21世纪高职高专规划教材

汇编语言程序设计

王成端 主 编
王宇晓 王 丰 李晓波 副主编

高等教育出版社

21 世纪高职高专规划教材

汇编语言程序设计

王成端　主编
王宇晓　王　丰　李晓波　副主编

高等教育出版社

内 容 提 要

本书以 8086/8088 系列微机为主要背景机，以 8086/8088 CPU 为基础，由浅入深地介绍了汇编语言程序设计相关知识。全书共 9 章，主要内容包括基础知识、8088/8086 的寻址方式和指令系统、汇编语言、基本程序设计、算术与非数值程序设计、输入/输出程序设计、DOS 与 BIOS 中断及程序设计、模块化程序设计和 80x86/Pentium 汇编语言介绍等。本书内容充实、重点突出，部分章节附有一定数量的实训项目，不同专业可根据需要选用。

本书适合作为高职高专计算机、自动化、机电类等专业的教材，也可作为工程技术人员参考用书。

本书所配教学电子教案及书中所有相关素材，均可从高等教育出版社网站上下载，网址为：

http://www.hep.edu.cn 或 http://cs.hep.com.cn

图书在版编目（CIP）数据

汇编语言程序设计 / 王成端主编. —北京：高等教育出版社，2003.9

ISBN 7-04-012926-4

Ⅰ.汇… Ⅱ.王… Ⅲ.汇编语言-程序设计

Ⅳ.TP313

中国版本图书馆 CIP 数据核字（2003）第 067018 号

出版发行	高等教育出版社	购书热线	010-64054588
社　　址	北京市西城区德外大街 4 号	免费咨询	800-810-0598
邮政编码	100011	网　　址	http://www.hep.edu.cn
总　　机	010-82028899		http://www.hep.com.cn
经　　销	新华书店北京发行所		
印　　刷	高等教育出版社印刷厂		
开　　本	787×1092　1/16	版　　次	2003 年 9 月第 1 版
印　　张	16	印　　次	2003 年 9 月第 1 次印刷
字　　数	390 000	定　　价	20.00 元

本书如有缺页、倒页、脱页等质量问题，请到所购图书销售部门联系调换。

前　言

汇编语言程序设计是计算机及相关专业的一门重要的专业基础课，是操作系统、计算机接口技术等课程的先修课。在众多的程序设计语言中，汇编语言是面向机器的低级语言。由于汇编语言可以直接对硬件资源进行编程，因而汇编语言程序具有更高的执行效率。与高级语言相比，它更适合于对执行速度和代码长度要求较高的场合。可见，汇编语言一方面可以帮助学生理解计算机工作原理，为后续课程打好基础，同时也可以用于实时控制及与硬件资源紧密结合的其他领域。

本书以 8086/8088 系列微机为主要背景机，以 8086/8088 CPU 为基础，由浅入深地介绍了汇编语言程序设计相关知识。同时，考虑到计算机本身的发展，书中最后一章专门介绍了 80x86 和 Pentium CPU 的新增指令，使学生了解最新的汇编语言知识。

全书共分 9 章。第 1 章讲述汇编语言有关基础知识，包括数据表示与运算、8086/8088 CPU 的结构及基本概念；第 2 章讲述 8086/8088 CPU 的寻址方式和指令系统；第 3 章讲述汇编语言的伪指令和汇编语言程序格式，包括 MASM 汇编程序和 DEBUG 调试程序的使用；第 4 章讲述基本程序设计，包括顺序程序设计、分支程序设计、循环程序设计和子程序设计；第 5 章讲述算术与非数值程序设计，包括算术运算和数据处理程序设计；第 6 章讲述输入/输出程序设计；第 7 章讲述 DOS 与 BIOS 中断及程序设计；第 8 章讲述模块化的程序设计，包括汇编语言与高级语言的接口设计；第 9 章讲述 80x86/Pentium 汇编语言介绍，包括 CPU 结构和新增指令。

