

计算机科学与技术系列教材

80X86宏汇编语言程序设计教程

主编 徐爱芸



WUHAN UNIVERSITY PRESS

武汉大学出版社

计算机科学与技术系列教材

80X86宏汇编语言程序设计教程

主 编 徐爱芸

副主编 向 华 马石安



WUHAN UNIVERSITY PRESS

武汉大学出版社

图书在版编目(CIP)数据

80X86 宏汇编语言程序设计教程/徐爱芸主编. —武汉:武汉大学出版社,
2009. 9

计算机科学与技术系列教材

ISBN 978-7-307-07274-9

I. 8… II. 徐… 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(2009)第 141868 号

责任编辑:林 莉 责任校对:王 建 版式设计:支 笛

出版发行:武汉大学出版社 (430072 武昌 珞珈山)

(电子邮件:cbs22@whu.edu.cn 网址:www.wdp.com.cn)

印刷:通山金地印务有限公司

开本:787×1092 1/16 印张:18.25 字数:459千字

版次:2009年9月第1版 2009年9月第1次印刷

ISBN 978-7-307-07274-9/TP·341 定价:29.00元

版权所有,不得翻印;凡购买我社的图书,如有缺页、倒页、脱页等质量问题,请与当地图书销售部门联系调换。

计算机科学与技术系列教材书目

编译原理及其习题解答 (国家精品课程教材)	何炎祥等
Windows 2000 / XP 网络构建与系统管理	王化文等
现代数据库系统及应用教程	尹为民等
计算机操作系统	黄水松等
计算机网络应用设计	黄传河等
C 程序设计导论	谭成予等
新一代多媒体技术与应用	曹加恒等
软件工程	李伟波等
软件工程学习与实践	李伟波等
汇编语言程序设计	何友鸣等
离散数学	刘学书等
离散数学题解	程二虹等
计算机操作系统	刘腾红等
计算机导论	姚爱国等
数据结构	李春葆等
计算机组成原理	戴光明等
人工智能引论	朱福喜等
计算机系统结构	高辉等
计算机系统结构学习辅导及习题解答	高辉等
计算机图形学	李伟波等
算法设计与分析	夏红霞等
数据库原理与技术	尹为民等
电子政务	刘永祥等

C 语言程序设计教程	陈建新等
三维动画制作	向 华等
数据库原理与技术典型题解与实习指导	尹为民等
Java 语言与面向对象程序设计	朱福喜等
网页设计教程	刘 全等
Linux 原理与应用	郑 鹏等
Visual Basic 程序设计	刘 全等
面向对象程序设计 (C++)	张 俊等
面向对象程序设计 (C++) 学习与实践	张 俊等
80X86 宏汇编语言程序设计教程	徐爱芸等

