

8088/8086

汇编语言程序设计

实验指导书

李兆凤 等编

8088

8086



8088/8086汇编语言程序 设计实验指导书

李兆凤 等编

中央广播电视大学出版社

(京) 新登字163号

8088/8086 汇编语言程序设计

实验指导书

李兆凤 等编

*

中央广播电视大学出版社出版

新华书店总店科技发行所发行

北京联华印刷厂印装

*

开本787×1092 1/16 印张 7.25 千字 178

1993年10月第1版 1993年10月第1次印刷

印数 1—12000

定价 4.35 元

ISBN7-304-00836-9/TP·40

前 言

本书是《8088/8086 汇编语言程序设计》的配套教材，共分三部分。第一部分是实验的预备知识，介绍了汇编语言程序运行时所需要的一些应用程序；第二部分是实验题目，分为基本实验六个和选做实验四个；第三部分给出了实验程序参考清单，供读者参考。

为了充分体现电大多媒体教学的优越性，更好地帮助学生尽快掌握所学内容、熟悉汇编语言的上机及其使用方法，本书专门配制了一张实验软盘。该盘提供了汇编语言程序上机操作所需要的应用程序。并以自学的方式向读者介绍了这些应用程序的使用方法。它们是本课程必备的学习工具。

本书第一部分的第二节、第五节；第二部分的实验一、二、六和选做实验四由郭万平同志编写。第一部分的第三节、第四节；第二部分的实验三、四和选做实验一、二、三由张立红同志编写。第一部分的第一节、第六节、第七节和第二部分的实验五由李兆凤同志编写。全书由李兆凤同志统稿。

编 者

1993年3月

1/10/03/03

目 录

第一部分	实验预备知识	(1)
第一节	汇编语言程序的上机操作过程	(1)
第二节	编辑程序 (QE)	(2)
第三节	汇编程序 (ASM、MASM)	(11)
第四节	连接程序 (LINK)	(17)
第五节	调式程序 (DEBUG)	(22)
第六节	上机操作辅助程序介绍	(41)
第七节	PC 总线与实验台	(45)
第二部分	汇编语言程序设计试验	(47)
第一节	基本实验	(47)
实验一	分支循环程序实验	(47)
实验二	子程序设计实验	(48)
实验三	算术运算程序实验	(50)
实验四	字符处理程序实验	(52)
实验五	输入输出程序实验	(54)
实验六	中断程序实验	(57)
第二节	选做实验	(59)
实验一	排序和检索程序实验	(59)
实验二	系统调用程序实验	(61)
实验三	模块化程序设计实验	(66)
实验四	语言接口实验	(67)
第三部分	各实验程序参考清单	(70)
附录一	QE 命令一览表	(95)
附录二	汇编错误信息	(99)
附录三	连接时错误信息	(101)
附录四	LIB 错误信息	(105)
附录五	BIOS 错误信息	(107)
附录六	DEBUG 命令一览表	(109)

