可下载教学资料
http://www.tup.tsinghua.edu.cn

21世纪普通高校计算机公共课程规划教材

汇编语言程序设计

宋人杰 主编
牛 斗 王润辉 周欣欣 李红彪 编著

清华大学出版社

21世纪普通高校计算机公共课程规划教材

汇编语言程序设计

宋人杰 主编
牛 斗 王润辉 周欣欣 李红彪 编著

清华大学出版社
北京

内 容 简 介

“汇编语言程序设计”是高校计算机专业的主干课程之一。本书以 8086/8088 指令为主，以实模式下的 80x86 指令为辅，系统地介绍了汇编语言的基础理论知识和程序设计方法。主要内容包括：汇编语言程序设计基础知识、8086 指令寻址方式及指令系统、常用伪指令、程序设计方法、高级汇编技术、80x86 指令系统、汇编语言与 C 语言混合设计的方法。本书各章节内容重点突出、结构清晰、简洁易懂。

在本书的实验调试软件一章中，介绍了两种调试软件：基于 MASM 5.0 的 DEBUG 和基于 MASM 6.11 的 PWB、CodeView，为读者进行汇编语言程序设计提供了方便。

本书可作为本科、高职院校计算机及相关专业的教材，也可供科研及软件开发人员自学参考。

图书在版编目（CIP）数据

汇编语言程序设计/宋人杰主编；牛斗等编著. —北京：清华大学出版社，2008.6
（21 世纪普通高校计算机公共课程规划教材）
ISBN 978-7-302-17458-5

Ⅰ. 汇… Ⅱ. ①宋… ②牛… 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（2008）第 056589 号

责任编辑：梁　颖
责任校对：梁　毅
责任印制：孟凡玉
出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**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
印 刷 者：北京市昌平环球印刷厂
装 订 者：北京国马印刷厂
经　　销：全国新华书店
开　　本：185×260　**印　张**：14　**字　数**：340 千字
版　　次：2008 年 6 月第 1 版　　**印　　次**：2008 年 6 月第1 次印刷
印　　数：1～4000
定　　价：21.00 元

出版说明

随着我国改革开放的进一步深化，高等教育也得到了快速发展，各地高校紧密结合地方经济建设发展需要，科学运用市场调节机制，加大了使用信息科学等现代科学技术提升、改造传统学科专业的投入力度，通过教育改革合理调整和配置了教育资源，优化了传统学科专业，积极为地方经济建设输送人才，为我国经济社会的快速、健康和可持续发展以及高等教育自身的改革发展做出了巨大贡献。但是，高等教育质量还需要进一步提高以适应经济社会发展的需要，不少高校的专业设置和结构不尽合理，教师队伍整体素质亟待提高，人才培养模式、教学内容和方法需要进一步转变，学生的实践能力和创新精神亟待加强。

教育部一直十分重视高等教育质量工作。2007 年 1 月，教育部下发了《关于实施高等学校本科教学质量与教学改革工程的意见》，计划实施“高等学校本科教学质量与教学改革工程（简称‘质量工程’）”，通过专业结构调整、课程教材建设、实践教学改革、教学团队建设等多项内容，进一步深化高等学校教学改革，提高人才培养的能力和水平，更好地满足经济社会发展对高素质人才的需要。在贯彻和落实教育部“质量工程”的过程中，各地高校发挥师资力量强、办学经验丰富、教学资源充裕等优势，对其特色专业及特色课程（群）加以规划、整理和总结，更新教学内容、改革课程体系，建设了一大批内容新、体系新、方法新、手段新的特色课程。在此基础上，经教育部相关教学指导委员会专家的指导和建议，清华大学出版社在多个领域精选各高校的特色课程，分别规划出版系列教材，以配合“质量工程”的实施，满足各高校教学质量和教学改革的需要。

本系列教材立足于计算机公共课程领域，以公共基础课为主、专业基础课为辅，横向满足高校多层次教学的需要。在规划过程中体现了如下一些基本原则和特点。

（1）面向多层次、多学科专业，强调计算机在各专业中的应用。教材内容坚持基本理论适度，反映各层次对基本理论和原理的需求，同时加强实践和应用环节。

（2）反映教学需要，促进教学发展。教材要适应多样化的教学需要，正确把握教学内容和课程体系的改革方向，在选择教材内容和编写体系时注意体现素质教育、创新能力与实践能力的培养，为学生的知识、能力、素质协调发展创造条件。