计算机科学与技术系列教材

编 委 会

主 任：何炎祥，武汉大学计算机学院院长，教授

副 主 任：康立山，中国地质大学（武汉）计算机学院院长，教授

陆际光，中南民族大学计算机科学学院院长，教授

编 委：（以姓氏笔画为序）

王江晴，中南民族大学计算机科学学院副院长，教授

王春枝，湖北工业大学计算机学院副院长，教授

牛冀平，黄冈师范学院计算机系主任，副教授

石曙东，湖北师范学院计算机科学与技术系主任，教授

朱 英，桂林电子工业学院计算机系副教授

孙扬波，湖北中医学院信息技术系信息管理与信息系统教研室主任

刘腾红，中南财经政法大学信息学院副院长，教授

陈少平，中南民族大学电信学院副院长，教授

杜友福，长江大学计算机科学学院院长，教授

陆 迟，江汉大学数学与计算机科学学院计算机系主任，副教授

闵华松，武汉科技大学计算机科学与技术学院副院长，副教授

陈佛敏，咸宁学院信息工程学院计算机系主任，副教授

陈建新，孝感学院计算机科学系主任，副教授

李禹生，武汉工业学院计算机与信息工程系副主任，教授

李晓林，武汉工程大学计算机科学与工程学院副院长，副教授

张焕国，武汉大学计算机学院教授

张唯佳，湖北省信息产业厅信息化推进处处长

余敦辉，湖北大学数学与计算机科学学院计算机系副主任

肖 微，湖北警官学院信息技术系副教授

钟 璐，武汉理工大学计算机科学与技术学院院长，教授

钟阿林，三峡大学电气信息学院计算机系主任

姜洪溪，襄樊学院电气信息工程系副主任，副教授
桂 超，湖北经济学院计算机与电子科学系副主任，副教授
黄求根，武汉科技学院计算机科学学院院长，教授
阎 菲，湖北汽车工业学院计算中心主任，副教授
韩元杰，桂林电子工业学院计算机系教授
谢坤武，湖北民族学院信息工程学院计算机系主任，副教授
戴光明，中国地质大学（武汉）计算机学院副院长，教授
魏中海，华中农业大学理学院计算机系副教授

执行编委：林 莉，武汉大学出版社计算机图书事业部主任
支 笛，武汉大学出版社计算机图书事业部编辑



内 容 提 要

本书以目前广泛使用的 PC 机为平台，详细介绍了宏汇编语言的基本概念、程序设计的基本方法与技巧。全书共 10 章，主要内容包括：宏汇编语言、程序设计的基本技巧、输入/输出程序设计、32 位 CPU 扩展功能、汇编语言上机内容与要求。

本书内容详实，叙述细腻易懂，在章节安排上由浅入深，指令介绍与指令应用紧密结合，使得知识易于理解和掌握。

本书可作为高等院校计算机及相关专业“汇编语言程序设计”课程教材，也可作为独立学院、高职高专计算机及相关专业、有关工程技术人员的教学参考书。

前 言

《80X86 宏汇编语言程序设计教程》是高等院校计算机硬、软件及应用专业学生都必须学习的核心课程之一，它是计算机组成原理、操作系统及其他核心课程的基础课，也是微机原理、单片机应用等课程的学习基础。同时，该课程对于训练学生掌握程序设计技术、熟悉上机操作和程序调试技术都有重要作用。

《80X86 宏汇编语言程序设计教程》的内容十分丰富，知识量大，是一门教师普遍感到难教、学生普遍感到难学的课程。为了满足教师和学生的教学需要，作者根据本课程的特点及知识结构，在多年教授这门课程的教学经验的基础上编写了此书，以期能帮助学生快速熟练地掌握相关知识，同时为教师的教学提供指导。

本书共 10 章，分为两大部分和两个附录，各部分内容如下：

第一部分：第 1~9 章。详细介绍了 16 位 80X86 CPU 的宏汇编程序设计。第 1 章基础知识，介绍了学习汇编语言程序设计所需的数据编码与运算等若干预备知识。第 2 章 8086/8088 计算机内部体系结构，介绍 8086/8088 CPU 内部结构和存储器管理。第 3 章汇编语言语法及七种寻址方式，主要介绍了宏汇编语言中的各种符号、表达式、伪指令语句、寻址方式及常用的系统功能调用。第 4 章汇编语言指令系统，全面介绍了 8086 指令系统中数据传送指令、算术运算指令、逻辑运算指令、处理器控制指令。第 5 章汇编语言基本程序设计，系统介绍了程序的三种基本结构（顺序结构、分支结构及循环结构）及串操作指令的设计方法与技巧，同时对实现这些结构的转移指令、循环指令和串指令也作了详细介绍。第 6 章汇编语言模块化程序设计，介绍了子程序及模块化程序设计。第 7 章高级汇编语言程序设计，介绍了宏的定义与调用方法、重复汇编、条件汇编等技术，特别介绍了汇编语言与高级语言的混合编程。第 8 章输入/输出程序设计，介绍了输入/输出数据传送方式、中断技术、软中断程序设计及 DOS 与 BIOS 中断调用。第 9 章 32 位机新增指令及功能，以典型的 32 位 80386 CPU 为例，介绍了 32 位 CPU 的工作模式、内部寄存器、内存管理、寻址方式、新增指令等扩展功能。

