



浙江大学  
ZHEJIANG UNIVERSITY

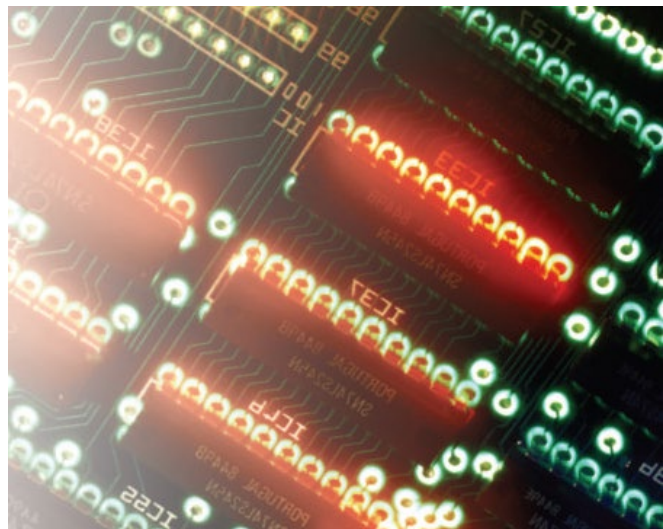
# 组合逻辑2

## Combinational Logic II

刘 鹏

浙江大学信息与电子工程学院

Email: liupeng@zju.edu.cn



# 复习

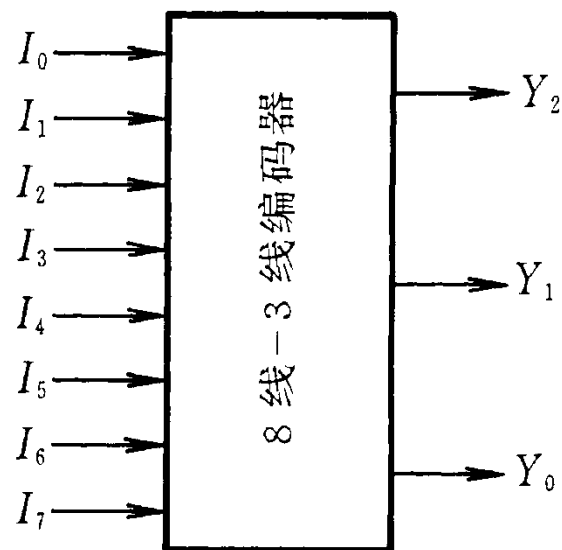
- ❑ 组合电路的基本概念
- ❑ 组合电路的设计方法
- ❑ 组合电路的模块设计
  - 优先编码器 (**Priority Encoder**)
  - 译码器 (**Decoder**)

# 本节内容

- ❑ 选择器(Multiplexer)
- ❑ 加法器(Adder)
- ❑ 比较器(Comparator)
- ❑ 采用模块组件实现组合电路

# 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



| 输 入            |                |                |                |                |                |                |                | 输 出            |                |                |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| I <sub>0</sub> | I <sub>1</sub> | I <sub>2</sub> | I <sub>3</sub> | I <sub>4</sub> | I <sub>5</sub> | I <sub>6</sub> | I <sub>7</sub> | Y <sub>2</sub> | Y <sub>1</sub> | Y <sub>0</sub> |
| 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              |

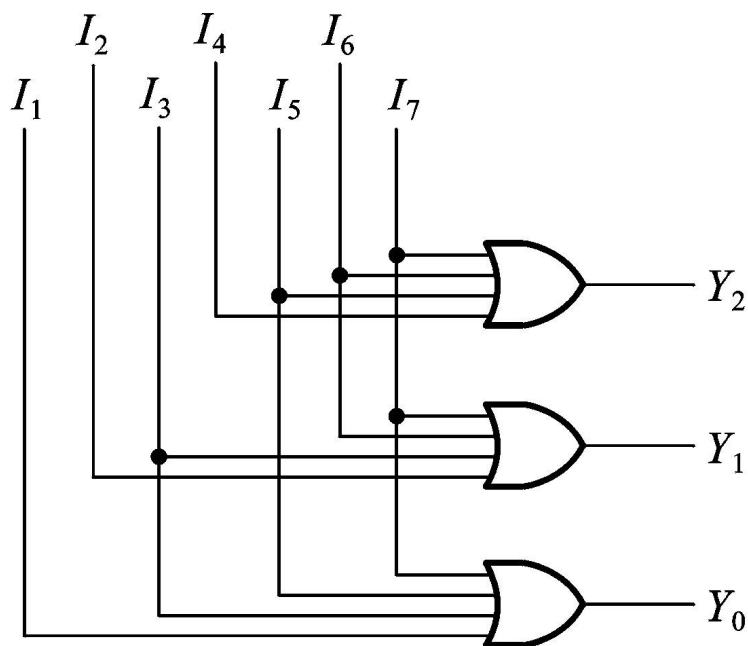
$$\begin{aligned}
 Y_2 = & I_7' I_6' I_5' I_4' I_3' I_2' I_1' I_0' + I_7' I_6' I_5' I_4' I_3' I_2' I_1' I_0 \\
 & + I_7' I_6' I_5' I_4' I_3' I_2' I_1' I_0' + I_7' I_6' I_5' I_4' I_3' I_2' I_1' I_0'
 \end{aligned}$$

