



普通高等教育“十一五”国家级规划教材
高等学校教材

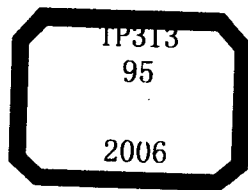
新版汇编语言 程序设计

● 钱晓捷 主编



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>



高等学校教材

新版汇编语言程序设计

钱晓捷 主编

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书以 Intel 80x86 指令系统和 MASM 6. x 为主体,全面而系统地介绍 16/32 位整数、浮点、多媒体指令的汇编语言程序设计方法。全书可分为基础和提高两部分。前 5 章作为基础部分,以当前“汇编语言程序设计”课程的教学为目标,为读者讲解 16 位基本整数指令及其汇编语言程序设计的知识。基础部分的主要内容是:汇编语言程序设计的基础知识,8086 指令详解,MASM 伪指令和操作符,程序格式,程序结构及其设计方法。后 4 章为提高部分,从不同的方面介绍汇编语言程序设计的深入内容和实际应用知识。提高部分各章的内容相对独立,主要有:32 位 80x86 CPU 的整数指令系统及其编程,汇编语言与 C/C++ 的混合编程,80x87 FPU 的浮点指令系统及其编程,多媒体扩展指令系统及其编程,64 位指令简介。

本书可作为高等院校《汇编语言程序设计》课程的教材或参考书,主要读者为计算机及相关学科的本科和高职、高专学生。本书内容广博、语言浅显、结构清晰、实例丰富,也适合于电子、自动控制等专业的高校学生和成教学生,计算机应用开发人员,深入学习微机应用技术的普通读者等。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有,侵权必究。

图书在版编目(CIP)数据

新版汇编语言程序设计/钱晓捷主编. —北京:电子工业出版社,2006.12

高等学校教材

ISBN 7-121-02696-1

I. 新… II. 钱… III. 汇编语言—程序设计—高等学校—教材 IV. TP313

中国版本图书馆 CIP 数据核字(2006)第 123013 号

责任编辑:赵家鹏

印 刷: 北京市铁成印刷厂
装 订:

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

开 本: 787×1 092 1/16 印张: 22 字数: 563.2 千字

印 次: 2006 年 12 月第 1 次印刷

印 数: 6 000 册 定价: 31.00 元

凡所购买电子工业出版社图书有缺损问题,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; 邮购电话:(010)88254888。

质量投诉请发邮件至 zltz@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线:(010)88258888。

前 言

本书以 Intel 80x86 指令系统和 MASM 6. x 为主体,在个人微机的 MS-DOS 和 Windows 操作系统平台中,依据循序渐进的原则,以浅显的语言、清晰的结构、丰富的实例,全面而系统地介绍整数指令、浮点指令、多媒体指令的汇编语言程序设计方法,以及汇编语言的模块化开发、32 位 Windows 应用程序的编写、与 C/C++ 混合编程等高级技术。全书共有 10 章,分为前 5 章的基础部分和后 5 章的提高部分。

基础部分

第 1 章总结性地给出进行汇编语言程序设计所需要的基本知识,包括微型计算机系统的软硬件组成和数据表示;并重点展开 8086 通用寄存器和寻址方式。

第 2 章详尽地讲述 8086 微处理器整数指令的功能与使用,并引导读者正确书写每条指令、理解程序段的功能,以及编写常见问题的汇编语言程序。

第 3 章结合汇编语言源程序格式,引出程序开发、语句格式、常用伪指令和操作符、段定义等内容。

第 4 章以程序设计技术为主线,结合大量的程序实例详述顺序、分支、循环、子程序结构的汇编语言程序设计方法。

第 5 章展开汇编语言的高级程序设计技术,包括 MASM 的高级语言特性、宏结构程序、模块化程序设计和 I/O 程序设计。

提高部分

第 6 章首先将 16 位指令及编程扩展到 32 位环境,其次介绍新增整数指令及其应用,最后重点展开利用汇编语言编写 32 位 Windows 控制台和窗口应用程序的开发环境和基本方法。

第 7 章讨论汇编语言与高级语言 Turbo C 和 Visual C++ 的混合编程,并说明如何利用汇编语言优化 C++ 代码,以及利用 Visual C++ 集成环境开发汇编语言程序的方法。

第 8 章介绍 Intel 80x87 浮点数据格式、浮点指令及其程序设计方法。

第 9 章介绍 MMX、SSE、SSE2 和 SSE3 多媒体指令及其编程方法。

第 10 章披露 EM64T 技术的 64 位指令。

每章最后配有相当数量的习题,既可以作为课后作业也可以作为上机练习。附录介绍 CodeView 调试程序的使用方法,提供汇编程序 MASM 6.11 的伪指令和 Intel 80x86 整数指令列表,还有配套本教材的输入输出子程序库的使用说明。

