MS-DOS 5.0 内核剖析

李凤华　周利华　赵丽松　编著

（中 册）

西安电子科技大学出版社

MS—DOS 5.0 内核剖析

（中册）

李凤华　周利华　赵丽松　编著

西安电子科技大学出版社
1992

(陕)新登字 010 号

内 容 简 介

作者对 DOS 操作系统的最新版本 MS－DOS 5.0 的核心程序进行了完整、详细的分析，同时结合从事 DOS 操作系统开发和改造的实践经验，全面地介绍了 DOS 操作系统的设计思想及 MS－DOS 5.0 的具体实现，为读者提供了 MS－DOS 5.0 内核的完整信息。

《MS－DOS 5.0 内核剖析》分上、中、下三册出版，全书分为十一个章节。它主要介绍了 DOS 操作系统的结构和设计思想、配置系统、引导过程、设备管理、内存管理、文件系统、进程管理、DOS 功能调用、DOS 核心文件的编程环境，并提供了 MS－DOS 5.0 的 IO.SYS 和 MSDOS.SYS 两个系统文件的源程序注释清单。第八章给出了包括内部保留(未公开)功能调用在内的所有 DOS 功能调用的详细信息，同时为了方便读者阅读 IO.SYS 和 MSDOS.SYS 源程序，书中还提供了这两个核心文件的源程序索引。

本书适合于计算机系统软件开发人员、微机开发和应用人员参考，也可作为大专院校"操作系统"、"系统程序设计"、"计算机外设联机技术"等有关课程的教学参考书，本书也可供从事微机的加/解密和计算机病毒防治等方面的技术人员参考。

MS－DOS 5.0 内核剖析(中册)
李凤华　周利华　赵丽松　编著
责任编辑　殷咸安

西安电子科技大学出版社出版发行
西安电子科技大学印刷厂印刷
新华书店经销
开本 787×1092　1/16　印张 27 4/16　字数 834 千字
1992 年 10 月第 1 版　1992 年 10 月第 1 次印刷　印数 1—4 000

ISBN7－5606－0213－4/TP・0076　　定价：80.00 元

目　录

第十章　IO.SYS 源程序

本章详细地介绍 MS－DOS 5.0 的 IO.SYS 系统文件的各个组成部分的内容和功能，并给出了 IO.SYS 源程序注释清单。这些资料是读者深入理解和掌握 MS－DOS 5.0 核心所需要的。

IO.SYS 系统文件的主要功能是进行 DOS 的系统引导和初始化、处理 CONFIG.SYS 文件和配置系统、提供系统基本配置的设备的设备驱动程序，它是 MSDOS.SYS 系统文件的基础。

§10.1　IO.SYS 的编程基础

IO.SYS 系统文件是建立在 ROM BIOS 的基础之上，它的运行环境是以下几个方面：

(1)硬盘分区信息表；

(2)引导记录提供的 BPB 参数块(对于 MS－DOS 5.0，引导记录提供扩充 BPB 参数块)；

(3)由 ROM BIOS 提供的软盘参数表和硬盘参数表；

(4)由 ROM BIOS 提供的中断服务程序——INT 10H～INT 1AH。

§10.2　IO.SYS 的组成

IO.SYS 系统文件由 Loader 模块、IO1 模块、IO2 模块和 IO3 模块组成(如表 10.1 所示)，有关这四个模块的各自作用将在下一节详细介绍。IO.SYS 系统文件的这种结构是根据如下思想编程的：

表 10.1

模块名	长度(字节数)	相对起始段内偏移	在源程序中的位置(行号)
Loader	05A6H	0000H	1～518
IO1	2570H	0000H	519～3817
IO2	1A60H	0030H	3818～7889
IO3	3D20H	0000H	7890～15417
IO.SYS	33430(字节)		

(1)Loader 是 IO.SYS 系统文件的装入程序；

(2)IO1 模块由驻留在常规内存低端的接口程序和数据区、以及系统初始化程序 SysInt－I 组成；

(3)IO2 模块是设备驱动程序集，它允许安装在常规内存(如图 5.7)或安装在 HMA(如图 5.9)；

(4)IO3 模块负责处理系统配置文件 CONFIG.SYS。

§10.3 IO.SYS 的各个模块的功能

10.3.1 Loader 模块

Loader 模块长 05A6H 字节，它由引导记录读入并存放在常规内存 0000:0070H 处。有关引导盘的信息(如引导盘的物理驱动器号、介质描述符和数据区的起始扇区号、IO.SYS 和 MSDOS.SYS 系统文件的目录项等)是由引导记录提供的。当 CPU 的执行控制权由引导记录转到 Loader 模块时，Loader 模块负责装入 IO.SYS 系统文件的其它三个模块——IO1 模块、IO2 模块和 IO3 模块，装入的过程如图 10.1 所示。

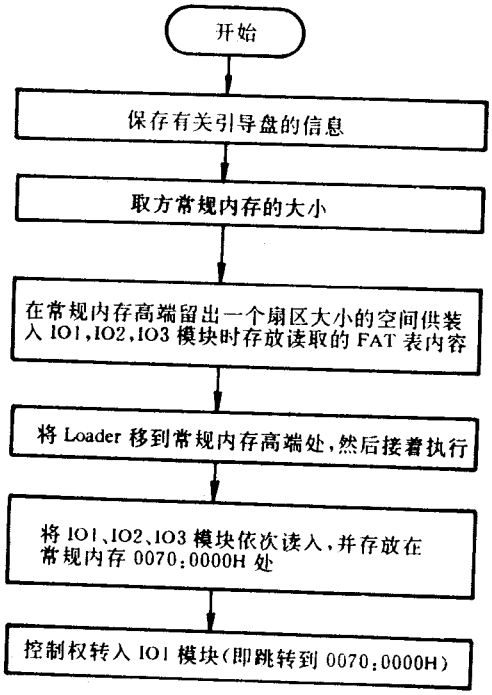

图 10.1 Loader 模块的工作过程

Loader 模块的作用如下：

(1)通过 Loader 模块安装 IO.SYS 系统文件的主体程序，使得 IO.SYS 的文件长度对引导记录透明，从而使得 MS—DOS 5.0 的引导记录能在以后的版本中保持不变；

(2)MS—DOS 5.0 允许 IO.SYS 和 MSDOS.SYS 系统文件的内容存于引导盘数据区的任意处，而不是引导盘数据区的起始位置。MS—DOS 5.0 只要求根目录的头两个目录项分别对应于 IO.SYS 和 MSDOS.SYS 即可。Loader 模块使得安装 MS—DOS 5.0 更方便。

MS—DOS 5.0 允许在不破坏系统盘数据的情况下安装，它的这一新特点正是由 Loader 模块来保证的。Loader 模块在 IO1 模块获得控制之后就失去了作用，并终止存在和被覆盖。

10.3.2 IO1 模块

IO1 模块长 2570H 字节，它由 Loader 读入并存放在常规内存 0070:0000H 处。由引导记录提供的有关引导盘的信息通过 Loader 模块转传给 IO1 模块。

一、IO1 模块的组成

IO1 模块由以下 7 部分组成：

(1)DOS 提供的所有设备驱动程序对应的设备驱动程序标题(如 CON、AUX、PRN、CLOCK $、块设备、COM1、LPT1、LPT2、LPT3、COM2、COM3、COM4，如图 4.1 所示。在源程序中的行号为 604～708)和调用 IO2 模块提供的设备驱动程序的接口程序(在源程序中的行号为 940～1014)；

(2)提供了 INT 1BH、INT 29H、INT 6CH 中断服务程序(在源程序中的行号分别为 931～939、1015～1034、1682～1939)；

(3)INT 13H 中断接管程序(在源程序中的行号为 1035～1071、1365～1590)；

(4)A20 地址线管理程序(在源程序中的行号为 1073～1140)；

(5)INT 19H 中断接管程序(在源程序中的行号为 1141～1191)，它是为保证系统重启动的环境而设置的(因为当 DOS 启动后，它接管或更改了 INT 10H、INT 13H、INT 15H、INT 19H、INT 1BH 中断服务程序；以外，当 CONFIG.SYS 文件中指定了 stacks 系统配置命令后，它还接管了 INT 02H、INT 08H～INT 0EH、INT 70H、INT 72H～INT 77H 中断服务程序，以便给它们提供动态堆栈)；

(6)IO1 模块的驻留部分及 IO2 模块的数据区和 BDPB 数据结构所需的内存；

(7)系统初始化程序 SysInt－Ⅰ(在源程序中的行号为 1941～3817)。

其中，1～6 项为 IO1 模块的驻留部分，但它们的驻留长度与具体的机器相关(即与微机系统配置的软硬盘数、硬盘分区数、机器型号、ROM BIOS 版本号等有关)，具体计算方法如下(参见源程序 2531～2625)。

二、SysInt－Ⅰ的功能

SysInt－Ⅰ是 DOS 操作系统的初始化程序，它具体建立了 DOS 内核(设备驱动程序和 MSDOS.SYS 系统文件的基本运行环境，分析 DOS 操作系统的读者从此开始逐步地深入将会了解 DOS 操作系统是如何启动的、以及如何管理磁盘等设备的。当 SysInt－Ⅰ执行完后，CPU 的控制权就转到 IO3 模块的系统初始化程序 SysInt－Ⅱ。SysInt－Ⅰ具体地完成以下五个方面的工作：

(1)确定并保存系统配置(包括保护和重新设置中断向量)；

(2)初始化串并端口；

(3)建立软硬盘的 BDPB 数据结构；

(4)确定 IO1 模块的驻留长度和 MSDOS1 模块在常规内存低端的最终位置；

(5)读取 MSDOS.SYS 系统文件内容到常规内存 083F:0000H 处(083FH＝0070H＋7CF0H/16，这里 7CF0H 是 IO.SYS 系统文件的 IO1、IO2 和 IO3 模块的长度和，即 MSDOS.SYS 系统文件暂时存于 IO3 模块之后)。

三、IO1 模块的驻留长度

IO1 模块的驻留长度(KL)是按如下算法计算的：

(1) 若机器配置了硬盘，则转到(3)。

(2) 若 ROM BIOS 不支持软盘的 Change Line 功能，则 KL＝08FEH，否则，KL＝0918HH。

(3) 若机器为 AT 档次机器、ROM BIOS 版本号为“01/10/84”，则 KL＝KL＋(140CH－12DEH)＝KL＋012EH；若机器为 COMPQA，ROM BIOS 是 1986 年 8 月 4 号以前开发的，则 KL＝KL＋(142DH－140CH)＝KL＋0021H。

(4) 若机器有实时时钟，则 KL＝KL＋(1509H－142EH)＝KL＋00DCH

(5) 若 INT 15H 中断服务程序支持 AX＝4101H 功能，则 KL＝KL＋(16D4－150A)＝KL＋01CBH。

根据上述算法，IO1 模块的驻留长度范围为 08FEH～(12DEH＋012EH＋00DCH＋01CBH)，即 08FEH～16B3H。

10.3.3 IO2 模块

IO2 模块长 1A60H 字节，它是 DOS 操作系统提供的设备驱动程序集，由 CON、PRN、AUX、CLOCK $ 和块设备等设备驱动程序组成(如表 10.2 和表 10.3a～10.3e 所示，其中表 10.3a 对应于 CON 设备驱动程序、表 10.3b 对应于 PRN 设备驱动程序、表 10.3c 对应于 AUX 设备驱动程序、表 10.3d 对应于 CLOCK $ 设备驱动程序、表 10.3e 对应于块设备驱动程序)；此外，它包含了 INT 13H 中断接管程序(在源程序中的行号为 6995～7344)和 INT 2FH 中断服务程序(在源程序中的行号为 6686～6840)。IO2 模块全部驻留内存，它的编程起始地址偏移为 0030H，这是因为当 IO2 模块安装在 HMA 起始位置时，在 8086/8088 指令体系下，访问 HMA 是通过使能 A20 地址线并利用段地址 0FFFFH＋段内偏移来实现的，因而 0FFFFH:0010H 才是 HMA 的起始地址；此外，HIMEM.SYS 等扩充内存管理程序使用 HMA 的起始 32 个字节来存放它们的安装标志和使用扩充内存的管理信息等。

表 10.2

设备名	起始行号	结束行号
CON	4002	4145
PRN	4146	4324
AUX	4325	4465
CLOCK $	4466	4638
块设备	4640	7882

表 10. 3a

命令码(H)	起始行号	结束行号
4	4002	4056
5	4057	4109
17	4130	4145
8 或 9	4110	4129

表 10. 3b

命令码(H)	起始行号	结束行号
4	4146	4156
8 和 9	4157	4195
0A	4196	4236
10	4237	4272
13	4273	4302
19	4303	4324

表 10. 3c

命令码(H)	起始行号	结束行号
4	4325	4347
5	4366	4389
7	4418	4429
8 和 9	4430	4454
0A	4390	4404

表 10. 3d

命令码(H)	起始行号	结束行号
4	4562	4617
8 和 9	4509	4561

表 10. 3e

命令码(H)		起始行号	结束行号
0		7346	7356
1		4687	4745
2		4775	4809
4、8 和 9		5186	5210
0D		5129	5143
0E		5144	5160
0F		5161	5173
13 子功能码(H)	40	5837	5927
	41	6073	6171
	42	5928	6028
	46	6487	6531
	47	6597	6612
	60	5797	5836
	61	6073	6171
	62	6029	6072
	66	6457	6486
	67	6580	6596
	68	6643	6684
17 和 18		6391	6427
19		6613	6642

CON 设备驱动程序是标准的显示器和键盘的驱动程序，所有键盘输入和以电传方式显示服务都是由它提供的。但是显示器和键盘是两个独立的设备，应用程序(包括系统程序)是由 CON 设备驱动程序从键盘上输入一个字符，然后将该输入字符输出到 CON 设备驱动程序，显示在显示器上。用户所见到的"输入并回显"就是分上述两步完成的。

PRN 设备驱动程序是标准并行打印机设备驱动程序，它在程序控制下发送单字节数据，不支持中断驱动式的打印。设备名 LPT1 等于 PRN，它们都对应于并行端口 1；LPT2 和 LPT3 分别对应于并行端口 2 和 3。

AUX 设备驱动程序是标准异步通讯(即串行通讯)设备驱动程序，它只能在程序控制下发送或接收字节数据，也不支持中断驱动式的数据传输和 XON/XOFF 这样的协议。设备名 COM1 等于 AUX，它们都对应于串行端口 1；COM2、COM3 和 COM4 分别对应于串行端口 2、3 和 4。

CLOCK $ 设备驱动程序提供了标准时间服务，DOS 内核通过它取出或设置当前日期或时间。

块设备驱动程序提供了磁盘服务，用户通过逻辑驱动器符来访问具体的磁盘介质。

10.3.4 IO3 模块

IO3 模块长 3D20H 字节，它的主要内容是系统初始化程序 SysInt－Ⅱ。SysInt－Ⅱ负责处理 CONFIG.SYS 系统配置文件，为 DOS 设置各种数据结构和其它工作环境。当位于 IO1 模块里的 SysInt－Ⅰ执行完后，CPU 的控制权就转到这里。

一、IO3 模块的组成

IO3 模块由以下十个部分组成：

(1)INT 02H、INT 08H～INT 0EH、INT 70H、INT 72H～INT 74H、INT 76H～INT 77H 的中断接管程序，它们为各自的中断提供动态堆栈。当 CONFIG.SYS 中有 stacks 系统配置命令，那么 SysInt－Ⅱ将驻留这部分程序(在源程序中的行号为 7943～8292)；

(2)系统初始化程序 SysInt－Ⅱ(在源程序中的行号为 8293～8819)；

(3)负责安装 IO2 模块和 MSDOS2 模块的程序(在源程序中的行号为 8820～9103)；

(4)DOS 启动时使用的 HMA 管理程序(在源程序中的行号为 9104～9263)；

(5)建立逻辑驱动器的当前目录结构(CDS)的子程序(在源程序中的行号为 9297～9375)；

(6)在作为 DOS 数据段的第一个内存块中为 DOS 内核建立 SFT 表数组、盘缓冲区、CDS 数组、动态堆栈等各种数据块的子程序(在源程序中的行号为 9376～9773)；

(7)处理 install 系统配置命令，并在 DOS 启动过程中安装 DOS 内存驻留程序的程序(在源程序中的行号为 9774～9925)；

(8)建立动态堆栈的控制块，并安装 INT 02H、INT 08H～INT 0EH、INT 70H、INT 72H～INT 74H、INT 76H～INT 77H 的中断接管程序(在源程序中的行号为 10013～10481)；

(9)取系统配置命令的参数(开关参数、整数参数、开关状态、关键字串、文件名说明串)程序，它在源程序中的行号为 10482～11752；

(10)CONFIG.SYS 允许的系统配置命令的参数说明信息和处理 CONFIG.SYS 的程序。它在源程序中的行号分别为 11753～12069、12071～15417。

二、SysInt－Ⅱ的执行过程

在 MS－DOS 5.0 中，IO2 模块、MSDOS2 模块和盘缓冲区可以安装在 HMA；在 80386 或更高档微机系统中，允许在 UMB 中安装某些设备驱动程序和其它一些应用程序；还允许使用 install 系统配置命令在 DOS 启动时安装 DOS 内存驻留程序。MS－DOS 5.0 的这些特点决定了对 CONFIG.SYS 的处理必须按一定的先后次序，CONFIG.SYS 的处理过程与先前旧的 DOS 版本有很大的差别。SysInt－Ⅱ的执行过程如图 10.2 所示，它反映了 SysInt－Ⅱ所完成的工作。

在分析 SysInt－Ⅱ程序时，应注意下列几个方面的问题：

1. 转换 CONFIG.SYS 的文件内容

完成 CONFIG.SYS 文件内容转换的程序为 14309～14537，它具体的转换方法和所完成的工作如下：

(1) 小写字符转换成大写字符；

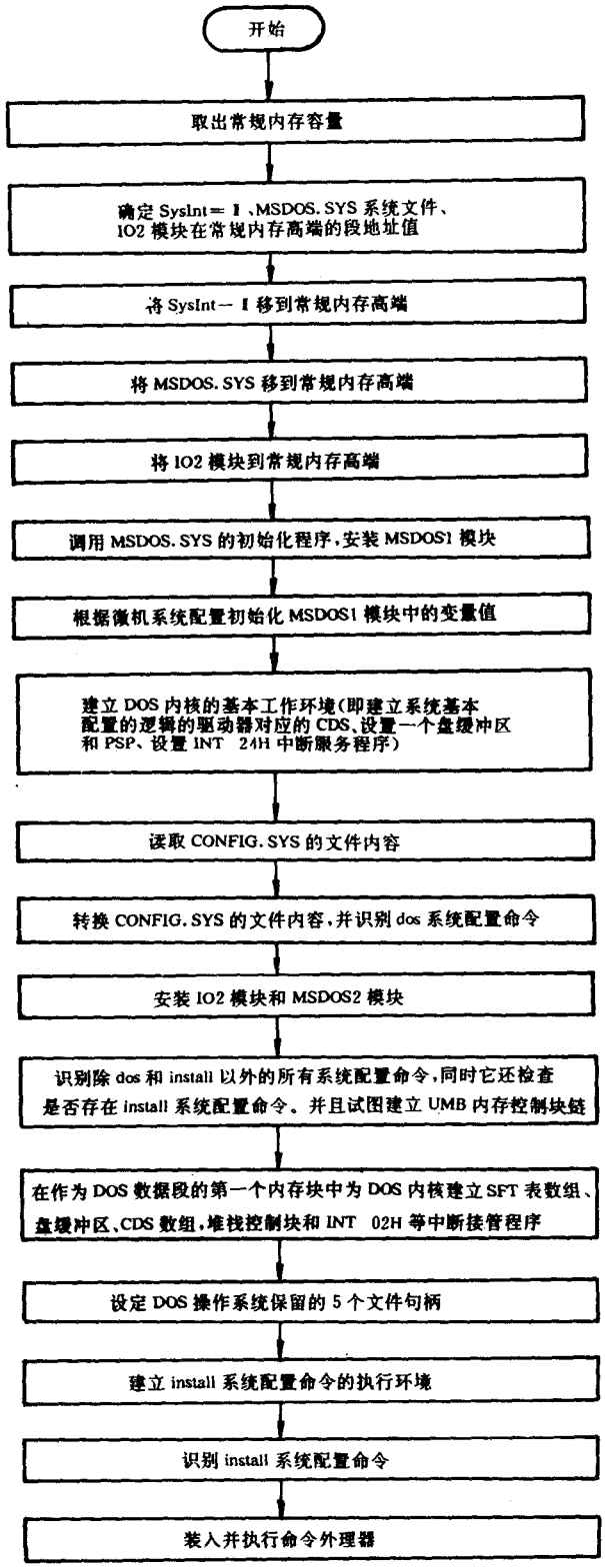

图 10.2 SysInt－Ⅱ的执行过程

(2) 丢掉每个 CONFIG.SYS 命令行的所有起始参数分隔符；

(3) 系统配置命令名用对应的缩写关键字符取代，如表 10.4 所示：

表 10.4

系统配置命令名	缩写字符	系统配置命令名	缩写字符	系统配置命令名	缩写字符
break	C	dos	H	multitrack	M
buffers	B	drivparm	P	rem	0
comment	Y	fcbs	X	shell	S
country	Q	files	F	stacks	K
device	D	install	I	switches	1
devicehigh	U	lastdrive	L		

(4) 识别 comment 系统配置命令，并丢掉在它之后的所有注释字符串；

(5) 命令行的结束符只使用换行符(0AH)，而回车符(0DH)被丢掉；

(6) 结束符(00H)代替文件名说明串后的参数分隔符；

(7) 用"Z"字符和换行符(0AH)代替整个非法的命令行；

(8) 给 devicehigh 命令行追加一个空参数项，以确保能正确地处理 devicehigh 系统配置命令的两种格式。

2. dos 系统配置命令

dos 系统配置命令的使用已在第二章中进行了介绍。由于 dos 系统配置命令是用于指定 DOS 的 IO2 模块、MSDOS2 模块和盘缓冲区的安装位置(即它们是被 SysInt－Ⅱ安装在常规内存低端或 HMA 中)；同时，它还指定设备驱动程序是否被安装在 UMB 中。鉴于 dos 系统配置命令的特殊作用，它必须首先被识别和处理。

3. 扩充内存管理程序 HIMEM.SYS

HIMEM.SYS 是扩充内存管理程序，它管理对扩充内存(包括高内存区 HMA)的使用。当 DOS 的 IO2 模块、MSDOS2 模块和盘缓冲区欲安装在 HMA 时，除 CONFIG.SYS 文件中有"dos=high"语句之外，还应有安装 HIMEME.SYS 的 device 命令(它必须安放在其它所有使用扩充内存的程序或设备驱动程序的 device 命令之前)。有关 HIMEM.SYS 的使用方法请参见第二章。

4. 高内存块管理程序 EMM386.EXE

EMM386.EXE 使得 DOS 操作系统和应用程序能够使用 80386 或更高档微机的高内存块(UMB)，同时，EMM386.EXE 还允许 SysInt－Ⅰ将设备驱动程序装入 UMB。使用 EMM386.EXE 之前必须安装 HIMEM.SYS。

EMM386.EXE 允许存取 UMB，MS－DOS 5.0 通过它可以分配和使用 UMB。有关 EMM386.EXE 的使用方法请参见第二章、第五章。

5. install 系统配置命令

DOS 操作系统的许多系统扩充(如 FASTOPEN、KEYB、NLSFUNC 和 SHARE 等)是通过 DOS 内存驻留程序提供的，它允许用户根据使用的需要有选择地过安装和使用。若用户不需要某一功能，就可以不安装对应的系统扩充命令，从而为用户留出更多的常规内存空间。

DOS 操作系统的这种功能分配使得 DOS 的核心文件尽可能地短小，使用户可以得到最佳的存贮——性能比或存贮——功能比。

在 DOS 4.0 以前，DOS 的这些系统扩充由 AUTOEXEC.BAT 或由用户通过命令行装入。在 DOS 启动时，这种处理方式使得设备驱动程序无法使用这些扩充，为了克服这个缺点，从 DOS 4.0 开始增加了 install 系统配置命令。

install 系统配置命令在 SysInt－Ⅱ执行过程中装入 DOS 内存驻留程序（如 SHARE.EXE、KEYB.COM 等），它不为装入程序生成环境块，因而它比从 AUTOEXEC.BAT 批文件或命令行下装入程序占用的内存少。

由于 install 系统配置命令装入程序时命令处理器尚未被装入运行，因而它不能装入要求 COMMAND.COM 管理临界错误的程序。另外，它也不能保证需要使用环境变量的装入程序的正常运行。

6. stacks 系统配置命令

stacks 系统配置命令允许用户指定多达 64 个动态堆栈，这些动态堆栈是提供给 INT 02H、INT 08H～INT 0EH、INT 70H、INT 72H～INT 74H、INT 76H～INT 77H 中断服务程序使用的。DOS 操作系统通过接管上述硬中断的中断服务程序方式来提供动态堆栈服务的。当上述硬中断发生时，DOS 就从用户定义的动态堆栈中分配一个堆栈，但当 stacks＝n，s 的 n 值指定为 0 时，DOS 将不接管上述硬中断对应的中断服务程序，此时，每个运行程序必须有足够的栈空间供硬中断各自对应的中断服务程序使用。

（1）stacks 系统配置命令执行后需要的内存容量

①INT 02H、INT 08H～INT 0EH、INT 70H、INT 72H～INT 74H、INT 76H～INT 77H 中断接管程序的长度为 0267H 字节；

②动态堆栈需要的内存字节数＝(8＋s)＊n，其中，8 为每个动态堆栈的控制块所需的字节数；

③堆栈对应的子段控制块的字节数＝16。

整个 stacks 系统配置命令执行后所需的内存是上述三项之和。

（2）堆栈控制块(SCB)

动态堆栈控制块(Stack Control Block，简称为 SCB)的数据结构如表 10.5 所示，它保存了堆栈的使用状态、先前的堆栈指针寄存器值和它对应的动态堆栈的指针值。

表 10.5

```
SCB                 Struc
SCB_UseMark         DB   ?      ;动态堆栈的使用标志，其值的意义如下：
                                    00：动态堆栈空闲
                                    01：动态堆栈已被分配
                                    03：动态堆栈被破坏，该动态堆栈不再被分配
                    DB   ?      ;未使用
SCB_SPValue         DW   ?      ;保存先前的 SP 寄存器值
SCB_SSValue         DW   ?      ;保存先前的 SS 寄存器值
SCB_AddrOfStack     DW   ?      ;指向对应的动态堆栈的栈顶(它的值为新的 SP 寄存器值)
SCB                 Ends
```

§10.4 IO.SYS 源程序注释清单

10.4.1 IO.SYS 源程序的编译和连接方式：

IO.SYS 源程序的编译和连接方法如下

C>MASM LOADER;↙

C>LINK LOADER;↙

C>EXE2BIN LOADER.EXE LOADER.COM;↙

C>MASM IO1;↙

C>MASM IO3;↙

C>LINK IO1+IO3;↙

C>EXE2BIN IO1.EXE IO1.COM↙

C>COPY/B LOADER.COM+IO1.COM IO.SYS↙

10.4.2 BIO.STR 的源程序清单

```
CR                          EQU     0DH
LF                          EQU     0AH
TAB                         EQU     9
SysInt_II_OFS               EQU     267H
SysInt_II_SEG               EQU     46DH
ROM_SEG                     EQU     0F000H
BIO_SEG                     EQU     70H
BDPB_Len                    EQU     64H
BPB_Len1                    EQU     19H
BPB_Len2                    EQU     1FH
DOS_0024                    EQU     24H
DOS_008C                    EQU     8CH
FileNameLen                 EQU     0BH
;
BDPB                        STRUC
  BDPB_NextPtr_OFS          DW      -1          ;0
  BDPB_NextPtr_SEG          DW      BIO_SEG     ;2
  BDPB_DriveNumber          DB      ?           ;4
  BDPB_LogicalNumber        DB      ?           ;5
;Start of Build BPB
  BDPB_BytesPerSector1      DW      512         ;6
  BDPB_SectorsPerCluster1   DB      ?           ;8
  BDPB_ReservedSectors1     DW      1           ;9
  BDPB_NumberOfFAT1         DB      2           ;B
  BDPB_TotalOfRootDir1      DW      ?           ;C
```

```
    BDPB_TotalSector1           DW      ?               ;E
    BDPB_MediaByte1             DB      ?               ;10
    BDPB_SectorsPerFAT1         DW      ?               ;11
    BDPB_SectorsPerTrack1       DW      ?               ;13
    BDPB_NumberOfHead1          DW      ?               ;15
    BDPB_HiddenSectors1         DD      0               ;17
    BDPB_BigTotalSector1        DD      0               ;1B
;End of Build BPB
    BDPB_SizeFlag               DB      0               ;1F
    BDPB_DeviceOpen             DW      0               ;20
    BDPB_DeviceType             DB      3               ;22
    BDPB_Attr_Status            DW      20H             ;23
    BDPB_NumberOfCylinder       DW      28H             ;25
;Start of Default BPB
    BDPB_BytesPerSector2        DW      512             ;27
    BDPB_SectorsPerCluster2     DB      ?               ;29
    BDPB_ReservedSectors2       DW      ?               ;2A
    BDPB_NumberOfFAT2           DB      ?               ;2C
    BDPB_TotalOfRootDir2        DW      ?               ;2D
    BDPB_TotalSector2           DW      ?               ;2F
    BDPB_MediaByte2             DB      ?               ;31
    BDPB_SectorsPerFAT2         DW      ?               ;32
    BDPB_SectorsPerTrack2       DW      ?               ;34
    BDPB_NumberOfHead2          DW      ?               ;36
    BDPB_HiddenSectors2         DD      0               ;38
    BDPB_BigTotalSector2        DD      0               ;3C
;End of Default BPB
    BDPB_ReservedFeild          DB      6 Dup(0)        ;40
    BDPB_CylinderOperated       DB      -1              ;46
    BDPB_Clock1orCylinderFlag   DW      -1              ;47
    BDPB_Clock2orCylinder       DW      -1              ;49
    BDPB_VolumeID               DB      ' NO NAME',0    ;4B
    BDPB_SerialNumber           DD      0               ;57
    BDPB_FileSystemType         DB      ' FAT12' ,0     ;5B
BDPB                            ENDS
;
BPB                             STRUC
    BPB_BytesPerSector          DW      512             ;0
    BPB_SectorsPerCluster       DB      ?               ;2
    BPB_ReservedSector          DW      ?               ;3
    BPB_NumberOfFAT             DB      ?               ;5
    BPB_TotalOfBootDir          DW      ?               ;6
    BPB_TotalSector             DW      ?               ;8
```

```
  BPB_MediaByte          DB    ?              ;A
  BPB_SectorsPerFAT      DW    ?              ;B
  BPB_SectorsPerTrack    DW    ?              ;D
  BPB_NumberOfHead       DW    ?              ;F
  BPB_HiddenSectors      DD    ?              ;11
  BPB_BigTotalSector     DD    0              ;15
BPB                      ENDS
;
FPT                      STRUC
  FPT_ControlByte1       DB    ?              ;0
  FPT_ControlByte2       DB    ?              ;1
  FPT_Motor_DelayTime    DB    ?              ;2
  FPT_SectorSize         DB    ?              ;3
  FPT_SectorsPerTrack    DB    ?              ;4
  FPT_ByteBetweenSector  DB    ?              ;5
  FPT_BytesPerSector     DB    ?              ;6
  FPT_LengthOfGapInFMT   DB    ?              ;7
  FPT_WriteByteInFMT     DB    ?              ;8
  FPT_HeadLoadTime       DB    ?              ;9
  FPT_StartupWaitTime    DB    ?              ;A
FPT                      ENDS
;
FDT                      STRUC
  FDT_FileName           DB    8  Dup(' ')    ;0
  FDT_FileType           DB    3  Dup(' ')    ;8
  FDT_FileAttribute      DB    ?              ;B
  FDT_Reserved           DB    10 Dup(0)      ;C
  FDT_TimeLastUpdated    DW    ?              ;16
  FDT_DateLastUpdated    DW    ?              ;18
  FDT_StartingCluster    DW    ?              ;1A
  FDT_FileLength         DD    0              ;1C
FDT                      ENDS
;
MCB                      STRUC
  MCB_Location           DB    ?              ;0
  MCB_ProcessID          DW    ?              ;1
  MCB_AllocationAmount   DW    ?              ;3
  MCB_Reserved           DB    3 Dup(0)       ;5
  MCB_ProgramName        DB    8 Dup(?)       ;8
MCB                      ENDS
;
DriverRequestHeader      STRUC
  DRH_Length             DB    ?              ;0
```

```
    DRH_SubUnit                 DB      ?               ;1
    DRH_Command                 DB      ?               ;2
    DRH_Status                  DW      ?               ;3
    DRH_Reserved                DB      8 dup(?)        ;5
    DRH_Media                   DB      ?               ;D
    DRH_TransAddr               DD      ?               ;E
    DRH_Count                   DW      ?               ;12
    DRH_StartingSector          DW      ?               ;14
    DRH_VolumeIDPtr             DD      ?               ;16
    DRH_BigStartingSector       DD      ?               ;1A
DriverRequestHeader             ENDS
;
StackCB                         STRUC
    SCB_UseMark                 DB      0               ;0
    SCB_Reserved                DB      0               ;1
    SCB_SPValue                 DW      0               ;2
    SCB_SSValue                 DW      0               ;4
    SCB_AddrOfStack             DW      0               ;6
StackCB                         ENDS
;
EXECParaBlock                   STRUC
    EXEC_Envioronment           DW      0               ;0
    EXEC_CMDLine_OFS            DW      ?               ;2
    EXEC_CMDLine_SEG            DW      ?               ;4
    EXEC_FCB1_OFS               DW      ?               ;6
    EXEC_FCB1_SEG               DW      ?               ;8
    EXEC_FCB2_OFS               DW      ?               ;A
    EXEC_FCB2_SEG               DW      ?               ;C
EXECParaBlock                   ENDS
```

10.4.3 Loader 模块的源程序注释清单

```
1: PAGE 60,132
2: ;Program Name: Loader.ASM
3: ;Version: MS—DOS5.00
4: ;Compiler: Macro Assembler Version 5.10
5: ;
6: ;Disk Boot Record Layout(DBRL)
7: DBRL_BytesPerSector              EQU     7C0BH   ;每扇区字节数
8: DBRL_SectorsPerCluster           EQU     7C0DH   ;每簇扇区数
9: DBRL_ReservedSectors             EQU     7C0EH   ;保留扇区数
10: DBRL_MaxNumOfRootDirEntries     EQU     7C11H   ;最大根目录登记项数
11: DBRL_TotalOfSector_16Bits       EQU     7C13H   ;总扇区数(本字段定义小于 32MB 的硬盘分
                                                    ;区或软盘。若本字段为 0 表示使用 32 位的
```

```
                                                    ;双字单元存放总扇区数)
12: DBRL_SectorsPerFAT              EQU     7C16H   ;每个 FAT 表占用的扇区数
13: DBRL_SectorsPerTrack            EQU     7C18H   ;每道扇区数
14: DBRL_NumberOfHead               EQU     7C1AH   ;磁头数
15: DBRL_HiddenSectors              EQU     7C1CH   ;隐含扇区数
16: DBRL_TotalOfSector_32Bits       EQU     7C20H   ;总扇区数(本字段定义大于 32MB 的硬盘分
                                                    ;区)
17: DBRL_ExtBootRecordMark          EQU     7C26H   ;扩充引导记录的标志
18: BIO_SEG                         EQU     70H     ;DOS 的 BIOS 模块在常规内存中的段地址值
19: Vector_1EH_OFS                  EQU     78H     ;┐
20: Vector_1EH_SEG                  EQU     7AH     ;| 中断向量的存放地址
21: Vector_2FH_OFS                  EQU     0BCH    ;|
22: Vector_2FH_SEG                  EQU     0BEH    ;┘
23: Info_51AH                       EQU     51AH    ;存放"IO.SYS 系统文件的起始簇号"的内存
                                                    ;地址
24: Info_522H                       EQU     522H    ;存放"软盘参数表"的 BIOS 通讯区的地址
25: ;=================================================================
26: ;功能:读取 IO.SYS 系统文件的实际有效程序部分(对应于 IO2.ASM、IO3.ASM)
27: ;入口参数:AX:BX=引导盘数据区的起始扇区号(由于 MS-DOS5.00 允许 IO.SYS 系统文件的内
        容存放在引导盘数据区的任意地方,因而 AX:BX 指定的扇区号不一定是 IO.SYS 文件
        的起始扇区号,这一点与 MS-DOS3.30 以下版本不同)
28: ;          CH=引导盘的介质描述符
29: ;          DL=引导盘的物理驱动器号
30: ;          DS:SI 指向系统引导前(ROM 自检时)设定的软盘参数表(即 INT 1EH 中断向量)
31: ;=================================================================
32: .286c
33: ;
34: Loader  SEGMENT  'CODE'
35:     ASSUME      CS:Loader, DS:Loader
36:     ORG         0
37: IO_Loader       PROC    FAR
38:     JMP         Loader_1
39:     DW          5
40:     DW          128     Dup(0)      ;┐ IO.SYS 系统文件的装入程序使用的堆栈
41: StackOfLoader   EQU     $           ;┘
42: NumberOfHead    DW      0           ;保存引导盘的磁头数
43: BytesPerCluster DW      0           ;保存引导盘的每簇扇区数
44: StartSectorNo   DD      0           ;存放当前欲读取的 FAT 表或 IO.SYS 系统文件内容
                                        ;对应的起始扇区号
45: TempWordVar     DW      0           ;存放磁道号/圆柱号的高 16 位值
46: ClusterNo       DW      0           ;当簇项值位于相邻两个扇区时,它存放簇项值的高、
                                        ;低字节值
47: SectorNoOfFAT   DW      -1          ;保存上次读取的 FAT 表内容对应的相对扇区号
```

```
48: ReadingSectors      DW   0          ;存放欲读取的扇区数
49: SectorsPerFAT       DW   0          ;保存引导盘上每个 FAT 表占用的扇区数
50: HiddenSectors       DD   0          ;保存引导盘的隐含扇区数
51: BytesPerSector      DW   0          ;保存引导盘的每扇区字节数
52: ReservedSectors     DW   0          ;保存引导盘的保留扇区数
53: CurrentCluster      DW   0          ;在读取 IO.SYS 系统文件时,它存放当前欲读取的文
                                        ;件内容对应的簇号
54: LoaderUsedAP        DW   0          ;在读取 IO.SYS 系统文件时,它存放读取的文件内容
                                        ;在内存中存放地址的偏移值
55: FirstDataSectorNo   DD   0          ;保存引导盘数据区的起始扇区号
56: TotalOfSector       DD   0          ;保存引导盘的总扇区数(对于硬盘,它为活动分区的
                                        ;总扇区数)
57: SectorsPerTrack     DW   0          ;保存引导盘的每道扇区数
58: BootDriveNo         DB   0          ;保存引导盘的物理驱动器号
59: TypeOfFAT           DB   0          ;存放引导盘 FAT 表的类型码(1:12 位 FAT 表;4:16
                                        ;位 FAT 表)
60: MediaDescriptor     DB   0          ;保存引导盘的介质描述符
61: EOF_IOSYS           DB   0          ;IO.SYS 系统文件的读操作标志(0:IO.SYS 系统文件
                                        ;未读完;-1:IO.SYS 系统文件内容已被全部读出)
62: Old_INT1EH_Vector   DD   0          ;保存系统引导前(ROM 自检时)设定的 INT 1EH 中断
                                        ;向量
63: AddrOfBuffer_SEG    DW   0          ;存放读取 FAT 表时使用的传送缓冲区的段地址值
64: SectorsPerCluster   DB   0          ;保存引导盘的每簇扇区数
65: Loader_1:

66:   MOV   WORD PTR CS:FirstDataSectorNo,BX    ;保存引导盘数据区的起始扇区号的低
                                                ;字值
67:   MOV   CS:MediaDescriptor,CH               ;保存引导盘的"介质描;述符"
68:   MOV   CS:BootDriveNo,DL                   ;保存引导盘的物理驱动器号
69:   MOV   WORD PTR CS:Old_INT1EH_Vector,SI    ;┐保存系统引导前设定的 INT 1EH 中
70:   PUSH  DS                                  ;│断向量
71:   POP   WORD PTR CS:Old_INT1EH_Vector+2     ;┘
72:   XOR   CX,CX                               ;┐
73:   MOV   DS,CX                               ;│ DS、ES 指向中断向量表
74:   PUSH  ES                                  ;│
75:   MOV   ES,CX                               ;┘
76:   MOV   SI,DS:Vector_1EH_OFS          ;┐ DS:SI 指向软盘参数表
77:   MOV   DS,DS:Vector_1EH_SEG          ;┘
78:   MOV   DI,Info_522H                  ;┐
79:   MOV   CX,0BH                        ;│将 ROM BIOS 设定的软盘参数表复制到
80:   CLD                                 ;│ BIOS 通讯区(0:0522H~0:52CH)中
81:   REP   MOVSB                         ;┘
82:   PUSH  ES                            ;┐ DS 指向中断向量表
```

```
83:     POP     DS                                  ;┘
84:     MOV     WORD PTR DS:Vector_1EH_OFS,Info_522H  ;┐ 设置 INT 1EH 中断向量(即
85:     MOV     WORD PTR DS:Vector_1EH_SEG,DS         ;┘ 保存软盘参数表的起始地址)
86:     POP     ES
87:     MOV     CX,DS:DBRL  BytesPerSector          ;┐ 取出并保存"每扇区字节数"
88:     MOV     CS:BytesPerSector,CX                ;┘
89:     MOV     CL,DS:DBRL  SectorsPerCluster       ;┐ 取出并保存"每簇扇区数"
90:     MOV     CS:SectorsPerCluster,CL             ;┘
91:     MOV     CX,DS:DBRL  SectorsPerTrack         ;┐ 取出并保存"每道扇区数"
92:     MOV     CS:SectorsPerTrack,CX               ;┘
93:     MOV     CX,DS:DBRL  NumberOfHead            ;┐ 取出并保存"磁头数"
94:     MOV     CS:NumberOfHead,CX                  ;┘
95:     MOV     CX,DS:DBRL  SectorsPerFAT           ;┐ 取出并保存"每个 FAT 表占用的扇区
96:     MOV     CS:SectorsPerFAT,CX                 ;┘ 数"
97:     MOV     CX,DS:DBRL  ReservedSectors         ;┐ 取出并保存"保留扇区数"
98:     MOV     CS:ReservedSectors,CX               ;┘
99:     MOV     CX,DS:DBRL  HiddenSectors           ;┐ 取出并保存"隐含扇区数"的低字值
100:    MOV     WORD PTR CS:HiddenSectors,CX        ;┘
101:    MOV     CX,DS:DBRL  TotalOfSector_16Bits    ;┐ 取出并保存"总扇区数"
102:    MOV     WORD PTR CS:TotalOfSector,CX        ;┘
103:    CMP     BYTE PTR DS:DBRL  ExtBootRecordMark,29H ;┐ 若引导记录中没有使用扩充的
104:    JNE     Loader_2                                ;┘ BPB 参数块,则跳转
105:    MOV     WORD PTR CS:FirstDataSectorNo+2,AX      ;保存引导盘数据区的起始扇区号
                                                        ;的高字值
106:    MOV     AX,DS:DBRL_HiddenSectors+2              ;┐ 取出并保存"隐含扇区数"的高
107:    MOV     WORD PTR CS:HiddenSectors+2,AX          ;┘ 字值
108:    CMP     CX,0                                ;┐ 若引导软盘或硬盘活动分区的总扇区
109:    JNE     Loader_2                            ;┘ 数用 16 位表示,则跳转
110:    MOV     AX,DS:DBRL  TotalOfSector_32Bits    ;┐
111:    MOV     WORD PTR CS:TotalOfSector,AX        ;├ 取出并保存用 32 位双字表示的总扇区
112:    MOV     AX,DS:DBRL  TotalOfSector_32Bits+2  ;├ 数
113:    MOV     WORD PTR CS:TotalOfSector+2,AX      ;┘
114: Loader_2:
115:    CLD
116:    XOR     SI,SI
117:    MOV     DI,SI
118:    INT     12H                                 ;取出系统的常规内存大小→AX(以 1KB 为单
                                                    ;位)
119:    MOV     CL,6                                ;┐ 将系统的常规内存大小变换成以节为单位
120:    SHL     AX,CL                               ;┘
121:    XOR     BX,BX                               ;┐
122:    MOV     DS,BX                               ;├ DS:BX←INT 2FH 中断服务
123:    MOV     BX,DS:Vector_2FH_OFS                ;├ 程序的入口地址
```

```
124:    MOV     DS,DS:Vector_2FH_SEG            ;┘
125:    CMP     WORD PTR [BX+3],' PR'
126:    JNE     Loader_3
127:    CMP     BYTE PTR [BX+5],' L'
128:    JNE     Loader_3
129:    MOV     DX,AX
130:    MOV     AX,4A06H
131:    INT     2FH
132:    MOV     AX,DX
133: Loader_3:
134:    MOV     CL,4                            ;┐ 将"每扇区字节数"变换 ┐ 设定读取 FAT
135:    MOV     DX,CS:BytesPerSector            ;│ 成以节为单位(DX=以   │ 表时使用的
136:    SHR     DX,CL                           ;│ 节为单位的每扇区(字  │ 传送缓冲区
137:    INC     DX                              ;┘ 节数)                ┘ 的段地址值
138:    SUB     AX,DX                           ;留出一个扇区的内存空间
139:    MOV     CS:AddrOfBuffer_SEG,AX          ;
140:    MOV     DX,offset IO_Loader_End         ;┐ 计算 IO.SYS 系统文件的装入程序的
141:    SHR     DX,CL                           ;│ 节数→DX
142:    INC     DX                              ;┘
143:    SUB     AX,DX                           ;留出 IO.SYS 系统文件的装入程序所需的内
                                                ;存空间
144:    MOV     ES,AX                           ;┐
145:    PUSH    CS                              ;│ 将 IO.SYS 系统文件的装
146:    POP     DS                              ;│ 入程序移到常规内存的
147:    MOV     CX,offset IO_Loader_End         ;│ 高端区域
148:    REP     MOVSB                           ;┘
149:    PUSH    ES                              ;┐
150:    MOV     AX,offset Loader_4              ;│ 控制权转到常规内存的高端,接着执行
151:    PUSH    AX                              ;│ IO.SYS 装入程序的后续程序
152:    RET                                     ;│
153: Loader_4:                                  ;┘
154:    MOV     AX,CS                           ;┐
155:    MOV     SS,AX                           ;│ 建立堆栈指针
156:    MOV     SP,offset StackOfLoader         ;┘
157:    XOR     AX,AX                           ;┐ DS←引导记录在常规内存中的段地址值
158:    MOV     DS,AX                           ;┘
159:    MOV     AX,DS:DBRL_BytesPerSector       ;┐
160:    XOR     BX,BX                           ;│ 计算"每簇字节数"→DX:AX
161:    MOV     BL,DS:DBRL_SectorsPerCluster    ;│
162:    MUL     BX                              ;┘
163:    MOV     CS:BytesPerCluster,AX           ;保存"每簇字节数"
164:    MOV     CS:TypeOfFAT,1                      ;设置 FAT 表的类型码(1.引导盘  ┐
                                                    ;使用 12 位 FAT 表)            │
```

```
165:    MOV     DX,WORD PTR CS:TotalOfSector+2  ;┐ DS:AX←总扇区数          |
166:    MOV     AX,WORD PTR CS:TotalOfSector    ;┘                        |
167:    SUB     AX,CS:ReservedSectors           ;┐ 减去"保留扇区数"         |
168:    SBB     DX,0                            ;┘                        |
169:    MOV     BX,CS:SectorsPerFAT             ;┐                        |
170:    SHL     BX,1                            ;├ 减去"FAT 表占用的扇区数"  |
171:    SUB     AX,BX                           ;|                        |
172:    SBB     DX,0                            ;┘                        |设
173:    MOV     BX,DS:DBRL_MaxNumOfRootDirEntries ;┐                      |置
174:    MOV     CL,4                            ;|                        | FAT
175:    SHR     BX,CL                           ;├ 减去"根目录区占用的扇区数" |表
176:    SUB     AX,BX                           ;|                        |的
177:    SBB     DX,0                            ;┘                        |类
178:    XOR     CX,CX                           ;┐ CX←每簇扇区数           |型
179:    MOV     CL,DS:DBRL_SectorsPerCluster    ;┘                        |码
180:    PUSH    AX                              ;                         |
181:    MOV     AX,DX                       ;┐ 计算引导盘"簇数的高 16       |
182:    XOR     DX,DX                       ;├ 位"值→AX                   |
183:    DIV     CX                          ;┘                           |
184:    MOV     CS:TempWordVar,AX           ;保存引导盘"簇数的高 16 位"值  |
185:    POP     AX                          ;┐ 计算引导盘"簇数的低          |
186:    DIV     CX                          ;┘ 16 位"值→AX                |
187:    CMP     AX,0FF6H                    ;┐ 若簇数<4086,则跳转           |
188:    JB      Loader_5                    ;┘                           |
189:    MOV     CS:TypeOfFAT,4              ;设置 FAT 表的类型码(4:引导盘使用 16 |
                                            ;位 FAT 表)                   |
190: Loader_5:                              ;                             ┘
191:    MOV     AX,DS:Info_51AH             ;AX←IO.SYS 系统文件的起始簇号
192:    DEC     AX                          ;┐ 设置欲读取的 IO.SYS 系统文件
193:    DEC     AX                          ;├ 内容对应的簇号
194:    MOV     CS:CurrentCluster,AX        ;┘
195:    MOV     AX,3                        ;AX←3(引导记录 Bootstrap 已读出的 IO.SYS
                                            ;系统文件的扇区数)
196:    DIV     CS:SectorsPerCluster        ;计算下次欲读取的 IO.SYS 系统文件内容对
                                            ;应的簇号
197:    CMP     AH,0                        ;┐ 若下次读取 IO.SYS 系统文件时是从整簇
198:    JE      Loader_6                    ;┘ 开始的,则跳转
199:;引导记录在读出 3 个扇区的 IO.SYS 系统文件内容之后,下次读取的位置不是从整簇开始的情况
200:    XOR     AH,AH                       ;AX=已读取的 IO.SYS 系统文件内容对应的
                                            ;整簇数
201:    PUSH    AX                              ;┐
202:    MOV     CX,WORD PTR CS:FirstDataSectorNo ;|
203:    MOV     WORD PTR CS:StartSectorNo,CX    ;|
```

```
204:    MOV     CX,WORD PTR CS:FirstDataSectorNo+2  ;|
205:    MOV     WORD PTR CS:StartSectorNo+2,CX      ;|
206:    MUL     CS:SectorsPerCluster                ;| 计算由引导记录在读
207:    ADD     WORD PTR CS:StartSectorNo,AX        ;| 出 3 个扇区的 IO.SYS
208:    ADC     WORD PTR CS:StartSectorNo+2,0       ;| 系统文件内容之后，未
209:    MOV     AX,WORD PTR DS:Info_51AH            ;| 全部读出整簇的文件
210:    DEC     AX                                  ;| 内容对应的起始扇区号
211:    DEC     AX                                  ;|
212:    XOR     BX,BX                               ;|
213:    MOV     BL,CS:SectorsPerCluster             ;|
214:    MUL     BX                                  ;|
215:    ADD     WORD PTR CS:StartSectorNo,AX        ;|
216:    ADC     WORD PTR CS:StartSectorNo+2,DX      ;|
217:    POP     AX                                  ;┘
218:    PUSH    AX
219:    MUL     CS:BytesPerCluster          ;┐
220:    MOV     DI,BIO_SEG * 16             ;|
221:    ADD     DI,AX                       ;| 传送缓冲区的地址→ES:DI
222:    XOR     AX,AX                       ;|
223:    MOV     ES,AX                       ;┘
224:    MOV     AL,CS:SectorsPerCluster     ;┐ 设定欲读取的 IO.SYS 系统文件内容的扇
225:    MOV     CS:ReadingSectors,AX        ;┘ 区数
226:    CALL    ReadDisk                    ;读取一个簇的 IO.SYS 系统文件内容
227:    POP     AX                          ;┐ AX＝IO.SYS 系统文件已被读出的簇数
228:    INC     AX                          ;┘
229: Loader_6:                              ;AX＝IO.SYS 系统文件已被读出的簇数
230:    INC     AX                          ;┐ 设定下次读取 IO.SYS 系统文件时的起
231:    ADD     CS:CurrentCluster,AX        ;┘ 始簇号
232:    DEC     AX
233:    PUSH    DS
234:    MUL     CS:BytesPerCluster          ;计算 IO.SYS 系统文件已被读出的字节数
                                            ;→AX
235:    SUB     AX,offset IO_Loader_End     ;减去 IO.SYS 装入程序的程序长度，此时
                                            ;AX＝已读出的 IO.SYS 主体程序的字节数
236:    MOV     CX,AX                       ;┐
237:    MOV     AX,BIO_SEG                  ;|
238:    MOV     DS,AX                       ;| 将已读出的 IO.SYS 系统
239:    MOV     ES,AX                       ;| 文件的主体程序复制
240:    MOV     SI,offset IO_MainCode       ;| 到内存的正确位置处
241:    XOR     DI,DI                       ;|
242:    REP     MOVSB                       ;┘
243:    MOV     CS:LoaderUsedAP,DI          ;设置存放下次读出的 IO.SYS 系统文件内
                                            ;容的内存地址偏移值
```

```
244:        POP     DS
245: Loader_7:                              ;按整簇方式并以簇为单位依次读取 IO.SYS
                                            ;系统文件的剩余文件内容
246:        XOR     AH,AH                   ;┐ AX←每簇扇区数
247:        MOV     AL,CS:SectorsPerCluster ;┘
248:        MOV     CS:ReadingSectors,AX    ;设置欲读取的扇区数(即读取整个簇的文件
                                            ;内容)
249:        PUSH    CS:ReadingSectors       ;┐
250:        CALL    Get_NextCluster         ;│ 取出并保存当前欲读取的 IO.SYS 系统
251:        POP     CS:ReadingSectors       ;│ 文件内容对应的簇号
252:        MOV     CS:CurrentCluster,AX    ;┘
253:        CMP     CS:EOF_IOSYS,0FFH       ;┐ 若 IO.SYS 系统文件已被全部读入常规内
254:        JE      ExecuteIO               ;┘ 存,则跳转
255:        XOR     DX,DX                   ;┐
256:        SUB     AX,2                    ;│
257:        XOR     CH,CH                   ;│ 计算当前簇号对应的相对扇区号
258:        MOV     CL,CS:SectorsPerCluster ;│ →DX:AX
259:        MUL     CX                      ;┘
260:        ADD     AX,WORD PTR CS:FirstDataSectorNo    ;┐ 加上引导盘数据区的起始扇区号
261:        ADC     DX,WORD PTR CS:FirstDataSectorNo+2  ;┘
262:        MOV     WORD PTR CS:StartSectorNo,AX        ;┐ 保存当前欲读取的 IO.SYS 系统文
263:        MOV     WORD PTR CS:StartSectorNo+2,DX      ;┘ 件内容对应的起始扇区号
264:        MOV     DI,CS:LoaderUsedAP      ;DI←存放当前欲读取的 IO.SYS 系统文件
                                            ;内容的内存地址偏移值
265:        PUSH    CS:ReadingSectors
266:        MOV     AX,BIO_SEG              ;┐ ES←IO.SYS 系统文件装入常规内存时的
267:        MOV     ES,AX                   ;┘ 段地址值
268:        CALL    ReadDisk                ;读取 IO.SYS 系统文件的当前簇内容到常
                                            ;规内存中
269:        POP     AX                      ;┐ 修改存放下次读取的 IO.SYS 系统文件
270:        MUL     CS:BytesPerSector       ;│ 内容的内存地址偏移值
271:        ADD     CS:LoaderUsedAP,AX      ;┘
272:        JMP     SHORT Loader_7          ;转去读取 IO.SYS 系统文件的后续内容
273: ExecuteIO:                            ;
274:        MOV     CH,CS:MediaDescriptor   ;CH←引导盘的介质描述符
275:        MOV     DL,CS:BootDriveNo       ;DL←引导盘的物理驱动器号
276:        MOV     BX,WORD PTR CS:FirstDataSectorNo    ;┐ AX:BX←引导盘数据区的起始扇区
277:        MOV     AX,WORD PTR CS:FirstDataSectorNo+2  ;┘ 号
278:        DB      0EAH                    ;┐
279:        DW      0,BIO_SEG               ;│ 控制权转到 IO.SYS 的后续部分,即
280:        ;JMP    FAR PTR 0070:0000       ;┘ IO.SYS 的初始化程序(对应于 IO1.ASM)
281: IO_Loader  ENDP
282: ;=============================================================
```

```
283: ;相对地址偏移:03B4
284: ;功能:读取指定个数的扇区
285: ;入口参数:ES:DI 指向传送缓冲区
286: ;出口参数:无
287: ;=========================================================
288: ReadDisk   PROC    NEAR
289:     MOV    CX,5                          ;CX←出错重试次数
290: ReadDisk_1:
291:     PUSH   CX                            ;保护出错重试次数
292:     MOV    AX,WORD PTR CS:StartSectorNo      ;┐ DX:AX←当前欲读取的文件内容对
293:     MOV    DX,WORD PTR CS:StartSectorNo+2    ;┘ 应的起始扇区号
294:     PUSH   AX
295:     MOV    AX,DX                         ;┐
296:     XOR    DX,DX                         ;│ 计算并保存磁道号的高 16 位值
297:     DIV    CS:SectorsPerTrack            ;│
298:     MOV    CS:TempWordVar,AX             ;┘
299:     POP    AX                            ;┐ 计算磁道号的低 16 位值
300:     DIV    CS:SectorsPerTrack            ;┘
301:     MOV    BX,CS:SectorsPerTrack         ;┐
302:     SUB    BX,DX                         ;│ 计算当前道内允许读取的扇区数→SI
303:     MOV    SI,BX                         ;┘
304:     CMP    CS:ReadingSectors,SI          ;┐ 若欲读取的内容在两个磁道上,则跳转
305:     JAE    ReadDisk_2                    ;┘
306:     MOV    SI,CS:ReadingSectors          ;SI←当前读取的扇区数
307: ReadDisk_2:                              ;AX=磁道号;SI=当前读取的扇区数
308:     INC    DL                            ;┐ BL←道内起始扇区号
309:     MOV    BL,DL                         ;┘
310:     MOV    DX,CS:TempWordVar             ;┐
311:     PUSH   AX                            ;│
312:     MOV    AX,DX                         ;│ 计算并保存圆柱号的高 16 位值
313:     XOR    DX,DX                         ;│
314:     DIV    CS:NumberOfHead               ;│
315:     MOV    CS:TempWordVar,AX             ;┘
316:     POP    AX                            ;┐ 计算圆柱号的低 16 位值→AX
317:     DIV    CS:NumberOfHead               ;┘
318:     MOV    DH,DL                         ;DH←磁头号
319:     MOV    CL,6                          ;┐
320:     SHL    AH,CL                         ;│
321:     OR     AH,BL                         ;│ 拼成 INT 13H 所规定的入口参数格式
322:     MOV    CH,AL                         ;│
323:     MOV    CL,AH                         ;┘
324:     MOV    BX,DI                         ;ES:BI 指向传送缓冲区
325:     MOV    DL,CS:BootDriveNo             ;DL←引导盘的物理驱动器号
```

```
326:    MOV     AX,SI                           ;AL←当前欲读取的扇区数
327:    MOV     AH,2                            ;┐
328:    PUSH    AX                              ;|
329:    PUSH    DI                              ;| 读取指定个数的扇区内容
330:    INT     13H                             ;|
331:    POP     DI                              ;|
332:    POP     AX                              ;┘
333:    POP     CX
334:    JNC     ReadDisk_4                      ;若读操作成功,则跳转
335:    MOV     BX,DI                           ;BX←传送缓冲区的地址偏移值
336:    XOR     AH,AH                           ;┐
337:    PUSH    CX                              ;|
338:    MOV     DL,CS:BootDriveNo               ;|
339:    PUSH    DI                              ;| 复位引导盘对应的驱动器
340:    INT     13H                             ;|
341:    POP     DI                              ;|
342:    POP     CX                              ;┘
343:    DEC     CX                              ;┐ 若重试过5次仍未成功,则跳转
344:    JZ      ReadDisk_3                      ;┘
345:    JMP     ReadDisk_1                      ;转去重新读取指定的文件内容
346: ReadDisk_3:
347:    JMP     RestartOS                       ;转去显示"系统引导失败"的错误提示信息,
                                                ;并试图重新引导DOS操作系统
348: ReadDisk_4:
349:    XOR     AH,AH                           ;┐ 修改未完成的扇区数
350:    SUB     CS:ReadingSectors,AX            ;┘
351:    JZ      ReadDisk_RET                    ;若全部读完指定的扇区数,则直接返回
352:    ADD     WORD PTR CS:StartSectorNo,AX    ;┐ 修改下次读取的文件内容所对应的起始
353:    ADC     WORD PTR CS:StartSectorNo+2,0   ;┘ 扇区号
354:    XOR     BX,BX                           ;┐
355:    MOV     BL,AL                           ;|
356:    MOV     AX,CS:BytesPerSector            ;| 修改传送缓冲区的地址
357:    MUL     BX                              ;|
358:    ADD     DI,AX                           ;┘
359:    JMP     ReadDisk                        ;转去继续读取后续的文件内容
360: ReadDisk_RET:
361:    RET
362: ReadDisk ENDP
363: ;=============================================================
364: ;相对地址偏移:0463
365: ;功能:取出IO.SYS系统文件的后续簇号
366: ;入口参数:无
367: ;出口参数:AX=取出的IO.SYS系统文件的后续簇号
```

```
;==========================================================
Get_NextCluster        PROC    NEAR
        PUSH    ES
        MOV     AX,CS:AddrOfBuffer_SEG      ;┐ ES←常规内存高端的传送缓冲区(一个扇
        MOV     ES,AX                       ;┘ 区大小)的段地址值
        MOV     CS:EOF_IOSYS,0FFH           ;设置 IO.SYS 系统文件已被全部读完的标志
        MOV     AX,CS:CurrentCluster        ;AX←当前簇号
        CMP     CS:TypeOfFAT,1              ;┐ 若引导软盘或硬盘活动分区使用 16 位
        JNE     Get_NextCluster_5           ;┘ FAT 表,则跳转
;12 位 FAT 表,AX=簇号
        MOV     SI,AX                       ;┐
        SHR     AX,1                        ;|
        ADD     SI,AX                       ;| 计算 AX 指定簇号的簇项值在 FAT 表中
        PUSH    DX                          ;| 的偏移→DX:SI
        XOR     DX,DX                       ;┘
        CALL    ReadFATTable                ;读取指定簇号的簇项值所在的扇区
        POP     DX
        JNZ     Get_NextCluster_1           ;若指定簇号的簇项值全部在当前读出的扇
                                            ;区中,则跳转
        MOV     AL,ES:[BX]                  ;┐ 保存指定簇号的簇项值低字节值
        MOV     BYTE PTR CS:ClusterNo,AL    ;┘
        INC     SI                          ;┐ 重新设定指定簇号的簇项值在 FAT 表中
        PUSH    DX                          ;| 的偏移→DX:SI,以便读取存放在下一个
        XOR     DX,DX                       ;┘ 扇区中的簇项值高字节值
        CALL    ReadFATTable                ;读取指定簇号的簇项值高字节所在的扇区
        POP     DX
        MOV     AL,BYTE PTR ES:[0]          ;┐ 保存指定簇号的簇项值高字节值
        MOV     BYTE PTR CS:ClusterNo+1,AL  ;┘
        MOV     AX,CS:ClusterNo             ;AX←指定簇号的簇项值
        JMP     SHORT Get_NextCluster_2
Get_NextCluster_1:
        MOV     AX,ES:[BX]                  ;AX←指定簇号的簇项值
Get_NextCluster_2:
        TEST    CS:CurrentCluster,1         ;┐ 若是取奇数簇号的簇项值,则跳转
        JNZ     Get_NextCluster_3           ;┘
        AND     AX,0FFFH                    ;校正簇项值(去掉无效的高 4 位值)
        JMP     SHORT Get_NextCluster_4
Get_NextCluster_3:
        MOV     CL,4                        ;┐ 校正簇项值(去掉无效的低 4 位值)
        SHR     AX,CL                       ;┘
Get_NextCluster_4:
        CMP     AX,0FF8H                    ;┐ 若当前簇号是 IO.SYS 系统文件的最后
        JAE     Get_NextCluster_7           ;┘ 簇,则跳转
```

```
410:    JMP     SHORT Get_NextCluster_6          ;转去置 IO.SYS 系统文件未结束的标志
411: Get_NextCluster_5:                          ;16 位 FAT 表,AX=簇号
412:    PUSH    DX
413:    XOR     DX,DX                            ;┐
414:    SHL     AX,1                             ;| 计算 AX 指定簇号的簇项值在 FAT 表中
415:    ADC     DX,0                             ;| 的偏移→DX:SI
416:    MOV     SI,AX                            ;┘
417:    CALL    ReadFATTable                     ;读取指定簇号的簇项值所在的扇区
418:    POP     DX
419:    MOV     AX,ES:[BX]                       ;AX←指定簇号的簇项值
420:    CMP     AX,0FFF8H                        ;┐ 若当前簇号是 IO.SYS 系统文件的最后
421:    JAE     Get_NextCluster_7                ;┘ 簇,则跳转
422: Get_NextCluster_6:
423:    MOV     CS:EOF_IOSYS,0                   ;设置 IO.SYS 系统文件未结束的标志
424: Get_NextCluster_7:
425:    POP     ES
426:    RET
427: Get_NextCluster        ENDP
428: ;=============================================================
429: ;相对地址偏移:04E2
430: ;功能:读取一个扇区的 FAT 表内容
431: ;入口参数:DX:SI=欲读取的 FAT 表内容的起始位置(字节数)
432: ;          ES=读取 FAT 表时使用的传送缓冲区的段地址值
433: ;出口参数:BX=存放簇项值的地址偏移量
434: ;          ZF=0,簇项值只在当前扇区中;ZF=1,簇项值在两个相邻的扇区中
435: ;=============================================================
436: ReadFATTable        PROC    NEAR
437:    PUSH    AX
438:    PUSH    SI
439:    PUSH    DI
440:    MOV     AX,SI                            ;┐ 计算相对扇区号,其中 AX=相对扇区
441:    MOV     CX,CS:BytesPerSector             ;| 号 DX=扇区内的偏移量
442:    DIV     CX                               ;┘
443:    CMP     AX,CS:SectorNoOfFAT              ;┐ 若欲读取的 FAT 表内容所在的扇区先
444:    JE      ReadFATTable_1                   ;┘ 前已读出,则跳转
445:    MOV     CS:SectorNoOfFAT,AX              ;保存当前欲读取的 FAT 表内容对应的相
                                                 ;对扇区号
446:    PUSH    DX
447:    XOR     DX,DX                            ;DX:AX=欲读取的 FAT 表内容对应的相
                                                 ;对扇区号
448:    ADD     AX,WORD PTR CS:HiddenSectors     ;┐ 加上"隐含扇区数"
449:    ADC     DX,WORD PTR CS:HiddenSectors+2   ;┘
450:    ADD     AX,CS:ReservedSectors            ;┐ 加上"保留扇区数"
```

```
451:    ADC     DX,0                                ;┘
452:    MOV     WORD PTR CS:StartSectorNo,AX        ;┐ 保存当前欲读取的 FAT 表内容对应的
453:    MOV     WORD PTR CS:StartSectorNo+2,DX      ;┘ 起始扇区号
454:    MOV     CS:ReadingSectors,1                 ;设置欲读取的扇区数
455:    XOR     DI,DI                               ;设置传送缓冲区的地址偏移值
456:    CALL    ReadDisk                            ;读取 FAT 表的内容
457:    POP     DX
458:    MOV     CX,CS:BytesPerSector                ;CX←每扇区字节数
459: ReadFATTable_1:
460:    DEC     CX                                  ;┐ 判定簇项值是否存放在两个相邻的扇区
461:    CMP     DX,CX                               ;┘ 中(ZF=1,是;ZF=0,不是)
462:    MOV     BX,DX                               ;BX←簇项值在扇区中的偏移
463:    POP     DI
464:    POP     SI
465:    POP     AX
466:    RET
467: ReadFATTable    ENDP
468: ;=============================================================
469: ;相对地址偏移:0532
470: ;功能:显示"系统引导失败"的错误提示信息,并试图重新引导 DOS 操作系统
471: ;入口参数:无
472: ;出口参数:无
473: ;=============================================================
474: RestartOS       PROC    NEAR
475:    PUSH    CS                          ;┐ DS:SI 指向"系统引导失败"的错误提示
476:    POP     DS                          ;| 信息串
477:    MOV     SI,offset PromptingMessage  ;┘
478:    CALL    DisplayString               ;显示错误提示信息串
479:    XOR     AH,AH                       ;┐ 等待键盘输入
480:    INT     16H                         ;┘
481:    XOR     BX,BX                       ;┐
482:    MOV     DS,BX                       ;| 恢复系统引导前设定的软盘参数表
483:    LES     BX,Old_INT1EH_Vector        ;| (说明:此处程序有错,LES BX,Old_
484:    MOV     SI,Vector_1EH_OFS           ;| INT1EH_Vector 应为 LES BX,CS:
485:    MOV     [SI],BX                     ;| Old_INT1EH_Vector)
486:    MOV     [SI+2],ES                   ;┘
487:    INT     19H                         ;重新引导操作系统
488: RestartOS    ENDP
489: ;=============================================================
490: ;相对地址偏移:0550
491: ;功能:显示 DS:SI 指定的以 0 结尾的字符串
492: ;入口参数:DS:SI 指向以 0 结尾的字符串
493: ;出口参数:无
```

```
;=============================================================
DisplayString PROC    NEAR
    LODSB                                   ;AL←当前字符
    OR      AL,AL                           ;┐若字符串结束,则直接返回
    JZ      DisplayString_RET               ;┘
    MOV     AH,0EH                          ;┐
    MOV     BL,7                            ;├以电传方式显示当前字符
    INT     10H                             ;┘
    JMP     SHORT DisplayString             ;转去显示字符串的后续字符
DisplayString_RET:
    RET
DisplayString       ENDP
;"系统引导失败"的错误提示信息串
PromptingMessage DB     0DH, 0AH, ' Non-System disk or disk error' , 0DH, 0AH
                 DB     ' Replace and press any key when ready' , 0DH, 0AH, 0
;相对地址偏移:05A6
IO_Loader_End    EQU    $                ;IO.SYS 装入程序的终止地址
IO_MainCode      EQU    $                ;IO.SYS 主体程序的起始地址
;
Loader    ENDS
    END
```

