

第四章 组合逻辑电路

主要内容

- ❖ 组合逻辑电路的特点
- ❖ 组合逻辑电路的分析与设计方法
- ❖ 几种常用的组合逻辑电路
 - 编码器、译码器、数据选择器、加法器和数值比较器
- ❖ 竞争-冒险现象及其消除方法

组合逻辑电路定义

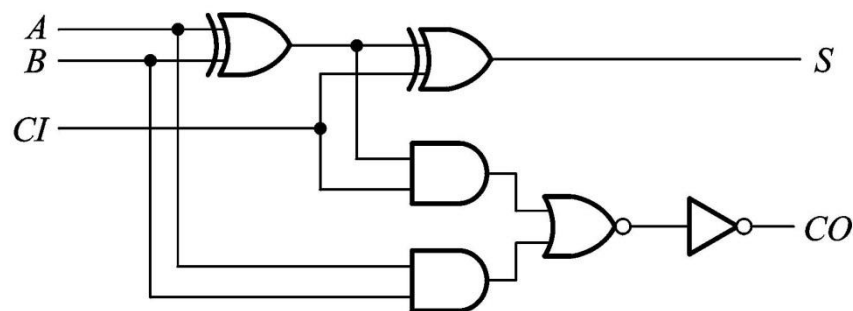
4.1 概述

一、组合逻辑电路的特点

1. 从功能上
2. 从电路结构上

任意时刻的输出仅
取决于该时刻的输入

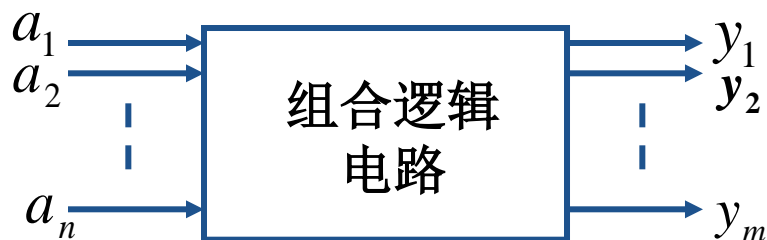
不含记忆（存储）
元件



4-1-1

二、逻辑功能的描述

组合逻辑电路的框图



逻辑函数式

$$y_1 = f_1(a_1 a_2 \cdots a_n)$$

$$y_2 = f_2(a_1 a_2 \cdots a_n)$$

$$\vdots$$

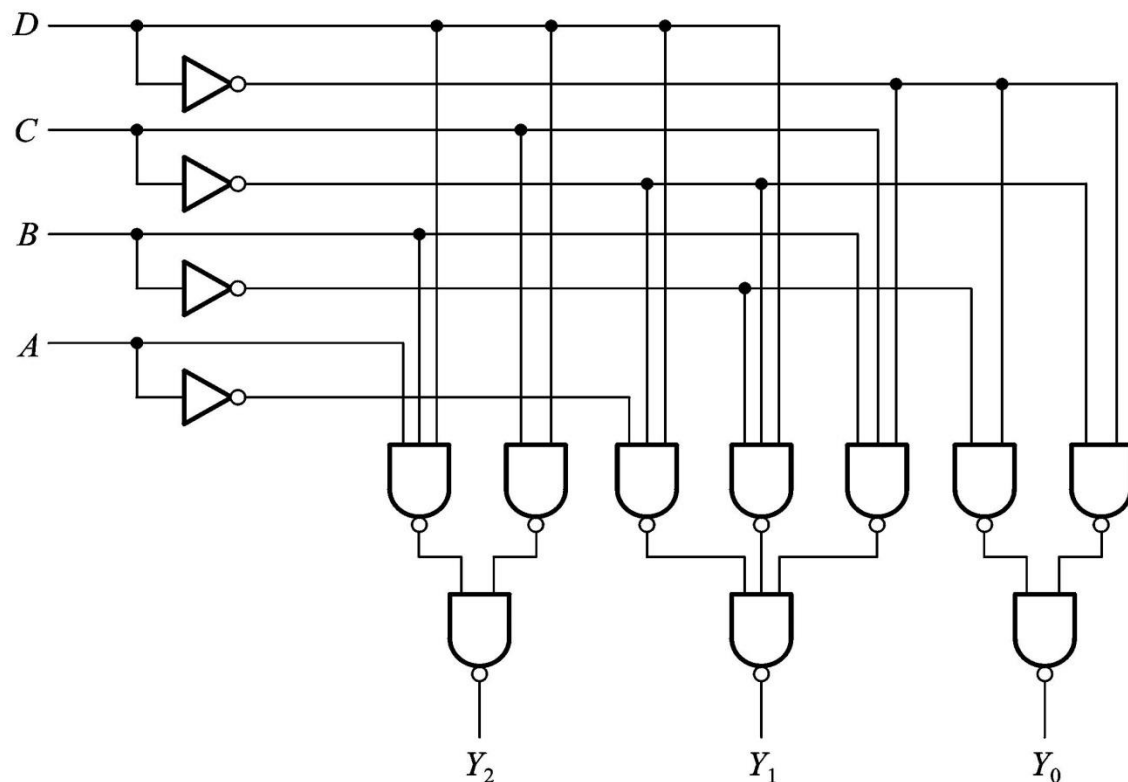
$$y_m = f_m(a_1 a_2 \cdots a_n)$$

$$\Rightarrow Y = F(A)$$

组合逻辑电路分析方法

4.2.1 组合逻辑电路的分析方法

- ❖ 所谓分析一个给定的逻辑电路，就是要通过分析找出电路的逻辑功能来。
- ❖ 分析过程
 - 从电路的输入到输出逐级写出逻辑函数式，最后得到表示输出与输入关系的**逻辑函数式**
 - 用公式化简法或卡诺图化简法将得到的**函数式化简或变换**，使逻辑关系简单明了。
 - 为了使电路的逻辑功能更加直观，有时还可以将逻辑函数式转换为**真值表**的形式。



$$\begin{cases} Y_2 = ((DC)'(DBA)')' = DC + DBA \\ Y_1 = ((D'CB)'(DC'B')'(DC'A'))' = D'CB + DC'B' + DC'A' \\ Y_0 = ((D'C')'(D'B'))' = D'C' + D'B' \end{cases}$$

输 入				输 出		
D	C	B	A	Y_2	Y_1	Y_0
0	0	0	0	0	0	1
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	0	0	1
0	1	0	1	0	0	1
0	1	1	0	0	1	0
0	1	1	1	0	1	0
1	0	0	0	0	1	0
1	0	0	1	0	1	0
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	1	0	0

组合逻辑电路设计方法

4.2.2 组合逻辑电路的设计方法

- ❖ 根据给出的实际逻辑问题，求出实现这一逻辑功能的最简单逻辑电路，这就是设计组合逻辑电路时要完成的工作。
- ❖ 所谓的“最简”
 - 电路所用的器件数量最少
 - 器件的种类最少
 - 器件之间的连线最少

设计步骤

一、逻辑抽象

- ❖ 分析因果关系，确定输入/输出变量
- ❖ 定义逻辑状态的含意（赋值）
- ❖ 列出真值表

二、写出函数式

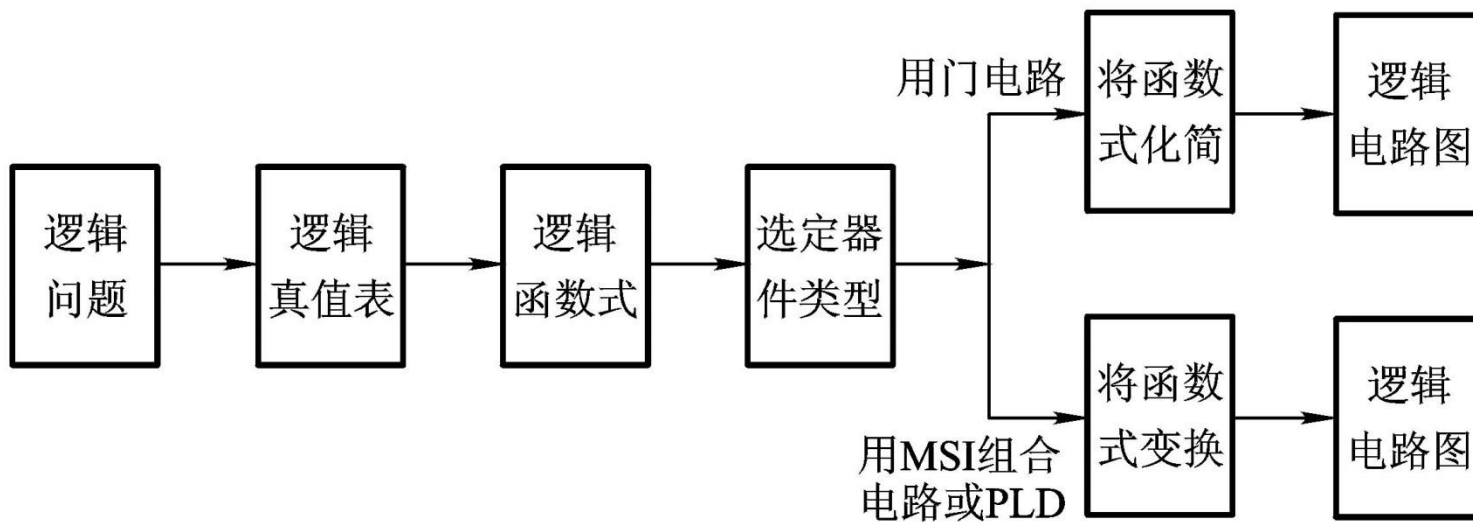
三、选定器件类型

- ## 四、根据所选器件：
- 对逻辑式化简（用门）
 - 变换（用中规模集成电路MSI）
 - 或进行相应的描述（可编程逻辑器件PLD）

五、画出逻辑电路图，或下载到PLD

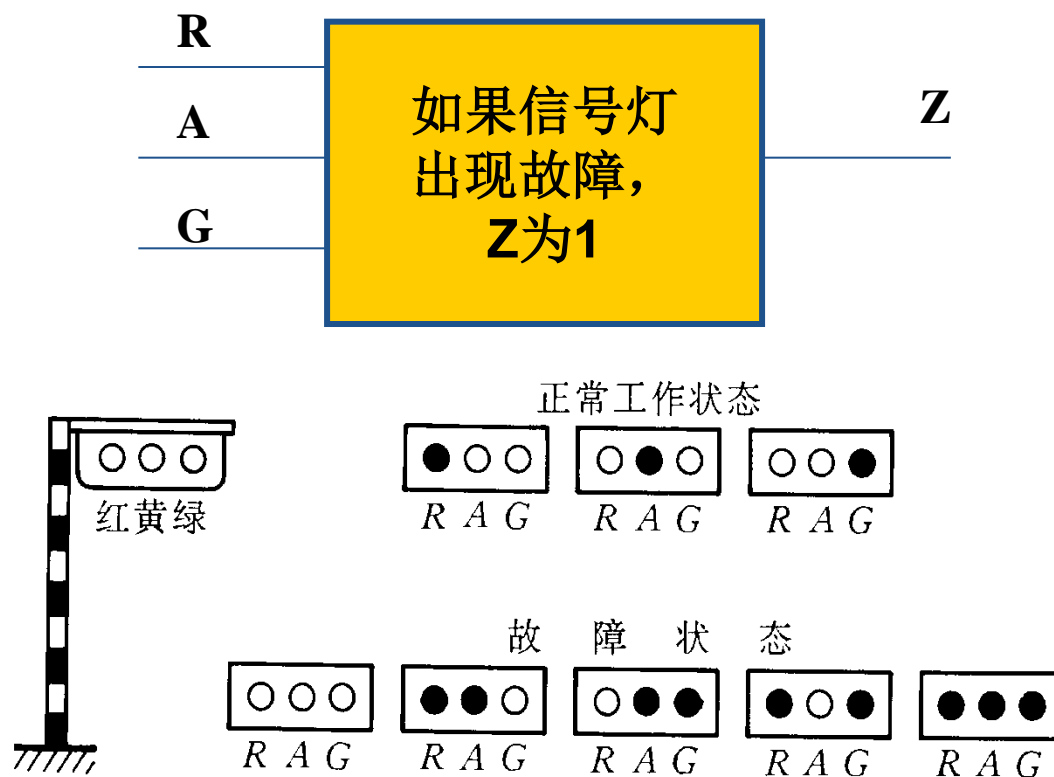
六、工艺设计

设计步骤



设计举例：

❖ 设计一个监视交通信号灯状态的逻辑电路



设计举例：

1.抽象

❖ 输入变量：

红 (R)、黄 (A)、绿 (G)

❖ 输出变量：

故障信号 (Z)

2.写出逻辑表达式

$$Z = R'A'G' + R'AG + RA'G + RAG' + RAG$$

输入变量			输出
R	A	G	Z
0	0	0	1
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

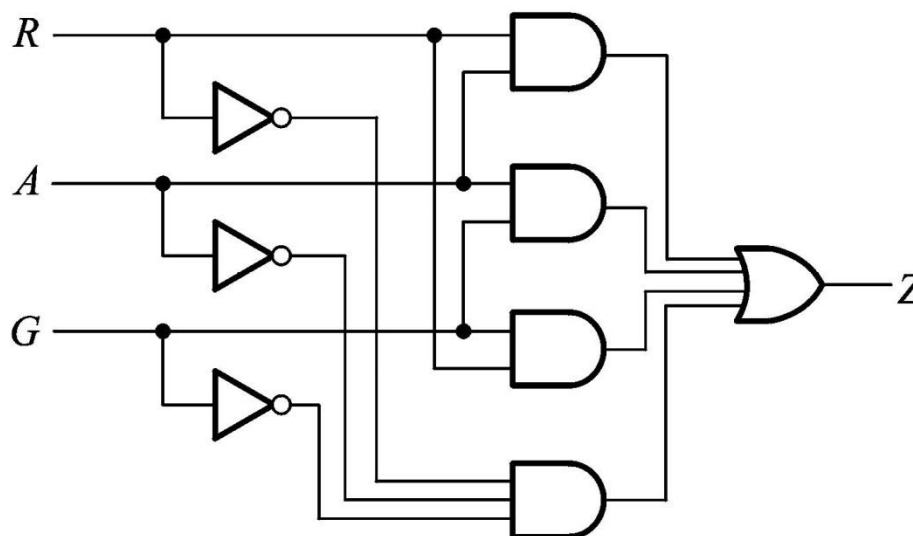
设计举例：

3. 选用小规模SSI器件

4. 化简

$$Z = R' A' G' + RA + RG + AG$$

5. 画出逻辑图



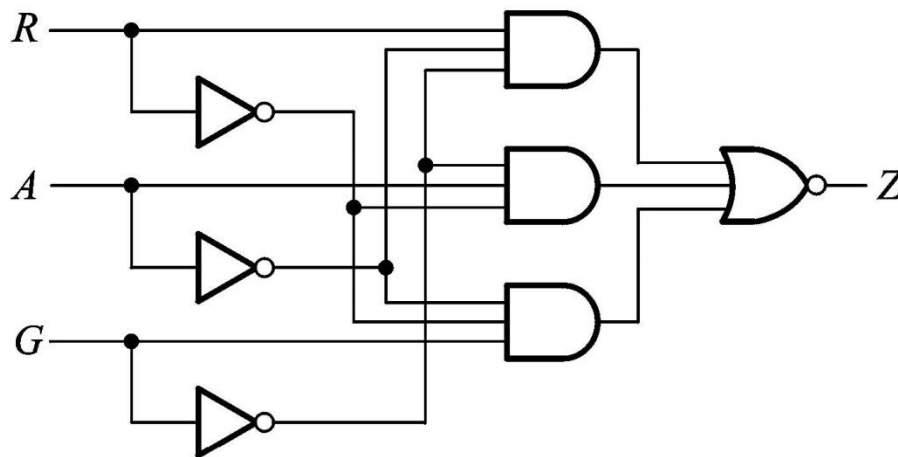
设计举例：

4. 化简（与或非）

$$Z = (RA'G' + R'AG' + R'A'G)'$$

5. 画出逻辑图（与或非）

		AG			
		00	01	11	10
R	0	1	0	1	0
	1	0	1	1	1



普通编码器

4.3 若干常用组合逻辑电路

4.3.1 编码器

❖ 编码：将输入的每个高、低电平信号编成一个对应的二进制代码

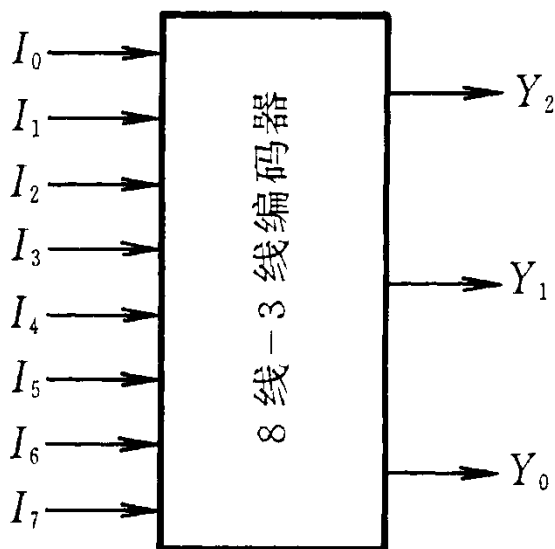
❖ 普通编码器

❖ 优先编码器

一、普通编码器

❖ 特点：任何时刻只允许输入一个编码信号。

❖ 例：3位二进制普通编码器(8线-3线)



输 入								输 出		
I ₀	I ₁	I ₂	I ₃	I ₄	I ₅	I ₆	I ₇	Y ₂	Y ₁	Y 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

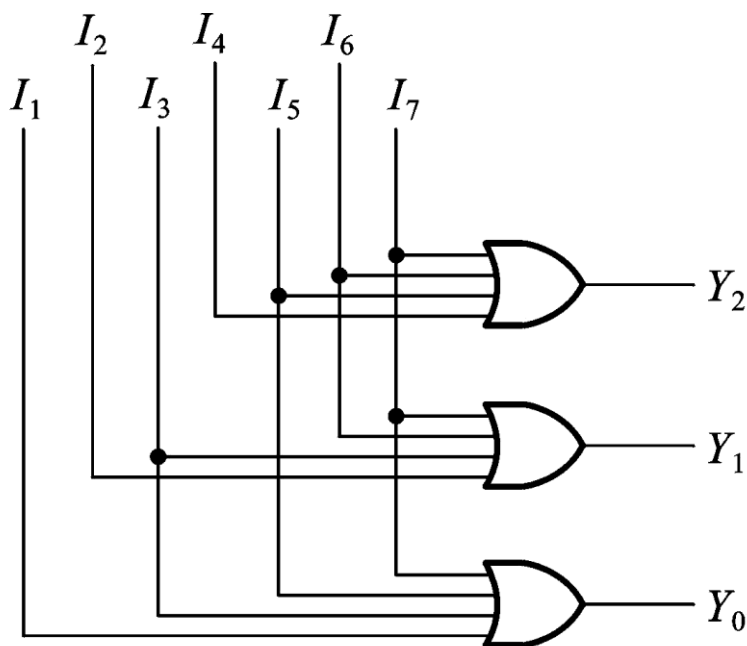
$$Y_2 = I_7' I_6' I_5' I_4' I_3' I_2' I_1' I_0' + I_7' I_6' I_5' I_4' I_3' I_2' I_1' I_0' + I_7' I_6' I_5' I_4' I_3' I_2' I_1' I_0' + I_7' I_6' I_5' I_4' I_3' I_2' I_1' I_0'$$

