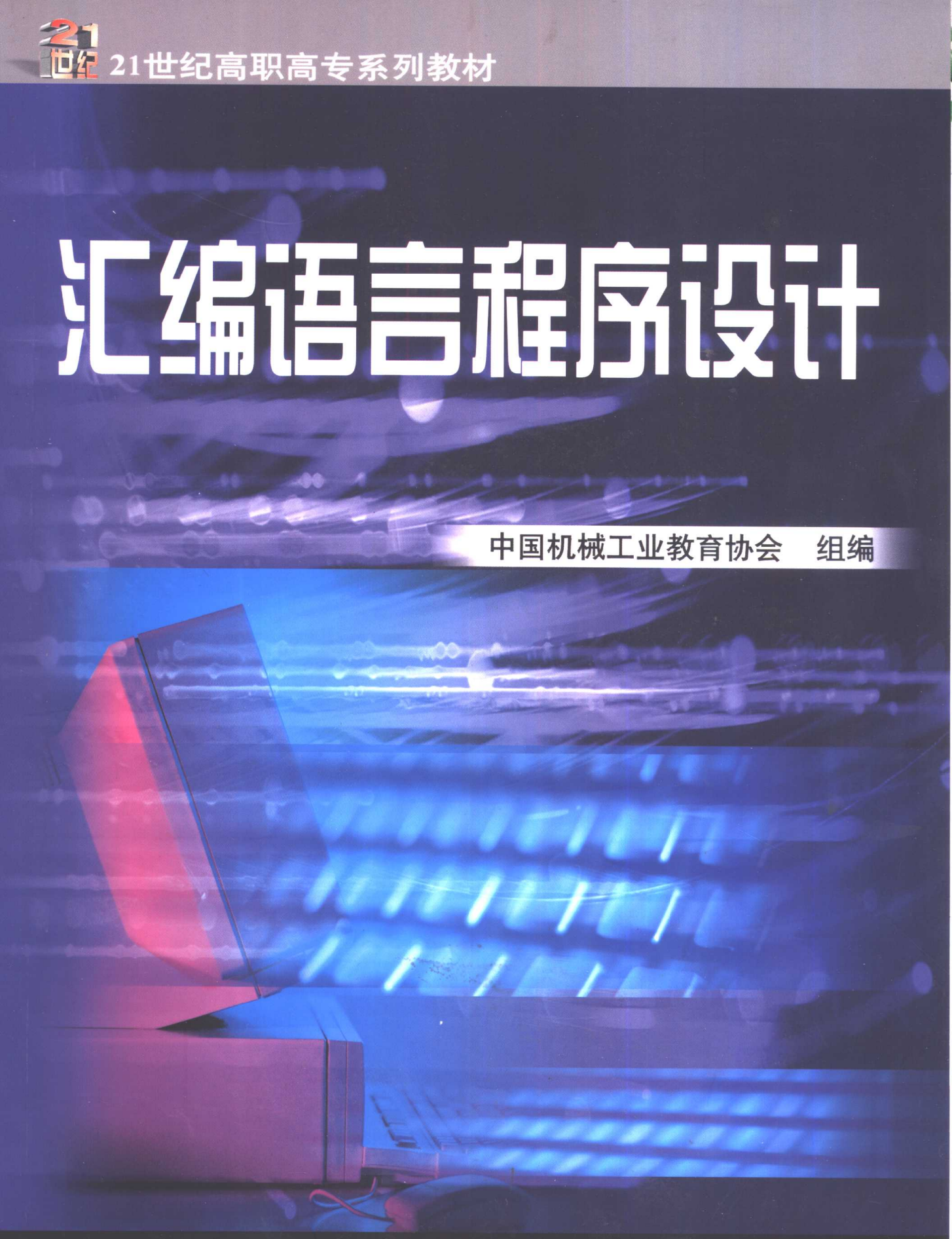
21世纪高职高专系列教材

汇编语言程序设计

中国机械工业教育协会　组编



机械工业出版社
China Machine Press

21世纪高职高专系列教材

汇编语言程序设计

中国机械工业教育协会　组编

主　编　北京科技大学职业技术学院　邹广慧

参　编　日照职业技术学院　焦卫峰　王立峰

天津理工学院职业技术学院　赵　颖

主　审　南京师范大学　李　芷

机械工业出版社

本书以 Intel 8086/8088 系列微型计算机为背景，系统介绍了汇编语言的基本概念、基本原理以及程序设计的常用方法和技术，还介绍了用计算机解决实际问题的全过程，以及调试运行汇编源程序的方法，同时还从汇编语言程序设计的角度，阐述了 80386/80486 与 8086 之间的主要区别。

全书共分 12 章。从内容的组织、概念的引入，到文字叙述、例题和习题的选择等，均以“易于学习”为目的，由浅入深循序渐进，力求遵循面向应用、重视实践、便于自学的原则，着重培养学生动手能力和思维方法。

