重点回顾

- **数据分配器**
  - 按照地址线，将输入送到指定道上输出
  - 可用译码器74x138实现

- **数据选择器**
  - 按照地址线，将指定道上的输入送到输出
  - 扩展方法（字扩展，位扩展）
  - 数据选择器实现组合逻辑

内容提纲

- 比较器
- 加法器

数值比较器

- 判断两个二进制数大小关系的逻辑电路
- 设计一位数值比较器
  - 输入：2个1位数A和B
  - 输出：$F_{A>B}$、$F_{A<B}$、$F_{A=B}$

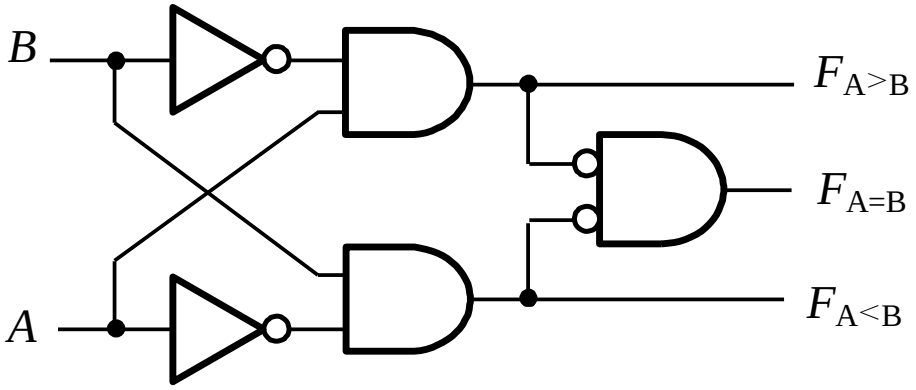
$$F_{A>B} = A\overline{B}$$

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

$$F_{A=B} = \overline{A}\,\overline{B} + AB$$

真值表

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



逻辑图

两位数值比较器

- 利用一位数值比较器设计两位数值比较器
- 待比较的两个2位二进制数：$A = A_1A_0$，$B = B_1B_0$
  - 先比较高位A_1和B_1
  - 只有高位相等，才比较低位A_0和B_0

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

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

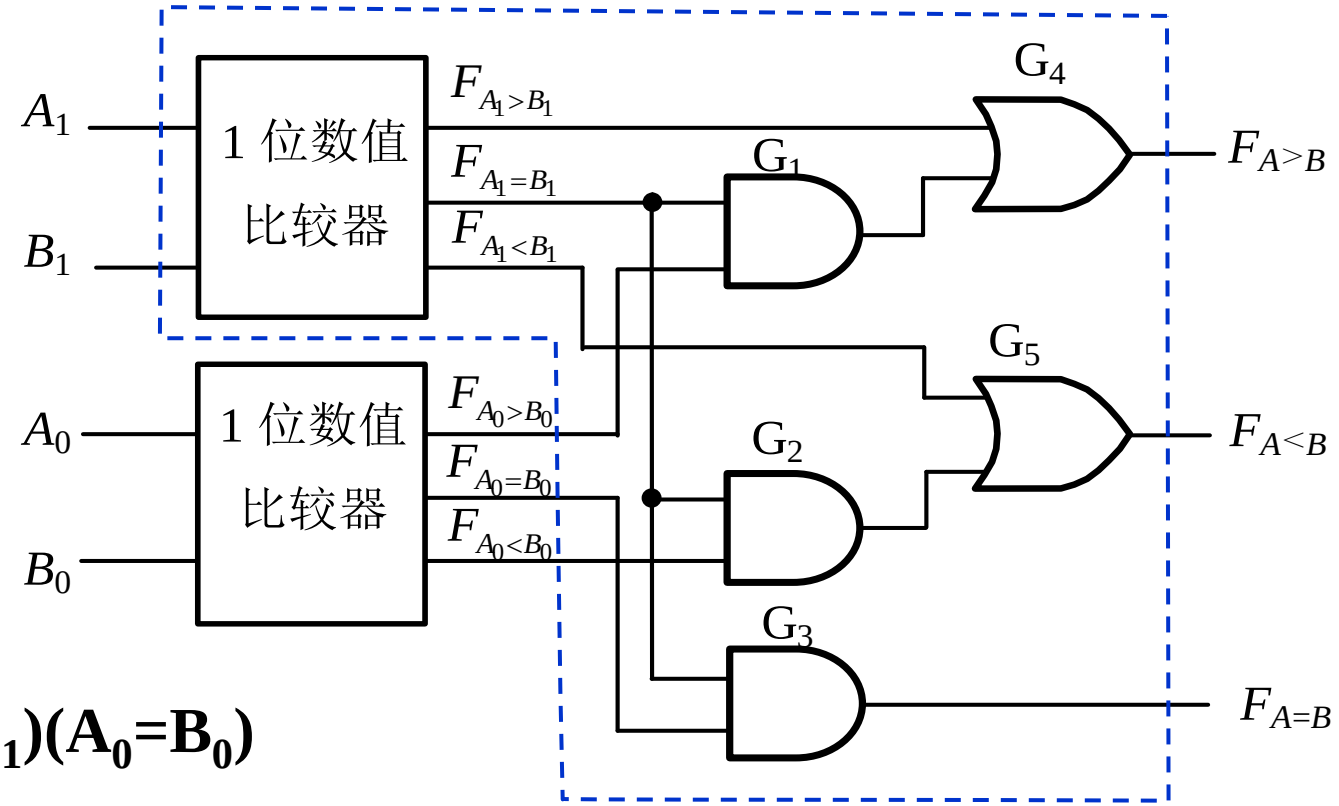
$$F_{A=B} = (A_1=B_1)(A_0=B_0)$$

真值表

A_1 B_1	A_0 B_0	$F_{A>B}$	$F_{A<B}$	$F_{A=B}$
$A_1>B_1$	x	1	0	0
$A_1<B_1$	x	0	1	0
$A_1=B_1$	$A_0>B_0$	1	0	0
	$A_0<B_0$	0	1	0
	$A_0=B_0$	0	0	1