第一部分 实验预备知识

汇编语言程序设计是一门实践性很强的课程。编写程序，上机调试，运行程序是进一步学习和掌握汇编语言程序设计的必要手段。建立程序，调试程序需要一些应用程序的帮助。

下面将介绍一些与上机操作有关的应用程序，以为第二部分的实验做出必要的准备。

第一节 汇编语言程序的上机操作过程

一、运行汇编语言程序所需的应用程序

本课程的实验所用计算机为 IBM-PC 及其兼容机。

操作系统使用 MS-DOS2.0 以上版本。

编辑程序可用行编辑程序 EDLIN、全屏幕编辑程序 QE 或其它可用于文件编辑的应用程序。

汇编程序可用宏汇编程序 MASM 1.0 以上版本，或汇编程序 ASM。

连接程序使用 LINK 程序。

调试程序使用 DEBUG 程序。

上述软件可复制在一张软磁盘上。在对汇编语言源程序进行操作时，将调用盘上有关程序。如果计算机带有硬盘，也可以预先将它们复制到硬盘中，则使用起来就更方便。

其中应用程序将在本部分进行讲述。

二、汇编语言源程序上机操作过程

从编写好的汇编语言源程序到计算机可以运行这个程序，中间要经历一些步骤。可以归纳如下图所示。

1. 编辑源文件

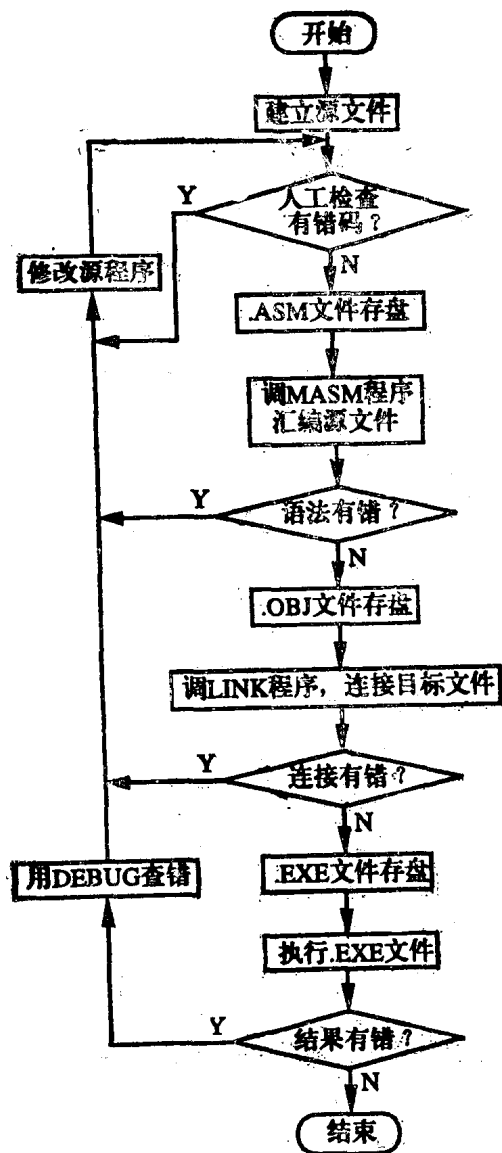
根据给定的任务，可编写程序。如果经过检查认为所编程序无错时，就可从在计算机上通过编辑程序建立源程序文件了。一般在编辑过程中总会有错误产生，因此还要进行人工检查、修改以至重新编辑，直到所编辑的源程序认为无误为止。把它以一个源程序文件（ASM 文件）存盘。

2. 汇编源程序文件

由编辑程序所建立的，ASM 文件，只是个文本文件，计算机无法执行。需要经过汇编程序的汇编，使其转为目标代码文件。它是通过调用宏汇编 MASM（或汇编程序 ASM）程序完成这一汇编操作的。汇编程序还将对源文件进行语法检查，如发现错，将给出错误原因。程序设计者可根据它分析和修改源程序，并再经过汇编，直到无错时为止，此时得到的是一个目标文件（.OBJ 文件）。

3. 连接目标文件

目标文件为浮动的相对地址，是不可直接执行的。必须经过连接，把有关模块连接在一起。



源程序上机过程示意图

构成统一的编址，从而为运行提供了条件。在运行时，装填到内存中的具体位置由段地址来确定。连接过程是由 LINK 连接程序实现的。此时得到的是一个可执行文件（.EXE 文件）。

4. 运行可执行文件

得到可执行文件后，就可以直接调用可执行文件名，从盘上调到内存中运行。如果程序设计正确，则可获得正确的运行结果。但一个较大的程序，往往会有很多设计上的逻辑错误，不可能一次就得到正确的结果。

对于较简单的程序，一般通过分析源程序就会找到出错的原因。但对于较复杂的程序，有时靠人工分析是很难发现的。此时就需要借用调试程序（如 DEBUG 程序）来调试了。在 DEBUG 调试程序控制下，可通过设置单步或断点等方法，显示有关存储单元或寄存器内容，而较容易地找到程序中的错误。总之，为试通一个汇编语言的源程序，往往要经过多次反复查错、改错，最后才能达到预期的结果。

