

80x86 汇编语言程序设计

沈美明 温冬婵 编著

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书由基础理论、编程工具、编程方法和实际应用 4 部分组成,共 13 章。第 1、2 章为基础理论部分,包括数制、码制等基础知识,计算机组成及基本原理;第 3、4 章介绍编程工具,包括指令系统、寻址方式、伪操作和汇编语言格式;第 5~9 章和第 13 章讲述编程方法,包括循环、分支、子程序等基本程序结构,宏汇编技术,中断等输入输出程序设计方法, BIOS 和 DOS 系统功能调用方法,以及多个模块的连接技术;第 10~12 章为实际应用部分,包括图形显示、发声和磁盘文件存取技术。

本书不仅可以作为高等院校“汇编语言程序设计”课程的教材,也可以供需用汇编语言的工程技术人员和科研人员使用。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

书 名: 80x86 汇编语言程序设计

作 者: 沈美明 温冬婵 编著

出版者: 清华大学出版社(北京清华大学学研大厦,邮编 100084)

<http://www.tup.tsinghua.edu.cn>

印刷者: 北京市清华园胶印厂

发行者: 新华书店总店北京发行所

开 本: 787×1092 1/16 印张: 39.25 字数: 978 千字

版 次: 2001 年 9 月第 1 版 2001 年 9 月第 1 次印刷

书 号: ISBN 7-302-04540-2/TP·2688

印 数: 0001~4000

定 价: 46.00 元

前 言

汇编语言是计算机能提供给用户的最快而又最有效的语言,也是能够利用计算机所有硬件特性并能直接控制硬件的惟一语言,因而在对于程序的空间和时间要求很高的场合,汇编语言是必不可少的。至于在很多需要直接控制硬件的应用场合,则更是非用汇编语言不可了。

本书是高等院校计算机科学与技术专业必修课“汇编语言程序设计”所用教材。它的前一版本《IBM-PC 汇编语言程序设计》(1991 年发行)曾被评为 1992 年第四届全国科技类优秀畅销书;获 1996 年电子工业部第三届工科电子类专业优秀教材一等奖,1999 年教育部科技进步一等奖,以及 1999 年国家科技进步三等奖。

在《IBM-PC 汇编语言程序设计》中,我们选用了以 8086 为 CPU 的 PC 机作为基础机型来组织教学。本书是在《IBM-PC 汇编语言程序设计》的基础上增加了有关技术发展的新内容,其中包括 8086 后继机型(80x86)所提供的指令及寻址方式、汇编程序 MASM 新版本所提供的伪操作及高级汇编语言技术、保护模式的编程基础等,以便满足广大读者使用高档微机的需要。本书适于初学者使用,只要有一种高级语言程序设计的基础,都可以通过学习本书掌握汇编语言程序设计技术。因此,本书不仅可以作为高等院校“汇编语言程序设计”课程的教材,也可以供需用汇编语言的工程技术人员和科研人员使用。

全书由基础理论、编程工具、编程方法和实际应用 4 部分共 13 章组成。第 1、2 章为基础理论部分,包括数制、码制等基础知识,计算机组成及基本原理。第 3、4 章介绍编程工具,包括指令系统、寻址方式、伪操作和汇编语言格式。第 5 章至第 9 章以及第 13 章说明编程方法,包括循环、分支、子程序等基本程序结构,宏汇编技术,中断等输入输出程序设计方法, BIOS 和 DOS 系统功能调用方法,以及多个模块的连接技术。第 10 章至第 12 章为实际应用部分,包括图形显示、发声和磁盘文件存取技术。这 4 个组成部分构成一个完整的系统。书中提供了大量程序例题,每章之后均有若干习题,便于读者复习及检查学习效果。同时为了能适应各种类型院校的不同要求,各章之间相互配合而又自成体系,易于为不同类型院校按其要求适当加以裁剪。所以本教材的适用面是比较宽的。

