浮点数的编码表示

阶码一般用**移码**表示，

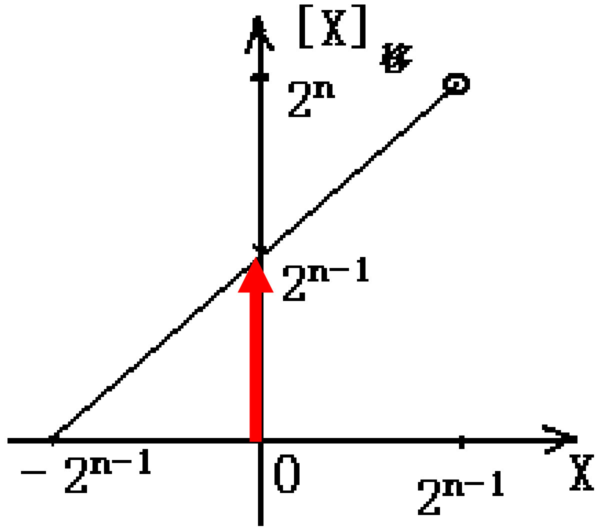
便于比较大小——通过机器数比较真值。

n-1	n-2	....	2	1	0

阶——移码（增码）

定义：设阶为 x ，阶码 E 位数为 n ，则：

$$[x]_{移}=2^{n-1}+x \quad (-2^{n-1} \le x \le 2^{n-1}-1)$$



X_1 = 0101 0101

$[X_1]_{补}$ = 0101 0101

$[X_1]_{移}$ = 1101 0101

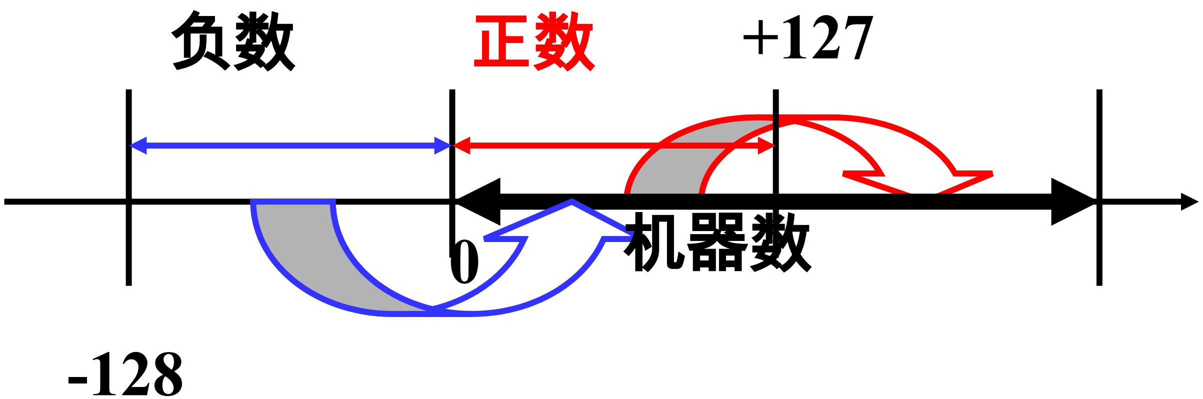
X_2 = -0101 0101

$[X_2]_{补}$ = 1010 1011

$[X_2]_{移}$ = 0010 1011

阶——移码（增码）

8 位机器数，移码表示：真值 → 在数轴上向右平移了 2^7(128) 个位置 → 移码。



用 0~255 表示 -128~127 。

阶——移码（增码）

$$[x]_{移} = 2^{n-1} + x \quad (-2^{n-1} \le x \le 2^{n-1}-1)$$

讨论：

1. 表示范围：$-2^{n-1} \le x \le 2^{n-1}-1$，与补码相同。

   10...0

2. $[-2^{n-1}]_{移} = 00...0$， $[-2^{n-1}]_{补} = ?$

3. 零的移码表示唯一：2^{n-1} = 10...0

   $[+0]_{移} = [-0]_{移} =$

移码

真值 x（十进制）	真值 x（二进制）	$x_{移}$ $(+2^7)$	$x_{补}$ $(+2^8)$
-128	-1000 0000	0000 0000	1000 0000
-127	-0111 1111	0000 0001	1000 0001
:			
-1	-0000 0001	0111 1111	1111 1111
0	0000 0000	1000 0000	0000 0000
+1	0000 0001	1000 0001	0000 0001
:			
+127	0111 1111	1111 1111	0111 1111

