数字逻辑电路

(2020级本科生课程)

清华大学计算机系

陶品

taopin@tsinghua.edu.cn

办公室：FIT 3－531 （13717813059）

3.3 常用的中规模组合逻辑电路

3.3.1 译码器

3.3.2 数据选择器

3.3.3 编码器

3.3.4 数据比较器

3.3.5 奇偶校验器

3.3.6可编程逻辑器件

⇨ 3.3.7 运算器（算数逻辑单元 ALU）

3.3.6 运算器

3.3.6 运算器（算数逻辑单元 ALU）

- 加法器
  - 一位加法器
  - 四位串行进位加法器
  - 快速加法器
  - 16位加法器
- 算术运算逻辑单元
  - 四位算术逻辑运算单元

3.3.6 运算器

3.3.6 运算器（算数逻辑单元 ALU）

加法器 Adder

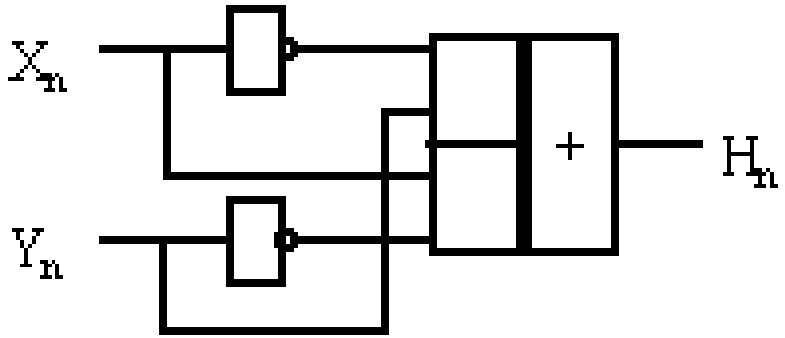
半加器（Half Adder）：不考虑低位进位输入和向高位的进位输出，两数码Xn、Yn相加，称半加。

一位半加器功能表

Xn	Yn	Hn
0	0	0
1	0	1
0	1	1
1	1	0

$$H_n = \overline{X_n}Y_n + X_n\overline{Y_n} = X_n \oplus Y_n$$

$$\overline{H_n} = X_nY_n + \overline{X_n}\overline{Y_n} = \overline{X_n \oplus Y_n}$$



3.3.6 运算器

3.3.6 运算器（算数逻辑单元 ALU）

加法器 Adder

全加器（Adder）：将Xn、Yn及低位进位C_{n-1}相加，并将进位输出C_n，称全加。

一位全加器功能表

Xn	Yn	C_{n-1}	F_n	C_n
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	0	1
0	0	1	1	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

3.3.6 运算器

一位全加器（Full Adder）

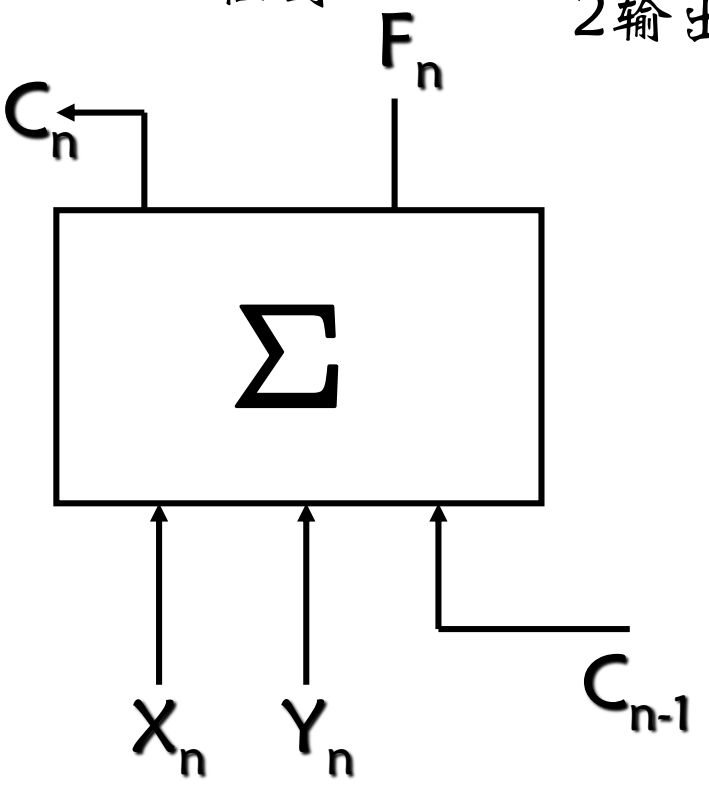
真值表

X_n	Y_n	C_{n-1}	F_n	C_n
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	0	1
0	0	1	1	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

框图

3输入

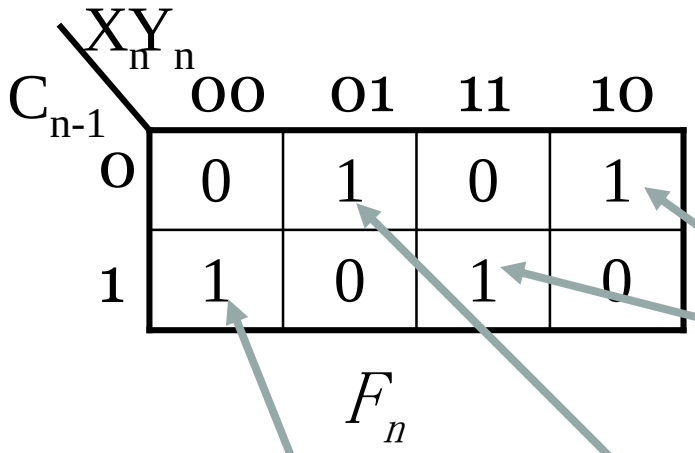
2输出



3.3.6 运算器

一位全加器（Full Adder）

卡诺图化简 Fn



$$F_n = \overline{X}_n\overline{Y}_nC_{n-1} + \overline{X}_nY_n\overline{C}_{n-1} + X_n\overline{Y}_n\overline{C}_{n-1} + X_nY_nC_{n-1}$$

$$= \overline{C}_{n-1}(\overline{X}_nY_n + X_n\overline{Y}_n) + C_{n-1}(\overline{X}_n\overline{Y}_n + X_nY_n)$$

$$= \overline{C_{n-1}(X_n \oplus Y_n)} + C_{n-1}\overline{(X_n \oplus Y_n)}$$

$$= C_{n-1} \oplus (X_n \oplus Y_n)$$

3.3.6 运算器

- 一位全加器（Full Adder）
  - 卡诺图化简Cn

$$C_n = X_n Y_n \overline{C}_{n-1} + X_n \overline{Y}_n C_{n-1} + \overline{X}_n Y_n C_{n-1} + X_n Y_n C_{n-1}$$

C_{n-1} \ XY_n	00	01	11	10
0	0	0	1	0
1	0	1	1	1

C_n

$$C_n = X_n Y_n + X_n C_{n-1} + Y_n C_{n-1}$$

$$= X_n Y_n + (X_n + Y_n) C_{n-1}$$

$$\overline{C_n} = \overline{X_n Y_n} + \overline{X_n C_{n-1}} + \overline{Y_n C_{n-1}}$$

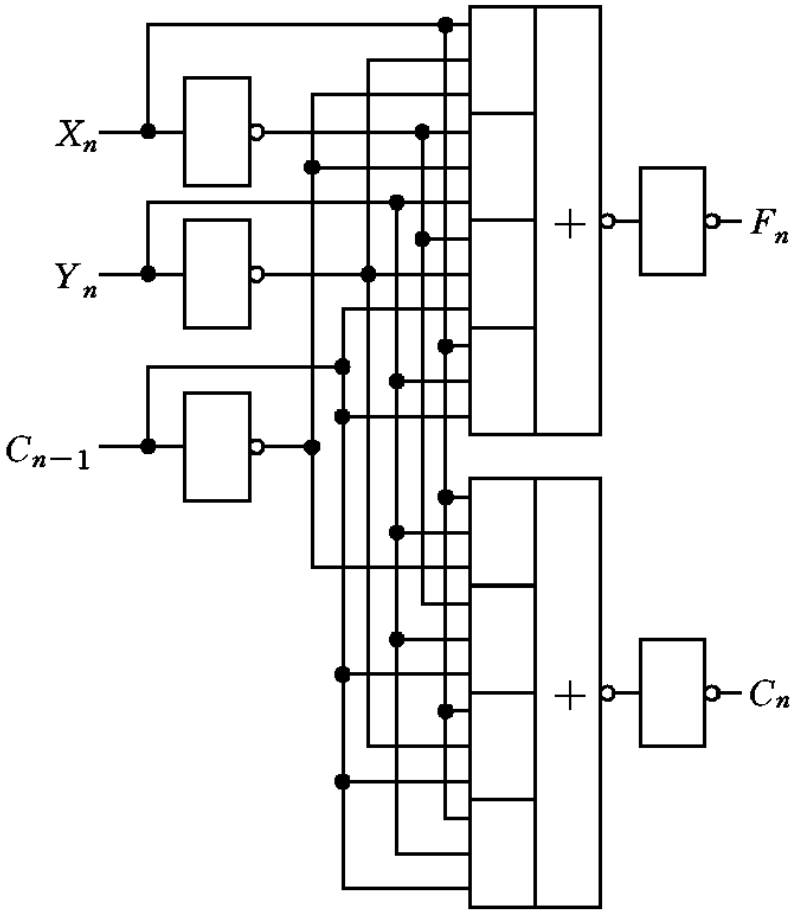
3.3.6 运算器

- 一位全加器实现方案1

$$F_n = X_n\overline{Y}_n\overline{C}_{n-1} + \overline{X}_nY_n\overline{C}_{n-1} + \overline{X}_n\overline{Y}_nC_{n-1} + X_nY_nC_{n-1}$$

$$C_n = X_nY_n\overline{C}_{n-1} + X_n\overline{Y}_nC_{n-1} + \overline{X}_nY_nC_{n-1} + X_nY_nC_{n-1}$$

不化简，用全部最小项实现，需要3级门。



3.3.6 运算器

- 一位全加器实现方案2

写$\overline{F}$ $\overline{C}$的表达式

$$\overline{F} = \overline{X_n Y_n C_{n-1}} + \overline{X_n} Y_n C_{n-1} + X_n \overline{Y_n} C_{n-1} + X_n Y_n \overline{C_{n-1}}$$

$$F = \overline{\overline{F}} = \overline{\overline{X_n Y_n C_{n-1}} + \overline{X_n} Y_n C_{n-1} + X_n \overline{Y_n} C_{n-1} + X_n Y_n \overline{C_{n-1}}}$$

$$\overline{C_n} = \overline{X_n Y_n} + \overline{X_n C_{n-1}} + \overline{Y_n C_{n-1}}$$

$$C_n = \overline{\overline{C_n}} = \overline{\overline{X_n Y_n} + \overline{X_n C_{n-1}} + \overline{Y_n C_{n-1}}}$$

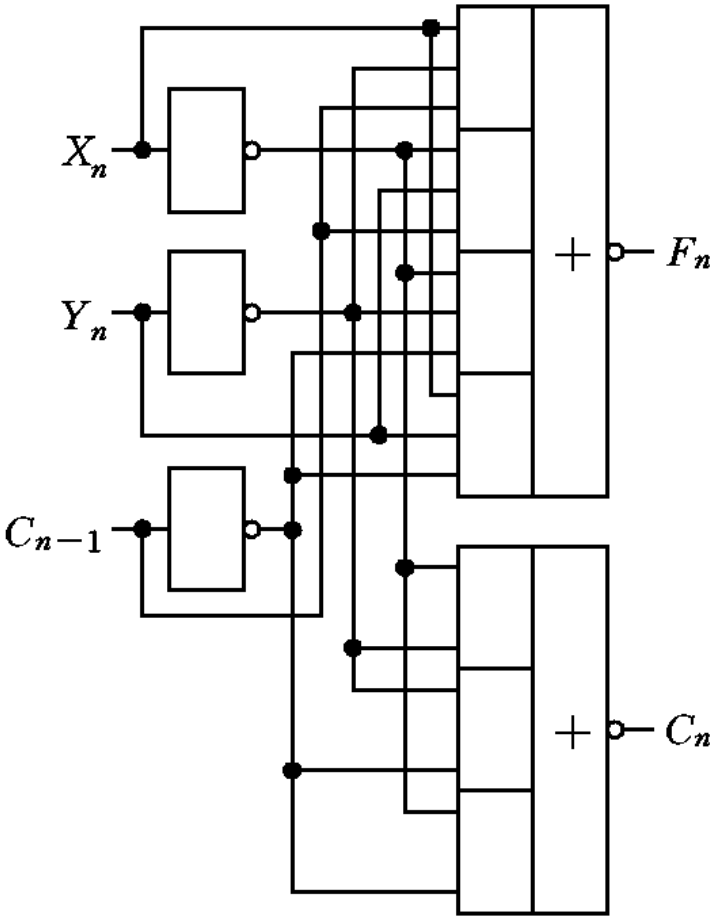
经变换后只要2级门。

3.3.6 运算器

- 一位全加器实现方案2(续)

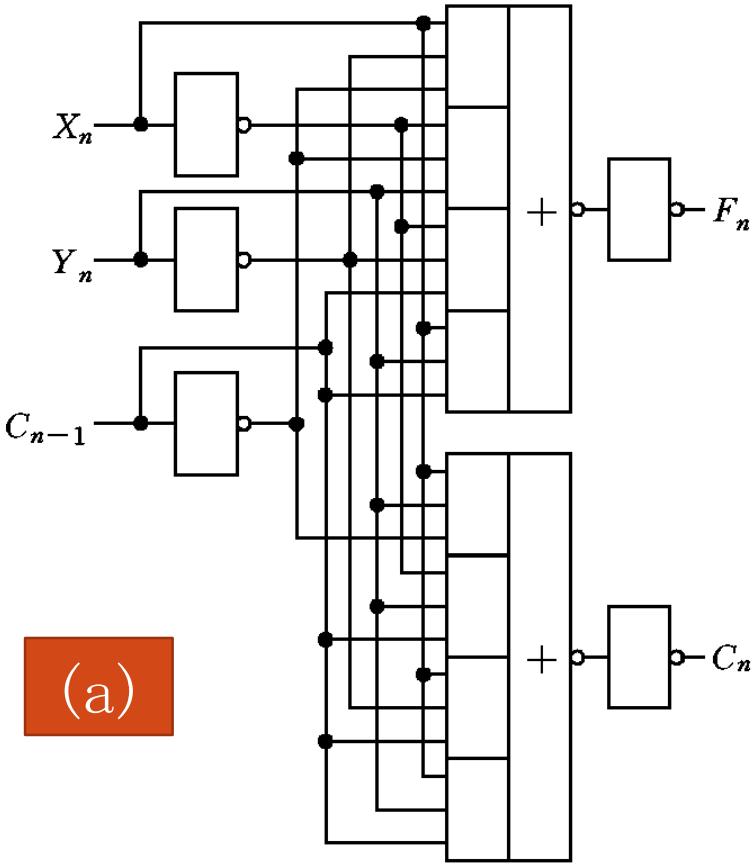
$$F = \overline{\overline{F}} = \overline{\overline{X_n Y_n C_{n-1}} \cdot \overline{X_n Y_n C_{n-1}} \cdot \overline{X_n Y_n C_{n-1}} \cdot \overline{X_n Y_n C_{n-1}}}$$

$$C_n = \overline{\overline{C_n}} = \overline{\overline{X_n Y_n} \cdot \overline{X_n C_{n-1}} \cdot \overline{Y_n C_{n-1}}}$$

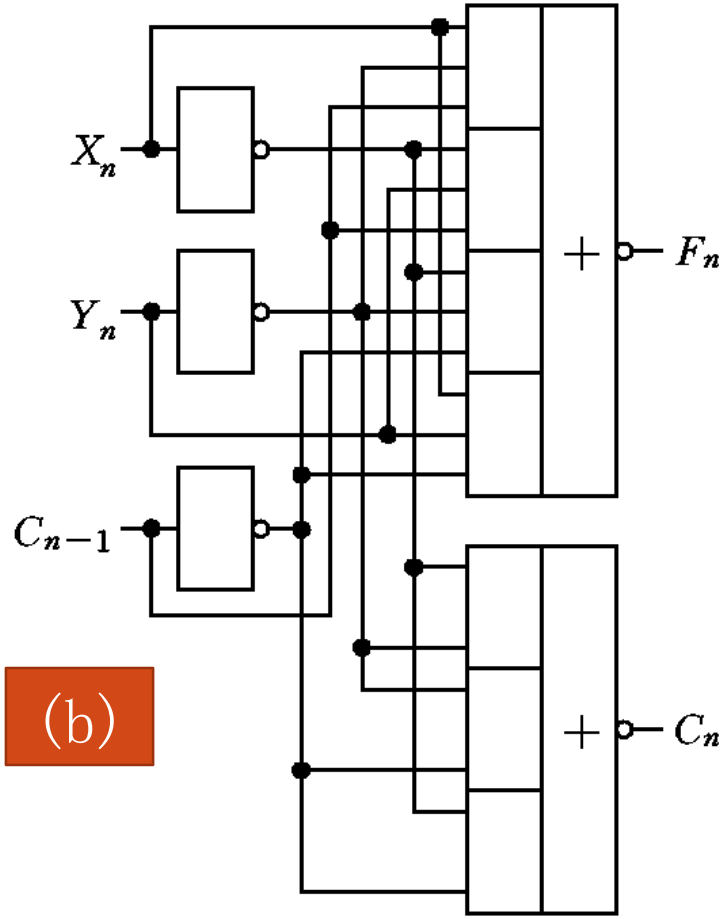


3.3.6 运算器

四种形式的全加器（1）



(a)



(b)

Cn和Fn的形成需要三级门延迟

Cn和Fn的形成需要二级门延迟

3.3.6 运算器

四种形式的1位全加器(2)