两位数值比较器(续)

- 逻辑图

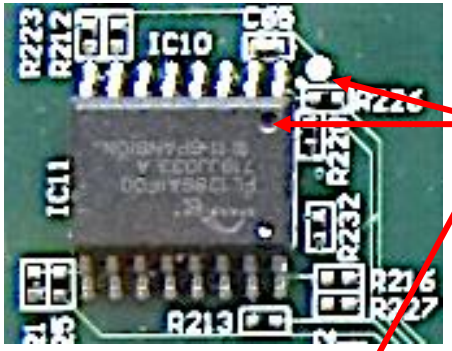


$$F_{A=B} = (A_1=B_1)(A_0=B_0)$$

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

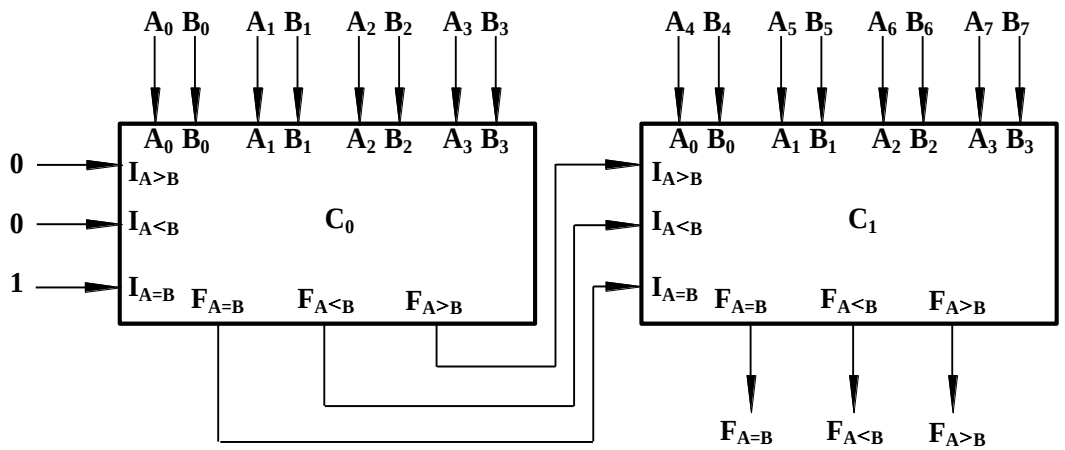
$$F_{A<B} = (A_1<B_1) + (A_1=B_1)(A_0<B_0)$$

四位数值比较器74x85

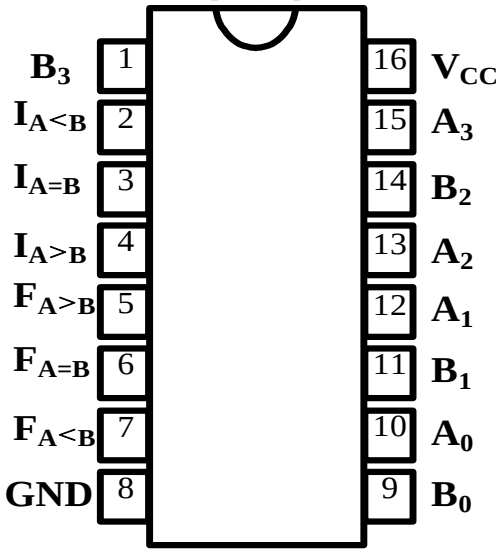


1脚标志

- 工作原理和两位数值比较器相同
- 提供附加输入端$I_{A<B}$、$I_{A=B}$和$I_{A>B}$，便于扩展应用
  - 从高位组比起，若不等，出结果，否则还需比较次高位组



八位串联数值比较器



引脚图

加法器

- **加法器是算术运算(加、减、乘、除)电路的基本单元**
- **1位加法器**
  - 1位半加器
  - 1位全加器
- **由1位加法器构成多位加法器**
  - 串行进位加法器
  - 超前进位加法器

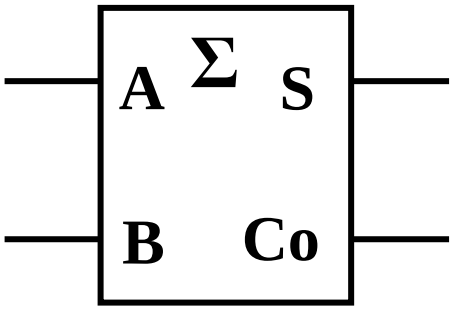
1位半加器

- **将两个1位数相加，产生1位和、 1位进位**
  - {Co, S} = A + B

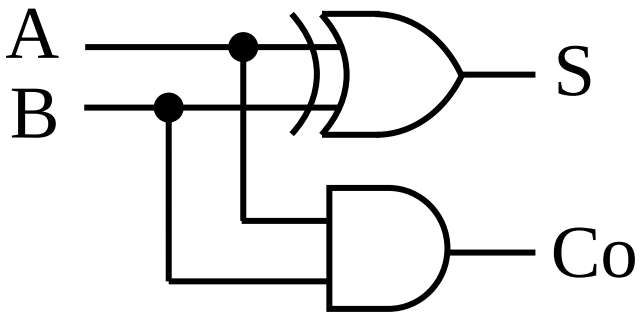
A	B	Co	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$S=\overline{A}B+A\overline{B}=A\oplus B$