4. 若将移码最高位**看成**是符号位，则有‘0’表示负数，‘1’表示正数，与其它 3 种码制相反；
5. 除符号位外，其余各位**与补码相同**。

阶——移码

1. 浮点数做加减运算时需**对阶**，即将阶码调整相同（小数点对齐）；
2. 移码的大小**直观**地反映了真值的大小，便于阶码比较；
3. 可将移码看作**无符号数**，直接按无符号数规则比较大小。

码制表示法小结

1. 若将最高位看作符号位：
   1. 原码，反码，补码：“0”表示正，“1”表示负；
   2. $[X]_{移}$：“1”表示正号，“0”表示负号。
2. 如果X为正数，则 $[X]_{原}=[X]_{反}=[X]_{补}=X$。
3. 0的补码和移码有唯一编码， 0的原码和反码有两种编码。
4. 移码与补码的形式相同，只是符号位相反。

码制表示法小结

数据的四种机器表示法中：

1. **移码**表示法主要用于表示浮点数的**阶码**。
2. **补码**表示对**加减法运算**十分方便，因此目前机器中广泛采用补码表示法。
   1) 一些机器中，数值用补码存储、补码运算。
   2) 有些机器中，数值用原码进行存储和传送，运算时改用补码。
   3) 有些机器在做加减运算时用补码表示，在做乘除运算时用原码表示。

浮点表示

例 1　将十进制数 65798 转换为下述浮点数格式 (32 位)

0 数符	1　　　7 阶码	8　　　31 尾数

0 位：数符 S

1-7 位：7 位阶码 E ，**移码**表示 (偏置常数 = 2^6)

8-31 位：24 位尾数 M ，**原码定点小数**

阶码的底：R=16

解：$(65798)_{10}=(10106)_{16}=(0.101060)_{16}\times 16^5$

数符：S=0 ;

阶码：E= $(2^6+5)_{10}=(100\ 0101)_2$

尾数：M = 0001 0000 0001 0000 0110 0000

浮点数表示为：

0	1000101	000100000001000001100000

0　1　　　7 8　　　31

45101060H

尾数规格化

一个浮点数有不同的表示：

$0.\underline{1011}\times 2^{0}=0.\underline{0101}\boxed{1}\times 2^{1}=0.\underline{0010}\boxed{1}1\times 2^{2}$

规格化的**目的**：

1. 为了充分利用尾数的**有效位数**，提高表示精度；
2. 为了数据表示的**唯一性**。避免浪费编码

规格化的尾数：绝对值大于或等于 1/R，R 为尾数的底。

尾数规格化

非 0 浮点数，尾数规格化后满足**条件**：

|M|1/2(R=2 时)

原码规格化后：正数 0.1×…× ，负数 1.1×…×

补码规格化后：正数 0.1×…× ，负数 1.0×…×

调整为 -1

一般机器规定，若底为 2 并用**补码**表示尾数，则**规格化数的标志**为：

尾数的符号位和数值部分最高位具有不同的代码。

浮点数的规格化处理

非规格化的尾数 $\longrightarrow$ 规格化

通过尾数移位和修改阶码实现。

$0.01011 \times 2^3 = 0.1011 \times 2^2$

如 $R=2$ 时：

左规：尾数左移 1 位，阶码减 1

右规：尾数右移 1 位，阶码加 1

$1.011 \times 2^3 = 0.1011 \times 2^4$

浮点表示

例：设某机器用 32 位表示一个实数，阶码部分 8 位（含 1 位阶符），用定点整数**补码**表示；尾数部分 24 位（含数符 1 位），用规格化定点小数**补码**表示，底为 2。

