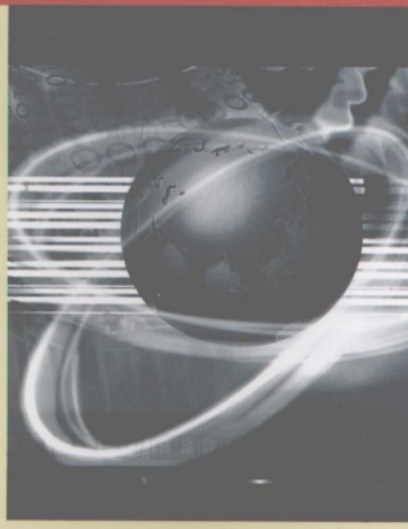
普通高等教育“十五”国家级规划教材



80X86
Assembly Language Programming

80X86
汇编语言程序设计

王元珍　曹忠升　韩宗芬　编著

华中科技大学出版社
http://press.hust.edu.cn

80X86
汇编语言程序设计

王元珍　曹忠升　韩宗芬　编著

华中科技大学出版社

图书在版编目(CIP)数据

80X86 汇编语言程序设计/王元珍　曹忠升　韩宗芬　编著
武汉:华中科技大学出版社,2005 年 4 月
ISBN 7-5609-3357-2

Ⅰ.8…
Ⅱ.①王…　②曹…　③韩…
Ⅲ. 汇编语言-程序设计-教材
Ⅳ.TP312

80X86 汇编语言程序设计　　王元珍　曹忠升　韩宗芬　编著

责任编辑:沈旭日　　封面设计:潘　群
责任校对:刘　竣　　责任监印:张正林

出版发行:华中科技大学出版社　武昌喻家山　邮编:430074　电话:(027)87557437

录　　排:华中科技大学惠友文印中心
印　　刷:湖北新华印务有限公司

开本:787×1092　1/16　　印张:22.5　　字数:503 000
版次:2005 年 4 月第 1 版　　印次:2006 年 12 月第 2 次印刷　　定价:32.00 元
ISBN 7-5609-3357-2/TP・562

内容简介

本书为教育部普通高等教育“十五”国家级规划教材。

本书以目前使用最为广泛的80X86机为例，详细介绍了使用宏汇编语言进行程序设计的理论、方法和技巧。全书共分7章，主要内容包括：80X86宏汇编语言、程序设计的基本技术和模块化程序设计技术、输入/输出、中断异常和WIN32程序设计，同时还介绍了在DOS和Windows环境下调试、运行32位段与16位段汇编源程序的方法，每章后面均附有丰富的习题。

本书内容新颖、重点突出、例题习题丰富、语言精炼易懂。可供各类高等学校计算机及非计算机有关专业作为教材，亦可供广大工程技术人员和其他读者自学、参考。

前　言

“汇编语言程序设计”是计算机类专业的重要专业基础课，是从事计算机研究与应用、特别是软件研究的基础，是计算机专业人员必须接受的最重要的专业基础训练之一。目前，使用最为广泛的PC系列机都是以Intel的80X86系列微处理器为CPU的。本书从系统软件和应用软件设计的角度出发，以80X86机为例，详细介绍了宏汇编语言的基本概念、基本原理和程序设计的常用方法与技术，通过具体实例，叙述了用计算机解决实际问题的全过程，同时还介绍了在DOS和Windows环境下调试、运行16/32位段汇编源程序的方法。

