数据的机器级表示与运算

定点数的编码

主讲人: 邓倩妮

上海交通大学

部分内容来自：

1. 《深入理解计算机系统》第三版，机械工业出版社，作者：Bryant,R.E.等
2. Computer Organization and Design, 4th Edition, Patterson & Hennessy

上海交通大学 SHANGHAI JIAO TONG UNIVERSITY

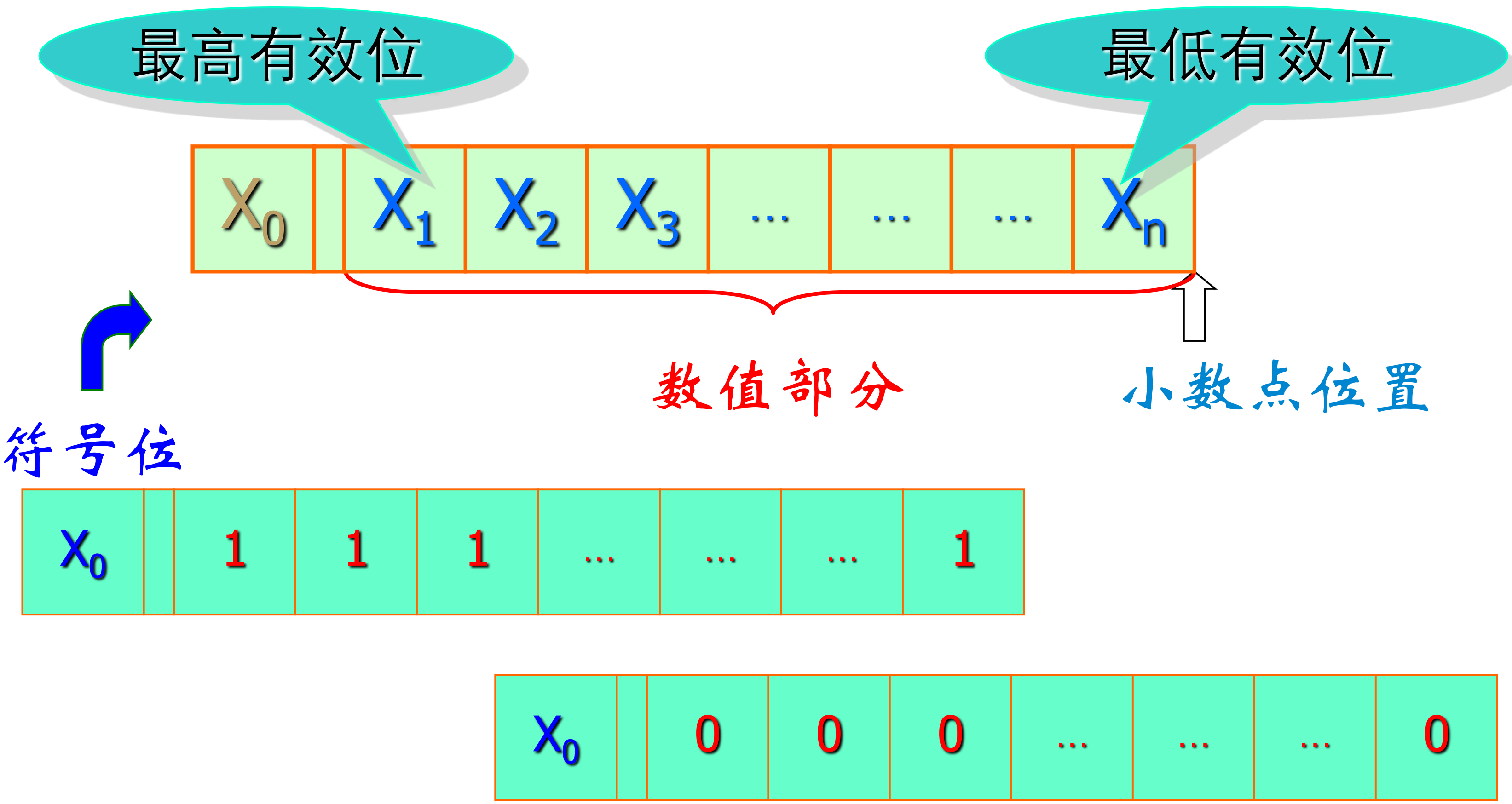
本节内容

- 计算机中的整数类型
  - short, int, long
- 计算机中的整数的编码
  - 原码、反码、补码
- 补码特点

数据类型—整数

- 整型数的表示范围：由各个编译器指定。整型数有三种存储方式，在C语言中占用的空间如下所示:
  - 基本型 int：
    - 4 byte　$-2^{31} \sim (2^{31}-1)$
  - 长整型long： long / long int
    - 4 byte （32位机器）　$-2^{31} \sim (2^{31}-1)$
    - 8 byte （64位机器）　$-2^{63} \sim (2^{63}-1)$
  - 短整型 short ：
    - 2 byte　$-2^{15} \sim (2^{15}-1)$