$Co=AB$



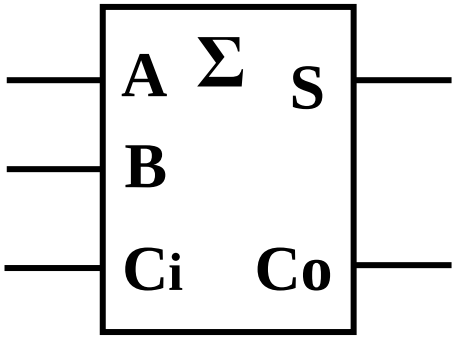
逻辑符号



1位全加器

- 将两个1位数与来自低位的进位相加，产生1位的和，以及向高位的进位
  - {Co, S} = A + B +Ci

C_i	A	B	C_O	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



逻辑符号

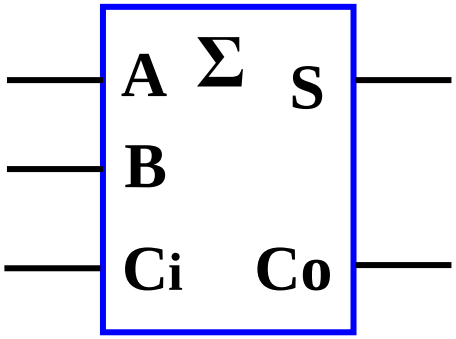
$$S = \overline{A}B\overline{C}_i + A\overline{B}\,\overline{C}_i + \overline{A}\,\overline{B}C_i + ABC_i$$

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

$$C_O = \underline{AB\overline{C}_i} + \underline{\overline{A}BC_i} + \underline{A\overline{B}C_i} + \underline{ABC_i}$$