本书特点

“汇编语言程序设计”是我国高校计算机及相关学科的一门专业基础课程。微型机技术的发展、教学内容的更新、教学方法的完善,以及教学手段的改进等都为该课程提出了更高的要求。我们在前 2 版“汇编语言程序设计”教材的基础上,结合普通高等教育“十一五”国家级教材规划项目精神,总结教学经验和改革思路,重新编写了新版“汇编语言程序设计”教材。新版教材具有以下多方面的特点。

• 知识全面

本书的编写参照国内高校和自学考试“汇编语言程序设计”课程的本、专科教学大纲,兼顾相关专业教学要求,既满足当前教学需求,又面向今后改革方向。

本书全面讲解 80x86 指令系统及编程,除完整的 8086 指令外,还包括 32 位指令、浮点指令、多媒体指令。本书不仅介绍基本的汇编语言程序设计知识,而且还介绍高级汇编语言程序设计技术,例如开发大型程序需要的模块化方法、实际应用当中的混合编程实用技术、32 位 Windows 应用程序编写等。

本书采用微软的 MASM 6.15 汇编程序,以简化段定义源程序格式为主,兼顾完整段定义源程序格式,涉及 DOS 和 Windows 操作系统平台的汇编语言程序设计。

• 教材实用

本书示例中的指令、程序段和完整的源程序都经验证,并运行通过。本书经过两个版本、6 年的使用时间,已经发现并纠正了绝大多数错误。

本书采用浅显、明晰、循序渐进的描述方法,具有前后对照、贯穿始终的风格,加上清晰的结构、丰富的示例,使得本书既适合课堂教学,又适合读者自学。

配合本书,作者制作有精美的多媒体电子教案(PowerPoint 演示文档),为教师利用现代化教学手段提供方便。同时,作者还维护“大学微机技术系列课程教学辅助网站”(钱晓捷之微辅网:<http://www2.zzu.edu.cn/qwfw>)。网站主要就“汇编语言程序设计”和“微机原理及接口技术”课程提供有关教学课件、教材勘误、疑难解答、补充资料等辅助资源,是本教材的动态延伸。微辅网还设置有留言簿和教学论坛,让学生在课外学习中获得帮助。

• 突出实践

本书特别强调上机实践,不仅在正文中引导读者通过调试程序或者带输出结果的源程序理解指令、程序功能,各章还配有丰富的习题和上机练习题;附录介绍调试程序 CodeView 的使用方法、调试指令和程序的步骤。

本书的结构安排适合尽早上机实践,并将实践过程贯穿始终。第 1 章为学生介绍命令行 MS-DOS 基本操作(基于 Windows),第 2 章充分利用调试程序学习指令功能和调试程序段,第 3 章引出完整的源程序格式,并给出程序开发方法。后续章节通过大量程序强化编程开发,还介绍开发 Windows 应用程序的 MASM32 环境、利用 Visual C++ 开发调试汇编语言程序的方法。

写给教师

“汇编语言程序设计”的教学内容在高校相关学科的教学中有两种处理方式。一种作为独立的课程,这主要是用在计算机专业的本科、专科教学中;而在电子、通信、自动控制等专业则将汇编语言作为主体融入微机原理课程。本书内容自成一个相对完整的知识体系,适合作为独立课程的教材,同时也可以作为微机原理课程的参考和补充教材。

传统上,“汇编语言程序设计”课程在 MS-DOS 操作系统平台上,采用微软的 MASM 汇编程序,介绍 16 位 8086 微处理器指令系统的程序设计。当前,汇编语言正在将教学重点转向 32 位 IA-32 微处理器指令系统,操作系统平台也逐渐向 Windows 过渡。同时,也出现在 Linux 环境的汇编语言课程。针对不同层次、不同专业的学习,“汇编语言程序设计”课程教学进行了多方面的改革,形成了多元化的教学方案。

本书从简单的 16 位汇编语言入手,使其满足当前教学要求(包括与“微型机原理”或“微机

接口技术”等课程的配合);在 32 位汇编语言展开许多深入的内容,使其面向以后的教学需求。本书在组织教学内容上,体现了许多新的理念。例如没有从纯软件角度介绍汇编语言,教学的重点是硬指令不是伪指令,强调程序设计不是程序格式,引出实用技术但淡化具体应用,通过程序实例和上机实践掌握程序设计方法,不是通过大量细节的描述讲解程序设计。