本书为清华大学计算机科学与技术系“汇编语言程序设计”课程的教材。该课程课内 80 学时,其中讲课 48 学时,上机实践 32 学时,课内外学时比例为 1:1.5。讲课内容为第 1 至第 9 章和第 13 章,第 10 至第 12 章结合实验由学生自学并上机。采用本教材的各校可根据教学计划规定的学时灵活安排。为便于查阅,本书把指令系统集中在第 3 章,因此,所占篇幅较大。在讲课过程中,为使学生尽可能早些上机,开始编程训练,可把有关指令分散到其后各章讲述。例如,把转移类指令放在循环与分支程序设计一章,把转子与返回指令放在子程序结构一章,把中断指令放在输入输出程序设计一章等。课程的上机安排可参考与本书配套的《IBM-PC 汇编语言程序设计实验教程》,根据课程上机时数及学生的水平,选用相

应的实验。

本书的第 1 章至第 7 章及第 13 章由沈美明编写,第 8 章至第 12 章由温冬婵编写。书中如有错误或不当之处,欢迎读者不吝批评指正。

编著者

2000 年 8 月

目 录

前言	I
第 1 章 基础知识	1
1.1 进位记数制与不同基数的数之间的转换	1
1.1.1 二进制数	1
1.1.2 二进制数和十进制数之间的转换	2
1.1.3 十六进制数及其与二进制数、十进制数之间的转换	3
1.2 二进制数和十六进制数运算	5
1.2.1 二进制数运算	5
1.2.2 十六进制数运算	5
1.3 计算机中数和字符的表示	6
1.3.1 数的补码表示	6
1.3.2 补码的加法和减法	8
1.3.3 无符号整数	10
1.3.4 字符表示法	10
1.4 几种基本的逻辑运算	11
1.4.1 “与”运算(AND)	11
1.4.2 “或”运算(OR)	12
1.4.3 “非”运算(NOT)	12
1.4.4 “异或”运算(XOR, exclusive-OR)	12
习题	13
第 2 章 80x86 计算机组织	14
2.1 80x86 微处理器	14
2.2 基于微处理器的计算机系统构成	16
2.2.1 硬件	16
2.2.2 软件	17
2.3 中央处理机	18
2.3.1 中央处理机(CPU)的组成	18
2.3.2 80x86 寄存器组	19
2.4 存储器	23
2.4.1 存储单元的地址和内容	23
2.4.2 实模式存储器寻址	25
2.4.3 保护模式存储器寻址	29

2.5 外部设备	34
习题	35
第3章 80x86的指令系统和寻址方式	38
3.1 80x86的寻址方式	39
3.1.1 与数据有关的寻址方式	39
3.1.2 与转移地址有关的寻址方式	47
3.2 80x86机器语言指令概况	49
3.2.1 操作码的机器语言表示	50
3.2.2 寻址方式的机器语言表示	50
3.2.3 加法的机器指令举例	52
3.2.4 指令的执行时间	55
3.2.5 32位指令格式简介	57
3.3 80x86的指令系统	58
3.3.1 数据传送指令	58
3.3.2 算术指令	69
3.3.3 逻辑指令	86
3.3.4 串处理指令	92
3.3.5 控制转移指令	100
3.3.6 处理机控制与杂项操作指令	120
习题	123
第4章 汇编语言程序格式	134
4.1 汇编程序功能	134
4.2 伪操作	135
4.2.1 处理器选择伪操作	135
4.2.2 段定义伪操作	135
4.2.3 程序开始和结束伪操作	143
4.2.4 数据定义及存储器分配伪操作	144
4.2.5 表达式赋值伪操作 EQU	149
4.2.6 地址计数器与对准伪操作	150
4.2.7 基数控制伪操作	152
4.3 汇编语言程序格式	152
4.3.1 名字项	153
4.3.2 操作项	154
4.3.3 操作数项	154
4.3.4 注释项	159
4.4 汇编语言程序的上机过程	161
4.4.1 建立汇编语言的工作环境	161

