



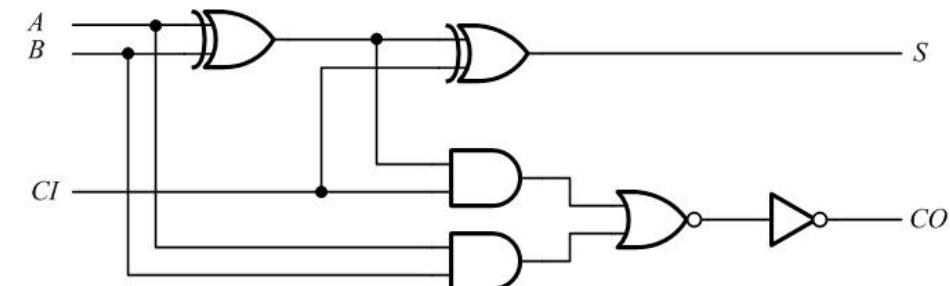
数字电路与系统

第四章、组合逻辑电路



第四章 组合逻辑电路

- ❖ 数字系统由数字电路模块构成，这些模块可分为两大类：
 - 组合逻辑电路；
 - 时序逻辑电路。
- ❖ 组合逻辑电路 — 功能上无记忆，结构上无反馈；
 - 电路任一时刻的输出状态只取决于该时刻各输入状态的组合，而与电路的原状态无关。



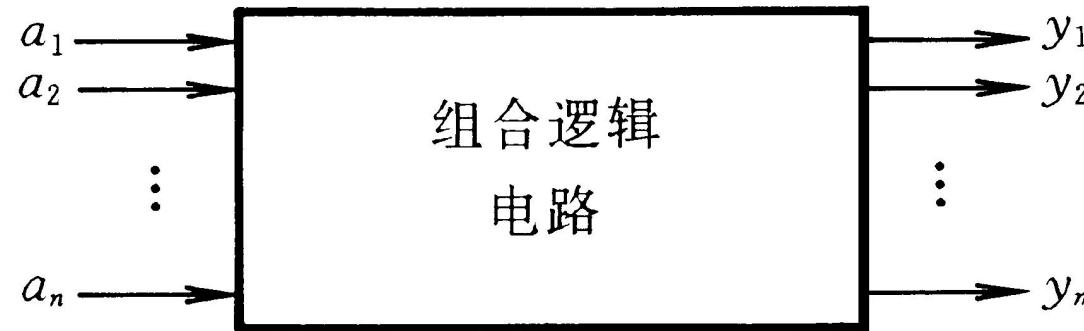
$$\begin{aligned}S &= (A \oplus B) \oplus CI \\CO &= (A \oplus B)CI + AB\end{aligned}$$

单比特加法器



第四章 组合逻辑电路

- 对于任何一个多输入多输出组合逻辑电路，可以用框图或者一组逻辑函数来表示



$$y_1 = f_1(a_1, a_2, \dots, a_n)$$

$$y_2 = f_2(a_1, a_2, \dots, a_n)$$

$$\mathbf{Y} = F(\mathbf{A})$$

...

$$y_m = f_m(a_1, a_2, \dots, a_n)$$

组合逻辑电路共同特点：不包含存储单元



第四章 组合逻辑电路

§ 4.1 组合逻辑电路的分析与设计

➤组合逻辑电路的分析

➤组合逻辑电路的设计

§ 4.2 组合逻辑电路模块及其应用

§ 4.3 组合逻辑电路中的竞争与冒险

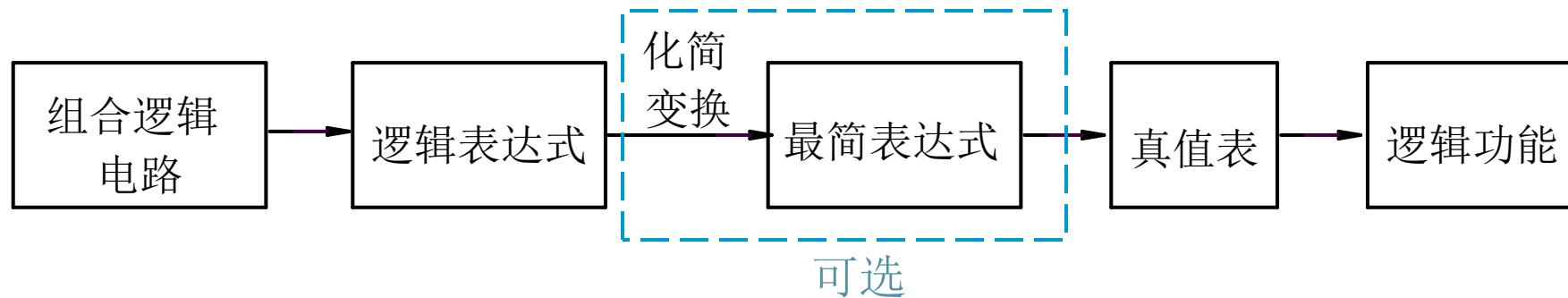


§ 4.1.1 组合逻辑电路分析

■ 组合逻辑电路的分析

➤ 组合逻辑电路的分析是用逻辑函数来**描述**已知的电路，找出输入、输出之间的关系，化简，得到真值表，从而判断电路**功能**；

➤ 分析步骤



§ 4.1.1 分析

- ✓ 写逻辑表达式

$$P = \overline{ABC}$$

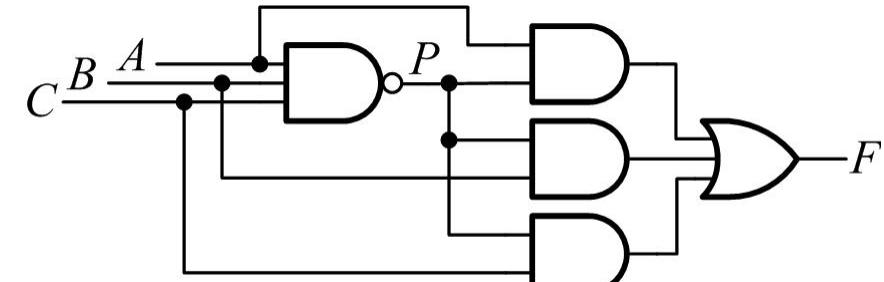
$$F = AP + BP + CP = \overline{A}\overline{ABC} + \overline{B}\overline{ABC} + \overline{C}\overline{ABC}$$

- ✓ 化简与变换

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

- ✓ 列出真值表

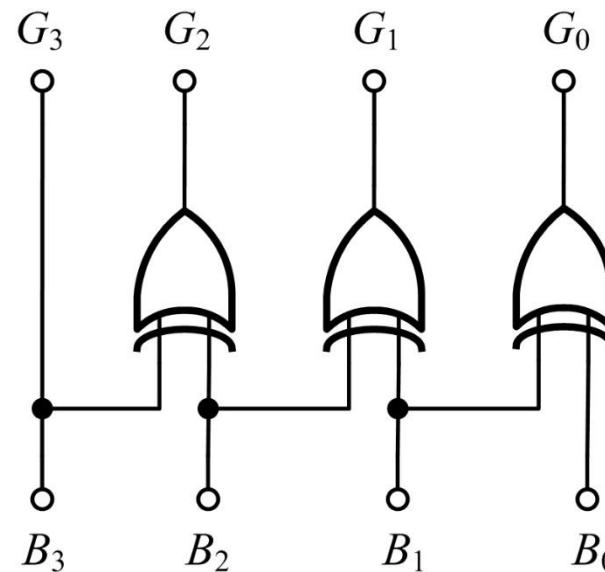
- ✓ 分析逻辑功能：“不一致电路”



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

§ 4.1.1 分析

✓ 自然二进制码至格雷码的转换电路



$$\begin{cases} G_3 = B_3 \\ G_2 = B_3 \oplus B_2 \\ G_1 = B_2 \oplus B_1 \\ G_0 = B_1 \oplus B_0 \end{cases}$$

B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

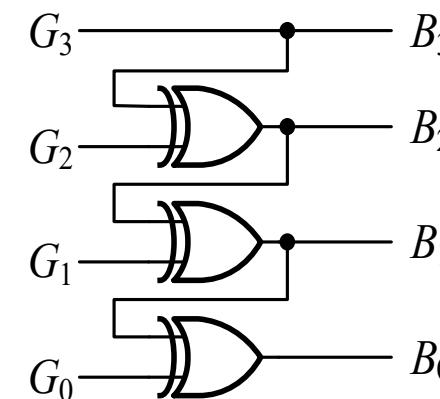


(续)

■ 思考：从格雷码转换到自然二进制码的公式和电路？

$$\begin{cases} G_3 = B_3 \\ G_2 = B_3 \oplus B_2 \\ G_1 = B_2 \oplus B_1 \\ G_0 = B_1 \oplus B_0 \end{cases}$$

$$\begin{cases} B_3 = G_3 \\ B_2 = B_3 \oplus G_2 = G_3 \oplus G_2 \\ B_1 = B_2 \oplus G_1 = G_3 \oplus G_2 \oplus G_1 \\ B_0 = B_1 \oplus G_0 = G_3 \oplus G_2 \oplus G_1 \oplus G_0 \end{cases}$$





第四章 组合逻辑电路

§ 4.1 组合逻辑电路的分析与设计

- 组合逻辑电路的分析
- 组合逻辑电路的设计

§ 4.2 组合逻辑电路模块及其应用

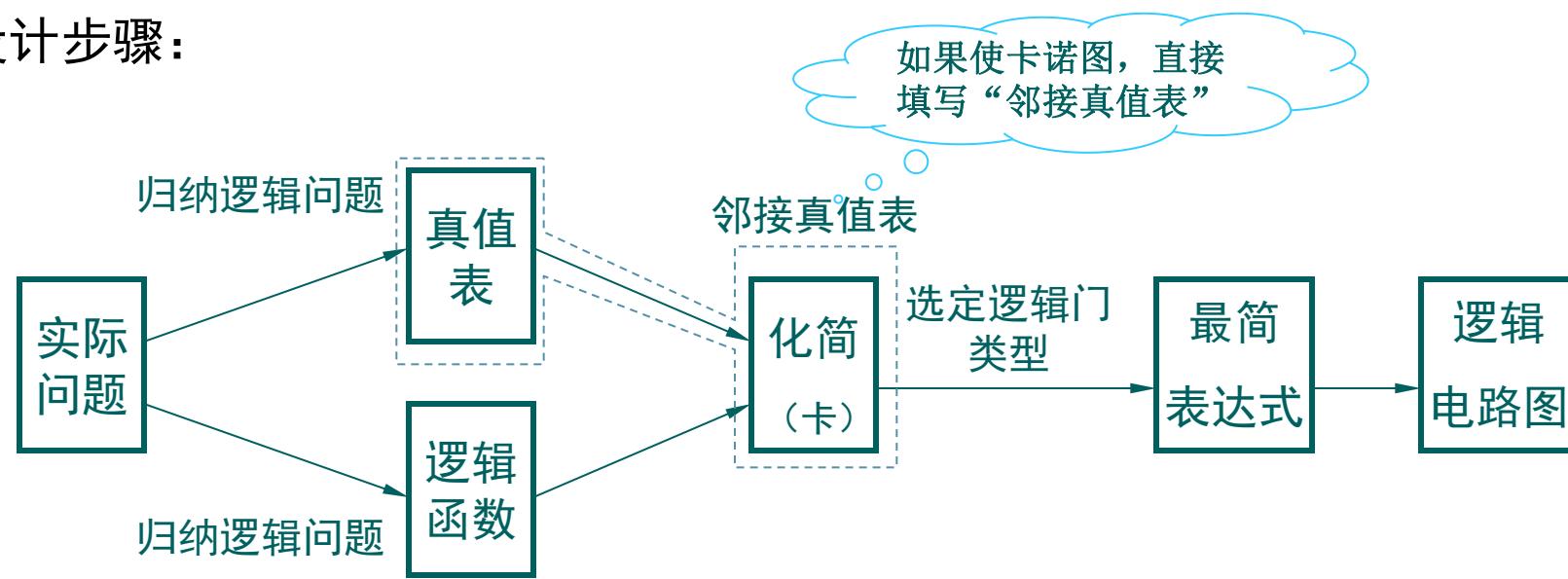
§ 4.3 组合逻辑电路中的竞争与冒险



§ 4.1.2 组合逻辑电路设计

组合逻辑电路的设计方法

- 组合逻辑电路的设计——即：根据给定的逻辑问题，得出能实现设计要求的逻辑电路
- 实现手段：
 - 采用门电路
 - 采用可编程逻辑器件
- 设计步骤：



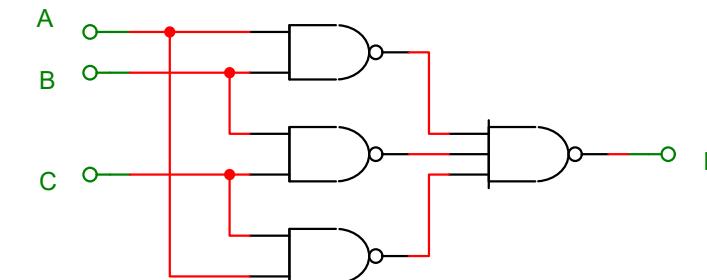
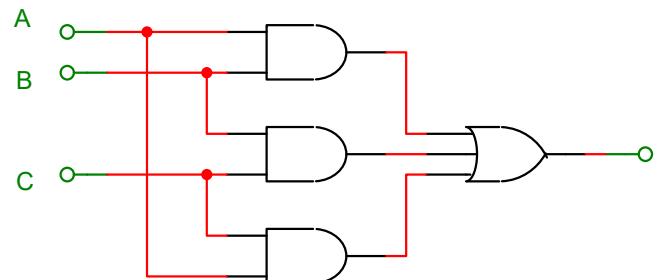


§ 4.1.2 组合逻辑电路设计

- 练习：按“少数服从多数”原则设计三人表决电路

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

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



§ 4.1.2 组合逻辑电路设计

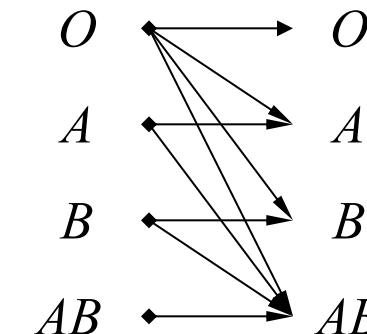
■ 练习：用与非门设计一输血与受血者血型关系检测电路

- 用 A 、 B 两个变量的4种组合表示献血者的血型
- 用 A_1 、 B_1 两个变量的4种组合表示受血者的血型
- 输出变量 F 表示两者之间的关系是否符合配合原则

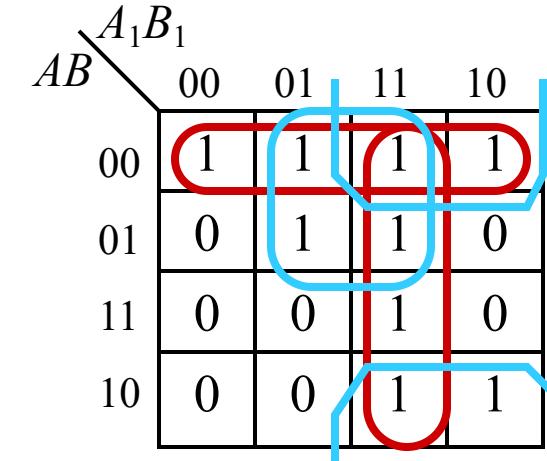
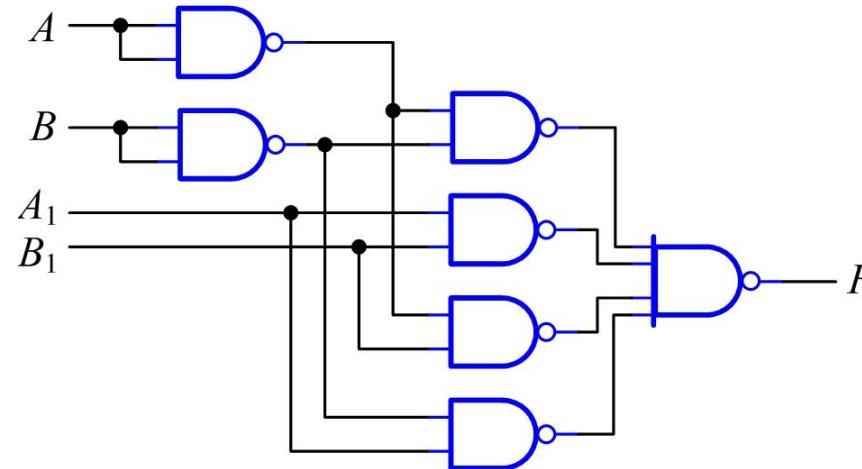
输入变量的表示

