

Micro soft 4.0 版

C 语 言

程序库

下 卷

北京中国科学院希望电脑公司

一九八八年三月

目 录

第一篇 C 语言编译连接

第一章 在DOS 系统下建立C 编译系统

1.1 概述	(1)
1.2 后备磁盘	(1)
1.3 磁盘内容	(1)
1.4 快速建立硬盘运行环境的方法	(4)
1.5 快速建立软盘运行环境的方法	(6)
1.6 理解编译软件	(10)
1.7 建立环境	(12)
1.8 建立 CONFIG.SYS 文件	(14)
1.9 使用8087或80287协处理器	(15)
1.10 使用80186、80188或80286处理器	(15)
1.11 转换现有的C程序	(15)
1.12 盘上软件的组织	(15)
1.13 实际范例	(16)
1.14 批命令文件的使用	(19)

第二章 编译器MS C

2.1 概述	(21)
2.2 运行编译器	(21)
2.3 列出编译器的任选项	(27)
2.4 命名目标文件	(28)
2.5 产生列表文件	(29)
2.6 控制预处理器	(34)
2.7 语法检查	(38)
2.8 选用浮点任选项	(39)
2.9 使用80186, 80188或80286处理机	(42)
2.10 理解错误信息	(42)
2.11 调试预备	(45)
2.12 优化	(46)
2.13 编译大程序	(47)

第三章 内存工具模型

3.1 概 述	(48)
---------------	------

3.3.2	使用标准内存模型	(49)
3.3.3	使用near, far, 和huge关键字	(51)
3.3.4	建立特定的内存模型	(56)
第四章: 进一步使用MS C 的选项		
4.1	概述	(59)
4.2	禁止特殊关键字	(59)
4.3	压缩结构成员	(59)
4.4	外部名长度的限制	(60)
4.5	标记目标文件	(60)
4.6	除去缺省库的选择	(60)
4.7	改变缺省的字符类型	(61)
4.8	控制栈和堆的分配	(61)
4.9	控制浮点运算	(62)
4.10	先进的优化	(63)
4.11	控制函数调用序列	(65)
4.12	控制二进制方式和正文方式	(66)
4.13	设置数据临界值	(67)
4.14	模块和段的命名	(67)
4.15	Windows的程序的编译	(68)
第五章: 连接		
5.1	概 述	(69)
5.2	运行连接器	(69)
5.3	连接C程序文件	(74)
5.4	列表文件的格式	(75)
5.5	覆盖的使用	(77)
5.6	使用选择来控制LTINK工作	(78)
5.7	LINK是怎样工作的	(84)
第六章: 编译器CL		
6.1	概述	(89)
6.2	命令语法和选项	(89)
6.3	CL 命令中的连接操作	(91)
6.4	附加选项	(92)
6.5	与 XENIX兼容的选项	(93)
第七章: 错误信息		
7.1	概述	
7.2	运行时错误信息	(95)
7.3	编译器错误信息	(95)
		(98)

第二篇 C 语言调试工具

第一章 Microsoft Code View 调试器简介	(119)
1.1 引言	(119)
1.2 概述	(119)
1.3 关于本手册的使用	(119)
1.4 记号的约定	(121)
第二章 准备工作	(123)
2.1 引言	(123)
2.2 启动演示程序	(123)
2.3 准备好要调试的 C 程序	(123)
2.3.1 C 源程序的编码	(124)
2.3.2 源文件的编译	(124)
2.3.3 目标文件的连接	(125)
2.4 启动 Code View 调试器	(125)
2.5 Code View 选项的使用	(127)
2.5.1 启用黑白屏幕显示	(128)
2.5.2 给定启动命令	(128)
2.5.3 设置屏幕切换方式	(129)
2.5.4 启动窗口或顺序方式	(130)
2.5.5 关闭鼠标	(131)
2.5.6 启用增强型图形适配器的43行显示方式	(131)
2.6 与宏汇编器一起使用调试器	(132)
第三章 CodeView 的显示	(133)
3.1 引言	(133)
3.2 启用窗口方式	(133)
3.2.1 通过键盘执行窗口命令	(134)
3.2.1.1 用键盘命令移动光标	(134)
3.2.1.2 用键盘命令改变屏幕	(135)
3.2.1.3 用键盘命令控制程序执行	(135)
3.2.1.4 用键盘从菜单中选择	(136)
3.2.2 用鼠标执行窗口命令	(137)

