南京理工大学
NANJING UNIVERSITY OF SCIENCE & TECHNOLOGY

第 2 章 数据信息的表示与运算单元

主讲 张功萱

本章学习内容

- 计算机中数值数据的表示
  - 机器数的概念
  - 原码、补码、反码、移码表示及运算方法
- 数的定点与运算单元
- 浮点表示及运算方法
- 非数值数据的表示
- 数据的长度与存储方式
- 数据校验码

计算机内部信息

- 信息
  - 控制信息
    - 指令
    - 控制字
  - 数据信息
    - 数值型数据
      - 定点数
      - 浮点数
    - 非数值型数据
      - 数字串
      - 字符与字符串
      - 汉字与汉字串

计算机中的数据表示

- 数据表示
- 能够由计算机硬件直接识别的数据类型，如定点数、浮点数等。

- 硬件直接识别
- 即某种数据类型可用计算机硬件直接表示出来，并能用计算机指令直接调用。

- 数据表示（取值范围、精度、类型）
  - 影响计算机性能的全局性问题
  - 直接影响算法的选择、硬件结构与组成
  - 随硬件技术和应用需求而变化和发展
  - 是复杂的系统设计问题

带符号数的表示

第2.1节

1. 机器数与真值怎么界定？
2. 机器怎么表示带符号数？
3. 补码表示的意义是什么？
4. 移码表示的作用是什么？

2.1.1 机器数与真值

- 由于计算机中的硬件电路只能直接表示和处理二进制数，所以需要研究带符号数的符号和小数点在计算机中如何表示。
- 1. 机器数
- 采用二进制表示形式的连同数符一起代码化了的数据，在计算机中统称为机器数或机器码。机器数是数在计算机中的二进制表示形式。
- 2. 真值
- 与机器数对应的用正、负符号加绝对值表示的实际数值称为真值。

3. 机器数的特点

- (1) 数的符号二进制代码化。
  “0”代表+，“1”代表−，且放在数据的最高位。
- (2) 小数点本身是隐含的，不占用存储空间。
- (3) 每个机器数数据所占的二进制位数受机器硬件规模的限制，与机器字长有关。超过机器字长的数值要舍去。

- 机器数可分为
- 无符号数：机器字长的所有二进制位均表示数值
- 带符号数：数值部分和符号均用二进制代码表示
- 例： 8位机器数为：11011011
- 若为无符号整数，则 11011011 表示二进制整数。
  其真值为 11011011＝(219)10
- 若为带符号整数，则最高位为符号，
  1 1011011 表示二进制整数 －1011011
  其真值为 －1011011＝(－91)10

- 机器数表示的数值是不连续的
- 例如：
- 8位二进制无符号数可以表示256个数

  00000000～11111111＝0～2^8－1
- 8位二进制带符号数可以表示－127～127，共256个数。

  11111111～10000000， 00000000～01111111
- 即－1111111～－0 和 ＋0～＋1111111。
- 其中：10000000表示－0， 00000000表示＋0

如何知道计算机表示的数据是否带符号

- 计算机在执行指令时，指令所处理的数据类型由指令操作码决定。

2.1.2 4种码制表示

- **1. 原码表示**：保持原有的数值部分的形式不变，只将符号用二进制代码表示。
- 原码表示是最简单的机器数表示方法。
- （1）原码的定义
- 纯小数原码定义：

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

- 例：$[0.10011001]_{原}=0.10011001$
- $[-0.10011001]_{原}=1.10011001$

- 纯整数原码定义：

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

- 式中 n 为除符号位以外的数值部分的位数

  - 例：
  - $[10011001]_{原}$＝**0**10011001
  - $[－10011001]_{原}$＝**1**10011001

(2) 原码中 0 的表示

- 原码中"0"有两种表示
- 纯小数原码

$[+0]_{原}=0.00\ldots0$

$[-0]_{原}=1.00\ldots0$

- 纯整数原码

$[+0]_{原}=00\ldots0$

$[-0]_{原}=10\ldots0$

(3) 原码的表数范围

- 对于纯小数，n+1位原码的表示范围：
- $-0.\underbrace{111\ldots11}_{n位} \sim +0.\underbrace{111\ldots11}_{n位}$
- 即 $-(1-2^{-n}) \sim (1-2^{-n})$
- 纯小数n+1位原码中有一位是符号
- 对于纯整数，n+1位原码的表示范围：
- $-\underbrace{111\ldots11}_{n位} \sim +\underbrace{111\ldots11}_{n位}$
- 即 $-(2^{n}-1) \sim (2^{n}-1)$
- 纯整数n+1位原码中有一位是符号
- 原码“0”有两种表示方式，n+1 位原码可表示 $2^{n+1}-1$ 个数。