汇编语言的基本语句是处理器指令。对于 Intel 80x86 系列微处理器,由于其属于复杂指令集计算机 CISC,再加上近 30 年发展的历史沉淀,它的指令系统非常庞大和复杂。虽然作为本书的一个特色,介绍了所有指令,但教材的重点教学内容却是常用的简单指令,全书的实例程序也主要采用各种处理器指令系统所共有的基本指令编写。作为教师,尤其应该注意这个问题,否则许多学生会面对繁杂的指令望而却步、失去进一步学习的兴趣。

各种高级语言程序设计的教学中,调试程序及程序的调试方法往往被忽略或回避,但作为低级程序语言的汇编语言不应避而不谈。在汇编语言的教学过程中,可以利用调试程序的单步执行和断点执行能力,直观地理解指令和程序的功能,进而掌握程序的动态调试和排错。对于专科层次或程度较低的学生,掌握调试程序本身就是一个似乎不可逾越的难关。一方面,教师可以通过多媒体教学手段,演示调试程序的使用;本书也配套制作了相关演示动画、记录了指令调试和程序开发过程(可以通过本书的教学网站下载);学生通过上机实践学习调试程序。另一方面,本书自编了一个显示输出和键盘输入的 I/O 子程序库,教师和学生可以直接使用编写具有显示结果的源程序,同时还可以配合列表文件,暂时避开调试程序这个难点。这个 I/O 子程序库还可以作为一个教学案例,用于组织学习中有余力的学生围绕输入输出子程序库进行项目开发。

对于使用过本书前续版本的教师,新版教材在主体教学内容上保持了兼容,仍然遵循由浅入深、循序渐进的原则:先 16 位 8086 指令系统、基本汇编语言编程技术,然后介绍 32 位指令编程,并将汇编语言知识加深,从混合编程、浮点编程、多媒体编程各个角度展开。基础部分的前 5 章主要删除了 DEBUG 调试说明,并将附录 A 用 CodeView 替换了 DEBUG 调试程序(删除的内容作为辅助资料在教学辅助网站上提供),使得同一个调试环境贯穿全书。提高部分的后续章节新增 32 位 Windows 应用程序、Visual C++ 函数优化、SSE3 指令和 64 位指令等。有鉴于此,教师在教学过程中,应该选择基本的、常用的指令,减少 16 位汇编语言部分的教学内容和教学时间。这样,就有时间展开对 32 位汇编语言的深入内容和实用技术的学习。

写给学生

学习汇编语言到底有什么用途?这是许多学生首先要提问的。

在计算机科学与技术的知识体系当中,“汇编语言程序设计”课程的教学内容属于计算机系统结构的一个方面。汇编语言配合“计算机组成原理”和“微机原理及接口技术”等相关课程,帮助学生从软件角度理解计算机工作原理;同时,为“操作系统”、“编译原理”等课程提供必须的基础知识,也是自动控制等与硬件相关应用领域的程序设计基础。“汇编语言程序设计”课程是继“高级语言程序设计”之后的又一门计算机语言程序设计课程,但它是一种低级语言程序设计。通过汇编语言的学习,学生能够比较全面地了解程序设计语言,利于更深入地学习和应用高级语言。

随着高级语言的发展、可视化开发工具的应用,汇编语言往往被应用程序开发人员所忽略,其应用领域也逐渐萎缩。但是,作为一个面向机器的程序设计语言,汇编语言具有直接有效控制硬件的能力,能够编写出运行速度快、代码量小的高效程序,在许多场合具有不可替代

的作用,例如操作系统的核心程序段、实时控制系统的软件、智能仪器仪表的控制程序、频繁调用的子程序或动态连接库、加密解密软件、分析和防治计算机病毒,等等。

学习什么样的汇编语言呢?这是许多学生感到困惑的。

汇编语言与处理器指令系统相关,不同的处理器指令系统具有不同的汇编语言。但是,作为一个底层开发语言,它还是有许多共性的。从指令系统的典型性、实用性、编程环境以及教学内容连续性等多方面考虑,Intel 80x86 指令系统作为“汇编语言程序设计”课程的主要教学内容具有显而易见的优势,理所当然地成为计算机及相关学科的首选。

日常工作和学习中广泛使用的个人微机(PC)采用 Intel 80x86 或与之兼容的微处理器。个人微机过去使用 DOS 操作系统,现在主要使用 Windows 或 Linux 操作系统,但后者具有 DOS 模拟环境。由于 DOS 操作系统平台比较简单,对程序员限制少,是一个相对理想的教学环境,所以本书也就利用这个平台展开汇编语言,选用 DOS 和 Windows 平台下最常用的微软 MASM 汇编程序。

