# 3.5　运算器组织与运算方法

以加法器为基础，可实现各种简单的算术、逻辑运算处理；在加法器的基础上，增加移位传送功能，并选择输入控制条件，可实现乘除法处理；但这还不是一个完整的运算器。

# 3.5　运算方法

## 3.5.1　定点加减运算

### 补码加减法

数用补码表示，符号位参加运算。

实际操作能否只取决于操作码？

结果需不需修正？

如何将减法转换为加法？

# 1. 基本关系式

$$(X+Y)_{补} = X_{补} + Y_{补} \quad (1)$$

$$(X-Y)_{补} = X_{补} + (-Y)_{补} \quad (2)$$

式（1）：操作码为“加”时，两数直接相加。

例. 求$(X+Y)_{补}$

1) $X=3$ $X_{补}=0\ 0011$

$Y=2$ $Y_{补}=0\ 0010$ ＋

$0\ 0101$（+5补码）

2) $X=-3$ $X_{补}=1\ 1101$

$Y=-2$ $Y_{补}=1\ 1110$ ＋

$1\ 1011$（–5补码）

3) $X=3$ $X_{补}=0\ 0011$

$Y=-2$ $Y_{补}=1\ 1110$ ＋

$0\ 0001$（+1补码）

4) $X=-3$ $X_{补}=1\ 1101$

$Y=2$ $Y_{补}=0\ 0010$ ＋

$1\ 1111$（–1补码）

$$(X+Y)_{补} = X_{补} + Y_{补} \quad (1)$$

$$(X-Y)_{补} = X_{补} + (-Y)_{补} \quad (2)$$

式（2）：操作码为“减”时，将减转换为加。

即将减数变补后与被减数相加。

$Y_{补}$ —将$Y_{补}$变补→ $(-Y)_{补}$：不管$Y_{补}$为正或负，将其符号连同尾数一起各位变反，末位加1。

例. 求$(X-Y)_{补}$

1) X= 4 $X_{补}$=0 0100

Y= –5 $Y_{补}$=1 1011

$(-Y)_{补}$=0 0101 +

0 1001（+9补码）

2) X= –4 $X_{补}$=1 1100

Y= 5 $Y_{补}$=0 0101

$(-Y)_{补}$=1 1011 +

1 0111（–9补码）

注意：某数的**补码表示**与某数**变补**的区别。

例. $1\ 0101_{原} \xrightarrow{\text{补码表示}} 1\ 1011$

$0\ 0101_{原} \xrightarrow{\text{补码表示}} 0\ 0101$

符号位不变；
负数尾数改变，
正数尾数不变。

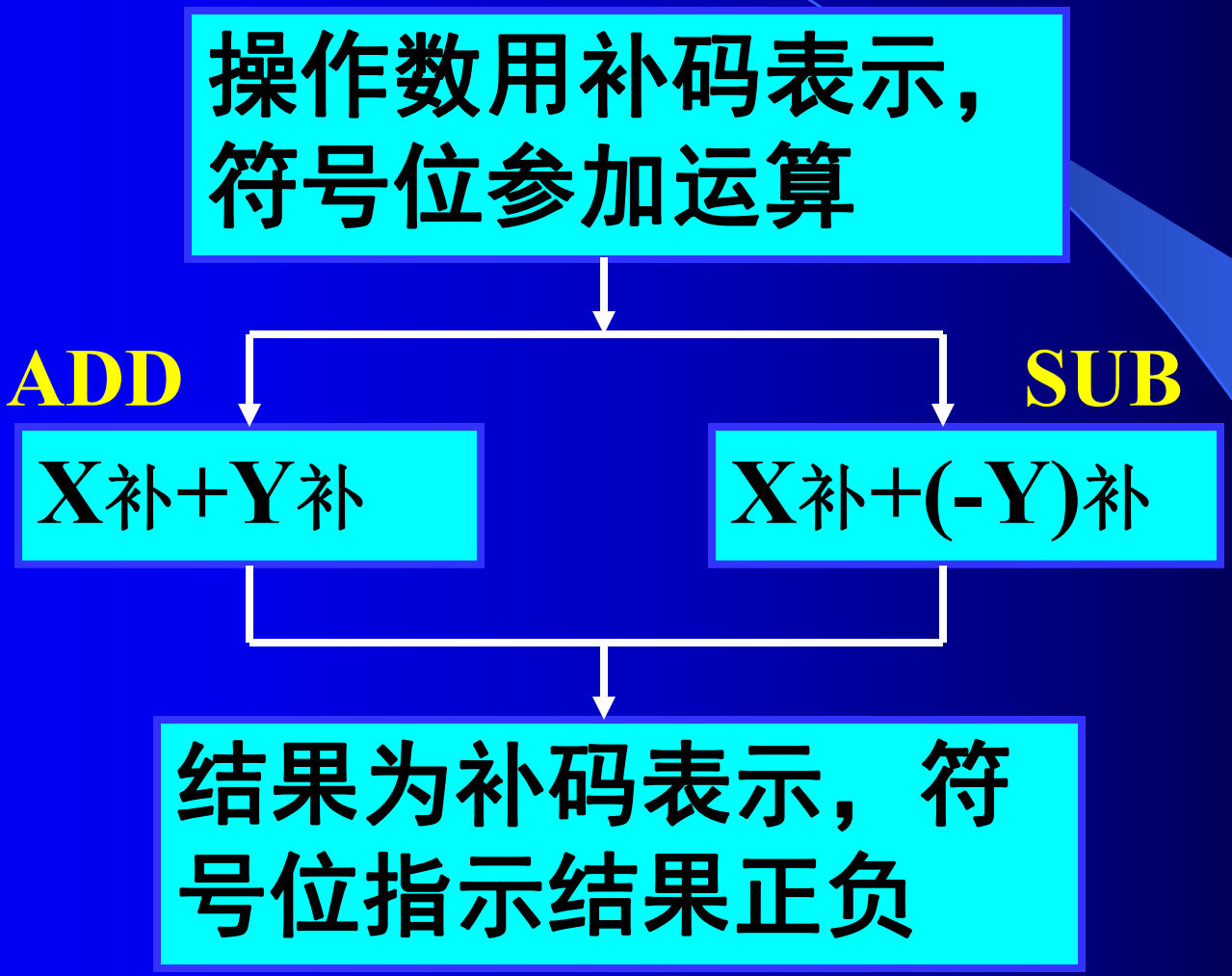
$1\ 0011_{补} \xrightarrow{\text{变补}} 0\ 1101$

