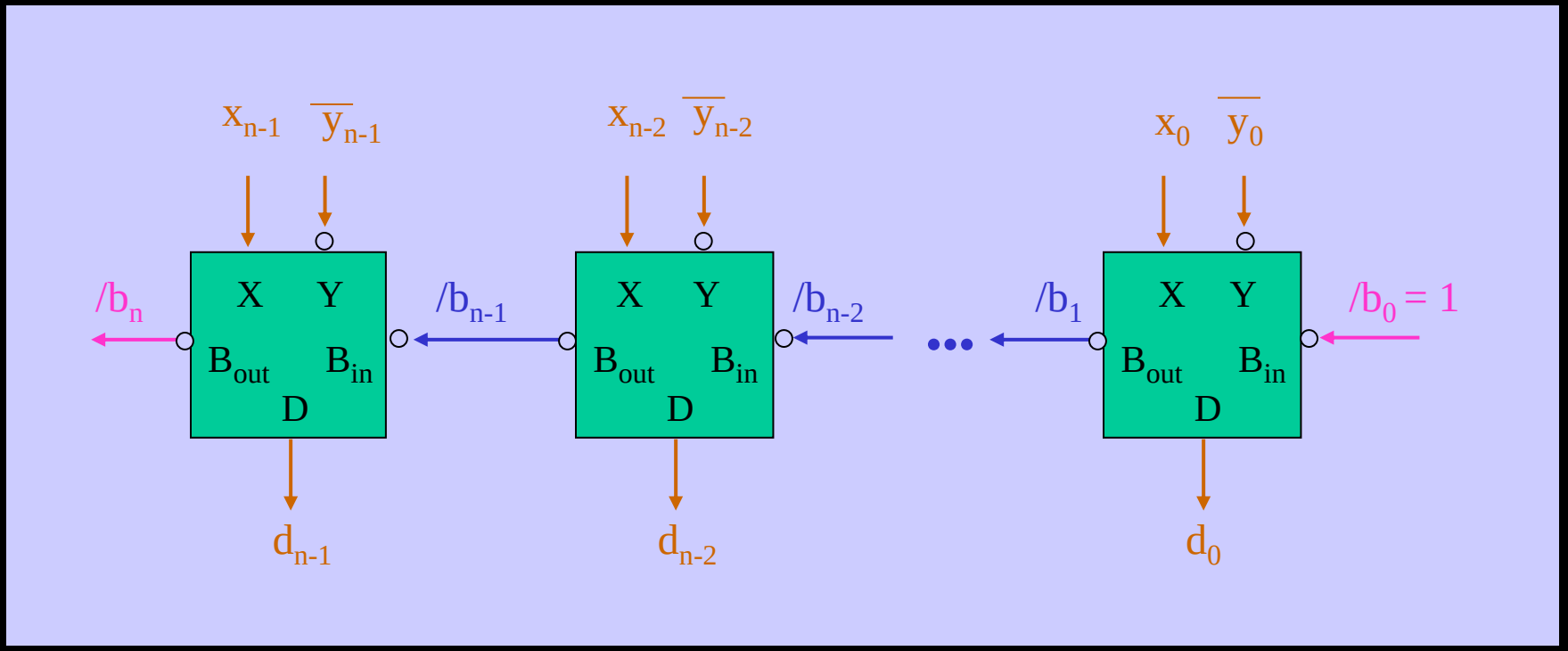
- 将全加器的**进位输入 C_i 和进位输出 C_{i+1}** 分别看成是全减器的两个低有效的**借位输入 $/b_i$ 和借位输出 $/b_{i+1}$**
- 全加器的**和 S_i** 即为全减器的**差 D_i**

则：$D = x \oplus \overline{y} \oplus \overline{b_i}$

$$\overline{b_{i+1}} = x \cdot \overline{y} + x \cdot \overline{b_i} + \overline{y} \cdot \overline{b_i}$$



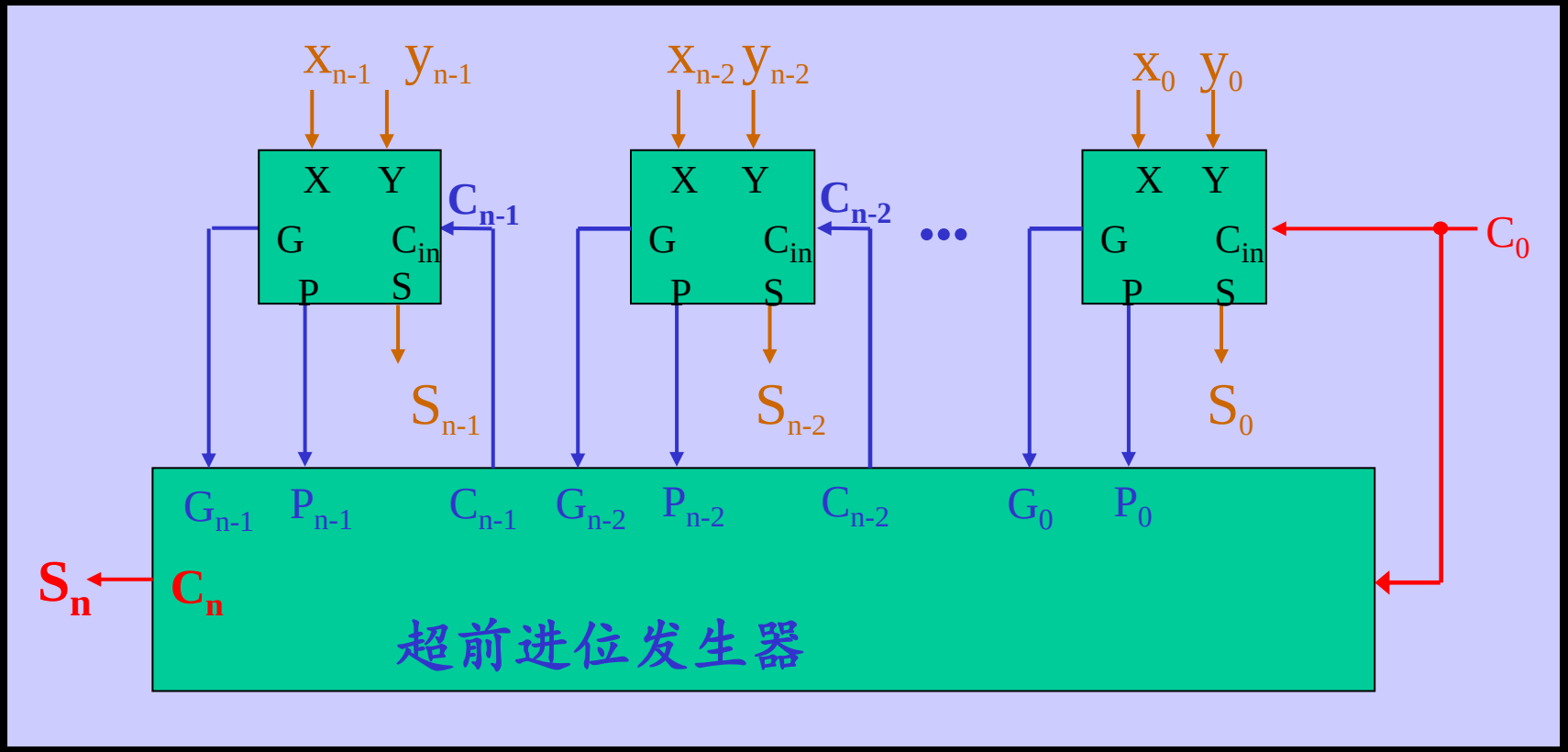
⑥ n 位行波减法器



其中：最低位的借位输入应为无效，即**$/b_0 = 1$**

3）超前进位加法器

结构框图如图所示：



输入处理模块的逻辑表达式为：进位产生项G、进位传递项P

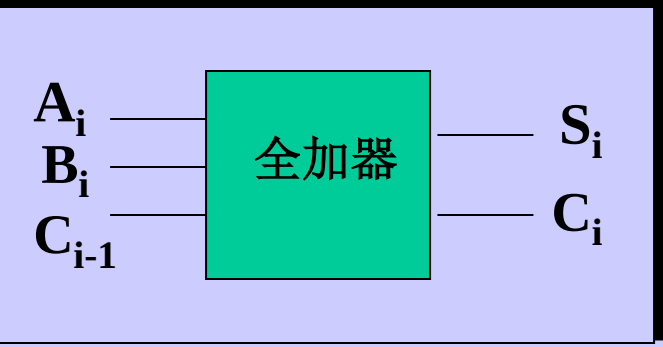
$S = x \oplus y \oplus C_{in} = P \oplus C_{in}$ ， $G = x \cdot y$ ， $P = x \oplus y$

