

定点乘法运算

乘法 \longrightarrow 部分积累加、移位。

1. 原码一位乘法

每次用一位乘数去乘被乘数。

(1) 算法分析

例. 0.1101×1.1011

$X_{\text{原}}$ $Y_{\text{原}}$

乘积 $P = |X| \times |Y|$

积符 $S_P = S_X \oplus S_Y$

1) 手算

0.1101 例. 0.1101×1.1011

$$\begin{array}{r} \times 0.1011 \\ \hline 1101 \end{array} \text{ —— 部分积}$$

1101

0000

+ 1101

0.10001111

上符号: 1.10001111

- 问题:
- 1) 加数增多 (由乘数位数决定)。
 - 2) 加数的位数增多 (与被乘数、乘数位数有关)。

改进: 将一次相加改为分步累加。

2) 分步乘法

每次将一位乘数所对应的部分积与原部分积的累加和相加，并移位。

设置寄存器：

A: 存放部分积累加和、乘积高位

B: 存放被乘数

C: 存放乘数、乘积低位

设置初值： 例. 0.1101×1.1011

$A = 00.0000$

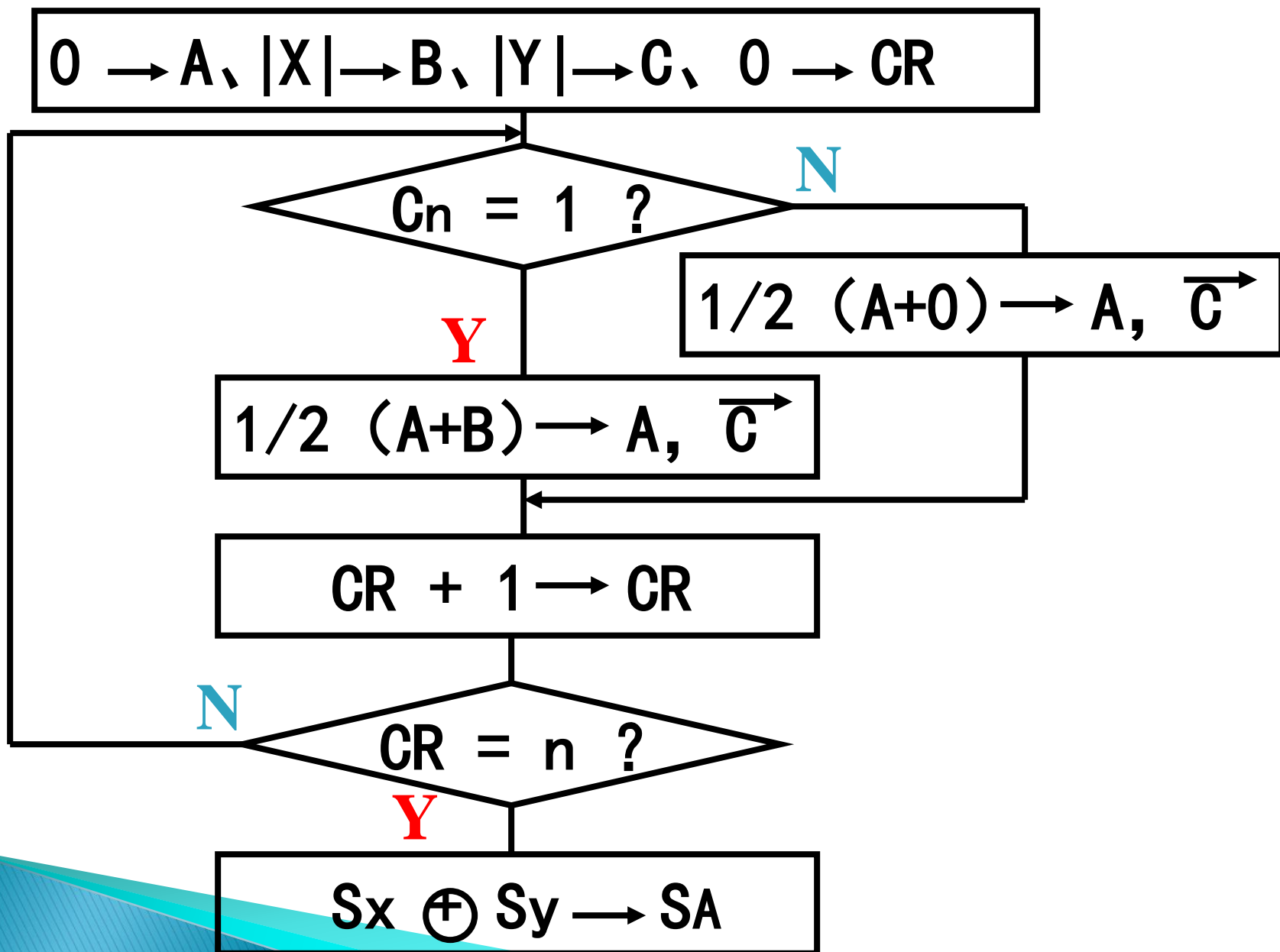
$B = |X| = 00.1101$

$C = |Y| = .1011$

步数	条件	操作	A	C ^{C_n}
			00. 0000	. 101 ¹
1)	C _n =1	+B	$ \begin{array}{r} 00. 0000 \\ + 00. 1101 \\ \hline 00. 1101 \end{array} $	
		→	00. 0110	1. 10 ¹
2)	C _n =1	+B	$ \begin{array}{r} 00. 0110 \\ + 00. 1101 \\ \hline 01. 0011 \end{array} $	
		→	00. 1001	11. 1 ⁰
3)	C _n =0	+0	$ \begin{array}{r} 00. 1001 \\ + 00. 0000 \\ \hline 00. 1001 \end{array} $	
		→	00. 0100	111. ¹
4)	C _n =1	+B	$ \begin{array}{r} 00. 0100 \\ + 00. 1101 \\ \hline 01. 0001 \end{array} $	
		→	00. 1000	1111