真值表

X_n	Y_n	C_{n-1}	F_n	C_n
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

分析全加器真值表中F_n和C_n的对应关系，F_n为“1”的条件有两个：

1、X_n Y_n C_{n-1}均为“1”

2、X_n Y_n C_{n-1}只有一个为“1”（有至少1个“1”且C_n为“0”）

用C_n表示F_n，先组合出C_n，再有F_n

$$F_n = X_nY_nC_{n-1} + X_n\overline{C_n} + Y_n\overline{C_n} + C_{n-1}\overline{C_n}$$

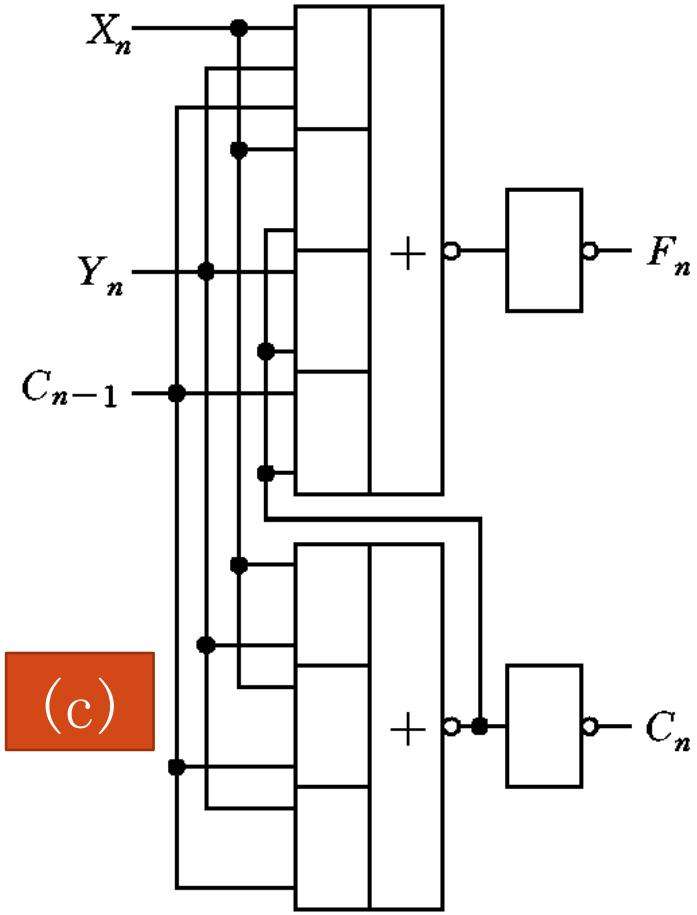
$$C_n = X_nY_n + (X_n + Y_n)C_{n-1}$$

$$F_n = \overline{\overline{X_n}\,\overline{Y_n}\,\overline{C_{n-1}}} + \overline{X_n}C_n + \overline{Y_n}C_n + \overline{C_{n-1}}C_n$$

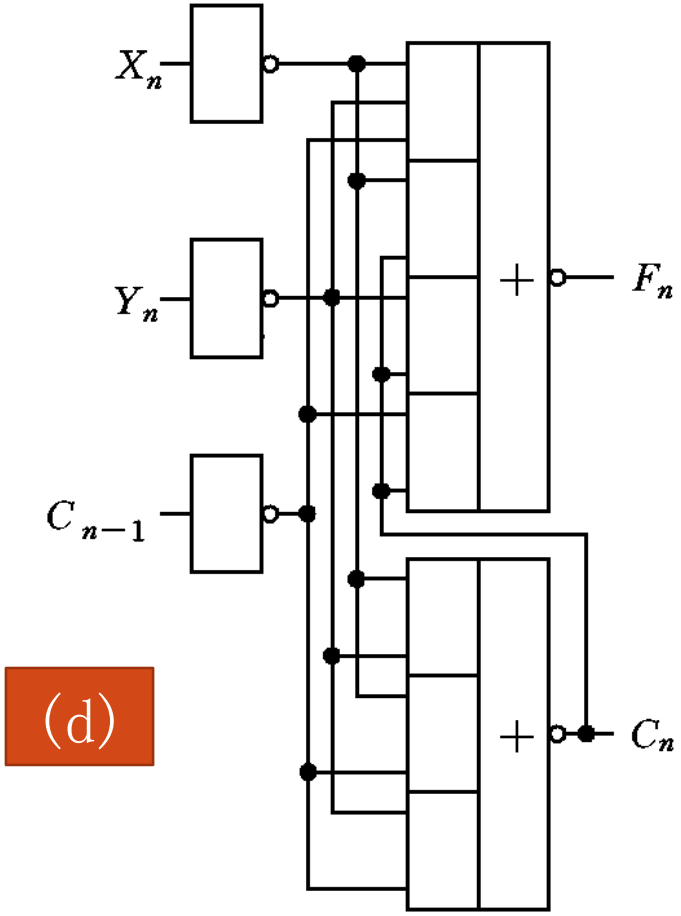
$$C_n = \overline{\overline{X_n}\,\overline{Y_n} + (\overline{X_n} + \overline{Y_n})\overline{C_{n-1}}}$$