（3）实施精品战略，突出重点，保证质量。本规划教材把重点放在公共基础课和专业基础课的教材建设上；特别注意选择并安排一部分原来基础比较好的优秀教材或讲义修订再版，逐步形成精品教材；提倡并鼓励编写体现教学质量和教学改革成果的教材。

（4）主张一纲多本，合理配套。基础课和专业基础课教材配套，同一门课程有针对不同层次、面向不同专业的多本具有各自内容特点的教材。处理好教材统一性与多样化，基本教材与辅助教材、教学参考书，文字教材与软件教材的关系，实现教材系列资源配套。

（5）依靠专家，择优选用。在制定教材规划时要依靠各课程专家在调查研究本课程教

材建设现状的基础上提出规划选题。在落实主编人选时，要引入竞争机制，通过申报、评审确定主题。书稿完成后要认真实行审稿程序，确保出书质量。

繁荣教材出版事业，提高教材质量的关键是教师。建立一支高水平教材编写梯队才能保证教材的编写质量和建设力度，希望有志于教材建设的教师能够加入到我们的编写队伍中来。

21 世纪普通高校计算机公共课程规划教材编委会
联系人：梁颖 liangying@tup.tsinghua.edu.cn

前言

汇编语言程序设计技术是计算机专业学生必须掌握的基本技能之一。使用汇编语言编写程序，用户可以直接访问计算机系统内部各资源，具有实时性强、占用存储资源少、执行速度快、代码效率高等优点。另外，通过汇编语言的学习，学生可以更好地理解计算机系统的组成及工作原理。因此“汇编语言程序设计”这门课程始终是高校计算机专业及相关学科的经典课程之一。

面对计算机技术的迅猛发展，传统的基于 DOS 平台的汇编语言程序设计已经不能满足需要。因此，本书从便于教学出发，在内容编排上既兼顾了以传统的 Intel 8086/8088 为代表的 16 位汇编语言程序设计，同时又以较大的篇幅介绍了 80x86 指令系统和相关的程序设计方法。

全书共分 10 章。第 1 章介绍了学习 80x86 汇编语言程序设计所需要的基础知识；第 2 章介绍了伪指令及汇编语言程序设计结构；第 3 章介绍了 8086 的寻址方式及指令系统；第 4 章系统地介绍了顺序、分支及循环程序设计的基本方法和技巧；第 5 章重点介绍了子程序和宏汇编程序设计的基本方法；第 6 章介绍了 32 位指令的寻址方式、指令系统及相关的程序设计方法；第 7 章介绍了汇编程序应用实例；第 8 章介绍了输入输出程序设计和中断程序设计的概念及方法，以及 DOS 和 BIOS 中断调用的调用方法；第 9 章介绍了 C 语言与汇编语言混合编程方法；第 10 章介绍了 Debug、PWB、Code View 等调试工具的使用方法。

本书由宋人杰教授负责组织编写，其中，第 1、4、5 章由宋人杰编写；第 6、7、10 章由牛斗编写；第 8、9 章由王润辉编写；第 2、3 章由周欣欣编写；其他辅助工作由李红彪完成。

由于编者水平有限，书中如有错误和不妥之处，敬请广大读者批评指正。

作　者

2008 年 3 月

目 录

第 1 章 汇编语言基础知识 …… 1

1.1 微型计算机概述 …… 1
1.2 Intel 公司微处理器简介 …… 2
1.3 计算机语言及汇编语言特点 …… 3
1.3.1 计算机语言概述 …… 3
1.3.2 汇编语言的特点 …… 5
1.4 程序可见寄存器组 …… 5
1.5 存储器 …… 9
1.5.1 基本概念 …… 9
1.5.2 实模式存储器寻址 …… 10
1.6 外部设备 …… 11
习题 …… 12

第 2 章 汇编语言源程序格式 …… 13

2.1 汇编语言语句格式 …… 13
2.1.1 汇编语言语句类型 …… 13
2.1.2 汇编语言指令格式 …… 13
2.2 伪指令 …… 20
2.2.1 处理器选择伪指令 …… 21
2.2.2 数据定义伪指令 …… 21
2.2.3 模块命名和标题伪指令 …… 24
2.2.4 程序结束伪指令 …… 24
2.2.5 完整段定义伪指令 …… 25
2.2.6 简化段定义伪指令 …… 28
2.2.7 表达式赋值伪指令 …… 29
2.2.8 定位伪指令 …… 30
2.2.9 标号定义伪指令 …… 32
2.3 汇编语言源程序基本框架 …… 32
2.3.1 完整段定义框架 …… 32
2.3.2 简化段定义框架 …… 34
习题 …… 35