0. ~~X_原~~ × ~~Y_原~~ = 1. 10001111

(2) 算法流程



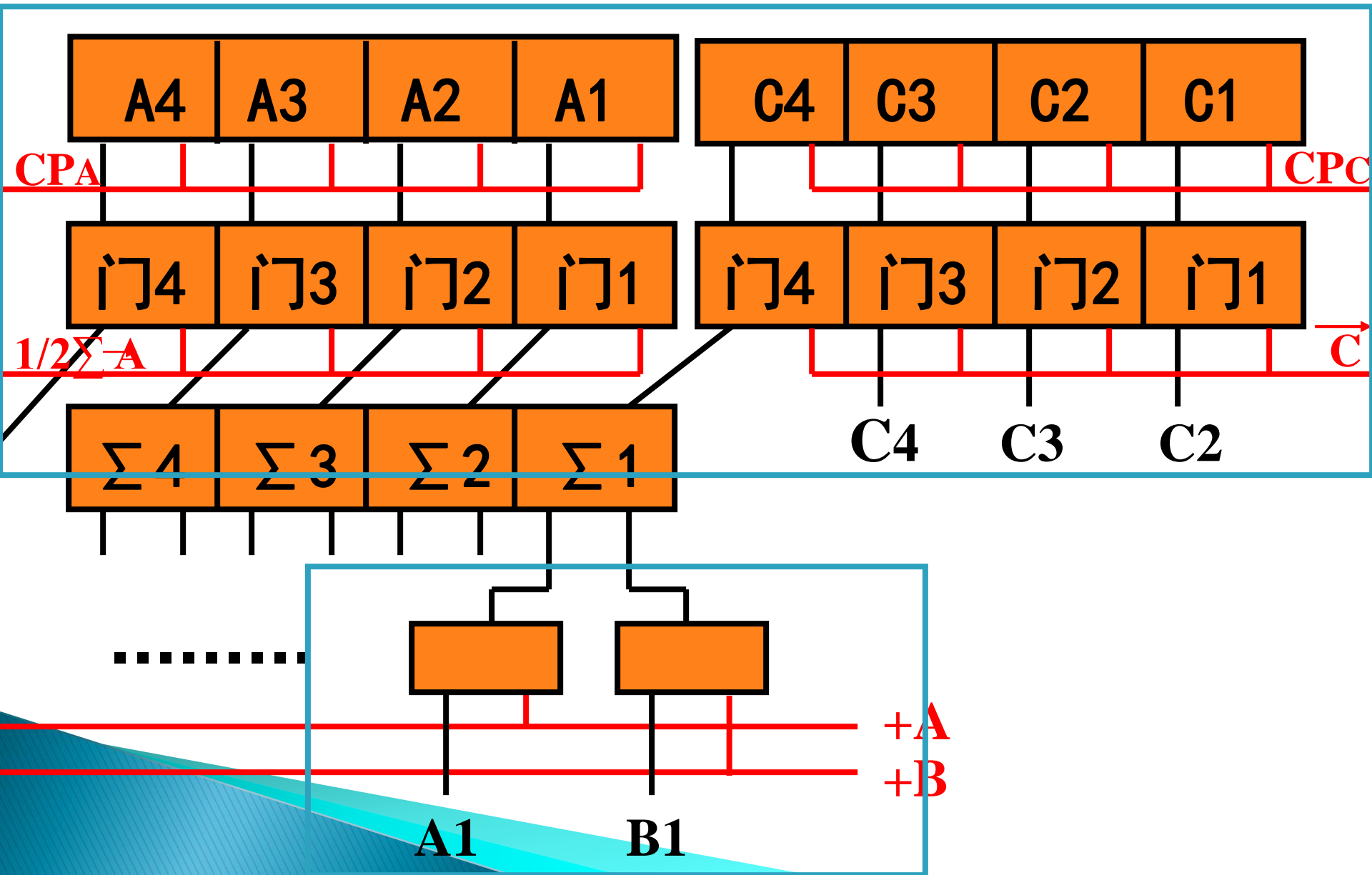
(3)运算规则

- ①操作数、结果用原码表示；
- ②被乘数(B)、累加和(A)取双符号位；
- ③乘数末位(C_n)为判断位，其状态决定下步操作；
- ④作 n 次循环（累加、右移）；
- ⑤绝对值运算，符号单独处理。

(4)逻辑实现

加法器输入端控制信号: $+A$ 、 $+B$

加法器输出端控制信号: $1/2\Sigma$ \vec{A} 、 \vec{C} 、 CP_A 、 CP_C



2 补码一位乘法

(1) 算法分析

$$X_{\text{补}} = X_0. X_1 X_2 \dots X_n$$

① Y 为正: $Y_{\text{补}} = 0. Y_1 Y_2 \dots Y_n$

$$(XY)_{\text{补}} = X_{\text{补}} (0. Y_1 Y_2 \dots Y_n)$$

② Y 为负: $Y_{\text{补}} = 1. Y_1 Y_2 \dots Y_n$

$$(XY)_{\text{补}} = X_{\text{补}} (0. Y_1 Y_2 \dots Y_n) + (-X)_{\text{补}}$$

③ Y 符号任意:

$$(XY)_{\text{补}} = X_{\text{补}} (0. Y_1 Y_2 \dots Y_n) + (-X)_{\text{补}} Y_0$$

符号位

④展开为部分积的累加和形式：

$$(XY)_{\text{补}} = X_{\text{补}} (0.Y_1Y_2\dots Y_n) + (-X)_{\text{补}} Y_0$$

$$= X_{\text{补}} (0.Y_1Y_2\dots Y_n) - X_{\text{补}} Y_0$$