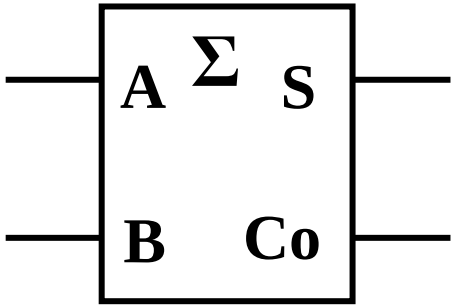
$$= AB + (A \oplus B)C_i$$

1位全加器逻辑图



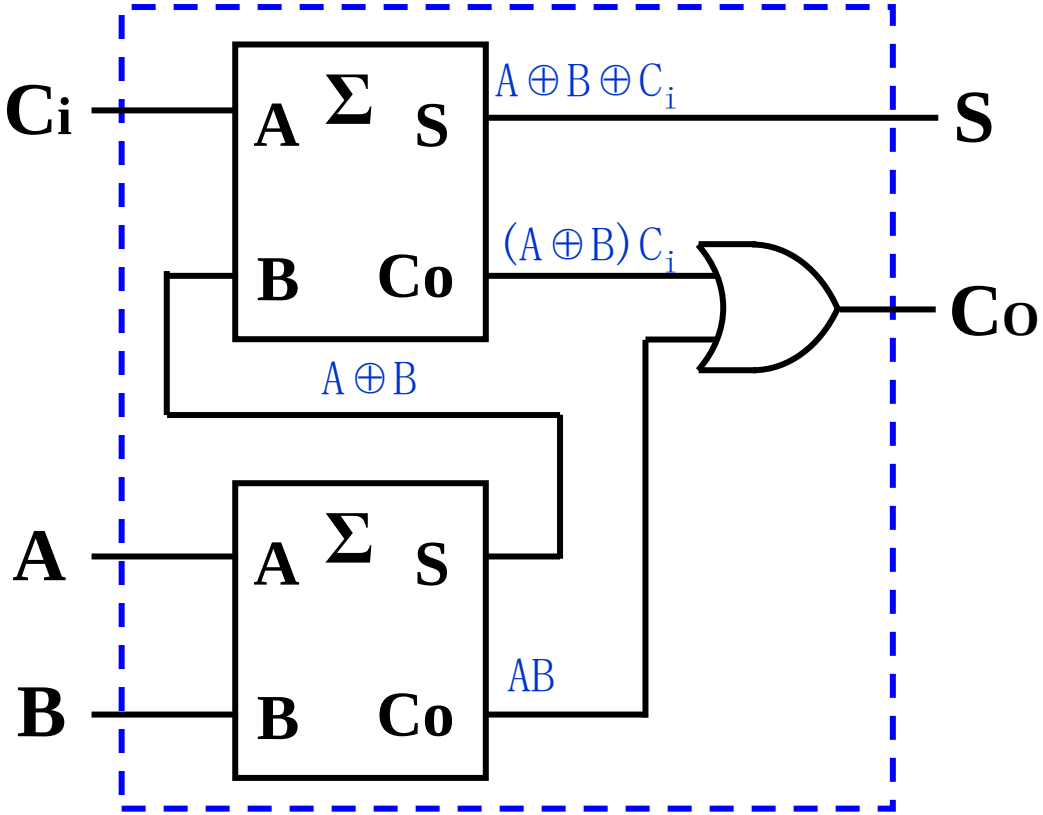
$$S = A \oplus B \oplus C_i$$

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



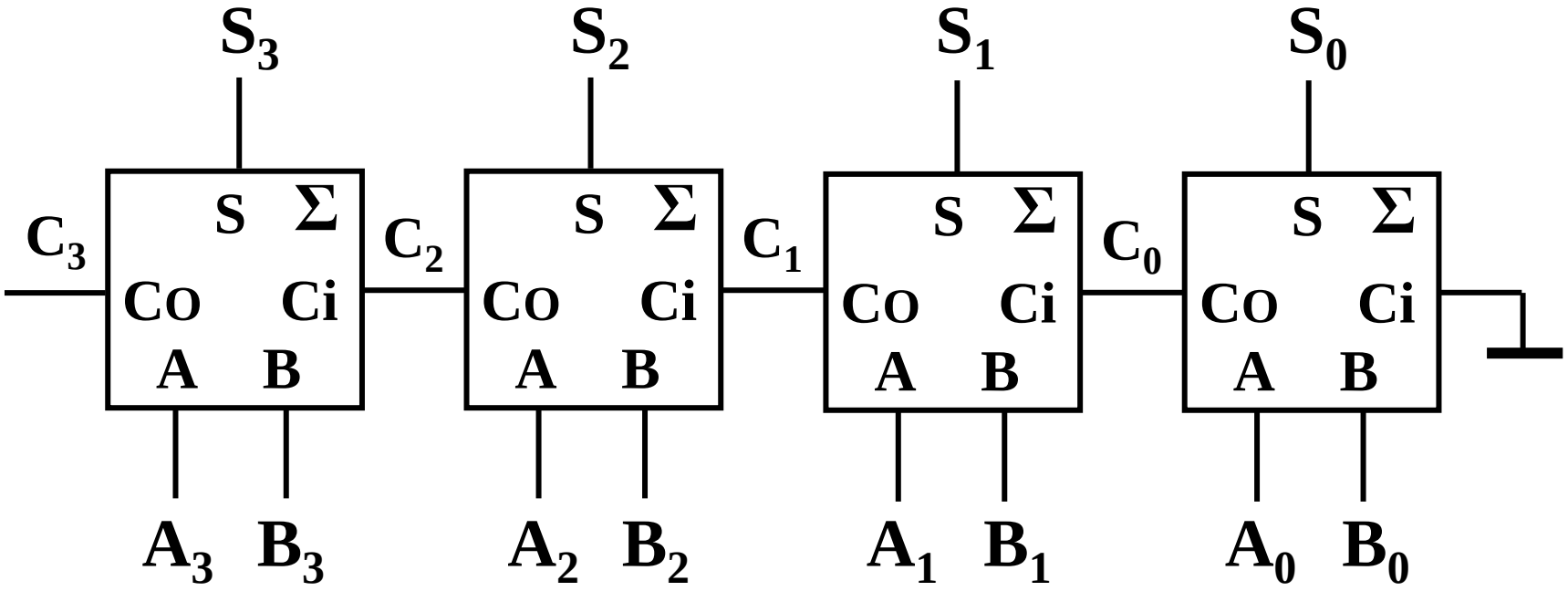
$$S = A \oplus B$$

$$C_O = A\,B$$



串行进位加法器

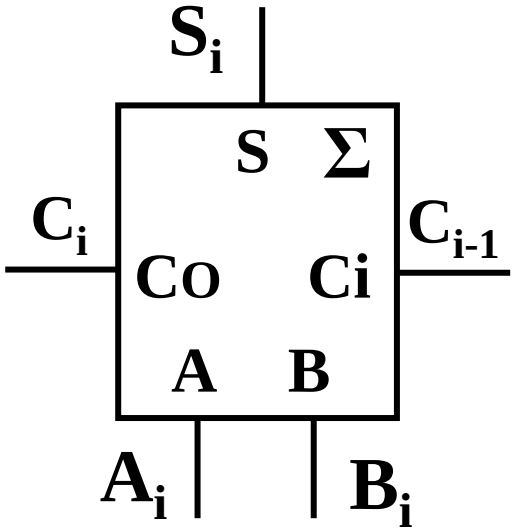
- 用1位全加器构造4位加法器



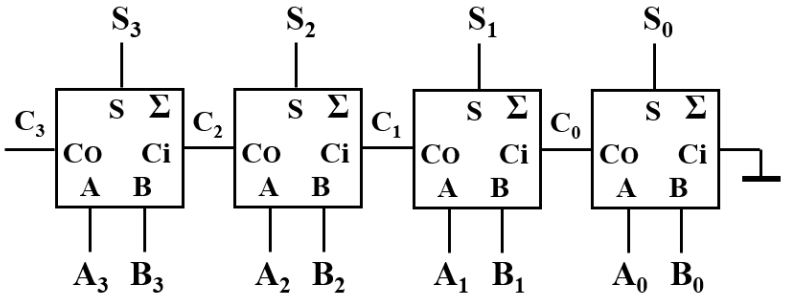
优点：简单，易于扩展；缺点：速度慢

超前进位加法器

- **基本原理**
  - C_{i-1}是A_{i-1}~A_0和B_{i-1}~B_0的函数
  - 设计每位进位信号产生电路：根据输入加数和被加数，同时获得该位全加的进位信号，无需等待最低位的进位信号
- **优点：速度快**
- **缺点：电路复杂**
- **4位超前进位加法器74x283**