10.4.4 IO1 模块的源程序注释清单

```
PAGE    60,132
;Program Name: IO1.ASM
;Version: MS-DOS5.00
;Compiler: Marco Assembler Version 5.10
;
PUBLIC  AllowHMA,HIMEMAddr,ReadKeyCode,GetKeyStatus,Switch_T,MultiTrack
PUBLIC  CompatibleFDiskFlag,Ptr_SearchSCB,Ptr_SCBArray,StackFlag,IO3_Flag
PUBLIC  SizeOfDynamicStk,D_08D0,FreeMemOFSofHMA,PreEntry7
;
IO3_0271        EQU     0271H           ;┐
IO3_0273        EQU     0273H           ;│
IO3_0275        EQU     0275H           ;├ SysInt-Ⅱ变量
IO3_0292        EQU     0292H           ;│
IO3_0296        EQU     0296H           ;│
IO3_03FF        EQU     03FFH           ;┘
Vector_0FH_OFS  EQU     3CH             ;┐
```

```
535: Vector_0FH_SEG  EQU   3EH          ;|
536: Vector_13H_OFS  EQU   4CH          ;|
537: Vector_13H_SEG  EQU   4EH          ;|
538: Vector_15H_OFS  EQU   54H          ;|
539: Vector_15H_SEG  EQU   56H          ;|
540: Vector_19H_OFS  EQU   64H          ;|
541: Vector_19H_SEG  EQU   66H          ;|
542: Vector_1BH_OFS  EQU   6CH          ;| 存放中断向量的地址
543: Vector_1BH_SEG  EQU   6EH          ;|
544: Vector_1EH_OFS  EQU   78H          ;|
545: Vector_1EH_SEG  EQU   7AH          ;|
546: Vector_29H_OFS  EQU   0A4H         ;|
547: Vector_29H_SEG  EQU   0A6H         ;|
548: Vector_2FH_OFS  EQU   0BCH         ;|
549: Vector_2FH_SEG  EQU   0BEH         ;|
550: Vector_6CH_OFS  EQU   1B0H         ;|
551: Vector_6CH_SEG  EQU   1B2H         ;┘
552: Info_17H        EQU   17H          ;┐
553: Info_42H        EQU   42H          ;|
554: Info_43H        EQU   43H          ;|
555: Info_44H        EQU   44H          ;|
556: Info_45H        EQU   45H          ;|
557: Info_46H        EQU   46H          ;| DOS 通讯区(段地址为 40H)
558: Info_47H        EQU   47H          ;|
559: Info_48H        EQU   48H          ;|
560: Info_74H        EQU   74H          ;|
561: Info_75H        EQU   75H          ;|
562: Info_76H        EQU   76H          ;┘
563: Info_496H       EQU   496H         ;┐
564: Info_500H       EQU   500H         ;|
565: Info_504H       EQU   504H         ;|
566: Info_522H       EQU   522H         ;| DOS 通讯区(段地址为 0)
567: Info_52BH       EQU   52BH         ;|
568: Info_52CH       EQU   52CH         ;|
569: Info_534H       EQU   534H         ;┘
570: MSDOS_Cluster   EQU   53AH         ;MSDOS.SYS 系统文件起始簇号的存放地址
                                        ;(段地址为 0)
571: Bootstrap_7C16H EQU   7C16H        ;引导扇区中 BPB 参数块参数(每个 FAT 表占
                                        ;用的扇区数,段地址为 0)
572: ROMAddr_000E    EQU   0EH          ;┐ ROM BIOS 常量(段地址为 0F000H)
573: ROMAddr_FFFE    EQU   0FFFEH       ;┘
574: ;
575: .286c
```

```
INCLUDE   BIO.STR
;
IO1     SEGMENT  ' CODE'
    ASSUME CS:IO1,DS:IO1,ES:IO1,SS:IO1
    ORG     0
    JMP     SysInt_I_Entry
MSDOS1_SEG    DW      0                 ;保存 MSDOS1 模块在常规内存低端的段地址值
;相对地址偏移:0005
    DB      0EAH                      ;┐由 IO1 模块定义的 INT 2FH 中断服务程序的程
    DW      ShiftInterface, BIO_SEG   ;│序转移接口(它供 MSDOS2 模块使用)
    ;JMP    0070:0893H                ;┘
    DW      0                         ;未使用
CON_Buffer      DB      0             ;CON 输入设备的输入缓冲区,它暂存等待输入字
                                      ;符的 ASCII 码
AllowHMA        DB      0             ;标志(0:IO2 模块和 MSDOS2 模块被安装在常规内
                                      ;存低端;-1:HMA 的自由内存区的起始地址偏移
                                      ;已确定,即 IO2 模块和 MSDOS2 模块已被安装在
                                      ;HMA 中)
HIMEMAddr       DD      0             ;保存"HIMEM.SYS 提供的扩充内存管理程序"的
                                      ;入口地址
DRH_Ptr         DD      0             ;保存设备驱动程序请求标题的起始地址
;作为 AUX 设备的输入缓冲区,它暂存等待输入字符的 ASCII 码
AUX_Buffer    DB        0             ;AUX(COM1)的输入缓冲区
    DB      0                       ;COM2 的输入缓冲区
    DB      0                       ;COM3 的输入缓冲区
    DB      0                       ;COM4 的输入缓冲区
ZeroValue DW      0                 ;中断向量表、BIOS 通讯区和 DOS 通讯区的段地址
                                    ;值(0000H)
DSValue   DW      0                 ;保存 INT 13H 的入口参数 DS 寄存器值
AXValue   DW      0                 ;保存 INT 13H 的入口参数 AX 寄存器值
UncompletedSectors    DB      0     ;存放"未读完的扇区数"
;相对地址偏移:0021
Device_No DB      0                 ;存放设备单元号
          DB      0                 ;未使用
;相对地址偏移:002
;CON Device Driver Header
CON_DriverHeader    DW  AUX_DriverHeader ;┐指向设备驱动程序标题链中下一个设备驱动程
    DW      BIO_SEG                  ;┘序标题
    DW      8013H                    ;CON 设备属性
    DW      Strategy                 ;策略过程入口
    DW      CON_Intr                 ;中断过程入口
    DB      ' CON '                  ;设备名
```

```
612: ;相对地址偏移:0035
613: ;AUX Device Driver Header
614: AUX_DriverHeader    DW    PRN_DriverHeader ;┐ 指向下一个设备驱动程序标题
615:     DW       BIO_SEG                      ;┘
616:     DW       8000H                        ;AUX 设备属性
617:     DW       Strategy                     ;策略过程入口
618:     DW       AUX_Intr                     ;中断过程入口
619:     DB       ' AUX '                      ;设备名
620: ;相对地址偏移:0047
621: ;PRN Device Driver Header
622: PRN_DriverHeader    DW    CLOCK$_DriverHeader   ;┐ 指向下一个设备驱动程序标题
623:     DW       BIO_SEG                          ;┘
624:     DW       0A0C0H                       ;PRN 设备属性
625:     DW       Strategy                     ;策略过程入口
626:     DW       PRN_Intr                     ;中断过程入口
627:     DB       ' PRN '                      ;设备名
628: ;相对地址偏移:0059
629: ;CLOCK$ Device Driver Header
630: CLOCK$_DriverHeader  DW Disk_DriverHeader      ;┐ 下一个设备驱动程序标题
631:     DW       BIO_SEG                          ;┘
632:     DW       8008H                        ;时钟(CLOCK$)设备属性
633:     DW       Strategy                     ;策略过程入口
634:     DW       CLOCK$_Intr                  ;中断过程入口
635:     DB       ' CLOCK$ '                   ;设备名
636: ;相对地址偏移:006B
637: ;Disk Device Driver Header
638: Disk_DriverHeader   DW    COM1_DriverHeader ;┐ 指向下一个设备驱动程序标题
639:     DW       BIO_SEG                      ;┘
640:     DW       08C2H                        ;块设备的设备属性
641:     DW       Strategy                     ;策略过程入口
642:     DW       Disk_Intr                    ;中断过程入口
643: ;相对地址偏移:0075
644: TotalOfBlockDev  DB      4                ;系统允许访问的逻辑驱动器数
645: DiskNumber       DB      0FEH             ;暂存当前操作的物理驱动器号
646: ChangeLine       DB      0                ;Change Line 允许标志(0:软盘驱动器不支持
                                               ;Change Line;1:软盘驱动器支持 Change Line)
647: SingleFDisk      DB      0                ;逻辑驱动器 A、B 共用同一物理驱动器的标志(0:
                                               ;不共用;1:共用;2:A、B 共用同一物理驱动器,当
                                               ;前正在建立逻辑驱动器 B 的 BDPB)
648: Support_15H      DB      0                ;ROM BIOS 支持"等待事件"功能请求的标志(0:不
                                               ;支持;1:支持)
649: EnableWR_Flag    DB      0                ;DOS 通讯区的 0:504H 字节单元的写操作允许标
                                               ;志(0:不允许写;1:允许写)
```

```
650: ;相对地址偏移:007B
651: ;COM1 Device Driver Header
652: COM1_DriverHeader   DW   LPT1_DriverHeader ;┐ 指向下一个设备驱动程序标题
653:     DW      BIO_SEG                        ;┘
654:     DW      8000H                          ;COM1 设备属性
655:     DW      Strategy                       ;策略过程入口
656:     DW      COM1_Intr                      ;中断过程入口
657:     DB      ' COM1 '                       ;设备名
658: ;相对地址偏移:008D
659: ;LPT1 Device Driver Header
660: LPT1_DriverHeader   DW   LPT2_DriverHeader    ;┐ 指向下一个设备驱动程序标题
661:     DW      BIO_SEG                          ;┘
662:     DW      0A0C0H                         ;LPT1 设备属性
663:     DW      Strategy                       ;策略过程入口
664:     DW      LPT1_Intr                      ;中断过程入口
665:     DB      ' LPT1 '                       ;设备名
666: ;相对地址偏移:009F
667: ;LPT2 Device Driver Header
668: LPT2_DriverHeader   DW   LPT3_DriverHeader ;┐ 指向下一个设备驱动程序标题
669:     DW      BIO_SEG                        ;┘
670:     DW      0A0C0H                         ;LPT2 设备属性
671:     DW      Strategy                       ;策略过程入口
672:     DW      LPT2_Intr                      ;中断过程入口
673:     DB      ' LPT2 '                       ;设备名
674:     DB      0, 0, 0                        ;未使用
675: ;相对地址偏移:00B4
676: INT13H_Shift_Entry  DD   0                 ;保存依据不同机器而设定的 INT 13H 中断接管程
                                                ;序的入口地址
677: ;相对地址偏移:00B8
678: ;LPT3 Device Driver Header
679: LPT3_DriverHeader DW   COM2_DriverHeader ;┐ 指向下一个设备驱动程序标题
680:     DW      BIO_SEG                        ;┘
681:     DW      0A0C0H                         ;LPT3 设备属性
682:     DW      Strategy                       ;策略过程入口
683:     DW      LPT3_Intr                      ;中断过程入口
684:     DB      ' LPT3 '                       ;设备名
685: ;相对地址偏移:00CA
686: ;COM2 Device Driver Header
687: COM2_DriverHeader   DW COM3_DriverHeader ;┐ 指向下一个设备驱动程序标题
688:     DW      BIO_SEG                        ;┘
689:     DW      8000H                          ;COM2 设备属性
690:     DW      Strategy                       ;策略过程入口
691:     DW      COM2_Intr                      ;中断过程入口
```

```
692:    DB      ' COM2 '                        ;设备名
693: ;相对地址偏移:00DC
694: ;COM3 Device Driver Header
695: COM3_DriverHeader  DW COM4_DriverHeader  ;┐指向下一个设备驱动程序标题
696:    DW      BIO_SEG                         ;┘
697:    DW      8000H                           ;COM3 设备属性
698:    DW      Strategy                        ;策略过程入口
699:    DW      COM3_Intr                       ;中断过程入口
700:    DB      ' COM3 '                        ;设备名
701: ;相对地址偏移:00EE
702: ;COM4 Device Driver Header
703: COM4_DriverHeader  DW      -1              ;┐由 IO1 模块提供的设备驱动程
704:    DW      BIO_SEG                         ;┘序标题链的尾结点指针域
705:    DW      8000H                           ;COM4 设备属性
706:    DW      Strategy                        ;策略过程入口
707:    DW      COM4_Intr                       ;中断过程入口
708:    DB      ' COM4 '                        ;设备名
709: ;相对地址偏移:0100
710: ;保存 ROM BIOS 自检时设定的中断向量的数据块,它确保了 INT 19H 热启动的工作环境
711: INT10H_Number      DB  10H                 ;INT 10H 的中断号
712:                    DD  0                   ;INT 10H 的中断向量
713:                    DB  13H                 ;INT 13H 的中断号
714: Old_INT13H_Vector  DD  0                   ;INT 13H 的中断向量
715:                    DB  15H                 ;INT 15H 的中断号
716: Old_INT15H_Vector  DD  0                   ;INT 15H 的中断向量
717:                    DB  19H                 ;INT 19H 的中断号
718: Old_INT19H_Vector  DD  0                   ;INT 19H 的中断向量
719:                    DB  1BH                 ;INT 1BH 的中断号
720:                    DD  0                   ;INT 1BH 的中断向量
721: ;相对地址偏移:0119
722: BDPB_Head          DW FDiskBDPB1           ;┐BDPB 链的头结点指针
723:                    DW BIO_SEG              ;┘
724: Operate_Times      DB  0                   ;在测试时钟计数时,它存放时钟计数值未改变的
                                                ;测试次数(时钟计数值未改变的计数器)
725: Drive_Number       DB  -1                  ;暂存当前操作的物理驱动器号。若为-1,则表明
                                                ;盘操作出错(或盘片已被更换)
726: DiskMedia          DB  0                   ;当前正在读的盘片的介质描述符
727: INT13H_CMDCode     DB  2                   ;存放 INT 13H 中断请求的命令码
728: VerifyFlag         DB  0                   ;写验证标志(0:不验证;1:验证)
729: SectorsOfRemain    DW  0                   ;存放未完成(待操作)的扇区数
730:                    DB  0                   ;未使用
731: NumberOfFDisk1     DB  1                   ;保存系统配置的软盘驱动器数目
732: StartupWaitTime    DB  0                   ;保存先前的软盘参数表的"马达启动时间"
```

```
733: HeadLoadTime          DB  0                ;保存先前的软盘参数表的"寻道后磁头的稳定时
                                                ;间"
734: NewHeadLoadTime       DB  0                ;新的软盘参数表的"寻道后磁头的稳定时间"
735:                       DB  0                ;未使用
736: HeadLoadTime_BAK      DB  0                ;保存先前的软盘参数表的"寻道后磁头的稳定时
                                                ;间"
737: SectorsPerTrack_BAK   DB  0                ;保存先前的软盘参数表的"每道扇区数"
738: MaxSectorsPerTrack    DB  9                ;保存每道最大扇区数
739: Old_INT1EH_Vector     DD  0                ;保存 INT 1EH 中断先前的中断向量
740: SectorNumber          DB  0                ;存放 INT 13H 中断请求时的道内扇区号
741: HeadNumber            DB  0                ;存放 INT 13H 中断请求时的磁头号
742: Cylinder              DW  0                ;存放 INT 13H 中断请求时的圆柱面号
743: SPValue               DW  0                ;SP 寄存器值的暂存单元
744: SecPerTK_FMT          DB  8                ;在格式化并验证逻辑设备的磁道时它存放"每道
                                                ;扇区数"
745: Head_FMT              DB  0                ;在格式化并验证逻辑设备的磁道时它存放"磁头
                                                ;号"
746: Cylinder_FMT          DW  0                ;在格式化并验证逻辑设备的磁道时它存放"圆柱
                                                ;面号"
747: GapLength_FMT    DB    50H                 ;在格式化时它存放扇区间的间隔长度
748: ;相对地址偏移:013C
749: INT13H_RetCode   DB  0CCH, 80H, 40H, 10H, 8, 6, 4, 3  ;INT 13H 中断请求返回的错误码表
750: INT13H_TS        DB    0                   ;INT 13H 中断请求返回的状态字节(AH 寄存器)
                                                ;值,它作为查 INT13H_RetCode 表的结束标志
751: Drive_ErrCode DB    0AH, 2, 6, 4, 4, 0FH, 8, 0, 0CH;用于转换 INT 13H 中断请求返回的错
                                                ;误码到设备驱动程序返回的错误码表
752: ;相对地址偏移:014E
753: DiskBuffer       DB    512    Dup(0)       ;盘 I/O 缓冲区
754: ;相对地址偏移:034E                         ;第 1 个软盘驱动器的 BDPB
755: FDiskBDPB1    BDPB <FDiskBDPB2,,0,0,,-1,,,40H,168H,0,2,9,1,,,,,,,,,1,1,2,0E0H,
     168H,0F0H,2,9,2,,,,,,,,>
756: ;相对地址偏移:03B2                         ;第 2 个软盘驱动器的 BDPB
757: FDiskBDPB2    BDPB <FDiskBDPB3,,0,0,,-1,,,40H,168H,0,2,9,1,,,,,,,,,1,1,2,0E0H,
     168H,0F0H,2,9,2,,,,,,,,>
758: ;相对地址偏移:0416                         ;第 3 个软盘驱动器的 BDPB
759: FDiskBDPB3    BDPB <FDiskBDPB4,,0,0,,-1,,,40H,168H,0,2,9,1,,,,,,,,,1,1,2,0E0H,
     168H,0F0H,2,9,2,,,,,,,,>
760: ;相对地址偏移:047A                         ;第 4 个软盘驱动器的 BDPB
761: FDiskBDPB4    BDPB <,,0,0,,-1,,,40H,168H,0,2,9,1,,,,,,,,,1,1,2,0E0H,168H,0F0H,2,
     9,2,,,,,,,,>
762: ;相对地址偏移:04DE
763: HDtoDD_BPB_Para  DB    3                   ;高密软盘驱动器作低密用时"每个 FAT 表占用的
                                                ;扇区数"
```

```
764:        DB      9                       ;高密软盘驱动器作低密用时"每道扇区数"
765:        DB      70H                     ;高密软盘驱动器作低密用时"最大根目录登记项
                                            ;数"
766:        DW      05A0H                   ;高密软盘驱动器作低密用时"总扇区数"
767:        DB      2                       ;高密软盘驱动器作低密用时"磁头数"
768:        DB      2                       ;高密软盘驱动器作低密用时"每簇扇区数"
769:  ;相对地址偏移:04E5
770:  ReadKeyCode   DB  0                   ;"读常规键盘/读扩展键盘"输入的命令码(00H/
                                            ;10H),其中 00H 对应于"switches=/K"
771:  GetKeyStatus  DB  1                   ;"取常规键盘/取扩展键盘"状态的命令码(01H/
                                            ;11H),其中 01H 对应于"switches=/K"
772:  OutDeviceNo   DB  0                   ;存放标准输出设备的设备名号(0=PRN,1
                                            ;=LPT1,2=LPT2,…)
773:  ;相对地址偏移:04E8
774:  PRN_RepeatCount DW  50H, 50H,50H,50H  ;PRN 设备(LPT1～LPT4)的出错重试次数表
775:  ;相对地址偏移:04F0
776:  DayCount   DW      0                   ;存放当前日期对应的总天数
777:  Switch_T   DB      0                   ;switches 系统配置命令的"/T 开关"标志(1:有/T
                                            ;开关,此时 INT 1AH AH=0 中断服务返回的 AL
                                            ;>0 表示日期天数加 1;0:无/T,开关此时 INT
                                            ;1AH AH=0 中断服务返回的 AL=日期天数增加
                                            ;值)
778:  RTCFlag    DB      0                   ;实时钟存在标志(0:不存在;1:存在)
779:  Century    DB      19                  ;存放世纪号
780:  Year       DB      80                  ;存放年号
781:  DayPerMonth  DB    31                  ;┐
782:  Febmary    DB      28                  ;│一年中每月的天数表
783:             DB      31, 30, 31, 30, 31, 31  ;│
784:             DB      30, 31, 30, 31      ;┘
785:  ;相对地址偏移:0502
786:  PreEntry1  DW      ConvertData         ;┐存放子程序(ConvertData)的入口地址,该子程
                                            ;│序由 IO1 模块定义,其功能是将 AL 指定的二进
787:             DW      BIO_SEG             ;┘制数据转换成 BCD 码格式
788:  PreEntry2  DW      BINtoBCD            ;┐存放子程序(BINtoBCD)的入口地址,该子程序
                                            ;│由 IO1 模块定义,其功能是将日期数据转换
789:             DW      BIO_SEG             ;┘成 BCD 码数据格式
790:  ;相对地址偏移:050A
791:  ChangeFlag    DB  0                   ;更换"卷系列号、卷标名和文件系统类型标志串"
                                            ;的控制标志(1:需要更换;2:已更换)
792:  FAT_12Bits    DB  'FAT12',0           ;文件系统类型标志串,它表明 FAT 表的每项长 12
                                            ;位
793:  FAT_16Bits    DB  'FAT16',0           ;文件系统类型标志串,它表明 FAT 表的每项长 16
                                            ;位
```

```
794: DefaultVolume  DB  ' NO NAME ' ,0          ;缺省的磁盘卷标名
795: ;相对地址偏移:0529
796: Temp_HW            DW   0                  ;它暂存参数值的高 16 位(如圆柱面数、磁道号等)
797: StartingSector_HW  DW   0                  ;保存 32 位起始扇区号的高 16 位值(即高字值)
798: StartingSector_LW  DW   0                  ;保存绝对的起始扇区号的低 16 位值(即低字值)
799: MultiTrack         DW   0                  ;MultiTrack 检查状态(1:OFF;80:ON。参见 IO3 模
                                                ;块)
800: CompatibleFDiskFlag  DB  0                 ;兼容的 3.5 英寸软盘驱动器标志(bit0=1 表示 A
                                                ;驱动器是一个兼容的 3.5 英寸软盘驱动器;bit1
                                                ;=1 表示 B 驱动器是一个兼容的 3.5 英寸软盘驱
                                                ;动器;…)
801: RetriesForUnECC    DW   0                  ;保存出错重试次数(非 ECC 纠正数据错误)的重
                                                ;试次数
802: RetriesForECC      DW   0                  ;保存出错重试次数(ECC 纠正数据错误)的重试次
                                                ;数
803: MultiTrack_OpFlag  DB   0                  ;多磁道操作的标志(0:只对一个磁道操作;1:多
                                                ;磁道操作)
804: ESDSValue          DW   0                  ;暂存段寄存器值(它暂存传送缓冲区的段地址值;
                                                ;或格式化磁道时,它保存 DS 寄存器值)
805: ;相对地址偏移:0539
806: ;地址域数组,它的每一个地址域格式如下:
807: ;第一个字节:圆柱面号
808: ;第二个字节:磁头号
809: ;第三个字节:扇区号
810: ;第四个字节:每扇区字节数的标志值(0=128,1=256,2=512,3=1024)
811: AddressField   DW   24H                    ;磁道中的扇区数(即地址域数组中的扇区数)
812:     DB      0, 0, 1, 2
813:     DB      0, 0, 2, 2
814:     DB      0, 0, 3, 2
815:     DB      0, 0, 4, 2
816:     DB      0, 0, 5, 2
817:     DB      0, 0, 6, 2
818:     DB      0, 0, 7, 2
819:     DB      0, 0, 8, 2
820:     DB      0, 0, 9, 2
821:     DB      0, 0, 0AH, 2
822:     DB      0, 0, 0BH, 2
823:     DB      0, 0, 0CH, 2
824:     DB      0, 0, 0DH, 2
825:     DB      0, 0, 0EH, 2
826:     DB      0, 0, 0FH, 2
827:     DB      0, 0, 10H, 2
828:     DB      0, 0, 11H, 2
```

```
829:    DB      0, 0, 12H, 2
830:    DB      0, 0, 13H, 2
831:    DB      0, 0, 14H, 2
832:    DB      0, 0, 15H, 2
833:    DB      0, 0, 16H, 2
834:    DB      0, 0, 17H, 2
835:    DB      0, 0, 18H, 2
836:    DB      0, 0, 19H, 2
837:    DB      0, 0, 1AH, 2
838:    DB      0, 0, 1BH, 2
839:    DB      0, 0, 1CH, 2
840:    DB      0, 0, 1DH, 2
841:    DB      0, 0, 1EH, 2
842:    DB      0, 0, 1FH, 2
843:    DB      0, 0, 20H, 2
844:    DB      0, 0, 21H, 2
845:    DB      0, 0, 22H, 2
846:    DB      0, 0, 23H, 2
847:    DB      0, 0, 24H, 2
848:    DB      27 * 4 Dup(0)
849: ;相对地址偏移:0637
850: MediaTypeFlag DB    0                  ;当驱动器中实际介质不能被确定时使用的介质类
                                           ;型(0:1.2MB 软驱存取 1.2MB 盘片;1:1.2MB
                                           ;软驱存取 360KB 盘片)
851: ModifyFlag     DB    0                ;INT 1EH 中断向量更换标志(0:未更换;1:更换,
                                           ;即不允许再修改软盘参数表的参数值,此时该软
                                           ;盘参数表是由"为格式化而设置介质类型"返
                                           ;回的)
852: FMTOpStatus    DB    0                ;"格式化磁道操作"失败标志(0:成功;1:失败)
853: Old_INT1EH_Vector_FMT   DD   -1       ;在格式化磁道时,它存放先前的 INT 1EH 中断向
                                           ;量
854: ModelByte_Init      DB  -1            ;机器型号字节
855: SubmodelByte_Init   DB  0             ;机器子型号字节
856: StackFlag           DB  0             ;动态堆栈建立标志(0:没有动态堆栈;1:通过
                                           ;stacks 系统配置命令设置了动态堆栈)
857: ;
858: PUBLIC Old_INT02H_Vector1, Old_INT08H_Vector1, Old_INT09H_Vector1
859: PUBLIC Old_INT0AH_Vector1, Old_INT0BH_Vector1, Old_INT0CH_Vector1, Old_INT0DH_Vector1
860: PUBLIC Old_INT0EH_Vector1, Old_INT70H_Vector1, Old_INT72H_Vector1, Old_INT73H_Vector1
861: PUBLIC Old_INT74H_Vector1, Old_INT76H_Vector1, Old_INT77H_Vector1
862: ;相对地址偏移:0641
863: ;部分硬中断的中断向量数组。在系统配置时,如果使用 stacks 系统配置命令指定了动态堆栈,那么
     ;DOS 操作系统在启动时就接管 INT 02H、INT 08H～INT 0EH、INT 70H、INT 72H～INT 74H、INT
```

```
;76H～INT 77H 的中断服务程序,以便给它们提供动态堆栈。此时,DOS 使用该数组存放先前的
;中断向量值。
HardIntVectorArray      EQU   $
                        DB    2              ;INT 02H 的中断号
Old_INT02H_Vector1      DD    -1             ;保存原来的 INT 02H 中断向量
                        DB    08             ;INT 08H 的中断号
Old_INT08H_Vector1      DD    -1             ;保存原来的 INT 08H 中断向量
                        DB    09             ;INT 09H 的中断号
Old_INT09H_Vector1      DD    -1             ;保存原来的 INT 09H 中断向量
                        DB    0AH            ;INT 0AH 的中断号
Old_INT0AH_Vector1      DD    -1             ;保存原来的 INT 0AH 中断向量
                        DB    0BH            ;INT 0BH 的中断号
Old_INT0BH_Vector1      DD    -1             ;保存原来的 INT 0BH 中断向量
                        DB    0CH            ;INT 0CH 的中断号
Old_INT0CH_Vector1      DD    -1             ;保存原来的 INT 0CH 中断向量
                        DB    0DH            ;INT 0DH 的中断号
Old_INT0DH_Vector1      DD    -1             ;保存原来的 INT 0DH 中断向量
                        DB    0EH            ;INT 0EH 的中断号
Old_INT0EH_Vector1      DD    -1             ;保存原来的 INT 0EH 中断向量
                        DB    70H            ;INT 70H 的中断号
Old_INT70H_Vector1      DD    -1             ;保存原来的 INT 70H 中断向量
                        DB    72H            ;INT 72H 的中断号
Old_INT72H_Vector1      DD    -1             ;保存原来的 INT 72H 中断向量
                        DB    73H            ;INT 73H 的中断号
Old_INT73H_Vector1      DD    -1             ;保存原来的 INT 73H 中断向量
                        DB    74H            ;INT 74H 的中断号
Old_INT74H_Vector1      DD    -1             ;保存原来的 INT 74H 中断向量
                        DB    76H            ;INT 76H 的中断号
Old_INT76H_Vector1      DD    -1             ;保存原来的 INT 76H 中断向量
                        DB    77H            ;INT 77H 的中断号
Old_INT77H_Vector1      DD    -1             ;保存原来的 INT 77H 中断向量
;相对地址偏移:0687
BPB_Ptr_Array DW        FDiskBDPB1+6         ;┐
              DW        FDiskBDPB2+6         ;├缺省的软盘 BPB 参数块指针数组
              DW        FDiskBDPB3+6         ;│
              DW        FDiskBDPB4+6         ;┘
              DW        22 DUP (0)           ;缺省的硬盘 BPB 参数块指针数组
;相对地址偏移:06BB
ReturnAddr    DW        0                    ;┐存放返回地址
              DW        0                    ;┘
RealTimeRetPara1    DB      0                ;存放实时时钟"日期/时间"操作返回的世纪号/小
                                             ;时数据
RealTimeRetPara2    DB      0                ;存放实时时钟"日期/时间"操作返回的年号/分数
```

```
                                                    ;据
904: RealTimeRetPara3    DB      0                  ;存放实时时钟"日期/时间"操作返回的月份号/秒
                                                    ;数据
905: RealTimeRetPara4    DB      0                  ;存放实时时钟"日期/时间"操作返回的日期值/夏
                                                    ;时制标志数据
906: ;相对地址偏移:06C3
907: ;当前月份以前的所有月份对应的天数累加和表
908: TotalOfDay_Table    DW      0                  ;1 月份以前的天数
909:          DW      31                           ;2 月份以前的天数
910:          DW      59                           ;3 月份以前的天数
911:          DW      90                           ;4 月份以前的天数
912:          DW      120                          ;5 月份以前的天数
913:          DW      151                          ;6 月份以前的天数
914:          DW      181                          ;7 月份以前的天数
915:          DW      212                          ;8 月份以前的天数
916:          DW      243                          ;9 月份以前的天数
917:          DW      273                          ;10 月份以前的天数
918:          DW      304                          ;11 月份以前的天数
919:          DW      334                          ;12 月份以前的天数
920: ;相对地址偏移:06DB
921: ClockFlag DW     0                            ;在实时时钟日期操作时,它存放错误码或当前日
                                                    ;期对应的总天数和
922:          DB      0                            ;未使用
923: PreEntry3 DW     43H                          ;┐存放设备驱动程序的统一中断过程的入口地
                                                    ;┘址,该统一中断过程由 IO2 模块定义
924:          DW      2C7H                         ;┘
925: PreEntry4 DW     396H                         ;┐保存"将时间转换成时钟计数值"的子程序的入
926:          DW      2C7H                         ;┘口地址,该子程序由 IO2 模块定义
927: PreEntry5 DW     1302H                        ;┐保存由 IO2 模块定义的 INT 2FH 中断服务程
928:          DW      2C7H                         ;┘序的入口地址
929: PreEntry6 DW     154BH                        ;┐保存由 IO2 模块定义的 INT 13H 中断服务程
930:          DW      2C7H                         ;┘序的入口地址
931: ;=============================================================
932: ;相对地址偏移:06EE
933: ;功能: INT 1BH 中断服务程序
934: ;=============================================================
935: INT1BH_ISR    PROC    FAR
936:     MOV     BYTE PTR CS:CON_Buffer,03        ;在 CON 输入设备的字符缓冲区中存放 ASCII 值
                                                    ;03H(Ctrl+Break)
937: DummyISR  LABEL   FAR                         ;虚拟的中断服务程序
938:     IRET
939: INT1BH_ISR ENDP
940: ;=============================================================
```

```
941: ;相对地址偏移:06F5
942: ;功能: 设备驱动程序的统一策略过程
943: ;=========================================================
944: Strategy    PROC    FAR
945:     MOV     WORD PTR CS:DRH_Ptr,BX      ;┐ 保存设备驱动程序请求标题的起始地址
946:     MOV     WORD PTR CS:DRH_Ptr+2,ES    ;┘
947:     RETF
948: Strategy    ENDP
949: ;=========================================================
950: ;相对地址偏移:0700
951: ;功能: 设备驱动程序的中断过程
952: ;=========================================================
953: IO1_Interrupt  PROC    FAR
954: CON_Intr   LABEL   NEAR                ;CON 设备驱动程序的中断过程入口
955:     CALL    PublicIntrProcess          ;控制权转入由 IO2 模块定义的统一中断过程,以
                                            ;便完成指定的功能
956:     DW      CON_CMDNum                 ;存放字节单元的地址,该字节单元存放"CON 设备
                                            ;驱动程序支持的最大命令码"
957: PRN_Intr   LABEL   NEAR                ;PRN 设备驱动程序的中断过程入口
958:     CALL    PublicIntrProcess          ;控制权转入由 IO2 模块定义的统一中断过程,以
                                            ;便完成指定的功能
959:     DW      PRN_CMDNum                 ;存放字节单元的地址,该字节单元存放"PRN 设备
                                            ;驱动程序支持的最大命令码"
960:     DW      0                          ;高字节为设备标识号(PRN=0);低字节为设备号
                                            ;(PRN=0)
961: LPT1_Intr  LABEL   NEAR                ;LPT1 设备驱动程序的中断过程入口
962:     CALL    PublicIntrProcess          ;控制权转入由 IO2 模块定义的统一中断过程,以
                                            ;便完成指定的功能
963:     DW      PRN_CMDNum                 ;存放字节单元的地址,该字节单元存放"LPT1 设
                                            ;备驱动程序支持的最大命令码"
964:     DW      0100H                      ;高字节为设备标识号(LPT1=1);低字节为设备号
                                            ;(LPT1=0)
965: LPT2_Intr  LABEL   NEAR                ;LPT2 设备驱动程序的中断过程入口
966:     CALL    PublicIntrProcess          ;控制权转入由 IO2 模块定义的统一中断过程,以
                                            ;便完成指定的功能
967:     DW      PRN_CMDNum                 ;存放字节单元的地址,该字节单元存放"LPT2 设
                                            ;备驱动程序支持的最大命令码"
968:     DW      0201H                      ;高字节为设备标识号(LPT2=2);低字节为设备号
                                            ;(LPT2=1)
969: LPT3_Intr  LABEL   NEAR                ;LPT3 设备驱动程序的中断过程入口
970:     CALL    PublicIntrProcess          ;控制权转入由 IO2 模块定义的统一中断过程,以
                                            ;便完成指定的功能
971:     DW      PRN_CMDNum                 ;存放字节单元的地址,该字节单元存放"LPT3 设
```

```
                                        ;备驱动程序支持的最大命令码"
972:    DW      0302H                   ;高字节为设备标识号(LPT3=3)低字节为设备号
                                        ;(LPT3=2)
973: AUX_Intr   LABEL   NEAR            ;AUX 设备驱动程序的中断过程入口
974: COM1_Intr  LABEL   NEAR            ;COM1 设备驱动程序的中断过程入口
975:    CALL    PublicIntrProcess       ;控制权转入由 IO2 模块定义的统一中断过程,以
                                        ;便完成指定的功能
976:    DW      AUX_CMDNum              ;存放字节单元的地址,该字节单元存放"COM1 设
                                        ;备驱动程序支持的最大命令码"
977:    DB      0                       ;设备号(COM1=0,AUX=0)
978: COM2_Intr  LABEL   NEAR            ;COM2 设备驱动程序的中断过程入口
979:    CALL    PublicIntrProcess       ;控制权转入由 IO2 模块定义的统一中断过程,以
                                        ;便完成指定的功能
980:    DW      AUX_CMDNum              ;存放字节单元的地址,该字节单元存放"COM2 设
                                        ;备驱动程序支持的最大命令码"
981:    DB      1                       ;设备号(COM2=1)
982: COM3_Intr  LABEL   NEAR            ;COM3 设备驱动程序的中断过程入口
983:    CALL    PublicIntrProcess       ;控制权转入由 IO2 模块定义的统一中断过程,以
                                        ;便完成指定的功能
984:    DW      AUX_CMDNum              ;存放字节单元的地址,该字节单元存放"COM3 设
                                        ;备驱动程序支持的最大命令码"
985:    DB      2                       ;设备号(COM3=2)
986: COM4_Intr  LABEL   NEAR            ;COM4 设备驱动程序的中断过程入口
987:    CALL    PublicIntrProcess       ;控制权转入由 IO2 模块定义的统一中断过程,以
                                        ;便完成指定的功能
988:    DW      AUX_CMDNum              ;存放字节单元的地址,该字节单元存放"COM4 设
                                        ;备驱动程序支持的最大命令码"
989:    DB      3                       ;设备号(COM4=3)
990: CLOCK$_Intr    LABEL   NEAR        ;CLOCK $ 设备驱动程序的中断过程入口
991:    CALL    PublicIntrProcess       ;控制权转入由 IO2 模块定义的统一中断过程,以
                                        ;便完成指定的功能
992:    DW      CLOCK$_CMDNum           ;存放字节单元的地址,该字节单元存放"CLOCK
                                        ;$ 设备动程序支持的最大命令码"
993: Disk_Intr  LABEL   NEAR            ;块设备驱动程序的中断过程入口
994:    CALL    PublicIntrProcess       ;控制权转入由 IO2 模块定义的统一中断过程,以
                                        ;便完成指定的功能
995:    DW      Disk_CMDNum             ;存放字节单元的地址,该字节单元存放"块设备驱
                                        ;动程序支持的最大命令码"
996: IO1_Interrupt      ENDP
997: ;=========================================================
998: ;相对地址偏移:0743
999: ;功能:通过将控制权转入 IO2 模块来调用对应的处理子程序,以便完成指定的功能
1000: ;=========================================================
```

```
1001: PublicIntrProcess        PROC     NEAR
1002:     CMP      BYTE PTR CS:AllowHMA,0        ;┐ 若 IO2 模块和 MSDOS2 模块被安装在常规内存
1003:     JE       PubIntrP_1                    ;┘ 低端,则跳转
1004:     PUSH     AX                            ;┐
1005:     MOV      AX,CS:MSDOS1_SEG              ;├ 若当前"执行设备驱动程序请求"不是 DOS
1006:     CMP      WORD PTR CS:DRH_Ptr+2,AX      ;├ kernel 模块发出的,则跳转
1007:     POP      AX                            ;├
1008:     JNZ      PubIntrP_2                    ;┘
1009: PubIntrP_1:
1010:     JMP      DWORD PTR CS:PreEntry3        ;控制权转入 IO2 模块提供的设备驱动程序的统一
                                                 ;中断过程
1011: PubIntrP_2:
1012:     CALL     Enable1_A20                   ;局部使能 A20 地址线
1013:     JMP      SHORT PubIntrP_1
1014: PublicIntrProcess        ENDP
1015: ;=========================================================
1016: ;相对地址偏移:0762
1017: ;功能: INT 29H 中断服务程序
1018: ;=========================================================
1019: INT29H_ISR PROC     FAR
1020:     PUSH     AX
1021:     PUSH     SI
1022:     PUSH     DI
1023:     PUSH     BP
1024:     PUSH     BX
1025:     MOV      AH,0EH                        ;┐
1026:     MOV      BX,7                          ;├ 以电传方式显示 AL 指定的字符
1027:     INT      10H                           ;┘
1028:     POP      BX
1029:     POP      BP
1030:     POP      DI
1031:     POP      SI
1032:     POP      AX
1033:     IRET
1034: INT29H_ISR ENDP
1035: ;=========================================================
1036: ;相对地址偏移:0774
1037: ;功能: INT 13H 中断服务程序
1038: ;=========================================================
1039: INT13H_ISR PROC     FAR
1040:     CMP      BYTE PTR CS:AllowHMA,0        ;┐ 若 IO2 模块和 MSDOS2 模块被安装在常规内存
1041:     JE       INT13H_ISR_1                  ;┘ 低端,则跳转
1042:     CALL     CheckA20                      ;检查 A20 地址线的状态
```

```
1043:     JNZ     INT13H_ISR_1                    ;若 A20 地址线已被使能,则跳转
1044:     CALL    Enable2_A20                     ;局部使能 A20 地址线
1045: INT13H_ISR_1:
1046:     MOV     WORD PTR CS:DSValue,DS          ;┐
1047:     PUSHF                                   ;| 调用由 IO2 模块定义的 INT 13H 中断服务程
1048:     CALL    DWORD PTR CS:PreEntry6          ;| 序,完成指定的功能
1049:     MOV     DS,WORD PTR CS:DSValue          ;┘
1050:     RETF    2                               ;中断返回
1051: ;=========================================================
1052: ;相对地址偏移:0797
1053: ;功能: INT 13H 中断服务程序,它供 IO2 模块调用,以便与先前的 INT 13H 中断服务程序链接起来
1054: ;=========================================================
1055: IO1  INT13H_ISR   LABEL     NEAR
1056:     MOV     DS,WORD PTR DSValue             ;┐
1057:     PUSHF                                   ;| 调用先前的 INT 13H 中断服务程序
1058:     CALL    DWORD PTR CS:INT13H_Shift_Entry ;|
1059:     MOV     WORD PTR CS:DSValue,DS          ;┘
1060:     PUSH    CS
1061:     POP     DS
1062:     PUSHF                                   ;保护 INT 13H 中断返回的标志寄存器值
1063:     CMP     BYTE PTR CS:AllowHMA,0          ;┐ 若 IO2 模块和 MSDOS2 模块被安装在常规内存
1064:     JE      INT13H_ISR_2                    ;┘ 低端,则跳转
1065:     CALL    CheckA20                        ;检查 A20 地址线的状态
1066:     JNZ     INT13H_ISR_2                    ;若 A20 地址线已被使能,则跳转
1067:     CALL    Enable2_A20                     ;局部使能 A20 地址线
1068: INT13H_ISR_2:
1069:     POPF
1070:     RETF
1071: INT13H_ISR ENDP
1072: ;
1073: HMAAddress   DW      0090H                   ;HMA 的内存地址,它表示 HMA 的相对段内偏移
                                                  ;值 0080H
1074:             DW      -1
1075: VectorAddress DW     0080H                   ;它表示常规内存低端的内存地址 0000:0080H
1076:             DW      0
1077: ;=========================================================
1078: ;相对地址偏移:07C3
1079: ;功能:局部使能 A20 地址线。若 A20 地址线先前已被使能,则直接返回
1080: ;入口参数:无
1081: ;出口参数:无
1082: ;=========================================================
1083: Enable1_A20      PROC    NEAR
1084:     CALL    CheckA20                        ;检查 A20 地址线的状态
```

```
1085:     JZ        Enable2_A20                    ;若 A20 地址线已被禁止,则跳转
1086:     RET
1087: Enable1_A20ENDP
1088: ;=========================================================================
1089: ;相对地址偏移:07C9
1090: ;功能:局部使能 A20 地址线
1091: ;入口参数:无
1092: ;出口参数:无
1093: ;=========================================================================
1094: Enable2_A20       PROC      NEAR
1095:     PUSH      AX
1096:     PUSH      BX
1097:     MOV       AH,5                           ;┐调用由 HIMEM.SYS 提供的扩充内存管理程
1098:     CALL      CS:HIMEMAddr                   ;┘序,局部使能 A20 地址线
1099:     POP       BX
1100:     POP       AX
1101:     RET
1102: Enable2_A20       ENDP
1103: ;=========================================================================
1104: ;相对地址偏移:07D5
1105: ;功能:检查 A20 地址线的状态
1106: ;入口参数:无
1107: ;出口参数:ZF=0,A20 地址线已被使能;ZF=1,A20 地址线已被禁止
1108: ;=========================================================================
1109: CheckA20    PROC      NEAR
1110:     PUSH      DS
1111:     PUSH      ES
1112:     PUSH      CX
1113:     PUSH      SI
1114:     PUSH      DI
1115:     LDS       SI,DWORD PTR CS:HMAAddress
1116:     LES       DI,DWORD PTR CS:VectorAddress
1117:     MOV       CX,8
1118:     REPE      CMPSW
1119:     POP       DI
1120:     POP       SI
1121:     POP       CX
1122:     POP       ES
1123:     POP       DS
1124:     RET
1125: CheckA20    ENDP
1126: ;=========================================================================
1127: ;相对地址偏移:07E5
```

```
;功能:局部禁止 A20 地址线
;入口参数:无
;出口参数:无
;=============================================================
DisableA20  PROC    NEAR
    PUSH    AX
    PUSH    BX
    MOV     AH,6                    ;┐ 调用由 HIMEM.SYS 提供的扩充内存管理程
    CALL    CS:HIMEMAddr            ;┘ 序,局部禁止 A20 地址线
    POP     BX
    POP     AX
    RET
DisableA20  ENDP
;=============================================================
;相对地址偏移:07FB
;功能:INT 19H 中断服务程序
;=============================================================
INT19H_ISR      PROC    FAR
    PUSH    CS
    POP     DS
    MOV     ES,ZeroValue            ;ES←0(中断向量表的段地址)
    MOV     CX,5                    ;CX←5(在热启动前应恢复的中断向量个数)
    MOV     SI,offset INT10H  Number    ;DS:SI 指向中断向量的数据块
INT19H_ISR_1:
    LODSB                           ;┐ AX←即将恢复中断向量的中断号
    CBW                             ;┘
    SHL     AX,1                    ;┐
    SHL     AX,1                    ;│ ES:DI=保存中断向量的双字单元地址
    MOV     DI,AX                   ;┘
    LODSW                           ;┐
    STOSW                           ;│ 恢复中断向量
    LODSW                           ;│
    STOSW                           ;┘
    LOOP    INT19H_ISR_1
    CMP     StackFlag,0             ;┐ 若没有因指定动态堆栈而接管中断服务程序,
    JE      INT19H_ISR_4            ;┘ 则跳转
    MOV     SI,offset HardIntVectorArray    ;┐
    MOV     CX,0EH                  ;│
INT19H_ISR_2:                       ;│
    LODSB                           ;│
    CBW                             ;│
    MOV     DI,AX                   ;│
    LODSW                           ;│
```

```
1171:    MOV     BX,AX                          ;|
1172:    LODSW                                  ;|
1173:    CMP     BX,-1                          ;| 恢复 INT 02H、INT 08H～INT 0EH、INT 70H、
1174:    JE      INT19H_ISR_3                   ;| INT 72H～INT 74H、INT 76H～INT 77H
1175:    CMP     AX,-1                          ;| 的中断向量
1176:    JE      INT19H_ISR_3                   ;|
1177:    ADD     DI,DI                          ;|
1178:    ADD     DI,DI                          ;|
1179:    XCHG    AX,BX                          ;|
1180:    STOSW                                  ;|
1181:    XCHG    AX,BX                          ;|
1182:    STOSW                                  ;|
1183: INT19H_ISR_3:                             ;|
1184:    LOOP    INT19H_ISR_2                   ;┘
1185: INT19H_ISR_4:
1186:    CMP     BYTE PTR AllowHMA,0            ;┐ 若 DOS 操作系统没有使用 HMA(即 IO2 模块、
                                                ;| MSDOS2 模块和盘缓冲区建在常规内存),则跳
1187:    JE      INT19H_ISR_5                   ;┘ 转
1188:    CALL    ClearStartingHMA               ;破坏 HMA 内存区的起始 32 个字节值,为热启动
                                                ;建立正常的工作环境
1189: INT19H_ISR_5:
1190:    INT     19H                            ;重新启动 DOS 操作系统(热启动)
1191: INT19H_ISR        ENDP
1192: ;=========================================================
1193: ;相对地址偏移:084A
1194: ;功能: INT 15H 中断服务程序
1195: ;=========================================================
1196: INT15H_ISR        PROC    FAR
1197:    CMP     AX,4F53H
1198:    JE      INT15H_ISR_1
1199:    JMP     CS:Old_INT15H_Vector           ;控制权转入先前的 INT 15H 中断服务程序
1200: INT15H_ISR_1:
1201:    PUSH    DS
1202:    PUSH    AX
1203:    MOV     AX,40H                         ;┐
1204:    MOV     DS,AX                          ;| AL 中保存 Alt、Ctrl 键的状态
1205:    MOV     AL,DS:Info_17H                 ;|
1206:    AND     AL,0CH                         ;┘
1207:    CMP     AL,0CH                         ;┐ 若 Alt、Ctrl 键未按下或未同时按下,则跳转
1208:    JNE     INT15H_ISR_2                   ;┘
1209:    PUSH    CS                             ;┐
1210:    POP     DS                             ;| 若 DOS 未使用 HMA,则跳转
1211:    CMP     BYTE PTR AllowHMA,0            ;|
```

```
1212:    JE      INT15H_ISR_2                    ;┘
1213:    CALL    ClearStartingHMA                ;清 HMA 内存区的起始 32 个字节值,它保存了
                                                 ;HIMEM.SYS 的安装标志
1214: INT15H_ISR_2:
1215:    POP     AX
1216:    POP     DS
1217:    STC                                     ;CF←1
1218:    JMP     CS:OLD_INT15H_Vector            ;控制权转入先前的 INT 15H 中断服务程序
1219: INT15H_ISR         ENDP
1220: ;=============================================================
1221: ;相对地址偏移:0878
1222: ;功能:清 HMA 内存区的起始 32 个字节值,它保存了 HIMEM.SYS 的安装标志。清除它是为了热启
      动时 DOS 能使用 HMA
1223: ;=============================================================
1224: ClearStartingHMA     PROC    NEAR
1225:    PUSH    AX
1226:    PUSH    CX
1227:    PUSH    DI
1228:    PUSH    ES
1229:    CALL    Enable1_A20                     ;局部使能 A20 地址线
1230:    MOV     AX,0FFFFH                       ;┐
1231:    MOV     ES,AX                           ;│
1232:    MOV     DI,10H                          ;├ 清除 HMA 内存区的起始 32 个字节值
1233:    MOV     CX,10H                          ;│
1234:    XOR     AX,AX                           ;│
1235:    REP     STOSW                           ;┘
1236:    POP     ES
1237:    POP     DI
1238:    POP     CX
1239:    POP     AX
1240:    RET
1241: ClearStartingHMA     ENDP
1242: ;=============================================================
1243: ;相对地址偏移:0893
1244: ;功能:程序转换接口
1245: ;=============================================================
1246: ShiftInterface     PROC    FAR
1247:    JMP     DWORD PTR CS:PreEntry5          ;控制权转入 IO2 模块中的 INT 2FH 中断服务程
                                                 ;序(即多用途程序接口的中断服务程序)
1248: BlkDevIntrEntry   LABEL   NEAR             ;块设备驱动程序的中断过程的入口
1249:    JMP     Disk_Intr
1250: RetfEntry    LABEL   NEAR                  ;长返回的接口
1251:    RETF
```

```
1252: ShiftInterface ENDP
1253: ;相对地址偏移:089C
1254: D_089C      DW      3
1255: D_089E      DW      0
1256: D_08A0      DW      0
1257:             DB      8 DUP (0)
1258:             DB      0AEH, 8, 70H, 0, 0, 0
1259:             DB      50H, 0, 2, 0, 0EH, 0, 50H, 0
1260:             DB      14H, 0, 0CH, 0, 70H, 0, 1, 0
1261: Ptr_SearchSCB DD     0                    ;保存"下次可供分配的动态堆栈控制块的指针值"
1262:             DB      2, 0
1263: Ptr_SCBArray DD     0                    ;保存动态堆栈控制块数组的起始地址
1264: SizeOfDynamicStk    DW   0               ;保存动态堆栈占用的总内存字节数(包括动态堆
                                               ;栈的控制块和动态堆栈本身)
1265:             DB      0,0,0,0
1266: D_08D0      DB      0
1267: ;=============================================================
1268: ;相对地址偏移:08D1
1269: ;功能:未知,它的功能与 DOS 内核无关
1270: ;=============================================================
1271: SUB_08D1   PROC    FAR
1272:     PUBLIC  SUB_08D1
1273:     PUSH    DI
1274:     PUSH    ES
1275:     PUSH    BX
1276:     PUSH    AX
1277:     XOR     DI,DI
1278:     MOV     ES,DI
1279:     MOV     BX,0015H
1280:     MOV     AX,1684H
1281:     INT     2FH
1282:     MOV     AX,ES
1283:     OR      AX,DI
1284:     JZ      L_08F2
1285:     PUSH    CS
1286:     MOV     AX,offset L_08F2
1287:     PUSH    AX
1288:     PUSH    ES
1289:     PUSH    DI
1290:     MOV     AX,1
1291:     RETF
1292: L_08F2:
1293:     POP     AX
```

```
1294:       POP     BX
1295:       POP     ES
1296:       POP     DI
1297:       RET
1298: SUB_08D1  ENDP
1299: ;相对地址偏移:08F7
1300: FreeMemOFSofHMA   DW      -1          ;存放高内存区(HMA)的自由(未被使用)内存区的
                                            ;起始地址偏移值
1301: PreEntry7         DW      0A84H       ;┐ IO3 模块中的子程序的入口地址,该子程序的
                                            ;┤ 功能是"试图在 HMA 安装 IO2 模块和 MSDOS2
1302:                   DW      46DH        ;┘ 模块,然后确定 HMA 自由内存区的起始地址"
1303: IO3_Flag          DB      0           ;IO3 模块有效标志 (0:IO3 模块不存在,即 IO3 模
                                            ;块已执行完;1:IO3 模块存在,即 IO3 模块正在执
                                            ;行)
1304: ;相对地址偏移:08FE
1305: FreeAddress1      EQU     $
1306:     DW      0                         ;未使用
1307: DefaultVolID      DB      'NO NAME',0 ;缺省的卷标字符串
1308: VolumeIDBuffer    DB      'NO NAME',0 ;存放驱动器上当前介质的卷标名的缓冲区
1309: ;相对地址偏移:0918
1310: FreeAddress2      EQU     $
1311:     DB      80H
1312: ;相对地址偏移:0919
1313: ;以下定义硬盘的基本 DOS 分区和扩充 DOS 分区对应的逻辑驱动器的 BDPB
1314: ExtendedBDPB1 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1315: ;
1316: ExtendedBDPB2 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1317: ;
1318: ExtendedBDPB3 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1319: ;
1320: ExtendedBDPB4 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1321: ;
1322: ExtendedBDPB5 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1323: ;
1324: ExtendedBDPB6 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1325: ;
1326: ExtendedBDPB7 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
```

```
      0,,,,,1,0,,,>
1327: ;
1328: ExtendedBDPB8 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1329: ;
1330: ExtendedBDPB9 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1331: ;
1332: ExtendedBDPB10 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1333: ;
1334: ExtendedBDPB11 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1335: ;
1336: ExtendedBDPB12 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1337: ;
1338: ExtendedBDPB13 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1339: ;
1340: ExtendedBDPB14 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1341: ;
1342: ExtendedBDPB15 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1343: ;
1344: ExtendedBDPB16 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1345: ;
1346: ExtendedBDPB17 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1347: ;
1348: ExtendedBDPB18 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1349: ;
1350: ExtendedBDPB19 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
1351: ;
1352: ExtendedBDPB20 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,>
1353: ;
1354: ExtendedBDPB21 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
      0,,,,,1,0,,,>
```

```
;
ExtendedBDPB22 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
    0,,,,,1,0,,,>
;
ExtendedBDPB23 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
    0,,,,,1,0,,,>
;
ExtendedBDPB24 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
    0,,,,,1,0,,,>
;
ExtendedBDPB25 BDPB <,0,50H,3,,1,1,2,10H,0,0F8H,1,0,0,,,,,,,,0,0,0,0,0,0,0,0,0,
    0,,,,,1,0,,,>
;
    DB      0
;=============================================================
;相对地址偏移:12DE
;功能:INT 13H 中断接管程序,它对应于 AT 机的 ROM BIOS 1/10/84 版本
;入口参数:无
;出口参数:CF=0,成功;CF=1,失败
;=============================================================
INT13H_ILR1     PROC    FAR
    CMP     DL, 80H                     ;┐ 若是对软盘进行操作,则跳转
    JB      INT13H_ILR1_1               ;┘
    CMP     AH, 2                       ;┐ 若是读硬盘操作,则跳转
    JZ      INT13H_ILR1_2               ;┘
    CMP     AH, 0AH                     ;┐ 若是"读长扇区"操作,则跳转
    JZ      INT13H_ILR1_2               ;┘
INT13H_ILR1_1:
    JMP     CS:Old_INT13H_Vector        ;控制权转到先前的 INT 13H 中断服务程序
INT13H_ILR1_2:
    PUSH    BX
    PUSH    CX
    PUSH    DX
    PUSH    DI
    PUSH    DS
    PUSH    ES
    PUSH    AX
    MOV     AX,40H                      ;┐ DS←BIOS 通讯区的段地址值
    MOV     DS,AX                       ;┘
    MOV     BYTE PTR DS:Info_74H,0      ;清硬盘操作的状态(0:无错误)
    AND     DL,7FH                      ;┐
    CMP     DL,BYTE PTR DS:Info_75H     ;├ 若存在指定的硬盘驱动器,则跳转
    JB      INT13H_ILR1_3               ;┘
```

```
1394:    MOV     BYTE PTR DS:Info_74H,1          ;设置硬盘操作的状态(1:指定的硬盘驱动器无效)
1395:    JMP     Short INT13H_ILR1_4
1396: INT13H_ILR1_3:
1397:    PUSH    BX                              ;┐
1398:    MOV     AX,ES                           ;│
1399:    SHR     BX,04                           ;│转换用户程序指定的传送缓冲区的地址值格
1400:    ADD     AX,BX                           ;│式,使得能对整个内存段进行存取而不发生段
1401:    MOV     ES,AX                           ;│溢出错
1402:    POP     BX                              ;│
1403:    AND     BX,0FH                          ;┘
1404:    PUSH    CS                              ;┐检查 DMA 是否会越界溢出
1405:    CALL    ROMBIOS6                        ;┘
1406:    JB      INT13H_ILR1_4                   ;若 DMA 会越界溢出,则跳转
1407:    POP     AX
1408:    PUSH    AX
1409:    CALL    BuildCDB                        ;建立"命令数据块"
1410:    MOV     DX,03F6H                        ;┐置附加磁头任选
1411:    OUT     DX,AL                           ;┘
1412:    CALL    SendCDB                         ;向硬盘控制器发送"命令数据块",并读取指定个
                                                 ;数的扇区内容
1413: INT13H_ILR1_4:
1414:    POP     AX
1415:    MOV     AH,DS:Info_74H                  ;┐
1416:    OR      AH,AH                           ;│若硬盘"读操作"成功,则跳转
1417:    JZ      INT13H_ILR1_5                   ;┘
1418:    STC                                     ;CF←1(失败)
1419: INT13H_ILR1_5:
1420:    POP     ES
1421:    POP     DS
1422:    POP     DI
1423:    POP     DX
1424:    POP     CX
1425:    POP     BX
1426:    RETF    2
1427: INT13H_ILR1        ENDP
1428: ;=============================================================
1429: ;相对地址偏移:1346
1430: ;功能:建立"命令数据块"
1431: ;入口参数:AL=欲读取的扇区数;DS=0040H(BIOS 通讯区的段地址值)
1432: ;出口参数:AL=硬盘参数表的"控制字节值"
1433: ;=============================================================
1434: BuildCDB                PROC    NEAR
1435:    MOV     BYTE PTR DS:Info_43H,AL         ;设置"欲读取的扇区数"
```

```
1436:     MOV     BYTE PTR DS:Info_48H,20H      ;设置命令码(20H:读)
1437:     CMP     AH,2                          ;┐若当前硬盘操作为"读操作",则跳转
1438:     JE      BuildCDB_1                    ;┘
1439:     MOV     BYTE PTR DS:Info_48H,22H      ;设置命令码(22H:读长扇区)
1440: BuildCDB_1:
1441:     MOV     AL,CL                         ;┐
1442:     AND     AL,3FH                        ;│设置"道内扇区号"
1443:     MOV     BYTE PTR DS:Info_44H,AL       ;┘
1444:     MOV     BYTE PTR DS:Info_45H,CH       ;┐
1445:     MOV     AL,CL                         ;│设置"圆柱面号"
1446:     SHR     AL,6                          ;│
1447:     MOV     BYTE PTR DS:Info_46H,AL       ;┘
1448:     MOV     AX,DX                         ;┐
1449:     SHL     AL,4                          ;│
1450:     AND     AH,0FH                        ;│设置"驱动器号和磁头号"
1451:     OR      AL,AH                         ;│
1452:     OR      AL,0A0H                       ;│
1453:     MOV     BYTE PTR DS:Info_47H,AL       ;┘
1454:     PUSH    ES
1455:     PUSH    BX
1456:     PUSH    CS                            ;┐取硬盘参数表的起始地址→ES:BX
1457:     CALL    ROMBIOS1                      ;┘
1458:     MOV     AX,ES:[BX+5]                  ;┐
1459:     SHR     AX,2                          ;│取置"写预补偿圆柱面号"
1460:     MOV     BYTE PTR DS:Info_42H,AL       ;┘
1461:     MOV     AL,ES:[BX+8]                  ;AL←硬盘参数表的"控制字节"值
1462:     POP     BX
1463:     POP     ES
1464:     MOV     AH,DS:Info_76H                ;┐
1465:     AND     AH,0C0H                       ;│设置新的"控制字节"值
1466:     OR      AH,AL                         ;│
1467:     MOV     DS:Info_76H,AH                ;┘
1468:     RET
1469: BuildCDB    ENDP
1470: ;=============================================================
1471: ;相对地址偏移:139E
1472: ;功能:向硬盘控制器发送"命令数据块",并读取指定个数的扇区内容
1473: ;入口参数:DS=0040H(BIOS 通讯区的段地址值)
1474: ;          ES:BX 指向用户指定的传送缓冲区
1475: ;出口参数:无
1476: ;=============================================================
1477: SendCDB    PROC    NEAR
1478:     MOV     DI,BX
```

```
1479:     PUSH    CS                          ;┐ 向硬盘控制器发送"命令数据块"
1480:     CALL    ROMBIOS2                    ;┘
1481:     JNZ     SendCDB_RET                 ;若控制器有故障,则直接返回
1482: SendCDB_1:
1483:     PUSH    CS                          ;┐ 等待数据请求中断
1484:     CALL    ROMBIOS3                    ;┘
1485:     JNZ     SendCDB_RET                 ;若设置超时,则直接返回
1486:     MOV     CX,100H                     ;        ┐
1487:     MOV     DX,1F0H                     ;        │
1488:     CLD                                 ;清除方式标志 │
1489:     CLI                                 ;        │从硬盘数据寄存器
1490: SendCDB_2:                             ;        │端口中读取1个扇
1491:     INSW                                ;        │区的数据(512字节)
1492:     LOOP    SendCDB_2                   ;        │
1493:     STI                                 ;        ┘
1494:     TEST    BYTE PTR DS:Info_48H,2      ;┐ 若不是"读长扇区"操作,则跳转
1495:     JZ      SendCDB_3                   ;┘
1496:     PUSH    CS                          ;┐ 等待数据请求
1497:     CALL    ROMBIOS4                    ;┘
1498:     JC      SendCDB_RET                 ;若设置超时,则直接返回
1499:     MOV     CX,4                        ;┐
1500:     MOV     DX,1F0H                     ;│从硬盘数据寄存器端口中读取4个字节的ECC
1501:     CLI                                 ;│数据
1502:     REP     INSB                        ;│
1503:     STI                                 ;┘
1504: SendCDB_3:
1505:     PUSH    CS                          ;┐ 检查硬盘状态
1506:     CALL    ROMBIOS5                    ;┘
1507:     JNZ     SendCDB_RET                 ;若有错误,则直接返回
1508:     DEC     BYTE PTR DS:Info_43H        ;┐ 若"欲读取的扇区数"未读完,则转去继续读取
1509:     JNZ     SendCDB_1                   ;┘ 指定的扇区内容
1510: SendCDB_RET:
1511:     RET
1512: SendCDB    ENDP
1513: ;=============================================================
1514: ;相对地址偏移:13DC
1515: ;功能:请求调用ROM BIOS的子程序,这些子程序是ROM BIOS的INT 13H中断服务程序的组成部
          分
1516: ;=============================================================
1517: CALL_ROMBIOS      PROC    NEAR
1518: ;功能:取硬盘参数表的起始地址
1519: ;入口:DL=物理驱动器号(80H:第1个硬盘;81H:第2个硬盘)
1520: ;出口:ES:BX指向对应的硬盘参数表
```

```
1521; ROMBIOS1  LABEL    NEAR                    ;13DC
1522;    PUSH    0FF65H                          ;压入长返回指令的地址偏移
1523;    DB      0EAH                            ;┐
1524;    DW      2F8EH, ROM_SEG                  ;├ 控制权转入 ROM BIOS
1525;    ;JMP    FAR 0F000H:2F8EH                ;┘
1526; ;功能:向硬盘控制器发送"命令数据块"
1527; ;入口:40:42H～40:48H 存放了"命令数据块"(参见 1428～1469)
1528; ;出口:ZF=1,成功;ZF=0,失败
1529; ROMBIOS2  LABEL    NEAR                    ;13E4
1530;    PUSH    0FF65H                          ;压入长返回指令的地址偏移
1531;    DB      0EAH                            ;┐ 控制权转入 ROM BIOS
1532;    DW      2E1EH, ROM_SEG                  ;├
1533;    ;JMP    0F000H:2E1EH                    ;┘
1534; ;功能:等待数据请求中断
1535; ;入口:无
1536; ;出口:ZF=1,有中断;ZF=0,设备超时
1537; ROMBIOS3  LABEL    NEAR                    ;13EC
1538;    PUSH    0FF65H                          ;压入长返回指令的地址偏移
1539;    DB      0EAH                            ;┐
1540;    DW      2E7FH, ROM_SEG                  ;├ 控制权转入 ROM BIOS
1541;    ;JMP    0F000H:2E7FH                    ;┘
1542; ;功能:等待数据请求
1543; ;入口:无
1544; ;出口:CF=0,有数据请求;CF=0,设备超时
1545; ROMBIOS4  LABEL    NEAR                    ;13F4
1546;    PUSH    0FF65H                          ;压入长返回指令的地址偏移
1547;    DB      0EAH                            ;┐
1548;    DW      2EE2H, ROM_SEG                  ;├ 控制权转入 ROM BIOS
1549;    ;JMP    0F000H:2EE2H                    ;┘
1550; ;功能:检查硬盘状态
1551; ;入口:无
1552; ;出口:ZF=0,无错误;ZF=0,有错误
1553; ROMBIOS5  LABEL    NEAR                    ;13FC
1554;    PUSH    0FF65H                          ;压入长返回指令的地址偏移
1555;    DB      0EAH                            ;┐
1556;    DW      2EF8H, ROM_SEG                  ;├ 控制权转入 ROM BIOS
1557;    ;JMP    0F000H:2EF8H                    ;┘
1558; ;功能:检查 DMA 是否会越界溢出(即超越 64KB 边界)
1559; ;入口:40:42H～40:48H 存放了"命令数据块"(参见 1428～1469)
1560; ;出口:CF=0,DMA 操作不会越界;CF=1,DMA 操作会越界
1561; ROMBIOS6  LABEL    NEAR                    ;1404
1562;    PUSH    0FF65H                          ;压入长返回指令的地址偏移
1563;    DB      0EAH                            ;┐
```

```
1564:     DW      2F69H, ROM_SEG                ;| 控制权转入 ROM BIOS
1565:     ;JMP    0F000H:2F69H                  ;┘
1566: CALL_ROMBIOS        ENDP
1567: TailOfINT13H_ILR1   EQU       $
1568: ;=============================================================
1569: ;相对地址偏移:140C
1570: ;功能:INT13H 中断接管程序,它对应于安装 1986 年 8 月 4 号以前开发的 ROM BIOS 的 COMPQA
          机器
1571: ;=============================================================
1572: INT13H_ILR2    PROC    FAR
1573:     CMP     AH,15H                        ;┐ 若功能号>15H,则跳转
1574:     JA      INT13H_ILR2_2                 ;┘
1575: INT13H_ILR2_1:
1576:     JMP     CS:Old_INT13H_Vector          ;控制权转入先前的 INT 13H 中断服务程序
1577: INT13H_ILR2_2:
1578:     CMP     DL,80H                        ;┐ 若是对软盘驱动器操作,则跳转
1579:     JB      INT13H_ILR2_1                 ;┘
1580:     PUSH    DS
1581:     PUSH    AX                            ;┐
1582:     MOV     AX,40H                        ;| DS←BIOS 通讯区的段地址值
1583:     MOV     DS,AX                         ;|
1584:     POP     AX                            ;┘
1585:     PUSHF                                 ;┐ 调用先前的 INT 13H 中断服务程序
1586:     CALL    CS:Old_INT13H_Vector          ;┘
1587:     POP     DS
1588:     RETF    2                             ;中断返回
1589: INT13H_ILR2    ENDP
1590: TailOfINT13H_ILR2   EQU     $
1591:     DB      0
1592: ;=============================================================
1593: ;相对地址偏移:142E
1594: ;功能:将二进制表示的日期数据转换成 BCD 码格式
1595: ;入口参数:无
1596: ;出口参数:CH=BCD 码格式的世纪号
1597: ;          CL=BCD 码格式的年号
1598: ;          DH=BCD 码格式的月份号
1599: ;          DL=BCD 码格式的日期号
1600: ;=============================================================
1601: BINtoBCD    PROC    FAR
1602:     PUSH    CS:DayCount                   ;保护从 1980 年 1 月 1 号起至今的总天数
1603:     CMP     WORD PTR CS:DayCount,7305     ;┐ 若世纪号为 20,则跳转
1604:     JNB     BIN_BCD1                      ;┘
1605:     MOV     BYTE PTR CS:Century, 19       ;设置世纪号
```

```
1606:     MOV     BYTE PTR CS:Year, 80          ;设置起始年号
1607:     JMP     SHORT BIN_BCD2
1608: BIN_BCD1:
1609:     MOV     BYTE PTR CS:Century, 20       ;设置世纪号
1610:     MOV     BYTE PTR CS:Year, 0           ;设置起始年号
1611:     SUB     WORD PTR CS:DayCount,7305     ;修改日期数据
1612: BIN_BCD2:
1613:     XOR     DX, DX                        ;┐
1614:     MOV     AX, CS:DayCount               ;|
1615:     MOV     BX, 1461                      ;|
1616:     DIV     BX                            ;|
1617:     MOV     CS:DayCount, DX               ;|
1618:     MOV     BL, 4                         ;|
1619:     MUL     BL                            ;|
1620:     ADD     CS:Year, AL                   ;|
1621:     INC     WORD PTR CS:DayCount          ;|
1622:     CMP     WORD PTR CS:DayCount,366      ;| 计算年号
1623:     JBE     BIN_BCD4                      ;|
1624:     INC     BYTE PTR CS:Year              ;|
1625:     SUB     WORD PTR CS:DayCount,366      ;|
1626:     MOV     CX, 3                         ;|
1627: BIN_BCD3:                                 ;|
