第一部分　汇编语言

第1章　进制及码元

进制和码元换算是计算机重要基础之一，计算机内采用的是二进制数值或编码，而在各种汇编语言中习惯使用十六进制，也可使用八进制、二进制和十进制，在C语言中可使用八进制、十六进制和十进制，特别是调试程序时更要与进制和码元换算打交道。所以掌握进制和码元换算的快速方法，对学好计算机相关课程特别是汇编语言、微机原理及接口技术非常重要。

本章所介绍的进制转换方法可以完成十进制、二进制、十六进制及八进制数之间的快速转换，一般可以在10s内完成万以内的数值转换。此外，本章所介绍的真值(有符号数)与补码(或无符号数)之间的直接转换也是前人未曾涉及的，负数与补码(或无符号数)之间转换也只要10s左右。

1.1　进制转换及计算

本节主要讲解进制的快速转换方法，学会此法可在10s内实现万以内的数值转换。

1. 进制

现实生活中除了最常用的十进制外，还有秒分时之间的六十进制、月年之间的十二进制以及古代钱两斤之间的十六进制等，在计算机语言中主要采用的是二进制(后缀B，Binary)、八进制(后缀O或Q，Octal，O易与0混淆，所以一般用Q替代O)、十进制(后缀D，Decimal，或不要后缀)和十六进制(后缀H，Hex)。4种进制基本信息如表1.1所示。

表1.1　计算机语言中的基本进制

进　制	英　文	尾　缀	数据位取值	举　例	算术运算
二	Binary	B	0、1	10110101B	逢二进一，借一等于二
八	Octal	O或Q	0～7	123O或357Q	逢八进一，借一等于八
十	Decimal	D或省略	0～9	68D或259	逢十进一，借一等于十
十六	Hex	H	0～9、A～F	2FCH	逢十六进一，借一等于十六

N进制的每个数据位取值范围为0～N－1，其算术运算规则同十进制，只不过是逢N进一、借一等于N而已。例如，二进制只有0和1两个数字，逢2进1，借1等于2；十六进制有0～9、A～F(分别代表10～15)16个数字，逢16进1，借1等于16。

2. 进制转换的一般方法

进制转换的一般方法如图1.1和图1.2所示。

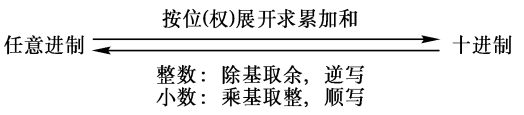

图1.1　任意进制数与十进制数之间转换关系图

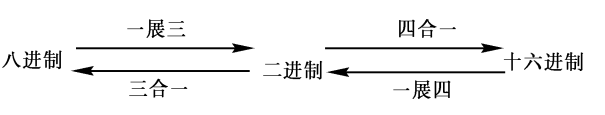

图1.2　二进制、八进制、十六进制之间转换关系图

例 1.1 $(101101)_2=101101\mathrm{B}=1\times2^5+0\times2^4+1\times2^3+1\times2^2+0\times2^1+1\times2^0=45$

例 1.2 $156.4\mathrm{Q}=1\times8^2+5\times8^1+6\times8^0+4\times8^{-1}=110.5$

例 1.3 $6\mathrm{C}.4\mathrm{H}=6\times16^1+12\times16^0+4\times16^{-1}=108.25$

下式中 a_i 代表 b 进制的第 i 位，任意的 b 进制转化为十进制的一般式子：

$$(a_na_{n-1}\cdots a_1a_0,a_{-1}\cdots a_{-m})_b=a_n\times b^n+a_{n-1}\times b^{n-1}+\cdots+a_0\times b^0+a_{-1}\times b^{-1}+\cdots+a_{-m}\times b^{-m}$$