浮点表示

求 y=-256.5 的浮点表示格式。

$y = -(256.5)_{10} = -(1\ 0000\ 0000.1)_2$

$= -0.1000000001 \times 2^9$

8 位阶码为：$[9]_{补}$=0000 1001

24 位尾数为：$[-0.100\ 0000\ 001]_{补}$

=1.011 1111 1110 0000 0000 0000

-256.5 的浮点表示格式为：

0000 1001 1011 1111 1110 0000 0000 0000

编码的 16 进制表示为 :09BFE000H

浮点数的溢出判断

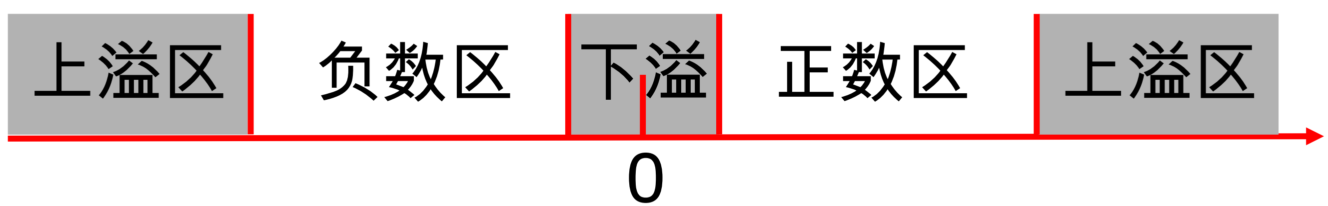
判断规格化后的**阶码**：　$x = M \times R^{E}$

上溢：浮点数阶码大于机器<u>最大阶码</u>

处理方法——**中断**。

下溢：浮点数阶码小于机器<u>最小阶码</u>

处理方法——**零处理**。



隐藏位技术

原码规格化后：正数 0.1×…× ，负数 1.1×…×

数值最高位必定为 1 。

隐藏位技术：

- 在保存浮点数到内存前，通过尾数左移，强行把最高位去掉；
- **用同样多的尾数位可多存一位二进制数**；
- 有利于提高数据表示精度。

说明：在取回浮点数到运算器执行运算时，必须先**恢复**隐藏位。

浮点表示

例 将十进制数 65798 转换为下述典型的 32 位浮点数格式。

0	1　　　　8	9　　　　　　　　31
数符	阶码	尾数

0 位：数符 S

1-8 位：8 位阶码 E, 移码表示 (偏置常数 =128)

