

数 字 逻 辑

Digital Logic Circuit

丁 贤 庆

ahhfdxq@163.com

Home work (P218)

- 1、周一晚上7:00-8:00在新安学堂105房间，答疑。
回答作业或者课本中疑难问题。
- 2、本周三（周四）的课上会有一次10分钟的随堂测验。
- 3、本周二晚上7:00开始有第一次实验，地点：电气楼505房间
- 4、今天的作业（不用抄题目）：
 - 4.4.20
 - 4.4.22
 - 4.4.25
 - 4.4.27 (1)

第一次实验时间

地点：电气楼505房间

物联网2班：本周二晚上19：---20:50

物联网1班：本周三晚上19：---20:50

计科1班：下周一晚上19：---20:50

计科2班：下周一上午8：---9:50

计科3班：下周二晚上19：---20:50

计科4班：下周三下午16：---17:50

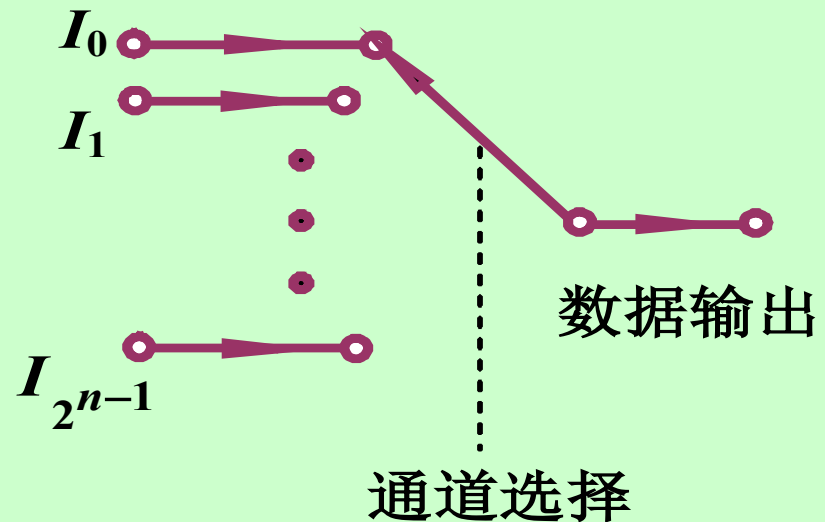
计科5班：下周三晚上19：---20:50

4.4.3 数据选择器 Multiplexers (Data Selectors)

1、数据选择器的定义与功能

数据选择器：能实现**数据选择功能**的逻辑电路。它的作用相当于**多个输入**的**单刀多掷开关**，又称“**多路开关**”。

数据选择的功能：在通道选择信号的作用下，将**多个通道**的数据分时传送到**公共的数据通道**上去的。



2选1数据选择器

简化真值表

完整的真值表

1位地址码
输入端

数据
输入
端

1路数据输出端

Y=1的情况共有两种情况：

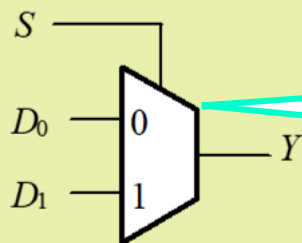
(1) $SD_0=01$

(2) $SD_1=11$

$$Y = \bar{S}D_0 + SD_1$$

选择 输入	输 出
S	Y
0	D_0
1	D_1

输入			输出
S	D_0	D_1	Y
0	1	x	1
0	0	x	0
1	x	1	1
1	x	0	0
...			



逻辑符号

4选1数据选择器

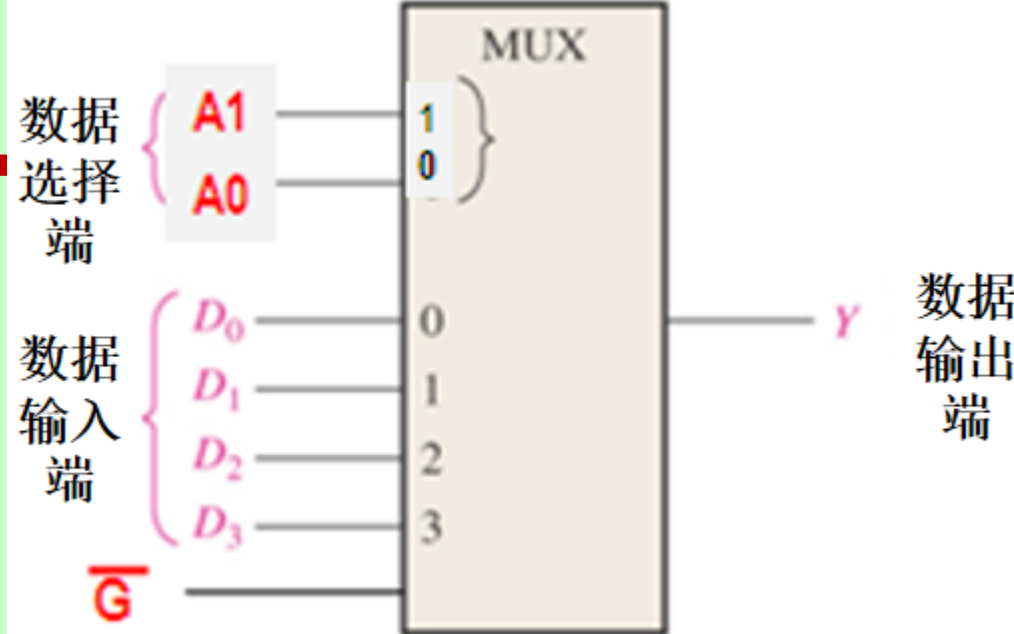
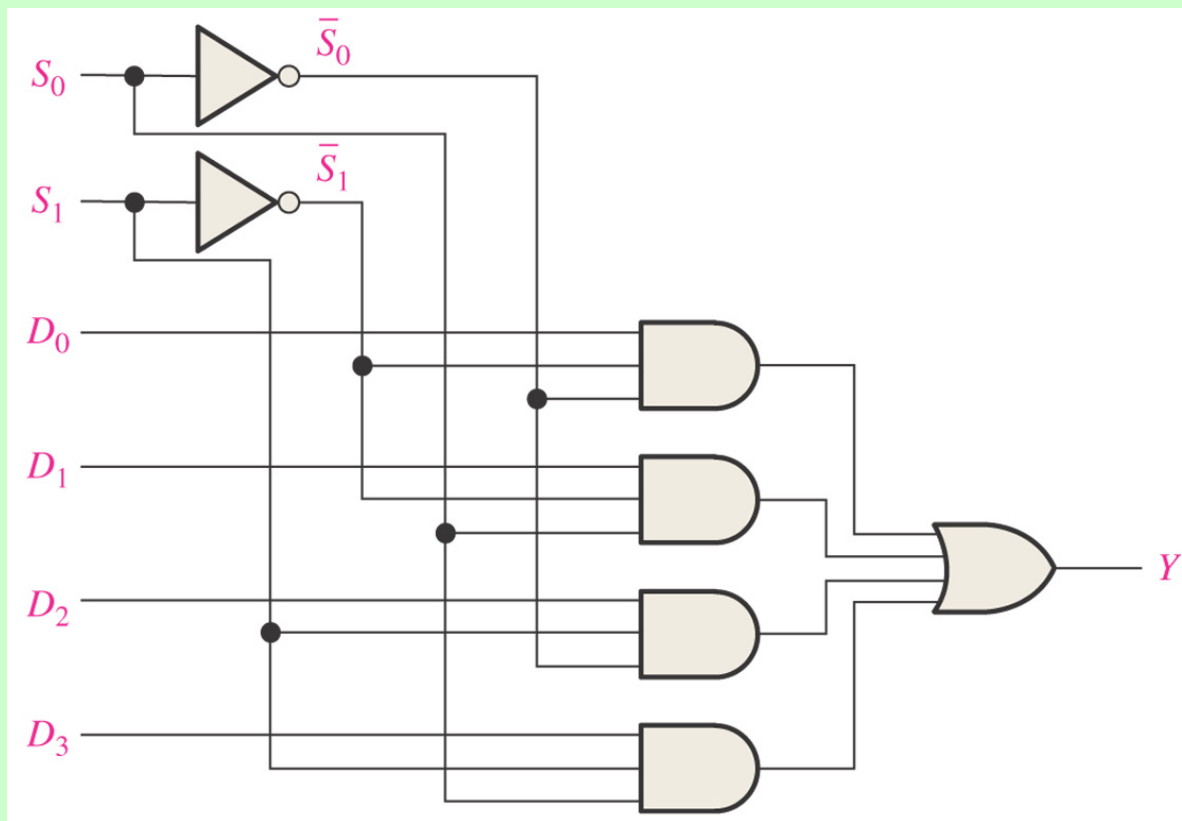


表 4.3.1 4 选 1 数据选择器功能表

输 入				输 出			
G	A_1	A_0	D_3	D_2	D_1	D_0	Y
1	×	×	×	×	×	×	0
0	0	0	×	×	×	0	0
			×	×	×	1	1
	0	1	×	×	0	×	0
			×	×	1	×	1
	1	0	×	0	×	×	0
			×	1	×	×	1
	1	1	0	×	×	×	0
			1	×	×	×	1

4选1数据选择器



$$Y = \bar{S}_1\bar{S}_0D_0 + \bar{S}_1S_0D_1 + S_1\bar{S}_0D_2 + S_1S_0D_3$$

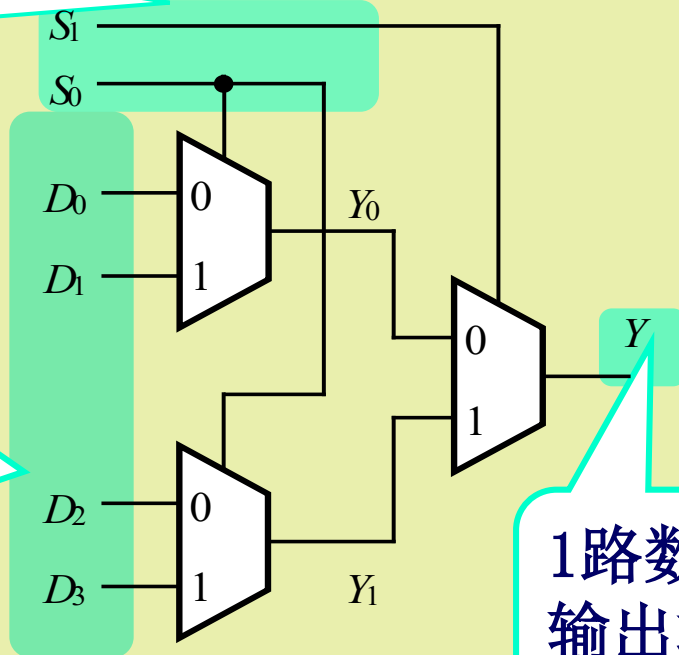
4选1数据选择器

(1) 逻辑电路

由3个2选1数据选择器构成4选1数据选择器。

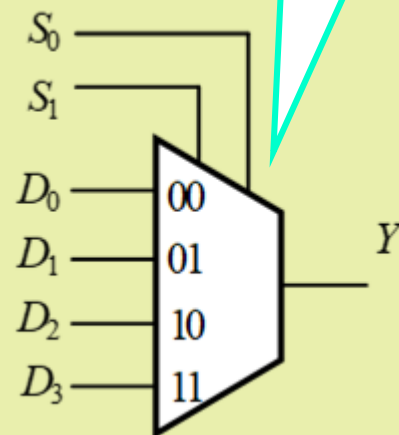
2 位地址
码输入端

数据
输入端

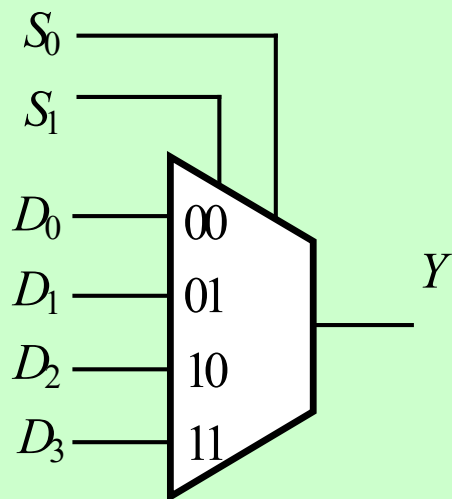


1路数据
输出端

逻辑符号



(2) 工作原理及逻辑功能



简化真值表

选择输入		输出
S_1	S_0	
0	0	
0	1	
1	0	
1	1	

$$Y = \overline{S_1} \overline{S_0} D_0 + \overline{S_1} S_0 D_1 + S_1 \overline{S_0} D_2 + S_1 S_0 D_3$$

