河海大学

第2章 运算方法和运算器

引入：深度学习里最常用的一种数值精度格式

格式	位数	精度	速度 / 显存	用途
FP32	32	高	较慢、占空间	推理、科学计算、训练收敛要求高
FP16	16	中	快、省显存	主流大模型训练
BF16	16	中	快、省显存	现代 AI 训练主流
INT8	8	低	很快、极省显存	模型量化、端侧推理

格式	总位宽	符号位	指数位	尾数位	特点
FP16	16	1	5	10	小数精度高，数值范围小
BF16	16	1	8	7	数值范围大，小数精度略低

第2章教学要求：

- 熟练掌握机器数与真值的区别,几种机器数之间的转换，对补码的透彻理解尤为重要。
- 掌握定点数、浮点数的表示范围、规格化形式等。
- 掌握定点数补码加、减法运算。
- 熟悉运算器的组成及提高运算速度的方法。

预备知识

- 十进制数：R=10， x_i为0,1,2,…, 9

  各位权10^3 , 10^2 , 10^1 ,10^0 , 10^{-1} , 10^{-2} , 10^{-3}

- 二 进 制：R=2， x_i为0,1；

  各位权为2^3 , 2^2 , 2^1 , 2^0 , 2^{-1} , 2^{-2}

常用进位制符号：

B（二进制），O或Q（八进制）

H（十六进制），D（十进制）

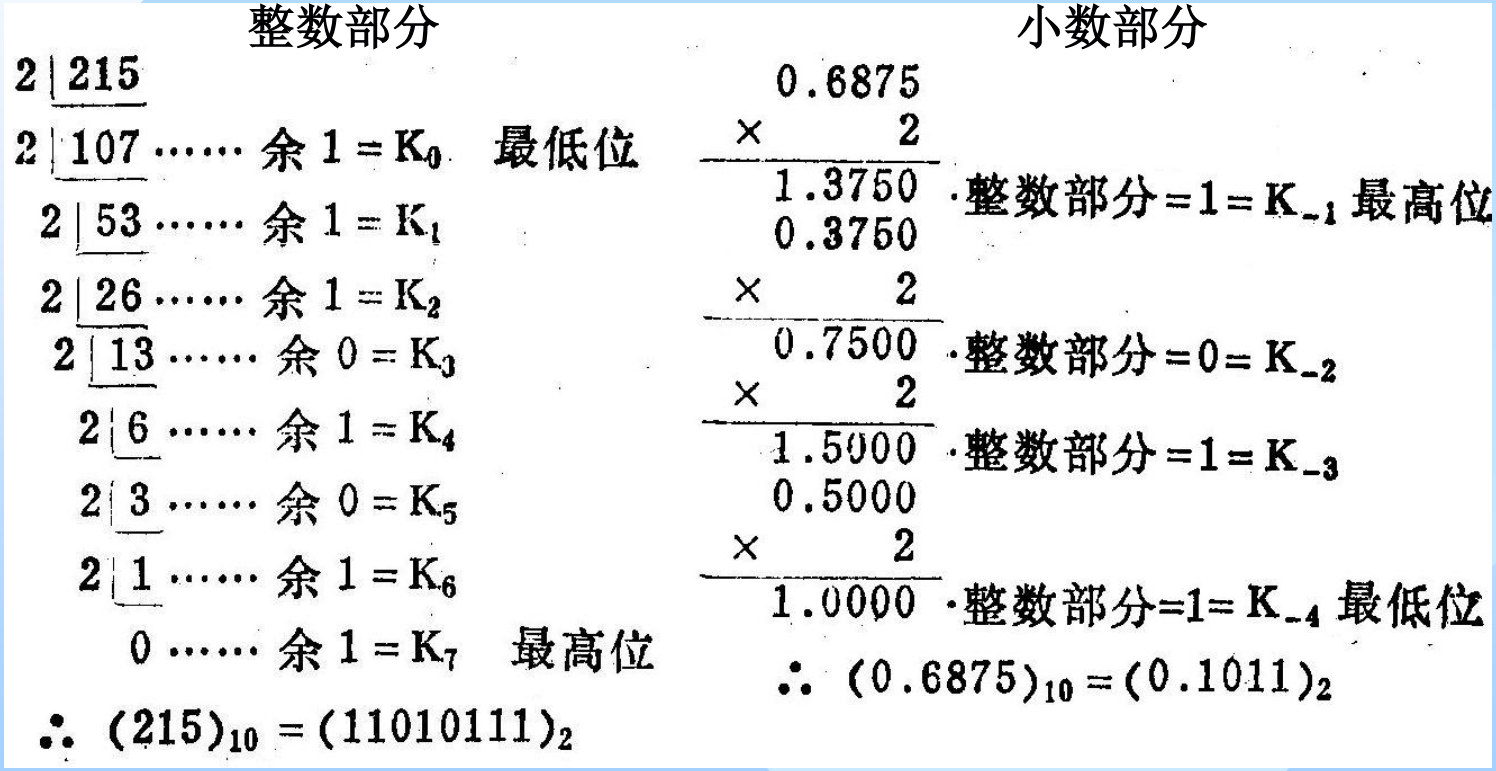
不同进位制的转换

转换原理：

如有两个有理数相等，则两数的整数部分和小数部分一定分别相等。

1．十进制转换为其他进制

例：215.6875D ⟶ ?B



215.6875D=110101111.1011B

2. 任意进制转换为十进制

转换方法：利用任意进制数定义式，将右边展开。

例：$4FCH = 4\times16^2 + 15\times 16^1 + 12\times 16^0$