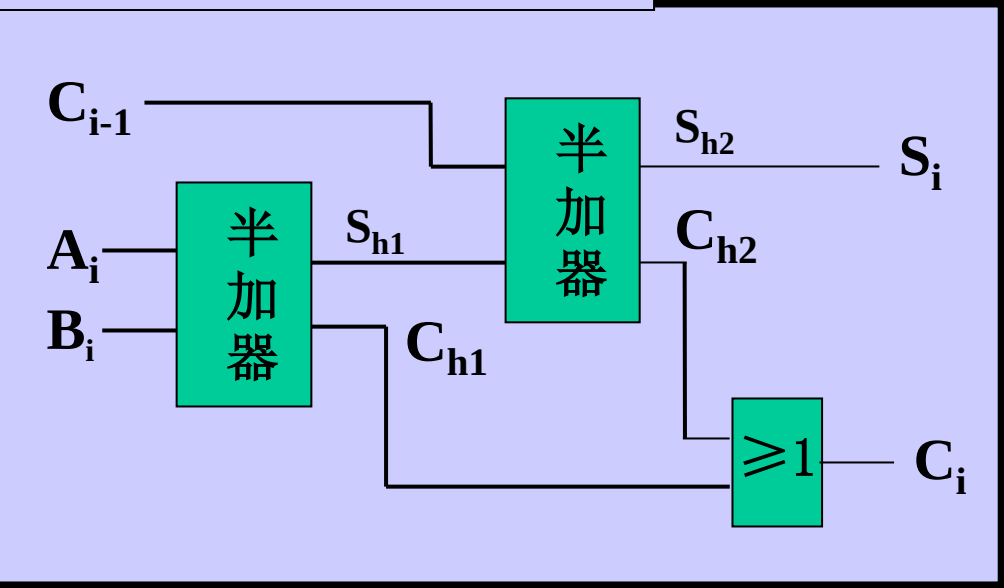
$C_{out} = x \cdot y + (x \oplus y) C_{in} = G + P \cdot C_{in}$

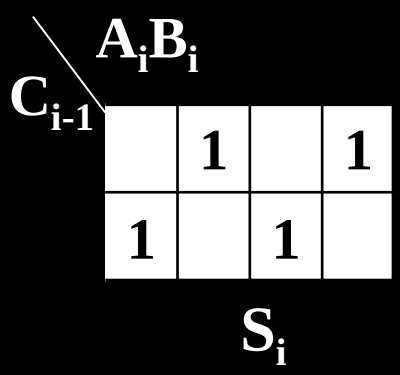
用“半加器”实现全加器：

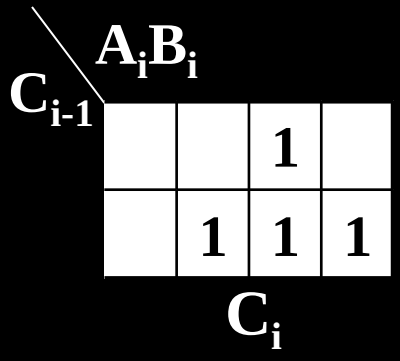
$$S_i = A_i \oplus B_i \oplus C_{i-1}$$
$$= S_{h1} \oplus C_{i-1}$$
$$= S_{h2}$$

$$C_i = A_iB_i + \overline{A_i}\,B_i\,C_{i-1} + A_i\,\overline{B_i}\,C_{i-1}$$
$$= C_{h1} + C_{i-1}\,(\overline{A_i}\,B_i + A_i\,\overline{B_i})$$
$$= C_{h1} + C_{i-1}\,(A_i \oplus B_i)$$
$$= C_{h1} + C_{i-1}\,S_{h1}$$
$$= C_{h1} + C_{h2}$$









超前进位加法器的设计思想是：

由 n 个输入处理模块及超前进位发生器组成。

即 $C_{i+1} = x_i \cdot y_i + (x_i \oplus y_i) \cdot C_i$

$= G_i + P_i \cdot C_i$

其中：输入处理模块产生两个中间变量，分别称为进位产生项G_i、进位传递项P_i

则 $G_i = x_i \cdot y_i$

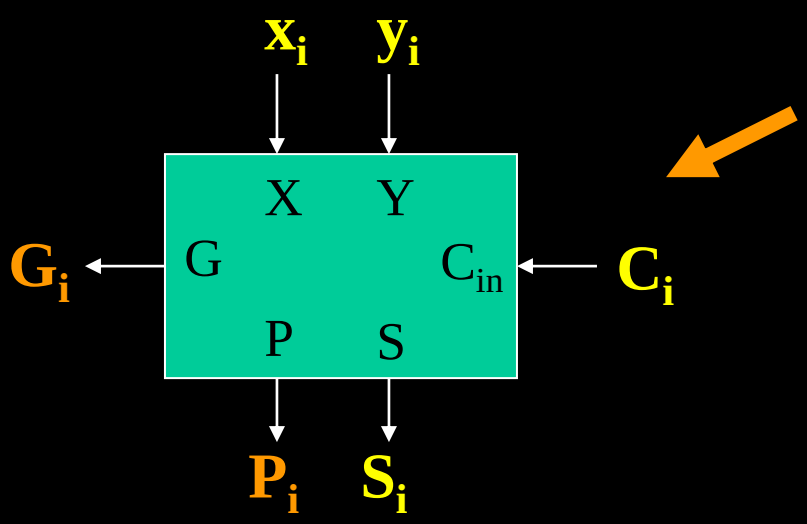
$P_i = x_i \oplus y_i$

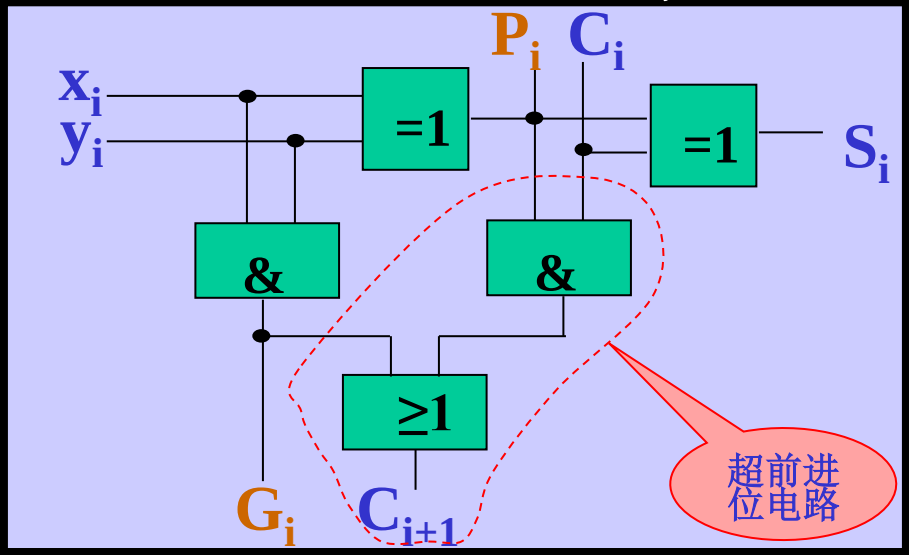
$C_{i+1} = G_i + P_i \cdot C_i = G_i + P_i \cdot (G_{i-1} + P_{i-1} \cdot C_{i-1}) = \cdots\cdots$

$= G_i + P_i \cdot (G_{i-1} + P_{i-1} \cdot (G_{i-2} + P_{i-2} \cdot (\cdots \cdot (G_0 + P_0 \cdot C_0))))$

- 第 i 位的进位输入 C_i 是否为 1，直接取决于 $x_0 \sim x_{i-1}$，$y_0 \sim y_{i-1}$及 C_0；
- 第 i 位是否产生进位输出 C_{i+1}(C_{i+1}= 1)，直接取决于 $x_0 \sim x_i$, $y_0 \sim y_i$及 C_0。

①可以由二个半加器组成的一个全加器构成**输入处理模块**，如下：





②三位二进制加法的进位输出可写成如下，此即**超前进位发生器**：

$$C_1 = G_0 + P_0 \cdot C_0 ,$$

$$C_2 = G_1 + P_1 \cdot C_1 = G_1 + P_1 \cdot (G_0 + P_0 \cdot C_0) = G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_0$$

$$C_3 = G_2 + P_2 \cdot C_2 = G_2 + P_2 \cdot (G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_0)$$

$$= G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot G_0 + P_2 \cdot P_1 \cdot P_0 \cdot C_0$$