进位信号超前产生



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

令 $G_i = A_iB_i$, $P_i = (A_i \oplus B_i)$ 则 $C_i = G_i + P_iC_{i-1}$

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

$\mathbf{C_0 = G_0 + P_0C_{-1}}$

$\mathbf{C_1 = G_1 + P_1C_0 = G_1 + P_1G_0 + P_1P_0C_{-1}}$

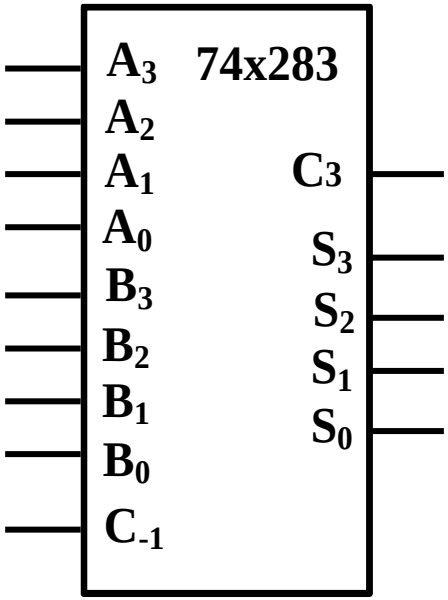
$$C_i = \sum_{j=0}^{i} \prod_{k=j+1}^{i} P_kG_j + \prod_{k=0}^{i} P_jC_{-1}$$

$\mathbf{C_2 = G_2 + P_2C_1 = G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_{-1}}$

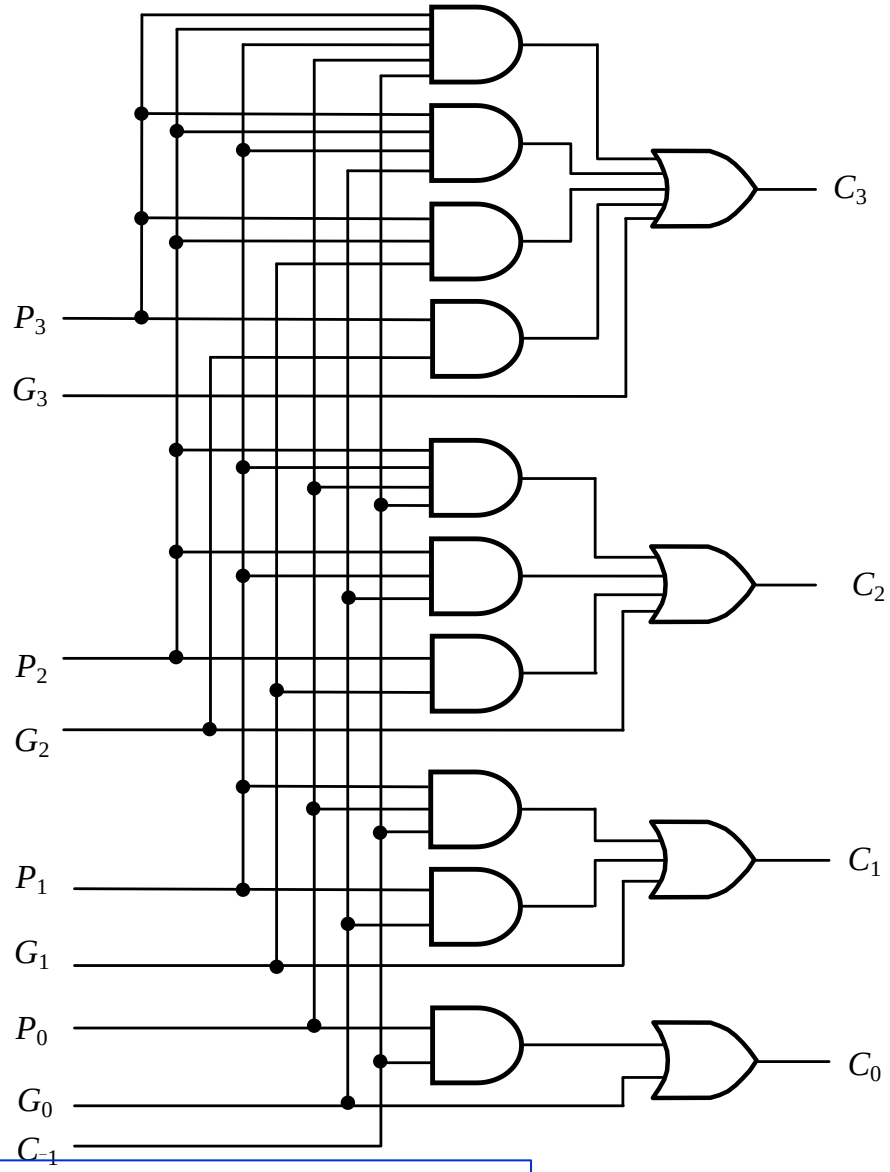
$\mathbf{C_3 = G_3 + P_3C_2 = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0}$

$\mathbf{+ P_3P_2P_1P_0C_{-1}}$

4位超前进位加法器74x283



逻辑符号



$$P_i = A_i \oplus B_i \qquad G_i = A_i B_i$$

$$C_i = \sum_{j=0}^{i} \prod_{k=j+1}^{i} P_k G_j + \prod_{k=0}^{i} P_j C_{-1}$$

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

门延迟 (gate delay)

In electronics, digital circuits and digital electronics, the propagation delay, or **gate delay**, is the length of time which starts when the input to a logic gate becomes stable and valid to change, to the time that the output of that logic gate is stable and valid to change. Often on manufacturers' datasheets this refers to the time required for the output to reach 50% of its final output level when the input changes to 50% of its final input level.

