Chapter 3.1 定点加减法运算

一、原理与溢出检测

1、补码加法. $[x]_c + [y]_c = [x+y]_c \pmod M$.

证明：1° $x>0$, $y>0$, $[x]_c = x$, $[y]_c = y$. 显然成立

2° $x>0$, $y<0$. 设 $M=2$.

此时 $[x]_c = x$, $[y]_c = 2+y$.

$$[x]_c+[y]_c = 2+(x+y) \begin{cases} >0\text{时} & \text{模2，得 } x+y \\ <0\text{时} & [2+x+y]_c = x+y. \end{cases}$$

3° 与2°对称

4° $[x]_c = 2+x$, $[y]_c = 2+y$,

$$[x]_c + [y]_c = 2+2+x+y \xlongequal{\text{模2}} 2+x+y = [x+y]_c. \quad \square$$

例1. 设 $x=-0.1010$, $y=-0.0100$, 求 $[x+y]_c$ 与 $x+y$.

$$\begin{array}{lr} [x]_c = & \cancel{1.00}\ 1.0110 \\ [y]_c = & 1.1100 \\ \hline & 1\ 1.0010 \end{array}$$

模数舍弃

$\Rightarrow [x+y]_c = 1.0010$

$x+y = -0.1110$.

2、补码减法. $\cancel{[x]_c =}$ $[x-y]_c = [x]_c + [-y]_c = [x]_c - [y]_c \pmod M$.

证: $[-y]_c - (-[y]_c) = [-y]_c + [y]_c = [-y+y]_c = 0$.

因而 $[-y]_c = -[y]_c$. $\square$

3、溢出检测.

① 双符号位

00.1011	11.0101
00.0111	11.0011
01.0010	10.1000
overflow	underflow
正+正→负	负+负→正

② 单符号位:

- 使用 正+正 $\xrightarrow{?}$ 负, 负+负 $\xrightarrow{?}$ 正 判定.
- 不可能溢出了"一圈"又回到正数.(取最大值分析).

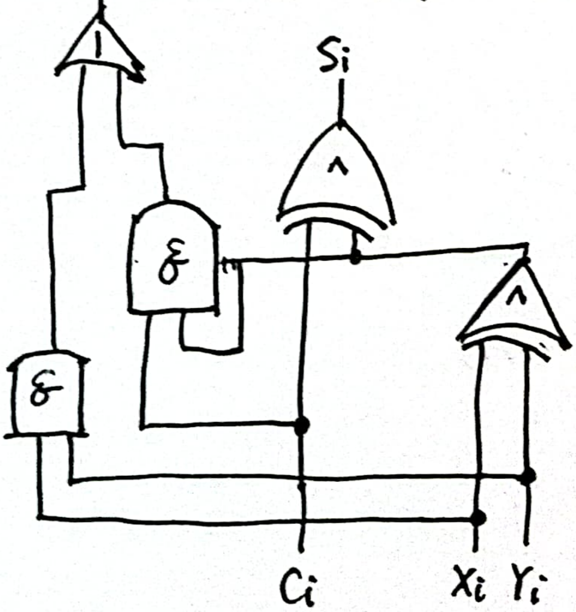
二、加减法的逻辑电路.

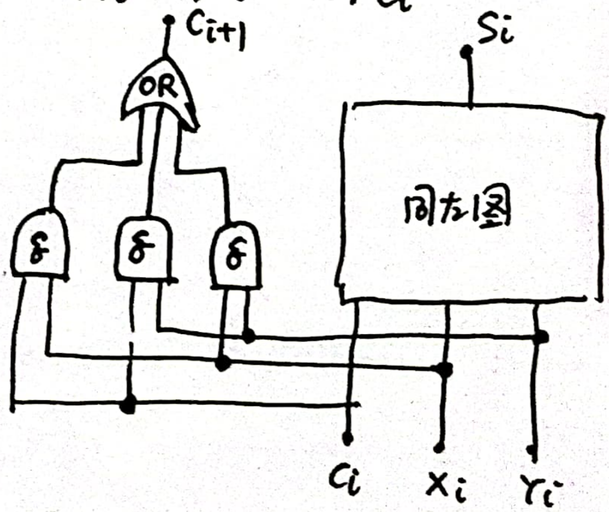
1、全加器

$\underbrace{X_i \quad Y_i}_{\text{adder}} \quad \underbrace{C_i}_{\text{carry on } i} \rightarrow \boxed{FA} \rightarrow \underbrace{S_i}_{\text{sum}}, \underbrace{C_{i+1}}_{\text{carry on } i+1}$