(4) 原码的移位规则

- 符号位不变，数值部分左移或右移，移出的空位填“0”。
- 例：　$[0.0110000]_{原} = 0.0110000$
- $[0.0110000]_{原} = 0.0011000$
- $2 \times [0.0110000]_{原} = 0.1100000$

注意

- 左移时不要将有效位移出，否则将会出错。

(5) 原码的特点

- ⑴ 原码表示直观、易懂，与真值的转换容易。
- ⑵ 原码表示中0有两种不同的表示形式，给使用带来了不便。
- 通常0的原码用$[+0]_{原}$表示，若在计算过程中出现了$[-0]_{原}$，则需要用硬件将$[-0]_{原}$变为$[+0]_{原}$。
- ⑶ 原码表示的加减运算复杂。
- 利用原码进行两数相加运算时，首先要判别两数符号，若同号则做加法，若异号则做减法，还要判别两数绝对值的大小，用绝对值大的数减去绝对值小的数，取绝对值大的数符号为结果的符号。
- 可见原码表示不便于实现加减运算

2. 补码表示

- 引入补码的目的是为了解决原码表示在加减运算时的不便。
- （1） **模的概念**
- 根据运算时“模”的概念
- 5－2＝5＋8＝3 （mod 10）
- 对于确定的模，某数减去一个数，用加上那个数的负数的补数来代替。
- $[x]_补 = M + x \pmod{M}$
- 当$x \geq 0$时，M＋x 大于M，把M丢掉，所以 $[x]_补 = x$， 即正数的补数等于其本身。
- 当$x < 0$时，$[x]_补 = M + x = M - |x|$，所以负数的补数等于模与该数绝对值之差。

(2) 补码的定义

- 在计算机中，由于数据是用二进制编码表示的，所以把补数称为补码。
- 对于纯小数表示，通常取模 M＝2
- 对于纯整数表示，通常取模 M＝2^{n+1}
  (n为除符号位以外数值位的位数)
- 纯小数的补码定义

$$[x]_{补}=\begin{cases} x & 0\le x<1 \\ 2+x & -1\le x<0 \end{cases} \quad (\mathrm{Mod}\ 2)$$

- 纯整数的补码定义

$$[x]_{补}=\begin{cases} x & 0\le x<2^{n} \\ 2^{n+1}+x & -2^{n}\le x<0 \end{cases} \quad (\mathrm{Mod}\ 2^{n+1})$$

- 例：
- ① $x = +0.1011$　$[x]_{补} = 0.1011$
- ② $x = -0.1011$
- $[x]_{补} = 2 + x = 10.0000 - 0.1011 = 1.0101$
- ③ $x = +1011$　$[x]_{补} = 01011$
- ④ $x = -1011$
- $[x]_{补} = 2^5 + x = 100000 - 1011 = 10101$

is called *two's complement* representation:

$00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000_{two} = 0_{ten}$

$00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000001_{two} = 1_{ten}$

$00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000010_{two} = 2_{ten}$

... ...

$01111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111101_{two} = 9{,}223{,}372{,}036{,}854{,}775{,}805_{ten}$

$01111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111110_{two} = 9{,}223{,}372{,}036{,}854{,}775{,}806_{ten}$

$01111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111_{two} = 9{,}223{,}372{,}036{,}854{,}775{,}807_{ten}$

$10000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000_{two} = -9{,}223{,}372{,}036{,}854{,}775{,}808_{ten}$

$10000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000001_{two} = -9{,}223{,}372{,}036{,}854{,}775{,}807_{ten}$

$10000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000010_{two} = -9{,}223{,}372{,}036{,}854{,}775{,}806_{ten}$

– ...