$0\ 0011_{补} \xrightarrow{\text{变补}} 1\ 1101$

符号位改变，
尾数改变。

**补码的机器负数**

# 2. 算法流程



# 3. 逻辑实现

## （1）控制信号

加法器输入端：

+A：打开控制门，将A送Σ。

+B：打开控制门，将B送Σ。

$+\overline{B}$：打开控制门，将$\overline{B}$送Σ。

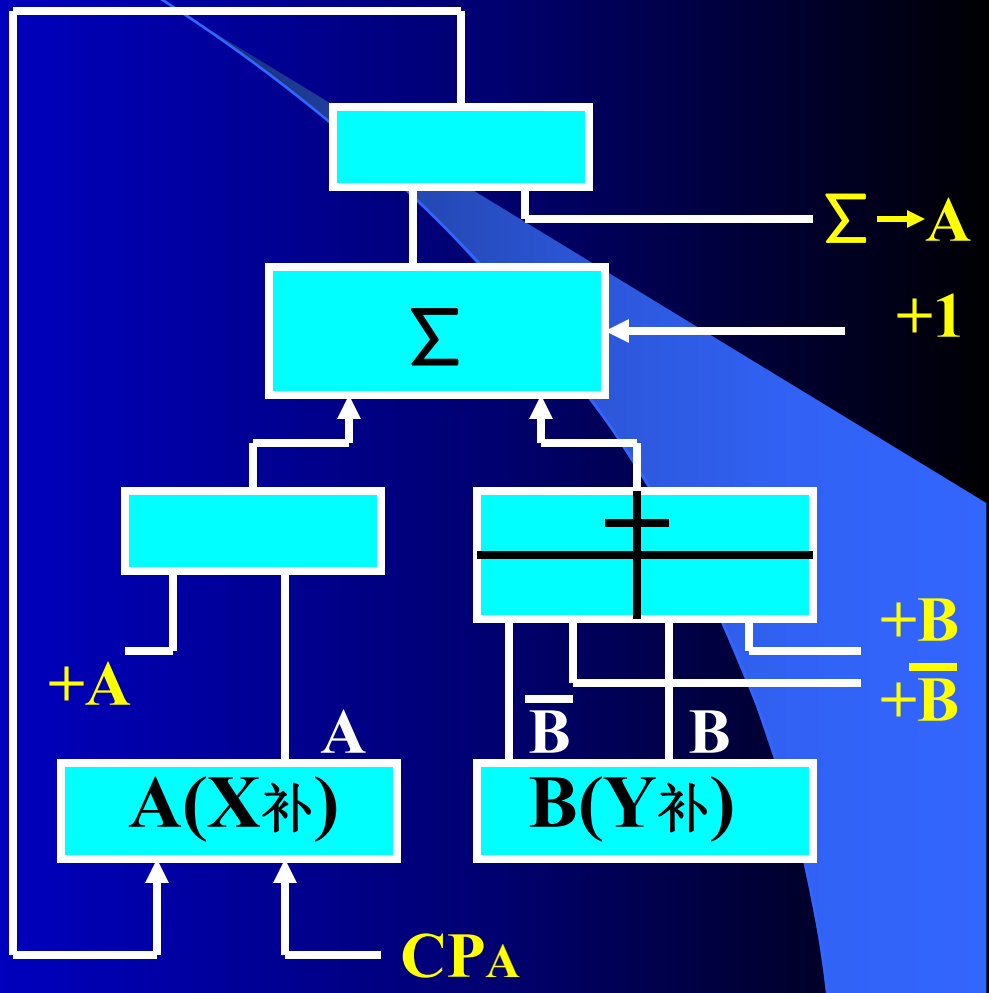
+1：控制末位加 1 。

加法器输出端：

Σ→A：打开控制门，将结果送A输入端。

CPA：将结果打入A。

## （2）补码加减运算器粗框



# 3.5.2 溢出判断与移位等

## 一.溢出判断

在什么情况下可能产生溢出？

例. 数A有4位尾数，1位符号SA

数B有4位尾数，1位符号SB

符号位参加运算

结果符号Sf

符号位进位Cf

尾数最高位进位C

（1）A=3 B=2

3+2： 0 0011

0 0010 +

0 0101 正确

（2）A=10 B=7

10+7： 0 1010

0 0111 +

1 0001 正溢

（3）A= -3 B= -2

-3+(-2)：1 1101

1 1110 +

1 1011 正确

（4）A= -10 B= -7

-10+(-7)：1 0110

1 1001 +

0 1111 负溢

（5）A=6 B= -4

6+(-4)：0 0110

1 1100 +

0 0010 正确

（6）A= -6 B=4

-6+4： 1 1010

0 0100 +

1 1110 正确

## 1. 硬件判断逻辑一（$S_A$、$S_B$与Sf的关系）

（2）A=10 B=7

10+7： 0 1010
+ 0 0111
1 0001

（4）A= -10 B= -7

-10+(-7)： 1 0110
+ 1 1001
0 1111

溢出= $\overline{S_A}\,\overline{S_B}\,S_f + S_A\,S_B\,\overline{S_f}$

## 2. 硬件判断逻辑二（Cf与C的关系）

（1）A=3 B=2

3+2： 0 0011

Cf=0 0 0010 +

C =0 0 0101 正确

（2）A=10 B=7

10+7： 0 1010

Cf=0 $0_{1}$0111 +

C =1 1 0001 正溢

（3）A= -3 B= -2

-3+(-2)：1 1101

Cf=1 $_{1}1_{1}$1110 +

C =1 1 1011 正确

（4）A= -10 B= -7

-10+(-7)：1 0110