$$S_i = X_i \oplus Y_i \oplus C_i$$

$$C_{i+1} = X_iY_i + (X_iY_i)C_i \quad 或 \quad X_iY_i + (X_i \oplus Y_i)C_i$$



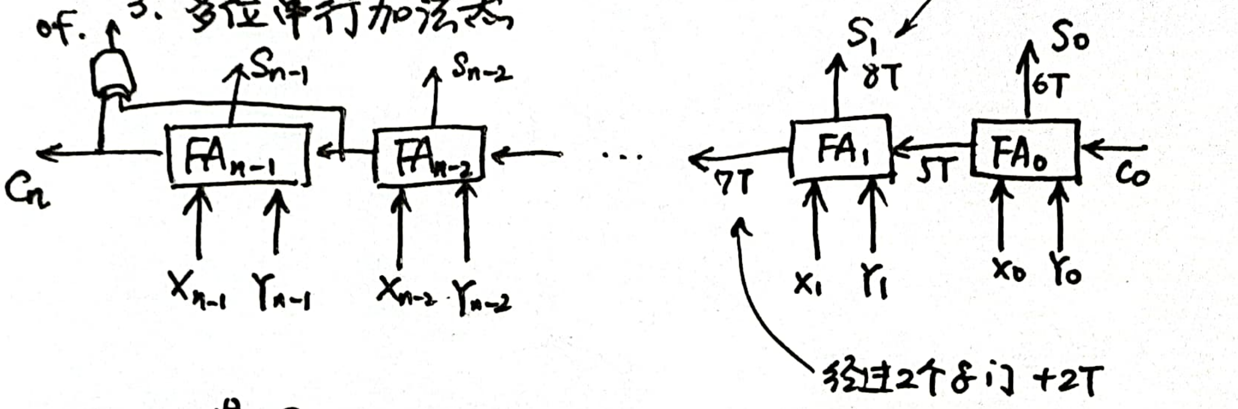


S_i	6T	6T
C_{i+1}	5T	2T

2、半加器　$X_i\ Y_i \rightarrow \boxed{HA} \rightarrow S_i\ C_{i+1}$

$S_i = X_i \oplus Y_i$ ，　$C_{i+1} = X_iY_i$.

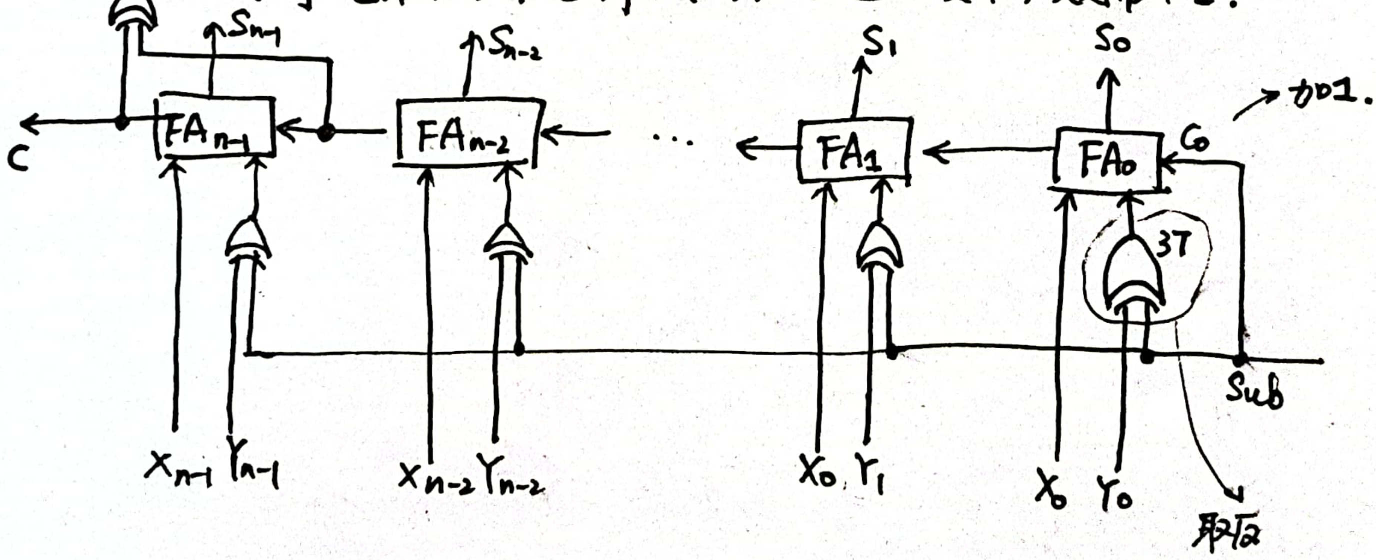
3、多位串行加法器



得 S_{n-1} 延迟为 $(2n+4)T$，$C_n=(2n+3)T$.