本书为教育部普通高等教育“十五”国家级规划教材。

本书在编写过程中力求遵循先进性和基础性相结合、理论性和应用性相结合、科学性和通俗性相结合的原则。在内容的选取、概念的引入、文字的叙述以及例题、习题的选择等方面力求新颖全面、重点突出、重视实践、便于自学。全书共分7章，其中，第1章介绍学习80X86宏汇编语言程序设计所需的若干预备知识；第2章详细介绍各种寻址方式的汇编格式、功能及使用方法，并重点阐述了它们之间的区别与联系；第3章主要介绍宏汇编语言中的表达式、常用的机器指令语句、伪指令语句和DOS系统功能调用的汇编格式及功能；第4章系统地介绍顺序、分支、循环、子程序的程序设计方法及技巧；第5章重点介绍串操作指令的汇编格式及使用方法、宏指令的定义与调用、模块化程序设计技术；第6章主要介绍中断及异常的概念、浮点运算指令，并以32位段汇编与DOS 16位段汇编对比的方式介绍WIN32编程，以帮助学生在学习了汇编语言后有更好的应用平台，能开发出应用面更广的程序；第7章全面介绍在DOS和Windows环境下调试、运行汇编源程序的方法与技术。附录提供了80X86指令系统简表、伪指令表、DOS的软中断与系统功能调用、常用BIOS子程序的功能及其调用参数、汇编连接程序错误信息等内容。书中带“*”号的章节为选学内容。为了帮助读者更好地掌握汇编语言程序设计的特点，书中结合应用安排了丰富的例题，希望读者用心地阅读这些例题，从中学习一些编程的基本规律。程序设计是一门实践性很强的学科，即包含复杂的脑力劳动，又是一种极富有创造性的活动。因此，读者在学习过程中应多阅读程序，多编程序，多上机，只有这样，才能真正掌握程序设计的方法与技巧。

本书由王元珍教授担任主编。其中，第1、3、5、7章由王元珍教授编写，第2、4、6章和附录由曹忠升副教授编写，韩宗芬教授参与了第2、4章的部分编写工作，王元珍统编全稿。在编写本书的过程中，得到了华中科技大学计算机学院、“汇编语言程序设计”教学组的领导和老师们的热情帮助和支持，得到了华中科技大学出版社有关领导和责任编辑沈旭日的关心和帮助，在此一并表示衷心感谢。

由于作者水平有限，书中不妥或错误之处在所难免，殷切希望广大读者批评指正。同时也欢迎读者，尤其是使用本书的教师和学生，共同探讨相关的教学内容和教学方法等问题。

作　　者

2005 年 1 月于华中科技大学(武汉)

普通高等教育“十五”国家级规划教材

华中科技大学百门精品课程教材

目　录

第1章 预备知识

汇编语言是一种面向机器的、能够充分利用计算机硬件特性的低级语言，它随机器结构的不同而不同。因此，要学会一种汇编语言，就必须首先了解与该机器有关的硬件结构。本章将从汇编语言程序设计的角度出发，介绍有关的预备知识，如：什么是汇编语言、Intel 80X86 微处理器中的寄存器组、主存储器的编址方式及物理地址的形成方式、数和符号在计算机中的表示方法，并以一个源程序为实例介绍汇编源程序的基本结构和格式，这些都是学习后继各章的必备知识。

1.1 机器语言与汇编语言

1.1.1 机器语言

计算机的工作必须接受人的操纵和控制，为了让计算机能按人的意愿工作，就必须与计算机之间交流信息，这个交流信息的工具就是计算机语言。目前所使用的计算机语言分3类：机器语言、汇编语言和高级语言。高级语言接近于自然语言，易学，易记，便于阅读，容易掌握，使用方便，通用性强且不依赖于具体的计算机。但是，计算机并不能识别高级语言，它只能识别在设计该机器时事先规定好的机器指令。

机器指令即为指挥计算机完成某一基本操作的命令。机器指令的一般形式为

操作码	地址码

操作码和地址码均是由0和1组成的二进制代码，可见每一条机器指令都是用一组二进制代码来表示的。操作码指出了操作的种类，如加、减、传送、移位等；地址码指出了参与运算的操作数和运算结果存放的位置。

例如，将偏移地址为1000H的字存储单元中的内容加3000H，再送回到原存储单元中去，如果用 Intel 80X86 的机器指令来完成该操作，则相应的机器指令为

7	0	对应的十六进制形式	
1 0 0 0 0 0 0 1		→	81H
0 0 0 0 0 1 1 0			06H
0 0 0 0 0 0 0 0			00H
0 0 0 1 0 0 0 0			10H
0 0 0 0 0 0 0 0			00H
0 0 1 1 0 0 0 0			30H

