定点乘法运算

乘法 ——→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

$X_{原}$　　$Y_{原}$

乘积 $P = |X| \times |Y|$

积符 $S_P = S_X \oplus S_Y$

1）手算　　　例. 0.1101×1.1011

$$
\begin{array}{r}
0.1101 \\
\times 0.1011 \\
\hline
1101 \\
1101 \\
0000 \\
+\ 1101 \\
\hline
0.10001111
\end{array}
$$

—— 部分积

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

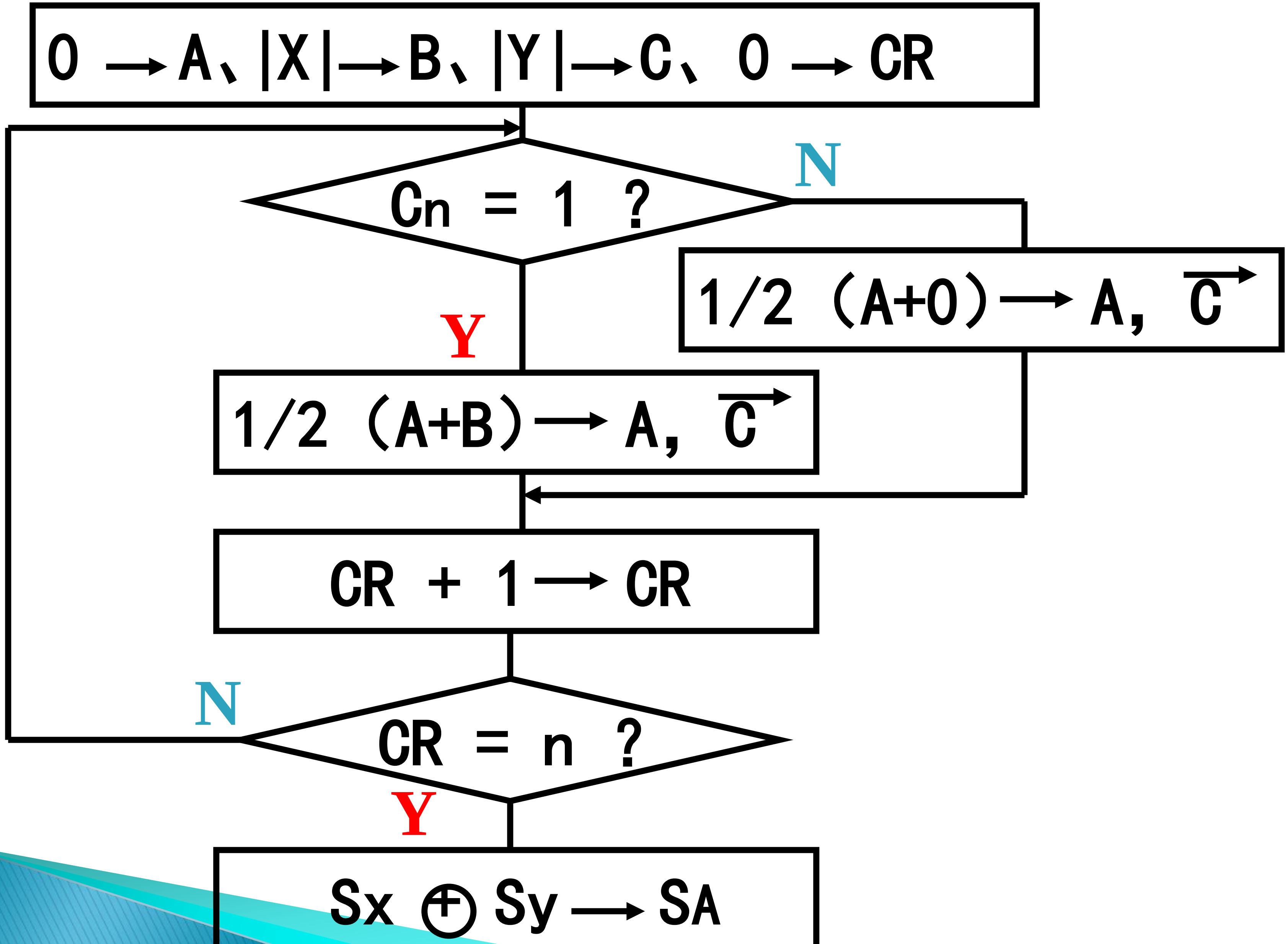
B = |X| = 00.1101

C = |Y| = .1011

步数	条件	操作	A	C (C_n)
			00.0000	.1011