第二部分：第 10 章实验指导，介绍了汇编语言上机实验，主要用于指导读者上机实验，提高编程能力，同时培养学生发现问题、解决问题的能力。

附录 A：ASCII 码。

附录 B：调试工具 DEBUG。

本书语言通俗易懂、讲叙细腻、内容丰富、例题典型、结构安排合理，且注重理论知识与实际训练相结合，具有较强的适应性，对学生学好汇编语言的相关知识有较大的辅助作用。

本书可作为计算机及相关专业本、专科学生《汇编语言程序设计》课程的学习和复习资料，也可作为软件工程技术人员的学习参考书和研究生入学考试的应试辅导资料。

为方便教学，本书配有电子教案，任课教师如有需要，可与武汉大学出版社联系，免费索取。



本书在写作的过程中得到了江汉大学有关领导和同行的指导与大力支持，谨在此表示衷心的感谢。

由于编写时间仓促，水平有限，虽然经过严格的审核，但书中不免还有疏漏和不足之处，敬请各位读者批评指正。

作 者

2009年6月

目 录

第 1 章 基础知识	1
1.1 计算机语言概述	1
1.1.1 机器语言	1
1.1.2 汇编语言	2
1.1.3 高级语言	3
1.1.4 非过程性语言	3
1.1.5 汇编语言的特点	4
1.2 计算机中数据的表示方法	4
1.2.1 进位计数制	4
1.2.2 数制转换	5
1.2.3 计算机中数据的表示	7
习题 1	14
第 2 章 8086/8088 计算机系统组织	15
2.1 8086/8088 CPU 的内部结构	15
2.1.1 8086 CPU 的功能结构	15
2.1.2 8086 CPU 的寄存器结构	15
2.2 存储器组织	19
2.2.1 存储器的地址和内容	19
2.2.2 存储器的分段	21
2.3 堆栈	26
2.3.1 堆栈的特点	26
2.3.2 堆栈的作用	26
2.3.3 堆栈的操作	27
习题 2	29
第 3 章 汇编语言语法与寻址方式	31
3.1 汇编语言语句类型及格式	31
3.1.1 语句类型	31
3.1.2 指令语句格式	31
3.2 伪指令	32
3.2.1 段定义伪指令	32
3.2.2 数据定义	35
3.2.3 符号定义伪指令	38
3.2.4 与地址有关的伪指令	40
3.2.5 常用伪指令	42
3.3 表达式	45

3.3.1	常量	45
3.3.2	数值表达式	46
3.3.3	地址表达式	47
3.4	寻址方式	47
3.4.1	立即数寻址方式	48
3.4.2	寄存器寻址方式	49
3.4.3	存储器型寻址方式	49
3.4.4	外设型寻址方式	55
3.5	汇编语言源程序结构	56
3.6	常用 DOS 功能调用	57
	习题 3	59
第 4 章 汇编语言指令系统		63
4.1	数据传送指令	63
4.2	算术运算指令	70
4.3	位操作指令	80
4.4	处理器控制指令	85
	习题 4	86
第 5 章 汇编语言基本程序设计		91
5.1	汇编语言程序设计步骤	91
5.2	顺序结构程序设计	92
5.3	分支结构程序设计	96
5.3.1	转移地址的寻址方式	96
5.3.2	无条件转移指令	97
5.3.3	条件转移指令	98
5.3.4	分支结构程序设计	104
5.4	循环结构程序设计	113
5.4.1	循环控制结构	114
5.4.2	循环控制指令	115
5.4.3	循环程序设计方法	120
5.4.4	多重循环程序设计举例	126
5.5	串操作程序设计	129
5.5.1	串操作概述	129
5.5.2	串操作指令及程序设计	132
	习题 5	138
第 6 章 汇编语言模块化程序设计		142
6.1	子程序设计	142
6.1.1	子程序概述	142
6.1.2	子程序的调用与返回指令	143
6.1.3	子程序的定义	145
6.1.4	主程序与子程序的参数传递	150
6.1.5	子程序的嵌套	156