利用无关项化简，得：

$$Y_2 = I_4 + I_5 + I_6 + I_7$$

$$Y_1 = I_2 + I_3 + I_6 + I_7$$

$$Y_0 = I_1 + I_3 + I_5 + I_7$$



优先编码器

二、优先编码器

❖ 特点：允许同时输入两个以上的编码信号，但只对其中优先权最高的一个进行编码。

❖ 例：8线-3线优先编码器

❖ （设 I_7 优先权最高... I_0 优先权最低）

输 入								输 出		
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	Y_2	Y_1	Y_0
X	X	X	X	X	X	X	1	1	1	1
X	X	X	X	X	X	1	0	1	1	0
X	X	X	X	X	1	0	0	1	0	1
X	X	X	X	1	0	0	0	1	0	0
X	X	X	1	0	0	0	0	0	1	1
X	X	1	0	0	0	0	0	0	1	0
X	1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0

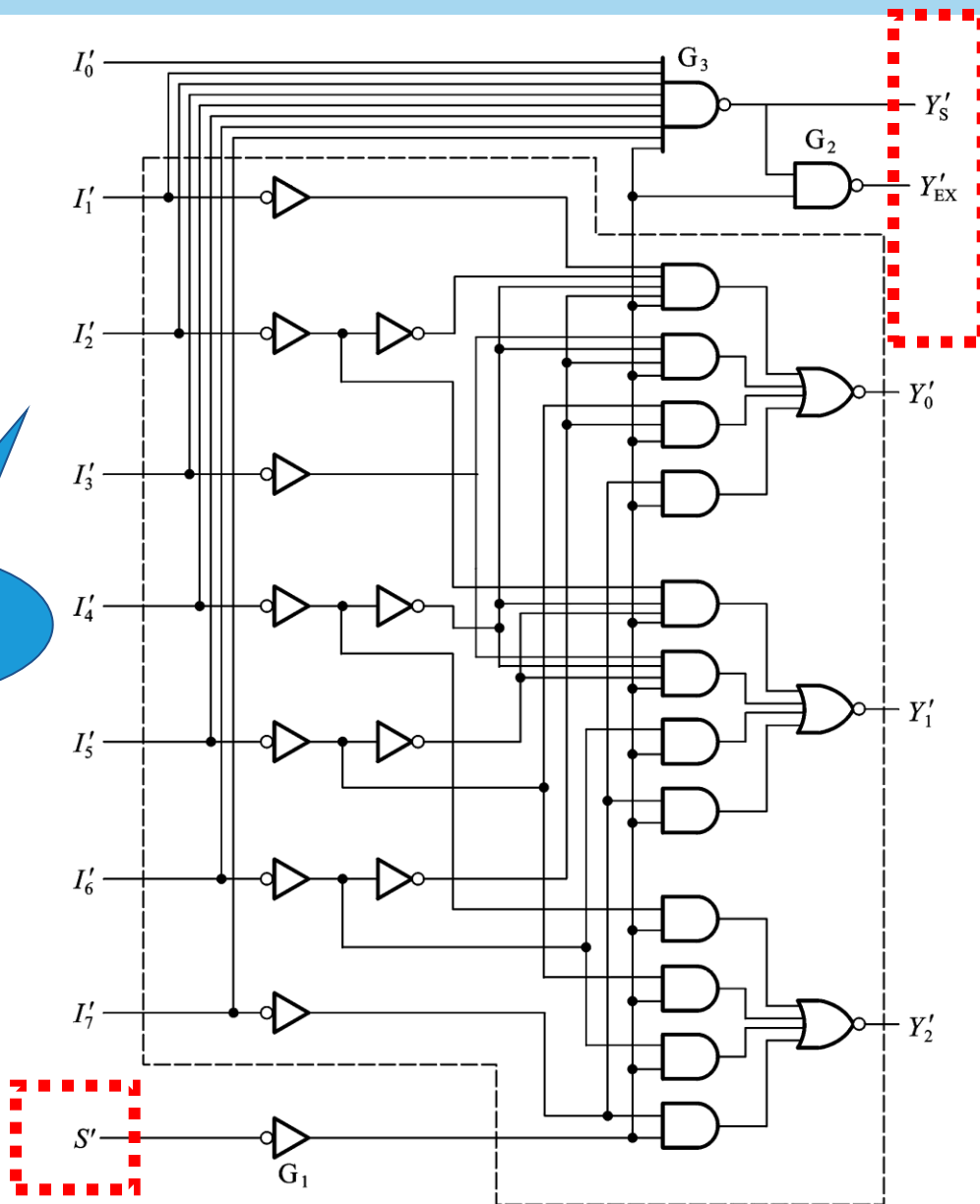
$$Y_2 = I_7 + I_7' I_6 + I_7' I_6' I_5 + I_7' I_6' I_5' I_4$$



$$A + A'B = A + B$$

$$Y_2 = I_7 + I_6 + I_5 + I_4$$

实例： 74HC148



低电
平

$$Y_2' = (I_7 + I_6 + I_5 + I_4)'$$



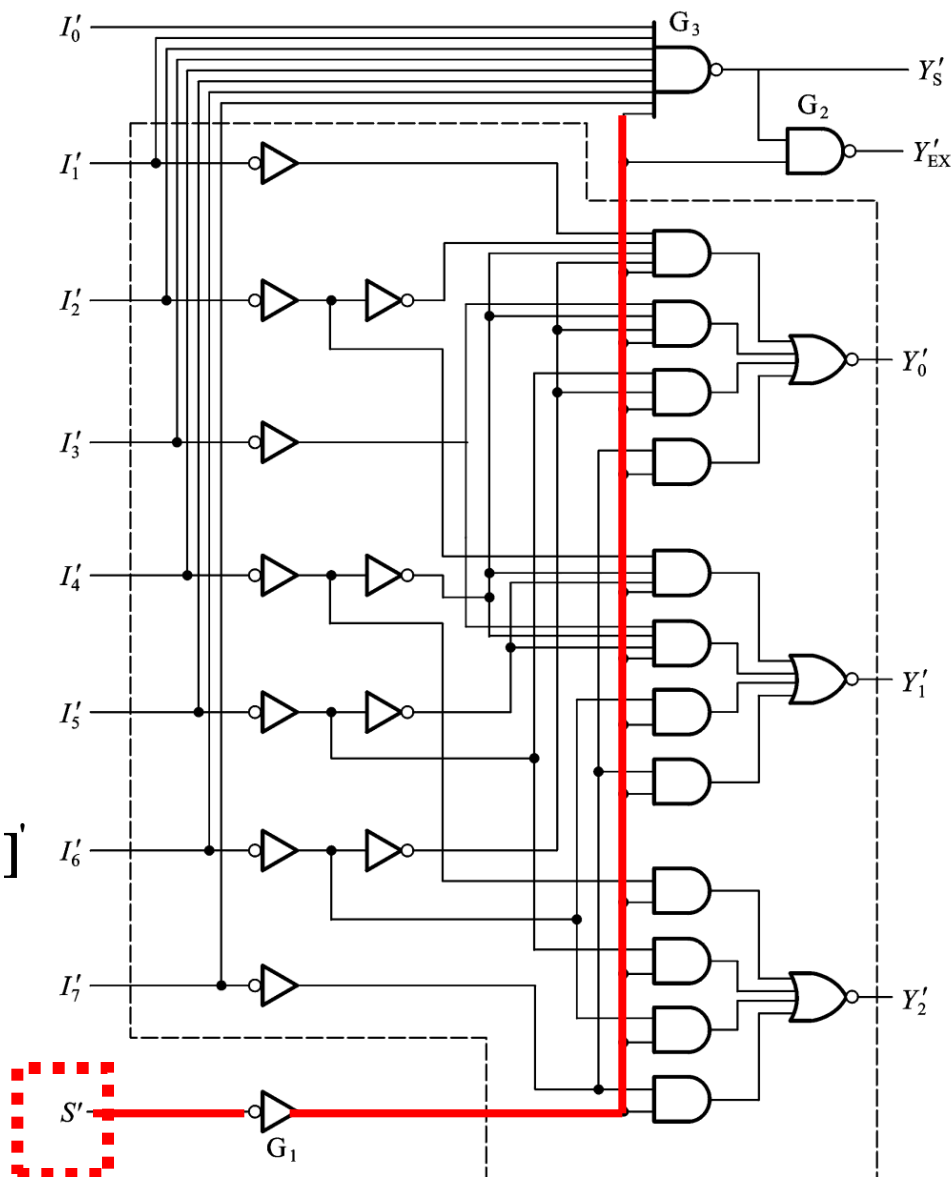
$$Y_2' = [(I_7 + I_6 + I_5 + I_4)S']$$

$$Y_2' = [(I_7 + I_6 + I_5 + I_4)S']$$

$$Y_1' = [(I_7 + I_6 + I_5 I_4' I_3' + I_2 I_4' I_5')S']$$

$$Y_0' = [(I_7 + I_6' I_5 + I_3 I_4' I_6' + I_1 I_2 I_4' I_6')S']$$

选通信号



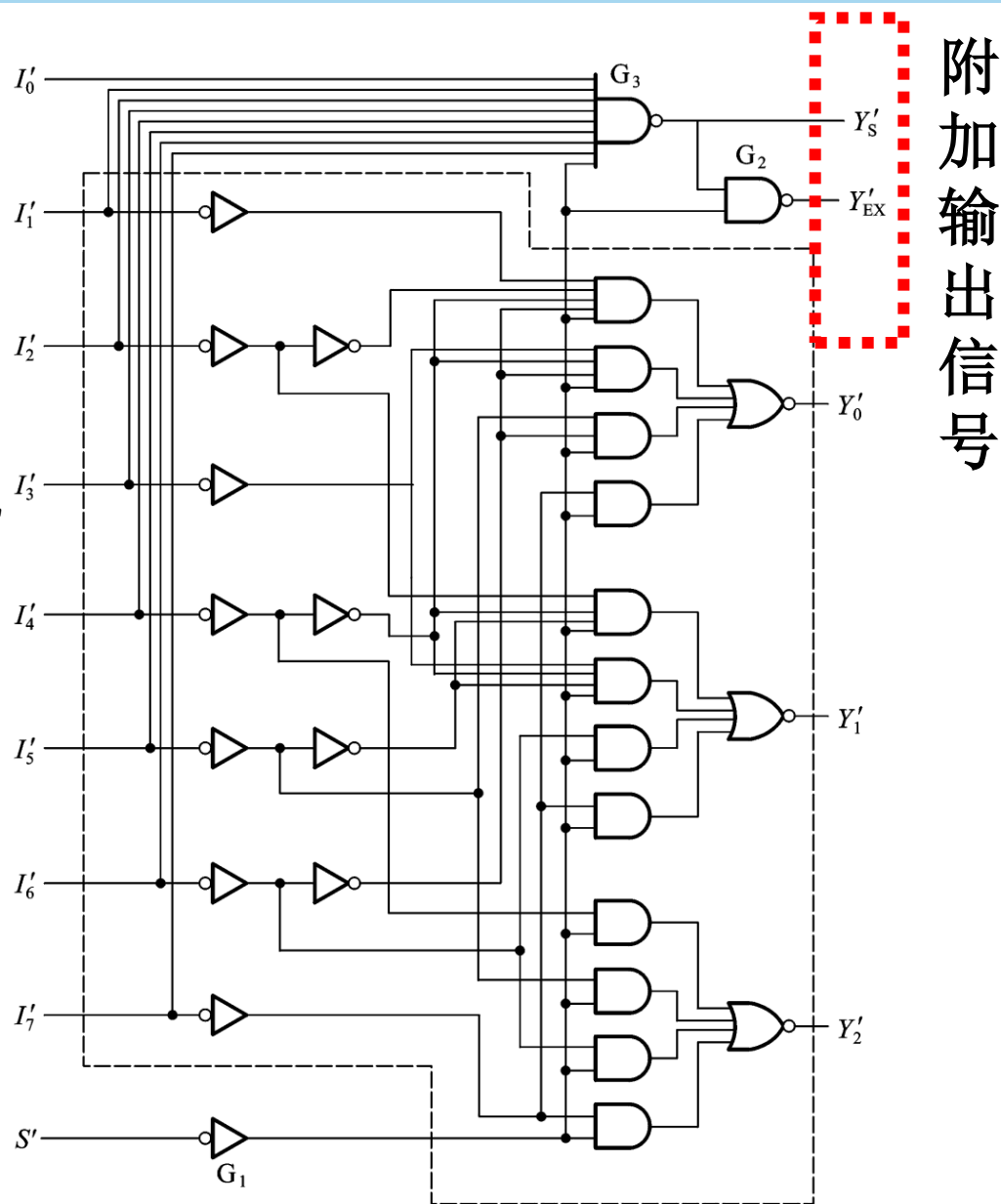
为0时，电路工作
无编码输入

$$Y'_S = (I'_7 I'_6 I'_5 I'_4 I'_3 I'_2 I'_1 I'_0 S)'$$

$$Y'_{EX} = [(I'_7 I'_6 I'_5 I'_4 I'_3 I'_2 I'_1 I'_0 S)' \quad S']$$

$$= [(I'_7 + I'_6 + I'_5 + I'_4 + I'_3 + I'_2 + I'_1 + I'_0) S']$$

为0时，电路工作
有编码输入



输 入									输 出				
S'	I_0'	I_1'	I_2'	I_3'	I_4'	I_5'	I_6'	I_7'	Y_2'	Y_1'	Y_0'	Y_S'	Y_{EX}'
1	X	X	X	X	X	X	X	X	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	0	1
0	X	X	X	X	X	X	X	0	0	0	0	1	0
0	X	X	X	X	X	X	0	1	0	0	1	1	0
0	X	X	X	X	X	0	1	1	0	1	0	1	0
0	X	X	X	X	0	1	1	1	0	1	1	1	0
0	X	X	X	0	1	1	1	1	1	0	0	1	0
0	X	X	0	1	1	1	1	1	1	0	1	1	0
0	X	0	1	1	1	1	1	1	1	1	0	1	0
0	0	1	1	1	1	1	1	1	1	1	1	1	0