3、可控加减法电路

注意：在 2的补码中，$-x = \sim x+1$，故可以直接拿它取反，末尾将减法信号送入其中，从而+1.



4、先行进位加法器.

动机：可否使加法进位更快？

- 若提前获得进位函数，则可以先行计算.

记得. $S_i = X_i \oplus Y_i \oplus C_i$ ，$C_{i+1} = X_iY_i + (X_i \oplus Y_i)C_i$

当 $X_iY_i = 1$ 时，C_{i+1} 一定为1.

记为 $G_i := X_iY_i$ 进位生成函数

C_i 输入，X_i, Y_i 互异，才传递进1.

$P_i := X_i \oplus Y_i$

依 $S_i = P_i \oplus C_i$, $C_{i+1} = G_i + P_i C_i$

$C_1 = G_0 + P_0 C_0$

$C_2 = G_1 + P_1 C_1 = G_1 + P_1 (G_0 + P_0 C_0) = G_1 + P_1 G_0 + P_1 P_0 C_0$

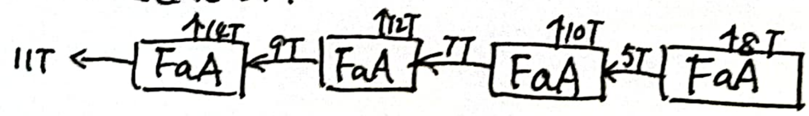
$C_3 = \cdots = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 G_0$

这表明

$$C_n = G_{n-1} + P_{n-1} G_{n-2} + P_{n-1} P_{n-2} G_{n-3} + \cdots + P_{n-1} P_{n-2} \cdots P_1 P_0 C_0$$

(见图1)(书上图3.5),是个快速加法器.(FastA).

• FastAdder 之连接方法



(1) 组内并行,组间串行

(2) 组内、组间并行.

定义 $G^* = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0$

$P^* = P_3 P_2 P_1 P_0$,

$C_4 = G^* + P^* C_0$ 与 $C_1 = G_0 + P_0 C_0$ 形式相同,亦可如上述进行并行.

(见图2)(书上图3.8)是如上描述的.

