高等学校教材

◎ 主编 陈 为

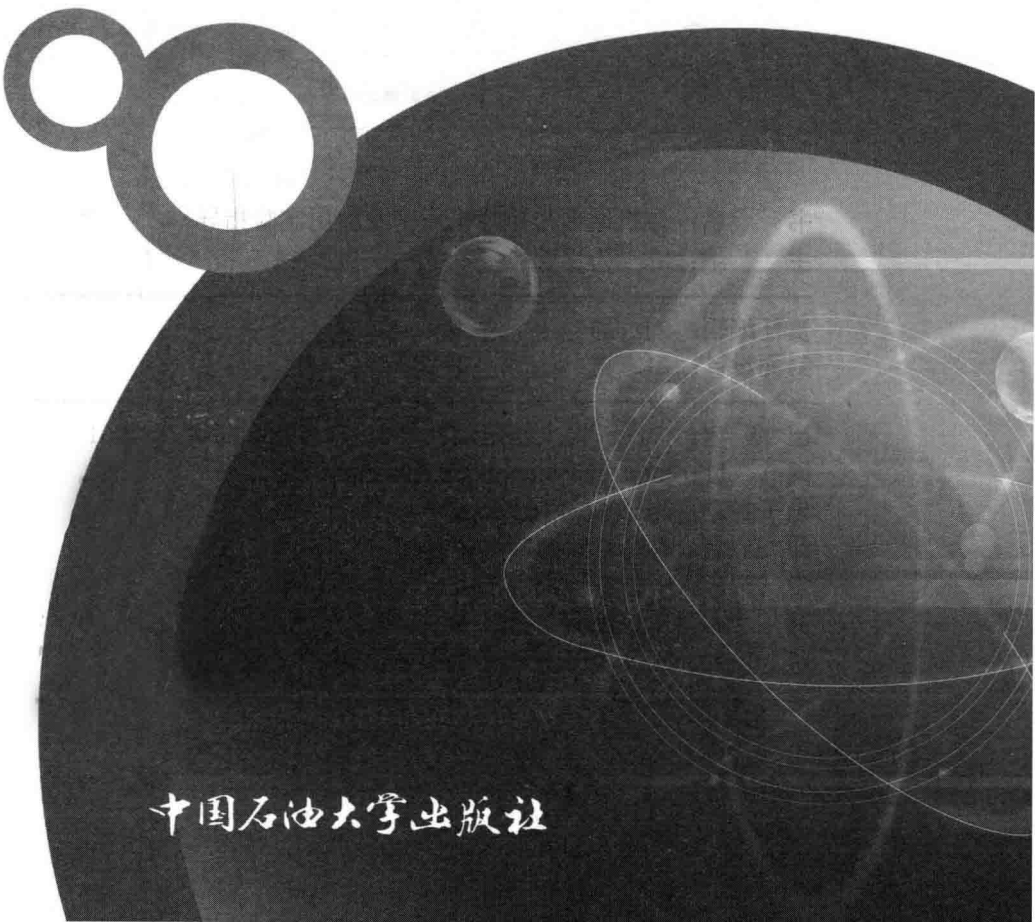
微机原理与单片机原理及应用实验指导

中国石油大学出版社

高等学校教材

微机原理与单片机原理及应用实验指导

◎ 主编　陈　为



中国石油大学出版社

图书在版编目（CIP）数据

微机原理与单片机原理及应用实验指导/陈为主编.
—东营：中国石油大学出版社，2011.4
ISBN 978-7-5636-3437-8

Ⅰ.①微… Ⅱ.①陈… Ⅲ.①微型计算机—理论②单片微型计算机—理论 Ⅳ.①TP36

中国版本图书馆 CIP 数据核字（2011）第 049046 号

书　　名：微机原理与单片机原理及应用实验指导
主　　编：陈　为

责任编辑：刘　静
封面设计：赵志勇

出 版 者：中国石油大学出版社（山东 东营，邮编 257061）
网　　址：http://www.uppbook.com.cn
电子邮箱：cbs2006@163.com
印 刷 者：青岛星球印刷有限公司
发 行 者：中国石油大学出版社（电话 0546-8391810）
开　　本：185×260　印张：8　字数：215 千字
版　　次：2011 年 4 月第 1 版第 1 次印刷
定　　价：14.80 元