1628:     CMP     CS:DayCount,365               ;|
1629:     JBE     BIN_BCD5                      ;|
1630:     INC     CS:Year                       ;|
1631:     SUB     CS:DayCount,365               ;|
1632:     LOOP    BIN_BCD3                      ;┘
1633: BIN_BCD4:
1634:     MOV     BYTE PTR CS:Febmary,29        ;修改二月份的天数值
1635: BIN_BCD5:
1636:     XOR     BX,BX                         ;┐
1637:     XOR     DX,DX                         ;|
1638:     MOV     AX,CS:DayCount                ;|
1639:     MOV     SI, offset DayPerMonth        ;|
1640:     MOV     CX,0CH                        ;|
1641: BIN_BCD6:                                 ;|
1642:     INC     BL                            ;| 计算月份号→BL
1643:     MOV     DL,[SI]                       ;| 计算日期号→AX
1644:     CMP     AX,DX                         ;|
1645:     JBE     BIN_BCD7                      ;|
1646:     INC     SI                            ;|
1647:     SUB     AX,DX                         ;|
1648:     LOOP    BIN_BCD6                      ;┘
```

```
1649: BIN_BCD7:
1650:     MOV     BYTE PTR CS:Febmary,28          ;设置缺省的二月份的天数
1651:     MOV     DL,BL                           ;DL←月份号
1652:     MOV     DH,CS:Year                      ;DH←年号
1653:     MOV     CL,CS:Century                   ;CL←世纪号
1654:     CALL    DWORD PTR CS:PreEntry1          ;┐计算日期号的 BCD 码格式数据→DL
1655:     XCHG    DL,AL                           ;┘
1656:     CALL    DWORD PTR CS:PreEntry1          ;┐计算月份号的 BCD 码格式数据→DH
1657:     XCHG    DH,AL                           ;┘
1658:     CALL    DWORD PTR CS:PreEntry1          ;┐计算年号的 BCD 码格式数据→CL
1659:     XCHG    CL,AL                           ;┘
1660:     CALL    DWORD PTR CS:PreEntry1          ;┐计算世纪号的 BCD 码格式数据→CH
1661:     MOV     CH,AL                           ;┘
1662:     POP     CS:DayCount                     ;恢复从 1980 年 1 月 1 号起至今的总天数
1663:     RET
1664: BINtoBCD    ENDP
1665: TailOfBINtoBCD        EQU     $
1666: ;=========================================================
1667: ;相对地址偏移:14FF
1668: ;功能:将二进制数转换成十进制 BCD 码数据
1669: ;入口参数:AL=二进制数
1670: ;出口参数:AL=十进制数
1671: ;=========================================================
1672: ConvertData  PROC    FAR
1673:     PUSH    CX
1674:     AAM
1675:     MOV     CL, 4
1676:     SHL     AH, CL
1677:     OR      AL, AH
1678:     POP     CX
1679:     RET
1680: ConvertData            ENDP
1681: TailOfConvertData      EQU     $
1682: ;=========================================================
1683: ;相对地址偏移:150A
1684: ;功能:INT 6CH 中断服务程序(设置时钟计数值)
1685: ;=========================================================
1686: INT6CH_ISR            PROC    FAR
1687:     PUSH    CS                              ;┐
1688:     POP     DS                              ;│保存中断返回地址,同时还弹出标志寄存器的
1689:     POP     ReturnAddr                      ;│值
1690:     POP     ReturnAddr+2                    ;│
1691:     POPF                                    ;┘
```

```
1692:    CALL    Calcuate_DayCount              ;┐
1693:    CLI                                    ;| 计算当前日期对应的总天数→SI
1694:    MOV     DayCount, SI                   ;|
1695:    STI                                    ;┘
1696:    CALL    Read_Clock2                    ;读实时钟时间,并转换成时钟计数→CX:DX 中
1697:    CLI                                    ;┐
1698:    MOV     AH, 1                          ;| 设置时钟计数
1699:    INT     1AH                            ;|
1700:    STI                                    ;┘
1701:    JMP     DWORD PTR ReturnAddr           ;转到原返回地址处执行
1702: INT6CH_ISR          ENDP
1703: ;=============================================================
1704: ;相对地址偏移:152B
1705: ;功能:读取实时钟日期,并计算当前日期对应的总天数
1706: ;入口参数:无
1707: ;出口参数:CF=0,SI=日期对应的总天数和;
1708: ;          CF=1,SI=实时钟日期操作的错误码(1=无实时钟工作,2=数据格式错,3=数据值错)
1709: ;=============================================================
1710: Calcuate_DayCount  PROC     NEAR
1711:    PUSH    AX                             ;┐
1712:    PUSH    CX                             ;|
1713:    PUSH    DX                             ;|
1714:    XOR     AH,AH                          ;| 读取时钟计数值
1715:    INT     1AH                            ;|
1716:    POP     DX                             ;|
1717:    POP     CX                             ;|
1718:    POP     AX                             ;┘
1719:    PUSH    AX
1720:    PUSH    BX
1721:    PUSH    CX
1722:    PUSH    DX
1723:    MOV     WORD PTR CS:ClockFlag,1        ;置无实时钟工作的错误码
1724:    MOV     AH,4                           ;┐
1725:    INT     1AH                            ;| 读实时钟日期,若实时钟工作,则跳转
1726:    JNB     Cal_Day1                       ;┘
1727:    JMP     Cal_Day6                       ;若实时钟不工作,则跳转
1728: Cal_Day1:
1729:    MOV     RealTimeRetPara1,CH            ;保存 BCD 码的世纪号
1730:    MOV     RealTimeRetPara2,CL            ;保存 BCD 码的年号
1731:    MOV     RealTimeRetPara3,DH            ;保存 BCD 码的月份号
1732:    MOV     RealTimeRetPara4,DL            ;保存 BCD 码的日期号
1733:    MOV     WORD PTR CS:ClockFlag,2        ;置实时钟日期数据格式错的错误码
1734:    CALL    Test_BCDData                   ;┐ 若实时钟日期数据不是 BCD 码格式,则跳转
```

```
1735:    JC      Cal_Day6                        ;┘
1736:    MOV     WORD PTR CS:ClockFlag,3         ;置实时钟日期数据值错的错误码
1737:    CALL    Test_DayData                    ;┐若实时钟日期数据值错误,则跳转
1738:    JC      Cal_Day6                        ;┘
1739:    MOV     WORD PTR CS:ClockFlag,0         ;清实时钟日期操作的错误码
1740:    CALL    Time_To_ClockCount              ;将 BCD 码格式表示的日期数据转换成二进制格式
1741:    MOV     AL,RealTimeRetPara2             ;┐ AX←年号
1742:    CBW                                     ;┘
1743:    CMP     RealTimeRetPara1,20             ;┐若世纪号不为 20,则跳转
1744:    JNE     Cal_Day2                        ;┘
1745:    ADD     AX,100                          ;修改年号
1746: Cal_Day2:
1747:    SUB     AX,80                           ;┐
1748:    MOV     CL,4                            ;|
1749:    DIV     CL                              ;|
1750:    MOV     BL,AH                           ;|计算年号对应的日期总天数→AX
1751:    CBW                                     ;|
1752:    MOV     CX,1461                         ;|
1753:    MUL     CX                              ;|
1754:    MOV     WORD PTR CS:ClockFlag,AX        ;┘
1755:    MOV     AL,BL                           ;┐
1756:    CBW                                     ;|若年号能被 4 整除,则跳转
1757:    OR      AX,AX                           ;|
1758:    JZ      Cal_Day3                        ;┘
1759:    MOV     CX,365                          ;┐
1760:    MUL     CX                              ;|累计上不足四年的年号日期天数和
1761:    ADD     WORD PTR CS:ClockFlag,AX        ;┘
1762:    JMP     SHORT Cal_Day4
1763: Cal_Day3:                                  ;二月份为 29 天
1764:    CMP     RealTimeRetPara3,2              ;┐若月份小于 3 月,则跳转
1765:    JBE     Cal_Day5                        ;┘
1766: Cal_Day4:
1767:    INC     WORD PTR CS:ClockFlag           ;因为二月份为 29 天,所以总日期天数加 1
1768: Cal_Day5:
1769:    MOV     CL,RealTimeRetPara4             ;┐
1770:    XOR     CH,CH                           ;|日期天数加上日期号
1771:    DEC     CX                              ;|
1772:    ADD     WORD PTR CS:ClockFlag,CX        ;┘
1773:    MOV     CL,RealTimeRetPara3             ;┐
1774:    XOR     CH,CH                           ;|
1775:    DEC     CX                              ;|
1776:    SHL     CX,1                            ;|查表得出当前月份号以前各月的天数和→AX
1777:    MOV     SI, offset TotalOfDay_Table     ;|
```

```
1778:     ADD     SI,CX                          ;|
1779:     MOV     AX,[SI]                        ;┘
1780:     ADD     WORD PTR CS:ClockFlag,AX       ;加上月份对应的日期天数
1781: Cal_Day6:
1782:     MOV     SI,WORD PTR CS:ClockFlag       ;SI←实时钟日期操作的错误码或当前日期对应的
                                                 ;总天数
1783:     POP     DX
1784:     POP     CX
1785:     POP     BX
1786:     POP     AX
1787:     RET
1788: Calcuate_DayCount    ENDP
1789: ;=============================================================
1790: ;相对地址偏移:15E6
1791: ;功能:将实时钟时间转换成时钟计数值
1792: ;入口参数:无
1793: ;出口参数:CX:DX=时钟计数值
1794: ;=============================================================
1795: Read_ClockCount      PROC    NEAR
1796: Read_Clock1:                               ;将时钟计数值清零
1797:     XOR     CX, CX
1798:     XOR     DX, DX
1799:     JMP     SHORT Read_Clock_RET
1800: Read_Clock2      LABEL    NEAR              ;将实时钟时间转换成时钟计数值
1801:     MOV     AH, 2                          ;┐
1802:     INT     1AH                            ;| 若无实时钟工作,则跳转
1803:     JB      Read_Clock1                    ;┘
1804:     MOV     RealTimeRetPara1, CH           ;保存 BCD 码格式的小时
1805:     MOV     RealTimeRetPara2, CL           ;保存 BCD 码格式的分
1806:     MOV     RealTimeRetPara3, DH           ;保存 BCD 码格式的秒
1807:     MOV     BYTE PTR RealTimeRetPara4, 0   ;置百分秒为 0
1808:     CALL    Test_BCDData                   ;┐ 若实时钟时间数据不是 BCD 码格式,则跳转
1809:     JB      Read_Clock1                    ;┘
1810:     CALL    Test_TimeData                  ;┐ 若实时钟时间数据值错,则跳转
1811:     JB      Read_Clock1                    ;┘
1812:     CALL    Time_To_ClockCount             ;将 BCD 码格式的时间数据转换成二进制格式
1813:     MOV     CH, RealTimeRetPara1           ;CH←小时
1814:     MOV     CL, RealTimeRetPara2           ;CL←分
1815:     MOV     DH, RealTimeRetPara3           ;DH←秒
1816:     MOV     DL, RealTimeRetPara4           ;DL←百分秒
1817:     CALL    DWORD PTR PreEntry4            ;将时间化成时钟计数值→CX:DX
1818: Read_Clock_RET:
1819:     RET
```

```
Read_ClockCount          ENDP
;=============================================================
;相对地址偏移:1625
;功能:将实时钟的 BCD 码格式的日期/时间数据转换成对应的二进制值
;入口参数:无
;出口参数:无
;=============================================================
Time_To_ClockCount  PROC    NEAR
    MOV     AL,RealTimeRetPara1          ;┐
    CALL    BCDtoBIN                     ;| 转换世纪号/小时
    MOV     RealTimeRetPara1, AL         ;┘
    MOV     AL, RealTimeRetPara2         ;┐
    CALL    BCDtoBIN                     ;| 转换年号/分
    MOV     RealTimeRetPara2, AL         ;┘
    MOV     AL, RealTimeRetPara3         ;┐
    CALL    BCDtoBIN                     ;| 转换月份/秒
    MOV     RealTimeRetPara3, AL         ;┘
    MOV     AL, RealTimeRetPara4         ;┐
    CALL    BCDtoBIN                     ;| 转换日期号/百分秒
    MOV     RealTimeRetPara4, AL         ;┘
    RET
Time_To_ClockCount  ENDP
;=============================================================
;相对地址偏移:164A
;功能:将 BCD 码格式的数据转换成二进制格式的数据
;入口参数:AL=BCD 码数据
;出口参数:AL=二进制数据
;=============================================================
BCDtoBIN    PROC    NEAR
    MOV     AH, AL
    AND     AL, 0FH
    MOV     CL, 4
    SHR     AH, CL
    AAD
    RET
BCDtoBIN    ENDP
;=============================================================
;相对地址偏移:1655
;功能:判定 BCD 码格式的日期数据值是否正确
;入口参数:无
;出口参数:CF=0,数据值正确;CF=1,数据值错误
;=============================================================
Test_DayData            PROC    NEAR
```

```
1863:     CMP     BYTE PTR RealTimeRetPara1,20H  ;┐ 若世纪号>20,则出错返回
1864:     JA      Test_DayD2                     ;┘
1865:     JZ      Test_DayD1                     ;若世纪号为 20,则跳转
1866:     CMP     BYTE PTR RealTimeRetPara1,19H  ;┐ 若世纪号<19,则出错返回
1867:     JB      Test_DayD2                     ;┘
1868:     CMP     BYTE PTR RealTimeRetPara2,80H  ;┐ 若年号<1980,则出错返回
1869:     JB      Test_DayD2                     ;┘
1870: Test_DayD1:                                ;世纪号、年号正确
1871:     CMP     BYTE PTR RealTimeRetPara2,99H  ;┐ 若号>99,则出错返回
1872:     JA      Test_DayD2                     ;┘
1873:     CMP     BYTE PTR RealTimeRetPara3,12H  ;┐ 若月份>12,则出错返回
1874:     JA      Test_DayD2                     ;┘
1875:     CMP     BYTE PTR RealTimeRetPara3,0    ;┐ 若月份≤0,则出错返回
1876:     JBE     Test_DayD2                     ;┘
1877:     CMP     BYTE PTR RealTimeRetPara4,31H  ;┐ 若日期号>31,则出错返回
1878:     JA      Test_DayD2                     ;┘
1879:     CMP     BYTE PTR RealTimeRetPara4,0    ;┐ 若日期≤0,则出错返回
1880:     JBE     Test_DayD2                     ;┘
1881:     CLC                                    ;置数据值正确标志(CF=0)
1882:     RET
1883: Test_DayD2:
1884:     STC                                    ;置数据值错误标志(CF=1)
1885:     RET
1886: Test_DayData        ENDP
1887: ;=============================================================
1888: ;相对地址偏移:1693
1889: ;功能:判定 BCD 码格式的时间数据值是否正确
1890: ;入口参数:无
1891: ;出口参数:CF=0,数据值正确;CF=1,数据值错误
1892: ;=============================================================
1893: Test_TimeData        PROC    NEAR
1894: Test_TimeD1:
1895:     CMP     BYTE PTR RealTimeRetPara1, 24H ;┐ 若小时>24,则出错返回
1896:     JA      Test_TimeD2                    ;┘
1897:     CMP     BYTE PTR RealTimeRetPara2, 59H ;┐ 若分>59,则出错返回
1898:     JA      Test_TimeD2                    ;┘
1899:     CMP     BYTE PTR RealTimeRetPara3, 59H ;┐ 若秒>59,则出错返回
1900:     JA      Test_TimeD2                    ;┘
1901:     CLC                                    ;置数据值正确标志(CF=0)
1902:     RET
1903: Test_TimeD2:
1904:     STC                                    ;置数据值出错标志(CF=1)
1905:     RET
```

```
1906: Test_TimeData      ENDP
1907: ;=============================================================
1908: ;相对地址偏移:16AC
1909: ;功能:判定实时钟日期/时间操作返回的数据是否是 BCD 码格式
1910: ;入口参数:无
1911: ;出口参数:CF=0,格式正确;CF=1,格式错
1912: ;=============================================================
1913: Test_BCDData      PROC    NEAR
1914:     MOV     CX,4                          ;数据个数
1915:     MOV     BX, offset RealTimeRetPara1
1916: Test_BCDD1:
1917:     MOV     AL,[BX]                       ;┐取出当前要测试的数据值
1918:     MOV     AH,AL                         ;┘
1919:     AND     AX,0F00FH                     ;┐
1920:     CMP     AL,0AH                        ;├若低位不是 BCD 码格式,则出错返回
1921:     JA      Test_BCDD2                    ;┘
1922:     SHR     AH,1                          ;┐
1923:     SHR     AH,1                          ;│
1924:     SHR     AH,1                          ;│
1925:     SHR     AH,1                          ;├若高位不是 BCD 码格式,则出错返回
1926:     AND     AH,0FH                        ;│
1927:     CMP     AH,0AH                        ;│
1928:     JA      Test_BCDD2                    ;┘
1929:     INC     BX                            ;┐
1930:     DEC     CX                            ;├若数据未检测完,则转去继续测试下一个数据
1931:     JNZ     Test_BCDD1                    ;┘
1932:     CLC                                   ;置数据格式正确标志(CF=0)
1933:     RET
1934: Test_BCDD2:
1935:     STC                                   ;置数据格式错标志(CF=1)
1936:     RET
1937: Test_BCDData      ENDP
1938: ;
1939: TailOfINT6CH_ISR EQU     $
1940:     NOP
1941: ;相对地址偏移:16D6
1942: BootDriveNo          DB    0             ;保存引导盘对应的逻辑驱动器号(0=A,2=C,…)
1943: BootMediaDescriptor  DB    0             ;保存引导盘的介质描述符
1944: SectorNumber_Init    DD    0             ;存放引导盘数据区的起始扇区号,或引导盘数据
                                               ;区的相对起始扇区号(减去了隐含扇区数)
1945: Sectors_Init         DW    0             ;在读取 MSDOS.SYS 系统文件时,它存放当前允许
                                               ;读取的扇区数
1946: FATType_Init         DB    0             ;文件系统类型码(bit7=1,簇数≥65536;bit6=1,
```

```
                                        ;16 位 FAT 表,bit6=0,12 位 FAT)
1947: Seg_FATBuffer     DW   0          ;在 DOS 操作系统启动时,它存放"读取 FAT 表所
                                        ;使用的传送缓冲区的段地址值"
1948: Seg_DPTBuffer     DW   0          ;在 DOS 操作系统启动时,它存放"读取硬盘分区
                                        ;信息表所使用的传送缓冲区的段地址值"
1949: DriveNo_Init      DB   80H        ;在建立硬盘驱动器的 BDPB 时,它暂存物理驱动
                                        ;器号
1950: BytesPerSector_Init DW 200H       ;保存引导盘的每扇区字节数
1951: ClusterNo_Init    DW   0          ;当簇项值位于相邻两个扇区时,它存放簇项值的
                                        ;高、低字节值
1952: SectorNoOfFAT_Init DW  -1         ;保存上次读取的 FAT 表内容对应的起始扇区号
1953: Heads_Init        DB   2          ;存放软盘驱动器支持的"磁头数"
1954: SectorsPerTrack_Init DB 9         ;存放软盘驱动器支持的"每道扇区数"
1955: Cylinders_Init    DB   28H        ;存放软盘驱动器支持的"最大圆柱面数"
1956: FDI_Flag_IO1      DB   0          ;软盘驱动器安装标志(0:安装了软驱;1:没有安装
                                        ;软驱)
1957: ;相对地址偏移:16EE
1958:     DB     0, 2, 0, 1, 40H, 0, 0      ;┐
1959:     DB     0, 0, 8, 1, 2, 70H,0       ;│
1960:     DB     0, 0, 0, 20H, 2, 4, 0      ;│ 无用数据
1961:     DB     1, 0, 0, 0A8H,7FH, 3, 8    ;│
1962:     DB     0, 2, 0, 0, -1, -1, 4      ;│
1963:     DB     10H, 0, 4, 0, 0            ;┘
1964: ;相对地址偏移:1716
1965: ;硬盘分区参数表数组,它由 7 个硬盘分区参数表组成,其中每一个表长 10 个字节,格式如下:
1966: ;第一项,双字表示的分区最大可用扇区数(低字在后,高字在前);
1967: ;第二项,字表示每簇扇区数的幂指数值(移位因子);
1968: ;第三项,字节表示每簇扇区数
1969: ;第四项,字表示最大根目录项数
1970: ;第五项,特标志字节(bit6=1,表示 FAT 表每项长 16 位,否则为 12 位)
1971: ;第六项,字节,保留待用
1972: FixedDiskPara  DW    0,07FA8H      ;┐
1973:     DB     3,8                      ;│ 扇区数≤32680
1974:     DW     200H                     ;│
1975:     DB     0, 0                     ;┘
1976:     DW     4,0                      ;┐
1977:     DB     2, 4                     ;│ 32680<扇区数≤262144
1978:     DW     200H                     ;│
1979:     DB     40H,0                    ;┘
1980:     DW     8,0                      ;┐
1981:     DB     3, 8                     ;│ 262144<扇区数≤524288
1982:     DW     200H                     ;│
1983:     DB     40H,0                    ;┘
```

```
1984:     DW      10H,0                    ;┐
1985:     DB      4, 10H                   ;| 524288<扇区数≤1048576
1986:     DW      200H                     ;|
1987:     DB      40H,0                    ;┘
1988:     DW      20H,0                    ;┐
1989:     DB      5, 20H                   ;| 1048 76<扇区数≤2097152
1990:     DW      200H                     ;|
1991:     DB      40H,0                    ;┘
1992:     DW      40H,0                    ;┐
1993:     DB      6, 40H                   ;| 2097152<扇区数≤4194304
1994:     DW      200H                     ;|
1995:     DB      40H,0                    ;┘
1996:     DW      80H,0                    ;┐
1997:     DB      7, 80H                   ;| 4194304<扇区数≤8388608
1998:     DW      200H                     ;|
1999:     DB      40H,0                    ;┘
2000: ;相对地址偏移:175C
2001: DriveNumber_Init   DB    0          ;当前硬盘驱动器的物理驱动器号
2002: FixedDiskNumber    DB    0          ;存放系统配置的硬盘驱动器数
2003: Ptr_BPBArray       DW    BPB_Ptr_Array  ;BPB 参数块指针数组的访问(写入)指针
2004: Ptr_FreeBDPB       DW    ExtendedBDPB1  ;空闲(可供使用)的 BDPB 的指针
2005: HeadsOfFixed_Init  DW    0          ;当前硬盘驱动器的磁头数
2006: SectorsPerTrackOfFixed  DW    0     ;当前硬盘的每道扇区数
2007: ROMBIOS_Version    DB    '01/10/84',0   ;ROM BIOS 版本标志串
2008:                    DB    90H
2009: ;相对地址偏移:1770
2010: DriveType0    DW    512            ;┐
2011:     DB      2                      ;|
2012:     DW      1                      ;|
2013:     DB      2                      ;|
2014:     DW      112                    ;|
2015:     DW      2D0H                   ;| 软盘驱动器类型 0(160KB/180KB 或 320KB/
2016:     DB      0FDH                   ;| 360KB 5.25 英寸软盘)对应的 Default BPB
2017:     DW      2                      ;| 参数块
2018:     DW      9                      ;|
2019:     DW      2                      ;|
2020:     DD      0                      ;|
2021:     DD      0                      ;┘
2022:     DB      90H
2023: ;相对地址偏移:178A
2024: DriveType1    DW    512            ;┐
2025:     DB      1                      ;|
2026:     DW      1                      ;|
```

```
2027:     DB      2                              ;|
2028:     DW      224                            ;|
2029:     DW      960H                           ;| 软盘驱动类型 1(1.2MB 5.25 英寸软盘)对应
2030:     DB      0F9H                           ;| 的 Default BPB 参数块
2031:     DW      7                              ;|
2032:     DW      0FH                            ;|
2033:     DW      2                              ;|
2034:     DD      0                              ;|
2035:     DD      0                              ;┘
2036:     DB      90H
2037: ;相对地址偏移:17A4
2038: DriveType2    DB      512                  ;┐
2039:     DB      2                              ;|
2040:     DW      1                              ;|
2041:     DB      2                              ;|
2042:     DW      112                            ;|
2043:     DW      5A0H                           ;| 软盘驱动器类型 2(720KB 3.5 英寸软盘)对应
2044:     DB      0F9H                           ;| 的 DefaultBPB 参数块
2045:     DW      3                              ;|
2046:     DW      9                              ;|
2047:     DW      2                              ;|
2048:     DD      0                              ;|
2049:     DD      0                              ;┘
2050:     DB      90H
2051: ;相对地址偏移:17BE
2052: DriveType9    DW      512                  ;┐
2053:     DB      2                              ;|
2054:     DW      1                              ;|
2055:     DB      2                              ;|
2056:     DW      240                            ;|
2057:     DW      1680H                          ;| 软盘驱动器类型 9(2.88MB 3.5 英寸软盘)对应
2058:     DB      0F0H                           ;| 的 Default BPB 参数块
2059:     DW      9                              ;|
2060:     DW      36                             ;|
2061:     DW      2                              ;|
2062:     DD      0                              ;|
2063:     DD      0                              ;┘
2064:     DB      90H
2065: ;相对地址偏移:17D8
2066: Ptr_BPB_Array  DW      DriveType0           ;┐
2067:     DW      DriveType1                     ;|
2068:     DW      DriveType2                     ;|
2069:     DW      DriveType2                     ;|
```

```
2070:     DW      DriveType2                    ;| 各类型的软盘驱动器对应的 Default BPB 参数
2071:     DW      DriveType2                    ;| 块指针数组
2072:     DW      DriveType2                    ;|
2073:     DW      DriveType2                    ;|
2074:     DW      DriveType2                    ;|
2075:     DW      DriveType9                    ;┘
2076: OFS_RetfIO2    DW      IO2_Inter3        ;存放 IO2 模块中 RETF 指令的偏移地址
2077: ;=============================================================
2078: ;相对地址偏移:17EE
2079: ;功能:调用由 BP 指定的在 IO2 模块中定义的子程序
2080: ;入口参数:BP=子程序的偏移地址
2081: ; 其它参数参见具体被调用的子程序
2082: ;出口参数:参见具体被调用的子程序
2083: ;=============================================================
2084: CallSubInIO2        PROC    NEAR
2085:     PUSH    CS:OFS_RetfIO2                ;压入长返回指令的偏移地址
2086:     PUSH    WORD PTR CS:PreEntry3+2       ;┐
2087:     PUSH    BP                            ;| 控制权转入 IO2 模块中由 BP 指定的子程序
2088:     RETF                                  ;┘
2089: CallSubInIO2 ENDP
2090: ;相对地址偏移:17FA
2091: NumberOfFDisk2  DB  0                     ;保存系统配置的软盘驱动器数目
2092: ;=============================================================
2093: ;相对地址偏移:17FB
2094: ;功能:系统初始化入口
2095: ;入口参数:AX:BX=引导盘数据区的起始扇区号;
2096: ; CH=引导盘的介质描述符;
2097: ; DL=引导盘的物理驱动器号
2098: ;出口参数:无
2099: ;=============================================================
2100: SysInt 1 Entry      PROC    NEAR
2101:     CLI                                   ;关中断
2102:     PUSH    AX                            ;┐
2103:     XOR     AX,AX                         ;| DS 指向中断向量表
2104:     MOV     DS,AX                         ;|
2105:     POP     AX                            ;┘
2106:     MOV     WORD PTR CS:SectorNumber_Init+2,AX   ;┐ 保存引导盘数据区的起始扇区号
2107:     MOV     WORD PTR CS:SectorNumber_Init,BX     ;┘
2108:     PUSH    CS                            ;┐ ES←IO1 模块在常规内存 ┐
2109:     POP     ES                            ;┘ 低端的段地址值          |
2110:     PUSH    CX                            ;                          |
2111:     PUSH    DI                            ;                          |
2112:     MOV     CX,5                          ;CX←要保存中断向量的中     |
```

```
2113:    MOV    SI,offset INT10H_Number          ;断个数 SI←保存中断向量   |
                                                 ;数据块的起始地址         |
2114: SysInt_I_E_1:                              ;                         |
2115:    LODS   BYTE PTR CS:[SI]                 ;┐ AX←中断号              |
2116:    CBW                                     ;┘                        | 保存 INT 10H、
2117:    SHL    AX,1                             ;┐                        | INT 13H、
2118:    SHL    AX,1                             ;|                        | INT 15H、
2119:    MOV    DI,AX                            ;|                        | INT 19H、
2120:    XCHG   SI,DI                            ;| 保存指定中断号的中断   | INT 1BH
2121:    LODSW                                   ;| 向量                   | 的中断向量
2122:    STOSW                                   ;|                        |
2123:    LODSW                                   ;|                        |
2124:    STOSW                                   ;|                        |
2125:    XCHG   SI,DI                            ;┘                        |
2126:    LOOP   SysInt_I_E_1                     ;                         |
2127:    POP    DI                               ;                         |
2128:    POP    CX                               ;                         ┘
2129:    MOV    AX,WORD PTR CS:Old_INT13H_Vector      ;┐
2130:    MOV    WORD PTR CS:INT13H_Shift_Entry,AX     ;| 设置(补缺)INT 13H 中断接
2131:    MOV    AX,WORD PTR CS:Old_INT13H_Vector+2    ;| 管程序的入口地址
2132:    MOV    WORD PTR CS:INT13H_Shift_Entry+2,AX   ;┘
2133:    MOV    WORD PTR DS:Vector_13H_OFS, offset INT13H_ISR ;┐ 重新设置 INT
2134:    MOV    DS:Vector_13H_SEG,CS                          ;┘ 13H 中断向量
2135:    MOV    WORD PTR DS:Vector_15H_OFS, offset INT15H_ISR ;┐ 重新设置 INT
2136:    MOV    DS:Vector_15H_SEG,CS                          ;┘ 15H 中断向量
2137:    MOV    WORD PTR DS:Vector_19H_OFS, offset INT19H_ISR ;┐ 重新设置 INT
2138:    MOV    DS:Vector_19H_SEG,CS                          ;┘ 19H 中断向量
2139:    STI                                     ;开中断
2140:    INT    11H                              ;取出系统配置的设备信息
2141:    TEST   AX,1                             ;┐ 若安装了软盘驱动器,则跳转
2142:    JNZ    SysInt_I_E_4                     ;┘
2143:    PUSH   AX
2144:    PUSH   BX
2145:    PUSH   CX
2146:    PUSH   DX
2147:    PUSH   DI
2148:    PUSH   ES
2149:    MOV    AH,8                             ;┐
2150:    MOV    DL,0                             ;| 取软盘驱动器参数
2151:    INT    13H                              ;┘
2152:    JC     SysInt_I_E_2
2153:    MOV    CS:NumberOfFDisk2,DL             ;保存系统中安装的软盘驱动器的数目
2154: SysInt_I_E_2:
```

```
2155:    POP     ES
2156:    POP     DI
2157:    POP     DX
2158:    POP     CX
2159:    POP     BX
2160:    POP     AX
2161:    JC      SysInt_I_E_4
2162:    CMP     CS:NumberOfFDisk2,0          ;┐ 若没有安装软盘驱动器,则跳转
2163:    JE      SysInt_I_E_3                 ;┘
2164:    MOV     AL,CS:NumberOfFDisk2         ;┐ AL←软盘驱动器数-1
2165:    DEC     AL                           ;┘
2166:    JMP     SHORT SysInt_I_E_5
2167: SysInt_I_E_3:
2168:    MOV     CS:FDI_Flag_101,1            ;置未安装软盘驱动器的标志
2169:    MOV     AX,1
2170:    JMP     SHORT SysInt_I_E_6
2171: SysInt_I_E_4:
2172:    ROL     AL,1                         ;┐ AL 低两位存放"软盘驱动器数"的信息
2173:    ROL     AL,1                         ;┘
2174: SysInt_I_E_5:
2175:    AND     AX,3                         ;AX=软盘驱动器数-1
2176:    JNZ     SysInt_I_E_7                 ;若有 1 个以上的软盘驱动器,则跳转
2177:    INC     AX                           ;虚设一个 B 软盘驱动器(即 A、B 两个逻辑驱动器
                                              ;共用同一个物理驱动器)
2178: SysInt_I_E_6:
2179:    INC     CS:SingleFDisk               ;设置系统只配置一个软盘驱动器的标志,即 A、B
                                              ;两个逻辑驱动器共用同一物理驱动器的标志
2180: SysInt_I_E_7:
2181:    INC     AX                           ;┐ CL←软盘驱动器数
2182:    MOV     CL,AL                        ;┘
2183:    TEST    DL,80H                       ;┐ 若是硬盘活动分区引导 DOS 操作系统,则跳转
2184:    JNZ     SysInt_I_E_8                 ;┘
2185:    XOR     AX,AX                        ;设置软盘 A 引导 DOS 操作系统的逻辑驱动器号
2186: SysInt_I_E_8:
2187:    XOR     DX,DX
2188:    CLI                                  ;┐
2189:    MOV     SS,DX                        ;| 建立堆栈指针
2190:    MOV     SP,BIO_SEG * 16              ;|
2191:    STI                                  ;┘
2192:    PUSH    CX                           ;保存软盘驱动器数
2193:    MOV     AH,CH                        ;┐ 保存引导盘的介质描述符和引导盘对应的逻辑
                                              ;| 驱动器号(AL=0,软盘引导 DOS 操作系统;否
                                              ;| 则,硬盘引导 DOS 操作系统
```

```
2194:       PUSH    AX                          ;┘
2195:       MOV     AH,0C0H                     ;┐ 取出系统配置参数,若 ROM BIOS 支持该功能,
2196:       INT     15H                         ;┘ 则 ES:BX 指向系统描述符表
2197:       JC      SysInt_1_E_9                ;┐ 若系统 ROM BIOS 不支持该功能(如 XT 档次机
2198:       CMP     AH,0                        ;│ 器),则跳转
2199:       JNE     SysInt_1_E_9                ;┘
2200:       MOV     AL,BYTE PTR ES:[BX+2]       ;┐ 取出并保存"机器型号"
2201:       MOV     CS:ModelByte_Init,AL        ;┘
2202:       MOV     AL,BYTE PTR ES:[BX+3]       ;┐ 取出并保存"机器子型号"
2203:       MOV     CS:SubmodelByte_Init,AL     ;┘
2204:       JMP     SHORT SysInt_1_E_10
2205: SysInt_1_E_9:
2206:       MOV     SI,0FFFFH                   ;┐
2207:       MOV     ES,SI                       ;│ 取出并保存 XT 档次机器的"机器型号"
2208:       MOV     AL,BYTE PTR ES:[000EH]      ;│
2209:       MOV     CS:ModelByte_Init,AL        ;┘
2210: SysInt_1_E_10:
2211:       MOV     AL,20H                      ;┐ 清除中断请求
2212:       OUT     20H,AL                      ;┘
2213:       CMP     CS:ModelByte_Init,0         ;                          ┐
2214:       JNE     SysInt_1_E_11               ;                          │
2215:       IN      AL,66H                      ;                          │
2216:       TEST    AL,20H                      ;                          │
2217:       JZ      SysInt_1_E_11               ;                          │
2218:       MOV     AL,0FH                      ;                          │
2219:       OUT     50H,AL                      ;                          │
2220:       IN      AL,50H                      ;                          │
2221:       TEST    AL,1                        ;                          │ 初始化
2222:       JZ      SysInt_1_E_12               ;                          │ COM1～
2223: SysInt_1_E_11:                           ;                          │ COM4 串
2224:       MOV     AL,3                        ;┐ 初始化 COM4 串行端      │ 行端口
2225:       CALL    Init_COM_Adapter            ;┘ 口                      │
2226:       MOV     AL,2                        ;┐ 初始化 COM3 串行端      │
2227:       CALL    Init_COM_Adapter            ;┘ 口                      │
2228:       MOV     AL,1                        ;┐ 初始化 COM2 串行端      │
2229:       CALL    Init_COM_Adapter            ;┘ 口                      │
2230:       XOR     AL,AL                       ;┐ 初始化 COM1 串行端      │
2231:       CALL    Init_COM_Adapter            ;┘ 口                      │
2232: SysInt_1_E_12:                           ;                          ┘
2233:       MOV     AL,2                        ;┐ 初始化 LPT3 串行端口
2234:       CALL    Init_PRN_Adapter            ;┘
2235:       MOV     AL,1                        ;┐ 初始化 LPT2 串行端口
2236:       CALL    Init_PRN_Adapter            ;┘
```

```
2237:   XOR     AL,AL                           ;┐ 初始化 LPT1 串行端口
2238:   CALL    Init_PRN_Adapter                ;┘
2239:   XOR     DX,DX                           ;┐
2240:   MOV     DS,DX                           ; │
2241:   MOV     ES,DX                           ; │
2242:   XOR     AX,AX                           ; │ 初始化 DOS 通讯区
2243:   MOV     DI,Info_534H                    ; │
2244:   STOSW                                   ; │
2245:   STOSW                                   ;┘
2246:   MOV     AX,CS
2247:   MOV     WORD PTR DS:Vector_1BH_OFS, offset INT1BH_ISR ;┐ 重新设置 INT
2248:   MOV     DS:Vector_1BH_SEG,AX                          ;┘ 1BH 中断向量
2249:   MOV     WORD PTR DS:Vector_29H_OFS, offset INT29H_ISR ;┐ 设置 INT 29H
2250:   MOV     DS:Vector_29H_SEG,AX                          ;┘ 中断向量
2251:   MOV     DI,4
2252:   MOV     BX,offset DummyISR              ;AX:BX 指向虚设的中断服务程序(即该中断服务
                                                ;程序直接中断返回)
2253:   XCHG    AX,BX                           ;┐
2254:   STOSW                                   ; │ 设置 INT 01H 中断向量
2255:   XCHG    AX,BX                           ; │
2256:   STOSW                                   ;┘
2257:   ADD     DI,4
2258:   XCHG    AX,BX                           ;┐
2259:   STOSW                                   ; │ 设置 INT 03H 中断向量
2260:   XCHG    AX,BX                           ; │
2261:   STOSW                                   ;┘
2262:   XCHG    AX,BX                           ;┐
2263:   STOSW                                   ; │ 设置 INT 04H 中断向量
2264:   XCHG    AX,BX                           ; │
2265:   STOSW                                   ;┘
2266:   MOV     DS:Info_500H,DX                 ;清 DOS 通讯区中的控制屏幕打印状态的标志字节
2267:   MOV     DS:Info_504H,DX                 ;在只有一个软盘驱动器的系统中,清 0 表明该软
                                                ;盘驱动器当前作为逻辑驱动器 A
2268:   MOV     AL,DS:Info_52CH                 ;┐ 保存软盘参数表的"马达启动时间"
2269:   MOV     CS:StartupWaitTime,AL           ;┘
2270:   CMP     CS:ModelByte_Init,0FDH          ;┐ 若是 PC/AT 等高档机器,则跳转
2271:   JB      SysInt_I_E_13                   ;┘
2272:   MOV     WORD PTR DS:Info_52BH,20FH      ;修改软盘参数表的"寻道后磁头的稳定时间"和
                                                ;"马达启动时间"
2273:   MOV     BYTE PTR DS:Info_522H,0DFH      ;修改软盘参数表的"步进速率"和"磁头卸载时间"
2274: SysInt_I_E_13:
2275:   INT     12H                             ;取出系统中的常规内存容量→AX(以 1KB 为单
                                                ;位)
```

```
2276:    MOV    CL,6                          ;┐ AX=以节为单位的常规内存容量
2277:    SHL    AX,CL                         ;┘
2278:    POP    CX                            ;┐ 保存引导盘的介质描述符和引导盘对应的逻辑
2279:    MOV    WORD PTR CS:BootDriveNo,CX    ;┘ 驱动器号
2280:    PUSH   AX
2281:    PUSH   DS
2282:    PUSH   BX
2283:    XOR    BX,BX                         ;┐
2284:    MOV    DS,BX                         ;│ DS:BX=INT 2FH 的中断向量
2285:    MOV    BX,WORD PTR DS:Vector_2FH_OFS ;│
2286:    MOV    DS,WORD PTR DS:Vector_2FH_SEG ;┘
2287:    CMP    WORD PTR [BX+3],' PR'         ;
2288:    JNE    SysInt_I_E_14                 ;
2289:    CMP    BYTE PTR [BX+5],' L'          ;
2290:    JNE    SysInt_I_E_14
2291:    MOV    DX,AX
2292:    MOV    AX,4A06H
2293:    INT    2FH
2294:    MOV    AX,DX
2295: SysInt_I_E_14:
2296:    POP    BX
2297:    POP    DS
2298:    SUB    AX,40H                        ;┐ 确定读取 FAT 表时使用的传送缓冲区的段地址
2299:    MOV    CS:Seg_FATBuffer,AX           ;┘ 值(缓冲区为 1KB)
2300:    SUB    AX,40H                        ;┐ 确定读取硬盘分区信息表时使用的传送缓冲区
2301:    MOV    CS:Seg_DPTBuffer,AX           ;┘ 的段地址值(缓冲区为 1KB)
2302:    POP    AX                            ;AX=以节为单位的常规内存容量
2303:    MOV    DX,SysInt_II_SEG              ;┐ DS SysInt—Ⅱ的段地址值
2304:    MOV    DS,DX                         ;┘
2305:    MOV    WORD PTR DS:IO3_0273,offset CON_DriverHeader   ;┐ 设置 SysInt—Ⅱ的变量(设
                                                               ;│ 备驱动程序标题链表头结点
2306:    MOV    WORD PTR DS:IO3_0275,CS                        ;┘ 指针)
2307:    MOV    WORD PTR DS:IO3_0292,AX       ;设置 SysInt—Ⅱ的变量(以节为单位的常规内存容
                                              ;量)
2308:    INC    CL                            ;┐ 设置 SysInt—Ⅱ的变量(引导盘对应的驱动器
2309:    MOV    BYTE PTR DS:IO3_0296,CL       ;┘ 号,1=A,2=B,…)
2310:    MOV    WORD PTR DS:IO3_0271,83FH     ;设置 MSDOS.SYS 系统文件初始读入时在常规内
                                              ;存中的段地址值(83FH=70H+7CF0H/16)
2311:    MOV    AX,SysInt_II_SEG              ;┐ ES←SysInt—Ⅱ的段地址值    ┐
2312:    MOV    ES,AX                         ;┘                          │
2313:    XOR    AX,AX                         ;┐ DS 指向中断向量表          │
2314:    MOV    DS,AX                         ;┘                          │
2315:    MOV    AX,DS:Vector_0FH_SEG          ;┐ 若 INT 0FH 中断向量由程序提 │
```

```
2316:    CMP     AX,FS:IO3_0292          ;┐供,则跳转                    │
2317:    JBE     SysInt_1_E_15           ;┘                            │修改
2318:    CMP     AX,0F000H               ;┐若 INT 0FH 中断向量由扩展卡   │INT 0FH
2319:    JNE     SysInt_1_E_16           ;┘提供,则跳转                  │中断向量
2320: SysInt_1_E_15:                     ;                             │
2321:    MOV     DS:Vector_0FH_OFS,offset Dummy_ISR                     │
2322:    MOV     DS:Vector_0FH_SEG,CS    ;                             │
2323: SysInt_1_E_16:                     ;                             ┘
2324:    XOR     CX,CX                   ;┐
2325:    MOV     DS,CX                   ;│CL←键盘状态字节值
2326:    MOV     CL,DS:Info_496H         ;┘
2327:    TEST    CL,10H                  ;┐若安装的键盘不是 101/102 键盘,则跳转
2328:    JZ      SysInt_1_E_17           ;┘
2329:    MOV     CS:ReadKeyCode,10H      ;设置"读扩展键盘(101/102 键盘)输入"的命令码
2330:    MOV     CS:GetKeyStatus,11H     ;设置"取扩展键盘(101/102 键盘)状态"的命令码
2331: SysInt_1_E_17:
2332:    PUSH    CS
2333:    POP     DS
2334:    PUSH    CS
2335:    POP     ES
2336:    CALL    Process_RTC             ;处理实时时钟,并计算当前日期对应的总天数
2337:    MOV     WORD PTR DS:[0],918H    ;清除 IO1 模块的第一条指令
2338:    POP     AX                      ;AL=软盘驱动器数
2339:    XOR     AH,AH
2340:    MOV     TotalOfBlockDev,AL      ;初置系统允许访问的逻辑驱动器数
2341:    MOV     NumberOfFDisk1,AL       ;保存系统配置的软盘驱动器数
2342:    SHL     AX,1                    ;┐修改 BPB 参数块指针数组的操作指针值,以便
                                         ;│使它等于第一个硬盘的 BPB 参数块指针值的
2343:    ADD     Ptr_BPBArray,AX         ;┘存放地址
2344:    MOV     DL,80H                  ;┐
2345:    MOV     AH,8                    ;│读取硬盘驱动器参数
2346:    INT     13H                     ;┘
2347:    JC      SysInt_1_E_18
2348:    MOV     FixedDiskNumber,DL      ;保存系统配置的硬盘驱动器数
2349: SysInt_1_E_18:
2350:    XOR     DL,DL                   ;DL←逻辑驱动器号(0=A)
2351:    PUSH    CS
2352:    POP     DS
2353:    MOV     MaxSectorsPerTrack,9    ;补缺"每道最大扇区数"值
2354:    MOV     DI,offset BDPB_Head     ;DI=BDPB 链的头结点指针
2355:    CMP     FDI_Flag_IO1,1          ;┐若安装了软盘驱动器,则跳转
2356:    JNE     SysInt_1_E_19           ;┘
2357:    MOV     DI,[DI]                 ;┐DS:DI 指向第一个软盘驱动器对应的 BDPB
```

```
2358:     MOV     DI,[DI]                    ;┘
2359:     MOV     WORD PTR [DI],0FFFFH       ;设置 BDPB 链尾结点的指针域
2360:     JMP     SysInt_I_E_32              ;转去建立初始的硬盘驱动器对应的 BDPB
2361: SysInt_I_E_19:                         ;建立软盘驱动器对应的 BDPB
2362:     CMP     DL,TotalOfBlockDev         ;┐若软盘驱动器对应的 BDPB 未初始化完,则跳转
2363:     JB      SysInt_I_E_20              ;┘
2364:     JMP     SysInt_I_E_31              ;转去建立初始的硬盘驱动器对应的 BDPB
2365: SysInt_I_E_20:
2366:     XOR     CX,CX
2367:     MOV     DI,[DI]                    ;DS:DI 指向当前软盘驱动器对应的 BDPB
2368:     MOV     DH,0
2369:     MOV     Cylinders_Init,40          ;补缺"圆柱面数"
2370:     PUSH    DS
2371:     PUSH    DI
2372:     PUSH    DX
2373:     PUSH    CX
2374:     PUSH    ES
2375:     MOV     AH,8                       ;┐读取 DL 指定的软盘驱动器的参数
2376:     INT     13H                        ;┘
2377:     JC      SysInt_I_E_27              ;若 DL 指定的软盘驱动器无效,则跳转
2378:     CMP     CH,0                       ;┐若最大圆柱面号低 8 位不为 0,则跳转
2379:     JNE     SysInt_I_E_21              ;┘
2380:     MOV     CH,27H                     ;CH←最大圆柱面号
2381:     MOV     CL,9                       ;CL←每道扇区数
2382:     MOV     DH,1                       ;DH←最大磁头号
2383: SysInt_I_E_21:
2384:     INC     DH                         ;DH←磁头数
2385:     INC     CH                         ;CH←圆柱面数
2386:     MOV     Heads_Init,DH              ;保存"磁头数"
2387:     AND     CL,3FH                     ;┐保存"每道扇区数"
2388:     MOV     SectorsPerTrack_Init,CL    ;┘
2389:     MOV     Cylinders_Init,CH          ;保存"圆柱面数"
2390:     CMP     CL,MaxSectorsPerTrack      ;┐
2391:     JBE     SysInt_I_E_22              ;├修改"每道最大扇区数"值
2392:     MOV     MaxSectorsPerTrack,CL      ;┘
2393: SysInt_I_E_22:
2394:     POP     ES
2395:     POP     CX
2396:     POP     DX
2397:     POP     DI
2398:     POP     DS
2399:     MOV     AH,15H                     ;┐读取软盘驱动器的 DASD 类型
2400:     INT     13H                        ;┘
```

```
2401:       JC      SysInt_1_E_23           ;若 ROM BIOS 不支持该功能调用,则跳转
2402:       CMP     AH,2                    ;┐ 若软盘驱动器不支持 Change Line,则跳转
2403:       JNE     SysInt_1_E_23           ;┘
2404:       OR      CL,2                    ;┐ 设置软盘驱动器支持 Change Line 标志
2405:       MOV     ChangeLine,1            ;┘
2406: SysInt_1_E_23:
2407:       CMP     Cylinders_Init,40       ;┐
2408:       JNE     SysInt_1_E_25           ; |
2409:       CMP     SectorsPerTrack_Init,9  ; |
2410:       JBE     SysInt_1_E_28           ; |
2411: SysInt_1_E_24:                        ; |
2412:       MOV     DH,7                    ; |
2413:       JMP     SHORT SysInt_1_E_28     ; | 确定软盘驱动器类型字节值→DH(1:1.2MB
2414: SysInt_1_E_25:                        ; | 5.25 英寸软盘驱;动器 2:720KB 3.5 英寸软盘
2415:       CMP     Cylinders_Init,80       ; | 驱动器;7:1.44MB 3.5 英寸软盘驱动器;9:
2416:       JNE     SysInt_1_E_24           ; | 2.88MB 3.5 英寸软盘驱动器)
2417:       MOV     DH,9                    ; |
2418:       CMP     SectorsPerTrack_Init,36 ; |
2419:       JE      SysInt_1_E_28           ; |
2420:       CMP     SectorsPerTrack_Init,15 ; |
2421:       JE      SysInt_1_E_26           ; |
2422:       CMP     SectorsPerTrack_Init,9  ; |
2423:       JNE     SysInt_1_E_24           ; |
2424:       MOV     DH,2                    ; |
2425:       JMP     SHORT SysInt_1_E_28     ; |
2426: SysInt_1_E_26:                        ; |
2427:       MOV     DH,1                    ; |
2428:       JMP     SHORT SysInt_1_E_28     ;┘
2429: SysInt_1_E_27:
2430:       POP     ES
2431:       POP     CX
2432:       POP     DX
2433:       POP     DI
2434:       POP     DS
2435:       MOV     AH,15H                  ;┐ 读取软盘驱动器的 DASD 类型
2436:       INT     13H                     ;┘
2437:       JC      SysInt_1_E_28           ;若 ROM BIOS 不支持该功能调用,则跳转
2438:       CMP     AH,2                    ;┐ 若软盘驱动器不支持 Change Line,则跳转
2439:       JNE     SysInt_1_E_28           ;┘
2440:       OR      CL,2                    ;┐ 设置软盘驱动器支持 Change Line 标志
2441:       MOV     ChangeLine,1            ;┘
2442:       MOV     Cylinders_Init,80       ;设置"圆柱面数"
2443:       MOV     DH,1                    ;DH←"磁头数"
```

```
2444:     MOV     AL,0FH                          ;AL←"每道扇区数"
2445:     CMP     AL,MaxSectorsPerTrack           ;┐
2446:     JBE     SysInt_I_E_28                   ;| 保存"每道最大扇区数"值
2447:     MOV     MaxSectorsPerTrack,AL           ;┘
2448: SysInt_I_E_28:
2449:     OR      CL,20H                          ;设置 BDPB 处于就绪状态(活动状态)的标志位
2450:     MOV     BH,DL                           ;BH←软盘的物理驱动器号
2451:     CMP     SingleFDisk,2                   ;┐ 若逻辑驱动器 A、B 各自对应一个物理驱动器,
2452:     JNE     SysInt_1_E_29                   ;┘ 则跳转
2453:     DEC     BH                              ;BH=逻辑驱动器 A、B 共用同一物理驱动器时,逻
                                                  ;辑驱动器 B 对应的物理驱动器号
2454:     XOR     CL,20H                          ;清除逻辑驱动器 B 对应的 BDPB 的就绪状态(活
                                                  ;动状态)标志位
2455: SysInt_I_E_29:
2456:     XOR     AX,AX                           ;┐
2457:     MOV     AL,Heads_Init                   ;| "磁头数"→BDPB
2458:     MOV     [DI]. BDPB_NumberOfHead2,AX     ;┘
2459:     MOV     AL,SectorsPerTrack_Init         ;┐ "每道扇区数"→BDPB
2460:     MOV     [DI]. BDPB_SectorsPerTrack2,AX  ;┘
2461:     MOV     [DI]. BDPB_Attr_Status,CX       ;"设备属性和 BDPB 状态标志"→BDPB
2462:     MOV     [DI]. BDPB_DeviceType,DH        ;"设备类型码"→BDPB
2463:     MOV     [DI]. BDPB_LogicalNumber,DL     ;"逻辑驱动器号"→BDPB
2464:     MOV     [DI]. BDPB_DriveNumber,BH       ;"物理驱动器号"→BDPB
2465:     MOV     BL,Cylinders_Init                       ;┐ "圆柱面数"→BDPB
2466:     MOV     BYTE PTR [DI]. BDPB_NumberOfCylinder,BL ;┘
2467:     CMP     SingleFDisk,1                   ;┐ 若逻辑驱动器 A、B 不用共同一物理驱动器,则
2468:     JNE     SysInt_1_E_30                   ;┘ 跳转
2469:     MOV     SingleFDisk,2                   ;更改逻辑驱动器 A、B 共用同一物理驱动器的标志
                                                  ;值,表明下次处理逻辑驱动器 B
2470:     DB      83H,0C9H                        ;┐
2471:     DB      10H                             ;| 设置当前 BDPB 的驱动器标志位(物理驱动器
2472: ;   OR      CX,+10H                         ;| 具有多个逻辑驱动器名)
2473:     OR      [DI]. BDPB_Attr_Status,CX       ;┘
2474:     MOV     DI,[DI]                         ;DS:DI 指向虚拟的逻辑驱动器 B 对应的 BDPB
2475:     INC     DL                              ;逻辑驱动器号加 1
2476:     JMP     SHORT SysInt_I_E_28             ;转去初始化虚拟的逻辑驱动器 B 对应的 BDPB
2477: SysInt_I_E_30:
2478:     INC     DL                              ;逻辑驱动器号加 1
2479:     JMP     SysInt_1_E_19                   ;转去初始化其它逻辑驱动器对应的 BDPB
2480: SysInt_I_E_31:
2481:     MOV     WORD PTR [DI],0FFFFH            ;设置 BDPB 链尾结点的指针域
2482: SysInt_1_E_32:                              ;建立硬盘驱动器对应的 BDPB
2483:     MOV     DH,FixedDiskNumber              ;DH←硬盘驱动器数
```

```
2484:    OR      DH,DH                    ;┐若系统中未安装硬盘,则跳转
2485:    JZ      SysInt_I_E_38            ;┘
2486:    MOV     DL,80H                   ;DL←第1个硬盘的物理驱动器号
2487: SysInt_I_E_33:
2488:    PUSH    DX
2489:    MOV     DI,Ptr_FreeBDPB          ;DS:DI指向当前可供使用的空闲BDPB
2490:    MOV     BL,TotalOfBlockDev       ;BL←逻辑驱动器号
2491:    MOV     BH,0                     ;BH←0(表明建立硬盘基本DOS分区对应的逻辑
                                          ;驱动器的BDPB)
2492:    CALL    Build_FixedDisk_BDPB     ;建立硬盘基本DOS分区对应的逻辑驱动器的
                                          ;BDPB
2493:    JC      SysInt_I_E_34            ;若不能建立BDPB,则跳转
2494:    CALL    CheckLogicDriveNum       ;检测逻辑驱动器是否超过26个(即是否还有空
                                          ;闲BDPB)
2495:    JNC     SysInt_I_E_34            ;若BDPB已全部用完,则跳转
2496:    CALL    App_New_Node             ;将当前BDPB插入到BDPB链尾
2497: SysInt_I_E_34:
2498:    POP     DX
2499:    INC     DL                       ;DL←下一个硬盘的物理驱动器号
2500:    DEC     DH                       ;DH←未建立基本DOS分区对应的BDPB的硬盘
                                          ;驱动器数
2501:    JNZ     SysInt_I_E_33            ;转去建立其它硬盘基本DOS分区对应的逻辑驱动
                                          ;器的BDPB
2502:    CALL    Build_ExtendedDisk_BDPB  ;建立硬盘扩展DOS分区对应的逻辑驱动器的
                                          ;BDPB
2503: ;建立硬盘其它(非第1个)基本DOS分区对应的逻辑驱动器的BDPB
2504:    MOV     DH,FixedDiskNumber       ;DH←硬盘驱动器数
2505:    MOV     DL,80H                   ;DL←第1个硬盘的物理驱动器号
2506: SysInt_I_E_35:
2507:    MOV     BH,1                     ;BH←1(表明建立硬盘其它基本DOS分区对应的
                                          ;逻辑驱动器的BDPB)
2508: SysInt_I_E_36:
2509:    PUSH    DX
2510:    PUSH    BX
2511:    MOV     DI,Ptr_FreeBDPB          ;DS:DI指向当前可供使用的空闲BDPB
2512:    MOV     BL,TotalOfBlockDev       ;BL←逻辑驱动器号
2513:    CALL    Build_FixedDisk_BDPB     ;建立硬盘其它基本DOS分区对应的逻辑驱动器的
                                          ;BDPB
2514:    JC      SysInt_I_E_37            ;若不能建立BDPB,则跳转
2515:    CALL    CheckLogicDriveNum       ;┐若没有其它的空闲BDPB,则跳转
2516:    JNC     SysInt_I_E_37            ;┘
2517:    CALL    App_New_Node             ;将当前BDPB插入到BDPB链尾
2518:    POP     BX
```

```
2519:     POP     DX
2520:     INC     BH                      ;┐ 转去建立后续的其它基本 DOS 分区对应的逻辑
2521:     JMP     SHORT SysInt_1_E_36     ;┘ 驱动器的 BDPB
2522: SysInt_1_E_37:
2523:     POP     BX
2524:     POP     DX
2525:     INC     DL                      ;DL←下一个硬盘的物理驱动器号
2526:     DEC     DH                      ;┐ 转去建立其它硬盘的其它基本 DOS 分区对应的
2527:     JNZ     SysInt_1_E_35           ;┘ 逻辑驱动器的 BDPB
2528:     CMP     NumberOfFDisk1,2        ;┐ 若系统安装的软盘驱动器数≤2,则跳转
2529:     JBE     SysInt_1_E_38           ;┘
2530:     CALL    RedefineDriveLogicName  ;重新编排驱动器的名符
2531: SysInt_1_E_38:
2532:     MOV     DI,Ptr_FreeBDPB         ;                              ┐
2533:     CMP     DI,offset ExtendedBDPB1 ;┐ 若机器配置了硬盘,则跳转       | DI=IO1
2534:     JNE     SysInt_1_E_39           ;┘                             | 模块当
2535:     MOV     DI,offset FreeAddress2  ;                              | 前可被
2536:     CMP     ChangeLine,0            ;                              | 覆盖的
2537:     JNE     SysInt_1_E_39           ;                              | 地址偏
2538:     MOV     DI,offset FreeAddress1  ;                              | 移值
2539: SysInt_1_E_39:                      ;                              ┘
2540:     PUSH    CS
2541:     POP     ES
2542:     CLD
2543:     CMP     ModelByte_Init,0FCH     ;┐ 若不是 AT 档次机器,则跳转
2544:     JNE     SysInt_1_E_40           ;┘
2545:     CMP     FixedDiskNumber,0       ;┐ 若没有安装硬盘,则跳转
2546:     JE      SysInt_1_E_40           ;┘
2547: ;DI=IO1 模块在常规内存中可被覆盖的地址偏移值
2548:     XCHG    AX,DI                   ;┐
2549:     MOV     SI,0F000H               ; |
2550:     MOV     ES,SI                   ; |
2551:     MOV     SI,offset ROMBIOS_Version ; |
2552:     MOV     DI,0FFF5H               ; | 若 ROM BIOS 不是"1984 年 1 月 10 号"的,则
2553:     MOV     CX,9                    ; | 跳转
2554:     REPE    CMPSB                   ; |
2555:     XCHG    AX,DI                   ; |
2556:     JNZ     SysInt_1_E_40           ;┘
2557:     MOV     CX,offset TailOfINT13H_ILR1 ;CX←INT13H 中断接管程序(INT13H_ILR1)的终
                                          ;止地址编移值
2558:     MOV     SI,offset INT13H_ILR1   ;SI←INT13H 中断接管程序(INT13H_ILR1)的起
                                          ;始地址偏移值
2559:     JMP     SHORT SysInt_1_E_42
```

```
2560: SysInt_1_E_40:
2561:     MOV     AX,0F000H                   ;┐ ES←ROM BIOS 的段地址值
2562:     MOV     ES,AX                       ;┘
2563:     CMP     WORD PTR ES:[0FFEAH],' OC' ;┐
2564:     JNE     SysInt_1_E_43               ;│
2565:     CMP     WORD PTR ES:[0FFECH],' PM' ;│ 若不是 COMPQA 机器,则跳转
2566:     JNE     SysInt_1_E_43               ;│
2567:     CMP     WORD PTR ES:[0FFEEH],' QA' ;│
2568:     JNE     SysInt_1_E_43               ;┘
2569:     MOV     AX,ES:[0FFFBH]              ;┐
2570:     XCHG    AH,AL                       ;│
2571:     CMP     AX,' 86'                    ;│
2572:     JA      SysInt_1_E_43               ;│
2573:     JC      SysInt_1_E_41               ;│
2574:     MOV     AX,ES:[0FFF5H]              ;│
2575:     XCHG    AH,AL                       ;│ 若 ROM BIOS 是 86 年 8 月 4 号以后开发的,则
2576:     CMP     AX,' 08'                    ;│ 跳转
2577:     JA      SysInt_1_E_43               ;│
2578:     JC      SysInt_1_E_41               ;│
2579:     MOV     AX,ES:[0FFF8H]              ;│
2580:     XCHG    AH,AL                       ;│
2581:     CMP     AX,' 04'                    ;│
2582:     JAE     SysInt_1_E_43               ;│
2583: SysInt_1_E_41:                          ;┘
2584:     MOV     CX,offset TailOfINT13H_ILR2 ;CX←INT 13H 中断接管程序(INT13_ILR2)的终
                                              ;止地址偏移值
2585:     MOV     SI,offset INT13H_ILR2       ;SI←INT 13H 中断接管程序(INT13_ILR2)的起
                                              ;始地址偏移值
2586: SysInt_1_E_42:
2587:     PUSH    CS                                                    ;┐ 将 INT 13H
2588:     POP     ES                                                    ;│ 中断接管
2589:     MOV     WORD PTR INT13H_Shift_Entry,DI  ;┐ 保存 INT 13H 中断接管程 ;│ 程序移到
                                                  ;│ 序的入口地址          ;│ IO1 模块在
2590:     MOV     WORD PTR INT13H_Shift_Entry+2,CS;┘                    ;│ 常规内存
2591:     SUB     CX,SI                                                 ;│ 中的正确
2592:     REP     MOVSB                                                 ;┘ 位置处
2593: SysInt_1_E_43:
2594:     PUSH    CS
2595:     POP     ES
2596:     CMP     RTCFlag,1                   ;┐ 若机器没有实时时钟,则跳转
2597:     JNE     SysInt_1_E_44               ;┘
2598:     MOV     PreEntry2,DI                ;设置"将二进制表示的日期数据转换成 BCD 码格
                                              ;式"的子程序入口地址偏移值
```

```
2599:    MOV    CX,offset TailOfBINtoBCD - offset BINtoBCD    ;┐ 将 BINtoBCD 子程序移到 IO1 模
2600:    MOV    SI,offset BINtoBCD                            ;│ 块在常规内存中的正确位置处
2601:    REP    MOVSB                                         ;┘
2602:    MOV    PreEntry1,DI                  ;设置"将 AL 指定的二进制数据转换成 BCD 码格
                                              ;式"的子程序入口地址偏移值
2603:    MOV    CX,offset TailOfConvertData - offset ConvertData ;┐ 将 ConvertData 子程序移到 IO1
2604:    MOV    SI,offset ConvertData                         ;│ 模块在常规内存中的正确位置处
2605:    REP    MOVSB                                         ;┘
2606: SysInt_1_E_44:
2607:    PUSH   DI                    ;保存 IO1 模块在常规内存中可被覆盖的地址偏移
                                      ;值
2608:    MOV    AX,4101H              ;AH=41H(等待一个事件);AL=1(事件类型)
2609:    MOV    BL,1                  ;BL←1(超过值)
2610:    MOV    BH,ES:[DI]            ;BH←事件标志
2611:    STC
2612:    INT    15H                   ;请求 INT 15H 中断服务
2613:    POP    DI
2614:    JC     SysInt_1_E_45
2615:    MOV    Support_15H,1         ;设置 ROM BIOS 支持"等待事件"功能请求的标志
                                      ;(1:支持)
2616:    PUSH   DS                    ;┐
2617:    XOR    AX,AX                 ;│
2618:    MOV    DS,AX                 ;│ 设置 INT 6CH 中断向量
2619:    MOV    DS:Vector_6CH_OFS,DI  ;│
2620:    MOV    DS:Vector_6CH_SEG,CS  ;│
2621:    POP    DS                    ;┘
2622:    MOV    SI,offset INT6CH_ISR                          ;┐ 将 INT 6CH 中断服务程序及其
                                                             ;│ 相关程序移到 IO1 模块在常规
2623:    MOV    CX,offset TailOfINT6CH_ISR - offset INT6CH_ISR ;│ 内存中的正确位置处
2624:    REP    MOVSB                                         ;┘
2625: SysInt_1_E_45:
2626:    PUSH   CS
2627:    POP    DS
2628:    ADD    DI,0FH                ;┐
2629:    SHR    DI,1                  ;│
2630:    SHR    DI,1                  ;│
2631:    SHR    DI,1                  ;│ 计算 MSDOS1 模块在常规内存低端的段地址
2632:    SHR    DI,1                  ;│ →DI
2633:    DB     81H,0C7H              ;│
2634:    DW     BIO_SEG               ;│
2635:    ;ADD   DI,0070H              ;┘
2636:    MOV    MSDOS1_SEG,DI         ;保存 MSDOS1 模块在常规内存低端的段地址值
2637:    MOV    AX,WORD PTR BootDriveNo  ;AL←引导盘的逻辑驱动器 ┐
```

```
                                        ;号                          | 取出引导盘对
2638:   MOV     BP,offset Get_BDPB_Node ;                            | 应的 BDPB 入口
2639:   PUSH    CS                      ;                            | 地址→ES:DI
2640:   CALL    CallSubInIO2            ;                            ┘
2641:   MOV     BP,offset Modify_BuildBPB ;┐ 根据引导盘的引导记录参数,设置引导盘对应
2642:   PUSH    CS                      ;| 的 BDPB 的"Build BPB 参数块"参数值
2643:   CALL    CallSubInIO2            ;┘
2644:   PUSH    ES
2645:   POP     DS
2646:   XOR     DI,DI                   ;┐
2647:   MOV     AL,ES:[DI]              ;| 修改引导盘的介质描述符
2648:   MOV     ES:BootMediaDescriptor,AL ;┘
2649:   MOV     AX,WORD PTR ES:BootDriveNo ;AL←引导盘的逻辑驱动器号;AH←引导盘的介质
                                        ;描述符
2650:   PUSH    ES                      ;┐
2651:   PUSH    DS                      ;|
2652:   POP     ES                      ;|
2653:   PUSH    CS                      ;|
2654:   POP     DS                      ;|
2655:   MOV     BP,offset Get_BDPB_Node ;| 取出引导盘对应的 BDPB 入口地址→ES:DI
2656:   PUSH    CS                      ;|
2657:   CALL    CallSubInIO2            ;|
2658:   PUSH    ES                      ;|
2659:   POP     DS                      ;|
2660:   POP     ES                      ;┘
2661:   MOV     BX,[DI].BDPB_BytesPerSector1 ;┐ 取出并保存引导盘的"每扇区字节数"
2662:   MOV     CS:BytesPerSector_Init,BX ;┘
2663:   MOV     BL,[DI].BDPB_SizeFlag   ;┐ 取出并保存引导盘的"文件系统类型码"
2664:   MOV     ES:FATType_Init,BL      ;┘
2665:   MOV     CL,[DI].BDPB_SectorsPerCluster1 ;                    ┐
2666:   MOV     AX,WORD PTR [DI].BDPB_HiddenSectors1 ;┐ 计算引导盘数  |
2667:   SUB     WORD PTR ES:SectorNumber_Init,AX ;| 据区的相对起  | CX←每
2668:   MOV     AX,WORD PTR [DI].BDPB_HiddenSectors1+2 ;| 始扇区号  | 簇扇区
2669:   SBB     WORD PTR ES:SectorNumber_Init+2,AX ;┘               | 数
2670:   XOR     CH,CH                   ;                            ┘
2671:   PUSH    DS                      ;┐
2672:   XOR     DI,DI                   ;|
2673:   MOV     DS,DI                   ;| 取出 MSDOS.SYS 系统文件的起始簇号→BX
2674:   MOV     BX,DS:MSDOS_Cluster     ;|
2675:   POP     DS                      ;┘
2676: Repeat_ReadMSDOS:
2677:   MOV     AX,SysInt_11_SEG        ;┐ ES←MSDOS.SYS 系统文件初始读入时在常规
2678:   MOV     ES,AX                   ;| 内存中的段地址值
```

```
2679:     MOV     ES,ES:IO3_0271                ;┘
2680:     CALL    Read_MSDOS                    ;读取 MSDOS.SYS 系统文件的内容
2681:     TEST    CS:FATType_Init,40H           ;┐
2682:     JNZ     SysInt_ReadDOS1               ;|
2683:     CMP     BX,0FF7H                      ;|
2684:     J2MP    SHORT SysInt_ReadDOS2         ;| 若 MSDOS.SYS 系统文件未读完,则继续读操作
2685: SysInt_ReadDOS1:                          ;|
2686:     CMP     BX,-9                         ;|
2687: SysInt_ReadDOS2:                          ;|
2688:     JB      Repeat_ReadMSDOS              ;┘
2689:     CALL    Build_BDPB_DefaultBPB         ;建立 BDPB 链中所有表结点的 Default BPB 参
                                                ;数块
2690:     DB      0EAH                          ;┐ 控制权转入 SysInt-Ⅱ初始化程序,处理 CON
2691:     DW      SysInt_II_OFS                 ;| FIG.SYS 系统配置文件及加载 COMMAND.
2692:     DW      SysInt_II_SEG                 ;| COM 等工作
2693:     ;JMP    FAR PTR 046D:0267H            ;┘
2694: SysInt_I_Entry        ENDP
2695: ;=============================================================
2696: ;相对地址偏移:1D58
2697: ;功能:当系统配置了两个以上的软盘驱动器时,该子程序重新编排驱动器的逻辑名符(即重新设置
        BDPB 的"逻辑驱动器号"字段值),其编排顺序如下:
2698: ;     1.第一个和第二个软盘驱动器,它们的逻辑驱动器名符分别为 A 和 B;
2699: ;     2.硬盘的基本 DOS 分区,它的逻辑驱动器名符从 C 开始;
2700: ;     3.硬盘的扩展 DOS 分区,它的逻辑驱动器名符紧接着第 2 步;
2701: ;     4.第三个和第三个以上的软盘驱动器,它们的逻辑驱动器名符紧接着第 3 步。
2702: ;     同时,该子程序还重新校正引导盘的逻辑驱动器号
2703: ;入口参数:无
2704: ;出口参数:无
2705: ;=============================================================
2706: RedefineDriveLogicName  PROC    NEAR
2707:     MOV     DI,CS:BDPB_Head               ;DS:DI 指向第 1 个 BDPB
2708: RedefDriveLN_1:
2709:     CMP     [DI].BDPB_DriveNumber,80H     ;┐
2710:     JE      RedefDriveLN_2                ;|
2711:     MOV     DI,[DI]                       ;|
2712:     CMP     DI,-1                         ;| 若系统未安装硬盘,则直接返回
2713:     JNE     RedefDriveLN_1                ;|
2714:     JMP     SHORT RedefDriveLN_RET        ;|