希望我们读者，无论是编程序还是做实验，都应该细心、耐心。经过一段时间的练习，一定会达到较为自由的境地。

第二节 编辑程序(QE)

编写好一个程序，要想在计算机中调试、运行，必须要有一个程序来支持程序的编辑、修改等操作。这种支援程序叫编辑程序。编辑程序有很多种，但按功能划分，主要有两大类。

一类是行编辑程序，例如大家熟知的 EDIT 编辑程序。另一类是全屏幕编辑程序。行编辑程序功能较弱，不如全屏幕编辑程序方便、灵活。在编辑汇编语言的源程序时，可以使用已熟悉的 EDIT 程序编辑，也可以使用下面将要介绍的全屏幕编辑程序。全屏幕编辑程序有几种，下面介绍一种使用较多的全屏幕字处理程序 quik Edit（缩写 QE）。

一、编辑程序 QE 的特点

1. 快速。主要表现在以下操作比较快速：

- (1) 从磁盘上读取文件；
- (2) 显示更改屏幕信息；
- (3) 移动编辑窗口；
- (4) 寻找文本区；

(5) 写盘操作。

2. 易于操作。提供弹出式窗口功能，使用起来易于操作，用户可任意配置帮助窗口，系统提供键盘宏定义功能。

3. 简洁。系统占空间较小，仅仅47KB空间。

4. 提供多窗口。多个文件同时操作功能，文件之间相互移植很方便。

5. 系统适用范围广。可适用于 IBM PS/2, PC/AT, PC/XT, PC, PC/jr 以及 IBM 兼容机。内存最小 128K，在 PC-DOS 或 MS-DOS 2.0 以上操作系统上均可使用。

在上述特点中，最突出的是 QE 的快速性和它的系统配置的灵活性，因此，采用 QE 作为编辑程序将大大提高工作效率。

二、QE 的进入和菜单说明

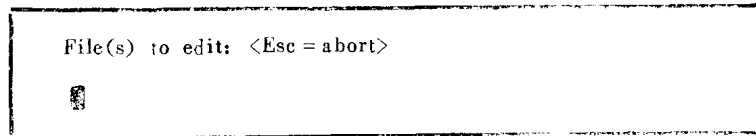
1. QE 的进入

将包含 QE 文件的工作磁盘插入驱动器 A。进入 QE 有两种选择方式

方式 1：直接打入命令：

A>QE ↵ 屏幕显示开启菜单：

A>QE ↵



注：图中方块代表光标所在位置，可键入文件名。

此时，①如果按 ESC 键，则退出 QE，返回 DOS。②如果想编辑一个程序，直接打入文件名再按回车，即可进入 QE 编辑状态。如盘中无此文件，QE 认为是新文件，则可从头编辑一新文件。如盘中有此文件，进入 QE 后，此文件在屏幕上显示出来。如下图所示：

```

L1 C1 Ins Indent                                A:/myprog, asm
-----
TITLE BELL__CLOCK
, =====
MY__STACK          SEGMENT PARA STACK 'STACK'
                   DB 256 DUP(0)
MY__STACK          ENDS
, =====
DATA              SEGMENT PARA PUBLIC 'DATA'
ADR1              DW 00
ADR2              DW 00

```

③如直接打入回车键，屏幕弹出如下窗口：


```

A: *.*
.. \
  masm          exe
  link          exe
  qe            exe
  qedit         doc
  qhelp         txt
  a1            asm
  a2            asm

```

可用上下光标键选择所需编辑的文件，按回车即进入编辑状态：

```

L1 C1 Ins Indent R W30 A:\a1.asm
-----
TITLE BELL_CLOCK
; =====
MY_STACK      SEGMENT PARA STACK 'STACK'
               DB 256 DUP(0)
MY_STACK      ENDS
; =====
DATA          SEGMENT PARA PUBLIC 'DATA'
ADR1          DW 00
ADR2          DW 00