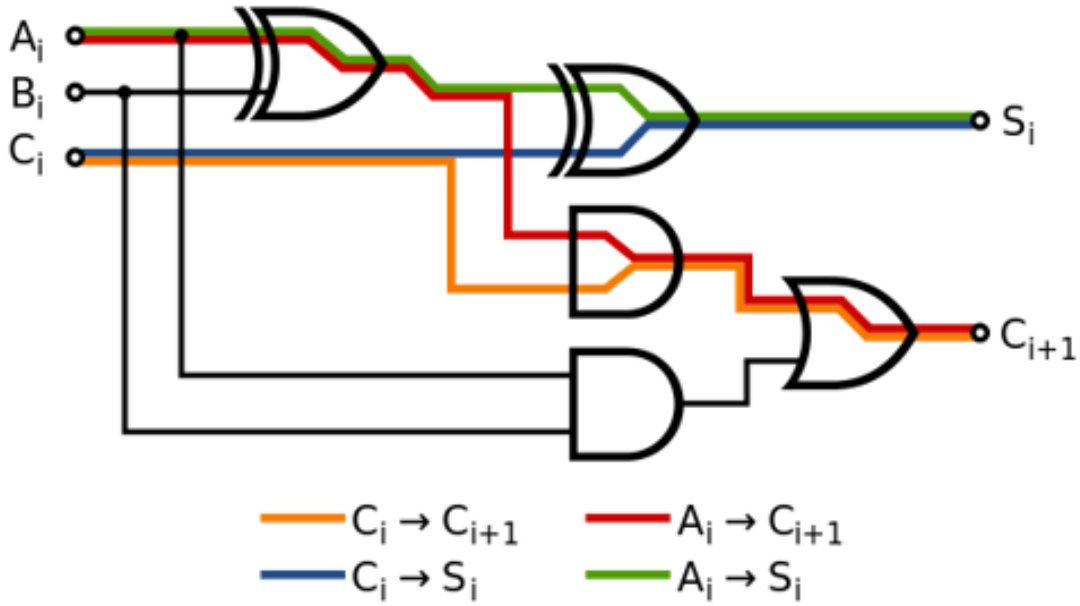
SN54147, SN74147 switching characteristics, V_{CC} = 5 V, T_A = 25°C (see Figure 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	WAVEFORM	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{PLH}	Any	Any	In-phase output	C_L = 15 pF, R_L = 400 Ω		9	14	ns
t_{PHL}						7	11	
t_{PLH}	Any	Any	Out-of-phase output			13	19	ns
t_{PHL}						12	19	

https://en.wikipedia.org/wiki/Propagation_delay

门延迟 (gate delay)

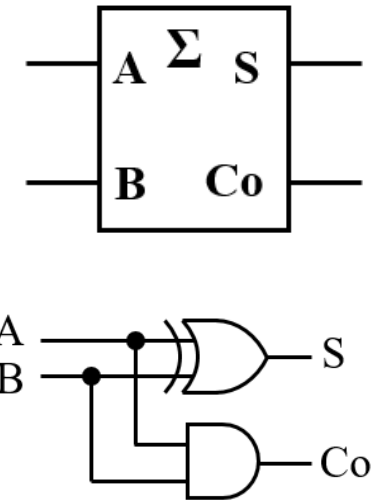
Reducing gate delays in digital circuits allows them to process data at a faster rate and improve overall performance. The determination of the propagation delay of a combined circuit requires identifying the longest path of propagation delays from input to output and by adding each tpd time along this path.



https://en.wikipedia.org/wiki/Propagation_delay

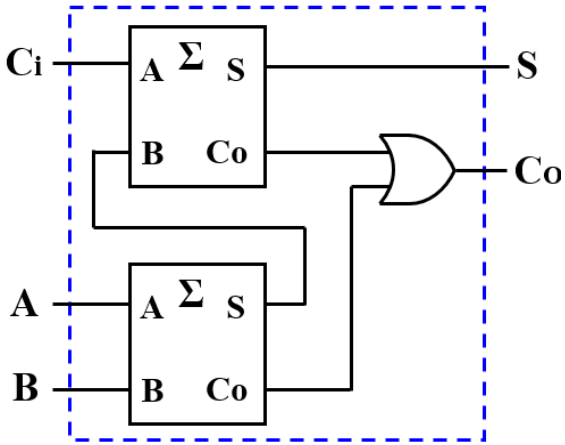
延迟分析

- 使用1位全加器实现4位全加器



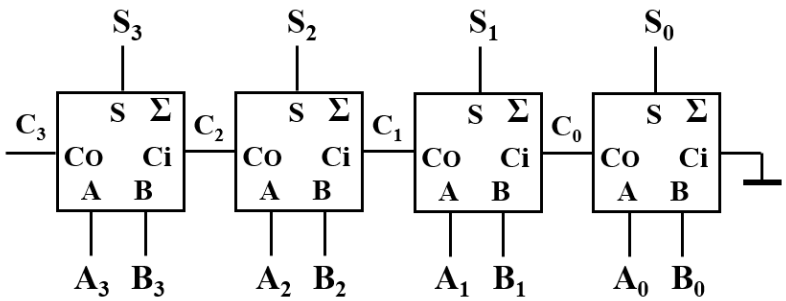
1位半加器

使用 2个门
S门延迟 1
Co门延迟 1



1位全加器

使用 2*2+1 = 5个门
S门延迟 2 (Ci->S,1; A/B->S,2)
Co门延迟 3 (Ci->Co,2;A/B->Co,3)