Cf=1 $_{1}$1 1001 +

C =0 0 1111 负溢

（5）A=6 B= -4

6+(-4)：0 0110

Cf=1 $_{1}1_{1}$1100 +

C =1 0 0010 正确

（6）A= -6 B=4

-6+4： 1 1010

Cf=0 0 0100 +

C =0 1 1110 正确

## 1. 硬件判断逻辑一（$S_A$、$S_B$与Sf的关系）

（2）A=10 B=7

10+7： 0 1010

0 0111 +

1 0001

（4）A= -10 B= -7

-10+(-7)：1 0110

1 1001 +

0 1111

溢出= $\overline{S_A}\,\overline{S_B}\,S_f + S_A\,S_B\,\overline{S_f}$

## 2. 硬件判断逻辑二（Cf与C的关系）

溢出= $C_f \oplus C$

## 3. 硬件判断逻辑三（双符号位）

（1）3+2：

00 0011

00 0010 +

00 0101 正确

（2）10+7：

00 1010

00 0111 +

01 0001 正溢

第一符号位Sf1

（3）-3+(-2)：

11 1101

11 1110 +

11 0111 正确

（4）-10+(-7)：

11 0110

11 1001 +

10 1111 负溢

第二符号位Sf2

（5）6+(-4)：

00 0110

11 1100 +

00 0010 正确

（6）-6+4：

11 1010

00 0100 +

11 1110 正确

## 1. 硬件判断逻辑一（$S_A$、$S_B$与Sf的关系）

（2）A=10 B=7

10+7： 0 1010

0 0111 +

1 0001

（4）A= -10 B= -7

-10+(-7)：1 0110

1 1001 +

0 1111

溢出= $\overline{S_A}\,\overline{S_B}\,S_f + S_A S_B \overline{S_f}$

## 2. 硬件判断逻辑二（Cf与C的关系）

溢出= $C_f \oplus C$

## 3. 硬件判断逻辑三（双符号位）

溢出= $S_{f1} \oplus S_{f2}$

# 二. 移位操作

## 1. 移位类型

逻辑移位 ： 数码位置变化，数值不变。

1 0 0 0 1 1 1 1

循环左移： 0 0 0 1 1 1 1 1

算术移位 ： 数码位置变化，数值变化，符号位不变。

1 0 0 1 1 1 1 (-15)

算术左移： 1 0 1 1 1 1 0 (-30)

