HZ BOOKS
华章教育

高等院校计算机教材系列

汇编语言程序设计

程学先　林　姗　程传慧　编著

为教师提供电子课件

机械工业出版社
China Machine Press

高等院校计算机教材系列

汇编语言程序设计

程学先 林姗 程传慧 编著

机械工业出版社
China Machine Press

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，并较深入地介绍了与汇编语言编程相关的硬件知识。本书注重程序设计思想与方法的学习，强调结构化与软件重用的思想，理论联系实际，以实例引导读者切实掌握汇编语言程序设计课程的主要知识点，进而提高程序设计能力。另外，本书针对社会上对具有一定硬件基础的计算机人才需求旺盛的特点，重点围绕输入、输出程序设计介绍了若干常用芯片的结构和程序设计方法，既加强了涉及硬件的程序设计技术的教学，也有助于提高设计实际应用系统的能力。

本书可作为高等院校计算机及相关专业应用类本科生的教材，也可供从事计算机应用与开发的各类人员学习和使用。

图书在版编目（CIP）数据

汇编语言程序设计/程学先，林姗，程传慧编著. —北京：机械工业出版社，2009.1
（高等院校计算机教材系列）

ISBN 978-7-111-25841-4

Ⅰ. 汇…　Ⅱ. ①程…　②林…　③程…　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201261号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）
责任编辑：刘立卿
三河市明辉印装有限公司印刷
2009年2月第1版第1次印刷
184mm×260mm · 20印张
标准书号：ISBN 978-7-111-25841-4
定价：36.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线（010）68326294

前　言

汇编语言是一门涉及硬件的程序设计语言，汇编语言程序设计是计算机专业的一门重要的专业基础课。汇编语言一般与某一种 CPU 提供的机器指令相对应，人们可以用它直接控制硬件系统进行工作，可以直接访问计算机系统内部各资源，汇编语言程序具有实时性强、执行速度快、代码效率高等优点。学习汇编语言程序设计时，由于软硬件知识交叉，因此对程序设计能力要求较高，学习难度较大。但学好本课程，对于了解计算机系统的组成结构与工作原理、体验底层编程的经验、更深入地学习程序设计方法很有意义，同时也会更加通透地理解数据结构、计算机组成原理、操作系统等课程中所学的知识，也为学习计算机接口、单片机及其他计算机控制类课程打下坚实的基础。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，较深入地介绍了与汇编语言编程相关的硬件知识，强调结构化与软件重用的思想。本教材以面向应用、深入浅出、重视实践、方便教学为宗旨，顺应人们实践—理论—实践的认识规律，面对枯燥、抽象的汇编语言程序，以实例先行，力求突出“怎么用”，再讨论“为什么这样用，规律是什么”，以及如何利用这些基本方法去解决实际问题，引导读者理论联系实际，切实掌握本课程主要知识点，进而提高程序设计能力。

汇编语言是一门程序设计语言，学习程序设计的思想与方法既是学习的重点也是难点，本书的目标就是要帮助读者提高程序设计的能力，较深入地掌握汇编语言程序设计的思想与方法。学习汇编语言时，有些读者已经学习过例如 C 语言等高级语言，这将为我们理解程序设计的思想打下良好基础；但也可能从未学习过其他语言，此时虽然框框较少，但建立计算机语言与程序设计的概念将是一大难题。汇编语言是多类知识交叉的学科，一部分知识往往还连带着其他的相关内容，如果处理不好这些连带的内容，将影响学习者对当前知识点的理解。但是如果先扫清外围，例如先介绍 CPU 与存储器的内部结构、数据形式，再介绍指令与指令系统、程序设计方法等等，如此虽然可以做到内容单一，也容易接受，但教学未以程序设计为线索展开，可能在开始学习后相当长一段时间里接触不到汇编语言程序设计的内容，这将令学习过程抽象无趣。本书考虑到这些情况，采用了实践先行的方法，以一个完整的程序实例引导，首先通过演示与动手，对程序设计及一些基本概念、基本方法建立感性认识，从而屏蔽了一些相关知识带来的干扰，在此基础上再逐一展开。学习的过程除了理解还有记忆与模仿，人们往往都是从死记、模仿走向创新的，实践引导的过程将为模仿创造条件，也提供了记忆的线索与内容。以程序为线索展开，将使我们在整个学习过程中的每一步都目标明确、主题清晰、基础扎实，在不知不觉中登上程序设计的巅峰。