4.4.2	建立 ASM 文件	162
4.4.3	用 MASM 程序产生 OBJ 文件	163
4.4.4	用 LINK 程序产生 EXE 文件	166
4.4.5	程序的执行	167
4.4.6	COM 文件	169
习题	170
第 5 章	循环与分支程序设计	175
5.1	循环程序设计	175
5.1.1	循环程序的结构形式	175
5.1.2	循环程序设计方法	176
5.1.3	多重循环程序设计	186
5.2	分支程序设计	191
5.2.1	分支程序的结构形式	191
5.2.2	分支程序设计方法	191
5.2.3	跳跃表法	195
5.3	如何在实模式下发挥 80386 及其后继机型的优势	200
5.3.1	充分利用高档机的 32 位字长特性	200
5.3.2	通用寄存器可作为指针寄存器	204
5.3.3	与比例因子有关的寻址方式	204
5.3.4	各种机型提供的新指令	207
习题	209
第 6 章	子程序结构	212
6.1	子程序的设计方法	212
6.1.1	过程定义伪操作	212
6.1.2	子程序的调用和返回	214
6.1.3	保存与恢复寄存器	214
6.1.4	子程序的参数传送	215
6.1.5	增强功能的过程定义伪操作	228
6.2	嵌套与递归子程序	233
6.2.1	子程序的嵌套	233
6.2.2	递归子程序	234
6.3	子程序举例	241
6.4	DOS 系统功能调用	255
习题	256
第 7 章	高级汇编语言技术	261
7.1	宏汇编	261

7.1.1	宏定义、宏调用和宏展开	261
7.1.2	宏定义中的参数	263
7.1.3	LOCAL 伪操作	268
7.1.4	在宏定义内使用宏	269
7.1.5	列表伪操作	271
7.1.6	宏库的建立与调用	274
7.1.7	PURGE 伪操作	276
7.2	重复汇编	277
7.2.1	重复伪操作	277
7.2.2	不定重复伪操作	279
7.3	条件汇编	281
7.3.1	条件伪操作 IF 的使用举例	282
7.3.2	条件伪操作 IF1 的使用举例	284
7.3.3	条件伪操作 IFNDEF 的使用举例	285
7.3.4	条件伪操作 IFB 的使用举例	290
7.3.5	条件伪操作 IFIDN 的使用举例	291
7.4	高级语言结构	293
7.4.1	.IF/.ELSEIF/.ELSE/.ENDIF	293
7.4.2	.WHILE/.ENDW	295
7.4.3	.REPEAT/.UNTIL 和 .REPEAT/.UNTILCXZ	296
7.4.4	.BREAK 和 .CONTINUE	298
7.4.5	高级语言结构中使用的表达式	300
	习题	301
第 8 章	输入输出程序设计	305
8.1	I/O 设备的数据传送方式	305
8.1.1	CPU 与外设	305
8.1.2	直接存储器存取方式	305
8.2	程序直接控制 I/O 方式	306
8.2.1	I/O 端口	306
8.2.2	I/O 指令	307
8.2.3	I/O 程序举例	308
8.3	中断传送方式	312
8.3.1	8086 的中断分类	313
8.3.2	中断向量表	315
8.3.3	中断过程	319
8.3.4	中断优先级的中断嵌套	320
8.3.5	中断处理程序	322
8.3.6	中断程序举例	323

8.4	80386 输入输出	335
8.4.1	80386 I/O 操作	335
8.4.2	I/O 允许位图	336
8.5	80386 的中断处理	336
8.5.1	80386 的中断和异常	337
8.5.2	实地址下的中断处理	339
8.5.3	保护方式下的中断处理	340
8.5.4	虚拟 8086 方式下的中断处理	342
	习题	343
第 9 章	BIOS 和 DOS 中断	345
9.1	键盘 I/O	347
9.1.1	字符码与扫描码	347
9.1.2	BIOS 键盘中断	348
9.1.3	DOS 键盘功能调用	349
9.2	显示器 I/O	354
9.2.1	字符属性	354
9.2.2	BIOS 显示中断	357
9.2.3	DOS 显示功能调用	365
9.3	打印机 I/O	366
9.3.1	DOS 打印功能	367
9.3.2	打印机的控制字符	368
9.3.3	BIOS 打印功能	372
9.4	串行通信口 I/O	375
9.4.1	串行通信接口	375
9.4.2	串行口功能调用	378
9.4.3	串行通信口中断	383
	习题	392
第 10 章	彩色图形程序设计	394
10.1	显示方式	394
10.1.1	显示分辨率	394
10.1.2	BIOS 设置显示方式	395
10.1.3	确定显示适配器	397
10.2	视频显示存储器	398
10.2.1	图形存储器映象	398
10.2.2	数据到颜色的转换	401
10.2.3	直接视频显示	402
10.3	EGA/VGA 图形程序设计	405

