

21
世纪

高等学校计算机应用型本科规划教材精选



汇编语言程序设计



朱耀庭 董焕芝 高飞 编著



清华大学出版社

013071084

TP313
146-2

21 世纪高等学校计算机应用型本科规划教材精选

汇编语言程序设计

朱耀庭 董焕芝 高飞 编著



清华大学出版社
北京



北航

C1680231

TP 313

146-2

内 容 简 介

本书详细讲述针对 80x86 微处理器的汇编语言程序设计方法,主要包括导论、80x86 实模式汇编的运行环境、8086 微处理器及其寻址方式、汇编语言程序设计基础、数据传送指令与顺序程序设计、算术运算指令与程序设计、逻辑运算、分支程序与循环程序设计、串操作指令与程序设计、过程、中断及中断处理程序、端口输入/输出与外设编程。

本书是计算机科学与技术、数字媒体技术及其相关专业本科生“汇编语言程序设计”课程的教材,特别适合应用型本科使用,也可供高职高专和各行各业计算机工作人员作为自学汇编语言程序设计的重要参考书。

全书贯穿学以致用用的指导原则,注重实例教学,通过大量实例引导读者由易到难一步步掌握汇编语言程序设计的核心技术。为方便上机操作,同时出版了与本教材配套的上机实习指导,供师生使用。无论是初学者还是专业人士,都可以从中得到启迪,熟悉并且掌握汇编语言程序设计的方法。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

汇编语言程序设计/朱耀庭,董焕芝,高飞编著.--北京:清华大学出版社,2013

21 世纪高等学校计算机应用型本科规划教材精选

ISBN 978-7-302-32464-5

I. ①汇… II. ①朱… ②董… ③高… III. ①汇编语言—程序设计—高等学校—教材
IV. ①TP313

中国版本图书馆 CIP 数据核字(2013)第 105115 号

责任编辑:付弘宇 王冰飞

封面设计:杨 兮

责任校对:李建庄

责任印制:杨 艳

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社 总 机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 装 者:北京国马印刷厂

经 销:全国新华书店

开 本:185mm×260mm

印 张:18.5 字 数:457 千字

版 次:2013 年 9 月第 1 版

印 次:2013 年 9 月第 1 次印刷

印 数:1~2000

定 价:33.00 元

产品编号:053456-01

编写委员会成员

(按姓氏笔画排序)

王慧芳	朱耀庭	孙富元	高福成
常守金	张洪定	王志军	渠丽岩
赵培军	于 萍	姬秀娟	桑 婧

序

PREFACE



“教

育部财政部关于实施高等学校本科教学质量与教学改革工程的意见”(教高[2007]1号)指出:“提高高等教育质量,既是高等教育自身发展规律的需要,也是办好让人民满意的高等教育、提高学生就业能力和创业能力的需要”,特别强调“学生的实践能力和创新精神亟待加强”。同时要求将教材建设作为质量工程的重要建设内容之一,加强新教材和立体化教材的建设;鼓励教师编写新教材,为广大教师和学生提供优质教育资源。

“21世纪高等学校计算机应用型本科规划教材精选”就是在实施教育部质量工程的背景下,在清华大学出版社的大力支持下,面向应用型本科的教学需要,旨在建设一套突出应用能力培养的系列化、立体化教材。该系列教材包括各专业计算机公共基础课教材;包括计算机类专业,如计算机应用、软件工程、网络工程、数字媒体、数字影视动画、电子商务、信息管理等专业方向的计算机基础课、专业核心课、专业方向课和实践教学的教材。

应用型本科人才教育重点是面向应用、兼顾继续深造,力求将学生培养成为既具有较全面的理论基础和专业基础,同时也熟练掌握专业技能的人才。因此,本系列教材吸纳了多所院校应用型本科的丰富办学实践经验,依托母体校的强大教师资源,根据毕业生的社会需求、职业岗位需求,适当精选理论内容,强化专业基础、技术和技能训练,力求满足师生对教材的需求。

本丛书在遴选和组织教材内容时,围绕专业培养目标,从需求逆推内容,体现分阶段、按梯度进行基本能力→核心能力→职业技能的培养;力求突出实践性,实现教材和课程系列化、立体化的特色。

突出实践性。丛书编写以能力培养为导向,突出专业实践教学内容,为有关专业实习、课程设计、专业实践、毕业实践和毕业设计教学提供具体、翔实的实验设计,提供可操作性强的实验指导,完全适合“从实践到理论再到应用”、“任务驱动”的教学模式。