$$=\sum_{i=-m}^{n}a_i\times b^i$$

例 1.4 $123.25=(1\ 111\ 011.01)_2=(173.2)_8=(7\mathrm{B}.4)_{16}$

解题步骤如图 1.3 所示。

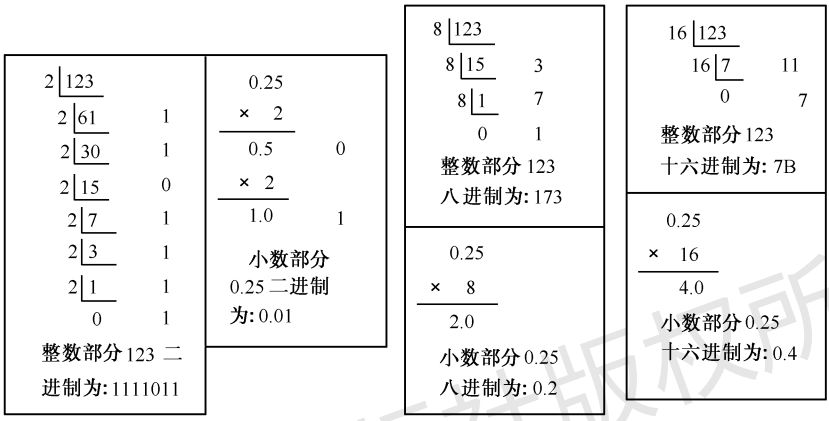

图 1.3　十进制转换为其他进制的一般方法

3. 进制快速转换方法

掌握进制快速转换方法的前提是记住 16 的倍数或 2 的 n 次方，如表 1.2 所示。

表 1.2　2 的指数及 16 的倍数表

n 的值	2^n	n 的值	$16\times n$	十六进制
−4	0.0625	1	16	10H
−3	0.125	2	32	20H
−2	0.25	3	48	30H
−1	0.5	4	64	40H
0	1	5	80	50H
1	2	6	96	60H
2	4	7	112	70H
3	8	8	128	80H
4	16	9	144	90H
5	32	10	160	A0H
6	64	11	176	B0H
7	128	12	192	C0H
8	256	13	208	D0H
9	512	14	224	E0H
10	1K(1 024)	15	240	F0H

（续表）

n 的值	2^n	n 的值	$16\times n$	十六进制
14	16K	1×16	256	100H
16	64K	2×16	512	200H
20	1M(1 024K)	3×16	768	300H
24	16M	4×16	1024(1K)	400H
30	1G(1 024M)	8×16	2 048(2K)	800H
40	1T(1 024G)	1×16×16	4 096(4K)	1 000H

记住表 1.2 的主要数据后，再结合图 1.4 及图 1.2 就可以在 10s 内完成进制转换。

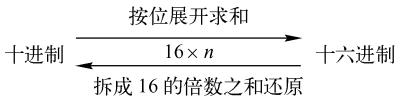

图 1.4　十进制与十六进制的快速转换思路

具体方法为：

将十进制转换为十六进制，只要把它拆成 16 的倍数之和(注：有时视情况可用 16 的倍数之差)还原成十六进制即可，再利用图 1.2 一展四转换为二进制，而后再用三合一转换为八进制。

例 1.5　280＝256＋16＋8＝118H＝100 011 000B＝430Q

例 1.6　2 000＝2 048－48＝800H－30H＝7D0H＝11 111 010 000B＝3 720Q

例 1.7　5 000＝4 096＋768＋128＋8＝1 388H＝1 001 110 001 000B＝11 610Q

也可先将十进制转换为二进制，只要把它拆成 2 的 n 次方之和(注：有时视情况可用 2 的 n 次方之差)，有 n 次方的二进制位写成 1，无 n 次方的二进制位写成 0 即可，再利用图 1.2 四合一转换为十六进制及用三合一转换为八进制。

例 1.8　$280=2^8+2^4+2^3=100011000B=118H=430Q$

例 1.9　$2000=2^{10}+2^9+2^8+2^7+2^6+2^4=11111010000B=7D0H=3720Q=2^{11}-2^5-2^4$

例 1.10　$5000=2^{12}+2^9+2^8+2^7+2^3=1001110001000B=1388H=11610Q$

4. 进制计算

进制计算主要有加、减、乘、除等算术运算及与、或、非等逻辑运算。其他进制加、减、乘、除等算术运算的运算方法与十进制的运算方法类似，要点是逢 N 进一、借一等于 N。与、或、非等逻辑运算一般是指变量取值为二值(0 或 1)的逻辑运算，将 1 当成真，将 0 当成假，与、或、非的真值表如图 1.5 所示。