全书内容根据教育部高职高专规划教材指导思想与原则的要求，充分考虑了高职高专学生的培养目标和教学特点。在内容的组织上，本着由浅入深、循序渐进的原则，注重基本知识和基本概念的介绍，结合实例重点介绍实用性较强的内容，对应用较少、难度过大的内容则少量介绍或不予介绍。全书注重层次，每一章的开头首先介绍本章主要内容，每一章的最后给出本章小结。使学生有的放矢，掌握所学内容。本书突出应用性，书中精选了一些实际应用例题，并介绍了汇编语言与高级语言混合编程的方法。目的是使学生学完本课程后能够用汇编语言解决一些实际问题。本书的另一个特色是理论教学与实践教学紧密结合，从第 3 章开始，每章都安排实训内容。理论部分主要讲述有关概念和程序设计方法，实训部分则给出这些基本知识的应用范例。由此做到理论教学与实践教学的同步融合，达到学以致用。

本书编者多年从事汇编语言教学与科研工作，对汇编语言的教学与应用有深刻的理解和丰富的经验。在内容的组织上结合了教学与科研等方面的经验，书中的许多例题就来自具体的科研项目，而且所有例题都已调试通过，其源程序代码可在高等教育出版社网站上下载。网址为：http://ww.hep.edu.cn 或 http://www.hep.com.cn。

本书由王成端担任主编，王宇晓、王丰、李晓波任副主编，其中王成端编写第 1、8、9 章，并负责全书的统稿，王宇晓编写第 4、5 章，王丰编写第 6、7 章，李晓波编写第 2、3 章。魏先民、边祥娟参与程序上机调试工作，刘磊、张风云参与部分文稿的录入、排版和绘图工作，在此表示感谢。本书编写前，对编写大纲进行了多次讨论，参与编写大纲讨论的有上海第二工业大学、深圳职业技

术学院、金陵科技学院、宁波高等专科学校、承德石油高等专科学校、华北航天工业学院、武汉职业技术学院、同济大学高等技术学院、上海电机技术高等专科学校、上海托普职业技术学院、浙江工商职业技术学院、南昌水利水电专科学校、西安联合大学、洛阳大学、河南机电高等专科学校、常熟高等专科学校、郑州经济管理干部学院、太原理工大学阳泉学院等。

山西运城学院的赵润林老师对全稿进行了仔细认真的审阅，并提出了许多宝贵意见。本书编写过程中，一直得到高等教育出版社计算机分社的大力支持和指导，在此一并表示衷心感谢。

由于作者水平有限，书中错误和不足之处在所难免，恳请读者批评指正。作者的 E - mail 为 wcd@wfu.edu.cn。

编 者

2003 年 3 月

策划编辑　雷顺加
责任编辑　雷顺加
封面设计　王凌波
责任印制　韩　刚

目　录

第1章

基础知识

本章导读

汇编语言是计算机系统提供给用户的最快、最有效的语言，也是能对硬件直接编程的语言。因此，对空间和时间要求很高的程序或需要直接控制硬件的程序，必须使用汇编语言进行程序设计。要掌握汇编语言，必须对计算机中的数据表示及运算、8088/8086 系统结构有一定的了解，本章中将介绍这些知识。

1.1 数据表示与运算

1.1.1 进位计数制与不同基数制之间的转换

1. 二进制数、八进制数和十六进制数

(1) 二进制数

日常生活中一般采用十进制进行计数，但计算机只能识别 0,1 代码，也就是说，计算机采用二进制进行计数。二进制数只有 0,1 两个数码，其基数为 2，遵循逢二进一的原则，它的第 k 位权以 2^k 表示。

二进制数 $a_n a_{n-1}\cdots a_0 . b_{-1} b_{-2}\cdots b_{-m}$ 的值是

$$a_n \times 2^n + a_{n-1} \times 2^{n-1} + \cdots + a_0 \times 2^0 + b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + \cdots + b_{-m} \times 2^{-m}$$

其中 a_i，b_j 为 0、1 两个数码中的一个。二进制的描述是在其尾部加注字母 B，例如：

$$10100101\text{B} = 1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$
$$= 128 + 32 + 4 + 1 = 165$$