逻辑图参见书P87图2.77。

特点：所有进位都是同时产生的，故电路延时时间与位数多少无关。

在位数较多时其运算速度比行波加法器的要快得多。

三位超前进位加法器逻辑图

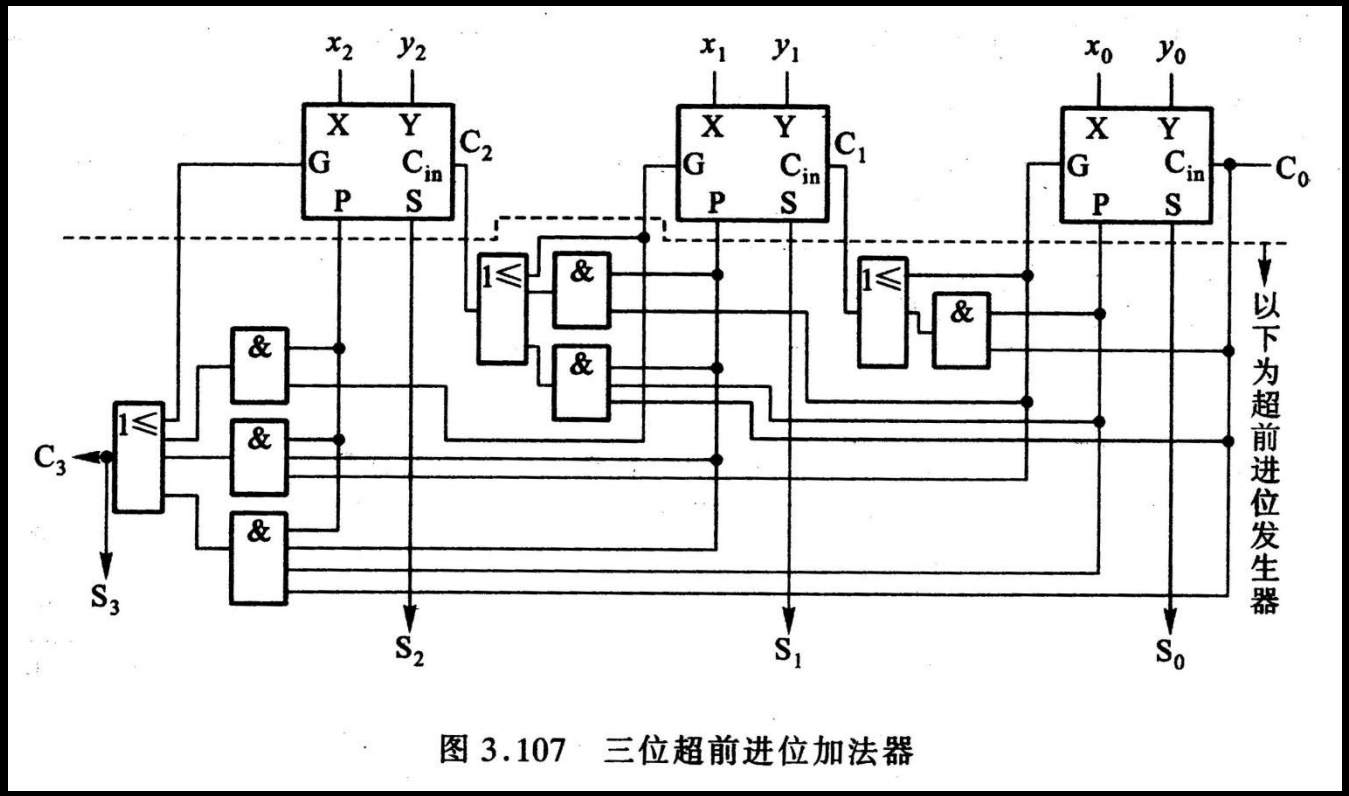


图 3.107 三位超前进位加法器

几级门延迟就可产生所有进位信号C_i?

$$C_1 = G_0 + P_0 \cdot C_0 ,$$

$$C_2 = G_1 + P_1 \cdot C_1 = G_1 + P_1 \cdot (G_0 + P_0 \cdot C_0) = G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_0$$

$$C_3 = G_2 + P_2 \cdot C_2 = G_2 + P_2 \cdot (G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_0)$$

$$= G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot G_0 + P_2 \cdot P_1 \cdot P_0 \cdot C_0$$

4） MSI加法器74LS283

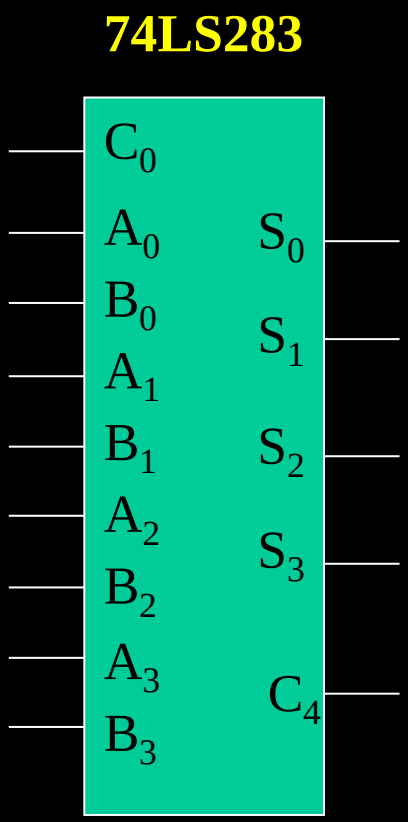
这是一个快速进位四位二进制加法器。

逻辑符号如下所示。

可以用**74LS283级联构成 n 位加法器**，

级联方式为：

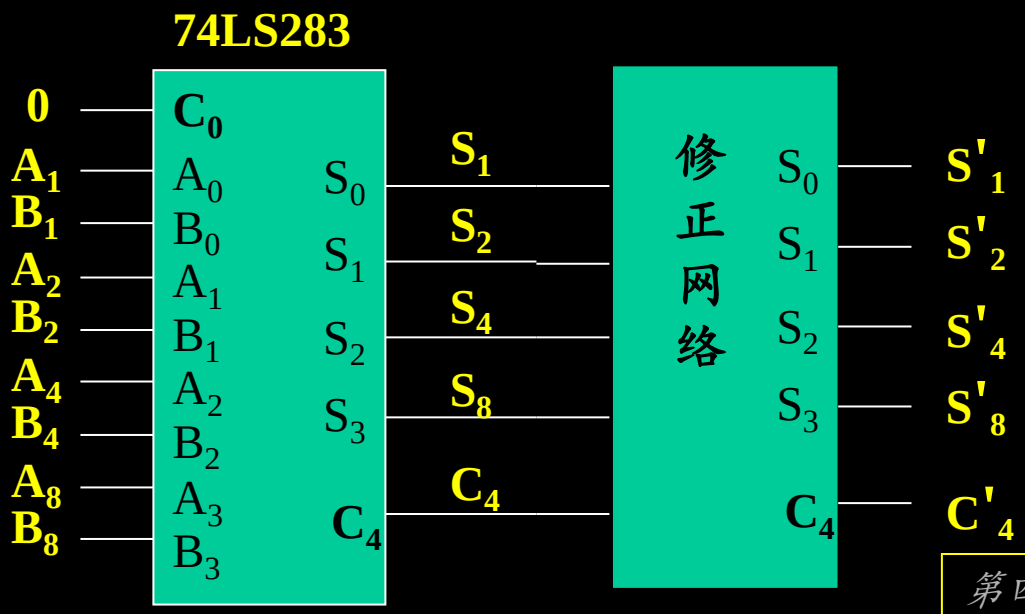
- 片内（4位）超前进位
- 片间为行波进位



➢ MSI加法器的应用

例1 用MSI四位二进制加法器实现两个一位十进制8421BCD码的加法器。

- 一位8421BCD码所能表示的值是 0~9；
- 两个8421BCD码相加所得和的范围是0~18；
- 如果其和的范围在 0~9 之间，则不需要进行校正；
- 如果其和在 10~18 之间，则必须进行校正。



两个一位8421BCD码相加之和的校正表(0+0~9+9，再加进位)

十进制数	未校正BCD码和 $C_4\ S_3\ S_2\ S_1\ S_0$	校正的BCD码和 $C'_4\ S'_3\ S'_2\ S'_1\ S'_0$	十进制数	未校正BCD码和 $C_4\ S_3\ S_2\ S_1\ S_0$	校正的BCD码和 $C'_4\ S'_3\ S'_2\ S'_1\ S'_0$
0	0 0 0 0	0 0 0 0	10	1 0 1 0	1 0 0 0 0
1	0 0 0 1	0 0 0 1	11	1 0 1 1	1 0 0 0 1
2	0 0 1 0	0 0 1 0	12	1 1 0 0	1 0 0 1 0
3	0 0 1 1	0 0 1 1	13	1 1 0 1	1 0 0 1 1
4	0 1 0 0	0 1 0 0	14	1 1 1 0	1 0 1 0 0
5	0 1 0 1	0 1 0 1	15	1 1 1 1	1 0 1 0 1
6	0 1 1 0	0 1 1 0	16	1 0 0 0 0	1 0 1 1 0
7	0 1 1 1	0 1 1 1	17	1 0 0 0 1	1 0 1 1 1
8	1 0 0 0	1 0 0 0	18	1 0 0 1 0	1 1 0 0 0
9	1 0 0 1	1 0 0 1	19	1 0 0 1 1	1 1 0 0 1