```

第一行是状态行，它将提示：

- (1) 当前光标所在的行号 L 和列号 C；
- (2) 当前光标是否处于插入状态 (INSERT ON)；
- (3) 是否处于缩排状态；
- (4) 宏记录开关状态；
- (5) 右边界值；
- (6) 被编辑文件路径和文件名。

屏幕的其余部分为文本编辑区，可以一行行的输入程序，也可以任意移动光标到任一行和任一列，还可进行任意删、插、修改。

方式 2：在 QE 名后跟文件名，如：

A>QE a1.asm ↵

则直接进入 QE，可以进行编辑操作。

2. QE 的命令窗口菜单

对于初用 QE 的用户来说，QE 本身的一个显著优点是其提供了弹出式窗口命令菜单，供用户选择命令使用。只要按下 ESC 键，屏幕上方将出现一条主菜单栏，再用光标键选择中心词，就可执行相应的命令功能。对应于主菜单栏的每部分功能分别为：文件操作、窗口操作、块操作、查找操作、打印功能、宏定义操作、编辑操作、其它操作和退出操作。其中

每一部分的具体操作内容和含义请参考附录一。

三、文本的编辑

1. 输入文本

可以通过键盘一行行的输入文本，也可以在屏幕的任意位置输入，直到输入完毕为止，如用户发现输入有误，可随时进行修改。

2. 文本内容的插入

使用插入方法可在文本需要的地方，插入一个字符，一个字，一行或若干行新的文本内容。

进行文本的插入前，需看状态行里是否显示有 INSERT ON。如果没有出现，按 \wedge V 或 \langle INS \rangle 键，使插入软开关处于 ON 状态，然后移动光标到适当位置、打入插入文本内容。例如要在 MOV AL, FCH 中 FCH 前插入 0 的步骤是：

(1) 按 \wedge V 或 \langle INS \rangle ，使状态行显示 INSERT ON。

(2) 移动光标，将光标停在 FCH 中的 F 下面。

(3) 打入 0，此时文本变成：

```
MOV    AL, 0FCH
```

插入结束后，再按一次 \wedge V (\wedge 表示 Ctrl 键) 或 \langle INS \rangle 键，使状态行 INSENT ON 消失，以便进行其他操作。

3. 文本内容的删除

删除可以删除一个字符，一个字，一行或若干行甚至整个文件。删除前，一定要给光标定位。所有修改、插入、删除操作均取决于光标当前位置。

(1) 删除一个字符

假设要将 This is a white book. 变成 This is a book. 步骤为：

①移动光标至 white 的 W 处；

②按 \wedge G 或 \langle DEL \rangle 键 5 次，便删除了 white。

(2) 删除字

删除字符时，如要将 white 删除，则需按 5 次 \wedge G，才能实现。其实 white 是一个单字，只要按一次 \wedge T 就可以删除。做法是：将光标移至 W 处，按一次 \wedge T，便删除了 white 及跟在后面的空白字符（标点符号不在删除之内）。

(3) 删除文本行

\wedge Y 删除文本中光标所在的整个一行字符。

\wedge QT 删除从光标到该行右端所有字符。

若要删除文本中的若干行，可连续按 \wedge Y。若使用文本块操作，只要将删除的文本部分前后加上块标记（见后），按下 \wedge KY 就可以迅速删除。

4. 查找功能

执行查找功能的步骤如下：

(1) 按 \wedge QF，请求查找。

(2) 屏幕上显示出提示信息如下：

```
Search For;  $\langle$ Esc=abort $\rangle$ 
```

要求你输入所要查找的信息。输入完后，按回车键，系统又显示如下信息：

```
Options [IWB] <Ignore, Word, Backward><Esc = abort>
I
```