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

这两个公式常用

(3) 数据选择器实现逻辑函数

例4.4.8 试用数据选择器实现下列逻辑函数

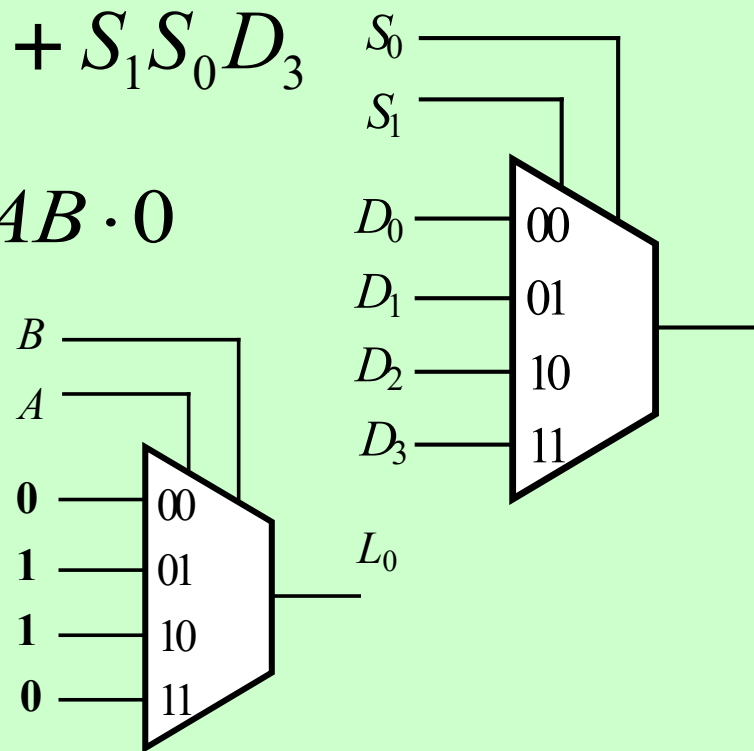
① 用4选1数据选择器实现 $L_0 = \overline{A}B + A\overline{B}$

② 用2选1数据选择器和必要的逻辑门实现 $L_1 = AB + A\overline{C} + BC$

$$Y = \overline{S_1}\overline{S_0}D_0 + \overline{S_1}S_0D_1 + S_1\overline{S_0}D_2 + S_1S_0D_3$$

$$L_0 = \overline{A}\overline{B} \cdot 0 + \overline{A}B \cdot 1 + A\overline{B} \cdot 1 + AB \cdot 0$$

① 当 $S_1 = A, S_0 = B,$
 $D_0 = D_3 = 0, D_1 = D_2 = 1$



② 用2选1数据选择器和必要的逻辑门实现 $L_1 = AB + \overline{A}\overline{C} + BC$

2选1数据选择器只有1个选通端接输入 A ，表达式有3个变量。因此数据端需要输入2个变量。考察真值表 B 、 C 与 L_1 的关系。

$$Y = \overline{S}D_0 + SD_1$$

当 $S=A$ 时，

$$\begin{aligned} L_1 &= AB + \overline{A}\overline{C} + BC \cdot (A + \overline{A}) \\ &= A \cdot (B + \overline{C} + BC) + \overline{A} \cdot BC \\ &= A \cdot (B + \overline{C}) + \overline{A} \cdot BC \end{aligned}$$

令 $S=A$ ，

$D_0=BC$

$D_1=B + \overline{C}$ 时，

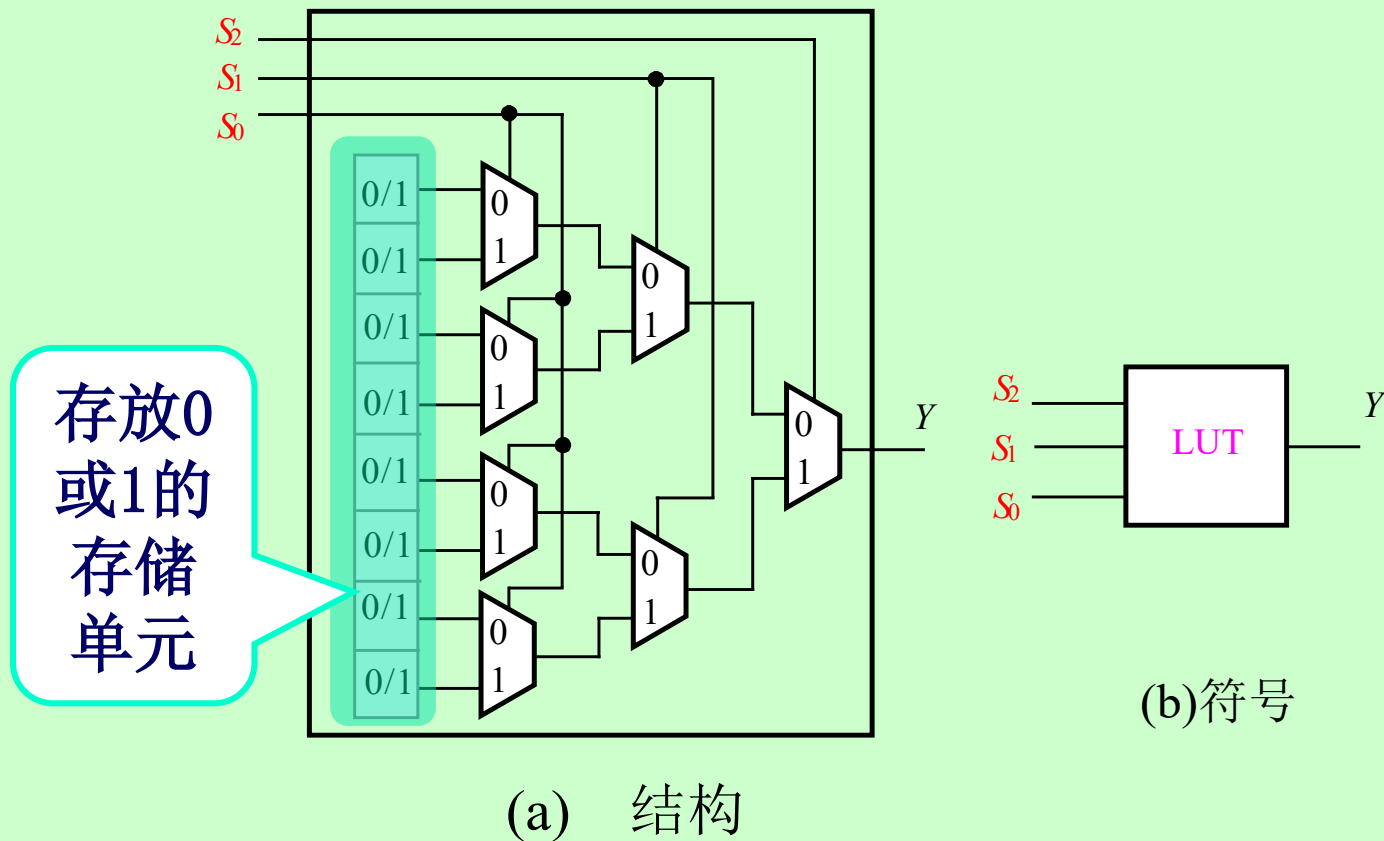
$Y=L_1$

输入	输出
S	Y
0	$Y=D_0$
1	$Y=D_1$

输 入			输 出	
	B	C	L_1	
	0	0	0	$L_1=BC$
	0	1	0	
	1	0	0	
	1	1	1	
	0	0	1	$L_1 = B + \overline{C}$
	0	1	0	
	1	0	1	
	1	1	1	

(4) 数据选择器构成查找表LUT

构成FPGA基本单元的逻辑块主要是查找表LUT。LUT实质是一个小规模的存储器，以真值表的形式实现给定的逻辑函数。3输入LUT的结构及逻辑符号如图。



用查找表LUT实现逻辑函数

$$L_1 = AB + \overline{A}\overline{C} + BC$$

用LUT实现逻辑函数，变量A、B、C接选择输入端，对存储单元进行编程。

根据前面例题已知

$$L_1 = AB(C + \overline{C}) + A(B + \overline{B})\overline{C} + (A + \overline{A})BC$$

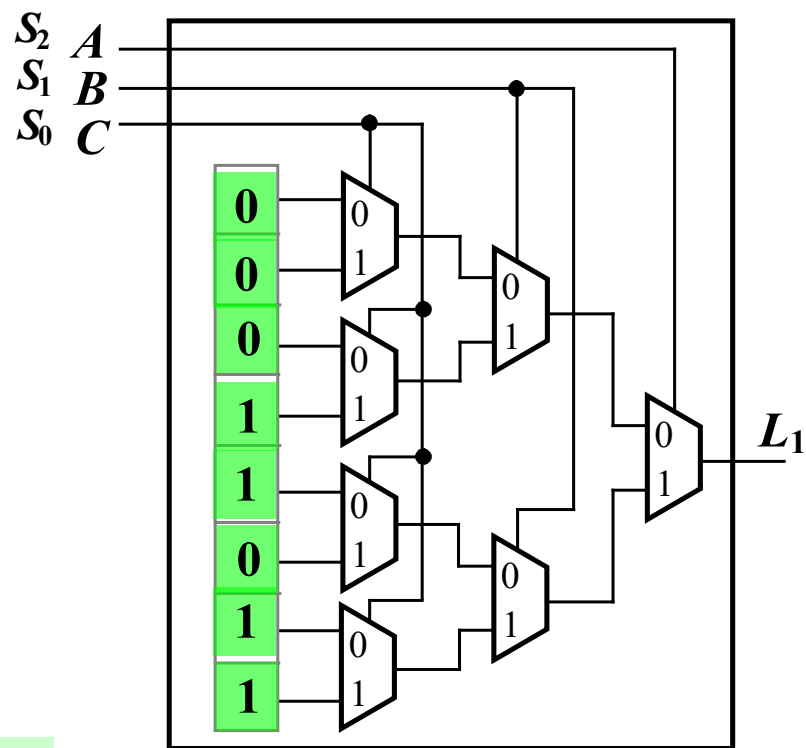
$$= m_3 + m_4 + m_6 + m_7$$

$$= 0 \cdot m_0 + 0 \cdot m_1 + 0 \cdot m_2 + 1 \cdot m_3 + 1 \cdot m_4 + 0 \cdot m_5 + 1 \cdot m_6 + 1 \cdot m_7$$