## 2.正数补码移位规则

（1）单符号位：

0 0111

左移 ← 0 1110

右移 → 0 0111

右移 → 0 0011

（2）双符号位：

00 0111

左移 ← 00 1110

左移 ← 01 1100

右移 → 00 1110

右移 → 00 0111

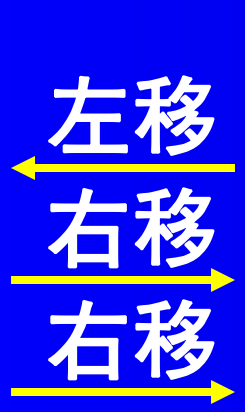
（3）移位规则

**数符不变**（单：符号位不变；双：第一符号位不变）。

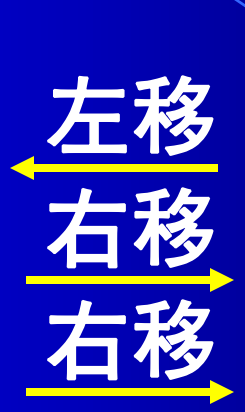
**空位补0**（右移时第二符号位移至尾数最高位）。

# 3.负数补码移位规则

（1）单符号位：




（2）双符号位：




（3）移位规则

**数符不变**（单：符号位不变；双：第一符号位不变）。

**左移空位补0**

**右移空位补1**（第二符号位移至尾数最高位）。

# 三. 舍入方法

## 1. 0舍1入（原码、补码）

例. 保留4位尾数：

0 00100原 ⟶ 0 0010原

1 00101原 ⟶ 1 0011原

1 11011补 ⟶ 1 1110补

## 2. 末位恒置1（原码、补码）

例. 保留4位尾数：

0 00100原 ⟶ 0 0011原

1 00101原 ⟶ 1 0011原

1 11011补 ⟶ 1 1101补

# 3.5.3 定点乘法运算

手算乘法如何实现？

例： 0.1101×0.1011 = ？

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

由手算到机器实现，要解决三个问题：

① 符号问题如何处理？

② 多项部分积相加，如何解决进位传递问题？

③ 乘数权值每高一位，新部分积需左移一位，才能保持两次部分积之间的位权对应关系，这导致加法器位数增加。为了保持加法器位数不变，能否改变移位方法？

对于符号位的处理方法，可采用原码乘法或补码乘法。

对于后两种问题的处理方法：一种乘法器是将N位乘法转换为 N次累加与移位循环，因而可用常规加法器实现。另一类乘法器结构，称为阵列乘法器。

本节主要讨论如何通过累加、移位实现分步乘法运算。

乘法 ⟶ 部分积累加、移位。

## 3.5.3.1 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

乘积 $P = |X| \times |Y|$

积符 $S_P = S_X \oplus S_Y$

（1）手算

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