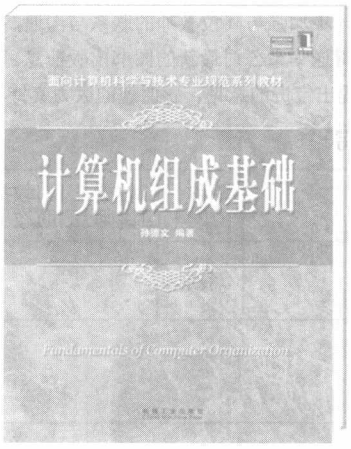
汇编语言的主要用途之一是涉及硬件的开发，目前社会上对具有一定硬件基础的计算机人才的需求较旺，本书围绕输入、输出程序设计介绍了若干常用芯片的结构与程序设计方法，既加强了涉及硬件的程序设计技术的教学，也有助于提高设计实际应用系统的能力。

全书共 9 章。第 1 章，通过一个完整的 8086 汇编语言程序示例建立对汇编语言程序及汇编过程的感性认识，在此基础上介绍数据的表示形式、基本的 DOS 系统功能调用、微处理器的构成、内存与外设的概念。第 2 章，介绍寻址方式、指令的格式与最基本的一些 8086 汇编指令，通过一些简单程序段加深对指令的深入理解，同时也充分表现了指令在程序设计中的意义。为

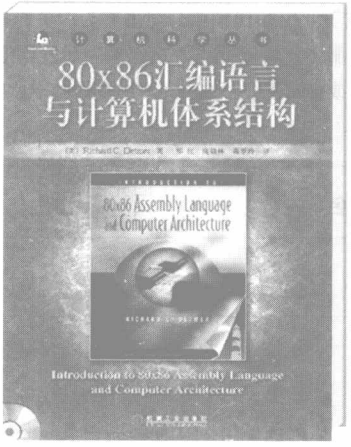
做到实践先行，第2章中有些程序（前面加有星号标志）涉及分支与循环，对于没有其他程序设计语言基础的读者，只要求对它们有所印象，实验时原样复制拷贝后编译执行即可，在学习第4章前不要求深入理解与掌握。第3章，介绍汇编语言语句格式与程序结构，伪指令及汇编语言程序开发环境，DEBUG程序调试方法。第4章，系统介绍DOS系统功能调用，程序流程概念，分支与循环程序及其设计方法，串处理程序设计方法。第5章，介绍子程序设计，结构化程序设计思想与方法，宏与宏程序设计方法。第6章，介绍BIOS系统调用，中断的概念，中断程序、输入与输出程序设计的一般方法。第7章，介绍一些常见的接口芯片，在此基础上介绍并行、串行程序，A/D与D/A转换程序，中断控制程序设计等设计方法。第8章，介绍磁盘文件的概念及其管理程序的设计方法。第9章，介绍汇编语言的其他技术，包括结构、重复块、条件汇编、多模块程序设计和C语言与汇编语言相互调用、驻留程序设计等。第10章，介绍80x86汇编的特点及概念。书后给出了几个附录，介绍了汇编语言程序设计常用的指令、DOS系统功能调用、BIOS功能调用等。本书第2、3、4、6章及7.1节、7.2节、7.4节由林姗编写，第5、8、9、10章由程传慧编写，程学先编写其他部分并修改全稿。陈永辉、史涵、鲁瑛、余小燕、周金松、堪章衡等参加校对与程序调试，在此表示感谢。

本书可作为高等院校计算机及相关专业应用类学生教材，也可供从事计算机应用与开发的各类人员学习和使用。建议本课程教学时数为90学时，其中包括实验36学时。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。

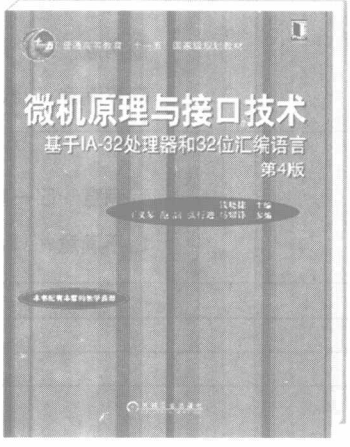
延伸阅读



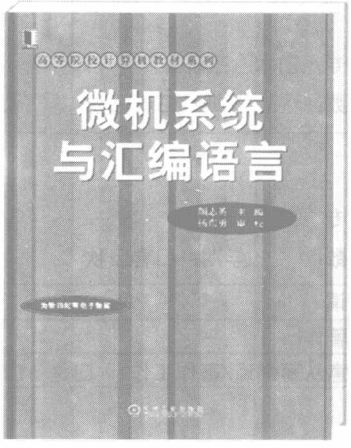
计算机组成基础
作 者：孙德文
ISBN：978-7-111-25261-0
定 价：29.00元