8选1数据选择器，输出表达式Y为(见下页PPT)：

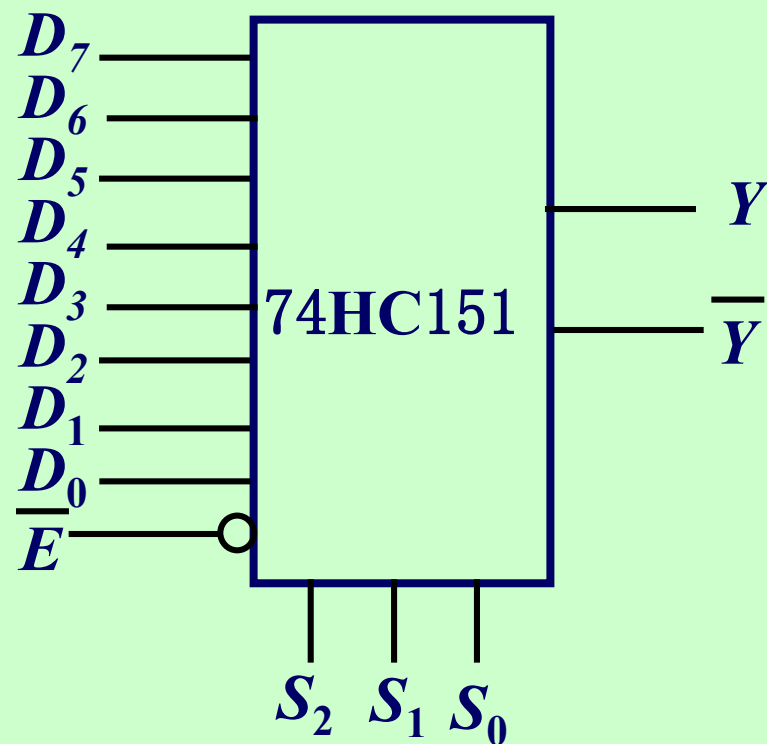
$$Y = D_0m_0 + D_1m_1 + D_2m_2 + D_3m_3 + D_4m_4 + D_5m_5 + D_6m_6 + D_7m_7$$

$$D_0 = D_1 = D_2 = D_5 = 0; D_3 = D_4 = D_6 = D_7 = 1$$



(6) 集成电路数据选择器

8选1数据选择器74HC151



74HC151逻辑符号

74HC151的功能表

$$Y = D_0m_0 + D_1m_1 + D_2m_2 + D_3m_3 + D_4m_4 + D_5m_5 + D_6m_6 + D_7m_7$$

•当 $\overline{E}=1$ 时, $Y=0$ 。

•当 $\overline{E}=0$ 时

Y=1的情况共有八种情况, 可以写出对应表达式:

(1) $S_2S_1S_0D_0=0001$ (5) $S_2S_1S_0D_4=1001$

(2) $S_2S_1S_0D_1=0011$ (6) $S_2S_1S_0D_5=1011$

(3) $S_2S_1S_0D_2=0101$ (7) $S_2S_1S_0D_6=1101$

(4) $S_2S_1S_0D_3=0111$ (8) $S_2S_1S_0D_7=1111$

$$Y = \overline{S_2}\overline{S_1}\overline{S_0}D_0 + \overline{S_2}\overline{S_1}S_0D_1 + \overline{S_2}S_1\overline{S_0}D_2$$

$$+ \overline{S_2}S_1S_0D_3 + S_2\overline{S_1}\overline{S_0}D_4 + S_2\overline{S_1}S_0D_5$$

$$+ S_2S_1\overline{S_0}D_6 + S_2S_1S_0D_7$$

$$Y = \sum_{i=0}^7 D_i m_i$$

输 入				输 出	
使能 E	选 择 S ₂ S ₁ S ₀			Y	\overline{Y}
1	X	X	X	L	H
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	$\overline{D_7}$

4.4.4 数值比较器

Comparators

本节了解

数值比较器：对两个1位数字进行比较（ A 、 B ），以判断其大小的逻辑电路。 Determine whether two numbers are equal

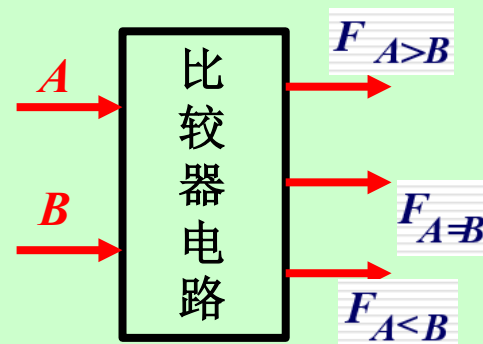
1. 1位数值比较器(设计)

输入：两个一位二进制数 A 、 B 。

输出： $F_{A>B}=1$ ，表示 A 大于 B

$F_{A<B}=1$ ，表示 A 小于 B

$F_{A=B}=1$ ，表示 A 等于 B



1, If $A=1$ and $B=0$, number A is greater than number B ;

2, If $A=0$ and $B=1$, number A is less than number B ;

3, If $A=B$, number A is equal to number B .

1位数值比较器

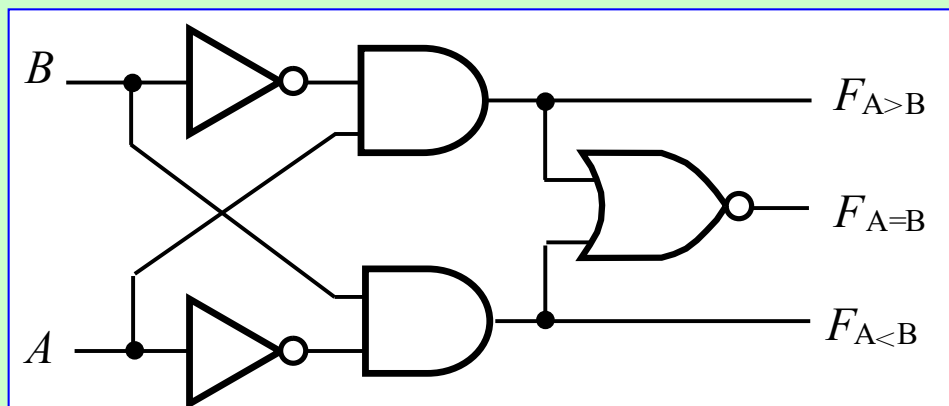
一位数值比较器真值表

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

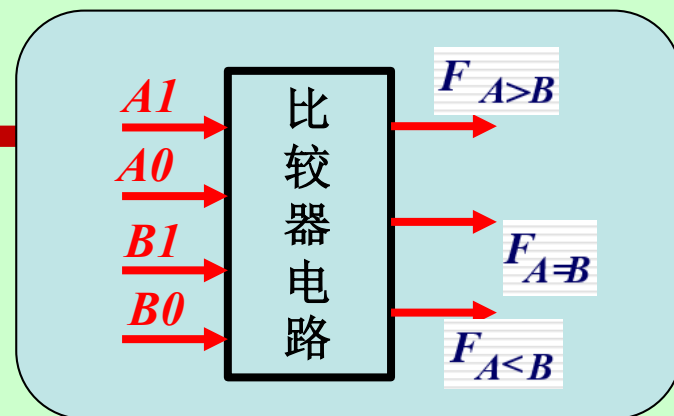
$$F_{A<B} = \bar{A} B$$

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

输 入		输 出		
A	B	$F_{A>B}$	$F_{A<B}$	$F_{A=B}$
0	0			
0	1			
1	0			
1	1			



2、2 位数值比较器:



比较两个2 位二进制数的大小的电路

输入：两个2位二进制数 $A=A_1A_0$ 、 $B=B_1B_0$

能否用1位数值比较器设计两位数值比较器？

用一位数值比较器设计多位数值比较器的原则

当高位 (A_1 、 B_1) 不相等时，无需比较低位 (A_0 、 B_0)，高位比较的结果就是两个数的比较结果。

当高位相等时，两数的比较结果由低位比较的结果决定。

真值表

输 入				输 出		
A_1	B_1	A_0	B_0	$F_{A>B}$	$F_{A<B}$	$F_{A=B}$
$A_1 > B_1$		\times				
$A_1 < B_1$		\times				
$A_1 = B_1$	$A_0 > B_0$					
$A_1 = B_1$	$A_0 < B_0$					
$A_1 = B_1$	$A_0 = B_0$					