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：

A = 00.0000

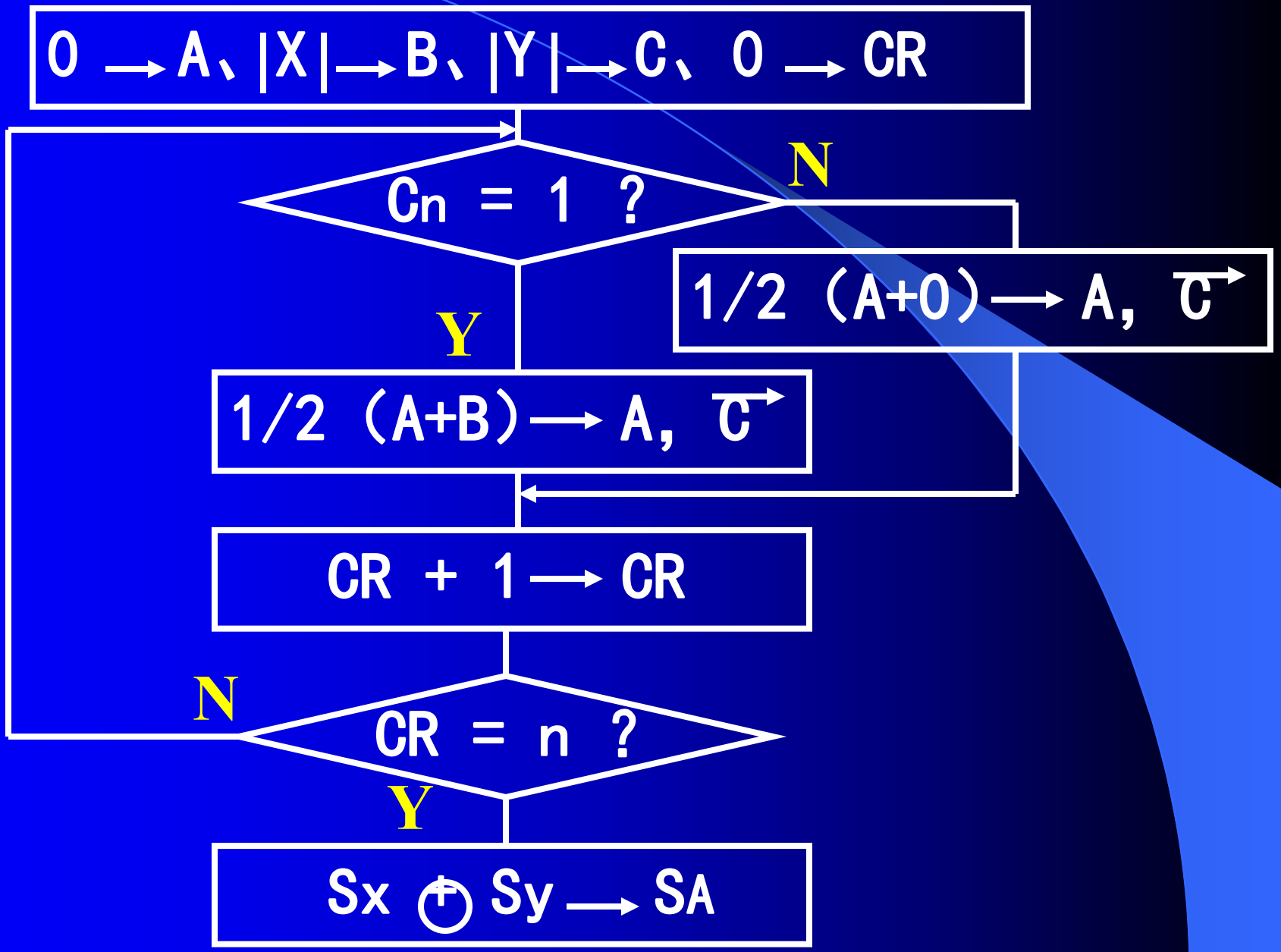
B = |X| = 00.1101

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）绝对值运算，符号单独处理；

（3）被乘数(B)、累加和(A)取双符号位；

（4）乘数末位($C_n$)为判断位，其状态决定下步操作；

（5）作n次循环（累加、右移）。

# 3.5.3.2 补码一位乘法

## 1. 算法分析

$X_{补} = X_0.X_1X_2\cdots\cdots X_n$

（1）Y为正：$Y_{补} = 0.Y_1Y_2\cdots\cdots Y_n$

$$(XY)_{补} = X_{补}(0.Y_1Y_2\cdots\cdots Y_n)$$

（2）Y为负：$Y_{补} = 1.Y_1Y_2\cdots\cdots Y_n$

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

（3）Y符号任意：

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

符号位

（4）展开为部分积的累加和形式：

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

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

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

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

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

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

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

*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 $C_nC_{n+1}$ | 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 | | |
| | | → | 11.1011 | 11111.0 | |
| 5） | 1 0 | −B | + 00.1101 | | |