80X86汇编语言与计算机体系结构
作 者：(美)Richard C.Detmer
译 者：郑红　庞毅林　蒋翠玲
ISBN：978-7-111-17617-0
定 价：49.00元



微机原理与接口技术：
基于IA-32处理器和32位汇编语言　第4版
作者：钱晓捷
ISBN：978-7-111-22926-1
定价：33.00元



微机系统与汇编语言
作者：颜志英
ISBN：978-7-111-22279-8
定价：30.00元

教师服务登记表

尊敬的老师：

您好！感谢您购买我们出版的 _____ 教材。

机械工业出版社华章公司本着为服务高等教育的出版原则，为进一步加强与高校教师的联系与沟通，更好地为高校教师服务，特制此表，请您填妥后发回给我们，我们将定期向您寄送华章公司最新的图书出版信息。为您的教材、论著或译著的出版提供可能的帮助。欢迎您对我们的教材和服务提出宝贵的意见，感谢您的大力支持与帮助！

个人资料（请用正楷完整填写）

<table>
<tr><td>教师姓名</td><td></td><td>□先生
□女士</td><td>出生年月</td><td></td><td>职务</td><td></td><td colspan="2">职称：□教授 □副教授
□讲师 □助教 □其他</td></tr>
<tr><td>学校</td><td colspan="2"></td><td>学院</td><td colspan="2"></td><td>系别</td><td colspan="2"></td></tr>
<tr><td rowspan="2">联系电话</td><td colspan="3" rowspan="2">办公：
宅电：
移动：</td><td>联系地址及邮编</td><td colspan="3"></td></tr>
<tr><td>E-mail</td><td colspan="3"></td></tr>
<tr><td>学历</td><td></td><td>毕业院校</td><td colspan="2"></td><td>国外进修及讲学经历</td><td colspan="3"></td></tr>
<tr><td>研究领域</td><td colspan="8"></td></tr>
<tr><td colspan="2">主讲课程</td><td colspan="3">现用教材名</td><td>作者及出版社</td><td>共同授课教师</td><td colspan="2">教材满意度</td></tr>
<tr><td colspan="2">课程：
□专 □本 □研
人数： 学期：□春□秋</td><td colspan="3"></td><td></td><td></td><td colspan="2">□满意 □一般
□不满意 □希望更换</td></tr>
<tr><td colspan="2">课程：
□专 □本 □研
人数： 学期：□春□秋</td><td colspan="3"></td><td></td><td></td><td colspan="2">□满意 □一般
□不满意 □希望更换</td></tr>
<tr><td colspan="9">样书申请</td></tr>
<tr><td colspan="2">已出版著作</td><td colspan="3"></td><td>已出版译作</td><td colspan="3"></td></tr>
<tr><td colspan="3">是否愿意从事翻译/著作工作 □是 □否</td><td>方向</td><td colspan="5"></td></tr>
<tr><td>意见和建议</td><td colspan="8"></td></tr>
</table>

填妥后请选择以下任何一种方式将此表返回：（如方便请赐名片）

地 址：北京市西城区百万庄南街1号 华章公司营销中心 邮编：100037

电 话：(010) 68353079 88378995 传真：(010)68995260

E-mail:hzedu@hzbook.com markerting@hzbook.com 图书详情可登录http://www.hzbook.com网站查询

目　录

第1章 基础知识

本章重点：

- 初步了解汇编程序编译过程
- 认识汇编语言源程序结构
- 三类语言的概念及其比较
- 各种进制数的表示方法与相互转换方法
- ASCII 码与 BCD 码
- 8086/8088 微处理器及其寄存器的结构与功能
- 80x86 实模式存储器地址表示与计算方法