10.3.1	读写像素	406
10.3.2	图形方式下的文本显示	411
10.3.3	彩色绘图程序	414
10.4	计算机动画	418
10.4.1	动画显示技术	419
10.4.2	交互式动画	422
10.4.3	游戏程序实例	424
	习题	445
第 11 章	发声系统的程序设计	447
11.1	可编程内部定时器 8253/54	447
11.1.1	编程结构	447
11.1.2	操作模式	447
11.1.3	控制字	450
11.1.4	IBM PC 8253/54 定时器的使用	451
11.2	通用发声程序	452
11.2.1	扬声器驱动方式	452
11.2.2	通用发声程序 GENSOUND	453
11.2.3	80x86 PC 的时间延迟	456
11.3	乐曲程序	457
11.3.1	音调与频率和时间的关系	457
11.3.2	演奏乐曲的程序	458
11.3.3	键盘控制发声程序	460
11.4	报警程序	467
	习题	472
第 12 章	磁盘文件存取技术	474
12.1	磁盘的记录方式	474
12.1.1	磁盘记录信息的地址	474
12.1.2	磁盘系统区和数据区	476
12.1.3	磁盘目录及文件分配表	476
12.2	文件代号式磁盘存取	478
12.2.1	路径名和 ASCII 串	479
12.2.2	文件代号和错误返回代码	479
12.2.3	文件属性	480
12.2.4	写磁盘文件	481
12.2.5	读磁盘文件	486
12.2.6	移动读写指针	491
12.3	字符设备的文件代号式 I/O	497

12.4	利用文件控制块的磁盘存取方式	500
12.4.1	文件控制块	501
12.4.2	建立磁盘文件	503
12.4.3	顺序读磁盘文件	509
12.4.4	随机存取磁盘文件	513
12.4.5	绝对磁盘 I/O	516
12.5	BIOS 磁盘存取功能	517
12.5.1	BIOS 磁盘操作	517
12.5.2	状态字节	519
12.5.3	BIOS 磁盘操作举例	519
	习题	522
第 13 章	模块化程序设计	524
13.1	汇编程序概述	524
13.1.1	汇编程序的主要工具	524
13.1.2	汇编过程	526
13.1.3	几个问题	527
13.2	连接程序及连接对程序设计的要求	529
13.2.1	连接程序的主要功能	529
13.2.2	连接对程序设计的要求	530
13.3	汇编语言程序与高级语言程序的连接	547
13.3.1	直接插入法	547
13.3.2	C 语言程序调用汇编语言过程法	548
13.4	模块化程序设计概述	552
13.4.1	模块化程序设计	553
13.4.2	结构化程序设计	555
13.4.3	程序设计举例	556
	习题	564
	参考文献	569
	附录 1 80x86 指令系统一览表	570
	附录 2 伪操作与操作符表	588
	附录 3 中断向量地址一览表	603
	附录 4 DOS 系统功能调用 (INT 21H)	605
	附录 5 BIOS 功能调用	611

第 1 章 基础知识

1.1 进位记数制与不同基数的数之间的转换

1.1.1 二进制数

进位记数制是一种记数的方法,习惯上最常用的是十进制记数法。一个任意的十进制数可以表示为:

$$a_n a_{n-1} \cdots a_i a_{i-1} \cdots a_0 \cdot b_1 b_2 \cdots b_j b_{j+1} \cdots b_m$$

其含义是

$$a_n \cdot 10^n + a_{n-1} \cdot 10^{n-1} + \cdots + a_0 \cdot 10^0 + b_1 \cdot 10^{-1} + b_2 \cdot 10^{-2} + \cdots + b_m \cdot 10^{-m}$$