血型	献血者		受血者	
	A	B	A_1	B_1
O	0	0	0	0
A	1	0	1	0
B	0	1	0	1
AB	1	1	1	1

血型配合原则



§ 4.1.2 设计



$$\begin{aligned}F &= \overline{\overline{AB}} + A_1B_1 + \overline{A}\overline{B}_1 + \overline{B}\overline{A}_1 \\&= \overline{\overline{AB}} \cdot \overline{\overline{A_1B_1}} \cdot \overline{\overline{A}\overline{B}_1} \cdot \overline{\overline{B}\overline{A}_1}\end{aligned}$$

卡诺图中两个相邻（包括闭合、轴对称）“1格”的最小项可以合并成一个与项，消去一个原变量和反变量

反复利用合并项法则，保留相同变量，消掉相反变量

§ 4.1.2 设计

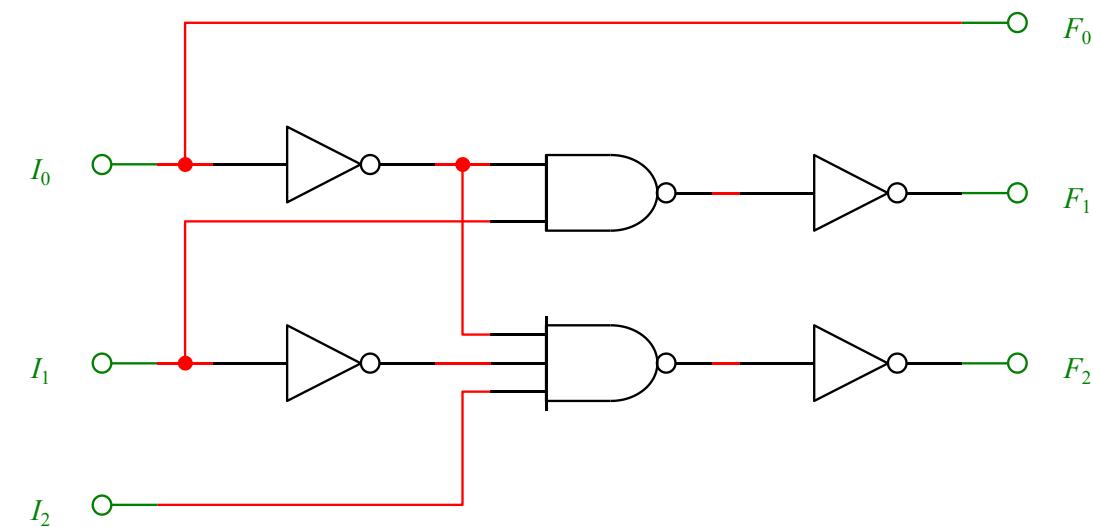
- 练习：设计一个电话信号控制电路：电路有 I_0 （火警）、 I_1 （盗警）和 I_2 （日常业务）三种输入信号，按优先次序分别从 F_0 、 F_1 、 F_2 输出，在同一时间只能有一个信号通过。要求用与非门实现。

输入输出关系表

输入			输出		
I_0	I_1	I_2	F_0	F_1	F_2
0	0	0	0	0	0
1	X	X	1	0	0
0	1	X	0	1	0
0	0	1	0	0	1

$$F_0 = I_0 \quad F_2 = \overline{I_0} \cdot \overline{I_1} \cdot I_2$$

$$F_1 = \overline{I_0} \cdot I_1$$





第四章 组合逻辑电路

§ 4.1 组合逻辑电路的分析与设计

§ 4.2 组合逻辑电路模块及其应用

§ 4.3 组合逻辑电路中的竞争与冒险



§ 4.2 组合逻辑电路模块及其应用

- 编码器
- 译码器
- 数据选择器
- 数值比较器
- 加法器



§ 4.2.1 组合逻辑电路模块 之 编码器

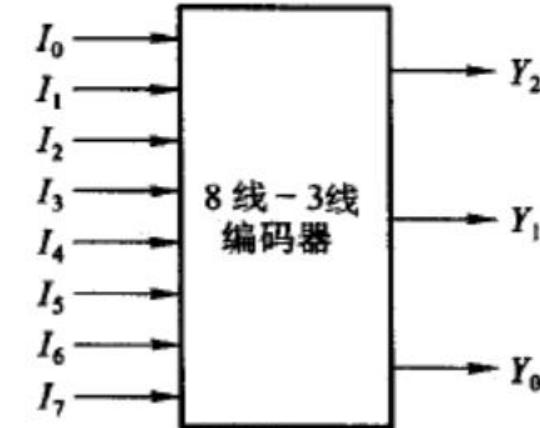
■ 编码器

- 编码器将字母、数字等信息符号编成一组二进制代码；
- 编码器对每一个**有效的**输入信号，都确定地产生的一组二进制代码与之对应；
- 编码器是一种**多输入多输出组合逻辑电路**；
- 通常 m 个输入信号，需要 n 位二进制编码， m 应不大于 2^n 。
- 常用的编码器有二进制编码器。

§ 4.2.1 编码器

■ 二进制编码器

- 用 n 位二进制代码对 2^n 个信号进行编码
- 三位二进制编码器 **8线—3线编码器**



输入								输出		
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	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

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	0	1	0
0	0	0	0	0	0	0	1	1	0	0

✓ 电路图

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

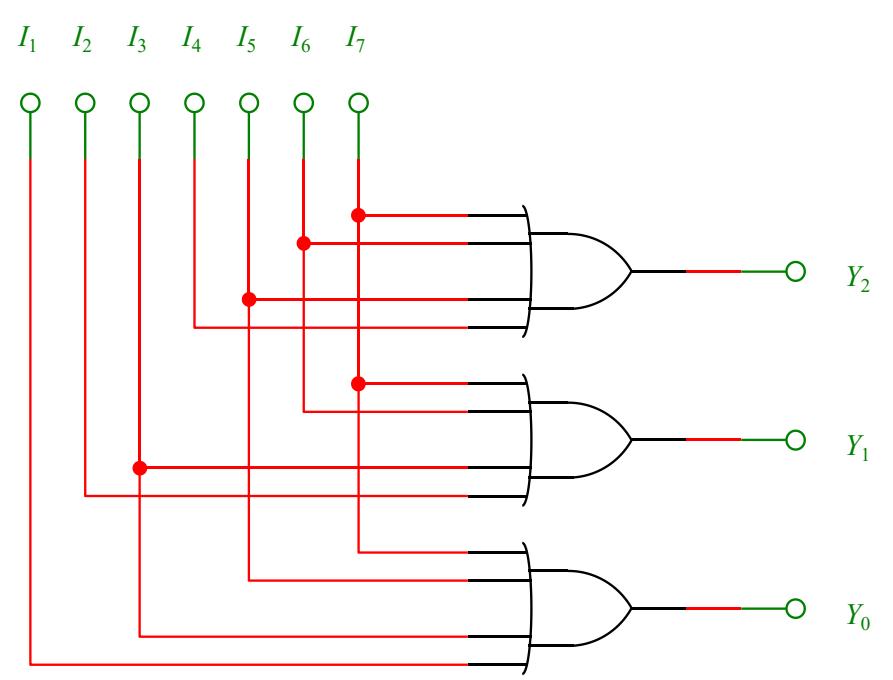
输入变量仅限表中取值，某
它为约束项，化简：

$$Y_1 = I_0'I_1'I_2'I_3'I_4'I_5'I_6'I_7 + I_0'I_1'I_2'I_3'I_4'I_5'I_6'I_7$$

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

$$Y_0 = I_0'I_1'I_2'I_3'I_4'I_5'I_6'I_7 + I_0'I_1'I_2'I_3'I_4'I_5'I_6'I_7$$

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



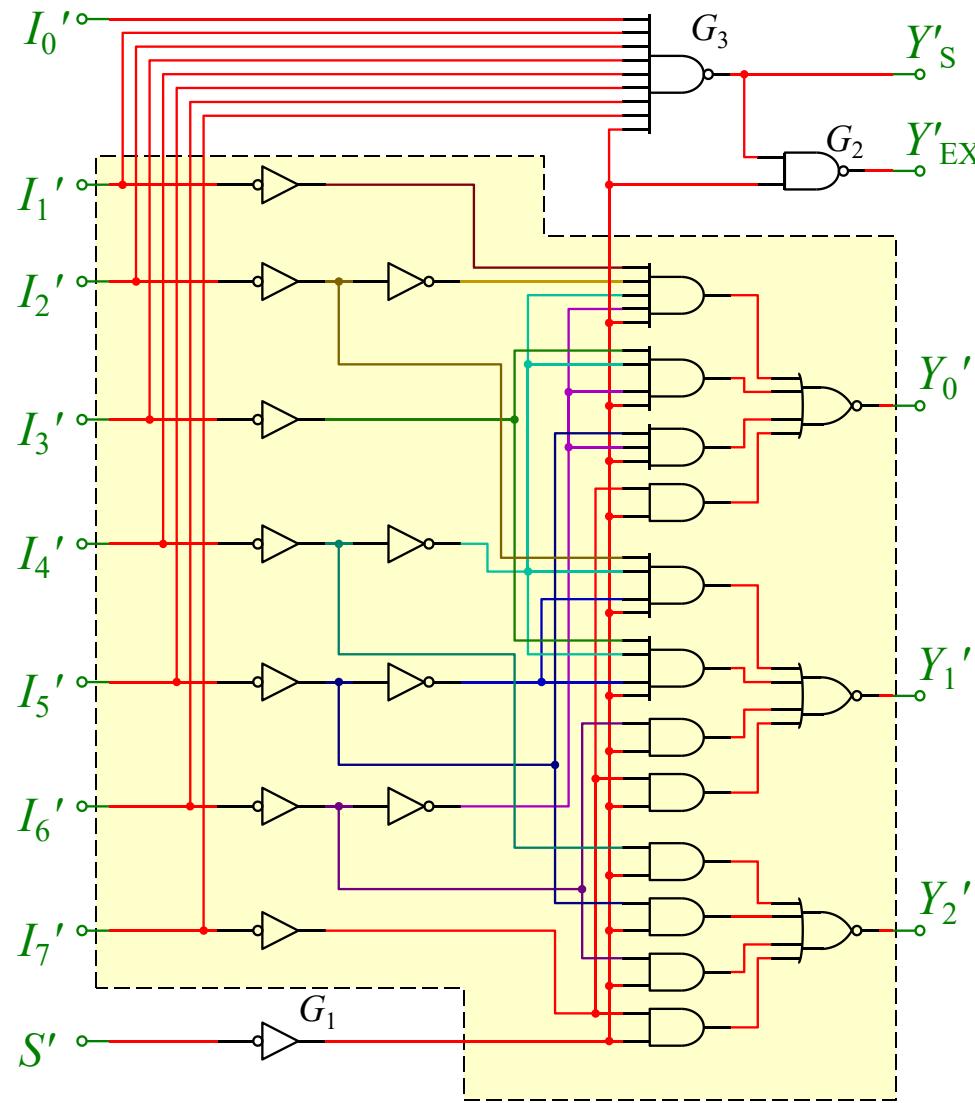


§ 4.2.1 编码器

■ 优先编码器

- 允许同时输入两个以上的编码信号，输入信号规定了优先顺序，当多个输入信号同时出现时，只对**优先级最高的**信号进行编码

- 常用的 8线—3线 优先编码器 74HC**148**
 - 除输入输出信号外，增加了输入、输出使能信号和输出扩展信号



例：74HC148

✓ 使能输入端 \bar{S}

✓ 虚框内为编码器主电路

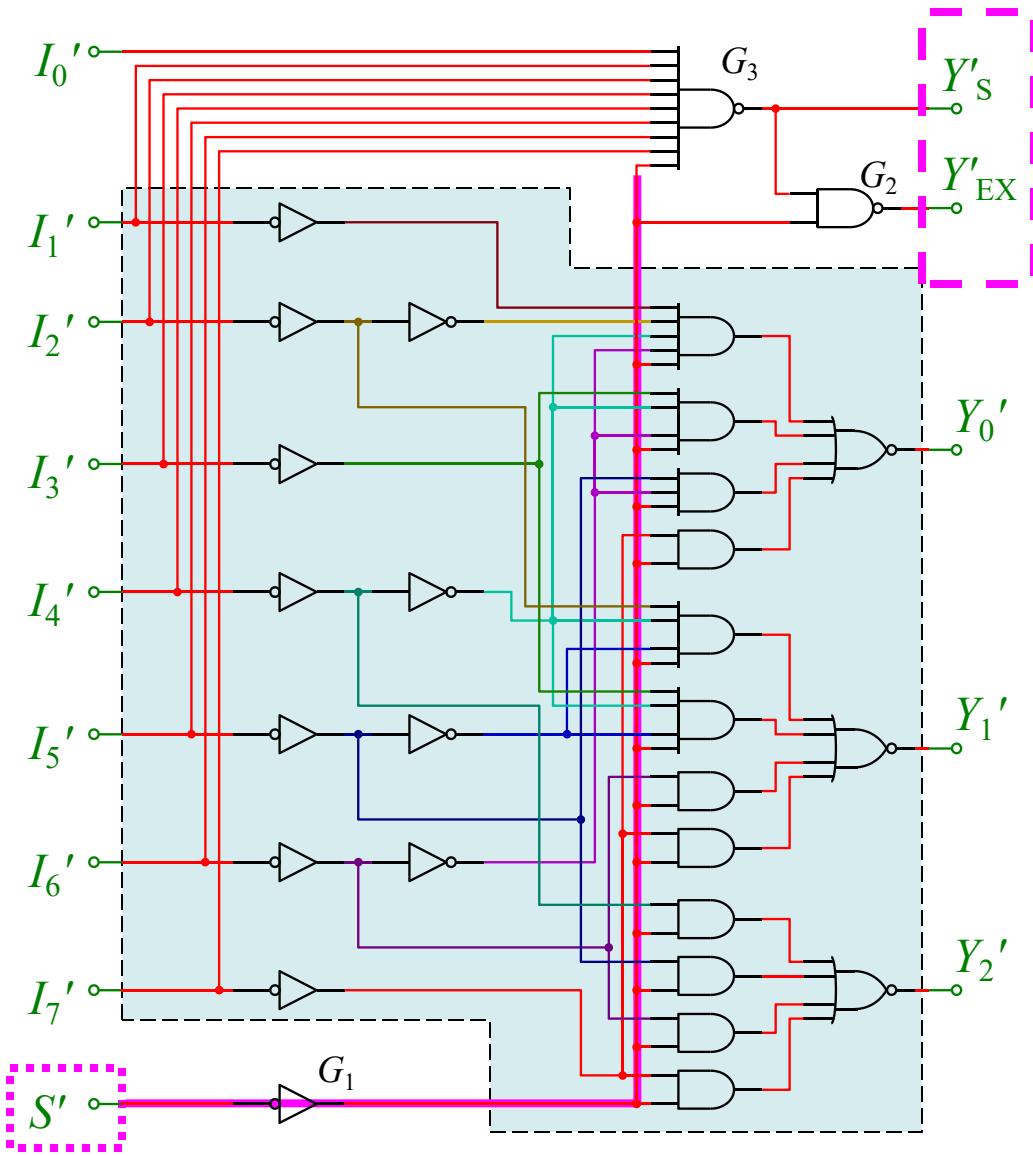
$$\bar{Y}_2 = \overline{(I_4 + I_5 + I_6 + I_7)} \cdot S$$

$$\bar{Y}_1 = \overline{(I_2 \bar{I}_4 \bar{I}_5 + I_3 \bar{I}_4 \bar{I}_5 + I_6 + I_7)} \cdot S$$

$$\bar{Y}_0 = \overline{(I_1 \bar{I}_2 \bar{I}_4 \bar{I}_6 + I_3 \bar{I}_4 \bar{I}_6 + I_5 \bar{I}_6 + I_7)} \cdot S$$

低电平为有效输入

为了扩展电路功能与增加灵活性，还增加了两个门



✓ 片选输入端 S' 低电平有效
高电平封锁

✓ 使能输出端 \bar{Y}_S

$$\bar{Y}_S = \overline{\overline{I_0} \cdot \overline{I_1} \cdots \overline{I_7} \cdot S}$$

只有所有输入为高且 $S=1$ 时，才为0；为低电平时表示电路正常，但无信号输入

✓ 扩展输出端 \bar{Y}_{EX}

$$\bar{Y}_{EX} = \overline{\overline{Y}_S \cdot S}$$

$$= \overline{(I_0 + I_1 + \cdots + I_7) \cdot S}$$

只要任意一个输入为低，且 $S=1$ 时，才为0；低电平表示电路工作且有信号输入



■ 优先编码器（续）

低电平为有效输入

I₇优先级最高，
容许多个同时输入

可以俗称为：

“有信号输入”和“无信号输入”指示

✓ 功能表

输入									输出				
\bar{S}	\bar{I}_0	\bar{I}_1	\bar{I}_2	\bar{I}_3	\bar{I}_4	\bar{I}_5	\bar{I}_6	\bar{I}_7	\bar{Y}_2	\bar{Y}_1	\bar{Y}_0	\bar{Y}_{EX}	\bar{Y}_S
1	×	×	×	×	×	×	×	×	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	×	×	×	×	×	×	×	0	0	0	0	0	1
0	×	×	×	×	×	×	0	1	0	0	1	0	1
0	×	×	×	×	0	1	1	1	0	1	0	0	1
0	×	×	×	0	1	1	1	1	0	1	1	0	1
0	×	×	0	1	1	1	1	1	1	0	0	0	1
0	0	1	1	1	1	1	1	1	1	1	0	0	1

