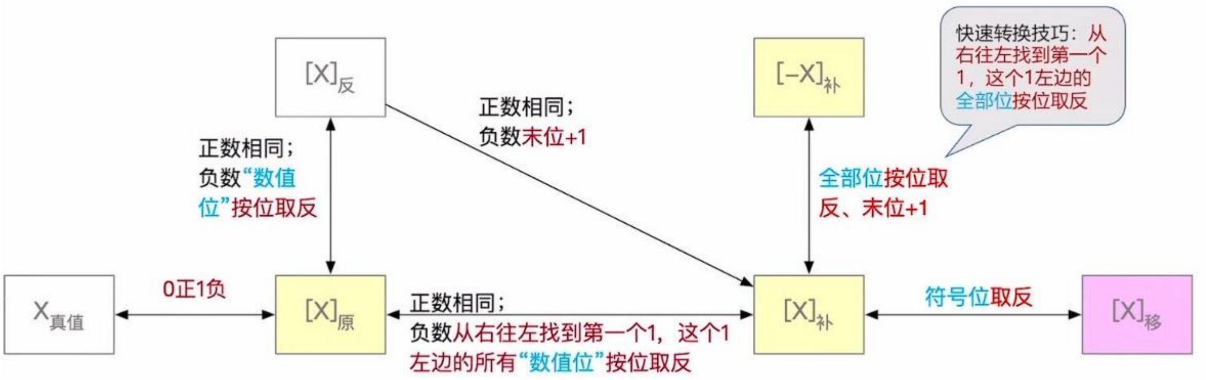
第二章：运算方法

原码、反码、补码、移码转换



原反补转换不改符号位

对于正数，反码和补码是其本身

行数	机器数	真值(十进制)				
		无符号数	原码	反码	补码	移码
1	0000 0000	0	+0	+0	+0,-0	-128
2	0000 0001	1	+1	+1	+1	-127
3	0000 0010	2	+2	+2	+2	-126
...	...	...	...	...	...	...
126	0111 1101	125	+125	+125	+125	-3
127	0111 1110	126	+126	+126	+126	-2
128	0111 1111	127	+127	+127	+127	-1
129	1000 0000	128	-0	-127	-128	0
130	1000 0001	129	-1	-126	-127	1
131	1000 0010	130	-2	-125	-126	2
...	...	...	...	...	...	...
253	1111 1100	252	-124	-3	-4	124
254	1111 1101	253	-125	-2	-3	125
255	1111 1110	254	-126	-1	-2	126
256	1111 1111	255	-127	-0	-1	127

溢出判断

1、双符号位法

（变形补码—扩大一倍范围）

$[x]_{补} = 2^{n+2} + x \quad (\mathrm{mod}\ 2^{n+2})$

S_{f1}	S_{f2}	
0	0	正确（正数）
0	1	正溢
1	0	负溢
1	1	正确（负数）

S_{f1} 表示正确的符号

[例] x=+1100，y=+1000，求 x+y。

解：$[x]_{补}$ = 001100，$[y]_{补}$ = 001000

	$[x]_{补}$	0 0 1 1 0 0
+	$[y]_{补}$	0 0 1 0 0 0
	$[x+y]_{补}$	0 1 0 1 0 0 (表示正溢)

符号扩展

Eg：8位→16位

正整数（原、反、补码的表示都一样）

0,1011010 → 0,00000000 1011010

正小数（原、反、补码的表示都一样）

0.1011010 → 0.1011010 00000000

负整数：

原码：	1,1011010	→ 1,00000000 1011010
反码：	1,0100101	→ 1,11111111 0100101
补码：	1,0100110	→ 1,11111111 0100110

负小数：

原码：	1.1011010	→ 1.1011010 00000000
反码：	1.0100101	→ 1.0100101 11111111
补码：	1.0100110	→ 1.0100110 00000000