其中 $a_i (i = 0, 1, \cdots, n), b_j (j = 1, 2, \cdots, m)$ 是 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 十个数码中的一个。

十进制数的基数为 10, 即其数码的个数为 10, 且遵循逢十进一的规则。上式中相应于每位数字的 10^k 称为该位数字的权, 所以每位数字乘以其权所得到的乘积之和即为所表示数的值。例如:

$$12345.67 = 1 \times 10^4 + 2 \times 10^3 + 3 \times 10^2 + 4 \times 10^1 + 5 \times 10^0 + 6 \times 10^{-1} + 7 \times 10^{-2}$$

十进制数是人们最熟悉、最常用的一种数制, 但它不是惟一的数制。例如计时用的时、分、秒就是按六十进制记数的。基数为 r 的 r 进制数的值可表示为:

$$a_n \cdot r^n + a_{n-1} \cdot r^{n-1} + \cdots + a_0 \cdot r^0 + b_1 \cdot r^{-1} + b_2 \cdot r^{-2} + \cdots + b_m \cdot r^{-m}$$

其中 a_i, b_j 可以是 0, 1, $\cdots, r-1$ 中的任一个数码, r^k 则为各位数相应的权。

计算机中为便于存储及计算的物理实现, 采用了二进制数。二进制数的基数为 2, 只有 0, 1 两个数码, 并遵循逢 2 进 1 的规则, 它的各位权是以 2^k 表示的, 因此二进制数 $a_n a_{n-1} \cdots a_0 \cdot b_1 b_2 \cdots b_m$ 的值是:

$$a_n \cdot 2^n + a_{n-1} \cdot 2^{n-1} + \cdots + a_0 \cdot 2^0 + b_1 \cdot 2^{-1} + b_2 \cdot 2^{-2} + \cdots + b_m \cdot 2^{-m}$$

其中 a_i, b_j 为 0, 1 两个数码中的一个。例如:

$$101101_2 = 1 \times 2^5 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^0 = 45_{10}$$

其中数的下标表示该数的基数 r , 即二进制的 101101 与十进制的 45 等值。

n 位二进制数可以表示 2^n 个数。例如 3 位二进制数可以表示 8 个数, 它们是:

二进制数	000	001	010	011	100	101	110	111
相应的十进制数	0	1	2	3	4	5	6	7

而 4 位二进制数则表示十进制的 0~15 共 16 个数, 如下所示:

二进制数	0000	0001	0010	0011	0100	0101	0110	0111
相应的十进制数	0	1	2	3	4	5	6	7
二进制数	1000	1001	1010	1011	1100	1101	1110	1111
相应的十进制数	8	9	10	11	12	13	14	15

为便于人们阅读及书写,经常使用八进制数或十六进制数来表示二进制数。它们的基数和数码如表 1.1 所示。

表 1.1 几种常用的进位记数制的基数和数码

进位记数制	基数	数 码
十六进制数	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
十进制数	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
八进制数	8	0, 1, 2, 3, 4, 5, 6, 7
二进制数	2	0, 1

按同样的方法,读者可以很容易地掌握八进制和十六进制数的表示方法。可以看出, 23_{10} 可以表示为 17_{16} 、 27_8 及 10111_2 , 1.375_{10} 可以表示为 1.6_{16} 、 1.3_8 及 1.011_2 等。在计算机里,通常用数字后面跟一个英文字母来表示该数的数制。十进制数一般用 D(decimal)、二进制数用 B(binary)、八进制数用 O(octal)、十六进制数用 H(hexadecimal)来表示。例如: $117D$, $1110101B$, $0075H$,…。当然也可以用这些字母的小写形式,本书的后面就采用了这种表示方法。