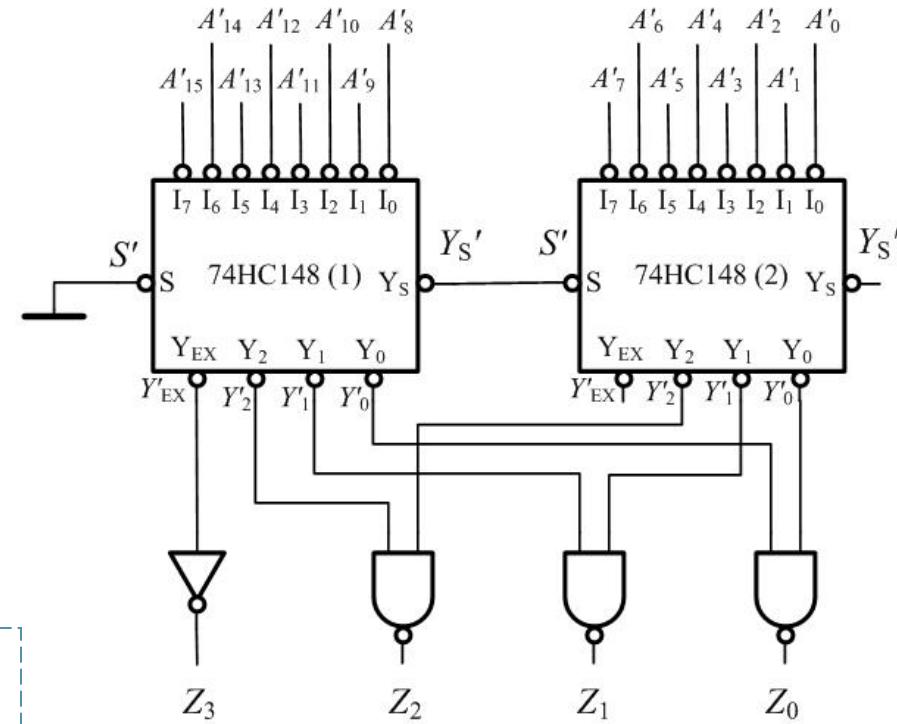
§ 4.2.1 编码器

■ 优先编码器（续）

➤ 编码器的扩展 →

- 16线-4线优先编码器

- ✓ 按优先级顺序，把高优先级
 - (1) 片的“无信号输入”端接低优先级(2)片的使能端；只有A15-A8均无输入信号时，才容许(2)的A7-A0编码
- ✓ 高优先级 (1) 片的“有信号输入”端作为编码输出最高位；
- ✓ 低三位为两片编码器输出信号的与非



A15优先级最高

注意：在中规模电路中，逻辑框图外部输入或输出端加小圆圈，同时在外部标注的输入或输出端信号加上非号，1、以低电平作为有效信号；2、门电路输入端加小圆圈有时候表示信号经反向后输入。需具体情况具体分析（通常会说明）。



§ 4.2.2 组合逻辑电路模块 之 译码器

- 与编码器相反，译码器将输入代码转换成特定的输出信号（**最常用的地址译码器**）
- 两种常用的译码器：
 - 二进制译码器
 - 数字显示译码器



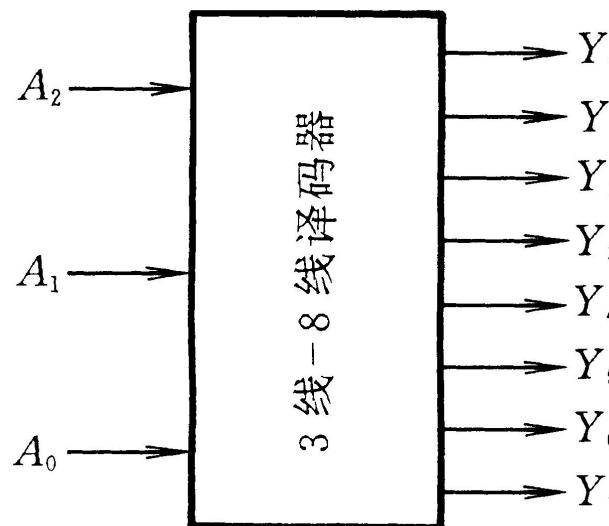
§ 4.2.2 译码器

■ 二进制译码器

- 译码器有 n 个输入信号和 m 个输出信号
- $m=2^n$ 二进制全译码器
 - 3线-8线译码器

§ 4.2.2 译码器

3线-8线译码器



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

$$Y_1 = \overline{A}_2 \overline{A}_1 A_0$$

$$Y_2 = \overline{A}_2 A_1 \overline{A}_0$$

$$Y_3 = \overline{A}_2 A_1 A_0$$

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

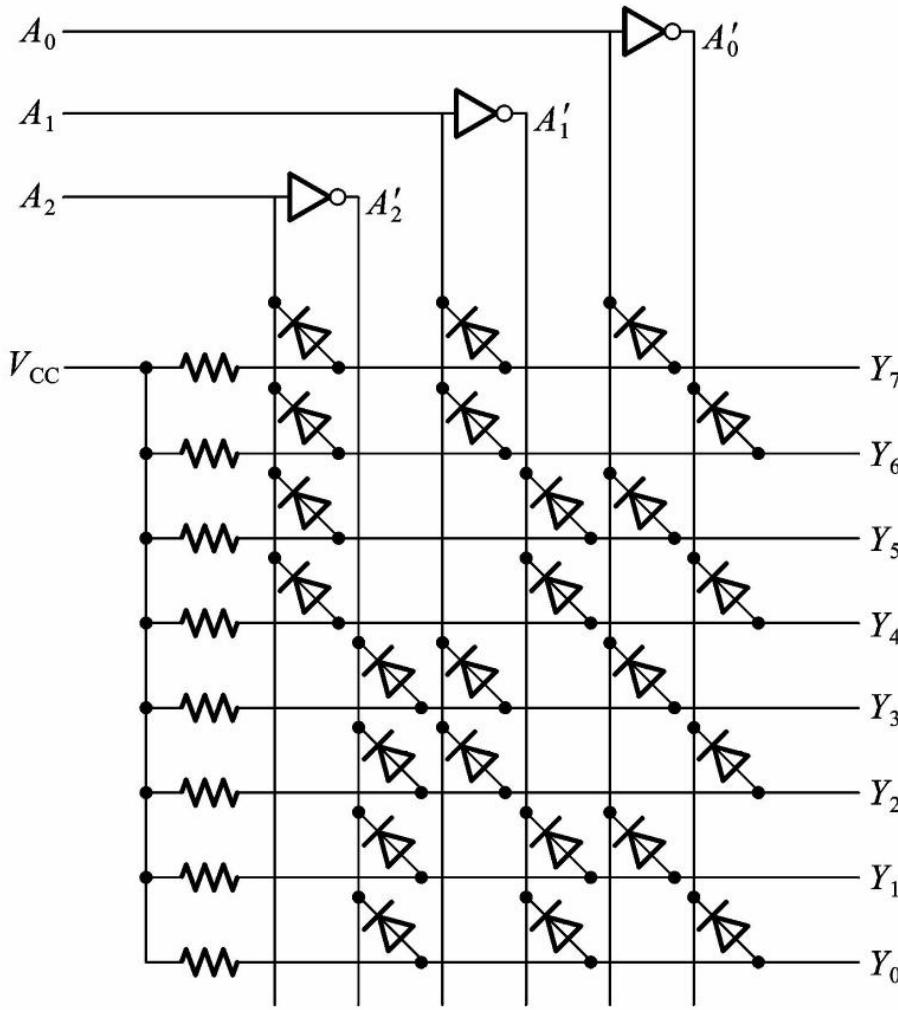
$$Y_5 = A_2 \overline{A}_1 A_0$$

$$Y_6 = A_2 A_1 \overline{A}_0$$

$$Y_7 = A_2 A_1 A_0$$

§ 4.2.2 译码器

用二极管与门阵列组成
3线-8线译码器



$$Y_0 = \overline{A_2} \overline{A_1} \overline{A_0}$$

$$Y_1 = \overline{A_2} \overline{A_1} A_0$$

$$Y_2 = \overline{A_2} A_1 \overline{A_0}$$

$$Y_3 = \overline{A_2} A_1 A_0$$

$$Y_4 = A_2 \overline{A_1} \overline{A_0}$$

$$Y_5 = A_2 \overline{A_1} A_0$$

$$Y_6 = A_2 A_1 \overline{A_0}$$

$$Y_7 = A_2 A_1 A_0$$



§ 4.2.2 译码器

- 集成译码器74HC138
- 常用的3线-8线译码器

S_1 、 $\overline{S_2}$ 和 $\overline{S_3}$

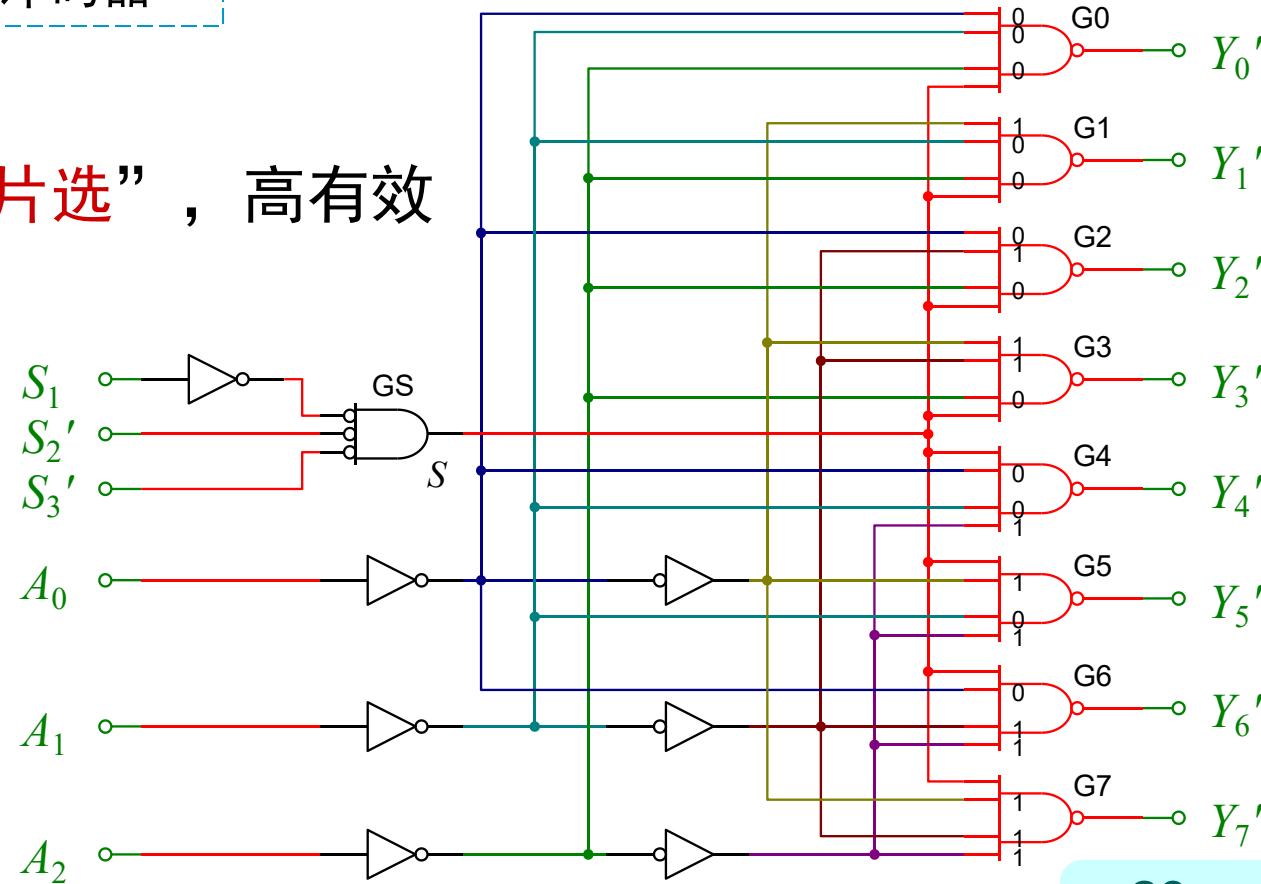
为使能端，又叫作“片选”，高有效

$$S = S_1 \overline{S_2} + \overline{S_3}$$

作用：

- ✓ $S=1$ 时，工作
- ✓ $S=0$ 时，禁止，所有输出封锁为高电平

输入					输出							
S_1	$\overline{S_2} + \overline{S_3}$	A_2	A_1	A_0	\overline{Y}_7	\overline{Y}_6	\overline{Y}_5	\overline{Y}_4	\overline{Y}_3	\overline{Y}_2	\overline{Y}_1	\overline{Y}_0
0	\times	\times	\times	\times	1	1	1	1	1	1	1	1
\times	1	\times	\times	\times	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	0	1	1	1	1



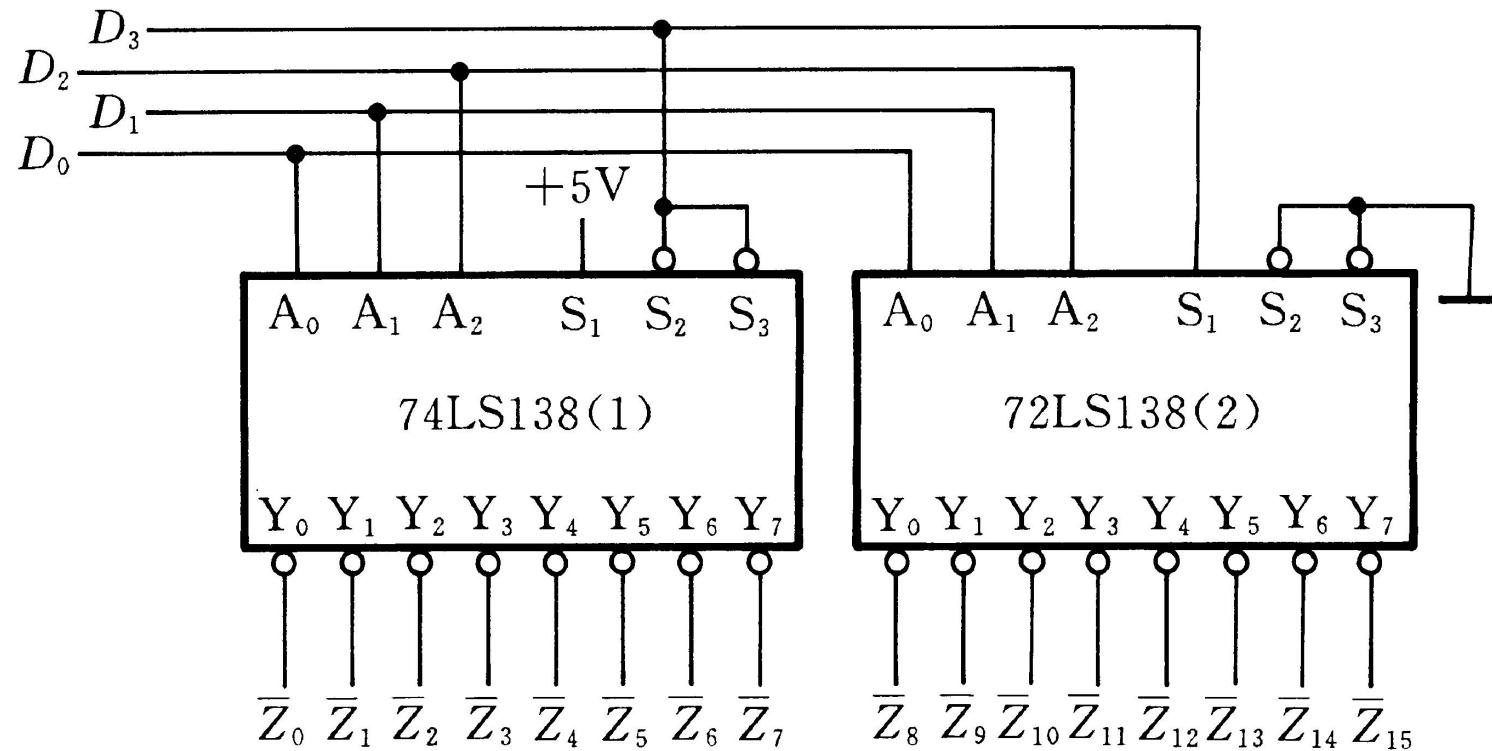
$$S = S_1 \overline{S_2} + \overline{S_3}$$

