第一章 二进制

一、二进制概述

二进制有四种表现形式，分别是：原码，反码，补码，移码。之所以会有这么多码，最根本的原因就是计算机只有加法运算器，减法运算需要转换成负数的加法。计算机中专门用一个位来表示数的符号，通常用最高位来表示符号位，0代表正，1代表负。在计算机中，负数是以其正数的补码形式表示的。

二、真值和字长

1、真值

例如有符号数10000101，其最高位1代表负，所以余下的“0000101”才是数值本身，所以其真正数值是-5，而如果是无符号数，则10000101所代表的是133。为区别起见，把带符号位的机器数所对应的真正数值称为机器数的“真值”。例：00100001的真值=0 0100001=+33（正号可以不写，可以直接写成33），10100011的真值=1 0100011=-35。

2、字长

字长是指计算机一次可处理的二进制数的码位长度，是计算机进行数据存储和数据处理的运算单位。例如：十进制-5对应字长为8位的二进制是：10000101，对应字长为16为的二进制是：10000000 00000101。所以字长越长代表计算机的处理能力越强，可以处理的数越大，8位能处理的数值范围：-2^7 到 2^7-1，16位能处理的数值范围：-2^15到2^15-1，依次类推……

三、二进制的四种表现形式

1、原码、反码、补码

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字长 | 十进制 | 二进制原码 | 二进制反码 | 二进制补码 |
| 8bit | 7 | 00000111 | 00000111 | 00000111 |
| 8bit | -7 | 10000111 | 11111000 | 11111001 |
| 16bit | 7 | 00000000 00000111 | 00000000 00000111 | 00000000 00000111 |
| 16bit | -7 | 10000000 00000111 | 11111111 11111000 | 11111111 11111001 |
| 注意：  在原码表示形式中，0有“+0”和“-0”之分，对应的原码分别是0 0000000和1 0000000。  在原反码表示形式中，0也有“+0”和“-0”之分，对应的反码分别是0 0000000和1 1111111。 | | | | |

①、正数的原码、反码和补码都是一样的；

②、负数的反码是对原码中除符号位外的其他各位取反；

③、负数的补码是再对其反码加1，即先对其原码中除符号位外的其他各位取反，然后 再在最低位加1。

④、负数补码的原码和负数原码取补码的过程一样，先对其除符号位外的其他各位全部取反，然后在最低位加1。

⑤、只有相同码制的数才能进行操作，结果是对应的码制，即原码与原码的运算，结果 是原码，反码数与反码数的运算结果也为反码，补码数与补码数的运算结果也为补 码。如果结果是负数，要判断结果是否正确，需要再将其对应的码制转换为原码。

2、移码

移码是一种比较特殊的二进制数表示形式，它的编码规则为：正数的符号位为1， 负数的符号位为0；真值部分与补码一样。

四、二进制的运算

1、加减法运算

所有的减法都可以转化成加法运算，对于所有二进制数的算术运算本质都应该求得其补码，然后通过两个数的补码运算得到为补码的结果，判断结果的正负，若为正，其原码就是其本身，所以直接得到正确结果；若为负，将其再转化成原码，得到正确结果。为了方便计算：

①对于双负数的加法可以直接采用原码计算（符号位不变）；

②对于减数小于被减数的正数减法可以直接采用原码计算；

③对于其他情况都需要采用补码进行计算；

|  |  |  |  |
| --- | --- | --- | --- |
| 运算法则 | 数1 | 数2 | 运算过程 |
| 加法 | (5)D=(00000101)B | (9)D=(00001001)B | 00000101  + 00001001  00001110  2^1+2^2+2^3=14 |
| (-3)D=(10000011)B | (-5)D=(10000101)B | 10000011  + 10000101  10001000  -(2^3)= -8 |
| (3)D=(00000011)B补 | (-5)D=(11111011)B补 | 00000011  + 11111011  补 11111110  反 10000001  原 10000010  -(2^1)= -2 |
| 减法 | (9)D=(00001001)B | (5)D=(00000101)B | 00001001  \_ 00000101  00000100  2^2= 4 |
| (-15)D=(11110001)B补 | (-13)D=(11110011)B补 | 11110001  \_ 11110011  11111110  10000001  10000010  -(2^1)= -2 |
| (-15)D=(11110001)B补 | (9)D=(00001001)B | 11110001  \_ 00001001  11101000  10010111  10011000  -(2^4+2^3)= 24 |

2、乘除法

与十进制乘除法原理相似。

二、常见进位制和表示方法

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 十进制 | 二进制 | 八进制 | 十六进制 |
| 英文表示 | Decimal | Binary | Octal | Hexadecimal |
| 下标 | D/10 | B/2 | O/8 | H/16 |
| 示例 | (125)D/(125)10 | (1111)B/(1111)2 | (127)O/(127)8 | (12E)H/(12E)16 |
| Java/c/c++表示 | 125 | 0b111 | 0123 | 0x10 |
| 对应关系 | | | | |
| 对应关系 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| 2 | 10 | 2 | 2 |
| 3 | 11 | 3 | 3 |
| 4 | 100 | 4 | 4 |
| 5 | 101 | 5 | 5 |
| 6 | 110 | 6 | 6 |
| 7 | 111 | 7 | 7 |
| 8 | 1000 | 10 | 8 |
| 9 | 1001 | 11 | 9 |
| 10 | 1010 | 12 | A |
| 11 | 1011 | 13 | B |
| 12 | 1100 | 14 | C |
| 13 | 1101 | 15 | D |
| 14 | 1110 | 16 | E |
| 15 | 1111 | 17 | F |

三、不同数值之间的转换

1、非十进制换十进制

|  |  |  |
| --- | --- | --- |
|  | 计算公式 | 示例 |
| B->D | (S)d = T\*2^n-1+T\*2^n-2+…+T\*2^0 (T=0,1) | (11011)b=1\*24+1\*23+1\*21+1=27 |
| O->D | (S)d = T\*8^n-1+T\*8^n-2+…+T\*8^0 (T=0~7) | (67)b=6\*81+7=55 |
| H->D | (S)d = T\*16^n-1+T\*16^n-2+…+T\*16^0 (T=0~15) | (AF5)= 10\*162+15\*161+5=2805 |

2、十进制转换成非十进制