6.2 模块化程序设计	159
6.2.1 模块的装配连接	160
6.2.2 模块间的通信	165
6.2.3 模块化程序设计举例	166
习题 6	170
第 7 章 高级汇编语言程序设计	173
7.1 宏汇编	173
7.1.1 宏指令的使用	173
7.1.2 宏与子程序的区别	176
7.1.3 与宏有关的伪指令	176
7.1.4 宏库的建立与使用	178
7.2 重复汇编	180
7.3 条件汇编	183
7.4 数据结构	185
7.5 数据记录	187
7.6 汇编语言与高级语言的连接	188
7.6.1 嵌入汇编语言	189
7.6.2 多模块混合编程	190
习题 7	194
第 8 章 输入/输出程序设计	196
8.1 输入/输出概述	196
8.1.1 I/O 端口的编址方式	197
8.1.2 数据传送方式	197
8.2 数据的输入/输出	199
8.2.1 I/O 指令	199
8.2.2 程序查询 I/O 方式	199
8.2.3 中断 I/O 方式	201
8.3 键盘输入控制	210
8.4 屏幕输出控制	212
8.5 打印机输出控制	220
8.6 磁盘文件操作	225
8.6.1 文件概述	226
8.6.2 文件操作的系统功能调用	227
8.6.3 文件管理程序设计	228
习题 8	229
第 9 章 32 位机新增指令及功能	232
9.1 80386 的工作机制	232
9.1.1 80386 CPU 的内部结构	232
9.1.2 80386 寄存器	232
9.1.3 80386 存储器管理	235
9.1.4 存储模型与段的简化定义	240

9.1.5	80386 的寻址方式	241
9.1.6	80386 新增指令	242
9.2	80486 系统	246
9.2.1	80486 CPU 结构	246
9.2.2	80486 的内存管理和高速缓存	246
9.2.3	80486 扩充指令	247
9.3	Pentium 系统	248
9.3.1	Pentium CPU 结构	248
9.3.2	Pentium 扩充指令	249
9.4	程序举例	250
	习题 9	252
第 10 章	实验指导	254
10.1	汇编语言上机过程及调试工具	254
10.2	分支程序设计	258
10.3	循环程序设计	259
10.4	子程序设计(一)	260
10.5	子程序设计(二)	263
10.6	字符处理程序设计	264
10.7	中断程序设计	264
10.8	磁盘文件管理程序设计	266
10.9	模块化程序设计	267
10.10	综合程序设计	269
附录 A	ASCII 码表	271
附录 B	调试工具 DEBUG	273
	参考文献	278

第1章 基础知识

自1946年第一台电子计算机问世以来,计算机已被广泛地应用于生产、生活的各个领域,推动着社会的进步与发展。特别是Internet出现后,传统的信息收集、传输及交换方式发生了革命性的改变。

计算机科学的发展依赖于计算机硬件和软件技术的发展,硬件是计算机的躯体,软件是计算机的灵魂,软件是用计算机语言编写的。计算机语言的发展经历了由机器语言(Machine Language)、汇编语言(Assembly Language)、高级语言(High Level Language)到非过程语言(Non-procedural Language)的历程。

1.1 计算机语言概述

1.1.1 机器语言

机器语言是计算机唯一能直接识别和执行的计算机语言。由于构成计算机的硬件本身的各个部件是基于地值逻辑的,这些部件只能识别“0”和“1”两个状态,其功能就是记忆、传输和加工二进制信息“0”或“1”。在计算机发展的初期,人们使用二进制代码来编写程序,使用二进制代码构成机器指令,这种二进制编码的计算机语言就是机器语言。机器语言描述的程序称为目标程序,只有目标程序CPU才能直接执行。指令是用于指出计算机进行的操作和操作对象的一组代码,一条机器指令通常由操作码和操作数两部分组成。其中,操作码指出计算机进行的具体操作,如做加法、减法等;操作数说明操作的对象。操作码比较简单,只需对每一种操作指定确定的二进制代码就可以了;操作数比较复杂,首先它可以有一个、二个或三个,分别称为单操作数、双操作数或三操作数,其次,操作数可能存放在不同的地方,既可存放在寄存器中,还可以存放在存储器中,甚至直接放在指令中。下面的二进制代码序列就是一条Intel 8086的机器指令:

39	24	23	0
10000000	00000110	01100100	00000000 00010100

指令中的前16位10000000进00000110为操作码,指出该指令为加法操作;后24位为操作数,第一个加数是00000000 01100100,表示操作数存放在内存地址为100的字节单元中,第二个加数量是二进制代码00010100,表示操作数20,它直接写在指令中。处理器逐条取指令、分析指令、执行指令,来完成人们所赋予的任务。

机器语言与汇编语言或高级语言相比,其执行效率高。但其可读性差,不易记忆;编写

程序既难又繁，容易出错；程序调试和修改难度巨大，不容易掌握和使用。此外，因为机器语言直接依赖于中央处理器，所以用某种机器语言编写的程序只能在相应的计算机上执行，无法在其他型号的计算机上执行，也就是说，可移植性差。

1.1.2 汇编语言

为了避免使用机器语言编程的烦琐，出现了汇编语言。汇编语言是用比较容易识别、记忆的助记符替代相应机器指令的操作码和操作数，并按照一定的格式书写的一种面向机器的程序设计语言。这样人们就能较容易地读懂程序，调试和维护也更方便了。但这些助记符号计算机无法识别，需要一个专门的程序将其翻译成机器语言，这种翻译程序被称为汇编程序。汇编语言指令与机器指令基本上是一一对应的，它依赖于具体的处理器体系结构，不能通用，处理器不同，相应的汇编语言就不同，因此，汇编语言也是低级语言。例如，对于前面的例子，用宏汇编语言来书写则应为：

```
ADD WORD PTR DS : [100] , 20
```

其中，ADD 为助记符，DS : [100] 表示在当前数据段中、偏移地址为 100 单元中的内容是目的操作数，而源操作数是 20，相加的结果存入目的操作数所在的单元中。

比较两种机器语言，可以看出，汇编语言比机器语言要直观得多，方便人们的书写与记忆。汇编语言能准确发挥计算机硬件的功能和特长，程序精炼而质量高，占用内存的字节数较少，执行效率仍接近于机器语言，因此，汇编语言至今仍是一种常用的软件开发工具。计算机中凡涉及与硬件有关的输入/输出和控制方面的程序，以及在单机控制、仪器、仪表及家用电器中用于控制方面的程序，大部分都是用汇编语言编写的。

用汇编语言编写的程序称为汇编源语言，计算机不能直接识别，必须将其翻译成由机器指令组成的程序后，CPU 才能执行，这一过程称为“汇编”。用于将汇编源程序翻译成机器语言的程序称为汇编程序，这种由源程序经过机器翻译转换成的机器语言程序也称目标程序。目标程序还不能直接交给 CPU 执行，它还需要通过连接程序装配成可执行程序才能被执行。它们之间的关系如图 1-1 所示。