校正算法：当和$S_{3\sim0}$的范围在 0~9 之间，则$S'_{3\sim0} = S_{3\sim0} + 0$；

当和$S_{3\sim0}$的范围在 10~18 之间，则$S'_{3\sim0} = S_{3\sim0} + 0110$

两个一位8421BCD码相加之和的校正表(0+0~9+9，再加进位)

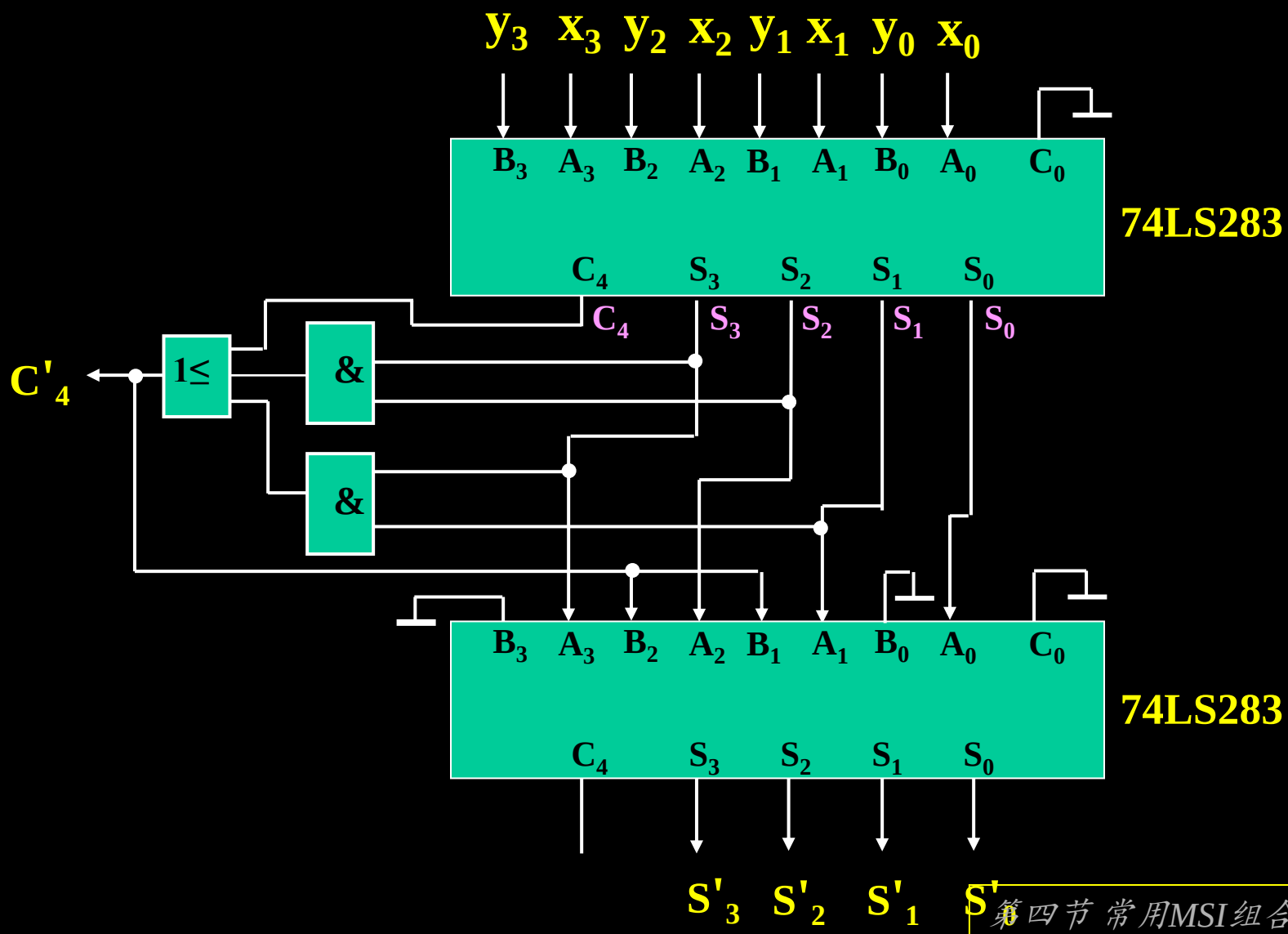
十进制数	未校正BCD码和 C_4	S_3	S_2	S_1	S_0	校正的BCD码和 C'_4	S'_3	S'_2	S'_1	S'_0	十进制数	未校正BCD码和 C_4	S_3	S_2	S_1	S_0	校正的BCD码和 C'_4	S'_3	S'_2	S'_1	S'_0
0	0	0	0	0		0	0	0	0		10	1	0	1	0		1	0	0	0	0
1	0	0	0	1		0	0	0	1		11	1	0	1	1		1	0	0	0	1
2	0	0	1	0		0	0	1	0		12	1	1	0	0		1	0	0	1	0
3	0	0	1	1		0	0	1	1		13	1	1	0	1		1	0	0	1	1
4	0	1	0	0		0	1	0	0		14	1	1	1	0		1	0	1	0	0
5	0	1	0	1		0	1	0	1		15	1	1	1	1		1	0	1	0	1
6	0	1	1	0		0	1	1	0		16	1	0	0	0	0	1	0	1	1	0
7	0	1	1	1		0	1	1	1		17	1	0	0	0	1	1	0	1	1	1
8	1	0	0	0		1	0	0	0		18	1	0	0	1	0	1	1	0	0	0
9	1	0	0	1		1	0	0	1		19	1	0	0	1	1	1	1	0	0	1

$$C'_4 = S_3\overline{S}_2S_1\overline{S}_0 + S_3\overline{S}_2S_1S_0 + S_3S_2\overline{S}_1\overline{S}_0 + S_3S_2\overline{S}_1S_0 + S_3S_2S_1\overline{S}_0 + S_3S_2S_1S_0 + C_4$$

$$= S_3S_1 + S_3S_2 + C_4$$

两个一位8421码加法器的逻辑图

当和需要校正(即$C_4'=1$)时，则需作+0110(+6)校正。

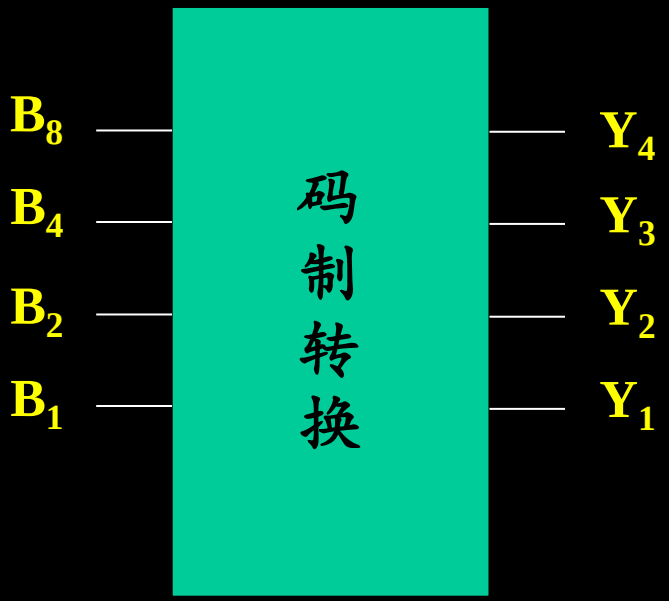


例2 用MSI四位二进制加法器实现码制转换。

1 用MSI四位二进制加法器实现8421BCD码转换成2421码（例2-18）

设：输入的8421BCD码为$B_8B_4B_2B_1$ (0~9)

输出对应的2421码为 $Y_4Y_3Y_2Y_1$



用74LS283实现8421BCD码到2421码的转换。

设：输入为8421码$B_8B_4B_2B_1$，输出为2421码$Y_4Y_3Y_2Y_1$

从真值表可以看出：$Y_4Y_3Y_2Y_1 = B_8B_4B_2B_1 + A_4A_3A_2A_1$

十进制数	$B_8B_4B_2B_1$	$Y_4Y_3Y_2Y_1$	$A_4A_3A_2A_1$
0	0 0 0 0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1	0 0 0 0
2	0 0 1 0	0 0 1 0	0 0 0 0
3	0 0 1 1	0 0 1 1	0 0 0 0
4	0 1 0 0	0 1 0 0	0 0 0 0
5	0 1 0 1	1 0 1 1	0 1 1 0
6	0 1 1 0	1 1 0 0	0 1 1 0
7	0 1 1 1	1 1 0 1	0 1 1 0
8	1 0 0 0	1 1 1 0	0 1 1 0
9	1 0 0 1	1 1 1 1	0 1 1 0