§ 4.2.2 译码器

思考：如果需要 6线-64线译码器，
需要几枚74138芯片？

■ 译码器的扩展

➤ 例题：将两片74HC138扩展为4线—16线译码器

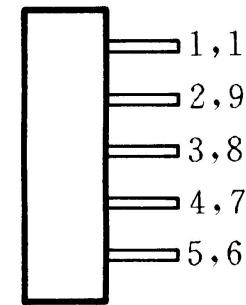
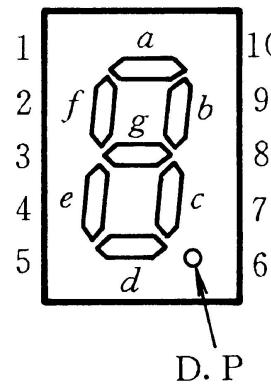


§ 4.2.2 译码器

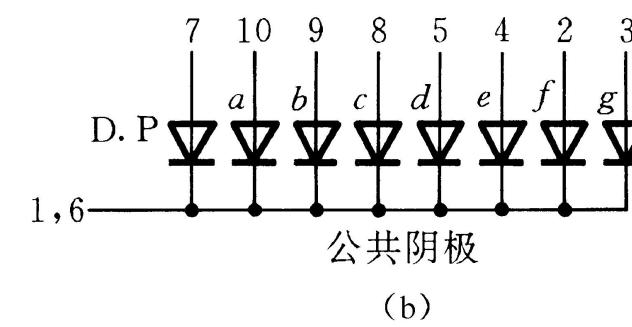
■ 数字显示译码器 (随处可见)

- 能够显示数字、字母或符号的器件称为**数字显示器**
- 能把数字量翻译成数字显示器所能识别的信号的译码器称为**数字显示译码器**。

■ 七段数字显示器 (半导体数码管BS201A, 含小数点八段)



(a)



(b)

§ 4.2.2 译码器

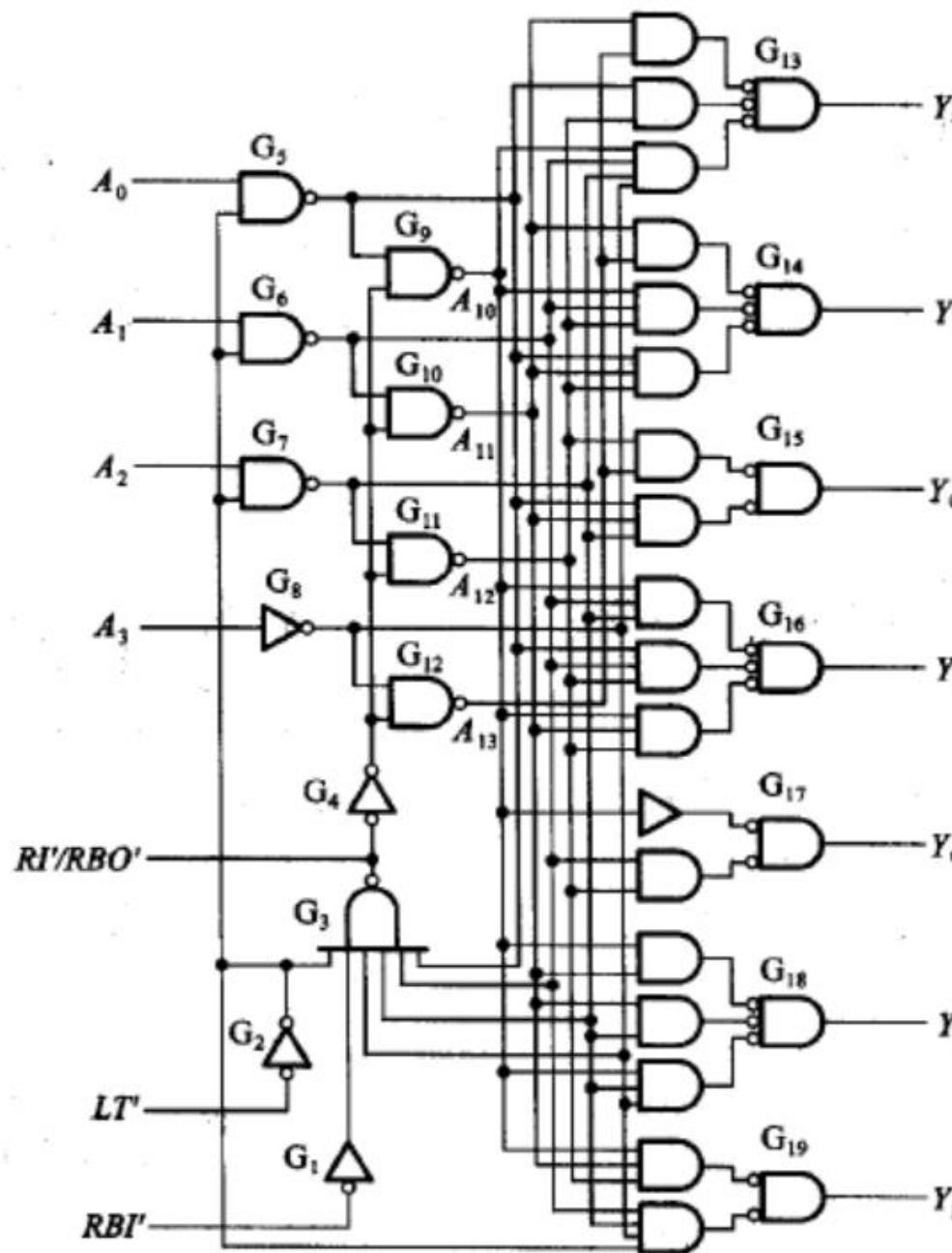
七段数字 显示译码

每个输入代码
多个输出有效



$$\left\{ \begin{array}{l} Y_a = (A'_3 A'_2 A'_1 A_0 + A_3 A_1 + A_2 A'_0)' \\ Y_b = (A_3 A_1 + A_2 A_1 A'_0 + A_2 A'_1 A_0)' \\ Y_c = (A_3 A_2 + A'_2 A_1 A'_0)' \\ Y_d = (A_2 A_1 A_0 + A_2 A'_1 A'_0 + A'_2 A'_1 A_0)' \\ Y_e = (A_2 A'_1 + A_0)' \\ Y_f = (A'_3 A'_2 A_0 + A'_2 A_1 + A_1 A_0)' \\ Y_g = (A'_3 A'_2 A'_1 + A_2 A_1 A_0)' \end{array} \right.$$

七段显示译码器7448逻辑公式

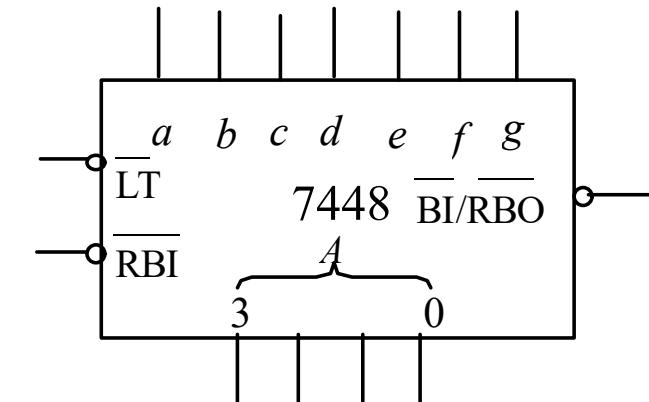


七段显示译码器7448逻辑图

§ 4.2.2 译码器

➤ 七段显示译码器7448

- ✓ 试灯 $\overline{LT}=0$ 所有灯亮
- ✓ 灭零 $\overline{RBI}=0$ 把不希望显示的零熄灭
- ✓ 正常译码显示 $\overline{LT}=1, \overline{RBI}=1$
- ✓ 控制端 $\overline{RI}/\overline{RBO}$ 可作输入端/输出端



输入 $\overline{RI}=0$, 数码管全灭, \overline{RI} 为灭灯输入端;

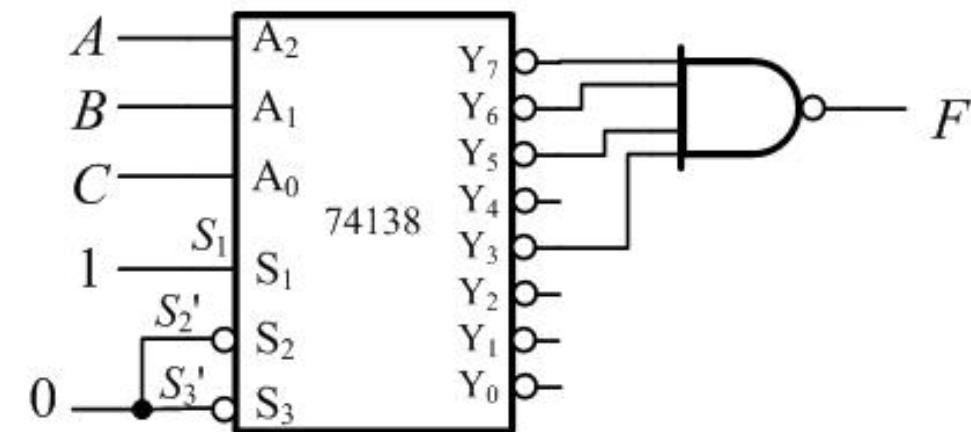
作为输出端使用时, 称为灭零输出端, 如
当 $RBI'=0$, 输入为0000时, RBO' 输出0,
指示该片处于灭零状态

- ✓ $\overline{BI}/\overline{RBO}$ 和 \overline{RBI} 配合使用, 可以实现多位数显示时的“无效0消隐”功能

§ 4.2.2 译码器

- 二进制全译码器的应用——实现组合逻辑函数
 - 译码器的每个输出端分别与一个最小项相对应

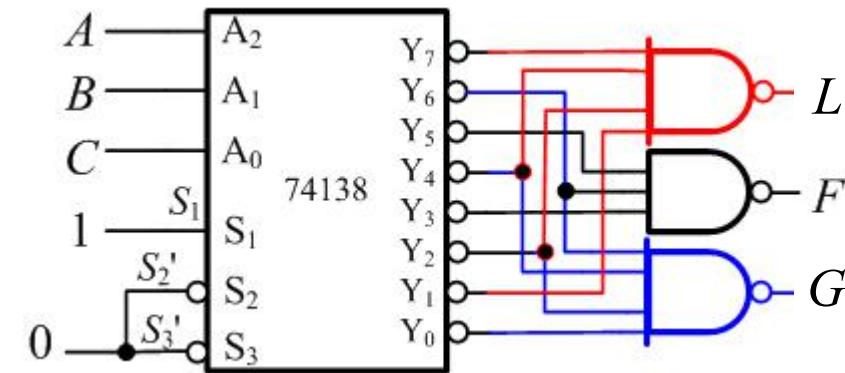
$$\begin{aligned}F &= AB + BC + AC \\&= \overline{ABC} + A\overline{B}C + A\overline{B}\overline{C} + ABC \\&= m_3 + m_5 + m_6 + m_7 \\&= \overline{m_3} \cdot \overline{m_5} \cdot \overline{m_6} \cdot \overline{m_7} \\&= m_3 \cdot m_5 \cdot m_6 \cdot m_7\end{aligned}$$





■ 实现组合逻辑函数

输入			输出		
A	B	C	L	F	G
0	0	0	0	0	1
0	0	1	1	0	0
0	1	0	1	0	1
0	1	1	0	1	0
1	0	0	1	0	1
1	0	1	0	1	0
1	1	0	0	1	1
1	1	1	1	0	0



$$L = \overline{A} \overline{B} C + \overline{A} B \overline{C} + A \overline{B} \overline{C} + ABC = m_1 + m_2 + \overline{\overline{m}_4 + m_7} = \overline{\overline{m}_1 \cdot \overline{m}_2 \cdot \overline{m}_4 \cdot \overline{m}_7}$$

$$F = \overline{A} BC + A \overline{B} C + AB \overline{C} = m_3 + m_5 + m_6 = \overline{\overline{m}_3 \cdot \overline{m}_5 \cdot \overline{m}_6}$$

$$G = \overline{A} \overline{B} \overline{C} + \overline{A} B \overline{C} + A \overline{B} \overline{C} + AB \overline{C} = m_0 + m_2 + m_4 + m_6 = \overline{\overline{m}_0 \cdot \overline{m}_2 \cdot \overline{m}_4 \cdot \overline{m}_6}$$

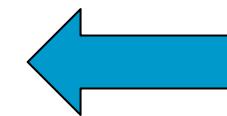
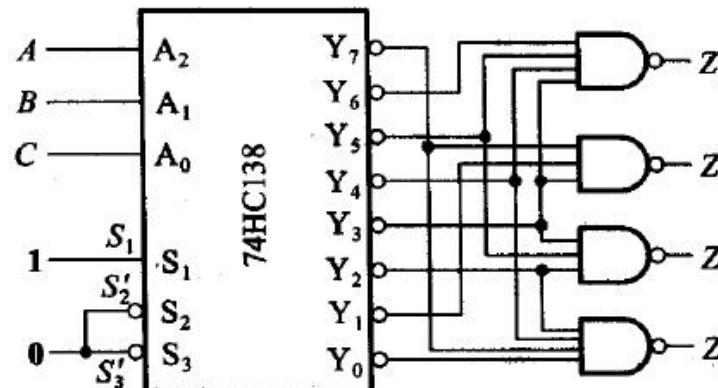


■ 用译码器实现组合逻辑函数

$$\begin{cases} Z_1 = AC' + A'BC + AB'C \\ Z_2 = BC + A'B'C \\ Z_3 = A'B + AB'C \\ Z_4 = A'BC' + B'C' + ABC \end{cases}$$



$$\begin{cases} Z_1 = ABC' + AB'C' + A'BC + AB'C = m_3 + m_4 + m_5 + m_6 \\ Z_2 = ABC + A'BC + A'B'C = m_1 + m_3 + m_7 \\ Z_3 = A'BC + A'BC' + AB'C = m_2 + m_3 + m_5 \\ Z_4 = A'BC' + AB'C' + A'B'C' + ABC = m_0 + m_2 + m_4 + m_7 \end{cases}$$

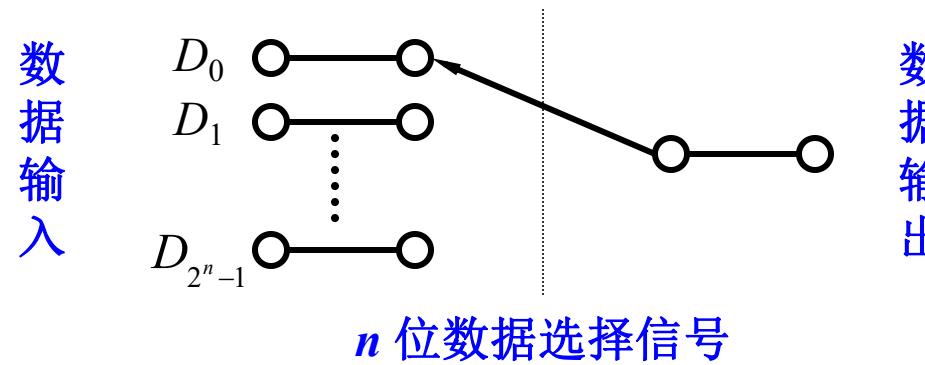


$$\begin{cases} Z_1 = (m'_3 \cdot m'_4 \cdot m'_5 \cdot m'_6)' \\ Z_2 = (m'_1 \cdot m'_3 \cdot m'_7)' \\ Z_3 = (m'_2 \cdot m'_3 \cdot m'_5)' \\ Z_4 = (m'_0 \cdot m'_2 \cdot m'_4 \cdot m'_7)' \end{cases}$$