1.1.2 二进制数和十进制数之间的转换

1. 二进制数转换为十进制数

各位二进制数码乘以与其对应的权之和即为该二进制数相对应的十进制数。例如:

$$1011100.10111B = 2^6 + 2^4 + 2^3 + 2^2 + 2^{-1} + 2^{-3} + 2^{-4} + 2^{-5} = 92.71875D$$

2. 十进制数转换为二进制数

十进制数转换为二进制数的方法很多,这里只说明比较简单的降幂法及除法两种。

(1) 降幂法

首先写出要转换的十进制数,其次写出所有小于此数的各位二进制权值,然后用要转换的十进制数减去与它最相近的二进制权值,如够减,则减去并在相应位记以 1;如不够减,则在相应位记以 0 并跳过此位;如此不断反复,直到该数为 0 为止。

例 1.1 $N=117D$, 小于 N 的二进制权为:

$$64 \quad 32 \quad 16 \quad 8 \quad 4 \quad 2 \quad 1$$

对应的二进制数是 $1 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1$

计算过程如下:

$$117 - 2^6 = 117 - 64 = 53 \quad (a_6 = 1)$$

$$53 - 2^5 = 53 - 32 = 21 \quad (a_5 = 1)$$

$$\begin{aligned}
21 - 2^4 &= 21 - 16 = 5 & (a_4 = 1) \\
& & (a_3 = 0) \\
5 - 2^2 &= 5 - 4 = 1 & (a_2 = 1) \\
& & (a_1 = 0) \\
1 - 2^0 &= 1 - 1 = 0 & (a_0 = 1)
\end{aligned}$$

所以 $N = 117D = 1110101B$ 。

例 1.2 $N = 0.8125D$, 小于此数的二进制权为:

$$0.5 \quad 0.25 \quad 0.125 \quad 0.0625$$

对应的二进制数是 $1 \quad 1 \quad 0 \quad 1$

计算过程如下:

$$\begin{aligned}
0.8125 - 2^{-1} &= 0.8125 - 0.5 = 0.3125 & (b_1 = 1) \\
0.3125 - 2^{-2} &= 0.3125 - 0.25 = 0.0625 & (b_2 = 1) \\
& & (b_3 = 0) \\
0.0625 - 2^{-4} &= 0.0625 - 0.0625 = 0 & (b_4 = 1)
\end{aligned}$$

所以 $N = 0.8125D = 0.1101B$ 。

(2) 除法

把要转换的十进制数的整数部分不断除以 2, 并记下余数, 直到商为 0 为止。

例 1.3 $N = 117D$

$$\begin{aligned}
117/2 &= 58 & (a_0 = 1) \\
58/2 &= 29 & (a_1 = 0) \\
29/2 &= 14 & (a_2 = 1) \\
14/2 &= 7 & (a_3 = 0) \\
7/2 &= 3 & (a_4 = 1) \\
3/2 &= 1 & (a_5 = 1) \\
1/2 &= 0 & (a_6 = 1)
\end{aligned}$$

所以 $N = 117D = 1110101B$ 。

对于被转换的十进制数的小数部分则应不断乘以 2, 并记下其整数部分, 直到结果的小数部分为 0 为止。

例 1.4 $N = 0.8125D$

$$\begin{aligned}
0.8125 \times 2 &= 1.625 & (b_1 = 1) \\
0.625 \times 2 &= 1.25 & (b_2 = 1) \\
0.25 \times 2 &= 0.5 & (b_3 = 0) \\
0.5 \times 2 &= 1.0 & (b_4 = 1)
\end{aligned}$$

所以 $N = 0.8125D = 0.1101B$ 。

1.1.3 十六进制数及其与二进制数、十进制数之间的转换

我们知道, 在计算机内部, 数的运算和存储都是采用二进制的。但是, 二进制数对于人

的阅读、书写及记忆都是很方便的。十进制数虽然是人们最熟悉的一种进位记数制,但它与二进制数之间并无直接的对应关系。为了便于人们对二进制数的描述,应该选择一种易于与二进制数相互转换的数制。显然,使用 2^n 作为基数的数制是能适合人们的这种要求的,常用的有八进制数和十六进制数,我们在这里主要介绍十六进制数。