2715: RedefDriveLN_2:                           ;┘
2716:     MOV     AL,2                          ;AL←第 1 个硬盘的起始逻辑驱动器号
2717: RedefDriveLN_3:                           ;┐
2718:     MOV     [DI].BDPB_LogicalNumber,AL    ;|
2719:     MOV     DI,[DI]                       ;| 设置硬盘基本 DOS 分区和扩展 DOS 分区对应
```

```
2720:     INC     AL                              ;| 的 BDPB 的"逻辑驱动器号"字段值
2721:     CMP     DI,--1                          ;|
2722:     JNE     RedefDriveLN_3                  ;┘
2723:     MOV     DI,CS:BDPB_Head                 ;┐ DS:DI 指向第 2 个 BDPB
2724:     MOV     DI,[DI]                         ;┘
2725:     MOV     AH,CS:NumberOfFDisk1            ;┐ AH←软盘驱动器数－2
2726:     SUB     AH,2                            ;┘
2727: RedefDriveLN_4:                            ;┐
2728:     MOV     DI,[DI]                         ;|
2729:     MOV     [DI].BDPB_LogicalNumber,AL      ;| 设置软盘驱动器对应的 BDPB 的"逻辑驱动器
2730:     INC     AL                              ;| 号"字段值
2731:     DEC     AH                              ;|
2732:     JNZ     RedefDriveLN_4                  ;┘
2733:     MOV     AL,CS:BootDriveNo               ;┐
2734:     CMP     AL,2                            ;| 若由 A 驱动器引导 DOS 操作系统,则直接返回
2735:     JB      RedefDriveLN_RET                ;┘
2736:     SUB     AL,CS:NumberOfFDisk1            ;┐
2737:     JC      RedefDriveLN_5                  ;|
2738:     ADD     AL,2                            ;|
2739:     JMP     SHORT RedefDriveLN_6            ;| 校正引导盘的逻辑驱动器号(0=A,1=B,…)
2740: RedefDriveLN_5:                            ;|
2741:     ADD     AL,CS:TotalOfBlockDev           ;|
2742: RedefDriveLN_6:                            ;|
2743:     MOV     CS:BootDriveNo,AL               ;┘
2744:     INC     AL                              ;┐
2745:     PUSH    DS                              ;|
2746:     MOV     DI,SysInt_11_SEG                ;| 设置 SysInt－Ⅱ 的变量(引导盘的驱动器号,1=
2747:     MOV     DS,DI                           ;| A,2=B,…)
2748:     MOV     DS:[03_0296],AL                 ;|
2749:     POP     DS                              ;┘
2750: RedefDriveLN_RET:
2751:     RET
2752: RedefineDriveLogicName    ENDP
2753: ;=============================================================
2754: ;相对地址偏移:1DBD
2755: ;功能:读取硬盘的第一个扇区(即硬盘的主引导记录),并判定它是否是引导记录
2756: ;入口参数:无
2757: ;出口参数:CF=0,是引导记录;CF=1,不是引导记录
2758: ;=============================================================
2759: Read_BootstrapSector1_PROC    NEAR
2760:     MOV     AX,CS:Seg_DPTBuffer             ;┐
2761:     MOV     ES,AX                           ;|
2762:     MOV     BX,200H                         ;|
```

```
2763:    MOV     AX,201H                        ;| 读取硬盘的第一个扇区
2764:    XOR     DH,DH                          ;|
2765:    MOV     CX,1                           ;|
2766:    INT     13H                            ;┘
2767:    JC      Read_BSS1                      ;若读操作失败,则跳转
2768:    CMP     ES:[3FEH],0AA55H               ;┐ 若是引导记录,则直接返回
2769:    JE      Read_BSS_RET                   ;┘
2770: Read_BSS1:
2771:    STC
2772: Read_BSS_RET:
2773:    RET
2774: Read_BootstrapSector1 ENDP
2775: ;=============================================================
2776: ;相对地址偏移:1DDD
2777: ;功能:依据硬盘的 DOS 分区信息和驱动器参数填写对应的 BDPB 中的 Build BPB 参数块
2778: ;入口参数:BL=逻辑驱动器号;
2779: ;        BH=控制字节(0:建立硬盘基本 DOS 分区对应的 BDPB,此时活动分区优先于非活动分
          区;BH≥1:建立硬盘的其它基本 DOS 分区对应的 BDPB);
2780: ;        DL=物理驱动器号;
2781: ;        DS:DI 指向 BDPB
2782: ;出口参数:CF=0,成功建立了一个 BDPB;CF=1,失败,未能建立一个 BDPB
2783: ;=============================================================
2784: Build_FixedDisk_BDPB  PROC    NEAR
2785:    PUSH    DI
2786:    PUSH    BX
2787:    PUSH    DS
2788:    PUSH    ES
2789:    MOV     [DI].BDPB_LogicalNumber,BL          ;"逻辑驱动器号"→BDPB
2790:    MOV     [DI].BDPB_DriveNumber,DL            ;"物理驱动器号"→BDPB
2791:    OR      BYTE PTR [DI].BDPB_Attr_Status,01   ;设置"介质不可装卸(移动)标志位"
2792:    MOV     BYTE PTR [DI].BDPB_DeviceType,5     ;设置"设备类型"为硬盘
2793:    MOV     FATType_Init,0                      ;清文件系统类型码(FAT 表每项长 12 位)
2794:    MOV     DH,BH                               ;DH←控制字节值
2795:    PUSH    DX
2796:    MOV     AH,8                           ;┐ 读硬盘驱动器参数
2797:    INT     13H                            ;┘
2798:    INC     DH                                    ;┐ "磁头数"→BDPB
2799:    MOV     BYTE PTR [DI].BDPB_NumberOfHead1,DH   ;┘
2800:    POP     DX
2801:    JC      Build_FixedDisk_BDPB_7         ;若指定的硬盘驱动器无效,则跳转
2802:    AND     CL,3FH                                ;┐ 每道扇区数→BDPB
2803:    MOV     BYTE PTR [DI].BDPB_SectorsPerTrack1,CL ;┘
2804:    PUSH    DX                             ;┐
```

```
2805:    CALL    Read_BootstrapSector1          ;| 读取硬盘主引导记录
2806:    POP     DX                             ;┘
2807:    JC      Build_FixedDisk_BDPB_7         ;若读操作失败,则跳转
2808:    MOV     BX,3C2H                        ;ES:BX 指向第一个硬盘分区表的"分区类型码"字
                                                ;段
2809: Build_FixedDisk_BDPB_1:
2810:    TEST    BYTE PTR ES:[BX-4],80H         ;┐ 若当前硬盘分区不是活动分区,则跳转
2811:    JE      Build_FixedDisk_BDPB_3         ;┘
2812:    CMP     BYTE PTR ES:[BX],1             ;┐
2813:    JE      Build_FixedDisk_BDPB_2         ;|
2814:    CMP     BYTE PTR ES:[BX],4             ;|
2815:    JE      Build_FixedDisk_BDPB_2         ;| 若当前硬盘活动分区不是基本 DOS 分区,则跳
2816:    CMP     BYTE PTR ES:[BX],6             ;| 转
2817:    JNE     Build_FixedDisk_BDPB_3         ;|
2818: Build_FixedDisk_BDPB_2:                   ;┘
2819:    OR      DH,DH                          ;┐ 若是建立硬盘活动 DOS 分区对应的 BDPB,则跳
2820:    JE      Build_FixedDisk_BDPB_8         ;┘ 转
2821:    DEC     DH
2822: Build_FixedDisk_BDPB_3:
2823:    ADD     BX,10H                         ;ES:BX 指向下一个硬盘分区表
2824:    CMP     BX,402H                        ;┐ 若还有其它硬盘分区,则继续
2825:    JNE     Build_FixedDisk_BDPB_1         ;┘
2826:    MOV     BX,3C2H                        ;ES:BX 指向第一个硬盘分区表的"分区类型码"字
                                                ;段
2827: Build_FixedDisk_BDPB_4:
2828:    TEST    BYTE PTR ES:[BX-4],80H         ;┐ 若当前硬盘分区是活动分区,
2829:    JNE     Build_FixedDisk_BDPB_6         ;┘ 则跳转
2830:    CMP     BYTE PTR ES:[BX],1             ;┐
2831:    JE      Build_FixedDisk_BDPB_5         ;|
2832:    CMP     BYTE PTR ES:[BX],4             ;|
2833:    JE      Build_FixedDisk_BDPB_5         ;| 若当前硬盘分区不是基本 DOS 分区,则跳转
2834:    CMP     BYTE PTR ES:[BX],6             ;|
2835:    JNE     Build_FixedDisk_BDPB_6         ;|
2836: Build_FixedDisk_BDPB_5:                   ;┘
2837:    OR      DH,DH                          ;┐ 若建立硬盘非活动的基本 DOS 分区对应的
2838:    JE      Build_FixedDisk_BDPB_8         ;┘ BDPB,则跳转
2839:    DEC     DH
2840: Build_FixedDisk_BDPB_6:
2841:    ADD     BX,10H                         ;ES:BX 指向下一个硬盘分区表
2842:    CMP     BX,402H                        ;┐ 若还有其它硬盘分区,则继续
2843:    JNE     Build_FixedDisk_BDPB_4         ;┘
2844: Build_FixedDisk_BDPB_7:
2845:    STC                                    ;┐ 出错返回
```

```
2846:     JMP     Build_FixedDisk_BDPB_36          ;┘
2847: Build_FixedDisk_BDPB_8:
2848:     MOV     CS:DriveNo_Init,DL               ;保存物理驱动器号
2849:     MOV     AX,ES:[BX+4]                     ;┐ DX:AX=本 DOS 分区的起始扇区号
2850:     MOV     DX,ES:[BX+6]                     ;┘
2851:     SUB     AX,1                             ;┐
2852:     SBB     DX,0                             ;│ 加上本 DOS 分区使用的扇区数,此时 DX:AX=
2853:     ADD     AX,ES:[BX+8]                     ;│ 本 DOS 分区的结束扇区号
2854:     ADC     DX,ES:[BX+0AH]                   ;┘
2855:     JNB     Build_FixedDisk_BDPB_9           ;若本 DOS 分区占用的扇区数大于 1,则跳转
2856:     OR      FATType_Init,80H
2857: Build_FixedDisk_BDPB_9:
2858:     MOV     AX,ES:[BX+4]                             ;┐
2859:     MOV     WORD PTR [DI].BDPB_HiddenSectors1,AX     ;│ "隐含扇区数"→BDPB
2860:     MOV     AX,ES:[BX+6]                             ;│
2861:     MOV     WORD PTR [DI].BDPB_HiddenSectors1+2,AX   ;┘
2862:     MOV     DX,ES:[BX+0AH]                           ;┐
2863:     MOV     AX,ES:[BX+8]                             ;│ "总扇区数"→BDPB
2864:     MOV     WORD PTR [DI].BDPB_BigTotalSector1+2,DX  ;│
2865:     MOV     WORD PTR [DI].BDPB_BigTotalSector1,AX    ;┘
2866:     CMP     DX,0                             ;┐
2867:     JA      Build_FixedDisk_BDPB_10          ;│
2868:     CMP     AX,40H                           ;│ 若本 DOS 分区占用的扇区数<64,则
2869:     JB      Build_FixedDisk_BDPB_7           ;│ 出错返回
2870: Build_FixedDisk_BDPB_10:                     ;┘
2871:     MOV     DX,WORD PTR [DI].BDPB_HiddenSectors1+2   ;┐ DS:AX←"隐含扇区数"(即本
2872:     MOV     AX,WORD PTR [DI].BDPB_HiddenSectors1     ;┘ DOS 分区的起始扇区号)
2873:     XOR     BX,BX                            ;┐ BX←"每道扇区数"
2874:     MOV     BL,BYTE PTR [DI].BDPB_SectorsPerTrack1   ;┘
2875:     PUSH    AX                               ;┐
2876:     MOV     AX,DX                            ;│
2877:     XOR     DX,DX                            ;│ 计算并保存 DOS 分区起始位置对应的磁道号
2878:     DIV     BX                               ;│ 高 16 位值
2879:     MOV     CS:Temp_HW,AX                    ;│
2880:     POP     AX                               ;┘
2881:     DIV     BX                               ;┐ AX←DOS 分区起始位置对应的磁道号低 16 位
2882:     MOV     CL,DL                            ;│ 值 CL←DOS 分区起始位置对应的道内扇区号
2883:     INC     CL                               ;┘
2884:     XOR     BX,BX                                    ;┐ BX←"磁头数"
2885:     MOV     BL,BYTE PTR [DI].BDPB_NumberOfHead1      ;┘
2886:     PUSH    AX                               ;┐
2887:     XOR     DX,DX                            ;│
2888:     MOV     AX,CS:Temp_HW                    ;│ 计算并保存 DOS 分区起始位置对应的圆柱号高
```

```
2889:   DIV     BX                              ;| 16 位
2890:   MOV     CS:Temp_HW,AX                   ;|
2891:   POP     AX                              ;┘
2892:   DIV     BX                              ;AX←DOS 分区起始位置对应的圆柱号的低 16 位
                                                ;值;DX←DOS 分区起始位置对应的磁头号
2893:   CMP     CS:Temp_HW,0                    ;┐
2894:   JA      Build_FixedDisk_BDPB_17         ;| 若 DOS 分区起始位置对应的圆柱号>1024,
2895:   CMP     AX,400H                         ;| 则跳转
2896:   JA      Build_FixedDisk_BDPB_17         ;┘
2897:   CMP     WORD PTR [DI].BDPB_Clock1orCylinderFlag,1;┐ 校正本 DOS 分区的起始圆柱面号
2898:   JNE     Build_FixedDisk_BDPB_11         ;| (因为该 DOS 分区为扩展 DOS 分
2899:   ADD     AX,[DI].BDPB_Clock2orCylinder   ;| 区)
2900: Build_FixedDisk_BDPB_11:                  ;┘
2901:   ROR     AH,1                            ;┐                          ┐
2902:   ROR     AH,1                            ;| 将圆柱面号和道内扇区号    |
2903:   AND     AH,0C0H                         ;| 拼成 INT 13H 所规定的入口 |
2904:   OR      CL,AH                           ;| 参数格式                  |
2905:   MOV     CH,AL                           ;┘                          | 读取当
2906:   MOV     DH,DL                           ;DH←磁头号                  | 前 DOS
2907:   MOV     DL,CS:DriveNo_Init              ;DL←物理驱动器号            | 分区的
2908:   PUSH    CS                              ;┐                          | 引导记
2909:   POP     ES                              ;| ES:BX 指向传送缓冲区     | 录
2910:   MOV     BX,offset DiskBuffer            ;┘                          |
2911:   MOV     AX,201H                         ;                           |
2912:   INT     13H                             ;                           ┘
2913:   MOV     BX,offset DiskBuffer            ;CS:BX 指向当前 DOS 分区的引导记录
2914:   PUSH    BX
2915:   PUSH    AX
2916:   CMP     BYTE PTR CS:[BX],0E9H           ;┐
2917:   JE      Build_FixedDisk_BDPB_12         ;|
2918:   CMP     BYTE PTR CS:[BX],0EBH           ;|
2919:   JNE     Build_FixedDisk_BDPB_14         ;| 若 DOS 分区的引导记录格式不对,则跳转
2920:   CMP     BYTE PTR CS:[BX+2],90H          ;|
2921:   JNE     Build_FixedDisk_BDPB_14         ;|
2922: Build_FixedDisk_BDPB_12:                  ;┘
2923:   MOV     BX,offset DiskBuffer+0BH        ;CS:BX 指向引导记录中的 BPB 参数块
2924:   MOV     AL,BYTE PTR CS:[BX].BPB_MediaByte ;AL←"介质描述符"
2925:   AND     AL,0F0H                         ;┐
2926:   CMP     AL,0F0H                         ;| 若介描述符无效,则跳转
2927:   JNE     Build_FixedDisk_BDPB_14         ;┘
2928:   CMP     WORD PTR CS:[BX],200H           ;┐ 若每扇区字节数≠512,则跳转
2929:   JNE     Build_FixedDisk_BDPB_14         ;┘
2930:   MOV     AL,BYTE PTR CS:[BX+2]           ;┐
```

```
2931:    OR      AL,AL                          ;| 若每簇扇区数=0,则跳转
2932:    JZ      Build_FixedDisk_BDPB_14        ;┘
2933: Build_FixedDisk_BDPB_13:                  ;┐
2934:    SHR     AL,1                           ;| 若每簇扇区数为2的幂,则跳转
2935:    JNC     Build_FixedDisk_BDPB_13        ;|
2936:    JZ      Build_FixedDisk_BDPB_15        ;┘
2937: Build_FixedDisk_BDPB_14:
2938:    POP     AX
2939:    POP     BX
2940:    JMP     Build_FixedDisk_BDPB_28
2941: Build_FixedDisk_BDPB_15:
2942:    POP     AX
2943:    POP     BX
2944:    CMP     WORD PTR CS:[BX+8],'.2'        ;┐
2945:    JNE     Build_FixedDisk_BDPB_19        ;| 若DOS版本号不是2.0,则跳转
2946:    CMP     BYTE PTR CS:[BX+0AH],'0'       ;|
2947:    JNE     Build_FixedDisk_BDPB_19        ;┘
2948: Build_FixedDisk_BDPB_16:
2949:    JMP     SHORT Build_FixedDisk_BDPB_21
2950: Build_FixedDisk_BDPB_17:
2951:    DB      0E9H
2952:    DW      0FEFBH
2953:    ;JMP    Build_FixedDisk_BDPB_16
2954: Build_FixedDisk_BDPB_18:
2955:    JMP     Build_FixedDisk_BDPB_29
2956: Build_FixedDisk_BDPB_19:
2957:    CALL    VerifyTotalSector              ;校证DOS分区的总扇区数
2958:    CMP     WORD PTR CS:[BX+8],'.0'        ;
2959:    JNE     Build_FixedDisk_BDPB_20
2960:    MOV     AL,BYTE PTR CS:[BX+7]
2961:    SUB     AL,31H                         ;'1'
2962:    AND     AL,0FEH
2963:    JZ      Build_FixedDisk_BDPB_21
2964:    JMP     Build_FixedDisk_BDPB_28
2965: Build_FixedDisk_BDPB_20:
2966:    CMP     WORD PTR CS:[BX+8],'.3'        ;┐
2967:    JB      Build_FixedDisk_BDPB_18        ;|
2968:    JNZ     Build_FixedDisk_BDPB_21        ;| 若DOS版本号<3.1,则跳转
2969:    CMP     BYTE PTR CS:[BX+0AH],'1'       ;|
2970:    JB      Build_FixedDisk_BDPB_18        ;┘
2971: Build_FixedDisk_BDPB_21:
2972:    CMP     BYTE PTR CS:DiskBuffer+26H,29H ;┐若不是扩充BPB参数块,则跳转
2973:    JNE     Build_FixedDisk_BDPB_23        ;┘
```

```
2974:    CMP     BYTE PTR CS:DiskBuffer+10H,0  ;┐若 FAT 表个数≠0,则跳转
2975:    JNE     Build_FixedDisk_BDPB_23       ;┘
2976:    PUSH    DI
2977:    PUSH    DS
2978:    PUSH    DS
2979:    POP     ES
2980:    PUSH    CS
2981:    POP     DS
2982:    MOV     SI,offset DiskBuffer+0BH      ;DS:SI 指向扩充 BPB 参数块
2983:    ADD     DI,6                          ;ES:DI 指向 BDPB 的 Build BPB 参数块
2984:    CMP     WORD PTR CS:[SI+8],0          ;┐
2985:    JNE     Build_FixedDisk_BDPB_22       ;│
2986:    CMP     WORD PTR CS:[SI+15H],0        ;│若扩充 BPB 参数块的"总扇区数"字段值
2987:    JNE     Build_FixedDisk_BDPB_22       ;│非零,则跳转
2988:    CMP     WORD PTR CS:[SI+17H],0        ;│
2989:    JNE     Build_FixedDisk_BDPB_22       ;┘
2990:    MOV     AX,WORD PTR [DI].BDPB_SectorsPerCluster1  ;┐
2991:    MOV     WORD PTR CS:[SI+8],AX         ;│
2992:    MOV     AX,[DI].BDPB_NumberOfHead1    ;│按 DOS 分区表指定的"总扇区数"
2993:    MOV     WORD PTR CS:[SI].BPB_BigTotalSector,AX  ;│补缺扩充 BPB 参数块的"总扇区
2994:    MOV     AX,WORD PTR [DI].BDPB_HiddenSectors1    ;│数"
2995:    MOV     WORD PTR CS:[SI].BPB_BigTotalSector+2,AX ;┘
2996: Build_FixedDisk_BDPB_22:
2997:    MOV     CX,BPB_Len1                   ;CX←BPB 参数块长度
2998:    REP     MOVSB                         ;根据 DOS 分区引导记录中的扩充 BPB 参数块设
                                               ;置 BDPB 的"Build BPB 参数块"
2999:    POP     DS
3000:    POP     DI                            ;DS:DI 指向 BDPB
3001:    PUSH    ES                            ;┐
3002:    PUSH    DS                            ;│
3003:    POP     ES                            ;│
3004:    PUSH    CS                            ;│
3005:    POP     DS                            ;│依据扩充 BPB 参数块设置 BDPB 的"卷系
3006:    MOV     BP,offset Set_BDPBExtMediaState ;│列号、卷标名、文件系统类型"
3007:    PUSH    CS                            ;│
3008:    CALL    CallSubInIO2                  ;│
3009:    PUSH    ES                            ;┘
3010:    POP     DS
3011:    POP     ES
3012:    JMP     Build_FixedDisk_BDPB_35       ;转去设置 BDPB 的"特征状态字节"值
3013: Build_FixedDisk_BDPB_23:                 ;处理一般的 BPB 参数块(非扩充 BPB 参数块)
3014:    MOV     SI,offset DiskBuffer+0BH      ;CS:SI 指向 BPB 参数块
```

```
3015:    XOR     DX,DX                                  ;┐
3016:    MOV     AX,WORD PTR CS:[SI+18]                 ;│
3017:    OR      AX,AX                                  ;│
3018:    JNZ     Build_FixedDisk_BDPB_24                ;│
3019:    MOV     AX,WORD PTR CS:[SI+15H]                ;│
3020:    MOV     DX,WORD PTR CS:[SI+17H]                ;│ 根据 BPB 参数块的"总扇区数"
3021:    MOV     CX,DX                                  ;│ 更换 BDPB 的"总扇区数"字段值
3022:    OR      CX,AX                                  ;│
3023:    JZ      Build_FixedDisk_BDPB_25                ;│
3024: Build_FixedDisk_BDPB_24:                          ;│
3025:    MOV     WORD PTR [DI].BDPB_BigTotalSector1,AX  ;│
3026:    MOV     WORD PTR [DI].BDPB_BigTotalSector1+2,DX ;│
3027: Build_FixedDisk_BDPB_25:                          ;┘
3028:    MOV     AX,WORD PTR [DI].BDPB_BigTotalSector1  ;┐ DX:AX←DOS 分区的总扇区数
3029:    MOV     DX,WORD PTR [DI].BDPB_BigTotalSector1+2 ;┘
3030:    MOV     BX,CS:[SI].BPB_ReservedSector  ;┐ "保留扇区数"→BDPB
3031:    MOV     [DI].BDPB_ReservedSectors1,BX  ;┘
3032:    SUB     AX,BX                          ;┐ 减去"保留扇区数"
3033:    SBB     DX,0                           ;┘
3034:    MOV     BX,CS:[SI].BPB_SectorsPerFAT           ;┐ "每个 FAT 表占用的扇区数"→
3035:    MOV     WORD PTR DS:[DI].BDPB_SectorsPerFAT1,BX ;┘ BDPB
3036:    SHL     BX,1                           ;┐
3037:    SUB     AX,BX                          ;│ 减去"FAT 表占用的扇区数"
3038:    SBB     DX,0                           ;┘
3039:    MOV     BX,WORD PTR CS:[SI+0BH]        ;┐ "最大根目录项数"→BDPB
3040:    MOV     [DI].BDPB_TotalOfRootDir1,BX   ;┘
3041:    MOV     CL,4                           ;┐
3042:    SHR     BX,CL                          ;│ 减去"根目录占用的扇区数"
3043:    SUB     AX,BX                          ;│
3044:    SBB     DX,0                           ;┘
3045:    XOR     CX,CX                          ;┐
3046:    MOV     CL,BYTE PTR CS:[SI+2]          ;│ "每簇扇区数"→BDPB
3047:    MOV     [DI].BDPB_SectorsPerCluster1,CL ;┘
3048: ;DX:AX=数据区的扇区数,CX=每簇扇区数
3049:    PUSH    AX                             ;┐
3050:    MOV     AX,DX                          ;│
3051:    XOR     DX,DX                          ;│
3052:    DIV     CX                             ;│ 计算 DOS 分区的簇数
3053:    MOV     CS:Temp_HW,AX                  ;│
3054:    POP     AX                             ;│
3055:    DIV     CX                             ;┘
3056:    CMP     CS:Temp_HW,0                   ;┐ 若 DOS 分区的簇数≥65536,则跳转
3057:    JA      Build_FixedDisk_BDPB_27        ;┘
```

```
3058:    CMP     AX,0FF6H                    ;┐若 DOS 分区的簇数<4086,则跳转
3059:    JB      Build_FixedDisk_BDPB_26     ;┘
3060:    OR      FATType_Init,40H            ;置 FAT 表的特征标志位,表示 DOS 分区的簇数在
                                             ;4086～65535 之间,FAT 表每项长 16 位

3061: Build_FixedDisk_BDPB_26:
3062:    PUSH    ES                          ;┐
3063:    PUSH    DS                          ;│
3064:    POP     ES                          ;│
3065:    PUSH    CS                          ;│
3066:    POP     DS                          ;│依据扩充 BPB 参数块设置 BDPB 中的"卷系
3067:    MOV     BP,offset Set_BDPBExtMediaState ;│列号、卷标名、文件系统类型"
3068:    PUSH    CS                          ;│
3069:    CALL    CallSubInIO2                ;│
3070:    PUSH    ES                          ;│
3071:    POP     DS                          ;│
3072:    POP     ES                          ;┘
3073:    JMP     Build_FixedDisk_BDPB_34     ;转去处理 BDPB 的"总扇区数"字段值,并设置
                                             ;BDPB 的"特征状态字节"值

3074: Build_FixedDisk_BDPB_27:
3075:    OR      CS:FATType_Init,80H         ;置特征标志位,表示 DOS 分区的簇数≥65536
3076:    JMP     Build_FixedDisk_BDPB_35     ;转去设置 BDPB 的"特征状态字节"值
3077: Build_FixedDisk_BDPB_28:
3078:    OR      [DI].BDPB_Attr_Status,200H
3079: Build_FixedDisk_BDPB_29:               ;处理 DOS 3.1 版本以下的情况
3080:    MOV     DX,WORD PTR [DI].BDPB_BigTotalSector1+2 ;┐ DX:AX←DOS 分区的总扇
3081:    MOV     AX,WORD PTR [DI].BDPB_BigTotalSector1   ;┘ 区数
3082:    MOV     SI,offset FixedDiskPara     ;SI 指向硬盘分区参数表数组
3083: Build_FixedDisk_BDPB_30:
3084:    CMP     DX,CS:[SI]                  ;┐
3085:    JB      Build_FixedDisk_BDPB_32     ;│
3086:    JA      Build_FixedDisk_BDPB_31     ;│若找到了匹配的硬盘分区参数表,则跳转
3087:    CMP     AX,WORD PTR CS:[SI+2]       ;│
3088:    JBE     Build_FixedDisk_BDPB_32     ;│
3089: Build_FixedDisk_BDPB_31:               ;┘
3090:    ADD     SI,0AH                      ;SI 指向下一个硬盘分区参数表
3091:    JMP     SHORT Build_FixedDisk_BDPB_30 ;转去继续搜索匹配的硬盘分区参数表
3092: Build_FixedDisk_BDPB_32:               ;找到合适的硬盘分区参数表
3093:    MOV     CL,[SI+8]                   ;┐设置"文件系统类型码"
3094:    OR      FATType_Init,CL             ;┘
3095:    MOV     CX,WORD PTR CS:[SI+4]       ;CL←移位因子,CH←每簇扇区数
3096:    MOV     DX,WORD PTR CS:[SI+6]       ;┐"最大根目录数"→BDPB
3097:    MOV     [DI].BDPB_TotalOfRootDir1,DX ;┘
3098:    MOV     DX,WORD PTR [DI].BDPB_BigTotalSector1+2 ;┐ DX:AX←DOS 分区的总扇区数
```

```
3099:   MOV     AX,WORD PTR [DI].BDPB_BigTotalSector1      ;┘
3100:   MOV     [DI].BDPB_SectorsPerCluster1,CH ;"每簇扇区数"→BDPB
3101:   TEST    FATType_Init,40H                ;┐ 若 DOS 分区使用 16 位 FAT 表,则跳转
3102:   JNZ     Build_FixedDisk_BDPB_33         ;┘
3103:   XOR     BX,BX                                ;┐                    ┐
3104:   MOV     BL,CH                                ;|                    |
3105:   DEC     BX                                   ;|                    |
3106:   ADD     BX,AX                                ;| 计算 DOS 分区的簇数→ |
3107:   SHR     BX,CL                                ;| BX                 |
3108:   INC     BX                                   ;|                    | "每个 FAT
3109:   AND     BL,0FEH                              ;┘                    | 表占用的
3110:   MOV     SI,BX                                ;┐ 每个 FAT 表占用的字节 | 扇区数"
3111:   SHR     BX,1                                 ;| 数→BX              | →BDPB
3112:   ADD     BX,SI                                ;┘                    |
3113:   ADD     BX,1FFH                              ;┐ 每个 FAT 表占用的扇区 |
3114:   SHR     BH,1                                 ;┘ 数→BH              |
3115:   MOV     BYTE PTR [DI].BDPB_SectorsPerFAT1,BH;                       ┘
3116:   JMP     SHORT Build_FixedDisk_BDPB_34 ;转去处理 BDPB 的"总扇区数"字段值,并设置
                                              ;BDPB 的"特征状态字节"值
3117: Build_FixedDisk_BDPB_33:                ;DOS 分区使用 16 位 FAT 表
3118:   MOV     CL,4                          ;┐
3119:   PUSH    DX                            ;|
3120:   MOV     DX,[DI].BDPB_TotalOfRootDir1  ;|
3121:   SHR     DX,CL                         ;|
3122:   SUB     AX,DX                         ;|
3123:   POP     DX                            ;|
3124:   SBB     DX,0                          ;|
3125:   SUB     AX,1                          ;| 计算每个 FAT 表占用的扇区数→AX
3126:   SBB     DX,0                          ;|
3127:   MOV     BL,2                          ;|
3128:   MOV     BH,[DI].BDPB_SectorsPerCluster1 ;|
3129:   ADD     AX,BX                         ;|
3130:   ADC     DX,0                          ;|
3131:   SUB     AX,1                          ;|
3132:   SBB     DX,0                          ;|
3133:   DIV     BX                            ;┘
3134:   MOV     [DI].BDPB_SectorsPerFAT1,AX   ;"每个 FAT 表占用的扇区数"→BDPB
3135:   MOV     BL,FATType_Init               ;┐ "文件系统类型码"→BDPB
3136:   MOV     [DI].BDPB_SizeFlag,BL         ;┘
3137:   PUSH    DS                                ;┐
3138:   PUSH    DS                                ;|
3139:   POP     ES                                ;|
3140:   PUSH    CS                                ;| 设置 BDPB 的 缺省卷系列号、缺省卷标名,
```

```
3141:    POP     DS                                      ;| 以及文件系统类型标志串"
3142:    MOV     BP,offset Set_BDPBDefExtMediaState      ;|
3143:    PUSH    CS                                      ;|
3144:    CALL    CallSubInIO2                            ;|
3145:    POP     DS                                      ;┘
3146: Build_FixedDisk_BDPB_34:
3147:    MOV     DX,WORD PTR [DI]._BDPB_BigTotalSector1+2  ;┐ DX:AX←┐
                                                         ;| DOS 分区 |
                                                         ;| 的总扇区 |
3148:    MOV     AX,WORD PTR [DI]._BDPB_BigTotalSector1    ;┘ 数       |
3149:    CMP     DX,0                                    ;             |
3150:    JA      Build_FixedDisk_BDPB_35                 ;             | 若 DOS 分区使用
3151:    CMP     WORD PTR [DI]._BDPB_HiddenSectors1+2,0  ;             | 的扇区数号≥
3152:    JA      Build_FixedDisk_BDPB_35                 ;             | 65536,则跳转
3153:    ADD     AX,WORD PTR [DI]._BDPB_HiddenSectors1   ;             |
3154:    JC      Build_FixedDisk_BDPB_35                 ;             ┘
3155:    MOV     AX,WORD PTR [DI]._BDPB_BigTotalSector1  ;┐ DOS 分区的"总扇区数"用 16 位
3156:    MOV     [DI]._BDPB_TotalSector1,AX              ;| 表示
3157:    MOV     WORD PTR [DI]._BDPB_BigTotalSector1,0   ;┘
3158: Build_FixedDisk_BDPB_35:
3159:    MOV     BL,FATType_Init                 ;┐ "文件系统类型码"→BDPB
3160:    MOV     [DI]._BDPB_SizeFlag,BL          ;┘
3161:    CLC
3162: Build_FixedDisk_BDPB_36:
3163:    POP     ES
3164:    POP     DS
3165:    POP     BX
3166:    POP     DI
3167:    RET
3168: Build_FixedDisk_BDPB   ENDP
3169: ;=============================================================
3170: ;相对地址偏移:2156
3171: ;功能:校证 DOS 分区的总扇区数
3172: ;入口参数:无
3173: ;出口参数:无
3174: ;=============================================================
3175: VerifyTotalSector       PROC    NEAR
3176:    PUSH    AX
3177:    PUSH    DX
3178:    PUSH    SI
3179:    CMP     BYTE PTR CS:DiskBuffer+26H,29H  ;┐ 若 DOS 引导记录存放扩充 BPB 参数块,则
3180:    JE      VerifyTotalSec_2                ;┘ 跳转
3181:    CMP     WORD PTR CS:[BX+7],3031H
```

```
3182:     JNE     VerifyTotalSec_1
3183:     CMP     BYTE PTR CS:[BX+0AH],' 0'
3184:     JNE     VerifyTotalSec_2
3185: VerifyTotalSec_1:
3186:     MOV     SI,offset DiskBuffer + 0BH          ;CS:SI 指向 BPB 参数块
3187:     CMP     WORD PTR CS:[SI+8],0                ;┐若 DOS 分区的"总扇区数"用 32 位表示,则跳
3188:     JE      VerifyTotalSec_2                    ;┘转
3189:     MOV     AX,WORD PTR CS:[SI+8]               ;┐
3190:     ADD     AX,WORD PTR CS:[SI+11H]             ;├若 DOS 分区的结束扇区号<65536,或>
3191:     JNC     VerifyTotalSec_2                    ;├ 65536,则跳转
3192:     JNZ     VerifyTotalSec_2                    ;┘
3193:     DEC     WORD PTR CS:[SI+8]                        ;┐若 DOS 分区的结束扇区号=65536
3194:     SUB     WORD PTR [DI].BDPB_BigTotalSector1,1      ;├,则丢掉一个尾扇区
3195:     SBB     WORD PTR [DI].BDPB_BigTotalSector1+2,0    ;┘
3196: VerifyTotalSec_2:
3197:     POP     SI
3198:     POP     DX
3199:     POP     AX
3200:     RET
3201: VerifyTotalSector     ENDP
3202: ;
3203: Value_2     DW     2                    ;2/3 的分子值(用于计算 12 位 FAT 表占用的扇
                                              ;区数)
3204: Value_3     DW     3                    ;2/3 的分母值(用于计算 12 位 FAT 表占用的扇
                                              ;区数)
3205: BytesPerSector  DW  200H                ;每扇区字节数(用于计算 12 位 FAT 表占用的
                                              ;扇区数)
3206: ;
3207: ;=============================================================
3208: ;相对地址偏移:219C
3209: ;功能:建立 BDPB 链中所有表结点的 Default BPB 参数块
3210: ;入口参数:无
3211: ;出口参数:无
3212: ;=============================================================
3213: Build_BDPB_DefaultBPB   PROC    NEAR
3214:     XOR     BX,BX
3215:     LES     DI,DWORD PTR BDPB_Head          ;ES:DI 指向第一个 BDPB
3216: Build_BDPB_DB_1:
3217:     PUSH    ES
3218:     PUSH    DI
3219:     MOV     BL,ES:[DI].BDPB_DeviceType      ;┐若当前 BDPB 对应的不是硬盘驱动器,则
3220:     CMP     BL,5                            ;├跳转
3221:     JNE     Build_BDPB_DB_4                 ;┘
```

```
3222:     XOR     DX,DX                                          ;┐
3223:     MOV     AX,ES:[DI].BDPB_TotalSector1                   ;│
3224:     OR      AX,AX                                          ;│
3225:     JNE     Build_BDPB_DB_2                                ;│ DX:AX←DOS 分区使用的总
3226:     MOV     DX,WORD PTR ES:[DI].BDPB_BigTotalSector1+2     ;│ 扇区数
3227:     MOV     AX,WORD PTR ES:[DI].BDPB_BigTotalSector1       ;│
3228: Build_BDPB_DB_2:                                           ;┘
3229:     PUSH    DX                                             ;┐
3230:     PUSH    AX                                             ;│
3231:     MOV     AX,ES:[DI].BDPB_NumberOfHead1                  ;│
3232:     MUL     WORD PTR ES:[DI].BDPB_SectorsPerTrack1         ;│ 计算每个圆柱面的扇区
3233:     MOV     CX,AX                                          ;│ 数→CX
3234:     POP     AX                                             ;│
3235:     POP     DX                                             ;┘
3236:     PUSH    AX                                 ;┐
3237:     MOV     AX,DX                              ;│
3238:     XOR     DX,DX                              ;│ 计算并保存圆柱面数的高 16 位值
3239:     DIV     CX                                 ;│
3240:     MOV     CS:Temp_HW,AX                      ;│
3241:     POP     AX                                 ;┘
3242:     DIV     CX                                 ;┐
3243:     OR      DX,DX                              ;│
3244:     JE      Build_BDPB_DB_3                    ;│ 圆柱面数的低 16 位值→AX
3245:     INC     AX                                 ;│
3246: Build_BDPB_DB_3:                               ;┘
3247:     MOV     ES:[DI].BDPB_NumberOfCylinder,AX   ;"圆柱面数"→BDPB
3248:     PUSH    ES                                 ;┐ DS:SI 指向硬盘对应的 BDPB 的 Build BPB
3249:     POP     DS                                 ;│ 参数块
3250:     LEA     SI,[DI].BDPB_BytesPerSector1       ;┘
3251:     JMP     SHORT Build_BDPB_DB_9              ;转去设置硬盘对应的 BDPB 的 Default BPB 参数
                                                     ;块
3252: Build_BDPB_DB_4:
3253:     PUSH    CS
3254:     POP     DS
3255:     CMP     CS:FDI_Flag_IO1,01                 ;┐ 若未安装软盘驱动器,则跳转
3256:     JE      Build_BDPB_DB_10                   ;┘
3257:     CMP     BL,7                               ;┐ 若不是 1.44MB 3.5 英寸软盘驱动器,则跳转
3258:     JNE     Build_BDPB_DB_8                    ;┘
3259:     XOR     DX,DX
3260:     MOV     AX,[DI].BDPB_NumberOfCylinder      ;AX←"圆柱面数"
3261:     MUL     [DI].BDPB_NumberOfHead2            ;DX:AX="磁道数"
3262:     MUL     [DI].BDPB_SectorsPerTrack2         ;DX:AX←"总扇区数"
3263:     MOV     [DI].BDPB_TotalSector2,AX          ;"总扇区数"→BDPB
```

```
3264:    DEC     AX                              ;┐
3265:    MOV     DL,1                            ;|
3266: Build_BDPB_DB_5:                           ;|
3267:    CMP     AX,0FF6H                        ;|
3268:    JB      Build_BDPB_DB_6                 ;| 计算簇数、每簇扇区数(AX=)簇数;DL=每簇
3269:    SHR     AX,1                            ;| 扇区数
3270:    SHL     DL,1                            ;|
3271:    JMP     SHORT Build_BDPB_DB_5           ;|
3272: Build_BDPB_DB_6:                           ;┘
3273:    CMP     DL,1                            ;┐
3274:    JE      Build_BDPB_DB_7                 ;| 修改 BDPB 的"最大根目录项数"字段值
3275:    MOV     [DI].BDPB_TotalOfRootDir2,00F0H ;|
3276: Build_BDPB_DB_7:                           ;┘
3277:    MOV     [DI].BDPB_SectorsPerCluster2,DL ;"每簇扇区数"→BDPB
3278:    MUL     CS:Value_3                      ;┐
3279:    DIV     CS:Value_2                      ;|
3280:    XOR     DX,DX                           ;| 计算每个 FAT 表占用的扇区数→AX
3281:    DIV     CS:BytesPerSector               ;|
3282:    INC     AX                              ;┘
3283:    MOV     [DI].BDPB_SectorsPerFAT2,AX     ;"每个 FAT 表占用的扇区数"→BDPB
3284:    JMP     SHORT Build_BDPB_DB_10
3285: Build_BDPB_DB_8:
3286:    SHL     BX,1                            ;┐ DS:SI 指向当前软盘驱动器类型对应的 Default
3287:    MOV     SI,offset Ptr_BPB_Array         ;| BPB 参数块
3288:    MOV     SI,[BX+SI]                      ;┘
3289: Build_BDPB_DB_9:
3290:    LEA     DI,[DI].BDPB_BytesPerSector2    ;┐
3291:    MOV     CX,BPB_Len1                     ;| 建立 BDPB 的 Default BPB 参数块
3292:    REP     MOVSB                           ;┘
3293: Build_BDPB_DB_10:
3294:    POP     DI
3295:    POP     ES
3296:    LES     DI,DWORD PTR ES:[DI].BDPB_NextPtr_OFS   ;ES:DI 指向下一个 BDPB
3297:    CMP     DI,-1                           ;┐ 若 BDPB 链所有结点都处理完,则结束返回
3298:    JE      Build_BDPB_DB_RET               ;┘
3299:    JMP     Build_BDPB_DB_1                 ;转去继续建立后续 BDPB 的 Default BPB 参数块
3300: Build_BDPB_DB_RET:
3301:    RET
3302: Build_BDPB_DefaultBPB        ENDP
3303: ;=============================================================
3304: ;相对地址偏移:2257
3305: ;功能:初始化并行端口
3306: ;入口参数:AL=打印机号(0=LPT1,1=LPT2,2=LPT3)
```

```
3307: ;出口参数:AH=打印机状态
3308: ;=========================================================
3309: Init_PRN_Adapter    PROC    NEAR
3310:     CBW                                   ;┐ DX←打印机号
3311:     MOV     DX,AX                         ;┘
3312:     MOV     AH,1                          ;AH←"初始化打印机"的命令码
3313:     INT     17H                           ;请求并行打印机服务
3314:     RET
3315: Init_PRN_Adapter    ENDP
3316: ;=========================================================
3317: ;相对地址偏移:225F
3318: ;功能:初始化串行端口
3319: ;入口参数:AL=串行端口号(0=COM1,1=COM2,2=COM3,3=COM4)
3320: ;出口参数:AH=线路状态
3321: ;=========================================================
3322: Init_COM_Adapter    PROC    NEAR
3323:     CBW                                   ;┐ DX←串行端口号
3324:     MOV     DX,AX                         ;┘
3325:     MOV     AL,0A3H                       ;AL←串行端口初始化参数(波特率为2400,没有
                                                ;奇偶位、2个结束位、8位字符)
3326:     MOV     AH,0                          ;AH←"初始化串行通信端口"的命令码
3327:     INT     14H                           ;请求串行通信服务
3328:     RET
3329: Init_COM_Adapter    ENDP
3330: ;=========================================================
3331: ;相对地址偏移:2269
3332: ;功能:建立系统配置的所有硬盘扩展DOS分区对应的逻辑驱动器的BDPB
3333: ;入口参数:无
3334: ;出口参数:无
3335: ;=========================================================
3336: Build_ExtendedDisk_BDPB    PROC    NEAR
3337:     MOV     DH,FixedDiskNumber            ;DH←硬盘驱动器数
3338:     CMP     DH,0                          ;┐ 若未安装硬盘,则直接返回
3339:     JE      Build_ExtDisk_BDPB_RET        ;┘
3340:     MOV     DL,80H                        ;DL←第1台硬盘驱动器的物理驱动器号
3341: Build_ExtDisk_BDPB_1:
3342:     PUSH    DX                            ;
3343:     MOV     DriveNumber_Init,DL           ;保存硬盘驱动器的物理驱动器号
3344:     MOV     AH,8                          ;┐ 读取硬盘驱动器参数
3345:     INT     13H                           ;┘
3346:     INC     DH                            ;┐
3347:     XOR     AX,AX                         ;│ AX←"磁头数"
3348:     MOV     AL,DH                         ;┘
```

```
3349:    MOV     HeadsOfFixed_Init,AX                ;保存硬盘驱动器的"磁头数"
3350:    AND     CL,3FH                              ;┐
3351:    MOV     AL,CL                               ;| 保存硬盘的"每道扇区数"
3352:    MOV     SectorsPerTrackOfFixed,AX           ;┘
3353:    PUSH    ES
3354:    MOV     DL,DriveNumber_Init                 ;DL←硬盘驱动器的物理驱动器号
3355:    CALL    Read_BootstrapSector1               ;读取硬盘的第一个扇区,即硬盘的主引导记录
3356:    JC      Build_ExtDisk_BDPB_2                ;若读操作失败,则跳转
3357:    CALL    Init_ExtendedDisk_BDPB              ;建立当前硬盘扩展 DOS 分区对应的逻辑驱动器的
                                                     ;BDPB
3358: Build_ExtDisk_BDPB_2:
3359:    POP     ES
3360:    POP     DX
3361:    INC     DL                                  ;DL←下一个硬盘驱动器的物理驱动器号
3362:    DEC     DH                                  ;┐若还有硬盘,则转去建立它的扩展 DOS 分区对
                                                     ;| 应的逻辑驱动器的 BDPB
3363:    JNZ     Build_ExtDisk_BDPB_1                ;┘
3364: Build_ExtDisk_BDPB_RET:
3365:    RET
3366: Build_ExtendedDisk_BDPBENDP
3367: ;=============================================================
3368: ;相对地址偏移:22A4
3369: ;功能:建立并初始化一个硬盘扩展 DOS 分区对应的逻辑驱动器的 BDPB,及其子扩展 DOS 分区对
          应的逻辑驱动器的 BDPB,有关它实现的策略请参见第四章
3370: ;入口参数:ES:BX 指向存放硬盘分区表的引导记录的缓冲区
3371: ;出口参数:无
3372: ;=============================================================
3373: Init_ExtendedDisk_BDPB        PROC    NEAR
3374:    ADD     BX,1C2H                             ;ES:BX 指向第一个硬盘分区表的"分区类型码"字
                                                     ;段
3375: Init_ExtBDPB_1:
3376:    CMP     BYTE PTR ES:[BX],5                  ;┐若当前硬盘分区为扩展 DOS 分区,则跳转
3377:    JE      Init_ExtBDPB_2                      ;┘
3378:    ADD     BX,10H                              ;┐
3379:    CMP     BX,402H                             ;| 继续搜索其它硬盘分区
3380:    JNE     Init_ExtBDPB_1                      ;┘
3381:    JMP     SHORT Init_ExtBDPB_RET              ;硬盘没有扩展 DOS 分区,直接返回
3382: Init_ExtBDPB_2:
3383:    CALL    CheckLogicDriveNum                  ;检测逻辑驱动器是否超过 26 个
3384:    JNB     Init_ExtBDPB_RET                    ;若超过 26 个,则结束返回
3385:    MOV     DI,Ptr_FreeBDPB                     ;DS:DI 指向当前可供使用的空闲 BDPB
3386:    MOV     [DI].BDPB_Clock1orCylinderFlag,1    ;设置要校证起始圆柱面号的标志(因为该 BDPB 对
                                                     ;应于扩展 DOS 分区)
```

```
3387:    OR      [DI].BDPB_Attr_Status,01        ;设置"介质不可装卸"的标志位
3388:    MOV     [DI].BDPB_DeviceType,5          ;"设备类型"(硬盘)→BDPB
3389:    MOV     FATType_Init,0                  ;清文件系统类型码(FAT 表每项长 16 位)
3390:    MOV     AX,HeadsOfFixed_Init            ;┐"磁头数"→BDPB
3391:    MOV     [DI].BDPB_NumberOfHead1,AX      ;┘
3392:    MOV     AX,SectorsPerTrackOfFixed       ;┐"每道扇区数"→BDPB
3393:    MOV     [DI].BDPB_SectorsPerTrack1,AX   ;┘
3394:    MOV     AL,DriveNumber_Init             ;┐"物理驱动器号"→BDPB
3395:    MOV     [DI].BDPB_DriveNumber,AL        ;┘
3396:    MOV     AL,TotalOfBlockDev              ;┐"逻辑驱动器号"→BDPB
3397:    MOV     [DI].BDPB_LogicalNumber,AL      ;┘
3398:    CMP     WORD PTR ES:[BX+0AH],00         ;┐若扩展 DOS 分区使用   ┐
                                                 ;│的扇区数>65536,则    │若扩展 DOS 分
                                                 ;│跳转                 │区使用的扇区
3399:    JA      Init_ExtBDPB_3                  ;┘                     │数<64,则直接
3400:    CMP     WORD PTR ES:[BX+8],40H          ;┐若扩展 DOS 分区使用   │返回
3401:    JB      Init_ExtBDPB_RET                ;┘的扇区数<64,则跳转   │
3402: Init_ExtBDPB_3:                           ;                      ┘
3403:    SUB     BX,4                            ;ES:BX 指向当前扩展 DOS 分区表
3404:    MOV     DH,ES:[BX+2]                    ;┐
3405:    AND     DH,0C0H                         ;│
3406:    ROL     DH,1                            ;│"当前扩展 DOS 分区的起始圆柱面号"→BDPB
3407:    ROL     DH,1                            ;│
3408:    MOV     DL,ES:[BX+3]                    ;│
3409:    MOV     [DI].BDPB_Clock2orCylinder,DX   ;┘
3410:    MOV     CX,ES:[BX+2]                    ;CX←起始圆柱面号、扇区号
3411:    MOV     DH,ES:[BX+1]                    ;DH←起始磁头号
3412:    MOV     DL,DriveNumber_Init             ;DL←物理驱动器号
3413:    MOV     BX,200H                         ;┐
3414:    MOV     AX,201H                         ;│读取硬盘扩展 DOS 分区的分区信息表
3415:    INT     13H                             ;┘
3416:    JC      Init_ExtBDPB_RET                ;若读操作失败,则跳转
3417:    MOV     BX,3C2H                         ;┐
3418:    PUSH    ES                              ;│初始化硬盘扩展 DOS 分区对应的逻辑驱动器
3419:    CALL    Init_FixedBDPB                  ;│的 BDPB
3420:    POP     ES                              ;┘
3421:    JC      Init_ExtBDPB_4
3422:    CALL    App_New_Node                    ;将当前 BDPB 插入到 BDPB 链尾
3423: Init_ExtBDPB_4:
3424:    JMP     Init_ExtBDPB_1                  ;转去建立当前硬盘扩展 DOS 分区的子扩展 DOS
                                                 ;分区(或子 DOS 分区)对应的 BDPB
3425: Init_ExtBDPB_RET:
3426:    RET
```

```
3427: Init_ExtendedDisk_BDPB      ENDP
3428: ;=========================================================
3429: ;相对地址偏移:2336
3430: ;功能:初始化硬盘 DOS 分区对应的逻辑驱动器的 BDPB
3431: ;入口参数:ES:BX 指向硬盘分区信息表
3432: ;出口参数:CF=0,成功;CF=1,没有基本 DOS 分区
3433: ;=========================================================
3434: Init_FixedBDPB     PROC     NEAR
3435:     PUSH    DI
3436:     PUSH    BX
3437:     PUSH    DS
3438:     PUSH    ES
3439: Init_FixedBDPB_1:
3440:     CMP     BYTE PTR ES:[BX],1          ;┐
3441:     JE      Init_FixedBDPB_2            ;|
3442:     CMP     BYTE PTR ES:[BX],4          ;| 若当前硬盘分区是一个基本 DOS 分区,则跳转
3443:     JE      Init_FixedBDPB_2            ;|
3444:     CMP     BYTE PTR ES:[BX],6          ;|
3445:     JE      Init_FixedBDPB_2            ;┘
3446:     ADD     BX,10H                      ;ES:BX 指向下一个硬盘分区表
3447:     CMP     BX,402H                     ;┐ 若硬盘分区信息表未搜索完,则转去继续搜索
3448:     JNE     Init_FixedBDPB_1            ;┘
3449:     STC
3450:     POP     ES
3451:     POP     DS
3452:     POP     BX
3453:     POP     DI
3454:     RET
3455: Init_FixedBDPB_2:
3456:     JMP     Build_FixedDisk_BDPB_8      ;转去初始化硬盘 DOS 分区对应的逻辑驱动器的
                                              ;BDPB
3457: Init_FixedBDPB      ENDP
3458: ;=========================================================
3459: ;相对地址偏移:235E
3460: ;功能:检测逻辑驱动器是否超过 26 个
3461: ;入口参数:无
3462: ;出口参数:CF=1,未超过;0:超过
3463: ;=========================================================
3464: CheckLogicDriveNum   PROC     NEAR
3465:     CMP     TotalOfBlockDev,1AH         ;┐ 若逻辑驱动器未超过 26 个,则直接返回
3466:     JB      ChkLogicDriveNum_RET        ;┘
3467:     PUSH    ES                          ;┐
3468:     MOV     AX,SysInt_11_SEG            ;| 设置 SysInt—Ⅱ的变量(逻辑驱动器超过 26 个
```

```
3469:     MOV     ES,AX                          ;| 的标志)
3470:     MOV     BYTE PTR ES:103_03FF,1         ;|
3471:     POP     ES                             ;┘
3472: ChkLogicDriveNum_RET:
3473:     RET
3474: CheckLogicDriveNum   ENDP
3475: ;=============================================================
3476: ;相对地址偏移:2373
3477: ;功能:将当前 BDPB 结点追加到 BDPB 链表尾
3478: ;入口参数:DS:DI 指向当前逻辑驱动器对应的 BDPB
3479: ;出口参数:无
3480: ;=============================================================
3481: App_New_Node        PROC     NEAR
3482:     PUSH    SI
3483:     PUSH    BX
3484:     MOV     SI,BDPB_Head                   ;DS:SI 指向 BDPB 链的头结点
3485: App_New_Node_1:                            ;┐
3486:     CMP     WORD PTR [SI],0FFFFH           ;|
3487:     JE      App_New_Node_2                 ;| 搜索 BDPB 链的尾结点
3488:     MOV     SI,[SI]                        ;|
3489:     JMP     SHORT App_New_Node_1           ;┘
3490: App_New_Node_2:
3491:     MOV     [SI],DI                        ;┐ 将 DS:DI 指定的 BDPB 结点加入到 BDPB 链表
3492:     MOV     [SI].BDPB_NextPtr_SEG,DS       ;┘ 尾
3493:     MOV     WORD PTR [DI],0FFFFH           ;┐ 设置尾结点的指针域
3494:     MOV     [DI].BDPB_NextPtr_SEG,DS       ;┘
3495:     LEA     BX,[DI].BDPB_BytesPerSector1   ;┐
3496:     MOV     SI,Ptr_BPBArray                ;| 设置当前逻辑驱动器的 BPB 参数块指针值
3497:     MOV     [SI],BX                        ;|
3498:     ADD     Ptr_BPBArray,2                 ;┘
3499:     INC     TotalOfBlockDev                ;系统允许访问的逻辑驱动器数加 1
3500:     ADD     Ptr_FreeBDPB,BDPB_Len          ;修改空闲 BDPB 的指针,使之指向下次可供使用的
                                                 ;BDPB
3501:     POP     BX
3502:     POP     SI
3503:     RET
3504: App_New_Node        ENDP
3505: ;=============================================================
3506: ;相对地址偏移:23A8
3507: ;功能:处理实时时钟,并保存当前日期对应的总天数和
3508: ;入口参数:无
3509: ;出口参数:无
3510: ;=============================================================
```

```
3511: Process_RTC        PROC    NEAR
3512:     PUSH    AX
3513:     PUSH    CX
3514:     PUSH    DX
3515:     PUSH    BP
3516:     XOR     BP,BP
3517: Process_RTC1:
3518:     XOR     CX,CX                    ;┐
3519:     XOR     DX,DX                    ;│读实时时钟时间
3520:     MOV     AH,2                     ;│
3521:     INT     1AH                      ;┘
3522:     CMP     CX,0                     ;┐
3523:     JNE     Process_RTC3             ;│若实时时钟工作,则跳转
3524:     CMP     DX,0                     ;│
3525:     JNE     Process_RTC3             ;┘
3526:     CMP     BP,1                     ;┐若实时时钟未工作,则跳转
3527:     JE      Process_RTC4             ;┘
3528:     INC     BP
3529:     MOV     CX,4000H                 ;┐
3530: Process_RTC2:                       ;│延时一段时间
3531:     LOOP    Process_RTC2             ;┘
3532:     JMP     SHORT Process_RTC1       ;转去重新读取实时时钟时间
3533: Process_RTC3:
3534:     MOV     CS:RTCFlag,1             ;置实时时钟工作标志
3535:     CALL    Init_RTC                 ;初始化实时时钟
3536:     PUSH    SI
3537:     CALL    Calcuate_DayCount        ;读取实时时钟日期,并计算当前日期对应的总天
                                           ;数
3538:     CLI                              ;┐
3539:     MOV     DayCount,SI              ;│保存当前日期对应的总天数
3540:     STI                              ;┘
3541:     POP     SI
3542: Process_RTC4:
3543:     POP     BP
3544:     POP     DX
3545:     POP     CX
3546:     POP     AX
3547:     RET
3548: Process_RTC        ENDP
3549: ;=============================================================
3550: ;相对地址偏移:23E6
3551: ;功能:根据机器型号决定是否初始化实时时钟
3552: ;入口参数:无
```

```
3553: ;出口参数:无
3554: ;=============================================================
3555: Init_RTC          PROC    NEAR
3556:     PUSH    AX
3557:     CMP     CS:ModelByte_Init,0FCH          ;┐
3558:     JNE     Init_RTC2                       ;│
3559:     CMP     CS:SubmodelByte_Init,6          ;│若不是AT档次机器,则跳转
3560:     JE      Init_RTC1                       ;│
3561:     CMP     CS:SubmodelByte_Init,4          ;│
3562:     JAE     Init_RTC2                       ;┘
3563: Init_RTC1:
3564:     MOV     AL,8AH                          ;┐设置RTC状态寄存器A的值(即设置基准时钟
3565:     MOV     AH,26H                          ;│频率为32.768kHz,输出时钟频率为1.024kHz)
3566:     CALL    Set_RTCPara                     ;┘
3567:     MOV     AL,8BH                          ;┐取出RTC状态寄存器B的值→AL
3568:     CALL    Get_RTCPara                     ;┘
3569:     AND     AL,7                            ;┐
3570:     MOV     AH,AL                           ;│修改RTC状态寄存器B的值
3571:     MOV     AL,0BH                          ;│
3572:     CALL    Set_RTCPara                     ;┘
3573: Init_RTC2:
3574:     POP     AX
3575:     RET
3576: Init_RTC          ENDP
3577: ;=============================================================
3578: ;相对地址偏移:2416
3579: ;功能:取出AL指定的CMOS RAM地址单元值
3580: ;入口参数:AL=CMOS RAM地址
3581: ;出口参数:AL=指定的CMOS RAM单元内容
3582: ;=============================================================
3583: Get_RTCPara          PROC    NEAR
3584:     PUSHF
3585:     CLI
3586:     PUSH    BX
3587:     PUSH    AX
3588:     OR      AL,80H                  ;NMI无效
3589:     OUT     70H,AL                  ;设置CMOS RAM地址
3590:     NOP
3591:     IN      AL,71H                  ;取出上面指定的CMOS RAM单元内容
3592:     MOV     BX,AX                   ;BL←CMOS ROM单元内容
3593:     POP     AX
3594:     AND     AL,80H                  ;保留"NMI无效/有效"控制位
3595:     OR      AL,0FH                  ;设置CMOS RAM地址(0FH单元存放关机原因)
```

```
3596:    OUT     70H,AL                          ;┐
3597:    NOP                                     ;├ 取出关机原因→AL
3598:    IN      AL,71H                          ;┘
3599:    MOV     AX,BX                           ;AL←指定的 CMOS RAM 单元内容
3600:    POP     BX
3601:    PUSH    CS                              ;┐ 清除当前中断控制
3602:    CALL    Clear_Intrrupt                  ;┘
3603:    RET
3604: Get_RTCPara        ENDP
3605: ;=========================================================
3606: ;相对地址偏移:2435
3607: ;功能:清除当前中断控制
3608: ;入口参数:无
3609: ;出口参数:无
3610: ;=========================================================
3611: Clear_Intrrupt     PROC    NEAR
3612:    IRET
3613: Clear_Intrrupt     ENDP
3614: ;=========================================================
3615: ;相对地址偏移:2436
3616: ;功能:设置 AL 指定的 CMOS RAM 地址单元值
3617: ;入口参数:AL=CMOS RAM 单元的地址(其中:bit6~bit0,表示 CMOS RAM 的地址;bit7=1,表示
          NMI 无效,=0,表示 NMI 有效);
3618: ;          AH=指定单元被设定的值
3619: ;出口参数:无
3620: ;=========================================================
3621: Set_RTCPara      PROC    NEAR
3622:    PUSHF
3623:    PUSH    AX
3624:    CLI
3625:    PUSH    AX
3626:    OR      AL,80H                          ;NMI 无效
3627:    OUT     70H,AL                          ;设置 CMOS RAM 地址
3628:    NOP
3629:    MOV     AL,AH                           ;┐ 向上面指定的 CMOS RAM 单元写数据
3630:    OUT     71H,AL                          ;┘
3631:    POP     AX
3632:    AND     AL,80H                          ;保留"NMI 无效/有效"控制位
3633:    OR      AL,0FH                          ;设置 CMOS RAM 地址(0FH 单元存放关机原因)
3634:    OUT     70H,AL                          ;┐
3635:    NOP                                     ;├ 取出关机原因
3636:    IN      AL,71H                          ;┘
3637:    POP     AX
```

```
3638:    PUSH    CS                              ;┐ 清除当前中断控制
3639:    CALL    Clear_Intrrupt                  ;┘
3640:    RET
3641: Set_RTCPara        ENDP
3642: ;=========================================================
3643: ;相对地址偏移:2453
3644: ;功能:读取当前簇及其后续连续簇的 MSDOS.SYS 系统文件的内容
3645: ;入口参数:BX=当前准备读取的 MSDOS.SYS 系统文件内容对应的起始簇号
3646: ;          CX=每簇扇区数
3647: ;          ES:DI=当前准备读取的 MSDOS.SYS 系统文件内容在内存中存放的地址
3648: ;出口参数:BX=下次读取内容对应的簇号
3649: ;          ES:DI=下次读取 MSDOS.SYS 系统文件内容的存放地址
3650: ;=========================================================
3651: Read_MSDOS       PROC    NEAR
3652:    PUSH    CX
3653:    PUSH    DI
3654:    MOV     CS:Sectors_Init,CX              ;保存准备读取的扇区数
3655:    MOV     AX,BX                           ;┐
3656:    DEC     AX                              ;|
3657:    DEC     AX                              ;| 计算 MSDOS.SYS 系统文件的起始扇区
3658:    MUL     CX                              ;| 号→DX:AX
3659:    ADD     AX,WORD PTR CS:SectorNumber_Init ;|
3660:    ADC     DX,WORD PTR CS:SectorNumber_Init+2 ;┘
3661: Read_MSDOS_1:
3662:    PUSH    DS
3663:    PUSH    AX
3664:    PUSH    BX
3665:    MOV     SI,CS:Seg_FATBuffer             ;┐ DS←读取 FAT 表时使用的传送缓冲区的段地
3666:    MOV     DS,SI                           ;┘ 址
3667:    MOV     SI,BX                           ;SI←当前簇号
3668:    TEST    CS:FATType_Init,40H             ;┐ 若引导软盘或硬盘活动分区使用 16 位 FAT 表,
3669:    JNZ     Read_MSDOS_4                    ;┘ 则跳转
3670:    SHR     SI,1                            ;┐                    ┐
3671:    ADD     SI,BX                           ;| 计算 SI 指定簇号的簇项 | 读取指定簇
3672:    PUSH    DX                              ;| 值在 FAT 表中的偏移   | 号的簇项值
3673:    XOR     DX,DX                           ;┘ →DX:SI              | 所在的扇区
3674:    CALL    ReadASectorOfFAT                ;                      |
3675:    POP     DX                              ;                      ┘
3676:    MOV     AX,[BX]                         ;AX←指定簇号的簇项值
3677:    JNZ     Read_MSDOS_2                    ;若指定簇号的簇项值全部在当前读出的扇区中,
                                                 ;则跳转
3678:    MOV     AL,[BX]                         ;┐ 保存指定簇号的簇项值的低字节值
3679:    MOV     BYTE PTR CS:ClusterNo_Init,AL   ;┘
```

```
3680:    INC     SI                              ;┐
3681:    PUSH    DX                              ;|
3682:    XOR     DX,DX                           ;| 读取指定簇号的簇项值高字节所在的扇区
3683:    CALL    ReadASectorOfFAT                ;|
3684:    POP     DX                              ;┘
3685:    MOV     AL,BYTE PTR DS:[0]              ;┐ 保存指定簇号的簇项值高字节值
3686:    MOV     BYTE PTR CS:ClusterNo_Init+1,AL ;┘
3687:    MOV     AX,CS:ClusterNo_Init            ;AX←指定簇号的簇项值
3688: Read_MSDOS_2:
3689:    POP     BX                              ;┐
3690:    PUSH    BX                              ;| 若是取奇数簇号的簇项值,则跳转
3691:    SHR     BX,1                            ;|
3692:    JNC     Read_MSDOS_3                    ;┘
3693:    SHR     AX,1                            ;┐
3694:    SHR     AX,1                            ;|
3695:    SHR     AX,1                            ;|
3696:    SHR     AX,1                            ;| BX←指定簇号的簇项值
3697: Read_MSDOS_3:                              ;|
3698:    MOV     BX,AX                           ;|
3699:    AND     BX,0FFFH                        ;┘
3700:    JMP     SHORT Read_MSDOS_5
3701: Read_MSDOS_4:                              ;使用 16 位 FAT 表
3702:    PUSH    DX
3703:    XOR     DX,DX                           ;┐ 计算 SI 指定簇号的簇项值在 FAT 表中的偏移
3704:    SHL     SI,1                            ;| →DX:SI
3705:    ADC     DX,0                            ;┘
3706:    CALL    ReadASectorOfFAT                ;读取指定簇号的簇项值所在的扇区
3707:    POP     DX
3708:    MOV     BX,[BX]                         ;BX←指定簇号的簇项值
3709: Read_MSDOS_5:
3710:    POP     SI
3711:    POP     AX
3712:    POP     DS
3713:    SUB     SI,BX                           ;┐ 若 MSDOS.SYS 系统文件的两个逻辑簇在物理
3714:    CMP     SI,0FFFFH                       ;| 上不相邻(即当前簇号与后续簇号不连续),则
3715:    JNE     Read_MSDOS_6                    ;┘ 跳转
3716:    ADD     CS:Sectors_Init,CX              ;累加可一次连续读取的扇区数
3717:    JMP     SHORT Read_MSDOS_1              ;转去继续判定后续簇是否是连续的,以便使得一
                                                 ;次读取的 MSDOS.SYS 文件内容最多,减少读取次
                                                 ;数
3718: Read_MSDOS_6:
3719:    PUSH    BX                              ;                    ┐
3720:    PUSH    DX                              ;                    |
```

```
3721:    PUSH    AX                              ;                          |
3722:    MOV     AX,WORD PTR CS:BootDriveNo      ;AL←引导盘对应的逻辑驱     |
                                                 ;动器号                    | 读取 CX 个扇区
3723:    MOV     CX,CS:Sectors_Init              ;CX←当前欲读取的扇区数     | 的 MSDOS.SYS
3724:    POP     DX                              ;DX←起始扇区号的低字值     | 系统文件内容
3725:    POP     CS:StartingSector_HW            ;起始扇区号的高字值        | 到内存
3726:    PUSH    DS                              ;                          |
3727:    PUSH    CS                              ;                          |
3728:    POP     DS                              ;                          |
3729:    PUSH    CS                              ;                          |
3730:    MOV     BP,offset Disk_Read             ;                          |
3731:    CALL    CallSubInIO2                    ;                          |
3732:    POP     DS                              ;                          |
3733:    POP     BX                              ;                          |
3734:    POP     DI                              ;                          |
3735:    MOV     AX,CS:Sectors_Init              ;┐
3736:    XCHG    AH,AL                           ;| 确定存放下次读取的 MSDOS.SYS
3737:    SHL     AX,1                            ;| 文件内容的内存地址偏移值
3738:    ADD     DI,AX                           ;┘
3739:    POP     CX
3740:    RET
3741: Read_MSDOS          ENDP
3742: ;==========================================================
3743: ;相对地址偏移:2506
3744: ;功能:读取一个扇区的 FAT 表内容
3745: ;入口参数:DX:SI=欲读取的 FAT 表内容的起始位置(字节数)
3746: ;          DS=读取 FAT 表时使用的传送缓冲区的段地址值
3747: ;出口参数:BX=存放簇项值的地址偏移量
3748: ;          ZF=0,簇项值只在当前扇区中;ZF=1,簇项值在两个相邻的扇区中
3749: ;==========================================================
3750: ReadASectorOfFAT    PROC    NEAR
3751:    PUSH    AX
3752:    PUSH    CX
3753:    PUSH    DI
3754:    PUSH    SI
3755:    PUSH    ES
3756:    PUSH    DS
3757:    MOV     AX,SI                           ;┐ 计算相对扇区号,其中 AX=相对扇区号;DX=
3758:    MOV     CX,CS:BytesPerSector_Init       ;| 扇区内的偏移量
3759:    DIV     CX                              ;┘
3760:    NOP
3761:    PUSH    ES
3762:    PUSH    DS
```

```
3763:    PUSH    DI
3764:    PUSH    AX                              ;                          ┐
3765:    PUSH    CS                              ;                          │
3766:    POP     DS                              ;                          │
3767:    MOV     AX,WORD PTR CS:BootDriveNo      ;AL←引导盘对应的逻辑驱     │取出引导盘对应
3768:    MOV     BP,offset Get_BDPB_Node         ;动器号                    │的 BDPB 入口地
3769:    PUSH    CS                              ;                          │址→ES:DI
3770:    CALL    CallSubInIO2                    ;                          │
3771:    POP     AX                              ;                          ┘
3772:    ADD     AX,ES:[DI].BDPB_ReservedSectors1    ;AX←FAT 表的起始扇区号
3773:    POP     DI
3774:    POP     DS
3775:    POP     ES
3776:    CMP     AX,CS:SectorNoOfFAT_Init        ;┐若欲读取的 FAT 表内容所在的扇区先前已读
3777:    JE      ReadASectorOfFAT_1              ;┘出,则跳转
3778:    MOV     CS:SectorNoOfFAT_Init,AX        ;保存当前欲读取的 FAT 表内容对应的起始扇区号
3779:    PUSH    DX
3780:    MOV     CS:StartingSector_HW,0          ;┐设置欲读取的 FAT 表内容所在扇区的扇区号
3781:    MOV     DX,AX                           ;┘
3782:    MOV     CX,1                            ;CX←欲读取 FAT 表内容的扇区数
3783:    MOV     AX,WORD PTR CS:BootDriveNo      ;AL←逻辑驱动器号
3784:    PUSH    DS                              ;┐
3785:    POP     ES                              ;│ ES:DI 指向存放 FAT 表内容的传送缓冲区
3786:    XOR     DI,DI                           ;┘
3787:    PUSH    DS                              ;┐
3788:    PUSH    CS                              ;│
3789:    POP     DS                              ;│
3790:    PUSH    CS                              ;│读取一个扇区的 FAT 表内容
3791:    MOV     BP,offset Disk_Read             ;│
3792:    CALL    CallSubInIO2                    ;│
3793:    POP     DS                              ;┘
3794:    POP     DX
3795:    MOV     CX,CS:BytesPerSector_Init       ;CX←每扇区字节数
3796: ReadASectorOfFAT_1:
3797:    DEC     CX                              ;┐判定簇项值是否存放在两个相邻的扇区中(ZF
3798:    CMP     DX,CX                           ;┘=1,是;ZF=0,不是)
3799:    MOV     BX,DX                           ;BX←簇项值在扇区中的偏移
3800:    POP     DS
3801:    POP     ES
3802:    POP     SI
3803:    POP     DI
3804:    POP     CX
3805:    POP     AX
```

```
3806:     RET
3807: ReadASectorOfFAT    ENDP
3808: ;
3809: SysInt_1_End    EQU        $
3810: ;
3811: IO1   ENDS
3812: ;     END
3813: INCLUDE    IO2.ASM
3814:
3815:
3816:
3817:
```