## 利用无关项化简

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

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

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



任何时候只有一个输入时激活的，或有两个输入同时激活，则输入就会产生一个没有定义的组合。对于这个不确定因素，编码器必须建立优先机制，使得只有一个输出被编码

# 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     |

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

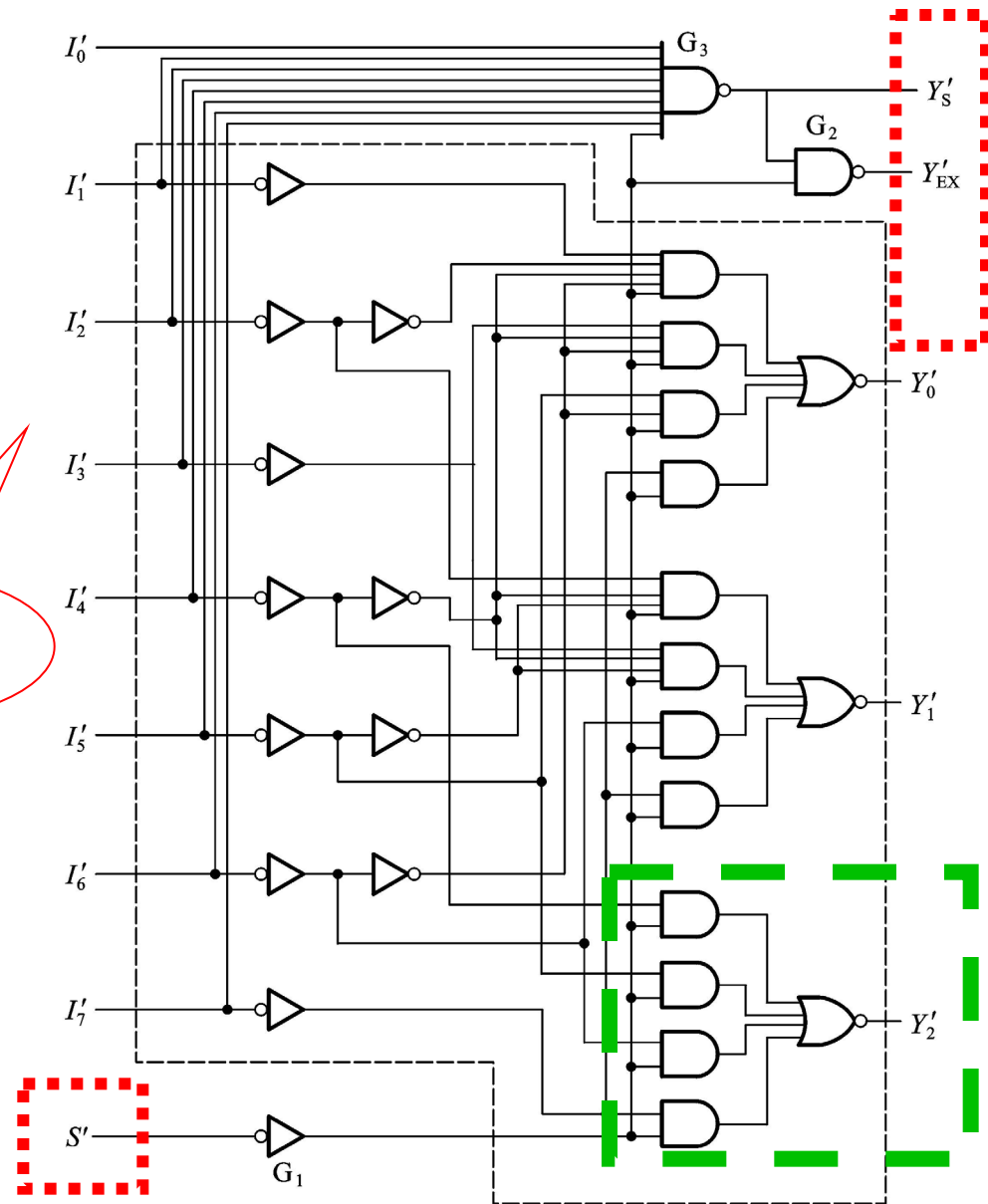


$$A + A'B = A + B$$

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

# 实例： 74HC148

低电平



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

选通信号

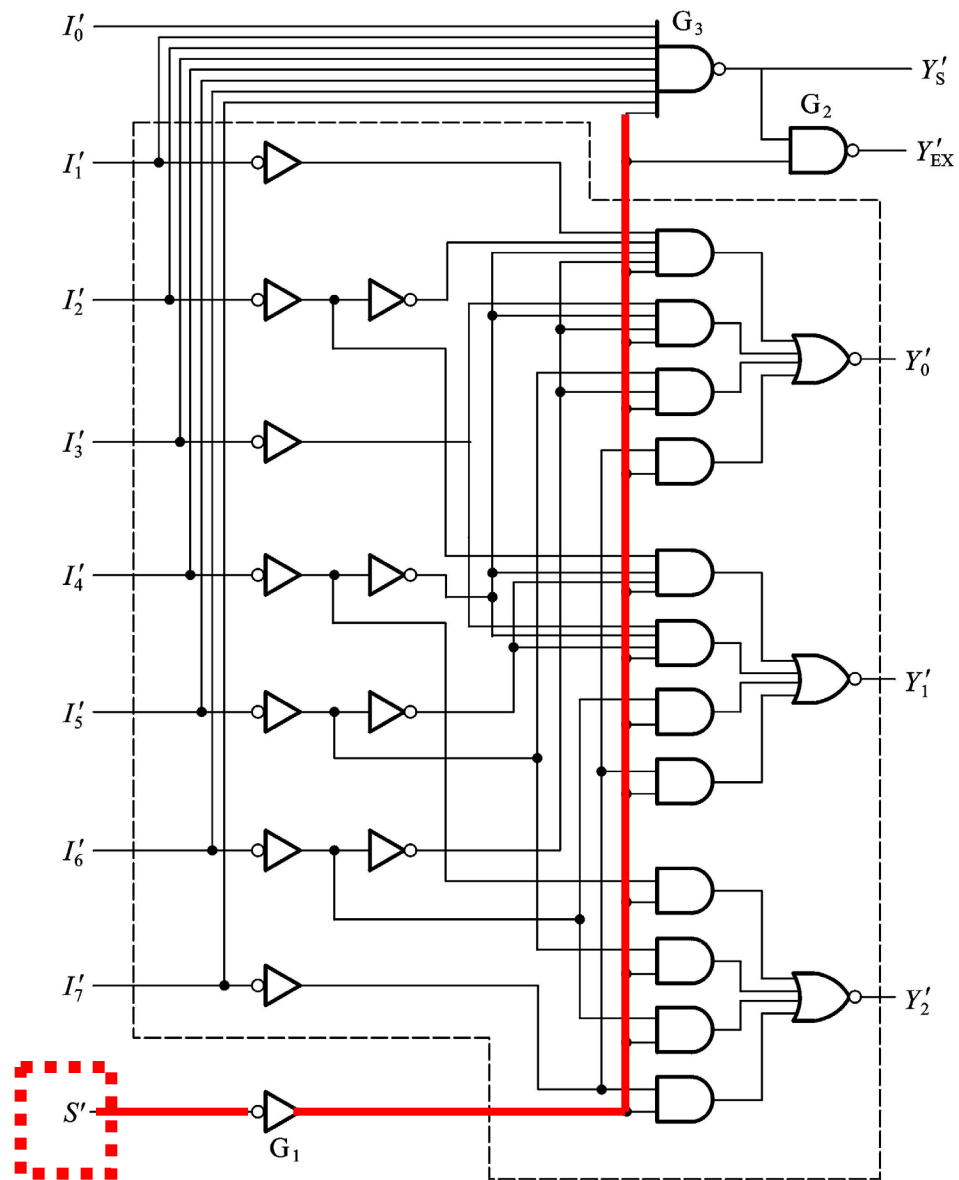
$$Y_2' = [(I_7 + I_6 + I_5 + I_4)S']$$

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

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

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

选通信号



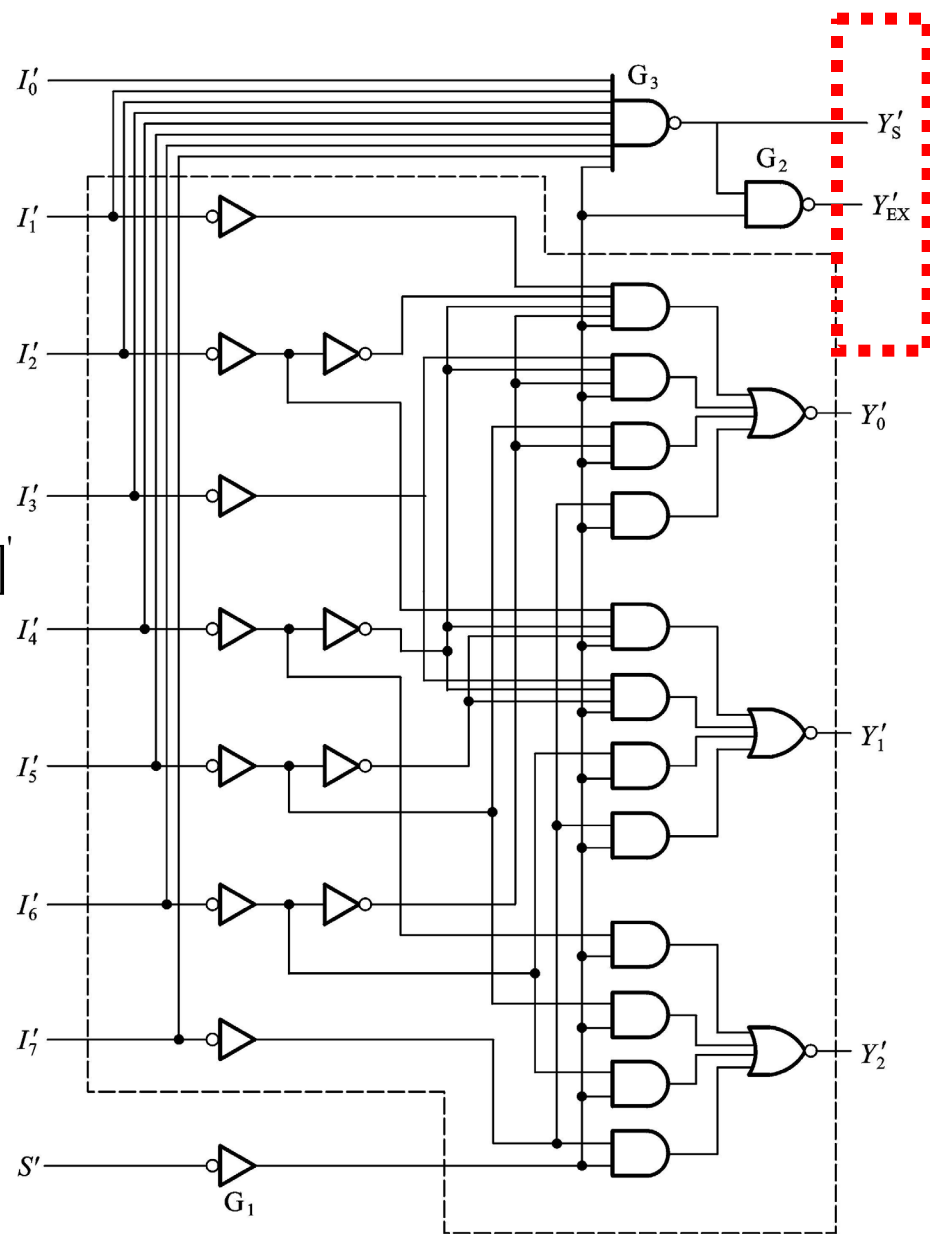
为0时，电路工作  
无编码输入

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

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

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

为0时，电路工  
作有编码输入



附加输出信号



| 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         |

# 74148 encoder 附加输出信号的状态及含义

| $Y'_S$ | $Y'_{EX}$ | 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    |