$A_4 = A_1 = 0 \quad A_3 = A_2$

B_2B_1 \ B_8B_4	00	01	11	10
00			d	1
01		1	d	1
11		1	d	d
10		1	d	d

A_2

$A_2 = B_8 + B_4B_2 + B_4B_1$

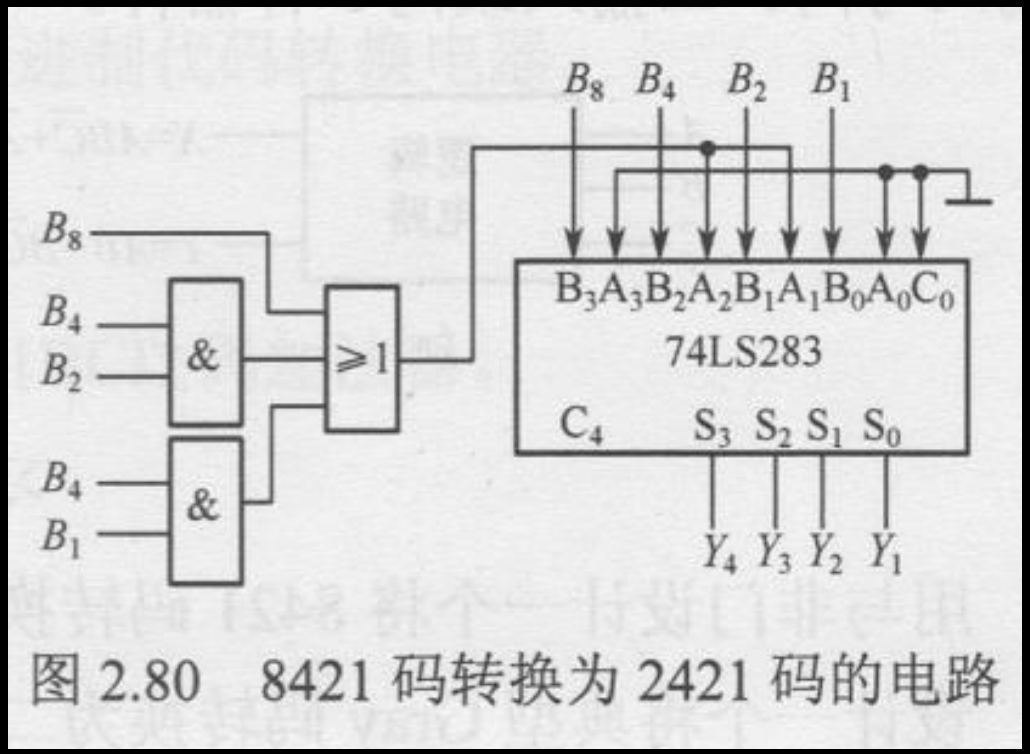


图 2.80 8421 码转换为 2421 码的电路

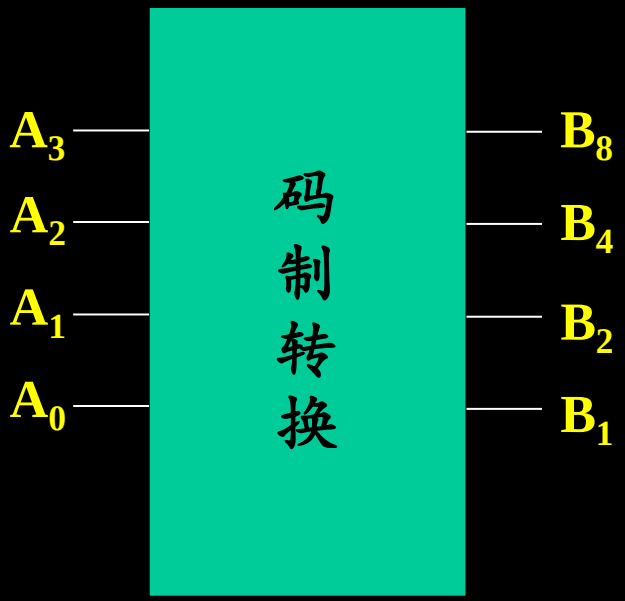
例2 用MSI四位二进制加法器实现码制转换。

练习

2 实现二进制码转换成8421BCD码 。

设：输入的二进制数为 $A_3\ A_2\ A_1 A_0$

输出对应的8421BCD码为$B_8\ B_4\ B_2\ B_1$ (0~9)？



例2 用MSI四位二进制加法器实现码制转换。

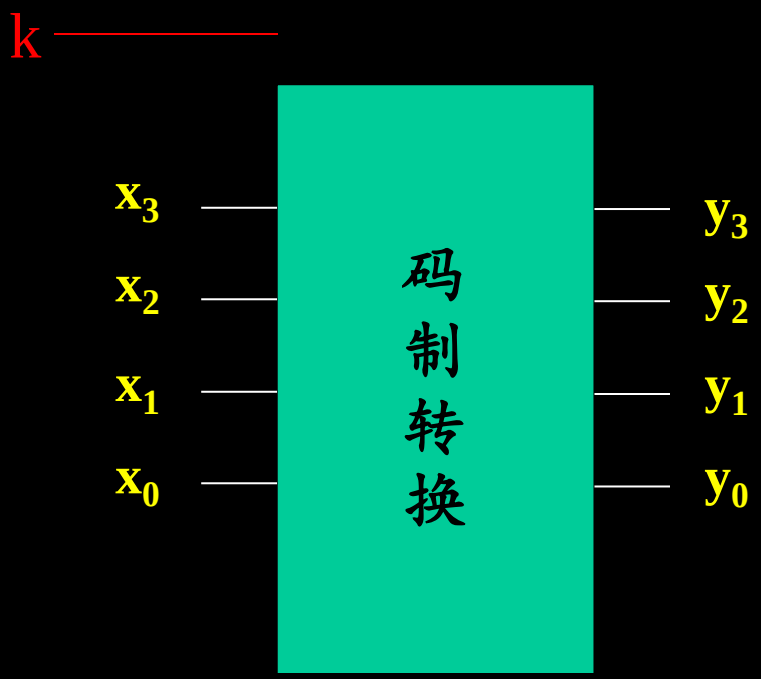
练习3 实现二进制码与8421BCD码的通用转换电路。

设：输入码为 $x_3\ x_2\ x_1 x_0$

输出码为 $y_3\ y_2\ y_1\ y_0$

输入控制端k：1——二进制码→8421码，

0——8421码→二进制码。



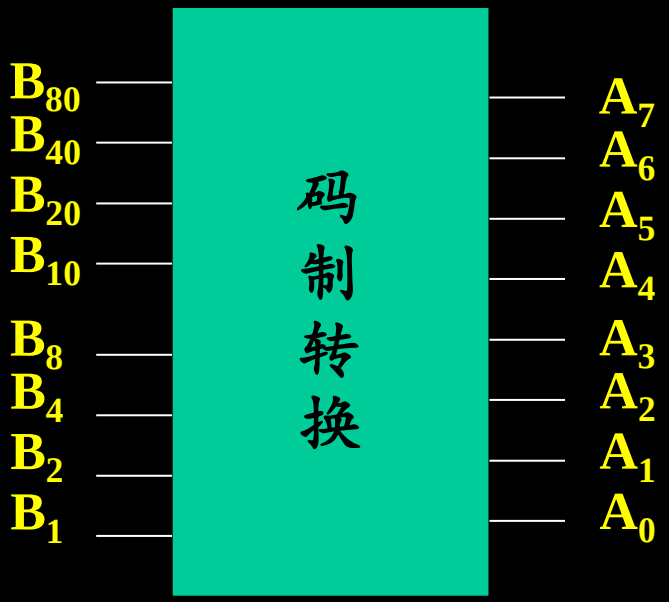
例3 用MSI四位二进制加法器实现两位8421BCD码转换成二进制码

练习

设：两位8421BCD码的高位为$B_{80}B_{40}B_{20}B_{10}$

低位为$B_8\ B_4\ B_2\ B_1$ (0~99)

输出对应的二进制数为$A_7\ A_6\ A_5\ A_4\ A_3\ A_2\ A_1A_0$ (<127)



练习

例3 用MSI四位二进制加法器实现两位8421BCD码转换成二进制码

则：$D = B_{80}B_{40}B_{20}B_{10}\,B_8B_4B_2B_1$

$= (B_{80}B_{40}B_{20}B_{10}) \times 10^1 + (B_8B_4B_2B_1) \times 10^0$

$= (B_{80} \times 8 + B_{40} \times 4 + B_{20} \times 2 + B_{10} \times 1) \times 10^1$