此时，用户可输入 I，B 或 W 任选项，来改变查找方式。其中：I 代表查找过程不区分大小写。B 代表向后查找。W 代表查找过程是按“字”查找。

(3) 按 Enter 键，光标定位在第一个要查找的字符串处，即光标指向第一个字的字母位置下，这时可对它进行需要的编辑工作，如插入、删除、修改等。

(4) 按 ^L，继续往下查找，光标定位到下一个字处。

(5) 重复 (4)，整个文件查找完毕为止。

5. 替换功能

替换操作的步骤为：

(1) 按 ^QA，请求查找并替换；

(2) 出现要替换的提示，输入后，按回车键；

(3) 屏幕上又出现要求输入替换内容的提示 change TO；按回车键，提示选择 IBW，回答后，按回车键。

(4) 输入完后，按 Enter 键，光标定位在第一个要替换的字第一个字符处，按回车键后，字将被替换；

(5) 按 ^L；可继续查找下一个字；

(6) 重复，一直替换完毕为止。

上述查找和替换过程，是从整个文本中一个一个找出指定字符串进行的，属于会话方式。在选择查找方式中，有下列选择。

I：不理睬大小写；

W：整字查找；

B：反向查找；

N：替换时不回答。

6. 文本块操作

QE 允许移动、复制、删除一个文本块。文本块是预先用命令定义的文本内容的一部分。一个文本块可以是一个句子、一段文章或整篇文章。文本块简称为块。块命令能够简化许多编辑工作。例如，文章的相互引用，部分内容前后移动位置等，都可以借助于块操作完成。

(1) 块的定义

文件块其长度可以从一个字到若干页，可以以块为单位进行传送、复制、删除，还可以把它作为一个新文件存贮在磁盘上，或把另一个文件作为块读入当前文件。一次只能定义一个块。块的大小是指以块首标记到块尾标记所包含的文本内容。

(2) 设置块标记

设置块首标记的步骤：

① 将光标移到需要作为块内容的开始位置。

② 按 ^KB (或 F7)，这时块开始处将被标记。

设置块尾标记的步骤：

① 将光标移到需要作为块内容的结束位置。

②按 \wedge KK (或 F8), 该块结束处被标记。

用户只要执行了设置块首、块尾标记, 即可对块进行各种操作。如想迅速移光标到块首或块尾处, 可按 \wedge QB 或 \wedge QK。

取消块标记 (\wedge KH)

设置好的块首、块尾标记, 可以通过按 \wedge KH 使块标记隐去。

使用块操作应注意以下几点:

①每次文本中只能定义一对块标记;

②块操作要求文本中的块首标记放在块尾标记之前;

③定义块后, 可以执行其它命令;

④只有进行块定义后, 才可执行块的操作命令。若块首或块尾没有标记时, 执行块操作时, 将见到‘块未定义’显示信息, 应重新设置缺少的块标记之后再操作。

(3) 块的移动 (\wedge KV)

文本打入后, 往往需要对某些内容调整位置, QE允许把需要移动的文本内容定义成块, 进行移动操作。移动之前, 把光标移到目标位置上, 按 \wedge KV, 定义的块便移到了光标处。块可以移到文本文件的任何地方。例如: 可移入任意行或一行之中, 块移动后, 文本的其余部分整个自动上移, 以填满移走块所留下的空间。移动时保持原块的格式, 不重新编排。且移动后, 不覆盖原先的内容, 原内容将自动下移。

例如: 移动前

```
SMY          PROC      FAR
              MOV       AX, PTR1
              CMP       AX, PTR2
              MOV       SI, PTR2
              <B> x x x x
              x x x x <K> } 定义块
              MOV AH, 02
              █ 光标位置
```

```
移动后  SMY          PROC      FAR
              MOV       AX, PTR1
              CMP       AX, PTR2
              MOV       SI, PTR2
              MOV       AH, 02
              <B> x x x x
              x x x x <K>
```

(4) 块的复制 (\wedge KC)

可以在文件中的多个地方复制定义的块。首先将光标移到复制的目标位置, 按 \wedge KC, 在光标处便复制了一个块的内容, 原块仍保留在原处。根据需要, 移动光标位置, 可在多处复制该块。连续按 \wedge KC, 可将块的内容复制多次。且复制的内容不覆盖原处内容。

(5) 块的删除 (\wedge KY)