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(见表 1-1)。

表 1-1 3 位二进制数与十进制数对应表

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

4位二进制数则表示十进制的0～15共16个数，如表1-2所示。

表1-2 4位二进制与十进制数的对应表

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

从上表可以看出，位数越多，二进制数越长，不便于人们阅读、书写和记忆，因此人们经常使用八进制数或十六进制数来表示二进制数。它们的基数和数码表示如表1-3所示。

(2) 八进制数

八进制数有0、1、2、3、4、5、6、7、8个数码，其基数为8，遵循逢八进一的原则，它的第k位权以8^k表示。八进制的描述是在其尾部加注字母O或Q。

八进制数 $a_n a_{n-1}\cdots a_0 . b_{-1} b_{-2}\cdots b_{-m}$的值是

$$a_n \times 8^n + a_{n-1} \times 8^{n-1} + \cdots + a_0 \times 8^0 + b_{-1} \times 8^{-1} + b_{-2} \times 8^{-2} + \cdots + b_{-m} \times 8^{-m}$$

例如：$534Q = 5 \times 8^2 + 3 \times 8^1 + 4 \times 8^0 = 5 \times 64 + 3 \times 8 + 4 \times 1 = 348$

(3) 十六进制数

十六进制数有0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F共16个数码，其中A、B、C、D、E、F表示10～15共6个数码，其基数为16，遵循逢十六进一的原则，它的第k位权以16^k表示。十六进制的描述是在其尾部加注字母H。

十六进制数 $a_n a_{n-1}\cdots a_0 . b_{-1} b_{-2}\cdots b_{-m}$的值是

$$a_n \times 16^n + a_{n-1} \times 16^{n-1} + \cdots + a_0 \times 16^0 + b^{-1} \times 16^{-1} + b_{-2} \times 16^{-2} + \cdots + b_{-m} \times 16^{-m}$$

例如：$2ACH = 2 \times 16^2 + 10 \times 16^1 + 12 \times 16^0 = 2 \times 256 + 10 \times 16 + 12 \times 1 = 684$

表1-3列出了几种常用的进位制的基数和数码。

表1-3 几种常用的进位制的基数和数码

进位计数制	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

2. 不同数制之间的转换

(1) 非十进制数转换为十进制数

各位非十进制数码乘以与其对应的权之和即为该数对应的十进制数。例如：

$1011100.1011B = 1 \times 2^6 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^{-1} + 1 \times 2^{-3} + 1 \times 2^{-4} = 92.6875\ D$

$A031H = 10 \times 16^3 + 3 \times 16^1 + 1 \times 16^0 = 41009$

$1001Q = 1 \times 8^3 + 1 \times 8^0 = 513$

(2) 十进制数转换为非十进制数

十进制数转换为非十进制数一般整数部分采用除基数取余法，小数部分采用乘基数取整法。除基数取余法的具体操作是把待转换的十进制数的整数部分不断除以要转换为的非十进制基数，并记下余数，直到商为0时为止；乘基数取整法的具体操作是把待转换的十进制数的小数部分不断乘以要转换为的非十进制基数，逐次记下乘积整数部分的值，直到小数部分为0时止。

现以十进制数转换为二进制数为例进行说明，十进制数转换为二进制数将“基数”换成2就成了除2取余法和乘2取整法，即对整数部分的处理是把待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时为止；对小数部分的转换是把待转换的十进制数的小数部分不断乘以2，逐次记下乘积整数部分的值，直到小数部分为0时止。