本书深度适中，适合大专层次计算机专业的学生使用，还可作为计算机应用人员的自学参考书。

图书在版编目（CIP）数据

汇编语言程序设计/中国机械工业教育协会组编 .—北京：机械工业出版社，2001.7
21 世纪高职高专系列教材
ISBN 7-111-08409-8

Ⅰ.汇… Ⅱ.中… Ⅲ.汇编语言—程序设计—高等学校：技术学校—教材 Ⅳ.TP313

中国版本图书馆 CIP 数据核字（2001）第 044518 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
责任编辑：邝 鸥 版式设计：冉晓华 责任校对：樊钟英
封面设计：姚 毅 责任印制：郭景龙
煤炭工业出版社印刷厂印刷·新华书店北京发行所发行
2001 年 8 月第 1 版·第 1 次印刷
787mm×1092mm 1/16·12.25 印张·303 千字
0 001－4000 册
定价：19.00 元

21世纪高职高专系列教材编委会名单

编委会主任 中国机械工业教育协会 郝广发

编委会副主任（单位按笔画排）

山东工程学院 仪垂杰
大连理工大学 唐志宏
天津大学 周志刚
甘肃工业大学 路文江
江苏理工大学 杨继昌
成都航空职业技术学院 陈玉华
机械工业出版社 陈瑞藻（常务）
沈阳工业大学 李荣德
河北工业大学 檀润华
武汉船舶职业技术学院 郭江平
金华职业技术学院 余党军

编委会委员（单位按笔画排）

广东白云职业技术学院 谢瀚华
山东省职业技术教育师资培训中心 邹培明
上海电机技术高等专科学校 徐余法
天津中德职业技术学院 李大卫
天津理工学院职业技术学院 沙洪均
日照职业技术学院 李连业
北方交通大学职业技术学院 佟立本
辽宁工学院职业技术学院 李居参
包头职业技术学院 郑 刚
北京科技大学职业技术学院 马德青
北京建设职工大学 常 莲
北京海淀走读大学 成运花
江苏理工大学 吴向阳
合肥联合大学 杨久志
同济大学 孙 章
机械工业出版社 李超群 余茂祚（常务）
沈阳建筑工程学院 王宝金
佳木斯大学职业技术学院 王跃国
河北工业大学 范顺成
哈尔滨理工大学工业技术学院 线恒录
洛阳大学 吴 锐
洛阳工学院职业技术学院 李德顺
南昌大学 肖玉梅
厦门大学 朱立秒
湖北工学院高等职业技术学院 吴振彪
彭城职业大学 陈嘉莉
燕山大学 刘德有

序

1999 年 6 月中共中央国务院召开第三次全国教育工作会议，作出了“关于深化教育改革，全面推进素质教育的决定”的重大决策，强调教育在综合国力的形成中处于基础地位，坚持实施科教兴国的战略。决定中明确提出要大力发展高等职业教育，培养一大批具有必备的理论知识和较强的实践能力，适应生产、建设、管理、服务第一线急需的高等技术应用性专门人才。为此，教育部召开了关于加强高职高专教学工作会议，进一步明确了高职高专是以培养技术应用性专门人才为根本任务；以适应社会需要为目标；以培养技术应用能力为主线设计学生的知识、能力、素质结构和培养方案；以“应用”为主旨和特征来构建课程和教学内容体系；高职高专的专业设置要体现地区、行业经济和社会发展的需要，即用人的需求；教材可以“一纲多本”，形成有特色的高职高专教材系列。

“教书育人，教材先行”，教育离不开教材。为了贯彻中共中央国务院以及教育部关于高职高专人才培养目标及教材建设的总体要求，中国机械工业教育协会、机械工业出版社组织全国部分有高职高专教学经验的职业技术学院、普通高等学校编写了这套《21 世纪高职高专系列教材》。教材首批 80 余本（书目附书后）已陆续出版发行。

本套教材是根据高中毕业 3 年制（总学时 1 600～1 800）、兼顾 2 年制（总学时 1 100～1 200）的高职高专教学计划需要编写的。在内容上突出了基础理论知识的应用和实践能力的培养。基础理论课以应用为目的，以必需、够用为度，以讲清概念、强化应用为重点；专业课加强了针对性和实用性，强化了实践教学。为了扩大使用面，在内容的取舍上也考虑到电大、职大、业大、函大等教育的教学、自学需要。

每类专业的教材在内容安排和体系上是有机联系、相互衔接的，但每本教材又有各自的独立性。因此各地区院校可根据自己的教学特点进行选择使用。

为了提高质量，真正编写出有显著特色的 21 世纪高职高专系列教材，组织编写队伍时，采取专门办高职的院校与办高职的普通高等院校相互协作编写并交叉审稿，以便实践教学和理论教学能相互渗透。

机械工业出版社是我国成立最早、规模最大的科技出版社之一，在教材编辑出版方面有雄厚的实力和丰富的经验，出版了一大批适用于全国研究生、大学本科、专科、中专、职工培训等各种层次的成套系列教材，在国内享有很高的声誉。我们相信这套教材也一定能成为具有我国特色的、适合 21 世纪高职高专教育特点的系列教材。