附加输出信号的状态及含意

Y'_S	Y'_{EX}	状态
1	1	不工作
0	1	工作，但无输入
1	0	工作，且有输入
0	0	不可能出现

优先编码器扩展

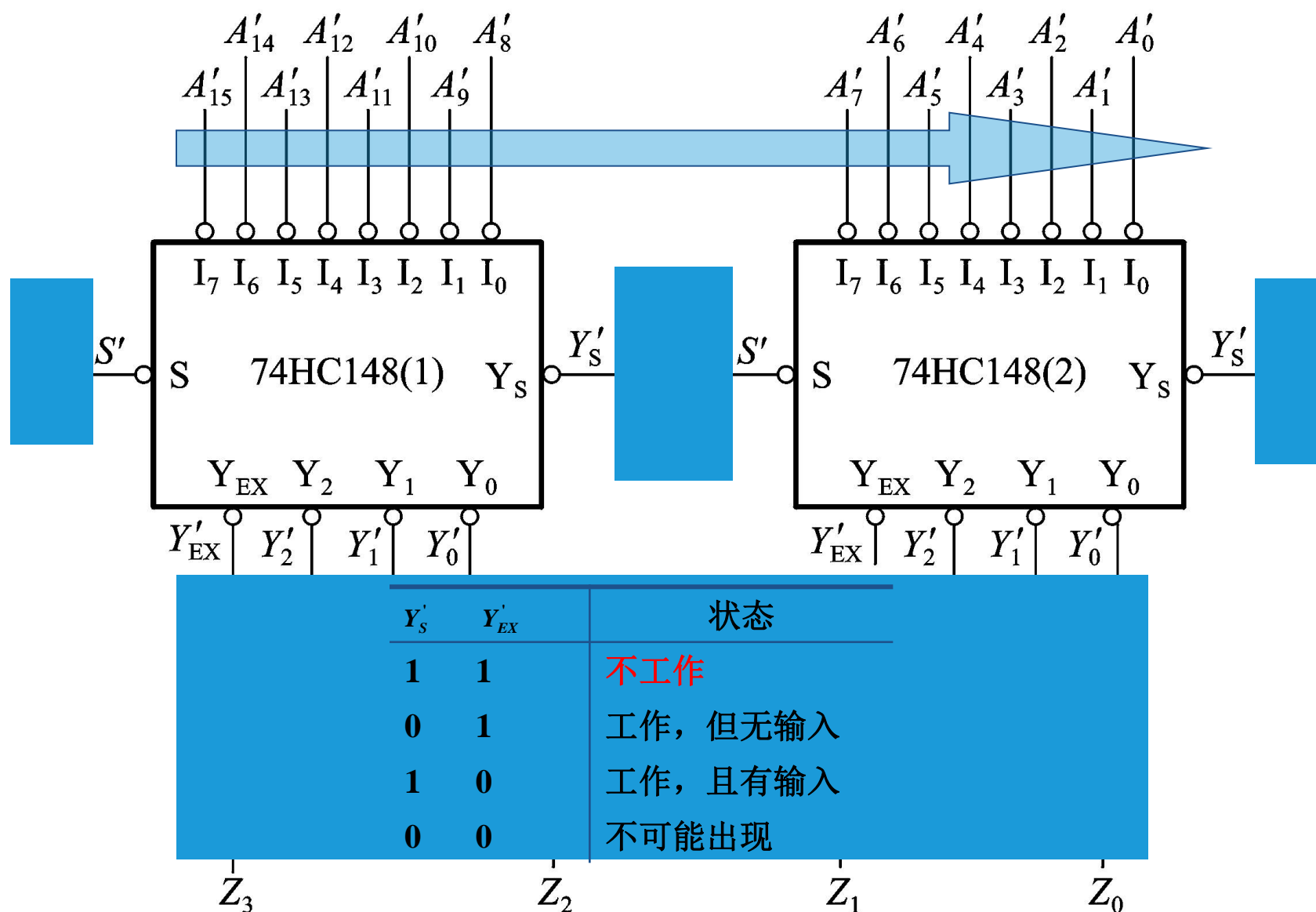
控制端扩展功能举例：

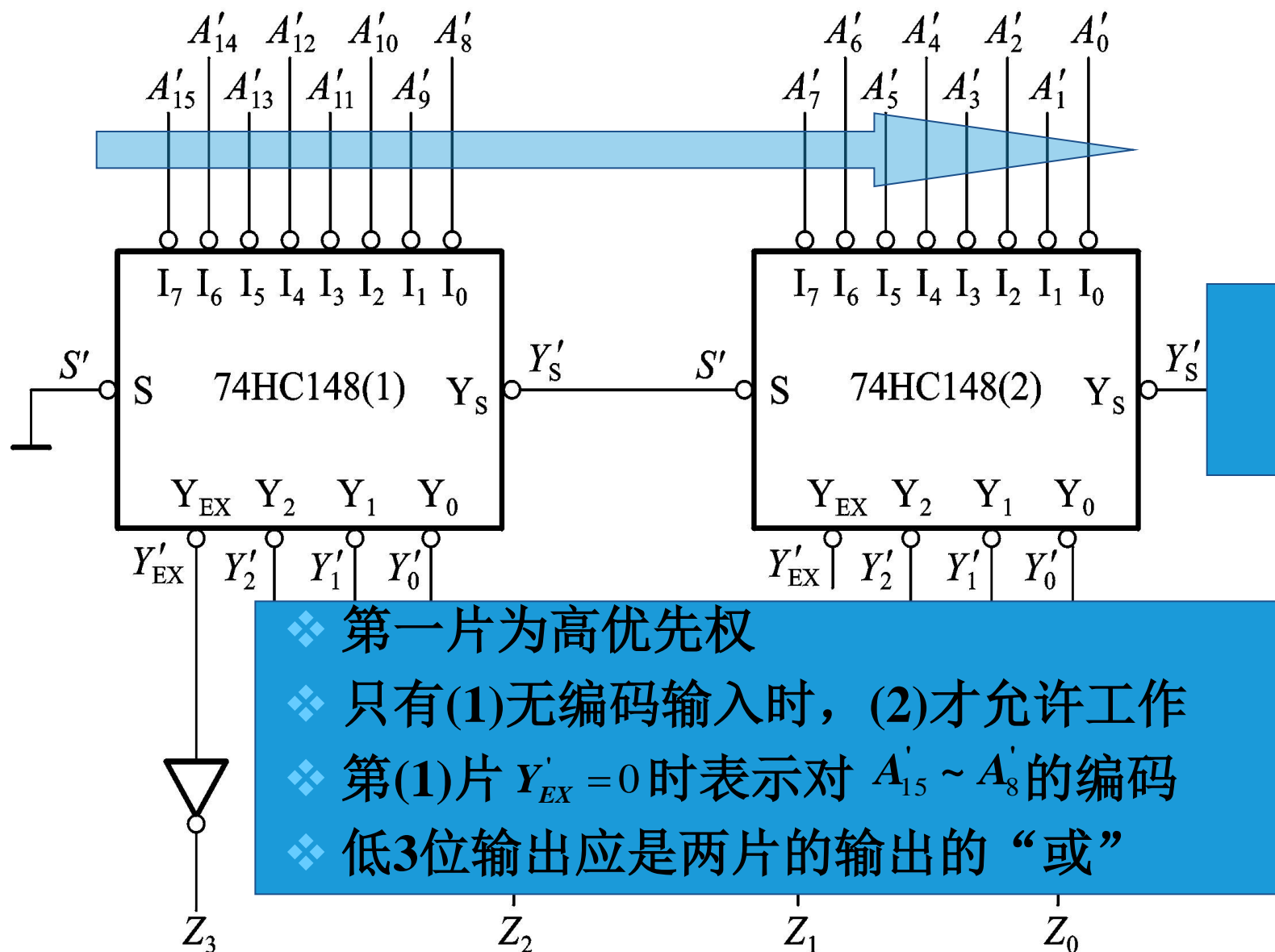
❖ 例：用两片8线-3线优先编码器



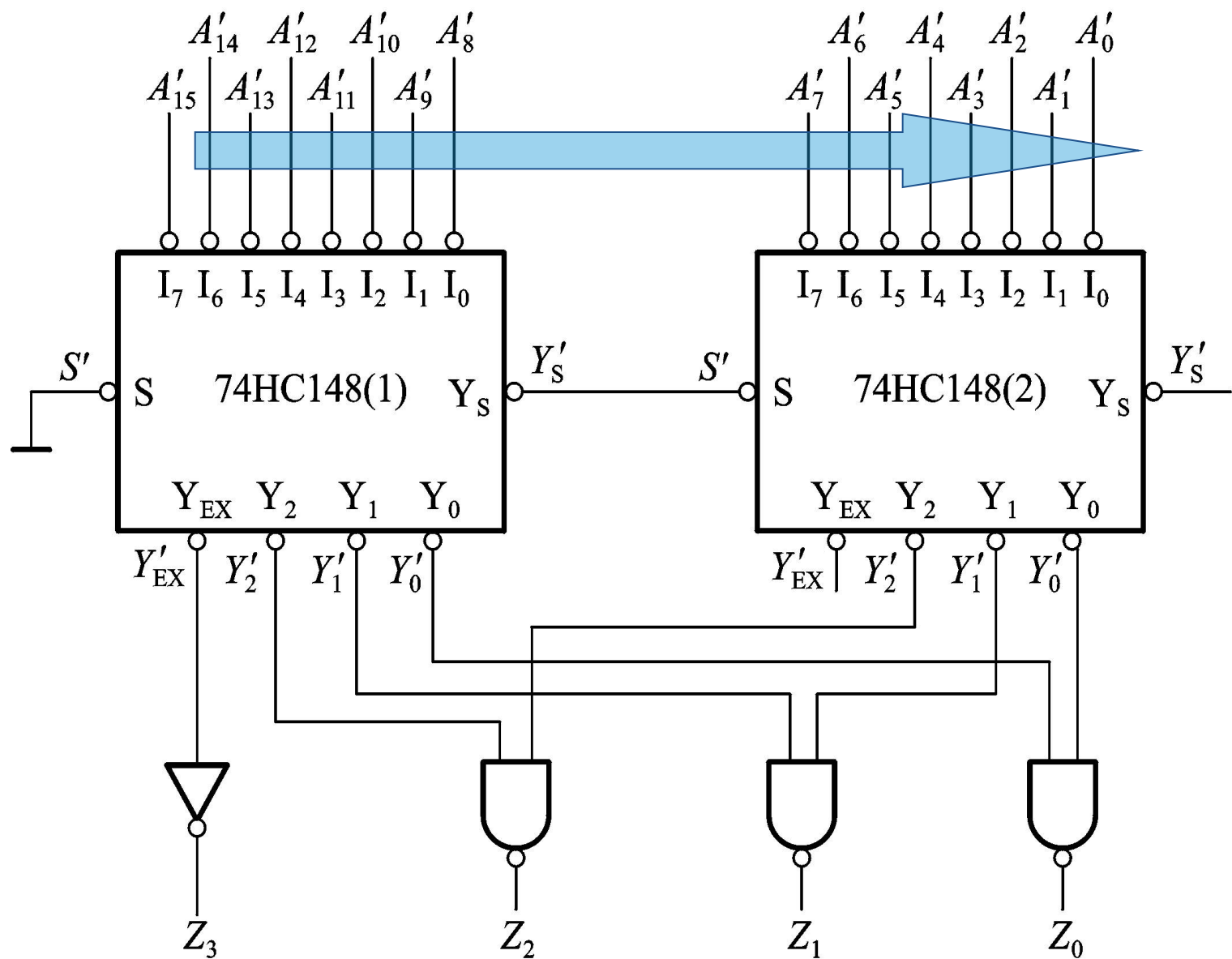
16线-4线优先编码器

其中， A'_{15} 的优先权最高...





- ❖ 第一片为高优先权
- ❖ 只有(1)无编码输入时，(2)才允许工作
- ❖ 第(1)片 $Y'_{EX} = 0$ 时表示对 $A'_{15} \sim A'_8$ 的编码
- ❖ 低3位输出应是两片的输出的“或”

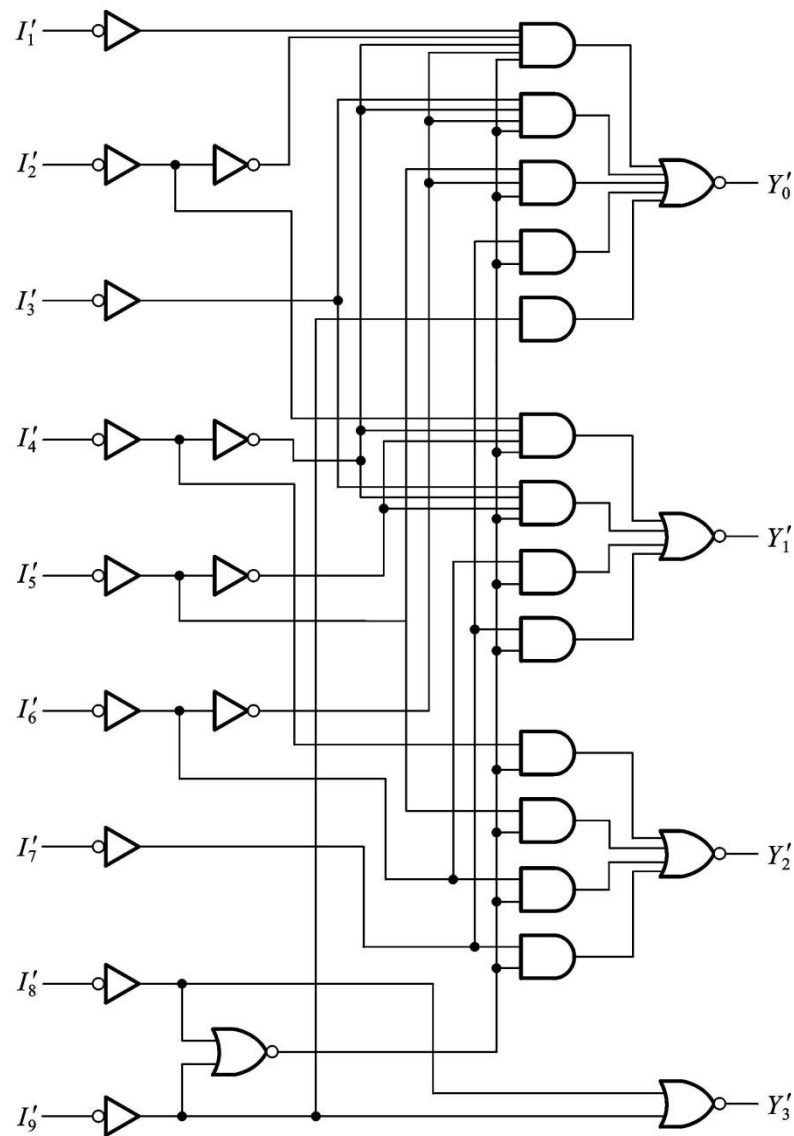


其它编码器

三、二-十进制优先编码器

- ❖ 将 $I'_9 \sim I'_1$ 编成 0110 ~ 1110 (BCD码的反码形式)
- ❖ I'_9 的优先权最高, I'_0 最低
- ❖ 输入的低电平信号变成一个对应的十进制的编码

二-十进制优先编码器74LS147



输 入									输 出			
I_1'	I_2'	I_3'	I_4'	I_5'	I_6'	I_7'	I_8'	I_9'	Y_3'	Y_2'	Y_1'	Y_0'
1	1	1	1	1	1	1	1	1	1	1	1	1
X	X	X	X	X	X	X	X	0	0	1	1	0
X	X	X	X	X	X	X	0	1	0	1	1	1
X	X	X	X	X	X	0	1	1	1	0	0	0
X	X	X	X	X	0	1	1	1	1	0	0	1
X	X	X	X	0	1	1	1	1	1	0	1	0
X	X	X	0	1	1	1	1	1	1	0	1	1
X	X	0	1	1	1	1	1	1	1	1	0	0
X	0	1	1	1	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	0

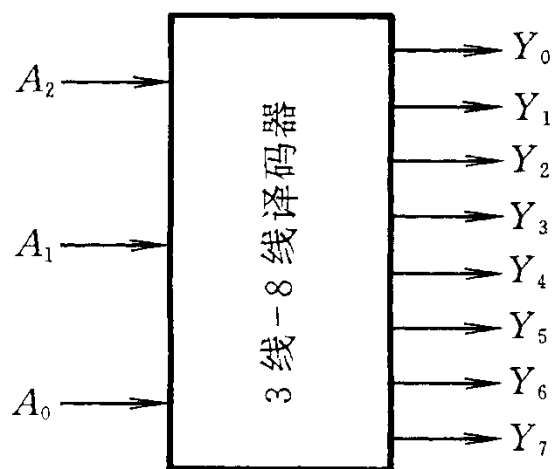
译码器

4.3.2 译码器

- ❖ 译码：将每个输入的二进制代码译成对应的输出高、低电平信号或另外一个代码。（译码是编码的反操作）
- ❖ 常用的有：二进制译码器，二-十进制译码器，显示译码器等。

一、二进制译码器

例：3线—8线译码器



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

真值表



逻辑表达式:

$$Y_0 = A_2' A_1' A_0' = m_0$$

$$Y_1 = A_2' A_1' A_0 = m_1$$

$$Y_2 = A_2' A_1 A_0' = m_2$$

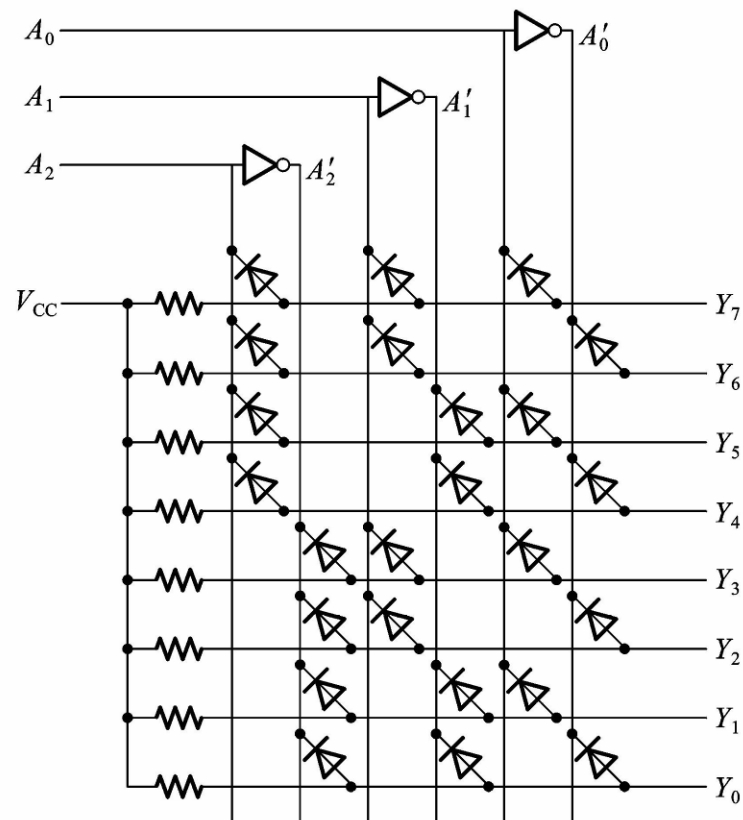
...

$$Y_7 = A_2 A_1 A_0 = m_7$$



用电路进行实现

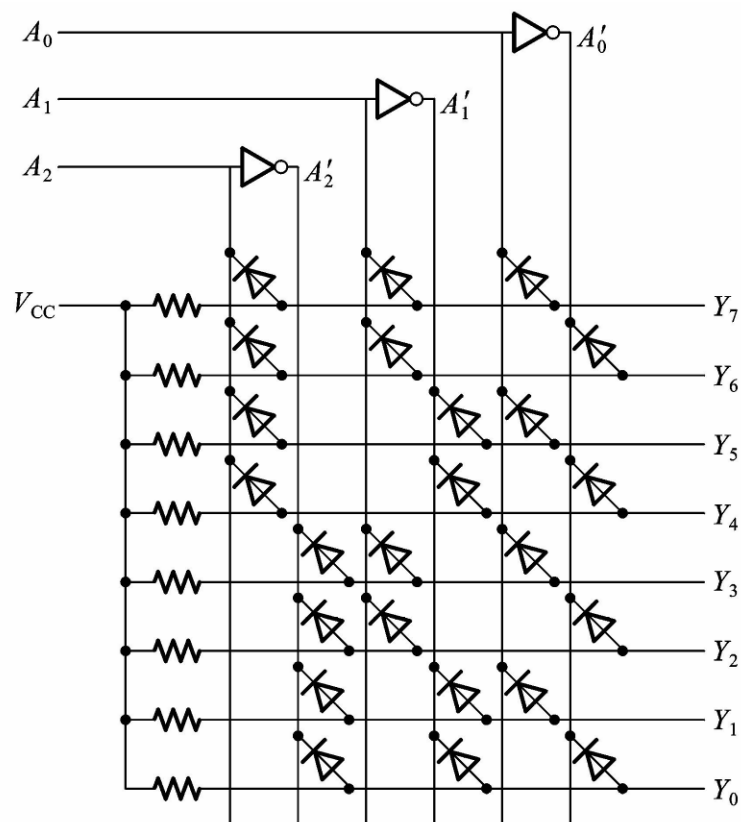
用二极管与门
阵列组成的3线
—8线译码器



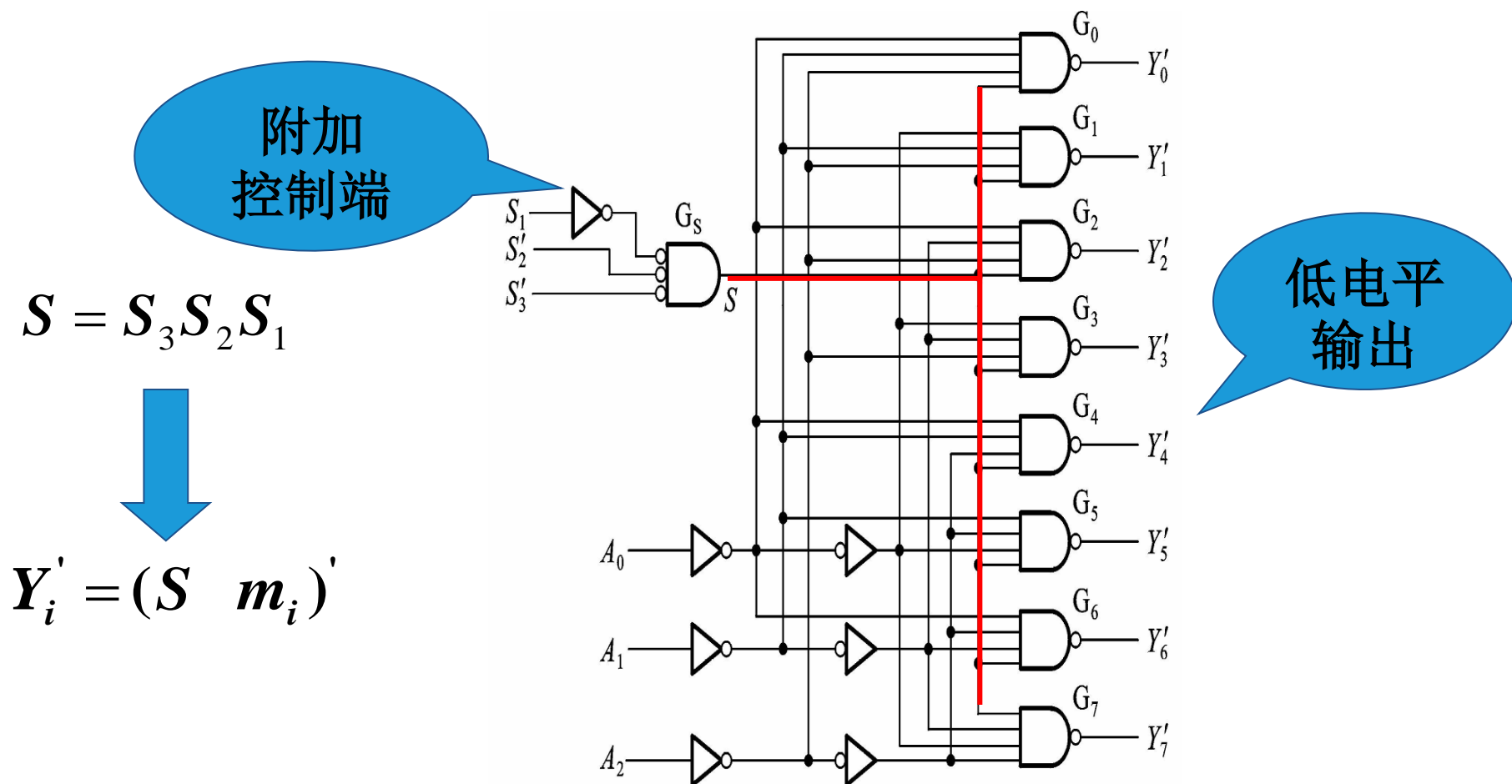
用二极管与门阵列组成的3线—8线译码器

●缺点:

1. 电路的输入电阻较低而输出电阻较高;
2. 输出的高、低电平信号发生偏移 (偏离输入信号的高、低电平)



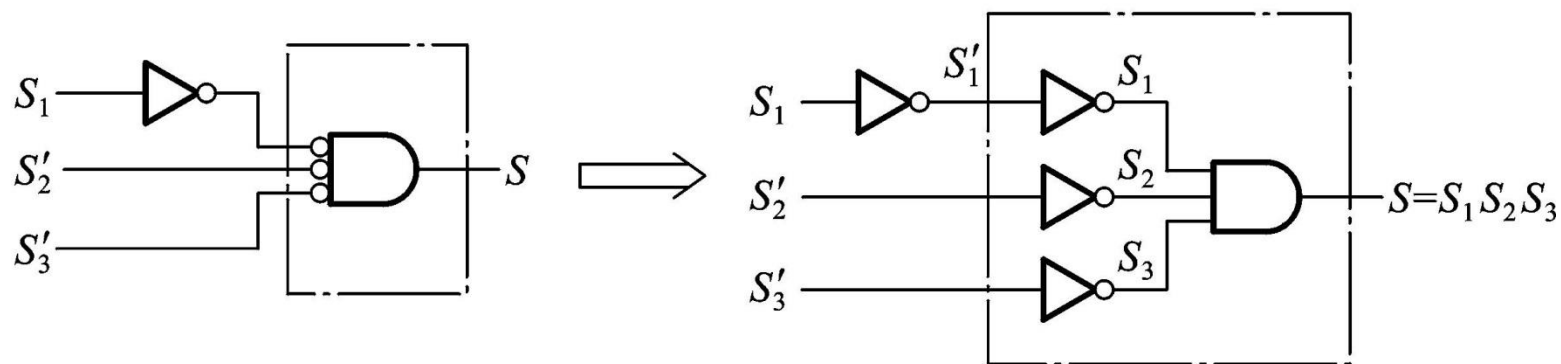
集成译码器实例：74HC138



74HC138的功能表:

输 入					输 出							
S_1	$S_2' + S_3'$	A_2	A_1	A_0	Y_7'	Y_6'	Y_5'	Y_4'	Y_3'	Y_2'	Y_1'	Y_0'
0	X	X	X	X	1	1	1	1	1	1	1	1
X	1	X	X	X	1	1	1	1	1	1	1	1
1	0	0	0	0	1	1	1	1	1	1	1	0
1	0	0	0	1	1	1	1	1	1	1	0	1
1	0	0	1	0	1	1	1	1	1	0	1	1
1	0	0	1	1	1	1	1	1	0	1	1	1
1	0	1	0	0	1	1	1	0	1	1	1	1
1	0	1	0	1	1	1	0	1	1	1	1	1
1	0	1	1	0	1	0	1	1	1	1	1	1
1	0	1	1	1	0	1	1	1	1	1	1	1

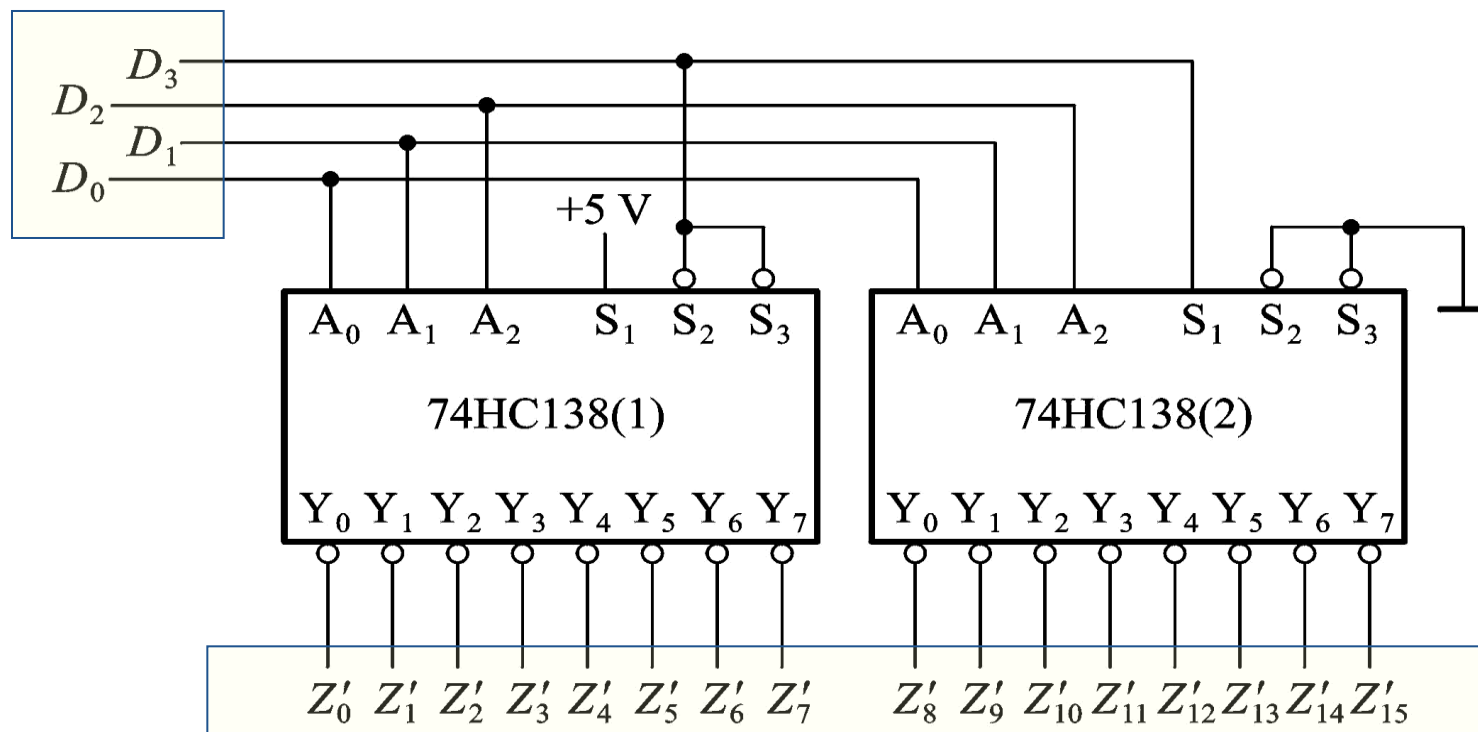
可以理解为地址

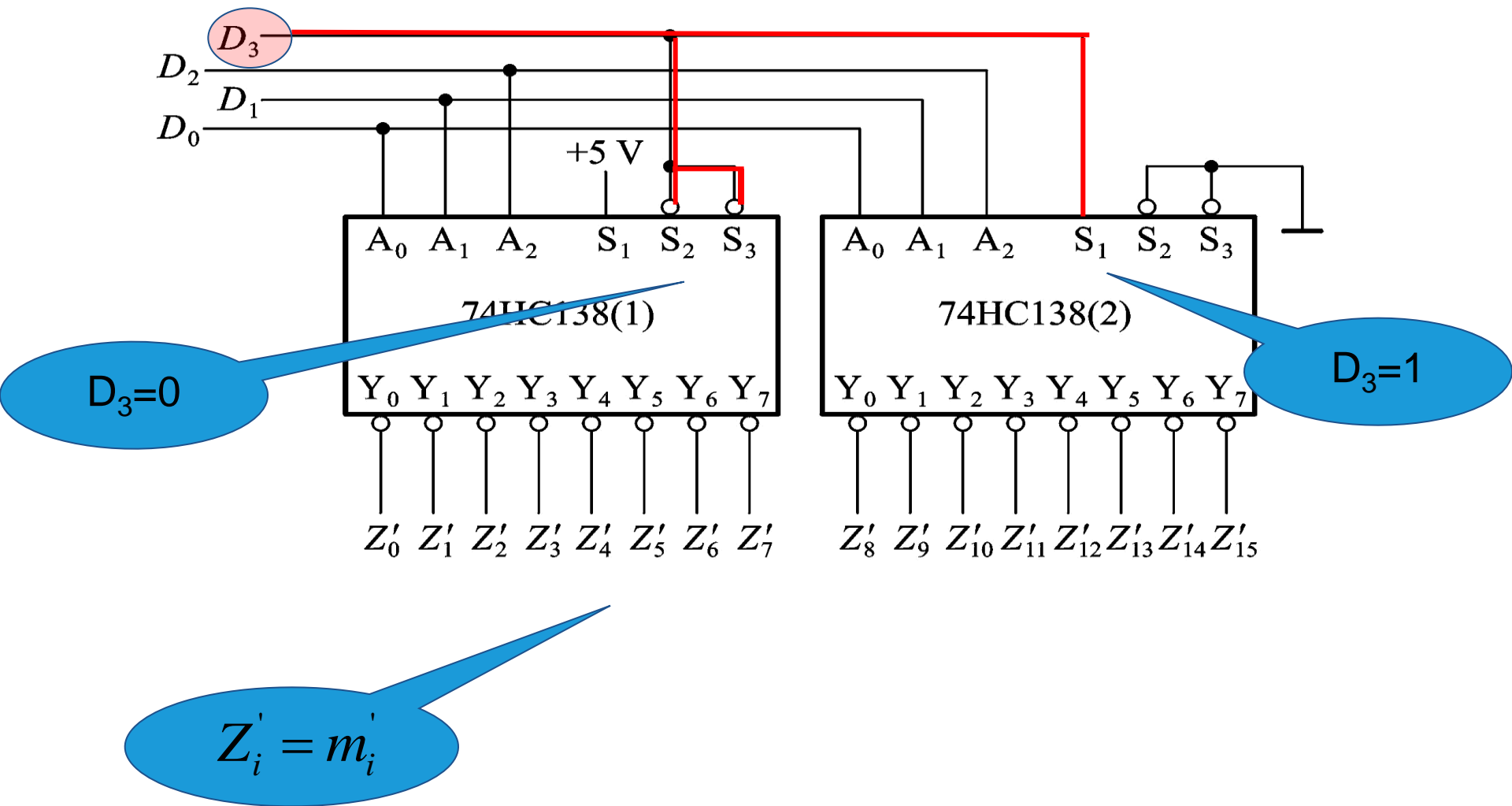


译码器扩展

❖ 利用附加控制端进行扩展

例：用74HC138（3线—8线译码器）组成
4线—16线译码器



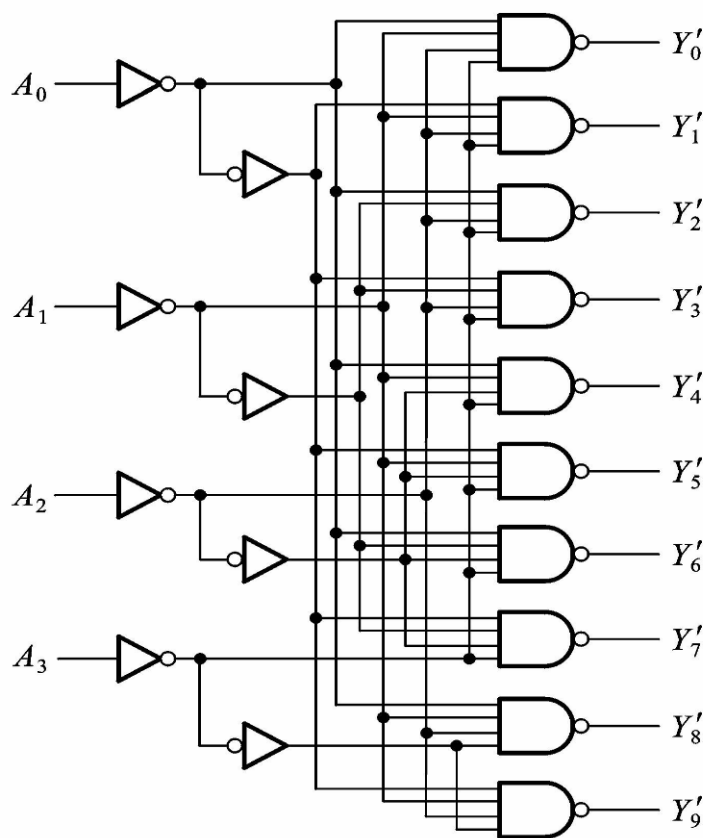


二、二—十进制译码器

- ❖ 将输入BCD码的10个代码译成10个高、低电平的输出信号
BCD码以外的伪码，输出均无低电平信号产生

- ❖ 例：74HC42

$$Y'_i = m'_i \quad (i = 0 \sim 9) \quad \longleftrightarrow$$

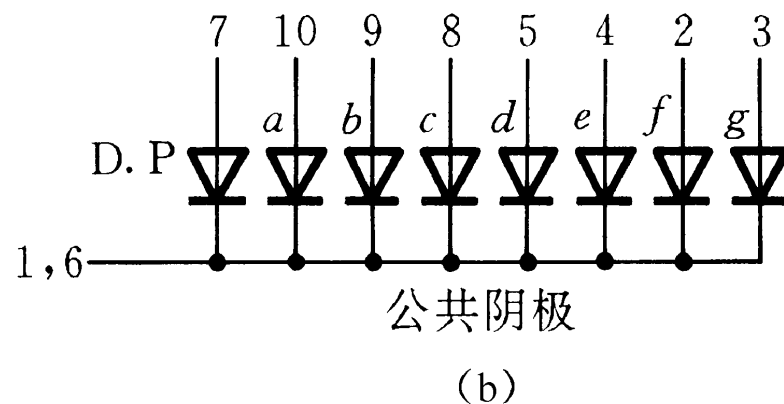
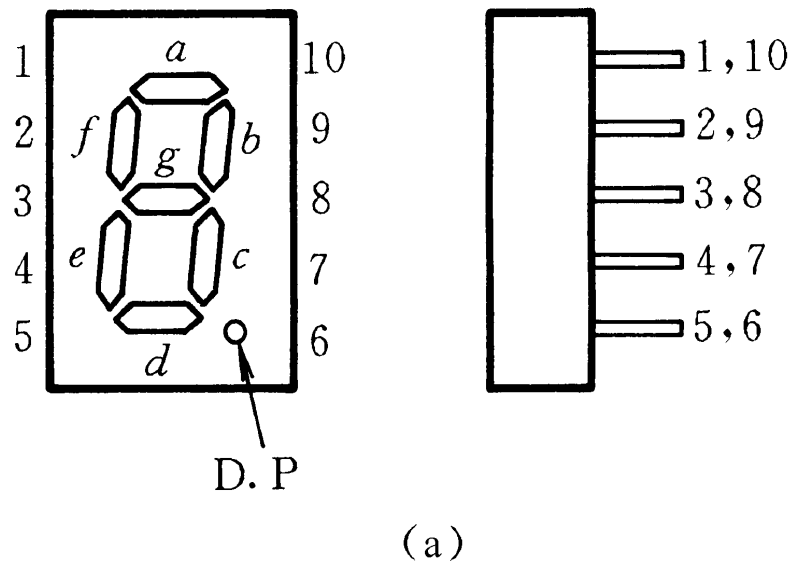


显示译码器

三、显示译码器

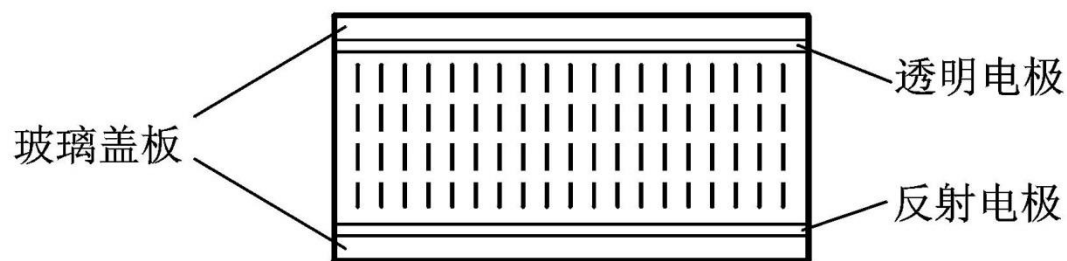
❖ 1. 七段字符显示器

如：半导体数码管BS201A

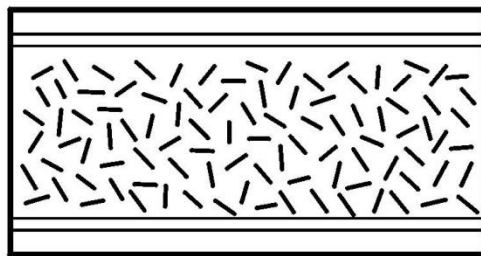


三、显示译码器

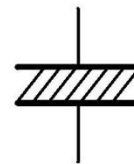
❖ 如：液晶显示器



(a)



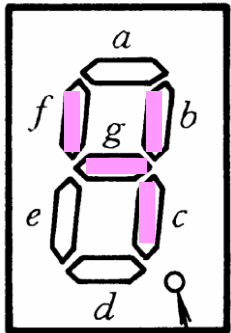
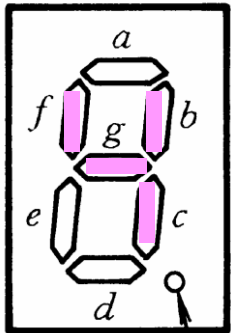
(b)

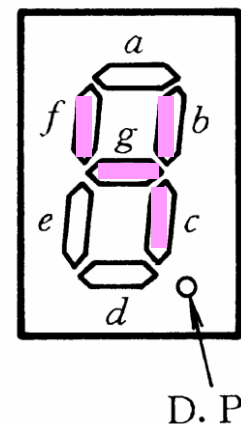


(c)