定点数：纯整数

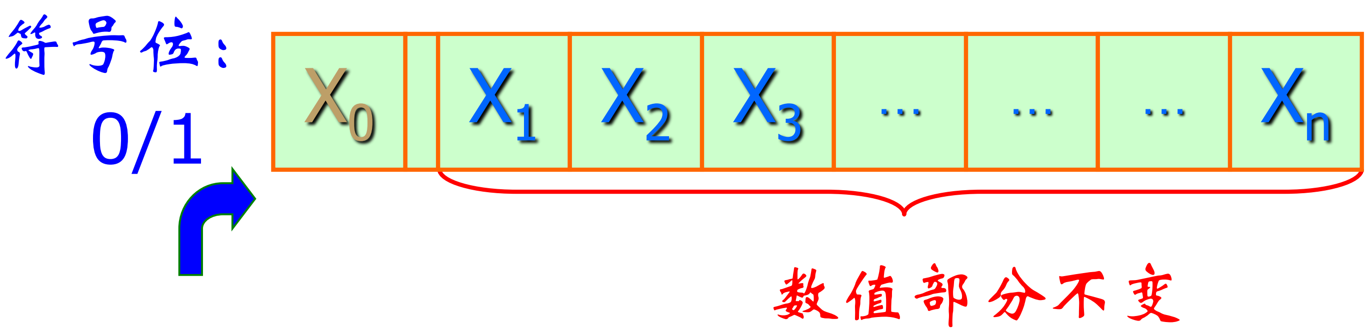


机器数/机器码

- 真值　(书写用)
  - 将用+ -表示正负的二进制数称为符号数的真值
- 机器码 (机器内部使用)
  - 将符号和数值一起编码表示的二进制数称为机器码
- 原码 Signed magnitude　反码 One’s complement
- 补码 Two’s complement

原码表示法（Signed magnitude）

- 计算机如何表示数的正负?
- 增加符号位 Add a sign bit
- 最高位为符号位，0为正，1为负，数值位不变



原码表示示例

- $[+0]_{原}=0\ 000\cdots0$
- $[-0]_{原}=\underline{1\ \ 000\cdots0}$

- $[\ 1110]_{原}\ =0\ 1110$
- $[-1110]_{原}=1\ 1110$

原码的缺点：不方便计算

$$
\begin{array}{rrcr}
 & 01011001_2 & = & 89_{10} \\
+ & \underline{11001101}_2 & = & \underline{-77}_{10} \\
 & 00100110_2 & = & 32_{10}
\end{array}
$$

两个计算数如果符号不相同，用原码计算会得到错误的计算结果

反码（ one's complement）

- 反码：如果是负数，二进制的各位数码取反
- 符号位表示方法与原码相同
- 例: $7_{10} = 00111_2$ ； $-7_{10} = 11000_2$
- 又叫 1的补码 One's Complement

负数符号位：
1

X_0		X_1	X_2	X_3	…	…	…	X_n

数值部分取反

反码的表示

- $[+0]_{反}=0\ 000\cdots0$
- $[-0]_{反}=1\ 111\cdots1$

- $[1110]_{反}=0\ 1110$
- $[-1110]_{反}=1\ 0001$

反码的缺点

- 算术运算仍然很复杂
- 仍然有两个零
    - $0x00000000 = +0_{ten}$
    - $0xFFFFFFFF = -0_{ten}$
- 不太被使用

补码 (two’s complement)

- 方法1
  - 正数：直接取其原来的二进制码（加符号位0）
  - 负数：对其二进制码按位取反之后再在最低位加1

    例： $[010101]_{补} = 0001\ 0101$

    $[-010101]_{补} = [-0001\ 0101]_{补} = 1110\ 1010+1 = 1110\ 1011$

- 方法2
  - 正数：同上；负数：从二进制码的最低位开始，对遇到的0和第一个1取其原来的二进制编码，从第一个1以后开始直到最高位均取其相反编码。

    例：$[101010]_{补} = 0010\ 1010$,

    $[-101010]_{补} = [-0010\ 1010]_{补} = 1101\ 0110$

例子：整数的补码

- X=+ 0111 1111

$[X]_{补}$ =???

$[X]_{补}$ = 0111 1111

- X=- 0111 1111

$[X]_{补}$ =???

$[X]_{补}$ =1000 0000

+ 0000 0001

= 1000 0001

- X=- 0000 0000

$[X]_{补}$ =???