汇编语言和计算机的机器语言有着直接的联系，要理解计算机的工作原理与工作过程，学习汇编语言很有必要。要学习计算机的程序设计，懂一点汇编语言也是很有好处的，如果进行涉及计算机控制、通信、动画、虚拟现实程序设计及许多对速度要求较高的软件设计，都常要求使用汇编语言设计；汇编语言程序经常被用来改进软件和硬件控制系统的效率，即使是使用如 C、C++、Java等高级语言进行应用系统的开发，也常要求嵌入汇编语言程序模块，以提高软件或硬件控制系统的运行速度；汇编语言还常用于高级语言的程序调试，解决一些涉及机器底层的问题。

汇编语言是一种面向机器的语言，不同 CPU 的计算机机器，其汇编语言都不相同。要学习某种汇编语言，就必须首先了解应用该汇编语言的计算机的硬件结构、数据类型及其在机内的表示方法。本书围绕 8086/8088 CPU 展开学习，纯粹的 8086 PC 机已经不存在了，但现在的任何一台 PC 机中的微处理器，只要是和 Intel 兼容的系列，都可以以 8086 的方式进行工作。可以将一个奔腾系列的微处理器当作一个快速的 8086 微处理器来用。学习 8086/8088 CPU 可以方便地进行实践，较容易理解并掌握其精髓。为了与现行机器对应，使方便应用，本书还介绍了 80x86 指令系统和相关的程序设计方法。

本章介绍汇编语言的最基本的概念及学习汇编语言所必须具备的计算机系统的相关知识。

1.1 汇编语言概述

从 1946 年第一台可编程计算机 ENIAC 诞生到如今，计算机经历了电子管、晶体管、集成电路和超大规模集成电路四个发展阶段，现正朝着巨型化、微型化、网络化和智能化的第五代计算机发展，已渗透到社会和生活的各个领域。人们与计算机进行交流的“语言”也从机器语言发展到汇编语言与高级语言，现正朝着“自然语言”的方向发展。

1.1.1 汇编语言程序示例

下面是汇编程序及汇编操作过程完整一例。

首先，利用纯文本编辑器（例如记事本）编写程序，内容如下：

```
DATA   SEGMENT                    ;定义段,本程序中为数据段,用于存放数据
       A    DB   4                ;定义标号 A,代表一个存储单元,字符类型,值为 4
       B    DB   2
DATA   ENDS                       ;数据段结束
```

```
STACK   SEGMENT   STACK                          ;定义堆栈段
        DB   200    DUP(0)                      ;为堆栈段安排200单元空间,初值为0
STACK   ENDS                                    ;堆栈段结束
CODE    SEGMENT                                 ;定义段,本程序中为代码段,用于存放程序代码
   ASSUME   DS:DATA,SS: STACK  ,CS:CODE         ;对准语句,定义数据、堆栈、代码段
START:MOV   AX,DATA                             ;程序开始,"START"代表第1句:将DATA段地址送AX
      MOV   DS,AX                               ;将寄存器AX中内容传送到段寄存器DS中
      MOV   DL,A                                ;将A中数据传送到寄存器DL中
      ADD   DL,B                                ;将寄存器DL中数据与B单元数据相加,结果放到DL中
      ADD   DL,30H                              ;将寄存器DL中数据加30H变成ASCII码
      MOV   AH,2                                ;调用DOS系统2号功能,显示DL中数据值
      INT  21H
      MOV   AH,4CH                              ;调用DOS系统4CH号功能,退回DOS系统
      INT  21H
CODE   ENDS                                     ;代码段结束
      END   START                               ;程序"START"结束
```

写入后存盘，要求文件名以 ASM 为扩展名。例如文件名为 A0. ASM，文件存放在文件夹 P1 中，文件夹名字是由设计者任意取的 DOS 系统允许的名字。在文件夹 P1 中要存入汇编编译程序、连接程序与调试程序，例如：MASM. EXE、LINK. EXE、DEBUG. EXE。