▼ 定点数运算

- 加减法带符号运算，乘除符号单独运算

定点加减法

1. **变形补码法**（双符号位）
    - 符号参与运算

变形补码算 x+y

(1) $x=11011$ $y=00001$ 双符号位

$[x]_补=0011011$ $[y]_补=0000011$

$$\begin{array}{r} 0011011 \\ +\ 0000011 \\ \hline 0011110 \end{array}$$ 无溢

$[x]_补+[y]_补=0011110$

$x+y=11110$

变形补码算 x-y

(1) $x=11011$ $y=-11111$ 符号参与

$[x]_补=0011011$ $[y]_补=1100001$

$[-y]_补=0011111$

$$\begin{array}{r} 0011011 \\ +\ 0011111 \\ \hline 0111010 \end{array}$$ 正溢

$x-y=11010$

溢出

【例 2.18】 $x=+1100$，$y=+1000$，求 $x+y$。

解 $[x]_补=001100$，$[y]_补=001000$

$$\begin{array}{lll} & [x]_补 & 001100 \\ + & [y]_补 & 001000 \\ \hline & & 010100 \end{array}$$

两个符号位出现“01”，表示正溢出，即结果大于$+2^n$。

【例 2.19】 $x=-1100$，$y=-1000$，求 $x+y$。

解 $[x]_补=110100$，$[y]_补=111000$

$$\begin{array}{lll} & [x]_补 & 110100 \\ + & [y]_补 & 111000 \\ \hline & & 101100 \end{array}$$

两个符号位出现“10”，表示负溢出，即结果小于-2^n。

定点乘法

原码阵列乘法器、补码阵列乘法器（算前算后多一步求补）

(1) $x=11011$ $y=-11111$ $[x]_补=011011$ $[y]_补=100001$

原码阵列乘法器

$|x|=11011$ $|y|=11111$

$$\begin{array}{r} 11011 \\ \times\ 11111 \\ \hline 11011 \\ 11011 \\ 11011 \\ 11011 \\ 11011 \\ \hline 1101000101 \end{array}$$

符 $0\oplus1=1$

$[x\times y]_原=1\ 1101000101$

$x\cdot y=(-1101000101)_2=(-837)_{10}$

补码阵列乘法器

$|x|=11011$ $|y|=11111$

$$\begin{array}{r} 11011 \\ \times\ 11111 \\ \hline 11011 \\ 11011 \\ 11011 \\ 11011 \\ 11011 \\ \hline 1101000101 \end{array}$$