4） 0 1 +B + 11.0011

11.0111

→ 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

## 4. 运算规则

(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。

# 3.5.4 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 ) 0.1011000
        - 0.11111
          ---------
          0.11010 0
        - 0.11111
          ---------
          0.10101 0
        - 0.11111
          ---------
          0.00010 1 1 0
```

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：$0.10110\times2^{-5}$

# 3.5.4.1 原码恢复余数法

## 1. 算法

比较两数大小可用减法试探。

2×余数-除数=新余数 {
- 为正：够减，商1。
- 为负：不够减，商0，恢复原余数。

## 2. 实例

X=-0.10110，Y=0.11111，求X/Y，给出商Q和余数R

设置：A：被除数、余数，B：除数，C：商

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 | Cn |
|---|---|---|---|---|---|
| | $S_A$ | | 00.10110 $r_0$ | 0.00000 | |
| 1） | | ← | 01.01100 $2r_0$ | | |
| | | −B | +11.00001 | | |
| | 0 | | 00.01101 $r_1$ | 0.00001 | $Q_1$ |
| 2） | | ← | 00.11010 $2r_1$ | | |
| | | −B | +11.00001 | | |
| | 1 | | 11.11011 $r_2'$ | 0.00010 | $Q_2$ |
| 3） | | +B | +00.11111 | | |
| | | 恢复余数 | 00.11010 $r_2$ | | |
| 4） | | ← | 01.10100 $2r_2$ | | |
| | | −B | +11.00001 | | |
| | 0 | | 00.10101 $r_3$ | 0.00101 | $Q_3$ |

| 步数 | 条件 | 操作 | A | C | Cn |
|---|---|---|---|---|---|
| | | | 00.10101 $r_3$ | 0.00101 | $Q_3$ |
| 5） | | ← | 01.01010 $2r_3$ | | |
| | | −B | +11.00001 | | |
| | 0 | | 00.01011 $r_4$ | 0.01011 | $Q_4$ |
| 6） | | ← | 00.10110 $2r_4$ | | |
| | | −B | +11.00001 | | |
| | 1 | | 11.10111 $r_5'$ | 0.10110 | $Q_5$ |
| 7） | | +B | +00.11111 | | |
| | | 恢复余数 | 00.10110 r5 | | |

$Q = -0.10110$

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

$$X/Y = -0.10110 + \frac{-0.10110 \times 2^{-5}}{0.11111}$$

## 3. 说明

（1）A、B双符号位，X、Y绝对值，|X| 小于|Y| 。

（2）运算结束后，余数乘以$2^{-n}$，与被除数同号。

## 3.5.4.2 原码不恢复余数法（加减交替法）

### 1. 算法分析

第二步：$2r_1-B=r_2'<0$

第三步：$r_2'+B=r_2$（恢复余数）

第四步：$2r_2-B=r_3$

$2r_2-B=2(r_2'+B)-B$

$=2r_2'+B=r_3$

第二步：$2r_1-B=r_2<0$

第三步：$2r_2+B=r_3$

（不恢复余数）

## 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 $r_0$ | 0.00000 | |
| 1） | | ← | 01.01100 $2r_0$ | | |
| | | −B | +11.00001 | | |
| | 为正 | | 00.01101 $r_1$ | 0.00001 | $Q_1$ |
| 2） | | ← | 00.11010 $2r_1$ | | |
| | | −B | +11.00001 | | |
| | 为负 | | 11.11011 $r_2$ | 0.00010 | $Q_2$ |
| 3） | | ← | 11.10110 $2r_2$ | | |
| | | +B | +00.11111 | | |
| | 为正 | | 00.10101 $r_3$ | 0.00101 | $Q_3$ |