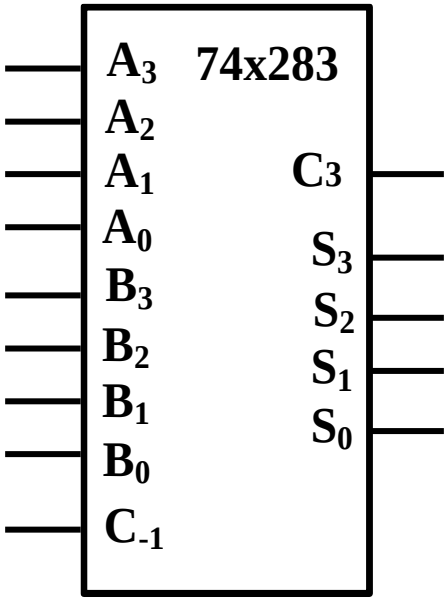
4位全加器

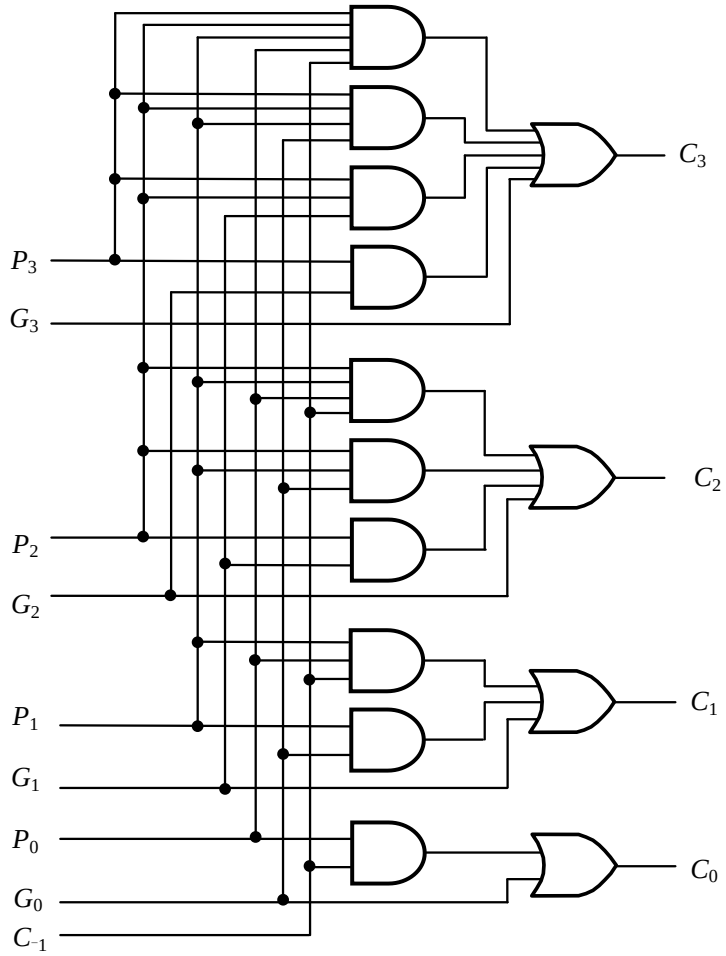
使用 5*4 = 20个门
S0门延迟 2,Co门延迟 3
S1门延迟3+1=4, C1门延迟3+2=5
S2门延迟5+1=6,C2门延迟 5+2=7
S3门延迟7+1=8, **C3门延迟7+2=9**

总共使用门数

4+4+14+4=30

Pi Gi Ci Si





$$P_i = A_i \oplus B_i \qquad G_i = A_i B_i$$

$$\boxed{C_i = \sum_{j=0}^{i} \prod_{k=j+1}^{i} P_k G_j + \prod_{k=0}^{i} P_j C_{-1}}$$

