第一部分 计算机基础

第2章 数据信息的表示方法

2.1 数值型数据的表示方法

计算机中要表示一个数要解决三个问题：

- 数的组合规则——进位计数制
- 小数点位置的确定——数的定点表示和浮点表示
- 符号的选择——带符号数的代码表示

2.1.1 进位计数制

数制的基与权

在任一数制中，其每一数位上允许选用的数码个数，称为该数制的基数。

每一数位所表示位置的值，称为权值。

常用的计数制有十进制、二进制、八进制和十六进制等。

对于任意的r进制来说，数X的表示方法有两种形式：

- 代码序列形式$(X)_r=(a_{n-1}a_{n-2}\ldots\ldots a_1a_0 . a_{-1}a_{-2}\ldots a_{-m})_r$
- 多项式表示形式（按权展开法）

$$(X)_r=(a_{n-1}\times r^{n-1}+\ldots+a_0\times r^0+a_{-1}\times r^{-1}+\ldots+a_{-m}\times r^{-m})$$

$$=\sum_{i=-m}^{n-1} a_i r^i$$

n：整数的位数；　　m：小数的位数；

a_i：$0\leqslant a_i\leqslant r-1$；　　r：进位制的基数。

例：$(246.52)_{10}$　　代码序列形式

多项式表示法:

$$2\times10^{2}+4\times10^{1}+6\times10^{0}+5\times10^{-1}+2\times10^{-2}$$

各位的权值分别是：10^{2}、10^{1}、10^{0}、10^{-1}、10^{-2}

例：$(101.01)_2$

$$=1\times2^{2}+0\times2^{1}+1\times2^{0}+0\times2^{-1}+1\times2^{-2}$$

$$=(5.25)_{10}$$

r进制的计数规则是：逢r进一。

一、常用计数制

1、 十进制

十个数码：0、1、2、3、4、5、6、7、8、9

写法：$(D)_{10}$——Decimal 或 $(D)_{10} = \sum_{i=-m}^{n-1} a_i 10^i$

基数：10； 进位：逢十进一

2、 二进制

有两个数码：0、1

写法：$(B)_2$——Binary 或 $(B)_2 = \sum_{i=-m}^{n-1} a_i 2^i$

基数：2； 进位：逢二进一

3、 八进制

有八个数码：0、1、2、3、4、5、6、7

写法：$(O)_{10}$——Octal 或 $(O)_8=\sum_{i=-m}^{n-1} a_i 8^i$

例如：$(37.41)_8=3\times8^1+7\times8^0+4\times8^{-1}+1\times8^{-2}$

基数：8

进位：逢八进一

4、十六进制

有十六个数码：0、1、2、…、9、A、B、C、D、E、F

写法：$(H)_{16}$——Hexadecimal 或 $(H)_{16}=\sum_{i=-m}^{n-1} a_i(16)^i$

例如：

$$(349)_{16}=3\times16^2+4\times16^1+9\times16^0=(841)_{10}$$

$$(3AB.11)_{16}=3\times16^2+A\times16^1+B\times16^0+1\times16^{-1}+1\times16^{-2}$$

$$\approx(939.0664)_{10}$$

基数：16

进位：逢十六进一

二、常用计数制之间的转换

1、非十进制数转换为十进制数

按权相加法：

即将对应的进位制数展开成多项式求和的形式，按照十进制的规则求出各项的数值，相加后的结果即为十进制数。

例 $(101.101)_2=1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}$

$=4+0+1+1/2+0+1/8=(5.625)_{10}$

例 $(147)_8=1\times8^2+4\times8^1+7\times8^0=(103)_{10}$

2、十进制整数转换为非十进制数

除基取余法：

$$\because \quad (D)_{10} = (a_{n-1} \times r^{n-1} + \cdots + a_0 \times r^0) = (N)_r$$

$$\therefore \quad D/r = (a_{n-1} \times r^{n-2} + \cdots + a_1 \times r^0)\text{，余数为}a_0\text{。}$$

$$\vdots$$

$$(N)_r = (a_{n-1} \cdots a_1 a_0)_r$$

例如： $(215)_{10}=(\ ?\ )_2$

2	215		
2	107	余1	最低位
2	53	余1	
2	26	余1	
2	13	余0	
2	6	余1	
2	3	余0	
2	1	余1	
	0	余1	最高位