3.3.6 运算器

四种形式的全加器（3）



Cn的形成需要二级门延迟
Fn的形成需要三级门延迟



Cn的形成需要二级门延迟
Fn的形成需要三级门延迟

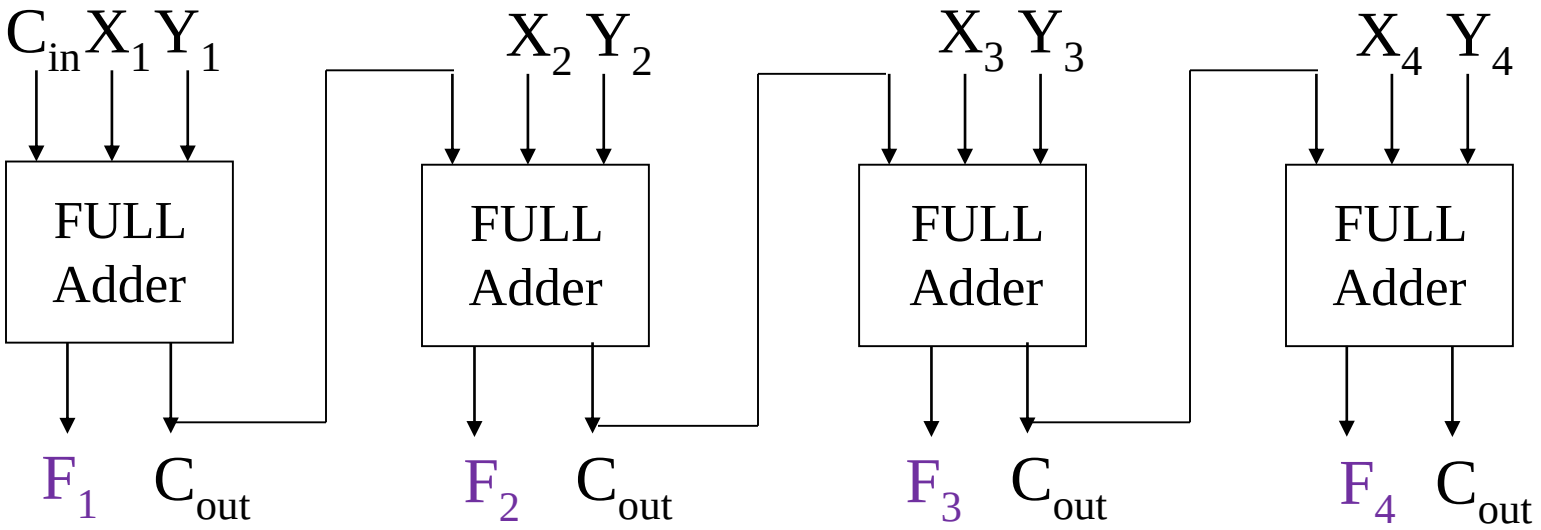
3.3.6 运算器

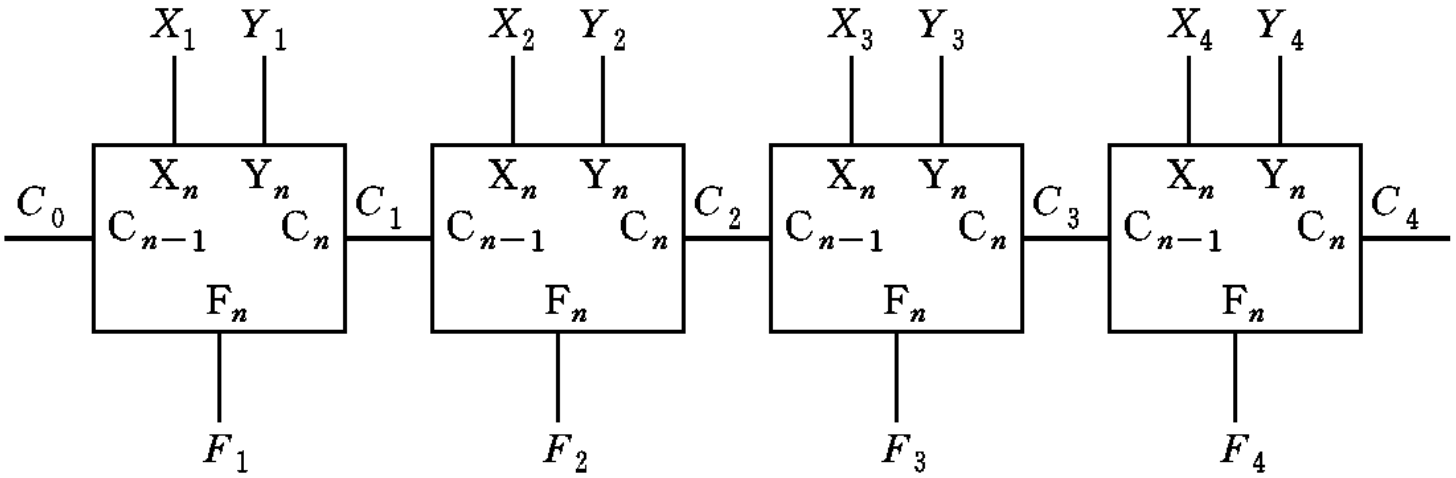
3.3.6 运算器（算数逻辑单元 ALU）

- 加法器
  - 一位加法器
  - 四位串行进位加法器
  - 快速加法器
  - 16位加法器
- 算术运算逻辑单元
  - 四位算术逻辑运算单元

3.3.6 运算器

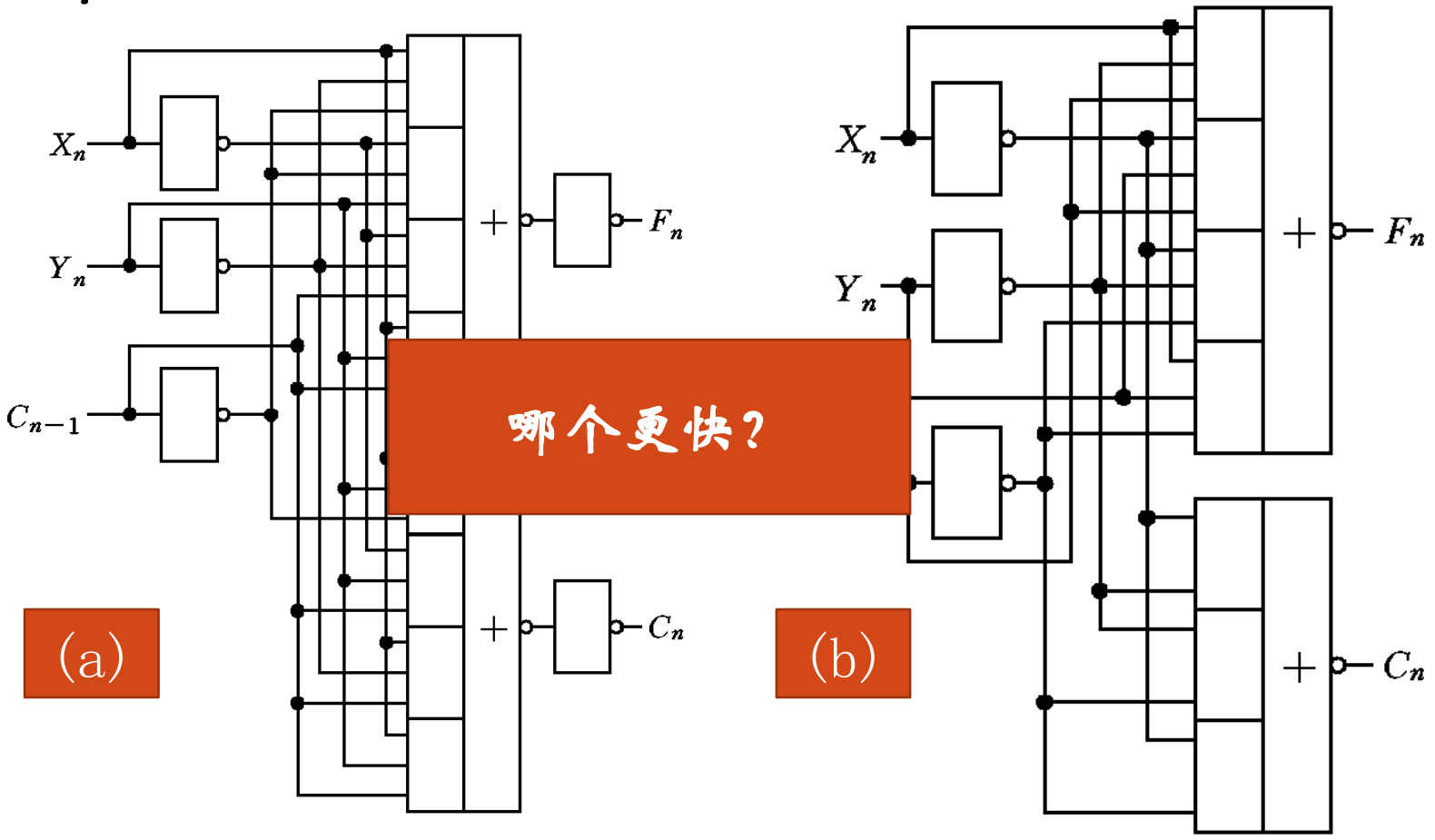
四位串行进位加法器





3.3.6 运算器

四种形式的全加器（1）

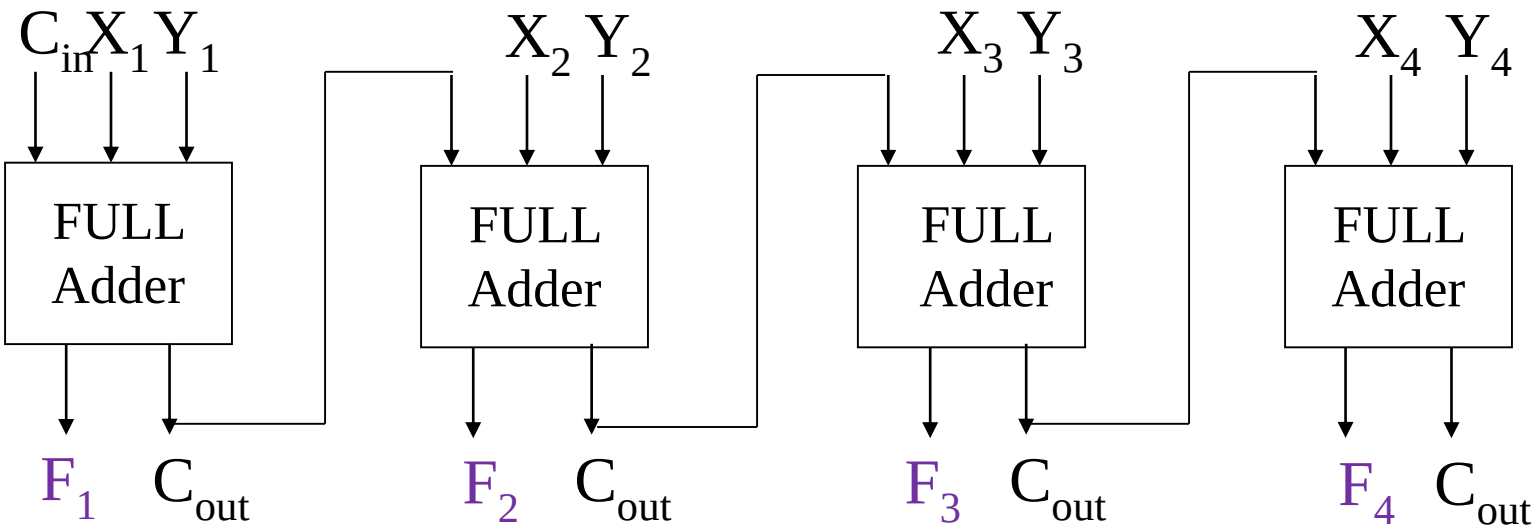


哪个更快？

Cn和Fn的形成需要三级门延迟　Cn和Fn的形成需要二级门延迟

3.3.6 运算器

四位串行进位加法器

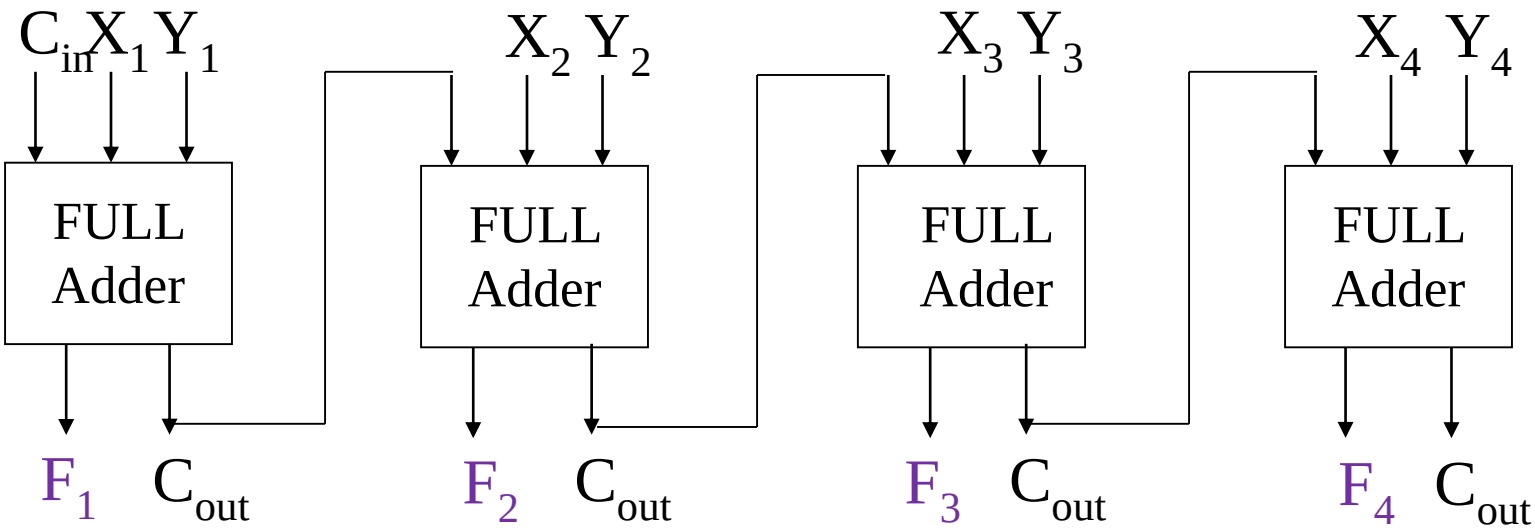


- 假设计算Fn需要2级，Cn需要2级，Cin、Xi和Yi同时到达。最后结果需要多少级延迟？

- F_1需要2级，C_1需要2级，F_2需要4级，C_2需要4级，F_3需要6级，C_3需要6级，F_4需要8级，C_4需要8级。

3.3.6 运算器

四位串行进位加法器

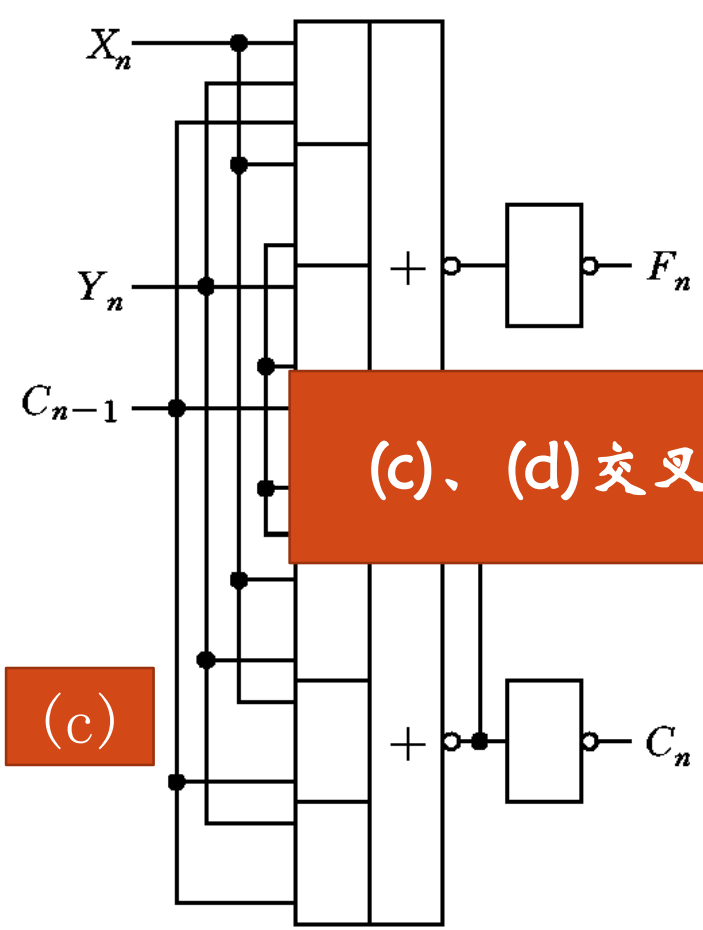


- 假设计算Fn需要3级，Cn需要2级，Cin、Xi和Yi同时到达。最后结果需要多少级延迟？

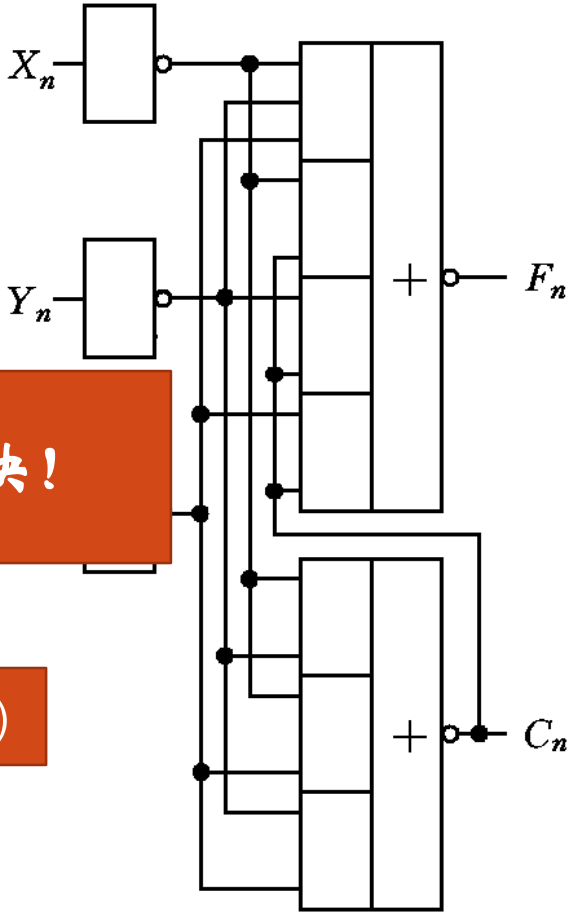
- F_1需要3级，C_1需要2级，F_2需要5级，C_2需要4级，F_3需要7级，C_3需要6级，F_4需要9级，C_4需要8级。

3.3.6 运算器

四种形式的全加器（3）



(c)



(d)

(c)、(d)交叉串联更快！

Cn的形成需要二级门延迟
Fn的形成需要三级门延迟

Cn的形成需要二级门延迟
Fn的形成需要三级门延迟

集成4位串行进位加法器

(c)、(d)两种一位加法器交叉串联，C4、F4经4级门延迟生成，不仅提高了速度，还节省了门的数量！

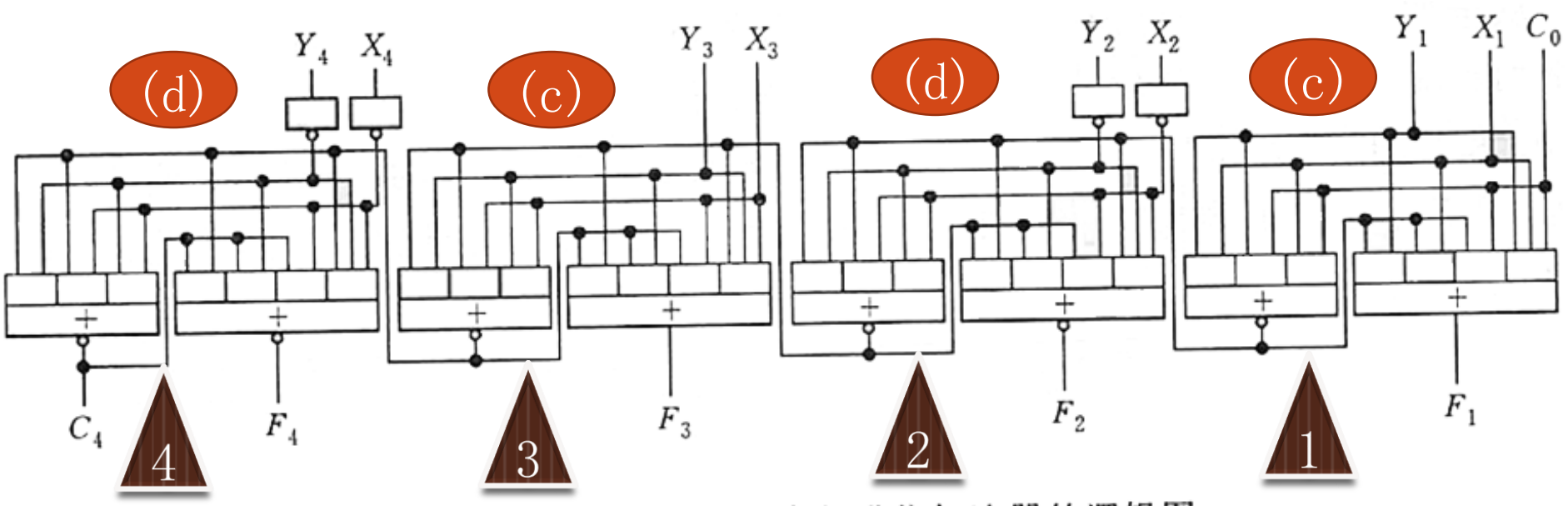
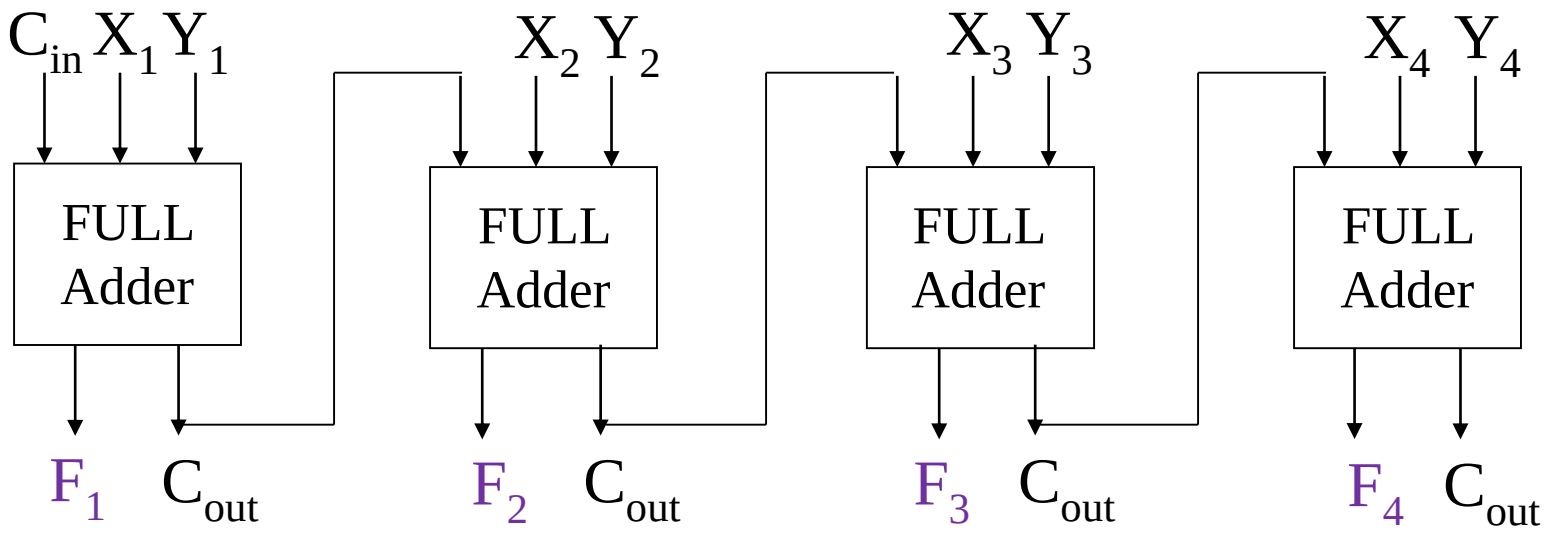


图 4-51　集成化的 4 位串行进位加法器的逻辑图

- F_1需要3级，C_1需要2级，F_2需要3级，C_2需要2级，F_3需要5级，C_3需要4级，F_4需要5级，C_4需要4级。

3.3.6 运算器(18)

- 四位串行进位加法器
  - 问题：由于前一个加法完成并提供进位后，下一个加法器才能开始运算。延迟长，速度慢。
  - 位数越多，加法完成的时间越长。
  - 是否可以用专用的进位电路提高速度？



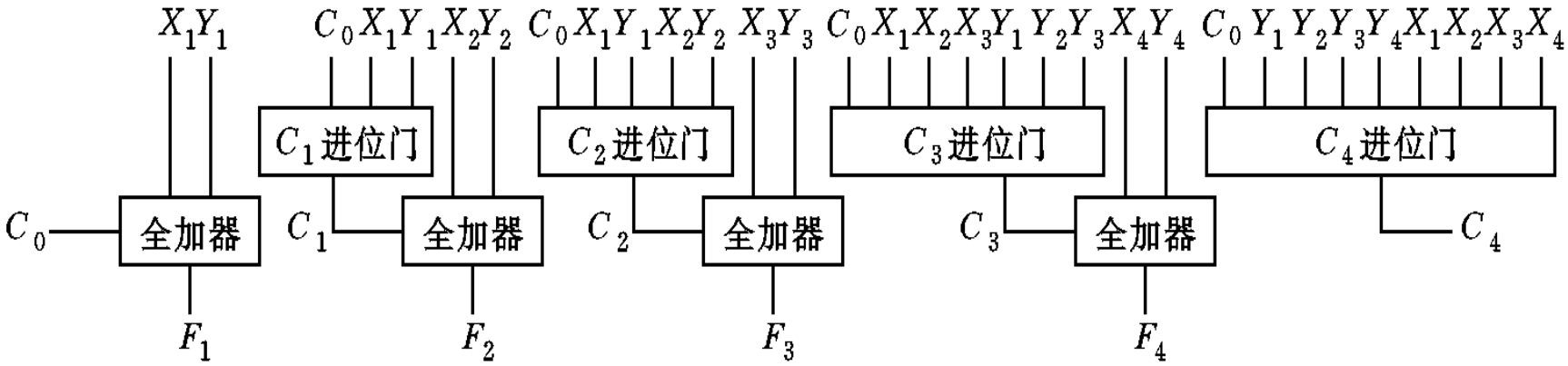
3.3.6 运算器(19)

3.3.6 运算器（算数逻辑单元 ALU）

- 加法器
  - 一位加法器
  - 四位串行进位加法器
  - 快速加法器
  - 16位加法器
- 算术运算逻辑单元
  - 四位算术逻辑运算单元

3.3.6 运算器(20)

- 快速加法器、超前进位加法器
  - 进位输入是由专门的“进位门”综合所有低位的加数、被加数及最低位进入输入后提供的。



C_1、C_2、C_3、C_4是怎样形成的？

3.3.6 运算器(21)

$G_i = X_iY_i$ 称为

产生进位函数

$P_i = X_i + Y_i$ 称为

传递进位函数

快速加法器、超前进位加法器

C_1、C_2、C_3、C_4形成的条件

$$C_1 = X_1Y_1 + (X_1 + Y_1)C_0$$

$$C_2 = X_2Y_2 + (X_2 + Y_2)X_1Y_1 + (X_2 + Y_2)(X_1 + Y_1)C_0$$

$$C_3 = X_3Y_3 + (X_3 + Y_3)X_2Y_2 + (X_3 + Y_3)(X_2 + Y_2)X_1Y_1 + (X_3 + Y_3)(X_2 + Y_2)(X_1 + Y_1)C_0$$

$$C_4 = X_4Y_4 + (X_4 + Y_4)X_3Y_3 + (X_4 + Y_4)(X_3 + Y_3)X_2Y_2 + (X_4 + Y_4)(X_3 + Y_4)(X_2 + Y_2)X_1Y_1 + (X_4 + Y_4)(X_3 + Y_3)(X_2 + Y_2)(X_1 + Y_1)C_0$$

3.3.6 运算器(22)

$$C_1 = X_1Y_1 + (X_1 + Y_1)C_0 = G_1 + P_1C_0$$

化简，得

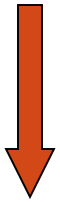
$$C_1 = \overline{\overline{X_1}\,\overline{Y_1} + (\overline{X_1} + \overline{Y_1})\overline{C_0}}$$

改写为

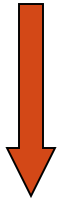
$$C_1 = \overline{\overline{X_1 + Y_1} + \overline{X_1Y_1}\,\overline{C_0}} = \overline{\overline{P_1} + \overline{G_1}\,\overline{C_0}}$$

3.3.6 运算器(23)

$$C_2 = X_2Y_2 + (X_2 + Y_2)X_1Y_1 + (X_2 + Y_2)(X_1 + Y_1)C_0$$
$$= G_2 + P_2G_1 + P_2P_1C_0$$

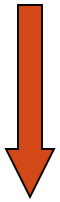


$$C_2 = \overline{\overline{X_2Y_2} + (\overline{X_2} + \overline{Y_2})X_1Y_1 + (\overline{X_2} + \overline{Y_2})(\overline{X_1} + \overline{Y_1})C_0}$$

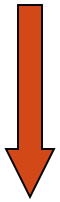


$$C_2 = \overline{\overline{X_2 + Y_2} + \overline{X_2Y_2(X_1 + Y_1)} + \overline{X_2Y_2X_1Y_1}C_0}$$
$$= \overline{\overline{P_2} + \overline{G_2P_1} + \overline{G_2G_1}C_0}$$

$$C_3 = X_3Y_3 + (X_3 + Y_3)X_2Y_2 + (X_3 + Y_3)(X_2 + Y_2)X_1Y_1$$
$$+ (X_3 + Y_3)(X_2 + Y_2)(X_1 + Y_1)C_0$$
$$= G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1C_0$$

