3.6 浮点运算

机器浮点数格式：

尾数：用定点小数表示，给出**有效数字**的位数，决定了浮点数的表示**精度**，一般为原码或补码；

阶码：用定点整数表示，指明**小数点在数据中的位置**，决定了浮点数的表示**范围**，一般用移码或补码表示。

E_s	E_1 E_2 …… E_m	M_s	M_1 M_2 …… M_n
阶符	阶码	数符	尾数

3.6.1 浮点加、减法

两个二进制浮点数 X 和 Y 可以表示为：

$$X = M_X \times 2^{E_X} \qquad Y = M_Y \times 2^{E_Y}$$

说明：浮点数 X 和 Y 是按规格化数存放的，它们运算后的结果也应该是规格化的。

十进制数的加法运算：

$$0.23*10^2+0.45*10^3=0.023*10^3+0.45*10^3$$

$$=(0.023+0.45)*10^3=0.473*10^3$$

浮点数加减法运算规则

$$X = M_X \times 2^{E_X} \qquad Y = M_Y \times 2^{E_Y}$$

不妨设 $E_x \le E_y$ ，浮点数 X 和 Y 加减法运算的规则为：

$$X + Y = (M_X \times 2^{E_X - E_Y} + M_Y) \times 2^{E_Y}$$

$$X - Y = (M_X \times 2^{E_X - E_Y} - M_Y) \times 2^{E_Y}$$

浮点数加减法运算步骤

第 1 步：对阶，调整尾数

第 2 步：尾数加减

第 3 步：尾数规格化

第 4 步：尾数的舍入处理

第 5 步：阶码溢出判断

计算机中实现 X 和 Y 加、减法运算的步骤为：

第 1 步：对阶

阶码的比较通过两阶码的减法来实现，对阶使得原数中较大的阶码成为两数的公共阶码；

小阶码的尾数按两阶码的差值决定右移的数量。

令$X \pm Y = M_b \times 2^{E_b}$

两阶码的差值表示为：$\Delta E = E_x - E_y$

1) 若$\Delta E \le 0$，则$E_b \leftarrow E_y$，$E_x \leftarrow E_y$，$M_x \leftarrow M_x * 2^{Ex-Ey}$

2) 若$\Delta E > 0$，则$E_b \leftarrow E_x$，$E_y \leftarrow E_x$，$M_y \leftarrow M_y * 2^{Ey-Ex}$

小阶码的尾数右移时应**注意**：

1) **原码**形式的尾数右移时，**符号位不参加移位**，数值位右移，空出位补 0 。

补码形式的尾数右移时，**符号位与数值位一起右移**，空出位填补符号位的值。

2) 尾数的右移，使得尾数中原来的 **$|\Delta E|$ 位有效位移出**。

说明：

1) 移出的这些位先不要丢掉，应保留，并且参加后续运算。这对运算结果的精确度有一定影响。

2) 保留的多余的位数称为**警戒位**。

第 2 步：尾数加减

对尾数进行加、减运算

$$M_b \leftarrow M_x \pm M_y$$

第 3 步：尾数规格化

设浮点数的尾数用补码表示，且加、减运算时采用双符号位，则规格化形式的尾数应是如下形式：

尾数为正数时： 001xx…x

尾数为负数时： 110xx…x

尾数违反规格化的情况有以下两种可能：

1) 尾数加、减法运算中产生溢出。

正溢出时，符号位为 01

负溢出时，符号位为 10

规格化采取的方法是：

尾数右移一位，阶码加 1 ，称为右规。

操作： $M_b \leftarrow M_b * 2^{-1}$ ， $E_b \leftarrow E_b + 1$ 。

2) 尾数的绝对值小于二进制的 0.1 。

补码形式的尾数表现为最高数值位与符号位同值。

尾数为正数时：$\underline{00}$ $\underline{00\ldots01\times\ldots\times}$ (k1)

符号位　数值位

尾数为负数时：$\underline{11}$ $\underline{11\ldots10\times\ldots\times}$ (k1)