§ 4.2.3 组合逻辑电路模块 之 数据选择器

■ 数据选择器

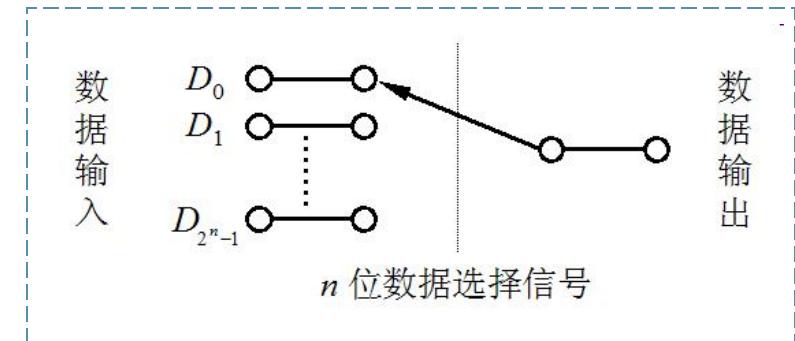
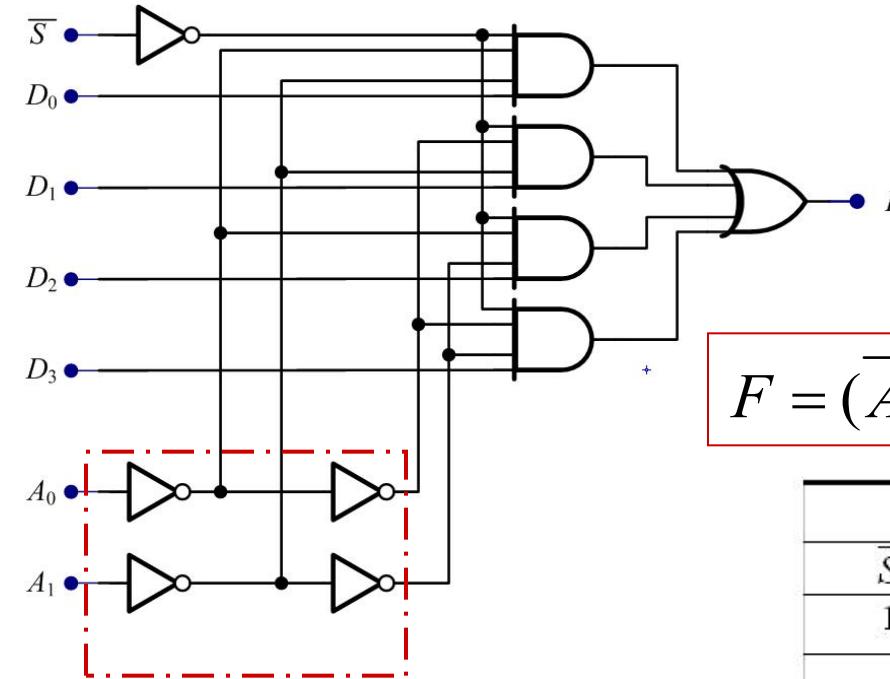
- 根据地址选择码从多路输入数据中选择一路输出



- ✓ 注意：选择信号位数n与数据输入信号个数满足的关系
- ✓ 常用的数据选择器有4选1、8选1、16选1等多种类型

§ 4.2.3 数据选择器

- 四选一数据选择器



逻辑表达式

$$F = (\overline{A_1} \overline{A_0} D_0 + \overline{A_1} A_0 D_1 + A_1 \overline{A_0} D_2 + A_1 A_0 D_3) \cdot S$$

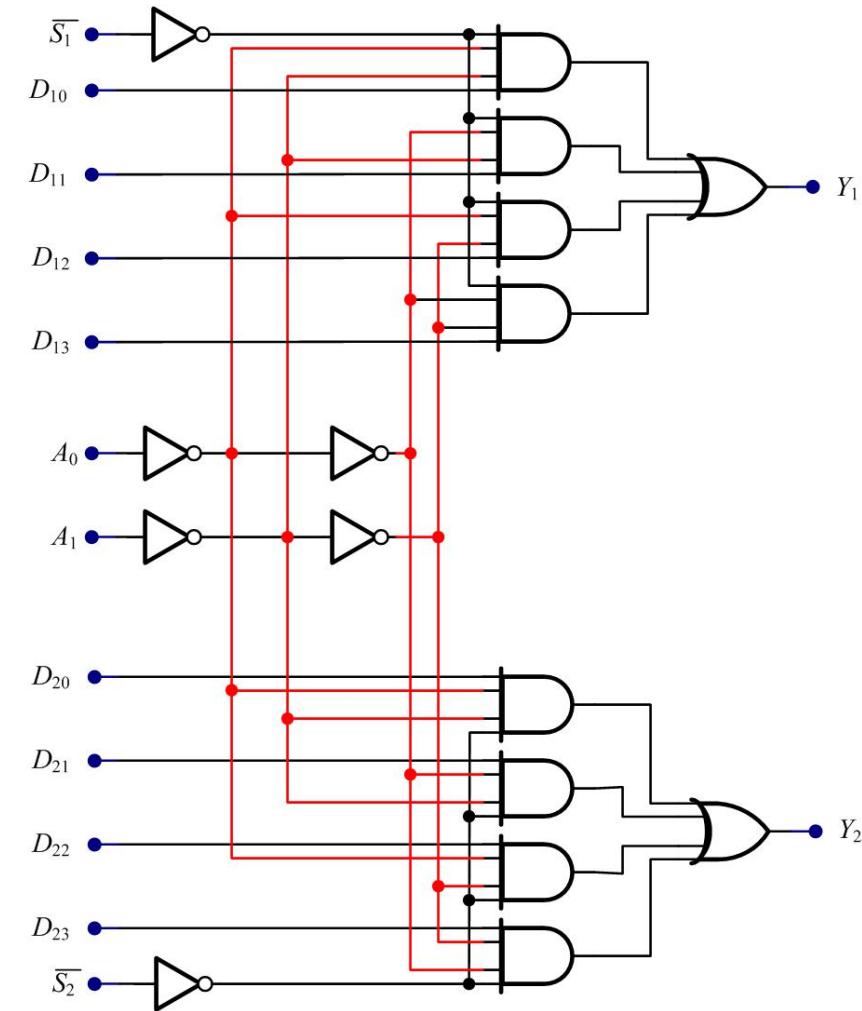
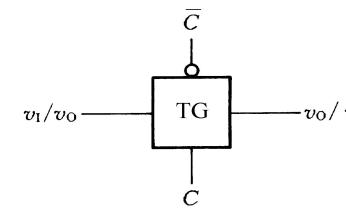
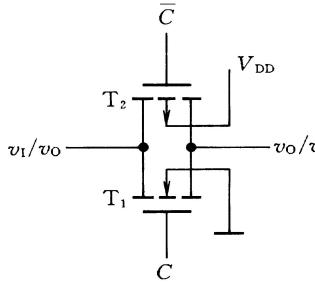
输入						输出	
\bar{S}	A_1	A_0	D_3	D_2	D_1	D_0	F
1	\times	\times	\times	\times	\times	\times	0
0	0	0	\times	\times	\times	0	0
			\times	\times	\times	1	1
	0	1	\times	\times	0	\times	0
			\times	\times	1	\times	1
	1	0	\times	0	\times	\times	0
			\times	1	\times	\times	1
1	1	\times	0	\times	\times	\times	0
		\times	1	\times	\times	\times	1



§ 4.2.3 数据选择器

- 双4选1数据选择器
74LS153

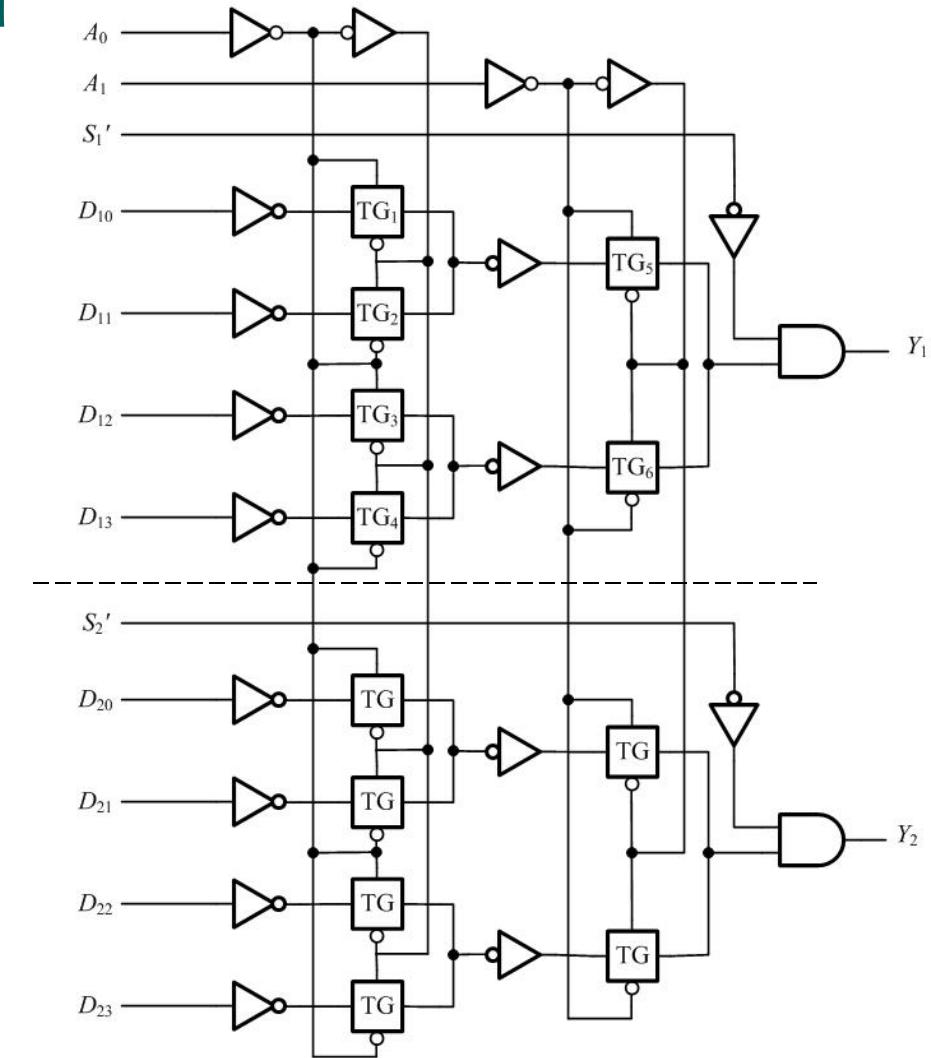
问题：利用我们之前学的哪个门电路可以很方便的实现数据选择器？





§ 4.2.3 数据选择器

- 用**传输门**构成的双4选1数据选择器
◆ 74HC153

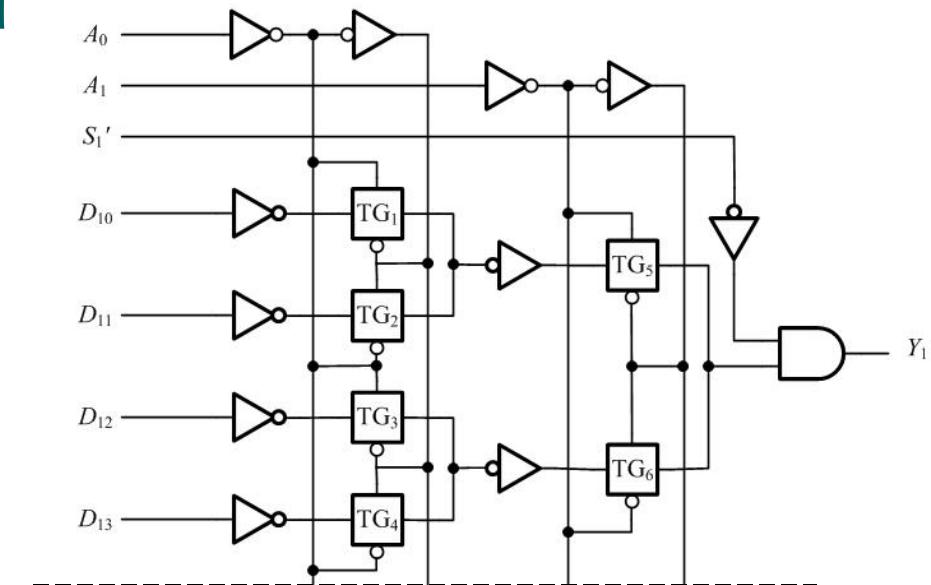


§ 4.2.3 数据选择器

- 用传输门构成的双4选1数据选择器

◆ 74HC153

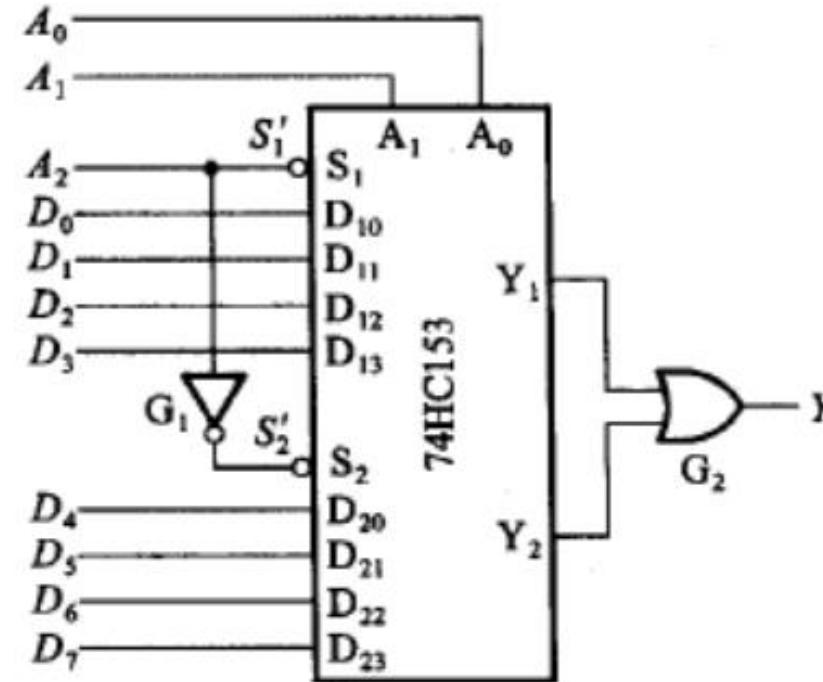
S_1'	A_1	A_0	Y_1
1	X	X	0
0	0	0	D_{10}
0	0	1	D_{11}
0	1	0	D_{12}
0	1	1	D_{13}



$$Y_1 = S_1[D_0(\overline{A}_1 \cdot \overline{A}_0) + D_1(\overline{A}_1 \cdot A_0) + D_2(A_1 \cdot \overline{A}_0) + D_3(A_1 \cdot A_0)]$$

§ 4.2.3 数据选择器

利用74HC153 两个4选1扩展成8选1?



$$\begin{aligned} Y = & (A_2'A_1'A_0')D_0 + (A_2'A_1'A_0)D_1 + (A_2'A_1A_0')D_2 + (A_2'A_1A_0)D_3 \\ & + (A_2A_1'A_0')D_4 + (A_2A_1'A_0)D_5 + (A_2A_1A_0')D_6 + (A_2A_1A_0)D_7 \end{aligned}$$

§ 4.2.3 数据选择器

■ 数据选择器的应用

➤ 实现组合逻辑函数

- 例题：用8选1数据选择器 74151 实现组合逻辑函数 F

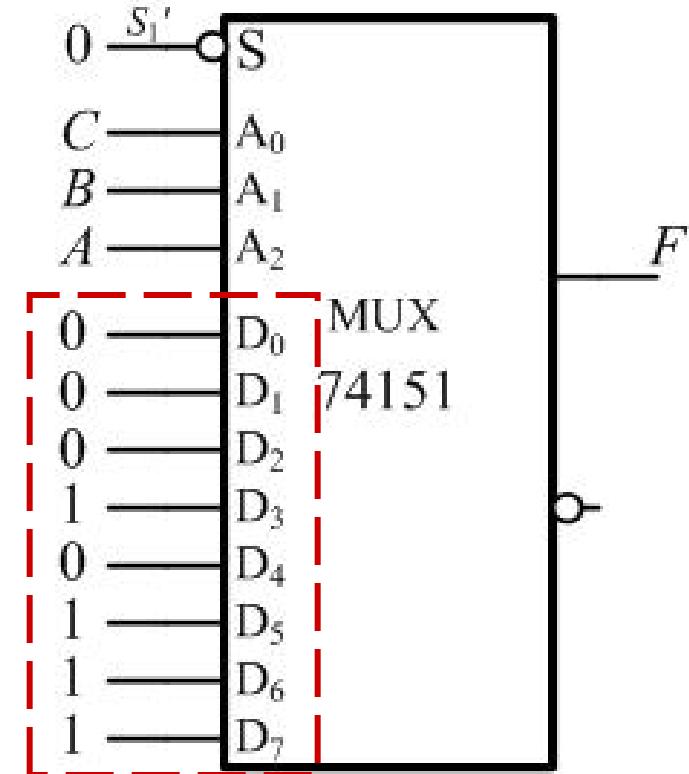
- $F = AB + BC + AC$

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

$$= m_3 + m_5 + m_6 + m_7$$

❖ 本例题中：逻辑函数的变量和地址输入变量个数相同

$$F = \overline{A}\overline{B}\overline{C}D_0 + \overline{A}\overline{B}CD_1 + \overline{A}B\overline{C}D_2 + \overline{A}BCD_3 + A\overline{B}\overline{C}D_4 + A\overline{B}CD_5 + AB\overline{C}D_6 + ABCD_7$$