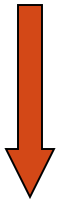


$$\overline{C_3} = \overline{X_3Y_3 + (X_3 + Y_3)X_2Y_2 + (X_3 + Y_3)(X_2 + Y_2)X_1Y_1}$$
$$\overline{+ (X_3 + Y_3)(X_2 + Y_2)(X_1 + Y_1)C_0}$$



$$C_3 = \overline{\overline{X_3 + Y_3} + \overline{X_3Y_3}(X_2 + Y_2)} + \overline{\overline{X_3Y_3}\,\overline{X_2Y_2}(X_1 + Y_1)} + \overline{\overline{X_3Y_3}\,\overline{X_2Y_2}\,\overline{X_1Y_1}C_0}$$
$$= \overline{\overline{P_3} + \overline{G_3}P_2} + \overline{\overline{G_3}\,\overline{G_2}P_1} + \overline{\overline{G_3}\,\overline{G_2}\,\overline{G_1}C_0}$$

$$C_4 = X_4Y_4 + (X_4 + Y_4)X_3Y_3 + (X_4 + Y_4)(X_3 + Y_3)X_2Y_2$$
$$+ (X_4+Y_4)(X_3 + Y_3)(X_2 + Y_2)X_1Y_1$$
$$+ (X_4 + Y_4)(X_3 + Y_3)(X_2 + Y_2)(X_1 + Y_1)C_0$$
$$= G_4 + P_4G_3 + P_4P_3G_2 + P_4P_3P_2G_1 + P_4P_3P_2P_1C_0$$



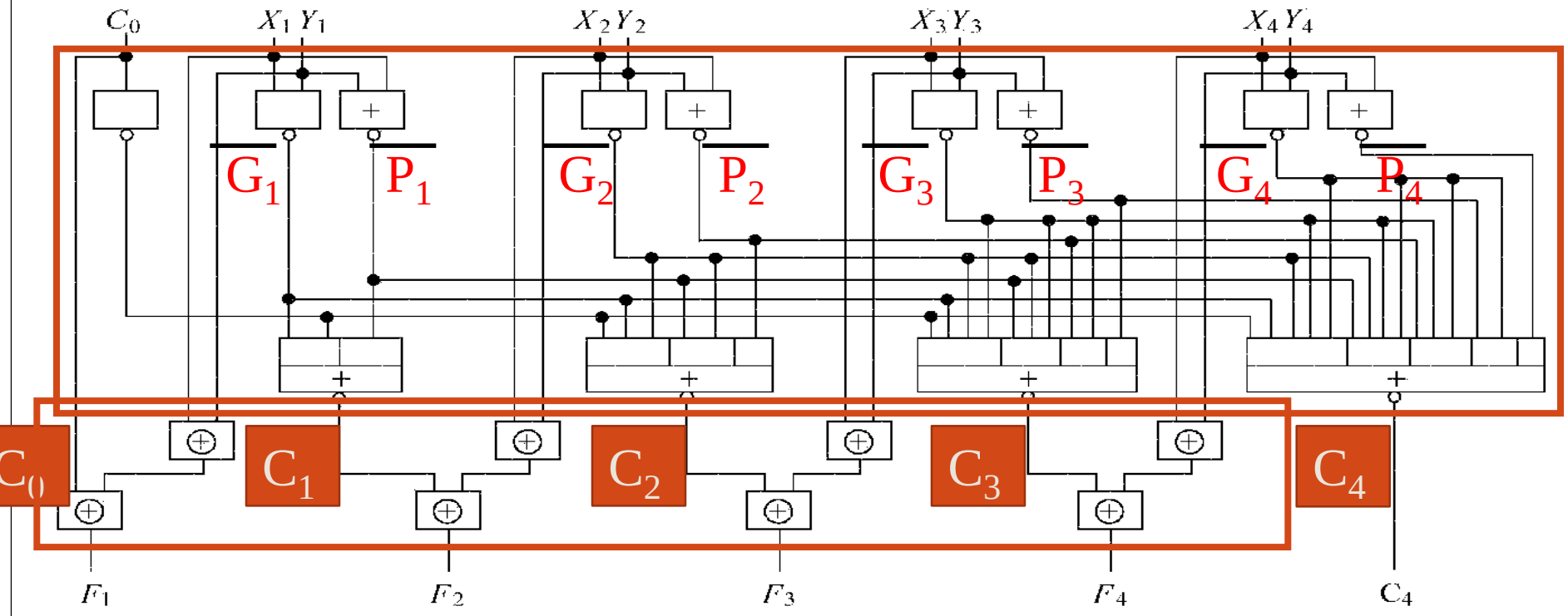
$$C_4 = \overline{\overline{X_4 + Y_4} + \overline{X_4Y_4}(X_3 + Y_3)} + \overline{X_4Y_4X_3Y_3}(X_2 + Y_2)$$
$$\overline{+ \overline{X_4Y_4X_3Y_3X_2Y_2}(X_1 + Y_1)} + \overline{X_4Y_4X_3Y_3X_2Y_2X_1Y_1C_0}$$
$$= \overline{\overline{P_4} + \overline{G_4P_3} + \overline{G_4G_3P_2} + \overline{G_4G_3G_2P_1} + \overline{G_4G_3G_2G_1C_0}}$$

$$C_1 = \overline{\overline{X_1 + Y_1 + X_1Y_1C_0}} = \overline{\overline{P_1 + G_1C_0}}$$

$$C_2 = \overline{\overline{X_2 + Y_2 + X_2Y_2(X_1 + Y_1) + X_2Y_2X_1Y_1C_0}} = \overline{\overline{P_2 + G_2P_1 + G_2G_1C_0}}$$

$$C_3 = \overline{\overline{X_3 + Y_3 + X_3Y_3(X_2 + Y_2) + X_3Y_3X_2Y_2(X_1 + Y_1) + X_3Y_3X_2Y_2X_1Y_1C_0}}$$

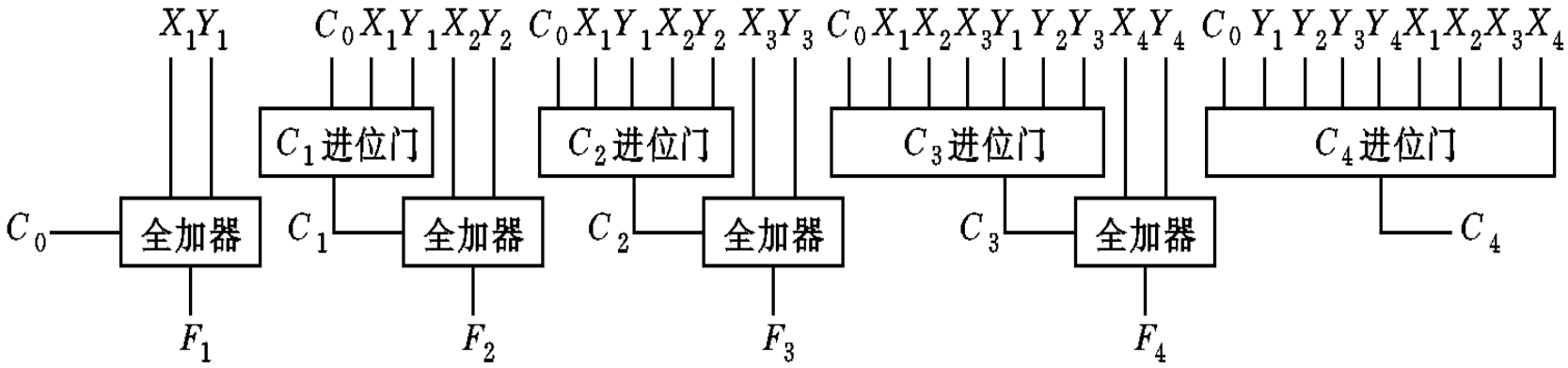
$$C_4 = \overline{\overline{X_4 + Y_4 + X_4Y_4(X_3 + Y_3) + X_4Y_4X_3Y_3(X_2 + Y_2) + X_4Y_4X_3Y_3X_2Y_2(X_1 + Y_1) + X_4Y_4X_3Y_3X_2Y_2X_1Y_1C_0}}$$



C_i延迟级数与位数无关：都是2级；F_i都是3级

3.3.6 运算器(24)

- 快速加法器、超前进位加法器
  - 进位输入是由专门的“进位门”综合所有低位的加数、被加数及最低位进位输入后提供的。
  - 由于进位不是由前一级加法器提供的，所以快速加法器又称超前进位加法器或并行进位加法器。

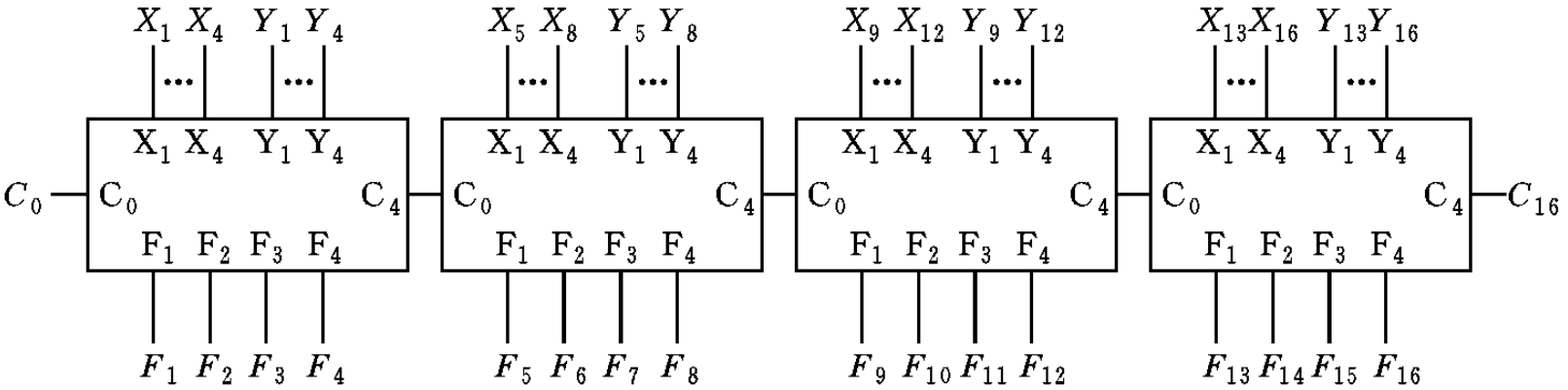


3.3.6 运算器(25)

- 3.3.6 运算器（算数逻辑单元 ALU）
  - 加法器
    - 一位加法器
    - 四位串行进位加法器
    - 快速加法器
    - 16位加法器
  - 算术运算逻辑单元
    - 四位算术逻辑运算单元

3.3.6 运算器(26)

- 16位串行加法器和16位并行加法器
  - 16位串行加法器
    - 用4片4位快速加法器组成16位快速加法器



此时,片内虽然是并行,但片间进位仍是串行逐片传递,

产生C_4, C_8, C_{12}, C_{16}的延迟各是几级? F_i要几级?(用P136图)

C_4需2级,F_1~F_4需3级; C_8需4级,F_5~F_8需5级; C_{12}需6级,

F_9~F_{12}需7级; C_{16}需8级,F_{13}~F_{16}需9级. $C_n=n/2$级

3.3.6 运算器(27)

- 16位并行进位加法器：

  用类似4位快速加法器中C1、C2、C3、C4形成的原理，去形成片间快速进位C4、C8、C12、C16

$$C_4 = (G_4 + P_4G_3 + P_4P_3G_2 + P_4P_3P_2G_1) + P_4P_3P_2P_1C_0$$
$$= G_{m1} + P_{m1}C_0$$

$$C_8 = G_8 + P_8G_7 + P_8P_7G_6 + P_8P_7P_6G_5 + P_8P_7P_6P_5C_4$$
$$= G_8 + P_8G_7 + P_8P_7G_6 + P_8P_7P_6G_5$$
$$+ P_8P_7P_6P_5(G_4 + P_4G_3 + P_4P_3G_2 + P_4P_3P_2G_1 + P_4P_3P_2P_1C_0)$$
$$= G_{m2} + P_{m2}(G_{m1} + P_{m1}C_0) = G_{m2} + P_{m2}G_{m1} + P_{m2}P_{m1}C_0$$

3.3.6 运算器(28)

$$C_{12} = G_{m3} + P_{m3}G_{m2} + P_{m3}P_{m2}G_{m1} + P_{m3}P_{m2}P_{m1}C_0$$

$$C_{16} = G_{m4} + P_{m4}G_{m3} + P_{m4}P_{m3}G_{m2} + P_{m4}P_{m3}P_{m2}G_{m1} + P_{m4}P_{m3}P_{m2}P_{m1}C_0$$

3.3.6 运算器(29)

$$C_4 = G_{m1} + P_{m1}C_0$$

$$C_8 = G_{m2} + P_{m2}G_{m1} + P_{m2}P_{m1}C_0$$

$$C_{12} = G_{m3} + P_{m3}G_{m2} + P_{m3}P_{m2}G_{m1} + P_{m3}P_{m2}P_{m1}C_0$$

$$C_{16} = G_{m4} + P_{m4}G_{m3} + P_{m4}P_{m3}G_{m2} + P_{m4}P_{m3}P_{m2}G_{m1} + P_{m4}P_{m3}P_{m2}P_{m1}C_0$$

$$G_{m1} = G_4 + P_4G_3 + P_4P_3G_2 + P_4P_3P_2G_1 \qquad P_{m1} = P_4P_3P_2P_1$$

$$G_{m2} = G_8 + P_8G_7 + P_8P_7G_6 + P_8P_7P_6G_5 \qquad P_{m2} = P_8P_7P_6P_5$$

$$G_{m3} = G_{12} + P_{12}G_{11} + P_{12}P_{11}G_{10} + P_{12}P_{11}P_{10}G_9 \qquad P_{m3} = P_{12}P_{11}P_{10}P_9$$

$$G_{m4} = G_{16} + P_{16}G_{15} + P_{16}P_{15}G_{14} + P_{16}P_{15}P_{14}G_{13} \qquad P_{m4} = P_{16}P_{15}P_{14}P_{13}$$

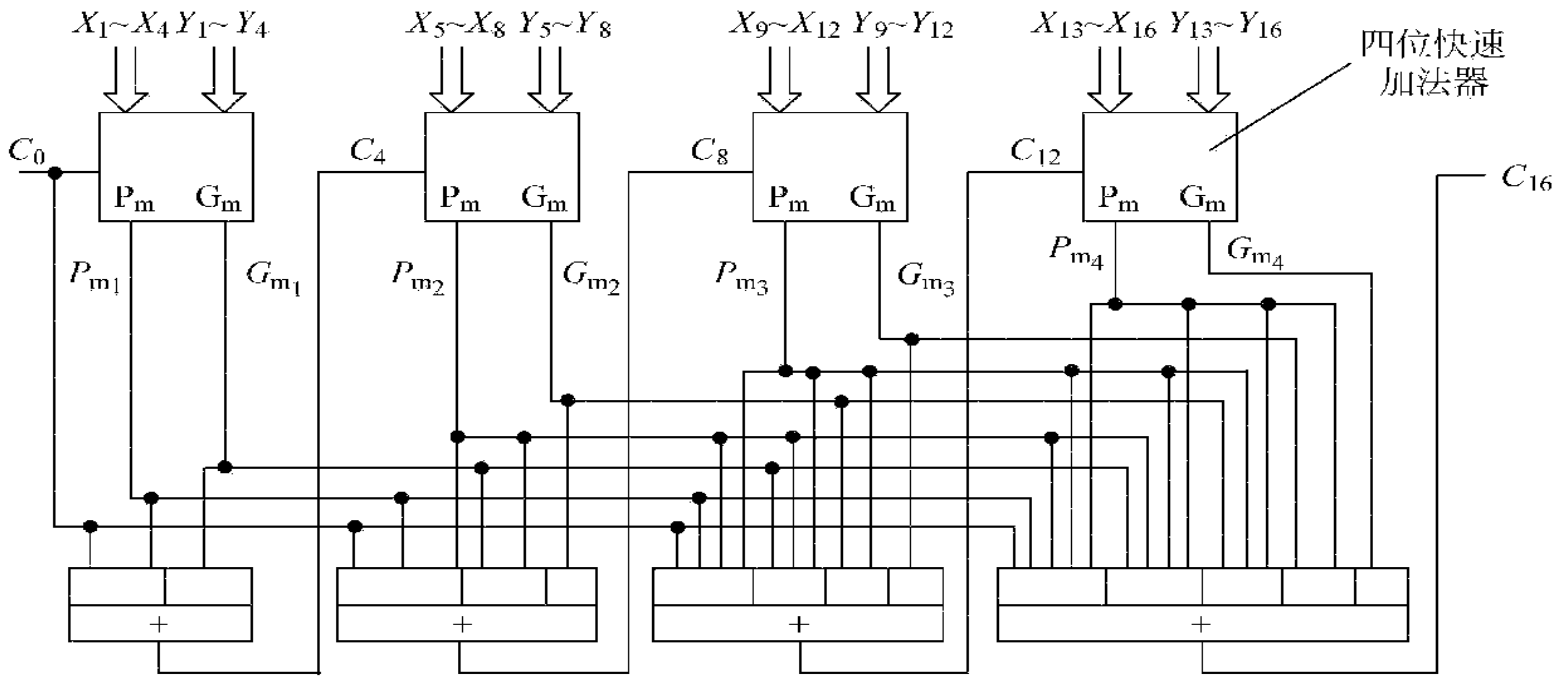
Gm和Pm有规律，可以根据输入X和Y直接计算得出，意味着4位加法器内部可以直接提供Gm和Pm。C4，C8，C12，C16可以设计一个“快速进位扩展器”形成。