许多学生总感觉 16 位指令、8086 微处理器、DOS 操作系统都是很“古老”和“陈旧”的内容,但它们实际上却是 32 位指令、Pentium 4 微处理器、Windows 或 Linux 操作系统的基础,都是最基本的知识。它们已经足够复杂、完全能够满足教学要求。当然,本书在提高部分也对使用 32 位指令编写 Windows 应用程序进行了详细讲解。因为有了 16 位汇编语言基础,这部分提高内容也就比较容易掌握了。如果直接从 32 位指令和 Windows 平台入门,往往需要大家学习很多其他内容才能够真正进入汇编语言的教学。国内外以所谓 32 位保护方式展开 Windows 汇编语言教学的教材,实际上是利用了 Windows 控制台环境。而 Windows 控制台环境的操作和外观与 DOS 操作系统一样,其基本教学内容几乎相同,所不同的仅仅是调用操作系统功能的方法。介绍用汇编语言编写 Windows 窗口应用程序的教材,都要求读者初步掌握汇编语言程序设计,并具有调用 Windows 应用程序接口 API 的编程经验(往往需要学习 Visual C++ 之后才能够达到这个要求)。本书从基础知识入手,囊括上述所有教学内容。相信读者阅读本书后,会对此有深刻的体会。

怎样才能学好汇编语言呢?这又是许多学生深感无助的。

其实,学习汇编语言程序设计对先修知识要求不高。学生如果具备微机软件的操作能力,尤其是 DOS 环境和常用命令的使用,可以更好地完成上机实践(本书第 1 章补充这方面的知识,并构造一个 MASM 开发环境)。学生的高级语言(如 C/C++ 或 Visual Basic)程序设计知识或经验,将有助于更好地理解程序结构和混合编程。程序设计属于软件方面的内容,但由于汇编语言与硬件相关的特点,学生在计算机或微型机原理方面的知识对于深刻体会指令功能大有益处。“汇编语言程序设计”课程的相关课程是“微机原理与接口技术”,后者是微型机的硬件知识,进一步加强了汇编语言在输入输出和中断等方面的应用。两者共同为学生建立微型计算机的完整知识体系。

循序渐进的学习是对任何课程都有效的。不以循序渐进方式进行学习,往往会浪费宝贵的时间去盲目探索,最终学习到的内容可能是相对零散的知识,不能建立完整的、系统的知识结构。所以,建议学生遵循循序渐进的方法进行学习。首先,学生应理解每条常用指令的功能,能够正确书写每条指令;其次,学生通过阅读示例程序,掌握完成常见功能程序段的编写;再次,学生利用伪指令将程序段扩展成完整的源程序文件;随后,学生就各种程序结构编写常见问题的程序;最后,学生才编写较大型程序和有一定难度的程序。

对程序设计类课程,没有上机编程的实践是无法真正掌握的。所以,希望学生加强实践环

节。学生应完成基本的上机指导编程要求,同时争取多进行编程实践;因为只有通过实际编程才能发现程序设计中的许多问题。不要轻视调试程序的作用,它是深入理解指令功能和程序执行过程的关键。请你不要直接拷贝源程序代码,在一条一条语句的录入编辑过程中,就是书写正确语句、加深语句理解的绝好机会。

最后引用一篇流行于互联网的文章。该文作者毕业于计算机科学与技术专业,若干年的工作经验使其有感而发,写下了洋洋洒洒的“计算机科学与技术反思录”。其中关于汇编语言课程的部分摘录如下:

“汇编语言和微机原理是两门特烦人的课。你的数学/理论基础再好,也占不到什么便宜。这两门课之间的次序就好比先有鸡还是先有蛋,无论你先学哪门,都会牵扯另一门课里的东西。所以,只能静下来慢慢琢磨。这就是典型的工程课,不需要太多的聪明和顿悟,却需要水滴石穿的渐悟。……”

有些人说不想了解计算机体系结构,也不想制造计算机,所以诸如计算机原理,汇编语言,接口之类的课觉得没必要学,这样合理吗?显然不合理,这些东西迟早得掌握,肯定得接触,而且,这是计算机专业与其他专业学生相比的少有的几项优势。做项目的时候,了解这些是非常重要的,不可能说,仅仅为了技术而技术,只懂技术的人最多做一个编码工人,而永远不可能全面地了解整个系统的设计,而编码工人是越老越不值钱的。”