$= 1024 + 240 + 12 = 1276D$

3. 二进制 ⇄ 十六进制

转换方法：以小数点为界，利用4位二进制数与1位十六进制数的对应关系转换。

例：1011011.100111B ⟶ ?H

0101 1011.1001 1100B ⟶ 5B.9CH {逆转换成立}

4. 二进制 ⇄ 八进制

转换方法：以小数点为界，利用3位二进制数与1位八进制数的对应关系转换。

例： 1011011.100111B ⟶?Q

001 011 011.100 111B ⟶133.47Q

- 下列数中最大的数是（　）。

A. $(10011001)_2$　　$(153)_{10}$

B. $(227)_8$　　$(151)_{10}$

C. $(98)_{16}$　　$(152)_{10}$

D. $(152)_{10}$　　$(152)_{10}$

[答案]A

河海大学

2.1 数据与文字的表示方法

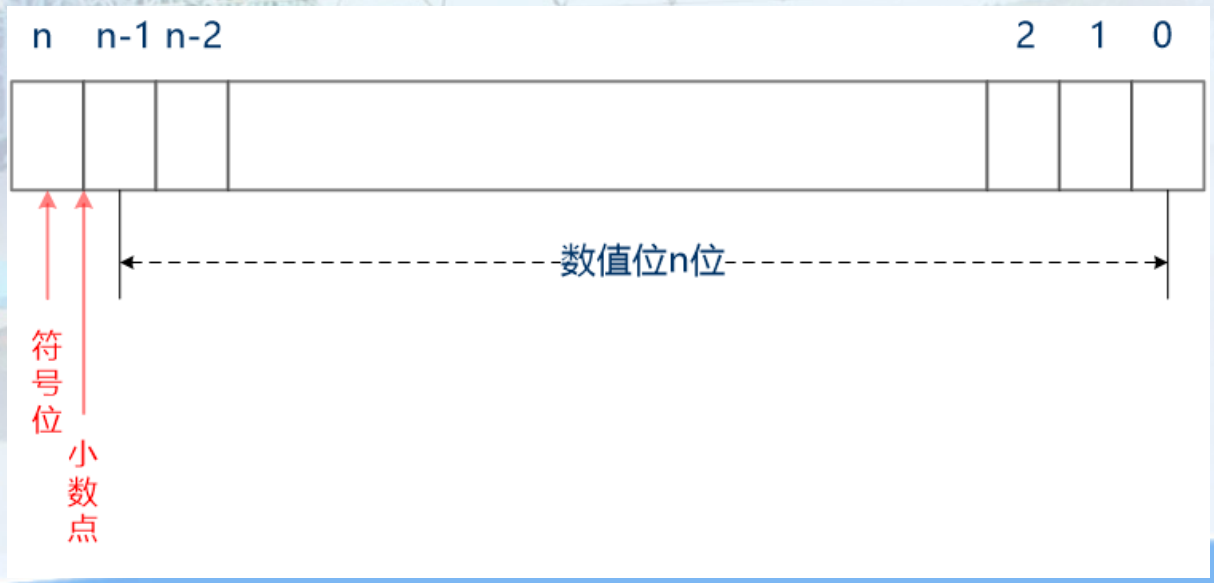
- 数据格式（2.1.1）
- 数的机器码表示（2.1.2）
- 字符与字符串的表示方法（2.1.3）
- 汉字的表示方法（2.1.4）
- 检验码（2.1.5）

2.1.1 数据格式

- 考虑因素
  - 数据类型（小数、整数、实数、复数）
  - 数值范围
  - 数值精度
  - 数据存储和处理所需要的硬件代价
- 常用格式
  - 定点数（纯小数、纯整数、无符号数）
  - 浮点数
  - 十进制数串（BCD码）

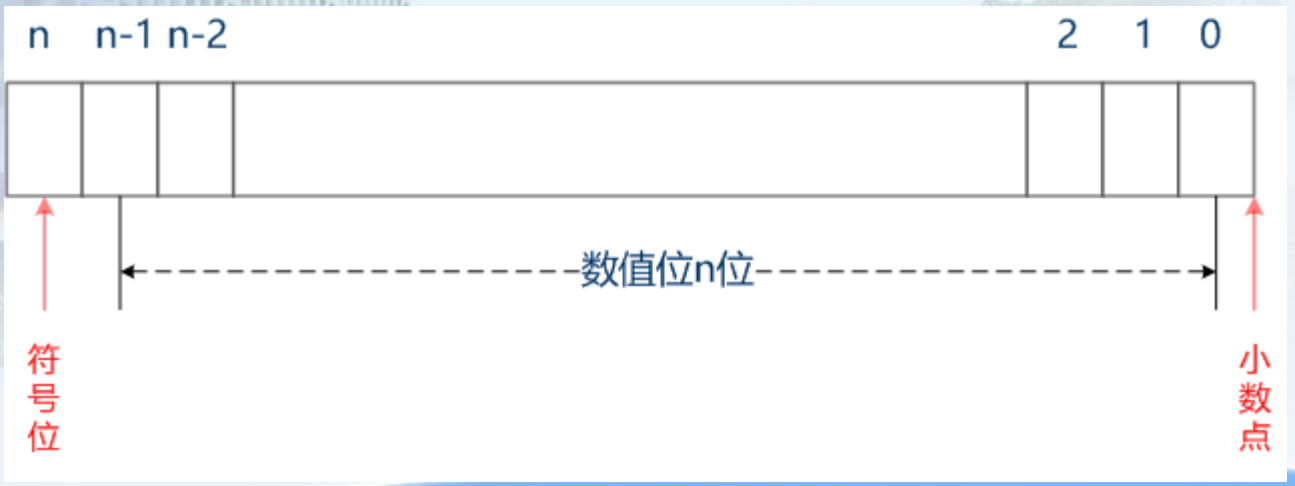
2.1.1 数据格式

- 纯小数
  - 由符号位和数值位（量值位）二部分构成
  - 小数点固定在符号位和最高数值位之间，不真正存储小数点
  - 符号位0表示正、1表示负
  - **±0.M**