从 Windows 系统桌面“开始”，用鼠标依次点击：开始→程序→附件→命令提示符。进入 DOS 命令执行窗口。图 1-1 显示了汇编生成执行程序的过程。

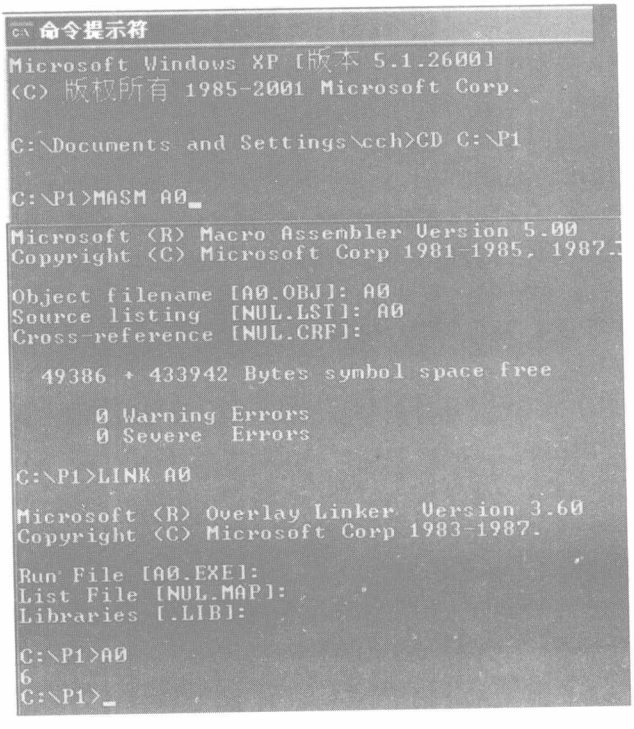


图 1-1　汇编生成执行程序全过程

如果打开文件 A0. LST，其主要内容如图 1-2 所示。

其中最左 1 列为其后语句编译后的指令代码存放的地址，第 2、3、4 列为指令代码，如果指令中有变量，其后用 R 标志，为变量所代表的数据的存放地址，再后面是源程序（原有注释内容已经删除）。

```
0000                        DATA   SEGMENT
0000  04                       A    DB   4
0001  02                       B    DB   2
0002                        DATA   ENDS
0000                        CODE   SEGMENT
                               ASSUME    DS:DATA,CS:CODE
0000  B8 ---- R             START:MOV    AX,DATA
0003  8E D8                       MOV    DS,AX
0005  8A 16 0000 R                MOV    DL,A
0009  02 16 0001 R                ADD    DL,B
000D  80 C2 30                    ADD    DL,30H
0010  B4 02                       MOV    AH,2
0012  CD 21                       INT    21H
0014  B4 4C                       MOV    AH,4CH
0016  CD 21                       INT    21H
0018                         CODE   ENDS
                                END   START
```

图 1-2　列表文件 A0. LST 的主要内容

说明：如果屏幕上提示“Object filename［A0. OBJ］”时直接回车，将按括号内的默认文件名（这里是 A0. OBJ）生成编译后得到的二进制码文件。如果输入其他名字，将按所给名字生成二进制码文件，在下面连接时注意写明新命名的文件名。

如果当屏幕上提示“Source listting［NUL. LST］”时，直接回车，意思是不要求生成列表文件。

关于汇编与调试程序更详细的方法与过程请见第 3 章关于汇编语言程序开发环境部分所介绍的内容。

1.1.2　机器语言

人们最初研制计算机的主要目的是利用它进行科学计算，要求将数字引入计算机，即要求用电子器件来表示数码，要求计算机能表示数字并能进行数学计算。人们首先想到的是十进制数字，但是，要表现十进制数字，就必须制造具有十个稳定状态的电子器件，以表示十进制中的 0 ~ 9 这 10 个数码，这是十分困难的，人类至今还没研制出具有 10 个稳态的电子器件。另外，十进制数学运算规则也很复杂，要利用机器来实现也特别困难。

人们研制与应用具有两个稳定状态的电子器件却很容易，其相关技术已非常成熟，其应用已经十分广泛。例如用双稳态电子器件的高电平表示二进制的数码 1，低电平表示二进制的数码 0，这很容易实现。又例如用有光表示 1、用无光表示 0；用磁粉磁化指向东西向为 1、南北向为 0 也都容易实现。总之，只要能实现两种不同的可以被识别的状态，就能分别表示 1 与 0。

二进制数学运算规则也十分简单，例如 1 位数的加法口诀只有 4 条：

1 + 1 = 0（进位 1）

1 + 0 = 1

0 + 1 = 1

0 + 0 = 0

利用机器实现相对容易。

如果用于控制开关的开与关，如图1-3所示，如果A点电位为高时能控制开关合上，使灯点亮；A点电位为0时能控制开关打开，使灯熄灭，我们如果定义高电平为1，0电平为0，那么我们就可以说，给A信号1可以控制灯亮，给A信号0可以控制灯灭。如果有如此这样的多个控制信号，分别控制不同开关，联合可以完成一件工作，我们就可以说，给定一个控制码，就可以实现一种功能。

