计算机组成原理

PRINCIPLES OF COMPUTER ORGANIZATION

第4次课：3.2带符号的二进制数据的表示方法（下）

杜国栋

信息科学与工程学院计算机科学与工程系

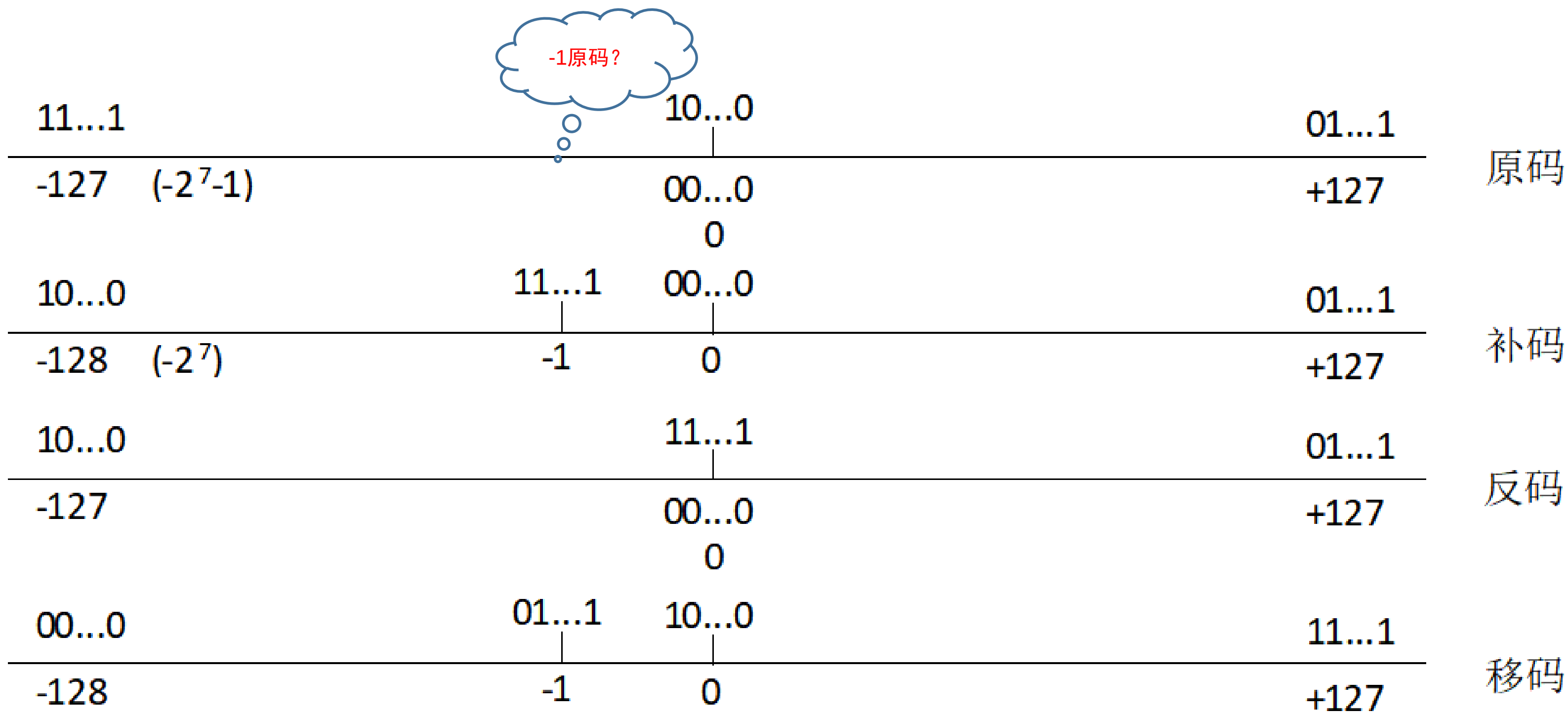
gddu@ysu.edu.cn

厚 德 · 博 学 · 求 是

燕山大學

YANSHAN UNIVERSITY

定点整数的原码、补码、反码和移码的表示范围（8位二进制数为例）



定点整数的原码、补码、反码和移码的表示范围（8位二进制数为例）

n=8则:

①原码:
- 正数: 0000 0000（+0）～ 0111 1111（+127） 共128个
- 负数: 1000 0000（-0）～ 1111 1111（-127） 共128个

共256个

∵ 一个字节8位(即n=8)　　∴ 共能表示256个数

即：-127~127

②反码:
- 正数: 0000 0000（+0）～ 0111 1111（+127） 共128个
- 负数: 1111 1111（-0）～ 1000 0000（-127） 共128个

共256个

∵ 一个字节8位(即n=8)　　∴ 共能表示256个数

即：-127~127

定点整数的原码、补码、反码和移码的表示范围（8位二进制数为例）

③补码:
- 正数: 0000 0000（+0） … 0111 1111（+127） 共128个
- 负数: 1111 1111（-1） … 1000 0001（-127） 共127个

共255个

∵ 一个字节8位(即n=8)，共能表示256个数

∴规定多出1000 0000表示-128

移码
- 正数: 1000 0000（+0） … 1111 1111（+127） 共128个
- 负数: 0111 1111（-1） … 0000 0001（-127） 共127个

共255个

∵ 一个字节8位(即n=8)，共能表示256个数

∴规定多出 0000 0000 表示-128

若小数点约定在8位二进制数的最右端（整数），试分别写出下列各种情况下的W、X、Y和Z的真值。

（1）$[W]_{补}=[X]_{原}=[Y]_{反}=[Z]_{移}=00H$

（2）$[W]_{补}=[X]_{原}=[Y]_{反}=[Z]_{移}=80H$

（3）$[W]_{补}=[X]_{原}=[Y]_{反}=[Z]_{移}=FFH$

若小数点约定在8位二进制数的最右端（整数），试分别写出下列各种情况下的W、X、Y和Z的真值。

（1）[W]补=[X]原=[Y]反=[Z]移=00H

（2）[W]补=[X]原=[Y]反=[Z]移=80H

（3）[W]补=[X]原=[Y]反=[Z]移=FFH

（1）X=W=Y=0 Z=-128

（2）X=-0 Y=-127 W=-128 Z=0

（3）X=-127 Y=-0 W=-1 Z=127

课程目标

- 掌握浮点数的表示方法；
- 熟悉IEEE 754浮点数标准格式；
- 了解计算机中数据的数值范围和精度。

浮点表示法

- 如何表示定点纯小数和定点整数之外的数值?
- 如何表示远超出定点纯小数和定点整数之外范围的数值?

例如:

①$(3.3125)_{10}=(11.0101)_2$

②天体计算或原子计算需要数据具有很大范围（跨度）

如何用有限位数的机器数表示更大范围的数？？

浮点表示法

$N = S \times r^{j}$　　浮点数的一般形式

S 尾数　j 阶码　r 尾数的基值

计算机中 r 取 2、4、8、16 等

当 $r = 2$　　$N = 11.0101$

二进制表示

$\checkmark = 0.110101 \times 2^{\boxed{10}}$ 规格化数

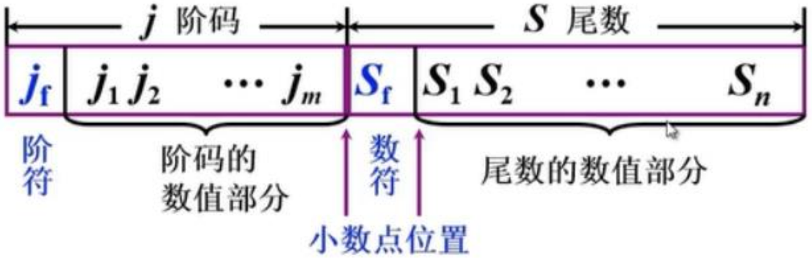
$= 1.10101 \times 2^{1}$

$= 1101.01 \times 2^{-10}$

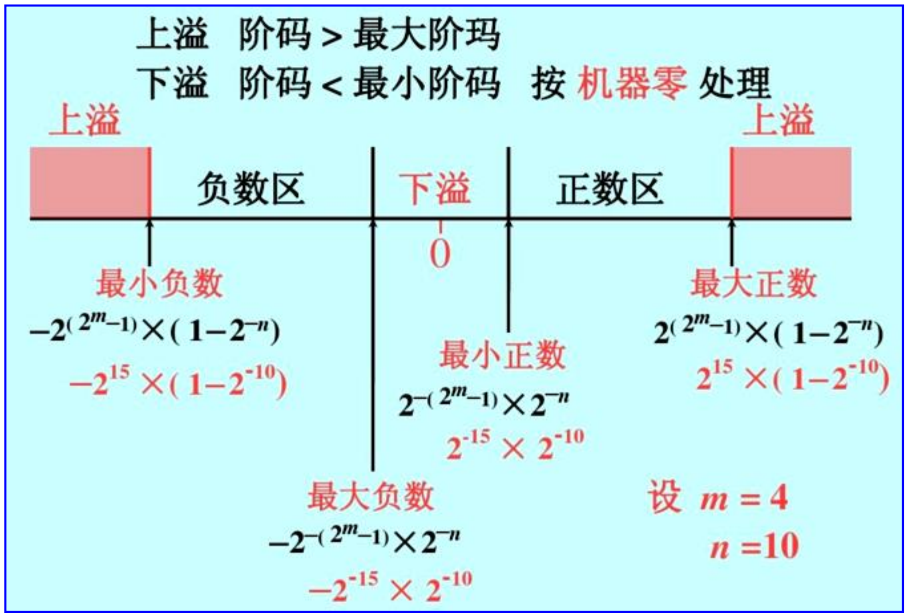
$\checkmark = 0.00110101 \times 2^{100}$