其中，第1、2行中的两个8位二进制数是操作码，表示要进行“加”操作，还指明了以何种方式取得两个加数；第3、4行中的两个8位二进制数指出了第一个加数所存放的偏移地址是1000H，因为结果也存放在该偏移地址中，所以第一个加数称目的操作数；第5、6行中的两个8位二进制数指出了第二个加数(称源操作数)是3000H。

机器指令也常常被称为硬指令，它是面向机器的，即每台计算机都规定了自己所特有的、一定数量的基本指令，这批指令的全体即为计算机的指令系统；这种机器指令的集合就是机器语言。用机器语言编写的程序称为机器语言程序。

1.1.2 汇编语言

由于机器指令是用二进制表示的，所以编写起程序来相当麻烦，写出的程序也难以阅读和调试。为了克服这些缺点，人们就想出了用助记符表示机器指令的操作码；用变量代替操作数的存放地址；在指令前冠以标号，用来代表该指令的存放地址等。这种用符号书写的、其主要操作与机器指令基本上一一对应的、并遵循一定语法规则的计算机语言就是汇编语言。用汇编语言编写的程序称为汇编源程序。由此可见，汇编语言是面向机器的语言。本教材中所介绍的是Intel 80X86宏汇编语言。

例如，对于前面的例子，可用宏汇编语言书写为

```
ADD WORD PTR DS:[1000H], 3000H
```

其中，ADD为加指令的助记符，代表了机器指令中的操作码；DS:[1000H]表示在当前数据段中，偏移地址为1000H单元中的内容是目的操作数；WORD PTR说明了这个目的操作数是16位二进制数，而源操作数是3000H，相加的结果送入目的操作数所在的单元中。

由于汇编语言是为了方便用户而设计的一种符号语言，因此，用它编写出的源程序并不能直接被计算机识别，必须将它翻译成由机器指令组成的程序后，计算机才能识别并执行。这种由源程序经过翻译转换，生成的机器语言程序也称为目标程序。目标程序中的二进制代码(即机器指令)称为目标代码。这个翻译工作一般都由计算机自己去完成，但人们事先必须将翻译方法编写成一个语言加工程序作为系统软件的一部分，在需要时让计算机执行这个程序才可完成对某一汇编源程序的翻译工作。

这种把汇编源程序翻译成目标程序的语言加工程序称为汇编程序。汇编程序进行翻译的过程叫做汇编。

在这里，汇编程序相当于一个翻译器，它加工的对象是汇编源程序，而加工的结果是目标程序。它们三者之间的关系如图1.1所示。

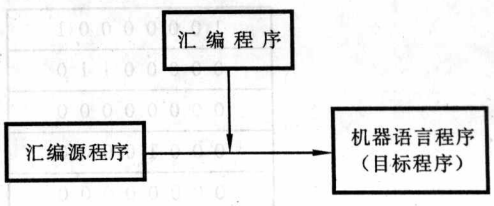


图1.1 汇编程序与汇编源程序、目标程序的关系

为了能让汇编程序正确地完成翻译工作，必须告诉汇编程序，源程序应从什么位置开始安放，汇编到什么位置结束，数据放在什么位置，数据的类型是什么，留多少内存单元作临时存储区等。这就要求源程序中应该有一套告诉汇编程序如何进行汇编工作的命令，这种

命令称伪指令(或称汇编控制命令)。

由此可见，指令助记符、语句标号、数据变量、伪指令及它们的使用规则构成了整个汇编语言的内容。

由于汇编语句基本上与机器指令对应，因而它的编写也是相当麻烦的。为了简化程序的编写，提高编程效率，现代的计算机系统一般都提供了宏汇编程序。它允许程序员用一个名字(宏指令名)来代替程序中重复出现的一组语句，然后在源程序中所需要的地方使用宏指令名字及不同的参数进行宏调用。熟练灵活地使用宏调用功能可使汇编源程序编写得像高级语言一样清晰、简洁、灵活，且容易使算法标准化。这样，不仅加速了源程序的编写效率，而且方便了源程序的修改和调试。Intel 80X86 把用宏汇编语言编写的源程序翻译成目标程序的工作是由宏汇编程序来完成的。