§ 4.2.3 数据选择器

- 例题：用4选1数据选择器实现逻辑函数：

$$F = AB + AC + BC$$

❖ 逻辑函数的变量**多于地址输入变量个数**

对比

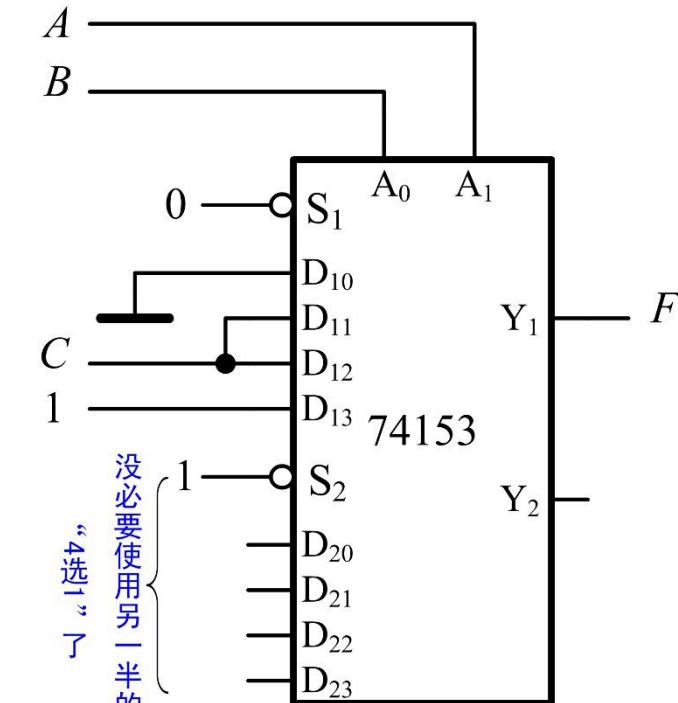
$$\begin{aligned} F &= AB + AC + BC \\ &= A\bar{B}\bar{C} + \bar{A}BC + \bar{A}\bar{B}C + ABC \\ &= \bar{A}\bar{B}C + \bar{A}\bar{B}C + AB(\bar{C} + C) \end{aligned}$$

$$F = \bar{A} \cdot \bar{B} \cdot 0 + \bar{A}\bar{B} \cdot C + \bar{A}\bar{B} \cdot C + AB \cdot 1$$

❖ 4选1选择器的输出表达式

$$F = \bar{A}_1 \bar{A}_0 D_0 + \bar{A}_1 A_0 D_1 + A_1 \bar{A}_0 D_2 + A_1 A_0 D_3$$

$$A = A_1, \quad B = A_0 \quad D_0 = 0 \quad D_1 = D_2 = C \quad D_3 = 1$$



§ 4.2.4 组合逻辑电路模块 之 数值比较器

■ 数值比较器

- 对两个位数相同的二进制整数进行数值比较

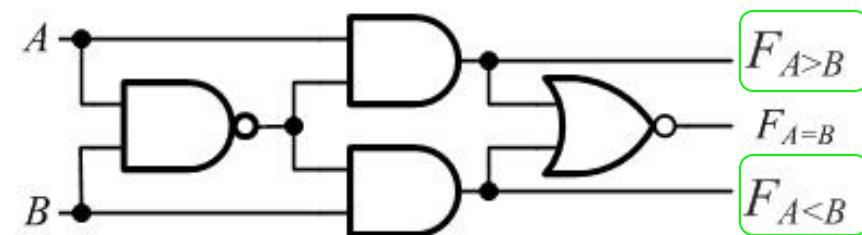
- 例如：1位数值比较器

$$F_{A>B} = A\bar{B}$$

$$F_{A<B} = \overline{A}\bar{B}$$

$$F_{A=B} = \overline{\overline{A}\bar{B}} + A\bar{B}$$

输入		输出		
A	B	$F_{A>B}$	$F_{A<B}$	$F_{A=B}$
0	0	0	0	1
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1





- 多位比较器 (如 $A_1A_0 \text{ vs } B_1B_0$) (考虑优先级)

数值输入		级联输入			输出		
$A_1\ B_1$	$A_0\ B_0$	$I_{A>B}$	$I_{A<B}$	$I_{A=B}$	$F_{A>B}$	$F_{A<B}$	$F_{A=B}$
$A_1 > B_1$	$\times\ \times$	\times	\times	\times	1	0	0
$A_1 < B_1$	$\times\ \times$	\times	\times	\times	0	1	0
$A_1 = B_1$	$A_0 > B_0$	\times	\times	\times	1	0	0
$A_1 = B_1$	$A_0 < B_0$	\times	\times	\times	0	1	0
$A_1 = B_1$	$A_0 = B_0$	1	0	0	1	0	0
$A_1 = B_1$	$A_0 = B_0$	0	1	0	0	1	0
$A_1 = B_1$	$A_0 = B_0$	0	0	1	0	0	1

$$F_{A>B} = (A_1 > B_1) + (A_1 = B_1) \cdot (A_0 > B_0) + (A_1 = B_1) \cdot (A_0 = B_0) \cdot I_{A>B}$$

$$F_{A<B} = (A_1 < B_1) + (A_1 = B_1) \cdot (A_0 < B_0) + (A_1 = B_1) \cdot (A_0 = B_0) \cdot I_{A<B}$$

$$F_{A=B} = (A_1 = B_1) \cdot (A_0 = B_0) \cdot I_{A=B}$$

- 多位比较器 —— 以2位为例

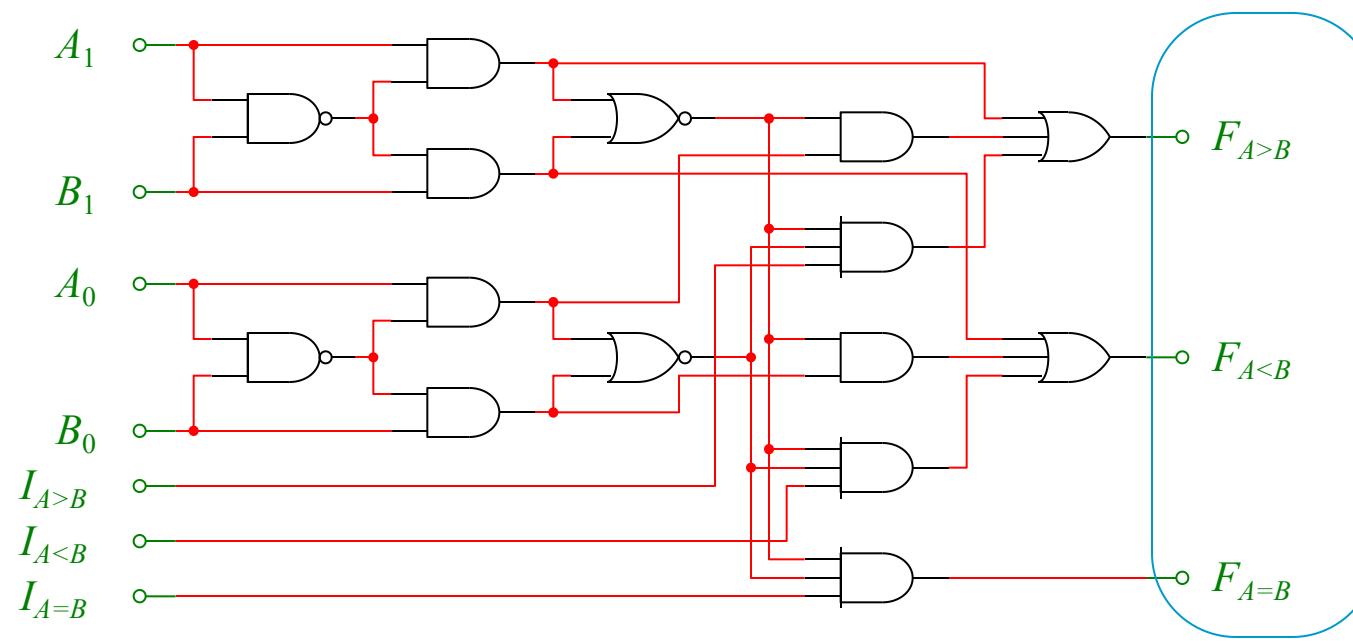
$$F_{A>B} = (A_1 > B_1) + (A_1 = B_1) \cdot (A_0 > B_0) + (A_1 = B_1) \cdot (A_0 = B_0) \cdot I_{A>B}$$

$$F_{A<B} = (A_1 < B_1) + (A_1 = B_1) \cdot (A_0 < B_0) + (A_1 = B_1) \cdot (A_0 = B_0) \cdot I_{A<B}$$

$$F_{A=B} = (A_1 = B_1) \cdot (A_0 = B_0) \cdot I_{A=B}$$

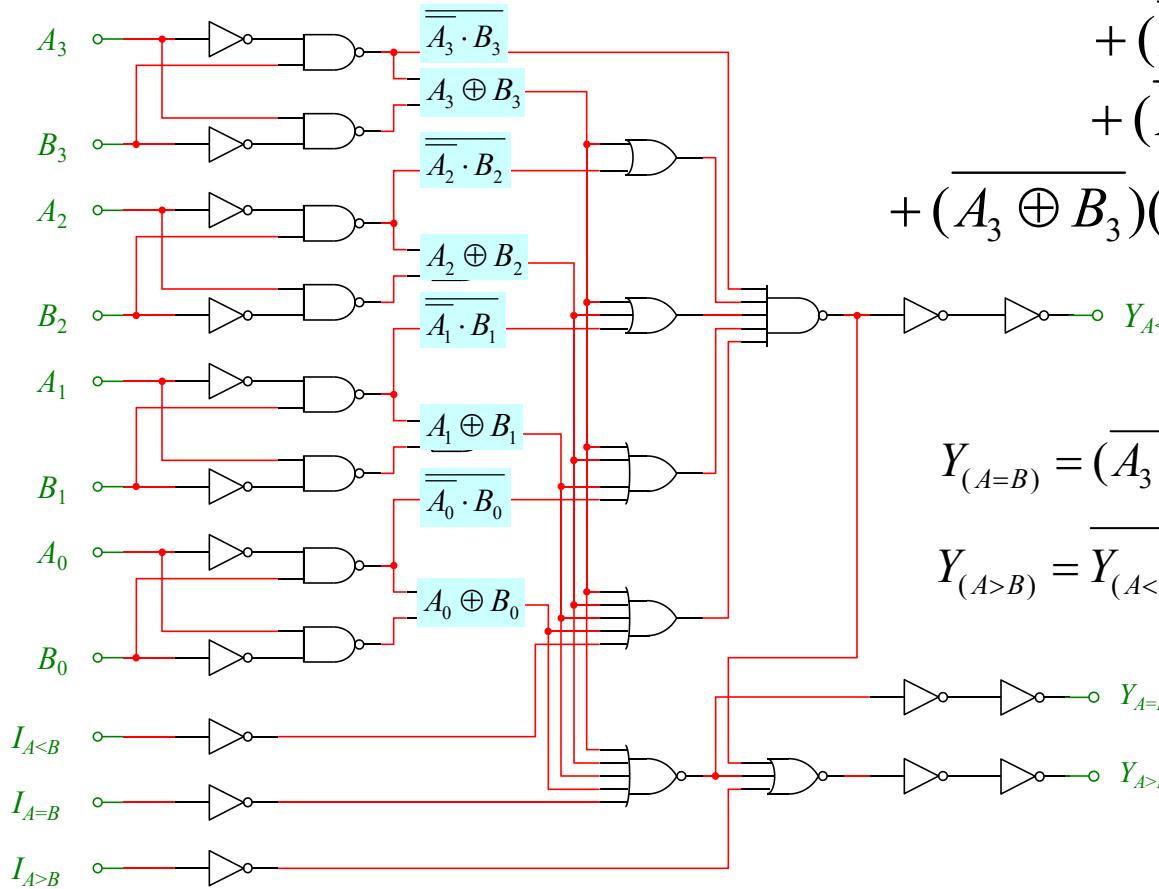
组合逻辑设计

简化：注意到 $A>B$, $A<B$ 和 $A=B$ 并不是互不相关的



● 多位比较器

—— 例如：4位数码比较器



$$\begin{aligned}
 Y_{(A < B)} &= \overline{\overline{A_3} \cdot B_3} + (\overline{A_3} \oplus B_3) \overline{\overline{A_2} \cdot B_2} \\
 &\quad + (\overline{A_3} \oplus B_3)(\overline{A_2} \oplus B_2) \overline{A_1} B_1 \\
 &\quad + (\overline{A_3} \oplus B_3)(\overline{A_2} \oplus B_2)(\overline{A_1} \oplus B_1) \overline{A_0} B_0 \\
 &\quad + (\overline{A_3} \oplus B_3)(\overline{A_2} \oplus B_2)(\overline{A_1} \oplus B_1)(\overline{A_0} \oplus B_0) I_{(A < B)}
 \end{aligned}$$

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

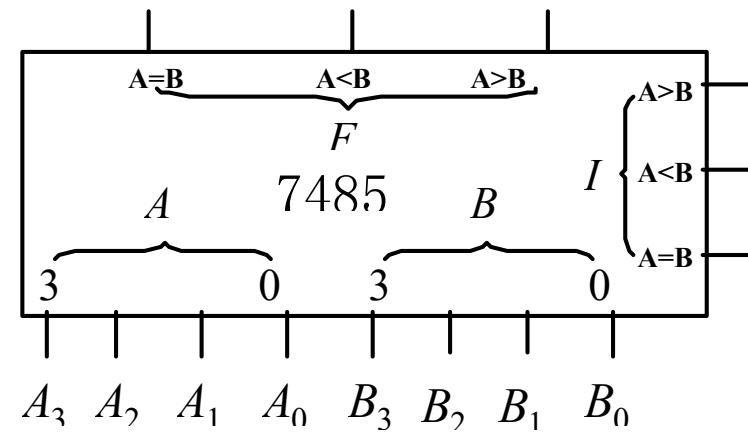
$$Y_{(A > B)} = \overline{Y_{(A < B)}} + Y_{(A = B)} + \overline{I_{(A > B)}}$$



§ 4.2.4 数值比较器

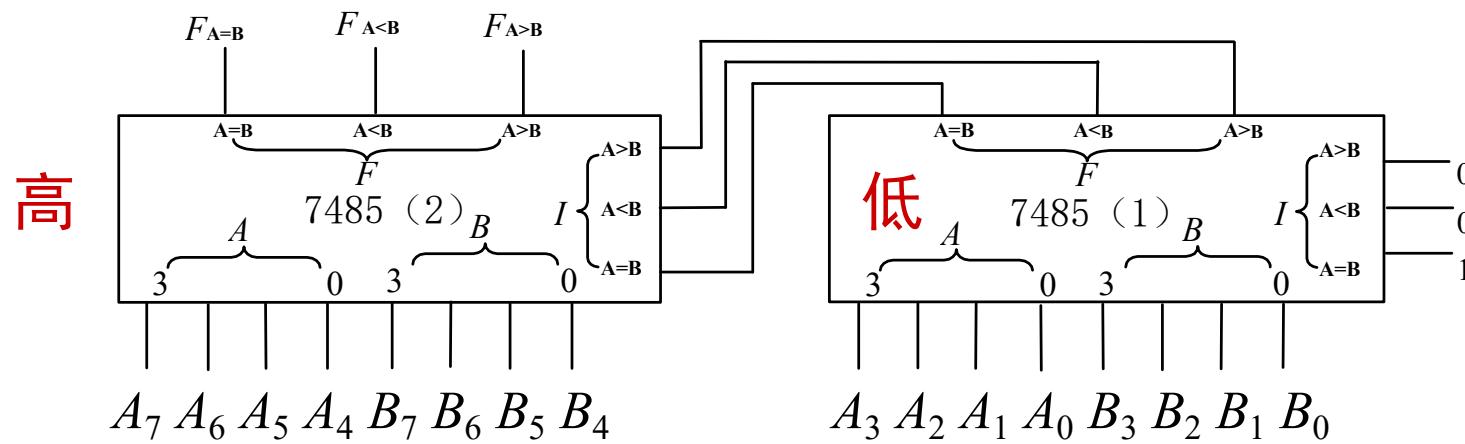
■ 数值比较器的扩展

- 例题：利用集成数值比较器7845进行
 - 比较器的位数串联扩展



§ 4.2.4 数值比较器

- 比较器的位数串联扩展
——通用的串联扩展方法



低位模块将比较结果馈入高位模块的级联输入端

§ 4.2.5 组合逻辑电路模块之加法器

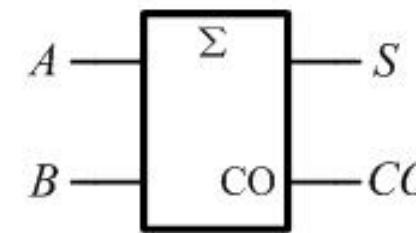
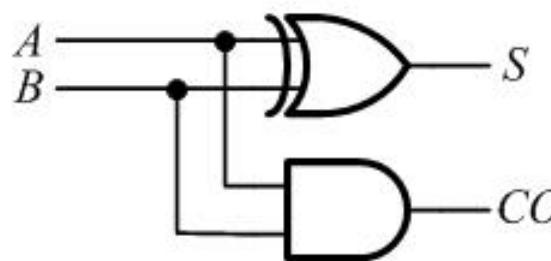
■ 加法器

➤ 半加器

✓ 仅考虑加数和被加数的运算

$$S = \overline{A}B + A\overline{B} = A \oplus B \quad CO = AB$$

输入		输出	
被加	加数	和数	进位
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



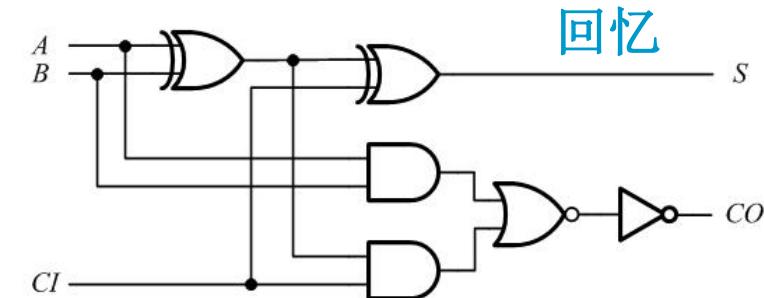
§ 4.2.5 加法器

➤ 全加器

✓ 考虑加数、被加数和相邻低位的进位的运算

全加器的真值表

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



$$S = A \oplus B \oplus CI$$

$$CO = AB + (A \oplus B) \cdot CI$$

对比...