10.4.5 IO2 模块的源程序注释清单

```
3818: PAGE   60,132
3819: ;Program Name: IO2.ASM
3820: ;Version: MS—DOS5.00
3821: ;Compiler: Marco Assembler Version 5.10
3822: ;
3823: IO2          SEGMENT PARA ' CODE'
3824:              ASSUME  CS:IO2, DS:IO1, ES:IO1
3825:              ORG   30H
3826: IO1_SEG     DW   BIO_SEG                    ;存放 IO1 模块在常规内存低端的段地址值
3827: ;=============================================================
3828: ;相对地址偏移:0032
3829: ;功能:修改被 IO1 模块调用但由 IO2 模块定义的子程序入口地址的段地址值
3830: ;入口参数:AX=IO2 模块当前在内存中的段地址值
3831: ;出口参数:无
3832: ;=============================================================
3833: ModifySegAddr   PROC   FAR
3834:     MOV     ES,CS:IO1_SEG                   ;┐ ES:DI 指向在 IO1 模块中保存由 IO2 模块定义
3835:     MOV     DI,offset PreEntry3+2           ;┘ 的子程序的入口地址表
3836:     MOV     CX,4                            ;CX←子程序个数
3837: ModifySegAddr_1:                            ;┐
3838:     STOSW                                   ;│ 修改在 IO1 模块中保存由 IO2 模块定义的子程
3839:     INC     DI                              ;│ 序的段地址值
3840:     INC     DI                              ;│
3841:     LOOP    ModifySegAddr_1                 ;┘
3842:     RET
3843: ModifySegAddr   ENDP
3844: ;=============================================================
3845: ;相对地址偏移:0043
3846: ;功能:由 DOS BIOS 模块提供的设备驱动程序的统一中断过程
```

```
3847: ;=========================================================
3848: IO2_Interrupt PROC  FAR
3849:     PUSH   SI
3850:     PUSH   AX
3851:     PUSH   CX
3852:     PUSH   DX
3853:     PUSH   DI
3854:     PUSH   BP
3855:     PUSH   DS
3856:     PUSH   ES
3857:     PUSH   BX
3858:     MOV    BP,SP
3859:     MOV    SI,[BP+12H]          ;┐ DS:SI 指向存放"设备驱动程序支持的最大命
3860:     MOV    DS,CS:IO1_SEG        ;┘ 令码"的单元地址
3861:     MOV    AX,[SI+02]           ;┐ 取出并保存"设备号"(如 LPT1=0,LPT2=1,
3862:     MOV    Device_No,AL         ;┘ …)
3863:     MOV    OutDeviceNo,AH       ;保存设备标识号(如 LPT1=1,LPT2=2,…)
3864:     MOV    SI,[SI]              ;CS:SI 指向设备驱动程序处理命令的子程序入口
                                      ;地址表
3865:     LES    BX,DRH_Ptr           ;ES:BX 指向设备驱动程序请求标题
3866:     MOV    AL,ES:[BX].DRH_SubUnit ;AL←块设备的部件号(即逻辑驱动器号,它对字符
                                      ;设备无意义)
3867:     MOV    AH,ES:[BX].DRH_Media ;AH←块设备的介质描述符(对字符设备无意义)
3868:     MOV    CX,ES:[BX].DRH_Count ;CX←计数值(对块设备为扇区个数,对字符设备为
                                      ;字符个数)
3869:     MOV    DX,ES:[BX].DRH_StartingSector ;DX←起始扇区号(对字符设备无意义)
3870:     CMP    SI,offset Disk_CMDNum ;┐ 若当前是请求字符设备驱动程序完成 I/O 操
                                      ;作,则跳转
3871:     JNZ    IO2_Inter1           ;┘
3872:     MOV    StartingSector_HW,0  ;清起始扇区号的高 16 位值(对字符设备无意义)
3873:     CMP    DX,-1                ;┐ 若起始扇区用 16 位表示,则跳转(对字符设备
3874:     JNZ    IO2_Inter1           ;┘ 无意义)
3875:     MOV    DX,WORD PTR ES:[BX].DRH_BigStartingSector+2 ;┐ 设置起始扇区号的高 16 位值
3876:     MOV    StartingSector_HW,DX                        ;┘ (对字符设备无意义)
3877:     MOV    DX,WORD PTR ES:[BX].DRH_BigStartingSector   ;DX←起始扇区号的低 16 位值
                                                             ;(对字符设备无意义)
3878: IO2_Inter1:
3879:     XCHG   AX,DI                ;保存部件号和介质描述符值
3880:     MOV    AL,ES:[BX].DRH_Command ;AL←命令码
3881:     CMP    AL,CS:[SI]           ;┐ 检查命令码的合法性,若是无效的命令码,则跳
3882:     JNB    IO2_Inter4           ;┘ 转
3883:     CBW                         ;┐
3884:     SHL    AX,1                 ;┘ SI←该命令码对应的处理程序的入口地址指针
```

```
3885:     ADD     SI,AX                          ;┘ 值
3886:     XCHG    AX,DI                          ;恢复部件号(即逻辑驱动器号)和介质描述符值
3887:     LES     DI,ES:[BX].DRH_TransAddr       ;ES:DI 指向传送缓冲区
3888:     CLD
3889:     CALL    WORD PTR CS:[SI+1]             ;执行命令码对应的处理程序
3890:     JB      IO2_Inter2                     ;若失败,则跳转
3891:     MOV     AH,1                           ;置完成位
3892: IO2_Inter2:
3893:     MOV     DS,CS:IO1_SEG                  ;┐ DS:BX 指向设备驱动程序请求标题
3894:     LDS     BX,DRH_Ptr                     ;┘
3895:     MOV     [BX].DRH_Status,AX             ;设置设备驱动程序的返回状态字
3896:     POP     BX
3897:     POP     ES
3898:     POP     DS
3899:     POP     BP
3900:     POP     DI
3901:     POP     DX
3902:     POP     CX
3903:     POP     AX
3904:     POP     SI
3905:     ADD     SP,2                           ;废弃子程序调用的返回地址
3906: IO2_Inter3:
3907:     RETF                                   ;中断过程的返回指令(它使控制权返回到请求者,
                                                 ;如 DOS Kernel 模块)
3908: IO2_Inter4:
3909:     CALL    InvalidCmd                     ;设置"错误代码、错误位和完成位"
3910:     JMP     Short IO2_Inter2               ;转去设置设备驱动程序的返回状态字
3911: IO2_Interrupt ENDP
3912:     DB      0,0                            ;未使用
3913: PUBLIC      IO2_00D0
3914: ;相对地址偏移:00D0
3915: IO2_00D0   DB   5 Dup(0)                   ;保存 DOS 的远调用指令(它的内容等于 0:00C0H
                                                 ;~0:00C4H)
3916: ;=========================================================
3917: ;相对地址偏移:00D5
3918: ;功能:设置设备驱动程序返回状态字的"错误位和完成位,以及错误码"
3919: ;入口参数:无
3920: ;出口参数:CF=1,失败,AX=设备驱动程序返回的状态字值
3921: ;=========================================================
3922: InvalidCmd   PROC   NEAR
3923:     MOV     AL,3                           ;AL←错误码 3(非法命令码)
3924: SetErrorDoneBit LABEL   NEAR               ;出错结束处理
3925:     LES     BX,DRH_Ptr                     ;DS:BX 指向设备驱动程序请求标题
```

```
3926:     SUB     ES:[BX].DRH_Count,CX          ;计算实际完成的字节或扇区数
3927:     MOV     AH,81H                        ;设置返回状态字的"错误位和完成位"
3928:     STC
3929:     RET
3930: InvalidCmd  ENDP
3931:     DB      0
3932: ;相对地址偏移:00E4
3933: CON_CMDNum  DB  0BH                       ;CON 设备驱动程序支持的最大命令码
3934: ;CON 设备驱动程序支持的命令码对应的处理程序入口地址表
3935: CON_CMDTab  DB  PUB_NOP_Sub              ;驱动程序初始化
3936:     DW      PUB_NOP_Sub                   ;介质检查
3937:     DW      PUB_NOP_Sub                   ;建立 BPB 参数块
3938:     DW      InvalidCmd                    ;I/O 控制读
3939:     DW      CON_RD                        ;读
3940:     DW      CON_ND_RD                     ;不等待的无破坏性读
3941:     DW      PUB_NOP_Sub                   ;输入状态
3942:     DW      CON_Inp_Flush                 ;刷新输入缓冲区
3943:     DW      CON_WR                        ;写
3944:     DW      CON_WR_Vertify                ;写同时验证
3945:     DW      PUB_NOP_Sub                   ;输入状态
3946: ;相对地址偏移:00FB
3947: PRN_CMDNum  DB  1AH                       ;打印机设备驱动程序支持的最大命令码
3948: ;PRN 设备驱动程序支持的命令码对应的处理程序入口地址表
3949: PRN_CMDTab  DW  PUB_NOP_Sub              ;驱动程序初始化
3950:     DW      PUB_NOP_Sub                   ;介质检查
3951:     DW      PUB_NOP_Sub                   ;建立 BPB 参数块
3952:     DW      InvalidCmd                    ;I/O 控制读
3953:     DW      PRN_RD                        ;读
3954:     DW      PUB_ND_RD                     ;不等待的无破坏性读
3955:     DW      PUB_NOP_Sub                   ;输入状态
3956:     DW      PUB_NOP_Sub                   ;刷新输入缓冲区
3957:     DW      PRN_WR                        ;写
3958:     DW      PRN_WR_Vertify                ;写同时验证
3959:     DW      PRN_Outp_Status               ;输出状态
3960:     DW      PUB_NOP_Sub                   ;刷新输出缓冲区
3961:     DW      PUB_NOP_Sub                   ;I/O 控制写
3962:     DW      PUB_NOP_Sub                   ;设备打开
3963:     DW      PUB_NOP_Sub                   ;设备关闭
3964:     DW      PUB_NOP_Sub                   ;可更换介质
3965:     DW      PRN_Outp_Busy                 ;输出直到忙
3966:     DW      PUB_NOP_Sub                   ;┐保留命令码
3967:     DW      PUB_NOP_Sub                   ;┘
3968:     DW      PRN_Generic_IOCTL             ;类属 IOCTL 请求
```

```
3969:    DW      PUB_NOP_Sub                    ;┐
3970:    DW      PUB_NOP_Sub                    ;|
3971:    DW      PUB_NOP_Sub                    ;| 保留命令码
3972:    DW      PUB_NOP_Sub                    ;|
3973:    DW      PUB_NOP_Sub                    ;┘
3974:    DW      PRN_19H                        ;检查类属 IOCTL 请求的类中子功能码
3975: ;相对地址偏移:0130
3976: AUX_CMDNum  DB  0BH                       ;AUX 设备驱动程序支持的最大命令码
3977: ;AUX 设备驱动程序支持的命令码对应的处理程序入口地址表
3978: AUX_CMDTab  DW  PUB_NOP_Sub              ;驱动程序初始化
3979:    DW      PUB_NOP_Sub                    ;介质检查
3980:    DW      PUB_NOP_Sub                    ;建立 BPB 参数块
3981:    DW      InvalidCmd                     ;I/O 控制读
3982:    DW      AUX_RD                         ;读
3983:    DW      AUX_ND_RD                      ;不等待的无破坏性读
3984:    DW      PUB_NOP_Sub                    ;输入状态
3985:    DW      AUX_Inp_Flush                  ;刷新输入缓冲区
3986:    DW      AUX_WR                         ;写
3987:    DW      AUX_WR_Vertify                 ;写同时验证
3988:    DW      AUX_Outp_Status                ;输出状态
3989: ;相对地址偏移:0147
3990: CLOCK$_CMDNum  DB  0AH                     ;时钟(CLOCK$)设备驱动程序支持的最大命令码
3991: ;CLOCK$设备驱动程序支持的命令码对应的处理程序入口地址表
3992: CLOCK$_CMDTab  DW  PUB_NOP_Sub            ;驱动程序初始化
3993:    DW      PUB_NOP_Sub                    ;介质检查
3994:    DW      PUB_NOP_Sub                    ;建立 BPB 参数块
3995:    DW      InvalidCmd                     ;I/O 控制读
3996:    DW      CLOCK$_RD                      ;读
3997:    DW      PUB_ND_RD                      ;不等待的无破坏性读
3998:    DW      PUB_NOP_Sub                    ;输入状态
3999:    DW      PUB_NOP_Sub                    ;刷新输入缓冲区
4000:    DW      CLOCK$_WR                      ;写
4001:    DW      CLOCK$_WR_Vertify              ;写同时验证
4002: ;==========================================================
4003: ;相对地址偏移:015C
4004: ;功能:"读"CON 设备的处理程序(命令码=4)
4005: ;入口参数:CX=要读取的字节数;
4006: ;          ES:DI 指向传送缓冲区
4007: ;出口参数:CF=0,成功
4008: ;==========================================================
4009: CON_RD  PROC  NEAR
4010:    JCXZ    CRD2                          ;若要读取的字节数为 0,则结束返回
4011: CRD1:
```

```
4012:     CALL    Get Char                    ;读取一个字符(字符的 ASCII 码值存于 AL 中)
4013:     STOSB                               ;将字符存入传送缓冲区中
4014:     LOOP    CRD1                        ;继续读取后续字符
4015: CRD2:
4016:     CLC                                 ;正常结束
4017:     RET
4018: CON RD  ENDP
4019: ;=============================================================
4020: ;相对地址偏移:0166
4021: ;功能:从标准输入设备(键盘)上读取一个字符
4022: ;入口参数:无
4023: ;出口参数:AL=读取的字符 ASCII 码(或扩展键的标志符)
4024: ;          AH=扫描码
4025: ;=============================================================
4026: Get Char  PROC  NEAR
4027: Get Chr Start:
4028:     MOV     AH,ReadKeyCode              ;AH←"读常规键盘输入/读扩展键盘输入"的命令
                                              ;码
4029:     XOR     AL,AL
4030:     XCHG    AL,CON Buffer               ;试图取出缓存的输入字符,并清字符缓冲区
4031:     OR      AL,AL                       ;┐若有缓存字符,则直接返回
4032:     JNZ     Get Chr RET                 ;┘
4033:     INT     16H                         ;等待并从键盘读取一个字符,同时也返回键盘的
                                              ;扫描码
4034:     OR      AX,AX                       ;┐若是 NUL,则转去重新读取一个字符
4035:     JZ      Get Chr Start               ;┘
4036:     CMP     AX,7200H                    ;┐若不是 Print Screen 键,则跳转
4037:     JNE     Get Chr1                    ;┘
4038:     MOV     AL,10H                      ;设置 ASCII 码为 DLE(即^P),当作 Ctrl+P 处理
4039:     JMP     ShortGet Chr RET
4040: Get Chr1:
4041:     CMP     ReadKeyCode,00              ;┐若读常规键盘输入,则跳转
4042:     JZ      Get Chr2                    ;┘
4043:     CMP     AL,0E0H                     ;┐若当前键入的不是扩充键,则跳转
4044:     JNZ     Get Chr2                    ;┘
4045:     OR      AH,AH
4046:     JZ      Get Chr RET
4047:     XOR     AL,AL
4048:     JMP     Short Get Chr3
4049: Get Chr2:
4050:     OR      AL,AL                       ;┐若当前键入的不是扩展键,则直接返回
4051: JNZGet Chr RET                          ;┘
4052: Get Chr3:
```

```
4053:    MOV    CON_Buffer,AH                ;缓存扩展键的扫描码
4054: Get_Chr_RET:
4055:    RET
4056: Get_Char  ENDP
4057: ;=============================================================
4058: ;相对地址偏移:019F
4059: ;功能:"不等待的无破坏性读"CON 设备的处理程序(命令码=5)
4060: ;入口参数:无
4061: ;出口参数:CF=0,成功;CF=1,失败,AH=返回状态字的高字节值
4062: ;=============================================================
4063: CON_ND_Read  PROC  NEAR
4064: CON_ND_RD  LABEL  NEAR                  ;CON 的"不等待的无破坏性读"处理程序入口
4065:    MOV    AL,CON_Buffer                ;取出缓存的输入字符
4066:    OR     AL,AL                        ;┐若有缓存的字符,则跳转
4067:    JNZ    CON_ND_RD5                   ;┘
4068:    MOV    AH,GetKeyStatus              ;┐读取键盘状态
4069:    INT    16H                          ;┘
4070:    JNZ    CON_ND_RD2                   ;若有字符输入,则跳转
4071:    CMP    Support_15H,0                ;┐若 ROM BIOS 不支持"等待事件"的功能请求,则
4072:    JE     CON_ND_RD1                   ;┘跳转
4073:    LES    BX, DRH_Ptr                  ;ES:BX 指向设备驱动程序请求标题
4074:    TEST   WORD PTR ES:[BX].DRH_Status,0400H
4075:    JZ     CON_ND_RD1
4076:    MOV    AX, 4100H
4077:    XOR    BL,BL
4078:    INT    15H
4079: CON_ND_RD1:
4080: PUB_ND_RD  LABEL  NEAR
4081:    STC                                 ;CF←1(失败)
4082:    MOV    AH,03                        ;AH←3(置设备忙位和完成位)
4083:    RET                                 ;失败结束
4084: CON_ND_RD2:
4085:    OR     AX, AX                       ;┐若不是 NUL,则跳转
4086:    JNZ    CON_ND_RD3                   ;┘
4087:    MOV    AH, ReadKeyCode              ;┐取出键盘输入的 NUL
4088:    INT    16H                          ;┘
4089:    JMP    Short CON_ND_RD              ;转去继续执行"不等待的无破坏性读"
4090: CON_ND_RD3:
4091:    CMP    AX, 7200H                    ;┐若不是 Print Screen 键,则跳转
4092:    JNZ    CON_ND_RD4                   ;┘
4093:    MOV    AL, 10H                      ;设置 ASCII 码为 DLE(即^P)
4094:    JMP    Short CON_ND_RD5
4095: CON_ND_RD4:
```

```
4096:    CMP     ReadKeyCode,0                ;┐若是对常规键盘操作,则跳转
4097:    JZ      CON_ND_RD5                   ;┘
4098:    CMP     AL,0E0H                      ;┐
4099:    JNZ     CON_ND_RD5                   ;│
4100:    CMP     AH,0                         ;│转换扩充键的 ASCII 码
4101:    JZ      CON_ND_RD5                   ;│
4102:    MOV     AL,0                         ;│
4103: CON_ND_RD5:                             ;┘
4104:    LES     BX, DRH_Ptr                  ;ES:BX 指向设备驱动程序请求标题
4105:    MOV     ES:[BX+0DH],AL               ;填写返回的字符,即将字符存入设备驱动程序请
                                              ;求标题中
4106: PUB_NOP_Sub  LABEL  NEAR
4107:    CLC                                  ;CF←0(成功)
4108:    RET                                  ;正常结束
4109: CON_ND_Read  ENDP
4110: ;=============================================================
4111: ;相对地址偏移:01FD
4112: ;功能:"写/写同时验证"CON 设备的处理程序(命令码=8/9)
4113: ;入口参数:CX=要写的字符个数;
4114: ;          ES:DI 指向传送缓冲区
4115: ;出口参数:CF=0,成功
4116: ;=============================================================
4117: CON_Write  PROC  NEAR
4118: CON_WR  LABEL  NEAR
4119: CON_WR_Vertify  LABEL  NEAR
4120:    JCXZ    PUB_NOP_Sub                  ;若待写的字符数为 0,则结束返回
4121: CON_WR_Ver1:
4122:    MOV     AL,ES:[DI]                   ;从传送缓冲区中取出一个字符
4123:    INC     DI                           ;修改传送缓冲区的地址
4124:    INT     29H                          ;以电传方式显示该字符
4125:    LOOP    CON_WR_Ver1                  ;继续写操作,直到全部写完
4126: CON_WR_Ver2:
4127:    CLC                                  ;CF←0(成功)
4128:    RET                                  ;正常结束
4129: CON_Write  ENDP
4130: ;=============================================================
4131: ;相对地址偏移:0209
4132: ;功能:"刷新(清洗)CON 的输入缓冲区"的处理程序(命令码=7)
4133: ;入口参数:无
4134: ;出口参数:CF=0,成功
4135: ;=============================================================
4136: CON_Inp_Flush  PROC  NEAR
4137:    MOV     BYTE PTR CON_Buffer,0        ;清 CON 的字符缓冲区
```

```
4138: CON_Inp1:
4139:     MOV     AH,1                    ;┐ 读键盘状态
4140:     INT     16H                     ;┘
4141:     JZ      CON_WR_Ver2             ;若没有键等待被输入,则跳转
4142:     XOR     AH,AH                   ;┐ 从键盘上读取当前字符
4143:     INT     16H                     ;┘
4144:     JMP     SHORT CON_Inp1          ;重新开始,直到键盘缓冲区空为止
4145: CON_Inp_Flush  ENDP
4146: ;=============================================================
4147: ;相对地址偏移:021A
4148: ;功能:"读"标准输出设备的的处理程序(命令码=4)
4149: ;入口参数:无
4150: ;出口参数:CF=0,成功
4151: ;=============================================================
4152: PRN_RD  PROC  NEAR
4153:     CALL    SetErrorDoneBit         ;设置返回状态字的"错误位和完成位"
4154:     CLC
4155:     RET
4156: PRN_RD  ENDP
4157: ;=============================================================
4158: ;相对地址偏移:021F
4159: ;功能:"写/写同时验证"标准输出设备的处理程序(命令码=8/9)
4160: ;入口参数:CX=要写的字符个数;
4161: ;          ES:DI 指向传送缓冲区
4162: ;出口参数:CF=0,成功;CF=1,失败,AH=返回状态字的高字节值
4163: ;=============================================================
4164: PRN_Output  PROC  NEAR
4165: PRN_WR  LABEL  NEAR
4166: PRN_WR_Vertify  LABEL  NEAR
4167:     JCXZ    PRN_Output7             ;若待写的字符数为 0,则结束返回
4168: PRN_Output1:
4169:     MOV     BX,2                    ;BX←2(出错重试次数)
4170: PRN_Output2:
4171:     CALL    GetPRNStatus            ;┐ 若打印机未准备好,则跳转
4172:     JNZ     PRN_Output4             ;┘
4173:     MOV     AL,ES:[DI]              ;取出当前要写的字符
4174:     XOR     AH,AH                   ;┐ 送一字符到指定的打印机
4175:     CALL    PRN_Request             ;┘
4176:     JZ      PRN_Output6             ;若成功地写了一个字符,则跳转
4177:     CMP     AH,-1                   ;┐
4178:     JNZ     PRN_Output3             ;├ 清 CON 设备的输入缓冲区
4179:     MOV     AL,0CH                  ;┤
4180:     MOV     CON_Buffer,0            ;┘
```

```
4181:    JMP     Short PRN_Output5
4182: PRN_Output3:
4183:    TEST    AH,01                          ;┐若打印机未超时,则跳转
4184:    JZ      PRN_Output6                    ;┘
4185: PRN_Output4:
4186:    DEC     BX                             ;┐若出错重试次数未完,则跳转
4187:    JNZ     PRN_Output2                    ;┘
4188: PRN_Output5:
4189:    JMP     SetErrorDoneBit                ;转去设置返回状态字的"错误位和完成位"
4190: PRN_Output6:
4191:    INC     DI                             ;修改传送缓冲区的地址
4192:    LOOP    PRN_Output1                    ;继续写操作,直到全部写完
4193: PRN_Output7:
4194:    CLC                                    ;CF←0(成功)
4195:    RET                                    ;正常结束
4196: ;取标准输出设备的"输出状态"的处理程序(命令码=0AH)
4197: PRN_Outp_Status  LABEL  NEAR
4198:    CALL    GetPRNStatus                   ;┐测试打印机的状态,若打印机未就绪,则跳转
4199:    JNZ     PRN_Output5                    ;┘
4200:    TEST    AH,80H                         ;┐若是打印机空闲,则跳转
4201:    JNZ     PRN_Output7                    ;┘
4202:    JMP     PUB_ND_RD                      ;转去设置"设备忙位和完成位"
4203: PRN_Output  ENDP
4204: ;=============================================================
4205: ;相对地址偏移:025E
4206: ;功能:取打印机状态
4207: ;入口参数:无
4208: ;出口参数:AL=驱动程序返回的错误码;AH=打印机状态
4209: ;          ZF=0,打印机未准备好;ZF=1,打印机已就绪
4210: ;=============================================================
4211: GetPRNStatus  PROC  NEAR
4212:    MOV     AH,2                           ;AH←2(取打印机状态)
4213: PRN_Request  LABEL  NEAR
4214:    MOV     DX,WORD PTR Device_No          ;DX←设备号
4215:    INT     17H                            ;请求 INT 17H 中断服务
4216:    PUSH    AX                             ;┐
4217:    AND     AH,30H                         ;|
4218:    CMP     AH,30H                         ;| 若打印机不处于"已被选择且缺纸"状态,则跳
4219:    POP     AX                             ;| 转
4220:    JNZ     GetPRNS1                       ;┘
4221:    AND     AH,0DFH                        ;清打印机"缺纸"的错误位
4222:    OR      AH,08                          ;置打印机"I/O 错"的错误位
4223: GetPRNS1:
```

```
4224:    TEST    AH,28H                    ;┐ 若打印机"没有 I/O 错且不缺纸",则跳转
4225:    JZ      GetPRNS2                  ;┘
4226:    MOV     AL,9                      ;设置"打印机缺纸"的错误码
4227:    TEST    AH,20H                    ;┐ 若打印机缺纸,则跳转
4228:    JNZ     GetPRNS_RET               ;┘
4229:    INC     AL                        ;设置"写失败"的错误码
4230: GetPRNS_RET:
4231:    RET
4232: GetPRNS2:
4233:    MOV     AL,2                      ;设置"设备未就绪"的错误码
4234:    TEST    AH,1                      ;判定打印机是否超时
4235:    RET
4236: GetPRNStatus  ENDP
4237: ;=============================================================
4238: ;相对地址偏移:028B
4239: ;功能:标准输出设备的"输出直到忙"的处理程序(命令码=10H)
4240: ;入口参数:CX=要输出的字符个数;
4241: ;          ES:DI 指向传送缓冲区
4242: ;出口参数:CF=0,成功;CF=1,失败,AH=返回状态字的高字节值
4243: ;=============================================================
4244: PRN_Outp_Busy  PROC  NEAR
4245:    MOV     SI, DI                    ;ES:SI 指向存放待输出字符的传送缓冲区
4246: PRN_Busy1:
4247:    PUSH    CX
4248:    PUSH    BX                        ;┐
4249:    XOR     BH,BH                     ;│
4250:    MOV     BL,OutDeviceNo            ;│ CX←标准输出设备出错时的重试次数
4251:    SHL     BX,1                      ;│
4252:    MOV     CX,PRN_RepeatCount[BX]    ;│
4253:    POP     BX                        ;┘
4254: PRN_Busy2:
4255:    CALL    GetPRNStatus              ;取出打印机的状态
4256:    JNZ     PRN_Busy3                 ;若设备未准备好(忙)或出错,则跳转
4257:    TEST    AH,80H                    ;┐ 若打印机不忙,则跳转
4258:    LOOPZ   PRN_Busy2                 ;┘
4259:    POP     CX                        ;CX=待输出的字符个数
4260:    JZ      PRN_Busy4
4261:    LODS    ES:BYTE PTR ES:[SI]       ;取出一个待输出的字符
4262:    XOR     AH, AH                    ;┐ 输出一个字符
4263:    CALL    PRN_Request               ;┘
4264:    JNZ     PRN_Busy4                 ;若输出错,则跳转
4265:    LOOP    PRN_Busy1                 ;继续输出,直到设备忙或全部输出完成为止
4266:    CLC                               ;CF←0(成功)
```

```
4267:     RET
4268: PRN_Busy3:
4269:     POP     CX
4270: PRN_Busy4:
4271:     JMP     SetErrorDoneBit                ;转去设置返回状态字的"错误位和完成位"
4272: PRN_Outp_Busy   ENDP
4273: ;=============================================================
4274: ;相对地址偏移:02BA
4275: ;功能:标准输出设备的"类属 IOCTL 请求"的处理程序(命令码=13H)
4276: ;入口参数:无
4277: ;出口参数:CF=0,成功;CF=1,失败,AX=返回状态字的值
4278: ;=============================================================
4279: PRN_Generic_IOCTL   PROC   NEAR
4280:     LES     DI,DRH_Ptr                     ;ES:DI 指向设备驱动程序请求标题
4281:     CMP     BYTE PTR ES:[DI+0DH],5         ;┐若是 LPTx 设备(即打印机设备),则跳转
4282:     JZ      PRN_Gen2                       ;┘
4283: PRN_Gen1:
4284:     JMP     InvalidCmd                     ;转去设置"未知命令"的错误码
4285: PRN_Gen2:
4286:     MOV     AL, BYTE PTR ES:[DI+0EH]       ;取出类属 IOCTL 请求的功能码
4287:     LES     DI, ES:[DI+13H]                ;ES:DI 指向类属 IOCTL 请求包
4288:     XOR     BH, BH                         ;┐
4289:     MOV     BL, OutDeviceNo                ;├ CX=当前指定的 LPTx 设备的出错重试次数
4290:     SHL     BX, 1                          ;│
4291:     MOV     CX, PRN_RepeatCount[BX]        ;┘
4292:     CMP     AL, 65H                        ;┐若是"取出重试次数"的功能,则跳转
4293:     JZ      PRN_Gen3                       ;┘
4294:     CMP     AL, 45H                        ;┐若不是"设置重试次数"的功能,则跳转
4295:     JNZ     PRN_Gen1                       ;┘
4296:     MOV     CX, ES:[DI]                    ;取出重新指定的重试次数值→CX
4297: PRN_Gen3:
4298:     MOV     PRN_RepeatCount[BX],CX         ;设置指定的标准输出设备的重试次数值
4299:     MOV     ES:[DI],CX                     ;设置设备驱动程序的返回值(即指定的标准输出
                                                 ;设备的重试次数)
4300:     CLC                                    ;CF←0(成功)
4301:     RET                                    ;正常结束
4302: PRN_Generic_IOCTL   ENDP
4303: ;=============================================================
4304: ;相对地址偏移:02F0
4305: ;功能:标准输出设备的"检查类属 IOCTL 请求的类中子功能码是否有效"的处理程序(命令码=
          19H)
4306: ;入口参数:无
4307: ;出口参数:CF=0,类属 IOCTL 请求的类中子功能码正确;CF=1,类属 IOCTL 请求的类中子功能码
```

```
                错误,此时 AX=返回状态字的值
;=============================================================
PRN_19H   PROC   NEAR
    LES     DI, DRH_Ptr                 ;ES:DI 指向设备驱动程序请求标题
    CMP     BYTE PTR ES:[DI+0DH],5      ;┐若是 LPTx 设备(即打印机设备),则跳转
    JNZ     PRN_19H2                    ;┘
    MOV     AL, BYTE PTR ES:[DI+0EH]    ;取出类属 IOCTL 请求的功能码
    CMP     AL, 65H                     ;┐若是"取出重试次数"的功能,则跳转
    JZ      PRN_19H1                    ;┘
    CMP     AL, 45H                     ;┐若不是"设置重试次数"的功能,则跳转
    JNZ     PRN_19H2                    ;┘
PRN_19H1:
    CLC                                 ;CF←0(成功)
    RET                                 ;正常结束
PRN_19H2:
    STC                                 ,CF←1(失败)
    JMP     InvalidCmd                  ;转去设置"未知命令"的错误码
PRN_19H   ENDP
;=============================================================
;相对地址偏移:030D
;功能:"读"标准辅助设备的处理程序(命令码=4)
;入口参数:CX=要读取的字符个数;
;         ES:DI 指向传送缓冲区
;出口参数:CF=0,成功,此时读取的数据存于传送缓冲区中
;=============================================================
AUX_RD   PROC   NEAR
    JCXZ    AUX_RD3                     ;若要读的字符数为 0,则结束返回
    CALL    GetChrBuf                   ;取出标准辅助设备的输入字符的暂存单元地址→
                                        ;BX
    XOR     AL, AL                      ;┐取出字符暂存单元中的内容,并清除该暂存单
    XCHG    AL, [BX]                    ;┘元
    OR      AL, AL                      ;┐若暂存单元中有待读取的字符,则跳转
    JNZ     AUX_RD2                     ;┘
AUX_RD1:
    CALL    GetChar                     ;从标准辅助设备上读取一个字符
AUX_RD2:
    STOSB                               ;保存当前字符
    LOOP    AUX_RD1                     ;继续读操作,直到所需的字符全部被读取为止
AUX_RD3:
    CLC                                 ;CF←0(成功)
    RET                                 ;正常结束
AUX_RD   ENDP
;=============================================================
```

```
4349:  ;相对地址偏移:0322
4350:  ;功能:从串行端口上读取一个字符
4351:  ;入口参数:无
4352:  ;出口参数:AL=读取字符的 ASCII 码
4353:  ;=========================================================
4354:  GetChar   PROC   NEAR
4355:      MOV     AH,2                    ;┐ 从指定的串行端口上读取一个字符,同时返回
4356:      CALL    GetChrStatus            ;┘ 串行端口的状态
4357:      TEST    AH,0EH                  ;┐ 检查串行端口的状态。若无错误,则跳转
4358:      JNZ     GetChar1                ;┘
4359:      RET
4360:  GetChar1:
4361:      POP     AX                      ;去掉程序的返回地址,保证堆栈指针正确
4362:      XOR     AL,AL
4363:      OR      AL,0B0H                 ;AL←"读失败"的错误码(此值应为 0BH)
4364:      JMP     SetErrorDoneBit         ;转去设置返回状态字的"错误位和完成位"
4365:  GetChar   ENDP
4366:  ;=========================================================
4367:  ;相对地址偏移:0335
4368:  ;功能:"不等待的无破坏性读"标准辅助设备的处理程序(命令码=5)
4369:  ;入口参数:CX=要读取的字符个数;
4370:  ;          ES:DI 指向传送缓冲区
4371:  ;出口参数:CF=0,成功;CF=1,失败,AH=返回状态字的高字节值
4372:  ;=========================================================
4373:  AUX_ND_RD   PROC   NEAR
4374:      CALL    GetChrBuf               ;取出标准辅助设备的输入字符的暂存单元地址→
                                            ;BX
4375:      MOV     AL, [BX]                ;AL←输入字符的暂存单元中的内容
4376:      OR      AL, AL                  ;┐ 若有先前输入的字符,则跳转
4377:      JNZ     AUX_ND_RD1              ;┘
4378:      CALL    GetPortStatus           ;读取指定串行端口的状态
4379:      TEST    AH, 01                  ;┐ 若串行端口数据未准备好,则跳转
4380:      JZ      AUX_ND_RD2              ;┘
4381:      TEST    AL, 20H                 ;┐ 若 modem 数据集未准备好,则跳转
4382:      JZ      AUX_ND_RD2              ;┘
4383:      CALL    GetChar                 ;从指定的串行端口中读取一个字符→AL
4384:      MOV     [BX], AL                ;缓存当前读取的字符
4385:  AUX_ND_RD1:
4386:      JMP     CON_ND_RD5              ;转去设置驱动程序返回的数据(即设置设备驱动
                                            ;程序请求标题)
4387:  AUX_ND_RD2:
4388:      JMP     PUB_ND_RD               ;转去设置"忙位和完成位"
4389:  AUX_ND_RD   ENDP
```

```
4390: ;=========================================================
4391: ;相对地址偏移:0355
4392: ;功能:取标准辅助设备的"输入状态"的处理程序(命令码=0AH)
4393: ;入口参数:无
4394: ;出口参数:CF=0,成功;
4395: ;=========================================================
4396: AUX_Outp_Status  PROC  NEAR
4397:     CALL    GetPortStatus              ;读取指定的串行端口的状态
4398:     TEST    AL, 20H                    ;┐若 modem 数据集未准备好,则跳转
4399:     JZ      AUX_ND_RD2                 ;┘
4400:     TEST    AH, 20H                    ;┐若发送保持寄存器不空,则跳转
4401:     JZ      AUX_ND_RD2                 ;┘
4402:     CLC                                ;CF←0(成功)
4403:     RET                                ;正常结束
4404: AUX_Outp_Status  ENDP
4405: ;=========================================================
4406: ;相对地址偏移:0363
4407: ;功能:取指定的串行端口的状态
4408: ;入口参数:无
4409: ;出口参数:AH=线路状态,AL=调制解调器的状态
4410: ;=========================================================
4411: GetPortStatus PROC  NEAR
4412:     MOV     AH, 3                      ;AH←3(功能:取串行端口的状态)
4413: GetChrStatus  LABEL  NEAR
4414:     MOV     DX,WORD PTR Device_No      ;DX←设备号
4415:     INT     14H                        ;返回指定的串行端口的状态
4416:     RET
4417: GetPortStatus  ENDP
4418: ;=========================================================
4419: ;相对地址偏移:036C
4420: ;功能:"清洗标准辅助设备的输入缓冲区"的处理程序(命令码=7)
4421: ;入口参数:无
4422: ;出口参数:CF=0,成功
4423: ;=========================================================
4424: AUX_Inp_Flush  PROC  NEAR
4425:     CALL    GetChrBuf                  ;取出标准辅助设备的输入字符的暂存单元地址→
                                             ;BX
4426:     MOV     BYTE PTR [BX], 0           ;清除输入字符的暂存单元内容,即刷新输入缓冲区
4427:     CLC                                ;CF←0(成功)
4428:     RET                                ;正常结束
4429: AUX_Inp_Flush  ENDP
4430: ;=========================================================
4431: ;相对地址偏移:0374
```

```
4432: ;功能:"写/写同时验证"标准辅助设备的处理程序(命令码=8/9)
4433: ;入口参数:CX=要写的字节数;
4434: ;            ES:DI 指向传送缓冲区
4435: ;出口参数:CF=0,成功;CF=1,失败,AX=返回状态字的值
4436: ;=============================================================
4437: AUX_Write  PROC  NEAR
4438: AUX_WR  LABEL  NEAR
4439: AUX_WR_Verify  LABEL  NEAR
4440:     JCXZ    AUX_RD3                    ;若待写的字符数为0,则跳转
4441: AUX_WR_Ver1:
4442:     MOV     AL,ES:[DI]                 ;AL←当前待写的字符
4443:     INC     DI                         ;修改待写内容的传送缓冲区的地址
4444:     MOV     AH,1                       ;┐向指定的串行端口写一个字符,并返回串行端
4445:     CALL    GetChrStatus               ;┘口的当前状态
4446:     TEST    AH,80H                     ;┐若操作成功,则跳转
4447:     JZ      AUX_WR_Ver2                ;┘
4448:     MOV     AL,0AH                     ;设置"写失败"的错误码
4449:     JMP     SetErrorDoneBit            ;转去设置返回状态字的"错误位和完成位"
4450: AUX_WR_Ver2:
4451:     LOOP    AUX_WR_Ver1                ;继续写操作,直到全部写完为止
4452:     CLC                                ;CF←0(成功)
4453:     RET                                ;正常结束
4454: AUX_Write  ENDP
4455: ;=============================================================
4456: ;相对基址偏移:038D
4457: ;功能:取出标准辅助设备的输入字符的暂存单元地址
4458: ;入口参数:无
4459: ;出口参数:BX=当前操作所指定的标准辅助设备的输入字符暂存单元地址
4460: ;=============================================================
4461: GetChrBuf  PROC  NEAR
4462:     MOV     BX,WORD PTR Device_No      ;BX<-当前操作所指定的串行端口号
4463:     ADD     BX,offset AUX_Buffer       ;BX←对应的暂存单元地址
4464:     RET
4465: GetChrBuf  ENDP
4466: ;=============================================================
4467: ;相对地址偏移:0396
4468: ;功能:将时间转换成时钟计数值
4469: ;入口参数:CH=小时,CL=分,DH=秒,DL=百分秒
4470: ;出口参数:CX:DX=时钟计数值,其中CX为高字,DX为低字
4471: ;=============================================================
4472: CalculateClockCount  PROC  FAR
4473:     MOV     AL,60                      ;┐
4474:     MUL     CH                         ;┘将小时转换成分,[illegible]存于AX中
```

```
4475:    MOV     CH,0                    ;|
4476:    ADD     AX,CX                   ;┘
4477:    MOV     CX,6000                 ;1 分钟=60 秒=6000 百分秒
4478:    MOV     BX,DX                   ;BX=秒,BL=百分秒
4479:    MUL     CX                      ;将分转换成百分秒→DX:AX
4480:    MOV     CX,AX                   ;将百分秒数的低字→CX
4481:    MOV     AL,100                  ;1 秒=100 百分秒
4482:    MUL     BH                      ;将秒转换成百分秒
4483:    ADD     CX,AX                   ;┐
4484:    ADC     DX,0                    ;|
4485:    MOV     BH,0                    ;| 计算总的百分秒数→DX:CX
4486:    ADD     CX,BX                   ;|
4487:    ADC     DX,0                    ;┘
4488:    XCHG    AX,DX                   ;┐ 总的百分秒数→CX:DX
4489:    XCHG    AX,CX                   ;┘
4490:    MOV     BX,59659                ;┐
4491:    MUL     BX                      ;|
4492:    XCHG    DX,CX                   ;|
4493:    XCHG    AX,DX                   ;| 计算百分秒×59659÷65536→AX:DX
4494:    MUL     BX                      ;|
4495:    ADD     AX,CX                   ;|
4496:    ADC     DX,0                    ;|
4497:    XCHG    AX,DX                   ;┘
4498:    MOV     BX,5                    ;┐                          ┐
4499:    DIV     BL                      ;| 计算时钟计数值的高字      |
4500:    MOV     CL,AL                   ;| 部分→CX                   |
4501:    MOV     CH,0                    ;┘                          | 百分秒×59659÷
4502:    MOV     AL,AH                   ;┐                          | 65536÷5=秒×
4503:    CBW                             ;| 计算时钟计数值的低字      | 18.20648193
4504:    XCHG    AX,DX                   ;| 部分→DX                   |
4505:    DIV     BX                      ;|                          |
4506:    MOV     DX,AX                   ;┘                          ┘
4507:    RET
4508: CalculateClockCount  ENDP
4509: ;=============================================================
4510: ;相对地址偏移:03DB
4511: ;功能:"写/写同时验证"时钟设备的处理程序(命令码=8/9)
4512: ;入口参数:传送缓冲区的内容格式如下:
4513: ;  偏移地址  长度      内容
4514: ;     0       DW       日期值(从 1980 年 1 月 1 号算起)
4515: ;     2       DB       分
4516: ;     3       DB       小时
4517: ;     4       DB       百分秒
```

```
;    5        DB        秒
;出口参数:CF=0,成功
;=============================================================
CLOCK$_Write   PROC   NEAR
CLOCK$_WR   LABEL   NEAR
CLOCK$_WR_Vertify   LABEL   NEAR
    MOV     AX, ES:[DI]                 ;AX←日期
    PUSH    AX                          ;
    CMP     BYTE PTR RTCFlag, 0         ;┐若系统中没有实时时钟,则跳转
    JZ      CLOCK$_WR_Ver1              ;┘
    MOV     AL, ES:[DI+3]               ;┐
    CALL    DWORD PTR PreEntry1         ;├将小时数据转换成 BCD 格式→CH
    MOV     CH, AL                      ;┘
    MOV     AL, ES:[DI+2]               ;┐
    CALL    DWORD PTR PreEntry1         ;├将分数据转换成 BCD 格式→CL
    MOV     CL, AL                      ;┘
    MOV     AL, ES:[DI+5]               ;┐
    CALL    DWORD PTR PreEntry1         ;├将秒数据转换成 BCD 格式→DH
    MOV     DH, AL                      ;┘
    MOV     DL, 0                       ;没有白昼时间选择(Daylight savings time option)
    CLI                                 ;┐
    MOV     AH, 03                      ;├设置实时时钟时间
    INT     1AH                         ;│
    STI                                 ;┘
CLOCK$_WR_Ver1:
    MOV     CX, ES:[DI+2]               ;小时→CH,分→CL
    MOV     DX, ES:[DI+4]               ;秒→DH,百分秒→DL
    CALL    DWORD PTR PreEntry4         ;将时间数据转换成时钟计数值→CX:DX
    CLI
    MOV     AH, 1                       ;┐设置时钟计数值
    INT     1AH                         ;┘
    POP     DayCount                    ;设置日期
    STI
    CMP     BYTE PTR RTCFlag, 0         ;┐若系统中没有实时时钟,则跳转
    JZ      CLOCK$_WR_Ver2              ;┘
    CALL    DWORD PTR PreEntry2         ;将二进制表示的日期数据转换成 BCD 格式
    CLI                                 ;┐
    MOV     AH, 5                       ;├设置实时时钟日期
    INT     1AH                         ;│
    STI                                 ;┘
CLOCK$_WR_Ver2:
    CLC                                 ;CF←0(成功)
    RET                                 ;正常结束
```

```
4561: CLOCK$  Write  ENDP
4562: ;=========================================================
4563: ;相对地址偏移:0435
4564: ;功能:"读"时钟设备的处理程序(命令码=4)
4565: ;入口参数:无
4566: ;出口参数:CF=0,成功,此时传送缓冲区的内容如下:
4567: ;  偏移地址  长度      内容
4568: ;     0       DW      日期值(从 1980 年 1 月 1 号算起)
4569: ;     2       DB      分
4570: ;     3       DB      小时
4571: ;     4       DB      百分秒
4572: ;     5       DB      秒
4573: ;=========================================================
4574: CLOCK$_RD  PROC  NEAR
4575:     CALL    ReadClockCount          ;读取时钟计数值并校正日期数据值
4576:     MOV     SI, DayCount            ;SI←日期值
4577:     MOV     AX, CX                  ;┐
4578:     MOV     BX, DX                  ;│
4579:     SHL     DX, 1                   ;│
4580:     RCL     CX, 1                   ;│
4581:     SHL     DX, 1                   ;│ 将时钟计数值×5→DX:AX
4582:     RCL     CX, 1                   ;│
4583:     ADD     DX, BX                  ;│
4584:     ADC     AX, CX                  ;│
4585:     XCHG    AX, DX                  ;┘
4586:     MOV     CX, 59659               ;59659÷5=11931.8(即 8253—5 计时芯片的时
                                          ;钟)
4587:     DIV     CX                      ;┐
4588:     MOV     BX, AX                  ;│
4589:     XOR     AX, AX                  ;│ (DX:AX)×65536÷59659→DX:AX
4590:     DIV     CX                      ;│
4591:     MOV     DX,BX                   ;┘
4592:     MOV     CX, 200                 ;DX:AX=百分秒数值,CX=200 百分秒
4593:     DIV     CX                      ;AX=秒(以 2 秒为单位)
4594:     CMP     DL, 100                 ;┐
4595:     JB      CLOCK$_RD1              ;│ DX=百分秒
4596:     SUB     DL, 100                 ;┘
4597: CLOCK$_RD1:
4598:     CMC                             ;决定进位标志
4599:     MOV     BL,DL                   ;BL←百分秒
4600:     RCL     AX,1                    ;┐
4601:     MOV     DL,0                    ;│ DX:AX=秒数
4602:     RCL     DX,1                    ;┘
```

```
4603:    MOV     CX,60                          ;┐ 将秒转换成分,AX=分,DX=秒
4604:    DIV     CX                             ;┘
4605:    MOV     BH,DL                          ;BH←秒
4606:    DIV     CL                             ;将分转换成小时,AL=小时,AH=分
4607:    XCHG    AL,AH                          ;AH=小时,AL=分
4608:    PUSH    AX
4609:    MOV     AX,SI                          ;┐ 将日期值存入传送缓冲区中
4610:    STOSW                                  ;┘
4611:    POP     AX                             ;┐ 将小时和分存入传送缓冲区中
4612:    STOSW                                  ;┘
4613:    MOV     AX,BX                          ;┐ 将秒和百分秒存入传送缓冲区中
4614:    STOSW                                  ;┘
4615:    CLC                                    ;CF←0(成功)
4616:    RET                                    ;正常结束
4617: CLOCK$  RD  ENDP
4618: ;=========================================================
4619: ;相对地址偏移:0486
4620: ;功能:读取时钟计数值,同时还校正日期数据值
4621: ;入口参数:无
4622: ;出口参数:CX:DX=时钟计数值
4623: ;=========================================================
4624: ReadClockCount  PROC  NEAR
4625:    XOR     AH, AH                         ;┐ 读时钟计数值
4626:    INT     1AH                            ;┘
4627:    CMP     Switch_T,00                    ;┐ 若系统配置命令 switches 指定了"/T"开关(即
                                                ;| switches=/T,它表明当 AL>0 时,日期天数加
4628:    JNZ     ReadClockCount_1               ;┘ 1),则跳转
4629:    XOR     AH,AH                          ;┐ 累计日期值
4630:    ADD     DayCount,AX                    ;┘
4631:    RET
4632: ReadClockCount_1:
4633:    OR      AL, AL                         ;┐ 若自开电源以来未超过 24 小时,则跳转
4634:    JZ      ReadClockCount_2               ;┘
4635:    INC     WORD PTR DayCount              ;日期单元值加 1
4636: ReadClockCount_2:
4637:    RET
4638: ReadClockCount  ENDP
4639:    DB      0
4640: ;相对地址偏移:04A2
4641: Disk_CMDNum  DB  1AH                       ;块设备驱动程序支持的最大命令码
4642: ;块设备驱动程序支持的命令码对应的处理程序入口地址表
4643: Disk_CMDTab  DW  Disk_Init                 ;块设备驱动程序的初始化
4644:    DW      Disk_Media_Check               ;介质检查
```

```
4645:     DW      Disk_Build_BPB                    ;建立 BPB 参数块
4646:     DW      InvalidCmd                        ;I/O 控制读
4647:     DW      Disk_RD                           ;读
4648:     DW      Disk_ND_RD                        ;不等待的无破坏性读
4649:     DW      Disk_NOP_Sub                      ;输入状态
4650:     DW      Disk_NOP_Sub                      ;刷新输入缓冲区
4651:     DW      Disk_WR                           ;写
4652:     DW      Disk_WR_Vertify                   ;写同时验证
4653:     DW      Disk_NOP_Sub                      ;输出状态
4654:     DW      Disk_NOP_Sub                      ;刷新输出缓冲区
4655:     DW      InvalidCmd                        ;I/O 控制写
4656: Disk_CMD0DH  DW  Disk_Dev_Open                 ;设备打开
4657:     DW      Disk_Dev_Close                    ;设备关闭
4658:     DW      Disk_Rem_Media                    ;可更换介质
4659:     DW      Disk_NOP_Sub                      ;输出直到忙
4660:     DW      Disk_NOP_Sub                      ;┐保留命令码
4661:     DW      Disk_NOP_Sub                      ;┘
4662:     DW      Disk_Generic_IOCTL                ;类属(一般的)IOCTL 请求
4663:     DW      Disk_NOP_Sub                      ;┐
4664:     DW      Disk_NOP_Sub                      ;├保留命令码
4665:     DW      Disk_NOP_Sub                      ;┘
4666:     DW      Disk_Get_Map                      ;取逻辑驱动器
4667:     DW      Disk_Set_Map                      ;设置逻辑驱动器
4668:     DW      Disk_19H                          ;检查类属 IOCTL 请求的类中子功能码
4669: ;=================================================
4670: ;相对地址偏移:04D7
4671: ;功能:取出 AL 指定的逻辑驱动器对应的 BDPB 的入口地址
4672: ;入口参数:AL=逻辑驱动器号
4673: ;出口参数:CF=0,成功,ES:DI 指向对应的 BDPB;CF=1,失败
4674: ;=================================================
4675: Get_BDPB_Node  PROC  NEAR
4676:     LES     DI,DWORD PTR BDPB_Head            ;ES:DI 指向第一个 BDPB
4677: Get_BDPB_Node1:
4678:     CMP     ES:[DI].BDPB_LogicalNumber,AL     ;┐若找到指定逻辑驱动器对应的 BDPB,则跳转
4679:     JE      Get_BDPB_Node2                    ;┘
4680:     LES     DI,DWORD PTR ES:[DI].BDPB_NextPtr_OFS   ;ES:DI 指向下一个 BDPB
4681:     CMP     DI,-1                             ;┐若当前结点不是尾结点,则转去继续查找
4682:     JNE     Get_BDPB_Node1                    ;┘
4683:     STC                                       ;置未找到对应的 BDPB 的错误标志
4684: Get_BDPB_Node2:
4685:     RET
4686: Get_BDPB_Node  ENDP
4687: ;=================================================
```

```
4688:  ;相对地址偏移:04EB
4689:  ;功能:"介质检查"的处理程序(块设备的命令码=1)
4690:  ;入口参数:AL=逻辑驱动器号
4691:  ;出口参数:CF=0,成功;CF=1,失败,AX=设备驱动程序返回的状态字值
4692:  ;=================================================
4693:  Disk_Media_Check PROC  NEAR
4694:      CALL    Get_BDPB_Node                   ;取出 AL 指定的逻辑驱动器对应的 BDPB
4695:      MOV     SI, 0001                        ;置介质更换码为 1(表示盘片未被更换)
4696:      TEST    BYTE PTR ES:[DI].BDPB_Attr_Status+1,01   ;┐ 若介质的磁道未被重新格式化,
4697:      JZ      Disk_Media_Chk1                          ;┘ 则跳转
4698:      AND     WORD PTR ES:[DI].BDPB_Attr_Status, 0FEFFH;清除介质格式化的标志位
4699:      MOV     BYTE PTR Drive_Number,-1        ;置介质更换标志
4700:      TEST    BYTE PTR ES:[DI].BDPB_Attr_Status, 1 ;┐ 若介质是可装卸的,则跳转
4701:      JZ      Disk_Media_Chk2                      ;┘
4702:      MOV     SI, -1                          ;置介质更换码为-1(表示介质已更换)
4703:      JMP     Short Disk_Media_Chk5
4704:  Disk_Media_Chk1:
4705:      TEST    BYTE PTR ES:[DI].BDPB_Attr_Status, 1 ;┐ 若驱动器是硬盘(即介质不可装卸),
4706:      JNZ     Disk_Media_Chk5                      ;┘ 则跳转
4707:  Disk_Media_Chk2:
4708:      XOR     SI, SI                          ;置介质更换码为 0(表示不知介质更换否)
4709:      CMP     ChangeLine,0                    ;┐ 若软盘驱动器不支持 Change Line,则跳转
4710:      JZ      Disk_Media_Chk3                 ;┘
4711:      CALL    ChkMediaRemove                  ;检查软盘介质是否被更换
4712:      JB      Disk_Media_Chk8                 ;若盘操作出错,则跳转
4713:      CALL    Test_ChangeLineStatus           ;┐ 若软盘驱动器支持 Change Line,则跳转
4714:      JNZ     Disk_Media_Chk5                 ;┘
4715:  Disk_Media_Chk3:
4716:      MOV     SI, 1                           ;置介质更换码为 1(表示介质未更换)
4717:      MOV     AL, Drive_Number                ;┐ 若检查的驱动器不是当前操作的驱动器(它表
4718:      CMP     AL, ES:[DI].BDPB_DriveNumber    ;│ 明不能确定介质是否被更换),则跳转
4719:      JNZ     Disk_Media_Chk4                 ;┘
4720:      CALL    Set_MediaChangedCode            ;依据同一软盘驱动器最近两次相邻的盘操作间隔
                                                   ;时间是否超过 2 秒来设置介质更换码
4721:      JMP     Short Disk_Media_Chk5
4722:  Disk_Media_Chk4:
4723:      DEC     SI                              ;置介质更换码为 0(表示不知介质更换否)
4724:  Disk_Media_Chk5:
4725:      PUSH    ES                              ;┐
4726:      LES     BX,DRH_Ptr                      ;│ 设置设备驱动程序返回的介质更换码
4727:      MOV     ES:[BX+0EH],SI                  ;│
4728:      POP     ES                              ;┘
4729:      OR      SI,SI                           ;┐ 若介质未被更换或不能断定介质是否被更换,
```

```
4730:     JNS     Disk_Media_Chk7                    ;┘ 则跳转
4731:     CMP     ChangeLine,0                       ;┐ 若软盘驱动器不支持 Change Line,则跳转
4732:     JZ      Disk_Media_Chk6                    ;┘
4733:     CALL    Set_LastVolumeID                   ;设置设备驱动程序返回的先前存取介质的卷标 ID
                                                     ;指针值
4734: Disk_Media_Chk6:
4735:     MOV     Drive_Number,-1                    ;置介质更换标志
4736: Disk_NOP_Sub  LABEL  NEAR
4737: Disk_Media_Chk7:
4738:     CLC                                        ;CF←0(成功)
4739:     RET                                        ;正常结束
4740: Disk_Media_Chk8:
4741:     CALL    Set_CodeOfDriver                   ;将 INT 13H 返回的错误码转换成设备驱动程序返
                                                     ;回的错误码
4742: Disk_Media_Chk9:
4743:     MOV     AH,81H                             ;设置设备驱动程序返回的状态字高字节值(置错
                                                     ;误位和完成位)
4744:     RET
4745: Disk_Media_Check ENDP
4746: ;=============================================================
4747: ;相对地址偏移:0560
4748: ;功能:根据同一软盘驱动器最近两次相邻的操作时间间隔是否小于 2 秒来设置介质更换码
4749: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
4750: ;出口参数:SI=介质更换码(0:未知介质更换否;1:介质未被更换)
4751: ;=============================================================
4752: Set_MediaChangedCode  PROC  NEAR
4753:     MOV     SI,1                               ;置介质更换码为 1(表示介质未被更换)
4754:     CALL    ReadClockCount                     ;读取时钟计数值→CX:DX
4755:     MOV     AX,ES:[DI].BDPB_Clock1orCylinderFlag   ;┐
4756:     SUB     DX,AX                                  ;├ 计算两次盘操作之间的间隔时钟计数
4757:     MOV     AX,ES:[DI].BDPB_Clock2orCylinder       ;├ 值→CX:DX
4758:     SBB     CX,AX                                  ;┘
4759:     JNZ     SMCC2                              ;若间隔时钟计数值>65536 时,则跳转
4760:     OR      DX,DX                              ;┐ 若间隔时钟计数值≠0,则跳转
4761:     JNZ     SMCC1                              ;┘
4762:     INC     Operate_Times                      ;时钟计数值未变化的计数器值加 1
4763:     CMP     Operate_Times,5                    ;┐ 若时钟在工作,但两次盘操作之间的间隔时间
4764:     JB      SMCC_RET                           ;┘ 很短(<<2 秒),则直接返回
4765:     DEC     Operate_Times                      ;┐ 若时钟可能不工作,则跳转
4766:     JMP     SHORT SMCC2                        ;┘
4767: SMCC1:
4768:     CMP     DX,24H                             ;┐ 若两次盘操作的间隔时间<2 秒,则直接返回
4769:     JBE     SMCC_RET                           ;┘
```

```
4770: SMCC2:
4771:     DEC     SI                              ;置介质更换码为 0(表示不知介质更换否)
4772: SMCC_RET:
4773:     RET
4774: Set_MediaChangedCode  ENDP
4775: ;=============================================================
4776: ;相对地址偏移:0592
4777: ;功能:"建立 BPB 参数块"的处理程序(块设备的命令码=2)
4778: ;入口参数:AL=逻辑驱动器号
4779: ;出口参数:CF=0,成功
4780: ;=============================================================
4781: Build_BPB  PROC  NEAR
4782: ;Build_BPB1:
4783:     JMP     Short Disk_Media_Chk8           ;无用指令
4784: Disk_Build_BPB  LABEL  NEAR                 ;建立 BPB 参数块
4785:     MOV     AH, ES:[DI]                     ;无用指令
4786:     CALL    Get_BDPB_Node                   ;取出 AL 指定的逻辑驱动器对应的 BDPB
4787:     TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,1   ;┐若介质不可装卸,则跳转
4788:     JNZ     Build_BPB2                      ;┘
4789:     CALL    Set_BDPBDefExtMediaState        ;设置 BDPB 的"缺省卷系列号、缺省卷标名,以及文
                                                  ;件系统类型标志串"
4790:     MOV     ChangeFlag,1                    ;┐建立 Build BPB 参数块,并且还设置"卷系列号、
4791:     CALL    Modify_BuildBPB                 ;┘卷标名和文件系统类型标志串"
4792:     JB      Disk_Media_Chk9                 ;若建立 Build BPB 参数块失败,则跳转
4793:     CMP     ChangeFlag,02                   ;┐
4794:     MOV     ChangeFlag,00                   ;├若已设置了"卷系列号、卷标名和文件系统类型
4795:     JZ      Build_BPB2                      ;┘标志串",则跳转
4796:     CMP     ChangeLine,0                    ;┐若软盘驱动器不支持 Change Line,则跳转
4797:     JZ      Build_BPB2                      ;┘
4798:     CALL    VolumeIDtoBDPB                  ;设置 BDPB 的卷标名
4799: Build_BPB2:
4800:     ADD     DI, 6                           ;ES:DI 指向 Build BPB 参数块
4801: Build_BPB3:
4802:     MOV     CX,ES
4803:     LES     BX,DRH_Ptr                      ;ES:BX 指向设备驱动程序请求标题
4804:     MOV     ES:[BX+0DH],AH                  ;设置介质描述符/块设备的部件数
4805:     MOV     ES:[BX+12H],DI                  ;┐设置块设备新的 Build BPB 参数块指针值/BPB
4806:     MOV     ES:[BX+14H],CX                  ;┘参数块指针数组的起始地址
4807:     CLC
4808:     RET                                     ;正常结束
4809: Build_BPB  ENDP
4810: ;=============================================================
4811: ;相对地址偏移:05D9
```

```
4812: ;功能:设置 BDPB 的"缺省卷系列号、缺省卷标名和文件系统类型标志串"
4813: ;入口参数:DS=IO1 模块在常规内存低端的段地址值
4814: ;          ES:DI 指向指定逻辑驱动器对应的 BDPB
4815: ;出口参数:无
4816: ;=========================================================
4817: Set_BDPBDefExtMediaState  PROC  NEAR
4818:     PUSH    DI
4819:     XOR     CX,CX                                  ;┐
4820:     MOV     WORD PTR ES:[DI].BDPB_SerialNumber,CX  ;├缺省卷系列号→BDPB
4821:     MOV     WORD PTR ES:[DI].BDPB_SerialNumber+2,CX;┘
4822:     MOV     CX,FileNameLen                 ;┐
4823:     MOV     SI,offset DefaultVolume        ;├缺省卷标名→BDPB
4824:     ADD     DI,4BH                         ;┘
4825:     REP     MOVSB
4826:     TEST    BYTE PTR ES:[DI].BDPB_SizeFlag,40H;                    ┐
4827:     MOV     SI,offset FAT_16Bits              ;SI 指向 16 位 FAT 表的|
                                                    ;文件系统标志串       |
4828:     JNZ     Set_BDPBDefExtMediaState_1        ;                     |
4829:     MOV     SI,offset FAT_12Bits              ;SI 指向 12 位 FAT 表的|"文件系统类型标志
                                                    ;文件系统标志串       |串"→BDPB
4830: Set_BDPBDefExtMediaState_1:                   ;                     |
4831:     MOV     CX,8                              ;CX←文件系统标志串    |
                                                    ;长度                 |
4832:     ADD     DI,5                              ;ES:DI 指向 BDPB 中存|
                                                    ;放文件系统标志串的   |
                                                    ;缓冲区               |
4833:     REP     MOVSB                             ;                     ┘
4834:     POP     DI
4835:     RET
4836: Set_BDPBDefExtMediaState  ENDP
4837: ;=========================================================
4838: ;相对地址偏移:0606
4839: ;功能:根据引导记录或介质描述符修改 Build BPB 参数块的参数值
4840: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
4841: ;出口参数:CF=0,成功;CF=1,失败
4842: ;=========================================================
4843: Modify_BuildBPB  PROC  NEAR
4844:     TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,5  ;┐若软盘驱动器的 Default BPB 参数块被
4845:     JZ      Modify_BuildBPB1                     ;┘修改了,则跳转
4846:     JMP     Modify_BuildBPB_RET          ;直接返回
4847: Modify_BuildBPB1:
4848:     PUSH    CX
4849:     PUSH    DX
```

```
4850:    PUSH    BX
4851:    CALL    Read_BootstrapSector2          ;读取引导记录
4852:    JC      Modify_BuildBPB2               ;若读操作失败,则跳转
4853:    OR      BX,BX                          ;┐若第一个扇区不是引导记录,则跳转
4854:    JNE     Modify_BuildBPB3               ;┘
4855:    CALL    Get_BPB_Para                   ;取出引导记录中 BPB 参数块的参数值
4856:    JMP     SHORT Modify_BuildBPB8         ;转去设置 Build BPB 参数块的参数值
4857: Modify_BuildBPB2:
4858:    JMP     Modify_BuildBPB10
4859: Modify_BuildBPB3:
4860:    CALL    Read_SecondSector              ;从第 2 个扇区中取出介质描述符
4861:    JC      Modify_BuildBPB2               ;若读第 2 个扇区时出错,则跳转
4862:    CMP     ChangeLine,0                   ;┐若软盘驱动器不支持 Change Line,则跳转
4863:    JZ      Modify_BuildBPB4               ;┘
4864:    CALL    Update_BuildBPB                ;依据软盘驱动器的类型决定是否修改 BPB 参数块
                                                ;的参数值
4865: Modify_BuildBPB4:
4866:    CMP     ES:[DI].BDPB_DeviceType,2      ;┐若软盘驱动器不是 720KB(3.5 英寸),则跳转
4867:    JNE     Modify_BuildBPB6               ;┘
4868:    CMP     AH,0F9H                        ;┐若不是高密软盘驱动器,则跳转
4869:    JNE     Modify_BuildBPB11              ;┘
4870: Modify_BuildBPB5:                         ;处理高密软盘驱动器存取低密盘片的情况
4871:    MOV     BX, offset HDtoDD_BPB_Para     ; BX 指向高密软盘驱动器作低密用时 BPB 参数块
                                                ;的参数值表
4872:    DB      8AH,87H                        ;┐
4873:    DW      0                              ;| AL←每个 FAT 表占用的扇区数
4874:    ;MOV    AL,[BX+0000H]                  ;┘
4875:    DB      8BH,8FH                        ;┐
4876:    DW      3                              ;| CX←总扇区数
4877:    ;MOV    CX,[BX+0003H]                  ;┘
4878:    DB      8BH,97H                        ;┐
4879:    DW      5                              ;| DH←每簇扇区数,DL←磁头数
4880:    ;MOV    DX,[BX+0005H]                  ;┘
4881:    DB      8BH,9FH                        ;┐
4882:    DW      1                              ;| BH←最大根目录项数,BL←每道扇区数
4883:    ;MOV    BX,[BX+0001H]                  ;┘
4884:    JMP     SHORT Modify_BuildBPB8
4885: Modify_BuildBPB6:
4886:    CMP     AH,0F8H                        ;┐若介质描述符不正确,则跳转
4887:    JC      Modify_BuildBPB11              ;┘
4888:    MOV     AL,1                           ;┐
4889:    MOV     BX,4008H                       ;| 单面、每道 8 个扇区的盘片的部分参数
4890:    MOV     CX,140H                        ;|
```

```
4891:    MOV     DX,101H                    ;┘
4892:    TEST    AH,2                       ;┐ 若每道 8 个扇区,则跳转
4893:    JNZ     Modify_BuildBPB7           ;┘
4894:    INC     AL                         ;每个 FAT 表占用的扇区数为 2
4895:    INC     BL                         ;每道扇区数为 9
4896:    ADD     CX,40                      ;总的扇区数为 360 个扇区
4897: Modify_BuildBPB7:
4898:    TEST    AH,1                       ;┐ 若是单面软盘,则跳转
4899:    JZ      Modify_BuildBPB8           ;┘
4900:    ADD     CX,CX                      ;总的扇区数加倍
4901:    MOV     BH,70H                     ;最大根目录项数为 112
4902:    INC     DH                         ;每簇扇区数为 2
4903:    INC     DL                         ;磁头数为 2
4904: Modify_BuildBPB8:                     ;设置 Build BPB 参数块的参数值
4905:    MOV     ES:[DI].BDPB_SectorsPerCluster1,DH          ;每簇扇区数→BDPB
4906:    MOV     BYTE PTR ES:[DI].BDPB_TotalOfRootDir1,BH    ;最大根目录项数→BDPB
4907:    MOV     ES:[DI].BDPB_TotalSector1,CX                ;总扇区数→BDPB
4908:    MOV     ES:[DI].BDPB_MediaByte1,AH                  ;介质描述符→BDPB
4909:    MOV     BYTE PTR ES:[DI].BDPB_SectorsPerFAT1,AL     ;每个 FAT 表占用的扇区数→BDPB
4910:    MOV     BYTE PTR ES:[DI].BDPB_SectorsPerTrack1,BL   ;每道扇区数→BDPB
4911:    MOV     BYTE PTR ES:[DI].BDPB_NumberOfHead1,DL      ;磁头数→BDPB
4912:    MOV     WORD PTR ES:[DI].BDPB_HiddenSectors1+2,0    ;┐ 清 BDPB 的"隐含扇区数"字段值
4913:    MOV     WORD PTR ES:[DI].BDPB_HiddenSectors1,0      ;┘
4914:    MOV     WORD PTR ES:[DI].BDPB_BigTotalSector1+2,0   ;清 BDPB 的"32 位双字表示的总扇
                                                             ;区数"字段值
4915: Modify_BuildBPB9:
4916:    POP     BX
4917:    POP     DX
4918:    POP     CX
4919: Modify_BuildBPB_RET:
4920:    RET
4921: Modify_BuildBPB10:
4922:    MOV     ChangeFlag,00              ;清更换"卷系列号、卷标名和文件系统类型标志
                                            ;串"的控制标志
4923:    CALL    Set_CodeOfDriver           ;将 INT 13H 返回的错误码转换成设备驱动程序返
                                            ;回的错误码
4924:    JMP     SHORT Modify_BuildBPB9
4925: Modify_BuildBPB11:
4926:    MOV     ChangeFlag,00              ;清更换"卷系列号、卷标名和文件系统类型标志
                                            ;串"的控制标志
4927:    MOV     AL,7                       ;置"未知介质"的错误码
4928:    STC                                ;CF←1(错误标志)
4929:    JMP     SHORT Modify_BuildBPB9
```

```
Modify_BuildBPB  ENDP
;=============================================================
;相对地址偏移:06C3
;功能:读取盘上第一个扇区,并判定它是否是 DOS 操作系统的引导记录
;入口参数:无
;出口参数:CF=0,成功,BX=0 表示第一个扇区为引导记录、BX=1 表示第一个扇区不是引导记
          录;CF=1,失败
;=============================================================
Read_BootstrapSector2  PROC  NEAR
    MOV     DH,0                        ;┐
    MOV     CX,1                        ;│ 读取第一个扇区(即引导记录)
    CALL    Read_Disk                   ;┘
    JC      Read_BS_RET                 ;若读操作失败,则直接返回
    XOR     BX,BX
    CMP     BYTE PTR DiskBuffer,69H     ;┐
    JE      Read_BS1                    ;│
    CMP     BYTE PTR DiskBuffer,0E9H    ;│
    JE      Read_BS1                    ;│ 判定是否符合引导记录的格式,若不符合,则跳
    CMP     BYTE PTR DiskBuffer,0EBH    ;│ 转
    JNE     Read_BS3                    ;│
    CMP     BYTE PTR DiskBuffer+2,90H   ;│
    JNE     Read_BS3                    ;│
Read_BS1:                               ;┘
    MOV     AL,DiskBuffer+15H           ;AL←介质描述符
    AND     AL,0F0H                     ;┐
    CMP     AL,0F0H                     ;│ 若介质描述符格式不对,则跳转
    JNE     Read_BS3                    ;┘
    MOV     AL,DiskBuffer+15H           ;AL←介质描述符
    CMP     AL,0F0H                     ;┐ 若是未定义的介质类型,则跳转
    JZ      Read_BS4                    ;┘
    TEST    AL,1                        ;┐ 若是双面盘,则跳转
    JNZ     Read_BS4                    ;┘
    CMP     WORD PTR DiskBuffer+8,'  .3'  ;┐
    JNE     Read_BS2                    ;│
    CMP     BYTE PTR DiskBuffer+0AH,' 2'  ;│ 若是 DOS 3.20 以下版本,则跳转
    JAE     Read_BS4                    ;│
Read_BS2:                               ;┘
    MOV     BYTE PTR DiskBuffer+0DH,1   ;置每簇扇区数为 1
    JMP     SHORT Read_BS4
Read_BS3:
    INC     BX
Read_BS4:
    CLC
```

```
4972:     RET
4973: Read_BS_RET:
4974:     RET
4975: Read_BootstrapSector2  ENDP
4976: ;=========================================================
4977: ;相对地址偏移:0719
4978: ;功能:取出引导记录中 BPB 参数块的参数值
4979: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
4980: ;出口参数:AL=每个 FAT 表占用的扇区数;
4981: ;          AH=介质描述符;
4982: ;          BL=每道扇区数;
4983: ;          BH=最大根目录登记项数;
4984: ;          CX=总扇区数;
4985: ;          DL=磁头数;
4986: ;          DH=每簇扇区数
4987: ;=========================================================
4988: Get_BPB_Para  PROC  NEAR
4989:     MOV     DH,BYTE PTR DiskBuffer+0DH    ;DH←每簇扇区数
4990:     MOV     BH,BYTE PTR DiskBuffer+11H    ;BH←最大根目录登记项数
4991:     MOV     CX,WORD PTR DiskBuffer+13H    ;CX←总扇区数
4992:     MOV     AH,BYTE PTR DiskBuffer+15H    ;AH←介质描述符
4993:     MOV     AL,BYTE PTR DiskBuffer+16H    ;AL←每个 FAT 表占用的扇区数
4994:     MOV     BL,BYTE PTR DiskBuffer+18H    ;BL←每道扇区数
4995:     MOV     DL,BYTE PTR DiskBuffer+1AH    ;DL←磁头数
4996:     CMP     ChangeFlag,01                 ;┐若不更改"卷系列号、卷标名和文件系统类型标
4997:     JNZ     Get_BPBP2                     ;┘志串",则跳转
4998:     CALL    Set_BDPBExtMediaState         ;依据扩充 BPB 参数块设置 BDPB 中的"卷系列号、
                                                ;卷标名和文件系统类型标志串"
4999:     JB      Get_BPBP1                     ;若未设置(即 BPB 参数块不是扩充 BPB 参数块),
                                                ;则跳转
5000:     MOV     ChangeFlag,02                 ;设置已成功地更换标志
5001: Get_BPBP1:
5002:     CMP     ChangeLine,1                  ;┐若软盘驱动器不支持 Change Line,则跳转
5003:     JNZ     Get_BPBP2                     ;┘
5004:     CALL    Clear_ModifyBPBFlag           ;清"重建 BPB 参数块"的标志位
5005: Get_BPBP2:
5006:     CLC                                   ;DS=IO1 模块在内存中的段地址
5007:     RET
5008: Get_BPB_Para  ENDP
5009: ;=========================================================
5010: ;相对地址偏移:0751
5011: ;功能:依据扩充 BPB 参数块设置 BDPB 中的"卷系列号、卷标名和文件系统类型标志串"
5012: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
```

```
;出口参数:CF=0,已设置;CF=1,未设置(即 BPB 参数块不是扩充 BPB 参数块)
;=============================================================
Set_BDPBExtMediaState  PROC  NEAR
    CMP     BYTE PTR DiskBuffer+26H,29H   ;┐若引导记录中存放一般 BPB 参数块,则跳转
    JNZ     Set_BDPBExtMediaState_1       ;┘
    PUSH    CX
    MOV     CX,WORD PTR DiskBuffer+27H                ;┐
    MOV     WORD PTR ES:[DI].BDPB_SerialNumber,CX     ;├卷系列号→BDPB
    MOV     CX,WORD PTR DiskBuffer+29H                ;│
    MOV     WORD PTR ES:[DI].BDPB_SerialNumber+2,CX   ;┘
    PUSH    DI
    PUSH    SI
    MOV     CX,FileNameLen                ;┐
    MOV     SI,offset DiskBuffer+2BH      ;├卷标名→BDPB
    ADD     DI,4BH                        ;│
    REP     MOVSB                         ;┘
    MOV     CX,8                          ;┐
    MOV     SI,offset DiskBuffer+36H      ;├文件系统类型标志串→BDPB
    ADD     DI,5                          ;│
    REP     MOVSB                         ;┘
    POP     SI
    POP     DI
    POP     CX
    CLC
    RET
Set_BDPBExtMediaState_1:
    STC
    RET
Set_BDPBExtMediaState  ENDP
;=============================================================
;相对地址偏移:0788
;功能:从第 2 个扇区(FAT 表)中取出介质描述符
;入口参数:无
;出口参数:CF=0,成功,AH=介质描述符;CF=1,失败
;=============================================================
Read_SecondSector  PROC  NEAR
    MOV     DH,0                          ;┐
    MOV     CX,2                          ;├读取第 2 个扇区,
    CALL    Read_Disk                     ;┘
    JC      Read_SS_RET                   ;若读操作失败,则结束返回
    MOV     AH,[BX]                       ;AH←介质描述符
Read_SS_RET:
    RET
```

```
5056: Read_SecondSector  ENDP
5057: ;=========================================================
5058: ;相对地址偏移:0795
5059: ;功能:读取一个指定的扇区内容,并修改 BDPB 中的"时钟计数和圆柱面号"的参数值
5060: ;入口参数:CL=扇区号(其中高 2 位为圆柱面号的高 2 位);
5061: ;          CH=圆柱面号低八位值;
5062: ;          DH=磁头号;
5063: ;          DS=IO1 模块在常规内存低端的段地址值(70H);
5064: ;          ES:DI 指向指定逻辑驱动器对应的 BDPB
5065: ;出口参数:CF=0,成功;CF=1,失败
5066: ;=========================================================
5067: Read_Disk  PROC  NEAR
5068:     PUSH    BP
5069:     MOV     BP,3                          ;BP←出错重试次数
5070:     MOV     DL,ES:[DI].BDPB_DriveNumber   ;DL←物理驱动器号
5071:     MOV     BX, offset DiskBuffer         ;BX=传送缓冲区的地址偏移值(该传送缓冲区位
                                                ;于 IO1 模块中)
5072: Read_Disk1:
5073:     PUSH    ES                            ;┐
5074:     PUSH    DS                            ;│
5075:     POP     ES                            ;│ 读取一个指定的扇区
5076:     MOV     AX,201H                       ;│
5077:     INT     13H                           ;│
5078:     POP     ES                            ;┘
5079:     JNC     Read_Disk6                    ;若读操作成功,则跳转
5080: Read_Disk2:
5081:     CALL    ResetDiskControler            ;复位软盘和硬盘控制器
5082:     JZ      Read_Disk5                    ;若复位失败并且不允许出错重试,则跳转
5083:     TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,1  ;┐ 若介质不可装卸,则跳转
5084:     JNZ     Read_Disk1                           ;┘
5085:     CMP     ModifyFlag,00                        ;┐ 若已修改 INT 1EH 中断 ┐
5086:     JNZ     Read_Disk3                           ;┘ 向量,则跳转          │
5087:     PUSH    AX                                   ;                       │
5088:     PUSH    DS                                   ;                       │
5089:     LDS     SI,Old_INT1EH_Vector                 ;DS:SI 指向软盘参数表   │
5090:     MOV     AL,[SI].FPT_HeadLoadTime             ;AL←"寻道后磁头的稳定时 │
                                                       ;间"                    │
5091:     MOV     BYTE PTR [SI].FPT_HeadLoadTime, 0FH  ;修改"寻道后磁头的稳定时 │ 修改"寻道后
                                                       ;间"                    │ 磁头的稳定
5092:     POP     DS                                   ;                       │ 时间",并重
5093:     MOV     HeadLoadTime_BAK,AL                  ;保存软盘参数表的先前   │ 新读取指定
                                                       ;"寻道后磁头的稳定时间" │ 的扇区
```

```
                                        ;参数值                    |
5094:    POP     AX                     ;                          |
5095: Read_Disk3:                       ;                          |
5096:    PUSH    ES                     ;┐                         |
5097:    PUSH    DS                     ;|                         |
5098:    POP     ES                     ;| 读取一个指定的扇区      |
5099:    MOV     AX,201H                ;|                         |
5100:    INT     13H                    ;|                         |
5101:    POP     ES                     ;┘                         ┘
5102:    PUSHF
5103:    CMP     ModifyFlag,00          ;┐ 若已修改 INT 1EH 中断向量,则跳转
5104:    JNZ     Read_Disk4             ;┘
5105:    PUSH    AX                     ;┐
5106:    MOV     AL, HeadLoadTime_BAK   ;|
5107:    PUSH    DS                     ;|
5108:    LDS     SI, Old_INT1EH_Vector  ;| 恢复原来的"寻道后磁头的稳定时间"参数值
5109:    MOV     [SI]. FPT_HeadLoadTime, AL ;|
5110:    POP     DS                     ;|
5111:    POP     AX                     ;┘
5112: Read_Disk4:
5113:    POPF
5114:    JNB     Read_Disk6             ;若读操作成功,则跳转
5115:    JMP     ShortRead_Disk2        ;转去重试刚才的读操作
5116: Read_Disk5:
5117:    MOV     DL,-1                  ;设置盘操作出错的标志
5118:    STC
5119: Read_Disk6:
5120:    MOV     DiskNumber,DL          ;┐ 保存当前操作的物理驱动器号,若为-1 表明盘
5121:    MOV     Drive_Number,DL        ;┘ 操作出错
5122:    MOV     ES:[DI]. BDPB_CylinderOperated,CH   ;当前盘操作的圆柱面号的低八位值→BDPB
5123:    PUSHF                          ;┐
5124:    CALL    Update_ClockCount      ;| 设置 BDPB 中的当前时钟计数值
5125:    POPF                           ;┘
5126:    POP     BP
5127:    RET
5128: Read_Disk  ENDP
5129: ;=========================================================
5130: ;相对地址偏移:080A
5131: ;功能:"打开块设备"的处理程序(块设备的命令码=0DH)
5132: ;入口参数:AL=逻辑驱动器号
5133: ;出口参数:CF=0,成功
5134: ;=========================================================
5135: Disk_Dev_Open  PROC  NEAR
```

```
5136:     CMP     ChangeLine,0                    ;┐ 若软盘驱动器不支持 Change Line,则跳转
5137:     JZ      Disk_Dev_Open1                  ;┘
5138:     CALL    Get_BDPB_Node                   ;取出 AL 指定的逻辑驱动器对应的 BDPB 设备
5139:     INC     ES:[DI].BDPB_DeviceOpen         ;打开计数器加 1
5140: Disk_Dev_Open1:
5141:     CLC
5142:     RET
5143: Disk_Dev_Open   ENDP
5144: ;=============================================================
5145: ;相对地址偏移:081A
5146: ;功能:"关闭块设备"的处理程序(块设备的命令码=0EH)
5147: ;入口参数:AL=逻辑驱动器号
5148: ;出口参数:CF=0,成功
5149: ;=============================================================
5150: Disk_Dev_Close   PROC   NEAR
5151:     CMP     ChangeLine,0                    ;┐ 若软盘驱动器不支持 Change Line,则跳转
5152:     JZ      Disk_Dev_Close1                 ;┘
5153:     CALL    Get_BDPB_Node                   ;取出 AL 指定的逻辑驱动器对应的 BDPB
5154:     CMP     ES:[DI].BDPB_DeviceOpen,0       ;┐
5155:     JZ      Disk_Dev_Close1                 ;├ 设备打开计数器减 1
5156:     DEC     ES:[DI].BDPB_DeviceOpen         ;┘
5157: Disk_Dev_Close1:
5158:     CLC
5159:     RET
5160: Disk_Dev_Close   ENDP
5161: ;=============================================================
5162: ;相对地址偏移:0831
5163: ;功能:"可更换介质"的处理程序(块设备的命令码=0FH)
5164: ;入口参数:AL=逻辑驱动器号
5165: ;出口参数:CF=0,介质可装卸;CF=1,AH=设备驱动程序返回状态字的高字节值(忙位=1,表明
                介质不能装卸)
5166: ;=============================================================
5167: Disk_Rem_Media   PROC   NEAR
5168:     CALL    Get_BDPB_Node                   ;取出 AL 指定的逻辑驱动器对应的 BDPB
5169:     TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,1  ;┐ 根据介质是否允许装卸来设置设备驱动
5170:     JNZ     Disk_ND_RD                           ;┘ 程序返回的状态字值
5171:     CLC                                     ;CF←0(介质可装卸)
5172:     RET
5173: Disk_Rem_Media   ENDP
5174: ;=============================================================
5175: ;相对地址偏移:083D
5176: ;功能:"不等待的无破坏性读"块设备的处理程序(块设备的命令码为 05H)
5177: ;入口参数:AL=逻辑驱动器号
```

```
;出口参数:CF=1,AH=设备驱动程序返回的状态字的高字节值
;=============================================
Disk_ND_RD  PROC  NEAR
    MOV   AH,3                          ;置设备忙位、完成位
    STC
Disk_ND_RD_RET:
    RET
Disk_ND_RD  ENDP
;=============================================
;相对地址偏移:0841
;功能:磁盘"读/写"操作的处理程序(块设备的命令码=4/8/9)
;入口参数:AL=逻辑驱动器号;
;         AH=介质描述符;
;         CX=欲读/写的扇区数;
;         DX=起始扇区号;
;         ES:DI 指向传送缓冲区
;出口参数:无
;=============================================