与机器语言相比，汇编语言易于理解和记忆，所编写的源程序也容易阅读和调试，所占用的存储空间、执行速度与机器语言相仿。

与高级语言相比，汇编语言具有直接和简捷的特点，用它编制的程序能精确地描述算法，充分发挥计算机硬件的功能，还可实现一些硬件难以实现的操作；而且用汇编语言编写的程序比高级语言编写的精短，占用空间小，运行速度快。例如，在通信、工业、计量、测试、监控等领域，由于输入/输出(简称 I/O)处理复杂、实时性强，有许多操作都得用汇编语言来实现的。再如，多媒体技术、网络通信技术、计算机安全、病毒的防治等方面，也由于需要与硬设备打交道而使用汇编语言。目前，系统程序的研制虽然普遍采用高级语言，但其产生的目标往往还是采用汇编语言的形式，而且还有一些系统程序和应用程序仍需要用汇编语言来编写。此外，实用中还有许多用汇编语言编写的系统程序和应用程序需要阅读、分析乃至修改。因此，几乎每一个计算机系统，都把汇编语言作为系统的基本配置，汇编程序成为系统软件的核心成分之一。汇编语言程序设计的方法与技巧又是进行其他高级语言程序设计的基础，是学习后继课程的基础。对于从事计算机研制和应用的广大科技工作者来说，熟练而牢固地掌握汇编语言程序设计技术是每一个计算机专业人员的基本功，是广大计算机工作者从事计算机研究与应用的基础，一个不经过汇编语言程序设计严格训练的计算机专业人员是不能很好地胜任其工作的。

1.1.3 书中使用符号的说明

在后面的学习中，需要引用以下一些符号。

(…)　表示地址“…”中的内容。例如，若偏移地址为 100 的存储单元中的内容为 50，则表示为(100)=50；寄存器 BX 的内容为 0FFFFH，则表示为(BX)=0FFFFH。

[…]　表示以地址“…”中的内容为偏移地址。例如，(BX)=03A2H，而(03A2H)=100，则([BX])表示以 BX 的内容为偏移地址，在该偏移地址中存放的数据。此处的([BX])=100。

EA　某一存储单元的偏移地址，指该存储单元到它所在段段首址的字节距

	离(详见1.2.2小节)。
PA	某一存储单元的物理地址(详见 1.3.3 小节)。
R	指某寄存器名字。
SR:[R]	指二维的逻辑地址(指针)。其中,SR 为段寄存器名;[R]表示某寄存器 R 为指示器,存放着该段内某一存储单元的偏移地址。
OPD	目的地址,即目的操作数存放的偏移地址。
OPS	源地址,即源操作数存放的偏移地址。
→	表示传送。例如,300H→BX 表示将操作数 300H 传送到寄存器 BX 中;设(100)=50,则(100)→BX 表示将偏移地址为 100 单元中的内容 50 传送到 BX 中。
↑(ESP/SP)	表示出栈(弹出)。其中,↑(ESP)表示从 32 位段堆栈中弹出数据;↑(SP)表示从 16 位段堆栈中弹出数据。
↓(ESP/SP)	表示进栈(压入)。其中,↓(ESP)表示往 32 位段堆栈中压入数据;↓(SP)表示往 16 位段堆栈中压入数据。
∧	表示逻辑乘。
∨	表示逻辑加。
⊻	表示按位加。
$\overbrace{\times\times\times}$	表示需要对数"×××"做求补运算。
$\overline{\times\times\times}$	表示需要对数"×××"做求反运算。
××H	表示数"××"为十六进制数。若其中的第一位数为 A～F,则应在其前面补 0。
××	表示数"××"为十进制数。
××B	表示数"××"为二进制数。
××BCD	表示数"××"为 BCD 码。
FLAGS	表示 16 位标志寄存器。
EFLAGS	表示 32 位标志寄存器。
/	表示或者。
$\underline{\times\times\times}$	表示横线上面的内容"×××"是通过键盘输入的。 例如,有以下书写形式: * $\underline{\text{ABC}\swarrow}$ 其中,"*"下面未加横线,表示是由显示器显示的内容;而"ABC"下面有横线,说明是由键盘输入的内容;"↙"为行终止符,用以表示一行内容的结束,在输入时,是通过敲键盘上的回车键实现的。