执行命令 \wedge KY, 可以将定义的整个块从文件中删除掉。为了防止不该删除的块被删除,

使文本受影响。在使用 ^KY 之前，可用 ^QB, ^QK 确定块的定义后，再执行删除命令。

7. 文件块的传送

用写块命令 ^KW，可以把当前文本中定义的块作为另一个新文件存入磁盘，还可以用读文件命令 ^KR 把磁盘上存放的一个文件的整个内容插入文本的当前光标位置。

(1) 写文件命令 (^KW)

把当前文本中定义的块作为一个新文件存入磁盘的操作如下：

- ① 在当前文本中定义块；
- ② 按 ^KW，请求写块到一个新文件；
- ③ 这时屏幕上显示提示信息：

File to write: <Esc=abort>

- ④ 输入文件名后按 <↵>。

这样，块内容作为一个新文件以输入的文件名存入磁盘中。

在第③步中，如果回答的文件名在磁盘中已存在，屏幕上将会给出确认性的询问：

Overlay existing file (Y/N-<CR> for Yes)?

允许以新文件代替已存在的老文件，可回答 Y；若回答 N，则停止写盘，可重新操作，输入其他文件名。

(2) 读文件命令 (^KR)

从另一个文件把整个文件读入到当前文本中光标所在位置。操作步骤如下：

- ① 光标移动到需要引用其他文件内容的位置；
- ② 按 ^KR；
- ③ 屏幕上出现：File to insent=<Esc=abort>
- ④ 输入要用的文件名后，按 <↵>。

这样，指定文件中的全部内容便插入在文本的当前位置。

8. 文件存贮与退出

(1) 存贮文件并继续编辑文本 (^KS)

输入文本需要花费较长时间，为了防止由于突然停电或其他原因而造成文本丢失，在编辑文件过程中，在任何位置，任何时候可按 ^KS 命令，磁盘上便会存入当前文件的内容。

(2) 存贮文件并返回操作系统

如编辑结束，想要返回 DOS，可有两种方法退回 DOS。

方法 1：按 Alt-X 键，如现行编辑的文件与最初调入 QE 编辑区的文件一样，则不存盘，直接退出 QE，返回 DOS。如有变动，则此时屏幕出现：Save Changes (Y/N-<CR> for Yes)。按回车键，则将文件存入磁盘，并返回 DOS。如打 N，按回车键，则不存盘，返回 DOS。

方法 2：按 Ctrl-KX，此时屏幕上出现：

File(s) to edit: <Esc=abort> 此时，如按 ESC 键，则返回 DOS，如想编辑另一文件，则直接打入文件名，按回车，QE 将正在编辑的文件存盘，然后调入另一文件进行编辑。

四、编辑修改一个源程序实例

下面结合一个实例，来说明一下 QE 的用法。如果编辑的文件如下，其文件名设定用 Myproy. asm。

```

; Display string (09H)
CODE      SEGMENT PUBLIC
          ASSUME CS:CODE, DS:CODE
          ORG      100H
START:    JMP      BEGIN
MSG       DB      'Hi! This is a dollar sign terminated string.', '$'
BEGIN:    MOV      AX, CS          ; set up ds
          MOV      DS, AX
          MOV      DX, OFFSET MSG ; set up to display message
          MOV      AH, 09H        ; display string function request
          INT      21H           ; call DOS
          MOV      AH, 00H        ; terminate program funct request
          INT      21H           ; call dos
CODE      ENDS                  ; end of code segment
          END      START         ; start is the entry point

```

首先打入: A>QE MyProg. asm ↵ 此时屏幕显示:

```

editing new file... a: \myprog. asm
█
<★★★ End of File ★★★>

```

按任一键, 屏幕最上行显示如下: L1 C2 Ins Indent a: \myprog. asm

屏幕最上边一栏代表 QE 的现行状态。L1、C2 代表现行光标在第一行、第二列, Ins 代表处于插入状态, 如按 Ins 键, 使插入和改写状态相互切换。Indent 代表编排状态, a: myprog. asm, 指明现行编辑的程序的驱动器、路经名和文件名。

