

浮点数加减法

$$X = 2^{E_x} M_x, Y = 2^{E_y} M_y$$

① $E_x = E_y, S = 2^{E_x} (M_x + M_y)$

② $E_x \neq E_y$, 必须先对阶

计算步骤: 对阶, 尾数运算, 规格化, 舍入, 溢出判断

① 对阶(使尾数可直接相加) \Rightarrow 小阶对大阶, 尾数右移 (通常通过保留附加位减少低位损失)

$$\text{eg. } 2^8 * (0.1100) + 2^6 * (0.0011)$$

大数小 $2^8 * 0.1100 \rightarrow 2^6 * 0.11000$ 溢出, 高位丢失

小数大 $2^6 * 0.0011 \rightarrow 2^8 * 0.00001111$ 低位丢失

② 尾数运算: 加法器

③ 规格化: 尾数非零时, 绝对值 ≥ 0.5 (尾数MSB=1); 否则修改阶码并移动尾数, 使其满足上述要求; 目的是保证浮点数的编码唯一性

方法: ① 右移以实现规格化 \Rightarrow 向左规格化 (右归, 阶码+1)

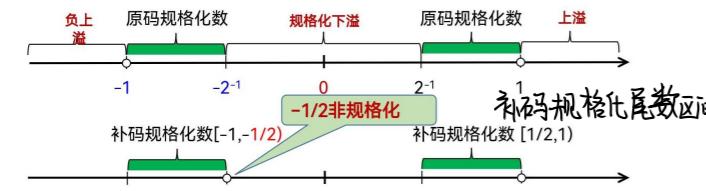
② 左移以实现规格化 \Rightarrow 向右规格化 (左归, 阶码-1)

■ 真值规格化数 $0.1XXXX - 0.1XXXX$

■ 原码规格化数 $0.1XXXX 1.1XXXX$

① 补码规格化数 $00.1XXXX 11.0XXXX \quad \left\{ \begin{array}{l} 11.0 \\ 00.1 \end{array} \right.$

■ 补码非规格化数 $00.0XXXX 11.1XXXX 01.XXXXXX 10.XXXXXX$



※ 补码尾数加减结果规格化简述:

① 尾数运算上溢时右归 (最多位) 尾号位之间关系

判断: 变形补码运算结果为 10XXX 或 0XXX 上溢

左归应连同尾号位一起右移, 阶码作加法

② 尾数运算下溢时左归 (可能多位) 尾号位与数值最高有效位关系

0.0XX, 0.000XX 或 1.0XX, 1.1110XX 下溢而非规格化

左归时 LSB 位补 0, 阶码作减法

④ 舍入的概念 (向右移位有可能丢失数据产生误差)

01100011 上溢右归

右归 $\rightarrow 00110001$

常用舍入方法: 0舍入, 截去法, 棍值置1法; 之后可能需要再次规格化

不同舍入方式影响精度与电路实现的复杂度

⑤ 溢出处理

浮点数溢出判定标准 \Rightarrow 阶码运算是否溢出

① 上溢 $|E| > 2^n$: 尾数上溢, 右归, 阶码正上溢

② 下溢 $|E| < -n$: 尾数下溢, 左归, 阶码负上溢

改善浮点数运算精度

采用更长尾数的浮点运算器 \Rightarrow IEEE754 在中间结果须在右加3附加位

采用更好的舍入处理 \Rightarrow IEEE754 舍入处理 ① 最近偶数舍入 ② 向0舍入 ③ 向+∞舍入 ④ 向-∞舍入

向+∞舍入

IEEE754 的舍入处理

最近偶数舍入 0000 111 最近舍入, 居中向偶数舍入
向0舍入 0000 110 向+∞ 0000 111
向-∞ 0000 110

舍入方式	1.2	1.6	1.5	2.5	2.6	-1.5
向零舍入	1	1	1	2	2	-1
向正无穷舍入	2	2	2	3	3	-1
向负无穷舍入	1	1	1	2	2	-2
最近偶数舍入	1	2	2	2	3	-2

IEEE754 浮点数加减法

① 对阶—遵循移码运算规则

移码表示阶码原因:

(1) 相当于转为无符号数, 阶码大小与浮点数大小一致, 较直观

(2) 使 0 表示最小阶码, 浮点数 0 表示 0, 使 255 表示最大阶码, 表示 $\pm \infty$

(3) 便于比较阶码大小(无符运算法则)

(4) 除法运算, 阶码相减后若为负数则直接判断下溢

(5) 使整个浮点数只有一个符号位, 即尾数的符号位

② 尾数加减 \Rightarrow 尾数用原码表示, 转为补码实现减法

原码表示尾数的原因

(1) 数据区间对称(补码规格化数的不对称区间)

(2) 便于实现乘除法运算

(3) 便于规格化

③ 规格化 \Rightarrow 使尾数成为 1.XXXX 格式, 其中左侧的 "1" 隐藏

④ 溢出判断

最小规格化阶码为 -126 (移码为 11, 左归则下溢为非规格化数)

最大规格化阶码为 127 (移码 254), 左归则上溢为正负无穷

⑤ 舍入 \Rightarrow 截去、就近偶数舍入, 向正∞舍入, 向负∞舍入; 舍入后可能需要再规格化

Guard 保护位: 最低位左边的位

Round 舍入位: 在保护位右边的位

Sticky 粘接力: 全位左边所有进位算成, 用于居中时辅助判断

浮点数乘法

$$X = 2^m M_x, Y = 2^n M_y$$

$$\therefore X \cdot Y = 2^{m+n} M_x \cdot M_y$$

① 阶码相加

阶码相加可能产生溢出, 要进行溢出判断, 如溢出计算机要进行处理

② 尾数相乘

尾数相乘可得积的尾数, 可按定点乘法运算方法运算

③ 结果规格化

可按浮点加减法运算规格化方式处理, 舍入方式也相同

浮点数除法

$$X = 2^m M_x, Y = 2^n M_y$$

$$\therefore X \div Y = 2^{m-n} M_x \div M_y$$

① 尾数调整

如被除数尾数大于除数尾数(绝对值), 则将被除数尾数右移一位, 阶码+1

② 阶码求差

商的阶码等于被除数的阶码减去除数的阶码

③ 尾数相除

从被除数的尾数除以除数的尾数以获得商的尾数, 尾数相除与定点除法运算相同

本运算器组织见PPT