$\therefore\ (215)_{10}=(11010111)_2$

3、十进制纯小数转换为非十进制数

乘基取整法:

$$\because\ (D)_{10} = (a_{-1}\times r^{-1}+\ldots+a_{-m}\times r^{-m}) = (X)_r$$

$$\therefore\ Dr = (a_{-1}\times r^{-1}+\ldots+a_{-m}\times r^{-m})r$$

$$= a_{-1}+(a_{-2}\times r^{-1}+\ldots+a_{-m}\times r^{-m+1})\text{，取出整数}a_{-1}$$

$$D' = (a_{-2}\times r^{-1}+\ldots+a_{-m}\times r^{-m+1})$$

$$\vdots$$

$$(X)_r = (0.a_{-1}a_{-2}\ldots a_{-m})_r$$

0.6875			0.6531	
× 2		最高位	× 2	
1.3750	取整数1	↓	1.3062	取整数1
× 2			× 2	
0.750	取整数0		0.6124	取整数0
× 2			× 2	
1.50	取整数1		1.2248	取整数1
× 2			× 2	
1.0	取整数1		0.4496	取整数0
		最低位	× 2	
			0.8992	取整数0
			× 2	
			1.7984	取整数1
			……	

$\therefore (0.6875)_{10}=(0.1011)_2 \qquad (0.6531)_{10}\approx(0.101001)_2$

说明：

（1）有些十进制的小数，不能用有限位的二进制小数表示时，可根据需要，表示到一定位数。

（2）对于具有小数和整数两个部分的十进制数，可以把整数和小数分别换算成二进制数的表示形式，然后相加起来即可。

例：$(215.6531)_{10} \approx (11010111.101001)_2$

例如： $(75.5)_{10}=(113.4)_8$

8	75		0.5	
8	9	余3	×8	
8	1	余1	4.0	取4
	0	余1		

4、二——八转换

将二进制数的整数部分由小数点向左，每三位分成一组。最后不足三位的，前面补零。小数部分的由小数点向右，每三位分为一组。最后不足三位的，后面补零。然后，把每三位二进制数，用对应的八进制数码代替即可。