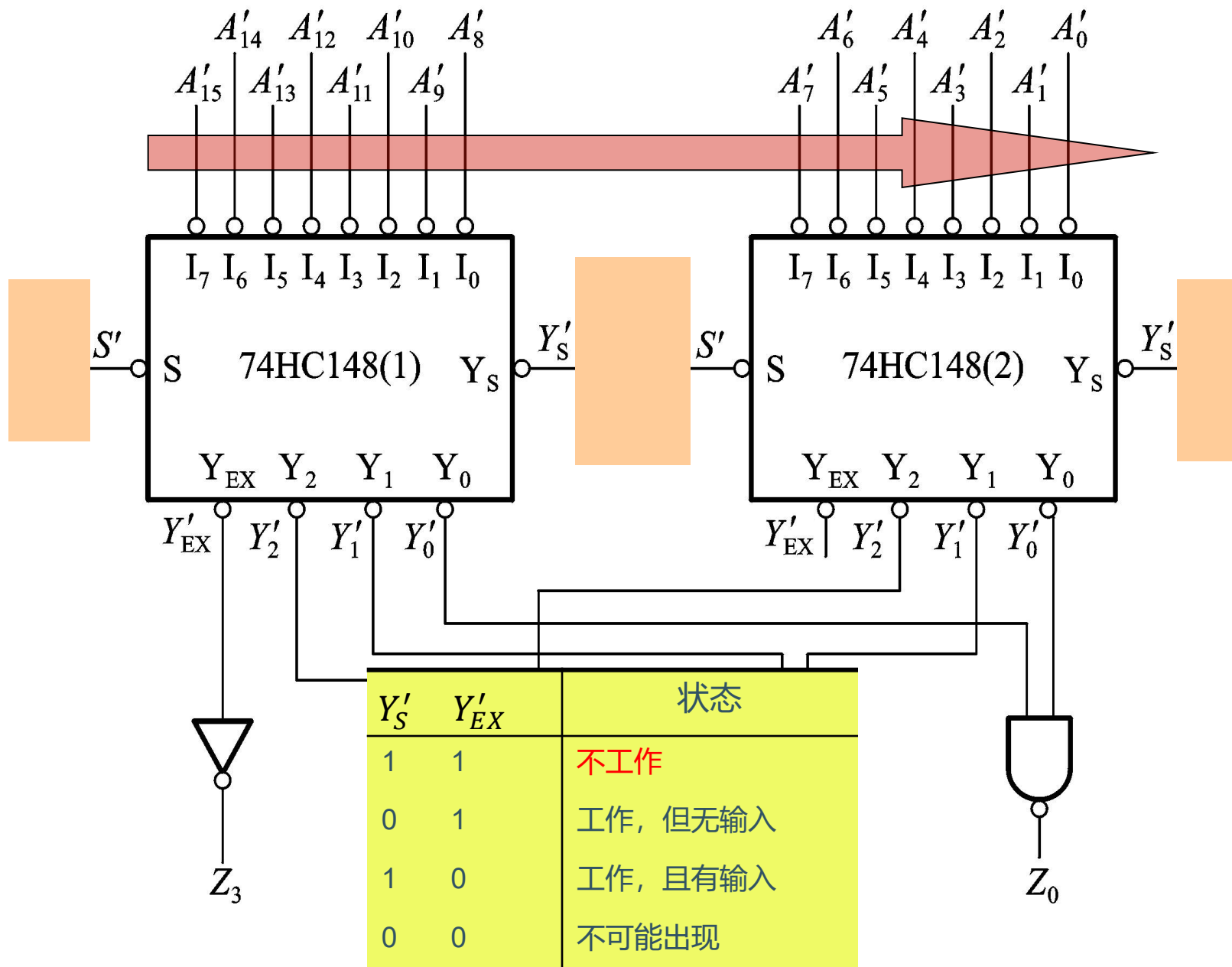
# 控制端扩展功能举例

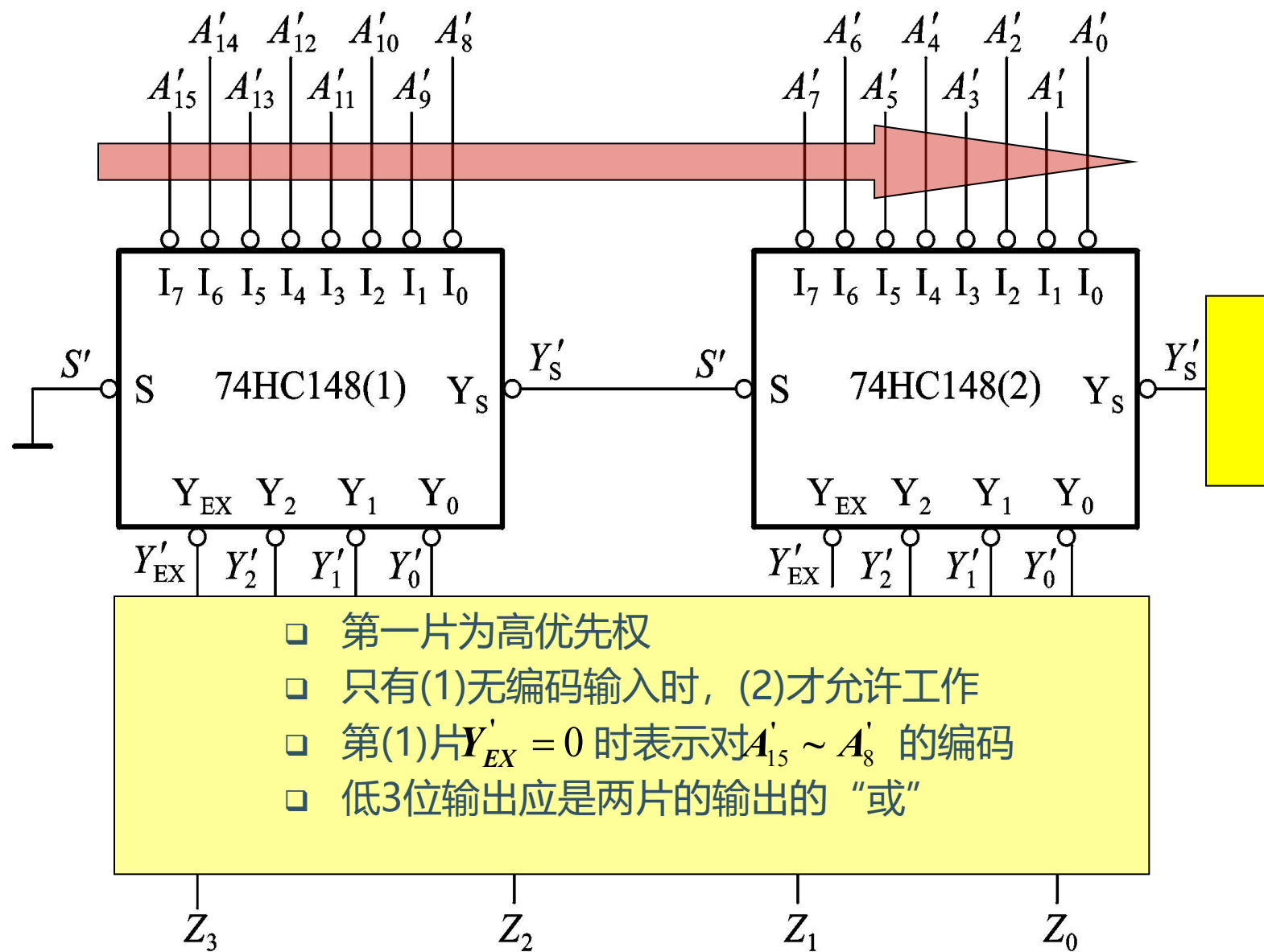
- 例：用两片8线-3线优先编码器74148



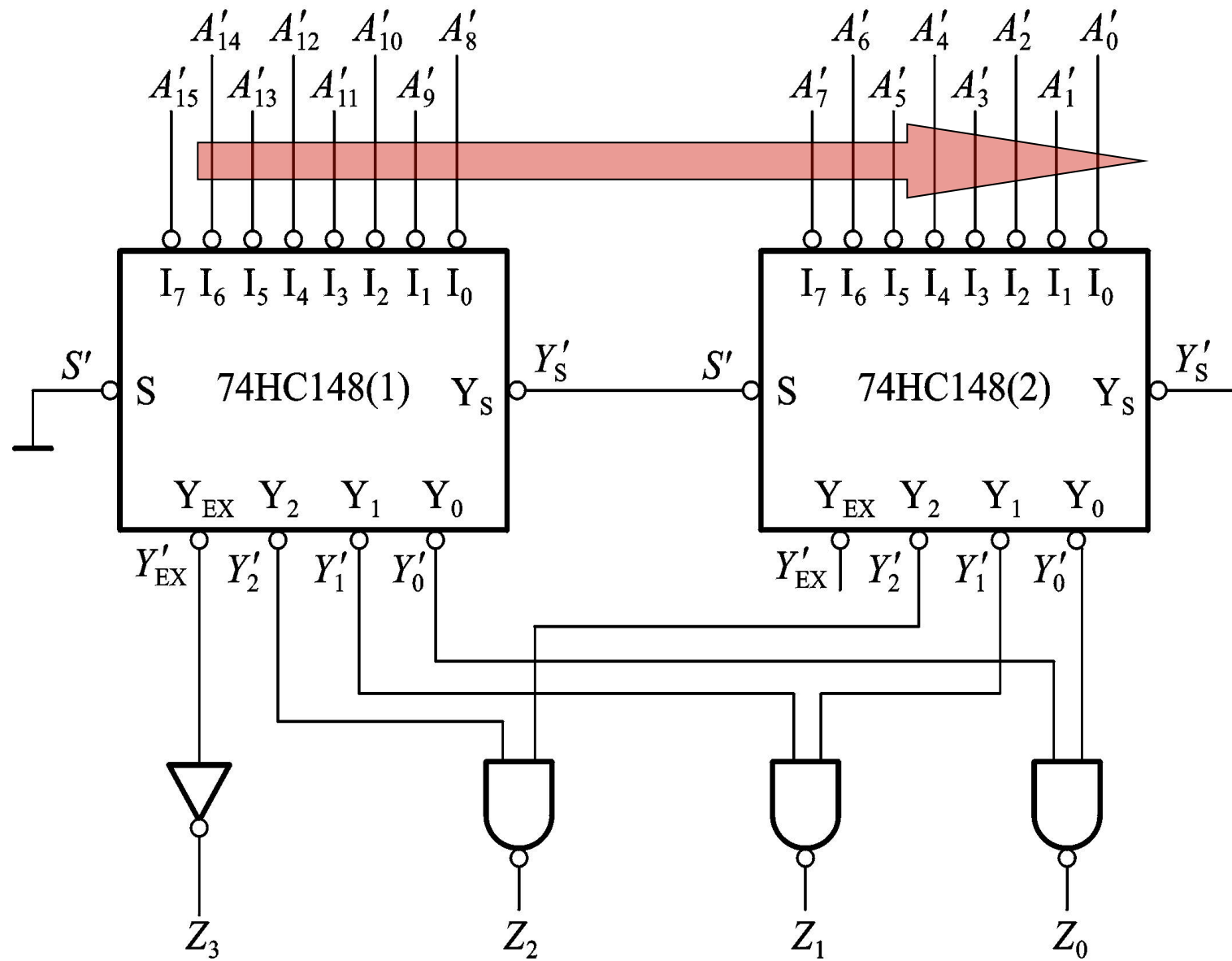
16线-4线优先编码器

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”

