

第 2 章 数据的表示

在计算机内部各类**基本数据**的表示（编码）方法。

数据类型

- **基本数据类型**：硬件能直接**实现**的数据类型。
分为**数值型数据**和**非数值型数据**。
- **复杂数据类型**：按某种结构描述方式在软件中实现。

本章学习导读

- (1) 计算机中如何来表示数据，包括数值数据和非数值数据。
- (2) 数值数据的编码表示：包括数制，原码、反码、补码，以及定点与浮点数。
- (3) 非数值数据如英文字符、汉字等的表示法。

2.2 数字化信息编码

编码

编码：就是用少量简单的基本符号的组合，表示大量复杂多样的信息。

在计算机系统中，凡是要进行**处理**、**存储**和**传输**的信息，都是用**二进制**进行编码的。

计算机内部采用二进制表示的原因

1. 只有 0、1 两个数码，易于用物理器件表示；
2. 电位的高低，脉冲的有无，电路通断，磁化方向等都比较**容易区别**，**可靠性高**；
3. 运算规则简单；如加减运算规则、乘法表。
。
4. 二进制的 0、1 与逻辑命题中的真假相对应，为计算机中实现**逻辑运算**和**逻辑判断**提供有利条件。

缺点：书写冗长，难认，难记，不易发现错误。
。

2.3 数值数据的编码表示

数值数据的数字化

数值数据

数值数据：表示数量多少，数值大小，可比较。
在计算机内部，数值数据的**表示方法**有两大类：

- 1) 二进制：直接用数据的二进制数表示；
- 2) 十进制：

十进制：采用二进制编码的十进制数
(Binary Coded Decimal Number，
简称 BCD) 表示。

例，十进制数 17，机器数长 8 位。

数值数据的表示

三个要素：

1. 进位计数制；
2. 符号的数字化？带符号数的编码表示？
3. 小数点？位置？定 / 浮点表示。

进位计数制

基数：允许使用的**基本符号**个数。

位权：不同数位的**权值**（**数量级别**）。

例：十进制数，

$$123.4 = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 + 4 \times 10^{-1}$$

基本符号： 0~9

基数： 10

位权： 10^i

进位计数制

在某个数字系统中，若采用 R 个基本符号 $(0, 1, 2, \dots, R-1)$ 表示各位上的数字，则称其为 **R 进制** 数字系统；

- R 被称为该数字系统的**基数**；
- 采用“**逢 R 进一，借一当 R** ”的运算规则；
- 第 i 位的权为 R^i 。

$$D_R = d_{n-1}d_{n-2}\dots d_1d_0 . d_{-1}d_{-2}\dots d_{-m}$$

$$D_R = d_{n-1} \times R^{n-1} + d_{n-2} \times R^{n-2} + \dots + d_1 \times R^1 + d_0 \times R^0 \\ + d_{-1} \times R^{-1} + d_{-2} \times R^{-2} + \dots + d_{-m} \times R^{-m}$$