000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

例：$(010\ 110\ 101.\ 001\ 111\ 010)_2=(\ 265.172\ )_8$

$\ \ 2\quad 6\quad 5\quad\ 1\quad 7\quad 2$

5、二——十六转换

与二——八转换相仿。但要四位分为一组

0000	0		1000	8
0001	1		1001	9
0010	2		1010	A
0011	3		1011	B
0100	4		1100	C
0101	5		1101	D
0110	6		1110	E
0111	7		1111	F

例：$(0101\ 1110.\ 1011\ 0010)_2=(\quad 5E.B2 \quad)_{16}$

5 E B 2

6、八——二转换和十六——二转换

与二——八转换和二——十六转换相反。

例：

$(\ 5 \quad 1 \quad 2. \quad 3 \quad 0 \quad 4\ )_8$

101 001 010 011 000 100

$=(\ 101001010.0110001\ )_2$

例：

$(\ 8 \quad F \quad A\ . \quad C \quad 6\ )_{16}$

1000 1111 1010 1100 0110

$=(\ 100011111010.1100011\ )_2$

2.1.2 带符号数的表示

1. 数的符号表示法

真值： ±“数值绝对值”；

机器数： “0”表示正号“+”；
“1”表示负号“-”。 → 约定为最高位

0	1	0	0	1	0	1	0

符号位 有效数值位

1	1	0	0	1	0	1	0

符号位 有效数值位

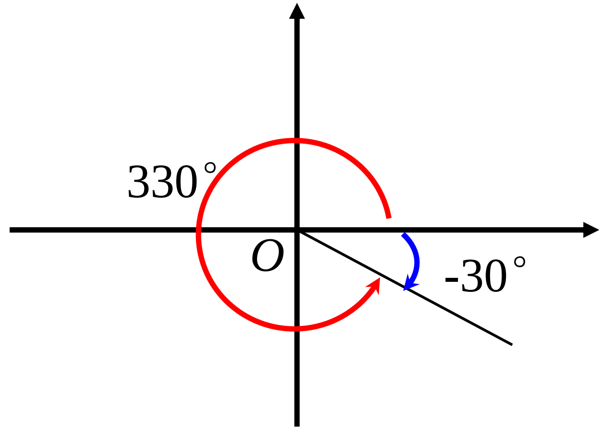
有效数值都为真值的绝对值时，则为原码表示形式

2. 原码和补码表示法

(1) **原码**： 最高位为符号位，其余为**有效数值位**，用**二进制真值的绝对值**来表示。

真值	+1001010	-1001010
原码	<u>0</u>1001010	<u>1</u>1001010

补码的定义



真值X对确定模M的补数称为该数的补码，即，

$$X_{补}=M+X \quad (\mathrm{mod}\ M)。$$

模是指一个计量器的容量，即二进制数最高位(即符号位)产生进位后，该进位位所代表的权值称为模。

如n位二进制整数，其最高位进位后该位的权值为2^n，则$M=2^n$；当纯小数时$M=2$。

(2) 补码： 最高位为符号位，对于正数，有效数值部分为二进制真值的绝对值；对于负数，有效数值部分是将真值的绝对值按位取反，且末位加1。

真值	+1001010	-1001010
		10110101
		+ 1
补码	01001010	10110110

（3）原码、补码之间的转换

当 $X \geq 0$ 时，$X_{原} = X_{补} = X$

1 原码→补码

当 $X < 0$ 时，**原码的符号位1不变，其余各位先变反，然后末位加1。**

2 补码→原码

当 $X < 0$ 时，**补码的符号位1不变，其余各位先变反，然后末位加1。**

【例】已知$[X]_{原}$= 10011010，求$[X]_{补}$

解：$[X]_{原}$= 1 0 0 1 1 0 1 0

$$
\begin{array}{rcccccccc}
[X]_{原}= & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
 & & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
+) & & & & & & & & 1 \\
\hline
[X]_{补}= & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0
\end{array}
$$

$[X]_{原}$= 10011010

$[X]_{补}$= 11100110

3. 反码表示法

最高位为符号位，对于正数，有效数值部分为二进制真值的绝对值；对于负数，有效数值部分是将真值的绝对值按位取反。

真值	+1001010	-1001010
反码	01001010	10110101

2.1.3 定点数与浮点数

1、定点表示法

程序中所有数的小数点固定在同一位置不变。

① 带符号的定点小数：约定所有数的小数点的位置固定在符号位之后（小数点隐含约定）。

1	. 0 1 0 1 1 1 0

符号位　小数点　数值部分

设字长= n+1位，表示范围为,

原码：$-(1-2^{-n}) \sim (1-2^{-n})$ ，补码：$-1 \sim (1-2^{-n})$

② **带符号的定点整数：**约定所有数的小数点的位置固定在最低数值位之后。

1	0 1 … 0 1 1 1 0 .

n+1位

小数点

设字长=n+1位，表示范围为，

原码： $-(2^n-1) \sim (2^n-1)$

补码： $-2^n \sim (2^n-1)$

③ **无符号定点整数：**约定所有数的小数点的位置固定在最低数值位之后。（不带符号的定点正整数）

若代码序列为$X_nX_{n-1}\ldots X_1X_0$，共$n+1$位，则有：

$$\boxed{\underbrace{0\ \ldots\ 1\ \ 0\ 1\ 1\ 1\ 0}_{n+1\text{位}}\quad .}$$

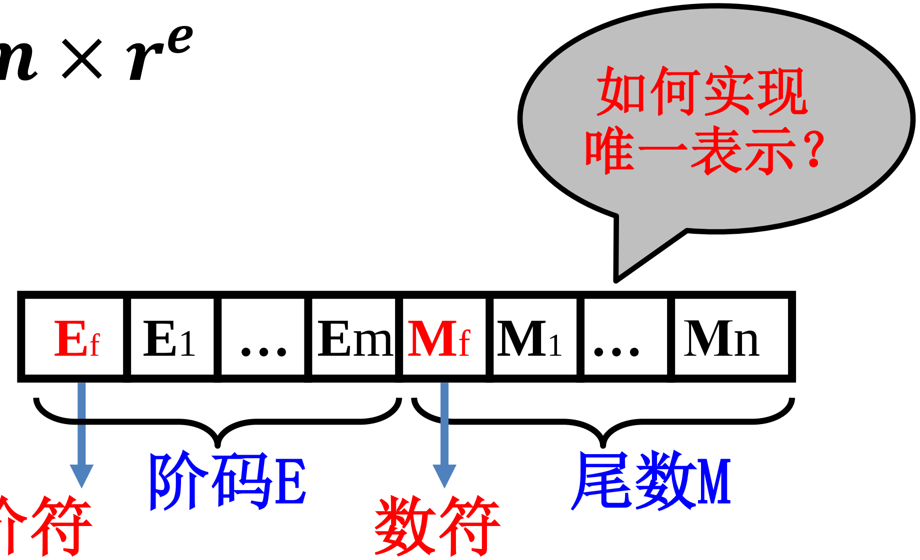
小数点

表示范围为： $0 \sim (2^{n+1} - 1)$

2、浮点表示法(原理性)