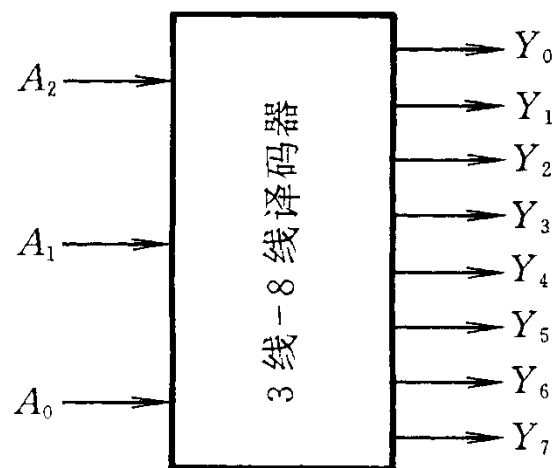


# 译码器

- 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
- 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

| 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     |

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

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

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

...

$$Y_7 = A_2 A_1 A_0 = m_7$$

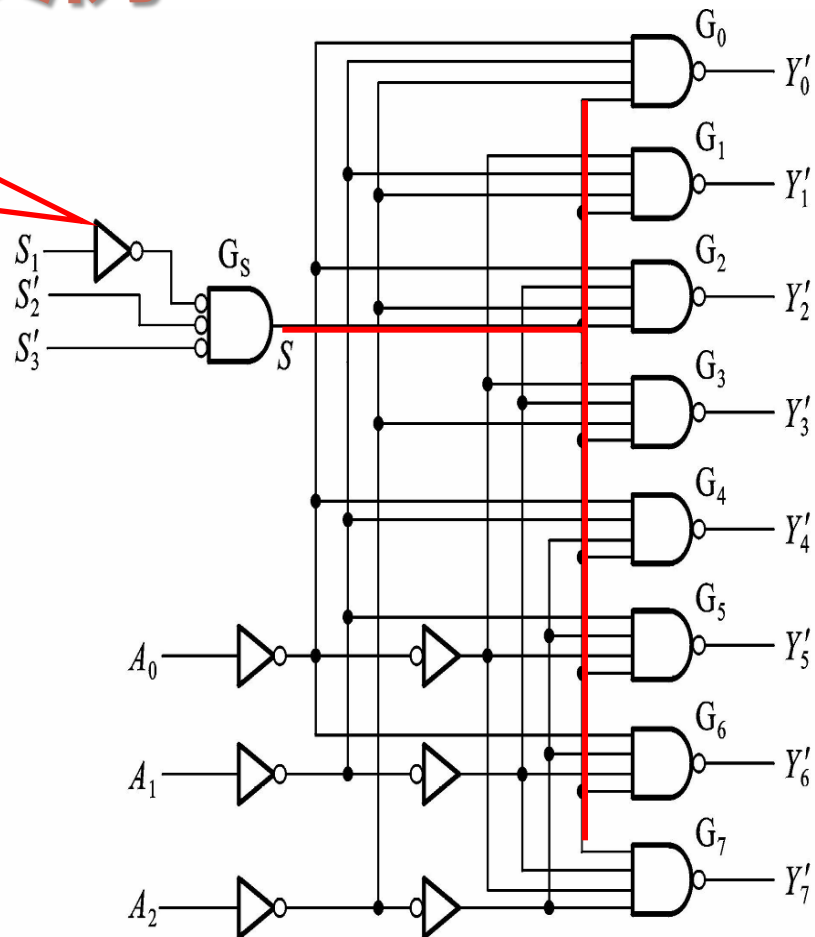


# 译码器Decoder实例：74HC138

$$S = S_3 S_2 S_1$$

$$Y'_i = (S \ m_i)'$$

附加  
控制端



低电平  
输出

# 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      | 0      | 1      | 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      |

# 用译码器设计组合逻辑电路

## □ 基本原理

- 一个译码器提供 $n$ 个输入变量的 $2^n$ 个最小项，译码器的输出由每一组输入唯一确定
- **任何布尔函数可以表示成最小项之和**
- 任何组合电路由 $n$ 个输入， $m$ 个输出可用 **$n$ -to- $2^n$** 译码器和 **$m$ 个或门**实现

# 译码器设计组合电路例子

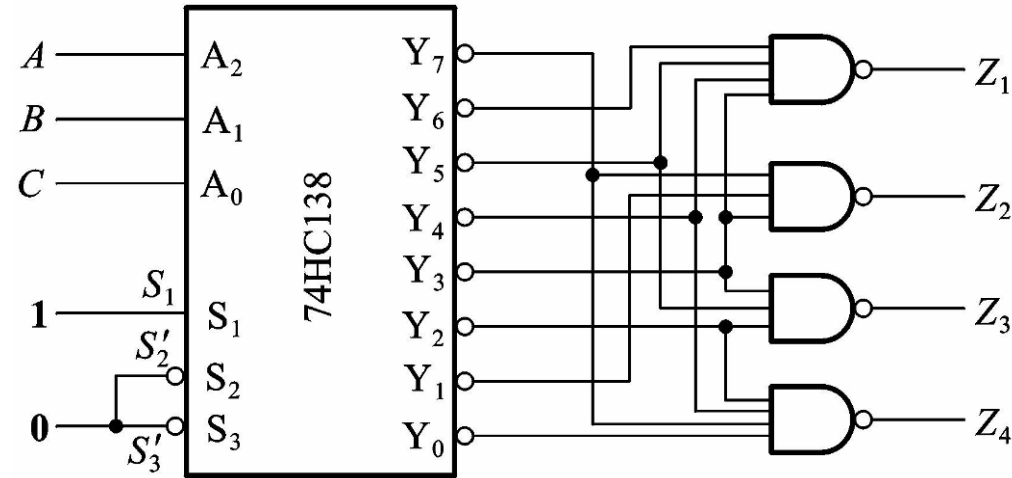
例：利用74HC138设计一个多输出的组合逻辑电路，输出逻辑函数式为：

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

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

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

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



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

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

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

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

$$Z_1 = \sum m(3,4,5,6) = (m'_3 m'_4 m'_5 m'_6)'$$

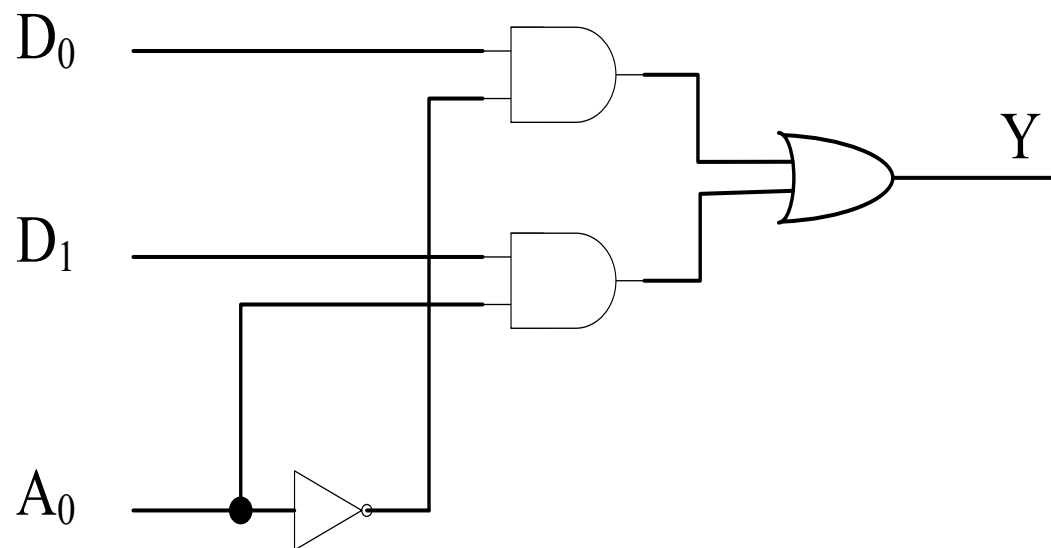
$$Z_2 = \sum m(1,3,7) = (m'_1 m'_3 m'_7)'$$