3.2.2.1	用鼠标改变屏幕	(137)
3.2.2.2	用鼠标控制程序执行	(138)
3.2.2.3	用鼠标从菜单中选择	(138)
3.2.3	使用菜单选择项	(139)
3.2.3.1	使用 File 菜单	(139)
3.2.3.2	使用 Search 菜单	(140)
3.2.3.3	使用 View 菜单	(141)
3.2.3.4	使用 Run 菜单	(142)
3.2.3.5	使用 Watch 菜单	(143)
3.2.3.6	使用 Options 菜单	(144)
3.2.3.7	使用 Calls 菜单	(146)
3.2.4	使用 Help 系统	(146)
3.3	使用顺序方式	(147)
第四章	使用会话命令	(149)
4.1	引言	(149)
4.2	键入命令和参数	(149)
4.2.1	使用特殊键	(149)
4.2.2	使用命令缓冲区	(349)
4.3	Code View 的命令及参数的格式	(150)
4.4	C 表达式	(150)
4.4.1	标识符	(152)
4.4.2	常数	(152)
4.4.3	寄存器	(153)
4.4.4	地址	(153)
4.4.5	地址域	(154)
4.4.6	行号	(15)
4.4.7	串	(155)
第五章	运行代码	(156)
5.1	引言	(156)
5.2	Trace (追踪) 命令	(156)
5.3	Program Step (程序步) 命令	(158)
5.4	Go (走) 命令	(160)
5.5	Execute (执行) 命令	(161)
5.6	Restart (重启) 命令	(162)
第六章	检查数据和表达式	(163)
6.1	引言	(163)

6.2	表达式显示命令	(163)
6.3	符号检查命令	(167)
6.4	内存显示命令	(169)
6.4.1	内存显示	(170)
6.4.2	内存字节显示	(171)
6.4.3	内存 ASCII 字符显示	(171)
6.4.4	内存整数显示	(172)
6.4.5	内存无符号整数显示	(172)
6.4.6	内存字显示	(172)
6.4.7	内存双字显示	(173)
6.4.8	内存短实数显示	(173)
6.4.9	内存长实数显示	(174)
6.4.10	内存10字节实数显示	(174)
6.5	寄存器命令	(174)
6.6	8087命令	(176)
第七章	断点管理	(178)
7.1	引言	(178)
7.2	Breakpoint Set (断点设置) 命令	(178)
7.3	Breakpoint Clear (断点清除) 命令	(180)
7.4	Breakpoint Disable (断点屏蔽) 命令	(181)
7.5	Breakpoint Enable (断点启动) 命令	(181)
7.6	Breakpoint List (断点列表) 命令	(182)
第八章	Watch (观察) 语句的管理	(184)
8.1	引言	(184)
8.2	设置表达式观察和内存观察语句	(184)
8.3	设置观察点	(187)
8.4	设置追踪点	(189)
8.5	删除观察语句	(191)
8.6	列出观察点和追踪点	(192)
第九章	检查代码	(194)
9.1	引言	(194)
9.2	Set Mode (方式设置) 命令	(194)
9.3	Unassemble (逆汇编) 命令	(195)
9.4	View (查看) 命令	(197)
9.5	Current Location (当前定位) 命令	(199)
9.6	Stack Trace (栈追踪) 命令	(199)

第十章 代码和数据的修改	(202)
10.1 引言	(202)
10.2 汇编命令	(202)
10.3 键入命令	(204)
10.3.1 键入命令	(206)
10.3.2 键入字节命令	(207)
10.3.3 键入 ASCII 字符命令	(207)
10.3.4 键入整数命令	(207)
10.3.5 键入无符号整数命令	(208)
10.3.6 键入字命令	(208)
10.3.7 键入双字命令	(209)
10.3.8 键入短实数命令	(209)
10.3.9 键入长实数命令	(209)
10.3.10 键入10字节实数命令	(210)
10.4 寄存器命令	(210)
第十一章 系统控制命令的使用	(213)
11.1 引言	(213)
11.2 帮助命令	(213)
11.3 退出命令	(214)
11.4 置基数命令	(214)
11.5 重新画屏命令	(215)
11.6 屏幕转换命令	(216)
11.7 查寻命令	(216)
11.8 Shell 调用 (Shell Escape) 命令	(218)
11.9 置制表符命令	(219)
11.10 重定向命令	(220)
11.10.1 Code View 输入重定向	(220)
11.10.2 Code View 输出重定向	(221)
11.10.3 Code View 输入输出重定向	(221)
11.10.4 与重定向有关的命令	(222)
11.10.4.1 注解命令	(222)
11.10.4.2 延迟命令	(223)
11.10.4.3 暂停命令	(223)