第 3 章　8086/8088 寻址方式及指令系统 …… 36

3.1　8086/8088 寻址方式 …… 36
3.1.1　数据寻址方式 …… 36
3.1.2　程序转移寻址方式 …… 42
3.2　8086/8088 指令系统 …… 44
3.2.1　数据传送指令 …… 44
3.2.2　算术运算指令 …… 49
3.2.3　逻辑操作指令 …… 55
3.2.4　串处理指令 …… 59
3.2.5　控制转移指令 …… 65
3.2.6　处理器控制指令 …… 72
习题 …… 74

第 4 章　顺序、分支与循环程序设计 …… 78

4.1　顺序程序设计 …… 78
4.2　分支程序设计 …… 80
4.2.1　分支结构 …… 80
4.2.2　用分支指令实现分支结构程序 …… 80
4.3　循环程序设计 …… 83
4.3.1　循环结构 …… 83
4.3.2　单循环程序设计 …… 85
4.3.3　多重循环程序设计 …… 89
习题 …… 92

第 5 章　子程序及宏指令设计 …… 93

5.1　子程序设计方法 …… 93
5.1.1　子程序定义 …… 93
5.1.2　寄存器内容的保存及恢复 …… 94
5.1.3　子程序的调用及返回 …… 95
5.1.4　子程序的参数传递 …… 95
5.1.5　子程序嵌套 …… 101
5.2　模块化程序设计 …… 102
5.2.1　模块划分 …… 102
5.2.2　源程序文件包含的伪指令 …… 102
5.2.3　模块间的连接 …… 103
5.3　宏汇编 …… 104
5.3.1　宏定义、宏调用和宏展开 …… 104
5.3.2　宏定义和宏调用中的参数 …… 106

5.3.3 宏指令的嵌套 …… 108
5.3.4 宏汇编中的伪指令 …… 110
5.3.5 重复汇编 …… 112
5.3.6 条件汇编 …… 113
习题 …… 114

第 6 章 32 位指令系统及程序设计 …… 116

6.1 32 位微处理器工作模式 …… 116
6.2 32 位指令的运行环境 …… 117
6.2.1 寄存器组 …… 117
6.2.2 80386 保护模式下的存储管理 …… 119
6.3 32 位 80x86 CPU 的寻址方式 …… 119
6.4 32 位微处理器指令 …… 120
6.4.1 使用 32 位 80x86 指令的注意事项 …… 120
6.4.2 80386 新增指令 …… 121
6.4.3 80486 新增指令 …… 123
6.4.4 Pentium 新增指令 …… 124
6.4.5 Pentium Pro 新增指令 …… 125
6.4.6 MMX 指令 …… 125
6.4.7 SIMD 指令 …… 130
6.5 程序设计举例 …… 132
6.5.1 基于 32 位指令的实模式程序设计 …… 132
6.5.2 基于 MMX 指令的实模式程序设计 …… 133
6.5.3 保护模式下的程序设计 …… 135
习题 …… 138

第 7 章 综合程序设计 …… 139

7.1 加密程序设计举例 …… 139
7.2 反跟踪程序设计举例 …… 141
习题 …… 145

第 8 章 输入输出与中断控制 …… 146

8.1 输入输出接口概述 …… 146
8.1.1 输入输出接口 …… 146
8.1.2 主机与外设之间交换数据的方式 …… 147
8.2 程序控制方式下的输入输出程序设计 …… 148
8.2.1 无条件传送方式 …… 148
8.2.2 程序查询方式 …… 152
8.3 中断传送方式 …… 154

8.3.1 中断系统 …… 155
8.3.2 中断优先级与中断嵌套 …… 158
8.3.3 中断处理程序 …… 158
8.4 DOS 与 BIOS 中断 …… 161
8.4.1 DOS 系统功能调用 …… 161
8.4.2 BIOS 功能调用 …… 164
习题 …… 172

第 9 章 C 语言与汇编语言混合编程 …… 174

9.1 嵌入式汇编 …… 174
9.1.1 嵌入式汇编程序中汇编指令格式 …… 174
9.1.2 嵌入式汇编程序设计 …… 175
9.1.3 编译连接的方法 …… 179
9.2 C 语言调用汇编模块 …… 179
9.2.1 C 语言调用汇编模块编程规则 …… 180
9.2.2 C 语言调用汇编模块的编译连接方法 …… 183
9.3 汇编语言引用 C 语言函数 …… 184
习题 …… 185