计算机系统中常用的进位计数制

二进制 (Binary): $R=2$, 基本符号为 0 和 1

八进制 (Octal): $R=8$, 基本符号为 0,1,2,3,4,5,6,7

十六进制 (Hexadecimal): $R=16$, 基本符号为 0,1,2,3,4,5,6,7,8,9,a,b,c,d,e,f

十进制 (Decimal): $R=10$, 基本符号为 0,1,2,3,4,5,6,7,8,9

举例说明。

进位计数制的相互转换

二、八、十六进制：**分段对应**转换。从小数点开始。

R 进制数转换成十进制数：**按权相加法**。

十进制数转换成 R 进制数：将整数和小数部分分别进行转换。除基取余法，乘基取整法。

以上转换的**特点**：

操作统一，算法简单，易于软件编程及硬件实现。

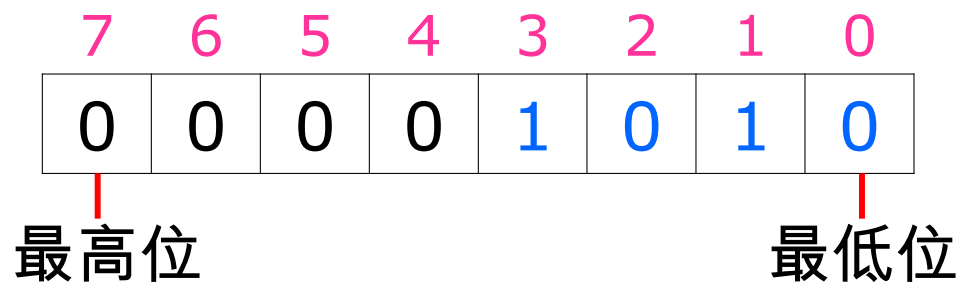
数值数据的编码表示

三个要素：

1. 进位计数制；
2. 符号？数字化？数的编码表示？
3. 小数点？位置？定 / 浮点表示

机器数位的编号

一个字节：



数值数据的编码表示

计算机用**数字**表示正负，**隐含规定**小数点（定点与浮点）。

计算机中常用的**数据表示格式**有两种：

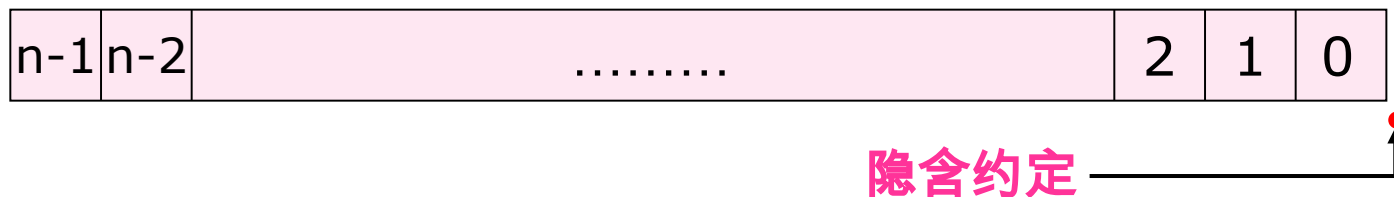
定点格式：容许的数值范围有限，但要求的处理硬件比较简单。

浮点格式：容许的数值范围很大，但要求的处理硬件比较复杂。

定点数的表示方法

定点表示：约定机器中**所有数据**的小数点位置是**固定不变**的。

定点整数（**纯整数**）：小数点固定在**最低位数**的后面；



定点小数（**纯小数**）：小数点固定在**最高位数**的后面。



定点无符号数的表示

无符号数：编码的**所有二进制位**都用来表示数值。

一般在全部是正数运算且不出现负值结果的情况下，可以省略符号位，使用无符号数表示。

计算机的运算部件与寄存器都有一定**字长限制**，如 8 位、16 位或 32 位，只能表示一定范围内的数据。

定点数的表示方法

问题：两种类型的定点数，无符号数的表示范围？



$$0 \sim 2^n - 1$$



$$0 \sim 2 - 2^{-(n-1)}$$

定点带符号数的表示



机器数与真值

真值

机器数

带符号的数

符号数字化的数

+ 0.1011



小数点的位置

- 0.1011



小数点的位置

+ 1100



小数点的位置

- 1100



小数点的位置

定点带符号数的表示

机器数：数值数据（有正负）在计算机内部的**编码**（只有 0、1）。

真值：机器数所表示的真正值，即原来带有正负号的数。



常用的编码表示方式 (机器数)

三种：原码、补码和反码。

对于这三种编码：

1. 正数的所有编码表示都是相同的：符号位取值为 0，数值部分是二进制真值；
2. 负数的所有编码表示，其符号位总是为 1，而数值部分对于不同的编码则有不同的取值。

原码表示法

原码：由符号位后直接跟上真值的**绝对值**构成。

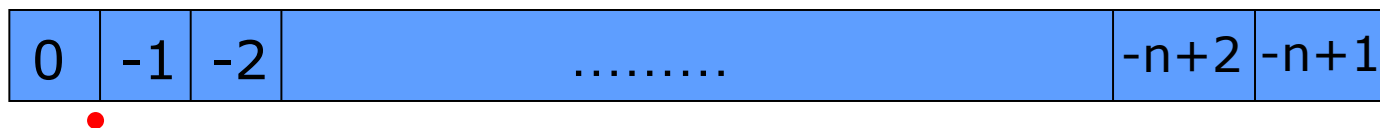
1. **最高位**是符号位，“0”正，“1”负；
2. 有效数值部分用二进制的**绝对值**表示；
3. 0 的原码有 +0 和 -0 两种形式。

$$[+0.1001]_{\text{原}} = 0.1001000 \quad [-0.1001]_{\text{原}} = 1.1001000$$

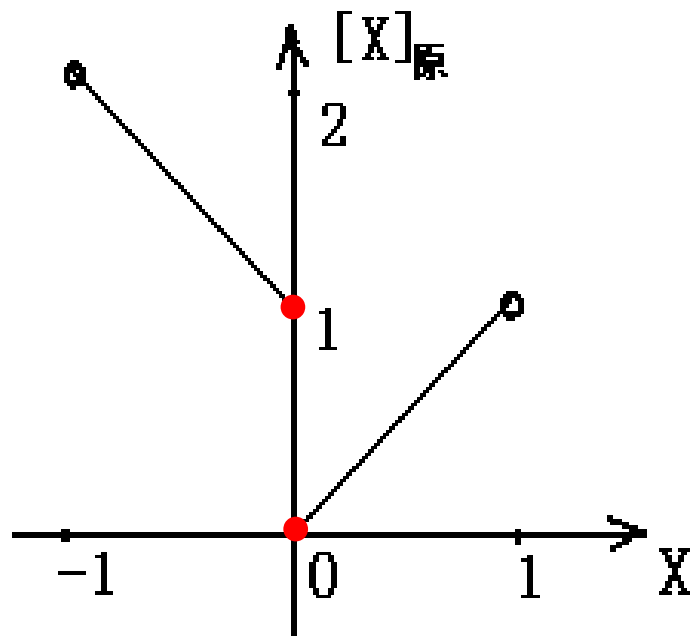
$$[+1001]_{\text{原}} = 00001001 \quad [-1001]_{\text{原}} = 10001001$$

$$[+0]_{\text{原}} = 00000000 \quad [-0]_{\text{原}} = 10000000$$

定点小数（纯小数）：



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



原码表示法

0	-1	-2	-n+2	-n+1
---	----	----	-------	------	------

原码定点小数的表示范围

设机器数字长为 n 位：

典型值	原码	真值
最大值	011...1	$1 - 2^{-(n-1)}$
最小值	111...1	$-(1 - 2^{-(n-1)})$

表示数的个数： $2^n - 1$ 为什么？

分辨率——绝对值最小正数？ $2^{-(n-1)}$

问题：字长 8 位的原码定点小数，表示的最大值、最小值？ $127/128$ ， $-127/128$

问题：求 0 的原码？ $00...0$ ， $10...0$

原码表示法

定点整数（纯整数）：公式和表示范围略。

原码表示法

问题：

1. 16 位整数的原码表示？如 ± 1001 ？
2. 16 位小数的原码表示？如 $\pm 0.101\ 1011$ ？

特点：

1. 数值位表示真值的绝对值，与真值的对应关系**直观**，因此与真值之间的**转换简单**；
2. 乘除运算较方便，加减运算规则复杂。

例， `mul R1, R2`
`add R1, R2`