教材立体化。丛书提供配套的纸质教材、电子教案、习题、实验指导和案例,并且在清华大学出版社网站(<http://www.tup.com.cn>)提供及时更新的数字化教学资源,供师生学习

与参考。

课程系列化。实验类课程均由“教程+实验指导+课程设计”三本教材构成一门课程的“课程包”，为教师教学、指导实验以及学生完成课程设计提供翔实、具体的指导和技术支持。

希望本丛书的出版能够满足国内对应用型本科学生的教学要求，并在大家的努力下，在使用中逐渐完善和发展，从而不断提高我国应用型本科人才的培养质量。

丛书编委会

2009年7月

前言

FOREWORD



如

果说学习生命科学不了解 DNA 不行,学习物理科学不了解基本粒子不行,那么在已经步入数字化信息时代的今天,学习信息科学不了解 bit 和 bit 流绝对不行。数字化信息时代的各种媒体:文字、图形、图像、动画、声音、视频等,统统都用 bit 表示,网络和网络上各种信息的传输及其安全保障全都离不开 bit 和 bit 流。计算机技术的发展已经深入到了媒体领域,图文声像及其涉足的新媒体领域都离不开数字媒体技术与数字媒体艺术。因此了解和掌控计算机中 bit 和 bit 流就成为计算机应用、数字媒体技术、数字媒体艺术及其应用的不可或缺的基本技能。现在计算机的应用也不仅仅是科学计算和信息处理,计算机的发展已经进入了后 PC 阶段,各种自动化设备、仪器仪表、通信设备、智能家电产品、电子声像文字设备等,复杂到机器人,简单到玩具,都离不开微处理器,而这一切又都与嵌入式系统密切相关。因此了解微处理器的物理结构和逻辑结构,掌握计算机底层编程技术,特别是嵌入式系统的原理,就显得尤为重要。而以上技术的掌握仅仅靠高级语言程序设计是不能解决问题的,必须掌握汇编语言。因为汇编语言与机器语言一样都是面向机器的语言,可以操作到计算机的底层,而汇编语言较之机器语言又容易掌握,因此国内外所有计算机相关专业都将汇编语言程序设计作为主修课程之一。以 80x86 为 CPU 的个人计算机毫无疑问已经成为当代的主流计算机,以 80x86 为 CPU 的个人计算机随处可见,因此从 20 世纪 90 年代初开始,国内汇编语言程序设计课程普遍选择 80x86 汇编语言。尽管不同微处理器的汇编语言并不相同,但原理是一样的。鉴于 80x86 微处理器的典型性和复杂程度,应该说学习了 80x86 汇编语言就可以很容易掌握其他微处理器的汇编语言,例如各种单片机的汇编语言;学习了 PC 的操作系统引导机制,也就不难理解各种嵌入式系统的设计原理。

早在 1986 年国内许多院校还在开设 Z80 汇编语言课程的时候,作者本人就已经在南开大学计算机专业和自动控制专业开设了 8086/8088 汇编语言,该课程曾经获得南开大学优秀课程和天津市教学成果二等奖。也是在 20 世纪 90 年代初,天津市应用技术研究所在承德举办了全国性的计算机应用培训班,作者应邀讲授了如何使用 IBM PC 和 80x86 汇编语言。由作者与于春凡教授编写、南开大学出版社出版的《IBM-PC(INTEL 8088/80x86)宏汇编语言程序设计》已经在全国发行 10 万册以上,同时为大专生编写的《汇编语言程序设计》也已经再版和多次印刷。从 2004 年开始本书作者又在南开大学滨海学院专门为计算机科学系讲授该课程,并且应清华大学出版社之邀,于 2009 年出版了《汇编语言程序设计》。这次几名任课教师又从培养应用型人才的角度出发,从应用型本科学生的实际情况出发,完成了本书的编写。教材内容充分考虑了微处理器的发展现状和应用需求,精心挑选针对性强的实例,重新组织、重新编排,每章都根据教学内容的需要安排了大量实例,许多题目都经过

几名作者的反复调试。为增强上机实习的针对性和提高实习的效果,我们还同时编写出版了与本书配套的上机实习指导。为授课教师教学方便,我们同时也提供了电子讲稿。这些改进相信会对读者有所帮助。