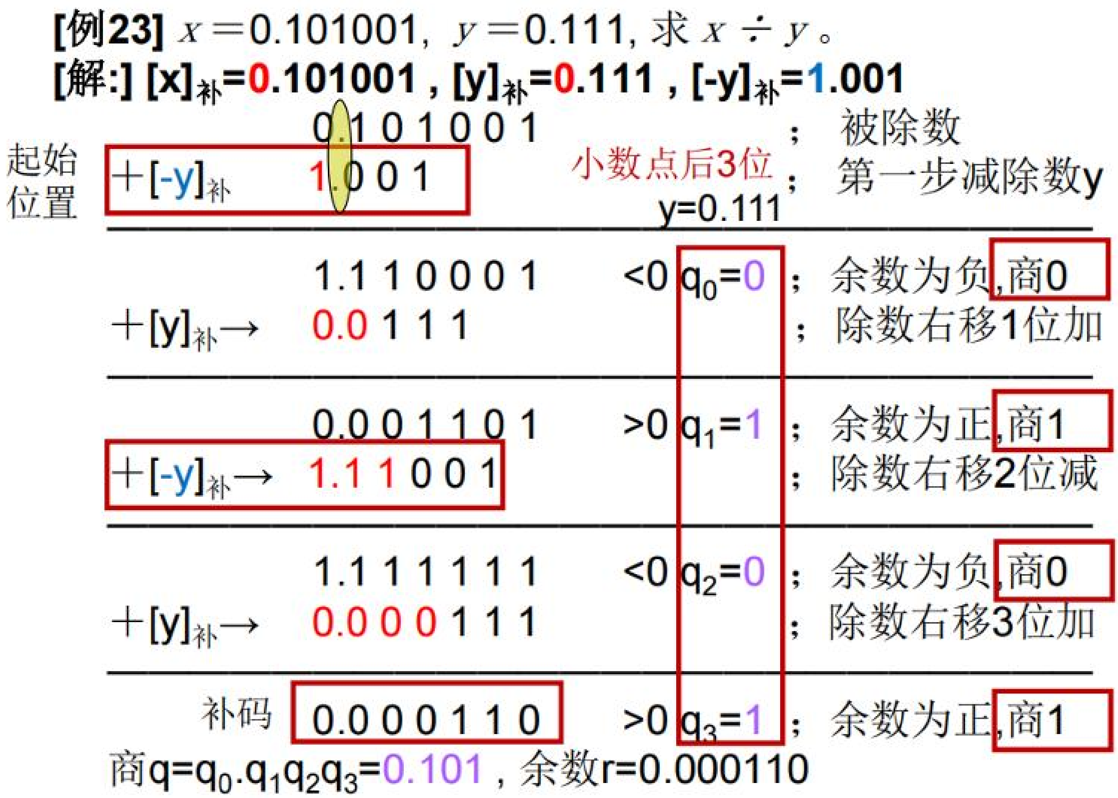
符 $0\oplus1=1$

$[x\times y]_补=1\ 0010111011$

定点除法

加减交替法（不恢复余数法）

1. 先乘比例因子变成小数
2. 符号单独运算
3. 除数为 $q_0.q_1q_2...q_n$ ，商算 $n+1$位（算出 $q_0.q_1q_2...q_n$）
4. 余数为正，商1，下次除数右移做减法，反之商0，下次除数右移做加法
5. 余数若为负，找到最后一次正余数作为余数
6. 补偿余数，余数成比例因子的倒数
7. 商的计算结果是不带符号的，再添上符号



习题[2-8-1)] x＝11000，y＝-11111，用原码除法，求 $x \div y$

- 符号位： 0⊕1=1
- |x|=11000，|y|=11111 小数点后5位
- 纯小数表示，小数点左移5位，|x|=0.11000，|y|=0.11111
- [| x |]=0.11000, [| y |]补=0.11111 , [-| y |]补=1.00001

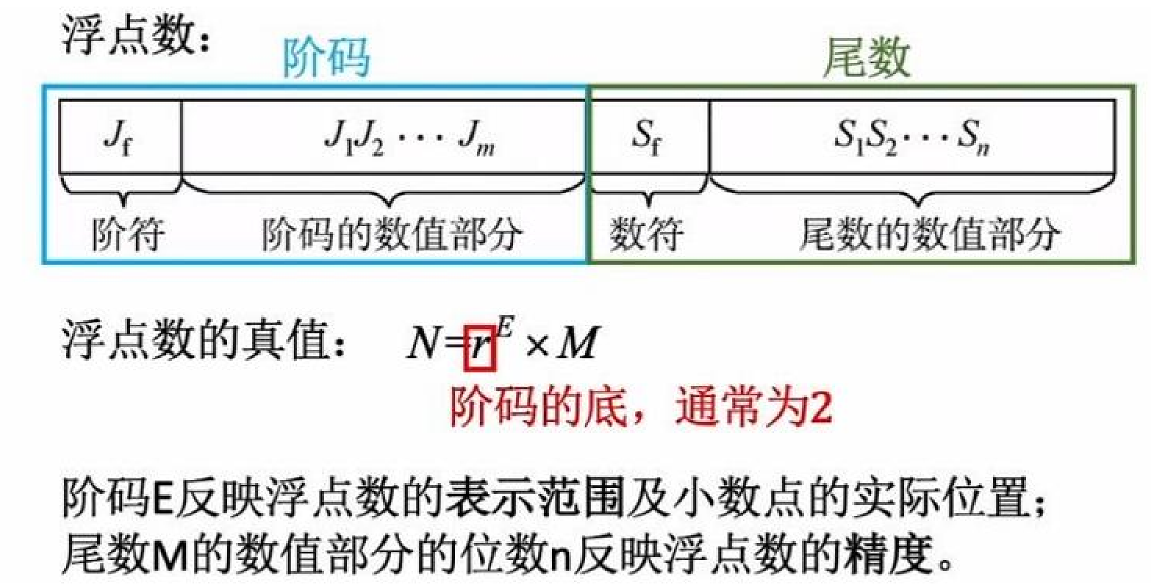
	0.1 1 0 0 0		； 被除数
+[-y]补	1. 0 0 0 0 1		； 第一步减除数y
	1.1 1 0 0 1	<0 q_0=0	； 余数为负,商0
+[y]补→	0.0 1 1 1 1 1		； 除数右移1位加
	0.0 1 0 0 0 1	>0 q_1=1	； 余数为正,商1
+[-y]补→	1.1 1 0 0 0 0 1		； 除数右移2位减
	0.0 0 0 0 0 0 1 1	>0 q_2=1	； 余数为正, 商1
+[-y]补→	1.1 1 1 0 0 0 0 1		； 除数右移3位减
	1.1 1 1 0 0 1 1 1	<0 q3=0	； 余数为负, 商0
+[y]补→	0.0 0 0 0 1 1 1 1 1		； 除数右移4位加
	1.1 1 1 1 0 1 1 0 1	<0 q4=0	； 余数为负, 商0
+[y]补→	0.0 0 0 0 0 1 1 1 1 1		； 除数右移5位加
	1. 1 1 1 1 1 1 1 0 0 1	<0 q5=0	； 余数为负， 商0