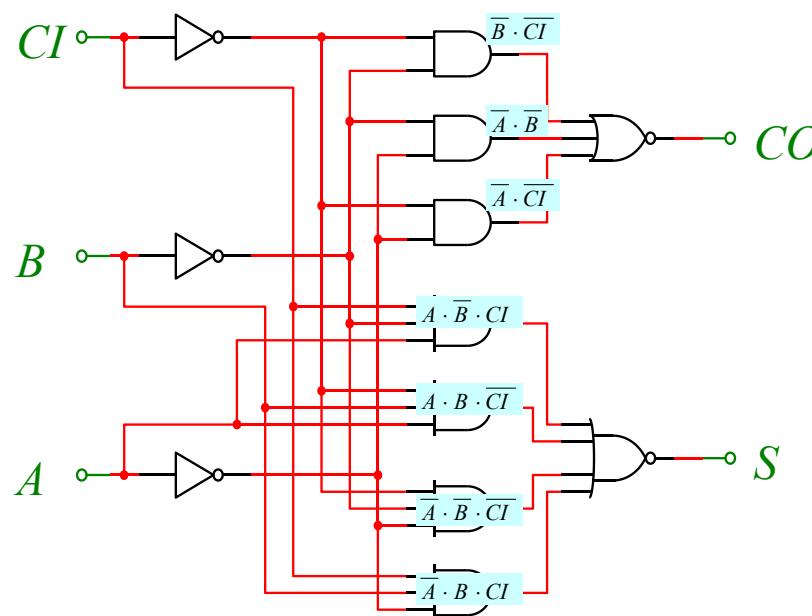
$$S = \overline{\overline{A} \cdot \overline{B} \cdot \overline{CI}} + \overline{A} \cdot B \cdot CI + A \cdot \overline{B} \cdot CI + A \cdot B \cdot \overline{CI}$$

$$CO = \overline{\overline{A} \cdot \overline{B}} + \overline{B} \cdot \overline{CI} + \overline{A} \cdot \overline{CI}$$

➤ 全加器

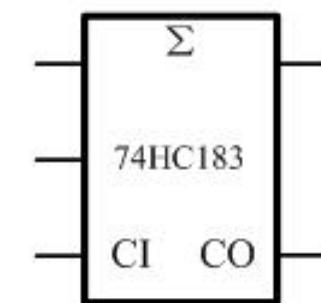
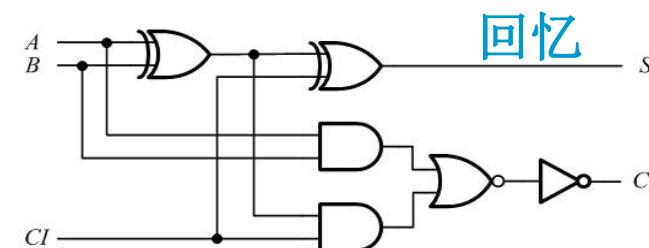
$$S = \overline{\overline{A} \cdot \overline{B} \cdot \overline{CI}} + \overline{ABC}I + A\overline{B}CI + ABC\overline{I}$$

$$CO = \overline{\overline{A} \cdot \overline{B}} + \overline{B} \cdot \overline{CI} + \overline{A} \cdot \overline{CI}$$



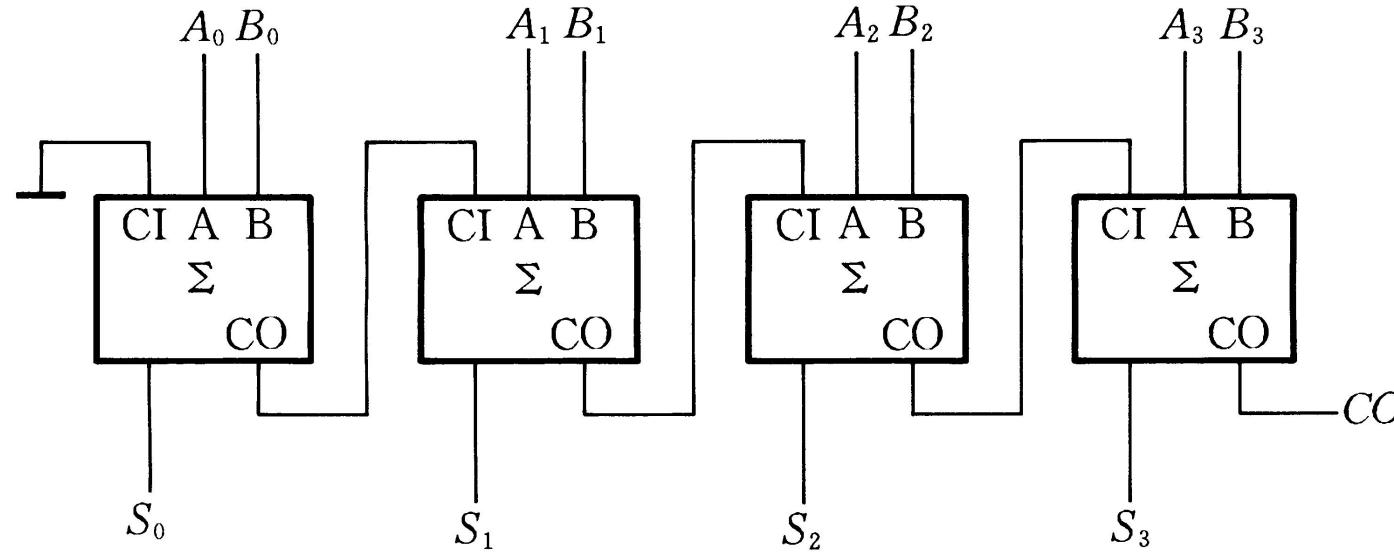
全加器的真值表

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



§ 4.2.5 加法器

➤ 多位数串行进位加法器



- ✓ 缺点：运算速度慢
- ✓ 事实上：第 i 位进位信号，是 $\{A_0, A_1, \dots, A_{i-1}\}$, $\{B_0, B_1, \dots, B_{i-1}\}$ 的组合逻辑函数；
- 超前进位加法器：Carry Look-ahead

§ 4.2.5 加法器

➤ 快速进位集成4位加法器74283

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

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

令 $G_i = A_i B_i$

$$C_1 = \overline{A_1 + B_1} + \overline{A_1 B_1} \cdot \overline{A_0 + B_0} + \overline{A_1 B_1} \cdot \overline{A_0 B_0} \cdot \overline{CI}$$

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

$$C_i = G_i + P_i C_{i-1}$$

例如：

$$C_0 = \overline{A_0 + B_0} + \overline{A_0 B_0} \cdot \overline{CI}$$

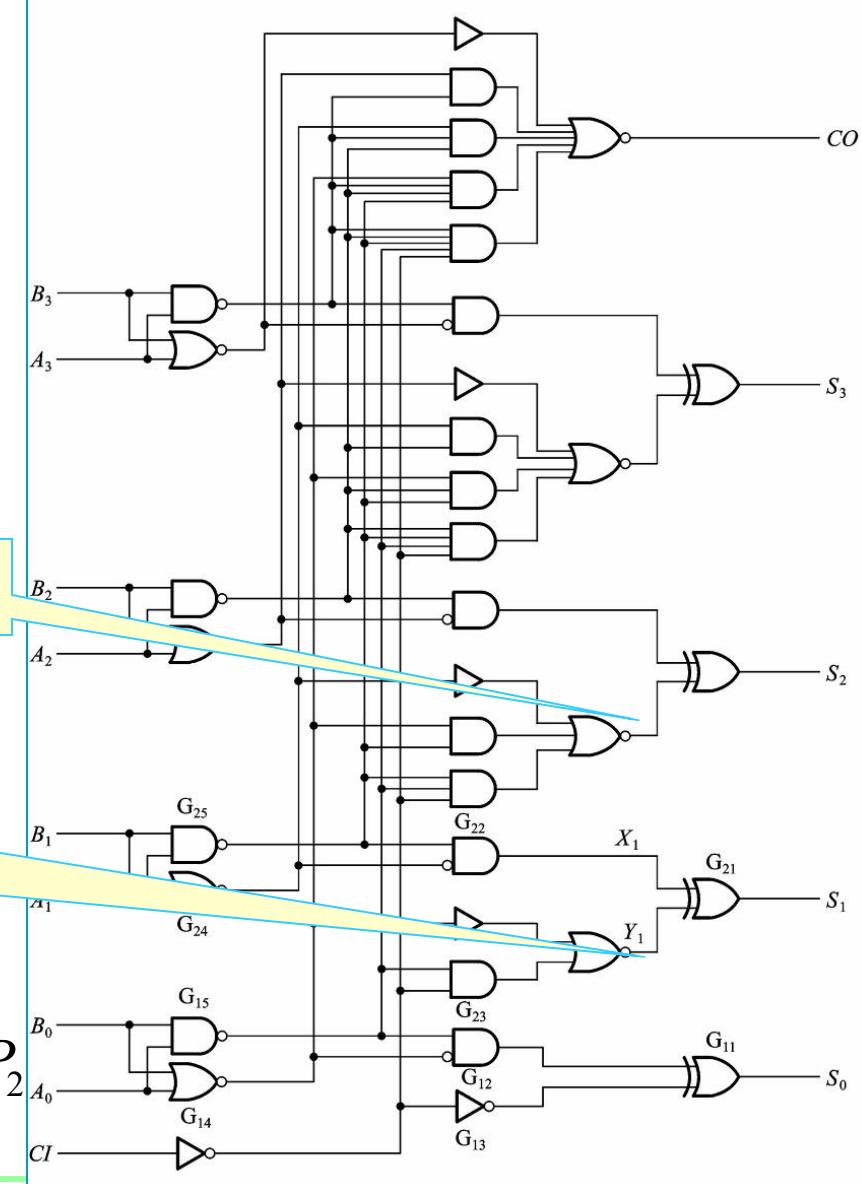
$$C_0 = G_0 + P_0 CI$$

$$C_1 = G_1 + P_1 C_0 = G_1 + P_1 G_0 + P_1 P_0 CI$$

$$C_2 = G_2 + P_2 C_1 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2$$

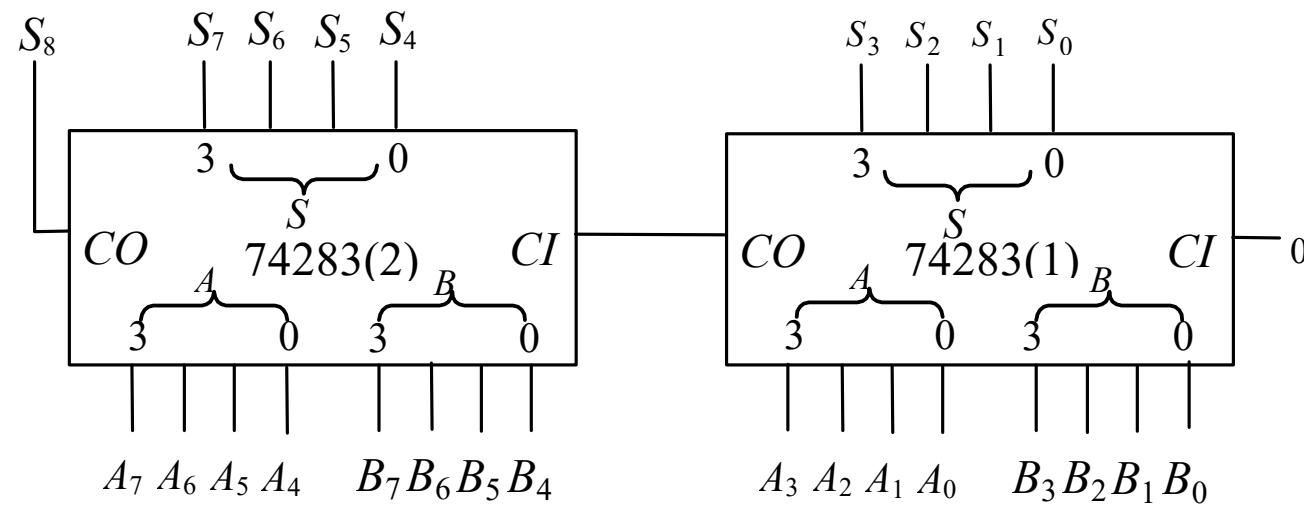
.....

$$CO = G_3 + P_3 C_2 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 CI$$



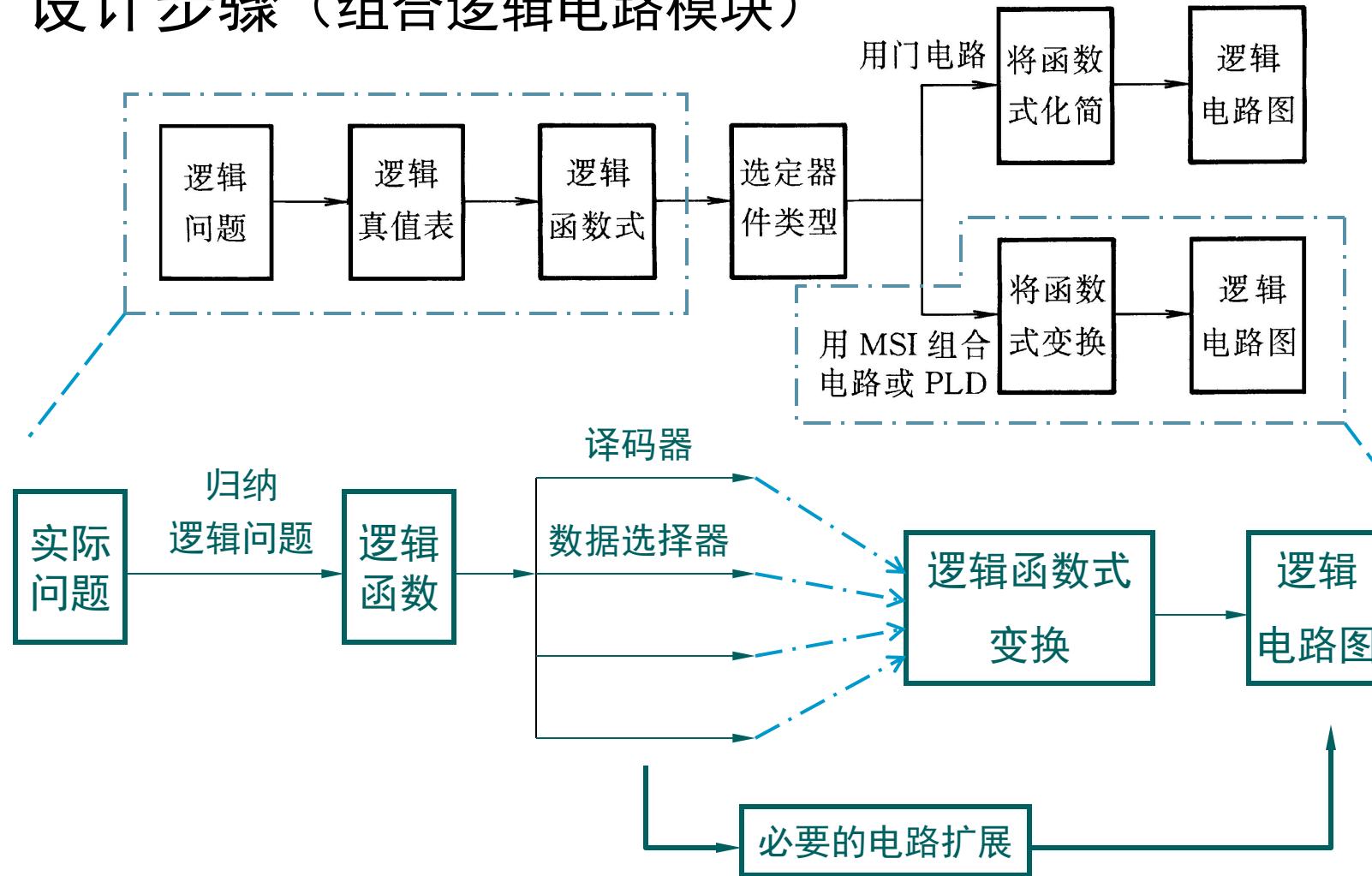
§ 4.2.5 加法器

■ 加法器的级联



归纳：组合逻辑电路设计

■ 设计步骤（组合逻辑电路模块）





第四章 组合逻辑电路

§ 4.1 组合逻辑电路的分析与设计

§ 4.2 组合逻辑电路模块及其应用

§ 4.3 组合逻辑电路中的竞争与冒险



§ 4.3 组合逻辑电路中的竞争与冒险现象

在前面的组合逻辑分析与设计中，考虑稳态的输入情况；而组合逻辑电路的工作正确性，还要考虑逻辑转换时的正确性...