中国机械工业教育协会

前　言

本书是根据中国机械工业教育协会对21世纪高职高专系列教材的编写要求，组织从事汇编语言教学的部分教师编写的。

汇编语言程序设计是计算机专业的一门专业基础课，是操作系统等其他课程的先修课，是计算机专业必修的核心课程之一。本书以 Intel 8086/8088 系列微型计算机为背景，系统介绍了汇编语言的基本概念、基本原理以及程序设计的常用方法和技术，介绍了用计算机解决实际问题的全过程，以及调试运行汇编源程序的方法，同时还从汇编语言程序设计的角度，阐述了 80386/80486 与 8086 之间的主要区别。

全书共分 12 章。第 1 章介绍学习汇编语言程序设计所必须掌握的计算机基础知识；第 2 章介绍 8086/8088 的寻址方式与指令系统；第 3 章介绍汇编语言中的表达式与运算符、常用伪指令和常用的 DOS 系统功能调用等内容；第 4 章～第 7 章分别介绍顺序程序、分支程序、循环程序以及子程序的设计方法；第 8 章介绍字符串操作、宏功能程序设计和模块化程序设计等；第 9 章介绍输入/输出和中断；第 10 章介绍汇编语言与高级语言的连接；第 11 章介绍 80386/80486 程序设计基础；第 12 章为上机操作与实验指导。

总结以往的教学经验，许多学生认为汇编语言入门比较困难。因此，本书内容的组织、概念的引入、文字的叙述、以及例题和习题的选择等，均以“易于学习”为出发点，由浅入深循序渐进，力求遵循面向应用、重视实践、便于自学的原则，着重培养学生的动手能力和思维方法。

本书深度适中，适合大专层次计算机专业的学生使用。还可作为计算机应用人员的自学参考书。

本书由北京科技大学职业技术学院邹广慧担任主编。其中第 1、2、3 章由邹广慧编写，第 4、5、6 章由焦卫峰和邹广慧合编，第 7 章由日照职业技术学院焦卫峰编写，第 8、9 章和第 12．1 节由天津理工学院职业技术学院赵颖编写，第 10、11 章由日照职业技术学院王立峰编写，实验指导由四位老师共同编写。全书由邹广慧统编、由南京师范大学李芷主审。俞馥敏教授和李芷副教授对本书的编写内容提出了许多宝贵意见，书末所附参考文献也给予我们很大帮助和启发，编写过程还得到北京科技大学职业技术学院马德青院长、教务处和信息系领导的大力支持，在此向有关专家和领导一并表示衷心的感谢。

由于编者水平有限，书中可能会有疏漏和不当之处，恳请读者批评指正。

编　者

目　录

第1章 基础知识

本章介绍学习汇编语言程序设计所必需的计算机基础知识。首先介绍什么是汇编语言以及为什么要学习汇编语言程序设计。然后介绍计算机的系统结构，包括微型计算机系统的组成、CPU 的功能结构、主存储器和堆栈等内容。最后以一个汇编源程序为例介绍汇编源程序的基本结构和格式，使读者尽快地对汇编语言源程序有一个初步了解。

1.1 汇编语言的一般概念

众所周知，用计算机可以解决科学计算、过程控制、信息处理和事务管理等多方面的问题，而计算机则总是把各种复杂的问题最终归结为执行特定程序来实现的。要在计算机上解决某个问题，就要针对这个问题编写一个程序，再把写好的程序输入到计算机中，然后让计算机执行这个程序。当程序执行完毕，输出相应的结果，这个问题也就解决了。

解决问题要编写程序，而程序是用程序设计语言编写的。

程序设计语言是程序设计人员和计算机对话所使用的语言，它遵循一定的规则和形式，是人机交流的工具。

随着计算机的发展，程序设计语言也是由低级向高级发展的。

1.1.1 机器语言·汇编语言·高级语言

1．面向机器的机器语言和汇编语言　机器语言是用二进制代码表示的语言，它只有 0 和 1 两种符号，是人们最早使用的一种程序设计语言。例如，下面的一组二进制代码可以完成 2+3 的运算操作：

```
10110000    00000010
00000100    00000011
```

由此可以看出，机器语言程序难读、难懂、难以调试，使用这种语言编写程序很不方便。但是，由于计算机的硬件只能识别二进制数 0 和 1，所以机器语言程序是唯一能被计算机直接识别而执行的程序。在计算机诞生的初期就是用这种机器语言编写程序的，现在除了用于编写机器的核心系统程序外，在实际中已很少直接使用这种机器语言了。

汇编语言用助记符号代替机器语言的二进制代码，是一种符号化的机器语言。例如，上面完成 2+3 运算操作的一组二进制代码，与下面的两条汇编指令功能相同：

```
MOV  AL，2              ；数值 2 传送到寄存器 AL，即 2→AL
ADD  AL，3              ；AL 中的 2 和数值 3 相加，结果在 AL 中，即 2+3→AL
```

上面两条汇编指令分别是汇编语言的传送数据指令（MOV）和加法指令（ADD）。汇编语言用人们习惯使用的英文单词或缩写符号（例如 ADD、MOV）指出计算机要进行的操作，因此可读性好，便于阅读、理解和记忆。