本书第1版由钱晓捷、管红英、穆玲玲、邱保志、关国利等编著,第2版由钱晓捷、穆玲玲修编。新版教材由主编钱晓捷进行修编改写,上述作者和张青、姚俊婷也给予了极大帮助,作者深表感谢。作者还要特别感谢使用本书前续版本的同行教师、学生和读者,大家的宝贵意见和建议催生了新版教材。新版教材仍然离不开大家的支持,希望大家通过本书的教学辅助网站(<http://www2.zzu.edu.cn/qwfw>)或主编的电子邮件(qianxiaojie@zzu.edu.cn)进行教学交流,共同提高“汇编语言程序设计”课程的教学质量。

钱晓捷
2006年6月

反侵权盗版声明

电子工业出版社依法对本作品享有专有出版权。任何未经权利人书面许可,复制、销售或通过信息网络传播本作品的行为;歪曲、篡改、剽窃本作品的行为,均违反《中华人民共和国著作权法》,其行为人应承担相应的民事责任和行政责任,构成犯罪的,将被依法追究刑事责任。

为了维护市场秩序,保护权利人的合法权益,我社将依法查处和打击侵权盗版的单位和个人。欢迎社会各界人士积极举报侵权盗版行为,本社将奖励举报有功人员,并保证举报人的信息不被泄露。

举报电话:(010)88254396;(010)88258888

传 真:(010)88254397

E-mail: dbqq@phei.com.cn

通信地址:北京市万寿路173信箱

电子工业出版社总编办公室

邮 编:100036

目 录

第 1 章 汇编语言基础知识	(1)
1.1 计算机系统概述	(1)
1.1.1 计算机的硬件	(1)
1.1.2 计算机的软件	(3)
1.1.3 计算机的程序设计语言	(3)
1.2 数据表示	(5)
1.2.1 数制	(5)
1.2.2 编码	(7)
1.2.3 有符号数的表示法	(7)
1.2.4 二进制运算	(10)
1.3 Intel 80x86 系列微处理器	(11)
1.4 微型计算机(PC)系统	(13)
1.5 8086 微处理器	(18)
1.5.1 8086 的功能结构	(18)
1.5.2 8086 的寄存器	(19)
1.5.3 存储器组织与段寄存器	(22)
1.6 8086 的寻址方式	(25)
1.6.1 8086 的机器代码格式	(25)
1.6.2 立即数寻址方式	(26)
1.6.3 寄存器寻址方式	(27)
1.6.4 存储器寻址方式	(27)
习题 1	(30)
第 2 章 8086 的指令系统	(32)
2.1 数据传送类指令	(33)
2.1.1 通用数据传送指令	(33)
2.1.2 堆栈操作指令	(35)
2.1.3 标志传送指令	(36)
2.1.4 地址传送指令	(37)
2.1.5 输入输出指令	(38)
2.2 算术运算类指令	(39)
2.2.1 加法指令	(39)
2.2.2 减法指令	(40)
2.2.3 乘法指令	(41)
2.2.4 除法指令	(42)
2.2.5 符号扩展指令	(43)

2.2.6	十进制调整指令	(43)
2.3	位操作类指令	(46)
2.3.1	逻辑运算指令	(46)
2.3.2	移位指令	(48)
2.3.3	循环移位指令	(49)
2.4	控制转移类指令	(50)
2.4.1	无条件转移指令	(51)
2.4.2	条件转移指令	(52)
2.4.3	循环指令	(54)
2.4.4	子程序指令	(55)
2.4.5	中断指令	(57)
2.4.6	系统功能调用	(58)
2.5	串操作类指令	(61)
2.6	处理机控制类指令	(65)
	习题2	(66)
第3章	汇编语言程序格式	(72)
3.1	汇编语言程序的开发	(72)
3.1.1	汇编语言程序的语句格式	(72)
3.1.2	汇编语言的程序格式	(73)
3.1.3	汇编语言程序的开发过程	(75)
3.2	参数、变量和标号	(79)
3.2.1	数值型参数	(79)
3.2.2	变量定义伪指令	(81)
3.2.3	变量和标号的属性	(85)
3.3	程序段的定义和属性	(88)
3.3.1	DOS的程序结构	(88)
3.3.2	简化段定义的格式	(90)
3.3.3	完整段定义的格式	(93)
3.4	复杂数据结构	(97)
3.4.1	结构	(97)
3.4.2	记录	(99)
	习题3	(100)
第4章	基本汇编语言程序设计	(103)
4.1	顺序程序设计	(103)
4.2	分支程序设计	(104)
4.3	循环程序设计	(108)
4.4	子程序设计	(112)
4.4.1	过程定义伪指令	(112)
4.4.2	子程序的参数传递	(114)
4.4.3	子程序的嵌套、递归与重入	(117)