$11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111101_{two} = -3_{ten}$

$11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111110_{two} = -2_{ten}$

$11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111\ 11111111_{two} = -1_{ten}$

64位补码，Pp76

(3) 特殊数的补码表示

- 补码中“0”的表示是唯一的
- $[+0]_补=[-0]_补=0.00\ldots0$ （纯小数）
- $[+0]_补=[-0]_补=00\ldots0$ （纯整数）
- 补码表示的最小数可以表示到-1或-2^n
- 对于纯小数
- $[-1]_补=2+(-1)=1.00\ldots0 \pmod 2$
- 对于纯整数
- $[-2^n]_补=2^{n+1}+(-2^n)=100\ldots0$ （mod 2^{n+1}）

- 因为补码可以表示-1（纯小数）和-2^n（纯整数），所以补码的表数范围比原码大。
- $[-1]_{补}=1.00…0$
- $[-2^n]_{补}=100…0$
- 对于-1和-2^n的补码，符号位上的1具有特殊意义，既表示符号也表示数值。

结论

- 补码中每一种编码都有独立的意义。
- 对于$n+1$位补码，其表数范围为：
- 纯小数 $-1\sim1-2^{-n}$ 共2^{n+1}个数
- 纯整数 $-2^{-n}\sim2^n-1$ 共2^{n+1}个数