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

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

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

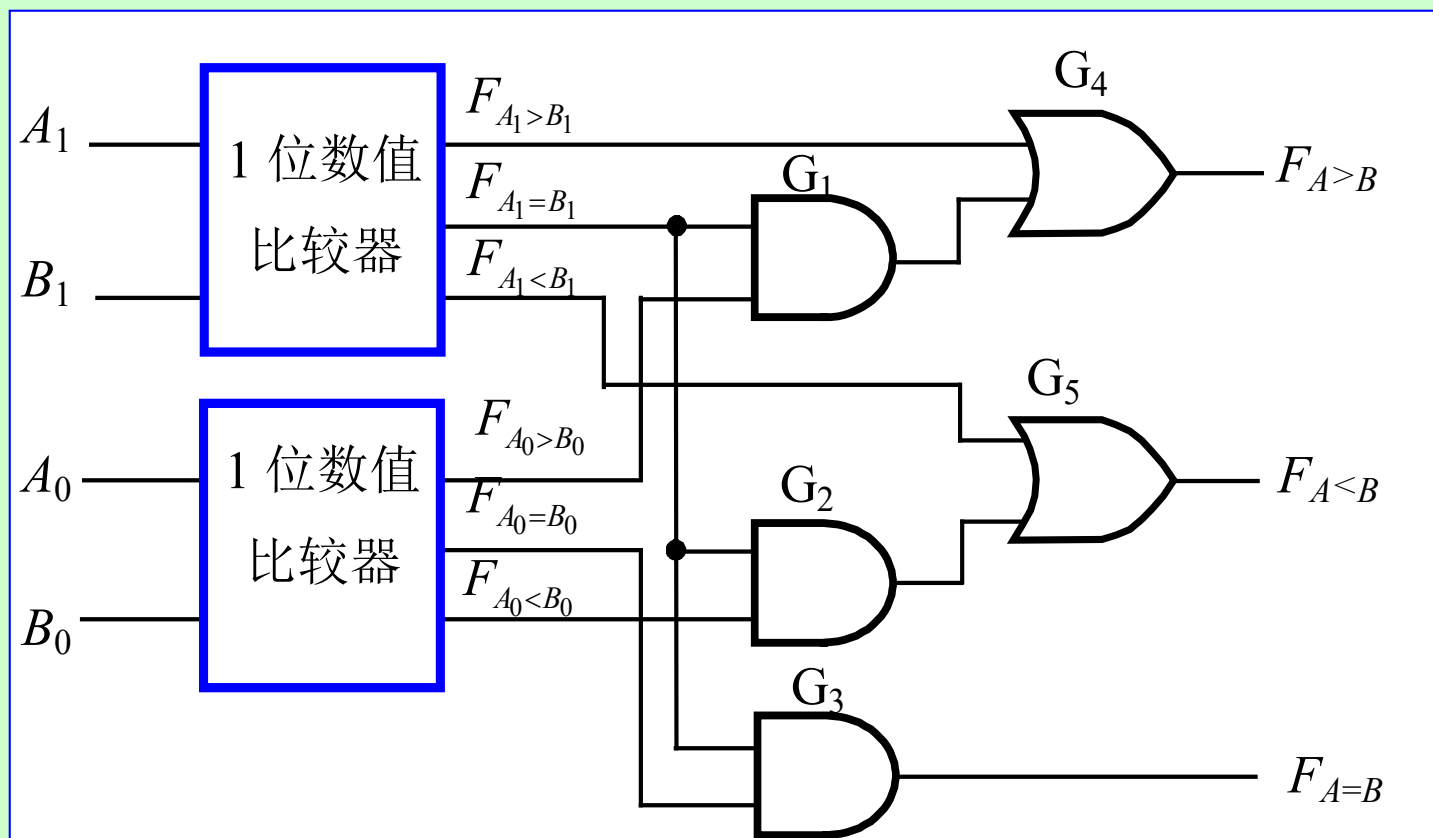
注意：上述不是真正的逻辑函数表达式，只示意逻辑关系。

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

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

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

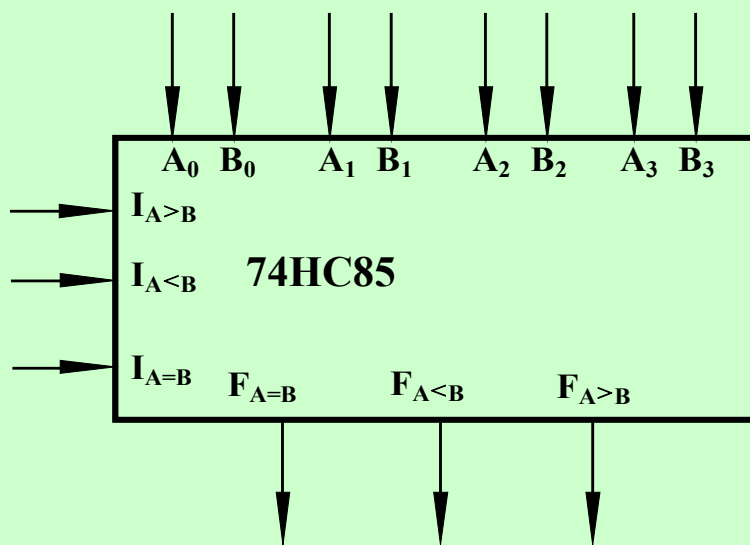
两位数值比较器逻辑图



3、集成数值比较器

(1.) 集成数值比较器74HC85的功能

74HC85是四位数值比较器，其工作原理和两位数值比较器相同。



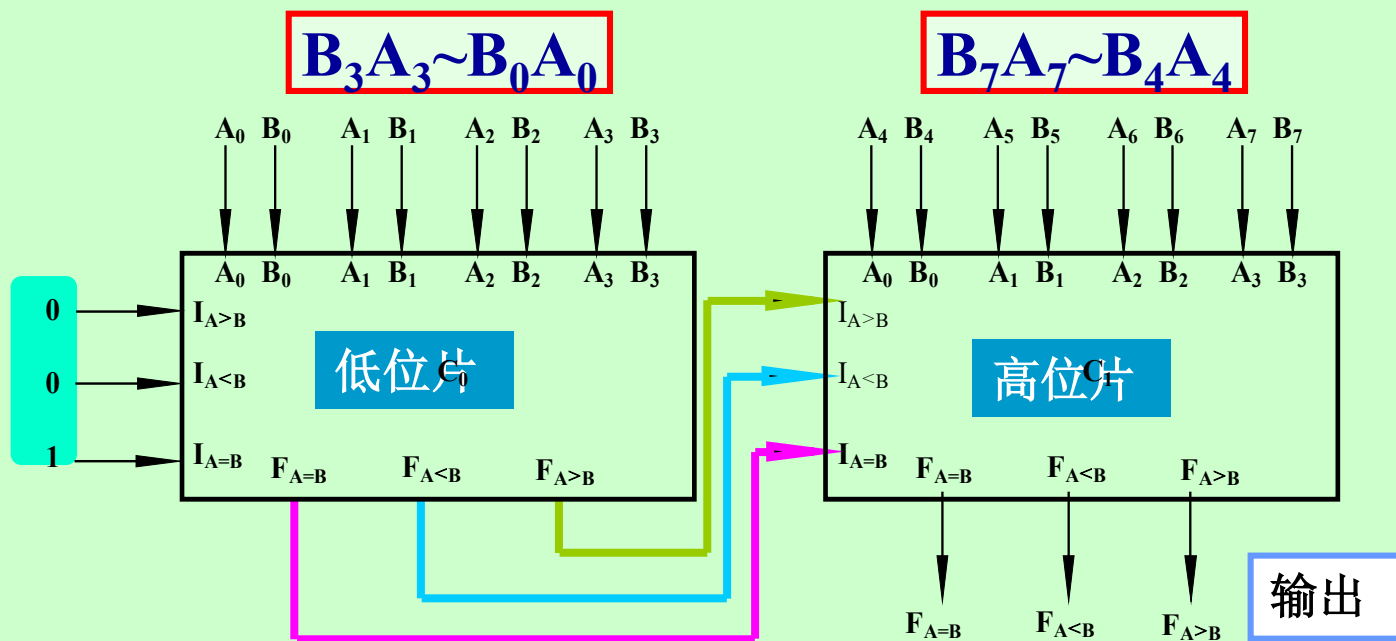
74HC85的示意框图

4、集成数值比较器的位数扩展（了解）

用两片74HC85组成8位数值比较器（串联扩展方式）。

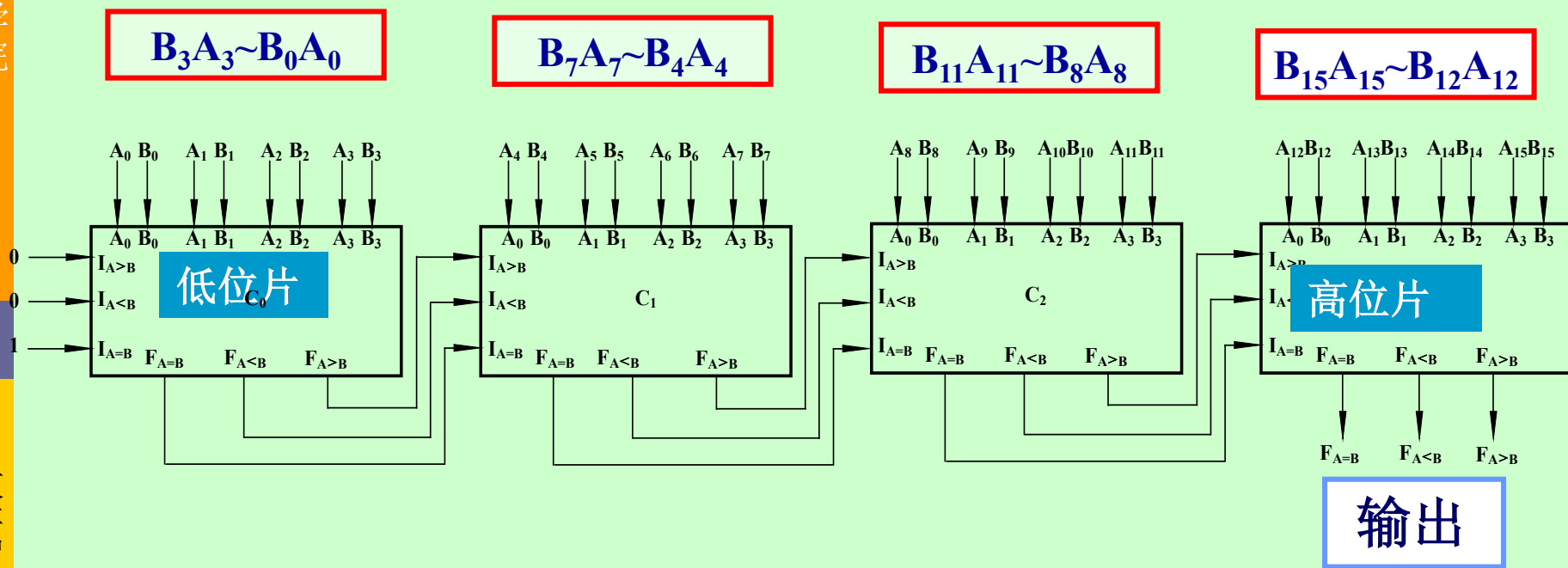
输入： $A=A_7A_6A_5A_4A_3A_2A_1A_0$ $B=B_7B_6B_5B_4B_3B_2B_1B_0$