浮点数真值：$X = \pm m \times r^{e}$

浮点数机器格式：



$$X_{浮} = R^{E} \times M$$

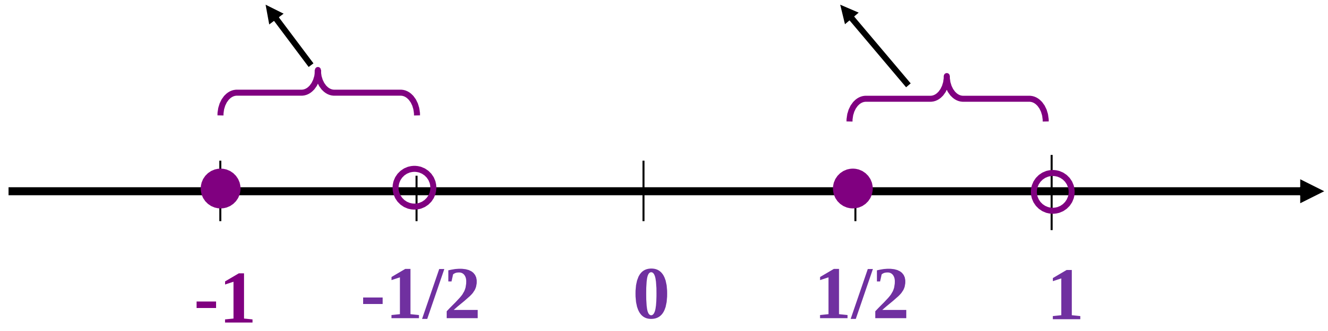
R：阶码底（基数），隐含约定，一般为2。

E：阶码，为定点整数，补码或移码表示。

M：尾数，为定点小数，补码或原码表示。

尾数规格化：$1/2 \le |M| < 1$（用原码表示时）

$-1 \le M < -1/2$ 或 $1/2 \le M < 1$（用补码表示时）



对于原码：规格化的特征是尾数最高有效位为“1”;

对于补码：

对于正数, 规格化的特征是最高有效位为“1”;