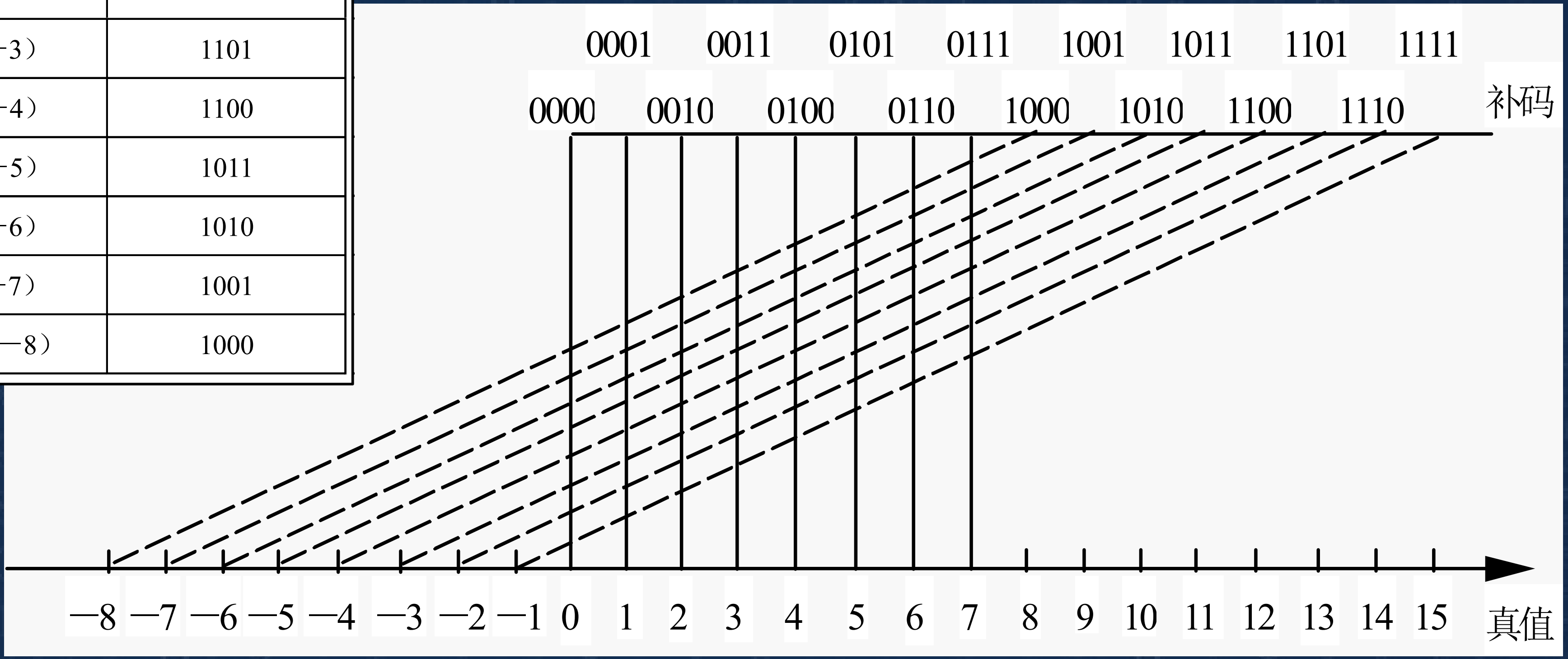
(4) 补码的简便求法

- 若x≥0，则$[x]_{补}=x$，符号位为0
- 若x<0，则将x的各位取反，然后在最低位上加1，符号位等于1，即得到$[x]_{补}$。
  - 例：
  - ①$x=+0.1011001$，$[x]_{补}=0.1011001$
  - ② $x=-0.1011001$
  - $[x]_{补}=1.0100110+0.0000001=1.0100111$
  - ③ $x=+1101010$，$[x]_{补}=01101010$
  - ④ $x=-1101010$，$[x]_{补}=10010110$

(5) 补码的几何性质

- 当n＝3时，纯整数的补码为：

真值	补码	真值	补码
＋000（＋0）	0000	－001（－1）	1111
＋001（＋1）	0001	－010（－2）	1110
＋010（＋2）	0010	－011（－3）	1101
＋011（＋3）	0011	－100（－4）	1100
＋100（＋4）	0100	－101（－5）	1011
＋101（＋5）	0101	－110（－6）	1010
＋110（＋6）	0110	－111（－7）	1001
＋111（＋7）	0111	－1000（－8）	1000



补码的几何性质

- ① 正数的补码就是其本身，负数的补码表示的实质是把负数映像到正值区域，因此加上一个负数或减去一个正数可以用加上另一个数(补码)来代替。
- ② 从表示符号的角度看，符号位的值代表了数的正确符号，**0**表示正数，**1**表示负数。从映像值来看，符号位的值是映像值的一个数位，因此在补码运算中，符号位与数值位一样参加运算。
- 补码的几何性质说明了补码运算的基础。

注意

- 原码运算时符号位不能参加运算。

(6) 补码的几个关系

- 1) 补码与原码的关系
- 若 $x \geq 0$，则 $[x]_{补} = [x]_{原}$
- 若 $x < 0$，则将除符号位以外的 $[x]_{原}$ 各位取反(符号位不变)，然后在最低位上加1，即得到 $[x]_{补}$。反之，将除符号位以外的 $[x]_{补}$ 的各位取反(符号位不变)，然后在最低位上加1，即得到 $[x]_{原}$。

注意

- 补码中特殊数－1（纯小数）和－2^n（纯整数）的表示，在原码中没有对应表示。

- 例：
- ① $x=+0.1001100$
- $[x]_{原}=0.1001100$ $[x]_{补}=0.1001100$
- ② $x=-0.1001100$
- $[x]_{原}=1.1001100$ $[x]_{补}=1.0110100$
- ③ $x=+1001100$
- $[x]_{原}=01001100$ $[x]_{补}=01001100$
- ④ $x=-1001100$
- $[x]_{原}=11001100$ $[x]_{补}=10110100$

2） 补码与机器负数的关系

- 在补码运算中称 $[x]_{补}$为机器正数， $[-x]_{补}$为机器负数。
- 已知$[x]_{补}$， 求机器负数的方法：
- 将$[x]_{补}$的各位(含符号位)取反，然后在最低位上加1，即可得到$[-x]_{补}$。反之亦然。
- 求$[-x]_{补}$，也称为对$[x]_{补}$的求补。

- 例：
- $[x]_{补}=1.0011010$　$[-x]_{补}=0.1100110$
- $[x]_{补}=10110010$　$[-x]_{补}=01001110$