A 与 B

B \ A	0	1
0	0	0
1	0	1

A 或 B

B \ A	0	1
0	0	1
1	1	1

非 A

A	0	1
$\overline{A}$	1	0

图 1.5　三种位逻辑运算真值表

在本书 3.3 节的汇编指令部分和 4.2 节的表达式部分将给出具体举例。

1.2　码制及其转换

本节介绍计算机主要使用的二进制编码，重点讲解真值(有符号数)与补码(或无符号数)间的快速转换方法。此方法使得 8 位或 16 位二进制补码的求解及有无符号数之间的转换变得轻而易举。

1. BCD 码

常见的 BCD 码有 8421 码、2421 码以及余 3 码等，一般使用 8421 码，它又分为压缩 BCD 码和非压缩 BCD 码。压缩 BCD 码是用 4 位二进制代码表示一位十进制，一个字节可以表示两位十进制(00～99)；而非压缩 BCD 码是用 8 位二进制代码中的低 4 位表示一位十进制、高 4 位无效，一个字节只能表示一位十进制(0～9)，高 4 位为 0 时则叫标准非压缩 BCD 码。例如，十进制数 35 的压缩 BCD 码为 35H，其标准非压缩 BCD 码为 0305H。它们的比较示意图如图 1.6 所示。

图 1.6　三种 8421 BCD 码的比较

2. ASCII 码

ASCII 码使用 8 位二进制编码，占一个字节，最高位为 0 的 ASCII 码称为基本 ASCII 码。重要的 8 个字符的 ASCII 码值如表 1.3 所示，其他字符参看附录 E。

'0'～'9' 的 ASCII 码依次加 1，'A'～'Z' 的 ASCII 码依次加 1，'a'～'z' 的 ASCII 码也是依次加 1，所以记住'0'、'A' 以及'a' 的 ASCII 码，也就记住了 62 个字符的 ASCII 码。'0'～'9' 的 ASCII 码是一种特殊的非压缩 BCD 码。例如'35' 是十进制数 35 的非压缩 BCD 码即 3335H。

表 1.3　重要的 ASCII 字符

字　　符	ASCII 码 十进制值	ASCII 码 十六进制值
NUL(空)	0	00H
LF(换行)	10	0AH
CR(回车)	13	0DH
SP(空格)	32	20H
'$'	36	24H
'0'～'9'	48～57	30H～39H
'A'～'Z'	65～90	41H～5AH
'a'～'z'	97～122	61H～7AH

3. 汉字内码

汉字在计算机及相关设备内存储、处理以及传输所用的编码称为汉字内码。我国目前主要采用的是国标内码(GB2312)，它在计算机内占用两个字节，每个字节的最高位为 1，最多可表示 $2^{14}=16\,384$ 个可区别代码。它与国标区位码的计算关系为：国标内码＝国标码(十六进制)＋8080H＝国标区位码(十六进制)＋A0A0H。GB2312—80 中有：一级汉字 3 755 个、按拼音顺序排列，二级汉字 3 008 个、按偏旁笔画数排列，字符 682 个。中国香港地区、中国台湾地区以及新加坡等繁体汉字区主要采用大五码(BIG5)，它在计算机内也是占用两个字节，每个字节的最高位也为 1。

为了统一表示世界上各国的文字，1993 年国际标准化组织公布了“通用多八位编码字符集”的国际标准 ISO/IEC10646，简称 UCS(Universal Code Set)，其中汉字部分叫 CJK(中、日、韩)统一汉字集。UCS 用 4 字节足以表示世界上所有的文字，包括英文、中文、日文、韩文、俄文以及法文等。我国的相应标准为 GB13000。

4. 原码、反码和补码

原码、反码和补码均为有符号数的编码，正、负号也用二进制编码来表示，它们所代表的实际数值称为“真值或原值”。

原码是直接在真值的绝对值之前增加一个符号位，并取正数的符号为 0，负数的符号为 1。正数的反码、补码与原码相同，负数的反码为原码的符号位不变其他位变反而得，负数的补码为原码的符号位不变其他位变反＋1 而得。负数的三种编码之间的转换关系如图 1.7 所示。

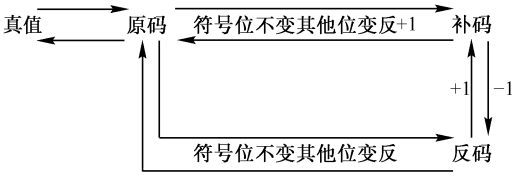

图 1.7　负数的原、反、补码之间转换关系图

补码是计算机中最基本的有符号数编码方案，最主要原因是因为采用补码后：减法可变加法如5－3＝5＋(－3)；加减时符号位如同数值位一样参加计算，具体例子请参看3.3.2节。