$$= X_{\text{补}} (-Y_0 + 2^{-1}Y_1 + 2^{-2}Y_2 + \dots + 2^{-n}Y_n)$$

$$= X_{\text{补}} \left[\underline{-Y_0} + \underline{(Y_1 - 2^{-1}Y_1)} + \underline{(2^{-1}Y_2 - 2^{-2}Y_2)} + \dots \right. \\ \left. + \underline{(2^{-(n-1)}Y_n - 2^{-n}Y_n)} \right]$$

$$= X_{\text{补}} \left[(Y_1 - Y_0) + 2^{-1}(Y_2 - Y_1) + 2^{-2}(Y_3 - Y_2) + \dots \right. \\ \left. + 2^{-n}(Y_{n+1} - Y_n) \right]$$

$$= X_{\text{补}} [(Y_1 - Y_0) + 2^{-1} (Y_2 - Y_1) + 2^{-2} (Y_3 - Y_2) + \dots + 2^{-n} (Y_{n+1} - Y_n)]$$

$$[A_0]_{\text{补}} = 0$$

$$[A_1]_{\text{补}} = 2^{-1} \{ [A_0]_{\text{补}} + (Y_{n+1} - Y_n) [X]_{\text{补}} \} [X]_{\text{补}} \{ 2^{-1} (Y_{n+1} - Y_n) \}$$

$$[A_2]_{\text{补}} = 2^{-1} \{ [A_1]_{\text{补}} + (Y_n - Y_{n-1}) [X]_{\text{补}} \}$$

$$[X]_{\text{补}} \{ 2^{-1} (Y_n - Y_{n-1}) + 2^{-2} (Y_{n+1} - Y_n) \}$$

...