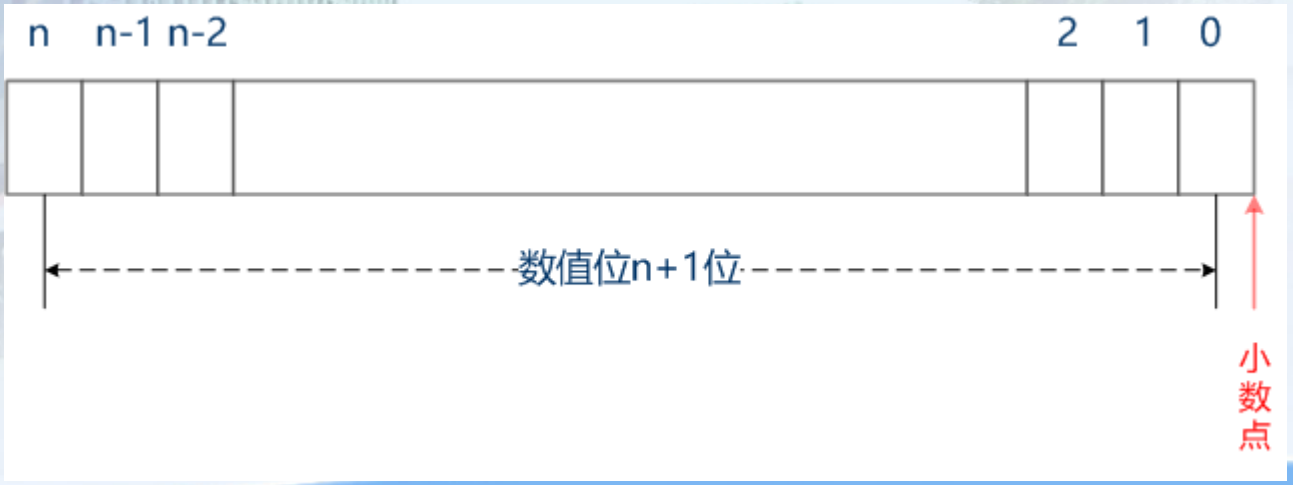
2.1.1 数据格式

- 纯整数
  - 由符号位和数值位（量值位）二部分构成
  - 小数点固定在最低数值位之后，不真正存储小数点
  - 符号位0表示正、1表示负
  - ±M



2.1.1 数据格式

- 无符号数（正整数）
  - 没有符号位，全部是数值位（量值位）
  - 小数点固定在最低数值位之后，不真正存储小数点
  - +M



2.1.1 数据格式

- 浮点数
  - R进制数N= M × R^E：M是尾数、E是指数（阶码）、R为基数
  - 二进制数N = M × 2^E：M是尾数、E是指数（阶码）
  - 存储形式：四个部分，

阶符	阶码	数符	尾数
E_s	$E_{m-1}E_{m-2}\cdots\cdots E_2E_1E_0$	M_s	$M_{n-1}M_{n-2}\cdots\cdots M_2M_1M_0$

【例】假设某机器中，浮点数的尾数用纯小数表示，则

0 00010011 1 1001011101 表示 - 0.1001011101 × $2^{+00010011}$

2.1.1 数据格式

- **字符串形式（非压缩型）：**
  - 一个字节存放一个十进制的数位或符号位
  - 例：1234　“1” “2” “3” “4”
  - 优点：与ASCII码兼容
- **压缩的十进制数串形式：**
  - 一个字节存放两个十进制的数位
  - 每个数值位数占用半个字节：BCD码
  - 符号位占半个字节：用四位编码中的六种冗余值

2.1.1 数据格式

- 十进制数串（BCD码）
  - BCD：Binary-Coded Decimal，即用4位二进制数来表示十进制数中的0~9这10个数码
  - 常用的BCD码
    - 8421码：最基本和最常用的BCD码，它和4位自然二进制码相似，从高到低各位的权值为8、4、2、1

$$(0011\ 0000\ 0001)_{BCD}=(\ 12D\ )_{16}$$

2.1.1 数据格式

- 约定
  - 符号位存放在最低数字位之后
  - 12（c）表示正号，13（d）表示负号
  - 数值位数与符号位数之和必须为偶数
- 例：

+123

1	2	3	C
0001	0010	0011	1100

-12

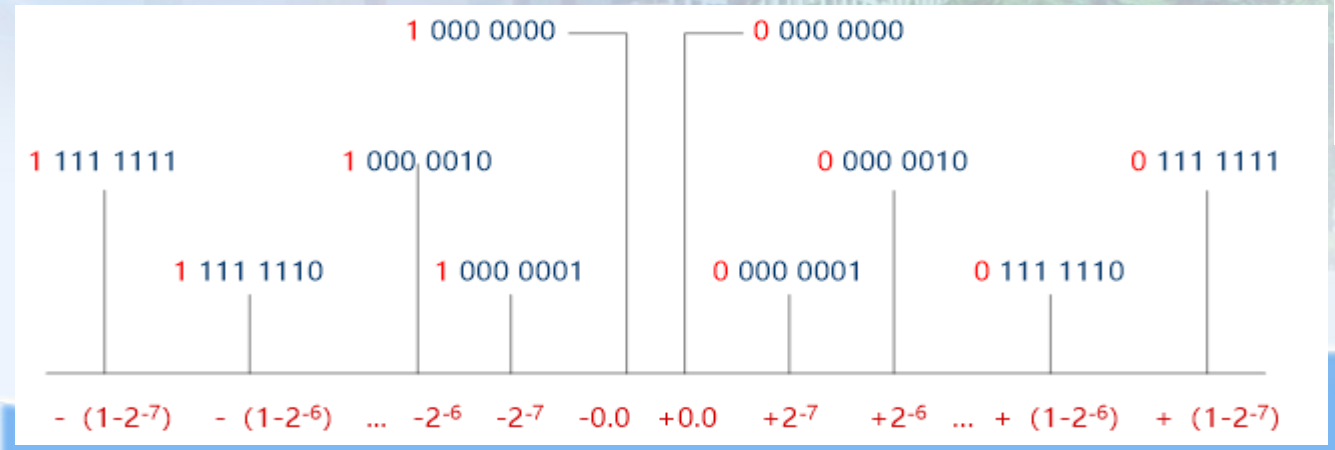
0	1	2	D
0000	0001	0010	1101

既节省存储空间，又便于直接完成十进制数的算术运算

2.1.2 数的机器码表示

- 原码表示法
  - 定点小数：符号位（0正1负）、数值位是二进制数

真值	机器字长=8	机器字长=16
+0.1001	0 100 1000	0 100 1000 0000 0000
-0.1001	1 100 1000	1 100 1000 0000 0000
+0.0	0 000 0000	0 000 0000 0000 0000
-0.0	1 000 0000	1 000 0000 0000 0000



机器字长	表示范围
8	- (1-2^{-7}) ~ + (1-2^{-7})
16	- (1-2^{-15}) ~ + (1-2^{-15})
32	- (1-2^{-31}) ~ + (1-2^{-31})
n	- (1-$2^{-(n-1)}$) ~ + (1-$2^{-(n-1)}$)