Code View 附录

A 命令和方式小结	(225)
A.1 引言	(225)

A.2	方式	(225)
A.3	选项	(226)
A.4	窗口命令	(226)
A.5	会话命令	(229)
A.6	类型描述符	(231)
B	正规表达式	(232)
B.1	引言	(232)
B.2	正规表达式中的特殊字符	(232)
B.3	特殊字符的查寻	(232)
B.4	使用句号	(233)
B.5	使用方括号	(233)
B.5.1	使用方括号中的减号	(233)
B.5.2	使用方括号中的箭头符号	(233)
B.5.3	匹配方括号中的括号	(234)
B.6	使用星号	(234)
B.7	匹配一行的头或尾	(234)
C	错误信息	(235)
D	词汇表	(239)

第三篇 维护工具

第一章 用LIB管理库程序

1.1	概述	(247)
1.2	LIB操作的综述	(247)
1.3	运行LIB	(248)
1.4	LIB的任务	(252)

第二章 用MAKE维护程序

2.1	概述	(256)
2.2	使用MAKE	(256)
2.3	维护程序的一个例子	(261)

第一章 在DOS系统下建立C编译系统

1.1 概述

本章解释如何装配编译软件并为编译器建立运行环境。同时，还描述了组成编译软件包的诸文件以及建议文件的组织方法。

本章提到了MS-DOS的几个程序。特别是MS-DOS的SET和PATH命令，它们被用来给“环境变量”置值，这些环境变量控制着编译环境。如果你不熟悉SET和PATH命令或是本章提到的其它的MS-DOS程序，请查阅你的操作系统手册。

本章包括一个磁盘建立实例（建立所需的文件）和一段实际会话范例来引导用户熟悉用Microsoft C编译器和覆盖连接器（LINK）编译和连接程序的过程。该实际会话段使用户能够验证自己的文件是否已适当地建立起来；尽管不是非要进行这种验证。本章还提供了MSC和LINK命令的要点概述。

为了使编译器运行起来，我们提出以下步骤：

- 1) 将磁盘进行后备复制
- 2) 检查磁盘内容。
- 3) 阅读README.DOC文件以便了解本手册印刷以后编译程发生的变化和增加的内容。
- 4) 采用适合于你们系统的“快速建立过程”（硬盘或软盘快速建立过程）来建立目录和从系统盘复制文件。

1.2 后备磁盘

在得到你的C系统盘后，应该做的第一件事情就是使用MS-DOS的COPY命令或DISKCOPY程序系统盘进行文件复制，保留原来的磁盘做为后备。

1.3 磁盘内容

当你首次打开编译软件包时，用户大概想证实自己有完整的一套软件。应该在磁盘上找到以下文件：

可执行文件：

文件名	描述
MSC.EXE	编译器的控制程序
C1.EXE	预处理及语法分析器
C2.EXE	代码生成器

C3.EXE	优化器, 连接代码发生器, 及汇编列表生成器
LINK.EXE	Microsoft 覆盖连接器
LIB.EXE	Microsoft 库管理器
EXEPACK.EXE	Microsoft 压缩EXE文件实用程序
EXEMOD.EXE	Microsoft .EXE 文件头标实用程序
SETENV.EXE	Microsoft扩展环境实用程序
CV.EXE	Microsoft Code View 调试器
MAKE.EXE	Microsoft 程序维护实用程序
CL.EXE	编译器的另外一个控制程序

嵌入文件:

文 件 名	描 述
ASSERT.H	断言宏定义
CONIO.H	控制台 I/O 函数说明
CTYPE.H	字符分类宏定义
DIRECT.H	目录控制函数说明
DOS.H	MS-DOS接口函数的数据类型定义及宏定义, MS-DOS 接口函数说明
ERRNO.H	整个系统的出错编号定义
FCNTL.H	open 函数中标志定义
FLOAT.H	用于浮点操作中的值的定义
IO.H	作用于文件号的函数说明 (低级函数)
LIMITS.H	各种数类的上下界限定义
MALLOC.H	内存分配函数的说明
MATH.H	数字函数及相关常数的定义
MEMORY.H	缓冲区操作函数说明
PROCESS.H	进程控制函数说明及spawn 函数中标志定义
SEARCH.H	查寻及排序函数说明
SETJMP.H	setjmp及longjmp函数说明和内存工作区的设置
SHARE.H	共享文件标志定义
SIGNAL.H	signal 函数说明及相应常数的定义
STDARG.H	处理可变数目参数表的宏定义 (依ANSI C标准草稿)
STDDEF.H	标准值 (如NULL和errno) 的定义
STDIO.H	流函数说明及相关的宏定义、常数定义和类型定义
STDLIB.H	所有未在其他嵌入文件中说明的C库函数说明
STRING.H	串操作函数说明
TIME.H	时间函数说明及时间函数所用到的结构类型定义
VARARGS.H	处理可变数目参数表的宏定义 (与STDARG.H相似, 与XENIX兼容)
V2TOV3.H	为帮助 Microsoft C 2.03版及更早版本的程序转换所使用的宏定义
SYS/LOCKING.H	文件加锁标志定义

SYS/STAT.H	stat和fstat函数说明, 和stat结构类型及相关常数的定义
SYS/TIMEB.H	ftime函数说明及timeb结构类型定义
SYS/TYPES.H	用于文件状态及时间信息的类型定义
SYS/UTIME.H	utime函数说明及utimbuf结构类型定义

库文件:

文 件 名	描 述
SLIBC.LIB	小模型标准C库
SLIBFP.LIB	小模型浮点数学库
SLIBFA.LIB	小模型补充数学库
MLIBC.LIB	中模型标准C库
MLIBFP.LIB	中模型浮点数学库
MLIBFA.LIB	中模型补充数学库
CLIBC.LIB	紧凑型标准C库
CLIBFP.LIB	紧凑型浮点数学库
CLIBFA.LIB	紧凑型补充数学库
LIBH.LIB	与模型不相关的帮助库
LLIBC.LIB	大模型标准C库
LLIBFP.LIB	大模型浮点数学库
LLIBFA.LIB	大模型补充数学库
EM.LIB	与模型不相关的浮点仿真库
87.LIB	与模型不相关的8087/80287浮点

其他文件:

文 件 名	描 述
BINMODE.OBJ	处理二进制数据子程序
SSETARGV.OBJ	处理万能匹配符的小模型子程序
MSETARGV.OBJ	处理万能匹配符的中模型子程序
CSETARGV.OBJ	处理万能匹配符的紧凑型子程序
LSETARGV.OBJ	处理万能匹配符的大模型子程序
SVARSTCK.OBJ	允许对未使用的栈空间进行动态堆分配的小模型子程序
CVARSTCK.OBJ	允许对未使用的栈空间进行动态堆分配的紧凑型子程序
MVARSTCK.OBJ	允许对未使用的栈空间进行动态堆分配的中模型子程序
LVARSTCK.OBJ	允许对未使用的栈空间进行动态堆分配的大模型子程序
EMOEM.ASM	浮点软件特定化的模块
CV.HLP	Code View 调试器的帮助文件
DEMO.C	C演示程序
README.DOC	本套手册中尚未包括的修改及附加的说明。如果你在盘上发现了未在上述表中出现的文件, 则它们将在README.DOC文件中解释。你的那套

软件中可能没有README.DOC 文件，所以在盘上没有找到此文件也不必惊慌。

Start-up sources 一组汇编程序子程序及嵌套文件构成的C程序启动代码。在README.DOC文件中列出。

1.4 快速建立硬盘运行环境的方法

以下环境建立实例适用于硬盘系统。该建立实例仅包含小存储型库文件。如果用户的所有程序都是小存储型或者根本不关心存储模型，则只需要小存储型库文件。但是，如果在程序设计中使⽤不止一个存储模型，那么可能要从4号盘“程序库盘（中型和紧凑型存储）”或是5号盘“程序库盘（大存储型）”中选择合适的库文件添加到LIB目录中去。

8087/80287浮点程序库和补充数学库没有出现在建立实例中，因为用户不会同时需要正规浮点库和其它浮点库。如果你想使⽤一种其它的浮点库，可以将这个库替换到LIB目录或添加到LIB目录。类似地，此建立实例中仅使⽤MSC.EXE控制程序。如果你想使⽤CL.EXE控制程序就要把它加到BIN目录中去，或是用它替换MSC.EXE。