9-31 位：23 位尾数 M, 原码定点小数。

规格化尾数的第一位总是 1 ，故不保存。

即虽只有 23 位，但可表示 24 位数据。

阶码的底：R=2 。

浮点表示

0	1　　　　　8	9　　　　　　　　　　31
数符	阶码	尾数

解：

$(65798)_{10}=(10106)_{16}=(1\ 0000\ 0001\ 0000\ 0110)_2$

$= (0.1000\ 0000\ 1000\ 0011)_2 \times 2^{17}$

数符：S=0

阶码：$E=(128+17)_{10}=(1001\ 0001)_2$

尾数：$M=1\underline{000\ 0000\ 1000\ 0011\ 0000\ 0000}$

浮点数表示为：

0	10010001	00000001000000110000000000

0　1　　　　8　9

31

16 进制表示：48808300H

浮点表示

例：若浮点数 x 的二进制存储格式为 $(41360000)_{16}$，求其 32 位浮点数的十进制真值。

解：

0100 0001 0011 0110 0000 0000 0000 0000

数符： 0

阶码： 1000 0010

尾数： 011 0110 0000 0000 0000 0000

指数： e = 阶码 -128=00000010=$(2)_{10}$

包括隐藏位 (1) 的尾数： 0.1011 0110

真值： $x = 0.1011\ 011 \times 2^{2}$

$=10.11011$

$=(2.84375)_{10}$

0	1 … 8	9 … 31
数符	阶码	尾数

浮点表示

同样字长的定点、浮点表示，浮点数的表示范围和精度都要高得多。

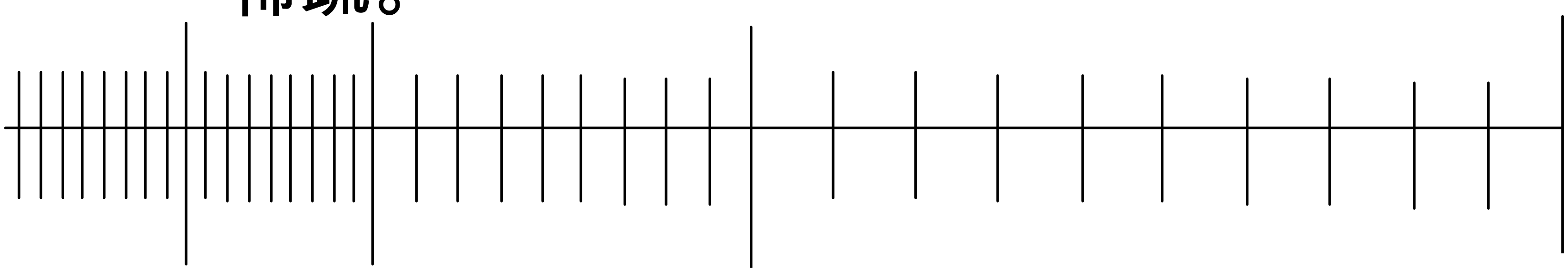
说明：表示范围：取决于阶码位数。

精度：取决于尾数位数。

定点表示与浮点表示

讨论：

1. 相同字长（如32位）的定点数与浮点数（规格化）能表示数的**个数**是相同的；
2. 定点数分布是等距且紧密的，而浮点数分布是不等距且稀疏的，越远离原点越稀疏。



浮点数的密度

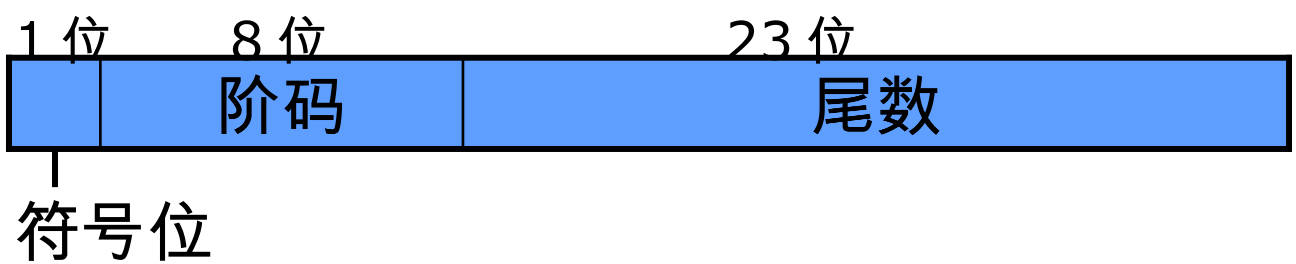
浮点数的标准格式

为了便于软件移植，使用 IEEE(电气和电子工程师协会)标准。

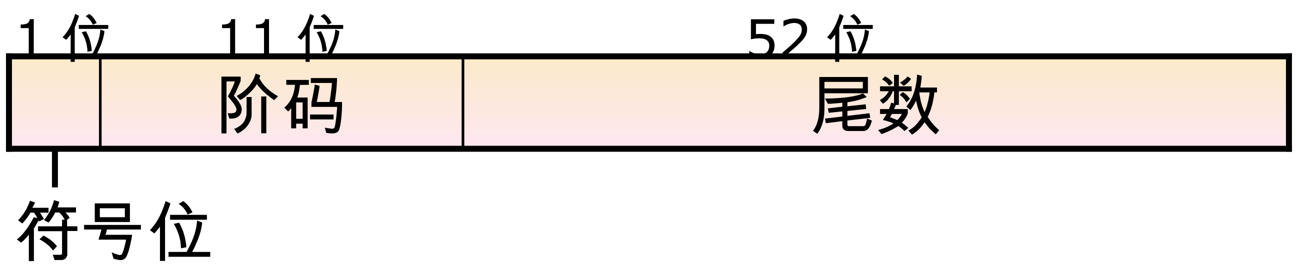
IEEE754 标准：尾数用**原码**；阶码用**移码**；底为 **2**。规格化，隐藏位。