原码表示法

- **定点小数**的原码形式为 $x_0\ x_1\ x_2 \cdots x_n$，则原码表示的定义是

$$[x]_{原}=\begin{cases} x & 1>x\geq 0 \\ 1-x=1+|x| & 0\geq x>-1 \end{cases}$$

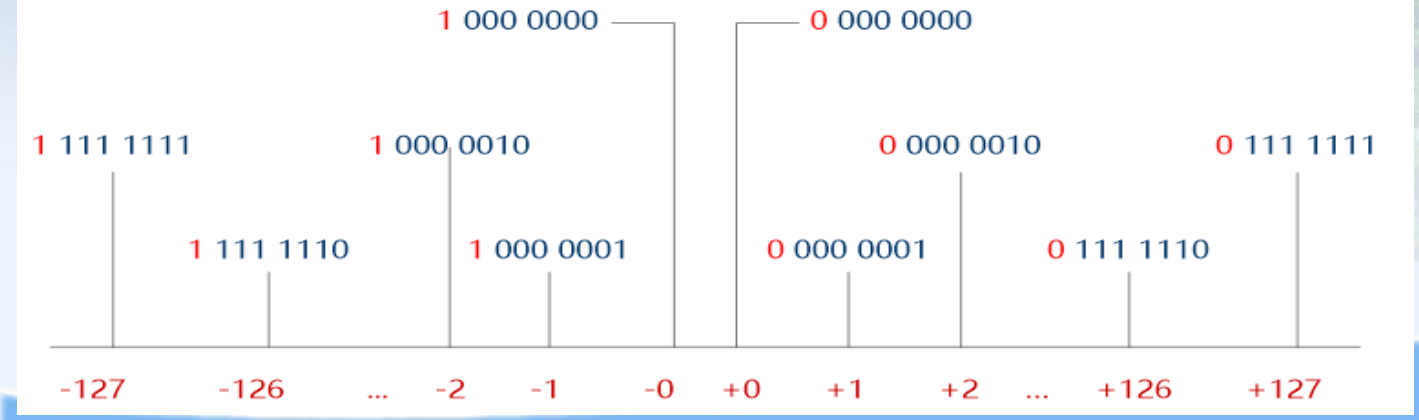
- $[x]_{原}$是机器数，x是真值

2.1.2 数的机器码表示

- 原码表示法
  - 定点整数：符号位（0正1负）、数值位是二进制数

真值	机器字长=8	机器字长=16
+1001	0 000 1001	0 000 0000 0000 1001
-1001	1 000 1001	1 000 0000 0000 1001
+0	0 000 0000	0 000 0000 0000 0000
-0	1 000 0000	1 000 0000 0000 0000

机器字长	表示范围
8	$-(2^7-1)\sim+(2^7-1)$
16	$-(2^{15}-1)\sim+(2^{15}-1)$
32	$-(2^{31}-1)\sim+(2^{31}-1)$
n	$-(2^{n-1}-1)\sim+(2^{n-1}-1)$



原码表示法

- **定点整数**的原码形式为 $x_0 x_1 x_2 \cdots x_n$，则原码表示的定义是

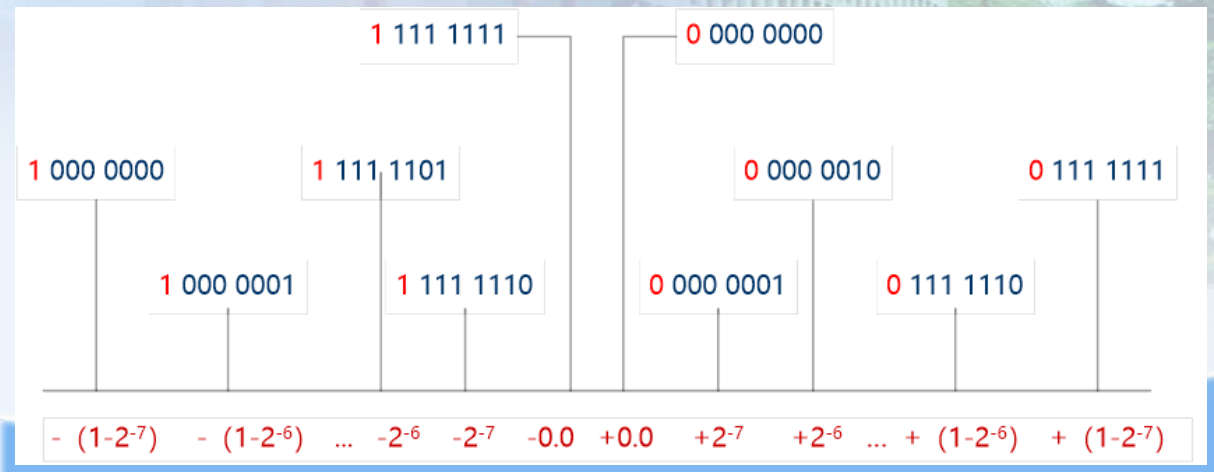
$$[x]_{原}=\begin{cases} x & 2^n > x \geq 0 \\ 2^n - x = 2^n + |x| & 0 \geq x > -2^n \end{cases}$$

2.1.2 数的机器码表示

- 反码表示法
  - 定点小数：符号位0正1负、负数数值位是二进制数按位取反

真值	机器字长=8	机器字长=16
+0.1001	0 100 1000	0 100 1000 0000 0000
-0.1001	1 011 0111	1 011 0111 1111 1111
+0.0	0 000 0000	0 000 0000 0000 0000
-0.0	1 111 1111	1 111 1111 1111 1111

机器字长	表示范围
8	$-(1-2^{-7}) \sim +(1-2^{-7})$
16	$-(1-2^{-15}) \sim +(1-2^{-15})$
32	$-(1-2^{-31}) \sim +(1-2^{-31})$
n	$-(1-2^{-(n-1)}) \sim +(1-2^{-(n-1)})$