例1.1 $N = 137.8125D$ 转换为二进制数。

整数部分137D，按除2取余法有：

$137/2 = 68$ $(a_0 = 1)$

$68/2 = 34$ $(a_1 = 0)$

$34/2 = 17$ $(a_2 = 0)$

$17/2 = 8$ $(a_3 = 1)$

$8/2 = 4$ $(a_4 = 0)$

$4/2 = 2$ $(a_5 = 0)$

$2/2 = 1$ $(a_6 = 0)$

$1/2 = 0$ $(a_7 = 1)$

故 $137D = 10001001B$

小数部分为0.8125D，按乘2取整法有：

$0.8125 \times 2 = 1.625$ $(b_{-1} = 1)$

$0.625 \times 2 = 1.25$ $(b_{-2} = 1)$

$0.25 \times 2 = 0.5$ $(b_{-3} = 0)$

$0.5 \times 2 = 1.0$ $(b_{-4} = 1)$

故 $0.8125D = 0.1101B$

所以 $N = 137.8125D = 10001001.1101B$

十进制数转换为十六进制数和八进制数的方法与十进制数转换为二进制数的方法类同。

(3) 十六进制数与二进制数之间的转换

因为 $16 = 2^4$，所以一个十六进制数中的每一位可以用4位二进制数表示，便可形成相应的二进制数。

例1.2

C	B	9	A
1100	1011	1001	1010

即 $CB9AH = 1100101110011010B$

反之，二进制数只要把它从低位到高位每4位组成一组，再用十六进制数来表示就可以了。

例1.3

0111	0101	1011	1111
7	5	B	F

即 0111010110111111B = 75BFH

1.1.2 二进制数和十六进制数运算

1. 二进制数的运算

加法规则：

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 10(1 为进位)

乘法规则：

0 × 0 = 0

0 × 1 = 0

1 × 0 = 0

1 × 1 = 1

2. 十六进制数的运算

(1) 十六进制加法

十六进制数的运算按照逢十六进一的规则进行，即当两个一位数之和 S 小于 16 时，与十进制数同样处理，如两个一位数之和 $S \geqslant 16$ 时，则应该用 $S - 16$ 及进位 1 来取代 S。

例 1.4

```
  15C3H
+ 3D45H
------
  5308H
```

(2) 十六进制数的减法

与十进制数类似，够减时可以直接相减，不够减时服从向高位借 1 为 16 的规则。

例 1.5

```
  3DA6H
- 0FC3H
------
  2DE3H
```

1.1.3 数据表示

计算机内的数据有多种形式，其中最主要的是数值数据和字符数据。

1. 数值数据的表示

数值数据可以用不同的码制来表示，常用的有原码、补码和反码表示法。由于做加减运算时补码表示法的符号位可以参加运算，而且不影响运算结果的正确性，所以多数机器的有符号整数都采用补码表示法。这里只介绍补码表示法。

(1) 数的补码表示

① 补码表示法中正数的表示

正数采用符号 - 绝对值表示，即数的最高有效位为 0 表示符号为正，数的其余部分则表示数的绝对值。

例如：假设机器字长为 8 位，则

$[+0]_{补} = 00000000B$

$[+1]_{补} = 00000001B$

$[+100]_{补} = 01100100B$

② 补码表示法中负数的表示

用补码表示法来表示负数时可以采用“求反加 1”的方法来完成:先写出与该负数相对应的正数的补码表示(用符号 - 绝对值法),然后将其按位求反(即 0 变为 1,1 变为 0),最后在末位(最低位)加 1,就可以得到该负数的补码表示了。

例 1.6 机器字长为 8 位,写出 $N = -27$ 的补码表示。

+27D 可表示为	0001	1011
按位求反为	1110	0100
末位加 1 后为	1110	0101
用十六进制数表示为	E5H	

即 $[-27]_{补} = E5H$

例 1.7 机器字长为 16 位,写出 N = - 32768 的补码表示。

32768D 可表示为	1000	0000	0000	0000
按位求反为	0111	1111	1111	1111
末位加 1 后为	1000	0000	0000	0000
用十六进制数表示为	8	0	0	0

即 $[-32768]_{补} = 8000H$。

③ 数的表示范围

• 有符号数的表示范围

一般说来,n 位二进制补码的表示范围是

$$-2^{n-1} \leqslant N \leqslant 2^{n-1} - 1$$

8 位二进制数可以表示 $2^8 = 256$ 个数。因为在补码表示法中 0 只有一种表示,即 00000000;对于 10000000 这个数,在补码表示法中被定义为 - 128。这样,8 为补码能表示的范围为 - 128 ~ 127。