PREFACE 前言

《微型计算机原理》、《单片机原理及应用》是自动化、机电一体化、检测、电子技术及应用专业的一门重要的基础课。近年来，计算机技术发展十分迅速，迄今已有很大的变化和发展。各个学校对《微型计算机原理》、《单片机原理及应用》的教学都比较重视，但与之配套的实验环节却发展十分缓慢，一直没有合适的实验讲义。我们在青岛科技大学自动化与电子工程学院的支持下，配合引进的SXL-100型微机接口实验仪，并结合学校的实际教学特点，编写了《微机原理与单片机原理及应用实验指导》。

本书针对目前在该类课程教学上的特点，将两门课的实验内容合并，采用同一套实验装置，在内容上相互依存，相互补充。其中，微机原理部分的实验侧重于基本工作原理和计算机控制的基本框架的建立，而单片机原理和应用部分的实验则侧重于将该原理用于实际应用。在编程方法上，微机原理部分主要以汇编语言为主，在单片机部分则采用汇编语言与C语言相结合的形式，使学生既能充分掌握基本原理，又可以更好地与实际应用相结合。在实验的方法上，既有基本的参考程序，也有相应的扩展应用问题，还有综合的设计内容，力求充分发挥学生的主动性和创造性，在完成基本实验内容的情况下，培养学生独立开发和设计的能力。

本书共分6章。第1章介绍实验的基本目的和要求、汇编语言的上机过程及方法；结合传统的DOS命令行及目前Windows操作的特点，采用了一种新的上机方法；强调了DEBUG的使用方法。第2章为SXL-100型微机接口实验仪的电路组成及连接方法。第3章为汇编语言程序实验，包括基本程序实验、DOS系统功能调用、代码变换、表的处理、子程序及多模块程序设计等实验。另外，为了发挥学生的主动性，锻炼其独立编写程序的能力，本章还安排了一个汇编语言综合实验。第4章为PC系列微机的硬件电路应用实验。这是学习微机原理的重要环节。本章配合教学内容共安排了5个实验，使学生对8255、8253、8259等接口芯片的使用及接口控制方法有一个全面的掌握。第5章为单片机原理及应用部分的实验。本章采用汇编语言与C语言相结合的形式，使学生既能充分掌握基本原理，又可以更好地与实际应用相结合。第6章为附录。

本书由青岛科技大学自动化与电子工程学院陈为主编，朱桂新、张彩红、庄克玉参与了部分编写工作。在编写过程中，作者参考了有关的书籍和资料，在此对相关作者表示感谢。

虽然作者在编写的过程中尽量避免失误，但由于时间紧促和经验不足，缺点和错误在所难免，有疏漏之处，恳请专家、同行和广大读者批评指正。

作　者

2011年4月

CONTENTS 目录

第1章 8086汇编语言程序设计基础

汇编语言是直接面向计算机硬件的语言，具有直接操作计算机硬件的能力。在学习汇编语言的过程中，实践环节十分重要，上机实验是快速掌握汇编语言程序设计的重要方法。本章主要介绍汇编语言程序设计的基本流程、相关工具软件的使用及程序调试方法。本章主要使学生掌握用汇编语言设计、编写、调试和运行程序的方法，从而为汇编语言实验及后续的接口实验奠定坚实的基础。

1.1 汇编语言程序设计开发过程

汇编语言程序设计开发要经过编辑源程序、汇编、连接、运行、调试等步骤。由于程序开发的不确定性，每一个步骤都经过反复调试才能得到最终正确的结果。汇编语言程序设计的主要过程如图 1-1 所示。

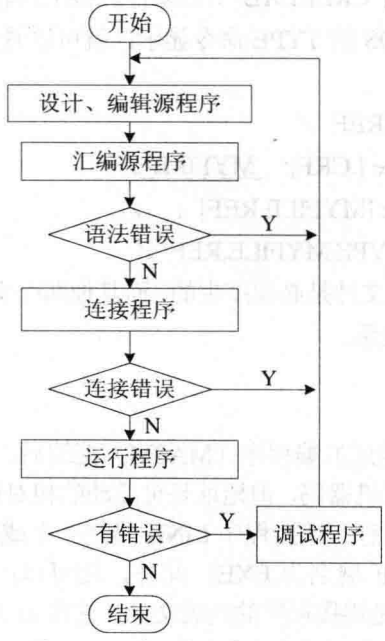


图 1-1 汇编语言程序设计的流程

1. 设计编辑源程序

该步骤用于建立源程序文件。可以用普通文本编辑器编辑并输入汇编语言源程序。常用的文本编辑器有 MS-DOS 下的 EDIT 文本编辑器、Windows 的记事本（NOTEPAD.EXE）、Windows 下的 UltraEdit 文本编辑器等。用户通过屏幕编辑程序键入源程序，检查无误后可将源程序存到磁