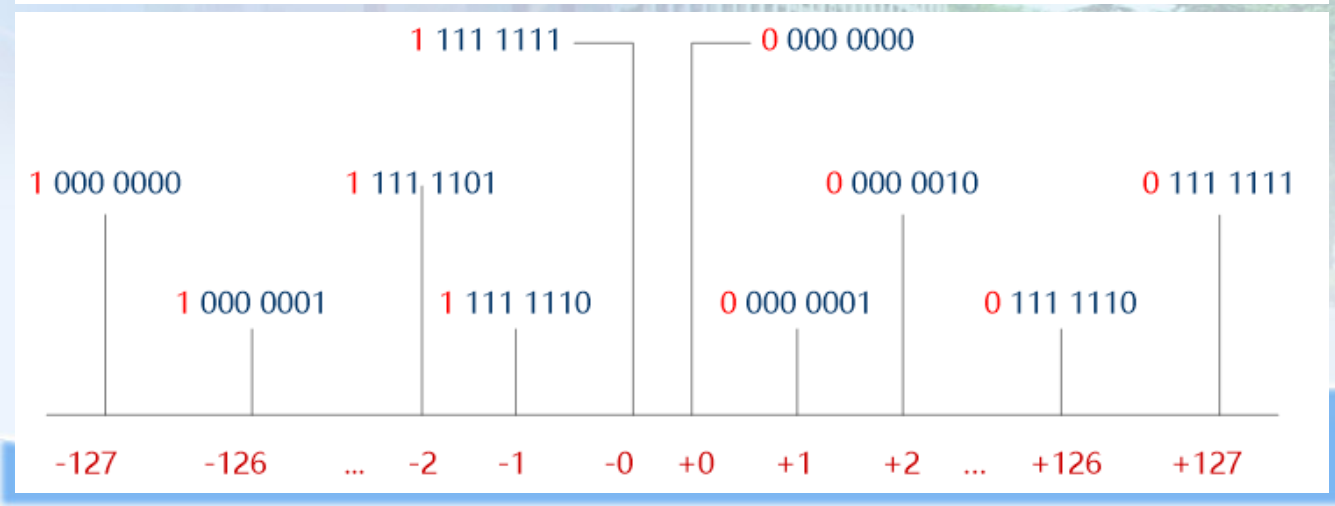


2.1.2 数的机器码表示

- 反码表示法
  - 定点整数：符号位0正1负、负数数值位是二进制数按位取反

真值	机器字长=8	机器字长=16
+1001	0 000 1001	0 000 0000 0000 1001
-1001	1 111 0110	1 111 1111 1111 0110
+0	0 000 0000	0 000 0000 0000 0000
-0	1 111 1111	1 111 1111 1111 0110

机器字长	表示范围
8	$-(2^7-1)\sim+(2^7-1)$
16	$-(2^{15}-1)\sim+(2^{15}-1)$
32	$-(2^{31}-1)\sim+(2^{31}-1)$
n	$-(2^{n-1}-1)\sim+(2^{n-1}-1)$

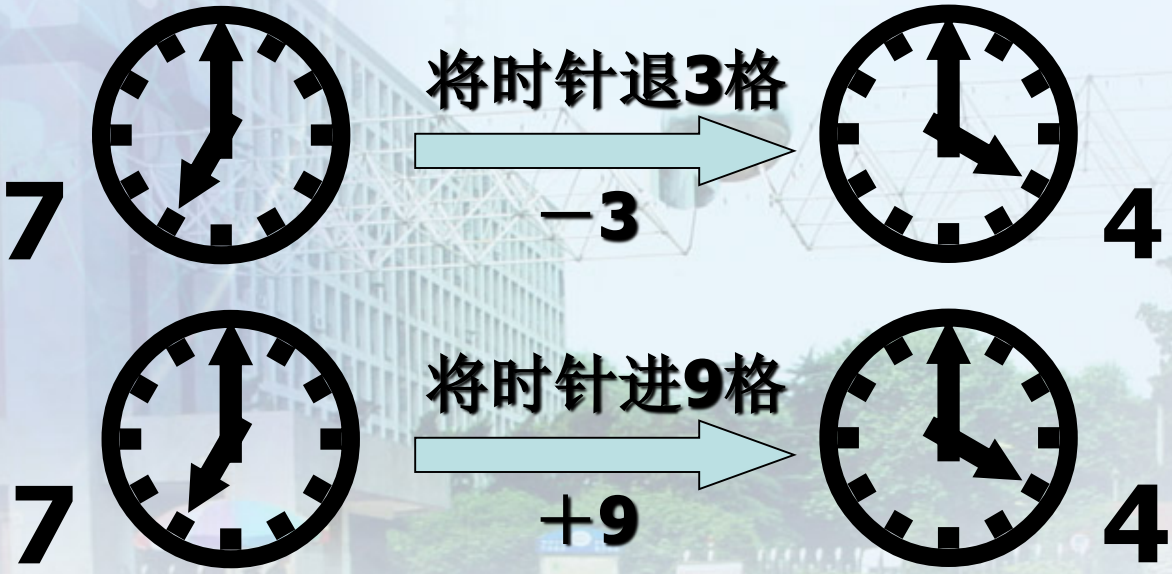


2.1.2 数的机器码表示

- 原码、反码表示法的优缺点
  - 表示法简单、容易理解
  - 零的表示不唯一：占用二个状态，少表示1个数。
  - 运算复杂：如原码两数相加时，如果是同号则数值相加；如果是异号，则要进行减法。而在进行减法时还要比较绝对值的大小，然后大数减去小数，并给结果确定符号。
  - 解决方案：使用补码表示法（零表示唯一、可以运算求得、可以求补电路实现、符号位数值化、减法转换为加法、基本加法单元就是半加器/全加器）。

2.1.2 数的机器码表示

- 补码表示法



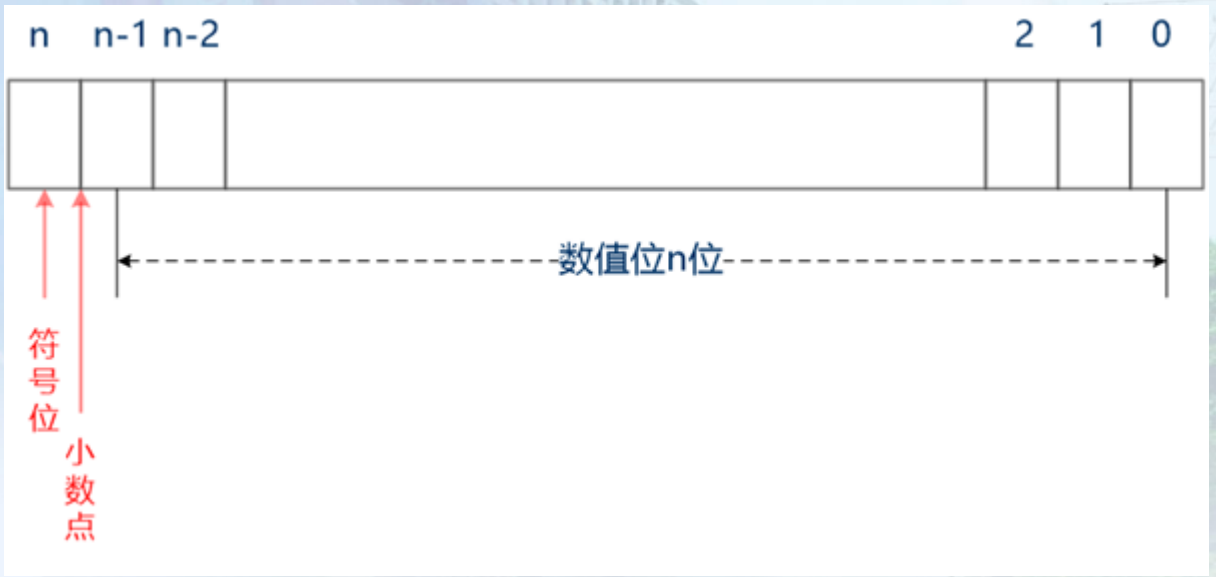
7-3 = 7+9 (mod12)=4

2.1.2 数的机器码表示

- 补码表示法
  - “取模”的启示
    - 计算机中由于机器字长有限，因此运算是有模的
    - 负数用补码表示时，可以把减法转化为加法，在计算机中实现方便