| 4） | | ← | 01.01010 $2r_3$ | | |
| | | −B | +11.00001 | | |
| | 为正 | | 00.01011 $r_4$ | 0.01011 | $Q_4$ |

| 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 | $r_5$ | | |

$Q = -0.10110$

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

$$X/Y = -0.10110 + \frac{0.10110 \times 2^{-5}}{-0.11111}$$

## 4. 运算规则

（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。

# 3.5.4.3 补码不恢复余数法（加减交替法）

## 如何判断是否够减?如何上商?如何确定商符?

### 1. 判够减

(1) 同号相除

| | | | |
|---|---|---|---|
| 1 | 0 | 1 | 0 |
| $4 \overline{)\ 7}$ | $7 \overline{)\ 4}$ | $-4 \overline{)\ -7}$ | $-7 \overline{)\ -4}$ |
| $-4$ | $-7$ | $-(-4)$ | $-(-7)$ |
| 3 | $-3$ | $-3$ | 3 |
| 够减 | 不够减 | 够减 | 不够减 |

够减：r与X、Y同号；不够减：r与X、Y异号。

(2) 异号相除

| | | | |
|---|---|---|---|
| 1 | 0 | 1 | 0 |
| $-4 \overline{)\ 7}$ | $-7 \overline{)\ 4}$ | $4 \overline{)\ -7}$ | $7 \overline{)\ -4}$ |
| $+(-4)$ | $+(-7)$ | $+4$ | $+7$ |
| 3 | $-3$ | $-3$ | 3 |
| 够减 | 不够减 | 够减 | 不够减 |

够减：r与X同号,与Y异号；不够减：r与X异号,与Y同号。

（3）判断规则

$\frac{X_{补}}{Y_{补}}$ 同号：作 $X_{补}-Y_{补}$ ｛ 够减：$r_{补}$ 与 $Y_{补}$ 同号；不够减：$r_{补}$ 与 $Y_{补}$ 异号

$\frac{X_{补}}{Y_{补}}$ 异号：作 $X_{补}+Y_{补}$ ｛ 够减：$r_{补}$ 与 $Y_{补}$ 异号；不够减：$r_{补}$ 与 $Y_{补}$ 同号

2. 求商值

$\frac{X_{补}}{Y_{补}}$ 同号：商为正 ｛ 够减商1（r、Y同号）；不够减商0（r、Y异号）

$\frac{X_{补}}{Y_{补}}$ 异号：商为负 ｛ 够减商0（r、Y异号）；不够减商1（r、Y同号）

上商规则：$Q_i=\overline{Sr_i \oplus S_Y}$

余数与除数同号商1，异号商0。

## 3. 算法

$$(r_{i+1})_{补}=2r_{i_{补}}+(1-2Q_{i_{补}})Y_{补}$$

$r_{i_{补}}$与$Y_{补}$同号，则$Q_{i_{补}}$为1，第i+1步作$2r_{i_{补}}-Y_{补}$；

$r_{i_{补}}$与$Y_{补}$异号，则$Q_{i_{补}}$为0，第i+1步作$2r_{i_{补}}+Y_{补}$。

## 4. 求商符

令$X_{补}=r0_{补}$　　$r0_{补}$与$Y_{补}$ { 同号：$Q0_{补}=1$ ；异号：$Q0_{补}=0$

商符

与实际商符相反

## 5. 商的校正

$$\frac{X_{补}}{Y_{补}}=\underbrace{(-1+2^{-n}+\sum_{i=0}^{n-1}2^{-i}Q_{i_{补}})}_{商}+\underbrace{\frac{2^{-n}r_{n_{补}}}{Y_{补}}}_{余数}$$

余数

$$\frac{X_{补}}{Y_{补}} = \underbrace{(-1+2^{-n}+\sum_{i=0}^{n-1}2^{-i}Qi_{补})}_{商} + \frac{2^{-n}r_{n_{补}}}{Y_{补}}$$

(1) $\sum_{i=0}^{n-1}2^{-i}Qi_{补}$ =Q0. Q1Q2……Qn-1 求n-1位商(假商)

(2) $2^{-n}$ 第n位商(末位商)恒置1

(3) -1 商符变反

真商=假商+1. $\underbrace{000\ldots01}_{n位}$

(4) 余数求至$r_n$

# 6. 实例

X=0.10110，Y=-0.11111，求X/Y，给出商Q和余数R。

初值：A $=X_{补}$=00.10110　　B $=Y_{补}$=11.00001

　　-B =00.11111　　C $=Q_{补}$=0.00000

| 步数 | 条件 | 操作 | A | | C |
|---|---|---|---|---|---|
| | r、Y异号 | 求商符 | 00.10110 | $r_0$ | 0.0000 $Q_0$ |
| 1） | | ← | 01.01100 | $2r_0$ | |