汇编语言是为方便用户使用而设计的一种符号语言。它比机器语言直观、方便，但随之产生的问题是用它编写的程序并不能被计算机的硬件直接识别，必须将它翻译成机器语言程序，计算机才能执行。用汇编语言编写的程序称为汇编源程序，汇编源程序经过翻译转换

成的机器语言程序称为目标程序，汇编源程序翻译成目标程序的过程称为汇编，汇编过程是由专门的软件——汇编程序来完成的。汇编程序是将汇编源程序翻译成目标程序的语言加工程序，它相当于一个翻译器，把汇编源程序翻译成机器语言的目标程序。汇编程序与汇编源程序和目标程序三者之间的关系如图 1-1 所示。目前最常用的汇编程序是 MASM 宏汇编。

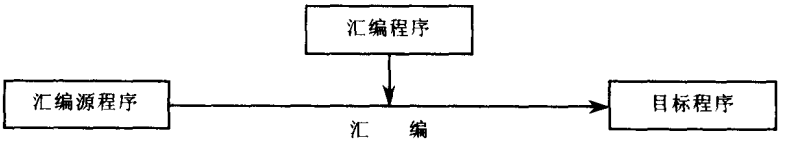


图 1-1　汇编程序与汇编源程序和目标程序之间的关系

机器指令是指挥计算机完成某一基本操作的命令。它是面向机器的，即每种类型的计算机都规定了自己所特有的、一定数量的基本指令，这些指令的全体就是计算机的指令系统。由于汇编指令与机器指令是一一对应的，不同类型的计算机，指令系统也不同，因此，汇编语言仍然是一种面向机器的语言。

2．面向过程的高级语言　高级语言是面向用户的语言。它更接近人们日常习惯使用的自然语言，更符合人的思维方式，容易学习和理解。高级语言程序由语句组成，每条语句相当于若干条机器指令，因此，高级语言的语句功能很强，方便用户的使用。

高级语言面向计算机求解问题的过程，不依赖具体的机器，因此通用性强，但同时也存在高级语言对硬件的控制能力较弱的缺点。典型的高级语言有 BASIC、PASCAL、FORTRAN、COBOL、C 语言，等等。

用高级语言编写的程序叫做高级语言源程序。计算机的硬件同样不能直接识别它，必须通过相应的编译程序，把它编译成机器语言目标程序，才能执行。有的高级语言源程序是通过解释程序，边解释边执行的，例如 BASIC 程序。

3．面向对象的高级语言　程序设计语言的更高形式是面向求解问题本身的高级语言。典型的面向对象的高级语言有 C++、Smalltalk、Eiffel 等。

综上所述，高级语言具有容易学习、使用方便、通用性强、便于交流等许多优点。既然如此，为什么还要学习和使用汇编语言呢？

1.1.2　为什么要学习和使用汇编语言

由于在汇编指令中可以直接使用寄存器、存储器地址等，因此用汇编语言编写程序，能充分发挥计算机硬件的作用，高效地使用机器。同时也需要程序设计人员了解并熟悉所用计算机的内部结构，特别是 CPU 的功能结构：有多少个寄存器、各有什么用途，CPU 是怎样访问存储器的，采用哪些寻址方式，等等。因此，学习汇编语言可以更清楚地了解计算机完成各种复杂操作的过程，从根本上认识和理解计算机的工作原理。

现在的计算机系统中，某些功能仍然是由汇编语言程序实现的。例如，机器自检、系统初始化、对输入/输出设备的控制操作等，都是用汇编语言编程来实现的，所以学习使用汇编语言仍然是很重要的。

虽然高级语言使用方便，但用于实时控制时，存在着明显的不足。与汇编语言程序相比，高级语言目标程序要多占用 0.5～1 倍的存储单元，执行时花费的时间要长 0.5～2 倍。也就是说汇编语言程序的效率通常高于高级语言程序。

现在很多高级语言都设置了与汇编语言接口的功能，以便于用户用汇编语言编写子程序，完成与机器联系紧密的特定功能，提高高级语言程序的效率。

综上所述，对于从事计算机研究和应用的人员来说，掌握汇编语言程序设计技术是非常重要的。本书以 IBM-PC 系列微型计算机及其兼容机为背景，介绍微型计算机的基本组成、内部结构和相应的指令系统。

1.2 微型计算机系统的组成

计算机系统由硬件（Hardware）和软件（Software）两部分组成。计算机硬件指的是组成计算机的那些电子元件、机械部件和设备，这些元件、部件和设备都是看得见、摸得着的有形实体。软件是相对于硬件而言的，是指计算机所使用的各种程序的集合，其中包括和这些程序有关的资料，如程序的使用说明及各种文档等。计算机系统的所有功能，都是由这两部分相辅相成来实现的。硬件和软件在发展上也是相互促进的，硬件为软件的发展奠定了基础，而软件的发展又对硬件提出了更高的要求。只有将它们有机地结合起来，才能充分发挥计算机的作用。