例1.11 (8位二进制数的原、反和补码)

－107 ＝－6BH＝－1101011B＝11101011B(原)＝10010100B(反)＝10010101B(补)

＝EBH(原)＝94H(反)＝95H(补)

107＝6BH(原)＝6BH(反)＝6BH(补)

5. 二进制数据的表示范围

二进制数据的表示范围要分有符号数还是无符号数。无符号数的所有二进制位(bit)均作为数值位；有符号数的最高位代表符号位，1代表负、0代表正，其余位才是数值位。n位二进制无符号数的表示范围为0～(2^n-1)。n位二进制有符号数的表示范围还取决于编码方案，补码为-2^{n-1}～$+(2^{n-1}-1)$；原码、反码的表示范围为$-(2^{n-1}-1)$～$+(2^{n-1}-1)$。计算机中内外存容量以字节(B,Byte)为单位，一个字节由8个二进制位构成(即1 B＝8 b)。8位二进制数(1字节)的无符号数表示范围为0～255，有符号补码表示范围为－128～＋127；16位二进制(2字节)的无符号数表示范围为0～65 535，有符号补码表示范围为－32 768～＋32 767。

6. 真值与补码(无符号数)之间的直接转换

正数的真值与补码(无符号数)完全相同，负数的真值与补码(无符号数)之间的直接转换方法如图1.8所示(0在用n位二进制补码表示时也代表2^n，即$0=2^n$)。

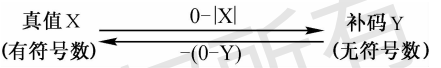

图1.8　负数的真值与补码之间转换关系图

例1.12　8位二进制时：

20＝14H(补)＝20(无)

－5＝0－5＝00H－05H＝FBH(补)＝251(无)＝256－5＝2^8－5

－120＝0－120＝00H－78H＝88H(补)＝136(无)＝256－120＝2^8－120

F8H(补)＝248(无)＝－(00H－F8H)＝－08H＝－(256－248)＝－8(有)

5CH(补)＝92(无)＝92(有)

16位二进制时：

20＝0014H(补)＝20(无)

－5＝0－5＝0 000H－0 005H＝FFFBH(补)＝65 531(无)＝65536－5＝2^{16}－5

－120＝0－120＝0 000H－78H＝FF88H(补)＝65 416(无)＝65536－120＝2^{16}－120

FFC6H(补)＝0 000H－(0 000H－FFC6H)＝65 536－58＝65 478(无)

＝－(0－FFC6H)＝－3AH＝－58(有)＝－(65 536－65 478)＝－58(有)

048FH＝1 024＋128＋15＝1 167(无)＝1 167(有)

7. 定点数和浮点数

机器数的表示是受设备限制的。计算机一般是以字为单位进行数据的处理、存储和传递的。所以运算器中的加法器、累加器以及其他一些寄存器，都选择与字长相同的位数。字长一定，则计算机所能表示数的范围也就确定了。例如，使用8位字长的计算机，它可以表示无符号整数的表示范围为0～255，补码的有符号数表示范围为－128～127。如果运算数值超出机器数所能表示的范围，机器就需要进行相应处理。这种现象称为溢出。

计算机中的数，既有整数，也有小数。如何确定小数点的位置呢？通常有两种约定：一种是规定小数点位置固定不变，这时的机器数称为定点数；另一种是小数点位置可以浮动，这样的机器数称为浮点数。

(1) 定点数

对于定点数，小数点位置可以固定在符号位之后，这样的机器表示的全是定点小数。例如，假定机器字长为 16 位，符号位占 1 位，数值占有 15 位，于是-2^{-15}用机器数原码表示如图 1.9 所示。

其相当于十进制数为-2^{-15}。

小数点位置固定在数的最后，则该机器表示的全是定点整数。例如，假设机器字长为 16 位，符号位占有 1 位，数值部分占 15 位，图 1.10 表示的机器数相当于十进制数为+32 767。