$$[A_n]_{\text{补}} = 2^{-1} \{ [A_{n-1}]_{\text{补}} + (Y_2 - Y_1) [X]_{\text{补}} \}$$

$$[XY]_{\text{补}} = [A_n]_{\text{补}} + (Y_1 - Y_0) [X]_{\text{补}}$$

比较法：用相邻两位乘数比较的结果决定+X补、-X补或+0。

(2) 比较法算法

Y_n (高位)	Y_{n+1} (低位)	操作 (A 补为部分积累加和)
0	0 (0)	$1/2A$ 补
0	1 (1)	$1/2(A$ 补 $+X$ 补)
1	0 (-1)	$1/2(A$ 补 $-X$ 补)
1	1 (0)	$1/2A$ 补

(3) 运算实例

$X = -0.1101$, $Y = -0.1011$, 求 (XY) 补。

初值: $A = 00.0000$, $B = X$ 补 $= 11.0011$,

$-B = (-X)$ 补 $= 00.1101$, $C = Y$ 补 $= 1.0101$

步数	条件 $C_n C_{n+1}$	操作	A	C
			00. 0000	$C_n C_{n+1}$ 1. 01010
1)	1 0	-B	+ 00. 1101	
			00. 1101	
		→	00. 0110	11. 0101
2)	0 1	+B	+ 11. 0011	
			11. 1001	
		→	11. 1100	111. 010
3)	1 0	-B	+ 00. 1101	
			00. 1001	
		→	00. 0100	1111. 01
4)	0 1	+B	+ 11. 0011	
			11. 0111	
	$Y_1 Y_2$	→	11. 1011	11111. 0

4)	0 1	+B	$\begin{array}{r} + 11.0011 \\ \hline 11.0111 \\ 11.1011 \end{array}$	
	$Y_1 \quad Y_2$	\rightarrow		1111.0
5)	1 0	-B	$\begin{array}{r} + 00.1101 \\ \hline 00.1000 \end{array}$	1111
		修正		

$$(XY)_{\text{补}} = 0.10001111$$

1.0 : -B修正
 0.1 : +B修正
 0.0 : 不修正
 1.1 : 不修正

$$[A_0]_{\text{补}} = 0$$

$$[A_1]_{\text{补}} = 2^{-1} \{ [A_0]_{\text{补}} + (Y_{n+1} - Y_n) [X]_{\text{补}} \}$$

$$[A_2]_{\text{补}} = 2^{-1} \{ [A_1]_{\text{补}} + (Y_n - Y_{n-1}) [X]_{\text{补}} \}$$

...

$$[A_n]_{\text{补}} = 2^{-1} \{ [A_{n-1}]_{\text{补}} + (Y_2 - Y_1) [X]_{\text{补}} \}$$

$$[XY]_{\text{补}} = [A_n]_{\text{补}} + (Y_1 - Y_0) [X]_{\text{补}}$$

步数	条件 $C_n C_{n+1}$	操作	A	C	$C_n C_{n+1}$
			00. 0000	1. 010	10
1)	1 0	-B	+ 00. 1101		

(1) A、B取双符号位，符号参加运算；

(2) C取单符号位，符号参加移位，以决定最后是否修正；

(3) C末位设置附加位 C_{n+1} ，初值为0， $C_n C_{n+1}$ 组成判断位，决定运算操作；

(4) 作n步循环，若需作第n+1步，则不移位，仅修正。

		→	00. 0100	1111. 01
4)	0 1	+B	+ 11. 0011	
			11. 0111	
		→	11. 1011	1111. 0
5)	1 0	-B	+ 00. 1101	

(4) 逻辑实现

加法器输入端控制信号： $+A$ 、 $+B$ 、 $+\bar{B}$ 、 $+1$

加法器输出端控制信号： $1/2\Sigma \rightarrow A$ 、 \overrightarrow{C} 、
 $\Sigma \rightarrow A$ 、 CP_A 、 CP_C

3.4 定点除法运算

除法——若干余数与除数加减、移位。

例. $0.10110 \div 0.11111$

$$\begin{array}{r} 0.10110 \\ 0.11111 \overline{) 0.101100} \\ \underline{-11111} \\ 110100 \\ \underline{-11111} \\ 101010 \\ \underline{-11111} \\ 0.0000010110 \end{array}$$