符号位　数值位

采取规格化的方法：

符号位不动，数值位逐次左移，阶码逐次减1；直到满足规格化形式的尾数，即最高数值位与符号位不同值为止。

这种规格化称为左规：

表示为：$M_b \leftarrow M_b * 2^k$，$E_b \leftarrow E_b - k$

第 4 步：尾数的舍入处理

对结果的尾数进行舍入处理的方法：

① 0 舍 1 入法

② 恒置 1 法

③ 恒舍法

① 0舍1入法

警戒位中的最高位为1时，就在尾数末尾加1

；

警戒位中的最高位为0时，舍去所有的警戒位；

该方法的最大误差：$\pm 2^{-(n+1)}$, n为有效尾数位数。

② 恒置1法

不论警戒位为何值，尾数的有效最低位恒置1。

恒置1法产生的最大误差为$\pm 2^{-n}$，n为有效尾数位数。

③ 恒舍法

无论警戒位的值是多少，都舍去。

尾数的结果就取其有效的 n 位的值。

称为**趋向零舍入** (Round toward zero) 。

说明：上述几种简单的舍入方法，

1) 对**原码**形式的尾数进行舍入处理，舍入的效果与**真值**舍入的效果是一致的；

2) 对于**补码**形式的**负**的尾数来说，所进行的舍入处理**与真值的舍入效果可能不一致**。

例 1 ：已知有 X = -0.101010 ，$[X]_{补}$ =1010110 ，
有效小数位数为 4 位。

分别对 X 和 $[X]_{补}$采用 0 舍 1 入法进行舍入处理，得：

X = - 0.1011 （入）

$[X]_{补}$ =10110 （入）

此时，对应的 X =- 0.1010

补码的舍入效果与真值的舍入效果不一致。

第 5 步：阶码溢出判断

1) 若阶码 E_b 正溢出，表示浮点数已超出允许表示范围的上限，置溢出标志。

2) 若阶码 E_b 负溢出，运算结果趋于零，置结果为机器零。

最后：浮点数加、减法运算正常结束，运算结果为 $M_b \times 2^{Eb}$ 。

例 2 ：已知 $X = 0.11011011*2^{010}$ ，

$Y = -0.10101100*2^{100}$ ；

用补码来表示浮点数的尾数和阶码。

$[X]_{浮}$ =0 0 010 11011011

$[Y]_{浮}$ =1 0 100 01010100

数符 阶符 阶码 尾数

$[X+Y]_{浮} = M_b*2^{Eb}$ ，执行 $[X+Y]_{浮}$ 的过程如下：

① 对阶

$\Delta E = E_x - E_y = 00\ 010 + 11\ 100 = 11\ 110$

$\Delta E = -2$ ， M_x 右移两位， M_x =0 0011011011

$E_b = E_x = E_y = 0\ 100$

② 尾数加法

$$M_b = M_x + M_y$$

$$
\begin{array}{r}
00\ 00110110\ \underline{11} \\
+11\ 01010100 \\
\hline
11\ 10001010\ \underline{11}
\end{array}
$$

因此 $M_b = 11\ 10001010\ 11$

③ 尾数规格化

尾数没有溢出，但符号位与最高数值位有 k=1 位相同，需左规：

M_b 左移 k=1 位：　M_b =11 00010101 1

E_b 减 1 ：　　　E_b 00 011

=

④ 舍入处理

尾数采用 0 舍 1 入法，

得， M_b =11 00010110

⑤ 阶码溢出判断

阶码无溢出， X+Y 正常结束，得：

$[X+Y]_{浮}$ ＝ 1 0 011 00010110

X+Y ＝ -0.11101010*2^{011}

3.6.2 浮点乘、除法运算

对两个规格化的浮点数 $X=M_x\times 2^{Ex}$ 和 $Y=M_y\times 2^{Ey}$ ，实现乘、除法运算的规则如下：

$$X\times Y=(M_x\times 2^{Ex})\times(M_y\times 2^{Ey})=(M_x\times M_y)\times 2^{Ex+Ey}$$

$$X/Y=(M_x\times 2^{Ex})/(M_y\times 2^{Ey})=(M_x/M_y)\times 2^{Ex-Ey}$$

说明：

1. 浮点数的乘、除法运算不需要对阶；
2. 两者对结果的后处理相同，包括：
    1) 结果数据规格化；
    2) 舍入处理；
    3) 阶码溢出判断。

1．浮点数乘法的运算步骤

（1）两浮点数相乘

乘积的阶码：相乘两数阶码之和；

乘积的尾数：相乘两数尾数之积。

表示为： $M_b = M_x \times M_y$ ， $E_b = E_x + E_y$

（2）尾数规格化

$1 \quad |M_x|, |M_y| \quad 0.1$

问题：尾数会溢出吗？需要右规吗？ 不

即 M_b 最多左移一位，阶码 E_b 减 1 。

（3）尾数舍入处理

$M_x \times M_y$ 产生双字长乘积，如果只要求得到单字长结果，那么低位乘积就当作警戒位进行结果舍入处理。

（4）阶码溢出判断

对 E_b 的溢出判断完全相同于浮点数加、减法的相应操作。

2. 浮点数除法的运算步骤

（1）除数是否为0，若 $M_y=0$，出错报告。

（2）两浮点数相除

商的阶码：被除数的阶码减去除数的阶码；
商的尾数：相除两数尾数的商。

表示为： $M_b= M_x/M_y$，$E_b= E_x-E_y$

（ 3 ）尾数规格化

$1 \quad |M_x|, |M_y| \quad 0.1$

若溢出，执行右规： M_b 右移一位，阶码 E_b 加 1

（ 4 ）尾数舍入处理

（ 5 ）阶码溢出判断

（ 4 ）、（ 5 ）两步操作相同于浮点数加、减法的相应操作。

3.6.2 浮点运算所需的硬件配置

浮点运算器的硬件配置比定点运算器复杂。

浮点运算分阶码和尾数两部分：

阶码：只有加减运算
尾数：加减乘除四种运算

运算器：主要由两个定点运算部件组成，

1) 阶码运算部件：加、减。
2) 尾数运算部件：加、减、乘、除，移位，判断规格化。
3) 阶码溢出判断。

浮点运算器：可做成独立的选件，如 Intel 80287 协处理器。

或编程实现浮点运算。