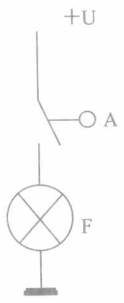


图1-3 开关控制

从上可见，无论是计算（数据处理），还是控制完成某件工作，利用二进制数据都能比较容易地实现。因此二进制就自然而然地引入了计算机，我们目前所讲的计算机都是指数字计算机，都采用二进制。这些计算机只认识0、1两个代码，人们与计算机进行信息交流：告诉计算机做什么与怎样做，都用一组0、1代码表示。

指示计算机做什么、怎样做的一组0、1代码，称作机器指令。

计算机所具有的各种机器指令的集合，称作计算机的机器指令系统，又称作机器语言。用机器语言编写的程序称作机器语言程序。机器指令指示计算机完成某一基本操作，一般由操作码和地址码两部分组成。操作码部分告诉计算机做什么，即表征机器指令的基本操作功能，比如加法，传送等；地址码部分告诉计算机怎样做，即表征指令的操作对象或操作对象所处的地址的表示形式。

Intel 8086的机器指令是多字节指令，一条指令可以由1～7个字节组成。例如，前面1.1.1节程序中，其中第1句起每条指令起始地址分别是0000H 0003H 0005H 0009H 000DH 0010H 0012H 0014H……，不难推出指令长度分别为3、2、4、4、3、2、2……。

机器指令由操作码、地址码两部分组成。例如上述程序中第5句完成将寄存器DL中数据与30H相加，结果放到DL中，该操作的Intel 8086机器指令为80 C2 30，单位为H（十六进制），变为二进制为：

10000000	操作码	
11000010	第一加数	地址码
00110000	第二加数	

机器指令中的操作码指明做什么操作及如何取得操作数，例如，上例中80H表示进行由寄存器AH或BH或BL或……与立即数做加法，后面紧跟的是寄存器名字，再后面是立即数数据大小。地址码由两部分组成，第一部分指明了存放在数据段中的第一加数，C2H表示DL、C4H表示AH、C7H表示BH、C3H表示BL，等等。第二部分指明了第二加数为立即数30H。

1.1.3 汇编语言

机器指令是用0、1代码表示的，例如前面从第1句起的机器指令为：1011100000010001000011001000111011011000100010100001011000000000000000000001000010111000000001000000001000000011000010001100001011010000000010……。

要读懂该程序十分困难，编写、调试也非常困难。如果改用十六进制形式：B8110C8ED88A1600000216010080C230B402……，依然难懂难写。为了解决读写等问题，人们考虑用符号来表示机器指令中的不同成分，例如用预先约定的助记符来表示操作码，用变量、标号表示内存存放的数据、指令的地址，用寄存器名来表示寄存器的编号，等等。由于约定的符号尽量与人们的自然语言靠近，简练且意义较容易让人理解，程序员写程序就方便了。再利用程序将由符号

构成的程序翻译成机器指令码，机器将能读懂并遵照执行。这样写成的由符号构成的程序能比较容易地让程序员理解，程序调试也变简单了。

这种用符号书写的，其主要操作与机器指令基本上一一对应的，并遵循一定语法规则的计算机语言称为汇编语言。

由汇编语言编写的程序，称为汇编语言源程序，前面所举的例子中 A0. ASM 中的内容就是一个汇编语言源程序。

汇编语言是面向机器的语言，对于不同的机器其汇编语言将不相同，但方法相同，我们如果学会了 Intel 80x86 的汇编语言，再学习其他机器的汇编语言也将容易得多。

前面 1. 1. 1 节中的例子，是用 Intel 8086 宏汇编语言的语句来写的，其第 5 句内容书写为：

```
ADD  DL,30H
```

该指令中，助记符 ADD 表示加法操作；符号 DL，表示第一个加数（目的操作数）为 8 位寄存器 DL；立即数 30H 在指令中直接给出，它为第二个加数（源操作数）；两个加数相加的和存入寄存器 DL 中。

由于计算机只认识 0、1 代码，所以计算机无法识别与执行用汇编语言所编写的源程序，要想让计算机识别就必须先把汇编语言源程序翻译成机器语言程序，又称目标程序，再通过连接程序（LINK）连接，生成可执行程序，才能被计算机识别执行。比较一下 A0. LST 中的内容与前面提到的机器指令 B8110C8ED88A16000002160100，可以看到它们存在不同：经过连接之后，原来的某些地址、变量，改为具体的数字了。如果一个程序是由多个程序段构成的，它们之间的区别将更大，由此可见连接程序所起的作用。