1.2.1 硬件系统

微型计算机硬件系统由主机和外部设备（简称外设）构成。主机由运算器、控制器、主存储器（内存）和输入／输出（I/O）接口组成，其中运算器和控制器集成在一个芯片上，叫做微处理器（Microprocessor），即中央处理器 CPU（Central Processor Unit）。主机内各部分由系统总线连接在一起。

外设一般包括外存储器（简称外存）和输入/输出设备。由于内存容量有限，所以使用外存作为内存的后援设备。外存的容量一般比内存大得多，常用的外存储器有：磁盘（硬盘和软盘）、光盘、磁带等。常用的输入/输出设备有键盘、显示器、打印机等。目前很多微机还配有调制解调器等其他外部设备。

微型计算机硬件系统的组成如图 1-2 所示。

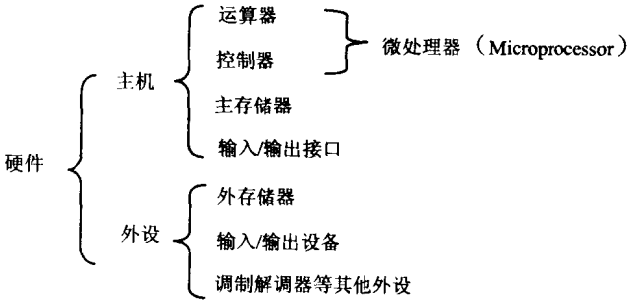


图 1-2 微型计算机硬件系统的组成

1.2.2 软件系统

微型计算机的软件系统由系统软件和应用软件两部分组成。系统软件一部分是由计算机生产厂家连同硬件一起提供给用户的，它们是用户使用机器所必需的基本软件；另一部分是用户根据实际需要从软件供应商处购买来的。系统软件一般包括：操作系统、编辑程序、编译和解释程序、汇编程序、连接定位程序、调试程序、输入/输出驱动程序、数据库管理

系统等。应用软件是为解决某个实际问题而编写的程序的集合，如：科学计算程序、数据处理程序、企业管理程序、会计电算化软件等。

微型计算机软件系统的组成如图 1-3 所示。

操作系统（Operating System）是系统软件的核心。它的主要作用是统一管理微型计算机的所有软、硬件资源。微型计算机的所有应用程序都是在操作系统的控制下自动而协调地运行的。目前微型计算机常用的操作系统有：MS-DOS、Windows 95/98/2000、Windows NT、OS/2、Unix、Netware 等。

编辑程序（Editor）可以用来输入和编辑文本，并将其存入存储器中。文本是指由数字、字母、符号等信息所组成的文件，它可以是用汇编语言或高级语言编写的源程序，也可以是一组数据或一份报告。用编辑程序也可以编辑使用说明等文档。MS-DOS 中的 EDIT 就是一个能方便地输入和编辑文本的全屏幕编辑程序。

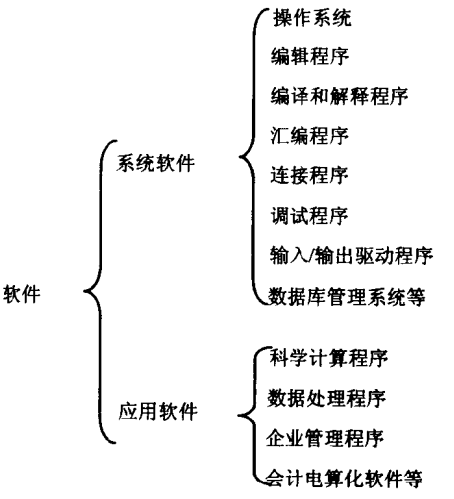


图 1-3　微型计算机软件系统的组成

编译程序（Compiler）和解释程序（Interpreter）是把高级语言源程序翻译成微型计算机能直接识别的二进制代码的系统翻译程序。编译程序是先把高级语言程序翻译成机器语言程序，然后再执行；而解释程序则是一边翻译一边执行。

汇编程序（Assembler）的作用是把汇编源程序翻译成机器语言目标程序。

连接程序（Link）把一个或多个独立的目标程序模块、库文件连接起来，形成微型机能够运行的可执行文件。

调试程序是用来监督和控制用户程序的一种工具，例如，MS-DOS 中的 DEBUG，它可以装入、修改、显示和逐条执行一个程序。

输入/输出驱动程序用来对输入/输出设备进行管理和控制。当系统程序或应用程序需要使用输入/输出设备时，通过调用输入/输出驱动程序对相应的设备发出命令，从而完成 CPU 和输入/输出设备之间的信息传送。

综上所述，我们可以这样认为，微型计算机的硬件就像人的躯壳，而软件就像人的灵魂，两者相互依存，缺一不可。我们将要学习的汇编语言程序设计技术属于软件的范畴，同时也需要我们了解有关硬件结构的知识。

1.3　CPU 的功能结构

IBM-PC 系列微型计算机的中央处理器采用的是 Intel 公司的 8086 或 8088 芯片，它们都是在 8 位微处理器的基础上发展起来的 16 位微处理器，具有 20 位地址总线和 16 位内部数据总线。它们的区别是：8088 的外部数据总线是 8 位的（称为准 16 位机），而 8086 的外部数据总线是 16 位的，也就是说它们仅在硬件接口上有些差别，在软件上则是完全兼容的，具有完全相同的指令系统，所以从软件设计的角度看，它们没有什么区别。