盘。该程序的扩展名为.ASM。

2. 汇编（MASM 或 ASM）源程序

用汇编语言编写的源程序必须是一个完整的源程序。宏汇编程序对汇编语言源程序的汇编过程包括语法检查和数据代码汇编两部分，最终生成目标程序和辅助信息文件。为了完成汇编任务，汇编程序一般采用两遍扫描的方法。第一遍扫描源程序产生符号表、处理伪指令等；第二遍扫描产生机器指令代码，确定数据等。

汇编程序有两种版本：一种是全型版本（MASM）；另一种是小型版本（ASM）。与小型版本相比，全型版本增加了宏汇编、条件汇编及错误信息全部打印输出功能。

源程序用宏汇编程序翻译（汇编）后，可以得到三个中间文件：

（1）目标代码文件，其扩展名为.OBJ。该文件将源程序的操作码部分变为机器码，但地址操作数是可浮动的相对地址，而不是实际地址，因此，需经后续的连接步骤才能形成可执行文件。

（2）列表文件，其扩展名为.LST。该文件把源程序和目标程序列表，以供检查程序用。列表程序由源程序和目标程序清单、段信息汇总表和符号汇总表等三部分组成。该文件可以在 DOS 状态下用 TYPE 命令显示或打印，以便分析调试源程序。

（3）交叉索引文件，其扩展名为.CRF。该文件是一个对源程序所用的各种符号进行前后对照的文件，它列出了源程序中定义的符号（包括标号、变量等）和程序中引用这些符号的情况。如果要查看这个符号表，必须使用 CREF.EXE 工具软件。该工具软件根据.CRF 文件建立一个扩展名为.REF 的文件，而后再用 DOS 的 TYPE 命令显示，就可以看到这个符号的使用情况表。具体操作方法如下：

```
D:\MASM> CREF ↙
  cref filename [.CRF]:  MYFILE ↙
  list filename [MYFILE.REF] :  ↙
D:\MASM> TYPE MYFILE.REF ↙
```

在这三个输出文件中，目标文件是必须产生的，而其他两个文件在需要时给予命令就可产生，对连接和执行汇编程序无直接关系。

3. 目标代码连接

用汇编语言编写的源程序经过汇编程序（MASM）汇编后，产生目标代码文件（.OBJ）。该文件将源程序操作码部分变成了机器码，但地址是可浮动的相对地址（逻辑地址），因此，必须经过连接程序 LINK 连接后才能运行。连接程序 LINK 是把一个或多个独立的目标程序模块装配成一个可重定位的可执行文件，扩展名为.EXE。此外，还可以产生一个内存映像文件，扩展名为.MAP。内存映像文件实际上是连接程序的列表文件，它给出了每个段的地址分配情况及长度。

4. 应用程序执行

当用连接程序 LINK 将目标程序（.OBJ）连接定位后，可产生可执行的应用程序文件（.EXE）。在 DOS 提示符下，键入连接程序所产生的可执行文件的文件名后，即可执行该程序。例如键入：

C:>MYFILE ↙ 或 C:> MYFILE.EXE ↙

就会把文件 MYFILE.EXE 装入内存，并从程序指定地址开始执行。在源程序 MYFILE 中，如果

有显示结果的指令，则在执行程序后可以看到执行结果；如果执行程序的结果不正确，需要动态调试应用程序 MYFILE.EXE，则可以借助动态调试程序 Debug.COM 来调试。

5. 调试程序

在编写和运行汇编程序的过程中，会遇到一些错误和问题，需要对程序进行分析和调试。调试程序 Debug 就是专为汇编语言设计的一种调试工具。它在调试汇编语言程序时有很强的功能，能使程序设计者接触到机器内部，能观察和修改寄存器和存储单元的内容，并能监视目标程序的执行情况，使用户真正接触到 CPU 内部，与计算机产生最紧密的工作联系。

1.2　汇编程序 MASM

目前，常用的汇编程序有 Microsoft 公司推出的宏汇编程序 MASM（Macro Assembler）和 Borland 公司推出的 TASM（Turbo Assembler）两种，两者的大部分内容是兼容的。本书以 MASM 为例介绍宏汇编程序的使用方法。