第 10 章 汇编语言程序实验工具软件介绍 …… 187

10.1 汇编语言实验上机步骤 …… 187
10.2 常用调试程序 Debug …… 189
10.2.1 Debug 的主要特点 …… 189
10.2.2 Debug 的启动 …… 189
10.2.3 Debug 的命令 …… 189
10.2.4 Debug 中的命令介绍 …… 190
10.2.5 Debug 程序的应用举例 …… 197
10.3 集成开发环境 PWB …… 198
10.3.1 PWB 的安装 …… 198
10.3.2 PWB 的运行和退出 …… 199
10.3.3 PWB 主菜单 …… 199
10.3.4 PWB 开发环境的设置 …… 200
10.3.5 PWB 的应用 …… 200
10.4 源代码级调试工具软件 CodeView …… 200
附录 A DOS 功能调用（INT 21H）一览表 …… 202
附录 B BIOS 中断调用表（INT N） …… 207

参考文献 …… 211

第 1 章 汇编语言基础知识

汇编语言是直接在硬件之上工作的编程语言，首先要了解硬件系统的结构，才能有效地应用汇编语言对其编程，因此，本章对硬件系统结构的问题进行部分探讨，首先介绍了计算机的基本结构、Intel 公司微处理器的发展、计算机的语言以及汇编语言的特点，在此基础上重点介绍寄存器、内存组织等汇编语言所涉及到的基本知识。

1.1 微型计算机概述

微型计算机由中央处理器（Central Processing Unit，CPU）、存储器、输入输出接口电路和总线构成。CPU 如同微型计算机的心脏，它的性能决定了整个微型计算机的各项关键指标。存储器包括随机存储器（Random Access Memory，RAM）和只读存储器（Read Only Memory，ROM）。输入输出接口电路用来连接外部设备和微型计算机。总线为 CPU 和其他部件之间提供数据、地址和控制信息的传输通道。如图 1.1 所示为微型计算机的基本结构。

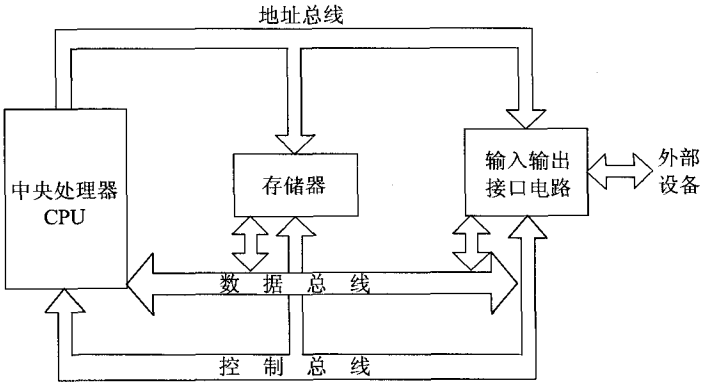

图 1.1 微型计算机基本结构

特别要提到的是微型计算机的总线结构，它使系统中各功能部件之间的相互关系变为各个部件面向总线的单一关系。一个部件只要符合总线结构标准，就可以连接到采用这种总线结构的系统中，使系统功能得到扩展。

数据总线用来在 CPU 与内存或其他部件之间进行数据传送。它是双向的，数据总线的位宽决定了 CPU 和外界的数据传送速度，8 位数据总线一次可传送一个 8 位二进制数据（即一个字节），16 位数据总线一次可传送两个字节。在微型计算机中，数据的含义是广义的，数据总线上传送的不一定是真正的数据，而可能是指令代码、状态量或控制量。

地址总线专门用来传送地址信息，它是单向的，地址总线的位数决定了 CPU 可以直接寻址的内存范围。如 CPU 的地址总线的宽度为 N，则 CPU 最多可以寻找 2^N 个内存单元。

控制总线用来传输控制信号，其中包括 CPU 送往存储器和输入输出接口电路的控制信号，如读信号、写信号和中断响应信号等；也包括其他部件送到 CPU 的信号，如时钟信号、中断请求信号和准备就绪信号等。