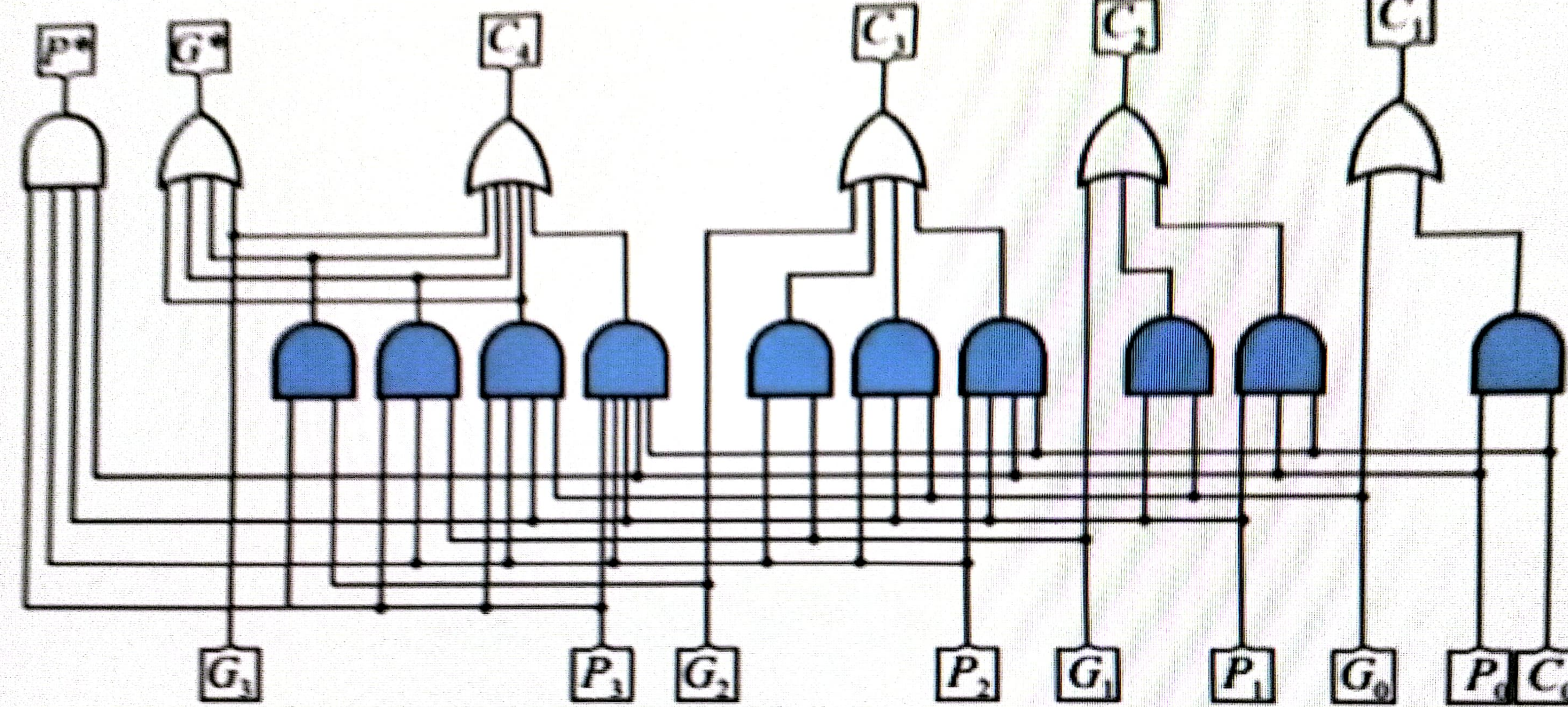


图 3.5　可级联的 4 位先行进位电路

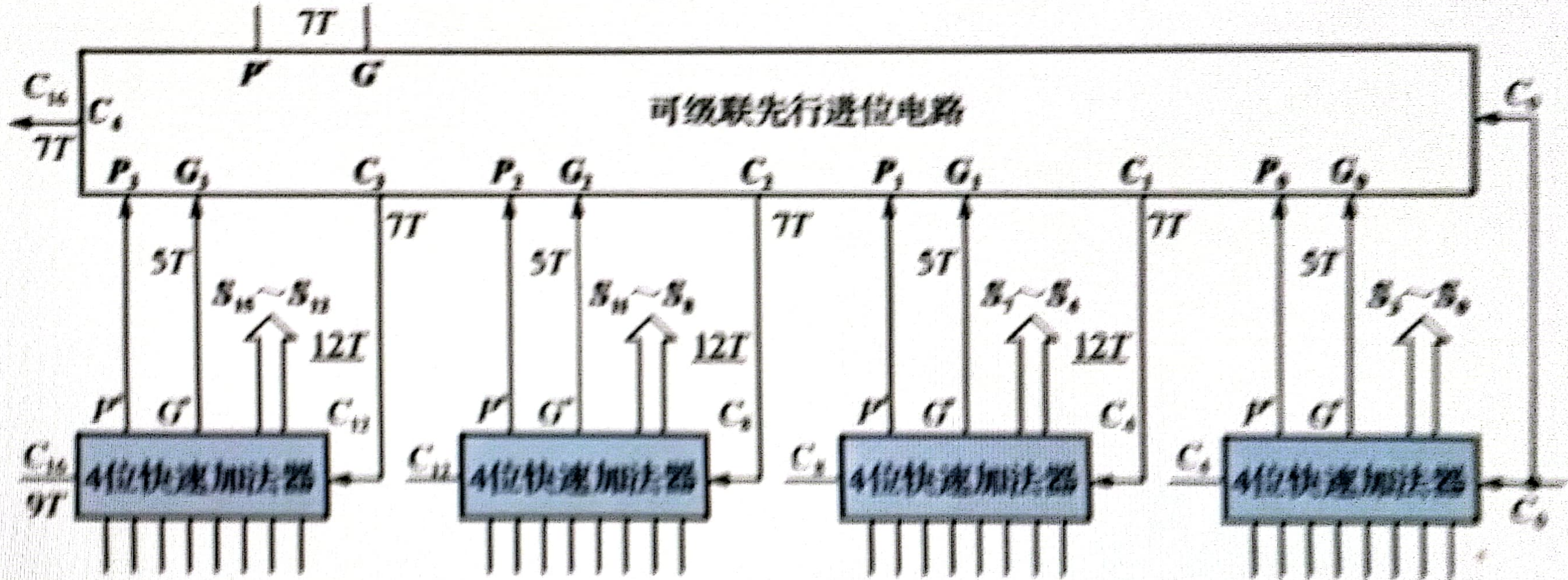


图 3.8　16 位组内并行、组间并行加法器

注意图中所有 4 位快速加法器产生成组生成的进位函数 G^*、P^* 的时间延迟为 $5T$（参考