MASM 宏汇编程序有很多版本，如 MASM5.00、MASM6.00 和 MASM6.11 等。MASM5.00 出现于 20 世纪 80 年代，支持 80386 指令集，适用于 DOS 环境汇编程序。20 世纪 90 年代出现了 MASM6.00，该版本支持.686、.686p 和.MMX 等指令集，MASM6.11 版本开始支持 Windows 环境下的 32 位汇编语言程序。

1. MASM 5.x 宏汇编程序的使用

MASM 5.x 及以前的版本主要包含两个文件：宏汇编程序 MASM.EXE 和连接程序 LINK.EXE。宏汇编程序的主要使用方式有以下两种：

1）提问方式

在 MS-DOS 下输入命令行：MASM 或 ASM，这时 MS-DOS 将装入并启动汇编程序，然后汇编程序就逐次向用户提问，用户必须根据要求予以回答。在回答信息的最后，可输入一个或多个开关，汇编程序将按照回答信息及开关的定义做出相应处理。在回答提示信息时，可以用字符“;”省略对后续提示的回答，也可以按下 Ctrl+C 键退出汇编过程。

例 1.1　用 MASM 命令汇编源程序。

```
D:\MASM>masm ↙
……
Microsoft® Macro Assemble Version 5.00
Copyright © Microsoft Corp 1981-1985,1987,All right reserved.
Source filename [.ASM ]: MYFILE ↙
Object filename [MYFILE.OBJ ]: MYFILE ↙
Source listing [NUL.LST ]: MYFILE ↙
Cross-reference [NUL.CRF]: MYFILE ↙
50678 + 410090 Bytes symbol space free
0 Warning Errors
0 Severe Errors
```

其中，划线部分为用户键入部分；MYFILE 为源程序名（MYFILE.ASM）；方括号内是机器规

定的默认文件名。如果用户认为方括号内的文件名就是要键入的文件名，则可只在划线部分键入回车符。如果不想要列表文件和交叉索引文件，则可在[NUL.LST]和[NUL.CRF]后不键入文件名而只键入回车符。

2）命令方式

以命令方式启动汇编程序时，必须在 MS-DOS 下键入下列格式的命令：

MASM <源文件>，[<目标文件>]，[<列表文件>]，[<交叉参考文件>] [/ 开关]

命令中，MASM 后面的项目分别顺次对应提问方式中对各提示信息的回答，各项目之间用逗号隔开。“[/开关]”可放在任一项目之后。如果对某一提示信息采用默认值，则只需在对应的项目处连续输入两个逗号。例如：

MASM FUN,, FUN/D/X, FUN

2. MASM 6.x 宏汇编程序的使用

从 6.0 版本以后，Microsoft 将 Assembler 和 Linker 组合成一个 ML 程序，以前需要两个步骤（编译、连接）才能生成一个 EXE 文件，现在只要一个步骤就可以完成。此时，程序扩展名一定要是 ASM，而且执行时一定要输入扩展名。例如：

ML DISPLAY. ASM

执行后将产生 DISPLAY.OBJ 和 DISPLAY.EXE。因为缺省只会产生 OBJ 和 EXE 文件，若要产生 LST、MAP 等文件，可在 ML 之后加/Fl、/Fm 等参数。各参数之间一定要用空格分开。例如：

ML /Fl /Fm DISPLAY. ASM

表 1-1 是各参数使用的意义，全部参数均为可选参数。

表 1-1 MASM6.11 部分参数使用说明

参 数	说 明
/Fl	建立一个 LST 文件
/Fm	建立一个 MAP 文件
/Fr	建立一个 SBR 文件
/c	只编译产生.OBJ 文件

1.3 调试工具 Debug

Debug 是专门为汇编语言设计的一种调试工具，它通过步进、设置断点等方式为汇编语言程序员提供了非常有效的调试手段。调试程序的启动方法是在操作系统提示符下，按如下格式键入命令：

C:>Debug [d:] [path] [文件名] [参数 1] [参数 2]

其中，文件名是被调试文件的名称，它须是执行文件（EXE）；两个参数是运行被调试文件时所需要的命令参数，在 Debug 程序调入后，出现提示符“-”，此时，可键入所需的 Debug 命令。

在启动 Debug 时，如果输入了文件名，则 Debug 程序把指定文件装入内存。用户可以通过 Debug 的命令对指定文件进行修改、显示和执行。如果没有文件名，则以当前内存的内容工作，或者用命名命令和装入命令把需要的文件装入内存，然后再用 Debug 的命令进行修改、显示和执行。