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

P_0,P_1,P_2,P_3 门延迟均为1

G_0,G_1,G_2,G_3 门延迟均为1

C_0,C_1,C_2,C_3 门延迟均为1+2=3

S_0,S_1,S_2,S_3 门延迟均为 3+1=4

且不随着位数增加而增加！

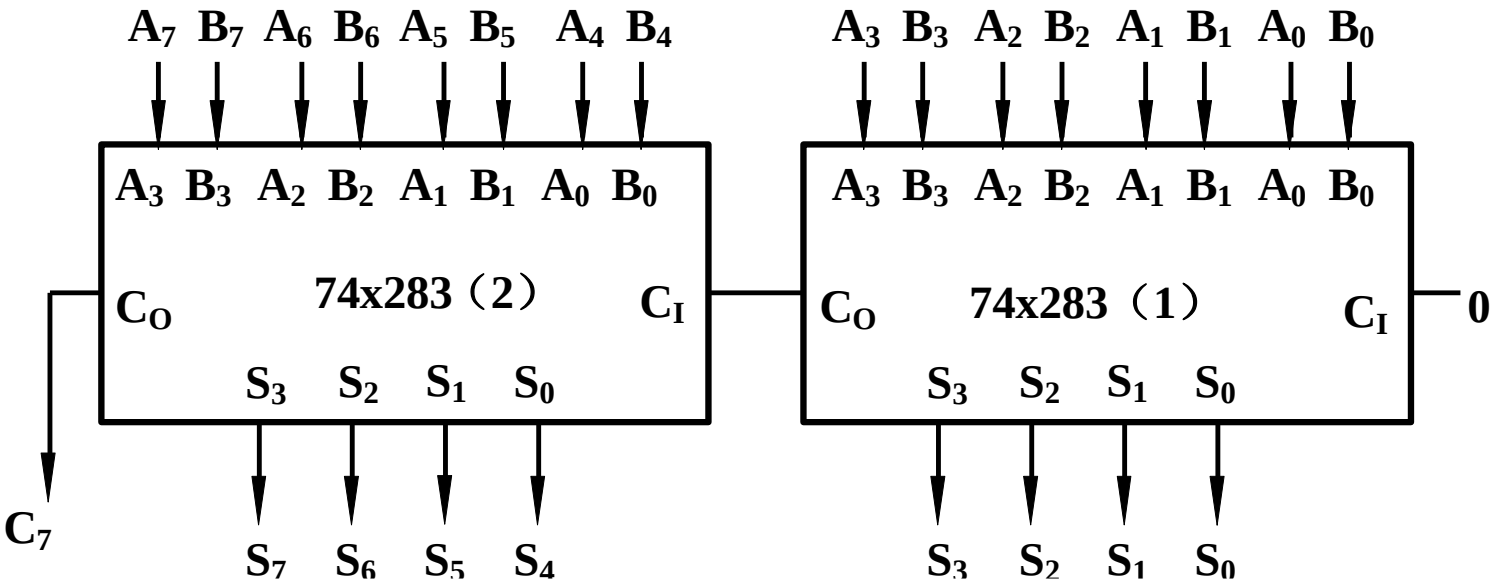
延迟分析

- **4位全加器**
  - 1位全加器串联：
    - 使用20个门，最大门延迟 3+2*3=9
  - 超前进位：
    - 使用30个门，最大门延迟4
- **8位全加器**
  - 1位全加器串联：
    - 使用40个门，最大门延迟 3+（2*7）=17
  - 超前进位：
    - 使用60个门，最大门延迟4

优点：速度快
缺点：电路复杂

74x283应用 (1)

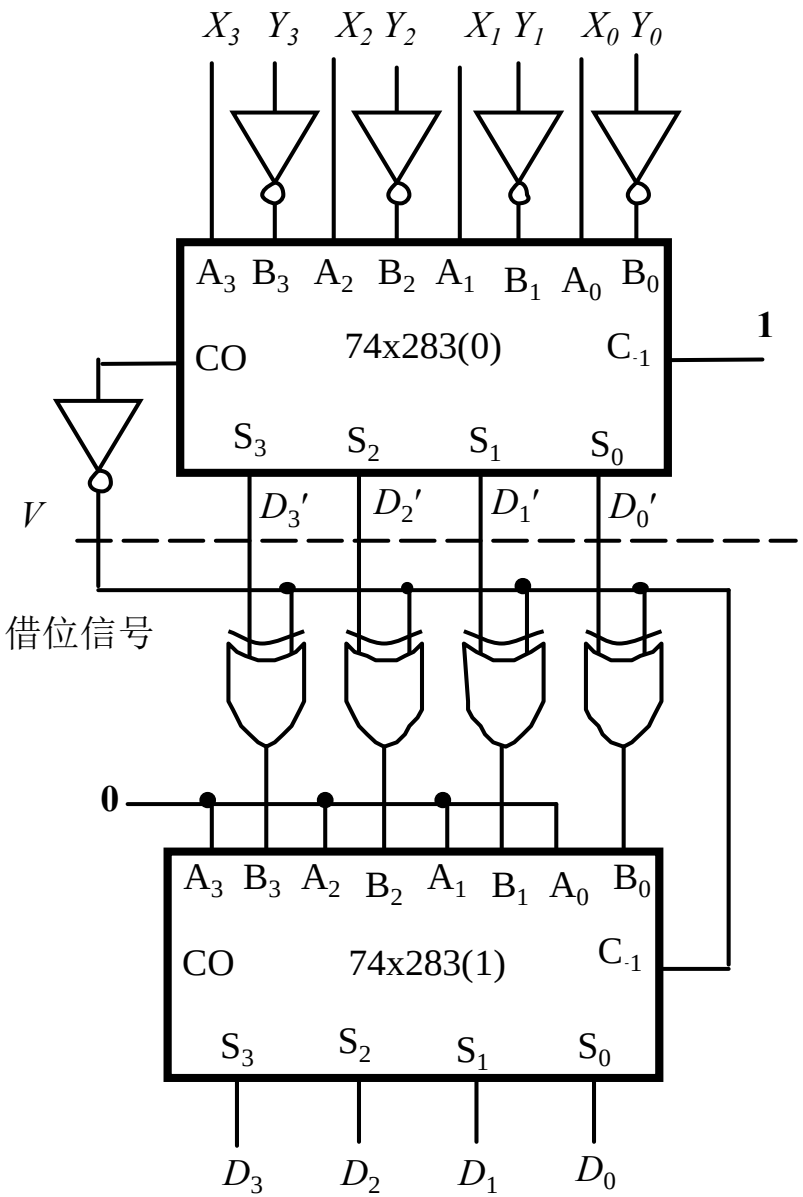
- **8位二进制数加法器**
  - 片内超前进位，片间串行进位



74x283应用 (2)

- **74x283(0)**
  - 按补码执行D’=X-Y运算
  - X ≥ Y : 无借位，V = 0
  - X < Y : 有借位，V = 1
- **74x283(1)**
  - V = 0 : D = 0+D’ = D
  - V = 1 : D = 0−D’ = −D

即 **D = |X−Y|**



隐藏条件

X和Y都大于零或者都小于零时才成立

输入			对 74x283(0)					对 74x283(0)				
X	Y	X 补	Y 补	/Y 补	X 补+/Y 补+1	Co	D’	V=/Co	B= V*/D’+/V*D’	D=0+B+V	D 原	X-Y
1	2	0001	0010	1101	01111	0	1111	1	0000	0001	1	-1
	-2		1110	0001	00011	0	0011	1	1100	1101	-3	3
-1	2	1111	0010	1101	11101	1	1101	0	1101	1101	-3	-3
	-2		1110	0001	10001	1	0001	0	0001	0001	1	1
2	1	0010	0001	1110	10001	1	0001	0	0001	0001	1	1
	-1		1111	0000	00011	0	0011	1	1100	1101	-3	3
-2	1	1110	0001	1110	11101	1	1101	0	1101	1101	-3	-3
	-1		1111	0000	01111	0	1111	1	0000	0001	1	-1