2.1.2 数的机器码表示

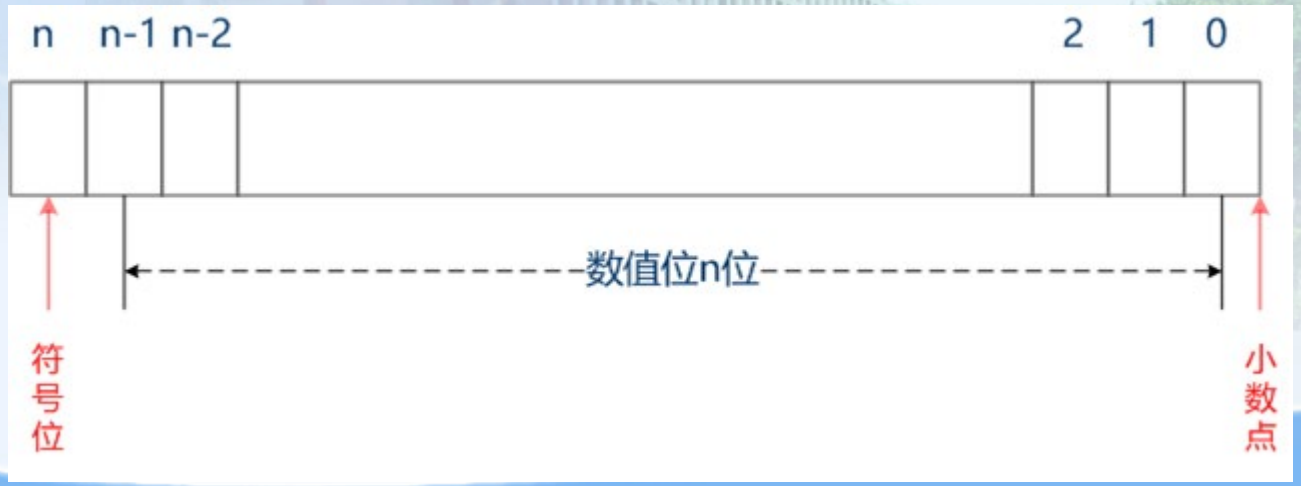
- 补码表示法
  - 定点小数的模：与机器字长无关，始终为2



$$
\begin{array}{rr}
 & 1.111 \quad \ldots\ldots \quad 1111 \\
+ & \qquad\qquad\qquad\quad 1 \\
\hline
 & 10.000 \quad \ldots\ldots \quad 0000
\end{array}
$$

2.1.2 数的机器码表示

- 补码表示法
  - 定点整数的模：与机器字长有关。
    - 机器字长=8，模=2^8=256
    - 机器字长=n，模=2^n
    - 机器字长=n+1，模=2^{n+1}



$$
\begin{array}{rl}
 & 1111\ \ldots\ldots\ 1111. \\
+ & \qquad\qquad\quad 1. \\
\hline
 & 10000\ \ldots\ldots\ 0000.
\end{array}
$$

2.1.2 数的机器码表示

- 补码表示法
  - 定点小数
    - 如果$0 \le X < 1$，$[X]_{补} = 0.X1X2\ldots Xn = X = [X]_{原}$
    - 如果$-1 \le X \le 0$，$[X]_{补} = 2 + X = 10.00\ldots0 - 0.X1X2\ldots Xn$

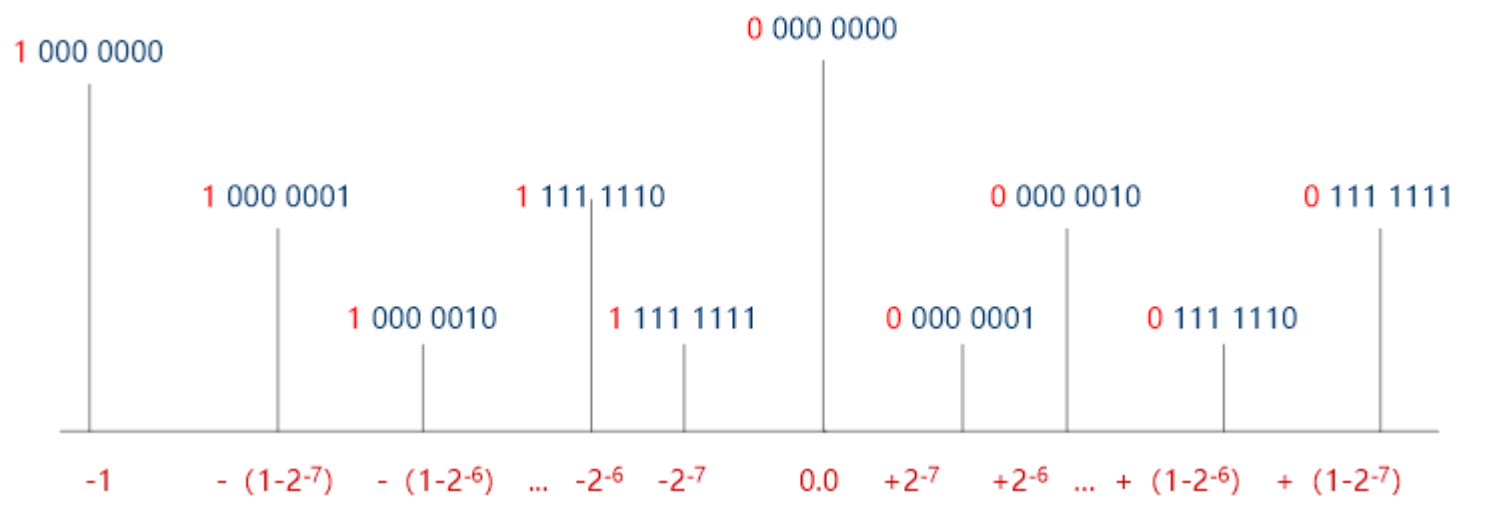
真值	机器字长=8	机器字长=16
+0.1001	0 100 1000	0 100 1000 0000 0000
-0.1001	1 011 1000	1 011 1000 0000 0000
+0.0	0 000 0000	0 000 0000 0000 0000
-0.0	0 000 0000	0 000 0000 0000 0000