1.3.1　Intel 8088 微处理器的结构

Intel 8088 微处理器按功能可分为两大部分：指令执行部件 EU(Execution Unit)和总线接

口部件 BIU(Bus Interface Unit)，其内部结构如图 1-4 所示。

图中虚线左边是指令执行部件（EU），右边是总线接口部件（BIU）。

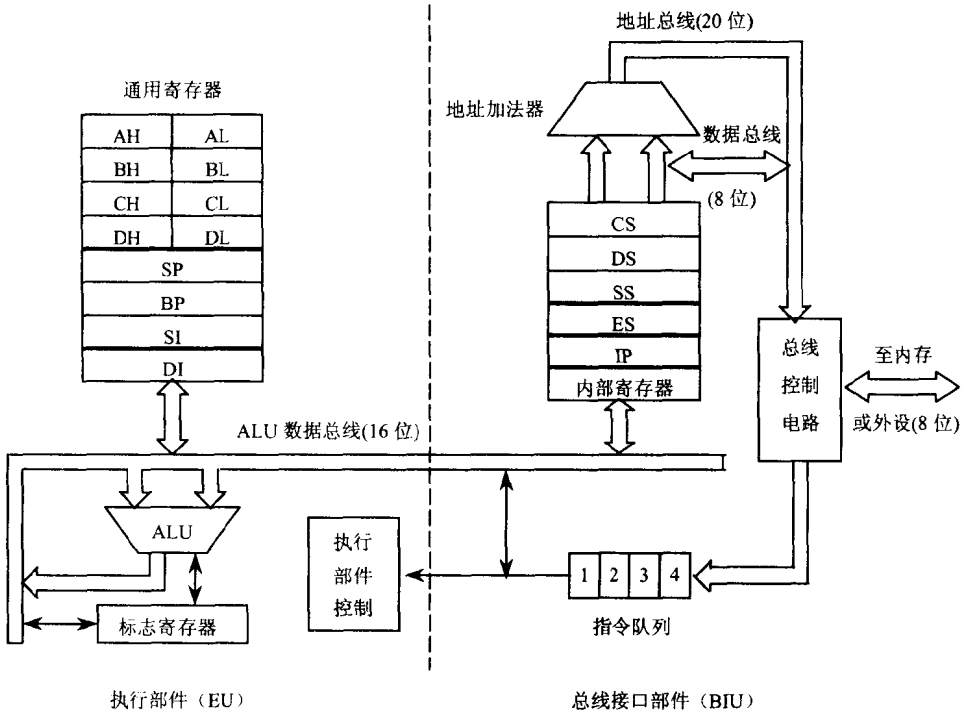


图 1-4　Intel 8088 微处理器的内部结构

1．执行部件（EU） 执行部件负责指令的执行。它由通用寄存器组、算术逻辑运算部件（ALU）、标志寄存器和执行部件控制系统组成。所有寄存器及数据通路都是 16 位的。EU 从 BIU 的指令队列中取得指令并执行，主要完成两种操作：一是根据指令进行算术 / 逻辑运算；二是计算出操作数存放的地址（内存数据的偏移地址或输入 / 输出数据的端口地址）。

2．总线接口部件（BIU） 总线接口部件负责 CPU 与存储器或外部设备之间的数据传送。它由地址加法器、段寄存器（CS、DS、SS、ES）、指令指针 IP、指令队列和总线控制电路组成。在 EU 执行指令的同时，BIU 从存储器中取出后续指令送到指令队列中排队，等待执行。

概括地说，BIU 从主存储器取出指令和数据送 EU 执行运算操作，EU 执行操作后再把运算结果通过 BIU 存入存储器或送输出设备输出。

由于 EU 和 BIU 是分开独立工作的，所以取指令和执行指令可以同时进行，从而节省了 CPU 因取指令而等待的时间，提高了处理速度。

1.3.2　程序的执行过程

程序的执行过程是由取指令和执行指令两种操作的循序反复完成的。为提高 CPU 的运行速度，8086/8088 采用并行工作方式，即取指令和执行指令同时进行。假设程序的指令代码已预先存放在存储器中，则 CPU 执行程序的过程如下：

1) BIU 从存储器中取出第一条指令放入指令队列中。

2) EU 从指令队列中取出指令并执行，同时 BIU 利用总线的空闲时间，从内存取出下一条要执行的指令放入指令队列中。

3) 如果 CPU 正在执行的指令有写存储器或输出设备的要求，则通知 BIU，由 BIU 把操作结果写入相关设备中。如果指令要求从存储器或输入设备中读取操作数，也由 BIU 来完成。