$$Z_3 = \sum m(2,3,5) = (m'_2 m'_3 m'_5)'$$

$$Z_4 = \sum m(0,2,4,7) = (m'_0 m'_2 m'_4 m'_7)'$$

# 数据选择器 (Multiplexers)

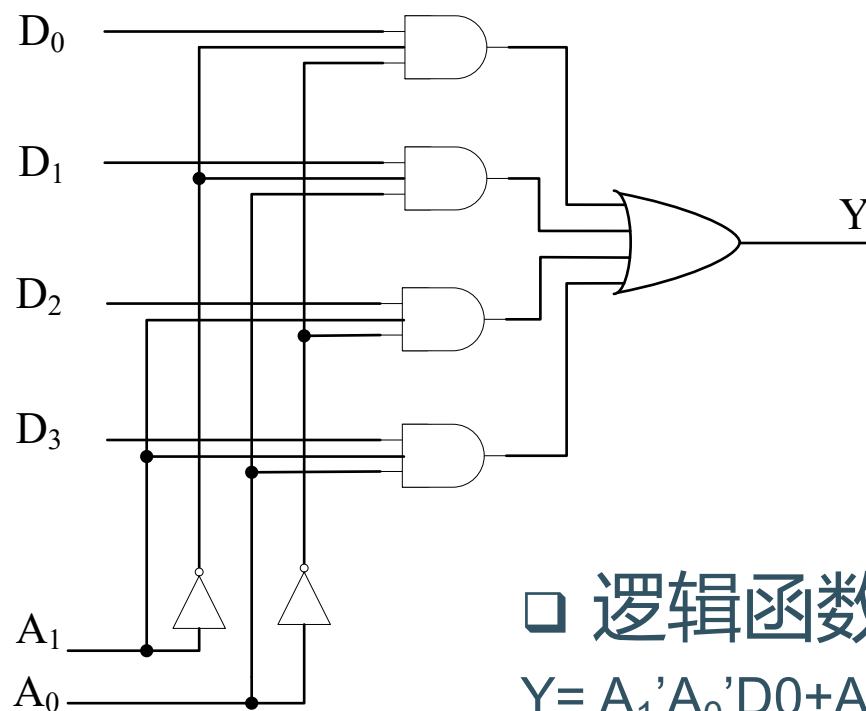
- ❑ 数据选择器是从多路输入线中选择其中的一路到输出线的一种组合电路
- ❑ 二选一数据选择器：
  - 数据输入线  $D_0$ - $D_1$
  - 选择线  $A_0$
  - 输出线  $Y$
- ❑ 电路图
- ❑ 表达式:  $Y = A'_0 D_0 + A_0 D_1$



# 4选1 选择器 (mux4to1)

□ 四选一数据选择器逻辑图

□ 功能表



| A1 | A0 | Y     |
|----|----|-------|
| 0  | 0  | $D_0$ |
| 0  | 1  | $D_1$ |
| 1  | 0  | $D_2$ |
| 1  | 1  | $D_3$ |

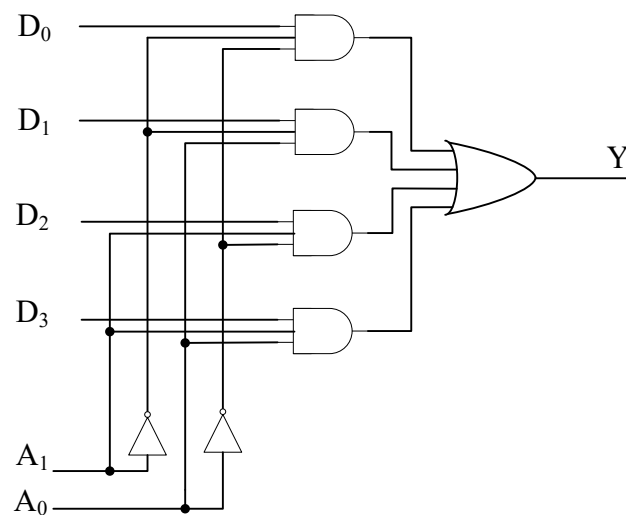
□ 逻辑函数式

$$Y = A_1'A_0'D_0 + A_1'A_0D_1 + A_1A_0'D_2 + A_1A_0D_3$$

# 采用数据选择器设计组合电路

## □ 基本原理

- $Y = D_0 A_1' A_0' + D_1 A_1' A_0 + D_2 A_1 A_0' + D_3 A_1 A_0$
- 具有n-1位地址输入的数据选择器，可实现n个变量布尔函数
- 数据选择器就是一个带或（OR）门的译码器



# 半加器(Half Adder, HA)

**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

我们指定符号S(sum) and CO(carry) 作为输出

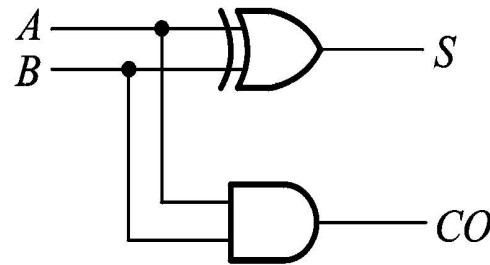
输入为A和B

真值表Truth Table

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

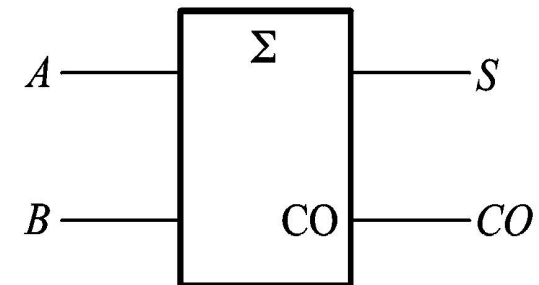
$$S = A \oplus B$$

$$CO = AB$$



(a)

一个异或门和一个与门



(b)



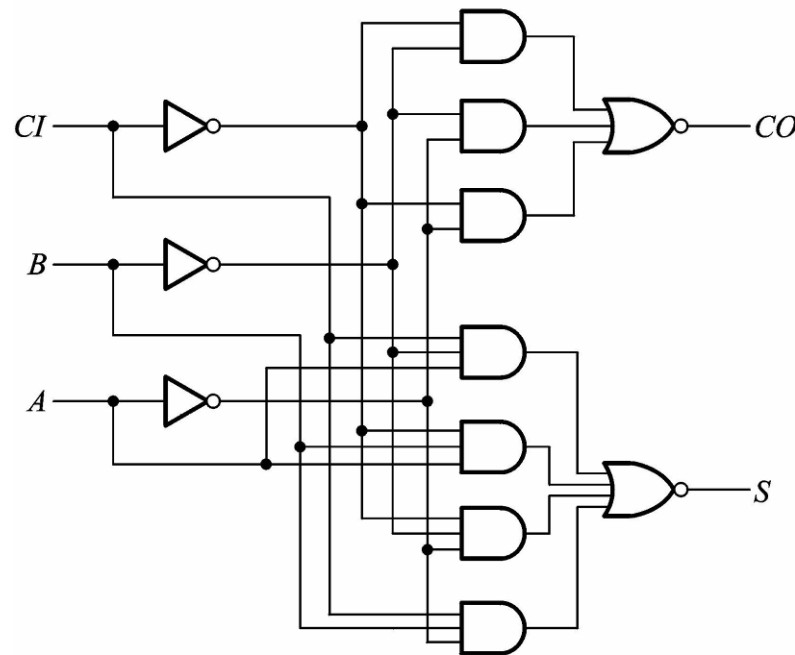
# 全加器(Full Adder, FA)

将两个1位二进制数A,B及来自低位的进位CI相加

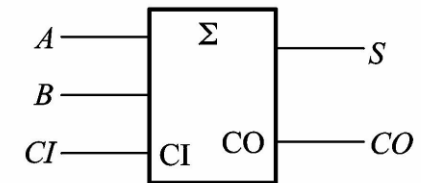
| 输 入 |   |    | 输 出 |    |
|-----|---|----|-----|----|
| A   | B | CI | S   | CO |
| 0   | 0 | 0  | 0   | 0  |
| 0   | 0 | 1  | 1   | 0  |
| 0   | 1 | 0  | 1   | 0  |
| 0   | 1 | 1  | 0   | 1  |
| 1   | 0 | 0  | 1   | 0  |
| 1   | 0 | 1  | 0   | 1  |
| 1   | 1 | 0  | 0   | 1  |
| 1   | 1 | 1  | 1   | 1  |