4.4.4 子程序的应用	(120)
习题4	(125)
第5章 高级汇编语言程序设计	(127)
5.1 高级语言特性	(127)
5.1.1 条件控制伪指令	(127)
5.1.2 循环控制伪指令	(129)
5.1.3 过程声明和过程调用伪指令	(131)
5.2 宏结构程序设计	(134)
5.2.1 宏汇编	(134)
5.2.2 重复汇编	(139)
5.2.3 条件汇编	(141)
5.3 模块化程序设计	(144)
5.3.1 源程序文件的包含	(144)
5.3.2 目标代码文件的连接	(149)
5.3.3 子程序库的调入	(151)
5.4 输入输出程序设计	(153)
5.4.1 程序直接控制输入输出	(153)
5.4.2 程序查询输入输出	(155)
5.4.3 中断服务程序	(156)
习题5	(163)
第6章 32位指令及其编程	(166)
6.1 32位指令运行环境	(166)
6.1.1 寄存器	(167)
6.1.2 寻址方式	(169)
6.1.3 机器代码格式	(170)
6.2 32位扩展指令	(172)
6.2.1 数据传送类指令	(172)
6.2.2 算术运算类指令	(174)
6.2.3 位操作类指令	(175)
6.2.4 串操作类指令	(175)
6.2.5 控制转移类指令	(176)
6.3 32位指令的程序设计	(179)
6.4 32位新增指令	(183)
6.4.1 80386 新增指令	(183)
6.4.2 80486 新增指令	(187)
6.4.3 Pentium 新增指令	(188)
6.4.4 Pentium Pro 新增指令	(192)
6.5 用汇编语言编写32位Windows应用程序	(193)
6.5.1 32位Windows应用程序的特点	(194)
6.5.2 32位Windows控制台程序	(194)

6.5.3	MASM32 开发环境	(199)
6.5.4	创建消息窗口	(200)
6.5.5	创建窗口应用程序	(201)
习题 6	(208)
第 7 章	汇编语言与 C/C++ 的混合编程	(211)
7.1	Turbo C 嵌入汇编方式	(211)
7.1.1	嵌入汇编语句的格式	(212)
7.1.2	汇编语句访问 C 语言的数据	(212)
7.1.3	嵌入汇编的编译过程	(214)
7.2	Turbo C 模块连接方式	(215)
7.2.1	混合编程的约定规则	(215)
7.2.2	汇编模块的编译和连接	(217)
7.2.3	混合编程的参数传递	(218)
7.2.4	汇编语言程序对 C 语言程序的调用	(226)
7.3	汇编语言在 Visual C++ 中的应用	(229)
7.3.1	嵌入汇编语言指令	(229)
7.3.2	调用汇编语言过程	(233)
7.3.3	使用汇编语言优化 C++ 代码	(237)
7.3.4	使用 Visual C++ 开发汇编语言程序	(240)
习题 7	(241)
第 8 章	80x87 浮点指令及其编程	(245)
8.1	浮点数据格式	(245)
8.1.1	实数和浮点格式	(245)
8.1.2	80x87 的数据格式	(248)
8.2	浮点寄存器	(249)
8.3	浮点指令的程序设计	(252)
8.3.1	浮点传送类指令	(253)
8.3.2	算术运算类指令	(255)
8.3.3	超越函数类指令	(258)
8.3.4	浮点比较类指令	(260)
8.3.5	FPU 控制类指令	(264)
习题 8	(269)
第 9 章	多媒体指令及其编程	(271)
9.1	MMX 指令系统	(271)
9.1.1	MMX 的数据结构	(271)
9.1.2	MMX 指令	(273)
9.1.3	MMX 指令的程序设计	(281)
9.2	SSE 指令系统	(283)
9.2.1	SIMD 浮点指令	(283)
9.2.2	SIMD 整数指令	(292)

9.2.3	高速缓存优化处理指令	(294)
9.2.4	SSE 指令的程序设计	(295)
9.3	SSE2 指令系统	(299)
9.3.1	SSE2 的数据类型	(300)
9.3.2	SSE2 浮点指令	(301)
9.3.3	SSE2 扩展指令	(307)
9.3.4	SSE2 指令的程序设计	(309)
9.4	SSE3 指令系统	(311)
9.4.1	SSE3 指令	(312)
9.4.2	SSE3 指令的程序设计	(314)
	习题 9	(314)
第 10 章	64 位指令简介	(316)
10.1	64 位方式的运行环境	(316)
10.2	64 位方式的指令	(318)
附录 A	调试程序 CodeView	(321)
附录 B	汇编程序 MASM 的伪指令和操作符	(327)
附录 C	80x86 整数指令系统	(328)
附录 D	输入输出子程序库 IO.LIB	(334)
参考文献	(336)