$n = 16$ 时的数的表示范围是

$$-32768 \leqslant N \leqslant +32767$$

• 无符号整数的表示范围

在做无符号数处理时,把最高有效位作为数值处理。因此,16 位无符号数的表示范围是 $0 \leqslant N \leqslant 65535$,8 位无符号数的表示范围是 $0 \leqslant N \leqslant 255$。

(2) 补码的运算

① 求补运算

已知 $[X]_{补}$,如何求 $[-X]_{补}$? 运算规则为:将 $[X]_{补}$ 各位(含符号位)取反,末位加 1,即:

$$[-X]_{补} = \overline{[X]_{补}} + 1$$

例 1.8 $[-117]_{补} = FF8BH$,求 $[+117]_{补}$。

对 $[-117]_{补}$ 做求补运算:

$[-117]_{补}$ 为	1111	1111	1000	1011
按位求反后得	0000	0000	0111	0100
末位加 1 后得	0000	0000	0111	0101

此数正是 $[+117]_{补} = 0075H$。

② 补码的加、减法运算

补码的加法规则是

$$[X+Y]_{补}=[X]_{补}+[Y]_{补}$$

补码的减法规则是

$$[X-Y]_{补}=[X]_{补}+[-Y]_{补}$$

其中的$[-Y]_{补}$只要对$[Y]_{补}$求补就可得到。

例 1.9 机器字长假定为 8 位,完成下列补码加法运算。

	十进制	二进制
①	23	00010111
	+36	+00100100
	59	00111011
②	36	00100100
	+(-23)	+11101001
	13	(进位 1)00001101
③	23	00010111
	+(-36)	+11011100
	-13	11110011
④	-23	11101001
	+(-36)	+11011100
	-59	(进位 1)11000101

可以看出,上述 4 个例子的计算结果都是正确的,在②和④中,从最高有效位向高位的进位由于机器字长的限制而自动丢失,但这并不会影响运算结果的正确性。

例 1.10 机器字长假定为 8 位,完成下列补码减法运算。

	十进制	补码	二进制
①	23	00010111	00010111
	-36	00100100	+11011100
	-13		11110011
②	36	00100100	00100100
	-(-23)	11101001	+00010111
	59		00111011
③	-23	11101001	11101001
	-(+36)	00100100	+11011100
	-59		(进位为 1) 11000101

④	-23	11101001	11101001
	-(-36)	11011100	+00100100
	13		(进位为1) 00001101

可以看出,补码减法是用对减数求补后把减法转换为加法进行的,它能自动地得到正确的结果。

2. 字符数据的表示

计算机内的常见字符包括:

字母: A、B、…、Z,a、b、…、z

数字: 0~9

标点符号: !、,、?、;、:、、等

算术运算符号: + 、- 、× 、/

关系运算符号: > 、< 、 =

专用字符: * 、$、@、%、# 、空格等

非打印字符: BEL(响铃)、LF(换行)、CR(回车)等

这些字符一般采用目前最常用的美国信息交换标准代码 ASCII 码来表示。这种代码用一个字节(8位二进制码)来表示一个字符,其中低7位为字符的 ASCII 值,最高位一般用做校验位。常见字符的 ASCII 码表见附录Ⅰ。

1.1.4 定点数与浮点数

计算机中根据小数点的位置是否固定,将数的表示方法分为定点数表示法和浮点数表示法。

1. 定点数

定点数是指小数点位置固定不变的数。小数点的位置通常只有两种约定,小数点约定在最低数位右面的称为定点整数,可用来表示一个纯整数。小数点约定在符号位右面,最高数位左面的数称为定点小数,可用来表示一个纯小数。

无符号定点整数,即正整数,不需要设符号位,所有各数位都用来表示数值大小,并约定小数点在最低数的右面。

在定点整数或定点小数的表示法中,参加运算的数以及运算的结果必须在该定点数所能表示的数值范围之内,否则溢出。当发生溢出时,CPU 中的状态标志寄存器中的溢出标志 OF 置1。