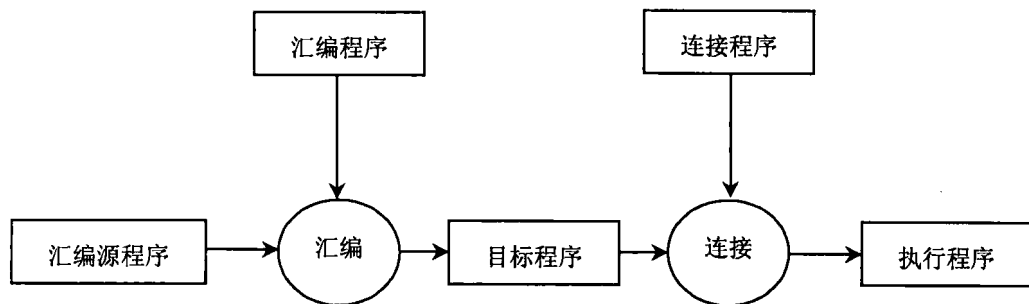


图 1-1 汇编程序与目标程序、可执行程序之间的关系

汇编程序 MASM.EXE 是一种专门用于将 Intel 8086 的汇编语言源程序翻译成机器语言程序的翻译器，它在执行汇编时首先进行语法检查，指出错误的位置及类型，编程人员要一一进行修改，直到源程序无错，汇编程序才执行翻译工作，因此，汇编程序决定了汇编语言的语法。连接程序 LINK.EXE 具有将多个目标程序装配在一起的功能，它也可以将目标程序与预先编写好的一些放在子程序库中的子程序连接在一起，构成较大的执行文件。汇编语言

的源程序一般以 .ASM 作为文件扩展名, 目标程序以 .OBJ 作为文件扩展名, 执行程序以 .EXE 作为文件扩展名。

1.1.3 高级语言

尽管汇编语言比机器语言方便, 但汇编语言仍然具有许多不便之处, 程序编写的效率远远不能满足需要。1954 年, 第一个高级语言 FORTRAN 问世了。高级语言是一种与具体的计算机硬件无关、独立于机器的通用语言, 它比较接近人类自然语言的语法及数学表达式, 人们用高级语言编程不必了解和熟悉机器的指令系统, 更容易被人们掌握和使用。高级语言采用接近自然语言的词汇, 其程序的通用性强, 易学易用, 这些语言面向计算机求解问题的过程, 不依赖具体机器, 与特定机器相分离。计算机不能直接执行高级语言程序, 高级语言程序必须先翻译转换成“目标程序”(即机器语言程序)才能被执行。这种翻译转换工作由被称为“编译程序”的专门软件来完成, 其过程如图 1-2 所示。

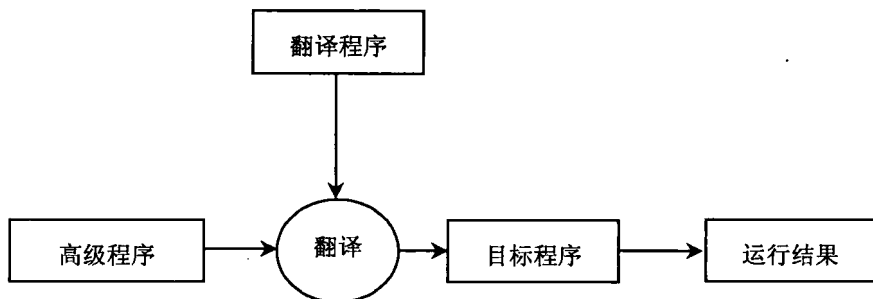


图 1-2 高级语言编译过程

高级语言翻译成机器语言有两种方式, 一种是先把高级语言程序翻译成机器语言程序, 然后在机器上执行, 这种翻译称为编译方式, 大多数高级语言如 PASCAL 语言、C 语言等都是采用这种方式; 另一种方式是直接把高级语言程序在机器上运行, 一边解释一边执行, 这种翻译方式称为解释, 如 BASIC 语言就采用这种方式。每一种高级语言都有自己的编译程序, 在一台计算机上运行某一种高级语言源程序的前提是该计算机系统配置了该语言的编译程序。

高级语言源程序在不对硬件编程时, 是在未考虑机器结构的特点下编写的, 经过翻译后的目标程序往往不够精练, 过于冗长, 加大了目标程序的长度, 导致存储空间占用较大和运行时间较长。

1.1.4 非过程性语言

面向过程的高级语言称为过程性语言, 用它编程序必须写出每一步如何进行的全过程。程序设计者必须具体指出执行的每一个细节(例如, 输入一个数给某一变量、进行某一公式的运算、进行什么条件判断、执行多少次循环等)。这要求程序设计人员考虑得十分周到, 稍有不慎(例如写错一个字母), 就会导致程序运行失败。

人们只要指出“做什么”, 而不必具体指出“如何做”, 由计算机自己去解决“如何做”的问题, 这就是“非过程性语言”(即不需指出解决问题的过程)。20 世纪 70 年代后期, 计