当一条指令执行完毕，EU 接着从指令队列中取出下一条指令继续执行，即重复执行 2)、3)的内容，直到程序结束。

1.3.3 8088 的寄存器

由于在汇编指令中可以直接使用寄存器，所以了解寄存器的作用是很重要的。8086/8088 内部有 14 个 16 位的寄存器，按功能可分为三大类：第一类是通用寄存器（8 个）；第二类是段寄存器（4 个）；第三类是控制寄存器（2 个）。8088 的寄存器如图 1-5 所示。

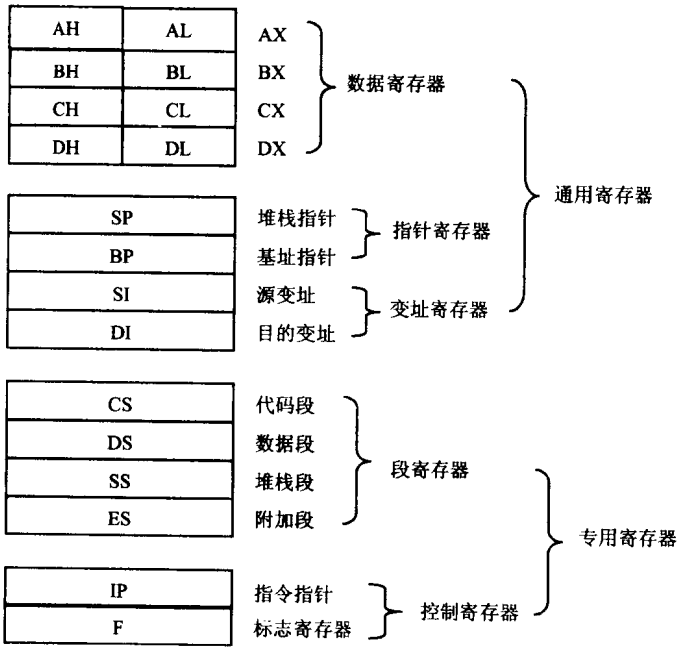


图 1-5 8088 的寄存器

1.3.3.1 通用寄存器组

通用寄存器组包括 4 个数据寄存器（AX、BX、CX、DX），2 个指针寄存器（SP、BP）和 2 个变址寄存器（SI、DI）。

1．数据寄存器 数据寄存器主要用来存放参加运算的操作数或运算结果等信息。它们既可以作为 16 位寄存器使用，又可以按高 8 位和低 8 位分开，作为 8 位寄存器使用。当作 16 位寄存器使用时，他们被分别称为 AX、BX、CX、DX；当作 8 位寄存器使用时，AX 的高 8 位称为 AH，低 8 位称为 AL；BX 的高 8 位称为 BH，低 8 位称为 BL；CX 的高 8 位称为 CH，低 8 位称为 CL；DX 的高 8 位称为 DH，低 8 位称为 DL。

虽然上述 4 个数据寄存器均具有通用性，但它们又有各自的习惯用法。AX（Accumulator）通常作为累加器，它是进行算术运算的主要寄存器。BX（Base）常作为基址寄存器，在计

算内存地址时，用来存放基址。CX（Count）常作为计数器使用，在循环和串操作指令中充当计数器。DX（Data）用作数据寄存器。当作双字运算时，用 DX 和 AX 合起来存放一个双字长数据（32 位），其中 DX 存放高 16 位。

如果没有数据寄存器，在指令执行过程中要用到操作数时，只能从存储器中取出，运算结果也必须立即送到存储器中保存起来。而从存储器中存取数据要占用总线和主存的时间，因此，用数据寄存器暂存参加运算的操作数和运算过程的中间结果，可以提高程序的执行速度。一般来说，处理器中包含的寄存器越多，处理器使用就越灵活，处理器执行程序的速度也就越快。

2．指针、变址寄存器　指针、变址寄存器 SP、BP、SI、DI 均为 16 位寄存器，一般用来存放操作数的偏移地址。其中堆栈指针 SP（Stack Point）用来指示堆栈栈顶的偏移地址，基址指针 BP（Base Point）用来指示堆栈中某一存储单元的偏移地址，SI（Source Index）和 DI（Destination Index）为变址寻址操作提供基地址。通常在串操作指令中，SI 存放源操作数的偏移地址，DI 存放目的操作数的偏移地址。

当 SI、DI 和 BP 不用做指针、变址寄存器时，也可以把它们当作数据寄存器使用，存放操作数和运算结果，但它们只能作为 16 位寄存器使用。SP 一般不做数据寄存器使用。

1.3.3.2　专用寄存器组

专用寄存器组包括 4 个段寄存器（CS、DS、SS、ES），2 个控制寄存器（指令指针 IP 和标志寄存器 F），它们都是 16 位的寄存器。

1．段寄存器　在 8088 中采用了存储空间分段技术，将存储器划分成若干段，把要运行的程序的各部分（代码、数据等）分别放在不同的存储段中，每个存储段用一个段寄存器指示它的首地址：代码段寄存器 CS（Code Segment）指示当前代码段的起始地址，该段存放当前正在运行的程序代码。数据段寄存器 DS（Data Segment）指示当前数据段的起始地址，该段存放当前运行程序所用的数据。堆栈段寄存器 SS（Stack Segment）指示当前堆栈段的起始地址，该段定义堆栈所在的区域，堆栈是按后进先出的原则存取数据的存储区。附加段寄存器 ES（Extra Segment）指示当前附加段的起始地址，该段是一个辅助的数据存储区。