2. BCD七段字符显示译码器

(代码转换器) 7448

输 入					输 出							字形
数字	A ₃	A ₂	A ₁	A ₀	Y _a	Y _b	Y _c	Y _d	Y _e	Y _f	Y _g	
0	0	0	0	0	1	1	1	1	1	1	0	
1	0	0	0	1	0	1	1	0	0	0	0	
2	0	0	1	0	1	1	0	1	1	0	1	
3	0	0	1	1	1	1	1	1	0	0	1	
4	0	1	0	0	0	1	1	0	0	1	1	
5	0	1	0	1	1	0	1	1	0	1	1	
6	0	1	1	0	0	0	1	1	1	1	1	
7	0	1	1	1	1	1	1	0	0	0	0	
8	1	0	0	0	1	1	1	1	1	1	1	
9	1	0	0	1	1	1	1	0	0	1	1	
10	1	0	1	0	0	0	0	1	1	0	1	
11	1	0	1	1	0	0	1	1	0	0	1	
12	1	1	0	0	0	1	0	0	0	1	1	
13	1	1	0	1	1	0	0	1	0	1	1	
14	1	1	1	0	0	0	0	1	1	1	1	
15	1	1	1	1	0	0	0	0	0	0	0	



A_1A_0		A_3A_2			
		00	01	11	10
A_3A_2	00	1	0	1	1
	01	0	1	1	0
	11	0	1	0	0
	10	1	1	0	0

(a)

A_1A_0		A_3A_2			
		00	01	11	10
A_3A_2	00	1	1	1	1
	01	1	0	1	0
	11	1	0	0	0
	10	1	1	0	0

(b)

A_1A_0		A_3A_2			
		00	01	11	10
A_3A_2	00	1	1	1	0
	01	1	1	1	1
	11	0	0	0	0
	10	1	1	1	0

(c)

A_1A_0		A_3A_2			
		00	01	11	10
A_3A_2	00	1	0	1	1
	01	0	1	0	1
	11	0	1	0	1
	10	1	0	1	1

(d)

A_1A_0		A_3A_2			
		00	01	11	10
A_3A_2	00	1	0	0	1
	01	0	0	0	1
	11	0	0	0	1
	10	1	0	0	1

(e)

A_1A_0		A_3A_2			
		00	01	11	10
A_3A_2	00	1	0	0	0
	01	1	1	0	1
	11	1	1	0	1
	10	1	1	0	0

(f)

A_1A_0		A_3A_2			
		00	01	11	10
A_3A_2	00	0	0	1	1
	01	1	1	0	1
	11	1	1	0	1
	10	1	1	1	1

(g)

BCD-七段显示译码器7448的逻辑图

$$Y_a = (A_3'A_2'A_1'A_0 + A_3A_1 + A_2A_0)'$$

$$Y_b = (A_3A_1 + A_2A_1A_0' + A_2A_1'A_0)'$$

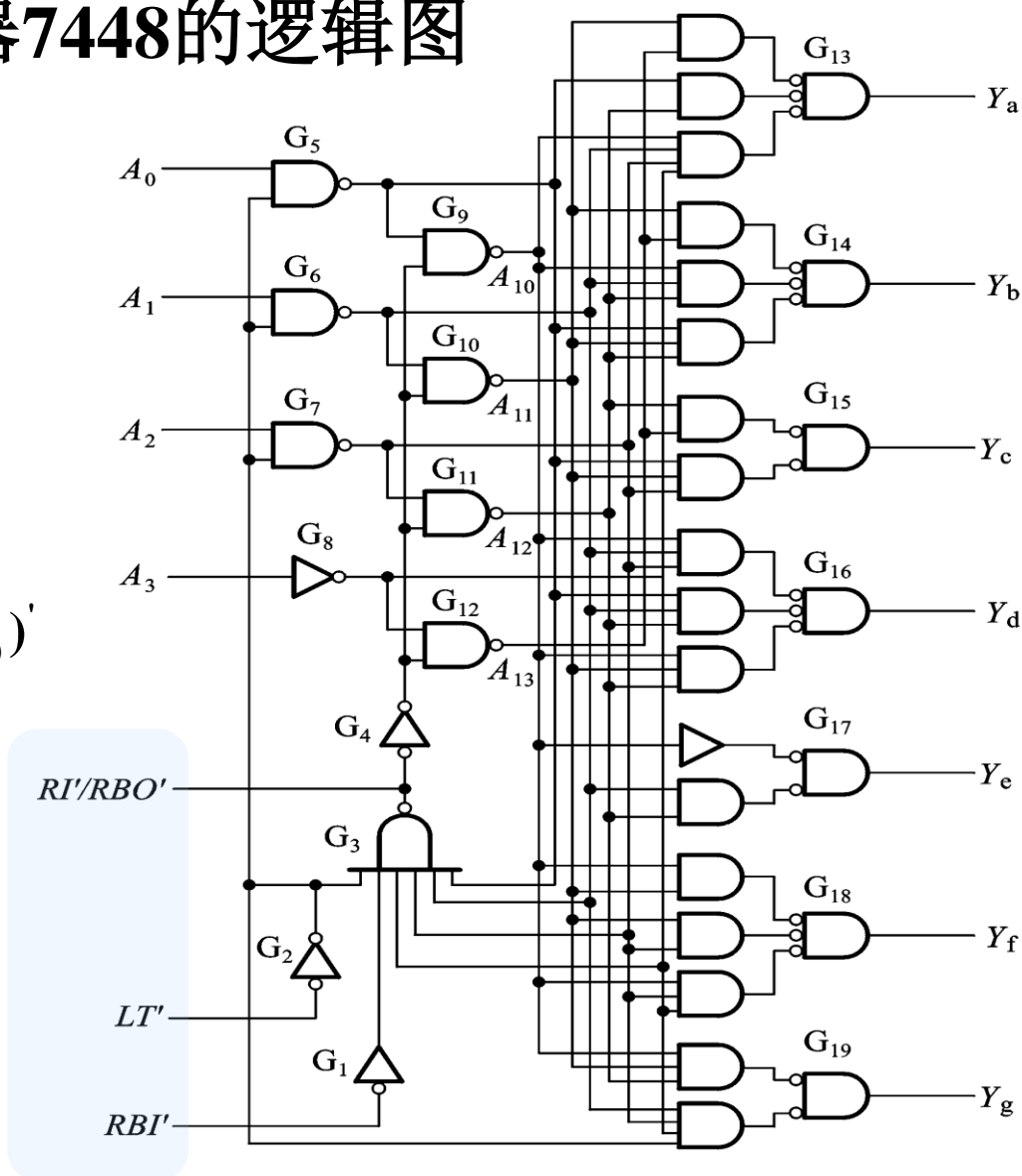
$$Y_c = (A_3A_2 + A_2'A_1A_0)'$$

$$Y_d = (A_2A_1A_0 + A_2A_1'A_0' + A_2'A_1'A_0)'$$

$$Y_e = (A_2A_1' + A_0)'$$

$$Y_f = (A_3'A_2'A_0 + A_2'A_1 + A_1A_0)'$$

$$Y_g = (A_3'A_2'A_1' + A_2A_1A_0)'$$

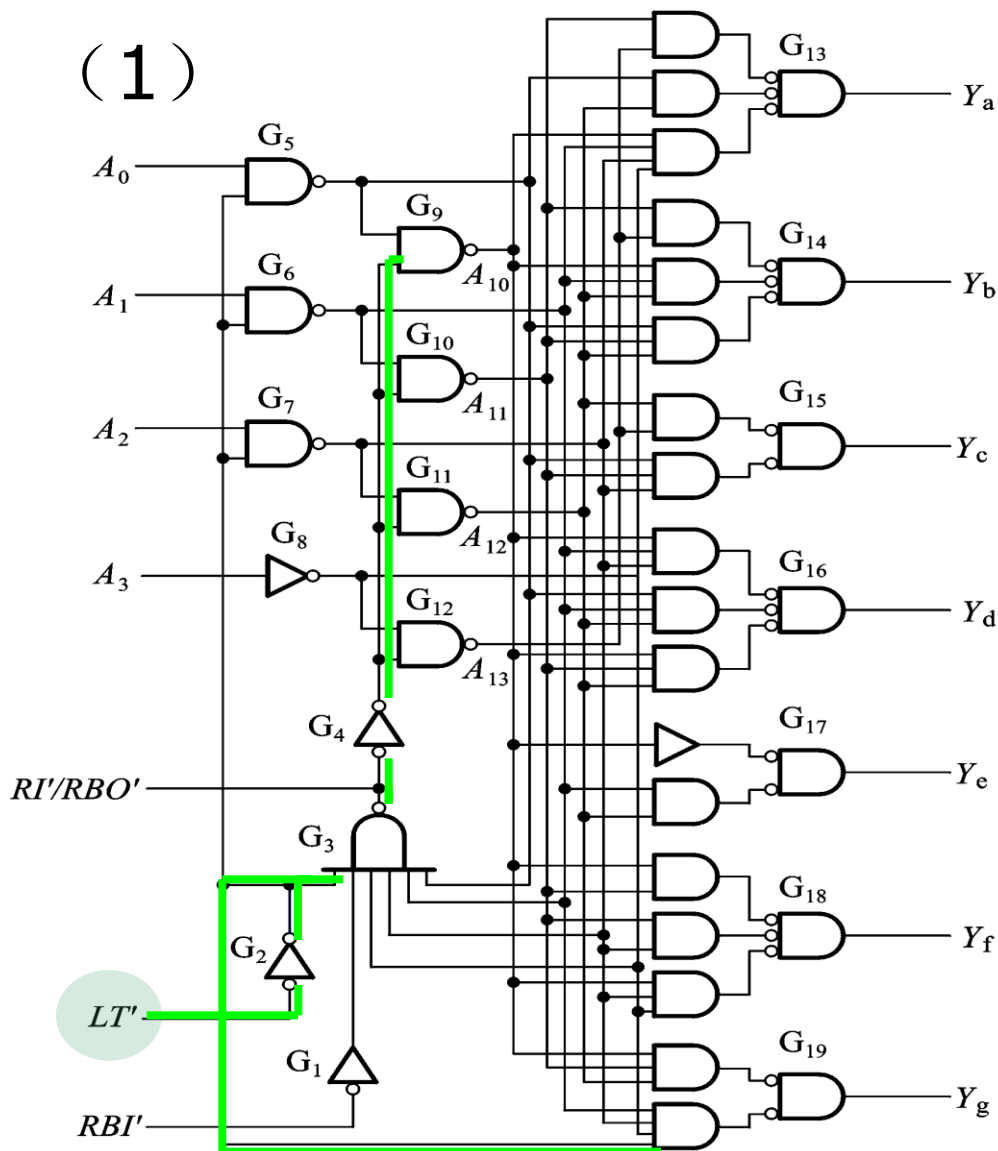


显示译码器附加控制端

7448的附加控制信号：(1)

❖ 灯测试输入 LT'

当 $T' = 0$ 时, $Y_a \sim Y_g$ 全部置为1

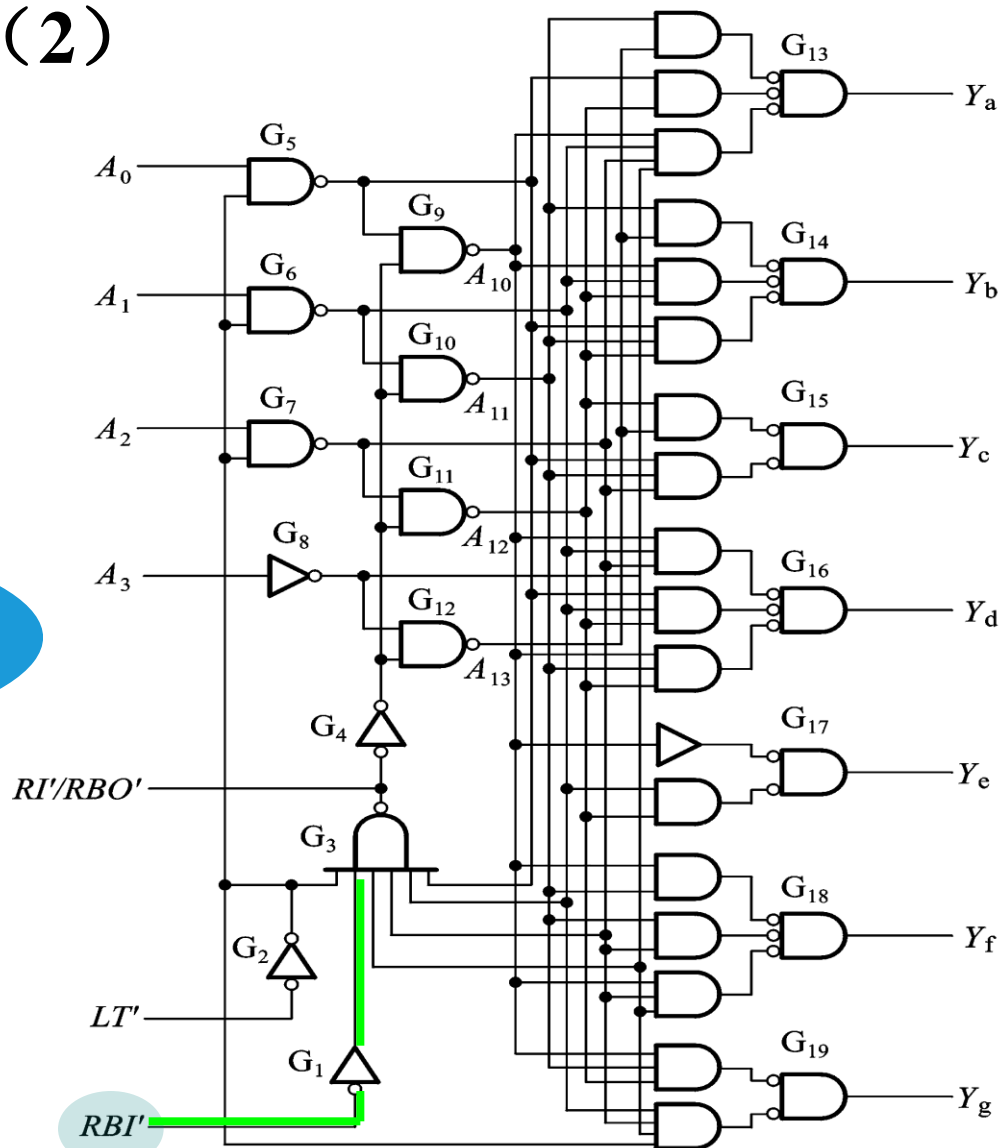


7448的附加控制信号：（2）

◆ 灭零输入 *RBI'*

把不希望显示的零熄灭

当 $A_3A_2A_1A_0 = 0000$ 时,
 $RBI' = 0$ 时, 则灭灯



7448的附加控制信号：（3）

❖ 灭灯输入/灭零输出 BI'/RBO'