在 Debug 的启动和执行过程中，需要注意以下事项：

（1）Debug 命令都是一个英文字母，后面跟着一个或多个有关参数。多个操作参数之间用“ ,”或空格隔开。

（2）输入 Debug 命令后，必须接着按 Enter 键，命令才有效。

（3）参数中不论是地址还是数据，均用十六进制数表示，但十六进制数后面不要用“H”。

（4）可以用 Ctrl+Break 键来停止一个命令的执行，返回到 Debug 提示符“-”下。

（5）用 Ctrl+NumLock 键中止正在上卷的输出行，再通过按任意键继续输出信息。

Debug 常用的命令一共有 18 条，这些命令如下：

1. 汇编命令 A

格式：　a. A [段寄存器名]:[偏移地址]
　　b. A [段地址]:[偏移地址]
　　c. A [偏移地址]
　　d. A

功能：用该命令可以将汇编语言程序直接汇编进入内存。

当键入 A 命令后，显示段地址和偏移地址，并等待用户键入汇编指令。每键入一条汇编指令按回车后，自动显示下一条指令的段地址和偏移地址，再键入下一条汇编指令，直到汇编语言程序全部键入，又显示下一地址时可直接键入回车返回到提示符“-”。

其中，a 的段地址在段地址寄存器中，因此，在使用该命令时必须将段地址寄存器送入段地址。c 的地址在 CS 中，d 的段地址在 CS 中，偏移地址为 100H。

2. 显示内存命令 D

格式：　a. D [地址]
　　b. D [地址范围]
　　c. D

功能：显示指定内存范围的内容。

显示的内容为两种形式：一种为十六进制内容；另一种为与十六进制相对应的 ASCII 码字符，对不可见字符以“•”代替。

a、c 每次显示 128 个字节内容，b 显示的字节数由地址范围来决定。

若命令中有地址，则显示的内容从指定地址开始；若命令中无地址（如 c），则从上一个 D 命令所显示的最后一个单元的下一个单元开始。若以前没有使用过 D 命令，则以 Debug 初始化的段寄存器的内容为起始段地址，起始偏移地址为 100H，即 CS:100。

a 中的地址为偏移地址，段地址为 CS 的内容。对于 b 中的地址范围，可以指定段地址、起始偏移地址和终止偏移地址。

3. 修改存储单元内容命令 E

格式：a. E [地址] [内容表]
　　b. E [地址]

功能：a. 用命令所给定的内容表代替指定地址范围的内存单元的内容。
　　b. 一个单元一个单元地连续修改单元内容。

其中，内容表既可以是一个十六进制数，也可以是用单引号括起来的一串字符。

4. 填充内存命令 F

格式：F [范围] [单元内容表]

功能：将单元内容表中的内容重复装入内存的指定范围内。

5. 内存搬家命令 M

格式：M [源地址范围] [目标起始地址]

功能：把源地址范围的内容搬至以目标起始地址开始的存储单元中。

其中，源地址范围和目标起始地址为偏移地址，段地址为 DS 的内容。

6. 比较命令 C

格式：C [源地址范围] ，[目标地址]

功能：从源地址范围是由起始的地址单元开始，逐个与目标起始地址以后的单元顺序比较每个单元的内容，比较到源终止地址为止。比较结果如果一致，则不显示任何信息；如果不一致，则以“[源地址] [源内容] [目的内容] [目的地址]”的形式显示失败单元的地址及内容。

其中，源地址范围是由起始地址和终止地址指出的一片连续的存储单元，目标地址为与源地址所指单元对比的目标地址的起始地址。

7. 搜索指定内容命令 S

格式：S [地址范围] [表]

功能：在指定地址范围内搜索表中的内容，搜索到就显示表中元素所在的地址。

8. 检查和修改寄存器内容命令 R

格式：a. R

b. R [寄存器名]

功能：a. 显示 CPU 内部所有寄存器的内容和全部标志位的状态。

b. 显示和修改一个指定寄存器的内容和标志位的状态。

其中，对状态标志寄存器 FLAG 以位的形式显示。显示时，8 个状态标志的显示次序和符号如表 1-2 所示。

表 1-2 状态标志显示形式

标 志 位	状 态	显示形式（置位/复位）
溢出标志 OF	有 / 无	OV/NV
方向标志 DF	增 / 减	DN/UP
中断标志 IF	开 / 关	EI/DI
符号标志 SF	负 / 正	NG/PL
零标志 ZF	零 / 非	ZR/NZ
辅助进位 AF	有 / 无	AC/NA
奇偶标志 PF	偶 / 奇	PE/PO
进位标志 CF	有 / 无	CY/NC