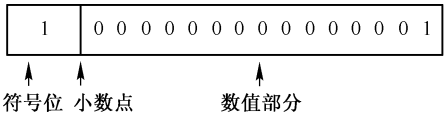

图 1.9　定点小数示意图

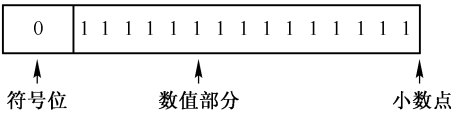

图 1.10　定点整数示意图

定点表示法表示的数值范围及精度有限，为了扩大定点数的表示范围或提高精度，可以采用多个字节来表示一个定点数，例如，采用 4 字节或 8 字节来表示。

(2) 浮点数

浮点数表示法就是小数点在数中的位置是浮动的。由于定点数表示的数的范围较窄，不能满足实际问题的需要，因此要采用浮点表示法。在同样字长情况下，浮点表示法能表示数的范围扩大了。

浮点表示法包括两部分：一部分是阶码，另一部分是尾数。浮点数在机器中的表示方法如图 1.11 所示。

图 1.11　浮点数示意图

由尾数部分隐含的小数点位置可知，尾数的绝对值总是小于 1 的数，它给出该浮点数的有效数字，为了有更多位有效数字，一般用规范化小数表示，即尾数的绝对值大于等于 0.5、小于 1。尾数部分的数符确定该浮点数的正负。阶码总是整数，它确定小数点浮动的位数。若阶符为正，尾数的小数点向右移动；若阶符为负，则向左移动。即浮点数的值为：尾数$\times 2^{阶码}$。

当浮点数的尾数为零或者阶码为最小值时，机器通常规定，把该数看做 0，称为“机器零”。在浮点数的表示和运算中，当一个数的阶码大于机器所能表示的最大阶码时，产生“上溢”，当一个数的阶码小于机器所能表示的最小阶码时，产生“下溢”。

浮点数的取值范围如图 1.12 所示。

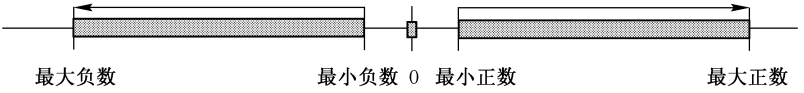

图 1.12　浮点数表示范围示意图

例 1.13　设阶码用 8 位补码表示，尾数部分用 16 位补码表示，则$-128.0625=-(2^7+2^{-4})=-(2^{-1}+2^{-12})\times 2^8=-0.100\ 000\ 000\ 001\ 000B\times 2^8$ 的尾数部分为-0.100 000 000 001 000B，补码为 1 011 111 111 111 000B；阶码部分为 8，即 00 001 000B，对应的十六进制数为 08BFF8H。

习题

1. 进制转换

129=_____ H=_____ B=_____ Q

298=_____ H=_____ B=_____ Q

1000=_____ H=_____ B=_____ Q

5DH=_____ B=_____ Q=_____ D

3E8H=_____ B=_____ Q=_____ D

357Q=_____ B=_____ H=_____ D

2. 进制计算

101101B+1101001B=_____ B

3FC9H−0FE6H=_____ H

一个字节的 NOT 8=_____ H=_____(有符号数)

两个字节的 NOT 8=_____ H=_____(有符号数)

5 AND 6=_____ D

5 OR 6=_____ D

3. 数据表示范围

一个字节的无符号数表示范围为_____,有符号数补码表示范围为_____。

两个字节的无符号数表示范围为_____,有符号数补码表示范围为 _____。

N 位二进制数的无符号数表示范围为_____,有符号数补码表示范围为_____。

4. 35H 代表的 ASCII 字符为_____,代表十六进制数时等价的十进制值为_____,代表压缩 8421 BCD 码等价的十进制值为_____,代表非压缩 8421 BCD 码等价的十进制值为_____。

5. FFH 代表无符号数时等价的十进制值为_____,代表补码有符号数时等价的十进制值为_____,代表反码有符号数时等价的十进制值为_____,代表原码有符号数时等价的十进制值为_____。

6. −20 的 8 位二进制补码为_____,原码为_____,反码为_____。

158 的 16 位二进制补码为_____,原码为_____,反码为_____。

7. 英文字符一般在计算机内占用_____个字节,每个字节的最高位一定为_____。全角英文字符在计算机内占用_____个字节,一个汉字在计算机内占用_____个字节,每个字节最高位为_____。

8. 设阶码用 8 位补码表示,尾数部分用 16 位补码表示,则−(1/32+1/128+1/512)的尾数部分及阶码分别为多少?