现在, 可以输入程序了。如输入一行完毕, 则按回车键, 光标回到上一行第一个字母列的位置上, 再继续输入, 直到输入完毕。如在输入过程中, 为防止意外, 可随时按 ctrl-ks 键, 将现行编辑的文件保存起来。如发现输入有错误, 可按 ↑, ↓, →, ← 键, 使光标到出错地方进行修改。如想删除或插入某一行, 可将光标移到指定位置, 按 Ctrl-Y 或直接打入回车即可, 假如文件已全部输入完毕。如下图示。

```

L1  C1  Ins  Indent          a: myprog. asm

```

```

; Display string (09H)
CODE      SEGMENT PUBLIC
          ASSUME CS:CODE, DS:CODE
          ORG      100H
START:    JMP      BEGIN
MSG       DB      'Hi! This is a dollar sign terminated string.', '$'
BEGIN:    MOV      AX, CS          ; set up ds

```

```

MOV      DS, AX
MOV      DX, OFFSET MSG ; set up to display message
MOV      AH, 09H        ; display string function request
INT      21H            ; call DOS
CODE     ENDS          ; end of code segment
END      START         ; start is the entry point

```

<★★★ End of File ★★★>

经检查12、13行忘输入了，此时可将光标移到第12行位置，按回车，则在此处插入一空行，然后输入即可。另外，如输入的文本文件中有两块内容完全一致，为节省时间，可采用块拷贝的方式。例如：在此例中，输入的两行和第10、11行相近，可以用 ^KB, ^KK 将第10, 11行定义为一块，如下所示。

```

L1      C1      Ins      Indent          a: myprog. asm
; Display string (09H)
CODE     SEGMENT PUBLIC
        ASSUME CS: CODE, DS: CODE
        ORG      100H
START:   JMP      BEGIN
MSG      DB 'Hi! This is a dollar sign terminated string.', '$'
BEGIN:   MOV      AX, CS          ; set up ds
        MOV      DS, AX
        MOV      DX, OFFSET MSG ; set up to display message
        MOV      AH, 09H        ; display string function request
        INT      21H            ; call DOS
CODE     ENDS          ; end of code segment
END      STARS         ; start is the entry point

```

<★★★ End of File ★★★>

然后将光标移到第12行开始处，按^KC，则将两行信息复制起来，按^KH，隐去定义块，最后，将这两行信息修改正确即可。

```

L1      C1      Ins      Indent          a: myprog. asm
; Display string (09H)
CODE     SEGMENT PUBLIC
        ASSUME CS: CODE, DS: CODE
        ORG      100H
START:   JMP      BEGIN
MSG      DB 'Hi! This is a dollar sign terminated string.', '$'

```

```

BEGIN:  MOV     AX, CS           ; set up ds
        MOV     DS, AX
        MOV     DX, OFFSET MSG ; set up to display message
        MOV     AH, 09H        ; display string function request
        INT     21H            ; call DOS
        MOV     AH, 4CH        ; terminate program funct request
        INT     21H            ; call DOS
CODE    ENDS                    ; end of code segment
        END     START          ; start is the entry point

```

<★★★ End of File ★★★>

当输入完毕后，按 ALT-X，存盘返回 DOS。

关于 QE 编辑程序的使用命令见附录一。

第三节 汇编程序 (ASM, MASM)

一、汇编程序的类别

经过编辑程序所建立的汇编语言源程序，是不能直接在计算机上运行的。必须经过汇编、连接后才能运行。目前使用的汇编程序有宏汇编程序 MASM 和小汇编程序 ASM。小汇编程序与宏汇编程序的区别在于：宏汇编程序有宏处理能力，而小汇编程序没有这种功能。另外，小汇编 (ASM) 处理的汇编语言程序可以在 64KB 的内存条件下运行，且只能有限制地使用伪指令。而宏汇编 (MASM) 则必须在 96KB 以上的内存条件下才能运行，且可以使用各种伪指令。ASM 和 MASM 的程序运行过程，除汇编时使用 ASM 或 MASM 进行汇编外，其它步骤完全相同。