1.2 Intel 公司微处理器简介

自 20 世纪 70 年代开始出现微型计算机以来，CPU 经历了飞速的发展。1971 年，Intel 设计成功了第一片 4 位微处理器 Intel 4004；随之又设计生产了 8 位微处理器 8008；1973 年推出了 8080；1974 年基于 8080 的个人计算机（Personal Computer，PC）问世，Microsoft 公司的创始人 Bill Gates 为 PC 开发了 BASIC 语言解释程序；1977 年 Intel 推出了 8085。自此之后，Intel 又陆续推出了 8086、80386、Pentium 等 80x86 系列微处理器。各种微处理器的主要区别在于处理速度、寄存器位数、数据总线宽度和地址总线宽度。下面简要介绍不同时期 Intel 公司制造的几种主要型号的微处理器，这些微处理器都曾经或正在广为流行。

1．80x86 系列微处理器

1）8088 微处理器

具有多个 16 位的寄存器、8 位数据总线和 20 位地址总线，可以寻址 1MB 的内存。虽然这些寄存器一次可以处理 2 个字节，但数据总线一次只能传送 1 个字节。该处理器只能工作在实模式。

2）8086 微处理器

指令系统与 8088 完全相同，具有多个 16 位的寄存器、16 位数据总线和 20 位地址总线，可以寻址 1MB 的内存，一次可以传送 2 个字节。该处理器只能工作在实模式。

3）80286 微处理器

比 8086 运行更快，具有多个 16 位的寄存器、16 位数据总线和 24 位地址总线，可以寻址 16MB 内存。它既可以工作在实模式，也可以工作在保护模式。

4）80386 微处理器

具有多个 32 位的寄存器、32 位数据总线和 32 位地址总线，可以寻址 4GB 内存。它提供了较高的时钟速度，增加了存储器管理和相应的硬件电路，减少了软件开销，提高了效率。它既可以工作在实模式，也可以工作在保护模式。

5）80486 微处理器

具有多个 32 位的寄存器、32 位数据总线和 32 位地址总线。它比 80386 增加了数字协处理器和 8KB 的高速缓存，提高了处理速度。它既可以工作在实模式，也可以工作在保护模式。

6）Pentium（奔腾）

具有多个 32 位的寄存器、64 位数据总线和 36 位地址总线。因为它采用了超标量体系结构，所以每个时钟周期允许同时执行两条指令，处理速度得到了进一步提高，性能比 80486 优越得多。它既可以工作在实模式，也可以工作在保护模式。

以上介绍了 Intel80x86 系列的一些主要微处理器，表 1.1 给出了该系列部分微处理器的数据总线和地址总线宽度。实际上 80x86 系列的功能还在不断改进和增强，它们的速度将会更快，性能将会更优越。但无论怎样变化，它们总会被设计成是完全向下兼容的，就像在 8086 上设计和运行的软件可以不加任何改变地在 Pentium 4 机上运行一样。对于汇编语言编程人员来讲，掌握 16 位计算机的编程十分重要，它是学习高档计算机及保护模式编程的基础，也是掌握实模式程序设计的唯一方法。

2．CPU 的主要性能指标

1）机器字长

机器字长和 CPU 内部寄存器、运算器、内部数据总线的位宽相一致。如 8086CPU，它的内部寄存器是 16 位的、运算器能完成两个 16 位二进制数的并行运算、数据总线的位宽为 16 位，则它的机器字长为 16 位，也称其为 16 位计算机。通常，机器字长越长，计算机的运算能力越强，其运算精度也越高。

2）速度

CPU 的速度是指单位时间内能够执行指令的条数。速度的计算单位不一，若以单字长定点指令的平均执行时间计算，用每秒百万条指令（Million Istructions Per Second，MIPS）作为单位；若以单字长浮点指令的平均执行时间计算，则用每秒百万条浮点运算指令（Million Floating-point Operations Per Second，MFLOPS）表示。现在，采用计算机中各种指令的平均执行时间和相应的指令运行权重的加权平均法求出等效速度作为计算机运算速度。

3）主频

主频又称为主时钟频率，是指 CPU 在单位时间内产生的时钟脉冲数，以 MHz/s（兆赫兹每秒）为单位。由于计算机中的一切操作都是在时钟控制下完成的，因此，对于机器结构相同或相近的计算机，CPU 的时钟频率越高，运算速度越快。

表 1.1　Intel 80x86 系列微处理器总线宽度

CPU	数据总线宽度	地址总线宽度	CPU	数据总线宽度	地址总线宽度
8086	16	20	Pentium	64	36
8088	8	20	Pentium Ⅱ	64	36
80286	16	24	Pentium Ⅲ	64	36
80386SX	16	24	Pentium 4	64	36
80386DX	32	32	Itanium	64	44
80486	32	32			