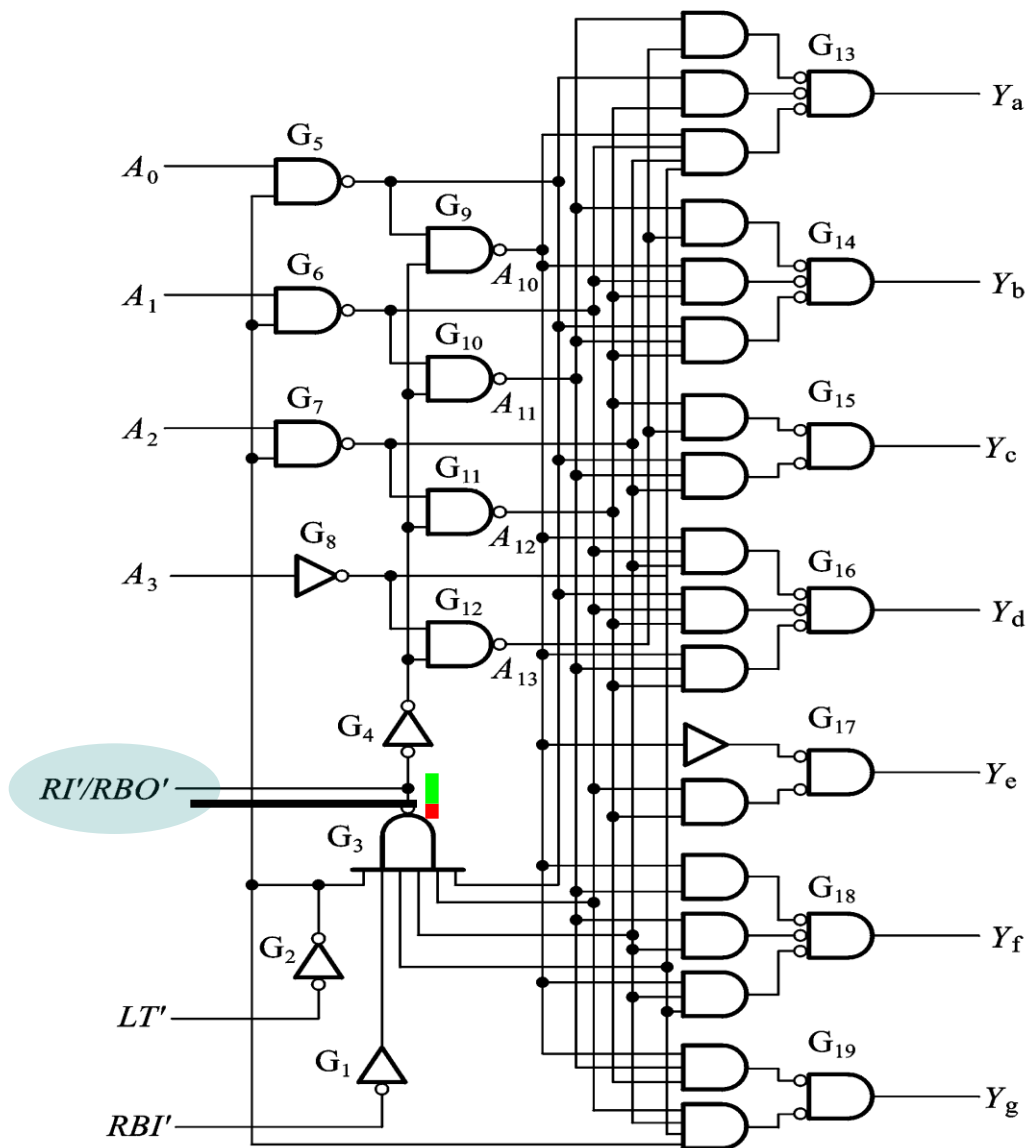
输入信号，称灭灯输入控制端：

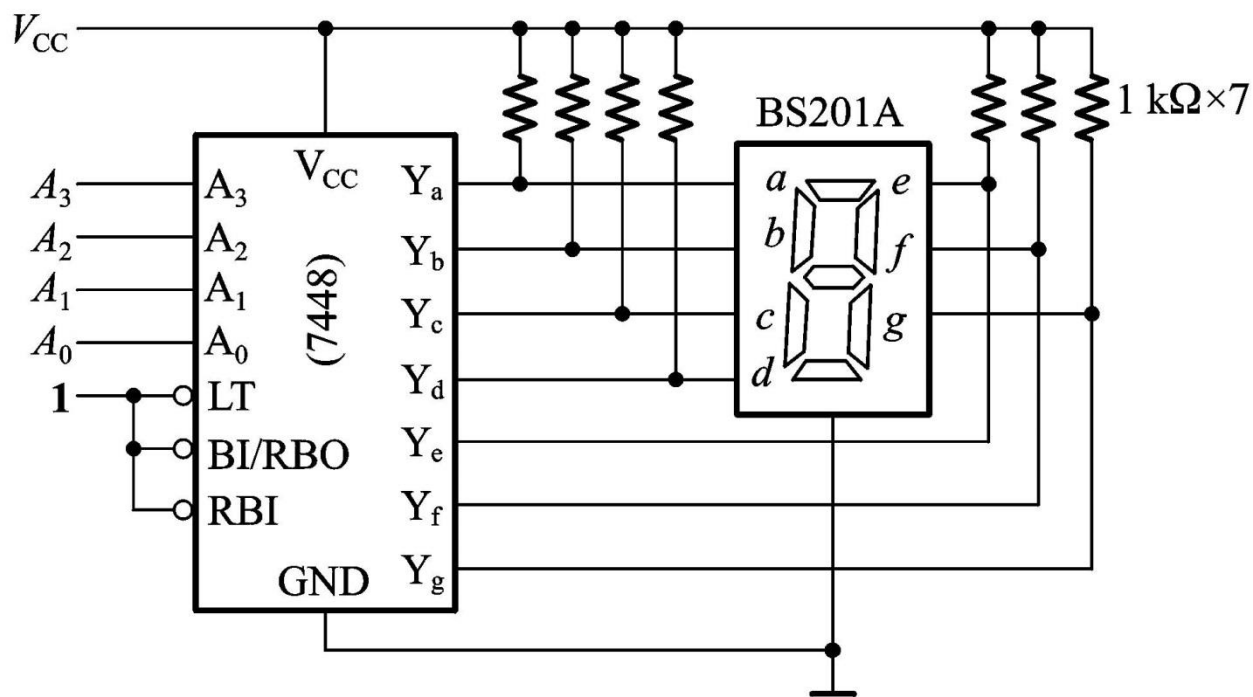
$BI' = 0$ 无论输入状态是什么，数码管熄灭

输出信号，称灭零输出端：

只有当输入 $A_3A_2A_1A_0 = 0$ ，且灭零输入信号 $RBI' = 0$ 时，
 RBO' 才给出低电平

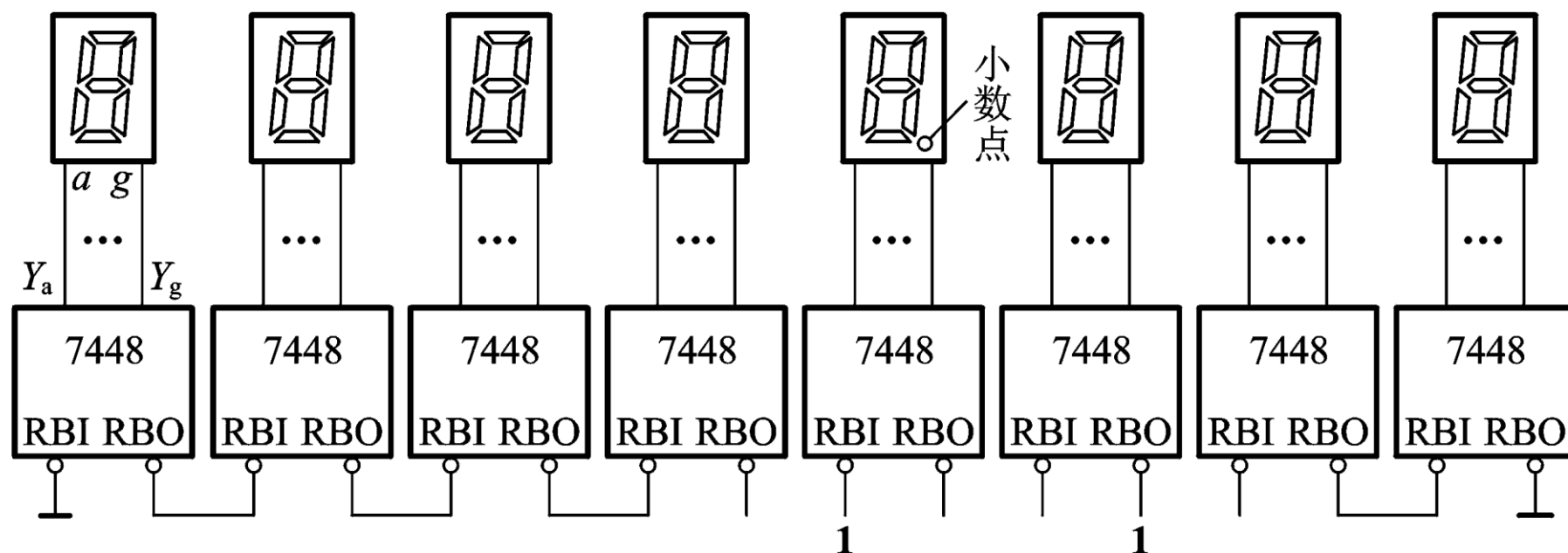
因此 $RBO' = 0$ 表示译码器将本来应该显示的零熄灭了





例：利用 RBI' 和 RBO 的配合，实现多位显示系统的灭零控制

- ❖ 整数部分：最高位是0，而且灭掉以后，输出 RBO' 作为次高位的 RBI' 输入信号
- ❖ 小数部分：最低位是0，而且灭掉以后，输出 RBO' 作为次低位的 RBI' 输入信号



译码器设计组合电路

四、用译码器设计组合逻辑电路

1. 基本原理

3位二进制译码器给出3变量的全部最小项;

。 。 。

n 位二进制译码器给出 n 变量的全部最小项;

任意函数

将 n 位二进制译码输出的最小项组合起来, 可获得任何形式的输入变量不大于 n 的组合函数

$$Y = \sum m_i$$

2. 举例

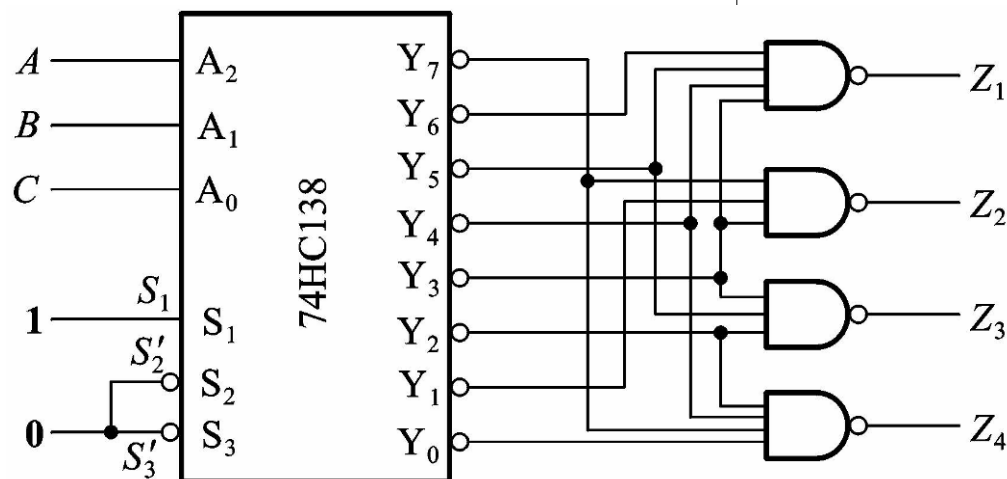
例：利用74HC138设计一个多输出的组合逻辑电路，输出逻辑函数式为：

$$Z_1 = AC' + A'BC + AB'C$$

$$Z_2 = BC + A'B'C$$

$$Z_3 = A'B + AB'C$$

$$Z_4 = A'BC' + B'C' + ABC$$



$$Z_1 = AC' + A'BC + AB'C = \sum m(3,4,5,6)$$

$$Z_1 = \sum m(3,4,5,6) = (m_3' m_4' m_5' m_6')'$$

$$Z_2 = BC + A'B'C = \sum m(1,3,7)$$

$$Z_2 = \sum m(1,3,7) = (m_1' m_3' m_7')'$$

$$Z_3 = A'B + AB'C = \sum m(2,3,5)$$

$$Z_3 = \sum m(2,3,5) = (m_2' m_3' m_5')'$$

$$Z_4 = A'BC' + B'C' + ABC = \sum m(0,2,4,7)$$

$$Z_4 = \sum m(0,2,4,7) = (m_0' m_2' m_4' m_7')'$$

思考：

利用74HC138设计一个多输出的组合逻辑电路，输出逻辑函数式为：

$$Y_1(ABC) = \sum_m (0,1,4,6)$$

$$Y_2(ABCD) = AB'C + ABD + A'B'D + BCD'$$

数据选择器

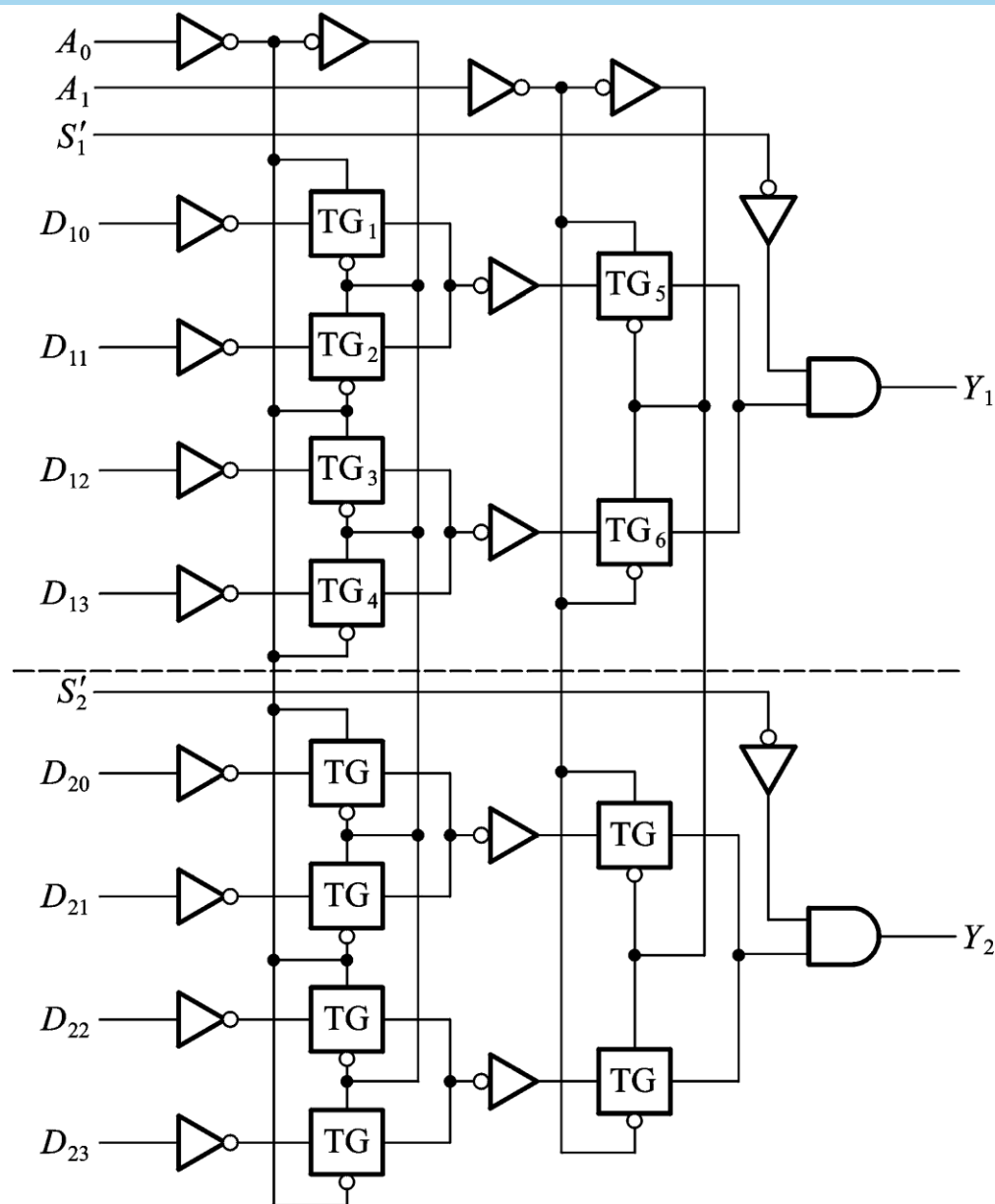
4.3.3 数据选择器

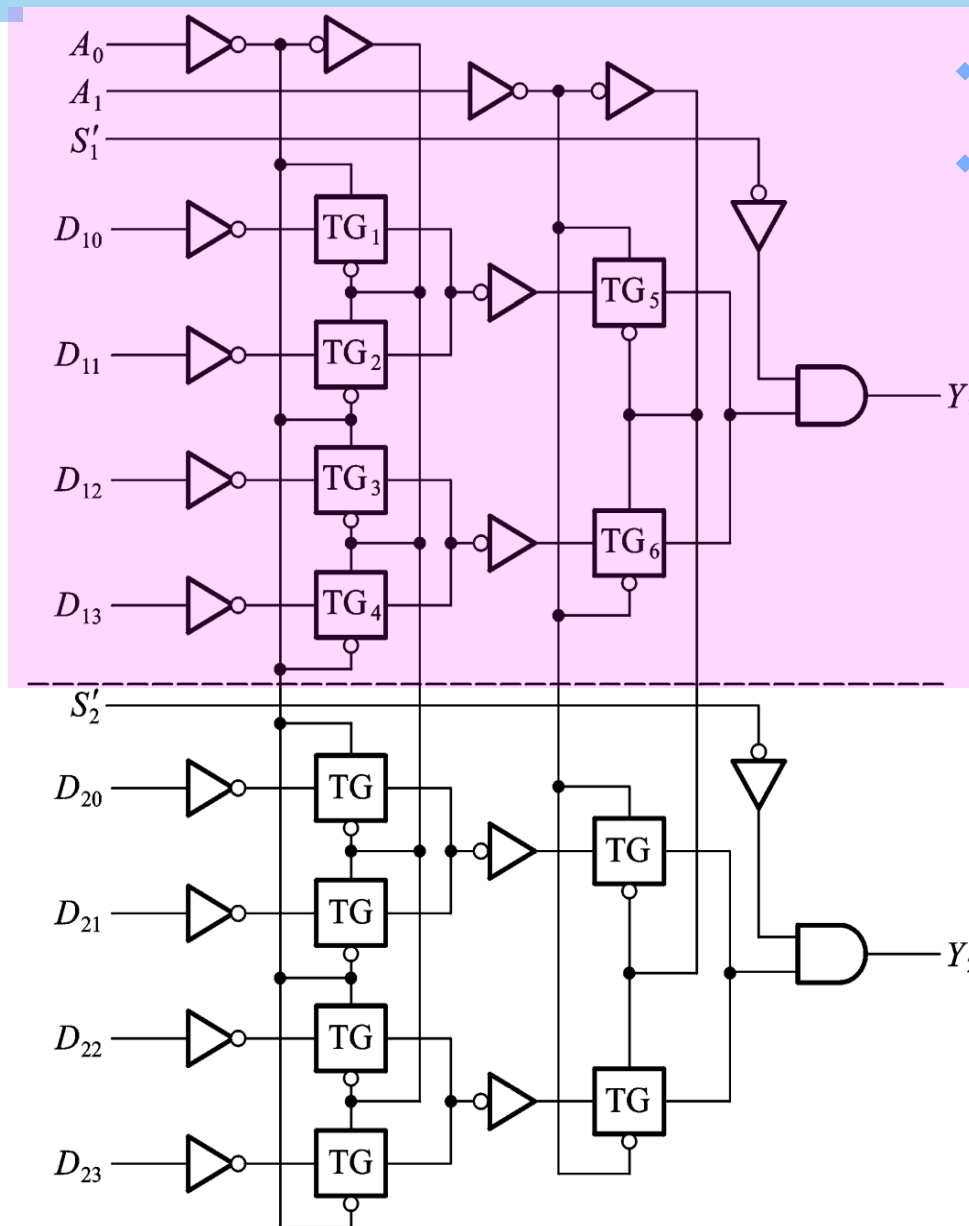
(多路开关)

一、工作原理

(从一组输入数据中选出某一个来)

❖ 例：“双四选一”，
74HC153，分析其中的一个“四选一”





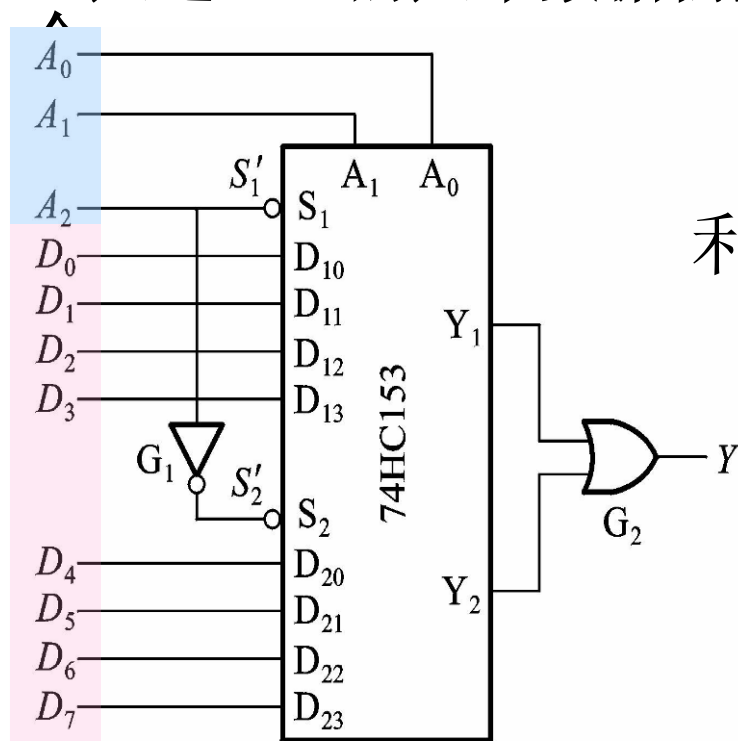
- ❖ 例：“双四选一”，74HC153
- ❖ 分析其中的一个“四选一”

$$Y_1 = S_1 [D_0(A_1'A_0) + D_1(A_1'A_0) + D_2(A_1A_0') + D_3(A_1A_0)]$$

S_1'	A_1	A_0	Y_1
1	X	X	0
0	0	0	D_{10}
0	0	1	D_{11}
0	1	0	D_{12}
0	1	1	D_{13}

例：用两个“四选一”接成“八选一”

- ❖ “四选一”只有2位地址输入，从四个输入中选中一个
- ❖ “八选一”的八个数据需要3位地址代码指定其中任何一



$$Y = (A_2' A_1' A_0') D_0 + (A_2' A_1' A_0) D_1 + (A_2' A_1 A_0') D_2 + (A_2' A_1 A_0) D_3 \\ + (A_2 A_1' A_0') D_4 + (A_2 A_1' A_0) D_5 + (A_2 A_1 A_0') D_6 + (A_2 A_1 A_0) D_7$$

数据选择器设计组合电路

二、用数据选择器设计组合电路

❖ 1. 基本原理

$$Y_1 = D_0(A_1'A_0') + D_1(A_1'A_0) + D_2(A_1A_0') + D_3(A_1A_0)$$

(三变量组合逻辑函数)