- 简单求补方法：在取反过程中，低位最后一个1不变，最后一个1后的0也都不变。

3） 补码的移位规则

- 补码的右移规则：
- 符号位不变，数值位各位向右移位，高位移空位置补与符号位相同的代码。
- 补码的左移规则：
- 连同符号位同时左移，低位移空位置补0。如果移位后符号位与移位前符号位不一致，说明移位出错，将有效位移出了。

- 例：
- $[x]_{补}=1.0011010$　　$[\frac{1}{2}x]_{补}=1.1001101$
- $[x]_{补}=0.0110010$　　$[\frac{1}{2}x]_{补}=0.0011001$
- $[x]_{补}=10110010$　　$[\frac{1}{2}x]_{补}=11011001$

- $[x]_{补}=1.1111010$　　$[2x]_{补}=1.1110100$
- $[x]_{补}=10110010$　　$[2x]_{补}=01100100$

出错！

（7） 补码的模

- 补码总是对确定的模而言的。如果补码运算结果超过了模，则模将自动丢失。
- 补码运算在运算过程中，模不能改变。
- 因为整数补码的模不同，所以不能将不同位数的补码直接进行运算。如需进行运算，需要进行符号扩展。
- 例2.18
- [x]补＋[y]补＝0110＋1101＝ $\boxed{1}$ 0011＝0011
- [x]补＋[y]补＝11010111＋1011

  ＝11010111＋11111011

  ＝ $\boxed{1}$ 11010010＝11010010

(8) 补码的特点

- ①在补码表示中，用符号位x_0表示数值的正负，形式与原码表示相同，即0正1负。但补码的符号可以看作是数值的一部分参加运算。
- ②在补码表示中，数值“0”只有一种表示方法，即00…0。
- ③负数补码的表示范围比负数原码的表示范围略宽。纯小数的补码可以表示到“－1”，纯整数的补码可以表示到“-2^n”。
- 由于补码表示中的符号位可以与数值位一起参加运算，并且可以将减法转换为加法进行运算，简化了运算过程，因此计算机中均采用补码进行加减运算。

3. 反码表示

- 反码实质上是补码的一个特例，其特别之处在于反码的模比补码的模小一个最低位上的 1。
- （1） 反码的定义
- 纯小数反码的定义（n为小数点后的数值位数）

$$[x]_{反}=\begin{cases} x & 0\le x<1 \\ \left(2-2^{-n}\right)+x & -1<x\le 0 \end{cases} \qquad (\mathrm{Mod}\left(2-2^{-n}\right))$$

- 纯整数反码的定义 （n为除符号外的数值位数）

$$[x]_{反}=\begin{cases} x & 0\le x<2^{n} \\ \left(2^{n+1}-1\right)+x & -2^{n}<x\le 0 \end{cases} \qquad (\mathrm{Mod}\left(2^{n+1}-1\right))$$

（2） 反码的求法

- 若$x \geq 0$ 则$[x]_反 = x$，符号位为0
- 若$x < 0$，则将x的各位取反，符号位等于1，即得到$[x]_反$。
- 例2.19
- $x = +0.1001100$ 则$[x]_反 = 0.1001100$
- $x = -0.1001100$ 则$[x]_反 = 1.0110011$
- $x = +1001100$ 则$[x]_反 = 01001100$
- $x = -1001100$ 则$[x]_反 = 10110011$

(3) 反码中“0”的表示

- 反码中“0”有两种表示
- 纯小数反码

$[+0]_{反}=0.00\ldots0$

$[-0]_{反}=1.11\ldots1$

- 纯整数反码

$[+0]_{反}=00\ldots0$

$[-0]_{反}=11\ldots1$

(4) 反码的表数范围

- 反码的表数范围与原码相同。
- 在纯小数反码中不能表示“-1”
- 在纯整数反码中不能表示“-2^n”

(5) 反码与原码的关系

- 若 $x \geq 0$，则$[x]_{反}=[x]_{原}$
- 若 $x<0$，则将除符号位以外的$[x]_{原}$各位取反(符号位不变)，即得到$[x]_{反}$。
- 若$x<0$，将除符号位以外的$[x]_{反}$的各位取反(符号位不变)，即得到$[x]_{原}$。