1.3　计算机语言及汇编语言特点

1.3.1　计算机语言概述

计算机语言的发展经历了由机器语言、汇编语言到高级语言这样一个由低级到高级的发展过程。

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。由于计算机硬件本身只能识别二进制代码，在计算机发展的初期，人们使用二进制代码构成机器指令来编写程序，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，只有目标程序才能被 CPU 直接执行。指令用于指出计算机所进行的操作和操作对象的代码，一条指令通常由操作码和操作数两部分组成。其中，操作码指出计算机所进行的具体操作，如加法、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、两个或三个，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，既可以存放在寄存器中，也可以存放在存储器中，甚至直接存放在指令中，通常要用寻址方式来说明。

一台计算机全部指令的集合构成该计算机的指令系统。指令系统是计算机基本功能的体现，不同的机器指令对应的二进制代码序列各不相同。机器语言是面向机器的，不同机器之间的语言是不通用的，这也是机器语言是“低级”语言的含义所在。用二进制代码编写程序相当麻烦，写出的程序也难以阅读和调试。

2. 汇编语言

早期的程序员们很快就发现了使用机器语言带来的麻烦，它是如此难于辨别和记忆，给整个产业的发展带来了障碍，于是产生了汇编语言。汇编语言是一种采用指令助记符、符号地址、标号等符号书写程序的语言，它便于人们书写、阅读和检查。汇编语言指令与计算机指令基本上是一一对应的，汇编语言与计算机有着密不可分的关系，处理器不同，汇编语言就不同，因此它是一种低级语言，同时它也是唯一能够充分利用计算机硬件特性并直接控制硬件设备的语言。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。

用汇编语言编写的程序称为汇编源程序（或称汇编语言程序），计算机不能直接识别，必须将其翻译成由计算机指令组成的程序后，CPU 才能执行，这一过程称为“汇编”。用于将汇编源程序翻译成计算机语言的程序称为汇编程序，这种由源程序经过计算机翻译转换成的计算机语言程序也称为目标程序。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。连接程序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，构成较大的可执行程序。它们之间的关系如图 1.2 所示。

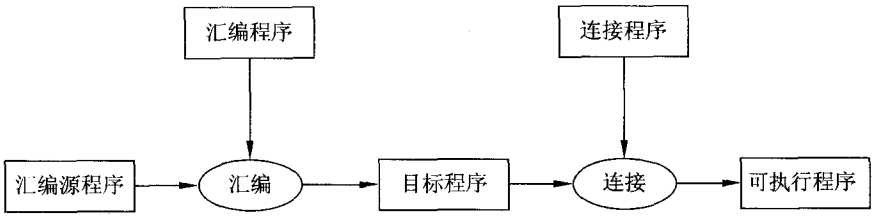

图 1.2　汇编程序与目标程序、可执行程序之间的关系

3. 高级语言

高级语言是一种与具体的计算机硬件无关，独立于计算机类型的通用语言，比较接近

人类自然语言的语法，用高级语言编程不必了解和熟悉计算机的指令系统，更容易掌握和使用。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，这些语言面向求解问题的过程，不依赖具体计算机。高级语言也要翻译成机器语言才能在计算机上执行。其翻译有两种方式，一种是把高级语言程序翻译成机器语言程序，然后经过连接程序连接成可执行文件，再在计算机上执行，这种翻译方式称为编译方式，大多数高级语言如 PASCAL 语言、C 语言等都是采用这种方式；另一种是直接把高级语言程序在计算机上运行，一边解释一边执行，这种翻译方式称为解释方式，如 BASIC 语言就采用这种方式。

高级语言源程序是在未考虑计算机结构特点情况下编写的，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，占用较大存储空间，执行时间较长。

1.3.2 汇编语言的特点

汇编语言使用助记符和符号地址，所以它要比机器语言易于掌握，与高级语言相比较，汇编语言有以下特点。

1）汇编语言与计算机关系密切

汇编语言中的指令是机器指令的符号表示，与机器指令是一一对应的，因此它与计算机有着密切的关系，不同类型的 CPU 有不同的汇编语言，也就有各种不同的汇编程序。汇编语言源程序与高级语言源程序相比，其通用性和可移植性要差得多。

2）汇编语言程序效率高

由于构成汇编语言主体的指令是用机器指令的符号表示的，每一条指令都对应一条机器指令，且汇编语言程序能直接利用计算机硬件系统的许多特性，如它允许程序员利用寄存器、标志位等编程。用汇编语言编写的源程序在编译后得到的目标程序效率高，主要体现在空间效率和时间效率上，即目标程序短、运行速度快这两个方面，在采用相同算法的前提下，任何高级语言程序在这两个方面的效率与汇编语言相比都望尘莫及。