把汇编语言源程序翻译成目标程序的程序称为汇编程序（MASM）。把汇编语言源程序翻译成目标程序的过程称为汇编。汇编语言源程序、汇编程序、目标程序、连接程序和执行程序之间的关系如图 1-4 所示。

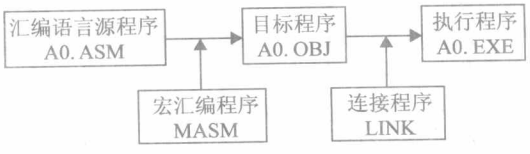


图 1-4　源程序、汇编程序和目标程序间的关系

图 1-4 中汇编语言源程序 A0. ASM 是汇编的对象，目标程序 A0. OBJ 是编译的结果，执行程序是汇编的结果，汇编程序是翻译器，连接程序完成生成执行程序的操作。

1. 1. 4　高级语言

虽然汇编语言用符号表示替代机器指令，从一定程度上简化了程序的编写、阅读和调试，但没经过汇编语言程序设计训练的人员，编起程序来还是十分困难，不利于计算机进一步推广和应用。

于是，人们尝试设计一种脱离具体的计算机、面向过程、更符合人们思维和更容易为人们所理解和学习的语言，称作高级语言。20 世纪 50 年代末，第一个主要用于科学计算的高级语言——FORTRAN 语言诞生了，随后各种高级语言如雨后春笋般地涌现出来。目前世界上使用的高级语言有数百种以上，而且具有高功效的新的高级语言不断在诞生。

由于计算机只能识别执行 0、1 代码组成的机器语言程序，所以各种高级语言的源程序只有经过相应的编译程序或解释程序翻译成目标程序，经过连接程序连接后，才能被计算机识别执行。高级语言源程序、编译程序或解释程序、目标程序之间的关系如图 1-5 所示。

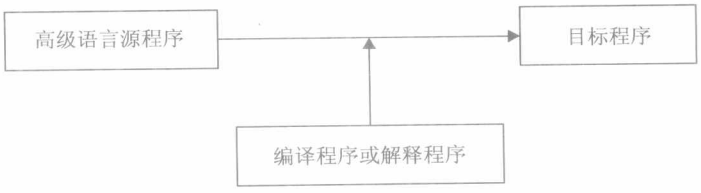


图 1-5 高级语言源程序、编译或解释程序和目标程序之间的关系

1.1.5 三种语言的特点比较

机器语言是由 0、1 代码组成的面向机器的语言。机器语言程序的编写、阅读和调试都十分困难，但它是计算机可直接识别执行的语言程序，占内存少，执行速度快。

汇编语言是用符号表示并替代机器指令的面向机器的语言。与机器语言相比，汇编语言较易于理解和记忆，也易于编写、阅读和调试。由于其语句与机器指令语句一一对应，所以具有占内存少、执行速度快的特点，并且能直接控制计算机的硬件设备，充分发挥计算机的硬件功能。但从本质上讲，汇编语言仍然是面向机器的语言，这就要求程序设计者要了解不同的计算机和熟悉不同的计算机的指令系统，从而给学习汇编语言带来了困难。另外，汇编语言的指令语句所能实现的都是一些简单的操作，如寄存器数据的移位、寄存器与内存之间数据的传递、寄存器间数据的逻辑运算等，与程序设计者所要完成的任务有很大的差别。例如 1.1.1 节程序中要完成“将某数据在显示器上显示”的任务时，程序设计者必须把这一任务转换为一条条汇编语言的指令语句：首先将该内存单元的数据传送到 DL，继而变成对应的 ASCII 码，再用 DOS 系统功能调用送显示器显示。假如要显示的数据超过一位，就更复杂了，先要将数据送到累加器中，然后用移位指令、逻辑指令、循环指令、传送指令将数据变成十进制或十六进制数，再从高位到低位依次取每一个字，并用加法指令转化成其对应的 ASCII 码，最后用 DOS 系统功能调用送显示器显示。这种转换工作是困难和耗费精力的，这促使人们设计并使用高级语言。

高级语言的共同特点是：脱离具体的机器，面向过程，是一种更类似于自然语言和数学描述语言的程序设计语言。这些语言的程序易于编写、阅读和调试，且可移植性好。但它们通用于不同的机器，不同机器各具特色的内容也要改变为大家共同的成分，这就使得它们难以产生有效的机器语言程序，运行速度较慢，占内存较大，这些地方它们是不如汇编语言程序的。