Disk_WR_RD  PROC  NEAR
Disk_WR_Vertify  LABEL  NEAR            ;写同时验证(命令码=9)
    MOV   WORD PTR INT13H_CMDCode,0103H ;置写操作的命令码和校验标志
    JMP   Short Disk_WRD1
Disk_WR  LABEL  NEAR                    ;磁盘写操作(命令码=8)
    MOV   WORD PTR INT13H_CMDCode,3     ;置写操作的命令码,并清校验标志
Disk_WRD1:
    CALL  Disk_Operation                ;进行磁盘写操作
Disk_WRD2:
    JNB   Disk_ND_RD_RET                ;正常结束返回
    JMP   SetErrorDoneBit               ;转去设置返回状态字的"错误位和完成位"
Disk_RD  LABEL  NEAR                    ;磁盘读操作(命令码=4)
    CALL  Disk_Read                     ;进行磁盘读操作
    JMP   Short Disk_WRD2
Disk_WR_RD  ENDP
;=============================================
;相对地址偏移:085C
;功能:处理一个物理软盘驱动器具有多个逻辑驱动器名符的情况,它通过给出提示信息由用户确
      认的方法来确保对指定的介质进行操作。同时,当A、B两个逻辑驱动器名共用同一物理驱动
      器时,它还设置DOS通讯区的驱动器工作状态变量
;入口参数:ES:DI 指向指定逻辑驱动器对应的BDPB
;出口参数:CF=0,成功;CF=1,失败
;=============================================
MultiDriveName  PROC  NEAR
    PUSH  AX
```

```
5219:    PUSH    BX
5220:    MOV     BX,ES:[DI].BDPB_Attr_Status     ;┐
5221:    TEST    BL,21H                          ;│若物理驱动器使用当前逻辑驱动器名,则结束
5222:    JNZ     MultiDriveName5                 ;┘返回
5223:    TEST    BL,10H                          ;┐若物理驱动器只使用一个逻辑驱动器名,则结
5224:    JZ      MultiDriveName5                 ;┘束返回
5225:    MOV     AL,ES:[DI].BDPB_DriveNumber     ;AL←物理驱动器号
5226:    PUSH    ES
5227:    PUSH    DI
5228:    LES     DI, DWORD PTR BDPB_Head         ;ES:DI 指向第一个 BDPB
5229: MultiDriveName1:
5230:    CMP     ES:[DI].BDPB_DriveNumber,AL     ;┐若物理驱动器号不同,则跳转
5231:    JNE     MultiDriveName4                 ;┘
5232:    MOV     BL,20H
5233:    TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,BL  ;┐若当前 BDPB 处于备用状态(即物理驱
5234:    JZ      MultiDriveName4                       ;┘动器当前不对应于该 BDPB),则跳转
5235:    XOR     BYTE PTR ES:[DI].BDPB_Attr_Status,BL
5236:    POP     DI
5237:    POP     ES
5238:    OR      BYTE PTR ES:[DI].BDPB_Attr_Status,BL
5239:    CMP     EnableWR_Flag,1                 ;┐若不允许写 DOS 通讯区,则跳转
5240:    JNZ     MultiDriveName2                 ;┘
5241:    CMP     ES:[DI].BDPB_DriveNumber,0      ;┐若 BDPB 不对应于第一个物理驱动器,则跳转
5242:    JNZ     MultiDriveName5                 ;┘
5243:    MOV     AL,ES:[DI].BDPB_LogicalNumber   ;┐
5244:    PUSH    ES                              ;│第一个物理驱动器对应的逻辑驱动器号→DOS
5245:    MOV     ES,ZeroValue                    ;│通讯区
5246:    MOV     ES:Info_504H,AL                 ;│
5247:    POP     ES                              ;┘
5248:    JMP     Short MultiDriveName5
5249: MultiDriveName2:
5250:    CMP     SingleFDisk,2                   ;┐若 A、B 不共用同一物理驱动器,则跳转
5251:    JNZ     MultiDriveName3                 ;┘
5252:    PUSH    AX                              ;┐
5253:    MOV     AL,ES:[DI].BDPB_LogicalNumber   ;│
5254:    MOV     AH,AL                           ;│
5255:    PUSH    ES                              ;│若 A、B 共用同一物理驱动器,并且逻辑驱动器
5256:    MOV     ES,ZeroValue                    ;│名未被改变,则跳转
5257:    XCHG    AL,BYTE PTR ES:Info_504H        ;│
5258:    POP     ES                              ;│
5259:    CMP     AH,AL                           ;│
5260:    POP     AX                              ;│
5261:    JZ      MultiDriveName5                 ;┘
```

```
5262: MultiDriveName3:
5263:     CALL    Disp_Insert_FDisk                ;显示在指定的逻辑驱动器中插入盘片的提示信息
5264:     JMP     SHORT MultiDriveName5
5265: MultiDriveName4:
5266:     LES     DI,DWORD PTR ES:[DI].BDPB_NextPtr_OFS   ;ES:DI 指向下一个 BDPB
5267:     CMP     DI,-1                            ;┐若还有 BDPB 未检查,则跳转
5268:     JNZ     MultiDriveName1                  ;┘
5269:     STC
5270:     POP     DI
5271:     POP     ES
5272: MultiDriveName5:
5273:     POP     BX
5274:     POP     AX
5275:     RET
5276: MultiDriveName  ENDP
5277: ;相对地址偏移:08DD
5278: Temp1  PROC  NEAR
5279: Temp1_1:
5280:     MOV     AL,8                             ;置"扇区未找到"的错误码
5281:     JMP     SHORTTemp1_3
5282: Temp1_2:
5283:     MOV     AL,7                             ;置"未知介质"的错误码
5284: Temp1_3:
5285:     STC                                      ;CF←1(错误标志)
5286: Temp1_RET:
5287:     RET
5288: Temp1  ENDP
5289: ;相对地址偏移:08E5
5290: Disk_Read  PROC  NEAR
5291:     MOV     BYTE PTR INT13H_CMDCode,2        ;置读操作的命令码
5292: ;=================================================
5293: ;相对地址偏移:08EA
5294: ;功能:完成磁盘的读/写操作
5295: ;入口参数:AL=逻辑驱动器号;
5296: ;          CX=欲读取/写的扇区数;
5297: ;          DX=起始扇区号的低 16 位值;
5298: ;          ES:DI 指向传送缓冲区
5299: ;出口参数:CF=0,成功;CF=1,失败,AL=设备驱动程序返回的错误码
5300: ;=================================================
5301: Disk_Operation  LABEL  NEAR
5302:     MOV     BX,DI                            ;ES:BX 指向传送缓冲区
5303:     MOV     ESDSValue,ES                     ;保存"传送缓冲区的段地址值"
5304:     CALL    Get_BDPB_Node                    ;取出 AL 指定的逻辑驱动器对应的 BDPB
```

```
5305:       MOV     AL,ES:[DI].BDPB_MediaByte1      ;┐ 保存盘介质描述符
5306:       MOV     DiskMedia,AL                    ;┘
5307:       JCXZ    Temp1_RET                       ;若要读/写的扇区数为 0,则直接返回
5308:       TEST    BYTE PTR ES:[DI].BDPB_Attr_Status+1,2 ;┐ 若驱动器中的介质未格式化,此时
                                                    ;| 它不允许磁盘访问,那么转去置
5309:       JNZ     Temp1_2                         ;┘ "未知介质"的错误码
5310:       MOV     SectorsOfRemain,CX              ;设置未完成的扇区数
5311:       MOV     SPValue,SP                      ;保存 SP 寄存器值
5312:       MOV     AX,DX                           ;AX←起始扇区号的低 16 位值
5313:       XOR     SI,SI                           ;┐ SI:DX←终止的扇区号(当介质的总扇区数用
5314:       ADD     DX,CX                           ;| 16 位表示时)
5315:       ADC     SI,0                            ;┘
5316:       CMP     ES:[DI].BDPB_TotalSector1,0     ;┐ 若介质的总扇区用 32 位表示,则跳转
5317:       JZ      Disk_Op1                        ;┘
5318:       CMP     SI,0                            ;┐
5319:       JNZ     Temp1_1                         ;| 若要读/写的扇区超出介质允许的范围,则跳转
5320:       CMP     DX,ES:[DI].BDPB_TotalSector1    ;|
5321:       JA      Temp1_1                         ;┘
5322:       JMP     SHORT Disk_Op2
5323: Disk_Op1:
5324:       ADD     SI,StartingSector_HW            ;SI:DX←终止的扇区号(当介质的总扇区数用
                                                    ;32 位表示时)
5325:       CMP     SI,WORD PTR ES:[DI].BDPB_BigTotalSector1+2 ;┐
5326:       JB      Disk_Op2                        ;|
5327:       JA      Temp1_1                         ;| 若要读/写的扇区超出介质允
5328:       CMP     DX,WORD PTR ES:[DI].BDPB_BigTotalSector1   ;| 许的范围,则跳转
5329:       JA      Temp1_1                         ;┘
5330: Disk_Op2:
5331:       MOV     DX,StartingSector_HW                       ;┐
5332:       ADD     AX,WORD PTR ES:[DI].BDPB_HiddenSectors1    ;| 加上隐含扇区数,此时 DX:AX
5333:       ADC     DX,WORD PTR ES:[DI].BDPB_HiddenSectors1+2  ;┘ =绝对的起始扇区号
5334:       MOV     StartingSector_LW,AX            ;保存绝对的起始扇区号的低 16 位值
5335:       PUSH    ES                              ;┐
5336:       MOV     ES,ZeroValue                    ;|
5337:       LES     SI,DWORD PTR ES:Vector_1EH_OFS  ;| 保存 INT 1EH(软盘参数表)中断向量
5338:       MOV     WORD PTR Old_INT1EH_Vector,SI   ;|
5339:       MOV     WORD PTR Old_INT1EH_Vector+2,ES ;|
5340:       POP     ES                              ;┘
5341:       TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,1 ;┐ 若是读/写硬盘(介质不可更换),则跳
5342:       JNZ     Disk_Op4                        ;┘ 转
5343:       CALL    MultiDriveName                  ;处理一个物理驱动器具有多个逻辑驱动
                                                    ;器名符的情况,确保对指定的介质进行
                                                    ;读/写操作
```

```
5344:     CMP     ChangeLine,0                          ;┐ 若软盘驱动器不支持 Change Line,则
5345:     JZ      Disk_Op3                              ;┘ 跳转
5346:     CALL    Rebuild_BPB                           ;试图重建 BPB 参数块
5347: Disk_Op3:
5348:     CALL    Update_FPT_Para                       ;修改软盘参数表的参数
5349: Disk_Op4:
5350:     MOV     AX,DX                                 ;┐
5351:     XOR     DX,DX                                 ;│
5352:     DIV     WORD PTR ES:[DI].BDPB_SectorsPerTrack1;│ 计算磁道号的道内扇区号
5353:     MOV     Temp_HW,AX                            ;│
5354:     MOV     AX,StartingSector_LW                  ;│
5355:     DIV     WORD PTR ES:[DI].BDPB_SectorsPerTrack1;┘
5356:     INC     DL                                    ;┐ 保存道内扇区号
5357:     MOV     SectorNumber,DL                       ;┘
5358:     MOV     CX,ES:[DI].BDPB_NumberOfHead1         ;┐
5359:     PUSH    AX                                    ;│
5360:     XOR     DX,DX                                 ;│
5361:     MOV     AX,Temp_HW                            ;│ 计算圆柱面号和磁头号
5362:     DIV     CX                                    ;│
5363:     MOV     Temp_HW,AX                            ;│
5364:     POP     AX                                    ;│
5365:     DIV     CX                                    ;┘
5366:     CMP     Temp_HW,0                             ;┐
5367:     JA      Disk_Op5                              ;│ 若圆柱面号>1024,则跳转
5368:     CMP     AX,0400H                              ;│
5369:     JA      Disk_Op5                              ;┘
5370:     MOV     HeadNumber,DL                         ;保存磁头号
5371:     MOV     Cylinder,AX                           ;保存圆柱面号
5372:     MOV     AX,SectorsOfRemain                    ;AX←待读/写的扇区数
5373:     CALL    Disk_Operate1                         ;执行盘操作
5374:     CALL    Set_Recover_Para                      ;处理软盘操作的后续工作
5375:     RET
5376: Disk_Op5:
5377:     JMP     Temp1_1
5378: Disk_Read   ENDP
5379: ;=============================================================
5380: ;相对地址偏移:09BD
5381: ;功能:修改软盘参数表的参数:每道扇区数,马达启动时间、寻道后磁头的稳定时间
5382: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
5383: ;出口参数:无
5384: ;=============================================================
5385: Update_FPT_Para  PROC  NEAR
5386:     MOV     AL,ES:[DI].BDPB_DriveNumber           ;AL←物理驱动器号
```

```
5387:     MOV     Drive_Number,AL                    ;设置当前操作的物理驱动器号
5388:     CMP     ModifyFlag,00                      ;┐若已重新设置了 INT 1EH 中断向量
5389:     JNZ     Update_FPT_Para_RET                ;┘(即更换了软盘参数表),则直接返回
5390:     MOV     AL,MaxSectorsPerTrack              ;AL←每道最大扇区数
5391:     PUSH    DS
5392:     LDS     SI,Old_INT1EH_Vector               ;DS:SI 指向软盘参数表
5393:     MOV     [SI].FPT_SectorsPerTrack,AL        ;修改软盘参数表的"每道扇区数"
5394:     MOV     AL,[SI].FPT_StartupWaitTime        ;AL←软盘参数表的"马达启动时间"
5395:     MOV     AH,[SI].FPT_SectorsPerTrack        ;AH←每道扇区数
5396:     POP     DS
5397:     MOV     StartupWaitTime,AL                 ;保存原来的软盘参数表的"马达启动时
                                                     ;间"
5398:     MOV     SectorsPerTrack_BAK,AH             ;保存软盘参数表的"每道扇区数"
5399:     PUSH    DS
5400:     LDS     SI,Old_INT1EH_Vector               ;DS:SI 指向软盘参数表
5401:     CMP     BYTE PTR ES:[DI].BDPB_DeviceType,2 ;┐若软盘驱动器不是 720KB(3.5 英寸),
5402:     JNE     Update_FPT_Para1                   ;┘则跳转
5403:     MOV     AL,4                               ;┐修改软盘参数表的"马达启动时间"
5404:     XCHG    AL,[SI].FPT_StartupWaitTime        ;┘
5405: Update_FPT_Para1:
5406:     XOR     AL,AL                              ;┐
5407:     INC     AL                                 ;│修改软盘参数表的"寻道后磁头的稳
5408:     XCHG    AL,[SI].FPT_HeadLoadTime           ;┘定时间"
5409:     POP     DS
5410:     MOV     HeadLoadTime,AL                    ;保存原来的软盘参数表的"寻道后磁头
                                                     ;的稳定时间"
5411:     MOV     AL,0FH                             ;┐设置新的"寻道后磁头的稳定时间"
5412:     MOV     BYTE PTR NewHeadLoadTime,AL        ;┘
5413: Update_FPT_Para_RET:
5414:     RET
5415: Update_FPT_Para  ENDP
5416: ;=============================================================
5417: ;相对地址偏移:0A06
5418: ;功能:处理软盘操作的后续工作(设置当前操作的时钟计数值、恢复软盘参数表的部分被修改的参
          数值)
5419: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
5420: ;出口参数:无
5421: ;=============================================================
5422: Set_Recover_Para  PROC  NEAR
5423:     TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,1 ;┐若介质不可移动(即硬盘),则直接返
5424:     JNZ     Set_RP_RET                          ;┘回
5425:     CALL    Update_ClockCount                   ;当前操作的时钟计数值→BDPB
5426: Set_Recover_Para1:
```

```
5427:    PUSHF
5428:    CMP     ModifyFlag,00                    ;┐ 若先前已更换软盘参数表,则结束返
5429:    JNZ     Set_Recover_Para2                ;┘ 回
5430:    PUSH    AX
5431:    PUSH    ES
5432:    LES     SI,Old_INT1EH_Vector             ;ES:SI 指向软盘参数表
5433:    MOV     AL,SectorsPerTrack_BAK           ;┐
5434:    MOV     ES:[SI].FPT_SectorsPerTrack,AL   ;│ 恢复软盘参数表中的"每道扇区数、寻
5435:    MOV     AL,HeadLoadTime                  ;│ 道后磁头的稳定时间、每个扇区字节
5436:    MOV     AH,StartupWaitTime               ;│ 数、马达启动时间"等参数值
5437:    MOV     ES:[SI].FPT_HeadLoadTime,AL      ;│
5438:    MOV     BYTE PTR ES:[SI].FPT_SectorSize,2 ;│
5439:    MOV     ES:[SI].FPT_StartupWaitTime,AH   ;│
5440:    POP     ES                               ;┘
5441:    POP     AX
5442: Set_Recover_Para2:
5443:    POPF
5444: Set_RP_RET:
5445:    RET
5446: Set_Recover_Para  ENDP
5447: ;=============================================================
5448: ;相对地址偏移:0A3D
5449: ;功能:依据具体情况完成盘操作
5450: ;入口参数:AX=欲读/写的扇区数;
5451: ;          BX=传送缓冲区的地址偏移值;
5452: ;          ES:DI 指向指定逻辑驱动器对应的 BDPB
5453: ;出口参数:无
5454: ;=============================================================
5455: Disk_Operate1  PROC  NEAR
5456:    OR      AX,AX                            ;┐ 若要读/写的扇区数为 0,则直接返回
5457:    JZ      Set_RP_RET                       ;┘
5458:    TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,1 ;┐ 若介质是可移动的,则跳转
5459:    JZ      Disk_Op1_1                       ;┘
5460:    TEST    BYTE PTR MultiTrack,80H          ;┐ 若"multitrack=OFF",则跳转
5461:    JZ      Disk_Op1_1                       ;┘
5462:    CALL    Disk_Operate2                    ;执行具体的读/写操作
5463:    XOR     AX,AX
5464:    RET
5465: Disk_Op1_1:
5466:    MOV     CL,BYTE PTR ES:[DI].BDPB_SectorsPerTrack1 ;┐
5467:    INC     CL                               ;│
5468:    SUB     CL,SectorNumber                  ;│
5469:    XOR     CH,CH                            ;┘ 计算当前允许读/写的扇区数
```

```
5470:    CMP     AX,CX                                  ;| →CX
5471:    JAE     Disk_Op1_2                             ;|
5472:    MOV     CX,AX                                  ;┘
5473: Disk_Op1_2:
5474:    PUSH    AX                                     ;AX=未完成读/写的扇区数
5475:    PUSH    CX                                     ;当前准备读/写的扇区数
5476:    MOV     AX,CX                                  ;┐ 读/写 AX 指定的扇区数
5477:    CALL    Disk_Operate2                          ;┘
5478:    POP     CX
5479:    POP     AX
5480:    SUB     AX,CX                                  ;修改未完成读/写的扇区数
5481:    SHL     CL,1                                   ;┐ 修改传送缓冲区的地址偏移值
5482:    ADD     BH,CL                                  ;┘
5483:    JMP     SHORT Disk_Operate1                    ;转去继续读/写未完成的扇区
5484: Disk_Operate1  ENDP
5485: ;
5486: Disk_Op2_1:
5487:    JMP     Disk_Op2_16
5488: ;=============================================================
5489: ;相对地址偏移:0A7B
5490: ;功能:完成读、写或验证等磁盘操作
5491: ;入口参数:AX=欲操作的扇区数
5492: ;          BX=传送缓冲区的地址偏移值
5493: ;出口参数:CX=未完成的扇区数
5494: ;=============================================================
5495: Disk_Operate2  PROC  NEAR
5496:    MOV     BP,5                                   ;┐
5497:    TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,1    ;|
5498:    JZ      Disk_Op2_2                             ;| 设置出错重试次数(BP=出错重试次
5499:    CMP     AH,4                                   ;| 数)
5500:    JZ      Disk_Op2_2                             ;|
5501:    MOV     BP,2                                   ;|
5502: Disk_Op2_2:                                       ;┘
5503:    MOV     RetriesForUnECC,BP                     ;设置"非 ECC 纠正数据错误"的重试次数
5504:    MOV     RetriesForECC,BP                       ;设置"ECC 纠正数据错误"的重试次数
5505:    MOV     AH,BYTE PTR INT13H_CMDCode             ;AH←INT 13H 的命令码
5506: Disk_Op2_3:
5507:    PUSH    AX
5508:    MOV     DX,Cylinder                            ;DX←圆柱面号
5509:    TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,1    ;┐ 若介质可移动,则跳转
5510:    JZ      Disk_Op2_4                             ;┘
5511:    CMP     WORD PTR ES:[DI].BDPB_ClocklorCylinderFlag,1 ;┐
5512:    JNE     Disk_Op2_4                             ;| 校正硬盘的圆柱面号
```

```
5513:     ADD     DX,ES:[DI].BDPB_Clock2orCylinder          ;|
5514: Disk_Op2_4:                                           ;┘
5515:     ROR     DH,1                                      ;┐
5516:     ROR     DH,1                                      ;|
5517:     OR      DH,SectorNumber                           ;| 将圆柱面号与道内扇区号拼成 INT
5518:     MOV     CX,DX                                     ;| 13H 的入口参数 CX 寄存器值的格式
5519:     XCHG    CH,CL                                     ;┘
5520:     MOV     DH,HeadNumber                             ;磁头号→DH
5521:     MOV     DL,ES:[DI].BDPB_DriveNumber               ;物理驱动器号→DL
5522:     CMP     ES:[DI].BDPB_DeviceType,5                 ;┐ 若是硬盘(介质不可移动),则跳转
5523:     JE      Disk_Op2_6                                ;┘
5524:     CMP     BYTE PTR DiskNumber,-1                    ;┐ 若盘片已更换,则跳转
5525:     JE      Disk_Op2_5                                ;┘
5526:     CMP     AH,2                                      ;┐ 若是读盘操作,则跳转
5527:     JE      Disk_Op2_6                                ;┘
5528:     CMP     AH,4                                      ;┐ 若是验证盘扇区,则跳转
5529:     JE      Disk_Op2_6                                ;┘
5530: Disk_Op2_5:
5531:     JMP     SHORT Disk_Op2_12
5532: Disk_Op2_6:
5533:     CALL    CALL  INT13H                              ;请求 INT 13H 中断服务以便完成指定的
                                                            ;工作
5534: Disk_Op2_7:
5535:     JB      Disk_Op2_1                                ;若盘操作失败,则跳转
5536:     MOV     DiskNumber, DL                            ;存放当前操作的物理驱动器号
5537:     MOV     ES:[DI].BDPB_CylinderOperated,CH          ;当前操作的圆柱面号的低八位值→BDPB
5538:     CMP     WORD PTR INT13H_CMDCode,0103H             ;┐ 若是写且验证,则跳转
5539:     JZ      Disk_Op2_14                               ;┘
5540: Disk_Op2_8:
5541:     POP     AX                                        ;AL=当前完成的扇区数
5542:     TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,1       ;┐ 若介质可移动,则跳转
5543:     JZ      Disk_Op2_9                                ;┘
5544:     TEST    BYTE PTR MultiTrack,80H                   ;┐ 若"multitrack=ON",则跳转
5545:     JNZ     Disk_Op2_11                               ;┘
5546: Disk_Op2_9:
5547:     AND     CL,3FH                                    ;CL←扇区号
5548:     XOR     AH,AH                                     ;┐ 计算未完成的扇区数
5549:     SUB     SectorsOfRemain,AX                        ;┘
5550:     ADD     CL,AL                                     ;┐ 计算下次操作的道内扇区号
5551:     MOV     SectorNumber,CL                           ;┘
5552:     CMP     CL,BYTE PTR ES:[DI].BDPB_SectorsPerTrack1;┐ 若不需要修改磁头号、道内扇区号、
5553:     JBE     Disk_Op2_11                               ;┘ 圆柱面号,则跳转
5554:     MOV     BYTE PTR SectorNumber,1                   ;┐
```

```
5555:    MOV     DH,HeadNumber                          ;|
5556:    INC     DH                                     ;|
5557:    CMP     DH,BYTE PTR ES:[DI].BDPB_NumberOfHead1;|
5558:    JB      Disk_Op2_10                            ;| 修改道内扇区号、磁头号、圆柱面号
5559:    XOR     DH,DH                                  ;|
5560:    INC     WORD PTR Cylinder                      ;|
5561: Disk_Op2_10:                                      ;|
5562:    MOV     HeadNumber,DH                          ;┘
5563: Disk_Op2_11:
5564:    CLC
5565:    RET
5566: Disk_Op2_12:
5567:    CMP     DL,BYTE PTR DiskNumber                 ;┐ 若驱动器工作不正常,则跳转
5568:    JNE     Disk_Op2_13                            ;┘
5569:    CMP     CH,ES:[DI].BDPB_CylinderOperated       ;┐ 若圆柱面号的低八位相同,则跳转
5570:    JE      Disk_Op2_6                             ;┘
5571: Disk_Op2_13:
5572:    CALL    Retry_INT13H                           ;变换软盘参数表的参数,重新请求 INT 13H 中
                                                        ;断服务
5573:    JMP     SHORT Disk_Op2_7
5574: Disk_Op2_14:
5575:    POP     AX                                     ;┐ AH=命令码,AL=当前操作的扇区数
5576:    PUSH    AX                                     ;┘
5577:    MOV     AH, 4                                  ;┐ 检证先前写的扇区
5578:    CALL    CALL_INT13H                            ;┘
5579:    JNB     Disk_Op2_8                             ;若检证正确,则跳转
5580:    CMP     AH,11H                                 ;┐ 若不是"ECC 纠正数据错误",则跳转
5581:    JNZ     Disk_Op2_15                            ;┘
5582:    DEC     WORD PTR RetriesForECC                 ;┐ 若规定的出错重试次数已用完,则跳转
5583:    JZ      Disk_Op2_8                             ;┘
5584:    CALL    Reset_Drive                            ;复位软盘和硬盘控制器
5585:    JMP     SHORT Disk_Op2_21
5586: Disk_Op2_15:
5587:    CALL    Reset_Drive                            ;复位软盘和硬盘控制器
5588:    DEC     WORD PTR RetriesForUnECC               ;允许"非 ECC 纠正数据错误"的出错重试次数减 1
5589:    JMP     SHORT Disk_Op2_19
5590: Disk_Op2_16:
5591:    CMP     ChangeLine,0                           ;┐ 若软盘驱动器不支持 Chagne Line,则跳转
5592:    JZ      Disk_Op2_17                            ;┘
5593:    CALL    Process_RemMedia                       ;若驱动器上的介质已被更换,则进行相应的处理
5594: Disk_Op2_17:
5595:    CMP     MultiTrack_OpFlag,01                   ;┐ 若当前的盘操作只在一个磁道中,则跳转
5596:    JNZ     Disk_Op2_18                            ;┘
```

```
5597:    MOV     BP,1                          ;BP←出错重试次数
5598:    MOV     MultiTrack_OpFlag,00          ;清"多磁道操作"的标志
5599: Disk_Op2_18:
5600:    CALL    ResetDiskControler            ;复位软盘和硬盘控制器,并检查是否允许出错重
                                               ;试
5601: Disk_Op2_19:
5602:    JZ      Disk_Op2_23                   ;若不允许出错重试,则跳转
5603:    TEST    BYTE PTR ES:[DI].BDPB_Attr_Status,1;┐若介质不可移动,则跳转
5604:    JNZ     Disk_Op2_20                   ;┘
5605:    CMP     AH,80H                        ;┐若是"驱动器未就绪"错,则跳转
5606:    JZ      Disk_Op2_23                   ;┘
5607: Disk_Op2_20:
5608:    CMP     AH,0CCH                       ;┐若写硬盘失败,则跳转
5609:    JZ      Disk_Op2_22                   ;┘
5610:    MOV     RetriesForECC,5               ;重新设置"ECC 纠正数据错误"的出错重试次数
5611: Disk_Op2_21:
5612:    POP     AX                            ;AL=欲操作的扇区数
5613:    JMP     Disk_Op2_3                    ;转去重新执行刚才的磁盘操作
5614: Disk_Op2_22:
5615:    MOV     BP,1                          ;BP←允许的出错重试次数
5616:    JMP     SHORT Disk_Op2_21
5617: Disk_Op2_23:
5618:    CALL    Set_CodeOfDriver              ;依据 INT 13H 返回的错误码设置设备驱动程序返
                                               ;回的错误码
5619: Disk_Op2_24:
5620:    MOV     Drive_Number,-1               ;置操作失败的标志
5621:    MOV     CX,SectorsOfRemain            ;CX=未完成的扇区数
5622:    MOV     SP,SPValue                    ;恢复堆栈指针值
5623:    JMP     Set_Recover_Para1             ;转去恢复软盘参数表的部分被修改的参数值
5624: Disk_Operate2  ENDP
5625: ;=============================================================
5626: ;相对地址偏移:0BB8
5627: ;功能:变换软盘参数表中的"寻道后磁头的稳定时间",并且重新请求 INT 13H 磁盘操作服务
5628: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
5629: ;出口参数:CF=0,成功;CF=1,失败,AH=错误码(INT 13H 返回的错误码)
5630: ;=============================================================
5631: Retry_INT13H  PROC  NEAR
5632:    CMP     ModifyFlag,00                 ;┐若先前已更换软盘参数表,则跳转
5633:    JNZ     CALL_INT13H                   ;┘
5634:    PUSH    ES                            ;┐
5635:    PUSH    AX                            ;|
5636:    MOV     AL,NewHeadLoadTime            ;|
5637:    LES     SI,Old_INT1EH_Vector          ;|
```

```
5638:    MOV     ES:[SI].FPT_HeadLoadTime,AL    ;| 在改变"寻道后磁头的稳定时间"之后发出 INT
5639:    POP     AX                             ;| 13H 中断请求
5640:    POP     ES                             ;|
5641:    CALL    CALL_INT13H                    ;|
5642:    PUSH    ES                             ;|
5643:    LES     SI,Old_INT1EH_Vector           ;|
5644:    MOV     ES:[SI].FPT_HeadLoadTime,1     ;|
5645:    POP     ES                             ;┘
5646:    RET
5647: Retry_INT13H  ENDP
5648: ;=============================================================
5649: ;相对地址偏移:0BDD
5650: ;功能:执行 INT 13H 的中断请求
5651: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB,其它参数参见具体的功能
5652: ;出口参数:CF=0,成功;CF=1,失败,AH=错误码(INT 13H 返回的错误码)
5653: ;=============================================================
5654: CALL_INT13H  PROC  NEAR
5655:    TEST    ES:[DI].BDPB_SizeFlag,80H      ;┐ 若介质超出了 DOS 的管理范围,则跳转
5656:    JNZ     CALL_INT13H1                   ;┘
5657:    PUSH    ES                             ;┐
5658:    MOV     ES,ESDSValue                   ;|
5659:    INT     13H                            ;| 请求 INT 13H 中断服务
5660:    MOV     ESDSValue,ES                   ;|
5661:    POP     ES                             ;┘
5662:    RET
5663: CALL_INT13H1:
5664:    STC                                    ;CF←1(失败)
5665:    MOV     AH,80H                         ;AH←超时的错误码
5666:    RET
5667: CALL_INT13H  ENDP
5668: ;=============================================================
5669: ;相对地址偏移:0BF5
5670: ;功能:依据 INT 13H 返回的错误码,经查表得到设备驱动程序返回的错误码
5671: ;入口参数:AH=INT 13H 返回的状态字节值(即错误码)
5672: ;出口参数:AL=设备驱动程序返回的错误码
5673: ;=============================================================
5674: Set_CodeOfDriver  PROC  NEAR
5675:    PUSH    CX
5676:    PUSH    ES
5677:    PUSH    DS
5678:    POP     ES
5679:    MOV     AL,AH                          ;AL←INT 13H 中断返回的状态字节值
5680:    MOV     INT13H_TS,AL                   ;保存 INT 13H 返回的状态字节值,它作为
```

```
                                        ;INT 13H_RetCode 表的结束标志
5681:   MOV     CX,9                    ;表的长度
5682:   MOV     DI, offset INT13H_RetCode   ;┐查表得出表的序号
5683:   REPNE   SCASB                   ;┘
5684:   DB      8AH,85H                 ;┐
5685:   DW      8                       ;│ AL←对应的设备驱动程序返回的错误码
5686:   ;MOV    AL,[DI+8]               ;┘
5687:   POP     ES
5688:   POP     CX
5689:   STC
5690:   RET
5691: Set_CodeOfDriver  ENDP
5692: ;=============================================================
5693: ;相对地址偏移:0C0E
5694: ;功能:设置指定 BDPB 的当前操作的时钟计数值
5695: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
5696: ;出口参数:无
5697: ;=============================================================
5698: Update_ClockCount  PROC  NEAR
5699:   PUSH    AX
5700:   CALL    ReadClockCount              ;读取时钟计数值→CX:DX
5701:   CMP     DX,ES:[DI].BDPB_Clock1orCylinderFlag ;┐
5702:   JNZ     Update_CC1                  ;│
5703:   CMP     CX,ES:[DI].BDPB_Clock2orCylinder     ;│ 判定是否要更换 BDPB 中的时钟计数值,
5704:   JE      Update_CC2                  ;│ 若不更换,则结束返回
5705: Update_CC1:                          ;┘
5706:   MOV     Operate_Times,0             ;清时钟计数未改变的计数器值
5707:   MOV     ES:[DI].BDPB_Clock1orCylinderFlag,DX ;┐ 时钟计数值→BDPB
5708:   MOV     ES:[DI].BDPB_Clock2orCylinder,CX     ;┘
5709: Update_CC2:
5710:   CLC
5711:   POP     AX
5712:   RET
5713: Update_ClockCount  ENDP
5714: ;=============================================================
5715: ;相对地址偏移:0C2E
5716: ;功能:复位软盘和硬盘控制器,并检查是否允许出错重试
5717: ;入口参数:BP=出错重试次数
5718: ;出口参数:ZF=0,复位成功或允许出错重试;ZF=1,复位失败并且不允许出错重试
5719: ;=============================================================
5720: ResetDiskControler  PROC  NEAR
5721:   CALL    Reset_Drive                 ;复位软盘和硬盘控制器
5722:   CMP     AH,6                        ;┐若软盘驱动器中没有盘片,则跳转
```

```
5723:    JZ      ResetDiskController_1          ;┘
5724:    DEC     BP                             ;出错重试次数减 1
5725:    RET
5726: ResetDiskControler_1:
5727:    OR      AH,AH                          ;设置返回标志值(ZF=0,复位成功)
5728:    RET
5729: ResetDiskControler  ENDP
5730:    DB      0
5731: ;相对地址偏移:0C3C
5732: Get_GenericIOCTL_CMDNum DB  8             ;块设备的类属 IOCTL 请求的类中子读取功能数
                                                ;-1
5733:    DW      Get_DevicePara                 ;取设备参数(类中子功能码为 60H)
5734:    DW      Read_DeviceTrack               ;读逻辑设备磁道(类中子功能码为 61H)
5735:    DW      VerifyTrack                    ;验证逻辑设备磁道(类中子功能为 62H)
5736:    DW      InvalidSubCmd                  ;保留待用的类中子功能码 63H
5737:    DW      InvalidSubCmd                  ;保留待用的类中子功能码 64H
5738:    DW      InvalidSubCmd                  ;保留待用的类中子功能码 65H
5739:    DW      Get_ExtMediaState              ;取磁盘的"卷系列号、卷标名和文件系统类型标志
                                                ;串"(类中子功能码为 67H)
5740:    DW      Get_AccessState                ;取访问标志(类中子功能码为 67H)
5741:    DW      Get_68H                        ;测试 INT 13H 中断服务程序是否支持功能 20H
                                                ;(类中子功能能码为 68H)
5742: ;相对地址偏移:0C4F
5743: Set_GenericIOCTL_CMDNum  DB  7            ;块设备的类属 IOCTL 请求的类中子设置功能数
                                                ;-1
5744:    DW      Set_DevicePara                 ;设置设备参数(类中子功能码为 40H)
5745:    DW      Write_DeviceTrack              ;写逻辑设备磁道(类中子功能码为 41H)
5746:    DW      FormatTrack                    ;格式化并验证逻辑设备磁道(类中子功能码为
                                                ;42H)
5747:    DW      InvalidSubCmd                  ;保留待用的类中子功能码 43H
5748:    DW      InvalidSubCmd                  ;保留待用的类中子功能码 44H
5749:    DW      InvalidSubCmd                  ;保留待用的类中子功能码 45H
5750:    DW      Set_ExtMediaState              ;设置磁盘的"卷系列号、卷标名和文件系统类型标
                                                ;志串"(类中子功能码为 46H)
5751:    DW      Set_AccessState                ;设置访问标志(类中子功能码为 47H)
5752: ;相对地址偏移:0C60
5753: ;块设备驱动程序支持的类属 IOCTL 请求的类中子功能码表
5754: GenericIOCTLFuncCode  DB  60H,40H
5755:    DB      61H,41H
5756:    DB      62H,42H
5757:    DB      66H,46H
5758:    DB      67H,47H
5759:    DB      68H
```

```
5760: ;=========================================================
5761: ;相对地址偏移:0C6B
5762: ;功能:"类属 IOCTL 请求"的处理程序(块设备的命令码=13H)
5763: ;入口参数:AL=逻辑驱动器号
5764: ;出口参数:CF=0,成功;CF=1,失败,AX=设备驱动程序返回的状态字值
5765: ;=========================================================
5766: Disk_Generic_IOCTL  PROC  NEAR
5767:     CALL    Get_BDPB_Node                   ;取出 AL 指定的逻辑驱动器对应的 BDPB
5768:     PUSH    ES
5769: LESBX, DRH_Ptr                             ;ES:BX 指向设备驱动程序请求标题
5770:     CMP     BYTE PTR ES:[BX+0DH], 08        ;┐
5771:     MOV     AL, BYTE PTR ES:[BX+0EH]        ;├若不是块设备类型(即设备类型不对),则跳转
5772:     POP     ES                              ;│
5773:     JNZ     Disk_Gen2                       ;┘
5774: ;AL=类中子功能码
5775:     MOV     SI, offset Get_GenericIOCTL_CMDNum  ;SI 指向读取操作的命令分转表
5776:     TEST    AL,20H                          ;┐若是取块设备信息的操作,则跳转
5777:     JNZ     Disk_Gen1                       ;┘
5778:     MOV     SI, offset Set_GenericIOCTL_CMDNum  ;SI 指向设置操作的命令分转表
5779: Disk_Gen1:
5780:     AND     AL, 0DFH                        ;┐
5781:     SUB     AL, 40H                         ;├若类中子功能码是未定义的,则跳转
5782:     CMP     AL,CS:[SI]                      ;│
5783:     JA      Disk_Gen2                       ;┘
5784:     CBW                                     ;┐
5785:     SHL     AX, 1                           ;│
5786:     INC     SI                              ;├调用类中子功能码对应的处理程序
5787:     ADD     SI, AX                          ;│
5788:     CALL    CS:[SI]                         ;┘
5789:     MOV     DS,CS:IO1_SEG                   ;DS←IO1 模块在常规内存中的段地址值
5790:     MOV     AH,81H                          ;AH←设备驱动程序返回状态字的高字节值
5791:     RET
5792: InvalidSubCmd  LABEL  NEAR
5793:     POP     DX
5794: Disk_Gen2:
5795:     JMP     InvalidCmd                      ;转去设置设备驱动程序返回的状态字值
5796: Disk_Generic_IOCTL  ENDP
5797: ;=========================================================
5798: ;相对地址偏移:0CA7
5799: ;功能:取块设备参数(类中子功能码=60H)
5800: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
5801: ;出口参数:CF=0,类属 IOCTL 请求的参数块中的数据有效;CF=1,类属 IOCTL 请求的参数块中的
          数据无效
```

```
5802: ;=========================================================
5803: Get_DevicePara  PROC  NEAR
5804:     LDS     BX, DRH_Ptr                    ;DS:BX 指向设备驱动程序请求标题
5805:     LDS     BX,[BX+13H]                    ;DS:BX 指向存放返回信息的类属 IOCTL 请求的
                                                 ;参数块
5806:     MOV     AL, ES:[DI].BDPB_DeviceType    ;┐"设备类型"→类属 IOCTL 请求的参数块
5807:     MOV     [BX+1], AL                     ;┘
5808:     MOV     AX, ES:[DI].BDPB_Attr_Status   ;┐
5809:     AND     AX,3                           ;├"设备属性"→类属 IOCTL 请求的参数块
5810:     MOV     [BX+2], AX                     ;┘
5811:     MOV     AX, ES:[DI].BDPB_NumberOfCylinder ;┐ 块设备支持的柱面数→类属 IOCTL 请求的
5812:     MOV     [BX+4], AX                     ;┘ 参数块
5813:     XOR     AL, AL                         ;┐ 设置类属 IOCTL 请求的参数块中的"介质类型"
5814:     MOV     [BX+6], AL                     ;┘ 为缺省值
5815:     LEA     SI, [DI].BDPB_BytesPerSector2  ;ES:SI 指向 Default BPB 参数块
5816:     TEST    BYTE PTR [BX], 1               ;┐ 若是取设备的 Default BPB 参数块,则跳转
5817:     JZ      Get_DevP1                      ;┘
5818:     PUSH    DS
5819:     MOV     DS,CS:IO1_SEG                  ;DS←IO1 模块在常规内存低端的段地址(70H)
5820:     CALL    MultiDriveName                 ;处理一个物理软盘驱动器具有多个逻辑驱动器名
                                                 ;的情况,确保对指定的介质进行操作
5821:     CALL    Modify_BuildBPB                ;建立 Build BPB 参数块
5822:     POP     DS
5823:     JB      Get_DevP_RET                   ;若建立 Build BPB 参数块失败,则跳转
5824:     LEA     SI, [DI].BDPB_BytesPerSector1  ;ES:SI 指向重建的 Build BPB 参数块
5825: Get_DevP1:
5826:     LEA     DI, [BX+07]                    ;┐
5827:     MOV     CX, BPB_Len1                   ;│
5828:     PUSH    DS                             ;│ 块设备的 BPB 参数块→类属 IOCTL 请求的参数
5829:     PUSH    ES                             ;│ 块
5830:     POP     DS                             ;│
5831:     POP     ES                             ;│
5832:     REP     MOVSB                          ;┘
5833:     CLC
5834: Get_DevP_RET:
5835:     RET
5836: Get_DevicePara  ENDP
5837: ;=========================================================
5838: ;相对地址偏移:0CF3
5839: ;功能:设置块设备的参数(类中子功能码=40H)
5840: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
5841: ;出口参数:CF=0,成功;CF=1,失败,AL=错误码(0CH)
5842: ;=========================================================
```

```
5843: Set_DevicePara  PROC  NEAR
5844:     LDS     BX,DRH_Ptr                    ;DS:BX 指向设备驱动程序请求标题
5845:     LDS     BX,[BX+13H]                   ;DS:BX 指向类属 IOCTL 请求的参数块
5846:     OR      WORD PTR ES:[DI].BDPB_Attr_Status,0140H;设置块设备参数变化的标志
5847:     TEST    BYTE PTR [BX], 02             ;┐若除 Track Layout 字段外忽略类属 IOCTL 请求
                                                ;│的参数块中的其它所有字段,则转去处理磁道
5848:     JNZ     Set_DevP5                     ;┘布局(Track Layout)字段
5849:     MOV     AL, [BX+1]                    ;┐"设备类型"→BDPB
5850:     MOV     ES:[DI].BDPB_DeviceType,AL    ;┘
5851:     MOV     AX, [BX+4]                        ;┐圆柱面数→BDPB
5852:     MOV     ES:[DI].BDPB_NumberOfCylinder,AX  ;┘
5853:     MOV     AX, [BX+2]                    ;AX←设备属性
5854:     PUSH    DS                            ;┐
5855:     MOV     DS,CS:IO1_SEG                 ;│
5856:     CMP     ChangeLine,0                  ;│若不允许取消 Change Line,则跳转
5857:     POP     DS                            ;│
5858:     JNZ     Set_Devp1                     ;┘
5859:     AND     AX,0FFFDH                     ;设置软盘驱动器不支持 Change Line 的标志位
5860: Set_Devp1:
5861:     AND     AX,3                          ;┐
5862:     MOV     CX,ES:[DI].BDPB_Attr_Status   ;│设置 BDPB 的"设备属性"字段值(其中,"介质装
5863:     AND     CX,0FDF4H                     ;│卸位、Change Line 功能位、磁道中扇区特性位、
5864:     OR      AX,CX                         ;│访问标志位"允许被改变)
5865:     MOV     ES:[DI].BDPB_Attr_Status,AX   ;┘
5866:     MOV     AL,[BX+6]                     ;┐
5867:     PUSH    DS                            ;│取出并保存当驱动器中实际介质不能被确定时
5868:     MOV     DS,CS:IO1_SEG                 ;│使用的介质类型
5869:     MOV     MediaTypeFlag,AL              ;│
5870:     POP     DS                            ;┘
5871:     OR      WORD PTR ES:[DI].BDPB_Attr_Status,0080H  ;置"设备参数"修改标志位
5872:     PUSH    DI
5873:     TEST    BYTE PTR [BX],1                          ;┐若"所有以后的 Build BPB 请求均
                                                           ;│返回类属 IOCTL 请求的参数块中
5874:     JNZ     Set_DevP3                                ;┘的 Device BPB",则跳转
5875:     TEST    WORD PTR ES:[DI].BDPB_Attr_Status,4      ;┐
5876:     JZ      Set_Devp2                                ;│设置 Default BPB 参数块被更换的
5877:     AND     WORD PTR ES:[DI].BDPB_Attr_Status,-05    ;│标志位值
5878: Set_Devp2:                                           ;┘
5879:     MOV     CX, BPB_Len2                             ;┐准备更换"Default BPB 参数块"
5880:     LEA     DI,[DI].BDPB_BytesPerSector2             ;┘
5881:     JMP     SHORT Set_DevP4                          ;转去更换 Default BPB 参数块
5882: Set_DevP3:
5883:     OR      WORD PTR ES:[DI].BDPB_Attr_Status,+4     ;置"Build BPB 参数块"被更换的标志
```

```
                                                    ;位
5884:   MOV     CX, BPB_Len1                        ;┐ 准备更换 Build BPB 参数块
5885:   LEA     DI,[DI]. BDPB_BytesPerSector1       ;┘
5886: Set_DevP4:
5887:   LEA     SI,[BX+7]                           ;┐ 更换"BPB 参数块"的参数值
5888:   REP     MOVSB                               ;┘
5889:   PUSH    DS                                  ;┐
5890:   MOV     DS,CS:IO1_SEG                       ;│ 试图恢复 INT 1EH 中断向量
5891:   CALL    Recover_INT1EH                      ;│
5892:   POP     DS                                  ;┘
5893:   POP     DI
5894: Set_DevP5:                                    ;处理"磁道布局"字段
5895:   MOV     CX,[BX+26H]                         ;CX←"磁道布局"字段中的总扇区数
5896:   PUSH    DS
5897:   MOV     DS,CS:IO1_SEG
5898:   MOV     AddressField,CX                     ;设置"磁道中扇区数"
5899:   POP     DS
5900:   AND     WORD PTR ES:[DI]. BDPB_Attr_Status,-09  ;设置"磁道中所有扇区大小不同"的
                                                    ;标志位值
5901:   TEST    BYTE PTR [BX],4                     ;┐ 若磁道中的所有扇区大小不同,
5902:   JZ      Set_DevP6                           ;┘ 则跳转
5903:   OR      WORD PTR ES:[DI]. BDPB_Attr_Status,+08  ;设置"磁道中所有扇区大小相同"的
                                                    ;标志位值
5904: Set_DevP6:
5905:   CMP     CX,3FH                              ;┐ 若"磁道布局"字段中的   ┐
5906:   JA      Set_DevP9                           ;┘ 扇区数>63,则跳转     │
5907:   JCXZ    Set_DevP8                           ;若"磁道布局"字段中的扇 │
                                                    ;区数为 0,则跳转        │
5908:   MOV     DI, offset AddressField+2           ;DI 指向存放地址域数组  │
                                                    ;的缓冲区               │
5909:   LEA     SI,[BX+28H]                         ;                       │
5910:   MOV     ES,CS:IO1_SEG                       ;                       │更
5911: Set_DevP7:                                    ;                       │换
5912:   INC     DI                                  ;┐ 跳过地址域中的圆柱面  │磁
5913:   INC     DI                                  ;┘ 号和磁头号           │道
5914:   LODSW                                       ;┐ 复制扇区号           │布
5915:   STOSB                                       ;┘                      │局
5916:   LODSW                                       ;AX←每扇区字节数        │
5917:   CALL    Set_BytesPerSector                  ;将"每扇区字节数"转换成 │
                                                    ;对应的标志值→AL        │
5918:   STOSB                                       ;设置"每扇区字节数"的标 │
                                                    ;志值                   ┘
```

```
5919:     LOOP    Set_DevP7                          ;继续设置地址域数组的参数|
5920: Set_DevP8:                                     ;                        ┘
5921:     CLC
5922:     RET
5923: Set_DevP9:
5924:     MOV     AL,0CH                             ;AL←"一般性错误"的错误码
5925:     STC
5926:     RET
5927: Set_DevicePara  ENDP
5928: ;=============================================================
5929: ;相对地址偏移:0DC1
5930: ;功能:格式化并验证逻辑设备的磁道(类中子功能码=42H)
5931: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
5932: ;出口参数:CF=0,成功;CF=1,AL=设备驱动程序返回的错误码
5933: ;=============================================================
5934: FormatTrack  PROC  NEAR
5935:     LDS     BX,DRH_Ptr                 ;DS:BX 指向设备驱动程序请求标题
5936:     LDS     BX,[BX+13H]                ;DS:BX 指向类属 IOCTL 请求的参数块
5937:     TEST    BYTE PTR [BX], 01          ;┐ 若是"格式化并验证调用",则跳转
5938:     JZ      FormatT1                   ;┘
5939: ;"格式化状态检查"的功能调用
5940:     PUSH    DS                         ;┐
5941:     MOV     DS,CS:IO1_SEG              ;| 进行格式化状态检查
5942:     CALL    Check_FMTStatus            ;|
5943:     POP     DS                         ;┘
5944:     MOV     [BX], AL                   ;格式化状态检查码→类属 IOCTL 请求的参数块
5945:     CLC
5946:     RET
5947: FormatT1:
5948:     CMP     BYTE PTR ES:[DI].BDPB_DeviceType, 05  ;┐ 若块设备不是硬盘,则跳转
5949:     JNZ     FormatT2                              ;┘
5950:     MOV     DS,CS:IO1_SEG              ;DS←IO1 模块在常规内存低端的段地址值
5951:     JMP     VerifyTrack                ;转去验证逻辑设备的磁道
5952: FormatT2:
5953:     MOV     CX,[BX+1]                  ;CX←磁头号
5954:     MOV     DX,[BX+3]                  ;DX←圆柱面号
5955:     TEST    BYTE PTR [BX],02           ;┐
5956:     MOV     DS,CS:IO1_SEG              ;| 若只格式化一个磁道,则跳转
5957:     JZ      FormatT3                   ;┘
5958:     JMP     VerifyTrack2               ;若同时格式化多个磁道,则转去设置错误码
5959: FormatT3:
5960:     CALL    Check_FMTStatus            ;进行格式化状态检查
5961:     CMP     AL,1                       ;┐ 若 ROM BIOS 不支持"格式化状态检查"的功能,
```

```
5962:       JE      FormatT4                    ;┘ 则跳转
5963:       CMP     AL,3                        ;┐ ROM BIOS 支持"格式化状态检查",并返回确切
5964:       JNE     FormatT5                    ;┘ 的信息,则跳转
5965:       JMP     SHORT FormatT10             ;若软盘驱动器中没有盘片,则跳转
5966: FormatT4:
5967:       PUSH    DX                          ;┐
5968:       CALL    Set_DASDType                ;│ 为格式化软盘而设置 DASD 类型码
5969:       POP     DX                          ;┘
5970: FormatT5:
5971:       CALL    MultiDriveName              ;处理一个物理驱动器具有多个逻辑驱动器名的情
                                                ;况,确保对指定的介质进行格式化操作
5972:       MOV     AX,DX                       ;┐ 设置圆柱面号
5973:       MOV     Cylinder_FMT,AX             ;┘
5974:       MOV     Head_FMT,CL                 ;设置"磁头号"
5975:       MOV     AH,CL                       ;┐
5976:       MOV     BX, offset AddressField+2   ;│
5977:       MOV     CX,AddressField             ;│
5978: FormatT6:                                 ;│ 设置地址域中的磁头号和圆柱面号
5979:       MOV     WORD PTR [BX],AX            ;│
5980:       ADD     BX,4                        ;│
5981:       LOOP    FormatT6                    ;┘
5982:       MOV     CX,5                        ;CX←格式化磁道的出错重试次数
5983: FormatT7:
5984:       PUSH    CX
5985:       MOV     BX, offset AddressField+2   ;ES:BX 指向存放"地址域数组"的缓冲区
5986:       MOV     AL, BYTE PTR AddressField   ;AL←要格式化的扇区数
5987:       MOV     AH,5                        ;┐
5988:       MOV     ESDSValue,DS                ;│ 格式化指定的磁道
5989:       CALL    Do_INT13H                   ;┘
5990:       POP     CX
5991:       JB      FormatT8                    ;若格式化指定的磁道失败,则跳转
5992:       PUSH    CX                          ;┐
5993:       PUSH    BX                          ;│
5994:       XOR     BX,BX                       ;│
5995:       MOV     ESDSValue,BX                ;│
5996:       MOV     AL,BYTE PTR AddressField    ;│ 验证指定的磁道
5997:       MOV     AH,4                        ;│
5998:       MOV     CL,1                        ;│
5999:       CALL    Do_INT13H                   ;│
6000:       POP     BX                          ;│
6001:       POP     CX                          ;┘
6002:       JNB     FormatT12                   ;经验证指定的磁道,若格式化指定的磁道成功,则
                                                ;跳转
```

```
6003: FormatT8:
6004:     CALL    Reset_Drive                   ;复位软盘和硬盘控制器
6005:     MOV     FMTOpStatus,1                 ;置格式化磁道操作的失败标志
6006:     PUSH    AX
6007:     PUSH    CX
6008:     PUSH    DX
6009:     CALL    Check_FMTStatus               ;进行格式化状态检查
6010:     CMP     AL,1                          ;┐若 ROM BIOS 支持"格式化状态检查",则跳转
6011:     JNE     FormatT9                      ;┘
6012:     CALL    Set_DASDType                  ;为格式化软盘而设置 DASD 类型码
6013: FormatT9:
6014:     POP     DX
6015:     POP     CX
6016:     POP     AX
6017:     LOOP    FormatT7                      ;试图重新格式化指定的磁道
6018: FormatT10:                                ;格式化指定的磁道失败
6019:     MOV     FMTOpStatus,1                 ;置格式化磁道操作的失败标志
6020:     CMP     AH,6                          ;┐
6021:     JNE     FormatT11                     ;├若软盘片被取出,则置"超时"错误码
6022:     MOV     AH,80H                        ;┘
6023: FormatT11:
6024:     JMP     Set_CodeOfDriver              ;转去依据 INT 13H 返回的错误码设置设备驱动程
                                                ;序返回的错误码
6025: FormatT12:
6026:     MOV     FMTOpStatus,0                 ;清格式化磁道操作的失败标志
6027:     RET
6028: FormatTrack  ENDP
6029: ;=============================================================
6030: ;相对地址偏移:0E86
6031: ;功能:验证逻辑设备的磁道(类中子功能码=62H)
6032: ;入口参数:ES:DI=指向指定逻辑驱动器对应的 BDPB
6033: ;出口参数:CF=0,成功;CF=1,失败,AL=设备驱动程序返回的错误码
6034: ;=============================================================
6035: VerifyTrack  PROC  NEAR
6036:     PUSH    DS
6037:     LDS     BX,DRH_Ptr                    ;DS:BX 指向设备驱动程序请求标题
6038:     LDS     BX,[BX+13H]                   ;DS:BX 指向类属 IOCTL 请求的参数块
6039:     MOV     CX,[BX+3]                     ;CX←圆柱面号
6040:     MOV     AX,[BX+1]                     ;AX←磁头号
6041:     MOV     DX,[BX+5]                     ;DX←磁道数
6042:     MOV     BL,[BX]                       ;BL←专用功能码(Special Functions)
6043:     POP     DS
6044:     MOV     INT13H_CMDCode,4              ;设置"验证扇区"的命令码
```

```
6045:    MOV     Cylinder,CX                        ;设置"圆柱面号"
6046:    MOV     HeadNumber,AL                      ;设置磁头号
6047:    MOV     CX,AddressField                    ;CX←每个磁道的扇区数
6048:    TEST    BL,2                               ;┐若只验证一个磁道,则跳转
6049:    JZ      VerifyTrack1                       ;┘
6050:    MOV     AX,DX                              ;AX←磁道数
6051:    OR      AH,AH                              ;┐若磁道数≥256,则跳转
6052:    JNZ     VerifyTrack2                       ;┘
6053:    MUL     CL                                 ;计算要验证的扇区数→AX
6054:    OR      AH,AH                              ;┐若一次验证的扇区数≥256,则跳转
6055:    JNZ     VerifyTrack2                       ;┘
6056:    MOV     CX,AX                              ;CX←要验证的扇区数
6057:    TEST    WORD PTR ES:[DI].BDPB_Attr_Status,1  ;┐若介质可移动(即软盘),则跳转
6058:    JZ      VerifyTrack1                       ;┘
6059:    TEST    WORD PTR MultiTrack,0080H          ;┐若"multitrack=OFF",则跳转
6060:    JZ      VerifyTrack1                       ;┘
6061:    MOV     MultiTrack_OpFlag,1                ;设置"多磁道操作"的标志
6062: VerifyTrack1:
6063:    XOR     AX,AX                              ;AX←磁道内的起始扇区号
6064:    XOR     BX,BX                              ;┐设置传送缓冲区的地址
6065:    MOV     ESDSValue,BX                       ;┘
6066:    CALL    Track_Operation                    ;请求验证磁道的操作
6067:    MOV     MultiTrack_OpFlag,00               ;清"多磁道操作"的标志
6068:    RET
6069: VerifyTrack2:
6070:    MOV     AH,1                               ;AH←1(错误码,其意义为"一般错误")
6071:    JMP     Set_CodeOfDriver                   ;转去依据刚设定的 INT 13H 的错误码来设置设备
                                                    ;驱动程序返回的错误码
6072: VerifyTrack  ENDP
6073: ;=============================================================
6074: ;相对地址偏移:0EE8
6075: ;功能:读/写逻辑设备的磁道(类中子功能码=61H/41H)
6076: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
6077: ;出口参数:无
6078: ;=============================================================
6079: Process_DevTrack  PROC  NEAR
6080: Read_DeviceTrack  LABEL  NEAR                 ;读逻辑设备的磁道(类中的子功能码=61H)
6081:    MOV     INT13H_CMDCode, 02                 ;设置"读磁盘"的命令码
6082:    JMP     SHORT Pro_DevTK1
6083: Write_DeviceTrac  LABEL  NEAR                 ;写逻辑设备的磁道(类中的子功能码=41H)
6084:    MOV     INT13H_CMDCode, 03                 ;设置"写磁盘"的命令码
6085: Pro_DevTK1:
6086:    PUSH    ES
```

```
6087:     LES     BX,DRH_Ptr                    ;ES:BX 指向设备驱动程序请求标题
6088:     LES     BX,ES:[BX+13H]                ;ES:BX 指向类属 IOCTL 请求的参数块
6089:     MOV     AX, ES:[BX+03]                ;┐取出并保存"圆柱面号"
6090:     MOV     Cylinder, AX                  ;┘
6091:     MOV     AX, ES:[BX+01]                ;┐取出并保存"磁头号"
6092:     MOV     HeadNumber, AL                ;┘
6093:     MOV     AX, ES:[BX+05]                ;AX←读/写的起始扇区号(磁道内的起始扇区号)
6094:     MOV     CX, ES:[BX+07]                ;CX←读/写的扇区数
6095:     LES     BX, ES:[BX+09]                ;ES:BX 指向传送缓冲区
6096:     MOV     ESDSValue,ES                  ;设置"传送缓冲区"的段地址值
6097:     POP     ES
6098: ;=============================================
6099: ;相对地址偏移:0F1C
6100: ;功能:进行磁道的读、写或验证操作
6101: ;入口参数:AX=磁道内的起始扇区号;
6102: ;          BX=传送缓冲区的地址偏移值;
6103: ;          CX=扇区数
6104: ;出口参数:无
6105: ;=============================================
6106: Track_Operation  LABEL  NEAR
6107:     MOV     SPValue,SP                    ;保存堆栈指针寄存器值
6108:     CALL    MultiDriveName                ;处理一个物理软盘驱动器具有多个逻辑驱动器名
                                                ;的情况,确保对指定的介质进行操作
6109:     CMP     ModifyFlag,1                  ;┐若已更换过软盘参数表,则跳转
6110:     JE      Pro_DevTK2                    ;┘
6111:     PUSH    AX                            ;┐
6112:     PUSH    CX                            ;│修改软盘基数表的部分参数值:每道扇区数、马
6113:     CALL    Update_FPT_Para               ;│达启动时间、磁头的平稳时间
6114:     POP     CX                            ;│
6115:     POP     AX                            ;┘
6116: Pro_DevTK2:
6117:     MOV     SI, offset AddressField+2     ;SI←"地址域缓冲区"的地址偏移值
6118:     SHL     AX,1                          ;┐
6119:     SHL     AX,1                          ;│计算指定的扇区号对应的"地址域"的地址偏移
6120:     ADD     SI,AX                         ;┘值→SI
6121:     MOV     DX,1                                  ;┐
6122:     TEST    WORD PTR ES:[DI].BDPB_Attr_Status,8   ;│通过判定磁道中所有扇区大小是否相
6123:     JZ      Pro_DevTK3                            ;│同来决定欲操作的扇区数是否一次完
6124:     XCHG    DX,CX                                 ;┘成
6125: Pro_DevTK3:                               ;CX=循环次数;DX=一次完成的扇区数
6126:     PUSH    CX
6127:     PUSH    DX
6128:     INC     SI                            ;┐
```

```
6129:    INC     SI                                  ;| 取出并保存道内扇区号
6130:    LODSB                                       ;|
6131:    MOV     SectorNumber,AL                     ;┘
6132:    TEST    WORD PTR ES:[DI].BDPB_Attr_Status,1 ;┐ 若介质可装卸,则跳转
6133:    JZ      Pro_DevTK4                          ;┘
6134:    TEST    WORD PTR MultiTrack,0080H           ;┐ 若"multitrack=OFF",则跳转
6135:    JZ      Pro_DevTK4                          ;┘
6136:    MOV     SectorsOfRemain,DX                  ;保存当前欲操作的扇区数
6137:    MOV     AX,DX                               ;┐ 对 AX 指定的扇区数进行操作
6138:    CALL    Disk_Operate2                       ;┘
6139:    POP     DX
6140:    POP     CX
6141:    CLC
6142:    RET
6143: Pro_DevTK4:
6144:    LODSB                                       ;AL←"每扇区字节数"的标志值
6145:    PUSH    AX
6146:    PUSH    SI
6147:    PUSH    DS                                  ;┐
6148:    PUSH    AX                                  ;|
6149:    MOV     AH,MaxSectorsPerTrack               ;|
6150:    LDS     SI,Old_INT1EH_Vector                ;| 修改软盘参数表中的"每扇区字节数、每道扇区
6151:    MOV     [SI].FPT_SectorSize,AL              ;| 数"的参数值
6152:    MOV     [SI].FPT_SectorsPerTrack,AH         ;|
6153:    POP     AX                                  ;|
6154:    POP     DS                                  ;┘
6155:    MOV     AL,DL                               ;┐ 设置欲操作的扇区数
6156:    MOV     SectorsOfRemain,AX                  ;┘
6157:    CALL    Disk_Operate2                       ;对 AX 指定的扇区数进行操作
6158:    POP     SI
6159:    POP     AX
6160:    CALL    Calculate_Bytes                     ;计算完成的字节数(即一个扇区的字节数)→AX
6161:    ADD     BX,AX                               ;修改传送缓冲区的地址偏移
6162:    POP     DX
6163:    POP     CX
6164:    LOOP    Pro_DevTK3                          ;继续盘操作,直到全部完成指定的扇区数为止
6165:    CMP     ModifyFlag,1                        ;┐ 若更换过软盘参数表的,则跳转
6166:    JE      Pro_DevTK5                          ;┘
6167:    CALL    Set_Recover_Para                    ;处理软盘操作的后续工作
6168: Pro_DevTK5:
6169:    CLC
6170:    RET
6171: Process_DevTrack  ENDP
```

```
;==========================================================
;相对地址偏移:0FA0
;功能:确定"每扇区字节数"的标志值(0=128,1=256,2=512,3=1024)
;入口参数:AX=每扇区字节数
;出口参数:AL="每扇区字节数"的标志值
;==========================================================
Set_BytesPerSector  PROC  NEAR
    CMP     AH,2
    JA      Set_BytesPerSector_1
    MOV     AL,AH
    RET
Set_BytesPerSector_1:
    MOV     AL,3
    RET
Set_BytesPerSector  ENDP
;==========================================================
;相对地址偏移:0FAB
;功能:计算一个扇区的字节数
;入口参数:AL="每扇区字节数"的标志值(0=128,1=256,2=512,3=1024)
;出口参数:AX=一个扇区的字节数
;==========================================================
Calculate_Bytes  PROC  NEAR
    MOV     CL,AL
    MOV     AX,80H
    SHL     AX,CL
    RET
Calculate_Bytes  ENDP
;==========================================================
;相对地址偏移:0FB3
;功能:为格式化软盘而设置 DASD 类型码,DASD 类型码的意义如下:
;          AL=1 320/360KB 盘片在 360KB 驱动器中
;          AL=2 360KB 盘片在 1.2MB 驱动器中
;          AL=3 1.2MB 盘片在 1.2MB 驱动器中
;          AL=4 720KB 盘片在 720KB 驱动器中
;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
;出口参数:无
;==========================================================
Set_DASDType  PROC  NEAR
    CMP     FMTOpStatus,1                    ;┐若磁道格式化操作失败,则跳转
    JE      Set_DASDT1                       ;┘
    TEST    WORD PTR ES:[DI].BDPB_Attr_Status,80H;┐若未修改"设备参数:设备类型、圆柱
    JZ      Set_DASDT4                               ;┘面数、设备属性",则跳转
    AND     WORD PTR ES:[DI].BDPB_Attr_Status,0FF7FH ;清"设备参数修改位"(它表明修改
```

```
                                                    ;的设备参数已被软盘驱动器接收)
6215: Set_DASDT1:
6216:     MOV     FMTOpStatus,0                     ;清磁道格式化操作的失败标志
6217:     MOV     GapLength_FMT,50H                 ;设置格式化时扇区间的间隔长度 ┐
6218:     MOV     AL,4                              ;720KB,3.5 英寸低密软盘驱动器 |
6219:     CMP     BYTE PTR ES:[DI].BDPB_DeviceType,2 ;┐ 若软盘驱动器是 720KB,则跳转 |
6220:     JE      Set_DASDT3                        ;┘                            |
6221:     CMP     BYTE PTR ES:[DI].BDPB_DeviceType,1 ;┐ 若软盘驱动器是 1.2MB,则跳转 |
6222:     JE      Set_DASDT2                        ;┘                            | 确定
6223:     MOV     AL,1                              ;320/360KB,5.25 英寸低密软盘驱 | DASD
                                                    ;动器                         | 类型
6224:     JMP     SHORT Set_DASDT3                  ;                             | 码→
6225: Set_DASDT2:                                   ;                             | AL
6226:     MOV     AL,2                              ;1.2MB 的高密软盘驱动器作低密 |
                                                    ;(360KB)用                    |
6227:     CMP     MediaTypeFlag,0                   ;┐ 若是高密软盘驱动器作低密用, |
6228:     JNE     Set_DASDT3                        ;┘ 则跳转                     |
6229:     MOV     AL,3                              ;1.2MB 的高密软盘驱动器存取   |
                                                    ;1.2MB 的盘片                 |
6230:     MOV     GapLength_FMT,54H                 ;设置格式化时扇区间的间隔长度 |
6231: Set_DASDT3:                                   ;                             ┘
6232:     PUSH    DS                                ;┐
6233:     PUSH    SI                                ;|
6234:     MOV     DS,ZeroValue                      ;|
6235:     LDS     SI,DWORD PTR DS:Vector_1EH_OFS    ;| 修改软盘参数表的"磁头平稳时间"
6236:     MOV     [SI].FPT_HeadLoadTime,0FH         ;|
6237:     POP     SI                                ;|
6238:     POP     DS                                ;┘
6239:     MOV     AH,17H                            ;┐
6240:     MOV     DL,ES:[DI].BDPB_DriveNumber       ;| 为格式化软盘而设置 DASD 类型码
6241:     INT     13H                               ;┘
6242: Set_DASDT4:
6243:     MOV     AH,BYTE PTR ES:[DI].BDPB_SectorsPerTrack1 ;┐ 取出并保存"每道扇区数"
6244:     MOV     SecPerTK_FMT,AH                           ;┘
6245:     RET
6246: Set_DASDType  ENDP
6247: ;=============================================================
6248: ;相对地址偏移:1017
6249: ;功能:进行格式化状态检查,决定是否支持指定的磁道数和每道扇区数的组合
6250: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
6251: ;出口参数:AL=0:ROM BIOS 支持"格式化状态检查",软盘驱动器支持磁道数和每道扇区数的指
          定组合;
6252: ;          AL=1:ROM BIOS 不支持"格式化状态检查";
```

```
6253: ;          AL=2:ROM BIOS 支持"格式化状态检查",但软盘驱动器不支持磁道数和每道扇区数
                 的指定组合;
6254: ;          AL=3:ROM BIOS 支持"格式化状态检查",但 ROM BIOS 不能决定是否允许该磁道数和
                 每道扇区数的指定组合,因为这个软盘驱动器是空的
6255: ;==========================================================
6256: Check_FMTStatus  PROC  NEAR
6257:     PUSH    CX
6258:     PUSH    DX
6259:     CMP     FMTOpStatus,1                  ;┐若"格式化磁道操作"失败,则跳转
6260:     JE      Check_FMTS2                    ;┘
6261:     XOR     AL,AL
6262:     CMP     ModifyFlag,1                   ;┐若格式化时未修改 INT 1EH 中断向量,
6263:     JNE     Check_FMTS1                    ;┘则跳转
6264:     JMP     Check_FMTS8
6265: Check_FMTS1:
6266:     PUSH    ES                             ;┐
6267:     PUSH    SI                             ;│
6268:     MOV     ES,ZeroValue                   ;│
6269:     LES     SI,DWORD PTR ES:Vector_1EH_OFS ;│取出并保存先前的 INT 1EH 中断向量,
6270:     MOV     WORD PTR Old_INT1EH_Vector,SI  ;│同时还修改软盘参数表的"磁头平稳时
6271:     MOV     WORD PTR Old_INT1EH_Vector+2,ES;│间"
6272:     MOV     ES:[SI].FPT_HeadLoadTime,0FH   ;│
6273:     POP     SI                             ;│
6274:     POP     ES                             ;┘
6275: Check_FMTS2:
6276:     MOV     CX,ES:[DI].BDPB_NumberOfCylinder  ;CX←设备支持的最大圆柱面数
6277:     DEC     CX                                ;┐
6278:     AND     CH,03                             ;│
6279:     ROR     CH,1                              ;│拼成 INT 13H 的入口参数格式(CX
6280:     ROR     CH,1                              ;│寄存器值)
6281:     XCHG    CH,CL                             ;│
6282:     OR      CL,BYTE PTR ES:[DI].BDPB_SectorsPerTrack];┘
6283:     MOV     DL,ES:[DI].BDPB_DriveNumber       ;DL←物理驱动器号
6284:     PUSH    ES
6285:     PUSH    DS
6286:     PUSH    SI
6287:     PUSH    DI
6288:     MOV     AH,18H                            ;┐为格式化而设置介质类型
6289:     INT     13H                               ;┘
6290:     JC      Check_FMTS4                       ;若失败,则跳转
6291:     CMP     FMTOpStatus,1                     ;┐若格式化操作失败过,则跳转
6292:     JE      Check_FMTS3                       ;┘
6293:     PUSH    ES                                ;┐
```

```
6294:    MOV    ES,ZeroValue                          ;| 保存并修改 INT 1EH 中断向量(即
6295:    LES    SI,DWORD PTR ES:Vector_1EH_OFS        ;| 保存先前的软盘参数表的入口地
6296:    MOV    WORD PTR Old_INT1EH_Vector_FMT,SI     ;| 址,同时更换当前使用的软盘参数
6297:    MOV    WORD PTR Old_INT1EH_Vector_FMT+2,ES   ;| 表)
6298:    MOV    ES,ZeroValue                          ;|
6299:    MOV    ES:Vector_1EH_OFS,DI                  ;|
6300:    POP    ES:Vector_1EH_SEG                     ;┘
6301:    MOV    ModifyFlag,1                          ;置 INT 1EH 中断向量的更换标志
6302: Check_FMTS3:
6303:    XOR    AL,AL                                 ;置"ROM BIOS 支持格式化状态检查,并允
                                                      ;许磁道数和每道扇区数的指定组合"的返
                                                      ;回码
6304:    MOV    FMTOpStatus,AL                        ;置"格式化状态检查"成功的标志
6305:    JMP    SHORT Check_FMTS7
6306: Check_FMTS4:
6307:    CMP    AH,0CH                                ;┐ 若请求的介质类型不存在(即不允许磁
6308:    JE     Check_FMTS5                           ;┘ 道数和每道扇区数的相合),则跳转
6309:    CMP    AH,80H                                ;┐ 若设备超时,则跳转
6310:    JE     Check_FMTS6                           ;┘
6311:    MOV    AL,1                                  ;置"ROM BIOS 不支持格式化状态检查"的
                                                      ;返回码
6312:    JMP    SHORT Check_FMTS7
6313: Check_FMTS5:
6314:    MOV    AL,2                                  ;置"ROM BIOS 支持格式化状态检查,但不
                                                      ;允许磁道数和每道扇区数的指定组合"的
                                                      ;返回码
6315:    JMP    SHORT Check_FMTS7
6316: Check_FMTS6:
6317:    MOV    AL,3                                  ;置"ROM BIOS 支持格式化状态检查,但软
                                                      ;盘驱动器为空"的返回码
6318: Check_FMTS7:
6319:    POP    DI
6320:    POP    SI
6321:    POP    DS
6322:    POP    ES
6323: Check_FMTS8:
6324:    POP    DX
6325:    POP    CX
6326:    RET
6327: Check_FMTStatus  ENDP
6328: ;=============================================================
6329: ;相对地址偏移:10B4
6330: ;功能:复位软盘和硬盘控制器
```

```
;入口参数:无
;出口参数:CF=0,成功,此时 AH=0;CF=1,失败,此时 AH=错误码
;=============================================================
Reset_Drive  PROC  NEAR
    PUSH   AX
    CMP    ModifyFlag,1                    ;┐ 若没有更换软盘参数表,则跳转
    JNZ    Reset_Drive1                    ;┘
    MOV    FMTOpStatus,1                   ;设置"磁道格式化操作"的失败标志
Reset_Drive1:
    XOR    AH,AH                           ;┐ 复位软盘和硬盘控制器
    INT    13H                             ;┘
    MOV    DiskNumber,-1                   ;置当前驱动器操作失败的标志
    POP    AX
    RET
Reset_Drive  ENDP

;=============================================================
;相对地址偏移:10CC
;功能:完成 AH 指定的 INT 13H 的磁盘操作服务
;入口参数:AL=扇区数;
;         AH=功能码;
;         CL=扇区号
;出口参数:CF=0 成功; CF=1,失败,此时 AH=错误码
;=============================================================
Do_INT13H  PROC  NEAR
    PUSH   BX
    PUSH   SI
    TEST   BYTE PTR ModifyFlag,1           ;┐ 若没有更换软盘参数表,则跳转
    JNZ    Do_13H2                         ;┘
    PUSH   AX                              ;                          ┐
    PUSH   ES                              ;                          |
    CMP    BYTE PTR ES:[DI].BDPB_DeviceType,2 ;                       |
    PUSHF                                  ;                          |
    MOV    ES,ZeroValue                    ;┐                         |
    LES    SI,DWORD PTR ES:Vector_1EH_OFS  ;| 保存 INT 1EH 的中断向量  | 修
    MOV    WORD PTR Old_INT1EH_Vector,SI   ;┘                         | 改
    MOV    WORD PTR Old_INT1EH_Vector+2,ES ;┘                         | 软
    MOV    AL,SecPerTK_FMT                 ;┐ 修改"每道扇区数"       | 盘
    MOV    ES:[SI].FPT_SectorsPerTrack,AL  ;┘                         | 参
    MOV    AL,GapLength_FMT                ;┐ 修改"格式化时扇区间的间" | 数
    MOV    ES:[SI].FPT_LengthOfGapInFMT,AL ;┘ 隔长度                 | 表
    MOV    ES:[SI].FPT_HeadLoadTime,0FH    ;修改"磁头平稳时间"       | 的
    POPF                                   ;                          | 部
```

```
6373:     JNE     Do_13H1                          ;若不是 720KB 的 3.5 英寸软盘驱 | 分
                                                   ;动器,则跳转                    | 参
6374:     MOV     ES:[SI].FPT_StartupWaitTime,04   ;修改"马达启动时间"            | 数
6375: Do_13H1:                                     ;                               |
6376:     POP     ES                               ;                               |
6377:     POP     AX                               ;                               ┘
6378: Do_13H2:
6379:     MOV     DX,Cylinder_FMT                  ;┐ CH←圆柱面号
6380:     MOV     CH,DL                            ;┘
6381:     MOV     DL,ES:[DI].BDPB_DriveNumber      ;DL←物理驱动器号
6382:     MOV     DH,Head_FMT                      ;DH←磁头号
6383:     PUSH    ES                               ;┐
6384:     MOV     ES,ESDSValue                     ;| 请求 AH 指定的 INT 13H 的磁盘操作服
6385:     INT     13H                              ;| 务
6386:     POP     ES                               ;┘
6387:     POP     SI
6388:     POP     BX
6389:     RET
6390: Do_INT13H  ENDP
6391: ;=================================================
6392: ;相对地址偏移:1124
6393: ;功能:取、置逻辑驱动器(块设备的命令码=17H/18H)
6394: ;入口参数:AL=逻辑驱动器号(0=A,1=B,…)
6395: ;出口参数:CF=0,成功
6396: ;=================================================
6397: Disk_Map  PROC  NEAR
6398: Disk_Get_Map  LABEL  NEAR                    ;取逻辑驱动器(块设备的命令码=17H)
6399:     CALL    Get_BDPB_Node                    ;取出 AL 指定的逻辑驱动器对应的 BDPB
6400:     MOV     AL, ES:[DI].BDPB_DriveNumber     ;AL←物理驱动器号
6401:     LES     DI, DWORD PTR BDPB_Head          ;ES:DI 指向第一个 BDPB
6402: Get_Map1:                                     ;┐
6403:     CMP     ES:[DI].BDPB_DriveNumber,AL      ;|
6404:     JNE     Get_Map2                         ;| 确定 AL 指定的物理驱动器当前对
6405:     TEST    WORD PTR ES:[DI].BDPB_Attr_Status,0020H;| 应的逻辑驱动器的 BDPB(即确定物
6406:     JNZ     Set_Map1                         ;| 理驱动器当前使用的逻辑驱动器
6407: Get_Map2:                                     ;| 名)
6408:     LES     DI,DWORD PTR ES:[DI].BDPB_NextPtr_OFS;|
6409:     JMP     SHORT Get_Map1                   ;┘
6410: ;相对地址偏移:1142
6411: Disk_Set_Map  LABEL  NEAR                    ;设置逻辑驱动器(块设备的命令码=18H)
6412:     CALL    Get_BDPB_Node                    ;取出 AL 指定的逻辑驱动器对应的 BDPB
6413:     MOV     EnableWR_Flag, 1                 ;┐ 处理一个物理软盘驱动器具有多个逻辑
6414:     CALL    MultiDriveName                   ;| 驱动器名的情况,确保对指定的介质进
```

```
6415:     MOV     EnableWR_Flag, 0                    ;┘ 行操作
6416: Set_Map1:                                       ;┐
6417:     XOR     CL,CL                               ;│
6418:     TEST    WORD PTR ES:[DI].BDPB_Attr_Status,10H ;│ 设置返回的参数值(即物理驱动器
6419:     JZ      Set_Map2                            ;│ 当前使用的逻辑驱动器号(0=当前
6420:     MOV     CL,ES:[DI].BDPB_LogicalNumber       ;│ 使用的驱动器名或物理驱动器只有
6421:     INC     CL                                  ;│ 一个逻辑驱动器名,1=A,2=B,
6422: Set_Map2:                                       ;┘ …)
6423:     LDS     BX,DRH_Ptr                          ;DS:BX 指向设备驱动程序请求标题
6424:     MOV     [BX+01],CL                          ; 设置返回的参数值(0=当前使用的驱动器
                                                      ;名或物理驱动器只有一个逻辑驱动器名,
                                                      ;1=A,2=B,…)
6425:     CLC                                         ;CF←0(成功)
6426:     RET                                         ;正常结束
6427: Disk_Map  ENDP
6428: ;=============================================================
6429: ;相对地址偏移:116B
6430: ;功能:试图恢复 INT 1EH 的中断向量
6431: ;入口参数:无
6432: ;出口参数:无
6433: ;=============================================================
6434: Recover_INT1EH  PROC  NEAR
6435:     PUSH    AX
6436:     XOR     AL,AL                               ;┐ 清"格式化磁道操作"的失败标志
6437:     MOV     FMTOpStatus,AL                      ;┘
6438:     XCHG    AL, ModifyFlag                      ;取出 INT 1EH 中断向量的更换标志,同时
                                                      ;还清除该标志
6439:     OR      AL, AL                              ;┐ 若未更换 INT 1EH 中断向量,则跳转
6440:     JZ      Rec_1EH1                            ;┘
6441:     PUSH    SI                                  ;┐
6442:     PUSH    DS                                  ;│
6443:     PUSH    ES                                  ;│
6444:     LDS     SI, DWORD PTR Old_INT1EH_Vector_FMT ;│ 恢复原先的 INT 1EH 中断向量(即
6445:     MOV     ES,CS:IO1_SEG                       ;│ 使用原来的软盘参数表)
6446:     MOV     ES,ES:ZeroValue                     ;│
6447:     MOV     ES:Vector_1EH_OFS, SI               ;│
6448:     MOV     ES:Vector_1EH_SEG, DS               ;│