初稿的整体架构由朱耀庭教授在原主编的《汇编语言程序设计》的基础上修改完成,董焕芝副教授负责第1章至第6章,以及第11章、第12章的编写和例题调试;高飞老师负责第7章至第10章的编写和例题调试。上机实习指导初稿的整体架构由朱耀庭教授设计。实验1至实验7由董焕芝副教授编写,实验8至实验12由高飞老师编写。全书最后由朱耀庭教授统编和修改完成。

本书从组稿、编审到出版始终得到了清华大学出版社的支持,特别是索梅编辑、贾斌编辑为此付出了辛勤的劳动,在此表示衷心感谢。

本书的出版得到了南开大学滨海学院的大力支持,在此一并致谢。

编者

2013年6月于天津南开大学滨海学院

目 录

CONTENTS



第 1 章 导论	1
1.1 计算机语言	1
1.1.1 面向机器的计算机语言	1
1.1.2 非面向机器的计算机语言	4
1.2 计算机中数的表示	5
1.2.1 数的表示	6
1.2.2 数制转换	8
1.2.3 ASCII 码	9
1.2.4 数的补码表示	10
1.2.5 IEEE 浮点数	12
1.2.6 从不同角度看待一个二进制数	16
1.3 理解和认识汇编语言	17
1.3.1 从设计师到设计师兼建筑师	17
1.3.2 冯·诺依曼计算机的设计思想	17
1.3.3 从高级语言的局限性看汇编语言	19
1.3.4 汇编语言的主要特性	20
习题 1	21
第 2 章 80x86 实模式汇编的运行环境	23
2.1 DOS 下 Edit 的使用	23
2.1.1 汇编源程序文件的编辑及相关知识	23
2.1.2 用 Edit 编辑汇编源程序	24
2.2 用 MASM(或 ASM)运行汇编语言程序	26
2.2.1 MASM 汇编语言程序的调试步骤	26
2.2.2 MASM 典型实例	28
2.3 用 DEBUG 调试汇编程序	31
2.3.1 用 DEBUG 运行汇编语言程序	31
2.3.2 典型 DEBUG 命令剖析	32
2.3.3 DEBUG 命令综述	36
2.3.4 DEBUG 命令综合应用实例	41
2.4 EMU8086 的使用	45

2.4.1	EMU8086 简介	45
2.4.2	EMU8086 的安装及使用	46
2.4.3	EMU8086 显示模拟	46
2.4.4	EMU8086 应用实例	47
习题 2		49
第 3 章 8086 微处理器及其寻址方式		51
3.1	8086CPU 的基本逻辑结构	51
3.1.1	8086 的基本逻辑结构	51
3.1.2	理解并运用基本逻辑结构图	52
3.1.3	8086CPU 的运行特点	52
3.1.4	8086 寄存器组	53
*3.2	指令与数据	55
3.3	8086 的存储器分段结构	56
3.3.1	8086 的存储器分段	56
3.3.2	段寄存器的引用	59
3.4	堆栈	61
3.4.1	什么是栈	61
3.4.2	8086 的栈机制	61
3.5	寻址方式	63
3.5.1	汇编指令	63
3.5.2	指令中的操作数	64
3.5.3	寻址规则	66
习题 3		72
第 4 章 汇编语言程序设计基础		74
4.1	源程序的书写格式	74
4.1.1	简化段定义格式	74
4.1.2	完整段定义格式	77
4.2	汇编语言的语句	82
4.3	汇编语言的数据组织及其访问	84
4.4	至今所用 DOS 功能调用及程序正常结束的方式	89
习题 4		91
第 5 章 数据传送指令与顺序程序设计		93
5.1	数据传送指令	93
5.1.1	数据传送指令分类	93
5.1.2	MOV 指令与顺序程序实例	94
5.1.3	堆栈操作指令	99

5.1.4	数据交换指令	102
5.1.5	查表转换指令	105
5.2	地址目标传送指令	107
5.3	标志位传送指令	110
5.4	端口输入/输出指令	111
5.5	顺序程序设计	111
	习题 5	112
第 6 章	算术运算指令与程序设计	114
6.1	算术运算指令、逻辑运算指令及其对标记寄存器的影响	114
6.1.1	算术运算指令	114
6.1.2	算术运算指令对标志寄存器的影响	115
6.1.3	条件转移指令与标志寄存器的关系	118
6.2	算术运算的数据格式	119
6.3	二进制数加减运算	120
6.3.1	二进制数加法	120
6.3.2	二进制数减法	123
6.3.3	二进制数加减运算的有效性	125
6.4	二进制数乘除运算	126
6.4.1	无符号二进制数乘除运算	126
6.4.2	带符号二进制数乘除运算	129
*6.5	非压缩型 BCD 码运算	133
*6.6	压缩型 BCD 码加减运算	138
	习题 6	141
第 7 章	逻辑运算	143
7.1	逻辑运算指令介绍	143
7.2	逻辑运算指令	144
7.3	移位指令	146
7.4	循环移位指令	149
7.5	逻辑运算指令应用	151
	习题 7	154
第 8 章	分支程序与循环程序设计	156
8.1	分支	156
8.1.1	标号	156
8.1.2	程序转移指令	158
8.1.3	无条件转移指令	159
8.1.4	条件转移	162