$+ (B_8 \times 8 + B_4 \times 4 + B_2 \times 2 + B_1 \times 1) \times 10^0$

$= B_{80} \times 80 + B_{40} \times 40 + B_{20} \times 20 + B_{10} \times 10$

$+ B_8 \times 8 + B_4 \times 4 + B_2 \times 2 + B_1 \times 1$

$= B_{80} \times (64+16) + B_{40} \times (32+8) + B_{20} \times (16+4)$

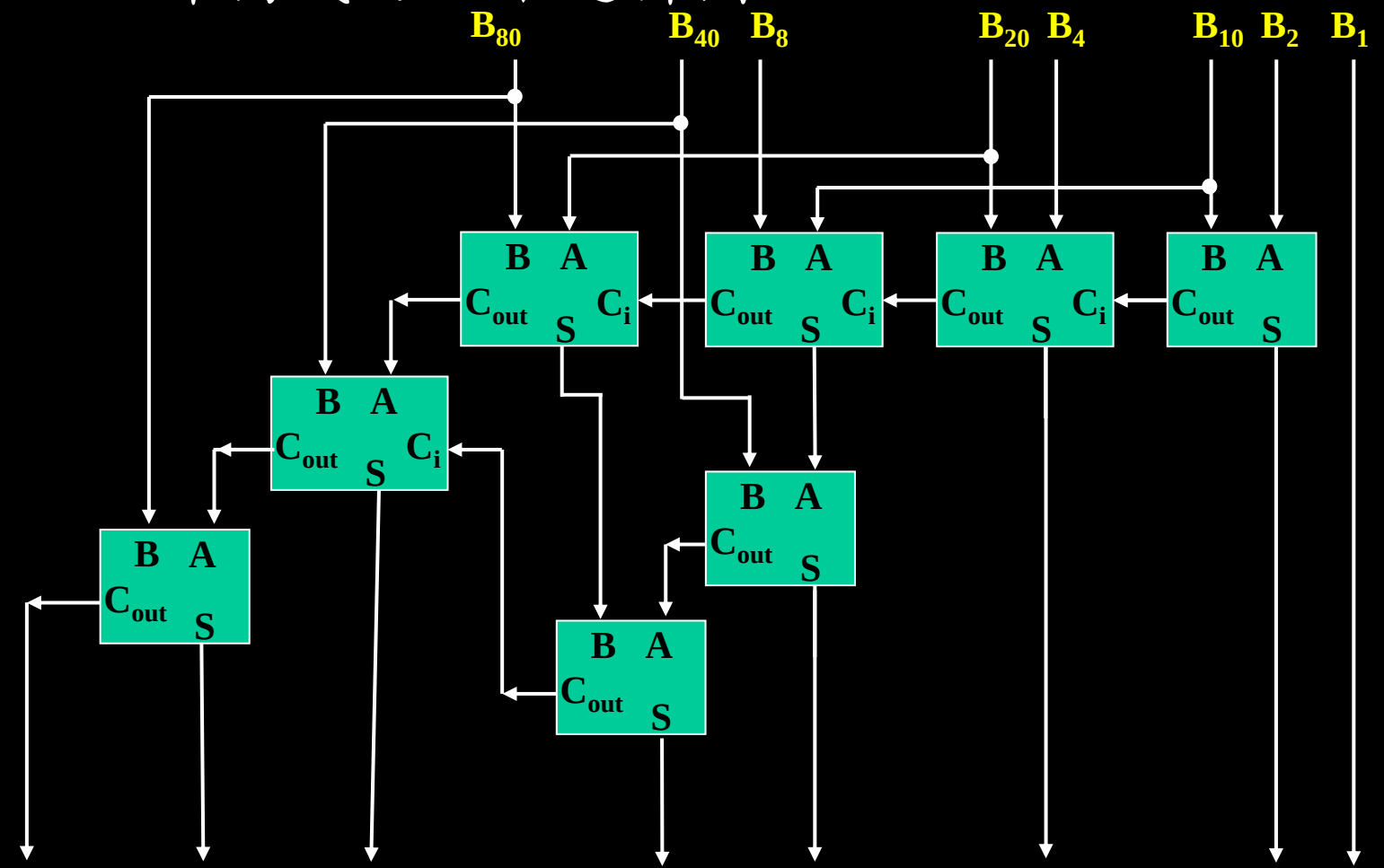
$+ B_{10} \times (8+2) + B_8 \times 8 + B_4 \times 4 + B_2 \times 2 + B_1 \times 1$

$= B_{80} \times 2^6 + B_{40} \times 2^5 + (B_{80} + B_{20}) \times 2^4 + (B_{40} + B_{10}$

$+ B_8) \times 2^3 + (B_{20} + B_4) \times 2^2 + (B_{10} + B_2) \times 2^1 + B_1 \times 2^0$

由分立元件构成的电路逻辑图

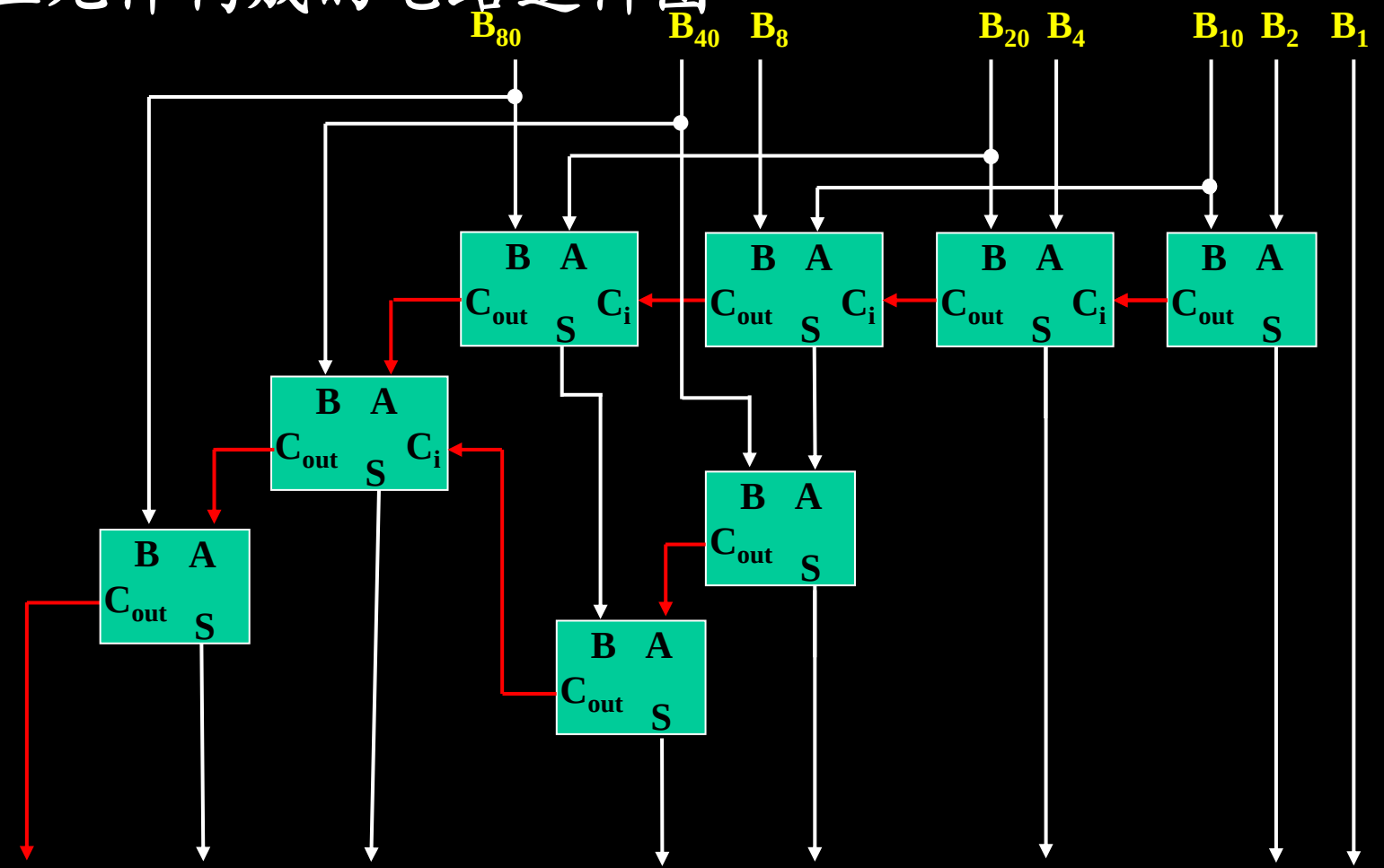
练习



$$= B_{80} \times 2^6 + B_{40} \times 2^5 + (B_{80} + B_{20}) \times 2^4 + (B_{40} + B_{10} + B_8) \times 2^3$$
$$+ (B_{20} + B_4) \times 2^2 + (B_{10} + B_2) \times 2^1 + B_1 \times 2^0$$