6449:     POP     ES                                  ;│
6450:     POP     DS                                  ;│
6451:     POP     SI                                  ;┘
6452: Rec_1EH1:
6453:     POP     AX
6454:     CLC
```

```
6455:     RET
6456: Recover_INT1EH   ENDP
6457: ;=========================================================
6458: ;相对地址偏移:119A
6459: ;功能:取磁盘的卷系列号、卷标名和文件系统类型标志串(类中子功能码=66H)
6460: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
6461: ;出口参数:CF=0,成功;CF=1,失败,AL=错误码(07H)
6462: ;=========================================================
6463: Get_ExtMediaState   PROC   NEAR
6464:     CALL    Set_RebuildBPBFlag                    ;设置软盘驱动器对应的 BDPB 的"重建 BPB
                                                      ;参数块标志位"
6465:     MOV     AL,BYTE PTR ES:[DI].BDPB_LogicalNumber ;AL←逻辑驱动器号   ┐
6466:     MOV     BYTE PTR INT13H_CMDCode,2              ;"读扇区"的命令码   │读取引导记
6467:     CALL    ReadBootSector                         ;                   ┘录
6468:     JC      Get_ExtMediaS_2                       ;若读引导记录失败,则跳转
6469:     CMP     BYTE PTR DiskBuffer+15H,0F0H          ;┐若"介质描述符"非法,则跳转
6470:     JB      Get_ExtMediaS_1                       ;┘
6471:     CMP     BYTE PTR DiskBuffer+26H,29H           ;┐若引导记录中没有扩充 BPB 参数块,则
6472:     JNE     Get_ExtMediaS_1                       ;┘跳转
6473:     LES     DI,DRH_Ptr                            ;┐ES:DI 指向类属 IOCTL 请求的参数块
6474:     LES     DI,DWORD PTR ES:[BX+13H]              ;┘
6475:     MOV     SI,offset DiskBuffer+27H              ;DS:SI 指向位于引导记录中的"卷系列号"
6476:     ADD     DI,2                                  ;┐
6477:     MOV     CX,17H                                ;│"卷系列号、卷标名和文件系统类型标志
6478:     REP     MOVSB                                 ;┘串"→类属 IOCTL 请求的参数块
6479:     CLC                                           ;CF←0(成功)
6480:     RET
6481: Get_ExtMediaS_1:
6482:     MOV     AL,7                                  ;AL←"未知介质"的错误码
6483:     STC                                           ;CF←1(失败)
6484: Get_ExtMediaS_2:
6485:     RET
6486: Get_ExtMediaState   ENDP
6487: ;=========================================================
6488: ;相对地址偏移:11D2
6489: ;功能:设置磁盘的卷系列号、卷标名和文件系统类型标志串(类中子功能码=46H)
6490: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
6491: ;出口参数:CF=0,成功;CF=1,失败,AL=错误码(07H)
6492: ;=========================================================
6493: Set_ExtMediaState   PROC   NEAR
6494:     CALL    Set_RebuildBPBFlag                    ;设置软盘驱动器对应的 BDPB 的"重建 BPB
                                                      ;参数块标志位"
6495:     MOV     AL,BYTE PTR ES:[DI].BDPB_LogicalNumber ;AL←逻辑驱动器号 ┐
```

```
6496:    MOV     DL,AL                              ;                          ┐
6497:    MOV     BYTE PTR INT13H_CMDCode,2          ;"读扇区"的命令码          │读取引导记录
6498:    PUSH    DX                                 ;                          │
6499:    CALL    ReadBootSector                     ;                          │
6500:    POP     DX                                 ;                          ┘
6501:    JC      Set_ExtMediaS_2                    ;若读引导记录失败,则跳转
6502:    CMP     BYTE PTR DiskBuffer+15H,0F0H       ;┐若"介质描述符"非法,则跳转
6503:    JB      Set_ExtMediaS_1                    ;┘
6504:    CMP     BYTE PTR DiskBuffer+26H,29H        ;┐若引导记录中没有扩充BPB参数块,则
6505:    JNE     Set_ExtMediaS_1                    ;┘跳转
6506:    PUSH    ES
6507:    PUSH    DI
6508:    PUSH    DS                                 ;┐ES:DI指向存放"卷系列号、卷标名和文
6509:    POP     ES                                 ;│件系统类型标志串"的缓冲区
6510:    MOV     DI,offset DiskBuffer+27H           ;┘
6511:    LDS     SI,DRH_Ptr                         ;┐DS:SI指向类属IOCTL请求的参数块
6512:    LDS     SI,DWORD PTR [SI+13H]              ;┘
6513:    ADD     SI,2                               ;┐更换引导记录中的"卷系列号、卷标名和
6514:    MOV     CX,17H                             ;│文件系统类型标志串"
6515:    REP     MOVSB                              ;┘
6516:    PUSH    ES
6517:    POP     DS
6518:    POP     DI
6519:    POP     ES
6520:    CALL    Set_BDPBExtMediaState              ;更换BDPB的"卷系列号、卷标名和文件系
                                                    ;统类型标志串"
6521:    MOV     AL,DL                              ;                  ┐将修改后的引
6522:    MOV     BYTE PTR INT13H_CMDCode,3          ;"写扇区"的命令码  │导记录写回到
6523:    CALL    ReadBootSector                     ;                  ┘对应的介质上
6524:    MOV     Drive_Number,0FFH                  ;置"介质更换"标志
6525:    RET
6526: Set_ExtMediaS_1:
6527:    MOV     AL,7                               ;AL←"未知介质"的错误码
6528:    STC                                        ;CF←1(失败)
6529: Set_ExtMediaS_2:
6530:    RET
6531: Set_ExtMediaState  ENDP
6532: ;=============================================================
6533: ;相对地址偏移:1226
6534: ;功能:读取引导记录所在扇区的内容
6535: ;入口参数:AL=逻辑驱动器号
6536: ;出口参数:CF=0,成功;CF=1,失败,AL=设备驱动程序返回的错误码
6537: ;=============================================================
```

```
6538: ReadBootSector  PROC  NEAR
6539:     PUSH    ES
6540:     PUSH    DI
6541:     PUSH    BX
6542:     PUSH    DS                              ;┐
6543:     POP     ES                              ;| ES:DI 指向传送缓冲区(该传送缓冲区
6544:     MOV     DI,offset DiskBuffer            ;┘ 位于 IO1 模块中)
6545:     XOR     DX,DX                           ;┐ 设置起始扇区号(即引导记录所在的扇
6546:     MOV     StartingSector_HW,DX            ;┘ 区)
6547:     MOV     CX,1                            ;CX←读取的扇区数
6548:     CALL    Disk_Operation                  ;读取引导记录所在的扇区
6549:     POP     BX
6550:     POP     DI
6551:     POP     ES
6552:     RET
6553: ReadBootSector  ENDP
6554: ;=============================================================
6555: ;相对地址偏移:123E
6556: ;功能:设置软盘驱动器对应的 BDPB 的"重建 BPB 参数块标志位"
6557: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
6558: ;出口参数:无
6559: ;=============================================================
6560: Set_RebuildBPBFlag  PROC  NEAR
6561:     MOV     DL,BYTE PTR ES:[DI].BDPB_DriveNumber  ;DL←物理驱动器号
6562:     OR      DL,DL                           ;┐ 若指定的逻辑驱动器对应于硬盘,则直
6563:     JS      Set_RebuildBPBF_RET             ;┘ 接返回
6564:     TEST    WORD PTR ES:[DI].BDPB_Attr_Status,4 ;┐ 若通过类属 IOCTL 请求重建过 Build BPB
6565:     JNZ     Set_RebuildBPBF_RET             ;┘ 参数块,则直接返回
6566:     CMP     ChangeLine,1                    ;┐ 若软盘驱动器支持 Change Line,则直接
6567:     JNE     Set_RebuildBPBF_RET             ;┘ 返回
6568:     CALL    Test_ChangeLineStatus           ;┐ 若软盘驱动器不支持 Change Line,则直
6569:     JZ      Set_RebuildBPBF_RET             ;┘ 接返回
6570:     MOV     AH,16H                          ;┐ 取软盘驱动器的 Change Line 状态→AH
6571:     INT     13H                             ;┘
6572:     JNC     Set_RebuildBPBF_RET             ;若 ROM-BIOS 支持该功能,则直接返回
6573:     PUSH    BX                              ;┐
6574:     MOV     BX,40H                          ;| 设置 BDPB 的"重建 BPB 参数块标志位"
6575:     CALL    Set_BDPB_DevInfo                ;|
6576:     POP     BX                              ;┘
6577: Set_RebuildBPBF_RET:
6578:     RET
6579: Set_RebuildBPBFlag  ENDP
6580: ;=============================================================
```

```
6581: ;相对地址偏移:1269
6582: ;功能:取访问标志(类中子功能码=67H)
6583: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
6584: ;出口参数:"访问标志"值→类属 IOCTL 请求的参数块
6585: ;=============================================
6586: Get_AccessState  PROC  NEAR
6587:     LDS     BX,DRH_Ptr                          ;DS:BX 指向设备驱动程序请求标题
6588:     LDS     BX,DWORD PTR [BX+13H]               ;DS:BX 指向类属 IOCTL 请求的参数块
6589:     MOV     AL,0                                    ;┐
6590:     TEST    WORD PTR ES:[DI].BDPB_Attr_Status,200H  ;│
6591:     JNZ     Get_AccessS_1                           ;│ AL←"访问标志"值(0:不允许磁盘
6592:     INC     AL                                      ;│ 访问;1:允许磁盘访问)
6593: Get_AccessS_1:                                     ;┘
6594:     MOV     BYTE PTR [BX+1],AL                  ;访问标志值→类属 IOCTL 请求的参数块
6595:     RET
6596: Get_AccessState  ENDP
6597: ;=============================================
6598: ;相对地址偏移:1280
6599: ;功能:设置访问标志(类中子功能码=47H)
6600: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
6601: ;出口参数:"访问标志"值→BDPB
6602: ;=============================================
6603: Set_AccessState  PROC  NEAR
6604:     LDS     BX,DRH_Ptr                          ;DS:BX 指向设备驱动程序请求标题
6605:     LDS     BX,DWORD PTR [BX+13H]               ;DS:BX 指向类属 IOCTL 请求的参数块
6606:     AND     WORD PTR ES:[DI].BDPB_Attr_Status,0FDFFH ;┐
6607:     CMP     BYTE PTR [BX+1],0                       ;│
6608:     JNE     Set_AccessS_RET                         ;│ "访问标志"值→BDPB
6609:     OR      WORD PTR ES:[DI].BDPB_Attr_Status,0200H ;│
6610: Set_AccessS_RET:                                    ;┘
6611:     RET
6612: Set_AccessState  ENDP
6613: ;=============================================
6614: ;相对地址偏移:129A
6615: ;功能:判定块设备的类属 IOCTL 请求的类中子功能码是否有效(块设备的命令码=19H)
6616: ;入口参数:无
6617: ;出口参数:CF=0,有效;CF=1,失败,AX=设备驱动程序返回的状态字值
6618: ;=============================================
6619: Disk_19H  PROC  NEAR
6620:     PUSH    ES
6621:     LES     BX,DRH_Ptr                          ;ES:BX 指向设备驱动程序请求标题
6622:     MOV     AX,WORD PTR ES:[BX+0DH]             ;AL←设备类型码,AH←类中子功能码
6623:     CMP     AL,8                                ;┐ 若设备不是块设备,则跳转
```

```
6624:    JNE     Disk_19H_2                          ;┘
6625:    PUSH    CS
6626:    POP     ES
6627:    MOV     CX,0BH                              ;CX←块设备驱动程序支持类属 IOCTL 请
                                                     ;求的类中子功能码个数
6628:    MOV     DI,offset GenericIOCTLFuncCode      ;ES:DI 指向块设备驱动程序支持的类属
                                                     ;IOCTL 请求的类中子功能码表
6629:    XCHG    AL,AH                               ;┐
6630:    REPNE   SCASB                               ;├若类属 IOCTO 请求的类中子功能码非
6631:    JNZ     Disk_19H_2                          ;┘法,则跳转
6632:    MOV     AX,100H                             ;AH←块设备驱动程序返回的状态字值
6633:    JMP     SHORT Disk_19H_1
6634: Disk_19H_1:
6635:    POP     ES
6636:    CLC                                         ;CF←0(有效)
6637:    RET
6638: Disk_19H_2:
6639:    POP     ES
6640:    JMP     InvalidCmd                          ;转去置"非法命令"的错误信息(返回状态
                                                     ;字的"错误位、完成位"以及错误码)
6641: Disk_19H   ENDP
6642:
6643: ;=============================================================
6644: ;相对地址偏移:12C1
6645: ;功能:测试 INT 13H 中断服务程序是否支持功能 20H,并取回相应的信息
6646: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
6647: ;出口参数:CF=0,成功;CF=1,失败,AX=设备驱动程序返回的状态字值
6648: ;=============================================================
6649: Get_68H  PROC  NEAR
6650:    LDS     BX,DRH_Ptr                          ;DS:BX 指向设备驱动程序请求标题
6651:    LDS     BX,DWORD PTR [BX+13H]               ;DS:BX 指向类属 IOCTL 请求的参数块
6652:    MOV     WORD PTR [BX],0                     ;清 INT 13H 中断服务程序支持功能 20H 的
                                                     ;标志
6653:    MOV     DL,BYTE PTR ES:[DI].BDPB_DriveNumber    ;DL←物理驱动器号
6654:    XOR     DH,DH
6655:    MOV     AH,20H
6656:    INT     13H
6657:    JC      Get_68H_3
6658:    INC     BYTE PTR [BX]                       ;设置 INT 13H 中断服务程序支持功能 20H 的标志
6659: Get_68H_1:
6660:    DEC     AL
6661:    CMP     AL,2
6662:    JE      Get_68H_2
```

```
6663:    ADD     AL,4
6664:    CMP     AL,7
6665:    JE      Get_68H_2
6666:    CMP     AL,9
6667:    JNE     Get_68H_4
6668: Get_68H_2:
6669:    MOV     BYTE PTR DS:[BX+1],AL
6670:    CLC                                  ;CF←0(成功)
6671:    RET
6672: Get_68H_3:
6673:    CMP     AH,32H
6674:    JE      Get_68H_1
6675:    MOV     AL,2                         ;AL←"驱动器未就绪"的错误码
6676:    CMP     AH,31H                       ;┐若指定的驱动器未就绪,则跳转
6677:    JE      Get_68H_5                    ;┘
6678: Get_68H_4:
6679:    MOV     AL,7                         ;AL←"未知介质"的错误码
6680: Get_68H_5:
6681:    MOV     AH,81H                       ;设置返回状态字的"错误位、完成位"
6682:    STC                                  ;CF←1(失败)
6683:    RET
6684: Get_68H  ENDP
6685:    DB      0
6686: ;=============================================================
6687: ;相对地址偏移:1302
6688: ;功能:由 IO2 模块提供的 INT 2FH 中断服务程序
6689: ;=============================================================
6690: IO2_INT2F_ISR  PROC  FAR
6691:    CMP     AH,13H                       ;┐若是"设置 INT 13H 中断服务程序的入口地址"
6692:    JZ      IO2_INT2F_ISR2               ;┘的功能,则跳转
6693:    CMP     AH,8                         ;┐若是"支持 driver.sys"的功能,则跳转
6694:    JE      IO2_INT2F_ISR4               ;┘
6695:    CMP     AH,16H                       ;┐若是"支持 windows"功能,则跳转
6696:    JE      IO2_INT2F_ISR9               ;┘
6697:    CMP     AH,4AH                       ;┐
6698:    JNE     IO2_INT2F_ISR1               ;├若是"访问高内存区(HMA)"的功能,则跳转
6699:    JMP     IO2_INT2F_ISR12              ;┘
6700: IO2_INT2F_ISR1:
6701:    IRET                                 ;中断返回
6702: IO2_INT2F_ISR2:                         ;设置 INT 13H 中断服务程序的入口地址
6703:    PUSH    AX
6704:    MOV     AX,DS
6705:    MOV     DS,CS:IO1_SEG                ;DS←IO1 模块在常规内存低端的段地址值
```

```
6706:    PUSH    WORD PTR DS:[0B4H]          ;┐ 保存原来的 INT 13H 中断服务程序的入口地址
6707:    PUSH    WORD PTR DS:[0B6H]          ;┘
6708:    PUSH    WORD PTR DS:[106H]          ;┐ 保存在系统热启动时恢复的 INT 13H 中断服务
                                             ;│ 程序的入口地址(在系统热启动时,一般应使用
6709:    PUSH    WORD PTR DS:[108H]          ;┘ ROM BIOS 自检时设定的中断向量)
6710:    MOV     WORD PTR DS:[0B4H],DX       ;┐ 设置新的 INT 13H 中断服务程序的入口地址
6711:    MOV     WORD PTR DS:[0B6H],AX       ;┘
6712:    MOV     WORD PTR DS:[106H],BX       ;┐ 更换在系统热启动时恢复的 INT 13H 中断服务
6713:    MOV     WORD PTR DS:[108H],ES       ;┘ 程序的入口地址
6714:    POP     ES                          ;┐ ES:BX=先前设定的在系统热启动时恢复的
6715:    POP     BX                          ;┘ INT 13H 中断服务程序的入口地址
6716:    POP     DS                          ;┐ DS:DX=先前设定的 INT 13H 中断服务程序的
6717:    POP     DX                          ;┘ 入口地址
6718:    POP     AX
6719: IO2_INT2F_ISR3:
6720:    IRET                                ;中断返回
6721: IO2_INT2F_ISR4:                        ;支持"driver.sys"运行的功能
6722:    CMP     AL,0F8H                     ;┐ 若子功能码不合法,则中断返回
6723:    JAE     IO2_INT2F_ISR3              ;┘
6724:    OR      AL,AL                       ;┐ 若不是"取安装状态",则跳转
6725:    JNZ     IO2_INT2F_ISR5              ;┘
6726:    MOV     AL,0FFH                     ;AL←-1(表示已安装)
6727:    JMP     SHORT IO2_INT2F_ISR3
6728: IO2_INT2F_ISR5:
6729:    CMP     AL,1                        ;┐ 若是"增加新的块设备",则跳转
6730:    JE      IO2_INT2F_ISR6              ;┘
6731:    CMP     AL,3                        ;┐ 若是"取 BDPB 链",则跳转
6732:    JE      IO2_INT2F_ISR7              ;┘
6733: ;处理"执行设备驱动程序请求"的子功能
6734:    PUSH    DS                          ;┐
6735:    MOV     DS,CS:IO1_SEG               ;│
6736:    MOV     WORD PTR DRH_Ptr,BX         ;│ 保存设备驱动程序请求标题的指针值
6737:    MOV     WORD PTR DRH_Ptr+2,ES       ;│
6738:    POP     DS                          ;┘
6739:    DB      0EAH                        ;┐
6740:    DW      offset BlkDevIntrEntry      ;│ 转入 IO1 模块提供的块设备驱动程序的中断过
6741:    DW      BIO_SEG                     ;│ 程
6742:    ;JMP    FAR PTR 0070:0898H          ;┘
6743: IO2_INT2F_ISR6:                        ;处理"增加新的设备"的子功能,此时 DS:DI 指向
                                             ;BDPB
6744:    PUSH    ES
6745:    PUSH    DS
6746:    PUSH    DS                          ;┐ ES:DI 指向 BDPB
```

```
6747:       POP     ES                          ;┘
6748:       MOV     DS,CS:IO1_SEG               ;DS←IO1 模块在常规内存低端的段地址值(70H)
6749:       CALL    AppendNewBDPB               ;将新增加的块设备对应的 BDPB 插入到 BDPB 链
                                                ;尾
6750:       POP     DS
6751:       POP     ES
6752:       JMP     SHORT IO2_INT2F_ISR3
6753: IO2_INT2F_ISR7:                           ;处理"取 BDPB 链"的子功能
6754:       MOV     DS,CS:IO1_SEG               ;┐ DS:DI 指向 BDPB 链的头结点
6755:       LDS     DI,DWORD PTR BDPB_Head      ;┘
6756: IO2_INT2F_ISR8:
6757:       JMP     SHORT IO2_INT2F_ISR3
6758: IO2_INT2F_ISR9:
6759:       PUSH    DS
6760:       MOV     DS,CS:IO1_SEG
6761:       CMP     AL,5
6762:       JE      IO2_INT2F_ISR10
6763:       CMP     AL,6
6764:       JNE     IO2_INT2F_ISR11
6765:       TEST    DX,1
6766:       JNZ     IO2_INT2F_ISR11
6767:       AND     D_08D0,0
6768:       JMP     SHORT IO2_INT2F_ISR11
6769: IO2_INT2F_ISR10:
6770:       TEST    DX,1
6771:       JNZ     IO2_INT2F_ISR11
6772:       OR      D_08D0,1
6773:       MOV     WORD PTR DS:D_089E,BX
6774:       MOV     D_08A0,ES
6775:       MOV     BX,offset D_089C
6776:       PUSH    DS
6777:       POP     ES
6778: IO2_INT2F_ISR11:
6779:       POP     DS
6780:       JMP     SHORT IO2_INT2F_ISR3
6781: IO2_INT2F_ISR12:                          ;处理"访问高内存区(HMA)"的功能
6782:       CMP     AL,1                        ;┐ 若不是"取 HMA 自由(未被使用)内存区的起始
6783:       JNE     IO2_INT2F_ISR14             ;┘ 地址和大小",则跳转
6784:       PUSH    DS
6785:       CALL    Get_OFS_FreeHMA             ;取出 HMA 的自由内存区的起始地址偏移
6786:       MOV     BX,-1                       ;┐ ES=高内存区的段地址值
6787:       MOV     ES,BX                       ;┘
6788:       MOV     BX,DI                       ;┐
```

```
6789:    NOT     BX                          ;|
6790:    OR      BX,BX                       ;| BX=HMA 的自由内存的大小(字节数)
6791:    JZ      IO2_INT2F_ISR13             ;|
6792:    INC     BX                          ;|
6793: IO2_INT2F_ISR13:                       ;┘
6794:    POP     DS
6795:    JMP     SHORT IO2_INT2F_ISR8        ;转中断返回
6796: IO2_INT2F_ISR14:
6797:    CMP     AL,2                        ;┐ 若不是"分配 HMA 内存",则跳转
6798:    JNE     IO2_INT2F_ISR17             ;┘
6799:    PUSH    DS
6800:    MOV     DI,-1                       ;┐ ES=HMA 的段地址值
6801:    MOV     ES,DI                       ;┘
6802:    CALL    Get_OFS_FreeHMA             ;取出 HMA 的自由内存区的起始地址偏移
6803:    CMP     DI,-1                       ;┐ 若 HMA 的内存已全部被占用完,则跳转
6804:    JE      IO2_INT2F_ISR16             ;┘
6805:    NEG     DI                          ;DI=HMA 的自由内存大小(字节数)
6806:    CMP     BX,DI                       ;┐ 若有足够的自由内存供分配,则跳转
6807:    JBE     IO2_INT2F_ISR15             ;┘
6808:    MOV     DI,-1                       ;置"没有足够的 HMA 内存"的标志
6809:    JMP     SHORT IO2_INT2F_ISR16
6810: IO2_INT2F_ISR15:
6811:    MOV     DI,FreeMemOFSofHMA          ;┐
6812:    ADD     BX,0FH                      ;|
6813:    AND     BX,-10H                     ;|
6814:    ADD     FreeMemOFSofHMA,BX          ;| 修改 HMA 的自由内存区的起始地址偏移值
6815:    JNZ     IO2_INT2F_ISR16             ;|
6816:    MOV     FreeMemOFSofHMA,0FFFFH      ;|
6817: IO2_INT2F_ISR16:                       ;┘
6818:    POP     DS
6819:    JMP     IO2_INT2F_ISR8              ;转中断返回
6820: IO2_INT2F_ISR17:
6821:    JMP     IO2_INT2F_ISR8              ;转中断返回
6822: IO2_INT2F_ISR   ENDP
6823: ;=============================================================
6824: ;相对地址偏移:1413
6825: ;功能:取出 HMA 的自由内存区的起始地址偏移
6826: ;入口参数:无
6827: ;出口参数:DI=HMA 的自由内存区的起始地址偏移值
6828: ;=============================================================
6829: Get_OFS_FreeHMA   PROC   NEAR
6830:    MOV     DS,CS:IO1_SEG               ;┐ DI←HMA 的自由内存区的起始地址偏移
6831:    MOV     DI,FreeMemOFSofHMA          ;┘
```

```
6832:    CMP     DI,-1                    ;┐若起始地址偏移值有效,则直接返回
6833:    JNE     Get_OFS_FreeHMA_RET      ;┘
6834:    CMP     IO3_Flag,00              ;┐若IO3模块无效(此时DI=-1,表明此起始
6835:    JE      Get_OFS_FreeHMA_RET      ;┘地址偏移有效),则直接返回
6836:    CALL    DWORD PTR PreEntry7      ;调用由IO3模块定义的子程序,试图在HMA中安
                                          ;装IO2模块和MSDOS2模块,然后确定HMA的自
                                          ;由内存区的起始地址
6837:    MOV     DI,FreeMemOFSofHMA       ;DI←HMA的自由内存区的起始地址偏移
6838: Get_OFS_FreeHMA_RET:
6839:    RET
6840: Get_OFS_FreeHMA  ENDP
6841: ;=============================================
6842: ;相对地址偏移:1431
6843: ;功能:传送一个扇区的数据
6844: ;入口参数:DS:SI指向存放源数据的缓冲区;
6845: ;          ES:DI指向目的缓冲区
6846: ;出口参数:无
6847: ;=============================================
6848: TransferData  PROC  NEAR
6849:    CLD
6850:    PUSH    CX
6851:    MOV     CX,100H
6852:    CMP     SI,0FE00H
6853:    JA      TransferData_1
6854:    CMP     DI,0FE00H
6855:    JA      TransferData_1
6856:    REP     MOVSW
6857:    POP     CX
6858:    RET
6859: TransferData_1:
6860:    SHL     CX,1
6861:    REP     MOVSB
6862:    POP     CX
6863:    RET
6864: TransferData  ENDP
6865: ;=============================================
6866: ;相对地址偏移:144C
6867: ;功能:校正INT 13H的入口参数
6868: ;入口参数:CL=扇区号(其中高2位为圆柱面号的高2位)
6869: ;          CH=圆柱面号;
6870: ;          DL=物理驱动器号;
6871: ;          DH=磁头号
6872: ;出口参数:CL=扇区号(其中高2位为圆柱面号的高2位)
```

```
;          CH=圆柱面号;
;          DL=物理驱动器号;
;          DH=磁头号
;=============================================================
Verify_INT13H_Para  PROC  NEAR
    PUSH   AX
    PUSH   BX
    PUSH   ES
    PUSH   DI
    CALL   Seek_BDPB                    ;取出 DL 指定的物理驱动器号对应的 BDPB
    JB     Verify_INT13HP3              ;若 DL 指定的驱动器无效,则结束返回
    TEST   WORD PTR ES:[DI].BDPB_Attr_Status,1 ;┐ 若介质是可装卸的,则结束返回
    JZ     Verify_INT13HP3                     ;┘
    MOV    BX,ES:[DI].BDPB_SectorsPerTrack1    ;BX←每道扇区数
    MOV    AX, CX                       ;┐
    AND    AX, 003FH                    ;│ 若扇区号≤每道扇区数,则跳转
    CMP    AX, BX                       ;│
    JBE    Verify_INT13HP3              ;┘
    DIV    BL                           ;┐
    OR     AH, AH                       ;│ AL←磁头号的增量
    JNZ    Verify_INT13HP1              ;│
    MOV    AH, BL                       ;│ AH←新的道内扇区号
    DEC    AL                           ;┘
Verify_INT13HP1:
    AND    CL, 0C0H                     ;┐ 形成新的道内扇区号→CL
    OR     CL, AH                       ;┘
    XOR    AH, AH                       ;┐
    INC    AX                           ;│ 形成新的磁头号→AX
    ADD    AL,DH                        ;│
    ADC    AH,0                         ;┘
    CMP    AX,ES:[DI].BDPB_NumberOfHead1 ;┐ 若磁头号正确,则跳转
    JBE    Verify_INT13HP4                ;┘
    PUSH   DX                           ;┐
    XOR    DX, DX                       ;│
    MOV    BX,ES:[DI].BDPB_NumberOfHead1 ;│
    DIV    BX                           ;│
    OR     DX, DX                       ;│ 计算圆柱面号的增量→AX
    JNZ    Verify_INT13HP2              ;│ 磁头号→DL
    MOV    DX, BX                       ;│
    OR     AX, AX                       ;│
    JZ     Verify_INT13HP2              ;│
    DEC    AX                           ;┘
```

```
6915: Verify_INT13HP2:
6916:     MOV     BH,DL                       ;┐
6917:     POP     DX                          ;| DH←磁头号
6918:     DEC     BH                          ;|
6919:     MOV     DH,BH                       ;┘
6920:     MOV     BH,CL                       ;┐ BH←道内扇区号
6921:     AND     BH,3FH                      ;┘
6922:     MOV     BL,6                        ;┐
6923:     XCHG    CL,BL                       ;| BL←圆柱面号的高二位值
6924:     SHR     BL,CL                       ;┘
6925:     ADD     CH,AL                       ;┐ BL:CH←圆柱面号
6926:     ADC     BL,AH                       ;┘
6927:     SHL     BL,CL                       ;┐
6928:     XCHG    BL,CL                       ;| 形成新的 CL 值
6929:     OR      CL,BH                       ;┘
6930: Verify_INT13HP3:
6931:     CLC
6932:     POP     DI
6933:     POP     ES
6934:     POP     BX
6935:     POP     AX
6936:     RET
6937: Verify_INT13HP4:
6938:     MOV     DH,AL                       ;┐ DH←磁头号
6939:     DEC     DH                          ;┘
6940:     JMP     SHORT Verify_INT13HP3
6941: Verify_INT13H_Para  ENDP
6942: ;=============================================================
6943: ;相对地址偏移:14C3
6944: ;功能:查找与 DL 指定的物理驱动器相对应的 BDPB
6945: ;入口参数:DL=物理驱动器号
6946: ;出口参数:CF=0,成功,ES:DI 指向对应的 BDPB;CF=1,失败
6947: ;=============================================================
6948: Seek_BDPB  PROC  NEAR
6949:     LES     DI,DWORD PTR BDPB_Head      ;ES:DI 指向第一个 BDPB
6950: Seek_BDPB_1:
6951:     CMP     ES:[DI].BDPB_DriveNumber,DL ;┐ 若物理驱动器号匹配(即找到与指定驱动器相
6952:     JE      Seek_BDPB_RET               ;┘ 关的 BDPB),则跳转
6953:     LES     DI,DWORD PTR ES:[DI].BDPB_NextPtr_OFS    ;ES:DI 指向下一个 BDPB
6954:     CMP     DI,-1                       ;┐ 若当前结点不是尾结点,则跳转
6955:     JNE     Seek_BDPB_1                 ;┘
6956:     STC                                 ;CF←1(未找到对应的 BDPB)
6957: Seek_BDPB_RET:
```

```
6958:     RET
6959: Seek_BDPB   ENDP
6960: ;=============================================================
6961: ;相对地址偏移:14D7
6962: ;功能:链接原来的 INT 13H 中断服务程序
6963: ;出口参数:DH=磁头号
6964: ;入口参数:无
6965: ;=============================================================
6966: Link_Old_INT13H   PROC   NEAR
6967:     DB      8AH,96H                    ;┐
6968:     DW      8                          ;| DL←物理驱动器号
6969:     ;MOV    DL,[BP+0008H]              ;┘
6970:     XOR     AH,AH                      ;
6971:     OR      AL,AL                      ;┐若盘操作的扇区数为 0,则直接返回
6972:     JZ      Link_Old13H_RET            ;┘
6973:     DB      8AH,0A6H                   ;┐
6974:     DW      3                          ;| AH←功能码
6975:     ;MOV    AH,[BP+0003H]              ;┘
6976:     DB      0FFH,0B6H                  ;┐
6977:     DW      10H                        ;| 恢复 INT 13H 中断请求者的状态寄存器值
6978:     ;PUSH   WORD PTR [BP+0010H]        ;|
6979:     POPF                               ;┘
6980:     DB      9AH                        ;┐
6981:     DW      offset IO1_INT13H_ISR      ;| 调用由 IO1 模块定义的 INT 13H 中断处理的链
6982:     DW      BIO_SEG                    ;| 接程序,以便完成指定的工作
6983:     ;CALL   0070:0797H                 ;┘
6984:     PUSHF                              ;┐
6985:     DB      8FH,86H                    ;| 设置中断返回的状态寄存器值
6986:     DW      10H                        ;|
6987:     ;POP    WORD PTR [BP+0010H]        ;┘
6988: Link_Old13H_RET:
6989:     RET
6990: Link_Old_INT13H   ENDP
6991: ;相对地址偏移:14F5
6992: ;BIOS 通讯区的"软盘驱动器(0 号和 1 号)介质状态"的起始单元地址(软盘驱动器 0 使用 40:90H
      单元;软盘驱动器 1 使用 40:91H 单元)
6993: AddrForMediaStatus   DW   90H
6994:     DW      40H
6995: ;=============================================================
6996: ;相对地址偏移:14F9
6997: ;功能:由 IO2 模块定义的 INT 13H 中断接管程序
6998: ;=============================================================
6999: INT13H_ILR   PROC   FAR
```

```
7000: INT13H_ILR_1:                          ;处理"格式化磁道柱面"的功能
7001:     CMP     ChangeLine,0               ;┐若软盘驱动器不支持 Change Line,则跳转
7002:     JE      INT13H_ILR_5               ;┘
7003:     PUSH    BX                         ;┐
7004:     MOV     BX,140H                    ;│设置 BDPB 的"重建 BPB 参数块标志位、磁道格
7005:     CALL    Set_BDPB_DevInfo           ;│式化标志位"
7006:     POP     BX                         ;┘
7007:     JMP     SHORT INT13H_ILR_5
7008: INT13H_ILR_2:
7009:     TEST    DL,DL                      ;┐若是对硬盘进行操作,则跳转
7010:     JS      INT13H_ILR_6               ;┘
7011: ;设置 BIOS 通讯区的"软盘驱动器介质状态"的单元值
7012:     PUSH    AX                         ;┐
7013:     PUSH    CX                         ;│
7014:     MOV     CL,DL                      ;│
7015:     MOV     AL,1                       ;│若 DL 指定的软盘驱动器不作为兼容的 3.5 英
7016:     SHL     AL,CL                      ;│寸软盘驱动器使用,则跳转
7017:     TEST    AL,CompatibleFDiskFlag     ;│
7018:     POP     CX                         ;│
7019:     POP     AX                         ;│
7020:     JZ      INT13H_ILR_6               ;┘
7021:     PUSH    BX                                 ;┐
7022:     PUSH    ES                                 ;│
7023:     LES     BX,DWORD PTR CS:AddrForMediaStatus ;│设置 BIOS 通讯区的"软盘驱动器介质状
7024:     ADD     BL,DL                              ;│态"的单元值(93H 表示:数据传输速率
7025:     ADC     BH,0                               ;│为 250KB/s、不需要双步进、已知介质、在
7026:     MOV     BYTE PTR ES:[BX],93H               ;│360KB 软盘驱动器中使用 360KB 的盘
7027:     POP     ES                                 ;│片)
7028:     POP     BX                                 ;┘
7029:     JMP     SHORT INT13H_ILR_6
7030: INT13H_ILR_3:
7031:     CMP     AXValue,8
7032:     JE      INT13H_ILR_4
7033:     CMP     AXValue,15H
7034:     JNE     INT13H_ILR_7
7035: INT13H_ILR_4:
7036:     PUSH    AX
7037:     MOV     AH,1                       ;┐
7038:     DB      9AH                        ;│
7039:     DW      offset IO1_INT13H_ISR      ;│取磁盘系统的状态
7040:     DW      BIO_SEG                    ;│
7041:     ;CALL   FAR PTR 0070:0797H         ;┘
7042:     POP     AX
```

```
7043:    JMP     SHORT INT13H_ILR_7
7044: ;=============================================================
7045: ;相对地址偏移:154B
7046: ;功能:由 IO2 模块定义的 INT 13H 中断接管程序的入口,它由 IO1 模块的 INT13H_ISR 通过远调用
          而被执行
7047: ;=============================================================
7048: IO2_INT13H_ISR  LABEL  FAR
7049:    PUSH    DS
7050:    MOV     DS,CS:IO1_SEG          ;DS←IO1 模块在常规内存低端的段地址值(70H)
7051:    MOV     AXValue,AX             ;保存 INT 13H 的入口参数 AX 寄存器值
7052:    CMP     AH,5                   ;┐若"格式化磁道柱面",则跳转
7053:    JE      INT13H_ILR_1           ;┘
7054: INT13H_ILR_5:
7055:    CMP     CompatibleFDiskFlag,0  ;┐若由 drivparm 系统配置命令指定了兼容的 3.5
7056:    JNE     INT13H_ILR_2           ;┘英寸软盘驱动器,则跳转
7057: INT13H_ILR_6:
7058:    DB      9AH                    ;┐
7059:    DW      offset IO1_INT13H_ISR  ;├调用由 IO1 模块定义的 INT 13H 中断处理的链
7060:    DW      BIO_SEG                ;├接程序,请求完成指定的工作
7061:    ;CALL   FAR PTR 0070:0797H     ;┘
7062:    PUSHF                          ;保护 INT 13H 中断处理返回的状态
7063:    CMP     ModelByte_Init,0FAH    ;┐若机器是"IBM PS/2 Model 30",ROM BIOS 为
7064:    JE      INT13H_ILR_3           ;┘"1986 年 9 月 2 号",则跳转
7065: INT13H_ILR_7:
7066:    POPF                           ;恢复 INT 13H 中断处理返回的状态
7067:    JC      INT13H_ILR_12          ;若 INT 13H 中断请求失败,则跳转
7068: INT13H_ILR_8:
7069:    POP     DS
7070:    RETF    2                      ;中断返回(废弃先前的标志寄存器值)
7071: INT13H_ILR_9:
7072:    JNC     INT13H_ILR_8           ;若指定的磁道操作成功,则跳转
7073: INT13H_ILR_10:
7074:    CMP     AH,6                   ;┐若不是"未装入软盘片"错,则跳转
7075:    JNE     INT13H_ILR_11          ;┘
7076:    OR      DL,DL                  ;┐若是对硬盘进行操作,则跳转
7077:    JS      INT13H_ILR_11          ;┘
7078:    CMP     ChangeLine,0           ;┐若软盘驱动器不支持 Change Line,则跳转
7079:    JE      INT13H_ILR_11          ;┘
7080:    PUSH    BX                     ;┐
7081:    MOV     BX,40H                 ;├设置 BDPB 的"重建 BPB 参数块标志位"
7082:    CALL    Set_BDPB_DevInfo       ;┘
7083:    POP     BX                     ;┘
7084: INT13H_ILR_11:
```

```
7085:    STC                                    ;CF←1(失败)
7086:    JMP     SHORT INT13H_ILR_8
7087: INT13H_ILR_12:
7088:    CMP     AH,9                           ;┐ 若是"DMA 段边界超越 64KB"错,则跳转
7089:    JE      INT13H_ILR_20                  ;┘
7090: INT13H_ILR_13:
7091:    CMP     AH,11H                         ;┐ 若不是"ECC 纠正数据错",则跳转
7092:    JNE     INT13H_ILR_10                  ;┘
7093:    CMP     ModifyFlag,1                   ;┐ 若更换过 INT 1EH 中断向量,则跳转
7094:    JE      INT13H_ILR_10                  ;┘
7095:    CMP     BYTE PTR AXValue+1,2           ;┐ 若不是"读扇区的操作",则跳转
7096:    JNE     INT13H_ILR_10                  ;┘
7097:    XOR     AH,AH                          ;┐
7098:    DB      9AH                            ;│
7099:    DW      offset IO1_INT13H_ISR          ;│ 复位软盘和硬盘控制器
7100:    DW      BIO_SEG                        ;│
7101:    ;CALL   FAR PTR 0070:0797H             ;┘
7102:    MOV     AX,AXValue                     ;┐
7103:    XOR     AH,AH                          ;│ 若只读 1 个扇区,则跳转
7104:    CMP     AL,1                           ;│
7105:    JE      INT13H_ILR_8                   ;┘
7106:    PUSH    BX
7107:    PUSH    CX
7108:    PUSH    DX
7109:    MOV     BYTE PTR UncompletedSectors,AL ;设置未读完的扇区数
7110: INT13H_ILR_14:
7111:    MOV     AX,201H                        ;┐
7112:    CALL    Verify_INT13H_Para             ;│
7113:    DB      9AH                            ;│ 读取一个扇区
7114:    DW      offset IO1_INT13H_ISR          ;│
7115:    DW      BIO_SEG                        ;│
7116:    ;CALL   FAR PTR 0070:0797H             ;┘
7117:    JNC     INT13H_ILR_15                  ;若读成功,则跳转
7118:    CMP     AH,9                           ;┐ 若是"DMA 段边界超越 64KB 错",则跳转
7119:    JE      INT13H_ILR_18                  ;┘
7120:    CMP     AH,11H                         ;┐ 若不是"ECC 纠正数据错",则跳转
7121:    JNE     INT13H_ILR_16                  ;┘
7122:    MOV     AH,0
7123:    XOR     AX,AX
7124: INT13H_ILR_15:
7125:    DEC     BYTE PTR UncompletedSectors    ;┐ 若指定的扇区数全部被读完,则跳转
7126:    JZ      INT13H_ILR_17                  ;┘
7127:    INC     CL                             ;扇区号加 1
```

```
7128:    INC     BH                      ;┐ 修改传送缓冲区的地址偏移值(BX=BX+
7129:    INC     BH                      ;┘ 200H)
7130:    JMP     SHORT INT13H_ILR_14     ;转去继续读取后续的扇区
7131: INT13H_ILR_16:
7132:    STC                             ;CF←1(失败)
7133: INT13H_ILR_17:
7134:    POP     DX
7135:    POP     CX
7136:    POP     BX
7137:    JMP     SHORT INT13H_ILR_9
7138: INT13H_ILR_18:                     ;┐
7139:    PUSH    ES                      ;|
7140:    PUSH    BX                      ;|
7141:    MOV     BX,offset DiskBuffer    ;|
7142:    PUSH    DS                      ;|
7143:    POP     ES                      ;|
7144:    MOV     AX,201H                 ;|
7145:    DB      9AH                     ;|
7146:    DW      offset IO1_INT13H_ISR   ;|
7147:    DW      BIO_SEG                 ;|
7148:    ;CALL   FAR PTR 0070:0797H      ;|
7149:    POP     BX                      ;|
7150:    POP     ES                      ;| 通过内部缓冲区(位于 IO1 模块中)读取当前扇
7151:    JNC     INT13H_ILR_19           ;| 区的数据
7152:    CMP     AH,11H                  ;|
7153:    JNE     INT13H_ILR_16           ;|
7154: INT13H_ILR_19:                     ;|
7155:    PUSH    SI                      ;|
7156:    PUSH    DI                      ;|
7157:    MOV     DI,BX                   ;|
7158:    MOV     SI,offset DiskBuffer    ;|
7159:    CALL    TransferData            ;|
7160:    POP     DI                      ;|
7161:    POP     SI                      ;|
7162:    JMP     SHORT INT13H_ILR_15     ;┘
7163: INT13H_ILR_20:
7164:    MOV     AX,AXValue              ;AX←INT 13H 的入口参数 AX 寄存器值
7165:    STI
7166:    CMP     AH,2                    ;┐ 若功能码<2,则跳转
7167:    JB      INT13H_ILR_22           ;┘
7168:    CMP     AH,4                    ;┐ 若是"验证扇区",则跳转
7169:    JE      INT13H_ILR_23           ;┘
7170:    CMP     AH,5                    ;┐ 若是"格式化磁道柱面",则跳转
```

```
7171:    JE      INT13H_ILR_25                ;┘
7172:    JA      INT13H_ILR_22                ;若功能码>5,则跳转
7173: ;对应于"读扇区、写扇区"
7174:    PUSH    DX
7175:    PUSH    CX
7176:    PUSH    BX
7177:    PUSH    AX
7178:    PUSH    BP
7179:    MOV     BP,SP
7180:    MOV     DX,ES                        ;┐
7181:    SHL     DX,1                         ;|
7182:    SHL     DX,1                         ;| 若传送缓冲区的地址不会因读/写一个扇区的
7183:    SHL     DX,1                         ;| 数据而发生 DMA 边界错,则跳转
7184:    SHL     DX,1                         ;|
7185:    ADD     DX,BX                        ;|
7186:    ADD     DX,1FFH                      ;|
7187:    JNC     INT13H_ILR_21                ;┘
7188:    JMP     INT13H_ILR_31
7189: INT13H_ILR_21:
7190:    SHR     DH,1                         ;┐
7191:    MOV     AH,80H                       ;| 当读/写 AL 个扇区的内容时,传送缓冲区的地
7192:    SUB     AH,DH                        ;| 址会发生 DMA 边界错,则跳转
7193:    CMP     AH,AL                        ;|
7194:    JB      INT13H_ILR_26                ;┘
7195:    MOV     DH,[BP+9]                    ;DH←磁头号
7196:    CALL    Link_Old_INT13H              ;链接原来的 INT 13H 中断服务程序,请求完成指
                                              ;定的功能
7197:    JMP     INT13H_ILR_35
7198: INT13H_ILR_22:
7199:    MOV     AH,9                         ;AH←INT 13H 中断返回的错误("DMA 边界错")
7200:    STC                                  ;CF←1(失败)
7201:    JMP     INT13H_ILR_8
7202: INT13H_ILR_23:                          ;验证扇区
7203:    PUSH    ES
7204:    PUSH    BX
7205:    PUSH    DS
7206:    POP     ES
7207: INT13H_ILR_24:
7208:    MOV     BX,offset DiskBuffer         ;ES:BX 指向由 IO1 模块定义的内部缓冲区
7209:    DB      9AH                          ;┐
7210:    DW      offset IO1_INT13H_ISR        ;| 调用由 IO1 模块定义的 INT 13H 中断处理的链
7211:    DW      BIO_SEG                      ;| 接程序,请求完成"验证扇区或格式化磁道柱
7212:    ;CALL   0070:0797H                   ;┘ 面"
```

```
7213:    POP     BX
7214:    POP     ES
7215:    JMP     INT13H_ILR_9
7216: INT13H_ILR_25:                          ;格式化磁道柱面
7217:    PUSH    ES
7218:    PUSH    BX
7219:    PUSH    SI                           ;┐
7220:    PUSH    DI                           ;|
7221:    PUSH    DS                           ;|
7222:    PUSH    ES                           ;|
7223:    PUSH    DS                           ;|
7224:    POP     ES                           ;| 传送缓冲区的数据→由 IO1 模块定义的内部缓
7225:    POP     DS                           ;| 冲区
7226:    MOV     SI,BX                        ;|
7227:    MOV     DI,offset DiskBuffer         ;|
7228:    CALL    TransferData                 ;|
7229:    POP     DS                           ;|
7230:    POP     DI                           ;┘
7231:    POP     SI
7232:    JMP     SHORT INT13H_ILR_24
7233: INT13H_ILR_26:
7234:    MOV     DX,[BP+8]                    ;DH←磁头号;DL←物理驱动器号
7235:    PUSH    CX
7236:    PUSH    ES
7237:    PUSH    DI
7238:    CALL    Seek_BDPB                    ;查找与 DL 指定的物理驱动器号相对应的 BDPB
7239:    MOV     CX,ES:[DI].BDPB_SectorsPerTrack1    ;CX←每道扇区数
7240:    TEST    WORD PTR ES:[DI].BDPB_Attr_Status,1
7241:    POP     DI
7242:    POP     ES
7243:    MOV     AL,AH                        ;AL←当传送缓冲区的地址不发生 DMA 边界错
                                              ;时,传送缓冲区允许存取的扇区数
7244:    JZ      INT13H_ILR_27                ;若是存取软盘,则跳转
7245:    MOV     AH,3FH                       ;┐ AH←道内扇区号表示方式允许扇区数
7246:    SUB     AH,CL                        ;┘
7247: INT13H_ILR_27:
7248:    POP     CX
7249: INT13H_ILR_28:
7250:    CMP     AH,AL                        ;┐
7251:    JAE     INT13H_ILR_29                ;|
7252:    PUSH    AX                           ;|
7253:    MOV     AL,AH                        ;| 保存当前允许存取的扇区数(AH=AL)
7254:    JMP     SHORT INT13H_ILR_30          ;|
```

```
7255: INT13H_ILR_29:                    ;|
7256:     MOV     AH,AL                 ;|
7257:     PUSH    AX                    ;┘
7258: INT13H_ILR_30:
7259:     CALL    Link_Old_INT13H       ;本次只读/写 AL 个扇区
7260:     JC      INT13H_ILR_35         ;若读/写操作失败,则跳转
7261:     POP     AX
7262:     SUB     [BP+2],AH             ;修改未读/写的扇区数
7263:     ADD     CL,AH                 ;修改道内扇区号
7264:     ADD     BH,AH                 ;┐修改传送缓冲区的地址偏移值
7265:     ADD     BH,AH                 ;┘
7266:     CMP     AH,AL
7267:     JE      INT13H_ILR_32
7268:     SUB     AL,AH
7269:     CALL    Verify_INT13H_Para    ;重新校正 INT 13H 的入口参数
7270:     JMP     SHORT INT13H_ILR_28
7271: INT13H_ILR_31:
7272:     MOV     DH,[BP+9]             ;DH←磁头号
7273: INT13H_ILR_32:                    ;处理因"会发生 DMA 边界错"而未完成的扇区
7274:     PUSH    BX
7275:     MOV     AH,[BP+3]             ;AH←功能码
7276:     CMP     AH,3                  ;┐若不是"写扇区"(即是"读扇区"),则跳转
7277:     JNE     INT13H_ILR_33         ;┘
7278: ;处理因"会发生 DMA 边界错"而未写入的扇区
7279:     PUSH    ES
7280:     PUSH    DS                    ;┐
7281:     PUSH    SI                    ;|
7282:     PUSH    DI                    ;|
7283:     PUSH    DS                    ;|
7284:     PUSH    ES                    ;|
7285:     POP     DS                    ;|
7286:     POP     ES                    ;| 将欲写的数据(一个扇区大小)从传送缓冲区中
7287:     MOV     DI,offset DiskBuffer  ;| 复制到由 IO1 模块定义的内部缓冲区
7288:     PUSH    DI                    ;|
7289:     MOV     SI,BX                 ;|
7290:     CALL    TransferData          ;|
7291:     POP     BX                    ;|
7292:     POP     DI                    ;|
7293:     POP     SI                    ;|
7294:     POP     DS                    ;┘
7295:     MOV     AL,1                  ;┐
7296:     MOV     DL,[BP+8]             ;| 将一个扇区的数据写到指定的介质上
7297:     CALL    Verify_INT13H_Para    ;|
```

```
7298:    CALL    Link_Old_INT13H                ;┘
7299:    POP     ES
7300:    JB      INT13H_ILR_35                  ;若写操作失败,则跳转
7301:    JMP     SHORT INT13H_ILR_34
7302: INT13H_ILR_33:                            ;处理因"会发生 DMA 边界错"而未读取的扇区
7303:    PUSH    ES                             ;┐
7304:    PUSH    BX                             ;│
7305:    PUSH    DS                             ;│
7306:    POP     ES                             ;│
7307:    MOV     BX,offset DiskBuffer           ;│ 将一个指定的扇区数据读到由 IO1 模块定义的
7308:    MOV     AL,1                           ;│ 内部缓冲区中
7309:    MOV     DL,[BP+8]                      ;│
7310:    CALL    Verify_INT13H_Para             ;│
7311:    CALL    Link_Old_INT13H                ;│
7312:    POP     BX                             ;│
7313:    POP     ES                             ;┘
7314:    JC      INT13H_ILR_35                  ;若读操作失败,则跳转
7315:    PUSH    SI                             ;┐
7316:    PUSH    DI                             ;│
7317:    MOV     DI,BX                          ;│ 将已读取的一个扇区数据从内部缓冲区中复制
7318:    MOV     SI,offset DiskBuffer           ;│ 到传送缓冲区
7319:    CALL    TransferData                   ;│
7320:    POP     DI                             ;│
7321:    POP     SI                             ;┘
7322: INT13H_ILR_34:                            ;修改传送缓冲区的地址偏移,并判定欲读/写的扇
                                                ;区数是否都完成
7323:    POP     BX
7324:    ADD     BH,2                           ;修改传送缓冲区的地址偏移值
7325:    INC     CX                             ;修改道内扇区号
7326:    MOV     AL,[BP+2]                      ;┐
7327:    CLC                                    ;│ 若欲读/写的扇区已全部完成,则跳转
7328:    DEC     AL                             ;│
7329:    JZ      INT13H_ILR_35                  ;┘
7330:    MOV     DL,[BP+8]                      ;┐
7331:    CALL    Verify_INT13H_Para             ;│ 读/写未完成的扇区
7332:    CALL    Link_Old_INT13H                ;┘
7333: INT13H_ILR_35:
7334:    MOV     SP,BP
7335:    POP     BP
7336:    POP     BX
7337:    POP     BX
7338:    POP     CX
7339:    POP     DX
```

```
7340:    JC      INT13H_ILR_36
7341:    JMP     INT13H_ILR_8
7342: INT13H_ILR_36:
7343:    JMP     INT13H_ILR_13
7344: INT13H_ILR  ENDP
7345:    DB      0
7346: ;=========================================================
7347: ;相对地址偏移:1742
7348: ;功能:块设备驱动程序的初始化处理程序(块设备的命令码=0)
7349: ;=========================================================
7350: Disk_Init  PROC  NEAR
7351:    MOV     AH,BYTE PTR TotalOfBlockDev    ;AH←块设备的部件数
7352:    MOV     DI,offset BPB_Ptr_Array        ;DS:DI 指向 BPB 参数块指针数组
7353:    PUSH    DS
7354:    POP     ES
7355:    JMP     Build_BPB3                     ;转去设置设备驱动程序返回的参数值
7356: Disk_Init  ENDP
7357: ;=========================================================
7358: ;相对地址偏移:174E
7359: ;功能:将 ES:DI 指定的 BDPB 插入到 BDPB 链尾
7360: ;入口参数:ES:DI 指向新增加的驱动器对应的 BDPB
7361: ;出口参数:无
7362: ;=========================================================
7363: AppendNewBDPB  PROC  NEAR
7364:    PUSH    DS
7365:    MOV     SI,offset BDPB_Head            ;DS:SI 指向第一个 BDPB
7366: AppendNewBDPB_1:
7367:    LDS     SI,DWORD PTR [SI]              ;DS:SI 指向当前的 BDPB
7368:    MOV     AL,ES:[DI].BDPB_DriveNumber    ;┐
7369:    CMP     DS:[SI].BDPB_DriveNumber,AL    ;├若物理驱动器名不同,则跳转
7370:    JNE     AppendNewBDPB_2                ;┘
7371:    MOV     BL,10H                                 ;┐
7372:    OR      BYTE PTR ES:[DI].BDPB_Attr_Status,BL   ;├设置 BDPB 的"驱动器标志位"
7373:    OR      BYTE PTR [SI].BDPB_Attr_Status,BL      ;┘
7374:    AND     BYTE PTR ES:[DI].BDPB_Attr_Status,0DFH    ;清当前 BDPB 的"活动状态位"
7375:    MOV     BL,BYTE PTR [SI].BDPB_Attr_Status      ;┐依据先前存在的同名物理驱动器对应
7376:    AND     BL,02                                  ;├的 BDPB,修改当前指定的 BDPB 的
7377:    OR      BYTE PTR ES:[DI].BDPB_Attr_Status,BL   ;┘"Change Line 功能位"
7378: AppendNewBDPB_2:
7379:    CMP     WORD PTR [SI],0FFFFH           ;┐若 BDPB 链未完,则跳转
7380:    JNE     AppendNewBDPB_1                ;┘
7381:    MOV     [SI].BDPB_NextPtr_SEG,ES       ;┐将当前的 BDPB 插入到 BDPB 链尾
7382:    MOV     [SI],DI                        ;┘
```

```
7383:    MOV     WORD PTR ES:[DI],0FFFFH      ;设置 BDPB 链尾结点的指针域
7384:    POP     DS
7385:    RET
7386: AppendNewBDPB  ENDP
7387: ;=============================================================
7388: ;相对地址偏移:1786
7389: ;功能:显示在指定的逻辑驱动器中插入盘片的提示信息
7390: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7391: ;出口参数:无
7392: ;=============================================================
7393: Disp_Insert_FDisk  PROC  NEAR
7394:    TEST    D_08D0,1
7395:    JZ      Disp_IFDsk1
7396:    DB      9AH
7397:    DW      offset SUB_08D1
7398:    DW      BIO_SEG
7399:    ;CALL   0070:08D1H
7400: Disp_IFDsk1:
7401:    PUSH    CX
7402:    PUSH    DX
7403:    MOV     DL,ES:[DI].BDPB_LogicalNumber ;DL←逻辑驱动器号
7404:    MOV     DH,DL
7405:    XOR     DH,1
7406:    SUB     CX,CX
7407:    MOV     AX,4A00H
7408:    INT     2FH
7409:    INC     CX
7410:    JZ      Disp_IFDsk3
7411:    ADD     DL,'A'                       ;┐设置逻辑驱动器名符
7412:    MOV     BYTE PTR CS:InsertFDiskSym,DL ;┘
7413:    MOV     SI,offset Str_InsertFDisk    ;CS:SI 指向提示插入相应盘片的提示信息
7414:    PUSH    BX
7415:    LODS    BYTE PTR CS:[SI]             ;┐
7416: Disp_IFDsk2:                            ;│
7417:    INT     29H                          ;│输出"在指定的逻辑驱动器中插入相应盘片的
7418:    LODS    BYTE PTR CS:[SI]             ;│提示信息"
7419:    OR      AL,AL                        ;│
7420:    JNZ     Disp_IFDsk2                  ;┘
7421:    CALL    CON_Inp_Flush                ;清键盘输入缓冲区
7422:    XOR     AH,AH                        ;┐等待键盘输入
7423:    INT     16H                          ;┘
7424:    POP     BX
7425: Disp_IFDsk3:
```

```
7426:     POP     DX
7427:     POP     CX
7428:     RET
7429: Disp_Insert_FDisk   ENDP
7430: ;相对地址偏移:17C8
7431: Str_InsertFDisk   DB   0DH, 0AH, ' Insert diskette for drive '
7432: InsertFDiskSym   DB   ' A: and press any key when ready' , 0DH, 0AH, 0AH, 0
7433: ;=============================================================
7434: ;相对地址偏移:1807
7435: ;功能:测试设备打开计数器值是否为零
7436: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7437: ;出口参数:ZF=1,为零;ZF=0,不为零
7438: ;=============================================================
7439: Test_DevOpCount   PROC   NEAR
7440:     CMP     WORD PTR ES:[DI].BDPB_DeviceOpen,0;判定设备打开计数器值为零否?
7441:     RET
7442: Test_DevOpCount   ENDP
7443: ;=============================================================
7444: ;相对地址偏移:180D
7445: ;功能:检查盘介质更换否
7446: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7447: ;出口参数:CF=0,成功,SI=介质更换码(0:不知介质更换否;1:介质未被更换;-1:介质已被更
          换);CF=1,失败,AL=设备驱动程序返回的错误码
7448: ;=============================================================
7449: ChkMediaRemove   PROC   NEAR
7450:     CALL MultiDriveName                    ;处理一个物理软盘驱动器具有多个逻辑驱动器名
                                                 ;的情况,确保对指定的介质进行操作
7451:     XOR     SI, SI                         ;SI←0(未知介质更换否)
7452:     CALL    Test_ChangeLineStatus          ;┐若驱动器不支持 Change Line,则直接返回
7453:     JZ      ChkMediaRemove_RET             ;┘
7454:     CALL    Test_ModifyBPBFlag             ;┐若需要重建 Build BPB 参数块,则跳转
7455:     JNZ     ChkMediaRemove1                ;┘
7456:     PUSH    AX                             ;┐
7457:     PUSH    DX                             ;|
7458:     MOV     DL, ES:[DI].BDPB_DriveNumber   ;| 测试 Change Line 状态
7459:     MOV     AH, 16H                        ;|
7460:     INT 13H                                ;|
7461:     POP     DX                             ;|
7462:     POP     AX                             ;┘
7463:     JB      ChkMediaRemove1                ;若 ROM BIOS 不支持 Change Line 检查,则跳转
7464:     MOV     SI,1                           ;SI←1(介质未被更换)
7465:     MOV     BL, Drive_Number               ;┐
7466:     CMP     ES:[DI].BDPB_DriveNumber,BL    ;| 若当前操作的驱动器未更换,则直接返回
```

```
7467:    JZ      ChkMediaRemove_RET          ;┘
7468:    PUSH    AX                          ;┐
7469:    PUSH    CX                          ;│
7470:    PUSH    DX                          ;│根据同一驱动器最近两次相邻操作的时间间隔
7471:    CALL    Set_MediaChangedCode        ;│是否小于2秒来设置介质更换码
7472:    POP     DX                          ;│
7473:    POP     CX                          ;│
7474:    POP     AX                          ;┘
7475:    OR      SI, SI                      ;┐若返回的是"未知介质更换否",则跳转
7476:    JZ      ChkMediaRemove1             ;┘
7477:    XOR     SI, SI                      ;SI←0(未知介质更换否)
7478: ChkMediaRemove_RET:
7479:    RET
7480: ChkMediaRemove1:
7481:    CALL    Modify_BuildBPB             ;重建 Build BPB 参数块
7482:    JB      ChkMediaRemove_RET          ;若重建 Build BPB 参数块失败,则直接返回
7483:    CALL    ChkWithVolume               ;检查卷系列号或卷标的一致性
7484:    JNB     ChkMediaRemove_RET          ;若读卷标时盘操作成功,则直接返回
7485:    JMP     Set_CodeOfDriver            ;根据 INT 13H 返回的错误码设置设备驱动程序返
                                             ;回的错误码
7486: ChkMediaRemove  ENDP
7487: ;=============================================================
7488: ;相对地址偏移:1854
7489: ;功能:试图重建 BPB 参数块,并测试驱动器中的介质是否已被更换
7490: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7491: ;出口参数:CF=0,介质未更换;CF=1,介质已被更换
7492: ;=============================================================
7493: Rebuild_BPB  PROC  NEAR
7494:    CALL    Test_DevOpCount             ;┐若"设备打开计数器"值为0,则直接返回
7495:    JZ      Rebuild_BPB_RET             ;┘
7496:    CALL    Test_ModifyBPBFlag          ;┐若不需要重建 BPB 参数块,则直接返回
7497:    JZ      Rebuild_BPB_RET             ;┘
7498:    CALL    Modify_BuildBPB             ;┐若重建 BPB 参数块出错,则跳转
7499:    JC      Rebuild_BPB2                ;┘
7500:    CALL    ChkWithVolume               ;根据卷系列号或卷标判断介质是否被更换
7501:    JC      Rebuild_BPB1                ;若读卷标时磁盘操作出错,则跳转
7502:    OR      SI,SI                       ;┐若介质未更换,则直接返回
7503:    JNS     Rebuild_BPB_RET             ;┘
7504:    CALL    Set_ReturnPara              ;设置设备驱动程序的返回参数(先前存取介质的
                                             ;卷标 ID 指针值)和 INT 13H 返回的错误码
7505: Rebuild_BPB1:
7506:    CALL    Set_CodeOfDriver            ;确定设备驱动程序返回的错误码
7507: Rebuild_BPB2:
```

```
7508:    STC
7509:    POP     SI
7510: Rebuild_BPB_RET:
7511:    RET
7512: Rebuild_BPB  ENDP
7513: ;=============================================================
7514: ;相对地址偏移:1875
7515: ;功能:根据介质描述符和卷系列号(或卷标名)来判定介质是否被更换
7516: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7517: ;出口参数:CF=0,SI=0,未知介质更换否;SI=-1,介质已被更换
7518: ; CF=1,读卷标时盘操作出错
7519: ;=============================================================
7520: ChkWithMediaDescript  PROC  NEAR
7521:    CALL    Test_MediaDescript              ;根据介质描述符来检查介质是否被更换
7522:    OR      SI,SI                           ;┐若介质被更换,则跳转
7523:    JS      ChkWith_2                       ;┘
7524: ChkWithVolume  LABEL  NEAR                 ;依据卷系列号或卷标名判定介质是否被更换
7525:    CMP     BYTE PTR DiskBuffer+26H,29H     ;┐若引导记录中存放扩充 BPB 参数块,则跳转
7526:    JZ      ChkWith_3                       ;┘
7527:    CALL    Test_ChangeLineStatus           ;┐若软盘驱动器不支持 Change Line,则直接返回
7528:    JZ      Rebuild_BPB_RET                 ;┘
7529:    XOR     SI,SI                           ;SI←0(未知介质更换否)
7530:    CMP     BYTE PTR DiskBuffer+10H,00      ;┐若 FAT 表个数为 0,则直接返回
7531:    JZ      ChkWith_RET                     ;┘
7532:    CALL    Get_VolumeID                    ;读取驱动器上当前盘片的卷标名
7533:    JC      ChkWith_RET                     ;若盘操作失败,则直接返回
7534:    CALL    Test_VolumeID                   ;判别卷标名是否一致
7535:    MOV     SI,-1                           ;SI←-1(介质已被更换)
7536:    JNZ     ChkWith_2                       ;若卷标中不一致,则跳转
7537:    INC     SI                              ;SI←0(未知介质更换否)
7538: ChkWith_1:
7539:    CALL    Clear_ModifyBPBFlag             ;清重建 BPB 参数块标志
7540:    CLC
7541: ChkWith_RET:
7542:    RET
7543: ChkWith_2:
7544:    CLC
7545:    MOV     Drive_Number,-1                 ;置介质被更换的标志
7546:    RET
7547: ChkWith_3:                                 ;根据卷系列号判定介质是否被更换
7548:    PUSH    AX
7549:    MOV     AX,WORD PTR DiskBuffer+27H              ;┐
7550:    CMP     AX,WORD PTR ES:[DI].BDPB_SerialNumber   ;|
```

```
7551:    JNZ     ChkWith_4                                    ;| 若卷系列号不一致(它表明介质
7552:    MOV     AX,WORD PTR DiskBuffer+29H                   ;| 自已被更换),则跳转
7553:    CMP     AX,WORD PTR ES:[DI].BDPB_SerialNumber+2;     |
7554:    JNZ     ChkWith_4                                    ;┘
7555:    XOR     SI,SI                          ;SI←0(未知介质更换否)
7556:    POP     AX
7557:    JMP     SHORT ChkWith_1
7558: ChkWith_4:
7559:    POP     AX
7560:    MOV     SI,-1                          ;SI←-1(介质已被更换)
7561:    CLC
7562:    JMP     SHORT ChkWith_2
7563: ChkWithMediaDescript  ENDP
7564: ;=============================================================
7565: ;相对地址偏移:18CA
7566: ;功能:若驱动器上的介质已被更换,则进行相应的处理
7567: ;入口参数:AH=INT 13H 返回的状态字节值;
7568: ;          ES:DI 指向指定逻辑驱动器对应的 BDPB
7569: ;出口参数:无
7570: ;=============================================================
7571: Process_RemMedia  PROC  NEAR
7572:    CMP     AH,6                           ;┐ 若不是"无效的介质更换错",则直接返回
7573:    JNE     ChkWith_RET                    ;┘
7574:    CALL    Test_DevOpCount                ;┐ 若"设备打开计数器"值为 0,则直接返回
7575:    JZ      ChkWith_RET                    ;┘
7576:    CALL    Modify_BuildBPB                ;建立 Build BPB 参数块
7577:    JC      Pro_RemM3                      ;若建立 Build BPB 参数块失败,则跳转
7578:    CALL    ChkWithMediaDescript           ;┐
7579:    JC      Pro_RemM2                      ;| 若介质已被更换,则跳转
7580:    OR      SI,SI                          ;|
7581:    JS      Pro_RemM1                      ;┘
7582:    INC     BP                             ;允许多一次出错重试
7583:    RET
7584: Pro_RemM1:
7585:    CALL    Set_ReturnPara                 ;设置设备驱动程序的返回参数
7586: Pro_RemM2:
7587:    STC
7588:    JMP     Disk_Op2_23
7589: Pro_RemM3:
7590:    JMP     Disk_Op2_24
7591: Process_RemMedia  ENDP
7592: ;=============================================================
7593: ;相对地址偏移:18EE
```

```
7594: ;功能:设置设备驱动程序的返回参数(先前存取介质的卷标 ID 指针值)和 INT 13H 返回的错误码
7595: ;入口参数:ES:DI 指向逻辑驱动器对应的 BDPB
7596: ;出口参数:CF=1,AH=INT 13H 返回的错误码;
7597: ;           ES:DI 指向逻辑驱动器对应的 BDPB
7598: ;=============================================================
7599: Set_ReturnPara  PROC  NEAR
7600:     MOV     SI,0016H
7601:     CALL    Set_LastVolumeID_1              ;设置"先前的卷标 ID 指针值"
7602:     MOV     AH, 06                          ;AH←错误码(无效的磁盘更换)
7603:     STC                                     ;CF←1(失败)
7604:     RET
7605: Set_ReturnPara ENDP
7606: ;=============================================================
7607: ;相对地址偏移:18F8
7608: ;功能:设置设备驱动程序的返回参数(先前存取介质的卷标 ID 指针值)
7609: ;入口参数:ES:DI 指向逻辑驱动器对应的 BDPB
7610: ;出口参数:ES:DI 指向逻辑驱动器对应的 BDPB
7611: ;=============================================================
7612: Set_LastVolumeID  PROC  NEAR
7613:     MOV     SI,000FH
7614: Set_LastVolumeID_1  LABEL  NEAR             ;DOS Version=4.01,5.00
7615:     PUSH    DS                              ;┐
7616:     LDS     BX,DRH_Ptr                      ;|
7617:     ADD     DI,4BH                          ;| 设置设备驱动程序的返回参数(先前存取介质
7618:     MOV     [BX+SI],DI                      ;| 的卷标 ID 指针值)
7619:     SUB     DI,4BH                          ;|
7620:     MOV     [BX+SI+02],ES                   ;┘
7621:     POP     DS
7622:     RET
7623: Set_LastVolumeID  ENDP
7624: ;=============================================================
7625: ;相对地址偏移:190D
7626: ;功能:设置 1.2MB(5.25 英寸)的高密软盘对应的 Build BPB 参数块的参数值
7627: ;入口参数:AH=介质描述符;
7628: ;           ES:DI 指向指定逻辑驱动器对应的 BDPB
7629: ;出口参数:无
7630: ;=============================================================
7631: Update_BuildBPB  PROC  NEAR
7632:     TEST    WORD PTR ES:[DI].BDPB_Attr_Status,2  ;┐ 若软盘驱动器不支持 Change Line,则
7633:     JZ      Update_BuildBPB_RET                  ;┘ 直接返回
7634:     CMP     BYTE PTR ES:[DI].BDPB_DeviceType,2   ;┐ 若软盘驱动器是 720KB(3.5 英寸软
7635:     JE      Update_BuildBPB_RET                  ;┘ 盘),则直接返回
7636:     CMP     AH,0F9H                              ;┐ 若不是高密软盘,则直接返回
```

```
7637:    JNE     Update_BuildBPB_RET                          ;┘
7638:    CMP     BYTE PTR ES:[DI].BDPB_DeviceType,7   ;┐ 若软盘驱动器是 1.44MB(3.5 英寸软
7639:    JE      Update_BuildBPB_1                            ;┘ 盘),则跳转
7640:    CMP     BYTE PTR ES:[DI].BDPB_DeviceType,9   ;┐ 若软盘驱动器是 2.88MB(3.5 英寸软
7641:    JE      Update_BuildBPB_1                            ;┘ 盘),则跳转
7642:    MOV     AL,7                              ;┐ 设置 BPB 参数块的参数,其中 AL=7(每个 FAT
7643:    MOV     BX,0E00FH                         ;│ 表占用的扇区数)、BL=15(每道扇区数)、BH=
7644:    MOV     CX,960H                           ;│ 224(最大根目录项数)、CX=2400(总扇区数)、
7645:    MOV     DX,102H                           ;┘ DL=2(磁头数)、DH=1(每簇扇区数)
7646:    ADD     SP,2                              ;去掉该子程序的返回地址
7647:    JMP     Modify_BuildBPB8                  ;转去设置 Build BPB 参数块的参数值
7648: Update_BuildBPB_1:
7649:    ADD     SP,2                              ;去掉该子程序的返回地址
7650:    JMP     Modify_BuildBPB5                  ;转去修改 Build BPB 参数块的参数值
7651: Update_BuildBPB_RET:
7652:    RET
7653: Update_BuildBPB  ENDP
7654: ;=============================================================
7655: ;相对地址偏移:1947
7656: ;功能:设置 BDPB 的设备属性和状态标志字
7657: ;入口参数:BX=BDPB 的设备属性和状态标志字的变化值
7658: ;          DL=物理驱动器号
7659: ;出口参数:无
7660: ;=============================================================
7661: Set_BDPB_DevInfo  PROC  NEAR
7662:    PUSH    ES
7663:    PUSH    DI
7664:    LES     DI,DWORD PTR BDPB_Head            ;ES:DI 指向第 1 个 BDPB
7665: Set_BDPB_DevInfo1:
7666:    CMP     ES:[DI].BDPB_DriveNumber,DL       ;┐ 若物理驱动器号不匹配,则跳转
7667:    JNE     Set_BDPB_DevInfo2                 ;┘
7668:    OR      ES:[DI].BDPB_Attr_Status,BX       ;设置 BDPB 的设备属性和状态标志字的变化值
7669: Set_BDPB_DevInfo2:
7670:    LES     DI,DWORD PTR ES:[DI].BDPB_NextPtr_OFS     ;ES:DI 指向下一个 BDPB
7671:    CMP     DI,-1                             ;┐ 若 BDPB 链表未搜索完,则跳转
7672:    JNE     Set_BDPB_DevInfo1                 ;┘
7673:    POP     DI
7674:    POP     ES
7675:    RET
7676: Set_BDPB_DevInfo  ENDP
7677: ;=============================================================
7678: ;相对地址偏移:1962
7679: ;功能:测试是否需要重建 BPB 参数块
```

```
7680:  ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7681:  ;出口参数:ZF=1,不需要重建 BPB 参数块;ZF=0,需要重建 BPB 参数块
7682:  ;=============================================================
7683:  Test_ModifyBPBFlag  PROC  NEAR
7684:      TEST    WORD PTR ES:[DI].BDPB_Attr_Status,0040H
7685:      RET
7686:  Test_ModifyBPBFlag  ENDP
7687:  ;=============================================================
7688:  ;相对地址偏移:1969
7689:  ;功能:清重建 BPB 参数块标志(bit6=0 表明 BPB 参数块已更换,不需要重建 BPB 参数块)
7690:  ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7691:  ;出口参数:无
7692:  ;=============================================================
7693:  Clear_ModifyBPBFlag  PROC  NEAR
7694:      AND     WORD PTR ES:[DI].BDPB_Attr_Status,-41H
7695:      RET
7696:  Clear_ModifyBPBFlag  ENDP
7697:  ;=============================================================
7698:  ;相对地址偏移:196F
7699:  ;功能:测试软盘驱动器是否支持 Change Line
7700:  ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7701:  ;出口参数:ZF=1,不支持;ZF=0,支持
7702:  ;=============================================================
7703:  Test_ChangeLineStatus  PROC  NEAR
7704:      TEST    WORD PTR ES:[DI].BDPB_Attr_Status,2
7705:      RET
7706:  Test ChangeLineStatus  ENDP
7707:  ;=============================================================
7708:  ;相对地址偏移:1976
7709:  ;功能:设置 BDPB 的"卷标名"参数值
7710:  ;入口参数:ES,DI 指向指定逻辑驱动器对应的 BDPB
7711:  ;出口参数:CF=0,成功;CF=1,失败
7712:  ;=============================================================
7713:  VolumeIDtoBDPB  PROC  NEAR
7714:      PUSH    DX
7715:      PUSH    AX
7716:      CALL    Test ChangeLineStatus        ;┐若软盘驱动器不支持 Change Line,则结束返回
7717:      JZ      VolumeIDtoBDPB1              ;┘
7718:      CALL    Get VolumeID                 ;取出或补缺驱动器上当前介质的卷标
7719:      JB      VolumeIDtoBDPB2              ;若磁盘操作失败,则结束返回
7720:      CALL    ModifyBDPB VolumeID          ;卷标字符串→BDPB
7721:      CALL    Clear ModifyBPBFlag          ;清重建 BPB 参数块的标志
7722:  VolumeIDtoBDPB1:
```

```
7723:    CLC
7724:    POP     AX
7725:    POP     DX
7726:    RET
7727: VolumeIDtoBDPB2:
7728:    POP     DX
7729:    POP     DX
7730:    RET
7731: VolumeIDtoBDPB  ENDP
7732: ;
7733: SectorOfBootDirectory  DW  0                ;存放当前待搜索的根目录登记项所在扇区的扇区
                                                 ;号
7734: ;
7735: ;=============================================================
7736: ;相对地址偏移:1991
7737: ;功能:读取驱动器上当前介质的卷标名。若没有卷标,则置卷标为缺省的字符串
7738: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7739: ;出口参数:CF=0,取出了卷标名或补缺了卷标名;CF=1,读目录扇区时盘操作失败
7740: ;=============================================================
7741: Get_VolumeID  PROC  NEAR
7742:    PUSH    DX
7743:    PUSH    CX
7744:    PUSH    BX
7745:    PUSH    AX
7746:    PUSH    ES
7747:    PUSH    DI
7748:    PUSH    DS
7749:    POP     ES
7750:    MOV     DI, offset VolumeIDBuffer       ;ES:DI 指向存放当前卷标的缓冲区地址
7751:    MOV     SI, offset DefaultVolID         ;DS:SI 指向缺省的卷标名
7752:    MOV     CX, FileNameLen+1               ;┐补缺卷标名
7753:    REP     MOVSB                           ;┘
7754:    POP     DI
7755:    POP     ES
7756:    MOV     AL,ES:[DI].BDPB_NumberOfFAT1;┐
7757:    MOV     CX,ES:[DI].BDPB_SectorsPerFAT1;├计算 FAT 表占用的扇区数→AX
7758:    MUL     CL                              ;┘
7759:    ADD     AX,ES:[DI].BDPB_ReservedSectors1    ;加上保留扇区数
7760:    MOV     CS:SectorOfBootDirectory,AX     ;设置当前待搜索的根目录登记项所在扇区的扇区
                                                 ;号
7761:    MOV     AX,ES:[DI].BDPB_TotalOfRootDir;AX←最大根目录登记项数
7762:    MOV     CL,4                            ;┐
7763:    SHR     AX,CL                           ;├计算根目录区占用的扇区数→CX
```

```
7764:    MOV    CX,AX                              ;┘
7765: Get_VolID1:
7766:    PUSH   CX
7767:    MOV    AX,CS:SectorOfBootDirectory        ;AX←当前待搜索的根目录登记项所在的扇区号
7768:    MOV    CX,ES:[DI].BDPB_SectorsPerTrack]CX←每道扇区数
7769:    XOR    DX,DX                              ;┐
7770:    DIV    CX                                 ;│计算磁道号和道内扇区号,其中 CL=道内扇区
7771:    INC    DX                                 ;│号、AX=磁道号
7772:    MOV    CL,DL                              ;┘
7773:    XOR    DX,DX                                    ;┐
7774:    DIV    WORD PTR ES:[DI].BDPB_NumberOfHead1 ;│计算圆柱面号和磁头号,其中 DH=磁
7775:    MOV    DH,DL                                    ;│头号、CH=圆柱面号
7776:    MOV    CH,AL                                    ;┘
7777:    CALL   Read_Disk                          ;读出当前待搜索的根目录登记项所在的一个目录
                                                   ;扇区
7778:    JC     Get_VolID9                         ;若读目录扇区失败,则跳转
7779:    MOV    CX,10H                             ;CX←每个目录扇区存放的目录登记项数
7780:    MOV    AL,8                               ;AL←卷标的文件属性值
7781: Get_VolID2:
7782:    CMP    BYTE PTR [BX],0                    ;┐若当前目录登记项为空,表明以后无有效的目
7783:    JE     Get_VolID8                         ;┘录登记项,则跳转
7784:    CMP    BYTE PTR [BX],0E5H                 ;┐若当前目录登记项为一个删除目录项,则跳转
7785:    JE     Get_VolID3                         ;┘
7786:    TEST   [BX].FDT_FileAttribute,AL          ;┐若当前目录登记项是卷标,则跳转
7787:    JNZ    Get_VolID5                         ;┘
7788: Get_VolID3:
7789:    ADD    BX,20H                             ;ES:BX 指向下一个目录登记项
7790:    LOOP   Get_VolID2                         ;继续搜索本扇区的后续目录登记项
7791:    POP    CX
7792:    INC    WORD PTR CS:SectorOfBootDirectory  ;修改下次待搜索的根目录登记项所在扇区的
                                                   ;扇区号
7793:    LOOP   Get_VolID1                         ;继续搜索后续的根目录占用的扇区
7794: Get_VolID4:
7795:    XOR    SI,SI
7796:    JMP    SHORT Get_VolID6
7797: Get_VolID5:
7798:    POP    CX
7799:    MOV    SI,BX                              ;DS:SI 指向卷标名
7800:    PUSH   ES
7801:    PUSH   DI
7802:    PUSH   DS                                 ;┐
7803:    POP    ES                                 ;│ ES:DI 指向存放当前卷标的缓冲区
7804:    MOV    DI, offset VolumeIDBuffer          ;┘
```

```
7805:    MOV     CX, FileNameLen              ;┐ 复制卷标名
7806:    REP     MOVSB                        ;┘
7807:    XOR     AL,AL                        ;┐ 置卷标字符串的结束符
7808:    STOSB                                ;┘
7809:    XOR     SI,SI
7810:    POP     DI
7811:    POP     ES
7812: Get_VolID6:
7813:    POP     AX
7814:    CLC                                  ;CF←0(成功)
7815: Get_VolID7:
7816:    POP     BX
7817:    POP     CX
7818:    POP     DX
7819:    RET
7820: Get_VolID8:
7821:    POP     CX
7822:    JMP     SHORT Get_VolID4
7823: Get_VolID9:
7824:    POP     SI
7825:    POP     SI
7826:    JMP     SHORT Get_VolID7
7827: Get_VolumeID  ENDP
7828: ;=============================================================
7829: ;相对地址偏移:1A29
7830: ;功能:依据当前操作的介质上指定的卷标名修改对应的 BDPB 的卷标名
7831: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7832: ;出口参数:无
7833: ;=============================================================
7834: ModifyBDPB_VolumeID  PROC  NEAR
7835:    PUSH    DI
7836:    PUSH    SI
7837:    PUSH    CX
7838:    MOV     SI,offset VolumeIDBuffer     ;DS:SI 指向存放由当前介质指定的卷标名的缓冲
                                             ;区
7839:    ADD     DI,4BH                       ;┐
7840:    MOV     CX, FileNameLen+1            ;│ 卷标字符串→BDPB
7841:    CLD                                  ;│
7842:    REP     MOVSB                        ;┘
7843:    POP     CX
7844:    POP     SI
7845:    POP     DI
7846:    RET
```

```
7847: ModifyBDPB_VolumeID  ENDP
7848: ;=========================================================
7849: ;相对地址偏移:1A3C
7850: ;功能:判定驱动器上由当前介质指定的卷标名是否与对应的 BDPB 中缓存的卷标名相同
7851: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7852: ;出口参数:ZF=0,卷标名不相同;ZF=1,卷标名相同
7853: ;=========================================================
7854: Test_VolumeID  PROC  NEAR
7855:     PUSH    DI
7856:     PUSH    CX
7857:     MOV     SI,offset VolumeIDBuffer        ;DS:SI 指向存放由当前介质指定的卷标名的缓冲
                                                  ;区
7858:     ADD     DI,4BH                          ;┐
7859:     MOV     CX,FileNameLen+1                ;├ 比较卷标的一致性
7860:     CLD                                     ;│
7861:     REPE    CMPSB                           ;┘
7862:     POP     CX
7863:     POP     DI
7864:     RET
7865: Test_VolumeID  ENDP
7866: ;=========================================================
7867: ;相对地址偏移:1A4D
7868: ;功能:判定当前操作的软盘的介质描述符是否与对应的 BDPB 中保存的介质描述符相同
7869: ;入口参数:ES:DI 指向指定逻辑驱动器对应的 BDPB
7870: ;出口参数:SI=0,介质描述符相同;SI=-1,介质描述符不相同
7871: ;=========================================================
7872: Test_MediaDescript  PROC  NEAR
7873:     PUSH    AX
7874:     XOR     SI,SI                           ;SI←0(介质描述符相同)
7875:     MOV     AL,DiskMedia                    ;┐
7876:     CMP     AL,ES:[DI].BDPB_MediaByte1      ;├ 若介质描述符相同,则跳转
7877:     JE      Test_MediaDecsript1             ;┘
7878:     DEC     SI                              ;SI←-1(介质描述符不同)
7879: Test_MediaDecsript1:
7880:     POP     AX
7881:     RET
7882: Test_MediaDescript  ENDP
7883:     DB      0,0,0,0;Debug
7884: IO2ENDS
7885:     END
7886:
7887:
7888:
```

```
7889:
```