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

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



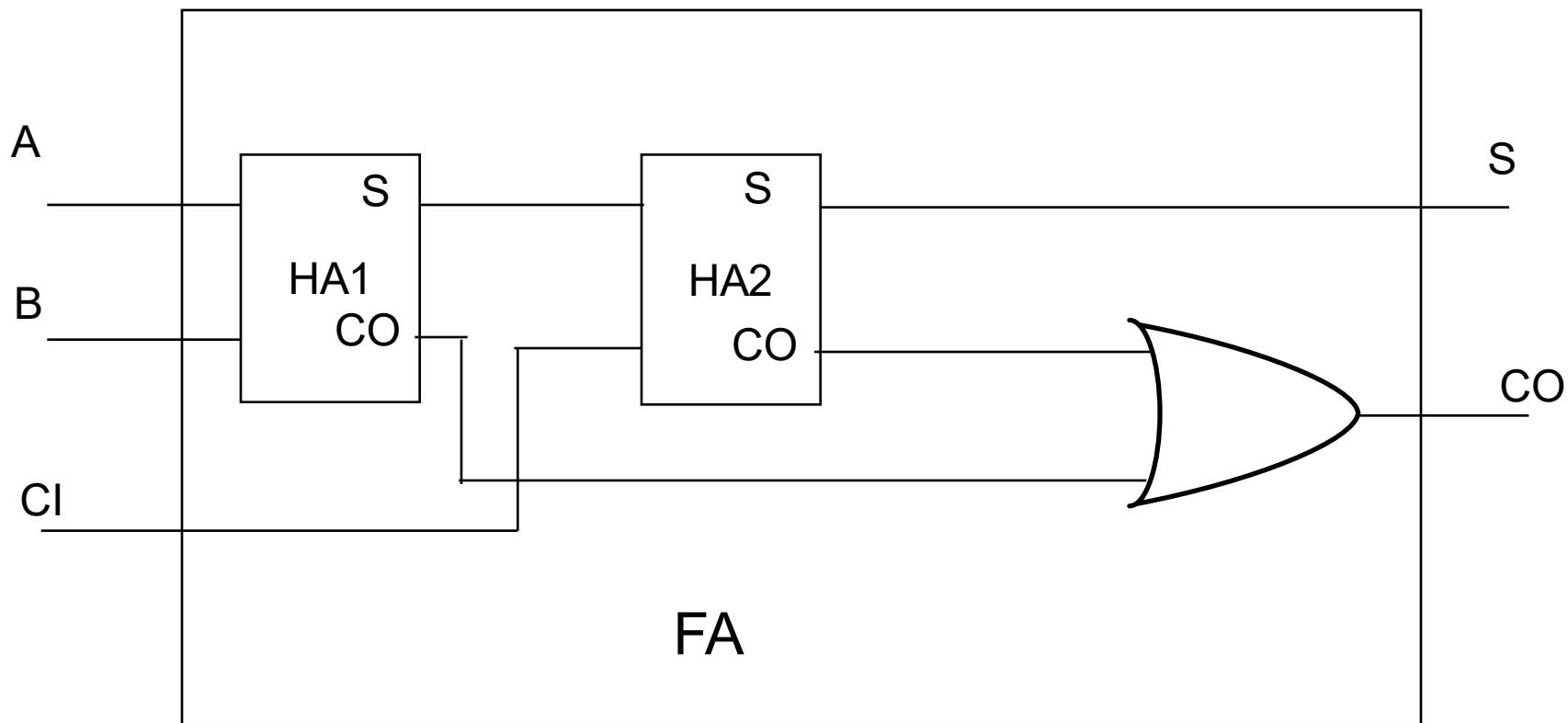
(a)



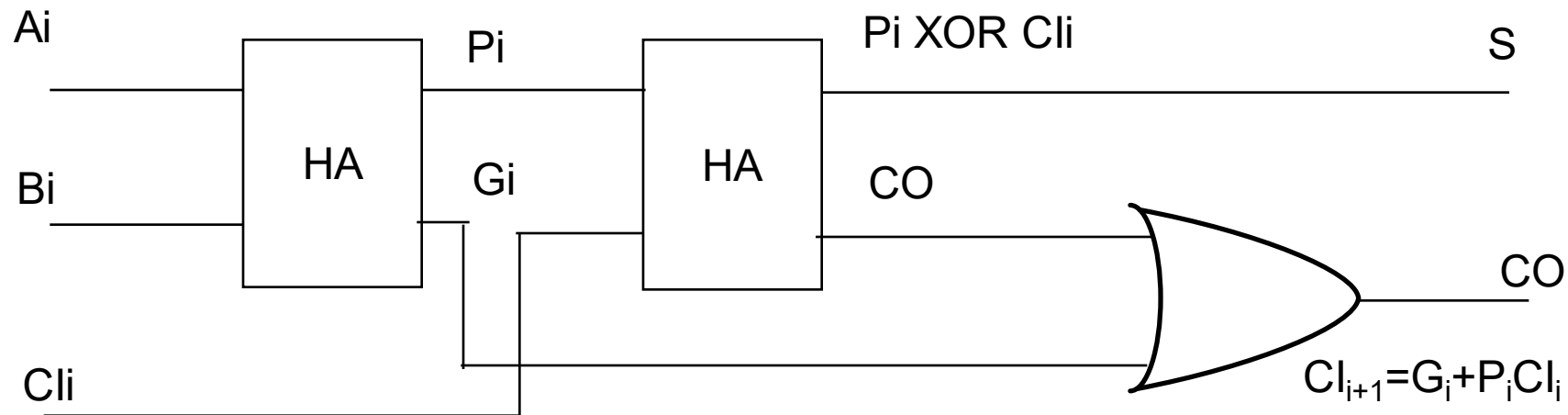
(b)

**74HC183**

# 两个半加器和1个或门实现全加器

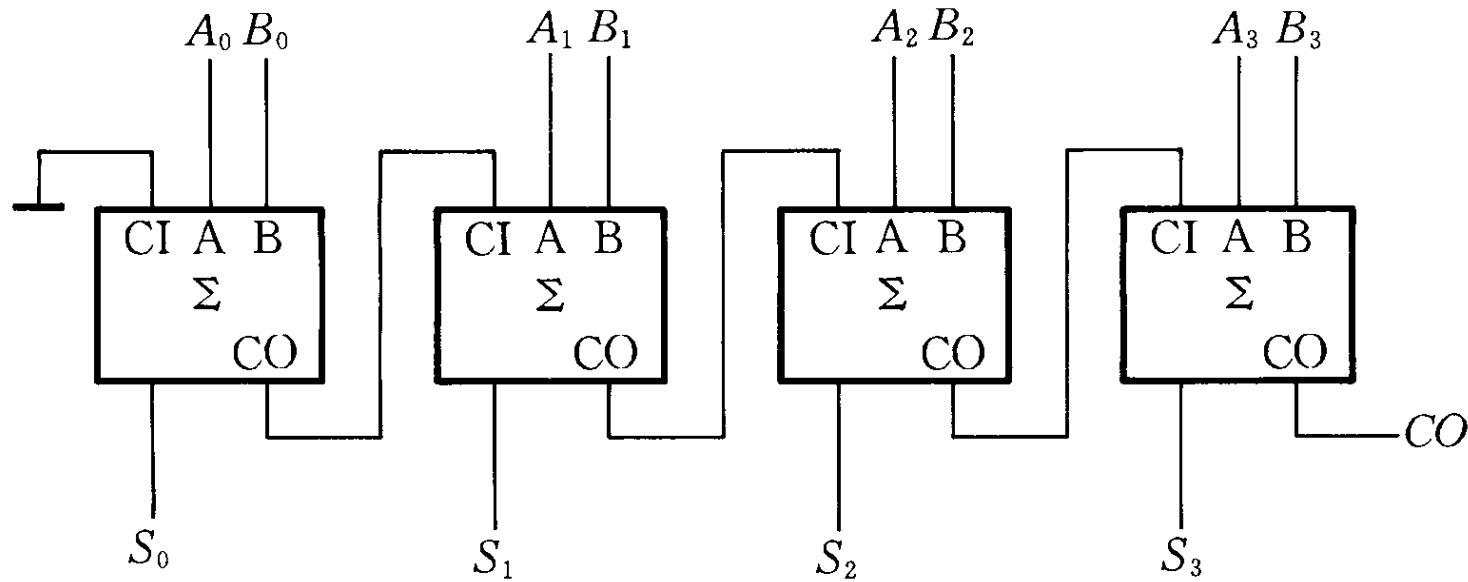


# 两个半加器和1个“或”门实现全加器



$P_i$ 进位传播  
 $G_i$ 进位产生

# 多位加法器:串行进位加法器



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

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

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

# 用加法器设计组合电路

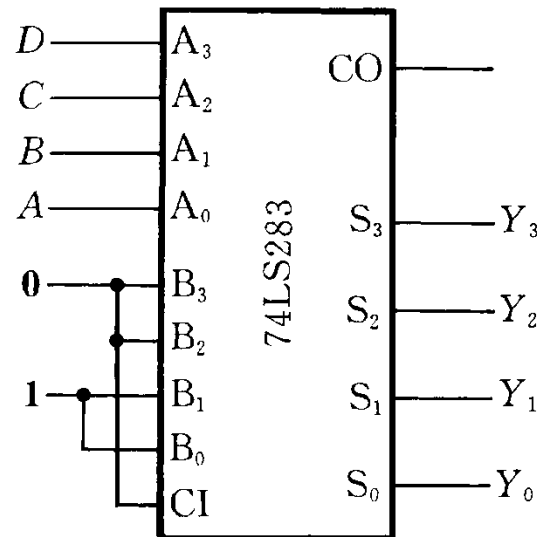
## □ 基本原理:

若能生成函数可变换成输入变量与输入变量相加

若能生成函数可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 | Y <sub>3</sub> | Y <sub>2</sub> | Y <sub>1</sub> | Y <sub>0</sub> |
| 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 (Magnitude Comparator)