输出： $F_{A>B}$ $F_{A<B}$ $F_{A=B}$



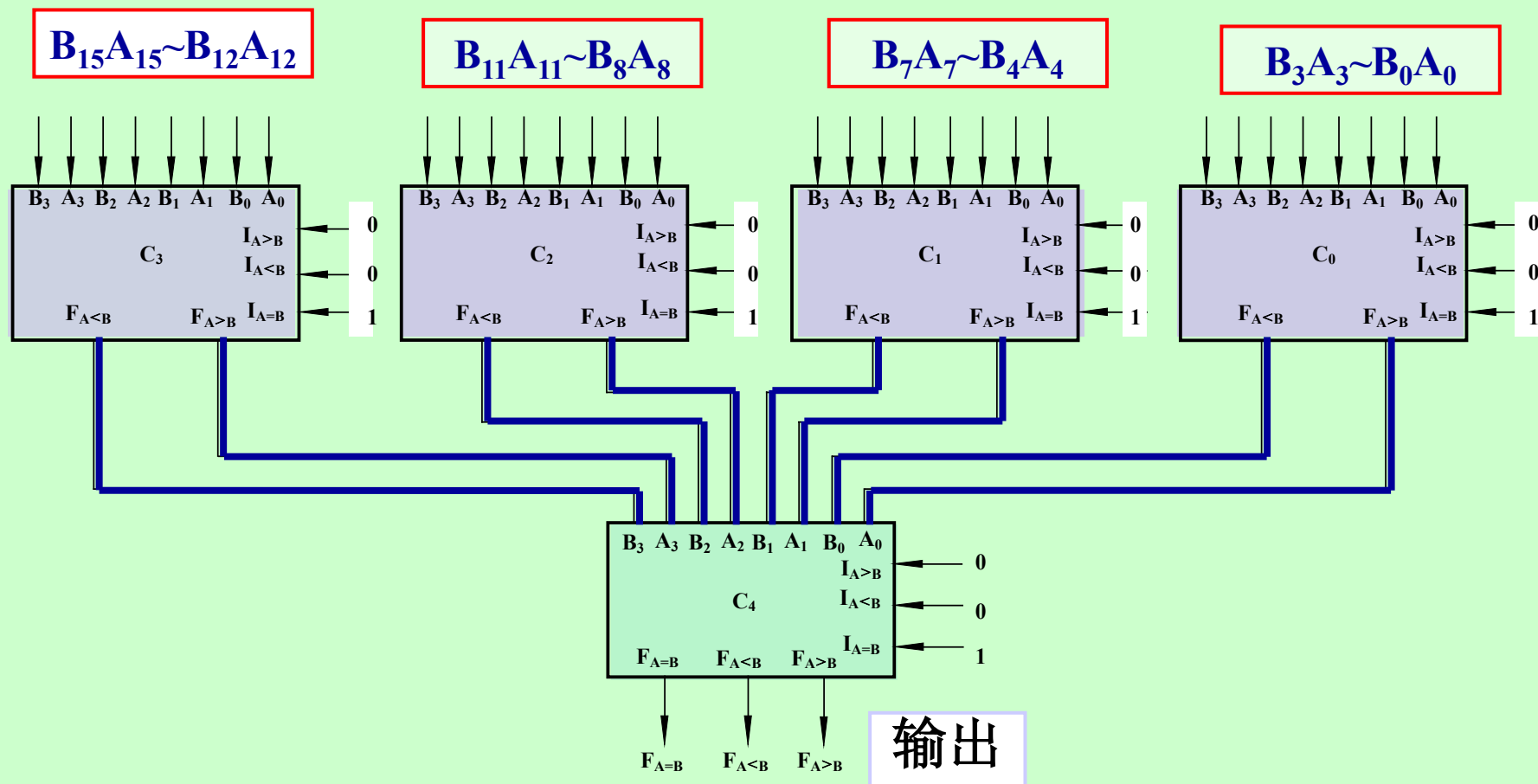
采用串联扩展方式数值比较器（了解）

用四片74HC85组成16位数值比较器（串联扩展方式）。



问题：如果每一片延迟时间为10ns，16位串行比较器延迟时间？

用74HC85组成16位数值比较器的并联扩展方式（了解）



问题：如果每一片延迟时间为10ns，16位并行比较器延迟时间？

4.4.5 算术运算电路

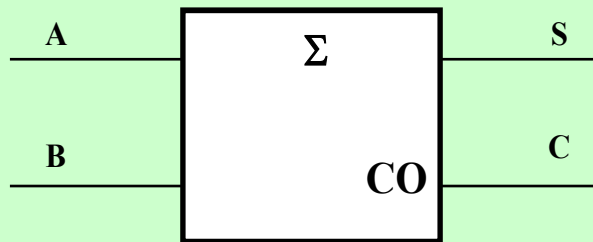
1、半加器和全加器

两个1位二进制数相加时，**不考虑低位来的进位**的加法 -----半加

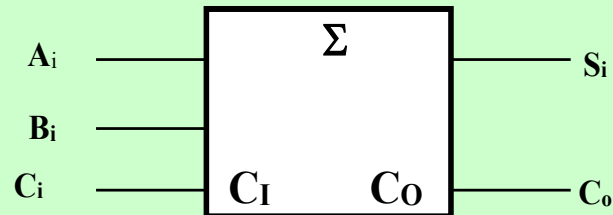
在两个1位二进制数相加时，**考虑低位进位**的加法 -----全加

加法器分为半加器和全加器两种。

半加器

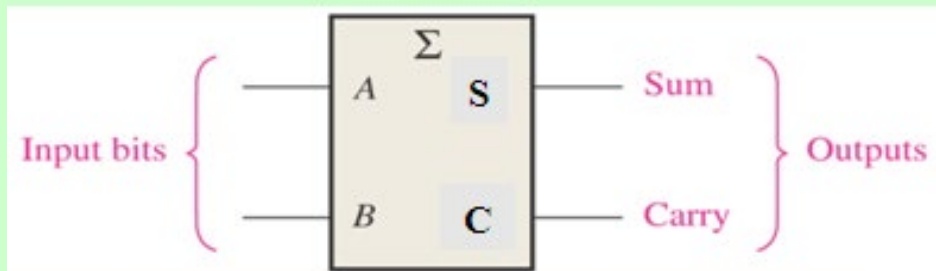


全加器



(1) 1位半加器 (Half Adder)

不考虑低位进位，将两个1位二进制数A、B相加的器件。



Truth table

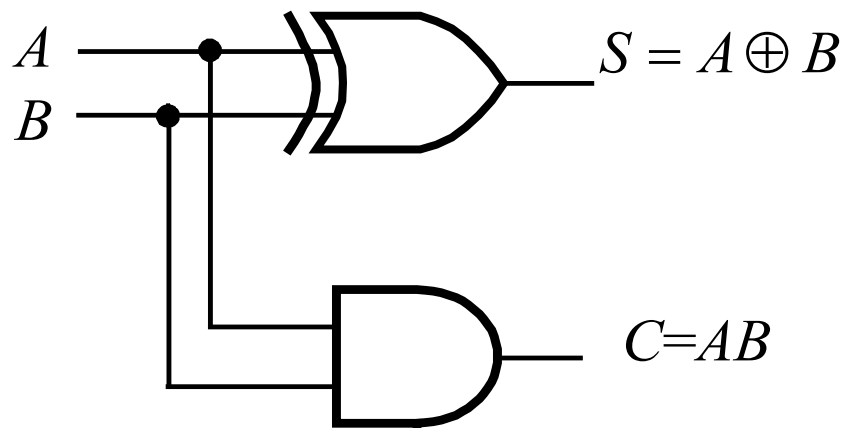
- 半加器的真值表

	A
+	B
<hr/>	
C	S

- 逻辑表达式

$$S = \bar{A}B + A\bar{B}$$

$$C = A B$$



逻辑图

logic diagram of half-adder

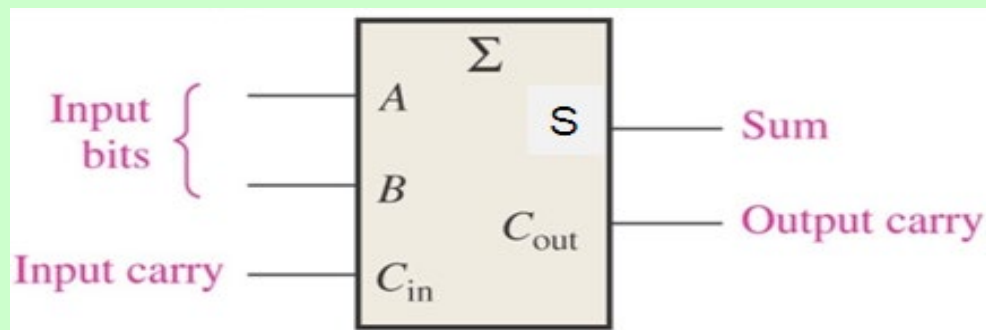
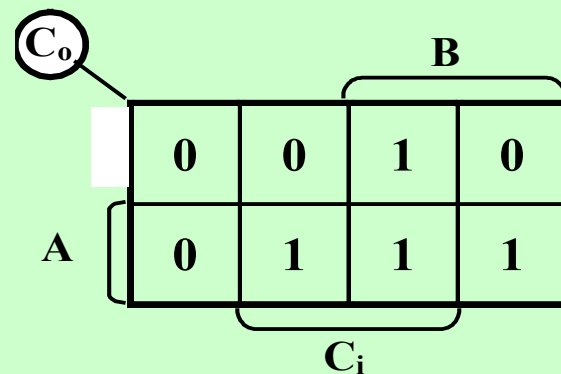
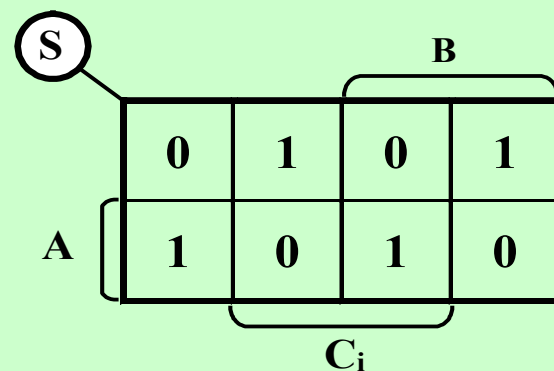
(2) 全加器 (Full Adder)

全加器能进行加数、被加数和低位来的进位信号相加，并根据求和结果给出该位的进位信号。

全加器真值表

A	B	C_i	S	C_o
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

$$\begin{array}{r} + \quad A \\ \quad B \\ \quad C_i \\ \hline C_o \quad S \end{array}$$



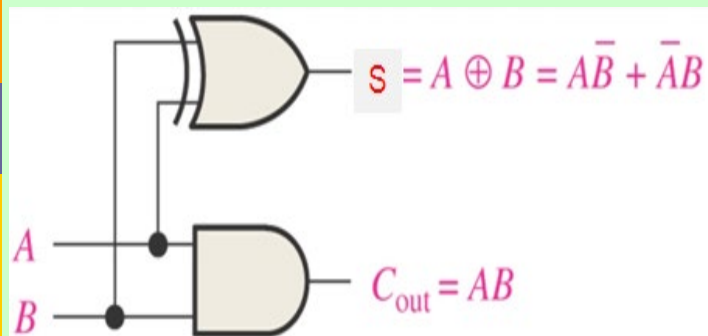
于是可得全加器的逻辑表达式为

$$S = \overline{A}\overline{B}C_i + \overline{A}B\overline{C}_i + A\overline{B}\overline{C}_i + ABC_i$$