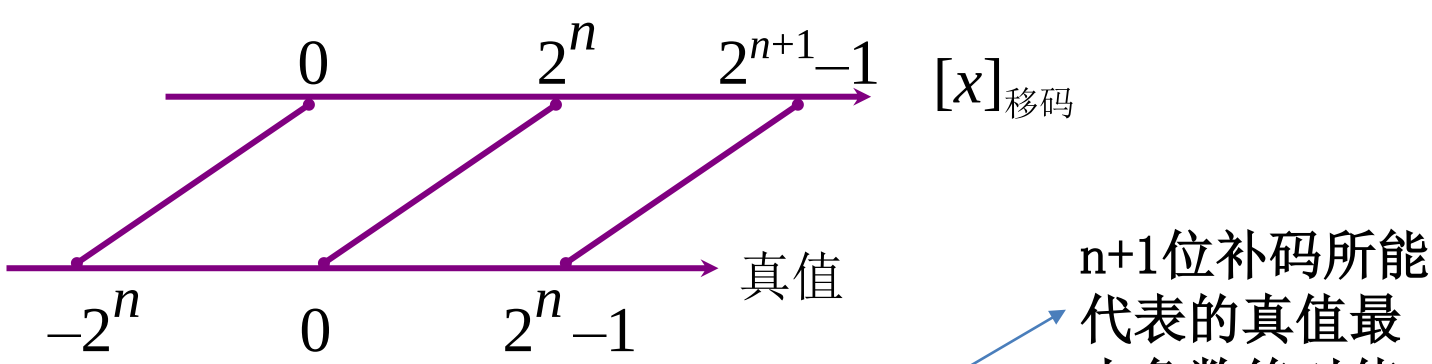
对于负数, 规格化的特征是最高有效位为“0”。

3、移码

在计算机中，移码通常用于表示浮点数的阶码。由于阶码一般取整数，所以移码通常只用于**整数**的表示。

对定点整数，移码的定义是： $[X]_{移}=2^n+X$

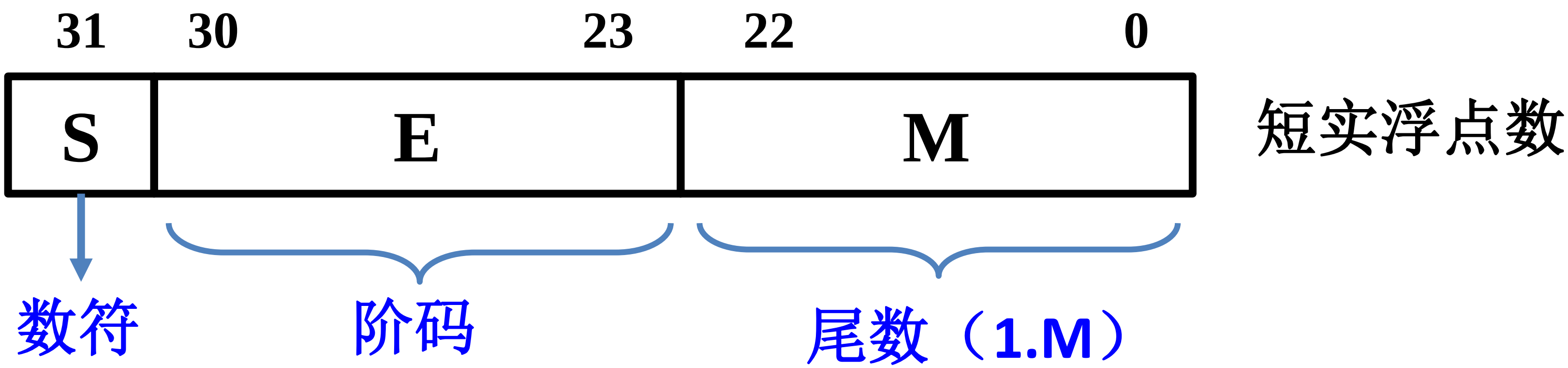
（X为阶码的真值，n+1为阶码的位数）



等价于将X正向平移或者增加2^n，因此称为**移码或增码**；所以移码也可以认为是**无符号整数**。

4、 IEEE754 的标准

若$X = (-1)^s \times (1.\mathrm{M}) \times 2^e$, 则32位浮点数的标准格式为



•S=浮点数的符号位，0表示正数，1表示负数。

•E=阶码，8位，采用**移码**方式来表示正负指数，但只偏移2^7-1=127， E=e+127。

•M=尾数，23位**原码**，用纯小数表示（不包括符号位），隐含约定尾数的最高位为1，即尾数为**1.M**。

[例1]：将十进制数20.59375转换成IEEE754 的32位标准浮点数的二进制格式来存储，并写出其16进制数。

解：首先分别将整数和分数部分转换成二进制数：

- $(20.59375)_{10} = (-1)^S(10100.10011)_2$
- 然后移动小数点，使其满足（1.M）$\times 2^e$ 形式
- $10100.10011 = 1.\underline{010010011} \times 2^4$
- 小数点被左移了4位，于是得到：$e = 4$
- $S = 0$，阶码 $E = 4 + 127 = 131 = (1000,0011)_2$
- $M = 010010011$
- 最后得到32位浮点数的二进制存储格式为：
- 0100 0001 1010 0100 1100 0000 0000 0000 （S E M）
- $= (41A4C000)_{16}$

例2，将$(-18.125)_{10}$转换成短浮点数格式的二进制形式

$$(-18.125)_{10}=(-10010.001)_2=-1.0010001\times 2^4$$

→ S=1

M=0010001 $\underline{0\cdots\cdots\cdots 0}$(连续16个0)

E=127+4=131, 即$(131)_{10}=(10000011)_2$

→ 11000001100100010000000000000000

例3，IEEE754单精度浮点数C0A00000H的十进制值是多少

$(C0A00000)_{16} = (1100,0000,1010,0000,\ 0000,\ 0000,$

$0000,0000)_2$

可得，符号位S是1

阶码位E是10000001→$(10000001)_2=(129)_{10}$

M为：010,0000,0000,0000,0000,0000

因此,由公式$(-1)^S \times 1.M \times 2^{E-127}$得

$(-1.01 \times 2^{129-127})_2 = (-1.25 \times 4)_{10} = (-5)_{10}$

2.2 字符的表示方法(非数值型数据)

ASCII码：128种常用字符，7位，最高位可以设置为奇偶校验位，共8位表示。

0-9：　　30H -39H

A-Z：　　41H -5AH

a-z：　　61H -7AH

常用符号

控制字符