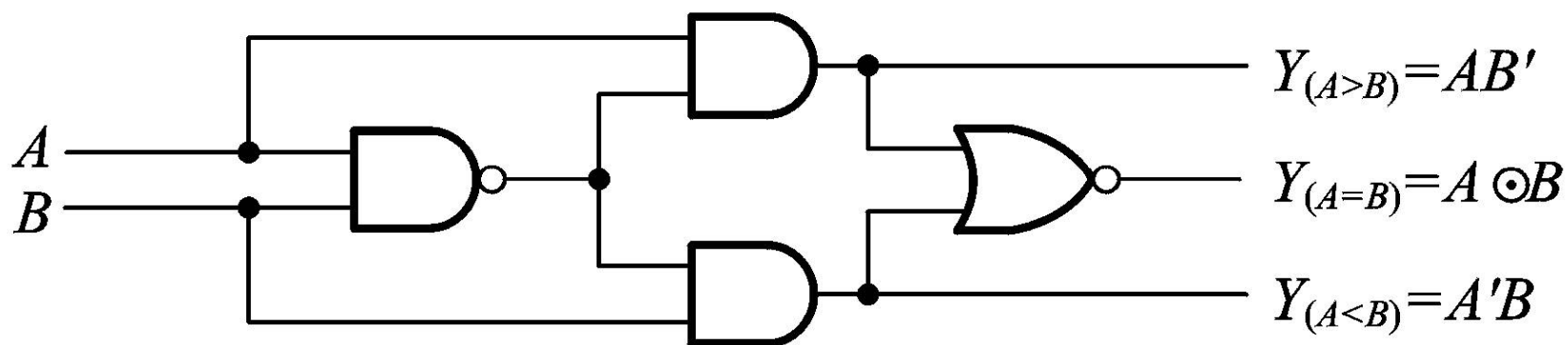
□ 用来比较两个二进制数的数值大小

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

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



# 多位数值比较器

原理：从高位比起，只有高位相等，才比较下一位

例如：

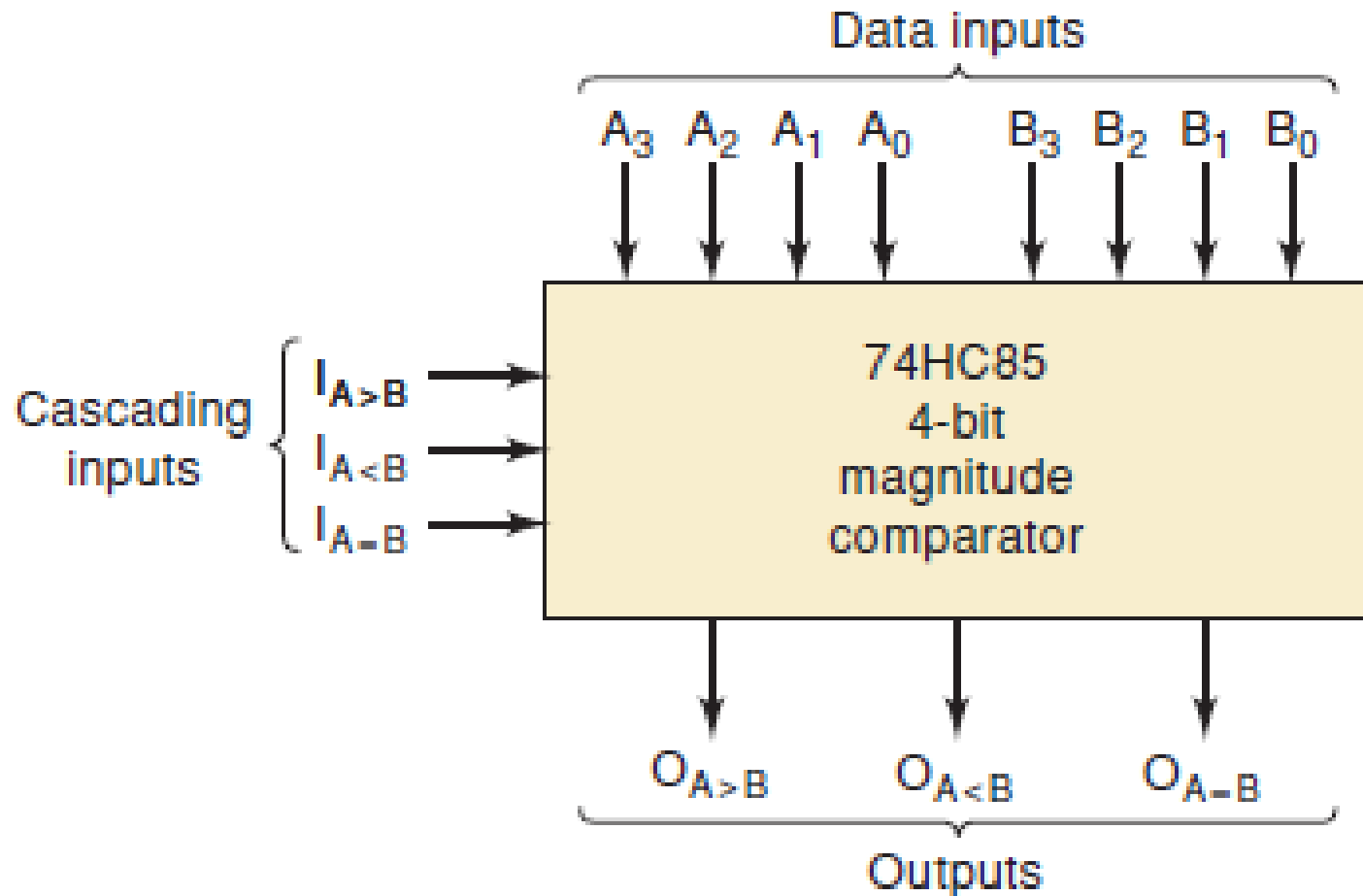
比较 $A_3A_2A_1A_0$ 和 $B_3B_2B_1B_0$

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

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

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

# 4位比较器 (Four-bit Magnitude Comparator)





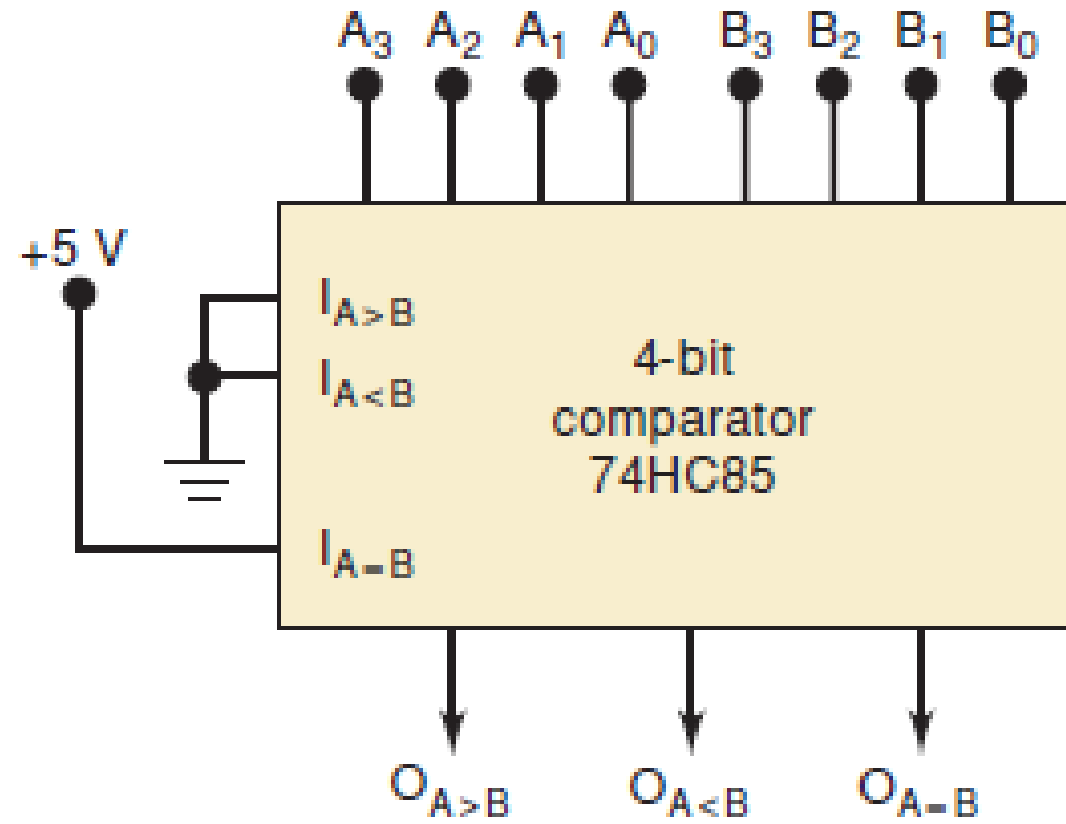
# 4位比较器的真值表

TRUTH TABLE

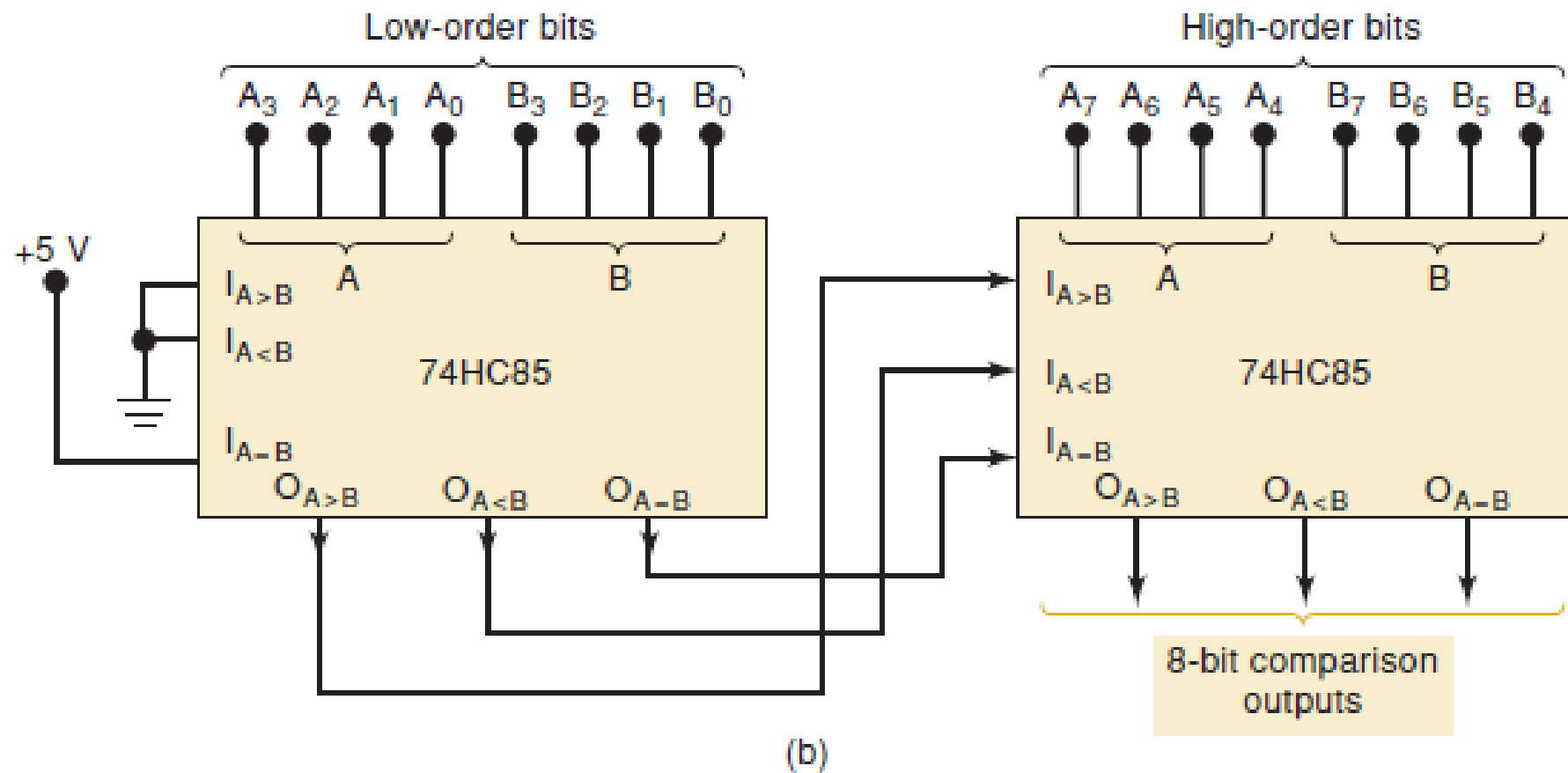
| COMPARING INPUTS |             |             |             | CASCADING INPUTS |           |           | OUTPUTS   |           |           |
|------------------|-------------|-------------|-------------|------------------|-----------|-----------|-----------|-----------|-----------|
| $A_3, B_3$       | $A_2, B_2$  | $A_1, B_1$  | $A_0, B_0$  | $I_{A>B}$        | $I_{A<B}$ | $I_{A=B}$ | $O_{A>B}$ | $O_{A<B}$ | $O_{A=B}$ |
| $A_3 > B_3$      | X           | X           | X           | X                | X         | X         | H         | L         | L         |
| $A_3 < B_3$      | X           | X           | X           | X                | X         | X         | L         | H         | L         |
| $A_3 = B_3$      | $A_2 > B_2$ | X           | X           | X                | X         | X         | H         | L         | L         |
| $A_3 = B_3$      | $A_2 < B_2$ | X           | X           | X                | X         | X         | L         | H         | L         |