1 0.000 0000	1 0.000 0000 0000 0000
- 0.100 1000	- 0.100 1000 0000 0000
1.011 1000	1.011 1000 0000 0000

2.1.2 数的机器码表示

- 补码表示法
  - 定点小数

机器字长	表示范围
8	$-1 \sim +(1-2^{-7})$
16	$-1 \sim +(1-2^{-15})$
32	$-1 \sim +(1-2^{-31})$
n	$-1 \sim +(1-2^{-(n-1)})$



2.1.2 数的机器码表示

- 补码表示法
  - 定点整数
    - 如果$0 \le X < 2^n$，$[X]_{补} = 0X1X2\ldots Xn = X = [X]_{原}$
    - 如果$-2^n \le X \le 0$，$[X]_{补} = 2^{n+1} + X$

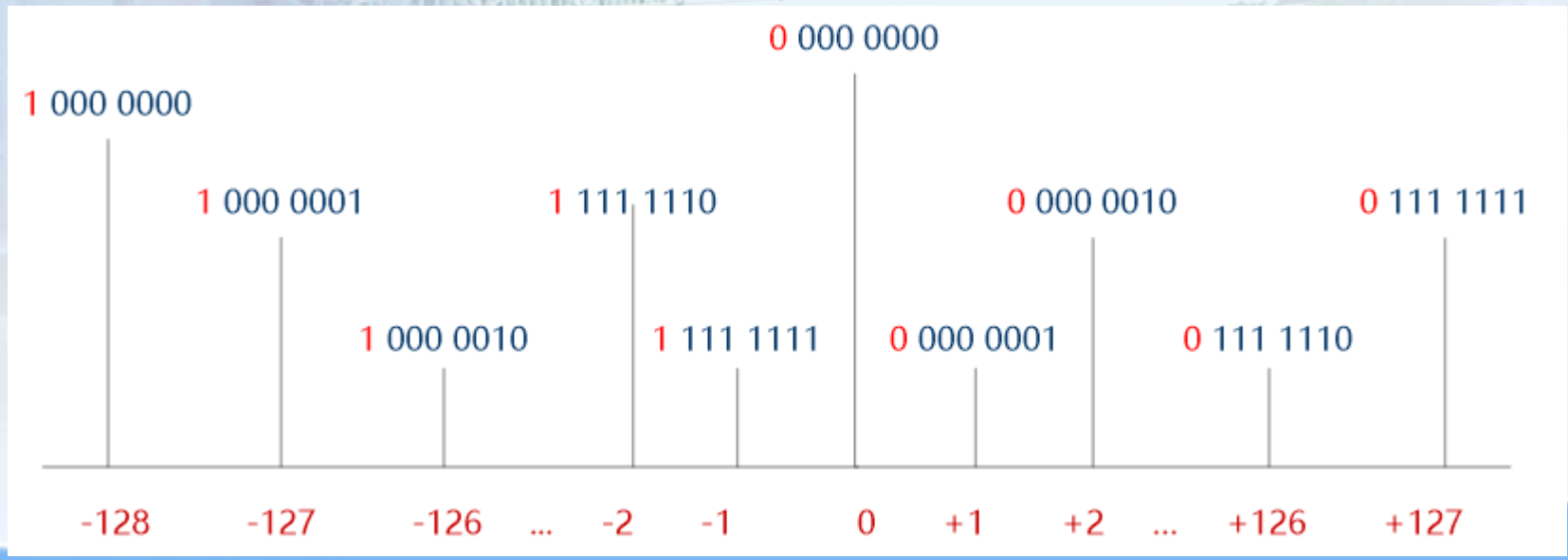
真值	机器字长=8	机器字长=16
+1001	0 000 1001	0 000 0000 0000 1001
-1001	1 111 0111	1 111 1111 1111 0111
+0	0 000 0000	0 000 0000 0000 0000
-0	0 000 0000	0 000 0000 0000 0000

```
  1 0000 0000.          1 0000 0000 0000 0000.
-      0000 1001.    -      0000 0000 0000 1001.
-----------------    ------------------------
     1111 0111.            1111 1111 1111 0111.
```

2.1.2 数的机器码表示

- 补码表示法
  - 定点整数

机器字长	表示范围
8	$-2^7 \sim +(2^7-1)$
16	$-2^{15} \sim +(2^{15}-1)$
32	$-2^{31} \sim +(2^{31}-1)$
n	$-2^{n-1} \sim +(2^{n-1}-1)$



2.1.2 数的机器码表示

- 移码表示法
  - 移码表示法通常用于浮点数的阶码表示，因此，主要讨论定点整数
  - 设机器字长为n+1位，$[X]_{移} = 2^n + X = 1,000\cdots000 + X$

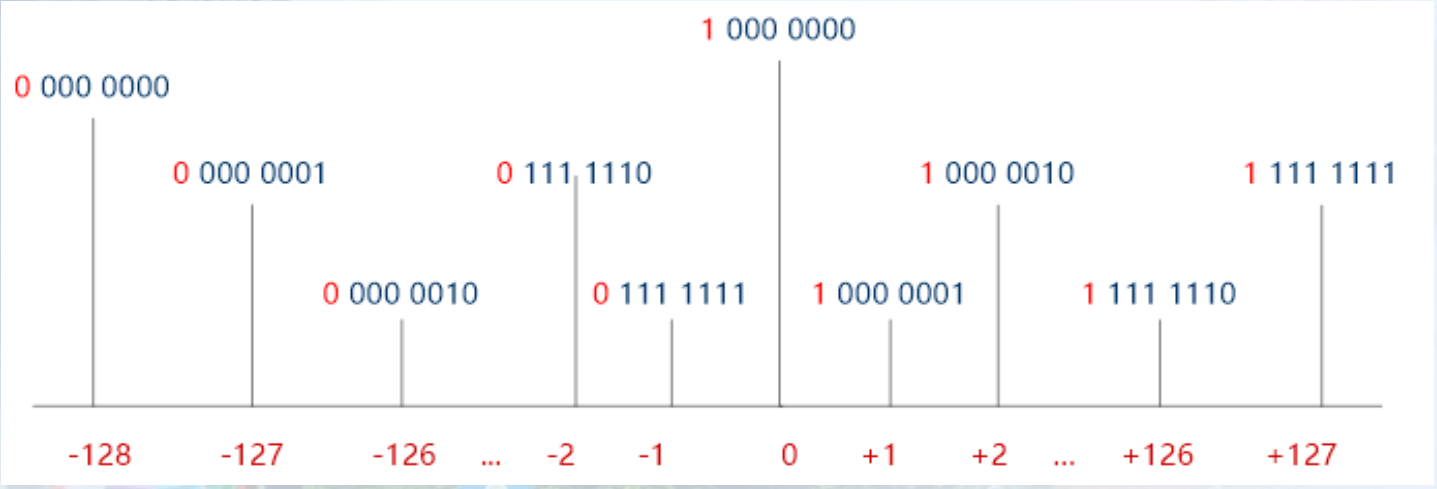
（相当于在符号位加1，1正0负）

真值	机器字长=8	机器字长=16
+1001	1 000 1001	1 000 0000 0000 1001
-1001	0 111 0111	0 111 1111 1111 0111
+0	1 000 0000	1 000 0000 0000 0000
-0	1 000 0000	1 000 0000 0000 0000

```
    1000 0000.          1000 0000.
+   0000 1001.      -   0000 1001.
-------------       -------------
    1000 1001.          0111 0111.
```