第 1 章 汇编语言基础知识

程序设计语言是开发软件的工具,它的发展经历了由低级语言到高级语言的过程。汇编语言是一种面向机器的低级程序设计语言。

汇编语言以助记符形式表示每一条计算机指令,每条指令对应着计算机硬件的一个具体操作。利用汇编语言编写的程序与计算机硬件密切相关,程序员可直接对处理器内的寄存器、主存储器的存储单元以及外设的端口等进行操作,从而能够有效地控制硬件。汇编语言程序运行速度快、占用主存容量少,这些都是高级语言无法替代的。所以,作为一个计算机专业人员,必须掌握汇编语言程序设计方法。

本章介绍用汇编语言进行程序设计所需要了解的基本知识,并引出有关基本概念。1.1 节是计算机系统的一般知识,介绍计算机系统的硬件、软件及程序设计语言的发展,特别说明了汇编语言的特点和应用场合。1.2 节简述计算机中数据的表示,诸如二进制和十六进制、BCD 码和 ASCII 码、补码和反码、二进制运算等。本书以 Intel 80x86 系列微处理器的指令系统为例讲解汇编语言程序设计方法,因此,必须对 80x86 微处理器和以它们为核心的 PC 系统有一定的了解,1.3 和 1.4 节分别介绍这两方面的内容。1.5 和 1.6 节展开 8086 微处理器的结构和寻址方式,这些内容是学习指令系统的基础,也是本章的重点。

1.1 计算机系统概述

计算机系统分为硬件和软件两大部分。硬件(Hardware)是计算机系统的机器部分,它是计算机工作的物质基础;软件(Software)则是为了运行、管理和维护计算机而编制的各种程序的总和,广义的软件还应该包括与程序有关的文档。利用汇编语言所编写的程序是软件;但是,每条汇编语言指令都使计算机某个具体硬件部件产生相应的动作;所以,利用汇编语言进行程序设计体现了计算机硬件和软件的结合。

1.1.1 计算机的硬件

计算机的硬件分成五大组成部件:运算器、控制器、存储器、输入设备和输出设备。其中,运算器和控制器是计算机的核心,合称中央处理单元 CPU(Central Processing Unit)或处理机、处理器(Processor);CPU 内部还有一些高速存储单元,被称为寄存器(Register)。输入设备和输出设备往往统称为外部设备(Peripheral),简称外设或 I/O 设备。在微型计算机中,CPU 由一个大规模集成电路芯片构成,被称为微处理器(Microprocessor)。

对汇编语言程序员来说,计算机各部件组成的结构如图 1.1 所示。它们主要由中央处理单元、存储器和输入输出设备组成,各部分之间通过系统总线连接。

1. 中央处理单元

中央处理单元包括运算器、控制器和寄存器。运算器执行所有的算术和逻辑运算;控制器负责把指令逐条从存储器中取出,经译码分析后向机器发出各种控制命令,如取数、执行、运算和存

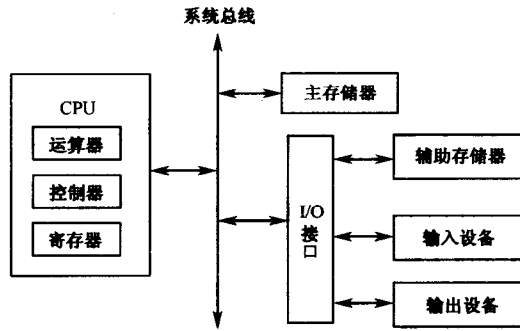


图 1.1 微型计算机的系统组成

数,从而正确完成程序所要求的功能。寄存器为处理单元提供各种操作所需要的数据。

2. 存储器

存储器(Memory)是计算机的记忆部件,它用来存放程序以及程序中所涉及的数据。存储器的内容并不因为被读出而消失,它可以重复取出;只有存入新的信息后,原来的旧信息才会被更改。

按所在位置,存储器可以分成主存储器和辅助存储器,分别被简称为主存(内存)和辅存(外存)。主存储器存放当前正在执行的程序和使用的数据,CPU可以直接存取,它由半导体存储器芯片构成,其成本高、容量小,但速度快。辅助存储器可用于长期保存大量程序和数据,CPU需要通过 I/O 接口访问,它由磁盘或光盘构成,其成本低、容量大,但速度较慢。计算机系统中,主存的容量是有限的,需要辅存来补充;辅存的容量比主存大得多,但存取速度却比主存慢得多。一般来说,程序和数据以文件的形式保存在辅存中,只有在用到它们时,才从辅存读到主存的某个区域中,由中央处理单元控制执行。