$[X]_{补}$ =1111 1111

+ 0000 0001

$=\underline{10}000\ 0000$

= 0000 0000

带符号整数（int）在机器内的编码

$0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000_{two} = 0_{ten}$

$0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0001_{two} = +1_{ten}$

$0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0010_{two} = +2_{ten}$

...

$0111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1110_{two} = +2{,}147{,}483{,}646_{ten}$

$0111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111_{two} = +2{,}147{,}483{,}647_{ten}$

$1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000_{two} = -2{,}147{,}483{,}648_{ten}$

$1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0001_{two} = -2{,}147{,}483{,}647_{ten}$

$1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0010_{two} = -2{,}147{,}483{,}646_{ten}$

...

$1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1101_{two} = -3_{ten}$

$1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1110_{two} = -2_{ten}$

$1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111_{two} = -1_{ten}$

补码表示法

$$[X]_{补}=\begin{cases} X & 0\le X<2^n \\ 2^{n+1}+X & -2^n\le X<0 \end{cases}$$

编码长度为n+1位

求值方法($[X]_{补}= x_0x_1 \cdots x_{n-1}\, x_n$)

$$x = -x_0 2^n + x_1 2^{n-1} + \cdots + x_{n-1}2 + x_n$$

例如：10000100的真值为-128+4=-124

补码表示法

$$[X]_{补}=\begin{cases} X & 0\le X<2^{n} \\ 2^{n+1}+X & -2^{n}\le X<0 \end{cases}$$

编码长度为n+1位

补码是以2^{n+1}为模的计量系统

模： 一个计量系统的计数范围

例如：编码长度为4，表达的数据范围为-8 ~ +7

5	0101
+ 6	0110
= - 5	1011

5+6= 16 + (- 5)

模： 16

同余的概念

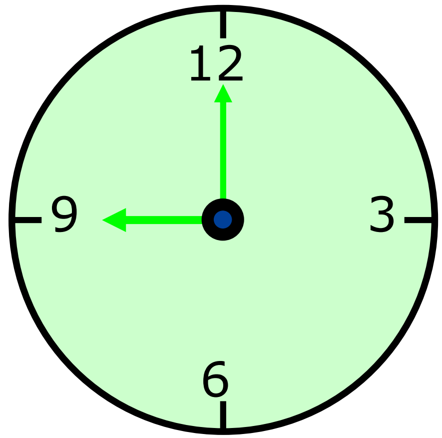
- 假定有两个数a和b，若用某一个整数m去除，所得的余数相同，即：
- a=km+b (k为整数),
- 就称a,b两个数对m同余,记作：

$$a \equiv b \pmod{m}$$

在一个模运算系统中，一个数与它除以“模”（m）后得到的余数是等价的, 即：同余的数是等价的。

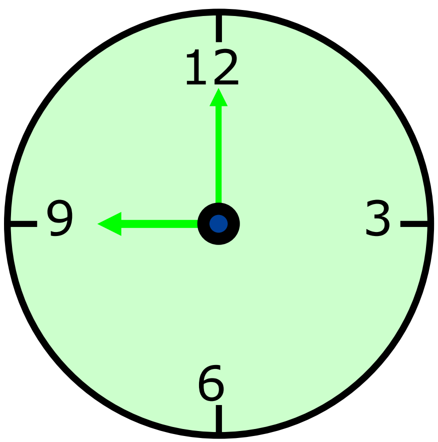
有趣的时钟

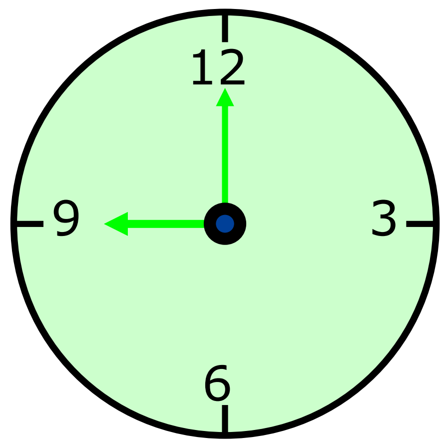
- 时钟的计数范围是0~11
- 以12为模
- 0和12等价



9+24= 9

9-24= 9





9+9 = 6

9-3=6

例子（减法变成加法）

- 7-4
- =7+(12-4)　　模为12
- =7+8
- =15
- =12+3
- = 3

补码特点

- 减法可以变成加法

  $[X+Y]_{补} = [X]_{补} + [Y]_{补}$

  $[X-Y]_{补} = [X]_{补} + [-Y]_{补}$

- 符号位可以直接参与运算
- 唯一的零
- 负数比正数多一个
- 补码加减法运算方便，得到了广泛的应用，目前计算机中广泛采用补码表达整数。

练习题

- 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

小结

- 整数类型： int， long , short
- 原码、反码、补码
- 目前计算机中广泛采用补码表达整数

谢谢！