汇编语言程序设计

（第2版）

丁　辉　主编　　陈书谦　朱海峰　副主编

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http: //www.phei.com.cn

汇编语言程序设计（第2版）

丁　辉　主编

陈书谦　朱海峰　副主编

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书主要内容包括：微机基础知识， 8086/8088 指令系统，80x86、Pentium 增强和扩展指令，程序设计方法，高级汇编技术，系统功能调用和模块化程序设计知识，汇编语言与 C/C++的混合编程技术，以及基于 Windows 平台的 WIN32 汇编语言程序设计基本方法，上机操作方法。每章附有习题，书后附有上机实验指导。

本书可作为高等学校计算机专业或相近专业汇编语言程序设计课程教材，微型计算机原理课程辅助教材，亦可用做工程技术人员的参考书。

图书在版编目(CIP)数据

汇编语言程序设计 / 丁辉主编. —2 版. —北京：电子工业出版社，2005.8
ISBN 7-121-01308-8

Ⅰ. 汇… Ⅱ. 丁… Ⅲ. 汇编语言—程序设计 Ⅳ. TP313

中国版本图书馆 CIP 数据核字（2005）第 051795 号

责任编辑：吕 迈
印 刷：北京市李史山胶印厂
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编 100036
经 销：各地新华书店
开 本：787×1 092 1/16 印张：19.25 字数：493 千字
印 次：2005 年 8 月第 1 次印刷
印 数：6 000 册 定价：24.00 元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话：（010）68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

前　言

在众多程序设计语言中，汇编语言属于低级语言。这里所谓的低级，是指汇编语言不同于高级语言。高级语言面向用户，而汇编语言则面向机器。正因为如此，汇编语言可以充分利用计算机的硬件特性，用以编制对时间和空间要求很高的程序，在需要直接控制硬件的场合，汇编语言更是无可替代，由此决定了汇编语言程序设计是计算机专业及相近专业人员的必备知识。

本书以 Intel 8086/8088 系列微机作为基础机型介绍汇编语言程序设计知识。在介绍 8086/8088 CPU 寻址方式和指令系统的基础上，详细介绍了汇编语言程序设计的基本方法和技巧。掌握这些内容，可以为 Intel 80x86 及 Pentium 系列微机的汇编语言程序设计奠定基础。考虑到 Intel 80x86 及 Pentium 系列微机的广泛应用，本书专门设置了一章关于 80x86 及 Pentium 的增强和扩展指令内容，在介绍各种程序设计方法的例题中也兼顾了这些指令的应用。本书的前两章提供了学习汇编语言的基础知识；第 10 章到第 13 章提供了进行高效率、大规模汇编语言程序设计的必备知识；第 14 章和第 15 章则分别涉及与 C/C++的混合编程，以及 WIN32 汇编语言程序设计的基本方法，在讲课中这两章可酌情取舍。

本书以编者长期使用的该课程讲稿为主体，力求难点分散、循序渐进。为避免大量的汇编语言指令集中堆砌，将部分指令融于相关程序设计方法的介绍之中。对于同类内容讲透一点，以点带面。例题和习题的设置力图紧扣重点，举一反三。每项实验均设有验证和设计两种类型的实验题，以便读者在巩固书本知识的基础上，提高应用和创新的能力。

本书由丁辉主编，陈书谦、朱海峰为副主编。第 6 章到第 10 章，第 13 章，以及上机实验指导由丁辉编写；第 1 章到第 5 章由陈书谦编写；第 11 、12、14、15 章由朱海峰编写；全书由丁辉统稿。伍俊明，傅扬烈，姜宏岸，张丽虹，邵峥嵘，冯亚东，常赵罡为本书的编写提出了不少有益的建议，并参与了资料的整理工作。由于编者水平有限，书中难免存在疏漏，敬请同行、专家指正。

在本书的编写过程中，得到了电子工业出版社的热情支持，在此一并表示衷心的感谢。

编者

2005 年 1 月

目　录

第1章 基础知识

本章提供了学习汇编语言程序设计所需的一些基础知识。首先对汇编语言程序设计进行了概述，其次对计算机常用的几种数制及其相互间的转换方法进行了讲解，并且介绍了数值数据和非数值数据在计算机中的表示方法。