注意，以下过程中假定你的硬盘是驱动器C，你在C盘开始工作：C作为你的当前目录和驱动器。

1) 启动你的系统，这时MS-DOS提示符显示出来。输入以下命令（这些命令用来设置环境变量，这样编译器就可以在第2步创建的目录中查找所需的可执行文件、程序库和嵌入文件）：

```
PATH C:\BIN
SET INCLUDE=C:\INCLUDE
SET LIB=C:\LIB
SET TMP=C:\TMP
```

注意，TMP设置指定驱动器C的TMP目录。由编译器产生的临时文件在编译处理完成之前就被删除了。MSC.EXE自动地删除临时文件，用户不负责删除它们。每次启动MS-DOS后，都要输入以上命令才能运行编译器。为了节省输入这些命令所花费的时间，你可以将这些命令放在一个批命令文件中（例如：SETC.BAT）而通过键入此文件的⺞字来设置环境变量（见“批命令文件的使用”）。

2) 紧跟着MS-DOS提示符按顺序输入以下命令（这些命令用来创建目录，你将在此目录中存储编译器文件、程序库和嵌入文件；如果你的硬盘上已经有名为BIN、LIB、TMP、INCLUDE或是INCLUDE\SYS的目录，就应该跳过创建这些目录的命令）：

```
CD \
MD BIN
MD LIB
MD INCLUDE
MD INCLUDE\SYS
MD TMP
```

3) 在驱动器A中插入1号盘“C编译器”，然后在MS-DOS提示符处输入以下命令：

```
COPY A:*.* \BIN
```

4) 在驱动器A中用2号盘“实用程序”替换1号盘,然后,在MS-DOS提示符后输入下列命令:

```
COPY A: *.EXE\BIN
```

5) 用3号盘替换2号盘,3号盘为“嵌入文件和库(小存储模型)”,然后,紧跟着MS-DOS提示符输入以下命令:

```
COPY A: LINK.EXE\BIN
```

6) 输入命令:

```
CD\BIN
```

```
DIR
```

查看下列文件是否都在你的BIN目录下:

EXEMOD.EXE	NSC.EXE
EXEPACK.EXE	C1.EXE
CV.EXE	C2.EXE
LIB.EXE	C3.EXE
LINK.EXE	SETENV.EXE
MAKE.EXE	

7) 3号盘仍留在驱动器A中,紧跟着MS-DOS提示符,输入这些命令:

```
COPY A: *.H\INCLUDE
```

```
COPY A:\SYS\*.H\INCLUDE\SYS
```

8) 在MS-DOS提示符后面输入命令:

```
CD \INCLUDE
```

```
DIR
```

来证实下列文件已被拷到INCLUDE目录:

ASSERT.H	FLOAT.H	SEARCH.H	STDLIB.H
CONIO.H	IO.H	SETJMP.H	STRING.H
CTYPE.H	LIMITS.H	SHARE.H	TIME.H
DIRECT.H	MALLOC.H	SIGNAL.H	V2TOV3.H
DOS.H	MATH.H	STDARG.H	VARARGS.H
ERRNO.H	MEMORY.H	STDDEF.H	MODULARC.H
FCNTL.H	PROCESS.H	STDIO.H	GRAPHICS.H

9) 其次,在MS-DOS提示符后输入这两条命令:

```
CD SYS
```

```
DIR
```

来证实这些嵌入文件已被拷到INCLUDE\SYS目录下:

```
LOCKING.H  
STAT.H  
TIMEB.H  
TYPES.H  
UTIME.H
```

10) 仍然使用驱动器 A 中的 3 号盘, 在提示符处输入以下命令:

```
COPY A:SLIBC.LIB \LIB
COPY A:SLIBEP.LIB \LIB
COPY A:EM.LIB \LIB
COPY A:LIBH.LIB \LIB
```

11) 输入:

```
CD \LIB
DIR
```

证实一下前一步中复制的四个文件已在 LIB 目录中。

按照以上建立实例的做法去做, 你可以在任何目录或盘中运行编译器和连接器 (事实上, 你可以运行刚才复制的任何 .EXE 文件)。

如果你的程序使用下列目标之一, 可以把这些文件放在你的 C 程序文件所在目录或放在 LIB 目录中。

文 件	用 途
xSETARGV.OBJ	允许使用万能匹配字符 (wild-card)
xVARSTCK.OBJ	允许使用栈/堆竞争 (x 为 S, C, M 或 L)
BINMODE.OBJ	改变缺省的正文处理方式