8.2	程序设计流程图	164
8.3	循环程序设计	166
8.3.1	循环控制指令	166
8.3.2	单重循环	170
8.3.3	多重循环	173
习题 8		176
第 9 章	串操作指令与程序设计	178
9.1	字符串操作综述	178
9.2	字符串操作指令	182
9.3	串操作指令的重复前缀	190
习题 9		191
第 10 章	过程	193
10.1	过程的定义与说明	193
10.2	过程调用	194
10.3	过程的返回	200
10.4	模块间的调用和转移	201
10.5	寄存器的保护和恢复	204
10.6	调用程序与被调用过程之间的数据传送	206
10.7	递归	208
10.8	过程部分综合实例	210
习题 10		214
第 11 章	中断及中断处理程序	216
11.1	中断及中断相关概念	216
11.1.1	中断的基本概念	216
11.1.2	中断向量表	216
11.1.3	中断源及中断源的识别	218
11.1.4	中断的分类	219
11.1.5	中断的优先级	221
11.1.6	中断嵌套	222
11.2	软中断	222
11.2.1	软中断的处理过程	222
11.2.2	软中断的分类	223
11.2.3	软中断的访问	225
11.3	DOS 中断服务和 BIOS 中断服务	226
11.3.1	DOS 中断服务	226
11.3.2	BIOS 中断服务	228

11.3.3	DOS 和 BIOS 功能调用实例	230
11.4	典型的中断处理程序实例	246
11.4.1	中断处理程序的编程原则	246
11.4.2	读取和设置中断向量	246
11.5	PC 引导机制	250
11.5.1	加电自检 POST 操作	250
11.5.2	操作系统加载	252
11.5.3	嵌入式系统与汇编	254
习题 11		255
第 12 章	端口输入/输出与外设编程	258
12.1	端口	258
12.2	8259A、8255、8253 典型芯片编程	259
12.2.1	8259A 芯片编程	259
12.2.2	8255 芯片编程	261
12.2.3	8253 芯片编程	264
12.3	综合实例	266
习题 12		270
附录 A	ASCII 表	271
附录 B	键盘扫描码 set1	275
附录 C	INT 21H 指令说明及使用方法	277
参考文献		282



导 论

汇编语言是面向机器的计算机语言。学习汇编语言必须了解计算机中机器指令、数据、符号的表示形式和存储格式,以及机器指令程序的运行方式等。而这些又都与现代计算机的基本工作原理,即冯·诺依曼原理有关。本章将详细介绍上述内容,使读者能够深入了解二进制数在计算机中的重要地位,掌握汇编语言与高级语言的区别和联系。

1.1 计算机语言

从电子计算机的诞生至今,已经有成百上千种计算机语言面世。然而只有一种计算机语言程序能够在计算机上直接执行,那就是机器语言。除机器语言之外的任何一种计算机语言程序,只有经过翻译变成机器语言程序后,才能够在计算机上直接执行。机器语言具有任何其他语言无法代替的特点,因此任何一个想真正了解计算机底层工作原理的人,任何一个想真正驾驭计算机的人,任何一个想对计算机设计进行改进或有所创新的人,都必须了解或掌握机器语言。机器语言通过代码表示一个一个的机器指令,掌握起来相当困难。汇编语言则用助记符表示机器指令,记忆起来相对方便。因此掌握了汇编语言就等价于掌握了机器语言。

1.1.1 面向机器的计算机语言

计算机语言有多种分类方法,按照是否面向机器可以分成面向机器的语言和非面向机器的语言。面向机器的语言又分为机器语言和汇编语言。面向机器的语言也称做计算机低级语言。其他计算机语言均为非面向机器的语言。非面向机器的语言又有多种分类方法:按照翻译方式可以分为编译型语言 and 解释型语言;按照其用途可以分为各种用途的语言,例如数据库管理语言、算法语言、人工智能语言等;相对于低级语言,可以分为高级语言、中级语言;按照程序设计方法可以分为结构型语言、面向对象语言等。