1.1 汇编语言程序设计概述

1.1.1 汇编语言

计算机程序设计语言是人机交流的重要工具，可分为机器语言、汇编语言和高级语言。

机器语言是机器指令的集合，是一种面向机器的程序设计语言。机器指令是由0和1构成的二进制代码，不同种类的计算机具有各自的机器语言。其优点是可为计算机直接接受，用其编写的机器语言程序执行速度快，占内存空间小，可充分利用计算机的硬件特性；缺点是指令难记，用其编写的机器语言程序难以阅读且通用性差。

高级语言是面向问题求解过程或面向对象的程序设计语言。典型的高级语言有 Pascal，FORTRAN，C++，Java 等。高级语言接近于人类的自然语言，而且通用于各种计算机。其优点是易学易记，用其编写的高级语言程序易读易改，通用性强；其缺点是高级语言程序需经过编译或解释方能被计算机接受，执行速度慢，占内存空间大，不能直接利用计算机的硬件特性。

汇编语言又称为符号语言，实际上是一种符号化的机器语言。它将机器指令的操作码、操作数由二进制代码改为人们所熟悉的符号，例如

```
ADD  AL，5
```

表示将数字 5 加到 AL 中。汇编语言程序需经过汇编才能为计算机接受，这一点不如机器语言方便。虽然所用符号为人们所熟悉，然而不如高级语言那样接近人类的自然语言，程序编写和交流较为困难。除此以外，汇编语言几乎具备了机器语言的所有优点，一定程度上弥补了机器语言的缺陷，而且不存在高级语言的上述缺点。可以认为，汇编语言是目前使用的惟一直接利用计算机硬件特性的程序设计语言。

1.1.2 汇编语言程序设计

汇编语言程序设计是指使用汇编语言设计程序的过程。为什么要学习汇编语言程序设计？其原因至少有以下几点：

1．通过汇编语言程序设计，人们可以高效地使用计算机解决现实问题。在解决同一现实问题时，汇编语言程序与高级语言程序相比，占用内存更小，执行速度更快。

2．通过汇编语言程序设计，人们可以直接利用计算机的硬件特性，准确计算解决某一问题所需的时间，从而可实现实时控制。这一点是高级语言程序难以替代的。

3．进行汇编语言程序设计，必然要从原理上认识、理解计算机的工作过程。因此，学习汇编语言程序设计不仅可以掌握一种高效的程序设计方法，而且对于学习计算机组成原理、

微机原理也大有帮助。

1.2 进位计数制

进位计数制是计数的一种方法。人们普遍习惯的进位计数制为十进制。十进制数的基为10，有10个数码：0、1、2、3、4、5、6、7、8及9，遵循逢十进一的规则。二进制数的基为2，有2个数码：0、1，遵循逢二进一的规则。以此类推，N进制数的基为N，有N个数码：0、1、2、……、N-1，遵循逢N进一的规则。常用的几种进位计数制的情况如表1.1所示。

表1.1 常用的进位计数制

数　制	基	数　码	尾　标
十六进制	16	0，1，2，3，4，5，6，7，8，9，A，B，C，D，E，F	H
十进制	10	0，1，2，3，4，5，6，7，8，9	D（或缺省）
八进制	8	0，1，2，3，4，5，6，7	Q（或O）
二进制	2	0，1	B

1.2.1 常用计数制及其数的算术运算

1. 十进制（Decimal）

首先观察以下十进制数的组成：

$$361.905D=3\times10^{2}+6\times10^{1}+1\times10^{0}+9\times10^{-1}+0\times10^{-2}+5\times10^{-3}$$

不难看出每一个数码根据它在这个数中所处的位置（数位）来决定实际数值。事实上，任意一个正的十进制数$S=K_{n-1}K_{n-2}\cdots K_0K_{-1}K_{-2}\cdots K_{-m+1}K_{-m}$都可以表示成以下形式：

$$S=K_{n-1}\times10^{n-1}+K_{n-2}\times10^{n-2}+\cdots+K_1\times10^{1}+K_0\times10^{0}+K_{-1}\times10^{-1}+\cdots+K_{-m+1}\times10^{-m+1}+K_{-m}\times10^{-m}$$