按读写能力,存储器可以分成随机存取存储器 RAM 和只读存储器 ROM。CPU 可以从 RAM 读出信息,也可以向 RAM 写入信息,所以 RAM 也被称为读写存储器;而 ROM 中的信息只能被读出,不能被修改。RAM 型半导体存储器可以按地址随机读写,但这类存储器在断电后不能保存信息;ROM 型半导体存储器通常只能被读出,但这类存储器在断电后仍能保存信息。磁盘存储器是可读可写的,而 CD-ROM 光盘则是只读的,这两种存储器都是辅存,都可以长期保存数据,但它们的存取速度都慢于半导体存储器。

主存储器由大量存储单元组成。为了区别每个单元,我们将它们编号,于是,每个存储单元就有了一个唯一的存储器地址(Address)。每个存储单元存放 1 字节量的数据,1 字节(B, byte)包含了 8 个二进制位(b, bit)。

存储容量是指存储器所具有的存储单元个数,其基本单位是字节(B)。为了表达更大的容量,常用的单位是 KB(千字节),MB(兆字节),GB(吉字节)甚至 TB(太字节)。1KB=2¹⁰B=1024B,1MB=2²⁰B,1GB=2³⁰B,1TB=2⁴⁰B。

3. 外部设备

外部设备是实现人机交互和机间通信的一些机电设备。微机系统中,常用的输入设备有键盘、鼠标器等;输出设备有显示器、打印机等;起辅存作用的磁盘和光盘也是以外设的形式连接到系统中的,可以认为它们既是输入设备也是输出设备。

由于外设的种类繁多、工作原理各异,所以每个外设必须通过输入输出接口电路(I/O 接口)与系统连接。程序员所见的 I/O 接口由一组寄存器组成。为了区别它们,对各个寄存器进行了编号,形成 I/O 地址,通常被称做 I/O 端口(Port)。系统实际上就是通过这些端口与外设进行通信的。

1.1.2 计算机的软件

软件是计算机系统的重要组成部分,它可以使机器更好地发挥其功能。软件可分为系统软件和应用软件。

1. 系统软件

系统软件是指为了方便使用、维护和管理计算机系统而编制的一类软件及其文档,它包括操作系统,语言翻译程序等。系统软件是面向计算机系统的、通常由计算机厂家提供的程序及其文档,它是用户使用计算机时为产生、准备和执行用户程序所必需的程序。

在系统软件中,最重要的软件是操作系统。操作系统负责管理整个系统的软硬件资源,向用户提供交互的界面,为所有其他程序的运行打下基础。用户借助操作系统使用计算机系统,程序员也要采用操作系统提供的驱动程序编写用户程序。

程序员采用某种程序设计语言编写源程序,利用语言翻译程序将源程序转变成可运行的程序。例如,本书介绍用汇编语言设计源程序的方法,但必须利用“汇编程序”完成源程序的翻译工作。高级语言则采用编译类或解释类程序来完成这个工作。

2. 应用软件

应用软件是解决某一问题的程序及其文档。它覆盖了计算机应用的所有方面,每个应用都有相应的应用程序。

微机系统具有多种多样的应用软件。例如,进行程序设计时要采用文本编辑软件编写源程序,带有丰富格式的字处理软件帮助用户书写文章,排版软件则用于书刊出版。

大型的程序设计项目往往要借助软件开发工具(包)。这个开发工具是进行程序设计所用到的各种软件的有机集合,所以也被称为集成开发环境。其中,有文本编辑器,有语言翻译程序,有用于形成可执行文件的连接程序,还组合有进行程序排错的调试程序等。

1.1.3 计算机的程序设计语言

进行程序设计的语言有很多,可以分成低级语言和高级语言。低级语言有机器语言和汇编语言,高级语言有 C/C++, Pascal, BASIC 等。

1. 机器语言

计算机能够直接识别的是二进制数 0 和 1 组成的代码。机器指令(Instruction)就是用二进制编码的指令,一条机器指令控制计算机完成一个操作。每种处理器都有各自的机器指令集,某处理器支持的所有指令的集合就是该处理器的指令集(Instruction Set)。指令集及使用它们编写程序的规则被称做机器语言(Machine Language)。

用机器语言形成的程序是计算机唯一能够直接识别并执行的程序,而用其他语言编写的程序必须经过翻译、变换成机器语言程序;所以,机器语言程序常称为目标程序(或目的程序)。

机器指令一般由操作码(Opcode)和操作数(Operand)构成。操作码表明处理器要进行的