1.2 Intel 系列机简介

微型计算机主要由微处理器(CPU)、主存储器(简称主存或内存)、外部设备及互连部件

组成，总线(数据总线、地址总线、控制总线)在各部件之间提供通信通道，其系统简略结构如图 1.2 所示。其中，CPU 是它的核心部分，主要由 Intel 80X86 微处理器(以下简称为 80X86)组成；主存用来保存程序和数据，受技术条件和成本的限制，主存空间不能过大。为了弥补主存容量小的不足，微型计算机还提供了价格低廉的大容量存储器作辅助存储器(也称外存储器)，如硬盘、光盘、U 盘、磁带等。除此以外，微型计算机还可配置多种外部设备，其中，键盘、显示器和打印机是最基本的配置。以上这些设备的结构、工作原理都是相当复杂的，详细地讨论它们不是本课程的内容，为了后面学习的需要，我们仅介绍与 Intel 80X86 有关的部分。

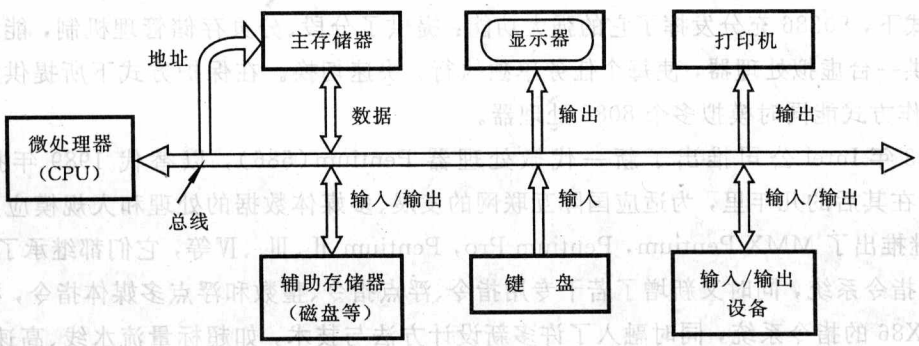


图 1.2　微型计算机的构成

1.2.1　Intel 80X86 微处理器简介

Intel 公司是世界上第一个生产微处理器的生产厂家，所生产的 80X86 系列微处理器一直是个人计算机的主流 CPU。

1971 年，Intel 公司生产的 4 位微处理器芯片 4004 宣告了微型计算机时代的到来。1972 年，Intel 公司又开发了 8 位的微处理芯片 8008；1974 年接着生产了 8080；1977 年，Intel 公司将 8080 及其支持电路集成在一块集成电路芯片上，形成了性能更高的 8 位微处理器 8085，并得到了广泛应用。

1978 年，Intel 正式推出了 16 位的 8086，也称 16 位 CPU。这是该公司生产的第一个 16 位芯片，内、外数据总线均为 16 位，地址总线为 20 位，主存寻址范围 1MB，时钟频率 5MHz。8086 的 16 位指令系统成为后来广泛应用的其他 80X86 的基本指令集。

由于 8086 的 16 位外部数据总线不易与当时广泛应用的 8 位外设接口，1979 年 Intel 推出了准 16 位微处理器 8088，它只是将外部数据总线设计为 8 位，其他(包括指令系统)均与 8086 相同，应用非常广泛。