$$=\sum_{i=-m}^{n-1}k_i\times10^{i}$$

其中K_i可以是0至9这十个数码中的任意一个，m和n均为自然数，10^i称为权。一般而言，一个正的N进制数$S=K_{n-1}K_{n-2}\cdots K_0K_{-1}K_{-2}\cdots K_{-m+1}K_{-m}$可以表示为以下通用形式：

$$S=\sum_{i=-m}^{n-1}k_i\times N^{i}$$

其中K_i可以是0至N-1中任意一个数字，m和n均为自然数，N^i为各数位相应的权。

2. 二进制（Binary）

在日常生活中，存在着大量的两种对立的现象，例如：是和非，开和关，通和断。电子元件、物理器件中两种状态易于实现，如电压、电流的有和无，晶体管的导通和截止，这两种状态是非常稳定的，而找到八种、十种或十六种稳定的状态则要复杂得多，所以计算机中采用二进制作为进位数制，以便于存储及计算的物理实现。

二进制数与人们常用的十进制数的对照关系如表1.2所示。

一个二进制数也可以用上述通用形式来表示。例如：

$$11101.011B=1\times2^4+1\times2^3+1\times2^2+0\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2}+1\times2^{-3}$$

二进制数的算术运算与十进制数的算术运算类似，区别仅在于该运算遵循逢二进一的规则。

【例 1.1】 （1）1101B+1011B=11000B

$$\begin{array}{r} 1101 \\ +1011 \\ \hline 11000 \end{array}$$

（2）1101B－1011B=0010B

$$\begin{array}{r} 1101 \\ -1011 \\ \hline 0010 \end{array}$$

（3）1101B×1011B=10001111B

$$\begin{array}{r} 1101 \\ \times 1011 \\ \hline 1101 \\ 1101 \\ 0000 \\ 1101 \\ \hline 10001111 \end{array}$$

（4）1111B÷101B=11B

$$\begin{array}{r} 11 \\ 101\overline{)\,1111} \\ 101 \\ \hline 101 \\ 101 \\ \hline 0 \end{array}$$

表 1.2 常用进位计数制对照表

十 进 制	二 进 制	八 进 制	十六进制
0	000	0	0
1	001	1	1
2	010	2	2
3	011	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

3. 十六进制（Hexadecimal）

十六进制数采用 0~9，A~F 十六个数码，这里 A 表示 10D，B 表示 11D，C 表示 12D，D 表示 13D，E 表示 14D，F 表示 15D。

十六进制数与人们常用的十进制数的对照关系如表 1.2 所示。

一个十六进制数也可以用上述通用形式来表示。例如

$$6B.0CH=6\times16^1+B\times16^0+0\times16^{-1}+C\times16^{-2}$$

十六进制数的算术运算与十进制数的算术运算类似，区别仅在于该运算遵循逢十六进一的规则。

【例 1.2】 （1）8A04H+110CH=9B10H　　（2）8A04H-110CH=78F8H

	8A04		8A04
+	110C	-	110C
	9B10		78F8

在汇编语言程序中，数据通常采用十六进制形式，汇编语言的调试、列表文件中显示的数据也都是使用十六进制形式，所以有必要熟练掌握这种数制及其数的运算。

4. 八进制(Octal)

八进制数采用 0~7 八个数码，与十进制数的前八个数码一一对应，如表 1.2 所示。

一个八进制数也可以用上述通用形式来表示。例如

$$36.53Q=3\times8^{1}+6\times8^{0}+5\times8^{-1}+3\times8^{-2}$$

1.2.2 数制转换

1. 非十进制数→十进制数

非十进制数转换为十进制数的方法是：将非十进制数用上述通用形式表示，即按权展开，计算的结果即为十进制数。不管非十进制数是否带有小数部分，均可用这种方法转换。

【例 1.3】 $1010110B=1\times2^{6}+0\times2^{5}+1\times2^{4}+0\times2^{3}+1\times2^{2}+1\times2^{1}+0\times2^{0}=86D$

$4D.8H=4\times16^{1}+13\times16^{0}+8\times16^{-1}=77.5D$

$1362Q=1\times8^{3}+3\times8^{2}+6\times8^{1}+2\times8^{0}=754D$

2. 十进制数→非十进制数

十进制数转换为二进制、八进制及十六进制等非十进制数的工作可分为整数的转换和小数的转换两种情况。一个十进制数与其对应的非十进制数相比，两者的整数部分和小数部分分别相等。于是，将一个十进制数转换为非十进制数时，可以对其整数部分和小数部分分别作转换，将两个转换结果结合起来就可以得到对应的非十进制数。