由分立元件构成的电路逻辑图

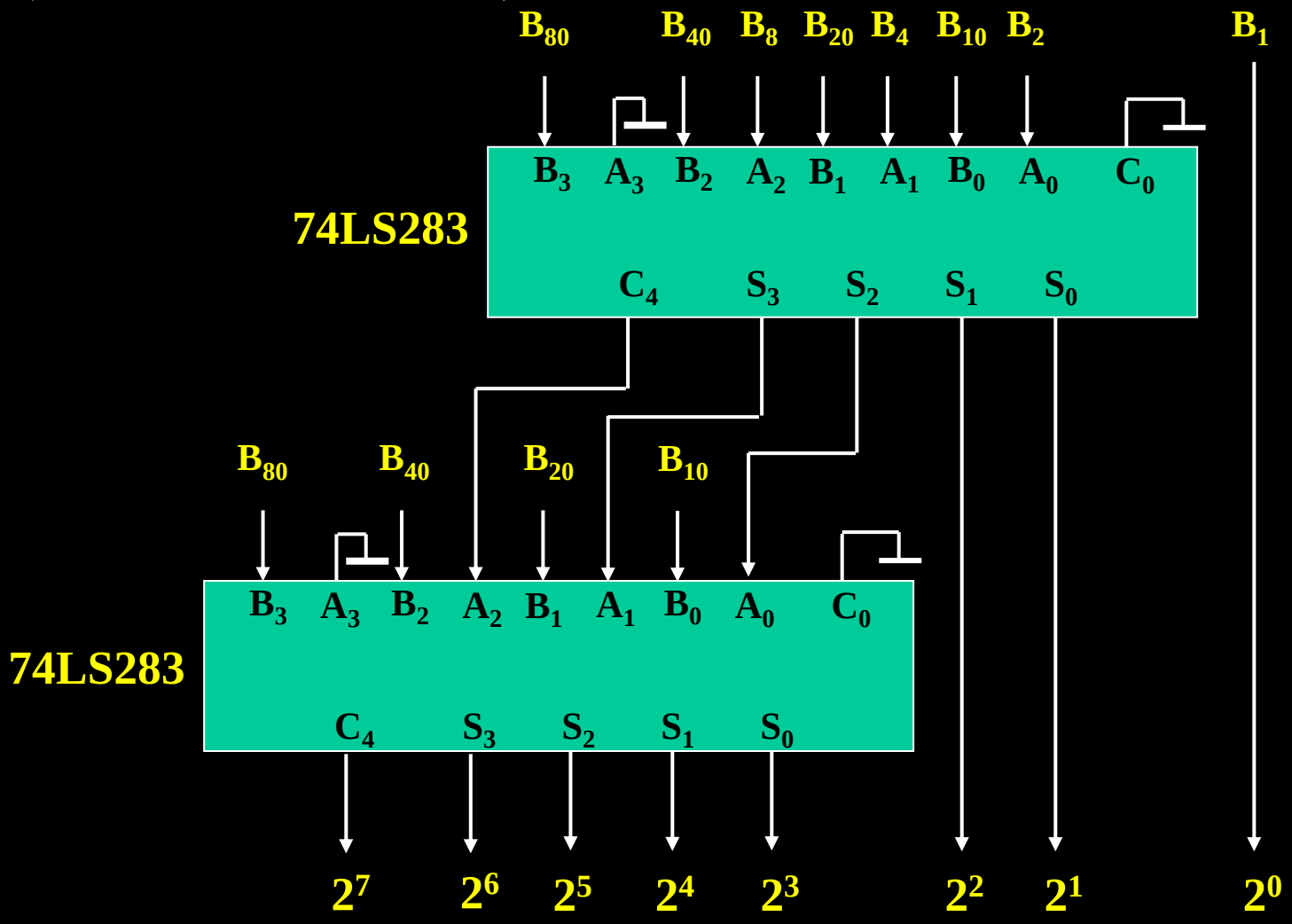
练习



$$= B_{80} \times 2^6 + B_{40} \times 2^5 + (B_{80} + B_{20}) \times 2^4 + (B_{40} + B_{10} + B_8) \times 2^3$$
$$+ (B_{20} + B_4) \times 2^2 + (B_{10} + B_2) \times 2^1 + B_1 \times 2^0$$

由MSI器件构成的电路逻辑图

练习



$$= B_{80} \times 2^6 + B_{40} \times 2^5 + (B_{80} + B_{20}) \times 2^4 + (B_{40} + B_{10} + B_8) \times 2^3$$
$$+ (B_{20} + B_4) \times 2^2 + (B_{10} + B_2) \times 2^1 + B_1 \times 2^0$$

例4 用MSI四位二进制加法器实现码制转换。

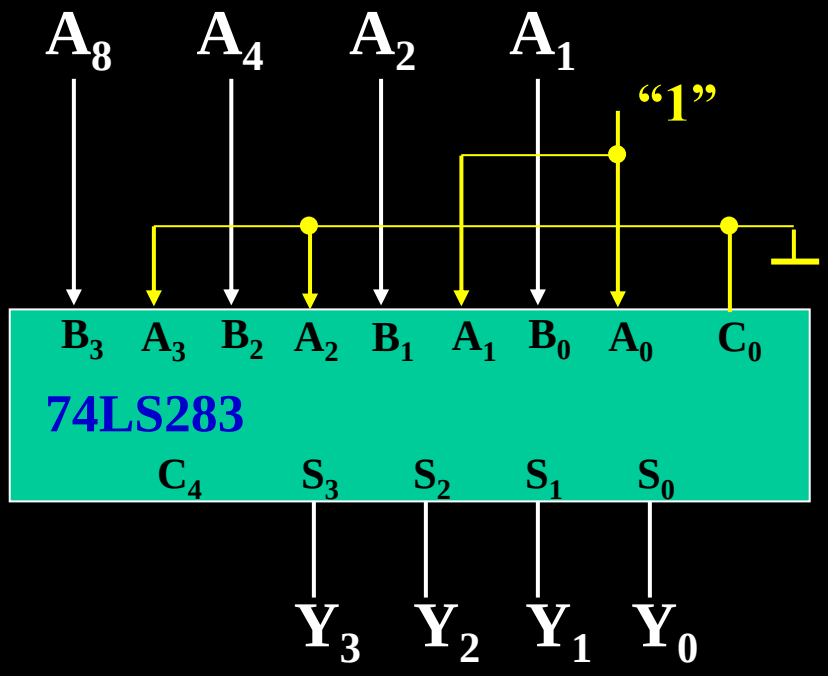
1. 实现8421BCD码到余3码的转换。

设：输入为8421码$A_8A_4A_2A_1$，输出为余3码$Y_3Y_2Y_1Y_0$

从真值表可以看出：$Y_3Y_2Y_1Y_0 = A_8A_4A_2A_1 + 3\ (0011)$

十进制数	$A_8A_4A_2A_1$	$Y_3Y_2Y_1Y_0$
0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 1 0 1
3	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 1 1
5	0 1 0 1	1 0 0 0
6	0 1 1 0	1 0 0 1
7	0 1 1 1	1 0 1 0
8	1 0 0 0	1 0 1 1
9	1 0 0 1	1 1 0 0