9. 跟踪与显示命令 T

格式：a. T [=地址] 或 T [地址]

b. T [=地址] [条数] 或 T [地址] [条数]

功能：a. 执行一条指定地址处的指令，然后暂停，显示 CPU 所有寄存器的内容和全部标志位的状态，以及下一条指令的地址和内容。

b. 为多条跟踪命令，从指定地址开始。若命令中用“[地址]”给定了起始地址，则从起始地址开始；若未给定，则从当前地址（CS:IP）开始。执行命令中的“[条数]”决定一共跟踪几条指令后返回 Debug 状态。

10. 反汇编命令 U

格式：a. U [地址]

b. U [地址范围]

功能：将指定范围内的代码以汇编语言的形式显示，同时显示该代码位于内存的地址和机器。

若在命令中没有指定地址，则以上一个 U 命令的最后一条指令地址的下一个单元作为起始地址；若没有输入过 U 命令，则以 Debug 初始化段寄存器的值作为段地址，以 0100H 作为偏移地址。

11. 命名命令 N

格式：N 文件名

功能：在调用 Debug 时，没有文件名，则需要用 N 命令将要调用的文件名格式化到 CS:5CH 的文件控制块中，才能用 L 命令把它调入内存进行调试（其他形式参考 DOS 手册）。

12. 读盘命令 L

格式：a. L [地址] [驱动器号] [起始扇区号] [所读扇区个数]

b. L [地址]

c. L

功能：a. 把指定驱动器和指定扇区范围的内容读到内存的指定区域中。其中，地址是读入内存的起始地址，若输入时没有给定地址，则隐含地址为 CS:100H。起始扇区号指逻辑扇区号的起始位置。所读扇区个数是指从起始扇区号开始读到内存几个扇区的内容。驱动器号为 0 或 1。0 表示 A 盘，1 表示 B 盘。

b. 读入已在 CS:5CH 中格式化的文件控制块所指定的文件。在使用该命令前，用 N 命令命名即可将要读入的文件名格式化到 CS:5CH 的文件控制块中。其中，地址为内存地址。

c. 同 b，地址隐含在 CS:100H 中。

若读入的文件有扩展名.COM 或.EXE，则始终装入 CS:100H 中，命令中即使指定了地址也不起作用。其中，BX 和 CX 中存放所读文件的字节数。

13. 写盘命令 W

格式：a. W [地址] [驱动器号] [起始扇区号] [所写扇区个数]

b. W [地址]

c. W

功能：a. 把在 DebugU 状态下调试的程序或数据写入指定的驱动器中。起始扇区号、所写扇区个数为要占盘中几个扇区。写盘指定扇区的操作应十分小心，如有差错将会破坏盘上的原有内容。如果在命令行中的地址只包含偏移地址，则 W 命令认为段地址在 CS 中。

b. 当键入不带参数的写盘命令时（或只键入地址参数的写盘命令），写盘命令把文件写到软盘上。在用 W 命令之前，该文件用命名命令 N 将文件格式化在 CS:5CH 的文件控制块中。

c. 只有 W 命令而没有任何参数时，与 N 命令配合使用进行写盘操作。

在用 W 命令之前，在 BX 和 CX 中应写入文件的字节数。

14. 输入命令 I

格式：I [端口地址]

功能：用来读取指定的 I/O 端口地址，并以 2 位十六进制数加以显示。

15. 输出命令 O

格式：O [端口地址] [字节值]

功能：向指定端口地址输出一个字节。

16. 运行命令 G

格式：G [=地址] [地址[地址...]]