3.3.6 运算器

16位并行进位加法器：

超前进位扩展器

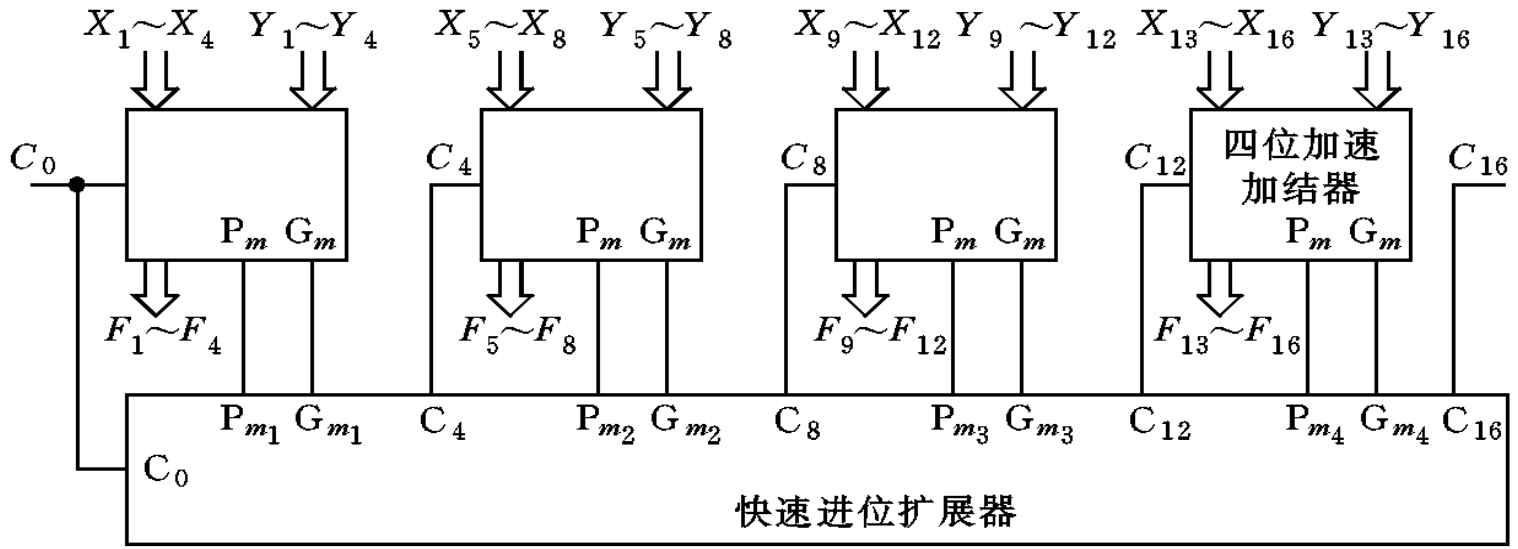
$$C_{16} = G_{m4} + P_{m4}G_{m3} + P_{m4}P_{m3}G_{m2} + P_{m4}P_{m3}P_{m2}G_{m1} + P_{m4}P_{m3}P_{m2}P_{m1}C_0$$



超前进位扩展器，使得C_4, C_8, C_{12}, C_{16}同时产生！（P138）

3.3.6 运算器

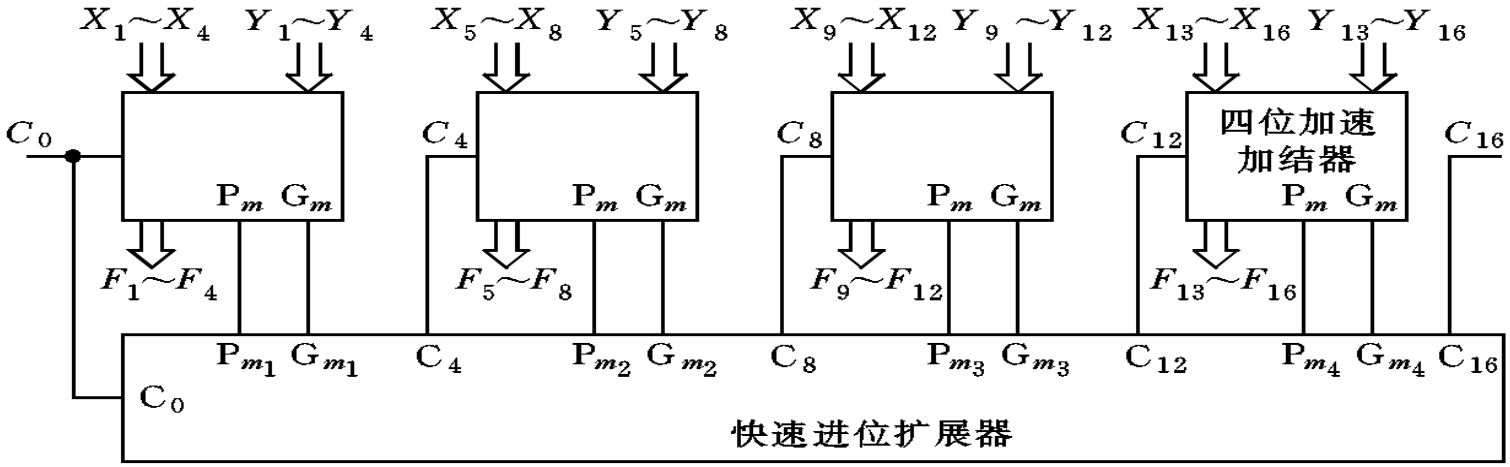
- 16位并行进位加法器逻辑结构图：
  - 用类似四位快速加法器中C1、C2、C3、C4形成的原理，去形成片间快速进位C4、C8、C12、C16



4位快速加法器的输出提供P_m、G_m需2级延迟，F需3级延迟。则产生C_4, C_8, C_{12}, C_{16}的延迟各是几级？F_i要几级？

3.3.6 运算器

- 16位快速加法器的结构图

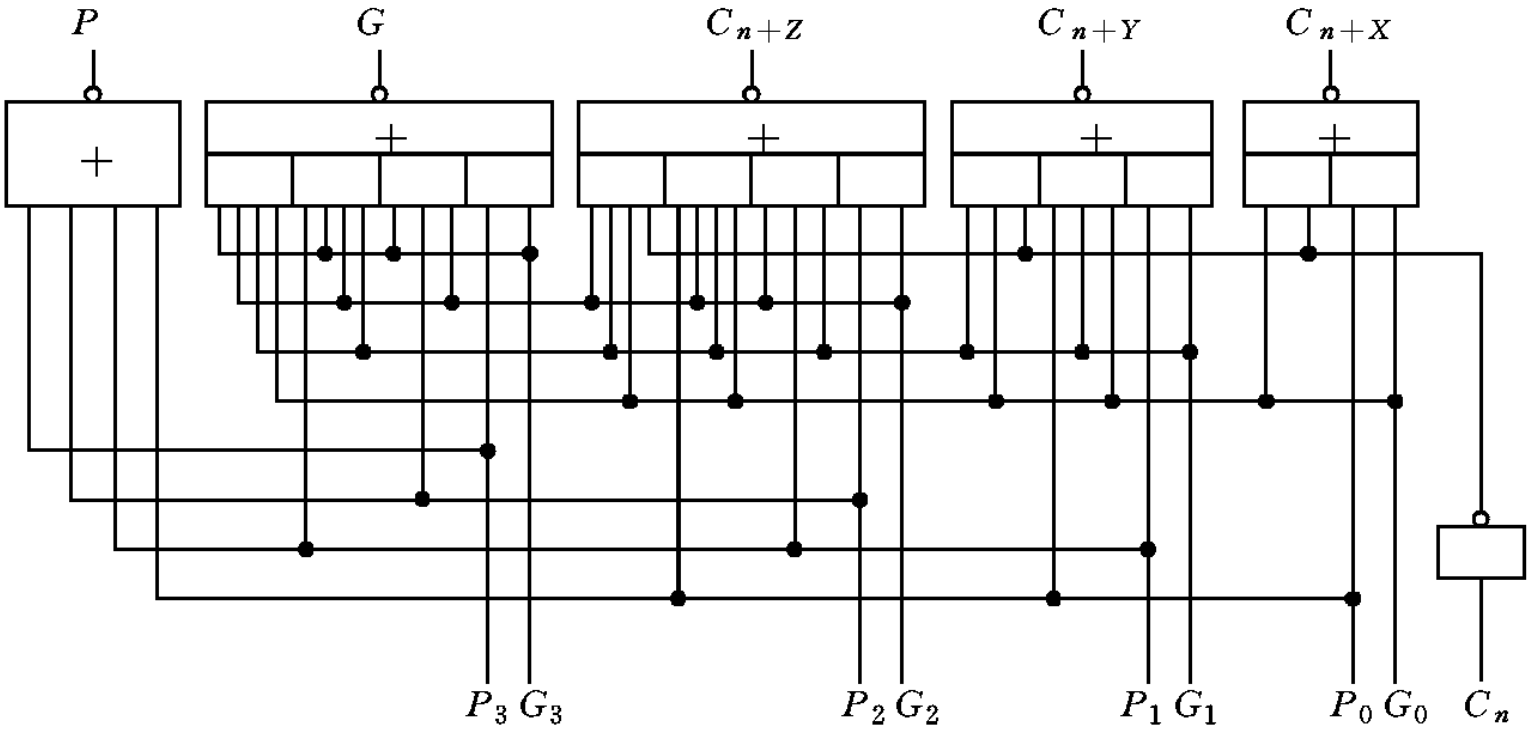


4位快速加法器的输出提供P_m、G_m需2级延迟。产生C_4, C_8, C_{12}, C_{16}的延迟3级，$F_1 \sim F_4$要3级，$F_5 \sim F_{16}$要6级。

请大家课后比较：由1位全加器构成16位串行加法器、4位全加器构成16位串行加法器、16位并行加法器计算结果所需要的级数。

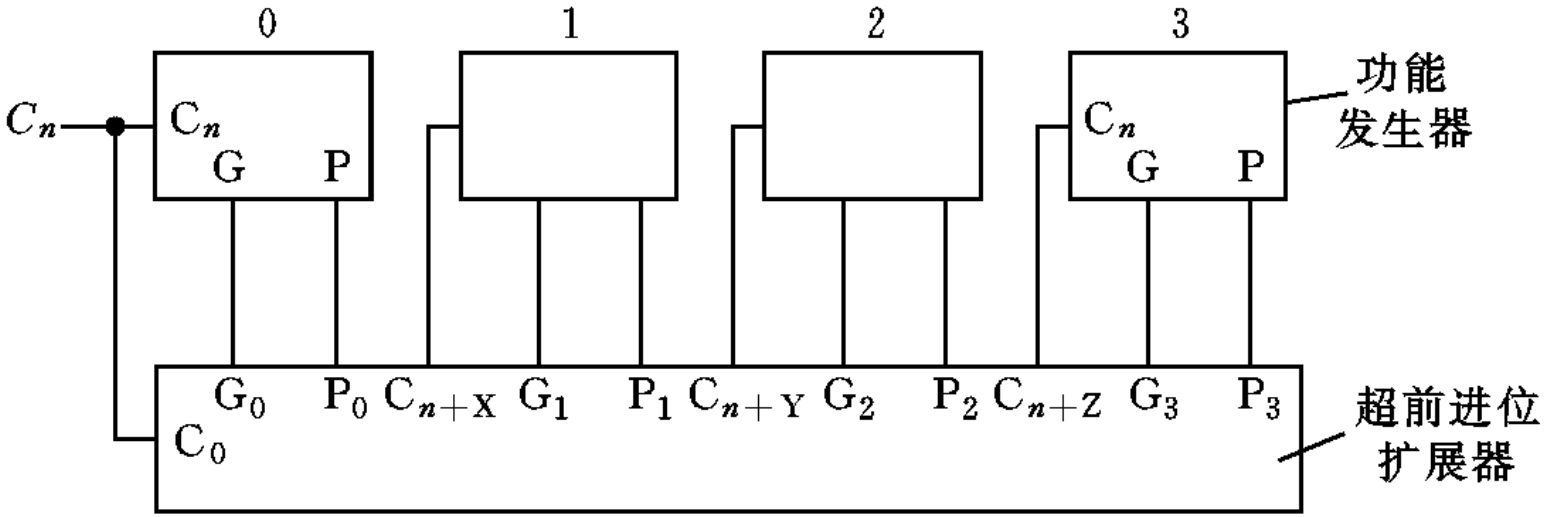
3.3.6 运算器

- 和4位加法器（或4位ALU）连用的超前进位扩展器专用器件(SN74182)（P146）



3.3.6 运算器

- 4位ALU和超前进位扩展器组成16位快速运算单元（P146）



3.3.6 运算器

3.3.6 运算器（算数逻辑单元 ALU）

- 加法器
  - 1位加法器
  - 4位串行进位加法器
  - 4位并行进位加法器和快速进位逻辑
  - 16位并行进位加法器
- 算术运算逻辑单元
  - 4位算术逻辑运算单元

3.3.6 运算器(34)

- 算术运算逻辑单元(ALU)
  - ALU是CPU的核心,不仅完成算术（加法、减法等）运算,而且完成逻辑运算.
  - ALU是多种功能集成在一起的器件,因此要有功能控制端.
    - 加
    - 减
    - 比较、与、或、非、移位

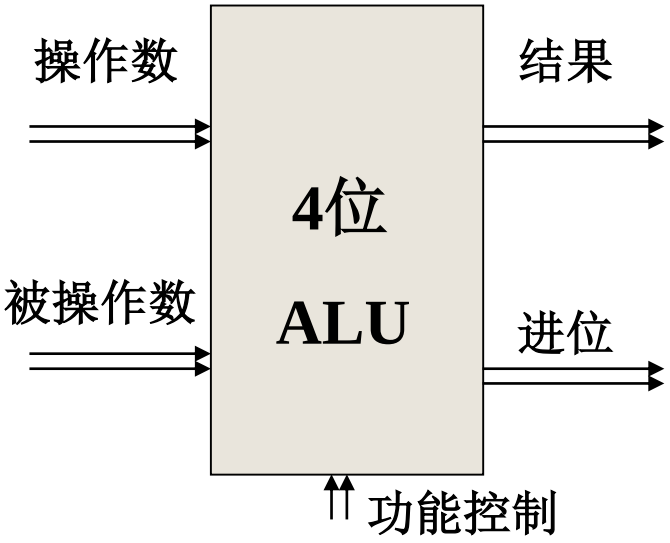
3.3.6 运算器(35)

3.3.6 运算器（算数逻辑单元 ALU）

- 加法器
  - 一位加法器
  - 四位串行进位加法器
  - 快速加法器
  - 16位加法器
- 算术运算逻辑单元
  - 4位算术逻辑运算单元

3.3.6 运算器(36)

- 四位算术逻辑运算单元
  - 4位ALU的核心是4位并行加法器,通过控制加法器的一些逻辑门或改变进位逻辑门能够获得多种功能。



3.3.6 运算器(37)

- 4位算术逻辑运算单元
  - 运算能力的获得方法
    - 方法1：控制4位加法器中的进位逻辑，获得多种运算能力
      - 简单、但运算种类少
    - 方法2：改变加法器中的Gi和Pi来获得多种运算能力
      - 运算种类多、但运算单元结构较复杂。

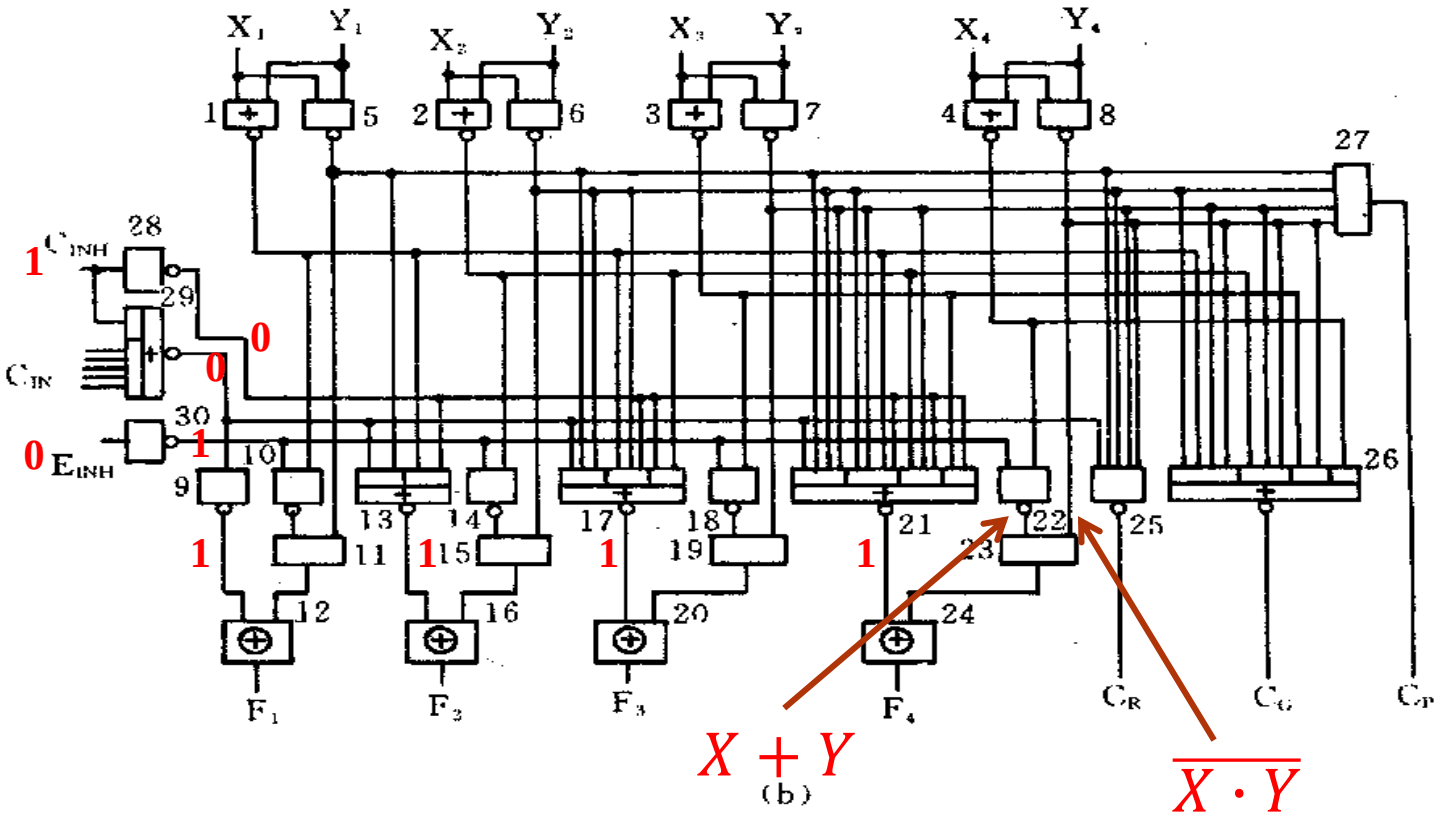
3.3.6 运算器(38)