3）特殊的使用场合

汇编语言可以实现高级语言难以胜任甚至不能完成的任务。汇编语言具有直接和简捷的特点，用它编制程序能精确地描述算法，充分发挥计算机硬件的功能。在过程控制、多媒体接口、设备通信、内存管理、硬件控制等方面的程序设计中，用汇编语言直接方便，执行速度快，效率高。

汇编语言提供了一些模块间相互连接的方法，一个大的任务可以分解成若干模块，将其中执行频率高的模块用汇编语言编写，可以大大提高大型软件的性能。

1.4 程序可见寄存器组

80386（含 80386）以上型号的 CPU 能够处理 32 位数据，其寄存器长度是 32 位的，但为了与早期的 8086 等 16 位机 CPU 保持良好的兼容性，80386 以上型号的 CPU 中程序可见寄存器组包括多个 8 位、16 位和 32 位寄存器，如图 1.3 所示。

1．通用寄存器

8086~80286 CPU 各有 8 个 16 位通用寄存器 AX、BX、CX、DX、SP、BP、SI、DI。对于 4 个 16 位数据寄存器 AX、BX、CX、DX，其每个又可以作为 2 个独立的 8 位寄存器

使用，它们被分别命名为 AH、AL、BH、BL、CH、CL、DH、DL。80386 以上型号的 CPU 各有 8 个 32 位通用寄存器，它们是相应 16 位寄存器的扩展，被分别命名为 EAX、EBX、ECX、EDX、ESP、EBP、ESI、EDI。在程序中每个 8 位、16 位、32 位寄存器都可以独立使用。

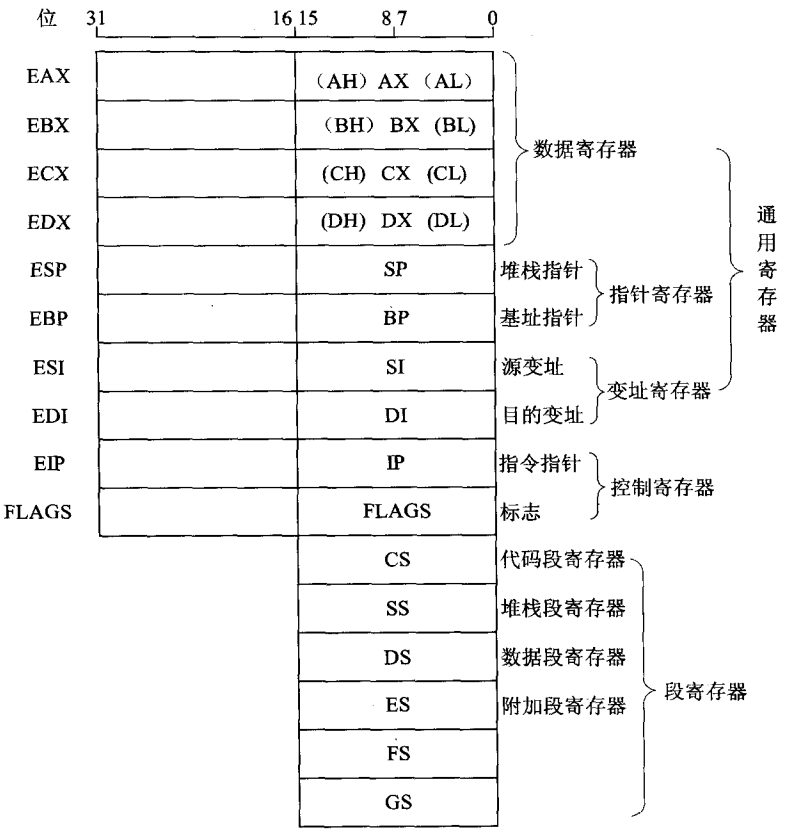

图 1.3　8086～Pentium CPU 程序可见寄存器组

SP、ESP 叫做堆栈指针寄存器，其中存放当前堆栈段栈顶的偏移量，它们总是与 SS 堆栈段寄存器配合存取堆栈中的数据。在实模式方式下使用 SP，在 80386 以上的保护模式下使用 ESP。