| | | +B | +11.00001 | | |
| | 异号 | | 00.01101 | $r_1$ | 0.0000 $Q_1$ |
| 2） | | ← | 00.11010 | $2r_1$ | |
| | | +B | +11.00001 | | |
| | 同号 | | 11.11011 | $r_2$ | 0.0001 $Q_2$ |

Cn-1

| 步数 | 条件 | 操作 | A | C | $C_{n-1}$ |
|---|---|---|---|---|---|
| | r、Y | | 11.11011 $r_2$ | 0.0001 | $Q_2$ |
| 3） | | ← | 11.10110 $2r_2$ | | |
| | | −B | +00.11111 | | |
| | 异号 | | 00.10101 $r_3$ | 0.0010 | $Q_3$ |
| 4） | | ← | 01.01010 $2r_3$ | | |
| | | +B | +11.00001 | | |
| | 异号 | | 00.01011 $r_4$ | 0.0100 | $Q_4$ |
| 5） | | ← | 00.10110 $2r_4$ | | |
| | | +B | +11.00001 | | |
| | | | 11.10111 $r_5$ | | |

假商=0.0100

真商=0.0100+1.00001=1.01001

$Q = -0.10111$　　$R = -0.01001 \times 2^{-5}$

$$X/Y = -0.10111 + \frac{-0.01001 \times 2^{-5}}{-0.11111}$$

7. 运算规则

（1）A、B取双符号位，符号参加运算，并且 |X|＜|Y| 。

（2）根据余数与除数的符号决定商值及下一步操作。

（3）求n-1位商，作n步操作（求出rn）。

（4）对商校正（商符变反，第n位商恒置1）。

## 3.5.5 浮点四则运算

# 3.5.5.1 浮点加减运算

步骤：

1. 检测能否简化操作。

判操作数是否为0 〈 尾数为0；阶码下溢

2. 对阶

例. $2^{2}\times 0.1001 \rightarrow 10.0\ 1 \rightarrow 010.01 \rightarrow 2^{3}\times 0.0101$

$2^{3}\times 0.1101 \rightarrow 11\ 0.1 \rightarrow 110.1 \rightarrow 2^{3}\times 0.1101$

(1) 对阶：使两数阶码相等(小数点实际位置对齐，尾数对应权值相同)。

(2) 对阶规则：小阶向大阶对齐。

(3) 对阶操作：小阶阶码增大，尾数右移。

例. AE>BE，则BE+1→BE，$\overrightarrow{BM}$，直到BE=AE

(4) 阶码比较：比较线路或减法。

## 3. 尾数加减.

$$AM \pm BM \longrightarrow AM$$

## 4. 结果规格化

(1)　1.0001
　　+0.1001
　　1.1010

|M|<1/2

应左移规格化

(2)　0.0101
　　+0.1101
　　1.0010

|M|>1

应右移规格化

(1) 11.0001

+00.1001

11.1010

$A_{f1}$ $A_{f2}$ $A_1$

若 $\overline{A}_{f1}\overline{A}_{f2}\overline{A}_1+A_{f1}A_{f2}A_1=1$，则左规：$\overleftarrow{A_M}$

(-1/2除外)

$A_E-1 \rightarrow A_E$

(2) 00.0101

+00.1101

01.0010

$A_{f1}$ $A_{f2}$

若 $A_{f1} \oplus A_{f2}=1$，则右规：$\overrightarrow{A_M}$

$A_E+1 \rightarrow A_E$

# 3.5.5.2 浮点乘法运算

设$A=2^{A_E}\times A_M$，$B=2^{B_E}\times B_M$　$A\times B=2^{A_E+B_E}\times(A_M\times B_M)$

浮点乘——→定点加、定点乘

步骤：

1. 检测操作数是否为0。
2. 阶码相加。

若阶码用移码表示，相加后要修正。

3. 尾数相乘。 相乘前不需对阶。
4. 结果规格化。一般左规。

# 3.5.5.3 浮点除法运算

设$A=2^{A_E}\times A_M$，$B=2^{B_E}\times B_M$　$A\div B=2^{A_E-B_E}\times(A_M\div B_M)$

浮点除 ⟶ 定点减、定点除

步骤：

1. 检测操作数是否为0。
2. $|A_M|<|B_M|$？
3. 阶码相减。

若阶码用移码表示，相减后要修正。

4. 尾数相除。　相除前不需对阶。
5. 结果不再规格化。

# 第三章作业（一）

简要回答下列问题

（1）原码一位乘法与补码一位乘法的主要区别是什么？

（2）原码加减交替除法和补码加减交替除法分别根据什么情况商1、商0？

（3）在浮点加减运算中如何进行对阶操作？

（4）在什么情况下需要左移规格化？什么情况下需要右移规格化？