- 方法1：控制4位加法器中的进位逻辑，获得多种运算能力。
  - 简单、运算种类较少。
  - 在控制信号C_{INH}、E_{INH}的作用下，可以完成4位数的加、比较和逻辑乘（与）运算。
  - 如图4－57所示（P139）

功能表

C_{INH}	E_{INH}	功　能
0	0	加
0	1	不用
1	0	$\overline{X_n \oplus Y_n}$
1	1	$X_n \cdot Y_n$

(a)



(b)

图 4-57　四位算术逻辑运算单元

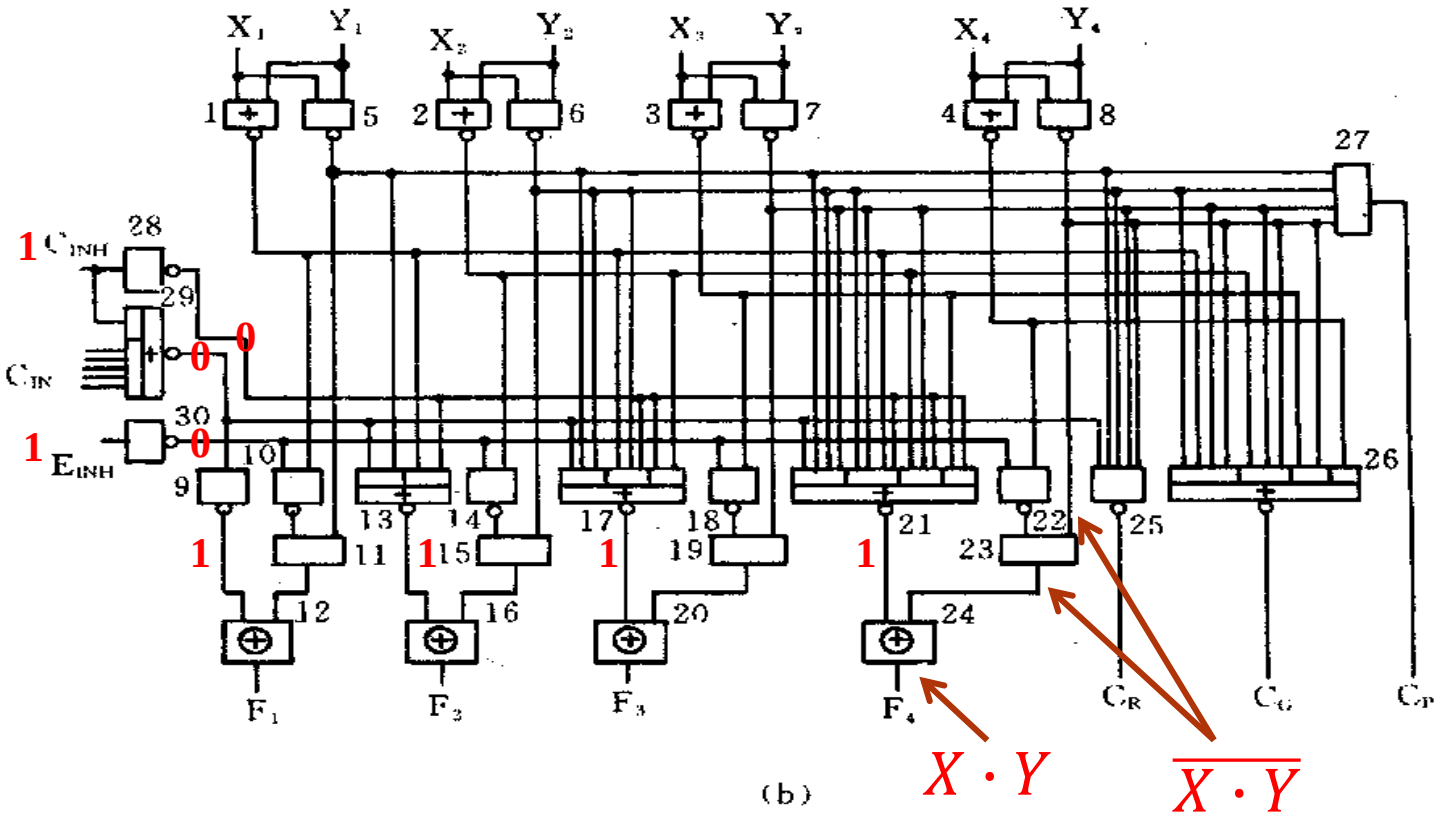
(a) 功能表；　(b) 逻辑图。

电路1: 功能控制加在进位门和半加器

功能表

C_{INH}	E_{INH}	功　能
0	0	加
0	1	不用
1	0	$\overline{X_n \oplus Y_n}$
1	1	$X_n \cdot Y_n$

(a)



(b)

图 4-57　四位算术逻辑运算单元

(a) 功能表；　(b) 逻辑图。

电路1: 功能控制加在进位门和半加器

3.3.6 运算器(39)

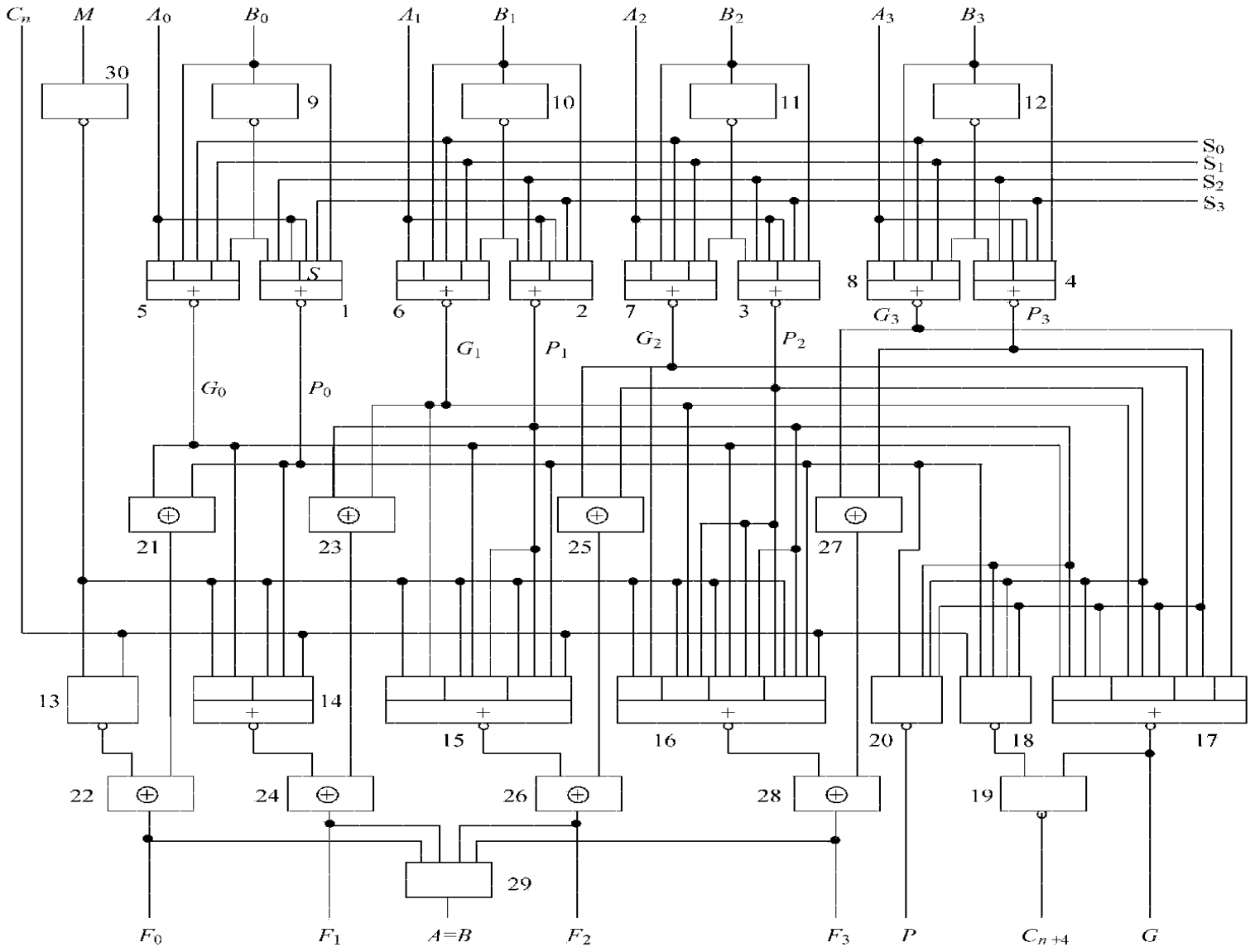
- 方法2：改变加法器中的Gi和Pi来获得多种运算能力。
  - 运算种类多、但运算单元结构较复杂。
  - 图4－59给出了这类集成化4位运算单元（简称功能发生器）的逻辑图。
  - 它能执行16种算术运算和16种逻辑运算。

3.3.6 运算器(40)

4位ALU功能表 (SN74181)

M是状态控制端，M＝”H”，执行逻辑运算，反之，执行算术运算。

S_3 S_2 S_1 S_0	正逻辑 M=H 逻辑运算	正逻辑 M=L 算术运算 C_n=1	正逻辑 M=L 算术运算 C_n=0	负逻辑 M=H 逻辑运算	负逻辑 M=L 算术运算 C_n=1	负逻辑 M=L 算术运算 C_n=0
L L L L	$\overline{A}$	A	A 加 1	$\overline{A}$	A 减 1	A
L L L H	$\overline{A+B}$	$A+B$	$(A+B)$加 1	$\overline{A \cdot B}$	$(A \cdot B)$减 1	$A \cdot B$
L L H L	$\overline{A} \cdot B$	$A+\overline{B}$	$(A+\overline{B})$加 1	$\overline{A}+B$	$(A \cdot \overline{B})$减 1	$A \cdot \overline{B}$
L L H H	“0”	减 1	0	1	减 1	0
L H L L	$\overline{A \cdot B}$	A 加$(A \cdot \overline{B})$	A 加$(A \cdot \overline{B})$加 1	$\overline{A+B}$	A 加$(A+\overline{B})$	A 加$(A+\overline{B}$ 加 1
L H L H	$\overline{B}$	$(A \cdot \overline{B})$加$(A+B)$	$(A \cdot \overline{B})$加$(A+B)$加 1	$\overline{B}$	$(A \cdot B)$加$(A+\overline{B})$	$(A \cdot B)$加$(A+\overline{B})$加 1
L H H L	$A \oplus B$	A 减 B 减 1	A 减 B	$\overline{A \oplus B}$	A 减 B 减 1	A 减 B
L H H H	$A \cdot \overline{B}$	$(A \cdot \overline{B})$减 1	$A \cdot \overline{B}$	$A+\overline{B}$	$A+\overline{B}$	$(A+\overline{B})$加 1
H L L L	$\overline{A}+B$	A 加$(A \cdot B)$	A 加$(A \cdot B)$加 1	$\overline{A} \cdot B$	A 加$(A+B)$	A 加$(A+B)$加 1
H L L H	$\overline{A \oplus B}$	A 加 B	A 加 B 加 1	$A \oplus B$	A 加 B	A 加 B 加 1
H L H L	B	$(A \cdot B)$加$(A+\overline{B})$	$(A \cdot B)$加$(A+\overline{B})$加 1	B	$(A \cdot \overline{B})$加$(A+B)$	$(A \cdot \overline{B})$加$(A+B)$加 1
H L H H	$A \cdot B$	$(A \cdot B)$减 1	$A \cdot B$	$A+B$	$A+B$	$(A+B)$加 1
H H L L	1	A 加 A	A 加 A 加 1	0	A 加 A	A 加 A 加 1
H H L H	$A+\overline{B}$	A 加$(A+B)$	A 加$(A+B)$加 1	$A \cdot \overline{B}$	A 加$(A \cdot B)$	A 加$(A \cdot B)$加 1
H H H L	$A+B$	A 加$(A+\overline{B})$	A 加$(A+\overline{B})$加 1	$A \cdot B$	A 加$(A \cdot \overline{B})$	A 加$(A \cdot \overline{B})$加 1
H H H H	A	A 减 1	A	A	A	A 加 1



74181

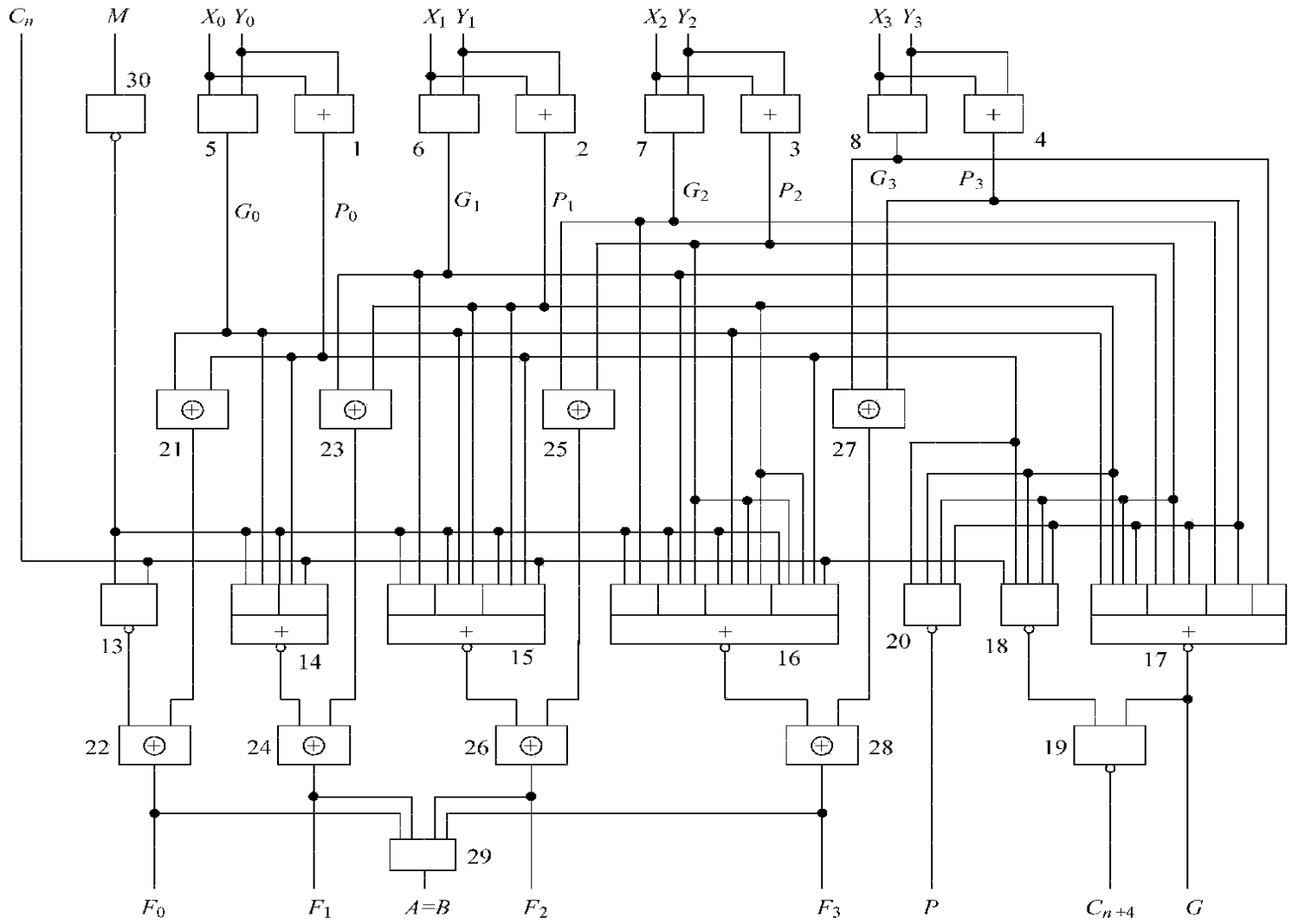
P141

电路2：功能控制加在输入端，改变进位产生函数G_i和进位传递函数P_i

4位ALU (SN74181)

（控制端 = HLLH）

简化逻辑图



4位ALU功能

1. M=L （控制端 = HLLH）

门13输出 $\overline{C_n}$

门14输出 $\overline{G_0 + P_0 C_n} = \overline{C_0}$

门15输出 $\overline{G_1 + P_1 G_0 + P_1 P_0 C_n} = \overline{C_1}$

门16输出 $\overline{G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_n} = \overline{C_2}$

门19输出 $G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_n = C_3$

门13~16形成以$X_{0\sim3}$、$Y_{0\sim3}$、C_n为输入的四位快速加法器的各进位的反码

4位ALU功能

门19形成的是第3位向第4位进位的原码

门21、23、25、27形成以X_n、Y_n为输入的半加和

门22 $\overline{C_n \oplus (X_0 \oplus Y_0)} = C_n \oplus (X_0 \oplus Y_0) =$ 第0位全加和