计算机中　S　小数、可正可负

j　整数、可正可负

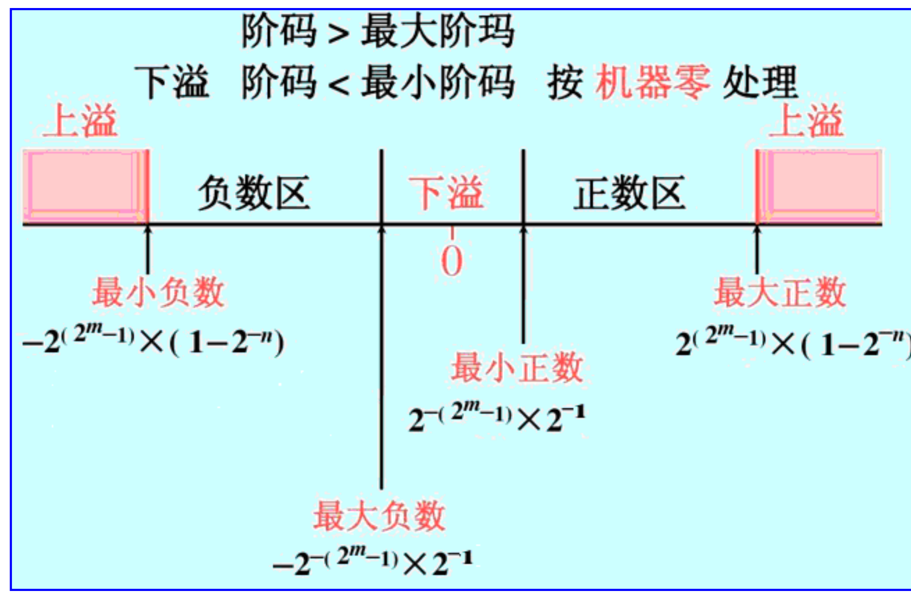


浮点数的范围



← 未规格化

规格化→



浮点数的规格化

±0.1xxxxx...xxxx

00.0xxxxx...xxxx
11.1xxxxx...xxxx

①左规

00.1xxxxx...xxxx
11.0xxxxx...xxxx

10.xxxxx...xxxx
01.xxxxx...xxxx

②右规

11.0xxxxx...xxxx
00.1xxxxx...xxxx

规格化的浮点数的尾数取值范围：$\frac{1}{2} < |M| < 1$；

①、当尾数小于此范围时（在 2^{-n}~1/2 之间），表示非规格化数，需要左规，即小数点右移，阶码减 1；

②、当尾数大于此范围时（>1，即溢出），表示溢出需要右规，即小数点左移，阶码加 1；

尾数边界值讨论（运算器根据补码符号位与有效位最高位是否不同来判断尾数是否为规格化数）：

（1）尾数为+1/2：补码为00.100000 ……，规格化数；

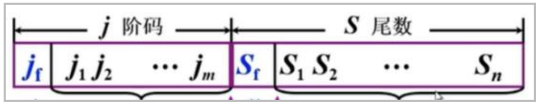
（2）尾数为-1/2：补码为11.100000……，非规格化数；

（3）尾数为+1：取不到，无补码，无效尾数；

（4）尾数为-1：补码为11.000000……，规格化数；

IEEE754浮点数标准格式

$$x = M_x \cdot 2^{Ex}$$



32 位和 64 位的 IEEE754 浮点数格式如下:

S(1 位)	E（8/11 位）	M(23/52 位)

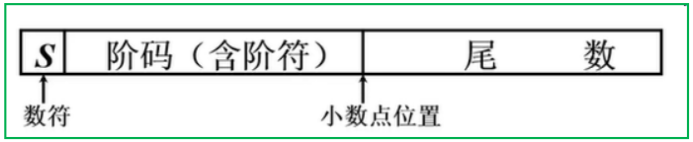
S 为浮点数的符号位，正 0，负 1;

M 为尾数小数值部分，且 IEEE754 的尾数标准形式为 1.xxxxxx，所以 1.是隐藏值，故 23 位尾数可表示 24 位的值;

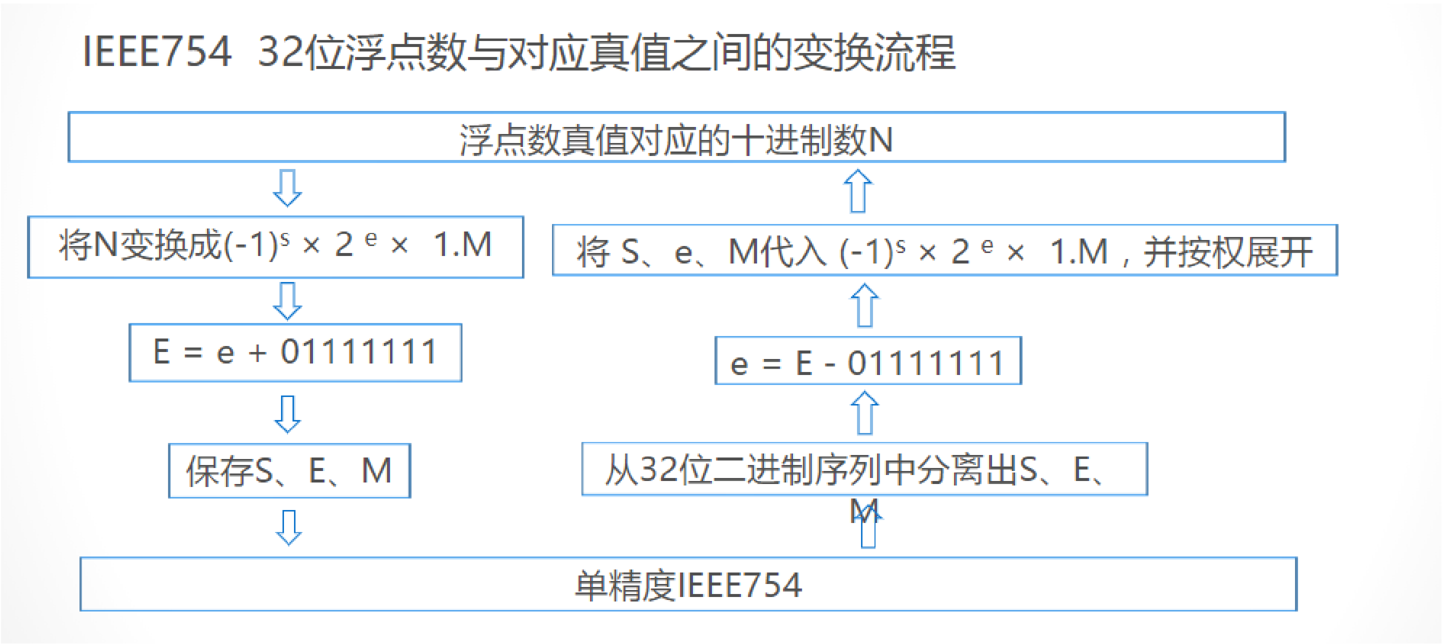
E 为阶码，包含 1 位符号位和 7 位阶码的数值部分。

单精度（32 位）浮点数 IEEE754 的真值:

$$x = (-1)^s \times (1.M) \times 2^{E-127}$$



IEEE754浮点数标准格式



若浮点数x的IEEE754标准存储格式为$(41360000)_{16}$，则其浮点数的十进制数值为$(\quad)_{10}$

例 1：　若浮点数 x 的 IEEE754 标准存储格式为$(41360000)_{16}$，求其浮点数的十进制数值？

解：二进制格式为：

0 100 0001 0011 0110 0000 0000 0000 0000

↑　↑　↑

S　E　M

指数 e=E-127=10000010 - 01111111=3

尾数 1.M=1.011 0110 0000 0000 0000 0000=1.011011

$x=(-1)^0\times(1.011011)\times2^{3}=(1.011011)\times2^3=1011.011=(11.375)_{10}$

将数$(20.59375)_{10}$转换成754标准的32位浮点数的二进制存储格式，则转换后的结果为(　　$)_{16}$

例 2：将数$(20.59375)_{10}$转换成 754 标准的 32 位浮点数的二进制存储格式。

解：首先分别将整数和分数部分转换成二进制数：

$(20.59375)_{10}=(10100.10011)_2$

然后移动小数点，使其在第 1，2 位之间

$10100.10011=1.010010011\times 2^4$

e=4 于是得到：

S=0, E=4+127=131, M=010010011

最后得到 32 位浮点数的二进制存储格式为：

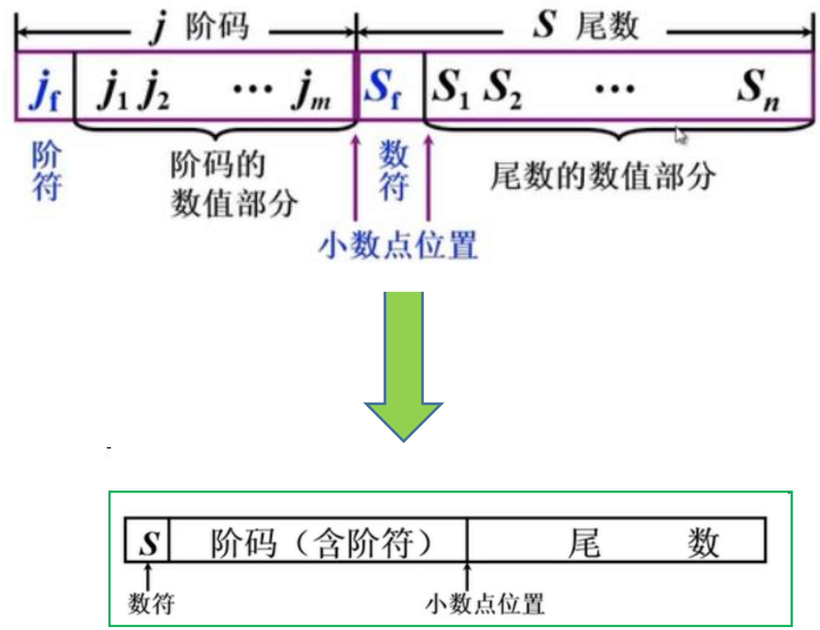
$01000001101001001100000000000000=(41A4C000)_{16}$

计算机中 S 小数、可正可负

j 整数、可正可负

总结

- 浮点表示法
- 浮点数的规格化
- IEEE754浮点数标准格式
- 反码表示法
- 移码表示法



$N = S \times r^j$ 浮点数的一般形式

S 尾数 j 阶码 r 尾数的基值

课后习题：P67 3.8 3.9 3.10 3.11

有问题欢迎随时跟我讨论

办公地点：西校区信息馆423
邮　　箱：gddu@ysu.edu.cn