(6) 反码的特点

- ①在反码表示中，用符号位x_0表示数值的正负，形式与原码表示相同，即0正1负。
- ②在反码表示中，数值0有两种表示方法。
- ③反码的表示范围与原码的表示范围相同。注意，纯小数的反码不能表示“-1”，纯整数的反码不能表示“-2^n”。
- 反码表示在计算机中往往作为数码变换的中间环节。

4. 移码表示

- 移码也称为增码、余码。在计算机中，移码主要用于表示浮点数的阶码。
- （1） 移码的定义
- 纯小数移码的定义
- $[x]_{移}=1+x \quad -1\leq x<1$
- 纯整数移码的定义
- $[x]_{移}=2^n+x \quad -2^n\leq x<2^n$
- n为除符号外的数值位数
- 由于移码通常用于表示浮点数的阶码。所以主要考虑整数的移码表示。

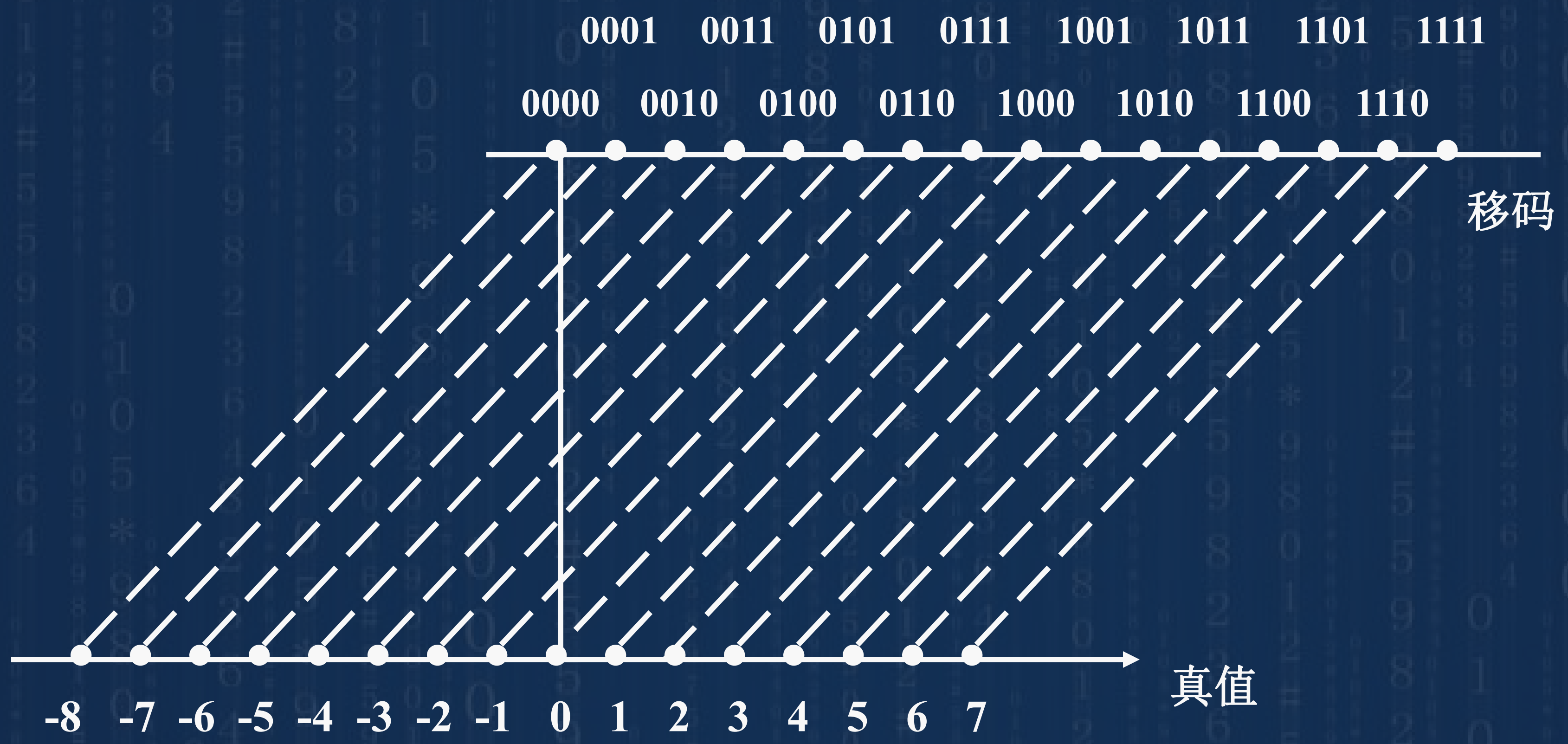
- 例：在字长为8位的机器中，$[x]_移=2^7+x$
- 设 $x=+1100101$
- 则 $[x]_移=2^7+1100101=10000000+1100101$
  $=11100101$
- 设 $x=-1100101$
- 则 $[x]_移=2^7+(-1100101)$
  $=10000000-1100101=00011011$