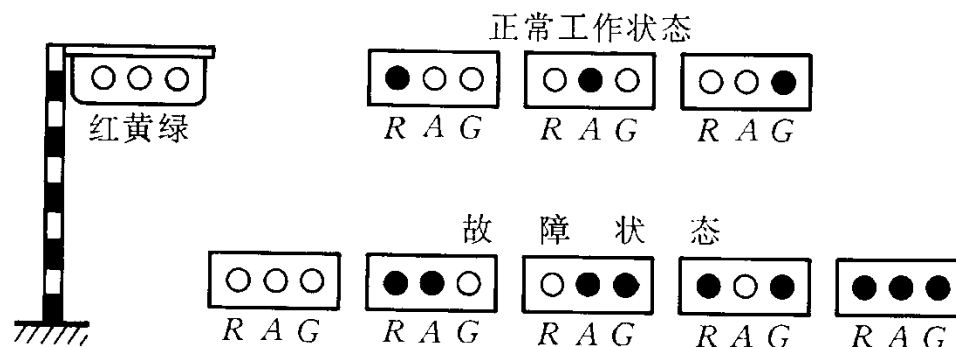
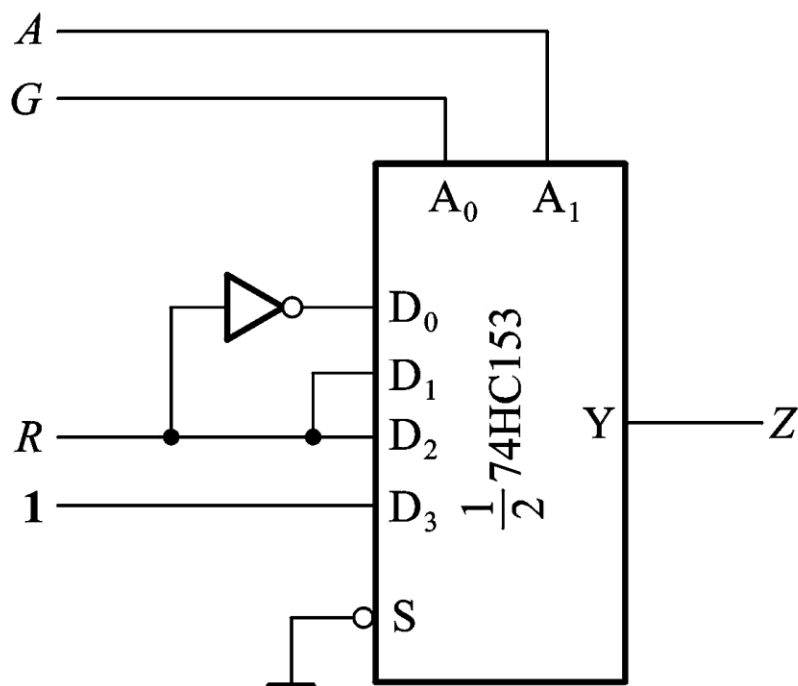
具有 n 位地址输入的数据选择器，可产生任何形式的输入变量不大于 $n+1$ 的组合函数

例如：

$$Z = R'A'G' + R'AG + RA'G + RAG' + RAG$$

$$= R'(A'G') + R(A'G) + R(AG') + 1 \cdot (AG)$$

$$Y_1 = S_1[D_0(A'_1A'_0) + D_1(A'_1A_0) + D_2(A_1A'_0) + D_3(A_1A_0)]$$



1位加法器

4.3.4 加法器

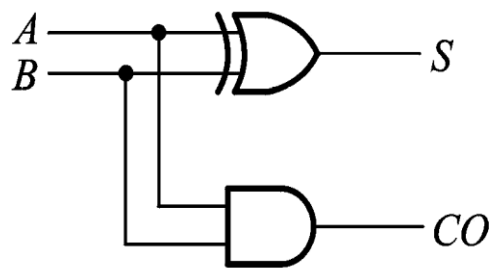
一、1位加法器

1. 半加器，不考虑来自低位的进位，将两个1位的二进制数相加

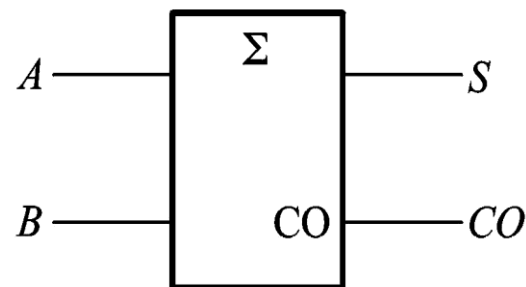
输 入		输 出	
A	B	S	CO
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$S = A \oplus B$$

$$CO = AB$$



(a)



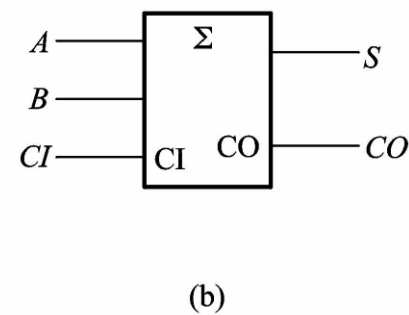
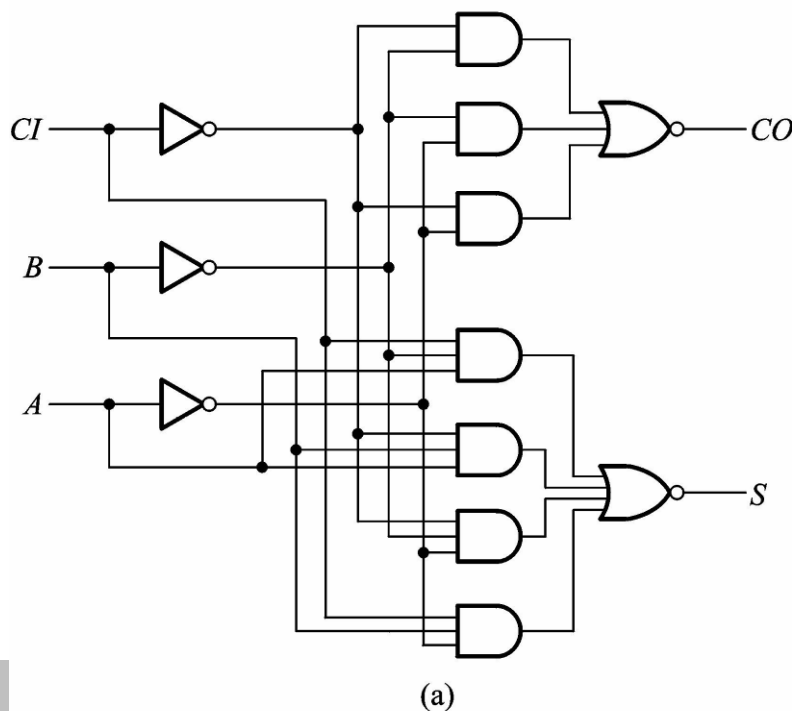
(b)

2. 全加器：将两个1位二进制数及来自低位的进位相加

输 入			输 出	
A	B	CI	S	CO
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

$$S = (A'B'CI' + A'B \cdot CI + AB'CI + ABCI')$$

$$CO = (A'B' + B'CI' + A'CI')$$



74LS183

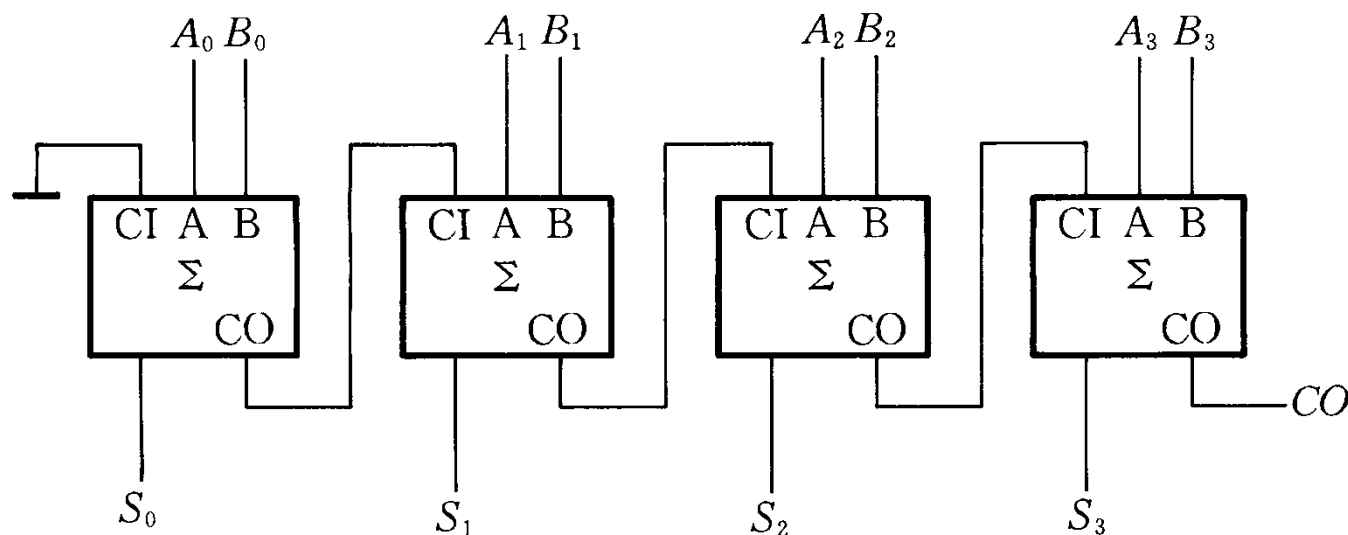
多位加法器

二、多位加法器

1. 串行进位加法器

优点：简单

缺点：慢



$$(CI)_i = (CO)_{i-1}$$

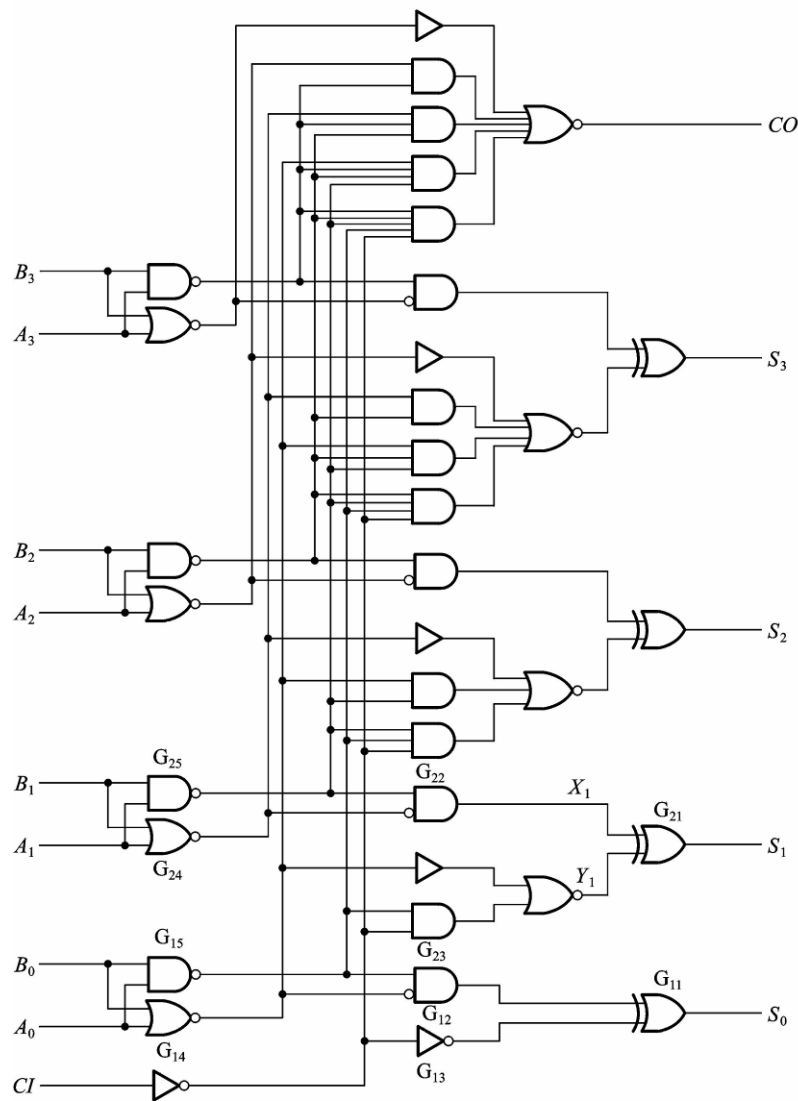
$$S_i = A_i \oplus B_i \oplus (CI)_i$$

$$(CO)_i = A_i B_i + (A_i + B_i)(CI)_i$$

2. 超前进位加法器

基本原理：加到第 i 位的进位输入信号是两个加数第 i 位以前各位（ $0 \sim j-1$ ）的函数，可在相加前由A,B两数确定。

优点：快，每1位的和及最后的进位基本同时产生。
缺点：电路复杂。



74LS283

$$i = 0: (CI)_0 = 0$$

$$S_0 = A_0 \oplus B_0 \oplus (CI)_0$$

$$(CO)_0 = A_0 B_0 + (A_0 + B_0)(CI)_0$$

$$i = 1: (CI)_1 = (CO)_0$$

$$S_1 = A_1 \oplus B_1 \oplus (CO)_0$$

$$= A_1 \oplus B_1 \oplus (A_0 B_0 + (A_0 + B_0)(CI)_0)$$

$$(CO)_1 = A_1 B_1 + (A_1 + B_1)(CO)_0$$

$$= A_1 B_1 + (A_1 + B_1)(A_0 B_0 + (A_0 + B_0)(CI)_0)$$

$$i = 2: (CI)_2 = (CO)_1$$

$$= A_1 B_1 + (A_1 + B_1)(A_0 B_0 + (A_0 + B_0)(CI)_0)$$

$$(CO)_2 = A_2 B_2 + (A_2 + B_2)(CI)_2$$

$$= A_2 B_2 + (A_2 + B_2)(A_1 B_1 + (A_1 + B_1)(A_0 B_0 + (A_0 + B_0)(CI)_0))$$

$$S_2 = A_2 \oplus B_2 \oplus (CI)_2$$

$$= A_2 \oplus B_2 \oplus (A_1 B_1 + (A_1 + B_1)(A_0 B_0 + (A_0 + B_0)(CI)_0))$$

⋮

进位输出

$$(CO)_i = A_i B_i + (A_i + B_i)(CI)_i$$

$$(CO)_i = G_i + P_i (CI)_i$$

$$(CO)_i = G_i + P_i G_{i-1} + \dots + P_i P_{i-1} \dots P_1 G_0 + P_i P_{i-1} \dots P_0 (CI)_0$$

相加的和

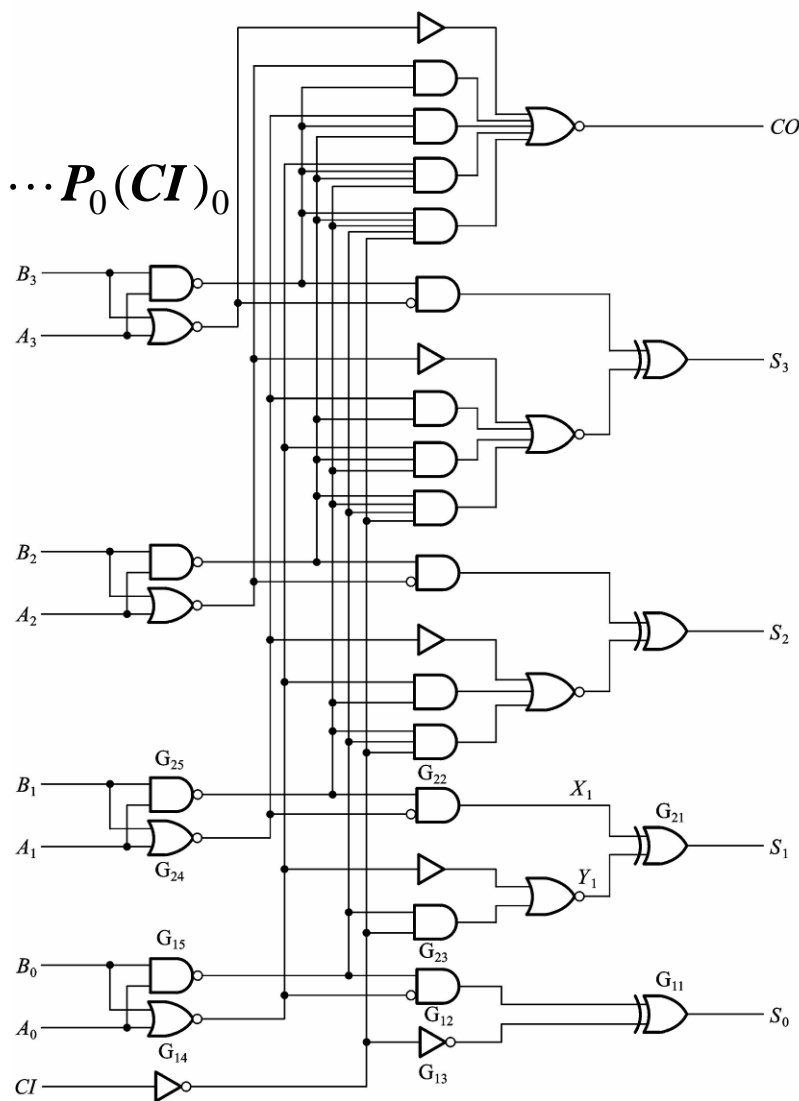
$$S_i = A_i \oplus B_i \oplus (CI)_i$$

举例 4位超前进位加法器(LS283)

$$X_1 = A_1 \oplus B_1$$

$$Y_1 = A_0 B_0 + (A_0 + B_0)(CI)_0 = (CO)_0 = (CI)_1$$

$$S_1 = X_1 \oplus Y_1 = A_1 \oplus B_1 \oplus (CI)_1$$



加法器设计组合电路

三、用加法器设计组合电路

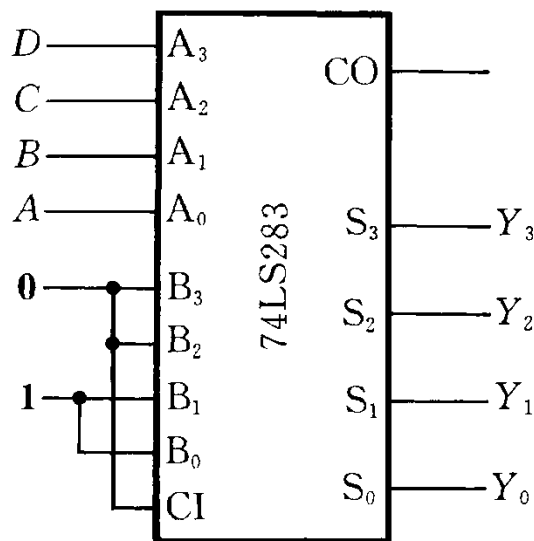
❖ 基本原理:

若能将函数变换成输入变量与输入变量相加

或能将函数变换成输入变量与常量相加

例：将BCD的8421码转换为余3码

$$Y_3Y_2Y_1Y_0 = DCBA + 0011$$



输 入				输 出			
D	C	B	A	Y3	Y2	Y1	Y0
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