除 SP、ESP 堆栈指针不能随意修改、需要慎用外，其他通用寄存器都可以直接在指令中使用，用以存放操作数，这是它们的通用之处。在后边讨论指令系统时，可以看到某些通用寄存器在具体的指令中还有其他用途，例如 EAX、AX、AL（通常分别被称为 32 位、16 位、8 位累加器），它们在乘除法、十进制运算、输入输出指令中有专门用途。另外有些通用寄存器也可以存放地址用以间接寻址内存单元，例如在实模式中 BX、BP、SI、DI 可以作为间接寻址的寄存器，用以寻址 64KB 以内的内存单元。在保护模式中 EAX、EBX、ECX、EDX、ESP、EBP、ESI、EDI 可以作为间接寻址的寄存器，用以寻址 4GB 以内的内存单元，详细内容见 3.1 节和 6.3 节。

2．段寄存器

在 IBM PC 机中存储器采用分段管理的方法，因此一个物理地址需要用段基地址和偏移量表示。一个程序可以由多个段组成，但对于 8086～80286 CPU，由于只有 4 个段寄存器，所以在某一时刻正在运行的程序只可以访问 4 个当前段，而对于 80386 及其以上的计算机，由于有 6 个段寄存器，则可以访问 6 个当前段。在实模式下段寄存器存放当前正在运行程序的段基地址的高 16 位，在保护模式下存放当前正在运行程序的段选择子，段选择子用以选择描述符表中的一个描述符，描述符描述段的基地址、长度和访问权限等，显然在保护模式下段寄存器仍然是选择一个内存段，只是不像实模式那样直接存放段基址罢了。

代码段寄存器 CS 指定当前代码段，代码段中存放当前正在运行的程序段。堆栈段寄存器 SS 指定当前堆栈段，堆栈段是在内存开辟的一块特殊区域，其中的数据访问按照后进先出（Last in First out，LIFO）的原则进行，允许插入和删除的一端叫做栈顶。IBM PC 机中 SP（或 ESP）指向栈顶，SS 指向堆栈段基地址。数据段寄存器 DS 指定当前运行程序所使用的数据段。附加数据段寄存器 ES 指定当前运行程序所使用的附加数据段。段寄存器 FS 和 GS 只对 80386 以上 CPU 有效，它们没有对应的中文名称，用于指定当前运行程序的另外两个存放数据的存储段。虽然 DS、ES、FS、GS（甚至于 CS、SS）所指定的段中都可以存放数据，但 DS 是主数据段寄存器，在默认情况下使用 DS 所指向段的数据。若要引用其他段中的数据，需要显式地说明。

3．控制寄存器

控制寄存器包括指令指针寄存器和标志寄存器。在程序中不能直接引用控制寄存器名。

1）IP、EIP

IP、EIP 叫做指令指针寄存器，它总是与 CS 段寄存器配合指出下一条要执行指令的地址，其中存放偏移量部分。在实模式方式下使用 IP，在 80386 以上的保护模式下使用 EIP。

2）标志寄存器（FLAGS）

标志寄存器也被称为状态寄存器，由运算结果特征标志和控制标志组成。8086～80286 CPU 为 16 位，80386 以上为 32 位。如图 1.4 所示，可以看出它们完全向下兼容。空白位为将来保留，暂未定义。

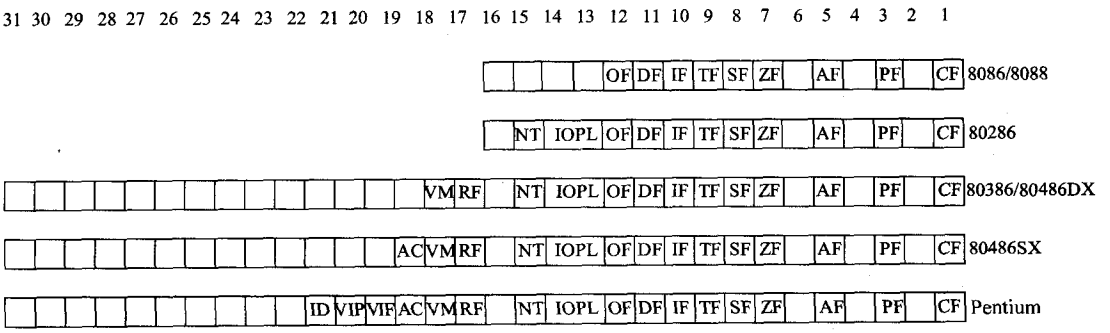

图 1.4　标志寄存器

（1）运算结果特征标志：用于记录程序中运行结果的特征，8086～Pentium CPU 的标