移码的几何性质

- 当n＝3时，纯整数的移码为：

真值	移码	真值	移码
＋000（＋0）	1000	－001（－1）	0111
＋001（＋1）	1001	－010（－2）	0110
＋010（＋2）	1010	－011（－3）	0101
＋011（＋3）	1011	－100（－4）	0100
＋100（＋4）	1100	－101（－5）	0011
＋101（＋5）	1101	－110（－6）	0010
＋110（＋6）	1110	－111（－7）	0001
＋111（＋7）	1111	－1000（－8）	0000

移码的几何性质


0001 0011 0101 0111 1001 1011 1101 1111
0000 0010 0100 0110 1000 1010 1100 1110
移码
真值
-8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7

- 移码表示的实质是把真值映像到一个正数域，因此移码的大小可直观地反映真值的大小。
- 不管正数还是负数，用移码表示时，都可以按无符号数比较大小。

移码中“0”的表示

- 移码中“0”的表示是唯一的
- $[+0]_{移}=[-0]_{移}=10\ldots0$ （纯整数）

- 移码的表数范围与补码一致
- 纯整数移码表示的最小数可以表示到-2^n

(2) 移码与补码的关系

- 整数补码的数值部分不变，符号取反，即得整数移码。反之亦然。即：
- $x \geq 0$时　　$[x]_{移} = [x]_{补} + 2^n$
- $x < 0$时　　$[x]_{移} = [x]_{补} - 2^n$

(3)　移码的特点

- ① 设$[x]_移=x_0x_1x_2 \ldots x_n$，符号位$x_0$表示真值x的正负：　$x_0=1$，x为正；$x_0=0$，x为负。
- ②真值0的移码表示只有一种形式。
- ③移码与补码的表示范围相同。
- 纯小数的移码可以表示到“-1”，
- $[-1]_移=0.0\ldots0$；
- 纯整数的移码可以表示到“-2^n”，
- $[-2^n]_移=00\ldots0$，　n为数值部分的长度。