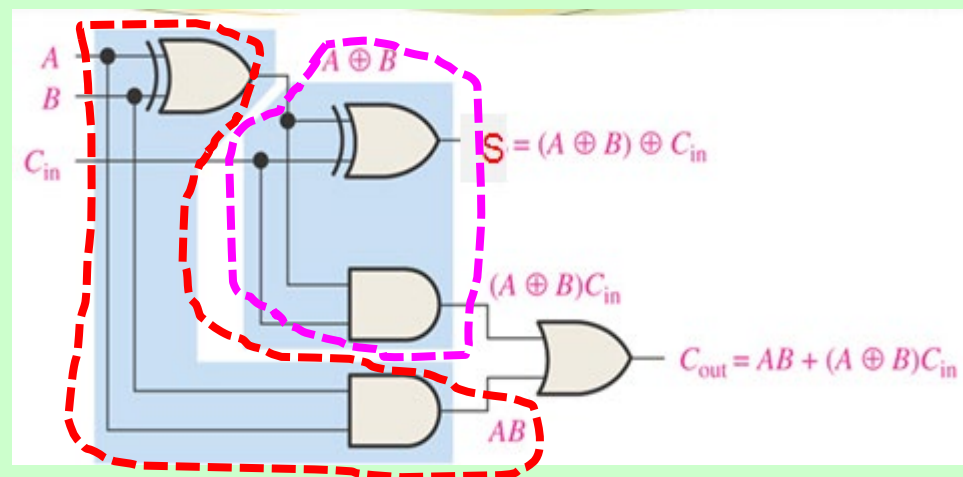
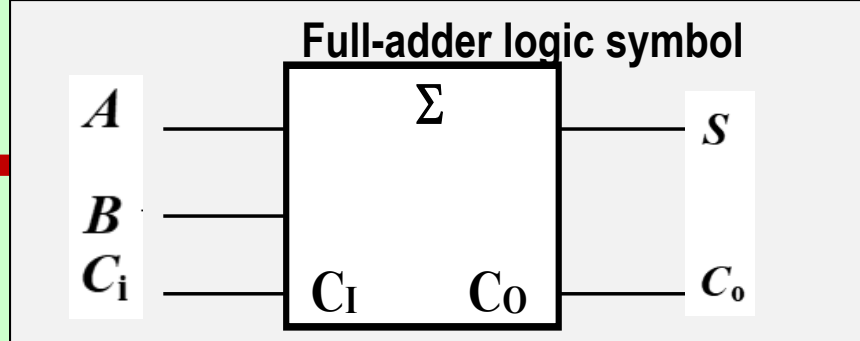
$$= A \oplus B \oplus C_i$$

$$C_o = AB + \overline{A}\overline{B}C_i + \overline{A}BC_i$$

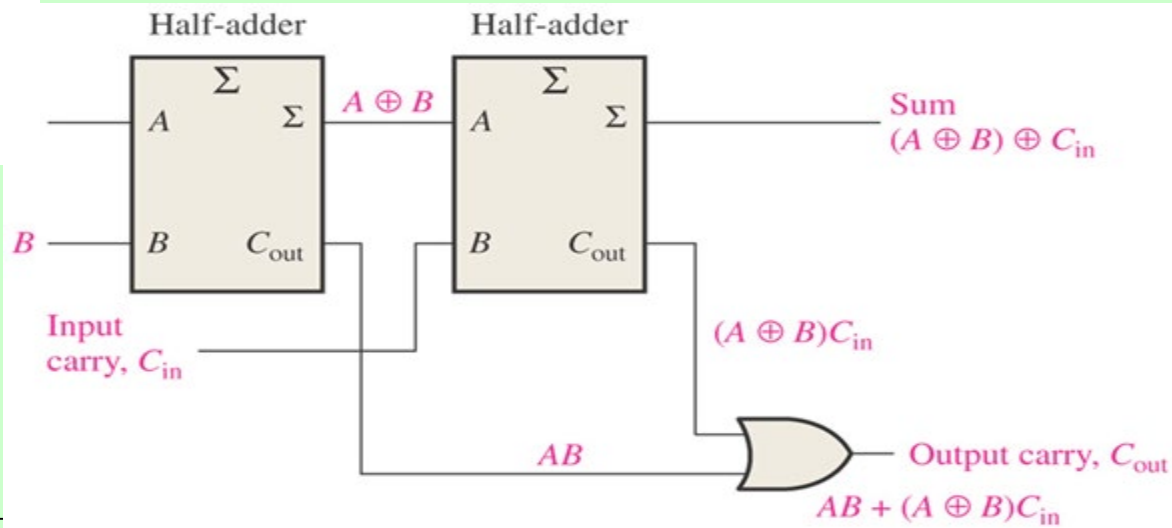
$$= AB + (A \oplus B)C_i$$



The Half-Adder



Complete logic circuit for a full-adder



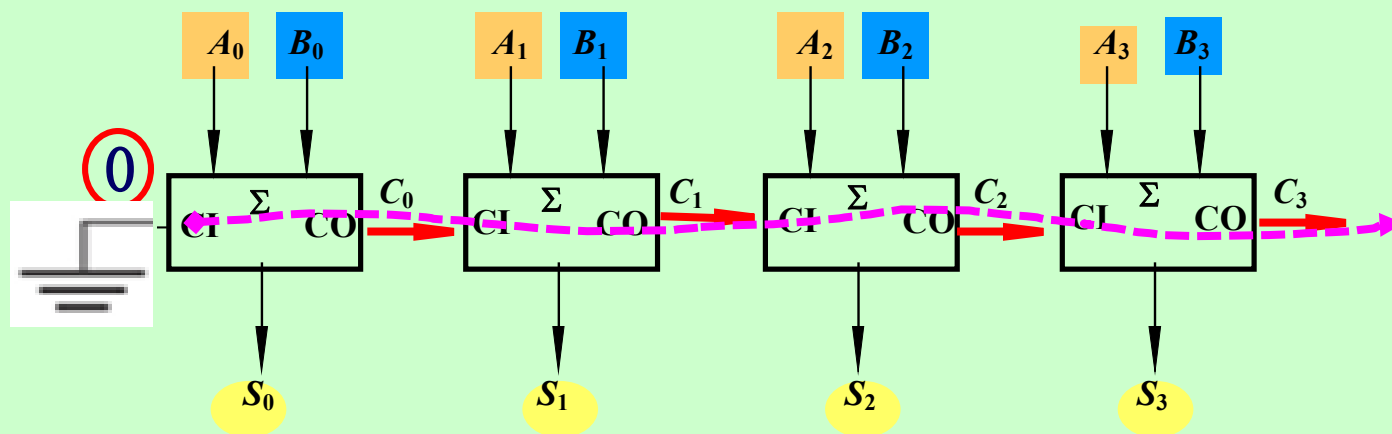
2、多位数加法器

- 如何用1位全加器实现两个四位二进制数相加？

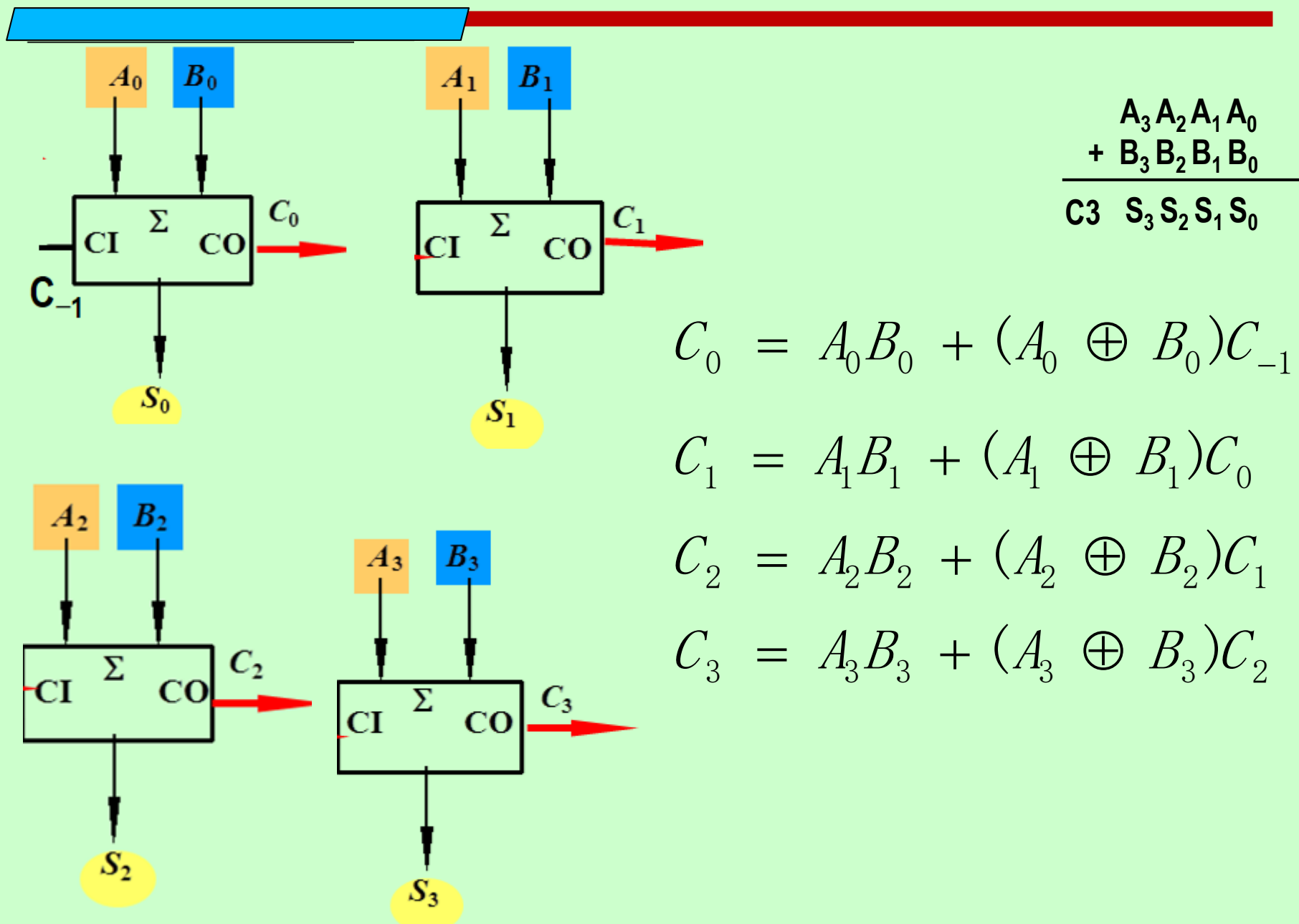
$$A_3 A_2 A_1 A_0 + B_3 B_2 B_1 B_0 = ?$$

$$\begin{array}{r} A_3 A_2 A_1 A_0 \\ + B_3 B_2 B_1 B_0 \\ \hline C_o S_3 S_2 S_1 S_0 \end{array}$$