注意: 不用 LIB 环境变量来查找 xSETARGV 或 BINMODE 文件; 如果这些文件不在你的当前工作目录下, 则你必须在连接时给定一个路径名。

1.5 快速建立软盘运行环境的方法

要在软盘上运行本编译软件, 你需要至少额外的三张软盘。以下给出的环境建立实例总是同时使用两个盘并且假定了如下条件:

- 当必要时, 你将在驱动器 A 交换你将命名为“编译器”和“连接/实用程序/库”的两张盘。

- 你将在驱动器 B 中用第三张单独的盘上 (你命名为“嵌入/源文件”), 开发自己的程序和建立列表文件。

- 你将在驱动器 B 中运行编译器, 这样, 对于输出文件 (目标文件、列表文件、映射文件和可执行文件), B 就是缺省的驱动器。

本建立实例仅包含小存储型库文件。你可以在一个盘上仅保留一套库文件, 以便节省空间, 因为任何给定的程序只使用一套库程序 (小型的、中型的、紧凑型的或大型的)。如果你的所有程序都是小型的, 或者你不用存储型, 那么你只需要小存储型的库文件。

以下范例不使用 8087/80287 浮点程序库和补充数学库, 因为你不会同时需要正规浮点程序库和其它浮点程序库。如果你想使用其它的浮点程序库, 你可以替换上该程序库。类似地, 以下范例仅使用 MSC.EXE 控制程序, 如果你想使用 CL.EXE 而不是 MSC.EXE, 则可以用它替换 MSC.EXE。

要实现以下建立过程, 每个盘驱动器必须具有 360K 容量。

1) 启动你的系统, 这时, MS-DOS提示符显示出来, 输入以下命令 (这些命令将驱动器 A 设置为当前驱动器, 然后设置环境变量, 以便编译器在下一步将要创建的目录中查找所需要的可执行文件、程序库和嵌入文件):

```
A:
PATH A:\,A:\BIN
SET INCLUDE = B:\INCLUDE
SET LIB = A:\LIB
SET TMP = B:\
```

注意: TMP设置仅简单地指定驱动器 B 的根目录。由编译器产生的临时文件在编译处理完成前就删除了, 因此你不需要建立一个单独的目录去存放它们 (MSC.EXE 自动地删除临时文件, 用户不负责删除它们)。

每次启动 MS-DOS后, 都要输入以上命令才能运行编译器。为了节省输入这些命令所花费的时间, 你可以将这些命令放在一个批命令文件中 (例如: SETC.BAT), 并且通过输入该文件的名字来设置环境变量 (见2.14节) “批命令文件的使用”)。

2) 在驱动器 B 中插入 1 号盘 “C 编译器” 并在驱动器 A 中插入一新格式化的软盘

3) 紧跟着MS-DOS提示符, 输入下列命令:

```
COPY B:*,*
```

4) 输入命令

```
DIR
```

来证实下列文件已复制到驱动器 A 中你的软盘上:

```
MSC.EXE
```

```
C1.EXE
```

```
C2.EXE
```

```
C3.EXE
```

5) 将驱动器 A 中的盘取出, 写上标记 “编译器” 并且将另外一张新格式化的软盘插入驱动器 A。

6) 在驱动器 B 中用 2 号盘 “实用程序” 换下 1 号盘。

7) 在MS-DOS命令之后, 按顺序输入以下命令:

```
MD BIN
```

```
CD BIN
```

```
COPY B;LIB.EXE
```

```
COPY B;MAKE.EXE
```

```
COPY B;EXEPACK.EXE
```

```
COPY B;EXEMOD.EXE
```

```
COPY B;SETENV.EXE
```

8) 在驱动器 B 中用 3 号盘 “嵌入文件和库 (小存储模型)” 换下 2 号盘, 然后, 按顺序输入这些命令:

```
COPY B;LINK.EXE
```

```
CD \
```

```
MD LIB
CD LIB
COPY B:\SLIBC.LIB
COPY B:\SLIBFP.LIB
COPY B\EM.LIB
COPY B\LIBH.LIB
```