数值比较器

4.3.5 数值比较器

❖ 用来比较两个二进制数的数值大小

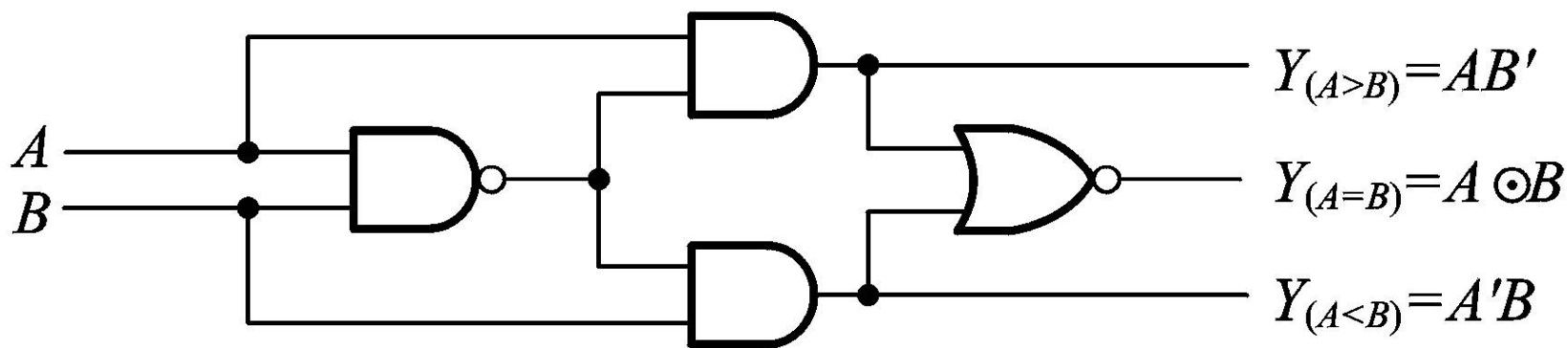
一、1位数值比较器

A,B比较有三种可能结果

* $A > B (A = 1, B = 0)$ 则 $AB' = 1, \therefore Y_{(A>B)} = AB'$

* $A < B (A = 0, B = 1)$ 则 $A'B = 1, \therefore Y_{(A<B)} = A'B$

* $A = B (A, B \text{ 同为 } 0 \text{ 或 } 1), \therefore Y_{(A=B)} = (A \oplus B)'$



二、多位数值比较器

1. 原理：从高位比起，只有高位相等，才比较下一位。

例如：

比较 $A_3A_2A_1A_0$ 和 $B_3B_2B_1B_0$

$$Y_{(A<B)} = A_3'B_3 + (A_3 \oplus B_3)' A_2'B_2 + (A_3 \oplus B_3)' (A_2 \oplus B_2)' A_1'B_1 \\ + (A_3 \oplus B_3)' (A_2 \oplus B_2)' (A_1 \oplus B_1)' A_0'B_0 + \\ (A_3 \oplus B_3)' (A_2 \oplus B_2)' (A_1 \oplus B_1)' (A_0 \otimes B_0)' I_{(A<B)}$$

$$Y_{(A=B)} = (A_3 \oplus B_3)' (A_2 \oplus B_2)' (A_1 \oplus B_1)' (A_0 \oplus B_0)' I_{(A=B)}$$

$$Y_{(A>B)} = (Y_{(A<B)} + Y_{(A=B)})' I_{(A>B)}$$

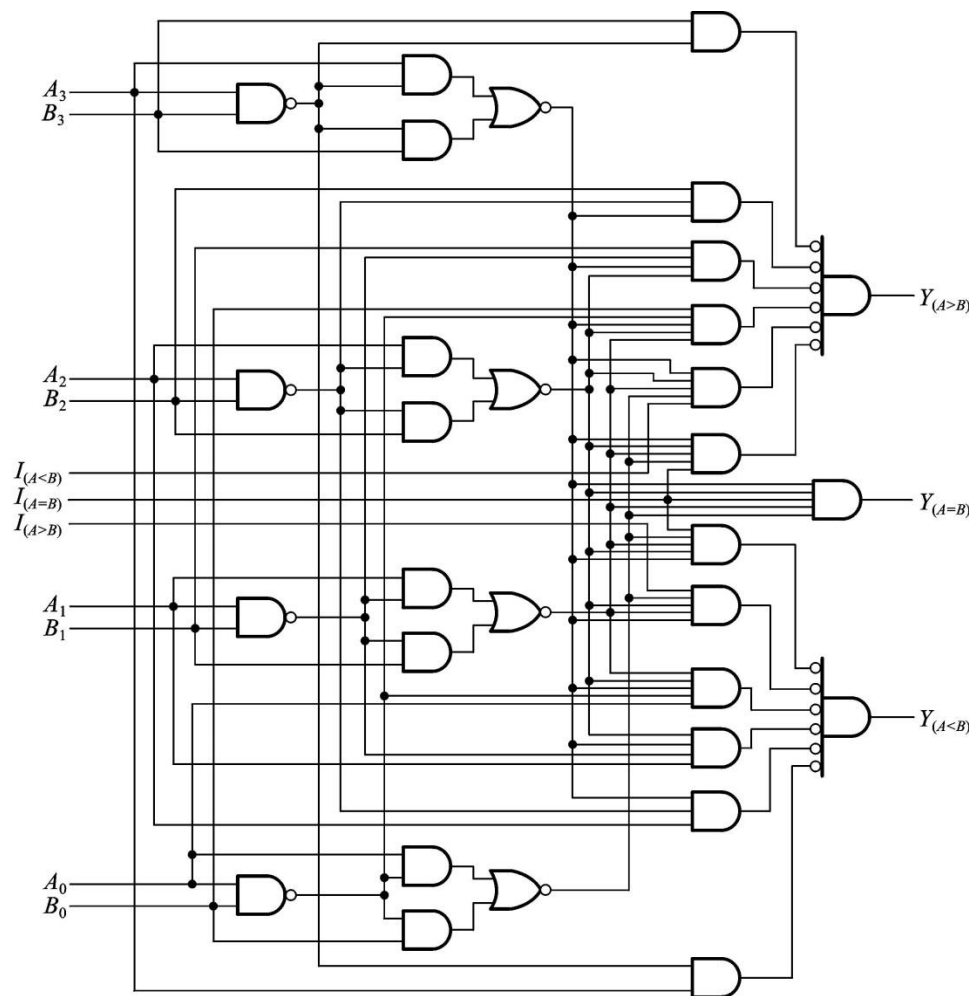
2. 集成电路74LS85 实现4位二进制数的比较

$I_{(A<B)}$, $I_{(A=B)}$ 和 $I_{(A>B)}$ 为附加端, 用于扩展

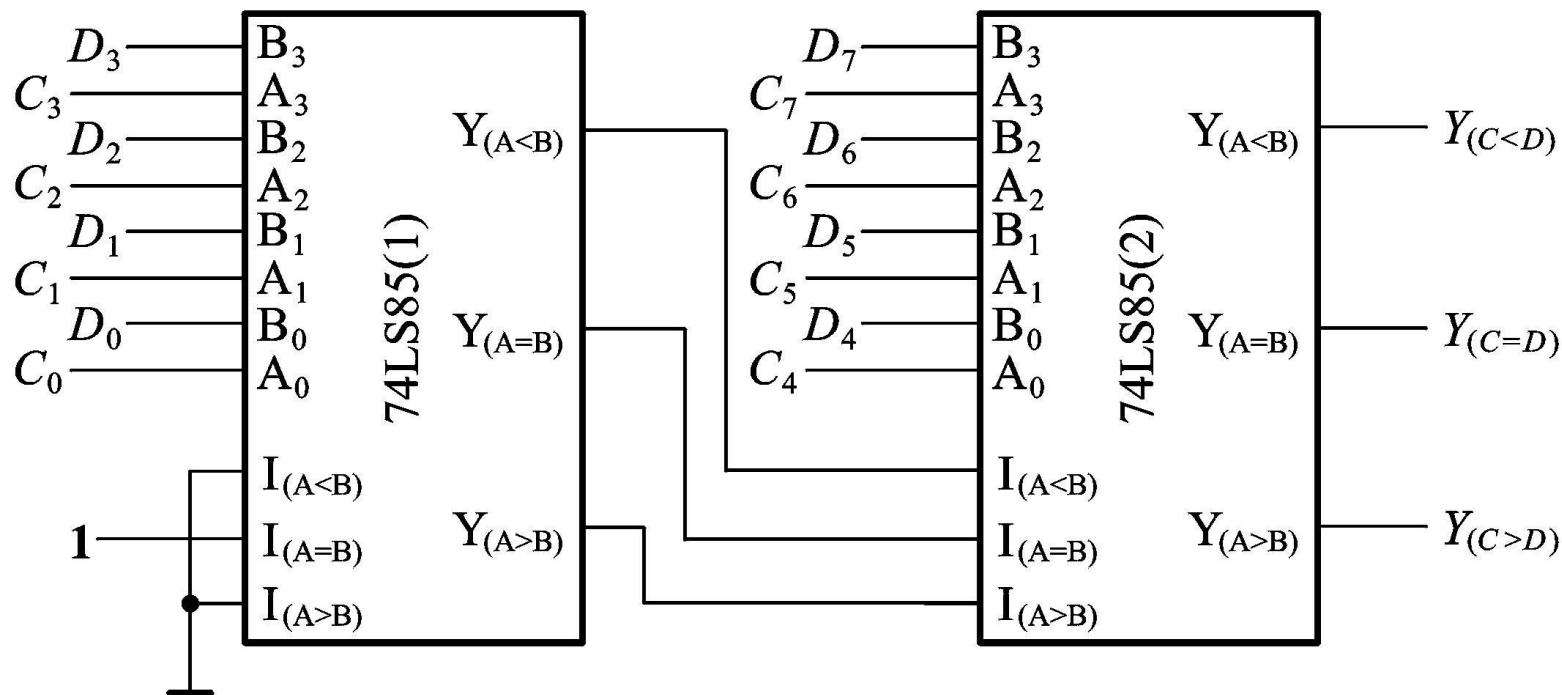
$I_{(A<B)}$, 来自低位的比较结果

$I_{(A=B)}$, 来自低位的比较结果

$I_{(A>B)}$, $A > B$ 输出允许信号



3. 比较两个8位二进制数的大小



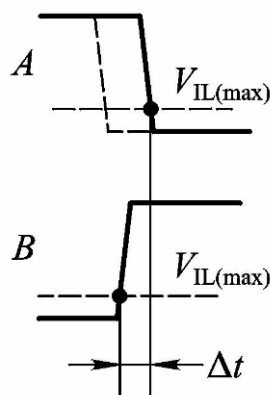
组合电路中的竞争-冒险现象

4.4 组合逻辑电路中的竞争-冒险现象

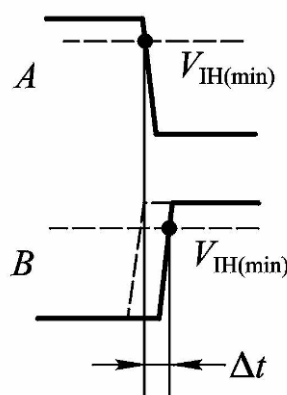
4.4.1 竞争-冒险现象及成因

一、什么是“竞争”

两个输入“同时向相反的逻辑电平变化”，称存在“竞争”



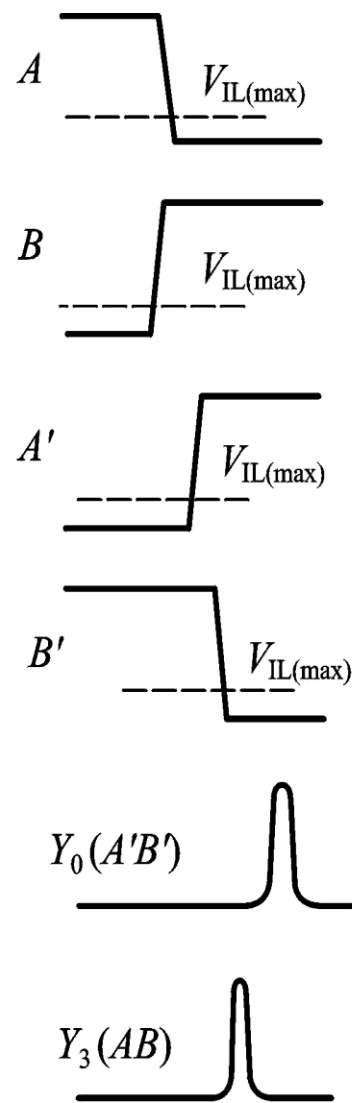
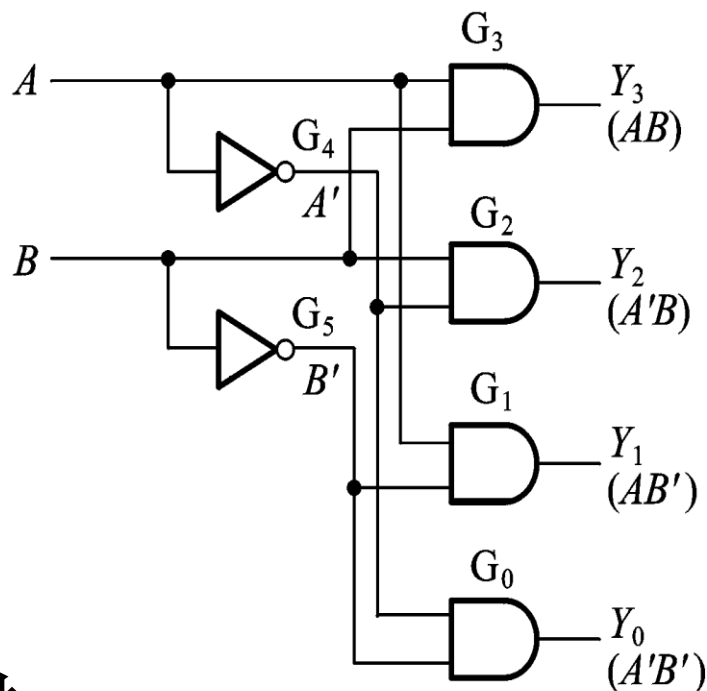
(a)



(b)

二、因“竞争”而可能在输出产生尖峰脉冲的现象，称为“竞争-冒险”。

三、2线—4线译码器中的竞争-冒险现象



当 AB 从 $10 \rightarrow 01$ 时，

在动态过程中可能出现

所以 Y_3 和 Y_0 输出端可能产生尖峰脉冲。

(a)

(b)

4.4.2 检查竞争-冒险现象的方法

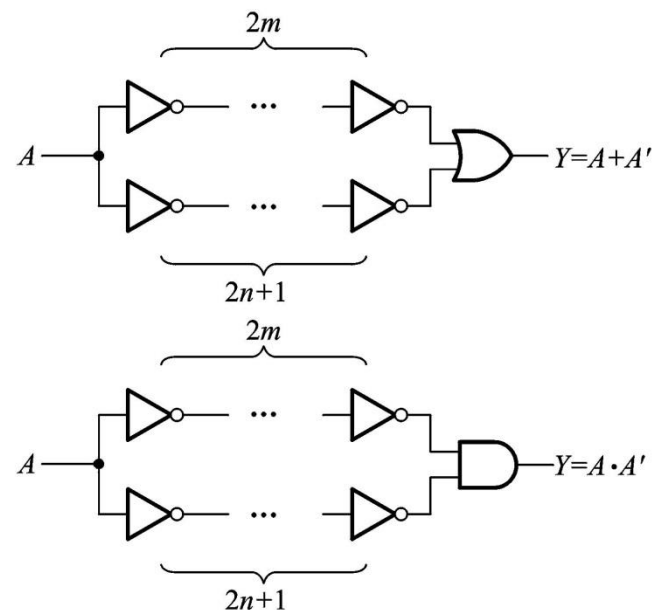
当输入变量每次只有一个改变状态的情况下：

如果输出端门电路的两个输入信号A和A'是输入变量A经过两个不同的传输途径而来的，那么当输入变量A的状态发生突变时输出端便有可能产生尖峰脉冲。

判断准则：只要输出端的逻辑函数在一定条件下能简化成

$$Y = A + A' \quad \text{或} \quad Y = A \cdot A'$$

则可判定存在竞争-冒险现象。



消除竞争-冒险现象的方法

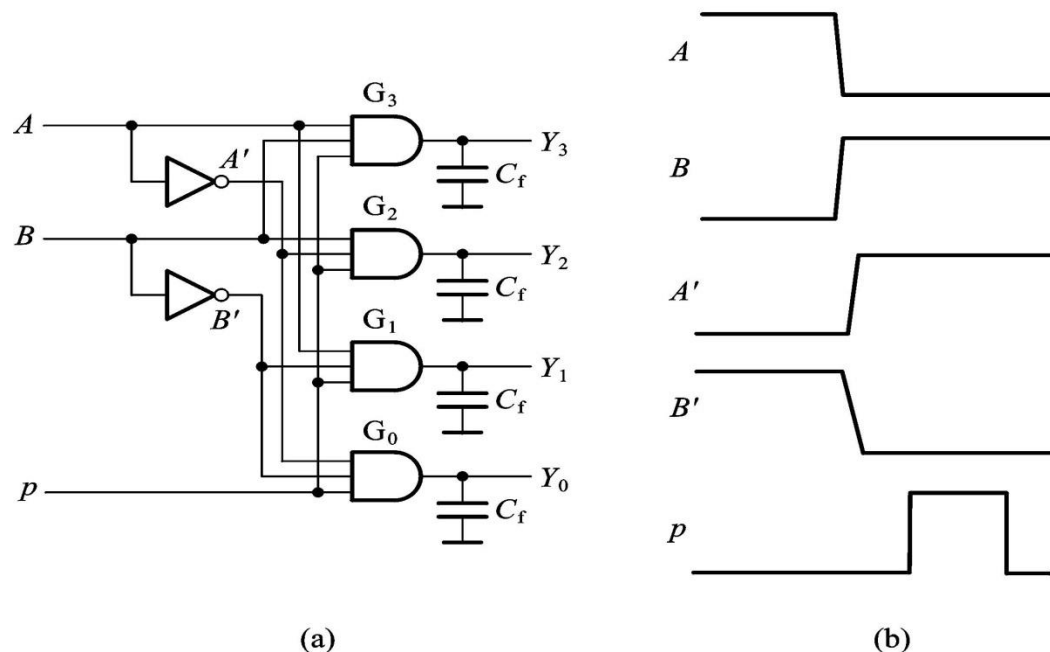
4.4.3 消除竞争-冒险现象的方法

一、接入滤波电容

尖峰脉冲很窄，用很小的电容就可将尖峰削弱到 V_{TH} 以下。

二、引入选通脉冲

利用选通脉冲，在电路达到稳定之后， P 的高电平期的输出信号不会出现尖峰。



三、修改逻辑设计

例: $Y = AB + A'C$

在 $B = C = 1$ 的条件下, $Y = A + A' \Rightarrow$ 稳态下 $Y = 1$

当 A 改变状态时存在竞争-冒险

$$Y = AB + A'C + BC$$