2. 浮点数

定点数的表示比较单一,要么纯整数,要么纯小数,表示数的范围比较小,运算过程中也很容易发生溢出。计算机中也引入了类似与十进制的科学标识法(如 1.23456×10^4)来表示二进制实数,这种方法用来表示值很大或很小的数,也可以用它来表示既有整数又有小数的数。这种表示法标为浮点表示法,其小数点的实际位置随指数(阶)的大小而浮动。

浮点数由两部分组成 阶码 E 和尾数 M。浮点数表示的数值为 $M \times R^E$。若尾数 M 为 m 位,阶码 E 为 e 位,则典型的浮点数格式如图 1-1 所示。

图 1-1 中,E 是阶码,即指数,为带符号定点整数,可用补码表示。E_s 阶符,表示阶的正负。阶为正,小数点实际位置向右浮动,阶为负,小数点实际位置向左浮动。

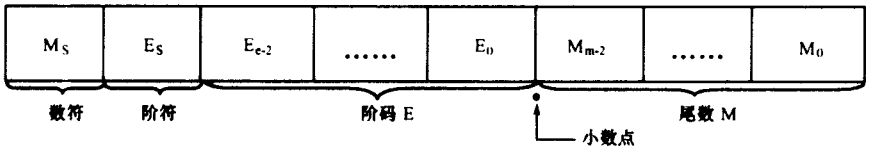

图 1-1 浮点数格式

M 是尾数，是带符号的定点小数，常用补码表示。M_s 是尾数的符号位，安排在最高位，表示该浮点数的正负。

小数点的位置约定在阶码最低位的右面，尾数最高数值位的左面，如图 1-1 所示。

R 是阶码的底，也就是尾数 M 的基(Radix)。一般基为 2，它是隐含约定的。

浮点数的表示范围主要由阶码的位数决定。精度则主要由尾数的位数决定。

1.2 8086/8088 系统结构

8086 微处理器是美国 Intel 公司 1987 年推出的一种高性能的 16 位微处理器。它采用硅栅 HMOS 工艺制造，在 1.45 cm^2 单个硅片上集成了 29 000 个晶体管。8086 的内部结构规模较小，采用 40 引脚的双列直插式封装。8086 的一个突出特点是多重处理能力，用 8086 CPU 与 8087 协处理器以及 8089 I/O 处理器组成的多处理器系统，可大大提高其数据处理和输入/输出能力。与 8086 配套的各种外围接口芯片非常丰富，方便用户开发各种系统。

1.2.1 8086/8088 CPU 的内部结构

8086 CPU 内部结构如图 1-2 所示。按功能可分为两大部分：总线接口单元 BIU(Bus Interface Unit)和执行单元 EU(Execution Unit)。

1. 总线接口单元 BIU

总线接口单元 BIU 是 8086 CPU 同存储器和 I/O 设备之间的接口部件，负责对全部引脚的操作，即 8086 所有对存储器和 I/O 设备的操作都是由 BIU 完成的。其具体任务是：负责从内存单元中预取指令，并将它们送到指令队列缓冲器暂存。CPU 执行指令时，总线接口单元要配合执行单元，从指定的内存单元或者 I/O 端口中取数据传送给执行单元，或者把执行单元的处理结果传送到指定的内存单元或 I/O 端口中。

总线接口单元 BIU 由 20 位地址加法器、4 个段寄存器、16 位指令指针 IP、指令队列缓冲器和总线控制逻辑电路等组成。

(1) 地址加法器和段寄存器

8086 CPU 的 20 条地址线，可直接寻址 1 MB 存储器物理空间。但 CPU 内部寄存器均为 16 位的寄存器。那么，16 位的寄存器如何实现 20 位地址寻址呢？它是由专门地址加法器将有关段寄存器内容(段的起始地址)左移 4 位后，与 16 位偏移地址相加，形成一个 20 位的物理地址，以对存储单元寻址。比如在取指令时，由 16 位指令指针(IP)提供一个有效地址(逻辑地址或偏移地