在程序运行的任何时刻，最多只能有 4 个当前段，即当前代码段、当前数据段、当前堆栈段、当前附加段。它们的首地址分别存放在段寄存器 CS、DS、SS、ES 中，这 4 个段寄存器分工不同，不能互换。

2．指令指针 IP（Instruction Pointer）　指令指针 IP 是指令存放地址的指针，具有自动增量功能。程序开始执行时，它指向该程序第一条指令的偏移地址。当从存储器取出第一条指令后，IP 自动增量指向下一条指令，即 IP 中存放着将要从存储器中取出执行的指令的偏移地址。正是由于 IP 的作用，程序才得以连续不断地执行下去。

3．标志寄存器 F（Flag）　标志寄存器用于记录指令运行结果的特征和 CPU 的状态信息。详细内容将在 1.6 节介绍。

1.4　主存储器和堆栈

1.4.1　主存储器

存储器（Memory）是存放程序和数据的装置，包括主存储器（简称主存或内存）和辅助存储器（也称外存储器或外存）。主存储器设在主机内部，用来存放当前运行的程序及其

所需的数据，内存容量较小但存取速度快；外存储器属于计算机的外部设备，用来存放暂时不运行的程序和数据，当需要运行这些程序时再调入内存执行，外存的特点是存储容量大，但存取速度相对较慢。

存储器是由存储单元组成的，存储单元的多少即为存储器的容量。

1．存储单元　主存的基本存储单位是位（bit），每个存储位可以存放一位二进制数 0 或 1。8 位二进制数组成一个字节（Byte），常用 B 表示，每个字节中的 8 位编号如图 1-6 所示。

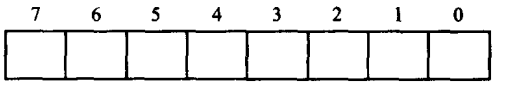


图 1-6　字节的 8 位编号

主存储器按字节组织排列成一个个存储单元，相邻的 2 个字节又可组成一个字。计算机的字长一般是字节的整数倍，如 8 位、16 位、32 位、64 位等。习惯上称 16 位为字，32 位为双字。

8086/8088 CPU 的字长为 16 位，由两个字节组成，各位编号如图 1-7 所示。

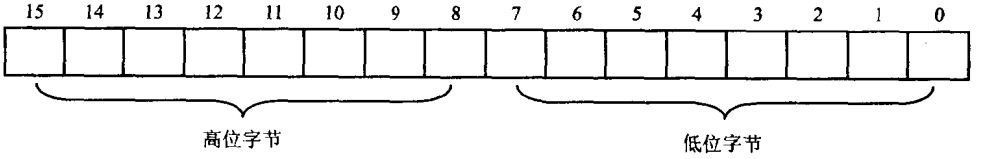


图 1-7　字的 16 位编号

2．存储单元的地址　8086/8088 CPU 具有 20 条地址线，寻址范围可达 1 兆（M）字节（2^{20}=1M），即可以在 1 兆字节（可写成 1MB）的存储器中寻找出所需要的一个存储单元。每个存储单元有一个编号，这个编号就是该单元的地址。在计算机内部存储单元的地址用无符号二进制数表示，从 0 开始编号，依序加 1。1M 字节的地址范围是 $\underbrace{00\cdots\cdots00}_{20\text{个}}$～$\underbrace{11\cdots\cdots11}_{20\text{个}}$，为了书写方便，习惯上常用 5 位十六进制数（后缀为“H”）表示存储单元的地址，1M 字节的地址范围是 00000H～FFFFFH。

3．存储单元的内容　存储单元中存放的信息称为该存储单元的内容。存储器的地址和内容如图 1-8 所示。从图 1-8 中可以看出 00100H 字节单元的内容为 27H，可表示为：

（00100H）＝27H

即用地址编号加括号表示该单元的值。

8086/8088 CPU 字长为 16 位，当以字为单位存储数据时，1 个字占用连续的 2 个字节单元，低位字节存入低地址单元，高位字节存入高地址单元。这样连续 2 个字节单元就构成了 1 个字单元，字单元的地址采用它的低地址表示。图 1-8 中 00100H 字单元的内容是 3427H，表示为：

（00100H）＝3427H

由此可见，同一个地址既可以看作是字节单元地址，又可看作是字单元地址，这要根据使用情况来确定。另外，存储单元的内容可以重复取出，直到存入新的信息，原来的内容才丢失。

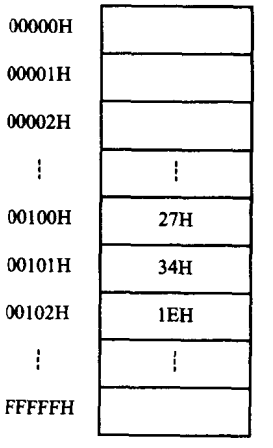


图 1-8　存储器的地址和内容