（1）十进制整数→非十进制整数

十进制整数转换为二进制、八进制及十六进制等非十进制整数可以采用除基取余法或减权记位法。前一方法较易掌握，但使用时较烦琐；后一方法使用时较方便，但需要熟悉非十进制各数位对应的权值。

① 除基取余法。将十进制整数及此间产生的商不断除以非十进制数的基，直至商为 0 为止，并记下每一次相除所得到的余数，按照从后往前的次序，将各余数作为 $K_{n-1}\,K_{n-2}\cdots K_0$，从而构成对应的非十进制数。

【例 1.4】 将 233D 转换为十六进制数。

按照题目要求，基应取 16，具体转换过程如下：

	除以基 2	余数	K_i
16	233	9	K_0
16	14	14（即 E）	K_1
	0		

转换结果为：233D=E9H。

【例 1.5】　将 233D 转换为二进制数。

按照题目要求，基应取 2，具体转换过程如下：

转换结果为：233D=11101001B。

	除以基 2	余数	K_i
2	233	1	K_0
2	116	0	K_1
2	58	0	K_2
2	29	1	K_3
2	14	0	K_4
2	7	1	K_5
2	3	1	K_6
2	1	1	K_7
	0		

② 减权记位法。减权记位法常用于十进制数到二进制数的转换，对于十进制数到其他进制数的转换使用这种方法则不够方便。这里仅介绍十进制数到二进制数转换时，减权记位法的具体内容。

将十进制整数与其最相近的权值 2^{n-1} 做比较，前者不小于后者则减去 2^{n-1}，并在 $n-1$ 位记 1；否则在 $n-1$ 位记 0。然后再与 2^{n-2} 做比较并做相同的工作，直至最低位被记为 1 或 0。从 $n-1$ 位、$n-2$ 位直至最低位所记的 1 或 0 就构成了二进制数 $K_{n-1}K_{n-2}\cdots K_0$。通常第 1 次比较所用的 2^{n-1} 取做小于等于十进制整数的最大二进位权值。

【例 1.6】　将 233D 转换为二进制数。

由于 $2^7<233D<2^8$，故开始用做比较的数值取 2^7。

具体转换过程如下：

比较，减权	记位	K_i
$233-2^7=105$	1	K_7
$105-2^6=41$	1	K_6
$41-2^5=9$	1	K_5
$9<2^4$	0	K_4
$9-2^3=1$	1	K_3
$1<2^2$	0	K_2
$1<2^1$	0	K_1
$1-2^0=0$	1	K_0

转换结果为：233D＝11101001B。

【例 1.7】　将 130D 转换为二进制数。

考虑到 130D 为 128D 与 2D 之和，即为 2^7 与 2^1 之和，于是对应的二进制数 $K_7K_6\cdots K_0$ 中 K_7=1，K_1=1，其余位均为 0，也即转换结果为 130D＝10000010B。

由该例可见，在熟悉二进制各数位对应的权值的基础上，参照减权记位法可以方便地进行十进制整数到二进制整数的转换。

（2）十进制小数→非十进制小数

十进制小数转换为二进制、八进制及十六进制等非十进制小数可以采用乘基取整法或减权记位法。前一方法较易掌握，但使用时较烦琐；后一方法使用时较方便，但需要熟悉非十进制数各数位对应的权值。

① 乘基取整法。将十进制小数以及此间产生的小数部分不断乘以非十进制数的基，并记下每次相乘所得到的整数部分，直至积的小数部分为 0 为止，按照从前往后的次序，将各整数部分作为 K_{-1}，K_{-2}，… K_{-m}，就构成了对应的非十进制数 $K_{-1}K_{-2}\cdots K_{-m}$。