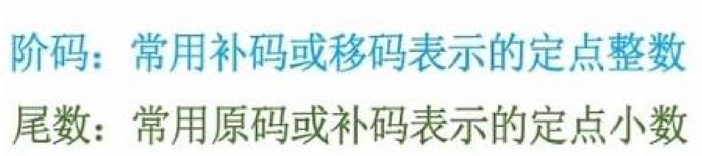
- 商真值|$x \div y$|=0.11000，原码除法 [$x \div y$]原=1.11000
- 余数：0.0000011
- 小数点右移5位（补偿）：0.11

	0.0 1 0 0 0 1	>0 q_1=1	； 余数为正,商1
+[-y]补→	1. 1 1 0 0 0 0 1		； 除数右移2位减
	0.0 0 0 0 0 1 1	>0 q_2=1	； 余数为正, 商1

0.0 0 0 0 0 0 1 1 >0 q_2=1 ； 余数为正, 商1

▼ 浮点数运算





▼ 规格化

用原码

正数 0.1xxxx

负数 1.1xxxx

用补码

正数 0.1xxxx

负数 1.0xxxx

左规：尾数算数左移（小数点右移）

右规：尾数算数右移（小数点左移）

1. 用原码表示的尾数进行规格化：
正数为0.1××…×的形式，其最大值表示为0.11…1；最小值表示为0.10…0。
尾数的表示范围为$1/2 \le M \le (1-2^{-n})$。
负数为1.1××…×的形式，其最大值表示为1.10…0；最小值表示为1.11…1。
尾数的表示范围为$-(1-2^{-n}) \le M \le -1/2$。

2. 用补码表示的尾数进行规格化：
正数为0.1××…×的形式，其最大值表示为0.11…1；最小值表示为0.10…0。
尾数的表示范围为$1/2 \le M \le (1-2^{-n})$。
负数为1.0××…×的形式，其最大值表示为1.01…1；最小值表示为1.00…0。
尾数的表示范围为$-1 \le M \le -(1/2+2^{-n})$。