(1) 串行进位加法器 Ripple Carry Adders



- 低位的进位信号送给邻近高位作为输入信号，采用串行进位加法器运算速度不高。



(2) 超前进位加法器

Look-Ahead Carry Adders

提高运算速度的**基本思想**：设计进位信号产生电路，在输入每位的加数和被加数时，同时获得该位全加的进位信号，而**无需**等待最低位的进位信号。

定义第*i*位的进位信号 (C_i)：

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

定义两个中间变量 G_i 和 P_i ：

$$G_i = A_i B_i$$

$$P_i = (A_i \oplus B_i)$$

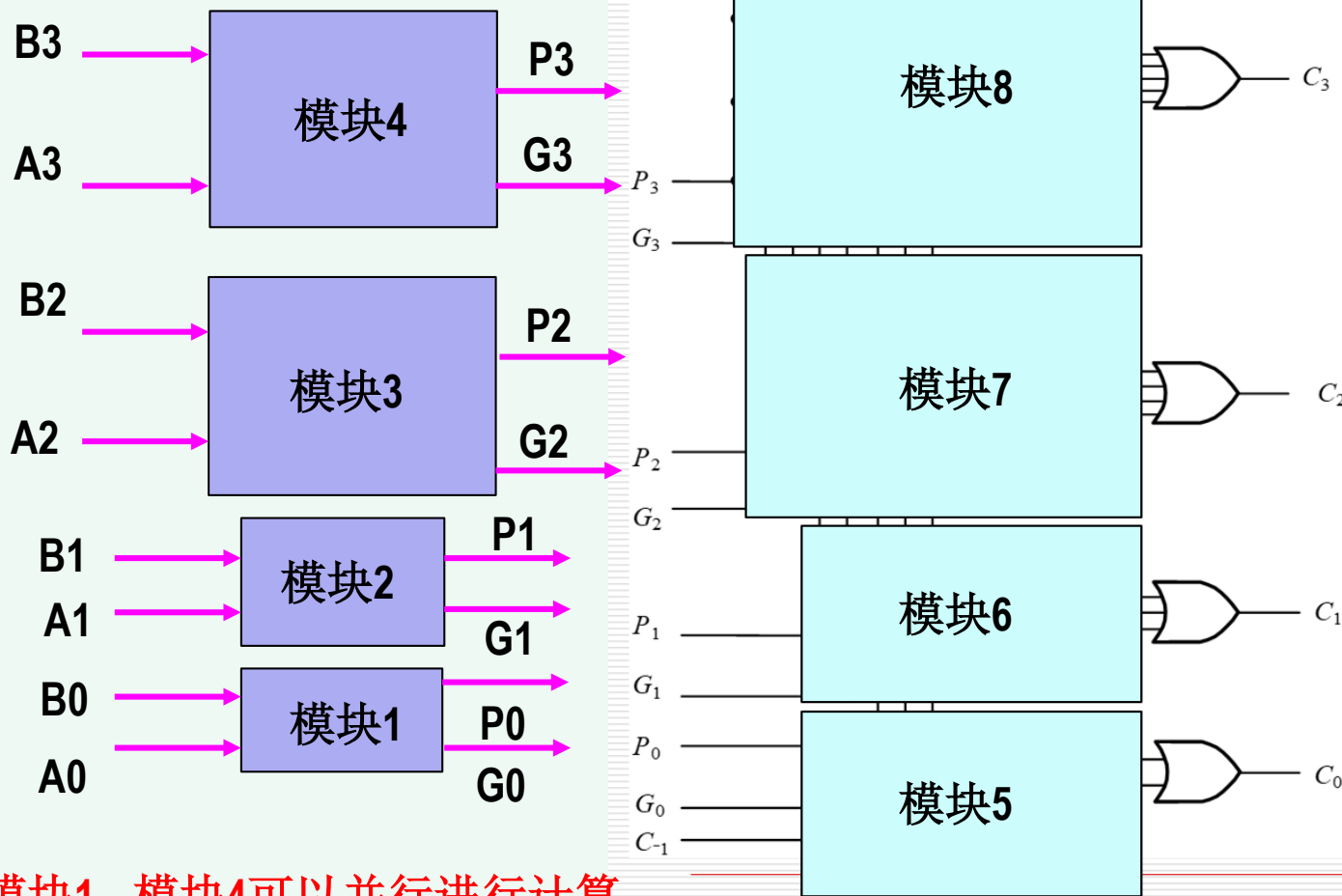
$$C_i = G_i + P_i C_{i-1}$$

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

超前进位产生设计思路

并行计算

并行计算



$$G_0 = A_0 B_0$$

$$p_0 = (A_0 \oplus B_0)$$

$$G_1 = A_1 B_1$$

$$p_1 = (A_1 \oplus B_1)$$

$$G_2 = A_2 B_2$$

$$p_2 = (A_2 \oplus B_2)$$

$$G_3 = A_3 B_3$$

$$p_3 = (A_3 \oplus B_3)$$

模块1...模块4可以并行进行计算

上页图中的模块5...模块8是否可以并行计算？

$$\text{由于 } C_i = G_i + P_i C_{i-1} \quad [G_i = A_i B_i \quad p_i = (A_i \oplus B_i)]$$

$$C_0 = G_0 + P_0 C_{-1} \quad (1)$$

$$C_1 = G_1 + P_1 C_0$$

$$C_1 = G_1 + P_1 G_0 + P_1 P_0 C_{-1} \quad (2)$$

$$C_2 = G_2 + P_2 C_1$$

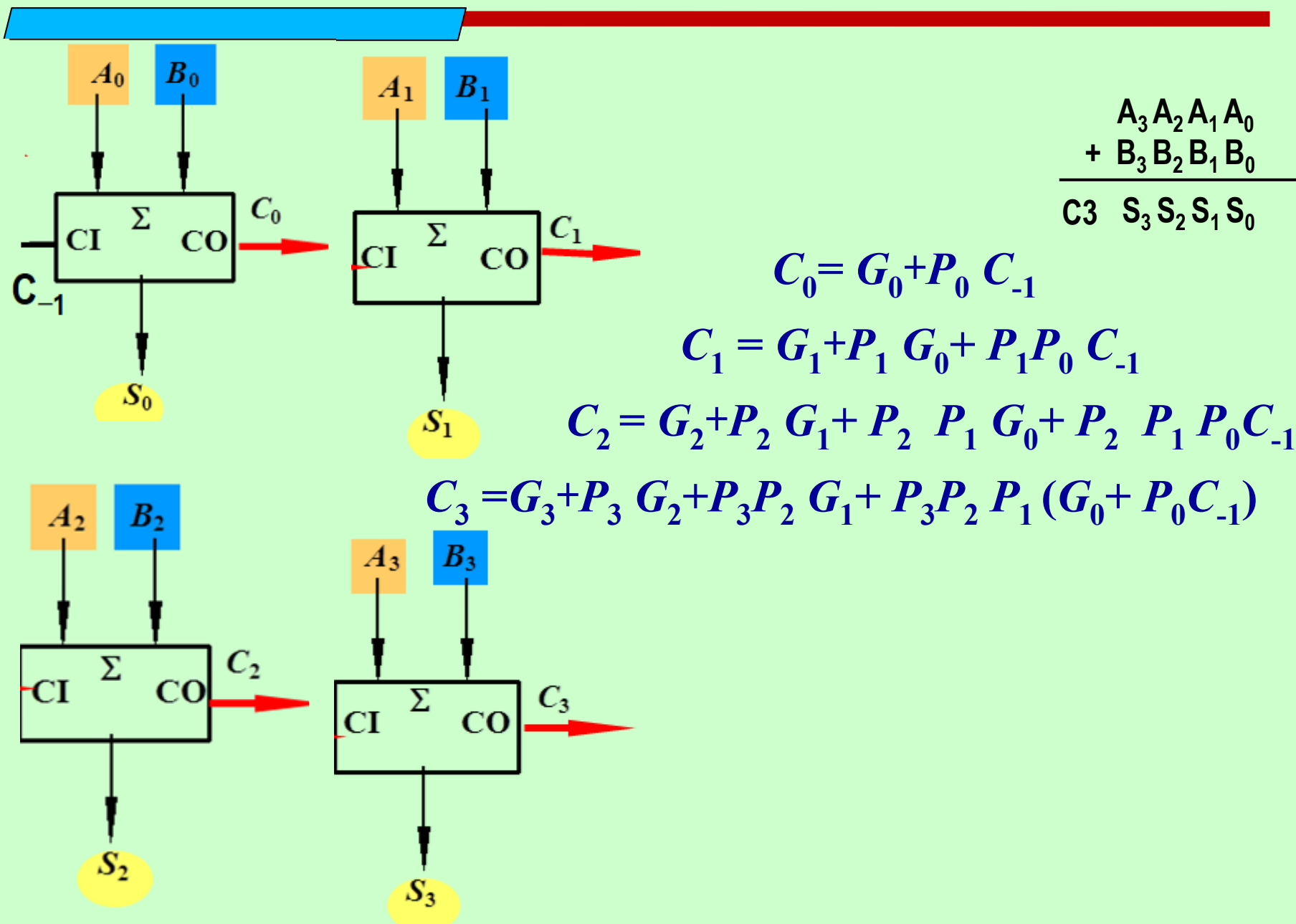
$$C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_{-1} \quad (3)$$

$$C_3 = G_3 + P_3 C_2 = G_3 + P_3 (G_2 + P_2 C_1) = G_3 + P_3 G_2 + P_3 P_2 C_1$$

$$= G_3 + P_3 G_2 + P_3 P_2 (G_1 + P_1 C_0)$$

$$C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 (G_0 + P_0 C_{-1}) \quad (4)$$

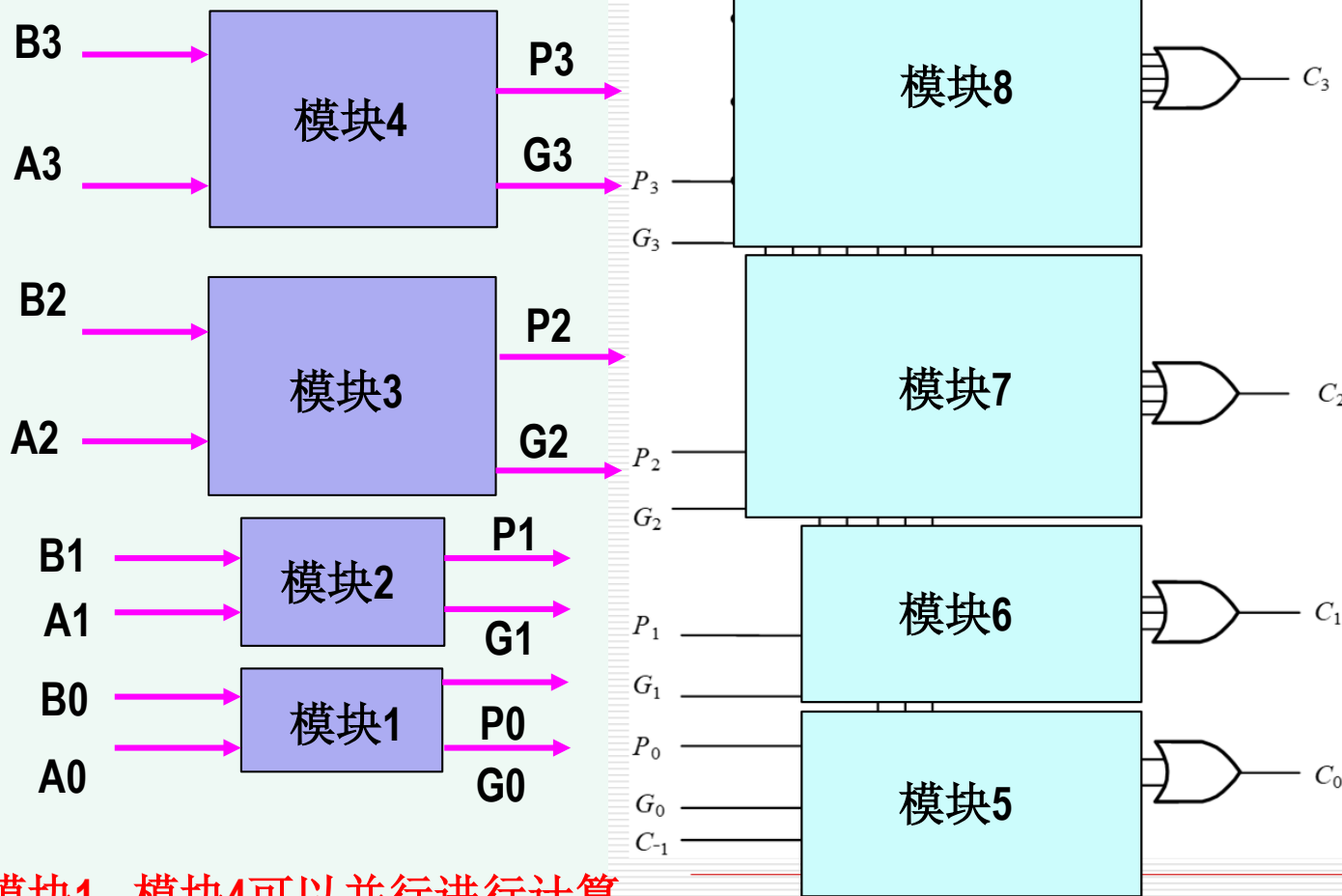
进位信号只由被加数、加数和C-1决定，而与其它低位的进位无关。提高了速度，但位数增加时，进位电路复杂度增加。