| $A_3 = B_3$      | $A_2 = B_2$ | $A_1 > B_1$ | X           | X                | X         | X         | H         | L         | L         |
| $A_3 = B_3$      | $A_2 = B_2$ | $A_1 < B_1$ | X           | X                | X         | X         | L         | H         | L         |
| $A_3 = B_3$      | $A_2 = B_2$ | $A_1 = B_1$ | $A_0 > B_0$ | X                | X         | X         | H         | L         | L         |
| $A_3 = B_3$      | $A_2 = B_2$ | $A_1 = B_1$ | $A_0 < B_0$ | X                | X         | X         | L         | H         | L         |
| $A_3 = B_3$      | $A_2 = B_2$ | $A_1 = B_1$ | $A_0 = B_0$ | H                | L         | L         | H         | L         | L         |
| $A_3 = B_3$      | $A_2 = B_2$ | $A_1 = B_1$ | $A_0 = B_0$ | L                | H         | L         | L         | H         | L         |
| $A_3 = B_3$      | $A_2 = B_2$ | $A_1 = B_1$ | $A_0 = B_0$ | X                | X         | H         | L         | L         | H         |
| $A_3 = B_3$      | $A_2 = B_2$ | $A_1 = B_1$ | $A_0 = B_0$ | L                | L         | L         | H         | H         | L         |
| $A_3 = B_3$      | $A_2 = B_2$ | $A_1 = B_1$ | $A_0 = B_0$ | H                | H         | L         | L         | L         | L         |

H = HIGH Voltage Level  
L = LOW Voltage Level  
X = Immaterial

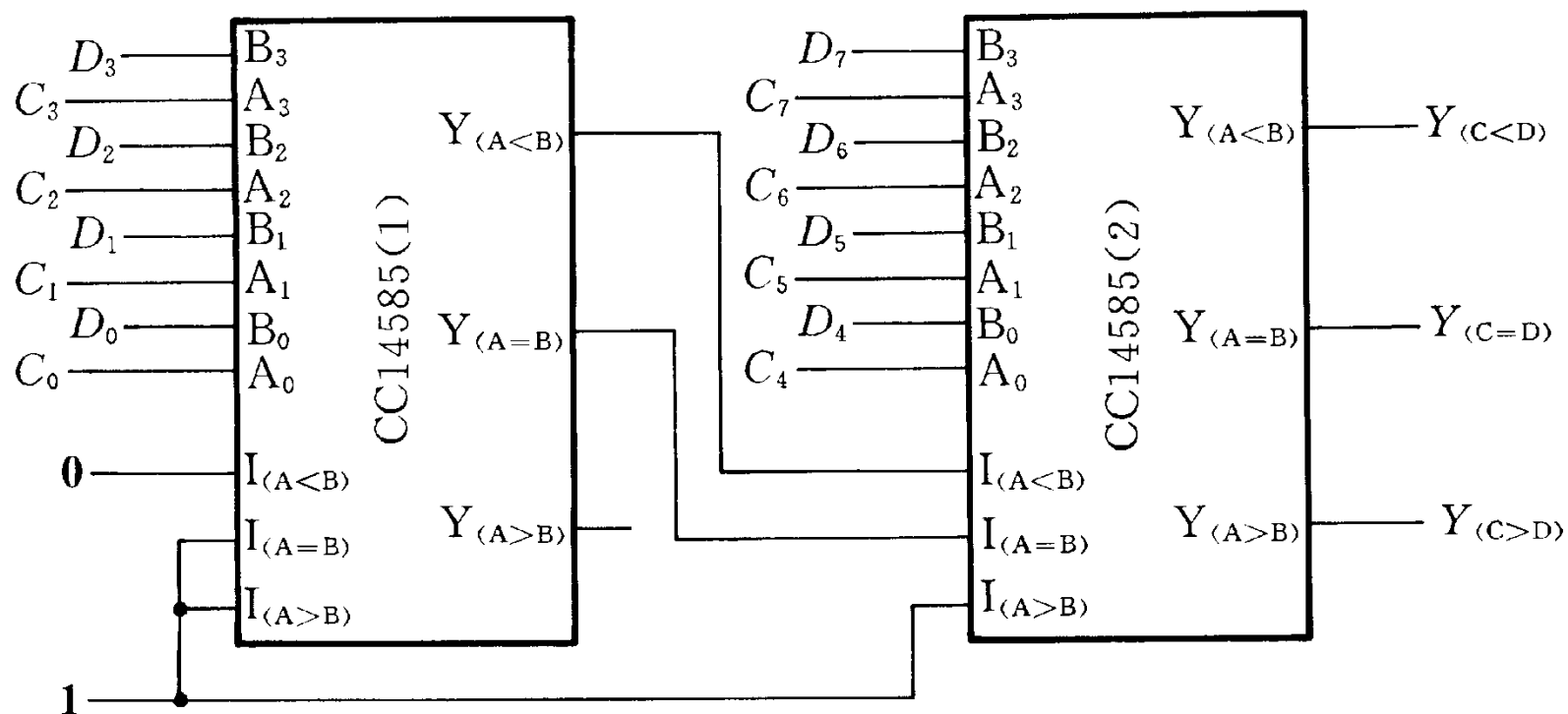
# 4位比较器



# 8位比较器



# 比较两个8位二进制数的大小



# 课后作业

## □ 回顾

- 选择器、加法器、比较器
- 采用模块设计组合电路

## □ 作业

- 学在浙大