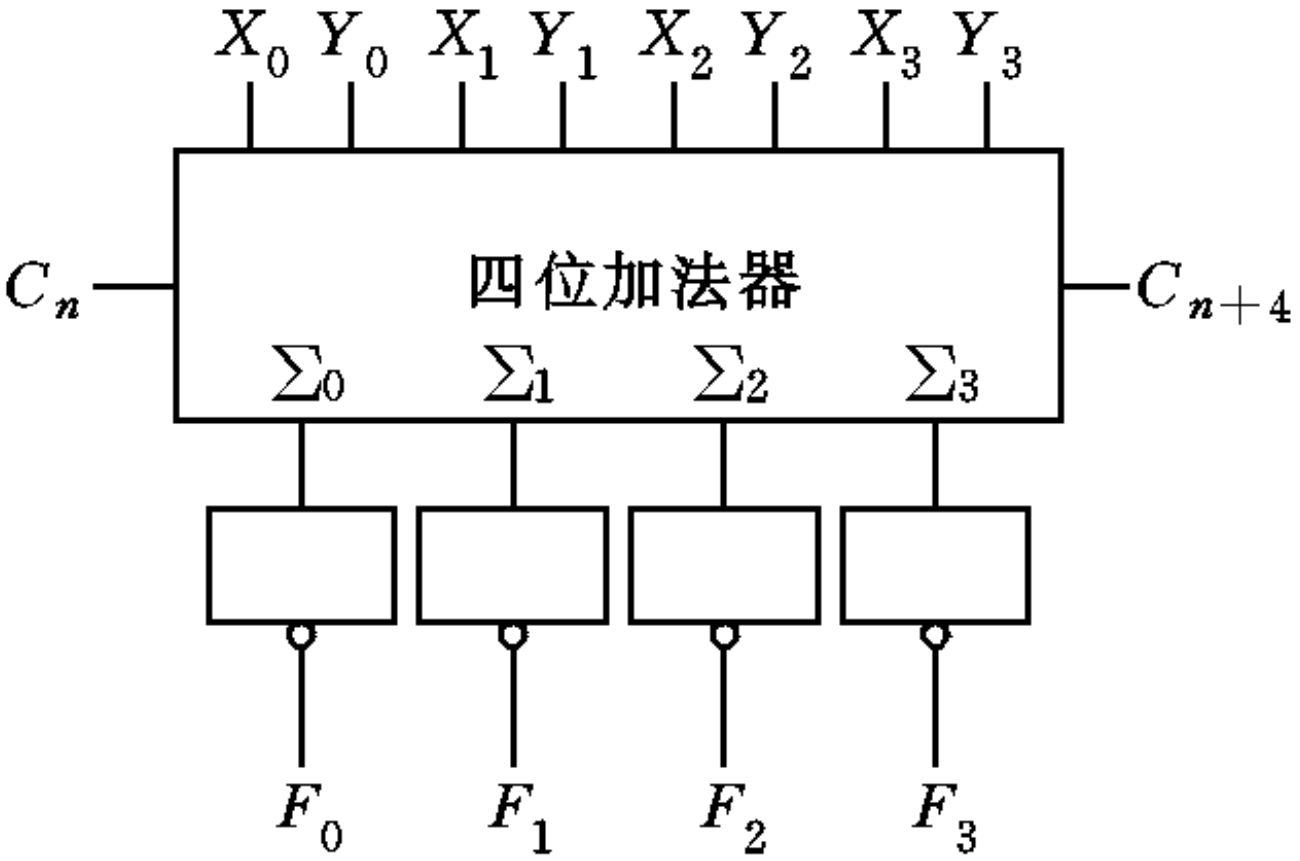
门24 $\overline{C_0 \oplus (X_1 \oplus Y_1)} =$ 第1位全加和

门26 $\overline{C_1 \oplus (X_2 \oplus Y_2)} =$ 第2位全加和

门28 $\overline{C_2 \oplus (X_3 \oplus Y_3)} =$ 第3位全加和

F_1--F_4是全加和的反码

4位ALU功能


X_0 Y_0 X_1 Y_1 X_2 Y_2 X_3 Y_3
C_n
四位加法器
C_{n+4}
Σ_0 Σ_1 Σ_2 Σ_3
F_0 F_1 F_2 F_3

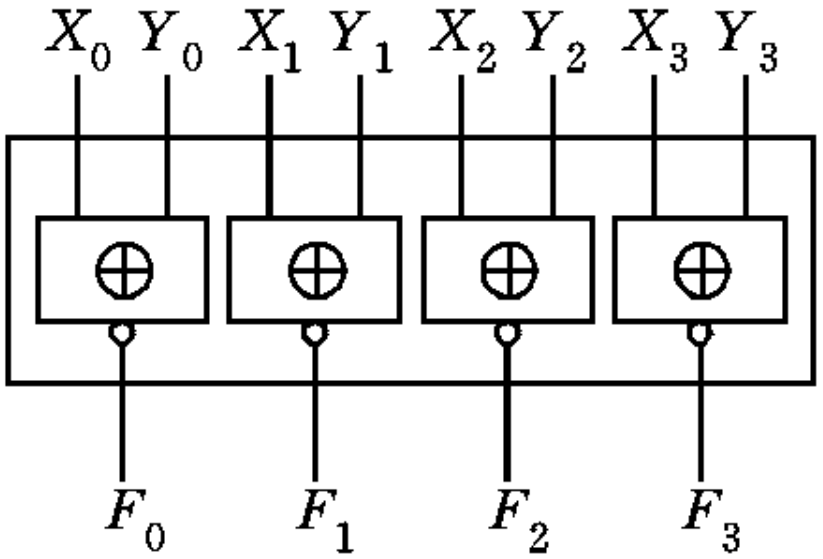
4位ALU功能

2. M=H（控制端 = HLLH）

门13~16均被封锁，位间没有操作，电路执行逻辑运算

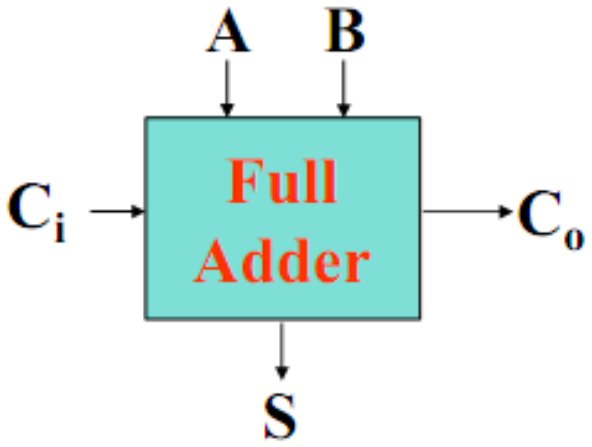
电路输出为 $F_n = \overline{X_n \oplus Y_n}$

电路是异或非逻辑（比较器）.



MIT数字逻辑电路课中的ALU课件

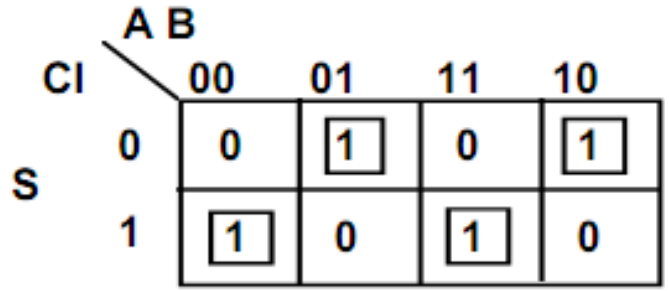
Binary Full Adder

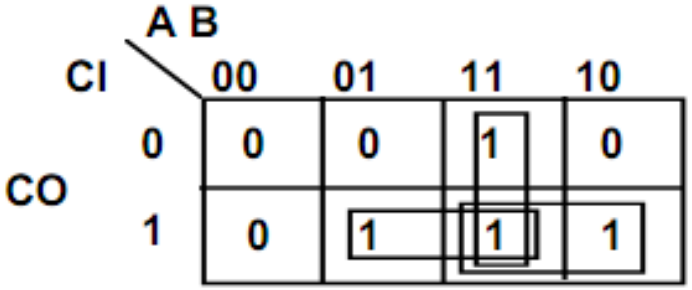


$$S = A \oplus B \oplus C_i$$
$$= A\bar{B}\bar{C}_i + \bar{A}B\bar{C}_i + \bar{A}\bar{B}C_i + ABC_i$$

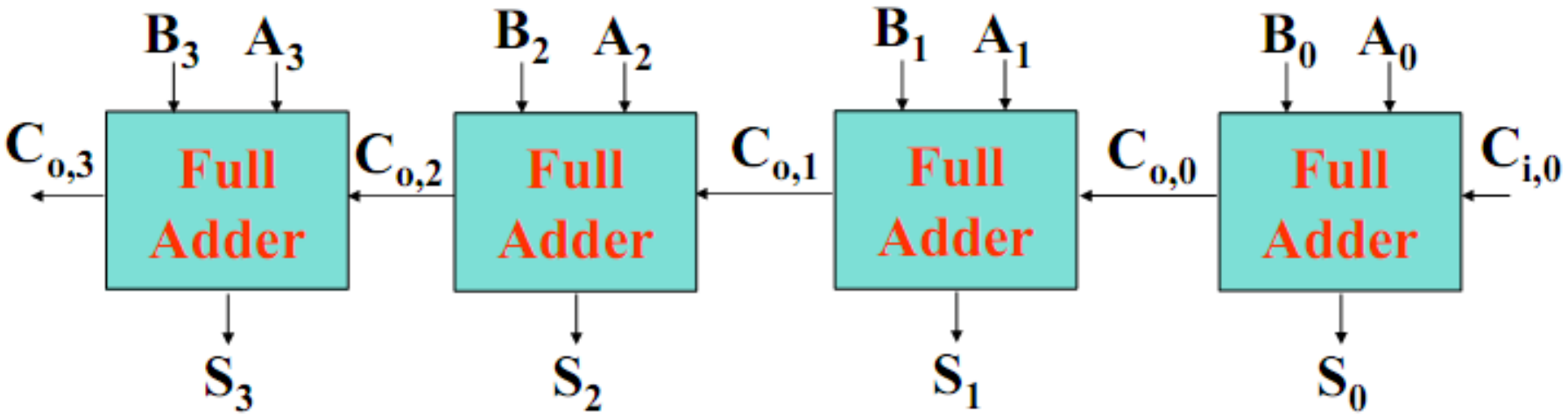
$$C_o = AB + C_i(A+B)$$

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





Ripple Carry Adder Structure



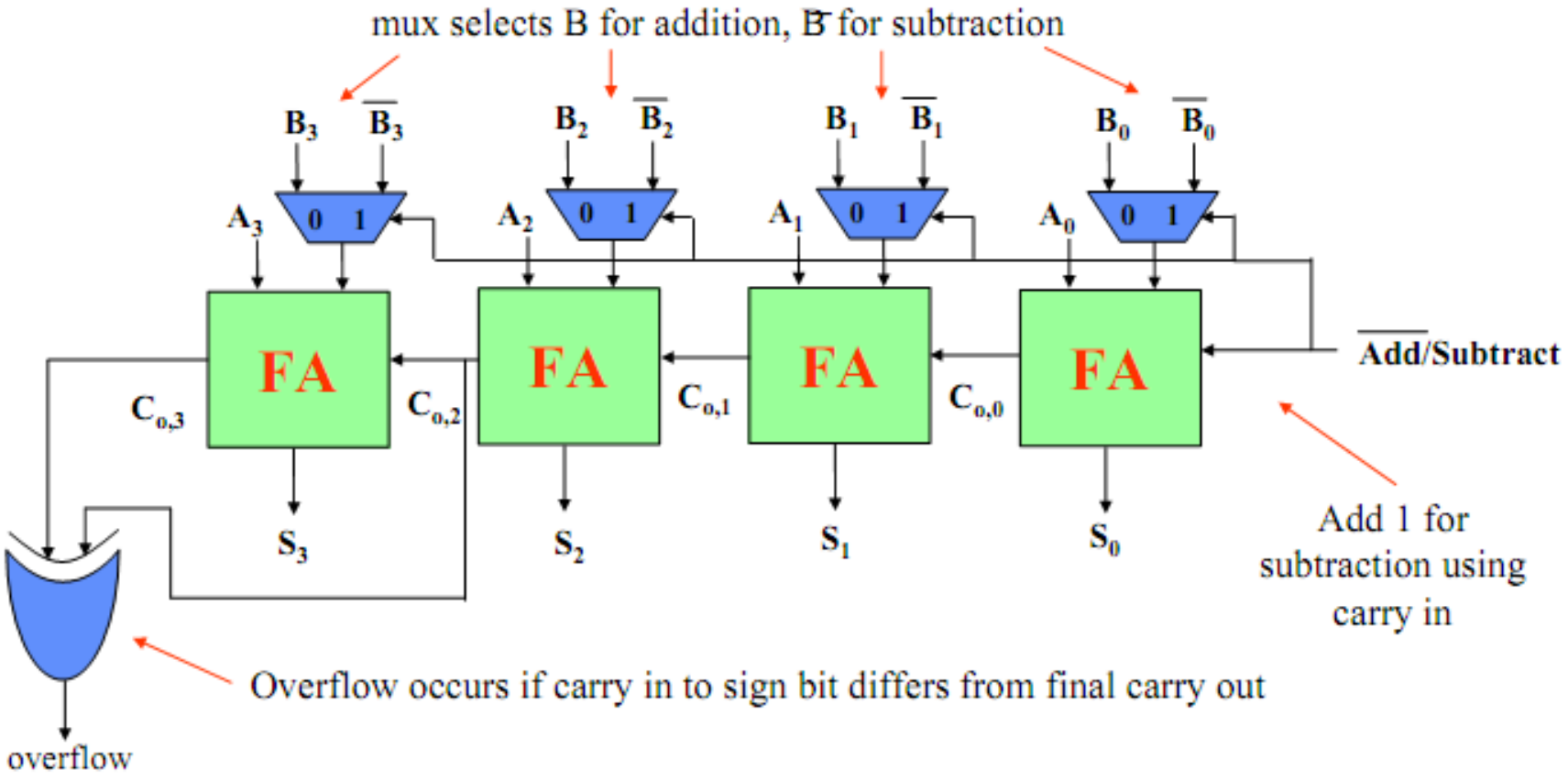
Worst case propagation delay linear with the number of bits

$$t_{adder} = (N-1)t_{carry} + t_{sum}$$

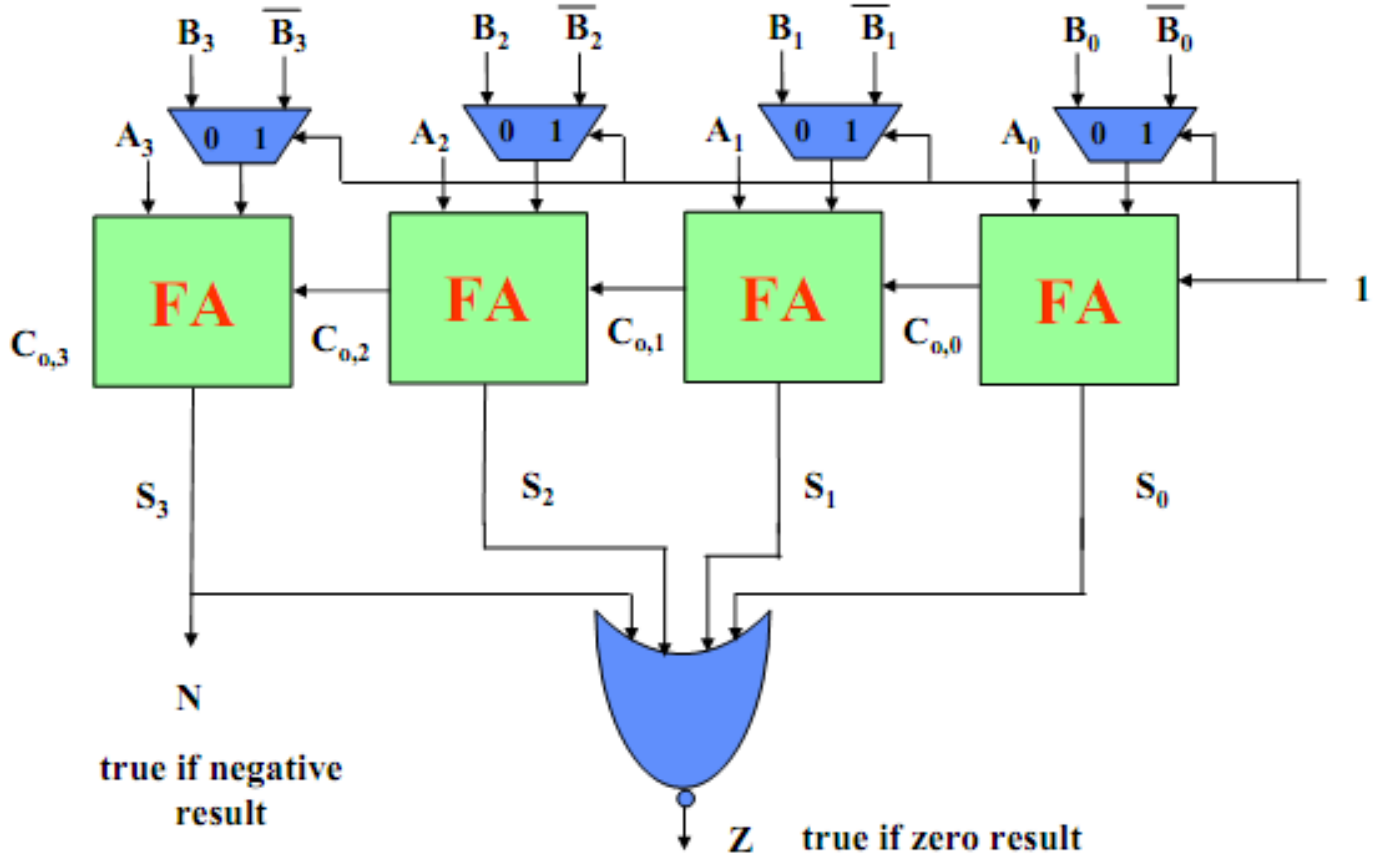
Extension to Subtraction

- Under twos complement, subtracting B is the same as adding the bitwise complement of B then adding 1

Combination addition/subtraction system:



Comparator (one approach)



$A < B = N$

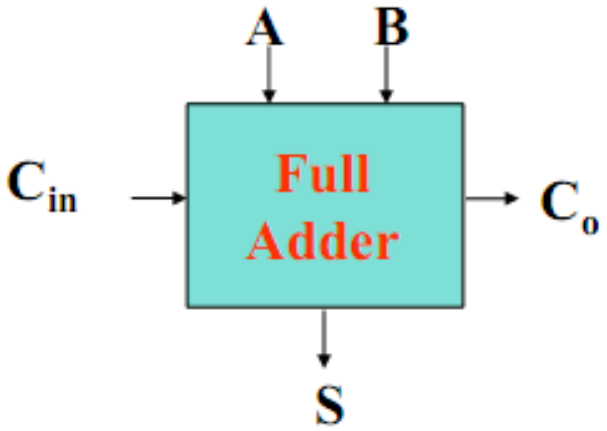
$A = B = Z$

$A \leq B = Z + N$

Alternate Adder Logic Formulation

How to Speed up the Critical (Carry) Path? (How to Build a Fast Adder?)