1982 年 Intel 公司推出了一种超级微处理器 80286。80286 具有 24 条地址线、16 条数据线，能直接寻址 16MB 主存。包含有 4 个独立的处理部件：执行部件、总线部件、指令部件和地址部件，这 4 个部件能并行工作，其系统性能和效率都比 8086 要高得多。80286 提供了两种工作方式：实地址方式(简称实方式)和保护方式。实方式的操作相当于一个快速的 8086；在

保护方式下，80286 提供了多任务并发执行(即多个程序都同时处于运行状态之中)的硬件控制，但由于当时环境的限制和 80286 本身的原因，未能很好地满足多任务处理的要求。

1985 年 Intel 公司正式推出 32 位微处理器 80386，并由此确定了 32 位指令系统的结构，被 Intel 公司称为英特尔结构(IA)，并明确宣布作为后续 80X86 微处理器的标准。因此，也将以后 80X86 中的 32 位微处理器统称为 32 位 CPU。

80386 采用了 32 位寄存器，具有 32 根地址总线和数据总线，全面支持 32 位的数据、指令和寻址方式，可访问 4GB 的存储空间。80386 提供了 3 种工作方式：实方式、保护方式和保护方式下的虚拟 8086 方式。实方式的操作相当于一个可进行 32 位快速运算的 8086。在保护方式下，80386 充分发挥了它的强大功能：提供了分段、分页存储管理机制，能为每个任务提供一台虚拟处理器，使每个任务单独执行，快速切换。在保护方式下所提供的虚拟 8086 工作方式能同时模拟多个 8086 处理器。

1993 年 Intel 公司推出了新一代微处理器 Pentium(586)，以替代 1989 年推出的 80486。在其后的几年里，为适应国际互联网的发展、多媒体数据的处理和大规模应用的要求，相继推出了 MMX Pentium，Pentium Pro，Pentium Ⅱ、Ⅲ、Ⅳ等，它们都继承了 80386 的 32 位指令系统，同时又新增了若干专用指令、浮点指令、整数和浮点多媒体指令，极大丰富了 80X86 的指令系统，同时融入了许多新设计方法与技术，如超标量流水线、高速缓存、快速浮点部件、动态分支预测等技术大大提高了系统的工作效率，有效增强了 80X86 微处理器的功能与性能。

1.2.2 Intel 80X86 微处理器结构

随着 80X86 系列机的发展，80X86 微处理器的结构变化是非常大的，但它的功能总是保持着向下兼容，详细介绍它各部分的结构和功能不是本门课程的内容。因此，本节从学习汇编语言的角度出发进行介绍。32 位 CPU 按其主要功能通常可分为六大部件：总线接口部件、执行部件、指令预取部件、指令译码部件、分段部件和分页部件等。其内部结构如图 1.3 所示。下面分别介绍这六大部件的基本结构及功能。

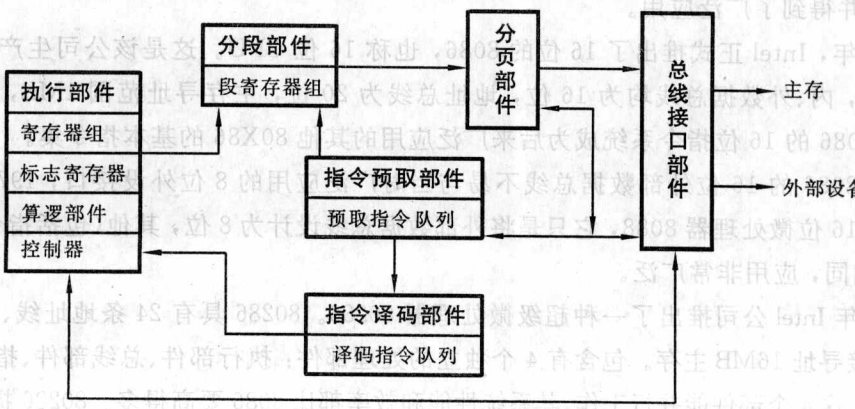


图 1.3 80X86 微处理器的基本结构