1. 十六进制数的表示

计算机中存储信息的基本单位为一个二进制位(bit),它可以用来表示0和1两个数码。此外,由于计算机中常用的字符是采用由8位二进制数组成的一个字节(byte)来表示的,因此字节也成为计算机中存储信息的单位。计算机的字长一般都选为字节的整数倍,如16位、32位、64位等。一个字节由8位组成,它可以用两个4位组(又称半字节)来表示,所以用十六进制数来表示二进制数是比较方便的。

十六进制数的基数是16,共有16个数码,它们是0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F。其中A表示十进制的10,余类推。它们与二进制和十进制数的关系如下:

二进制数	0000	0001	0010	0011	0100	0101	0110	0111
十进制数	0	1	2	3	4	5	6	7
十六进制数	0	1	2	3	4	5	6	7
二进制数	1000	1001	1010	1011	1100	1101	1110	1111
十进制数	8	9	10	11	12	13	14	15
十六进制数	8	9	A	B	C	D	E	F

对应于十六进制数中各位的权是 16^k 。

2. 十六进制数和二进制数之间的转换

由于十六进制数的基数是2的幂,所以这两种数制之间的转换是十分容易的。一个二进制数,只要把它从低位到高位每4位组成一组,直接用十六进制数来表示就可以了。

例 1.5

0011	0101	1011	1111
3	5	B	F

亦即 0011010110111111B=35BFH

反之,把十六进制数中的每一位用4位二进制数表示,就形成相应的二进制数了。

例 1.6

A	1	9	C
1010	0001	1001	1100

亦即 A19CH=1010000110011100B

3. 十六进制数和十进制数之间的转换

各位十六进制数与其对应权值的乘积之和即为与此十六进制数相对应的十进制数。

例 1.7 $N = \text{BF3CH}$

$$= 11 \times 16^3 + 15 \times 16^2 + 3 \times 16^1 + 12 \times 16^0$$

$$= 11 \times 4096 + 15 \times 256 + 3 \times 16 + 12 \times 1$$

$$= 48956\text{D}$$

十进制数转换为十六进制数也可使用降幂法和除法。

(1) 降幂法

首先写出要转换的十进制数,其次写出小于该数的十六进制权值,然后找出该数中包含

多少个最接近它的权值的倍数,这一倍数即相应位的值,用原数减去此倍数与相应位权值的乘积得到一个差值,再用此差值去找低一位的权值的倍数,如此反复直到差值为0为止。

例 1.8 $N=48956D$ 小于 N 的十六进制权值为

4096 256 16 1

对应的十六进制数 B F 3 C

计算过程如下:

$$48956 - 11 \times 4096 = 3900$$

$$3900 - 15 \times 256 = 60$$

$$60 - 3 \times 16 = 12$$

$$12 - 12 \times 1 = 0$$

所以 $N = 48956D = (11)(15)(3)(12)$

$= BF3CH$

(2) 除法

把要转换的十进制数的整数部分不断除以 16,并记下余数,直到商为 0 为止。

例 1.9 $N=48956D$

$$48956/16=3059 \quad (a_0=12)$$

$$3059/16=191 \quad (a_1=3)$$

$$191/16=11 \quad (a_2=15)$$

$$11/16=0 \quad (a_3=11)$$

所以 $N=48956D=BF3CH$ 。

对于要转换的十进制数的小数部分,则应不断地乘以 16,并记下其整数部分,直到结果的小数部分为 0 为止。由于其方法与二、十进制数的转换方法是相同的,这里不再举例说明。显然,为把一个十进制数转换为二进制数,可以先把该数转换为十六进制数,然后再转换为二进制数,这样可以减少计算次数;反之,要把一个二进制数转换为十进制数,也可采用同样的方法。

1.2 二进制数和十六进制数运算

1.2.1 二进制数运算

加法规则:

$$0+0=0$$

$$0+1=1$$

$$1+0=1$$

$$1+1=0(\text{进位 } 1)$$

乘法规则:

$$0 \times 0 = 0$$

$$0 \times 1 = 0$$

$$1 \times 0 = 0$$

$$1 \times 1 = 1$$

1.2.2 十六进制数运算

十六进制数的运算可以采用先把该十六进制数转换为十进制数,经过计算后再把结果