1. 机器语言

机器语言程序由机器指令构成,这些机器指令一般按照执行顺序依次排列,依次执行完

成某个特定的计算或任务。

机器指令是 CPU 能直接识别并执行的指令。机器指令由物理部件的若干位二进制 0、1 状态体现,机器指令程序存储在内存中,正确的机器指令一旦送入 CPU 被译码后就将变成动作予以执行。任何计算机语言程序最终都必须变成机器语言程序才能够执行。

机器指令面向机器,因机器而异,如图 1-1 所示。不同的机器具有不同的机器指令,因而具有不同的机器语言。所谓不同的机器,是指具有不同的 CPU(中央处理器)的计算机。由于机器指令与 CPU 紧密相关,所以不同种类的 CPU 所对应的机器指令也就不同,而且它们的指令系统往往相差很大。但对同一系列的 CPU 来说,为了满足各型号之间具有良好的兼容性,因此设计者一般要做到:新一代 CPU 的指令系统必须兼容先前开发的同系列 CPU 的指令系统。只有这样,先前开发出来的各类程序在新一代 CPU 上才能正常运行。



图 1-1 使用不同机器语言的计算机

机器指令通常由操作码和操作数两部分组成。操作码指出该指令所要完成的操作,即指令的功能;操作数指出参与运算的对象,以及运算结果所存放的位置等。

机器指令由物理部件的若干位二进制 0、1 状态体现,因此机器指令的最直接的记忆形式就是表示这一指令状态的二进制数,这个二进制数就称做该机器指令的二进制代码指令。由于二进制数和八进制数以及十六进制数的特殊关系,因此该二进制代码指令对应的八进制数和十六进制数分别称做该机器指令的八进制或十六进制代码指令。

例如, B233H 就是一条 80x86 的十六进制机器代码指令,这条指令的功能是将 33H 这个十六进制数送 DL 寄存器。

该机器指令的二进制代码指令是 1011001000110011B。

该机器指令的八进制代码指令是 131063Q。

显然二进制代码指令与其状态对应直观,但记忆困难;而十六进制代码指令则方便记忆。因此 80x86 的代码指令通常都用十六进制表示。

用机器指令所写的程序称做机器指令程序,机器指令程序也指用代码指令所写的程序。机器指令程序遵循冯·诺依曼原理:用二进制表示数;程序与数据都存储在同样的存储器中;程序指令一般顺序存取、顺序执行。

计算机出现初期,计算机只有一种语言,那就是机器语言。程序员只能用代码指令编写程序,即机器语言程序。用机器语言编写程序是相当艰苦的工作,必须由经过严格训练的专业技术人员承担,这是早期计算机所以不能够广泛应用的最大障碍之一。同时用机器语言编写的代码指令程序,由于都是由代码组成,因此具有不易读、出错率高、难以维护等诸多缺点,当然也就不能直观地反映用计算机解决问题的基本思路。由于用机器语言编写程序具有以上诸多不便,因此现在几乎已经没有程序员再用代码指令直接编写程序了。

一方面,机器语言十分重要,必须了解;一方面,机器代码指令程序使用又十分困难,因此汇编语言就成为唯一能够替代机器语言的计算机科技人员必须掌握的一种计算机语言。

2. 汇编语言

简单地说,用助记符表示的机器指令称做汇编指令,用汇编指令编写的程序称做汇编指令程序。

例 1-1 汇编程序和对应的机器指令程序。

本例给出的是一段由 4 条二字节汇编指令构成的 80x86 的汇编语言程序,它可以在 debug 调试环境下汇编运行,运行结果是在显示器上输出一个 ASCII 码字符‘3’。

```
MOV DL, 33
MOV AH, 2
INT 21
INT 20
```

注意:这里‘3’不能在 debug 下直接输入,必须用 33H 表示,H 省略不写。

其对应的十六进制机器代码指令程序如下:

```
B233H
B402H
CD21H
CD20H
```

虽然用机器语言编写程序有很高的要求和许多不便,但编写出来的程序执行效率高,CPU 严格按照程序员的要求去做,没有多余的额外操作。所以,在保留“程序执行效率高”的前提下,人们就开始着手研究一种能大大改善程序可读性的编程方法。