超前进位产生设计思路

并行计算

并行计算



$$G_0 = A_0 B_0$$

$$p_0 = (A_0 \oplus B_0)$$

$$G_1 = A_1 B_1$$

$$p_1 = (A_1 \oplus B_1)$$

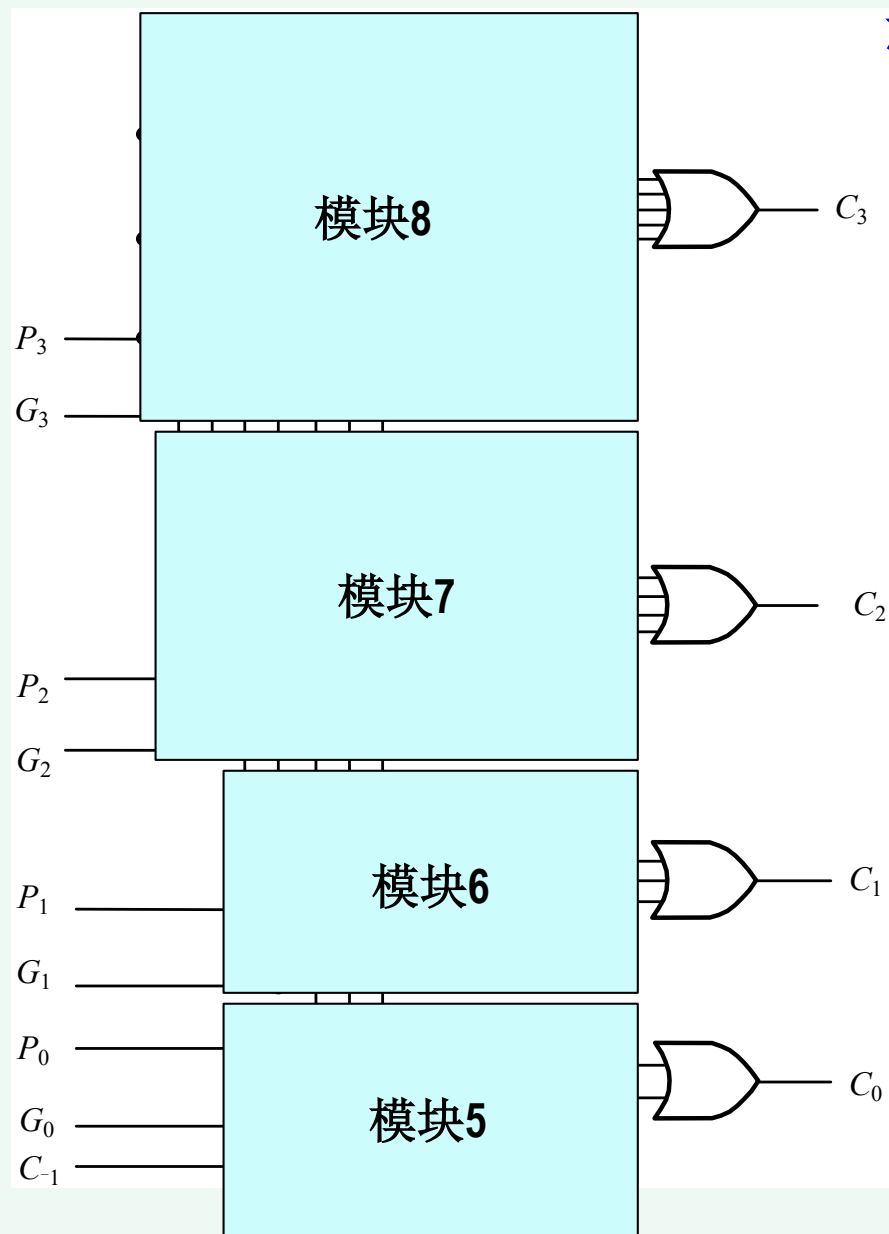
$$G_2 = A_2 B_2$$

$$p_2 = (A_2 \oplus B_2)$$

$$G_3 = A_3 B_3$$

$$p_3 = (A_3 \oplus B_3)$$

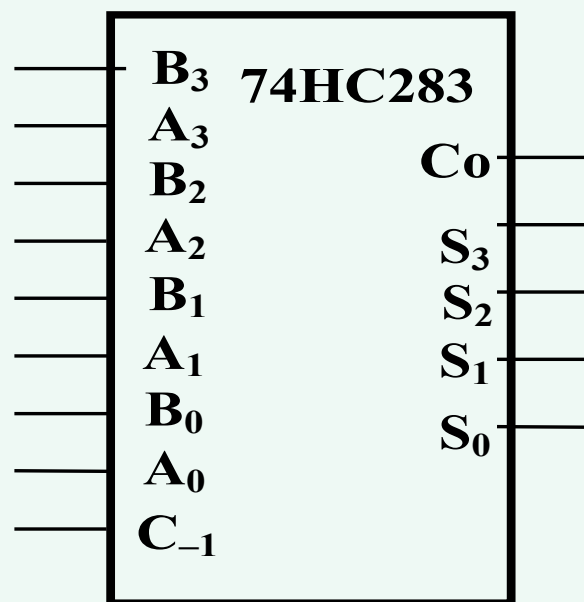
超前进位产生电路



模块5...模块8可以并行计算。
根据上页中的四个公式画出电路图

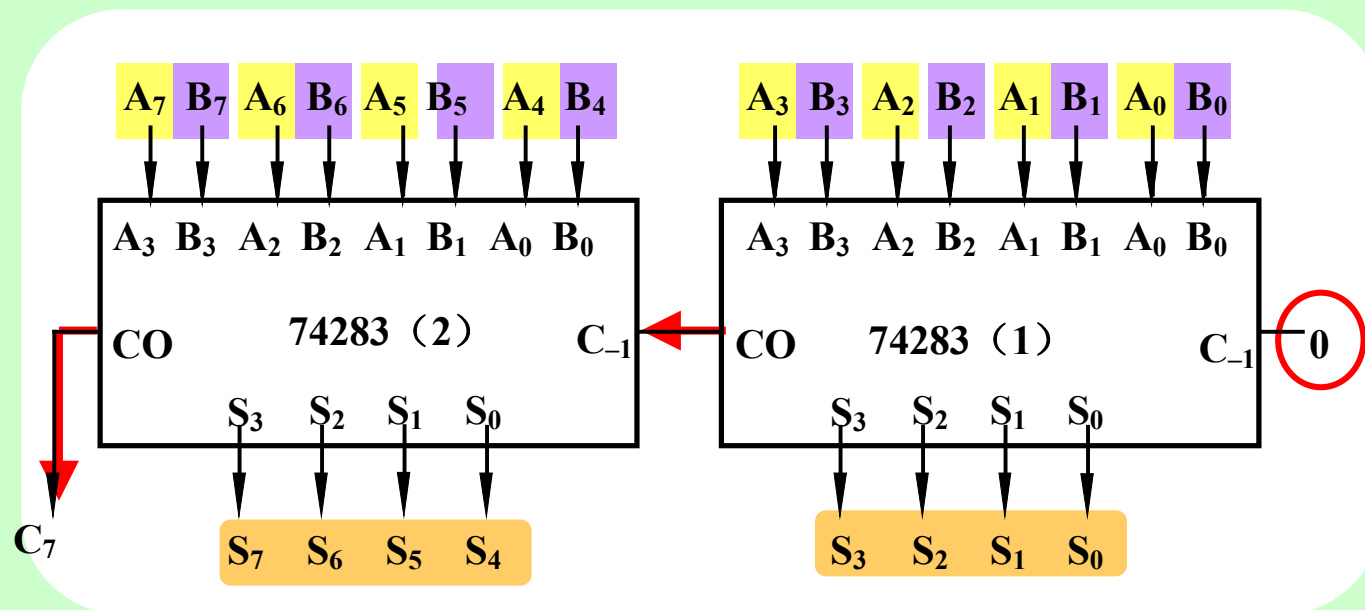
集成4位超前进位加法器74HC283

74HC283逻辑框图



3、超前进位加法器74LS283的应用

例1. 用两片74LS283构成一个8位二进制数加法器。



在片内是超前进位，而片与片之间是串行进位。

例. 用74283构成将8421BCD码转换为余3码的码制转换电路。

8421码

0000

0001

0010

⋮

+0011

+0011

+0011

余3码

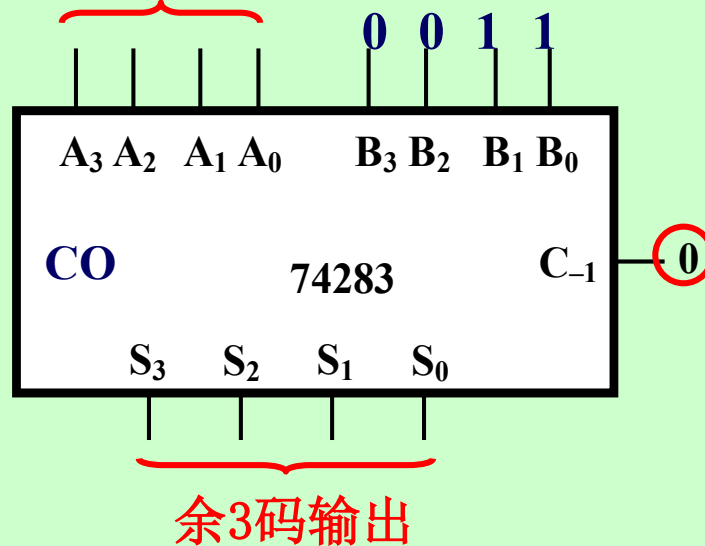
0011

0100

0101

⋮

8421码输入



余3码输出

4.5 组合可编程逻辑器件（推后讲）

4.5.1 PLD的结构、表示方法及分类

4.5.2 组合逻辑电路的PLD实现

4.6 用VerilogHDL描述组合逻辑电路（推后讲）

4.6.1 组合逻辑电路的行为级建模

4.6.2 分模块、分层次的电路设计