实现除法的关键：
比较余数、除数
绝对值大小，以
决定上商。

商： 0.10110

余数： 0.10110×2^{-5}

(1) 如何判断够减

先用逻辑电路进行比较判别

用减法试探 {
 恢复余数法 减后发现不够减，则商0，
 并加除数，恢复减前的余数
 不恢复余数除法 减后发现不够减，则在
 下一步改作加除数操作

(2) 如何处理符号位

原码除法

补码除法

$$\begin{array}{r} 0.11111 \overline{) 0.101100} \\ \underline{-11111} \\ 11010 \\ \underline{-11111} \\ +11111 \\ \hline 11010 \end{array}$$

1. 原码不恢复余数法（加减交替法）

(1) 算法分析

第 i 步: $2r_{i-1} - B = r_i' < 0$ 商0

第 $i+1$ 步: $r_i' + B = r_i$ (恢复余数)

第 $i+2$ 步: $2r_i - B = r_{i+1}$ 上商

第 i 步: $2r_{i-1} - B = r_i < 0$

第 $i+1$ 步: $2r_i + B = r_{i+1}$ (不恢复余数)

$$\begin{aligned} r_{i+1} &= 2r_i - B \\ &= 2(r_i' + B) - B \\ &= 2r_i' + B = r_{i+1} \end{aligned}$$

0.11111 $\overline{) 0.101100}$

0.101100

-11111

第 i 步 $\rightarrow r_i'$ { r_i 110100 } 第 $i+2$ 步 r_{i+1}

第 $i+1$ 步 \rightarrow { -11111 }

(2) 算法

$$r_{i+1} = 2r_i + (1 - 2Q_i)Y$$

r_i 为正, 则 Q_i 为 1, 第 $i+1$ 步作 $2r_i - Y$;

r_i 为负, 则 Q_i 为 0, 第 $i+1$ 步作 $2r_i + Y$ 。

(3) 实例

$X = 0.10110$, $Y = -0.11111$, 求 X/Y , 给出商 Q 和余数 R 。

初值: $A = |X| = 00.10110$

$B = |Y| = 00.11111$

$-B = 11.00001$

$C = |Q| = 0.00000$

步数	条件	操作	A	C
	r		00. 10110 r_0	0. 00000 C_n
1)		←	01. 01100 $2r_0$	
		-B	<u>+11. 00001</u>	
	为正		00. 01101 r_1	0. 00001 Q_1
2)		←	00. 11010 $2r_1$	
		-B	<u>+11. 00001</u>	
	为负		11. 11011 r_2	0. 00010 Q_2
3)		←	11. 10110 $2r_2$	
		+B	<u>+00. 11111</u>	
	为正		00. 10101 r_3	0. 00101 Q_3
4)		←	01. 01010 $2r_3$	
		-B	<u>+11. 00001</u>	
	为正		00. 01011 r_4	0. 01011 Q_4

步数	条件	操作	A	C
	为正		00.01011 r_4	0.01011 Q_4
5)		\leftarrow	00.10110 $2r_4$	
		$-B$	<u>+11.00001</u>	
	为负		11.10111 r_5'	0.10110 Q_5
6)		$+B$	<u>+00.11111</u>	
	恢复余数		00.10110 r_5	

$$Q = -0.10110$$

$$R = 0.10110 \times 2^{-5}$$

r_n 的位权应乘以 2^{-n} 。

商按同号相除为正，异号相除为负确定；
余数的实际符号与被除数的实际符号相同。

步数	条件	操作	A	C
	为正		00.01011 r_4	0.01011 Q_4
5)		\leftarrow	00.10110 $2r_4$	
		$-B$	<u>+11.00001</u>	

(4) r_n 的位权应乘以 2^{-n} 。

商按同号相除为正，异号相除为负确定；
余数的实际符号与被除数的实际符号相同。

$$Q = -0.10110$$

$$R = 0.10110 \times 2^{-5}$$

- (1) A、B取双符号位，X、Y取绝对值运算， $|X| < |Y|$ 。
 (2) 根据余数的正负决定商值及下一步操作。
 (3) 求n位商，作n步操作；若第n步余数为负，则第n+1步恢复余数，不移位。

(4)逻辑实现

加法器输入端控制信号：

$+2A$ 、 $+A$ 、 $+B$ 、 $+\bar{B}$ 、 $+1$

加法器输出端控制信号：

$\Sigma \rightarrow A$ 、 \bar{C} 、 $Q_i \rightarrow C_n$ 、 C_{PA} 、 C_{PC}

r_i 为正，则 Q_i 为1，第 $i+1$ 步作 $2r_i - Y$ ；

r_i 为负，则 Q_i 为0，第 $i+1$ 步作 $2r_i + Y$ 。