功能：执行用户正在调试的程序。

其中，地址为执行的起始地址，以 CS 中的内容为段地址，以等号后面的地址为偏移地址，再后面的地址为断点地址。若命令行中只有起始地址，没有断点地址，则程序在执行时不中断。Debug 规定最多设置 10 个断点地址。设置多个断点用于调试较大的程序，即程序中有多个模块、多个通路时用，比较方便，在执行时不论走哪条通路，程序都可以在断点处停下来，以便调整程序。

断点地址为程序中断处的偏移地址，段地址在 CS 中。

当执行在 Debug 状态下汇编的小段程序时，只用 G 命令即可。

17. 十六进制运算命令 H

格式：H 数据 1 数据 2

功能：将两个十六进制数相加、减，结果显示在屏幕上。

其中，数据 1 和数据 2 均为十六进制。

18. 结束 Debug 返回到 DOS 命令 Q

格式：Q

功能：程序调试完退出 Debug 状态，返回到 DOS 状态。

Q 命令不能把内存的文件存盘，要想存盘必须在退出 Debug 之前用 W 命令写盘。

1.4 UltraEdit 文本编辑工具

UltraEdit 是 Windows 环境下非常优秀的文本编辑软件，有很好的可配置性，并支持语法高亮等。通过配置 UltraEdit，不仅可以实现在 UltraEdit 环境下编辑汇编源程序，而且还可以实现汇编、

连接、调用 Debug 进行调试等。配置完成后，可通过快捷键调用各编译、连接、调试功能。

1. 打开 UltraEdit，编辑并保存汇编源文件

先新建一个文件，并保存为.asm 为后缀的文件，此后，可以有语法高亮指示。在新建文件中输入汇编程序并保存。图 1-2 所示是利用 UltraEdit 输入源程序的界面。

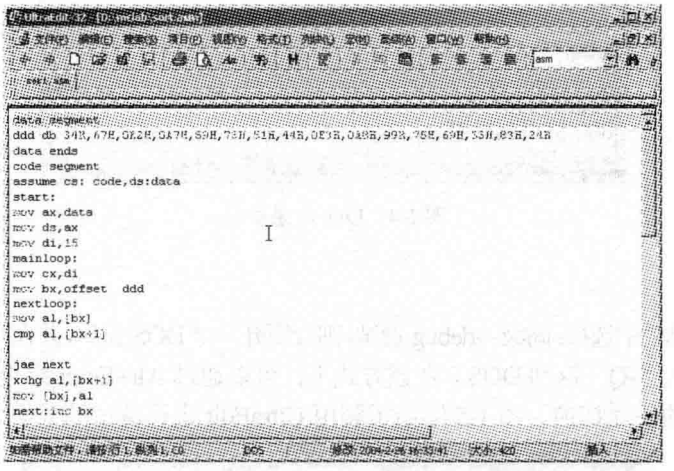


图 1-2　利用 UltraEdit 输入源程序的界面

2. 编译

从 UltraEdit 菜单栏中选择：高级→MASM 编译（或 Tasm 编译）菜单，则会调用 masm.exe（ml.exe）或 tasm.exe。输出结果放在当前编辑屏幕的下方，如图 1-3 所示。