- ④ 真值大时，对应的移码也大；
  真值小时，对应的移码也小。

- 当$[x]_{移}$＝ 0时，x为编码所能够表示的最小值。

（4） 移数值为K的移码

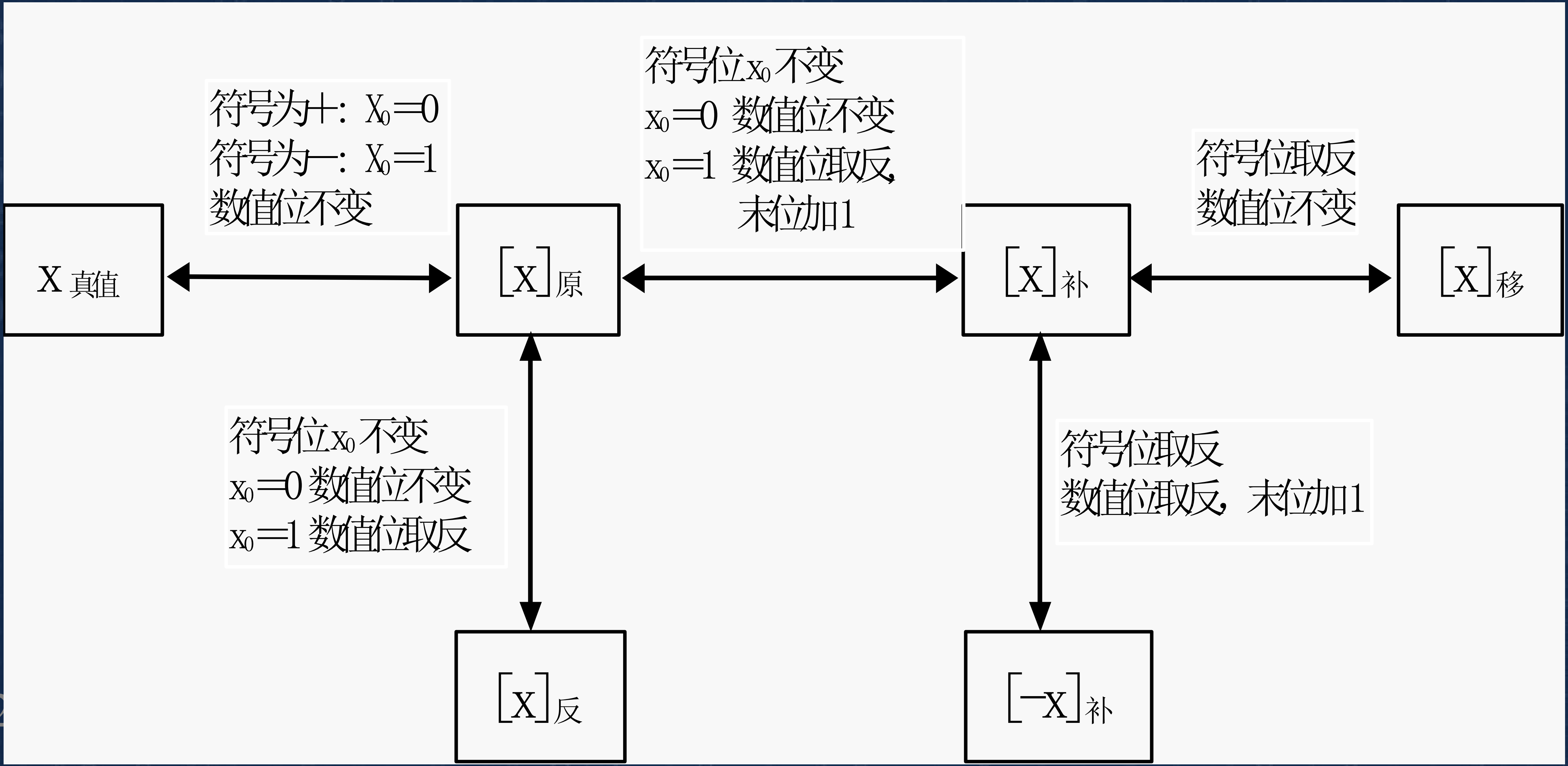
- 根据移码的几何性质，可以将移码的定义进行扩展，得到特殊的移码为：
- 移码＝K＋实际数值
- K：约定的移数值

- 例：移127码，移数值为127，即：
- 移127码＝127＋实际数值

- 例：求＋12和－3的8位移127码的二进制编码形式。
- 解： $(+12)_{10}=1100$，
- $[+12]_{移127码}=127+12=(139)_{10}$
$=(1111111+1100)_2=(10001011)_2$

- $(-3)_{10}=11$，
- $[-3]_{移127码}=127-3=(124)_{10}$
$=(1111111-11)_2=(01111100)_2$

不同码制之间的转换


X 真值
符号为+：X_0=0
符号为–：X_0=1
数值位不变
[X]原
符号位x_0不变
x_0=0 数值位不变
x_0=1 数值位取反，
末位加1
[X]补
符号位取反
数值位不变
[X]移
符号位x_0不变
x_0=0 数值位不变
x_0=1 数值位取反
[X]反
符号位取反
数值位取反，末位加1
[–X]补

- 例：设某计算机的字长为8位，采用整数表示。求表中机器数在不同表示形式中对应的十进制真值。

表示方法 / 机器数	原码	补码	反码	移码	无符号数
01001001	＋73	＋73	＋73	－55	73
10101101	－45	－83	－82	＋45	173
11111111	－127	－1	－0	＋127	255