单精度和双精度两种浮点数格式：

1. 单精度格式(32 位)



2. 双精度格式(64 位)



浮点数的标准格式

规格化：个位为 1 ，隐藏个位。

偏置常数： $2^{n-1}-1$

例：将 100.25 转换成短浮点格式。

(1) 将十进制转换为二进制数

$(100.25)_{10} = (1100100.01)_2$

(2) 规格化

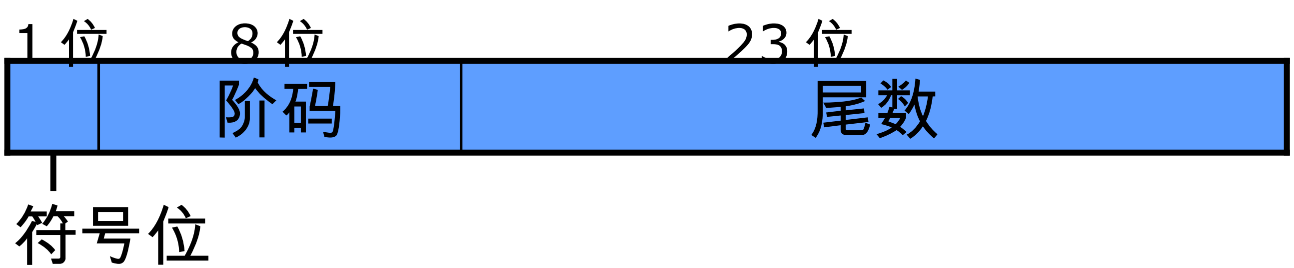
$1100100.01 = 1.100\ 1000\ 1 \times 2^6$

(3) 阶码 = 0000 0110+0111 1111=1000 0101

(4) 浮点编码

0 1000 0101 100 1000 1000 0000 0000 0000

浮点编码： 42C88000H



作业

1. 实现下列各数的转换

   $(101101.011)_2=(\ )_{10}=(\ )_8=(\ )_{16}=(\ )_{8421}$

2. 机器字长 8 位，1 位符号位，求原码、反码、补码。-0.010100

3. 已知原码，求补码、反码

   $[x]_{原}=1.00111$ $[x]_{原}=110100$

6. 已知补码，求真值

   $[x]_{补}=10000000$ $[x]_{补}=11010011$

7. 已知下列字符编码，求 e,f,7,G,Z,5 的 7 位 ASCII 码。

   A=100 0001 a=110 0001 0=011 0000

8. 在第七题的各个编码的高位前，加入奇校验位

浮点作业

1. 有一个字长为 32 位的浮点数，阶码 10 位（包括 1 位阶符），用移码表示；尾数 22 位（包括 1 位尾符）用补码表示，基数 R=2。请写出：

(1) 最大数的二进制表示；

(2) 最小数的二进制表示；

(3) 规格化数所能表示的数的范围；

(4) 最接近于零的正规格化数与负规格化数。

浮点作业

2. 将下列十进制数表示成浮点规格化数，阶码 4 位（包括 1 位阶符），用补码表示；尾数 10 位（包括 1 位尾符），用补码表示，底 R=2。

（1） 27/64

（2） -27/64

浮点作业

3 . 设浮点数的格式为 :

数符	阶码	尾数
1 位	5 位移码	6 位补码

1. 设阶码的底为 4 ，要求用这种格式表示下列十进制数： +19 ， -1/8 ；
2. 写出这种格式所能表示的范围，并与 12 位定点补码整数和定点补码小数的表示范围进行比较。