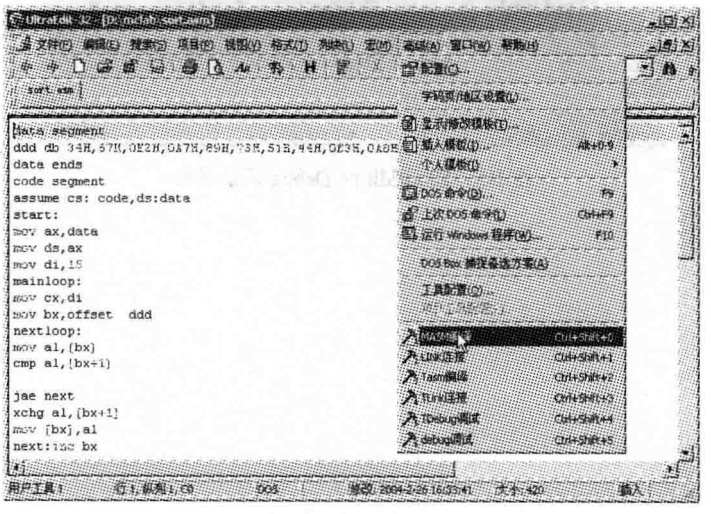


图 1-3　利用 UltraEdit 菜单选择编译功能

3. 连接

从 UltraEdit 菜单栏中选择：高级→LINK 连接，则会调用 Link.exe。输出结果也放在当前编辑屏幕的下方，如图 1-4 所示。

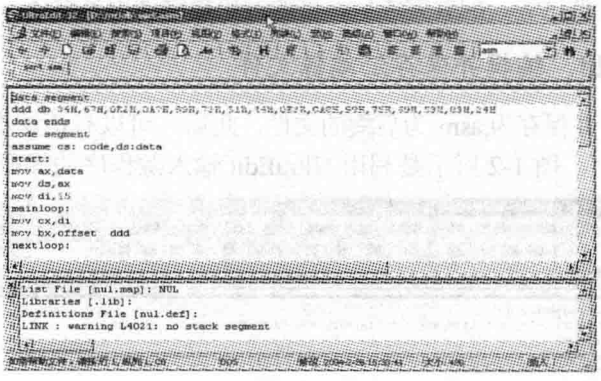


图 1-4 Link 结果显示

4. 调试

从 UltraEdit 菜单栏中选择: 高级→debug 调试, 则会新开一个 DOS 窗口, 并在其中运行 Debug.com。调试完毕，可以输入“-Q”返回 DOS。在该方式下，可以通过 Alt+Enter 组合键实现 DOS 窗口模式与全屏模式之间的快速切换。图 1-5 给出了利用 UltraEdit 进行调试的界面。

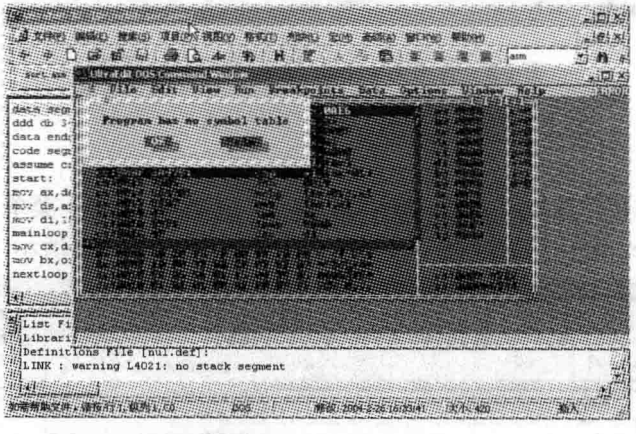


图 1-5 UltraEdit 的 Debug 调试界面

第2章　微机原理实验装置

2.1　电路结构

SXL-100 微机接口实验仪由实验扩展箱、连接电缆和 PCI 总线扩展卡组成。

1. PCI 总线扩展卡

PCI 总线扩展卡完成主控（Master）DMA 和 S5933 的 Pass_Thru 接口，即利用 S5933 的 Pass_Thru 方式来模拟产生 ISA 信号。（如果 PCI 总线扩展卡芯片是 PLX9052，则不支持主控。）

2. 连接电缆

普通 50 芯扁平电缆，用于连接 PCI 总线扩展卡和扩展板，其两端插头可互相调换。

3. 实验扩展板

SXL-100 实验扩展板的布局图如图 2-1 所示。

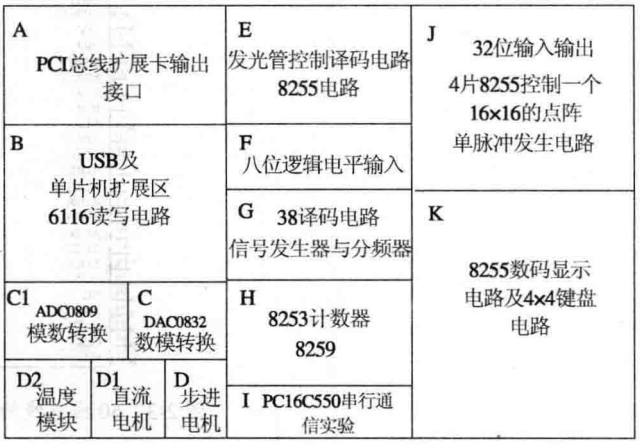


图 2-1　SXL-100 实验扩展板的布局图

4. 板图分区说明

板图分区说明如表 2-1 所示。

表 2-1　板图分区说明

A　PCI 总线扩展卡输出接口	B　USB 及单片机扩展区，6116 读写电路
C　DAC0832 数模转换电路	C1　ADC0809 模数转换电路
D　步进电机，8 MHz 信号发生器与分频器	D1　直流电机
D2　温度模块	E　发光管控制译码电路，8255 电路
F　八位逻辑电平输入	G　38 译码电路，信号发生器与分频器
H　8253 计数器，8259	I　PC16C550 串行通信实验
J　32 位输入输出，4 片 8255 控制 1 个 16×16 的点阵，单脉冲发生电路	K　8255 数码显示电路及 4×4 键盘电路