- 产生竞争冒险的原因
- 冒险现象的识别
- 冒险现象的消除

§ 4.3 竞争——冒险现象

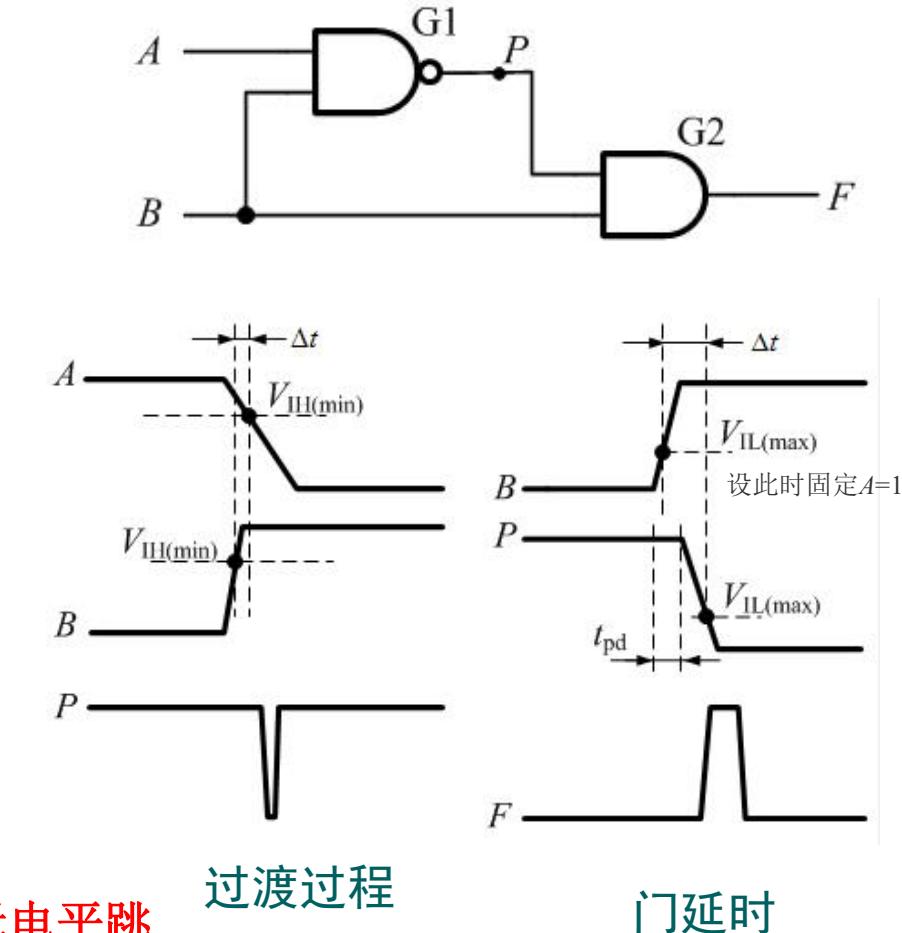
■ 产生竞争冒险的原因

➤ 例如：

$$P = \overline{AB} \quad F = P \cdot B = \overline{AB} \cdot B$$

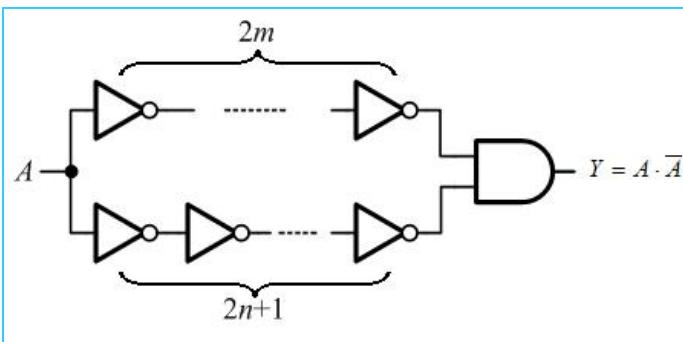
- ✓ 稳态下，正常工作；但是
- ✓ 信号变化时的**过渡过程不一致**和门电路的**传输延时**的存在，使电路产生瞬间错误输出。该现象称为**竞争-冒险**。

门电路两个输入信号“同时”向相反的逻辑电平跳变（一个从1变0，另一个从0变1）的现象叫作竞争



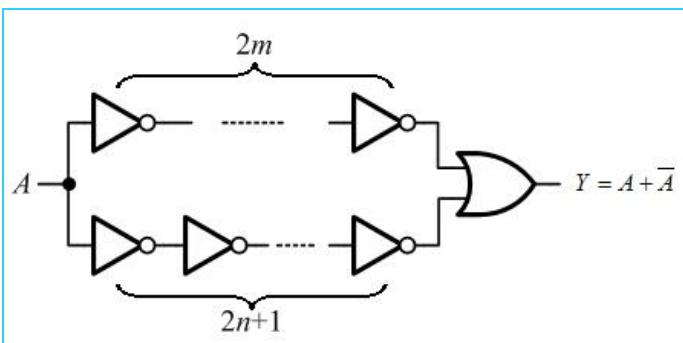
§ 4.3 竞争——冒险现象

- 冒险现象的识别 因为竞争在输出端产生尖峰脉冲的现象
- 某些逻辑变量取特定值时，表达式能转换为：



$$Y = A \bar{A}$$

✓ 存在“1”型冒险

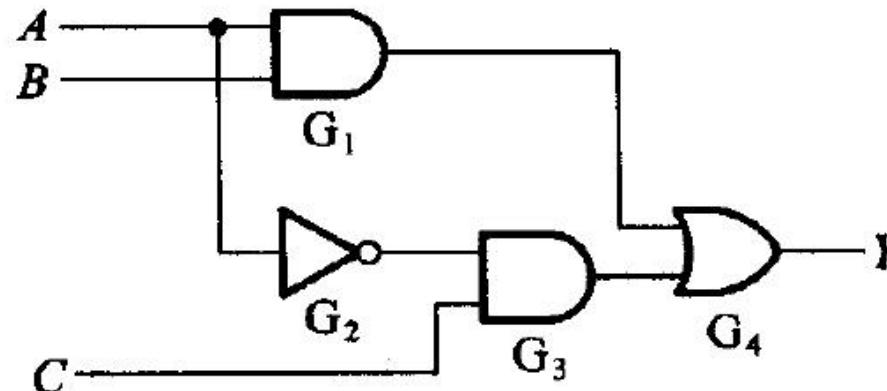


$$Y = A + \bar{A}$$

✓ 存在“0”型冒险

§ 4.3 竞争——冒险现象

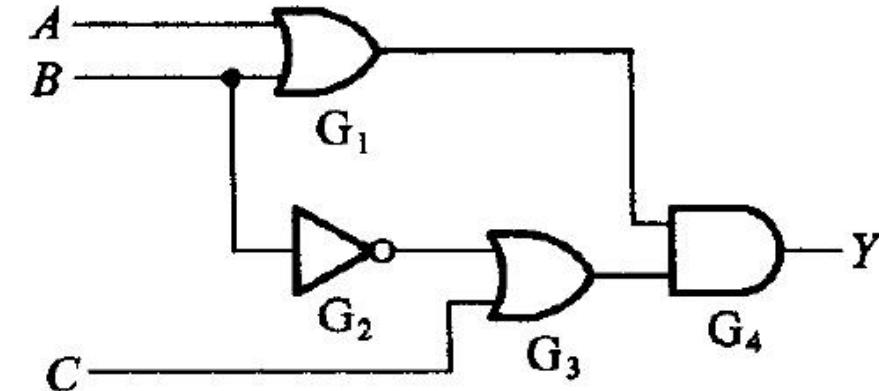
■ 冒险现象的识别



$$Y = AB + A'C$$

当 $B = C = 1$ 时, 上式将成为

$$Y = A + A'$$



$$Y = (A + B) \cdot (B' + C)$$

在 $A = C = 0$ 的条件下, 上式简化为

$$Y = B \cdot B'$$

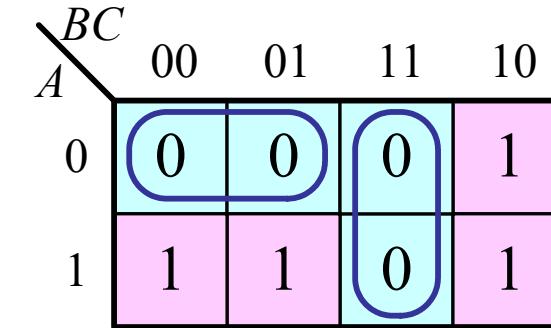
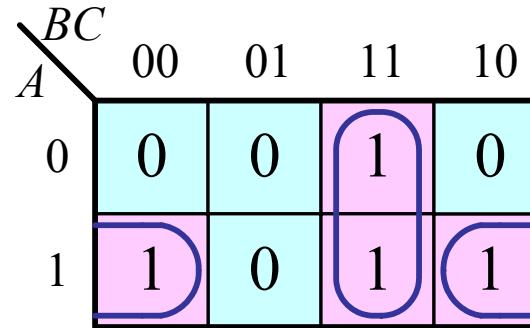
§ 4.3 竞争——冒险现象

■ 冒险现象的识别

$$F = A\bar{C} + BC \quad A=B=1 \text{ 时}, \quad F = C + \bar{C} \quad \checkmark \text{ 存在冒险}$$

$$F = (A+B)(\bar{B}+\bar{C}) \quad A=0, \quad C=1 \text{ 时}, \quad F = B \cdot \bar{B} \quad \checkmark \text{ 存在冒险}$$

\checkmark 通过卡诺图识别冒险



\checkmark 卡诺图上的圈相切，且相切处无其它圈包含

§ 4.3 竞争——冒险现象

■ 冒险现象的消除

- ✓ 增加冗余项

$$F = A\bar{C} + BC + AB$$

$$F = (A + B)(\bar{B} + \bar{C})(A + \bar{C})$$

		BC	00	01	11	10
		A	0	0	1	0
0	0	0	0	1	0	0
	1	1	0	1	1	0

		BC	00	01	11	10
		A	0	0	0	1
0	0	0	0	0	1	0
	1	1	1	0	0	1

- ✓ 变换逻辑式

$$F = (A + B)(\bar{B} + \bar{C}) = A\bar{B} + A\bar{C} + B\bar{C}$$

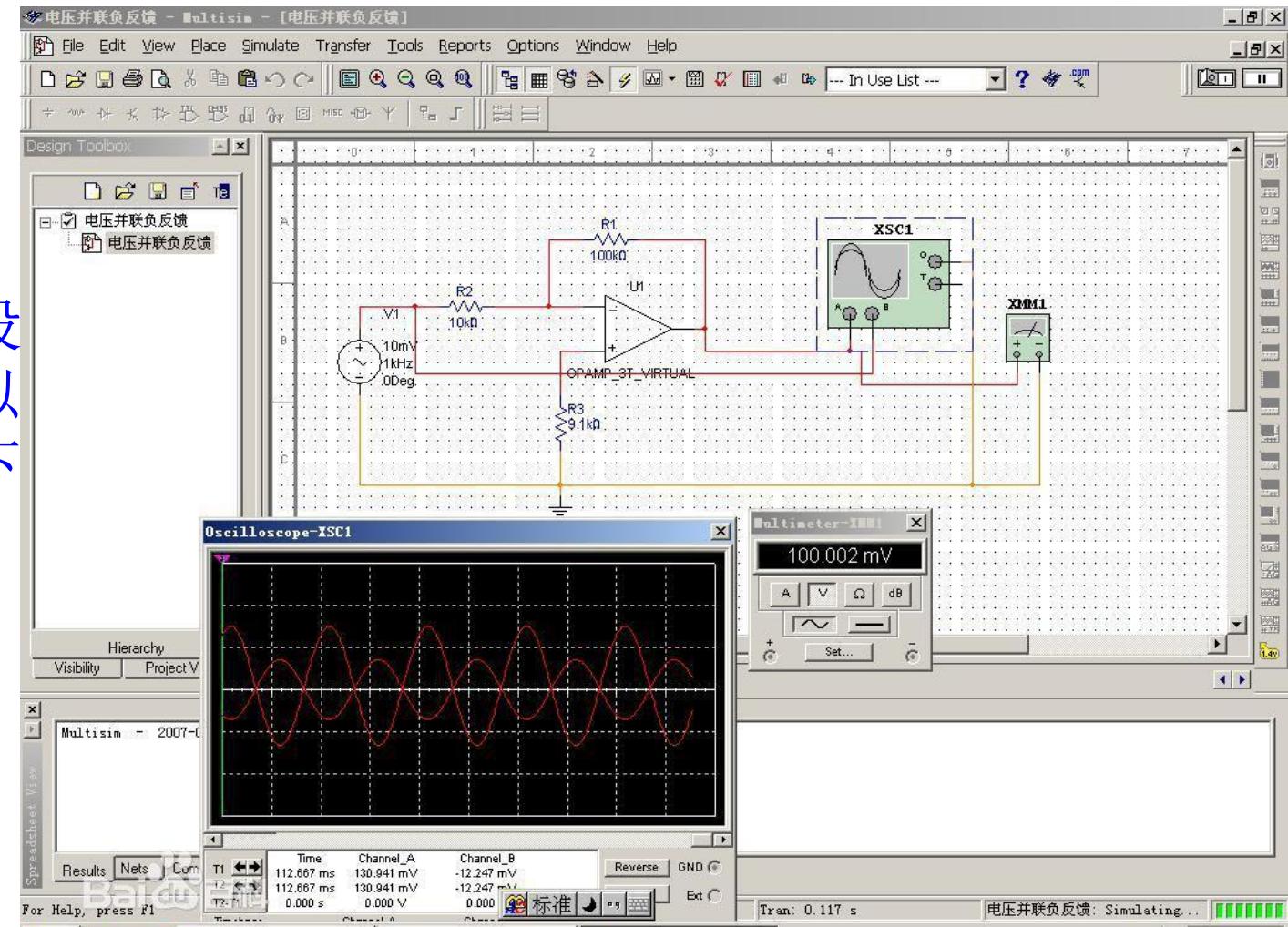
- ✓ 增加选通信号

- ✓ 增加输出滤波电容



补充：用Multisim分析组合逻辑电路

- Multisim是美国国家仪器有限公司推出的以Windows为基础的仿真工具，适用于板级的模拟/数字电路板的设计工作。它包含了电路原理图的图形输入、电路硬件描述语言输入方式，具有丰富的仿真分析能力。



有兴趣参加电子设计竞赛的同学可以课余时间学习一下



本章小结

- 组合逻辑电路的特点：输出只决定于该时刻各输入组合，与电路的原状态无关，电路中无记忆单元，没有反馈通路。
- 分析步骤：写出各输出端的逻辑表达式→化简和变换逻辑表达式→列出真值表→确定功能；
- 设计步骤：根据设计要求列出真值表→写出逻辑表达式（或填写卡诺图）→逻辑化简和变换→画出逻辑图。
- 用译码器和数据选择器实现组合逻辑功能。
- 常用的中规模组合逻辑模块包括：编码器、译码器、数据选择器、数值比较器、加法器…，为了增加使用的灵活性和便于功能扩展，设置了输入、输出使能端或扩展端。
- 组合电路中存在竞争和冒险现象，可通过表达式和卡诺图识别冒险现象，并应采取措施消除冒险现象。



第四章 课程作业

阎石 老师 第五版

- 4.3;
- 4.6;
- 4.8; 4.10
- 4.12; 4.17
- 4.28;
- 4.32 ;