1）	C_n=1	+B	+ 00.1101	
			00.1101	
		→	00.0110	1.101
2）	C_n=1	+B	+ 00.1101	
			01.0011	
		→	00.1001	11.10
3）	C_n=0	+0	+ 00.0000	
			00.1001	
		→	00.0100	111.1
4）	C_n=1	+B	+ 00.1101	
			01.0001	
		→	00.1000	1111

$X_{原} \times Y_{原} = 1.10001111$

(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；

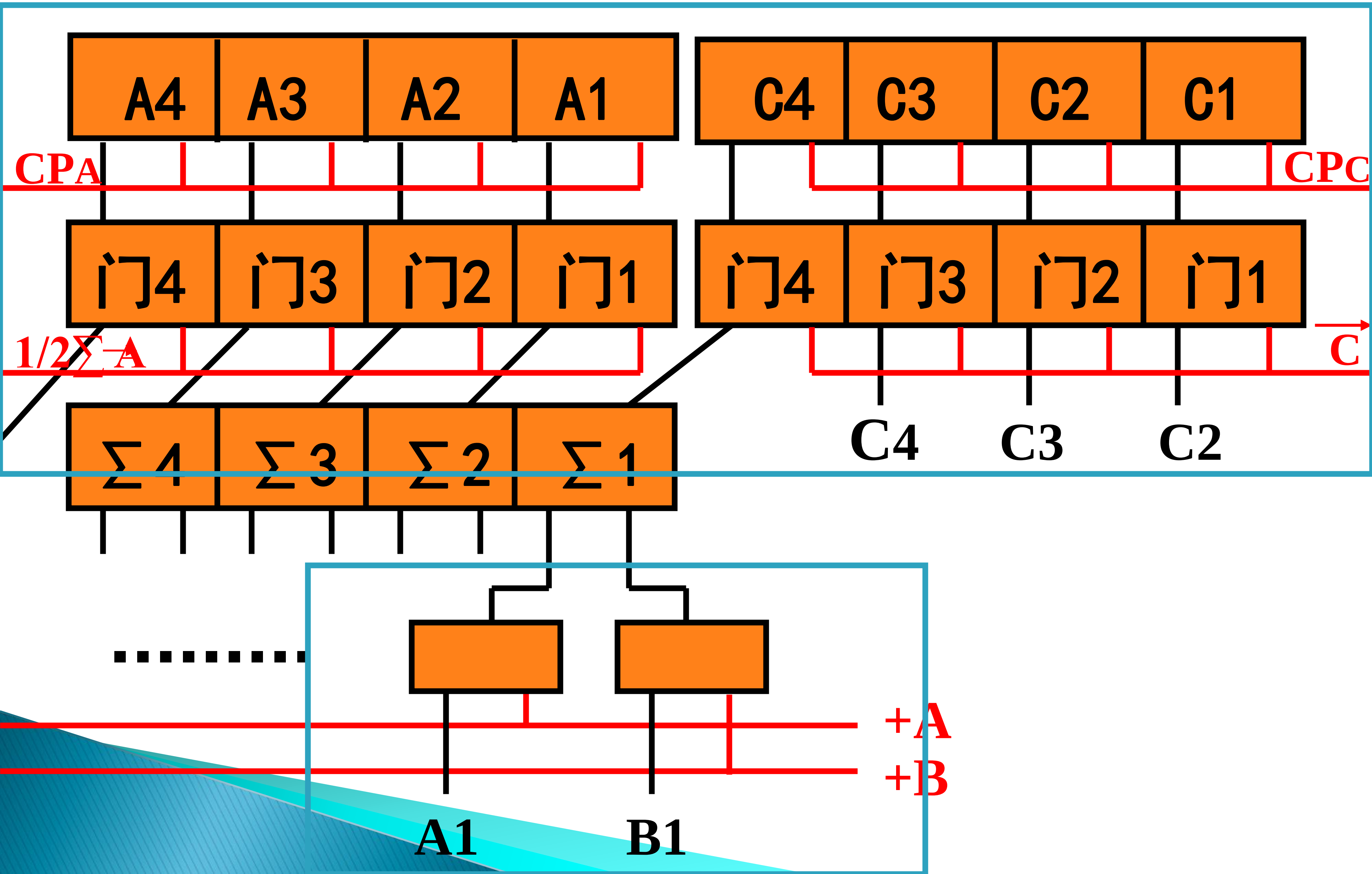
④作n次循环（累加、右移）；

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∑　$\vec{A}$、$\vec{C}$、CP_A、CP_C