二、汇编过程

宏汇编程序是二次扫描的汇编程序，它两次从头到尾逐行扫描源程序完成汇编过程。第一次扫描用来定义源程序每个行的相对位置。汇编程序中有一个软件计数器，称为定位计数器或地址计数器 LC (Location Counter)。当每个段开始时，LC 复位 (LC=0)。汇编程序在逐行扫描中，LC 按每行指令的字节数进行计数。读入一个源程序行后，先查看其是否为伪指令，若为伪指令则 LC 不计数，否则将指令长度 (指令字节数) 送 L。再检查此行语句是否有标号，若有则把标号写到一个符号表中，并且把现行 LC (即为上一条指令的计数结果) 赋给标号且填入符号表中，然后令 $LC+L \rightarrow LC$ ，形成下一条指令的地址，即这条指令在段中的相对位置，称为段内偏移量。若语句没有标号则直接做 $LC+L \rightarrow LC$ 及其后的操作，其它行按同样的方法处理，直至扫描到 END 时结束第一次扫描。这样通过第一次扫描可把源程序中的标号、变量的偏移量等填入符号表中，从而构成并得到了各条指令的相对地址和一张符号表。在第一次扫描时还计算出了源程序机器码的总长度，并可检查出一些语法错误，但不产生机器码。

汇编程序的第二次扫描是把汇编指令翻译成机器码，它根据汇编程序中的指令代码表找到汇编指令助记符对应的机器指令操作码，再根据第一次扫描产生的符号表，将指令中的符

号操作数在符号表中查到，并找到其偏移值，从而将每条汇编指令翻译成机器代码写入到目标程序文件中去。在第二次扫描时，还要进行语法检查，将第一次扫描时未检查到的语法错误查出并显示。在第二次扫描后除了产生出目标程序文件（.OBJ）外，还可根据需要产生列表文件（.LST），和交叉索引文件（.CRF）。

其中：

- 目标程序文件——其扩展名为 .OBJ。在这个程序文件中，操作码部分已变成了机器码可由机器识别，但地址操作数部分只是一个可浮动的相对地址，而不是内存中的绝对地址。

- 列表文件——其扩展名为 .LST。此文件包含源程序及其注释、地址和目标码（即汇编后的目标程序）。它们都是用 ASCII 码表示的文本文件，需要时可在 DOS 下，用 TYPE 命令将其打印出来以供查阅。

- 交叉索引文件——其扩展名为 .CRF。交叉索引文件是用来了解源程序中每个符号是在哪里定义的。哪些指令用到了这个符号，给出该指令的行号。交叉索引文件对较大的程序很有用，可以用来帮助查错。要把交叉索引文件打印出来，必须在 DOS 下用 CREF 命令把它变成可打印文件，然后用 TYPE 命令打印。

三、运行环境

为了对汇编语言的源程序文件进行汇编，必须有一台 PC/XT 档次以上的计算机。使用一张 PC-DOS 2.0 以上版本的系统盘。该盘应含有如下文件：

MASM (或 ASM)：宏汇编或小汇编程序

LINK：连接程序

CREF：索引程序

LIB：库管理程序

另备一张已格式化的空盘，用来存放用户程序。对较小程序也可直接使用系统盘存放，但系统盘应有足够的可用空间。

四、操作过程

首先启动 DOS，在 DOS 状态下将汇编系统盘插入 A 或 B 驱动器。（也可以预先装在硬盘上。下面均以 A 驱动器上操作为例进行说明）。然后调入 MASM 宏汇编程序。

屏幕显示与操作如下（其中下划线部分表示由用户键入，↓表示回车键）：

```
A>masm ↵
Microsoft (R) Macro Assembler Version 5.00
Copyright (C) Microsoft Corp 1981—1985, 1987. All rights reserved.
Source filename [.ASM]:myfile ↵
Object filename [myfile.OBJ]:↵
Source listing [NUL.LST]:↵
Cross-reference [NUL.CRF]:↵
51698 + 417454 Bytes symbol space free
0 Warning Errors
0 Severe Errors
```

宏汇编程序被调入以后，先显示版本号。然后依次提出提示，请求用户回答。