2.1.2 数的机器码表示

- 移码表示法
  - 移码的数值部分与补码相同，符号位相反



  - 如果把移码中所有的位都看作是数值位（即无符号数），则它的大小反映了实际数值的大小
  - 浮点数阶码采用移码表示，则对阶比较大小时比较方便

- **1、若某数X的真值为-0.1010，在计算机中该数表示为1.0110，则该数所用的编码方法是（　）码。**

A. 原

B. 补

C. 反

D. 移

[答案]B

- 2、已知X为整数，且$[X]_{补}$ = 10011011，则X的十进制数值是_____。

A. +155

B. –101

C. –155

D. +101

[答案]B

- 3、设寄存器位数为8位，机器数采用补码形式（一位符号位），对应于十进制数-27，寄存器内为_____。

A. $(27)_{16}$

B. $(9B)_{16}$

C. $(E5)_{16}$

D. $(5A)_{16}$

[答案]C

- **4、由3个“1”和5个“0”组成的8位二进制补码，能表示的最小整数是（　）。**

A. -126

B. -125

C. -32

D. -3

[答案]B

练习

1.若－1<x<0,问是否存在一个X值使等式$[X]_{补}=[X]_{原}$成立？若存在，该值是多少？

由于2+x=1-x，所以x=-0.5

2. 已知$[X]_{补}=11100$，求 $[-X]_{补}$，$[X/2]_{补}$及$[2X]_{补}$

$[-X]_{补}$ = 00100（全部取反，末位+1）
$[X/2]_{补}$= 11110（算术右移位规则）
$[2X]_{补}$ = 11000（算术左移位规则）

例题：把十进制数$x=(+128.75)\times 2^{-10}$写成浮点表示的机器数，阶码、尾数分别用原码、反码和补码表示。设阶码4位，阶符1位，尾数15位，尾数符号一位。

步骤：①化为二进制数 $X=10000000.11\times 2^{-10}$

②浮点真值表示：$X=0.10000000011\times 2^{-2}$

③浮点机器数表示：

$x=(+128.75)\times 2^{-10}$

$[x]_{原}=1\ \ 0010\ \ 0\ \ 10000000100000$

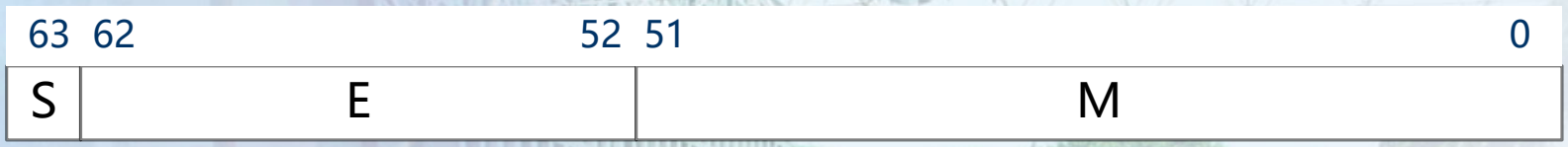
$[x]_{反}=1\ \ 1101\ \ 0\ \ 10000000100000$

$[x]_{补}=1\ \ \underbrace{1110}\ \ 0\ \ \underbrace{10000000100000}$

阶符　阶码　数符　尾数

2.1.2 数的机器码表示

- 浮点数
  - IEEE 754浮点数标准格式



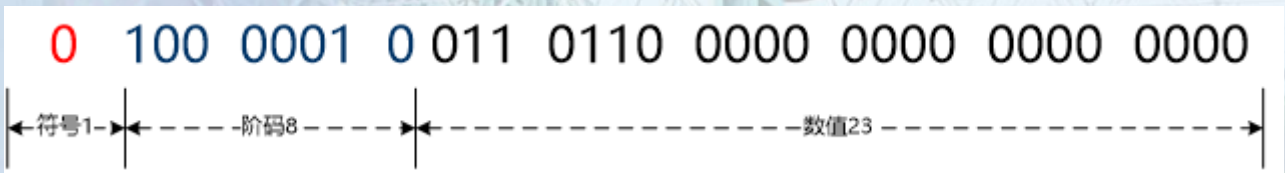
- 尾数=1.M，“1”不保存，尾数可多1位
- 32位浮点数真值=$(-1)^{s}\times(1.M)\times2^{E-127}$
- 64位浮点数真值=$(-1)^{s}\times(1.M)\times2^{E-1023}$

2.1.2 数的机器码表示

- 浮点数

【例】设浮点数x的754标准存储格式为41360000H，求其浮点数的十进制数值。

【解】



指数e=E-127=1000 0010 - 0111 1111 = 3

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

数值x=$(-1)^s \times 1.M \times 2^e = +1.011011 \times 2^3 = (11.375)_{10}$

2.1.2 数的机器码表示

- 浮点数

【例】将数$(20.59375)_{10}$转换成IEEE754标准的32位浮点数的二进制存储格式。

【解】 $(20.59375)_{10}$ =10100.10011=1.010010011 ×2^4

e=4，E=e+127=4+127=131=1000 0011

1.M=1.010 0100 1100 0000 0000 0000

0 100 0001 1 010 0100 1100 0000 0000 0000

41A4C000H

IEEE 754浮点数的编码表示

阶码全0和全1用作特殊值处理

单精度		双精度		表示的数
Exponent	Fraction	Exponent	Fraction	
0	000	0	000	0
0	非零	0	非零	±非规格化数
1~254	任意	1~2046	任意	±浮点数
255	0	2047	0	±无穷
255	非零	2047	非零	NaN（非数）

本次作业

P68习题：

1、2、4

河海大學

Q&A