A	B	C_i	S	C_o	*Carry status*
0	0	0	0	0	delete
0	0	1	1	0	delete
0	1	0	1	0	propagate
0	1	1	0	1	propagate
1	0	0	1	0	propagate
1	0	1	0	1	propagate
1	1	0	0	1	generate
1	1	1	1	1	generate



Generate (G) = AB

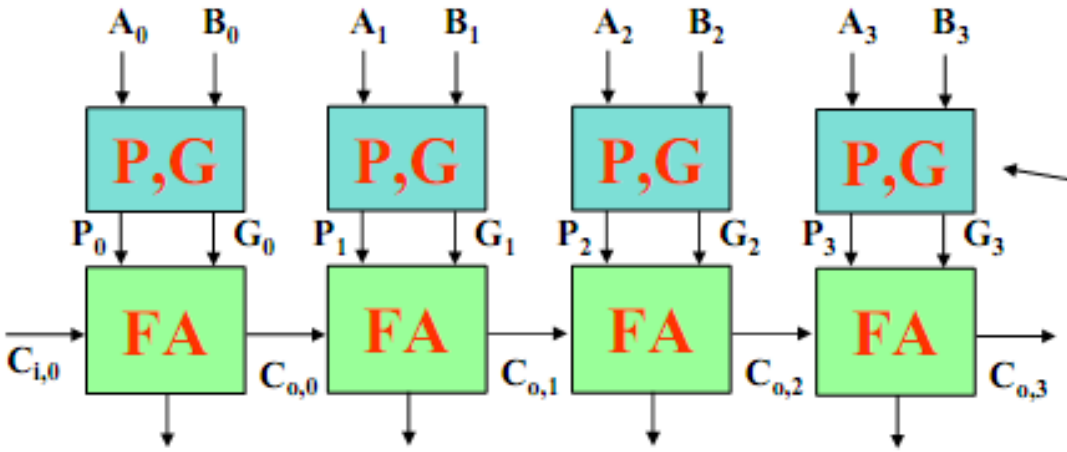
Propagate (P) = A ⊕ B

$$C_o(G,P) = G + PC_i$$

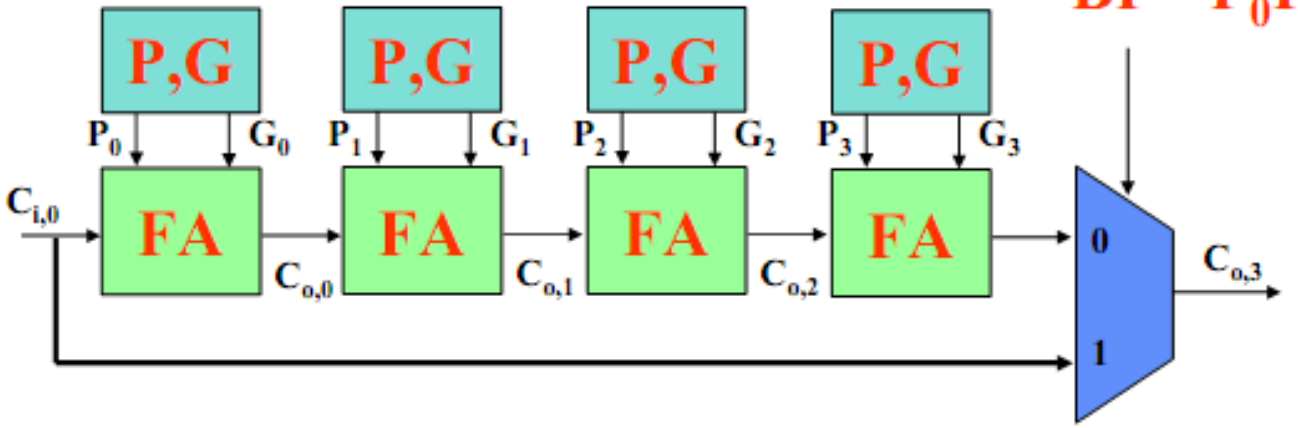
$$S(G,P) = P \oplus C_i$$

Note: can also use P = A + B for C_o

Carry Bypass Adder

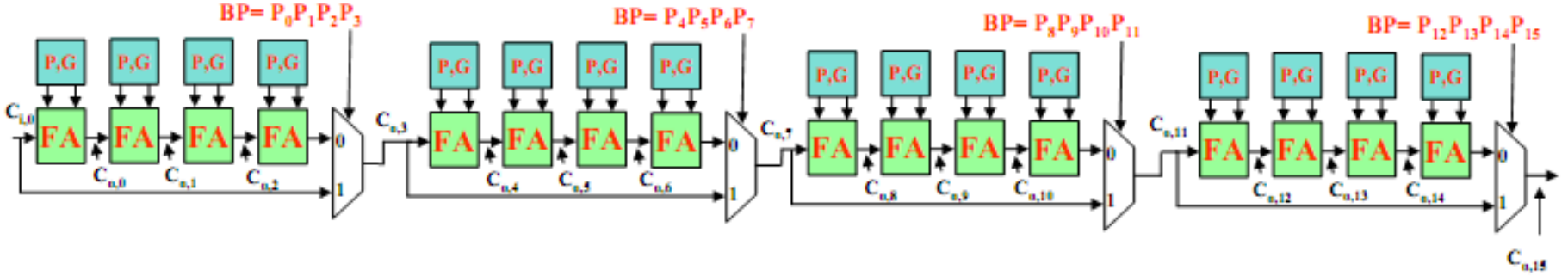


Can compute P, G in parallel for all bits



Key Idea: if $(P_0\ P_1\ P_2\ P_3)$ then $C_{o,3} = C_{i,0}$

16-bit Carry Bypass Adder



Assume the following for delay each gate:

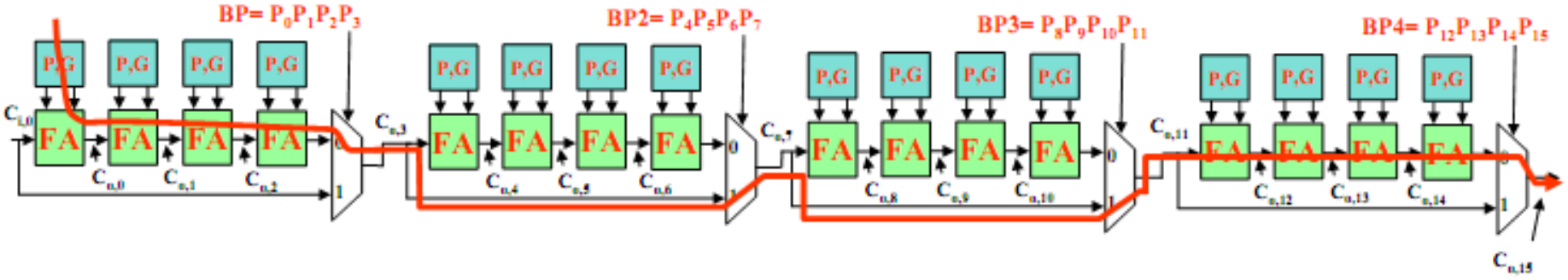
P, G from A, B: 1 delay unit

P, G, C_i to C_o or Sum for a FA: 1 delay unit

2:1 mux delay: 1 delay unit

What is the worst case propagation delay for the 16-bit adder?

Critical Path Analysis



For the second stage, is the critical path:

BP2 = 0 or BP2 = 1?

Message: Timing Analysis is Very Tricky – Must Carefully Consider Data Dependencies For *False Paths*

Carry Lookahead Adder

Re-express the carry logic as follows:

$C1 = G0 + P0\ C0$

$C2 = G1 + P1\ C1 = G1 + P1\ G0 + P1\ P0\ C0$

$C3 = G2 + P2\ C2 = G2 + P2\ G1 + P2\ P1\ G0 + P2\ P1\ P0\ C0$

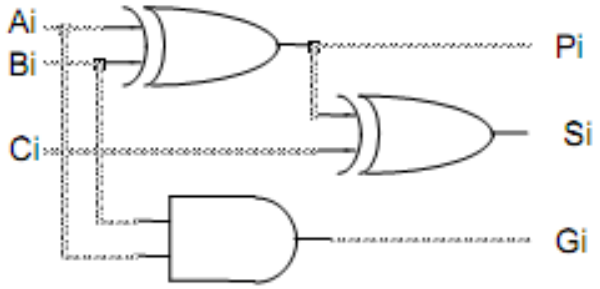
$C4 = G3 + P3\ C3 = G3 + P3\ G2 + P3\ P2\ G1 + P3\ P2\ P1\ G0 + P3\ P2\ P1\ P0\ C0$

…

- Each of the carry equations can be implemented in a two-level logic network
- Variables are the adder inputs and carry in to stage 0

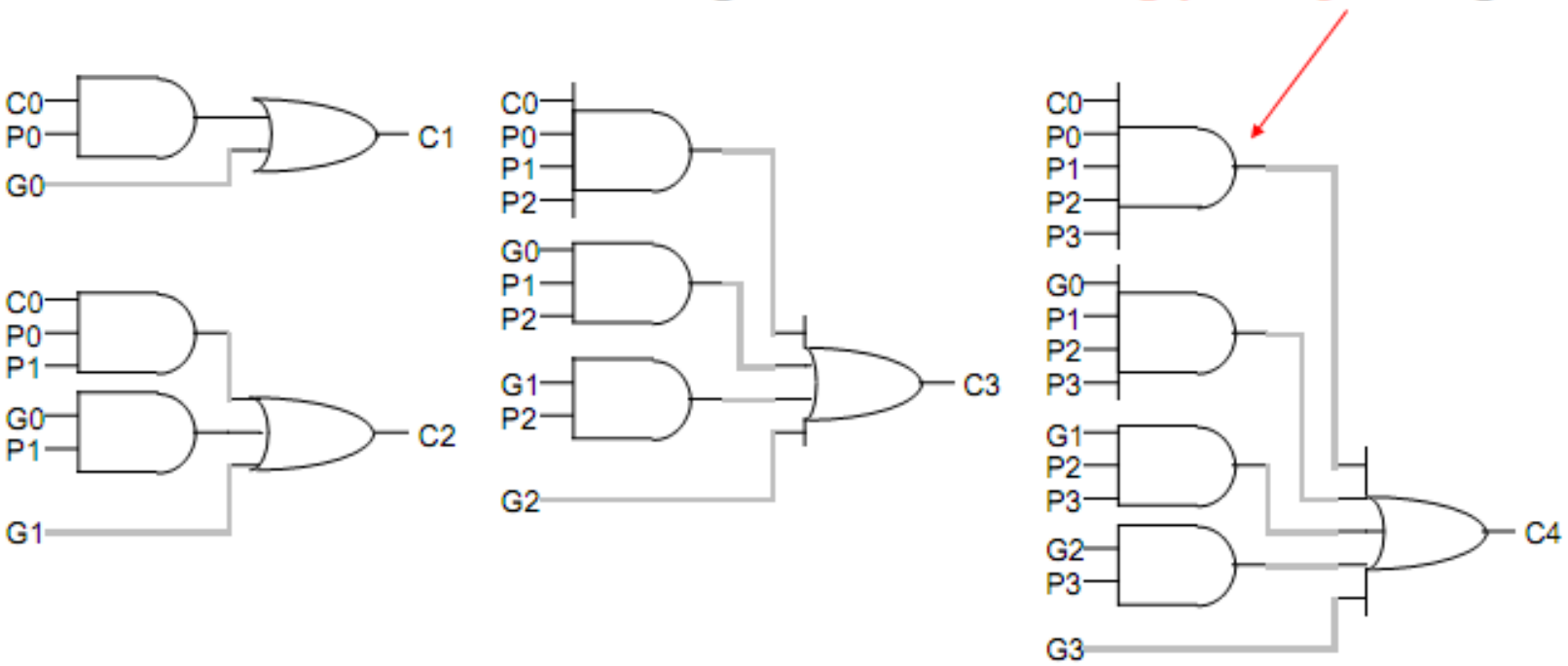
Ripple effect has been eliminated!

Carry Lookahead Logic

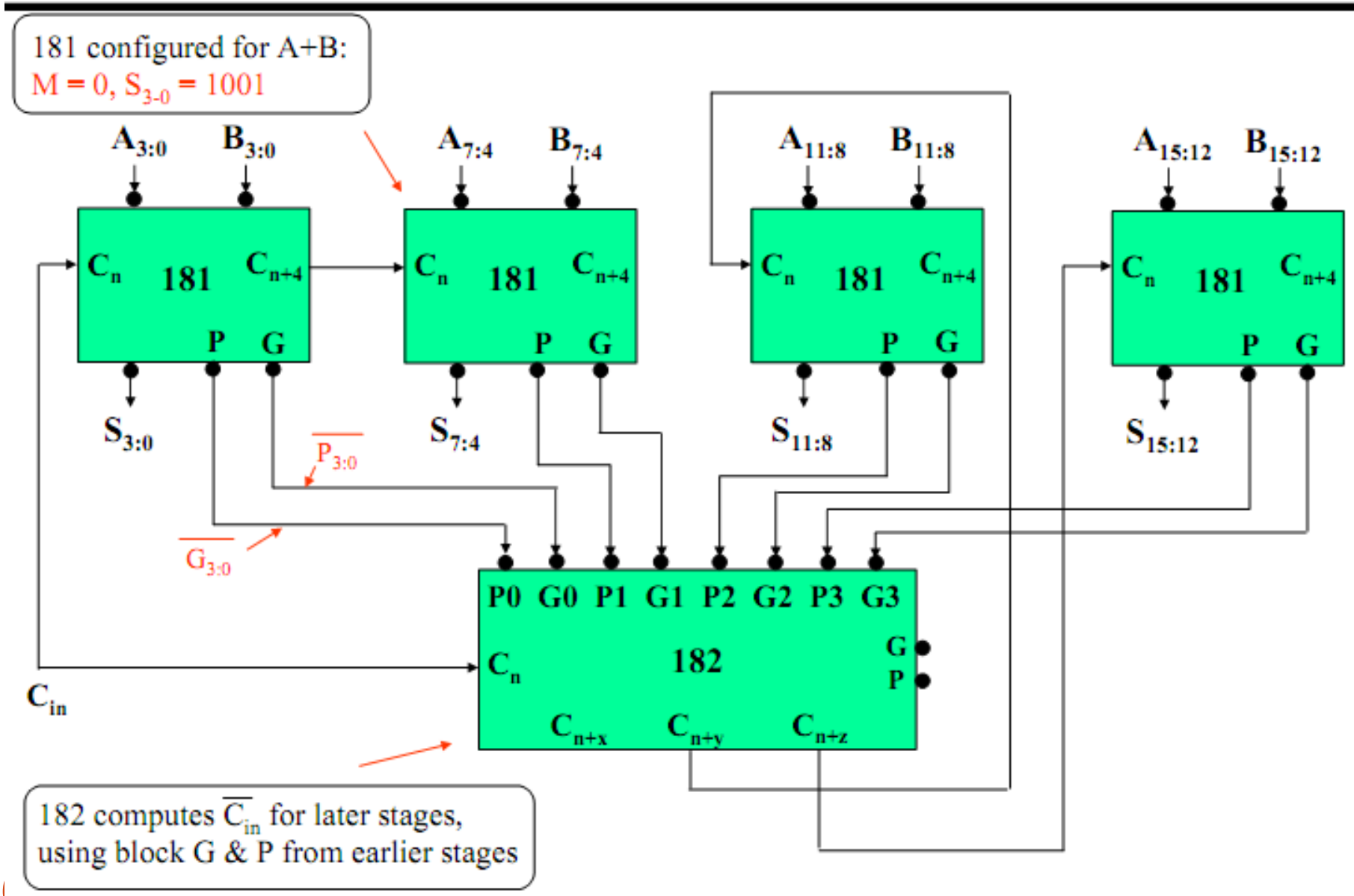


Adder with propagate and generate outputs

Later stages have increasingly complex logic



16-bit Carry Lookahead Schematic



作业：4.18, 4.23

4.18 用一个 3 输入 8 输出的变量译码器及两个 4 输入“与非”门组成一位全加器、全减器电路。

4.23 用双 4 选 1 数据选择器实现全加器。输入量为 A、B；进位输入为 C_{i-1}，输出量为全加和 S 及进位输出 C_i。

附加题：给出由最快的1位全加器构成16位串行加法器、4位并行全加器构成16位串行加法器、16位并行加法器计算结果（C_{16}、F_{16}）所需要的级数，要求描述计算过程。