【例 1.8】 将 0.6875D 转换为二进制数。

按照题目要求，基应取 2，具体转换过程如下：

$$
\begin{array}{rl}
 & 0.6875 \\
 & \times\quad 2 \\
\hline
K_{-1} \leftarrow & \boxed{1}.3750 \\
 & \times\quad 2 \\
\hline
K_{-2} \leftarrow & \boxed{0}.7500 \\
 & \times\quad 2 \\
\hline
K_{-3} \leftarrow & \boxed{1}.5000 \\
 & \times\quad 2 \\
\hline
K_{-4} \leftarrow & \boxed{1}.0000
\end{array}
$$

转换结果为：0.6875D=0.1011B。

【例 1.9】 将 0.6875D 转换为八进制数。

按照题目要求，基应取 8，具体转换过程如下：

$$
\begin{array}{rl}
 & 0.6875 \\
 & \times\quad 8 \\
\hline
K_{-1} \leftarrow & \boxed{5}.5000 \\
 & \times\quad 8 \\
\hline
K_{-2} \leftarrow & \boxed{4}.0000
\end{array}
$$

转换结果为：0.6875D=0.54Q。

当一个十进制小数对应的非十进制小数的位数过多时，可根据需要取前若干位作为近似结果。

② 减权计位法。与上述十进制整数转换为二进制整数时使用的减权计位法类似，但有两点区别：其一，十进制小数首先应与 2^{-1} 比较，然后与 2^{-2} 比较，依此类推；其二，有时要根据需要，取部分小数位作为近似转换结果。

【例 1.10】 将 0.6875D 转换为二进制数。

具体转换过程如下：

比较、减权	记　位	K_I
$0.6875-2^{-1}=0.1875$	1	K_{-1}
$0.1875<2^{-2}$	0	K_{-2}
$0.1875-2^{-3}=0.0625$	1	K_{-3}
$0.0625-2^{-4}=0$	1	K_{-4}

转换结果为：0.6875D=0.1011B。

如前所述，将一个十进制数转换为非十进制数时，可以对其整数部分和小数部分分别作转换，然后将整数部分和小数部分的转换结果结合成为最终结果。

【例 1.11】 将 233.6875D 转换为二进制数。

由前面的例题可知：

233D=1110 1001B

0.6875D=0.1011B

于是转换结果为：233.6875D=11101001.1011B。

3. 二进制数↔八进制数、十六进制数

由于一位八进制数对应三位二进制数，一位十六进制数对应四位二进制数，于是二进制数与八进制数、十六进制数之间的转换比较简单。

（1）二进制数→八进制数、十六进制数

将二进制数由小数点向左右每三位分为一组（不足三位则用 0 补充），每一组用对应的八进制数码表示，即可得到对应的八进制数。类似地，将二进制数由小数点向左右每四位分为一组（不足四位则用 0 补充），每一组用对应的十六进制数码表示，即可得到对应的十六进制数。

【例 1.12】 将 0101 1101.01B 分别转换为八进制数和十六进制数。

01 011 101.01B = 001 011 101 .010B = 135.2Q

0101 1101.01B = 0101 1101.0100B = 5D.4H

说明：在分组中若不足三位或不足四位时，一定要用 0 补充，否则极易出错。就该例而言，若不注意这一点，就极易将结果错写为 135.1Q 和 5D.1H。

（2）八进制数、十六进制数→二进制数

将八进制数的每一位数用三位二进制数码表示，即可得到对应的二进制数。类似地，将十六进制数的每一位数用四位二进制数码表示，即可得到对应的二进制数。必要时可以去掉转换结果中的前 0 和尾 0。

【例 1.13】 分别将 45.4Q 和 27CH 转换为二进制数。

45.4Q=100 101 .100B=100101.1B

27CH=0010 0111 1100B=100111 1100B

从以上介绍可以看出，在将十进制数转换为非十进制数时，转换为二进制数较为简单，转换为八进制数、十六进制则较为复杂。考虑到二进制数转换为八进制数、十六进制数比较简单，故在需要将十进制数转换为八进制数、十六进制数时，可以先将其转换为二进制数，然后再由二进制数转换为八进制数、十六进制数。

【例 1.14】 将 233.6875D 转换为十六进制数。

首先将 233.6875D 转换为二进制数，由【例 1.11】可知：

233.6875D = 1110 1001.1011B

然后将 11101001.1011B 转换为十六进制数。

1110 1001.1011B = E9.BH

于是十进制数到十六进制数的转换结果为

233.6875D = E9.BH