电路图如下所示。



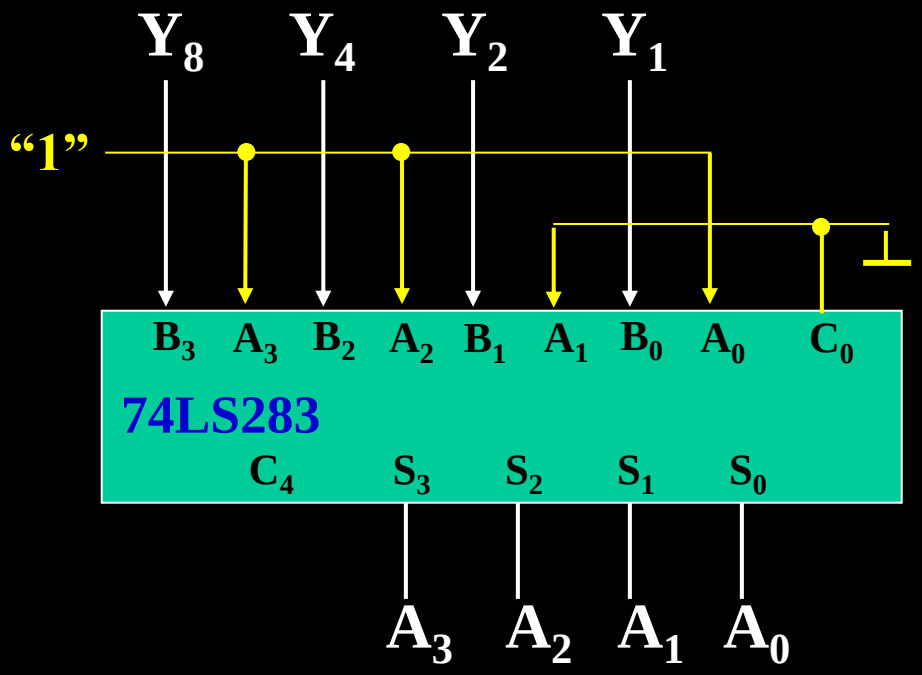
2. 实现余3码到8421BCD码的转换。

设：输入为余3码$Y_3Y_2Y_1Y_0$，输出为8421码$A_8A_4A_2A_1$

则：$A_8A_4A_2A_1 = Y_3Y_2Y_1Y_0 - 3\ (-\ 0011+10000)$

$= Y_3Y_2Y_1Y_0 + \ 1101$

电路图为：

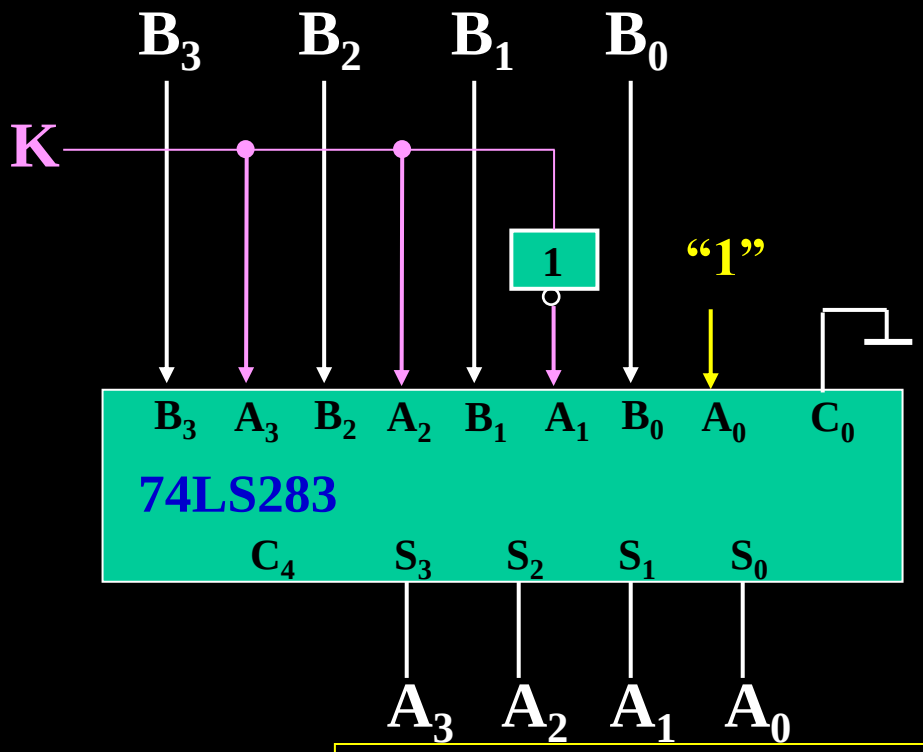


3. 设计8421码和余3码的通用转换器。

设：输入为$B_3B_2B_1B_0$，输出为$A_3A_2A_1A_0$

设置转换开关选择 K

则：$K=\begin{cases}0 & 8421\rightarrow \text{余3码} \quad +0011\\ 1 & \text{余3码}\rightarrow 8421 \quad +1101\end{cases}$



4. 实现2421到余3码的转换。

设：输入为2421码ABCD

输出为余3码$Y_3Y_2Y_1Y_0$

十进制数	A B C D	$Y_3Y_2Y_1Y_0$
0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 1 0 1
3	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 1 1
5	1 0 1 1	1 0 0 0
6	1 1 0 0	1 0 0 1
7	1 1 0 1	1 0 1 0
8	1 1 1 0	1 0 1 1
9	1 1 1 1	1 1 0 0

从真值表可以看出：

- 当十进制数为 0 ~4 时，

相应的余三码比2421码多3。

- 当十进制数为 5 ~ 9时，

相应的余三码比2421码少3。

即：当输入 A = 0 时

$Y_3Y_2Y_1Y_0 = ABCD + 3$

$= ABCD + 0011$

当输入 A = 1时

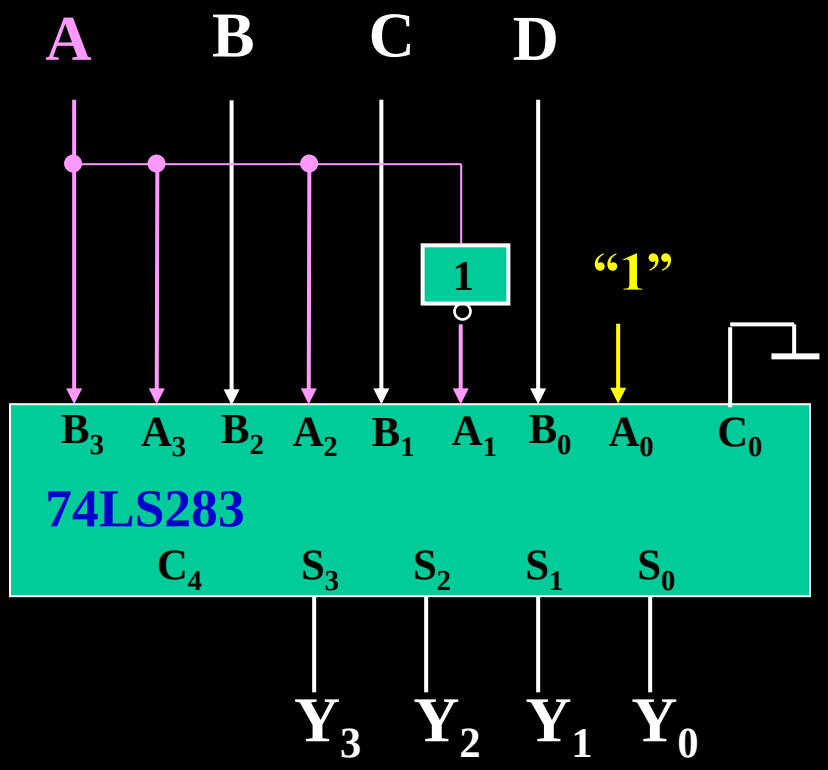
$Y_3Y_2Y_1Y_0 = ABCD - 3$

$= ABCD + 1101$

2421码转换成余3码的转换电路

A = 0： + 0011

A = 1： － 0011（+ 1101）

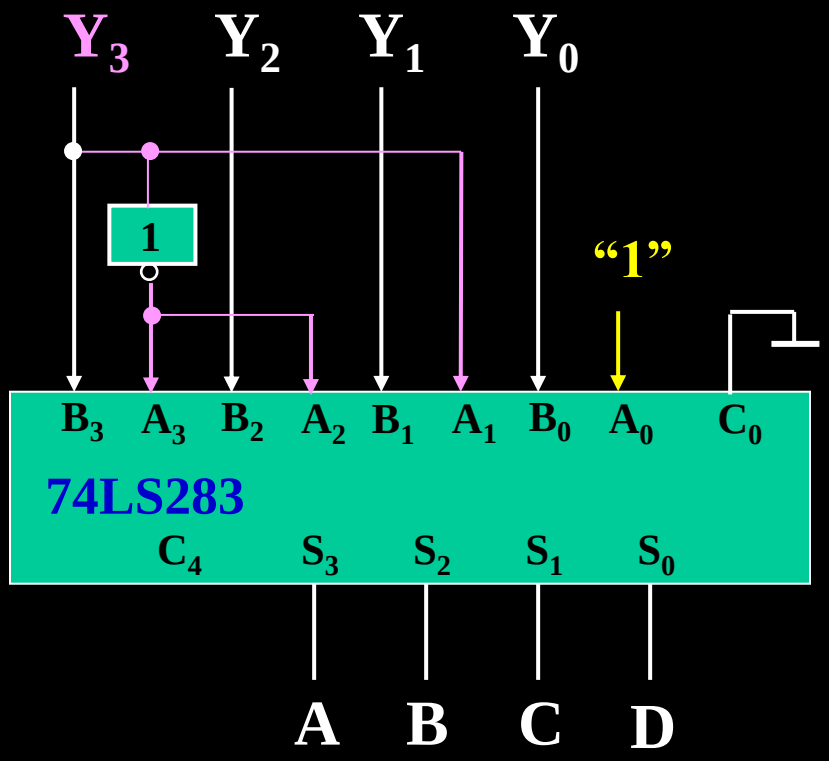


5. 余3码转换成2421码的转换电路

根据前例的分析，可以得出：

$Y_3 = 0$：－0011（+1101）

$Y_3 = 1$：+ 0011



练习

6. 设计2421码和余3码的通用转换器。

设：输入为$B_3B_2B_1B_0$，输出为$A_3A_2A_1A_0$

设置转换开关选择 K

则：$K=\begin{cases}0 & 2421\rightarrow \text{余3码}, \text{异或门不起作用}\\ 1 & \text{余3码}\rightarrow 2421, \text{异或门起非门作用}\end{cases}$