1.1.6 汇编语言源程序的格式

汇编语言程序设计的源程序格式性较强，有些指令语句几乎是在每个程序中都要出现的，我们可以结合 1.1.1 节的例子讨论程序结构。从宏观上看，汇编语言程序总是分段的，我们可以将一般汇编源程序的框架总结如下：

```
DATA    SEGMENT                                  ;定义一个段名为 DATA 的数据段
        …                                        ;定义程序中需要的常量与变量
DATA    ENDS
STACK   SEGMENT   STACK                          ;预定义堆栈段,其段名为 STACK
        …                                        ;预定义该堆栈的字节长度
STACK   ENDS

CODE    SEGMENT                                  ;定义代码段
        ASSUME   DS:DATA,SS:STACK,CS:CODE        ;对准语句
START:MOV   AX,DATA
        MOV   DS,AX                              ;数据段段寄存器赋初值
```

```
        ...                                ;数据处理代码
        MOV   AH,4CH
        INT     21H                        ;返回 DOS
CODE    ENDS                               ;代码段结束
        END     START                      ;该源程序结束
```

以上分号后为注释内容，写的目的是帮助程序员阅读与理解程序，在编译过程中不起作用。

可以不定义数据段与堆栈段。如果不定义数据段，会影响程序结构的清晰度，给修改数据带来不便。尤其是要注意，如果程序运行中需要将某些中间数据写到内存中，要防止与代码内容相冲突。如果不定义堆栈段，在编译之后会提示：error A2009：Symbol not defined：STACK。意思是说没有定义堆栈，但这只是一个通知有错（Warning Errors），将不影响其后的连接与执行程序的生成。如果是 Severe Errors（或 Errors，表示严重错误），表示编译失败，必须找出错误之处并修改程序之后重新编译。

1.2 计算机中数和字符的表示

计算机硬件只存放与处理二进制数码，不区分十进制还是十六进制，不区分正数、负数或小数，也不区分数字、字母或符号，人们需要约定怎样用二进制数来表示上述这些内容，从而使得计算机也能“认识”与“区分”它们。

下面复习进制的概念，介绍如何用二进制数表示不同的数据与符号，介绍不同表示之间变换的方法。

一个任意的 K 进制整数 N 可以表示为：

$$A_m A_{m-1} \cdots A_0$$

人们习惯于用十进制数去衡量任意进制的数的大小。N 的十进制码等于：

$$A_m \times K^m + A_{m-1} \times K^{m-1} + \cdots + A_0 \times K^0$$

其中，A_m、A_{m-1}、…、A_0 为各位数字，K^m、K^{m-1}、…、K^0 分别称为 K 进制数每位数字的权，K 称为基数。将 K 进制的每位数字乘以其对应的权，所得到的乘积之和即为该进制数所代表的值。

反过来，将任意一个某进制的数，按该进制的规则依次除以某新进制的基数，每次除后取其余数，就可以分别得到 A_0、A_1、…、A_{m-1}、A_m。倒写过来 $A_m A_{m-1} \cdots A_1 A_0$，就可得到新进制表示的数。以下联系几个常用的进制的数为例具体说明。

1.2.1 不同进位制的数及相互间的转换

人们习惯上最常用的是十进制。十进制是用 0、1、2、3、4、5、6、7、8、9 十个数码来表示数，且遵循逢十进一规则的数制。一个任意的十进制整数 N 可以表示为：

$$A_m A_{m-1} \cdots A_0$$

其含义为 N 等于：

$$A_m \times 10^m + A_{m-1} \times 10^{m-1} + \cdots + A_0 \times 10^0$$

式中，A_m、A_{m-1}、…、A_0 可取值 0～9 十个数码中的任意一个，10^m、10^{m-1}、…、10^0 分别称为十进制数每位数字的权，10 称为十进制的基数。将十进制的每位数字乘以其对应的权，所得到的乘积之和即为该十进制数所表示的值。

例如：$58679 = 5 \times 10^4 + 8 \times 10^3 + 6 \times 10^2 + 7 \times 10^1 + 9 \times 10^0$

反过来，58679 ÷ 10，得到 5867，余数为 9

5867 ÷ 10，得到 586，余数为 7

586 ÷ 10，得到 58，余数为 6