为了改善机器指令的可读性,人们选用了一些能反映机器指令功能的单词或词组来代表该机器指令,而不再关心机器指令的具体二进制编码,不妨称这些单词或词组为助记符。与此同时,也把 CPU 内部的各种资源符号化,使用该符号名也等于引用了该具体的物理资源。

例如,汇编指令

```
MOV AL, BL
```

对应的二进制指令为 1101100010001000B,十六进制代码指令则为 D888H,记忆相当困难。

而汇编指令中 MOV 源于 Move,表示传送指令;AL、BL 分别表示 80x86 的 AL 和 BL 寄存器。指令功能是将 BL 的内容传送到 AL 寄存器。如此一来,令人难懂的二进制机器代码指令或十六进制机器代码指令就可以用通俗易懂的、具有一定含义的符号指令来表示了,这就是汇编指令的原意。称这些具有一定含义的符号,例如上述指令中的 MOV,为助记符,用助记符、符号地址等组成的符号指令称为汇编格式指令或汇编指令。

汇编语言是汇编指令集、伪指令集及其使用规则的统称。能够一对一地翻译成机器指令的这种用助记符表示的机器指令称做汇编指令。伪指令是相对于汇编指令而言的,它是出现在汇编语言程序中所需要的一些辅助性的说明,它不对应具体的机器指令,是程序设计与汇编过程中的翻译程序(即汇编)、连接程序以及执行时如何装入的一种约定。有关内容在以后的各章节中会有详细叙述。

用汇编语言编写的程序称为汇编语言程序或汇编语言源程序,在本书中或特定的环境

下,也可简称为源程序。汇编语言程序要比用机器指令编写的程序容易理解和维护。

汇编语言源程序是一个文本文件,汇编语言源程序必须经过翻译才能够变成可以执行的机器语言程序,这个翻译过程称做汇编。汇编的核心过程是将汇编指令逐条翻译成机器指令,这正是汇编语言中“汇编”一词的含义。图 1-2 给出了一个简单汇编语言源程序与其机器指令程序之间的对应关系。

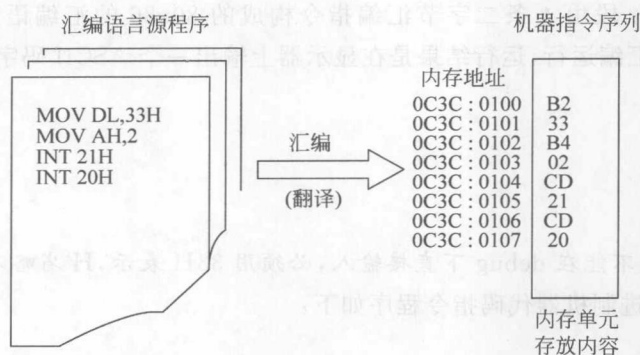


图 1-2 汇编程序翻译过程示意图

尽管不同的 CPU 有不同的汇编语言,但任何一种汇编语言都有其共同规律。因此学会一种机器的汇编语言,再学习其他汇编语言就相对容易多了。

1.1.2 非面向机器的计算机语言

1. 编译型计算机语言

非面向机器的语言按照翻译程序的类型分为编译型语言 and 解释型语言,多数计算机高级语言是编译型语言,例如算法语言 ALGOL、公式翻译语言 FORTRAN、商用语言 COBOL 等。各种 C 语言,例如 Microsoft C、Turbo C、VC 等,都是编译型语言。

这些计算机语言,无论是结构化的还是面向对象的,均采用类似于英语的语法结构,其源程序与汇编语言一样也是文本文件。用这些语言所写的程序,它所描述的算法和问题要比汇编语言直观明了。但其翻译成机器语言的过程远远比汇编的过程复杂。通常要经过预处理和多次扫描才能够变成机器语言,将这个过程称做编译。编译后的浮动二进制文件通常也要经过连接,才能够生成可执行文件。

以 Turbo C 为例,C 语言源程序文件扩展名为 .c,C++ 语言源程序文件扩展名为 .cpp,经过编译后的浮动二进制文件为 .obj,连接以后的执行文件扩展名为 .exe。这里的可执行文件是一个机器指令程序。一旦程序调试正确,何时执行只需要执行该文件即可,这和汇编以后的执行文件是同样的。不过其效率往往要低于用汇编语言直接写成的程序。

不同 CPU 的计算机如果要执行 C 语言程序,只要运用不同的 C 语言编译程序就可以了。图 1-3 表明了 C 语言的可移植性。其他非面向机器的语言与 C 语言具有同样的规律,具有可移植性,而低级语言则不具有可移植性。