2 补码一位乘法

(1)算法分析

$X_补 = X_0.X_1X_2......X_n$

①Y为正：$Y_补 = 0.Y_1Y_2......Y_n$

$(XY)_补 = X_补(0.Y_1Y_2......Y_n)$

②Y为负：$Y_补 = 1.Y_1Y_2......Y_n$

$(XY)_补 = X_补(0.Y_1Y_2......Y_n)+(-X)_补$

③Y符号任意：

$(XY)_补 = X_补(0.Y_1Y_2......Y_n)+(-X)_补Y_0$

符号位

④展开为部分积的累加和形式：

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

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

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

$$= X_{补}[\underline{-Y_0+(Y_1}-\underline{2^{-1}Y_1)+(2^{-1}Y_2}-2^{-2}Y_2)+\ldots\ldots$$

$$\underline{+(2^{-(n-1)}Y_n}-2^{-n}Y_n)]$$

$$= X_{补}[(Y_1-Y_0)+2^{-1}(Y_2-Y_1)+2^{-2}(Y_3-Y_2)+\ldots\ldots$$

$$+2^{-n}(Y_{n+1}-Y_n)]$$

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

$[A_0]_{补} = 0$

$$0$$

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

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

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

$$\cdots$$

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

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

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

Yn（高位）	Yn+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_nC_{n+1}	操作	A	C C_nC_{n+1}
			00.0000	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 + 11.0011

11.0111

Y_1 Y_2 ⟶ 11.1011 11111.0

5） 1 0 −B + 00.1101

修正 00.1000 1111

(XY)补 = 0.10001111

1.0： -B修正
0.1：+B修正
0.0：不修正
1.1：不修正

$$[A_0]_{补} = 0$$

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

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

…

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

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

步数	条件 C_nC_{n+1}	操作	A	C C_nC_{n+1}
			00.0000	1.01010
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_nC_{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	11111.0
5)	1 0	-B	+ 00.1101	

（4）逻辑实现

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

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÷0. 11111

```
               0.10110
0. 11111 ) 0. 101100
             -11111
               110100
              -11111
                101010
                -11111
0. 00000 10110
```

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

0.10

0. 11111) 0. 101100

−11111

11010

−11111

+11111

11010

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

$r_{i+1}=2r_i-B$

$=2(r_i'+B)-B$

$=2r_i'+B=r_{i+1}$

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)

0.10

0.11111) 0.101100

−11111

第i步→ r_i' r_i 110100

第i+1步 −11111

r_{i+1} 第i+2步

(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	C_n
	r		00.10110	r0	0.00000	
1）		←	01.01100	2r0		
		−B	+11.00001			
	为正		00.01101	r1	0.00001	Q1
2）		←	00.11010	2r1		
		−B	+11.00001			
	为负		11.11011	r2	0.00010	Q2
3）		←	11.10110	2r2		
		+B	+00.11111			
	为正		00.10101	r3	0.00101	Q3
4）		←	01.01010	2r3		
		−B	+11.00001			
	为正		00.01011	r4	0.01011	Q4

步数	条件	操作	A		C
					C_n
	为正		00.01011	r_4	0.01011 Q_4
5）		←	00.10110	$2r_4$	
		−B	+11.00001		
	为负		11.10111	r_5'	0.10110 Q_5
6）		+B	+00.11111		
		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)		←	00.10110	$2r_4$	
		−B	+11.00001		

C_n

（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：

$+2A$、$+A$、$+B$、$+\overline{B}$、$+1$

加法器输出端控制信号：

$\Sigma \rightarrow A$、$\overleftarrow{C}$、$Q_i \rightarrow C_n$、CP_A、CP_C

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$2r_i+Y$。