但对S<0时，有两种情况需特殊处理。

S=-1/2，则$[S]_{补}$=11.100…0。对于补码而言，它不满足于上面的规格化表示式。为了便于硬件判断，特规定-1/2是规格化的数(对补码而言)。

S=-1，则$[S]_{补}$=11.000…0。因小数补码允许表示-1，故-1视为规格化的数。

加减法运算

原码计算

小数都转化为 1. 的形式。

[例] 设$x=0.5_{10}$，$y=-0.4375_{10}$，尾数用原码，求$(x+y)_{浮}$

1、写出操作数（浮点数格式）

$x=0.1_2=0.1_2\times2^0=1.000\times2^{-1}$

$y=-0.0111_2=-0.0111_2\times2^0=-1.110\times2^{-2}$

2、对阶：y阶码更小，调整y使对齐

$y=-1.110\times2^{-2}=-0.111$（尾数右移1位）$\times2^{-1}$

3、尾数相加

$x+y=1.000-0.111=0.001$ （$\times2^{-1}$）

4、规格化

$x+y=0.001\times2^{-1}=1.000$（小数点右移3位）$\times2^{-4}$

5、舍入操作

尾数有效位为4位，不进行舍入操作

补码计算

- 对阶：阶码小向大对。$E_x-E_y>0$，E_y向 E_x 对阶。

  阶码+1，尾数小数点左移1。

超出精度的加括号

- 带符号位尾数相加
- 规格化
- 舍入
  - 就近舍入(0舍1入)：类似”四舍五入”
  - 朝0舍入：截尾
  - 朝＋∞舍入：正数多余位不全为0，进1；负数，截尾
  - 朝－∞ 舍入：负数多余位不全为0，进1；正数，截尾
- 溢出判断

阶3位，尾6位，算[x+y]，[x-y]

(1) $x = 2^{-011} \times 0.100101$

$y = 2^{-010} \times (-0.011110)$

$E_x = 11\ 101$ 双符号

$E_y = 11\ 110$

$E_x - E_y = [11101] + [00010] = 11111 < 0$

对阶 $x = 2^{-011} \times 0.100101 = 2^{-010} \times 0.010010(1)$

$$
\begin{array}{lr}
x+y & 00.010010(1) \\
 & +\ 11.100010 \\
\hline
 & 11.110100(1)
\end{array}
$$

规格化： 1.010010×2^{-100}

$\therefore\ x+y = 2^{-4} \times (-0.101110)$

$$
\begin{array}{lr}
x-y & 00.010010(1) \\
 & +\ 00.011110 \\
\hline
 & 00.110000(1)
\end{array}
$$

规格化：0.110001×2^{-010}
(并舍入)

$x-y = 2^{-2} \times 0.110001$

▼ 乘除法运算

- 符号位单独处理

原码计算

[例] 设$x=0.5_{10}$，$y=-0.4375_{10}$，尾数用原码，求$(x\times y)_{浮}$

1、写出操作数（浮点数格式）

$x=0.1_2=0.1_2*2^0=1.000*2^{-1}$

$y=-0.0111_2=-0.0111_2*2^0=-1.110*2^{-2}$

2、阶码相加

$-1+-2=-3$

3、尾数相乘

$x\times y=1.110\quad(\times 2^{-3})$

4、规格化与确定符号

$x\times y=-1.110\times 2^{-3}$

补码计算

浮点数(乘除) 阶3 尾6

(1) $(2^3\times\frac{13}{16})\times[2^4\times(-\frac{9}{16})]$

$[E_x]_{补}=0,011$ $[E_y]_{原}=0,100$ $E_z=E_x+E_y=0.111$

$[M_x]_{原}=0.110100$ $[M_y]_{原}=1.100100$

$|M_x|\times|M_y|$

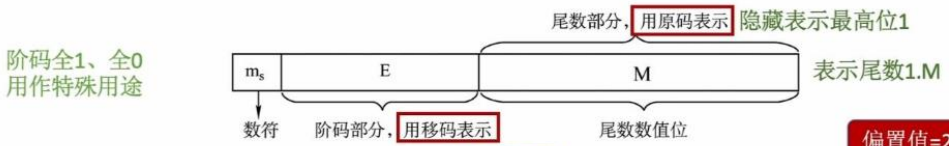
$$
\begin{array}{r}
0.1101 \\
\times \quad 0.1001 \\
\hline
01101 \\
00000 \\
00000 \\
01101 \\
00000 \\
\hline
0.01110101
\end{array}
$$