输入命令

```
DIR
```

来证实下列库文件已经被复制到驱动器 A 中的 LIB 目录中:

```
SLIBC.LIB
SLIBFP.LIB
EM.LIB
LIBH.LIB
```

然后, 输入:

```
CD\BIN
DIR
```

来证实以下程序已经被复制到 BIN 目录:

```
LINK.EXE           EXEPACK.EXF
MAKE.EXE           EXEMOD.EXE
LIB.EXE            SETENV.EXE
```

9) 从驱动器 A 中取出软盘, 将它标上“(连接/实用程序/库)”并在驱动器 A 中换上另一张格式化的软盘。

10) 按照所示顺序, 在 MS-DOS 提示符后面键入以下命令:

```
MD INCLUDE
CD INCLUDE
COPY B\INCLUDE\*.H
```

11) 输入命令

```
DIR
```

来证实一下以下文件已经被复制到你的 INCLUDE 目录中去了:

```
ASSERT.H    FLOAT.H    SEARCH.H    STDLIB.H
CONIO.H     IO.H       SETJMP.H    STRING.H
CTYPE.H     LIMITS.H   SHARE.H     TIME.H
DIRECT.H    MALLOC.H    SIGNAL.H    V2TOV3.H
DOS.H       MATH.H      STDARG.H    VARARGS.H
ERRNO.H     MEMORY.H    STDDEF.H    MODULARC.H
FCNTL.H     PROCESS.H   STDIO.H     GRAPHICS.H
```

12) 输入命令

```
MD SYS
```

以在 INCLUDE 目录中创建 SYS 子目录。

13) 按所示顺序输入以下命令:

```
CD SYS
```

```
COPY B: \INCLUDE\SYS\*.H
```

14) 键入命令:

```
DIR
```

来证实以下文件已经复制到SYS目录:

```
LOCKING.H
```

```
STAT.H
```

```
TIMEB.H
```

```
TYPES.H
```

```
UTIME.H
```

15) 从驱动器 A 中取出软盘, 并将它标上“嵌入文件/源文件”。

如果你使用xSETARGV.OBJ、xVARSTCK.OBJ或 BINMODE.OBJ 文件中的某一个文件(所有这些文件都在“快速建立硬盘运行环境的方法”中加以描述), 那么你可以把这个文件放在你的 C 程序文件所在的目录或是 LIB 目录中。

注意, LIB环境变量没有用于查找xSETARGV或BINMODE文件, 当这个文件不在你的当前目录下时, 你必须在连接时指定一个路径名。

如果你的程序设计中使用不止一个存储模型, 那么可能要为每种存储型建立一个单独的程序库盘。注意, 存储在你的“编译器”盘和“嵌入文件/源文件”盘上的文件(编译器的扫描程序和嵌入文件)不会受存储型式的影响而改变, 这样, 你就可以在编译阶段对于所有五种存储型式使用同样的盘片。

在每一个单独的程序库盘上, 你将存有对于那个特定存储模型的库文件, 加上 LINK 程序和LIB程序的复本, 以及任何你要使用的实用程序。尽管LINK和LIB程序不会由于存储模型不同而改变, 但是, 在每个程序库盘上保留他们的复本是方便的, 这样你可以在调用LINK和LIB时不需要一定用到你的“小存储型盘”。

最好在所有四个磁盘(小型、中型、紧凑型 and 大型库盘)上使用同样的目录结构, 这样, 当你改换磁盘时就不必去改变环境变量的值了。例如, 处理一个中存储型的程序, 该程序使用补充数学库而不是仿真程序, 你可以按以下建立一张盘, 将它放在驱动器 A 中使用:

```
BIN\LINK.EXE
```

```
BIN\LIB.EXE
```

```
LIB\MLIBC.LIB
```

```
LIB\MLIBEA.LIB
```

这种文件组织方式除了用中存储型标准库文件代替了小存储型文件和用中存储型补充数学库(MLIBFA.LIB)代替了EM.LIB与SLIBFP.LIB以外, 与前面“连接/实用程序/库”的建立是相同的。以上使用的PATH设置(A:\BIN)和TMP设置(B:\)对于这个盘也是适用的, 因为这个盘与前面建立的小存储型盘具有同样的目录结构。注意, 当你将小存储型盘改换为中存储型盘时, 必须使用同样的盘驱动器(驱动器 A), 否则, 你的环境设置就成为非法了。