符号 $1\oplus0=1$

规格化: 1.111011×2^6

$x\cdot y=(-0.111011)\times2^6$

▼ IEEE754



偏置值 $=2^{n-1}-1$

真值正常范围：-126~127

单精度浮点型　双精度浮点型

	类型	数符	阶码	尾数数值	总位数	偏置值 十六进制	偏置值 十进制
float	短浮点数	1	8	23	32	7FH	127
double	长浮点数	1	11	52	64	3FFH	1023
long double	临时浮点数	1	15	64	80	3FFFH	16383

float 1000 0001 1000 1010 0101 0000 1000 0000

double 1000 0001 1100 0010 0101 0000 1000 0000 0000 0000 0001 1111 0000 0000 0000 0000

规格化的短浮点数的真值为：$(-1)^s \times 1.M \times 2^{E-127}$　　阶码真值=移码-偏移量

规格化长浮点数的真值为：$(-1)^s \times 1.M \times 2^{E-1023}$

格式	规格化的最小绝对值	规格化的最大绝对值
单精度	E=1，M=0：$1.0 \times 2^{1-127}=2^{-126}$	E=254，M=.11...1：$1.11...1 \times 2^{254-127}=2^{127} \times (2-2^{-23})$
双精度	E=1，M=0：$1.0 \times 2^{1-1023}=2^{-1022}$	E=2046，M=.11...1：$1.11...1 \times 2^{2046-1023}=2^{1023} \times (2-2^{-52})$

IEEE 754 单精度浮点型能表示的最小绝对值、最大绝对值是多少？

最小绝对值：尾数全为0，阶码真值最小-126，对应移码机器数 0000 0001
此时整体的真值为 $(1.0)_2 \times 2^{-126}$

只有 $1 \le E \le 254$ 时，真值 $=(-1)^s \times 1.M \times 2^{E-127}$

隐含最高位变为0　　阶码真值固定视为 -126

当阶码E全为0，尾数M不全为0时，表示非规格化小数 $\pm(0.xx...x)_2 \times 2^{-126}$

当阶码E全为0，尾数M全为0时，表示真值 ±0

当阶码E全为1，尾数M全为0时，表示无穷大 ±∞

当阶码E全为1，尾数M不全为0时，表示非数值 “NaN” (Not a Number)

特殊情况

32位IEEE标准表示的范围

- 最大正数：0 11111110 111 1111 1111 1111 1111 1111=$2^{127}*(1+(1-2^{-23}))$
- 最小正数：0 00000001 000 0000 0000 0000 0000 0000=$2^{-126}*1$
- 最小负数：1 11111110 111 1111 1111 1111 1111 1111=$-2^{127}*(1+(1-2^{-23}))$
- 最大负数：1 00000001 000 0000 0000 0000 0000 0000=$-2^{-126}*1$

二进制转十进制

例：若浮点数x的754标准存储格式为$(41360000)_{16}$，求其浮点数的十进制数值。

解：将16进制数展开后，可得二制数格式

0 100 00010 011 0110 0000 0000 0000 0000

S 阶码(8位) 尾数(23位)

130

指数e=阶码-127=10000010-01111111=00000011=$(3)_{10}$

包括隐藏位1的尾数 使用十进制数算 130-127=3 =$(00000011)_2$

1.M=1.011 0110 0000 0000 0000 0000=1.011011

于是有：$x=(-1)S\times1.M\times2^{e}=+(1.011011)\times2^{3}=+1011.011=(11.375)_{10}$

十进制转二进制

例2 将数$(20.59375)_{10}$转换成754标准的32位浮点数的二进制存储格式。

解:首先分别将整数和分数部分转换成二进制数:

20.59375=10100.10011

然后移动小数点，使其在第1，2位之间

$10100.10011=1.\underline{010010011}\times2^{4}$

e=4于是得到: M

S=0, E=4+127=131, E=1000 0011

最后得到32位浮点数的二进制存储格式为:

0 10000011 010010011 00000000000000=$(41A4C000)_{16}$

▼ 加法器

【例 2.20】 参见图 2.5，已知两个不带符号的二进制整数 A=11011，B=10101，求每一部分乘积项 a_ib_j 的值与 $p_9p_8\cdots p_0$ 的值。

解

$$
\begin{array}{rr}
 & 1\,1\,0\,1\,1 = A\,(27_{10}) \\
\times & 1\,0\,1\,0\,1 = B\,(21_{10}) \\
\hline
 & 1\,1\,0\,1\,1 \\
 & 0\,0\,0\,0\,0 \\
 & 1\,1\,0\,1\,1 \\
 & 0\,0\,0\,0\,0 \\
+ & 1\,1\,0\,1\,1 \\
\hline
 & 1\,0\,0\,0\,1\,1\,0\,1\,1\,1 = P
\end{array}
$$

$a_4b_0=1,a_3b_0=1,a_2b_0=0,a_1b_0=1,a_0b_0=1$
$a_4b_1=0,a_3b_1=0,a_2b_1=0,a_1b_1=0,a_0b_1=0$
$a_4b_2=1,a_3b_2=1,a_2b_2=0,a_1b_2=1,a_0b_2=1$
$a_4b_3=0,a_3b_3=0,a_2b_3=0,a_1b_3=0,a_0b_3=0$
$a_4b_4=1,a_3b_4=1,a_2b_4=0,a_1b_4=1,a_0b_4=1$

$$P = p_9p_8p_7p_6p_5p_4p_3p_2p_1p_0 = 1000110111\,(567_{10})$$

四片74181ALU和一片74182CLA配合，具有进位传递功能，在片内进行**先行进位**，在片间进行**先行进位。**

- 加法器中每位的进位生成信号g为（X_iY_i）

$$F_i=X_i \oplus Y_i \oplus C_{n+i}$$

$$C_{n+i+1}=X_iY_i+Y_iC_{n+i}+C_{n+i}X_i$$

$$C_{n+1}=Y_0+X_0C_n$$

$$C_{n+2}=Y_1+X_1C_{n+1}=Y_1+Y_0X_1+X_0X_1C_n$$

$$C_{n+3}=Y_2+X_2C_{n+2}=Y_2+Y_1X_2+Y_0X_1X_2+X_0X_1X_2C_n$$

$$C_{n+4}=Y_3+X_3C_{n+3}=Y_3+Y_2X_3+Y_1X_2X_3+Y_0X_1X_2X_3+X_0X_1X_2X_3C_n$$

$$G=Y_3+Y_2X_3+Y_1X_2X_3+Y_0X_1X_2X_3$$

$$P=X_0X_1X_2X_3$$

G 称为进位发生输出 P 称为进位传送 输出

先行进位发生器(CLA)

1）串行进位方式：

$$C_1 = G_1 + P_1C_0 \qquad 其中，G_1 = A_1B_1，P_1 = A_1 \oplus B_1$$
$$C_2 = G_2 + P_2C_1 \qquad G_2 = A_2B_2，P_2 = A_2 \oplus B_2$$
$$C_3 = G_3 + P_3C_2 \qquad G_3 = A_3B_3，P_3 = A_3 \oplus B_3$$
$$C_4 = G_4 + P_4C_3 \qquad G_4 = A_4B_4，P_4 = A_4 \oplus B_4$$

$C_n = AB + (A \oplus B)C_{n-1}$

2）并行进位方式：

$$C_1 = G_1 + P_1C_0$$
$$C_2 = G_2 + P_2G_1 + P_2P_1C_0$$
$$C_3 = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1C_0$$
$$C_4 = G_4 + P_4G_3 + P_4P_3G_2 + P_4P_3P_2G_1 + P_4P_3P_2P_1C_0$$

其中，$G_1 \sim G_4$、$P_1 \sim P_4$表达式与串行进位方式相同。