10.4.6 IO3 模块的源程序注释清单

```
7890: PAGE 60,132
7891: ;Program Name: IO3.ASM
7892: ;Version: MS—DOS5.00
7893: ;Compiler: Marco Assembler Version 5.10
7894: ;
7895: EXTRN  AllowHMA:BYTE, HIMEMAddr:DWORD, IO2_00D0: BYTE, ReadKeyCode:BYTE
7896: EXTRN  GetKeyStatus:BYTE, Switch_T:BYTE, MultiTrack:WORD, Ptr_SearchSCB:DWORD
7897: EXTRN  Ptr_SCBArray:DWORD, SizeOfDynamicStk:WORD, CompatibleFDiskFlag:BYTE
7898: EXTRN  D_08D0:BYTE
7899: EXTRN  FreeMemOFSofHMA:WORD, PreEntry7: DWORD, IO3_Flag:BYTE
7900: EXTRN  StackFlag: BYTE,              Old_INT02H_Vector1: DWORD   ;┐
7901: EXTRN  Old_INT08H_Vector1: DWORD,  Old_INT09H_Vector1: DWORD   ; |
7902: EXTRN  Old_INT0AH_Vector1: DWORD,  Old_INT0BH_Vector1: DWORD   ; |
7903: EXTRN  Old_INT0CH_Vector1: DWORD,  Old_INT0DH_Vector1: DWORD   ; | 由 IO1 模块定义
7904: EXTRN  Old_INT0EH_Vector1: DWORD,  Old_INT70H_Vector1: DWORD   ; | 的变量
7905: EXTRN  Old_INT72H_Vector1: DWORD,  Old_INT73H_Vector1: DWORD   ; |
7906: EXTRN  Old_INT74H_Vector1: DWORD,  Old_INT76H_Vector1: DWORD   ; |
7907: EXTRN  Old_INT77H_Vector1: DWORD                               ;┘
7908: ;
7909: PUBLIC  Addr_MSDOS, Ptr_DriveLinker, RAMSize, DriveNumber_Boot, LogicDriveFlag
7910: ;
7911: Vector_02H_OFS EQU   8                ;┐
7912: Vector_08H_OFS EQU   20H              ; |
7913: Vector_09H_OFS EQU   24H              ; |
7914: Vector_0AH_OFS EQU   28H              ; |
7915: Vector_0BH_OFS EQU   2CH              ; |
7916: Vector_0CH_OFS EQU   30H              ; |
7917: Vector_0DH_OFS EQU   34H              ; |
7918: Vector_0EH_OFS EQU   38H              ; |
7919: Vector_0FH_OFS EQU   3CH              ; |
7920: Vector_13H_OFS EQU   4CH              ; |
7921: Vector_13H_SEG EQU   4EH              ; |
7922: Vector_19H_OFS EQU   64H              ; | 存放中断向量的地址
7923: Vector_19H_SEG EQU   66H              ; |
7924: Vector_2FH_OFS EQU   0BCH             ; |
7925: Vector_2FH_SEG EQU   0BEH             ; |
7926: Vector_30H_OFS EQU   0C0H             ; |
7927: Vector_70H_OFS EQU   1C0H             ; |
7928: Vector_72H_OFS EQU   1C8H             ; |
7929: Vector_73H_OFS EQU   1CCH             ; |
```

```
7930: Vector_74H_OFS EQU     1D0H                ;|
7931: Vector_76H_OFS EQU     1D8H                ;|
7932: Vector_77H_OFS EQU     1DCH                ;┘
7933: Info_13H       EQU     13H                 ;DOS 通讯区(段地址为40H)
7934: PSPAddr_36     EQU     36H                 ;PSP 结构块的偏移36H
7935: ROMAddr_FF01   EQU     0FF01H              ;┐ ROM BIOS 常量的地址偏移(段地址为
7936: ROMAddr_FFFE   EQU     0FFFEH              ;┘ 0F000H)
7937: ;
7938: INCLUDE BIO.STR
7939: ;
7940: IO3      SEGMENT PARA ' CODE'
7941:          ASSUME  CS:IO3, DS:IO3
7942:          ORG     0
7943:                          DW      0        ;未使用
7944: n_Stack                  DW      0        ;保存系统配置命令 stacks 设定的动态堆栈结构数
                                                ;(即 stacks=n,s 的 n 参数值)
7945: OFS_Stack                DW      0        ;保存动态堆栈区的起始地址偏移(它等于8 * n)
7946: s_Stack                  DW      0        ;保存系统配置命令 stacks 设定的每个堆栈的尺寸
                                                ;(字节数,即 stacks=n,s 的 s 参数值)
7947: OFS_ManageBlockHead      DW      0        ;┐ 保存动态堆栈的控制块数组的起始地址
7948: SEG_ManageBlockHead      DW      0        ;┘
7949: OFS_ManageBlockHead2     DW      8        ;保存动态堆栈的控制块的起始地址偏移值
7950: OFS_ManageBlockTail      DW      48H      ;保存最后一个堆栈的控制块的起始地址偏移值
7951: OFS_AllocatedManageBlock DW      48H      ;保存下次可供分配的动态堆栈对应的控制块的
                                                ;起始地址偏移值
7952: ;
7953: Old_INT02H_Vector2       DD      0        ;存放先前的 INT 02H 中断向量
7954: ;=========================================================
7955: ;相对地址偏移:0016
7956: ;功能:INT 02H 中断服务程序
7957: ;=========================================================
7958: INT02H_Entry  PROC    FAR
7959:     PUSH    AX
7960:     PUSH    ES                            ;┐
7961:     MOV     AX,ROM_SEG                    ;|
7962:     MOV     ES,AX                         ;| 判定机器型号,是 PC 机可转换型?
7963:     CMP     BYTE PTR ES:ROMAddr_FFFE,0F9H;|
7964:     POP     ES                            ;┘
7965:     JNZ     INT02H_E1                    ;若不是,则跳转
7966:     IN      AL,62H                       ;取系统状态和开关设置等管理信息
7967:     TEST    AL,80H
7968:     JZ      INT02H_E1
7969:     POP     AX
```

```
7970:    JMP     CS:Old_INT02H_Vector2          ;控制权转到先前的 INT 02H 中断服务程序
                                                ;(即链接原来的中断向量)
7971: INT02H_E1:
7972:    POP     AX
7973:    CALL    Link_PreviousISR               ;分配一个空闲的动态堆栈给 INT 02H,并在切换
                                                ;堆栈之后调用先前的 INT 02H 中断服务程序
7974:    DW      Old_INT02H_Vector2             ;存放"先前的 INT 02H 中断向量"的双字单元地址
7975: INT02H_Entry  ENDP
7976: Old_INT08H_Vector2      DD      0         ;存放先前的 INT 08H 中断向量
7977: ;=============================================================
7978: ;相对地址偏移:003C
7979: ;功能:INT 08H 中断服务程序
7980: ;=============================================================
7981: INT08H_Entry  PROC    FAR
7982:    CALL    Link_PreviousISR               ;分配一个空闲的动态堆栈给 INT 08H,并在切换
                                                ;堆栈之后调用先前的 INT 08H 中断服务程序
7983:    DW      Old_INT08H_Vector2             ;存放"先前的 INT 08H 中断向量"的双字单元地址
7984: INT08H_Entry  ENDP
7985: Old_INT09H_Vector2      DD      0         ;存放先前的 INT 09H 中断向量
7986: ;=============================================================
7987: ;相对地址偏移:0045
7988: ;功能:INT 09H 中断服务程序
7989: ;=============================================================
7990: INT09H_Entry  PROC    FAR
7991:    JMP     INT09H_E1
7992:    DB      0
7993: INT09H_E1:
7994:    CALL    Link_PreviousISR               ;分配一个空闲的动态堆栈给 INT 09H,并在切换
                                                ;堆栈之后调用先前的 INT 09H 中断服务程序
7995:    DW      Old_INT09H_Vector2             ;存放"先前的 INT 09H 中断向量"的双字单元地址
7996: INT09H_Entry  ENDP
7997: Old_INT70H_Vector2      DD      0         ;存放先前的 INT 70H 中断向量
7998: ;=============================================================
7999: ;相对地址偏移:0052
8000: ;功 能:INT 70H 中断服务程序
8001: ;=============================================================
8002: INT70H_Entry  PROC  FAR
8003:    CALL    Link_PreviousISR               ;分配一个空闲的动态堆栈给 INT 70H,并在切换
                                                ;堆栈之后调用先前的 INT 70H 中断服务程序
8004:    DW      Old_INT70H_Vector2             ;存放"先前的 INT 70H 中断向量"的双字单元地址
8005: INT70H_Entry  ENDP
8006: ;=============================================================
8007: ;相对地址偏移:0057
```

```
8008: ;功 能:INT 0AH 中断服务程序
8009: ;=============================================================
8010: INT0AH_Entry  PROC  FAR
8011:     JMP     Short INT0AH_E1
8012: Old_INT0AH_Vector2      DD      0       ;存放先前的 INT 0AH 中断向量
8013:     DB      ' KB' ,0
8014:     JMP     Short INT0AH_E2
8015:     DB      0,0,0,0,0,0,0
8016: INT0AH_E1:
8017:     CALL    Link_PreviousISR            ;分配一个空闲的动态堆栈给 INT 0AH,并在切换
                                              ;堆栈之后调用先前的 INT 0AH 中断服务程序
8018:     DW      Old_INT0AH_Vector2          ;存放"先前的 INT 0AH 中断向量"的双字单元地址
8019: INT0AH_E2:
8020:     IRET
8021: INT0AH_Entry  ENDP
8022: ;=============================================================
8023: ;相对地址偏移:006F
8024: ;功 能:INT 0BH 中断服务程序
8025: ;=============================================================
8026: INT0BH_Entry  PROC  FAR
8027:     JMP     Short INT0BH_E1
8028: Old_INT0BH_Vector2      DD      0       ;存放先前的 INT 0BH 中断向量
8029:     DB      ' KB' ,0
8030:     JMP     Short INT0BH_E2
8031:     DB      0,0,0,0,0,0,0
8032: INT0BH_E1:
8033:     CALL    Link_PreviousISR            ;分配一个空闲的动态堆栈给 INT 0BH,并在切换
                                              ;堆栈之后调用先前的 INT 0BH 中断服务程序
8034:     DW      Old_INT0BH_Vector2          ;存放"先前的 INT 0BH 中断向量"的双字单元地址
8035: INT0BH_E2:
8036:     IRET
8037: INT0BH_Entry  ENDP
8038: ;=============================================================
8039: ;相对地址偏移:0087
8040: ;功 能:INT 0CH 中断服务程序
8041: ;=============================================================
8042: INT0CH_Entry  PROC  FAR
8043:     JMP     Short INT0CH_E1
8044: Old_INT0CH_Vector2      DD      0       ;存放先前的 INT 0CH 中断向量
8045:     DB      ' KB' ,0
8046:     JMP     Short INT0CH_E2
8047:     DB      0,0,0,0,0,0,0
8048: INT0CH_E1:
```

```
8049:    CALL    Link_PreviousISR                ;分配一个空闲的动态堆栈给 INT 0CH,并在切换
                                                 ;换堆栈之后调用先前的 INT 0CH 中断服务程序
8050:    DW      Old_INT0CH_Vector2              ;存放"先前的 INT 0CH 中断向量"的双字单元地址
8051: INT0CH_E2:
8052:    IRET
8053: INT0CH_Entry  ENDP
8054: ;=============================================================
8055: ;相对地址偏移:009F
8056: ;功 能:INT 0DH 中断服务程序
8057: ;=============================================================
8058: INT0DH_Entry  PROC  FAR
8059:    JMP     Short INT0DH_E1
8060: Old_INT0DH_Vector2      DD      0          ;存放先前的 INT 0DH 中断向量
8061:    DB      'KB',0
8062:    JMP     Short INT0DH_E2
8063:    DB      0,0,0,0,0,0,0
8064: INT0DH_E1:
8065:    CALL    Link_PreviousISR                ;分配一个空闲的动态堆栈给 INT 0DH,并在切换
                                                 ;堆栈之后调用先前的 INT 0DH 中断服务程序
8066:    DW      Old_INT0DH_Vector2              ;存放"先前的 INT 0DH 中断向量"的双字单元地址
8067: INT0DH_E2:
8068:    IRET
8069: INT0DH_Entry  ENDP
8070: ;=============================================================
8071: ;相对地址偏移:00B7
8072: ;功 能:INT 0EH 中断服务程序
8073: ;=============================================================
8074: INT0EH_Entry  PROC  FAR
8075:    JMP     Short INT0EH_E1
8076: Old_INT0EH_Vector2      DD      0          ;存放先前的 INT 0EH 中断向量
8077:    DB      'KB',0
8078:    JMP     Short INT0EH_E2
8079:    DB      0,0,0,0,0,0,0
8080: INT0EH_E1:
8081:    CALL    Link_PreviousISR                ;分配一个空闲的动态堆栈给 INT 0EH,并在切换
                                                 ;堆栈之后调用先前的 INT 0EH 中断服务程序
8082:    DW      Old_INT0EH_Vector2              ;存放"先前的 INT 0EH 中断向量"的双字单元地址
8083: INT0EH_E2:
8084:    IRET
8085: INT0EH_Entry  ENDP
8086: ;=============================================================
8087: ;相对地址偏移:00CF
8088: ;功 能:INT 72H 中断服务程序
```

```
8089: ;=========================================================
8090: INT72H_Entry  PROC  FAR
8091:     JMP     Short INT72H_E1
8092: Old_INT72H_Vector2      DD      0          ;存放先前的 INT 72H 中断向量
8093:     DB      ' KB' ,0
8094:     JMP     Short INT72H_E2
8095:     DB      0,0,0,0,0,0,0
8096: INT72H_E1:
8097:     CALL    Link_PreviousISR               ;分配一个空闲的动态堆栈给 INT 72H,并在切换堆
                                                 ;栈之后调用先前的 INT 72H 中断服务程序
8098:     DW      Old_INT72H_Vector2             ;存放"先前的 INT 72H 中断向量"的双字单元地址
8099: INT72H_E2:
8100:     IRET
8101: INT72H_Entry  ENDP
8102: ;=========================================================
8103: ;相对地址偏移:00E7
8104: ;功 能:INT 73H 中断服务程序
8105: ;=========================================================
8106: INT73H_Entry  PROC  FAR
8107:     JMP     Short INT73H_E1
8108: Old_INT73H_Vector2      DD      0          ;存放先前的 INT 73H 中断向量
8109:     DB      ' KB' ,0
8110:     JMP     Short INT73H_E2
8111:     DB      0,0,0,0,0,0,0
8112: INT73H_E1:
8113:     CALL    Link_PreviousISR               ;分配一个空闲的动态堆栈给 INT 73H,并在切换堆
                                                 ;栈之后调用先前的 INT 73H 中断服务程序
8114:     DW      Old_INT73H_Vector2             ;存放"先前的 INT 73H 中断向量"的双字单元地址
8115: INT73H_E2:
8116:     IRET
8117: INT73H_Entry  ENDP
8118: ;=========================================================
8119: ;相对地址偏移:00FF
8120: ;功 能:INT 74H 中断服务程序
8121: ;=========================================================
8122: INT74H_Entry  PROC  FAR
8123:     JMP     Short INT74H_E1
8124: Old_INT74H_Vector2      DD      0          ;存放先前的 INT 74H 中断向量
8125:     DB      ' KB' ,0
8126:     JMP     Short INT74H_E2
8127:     DB      0,0,0,0,0,0,0
8128: INT74H_E1:
8129:     CALL    Link_PreviousISR               ;分配一个空闲的动态堆栈给 INT 74H,并在切换堆
```

```
                                        ;栈之后调用先前的 INT 74H 中断服务程序
8130:    DW      Old_INT74H_Vector2     ;存放"先前的 INT 74H 中断向量"的双字单元地址
8131: INT74H_E2:
8132:    IRET
8133: INT74H_Entry  ENDP
8134: ;=============================================================
8135: ;相对地址偏移:0117
8136: ;功 能:INT 76H 中断服务程序
8137: ;=============================================================
8138: INT76H_Entry  PROC  FAR
8139:    JMP     Short INT76H_E1
8140: Old_INT76H_Vector2    DD      0      ;存放先前的 INT 76H 中断向量
8141:    DB      ' KB' ,0
8142:    JMP     Short INT76H_E2
8143:    DB      0,0,0,0,0,0,0
8144: INT76H_E1:
8145:    CALL    Link_PreviousISR           ;分配一个空闲的动态堆栈给 INT 76H,并在切换堆
                                            ;栈之后调用先前的 INT 76H 中断服务程序
8146:    DW      Old_INT76H_Vector2         ;存放"先前的 INT 76H 中断向量"的双字单元地址
8147: INT76H_E2:
8148:    IRET
8149: INT76H_Entry  ENDP
8150: ;=============================================================
8151: ;相对地址偏移:012F
8152: ;功 能:INT 77H 中断服务程序
8153: ;=============================================================
8154: INT77H_Entry  PROC  FAR
8155:    JMP     Short INT77H_E1
8156: Old_INT77H_Vector2    DD      0      ;存放先前的 INT 77H 中断向量
8157:    DB      ' KB' ,0
8158:    JMP     Short INT77H_E2
8159:    DB      0,0,0,0,0,0,0
8160: INT77H_E1:
8161:    CALL    Link_PreviousISR           ;分配一个空闲的动态堆栈给 INT 77H,并在切换堆
                                            ;栈之后调用先前的 INT 77H 中断服务程序
8162:    DW      Old_INT77H_Vector2         ;存放"先前的 INT 77H 中断向量"的双字单元地址
8163: INT77H_E2:
8164:    IRET
8165: INT77H_Entry  ENDP
8166: ;=============================================================
8167: ;相对地址偏移:0147
8168: ;功能:分配一个空闲的动态堆栈给当前中断,并在切换堆栈之后调用原来的中断服务程序
8169: ;=============================================================
```

```
8170: Link_PreviousISR    PROC    NEAR
8171:     PUSH    AX
8172:     PUSH    BP
8173:     PUSH    ES
8174:     MOV     ES,CS:SEG_ManageBlockHead        ;┐ ES:BP←当前可供分配的动态堆栈控制块的起
8175:     MOV     BP,CS:OFS_AllocatedManageBlock   ;┘ 始地址
8176:     MOV     AL,1                             ;┐ 设置动态堆栈占用标志,并取出该动态堆栈先
8177:     XCHG    AL,ES:[BP].SCB_UseMark           ;┘ 前的使用状态
8178:     CMP     AL,0                             ;┐ 若当前动态堆栈不是空闲的,则跳转
8179:     JNZ     Short Link_PreviousISR_2         ;┘
8180:     SUB     CS:OFS_AllocatedManageBlock,8    ;设置下一个可供分配的动态堆栈控制块的起始地
                                                   ;址偏移值
8181: Link_PreviousISR_1:
8182:     MOV     ES:[BP].SCB_SPValue,SP           ;┐ 保存先前的堆栈指针寄存器值
8183:     MOV     ES:[BP].SCB_SSValue,SS           ;┘
8184:     MOV     AX,BP                            ;AX=当前动态堆栈控制块的起始地址偏移值
8185:     MOV     BP,ES:[BP].SCB_AddrOfStack       ;BP←当前堆栈区的栈顶地址偏移值
8186:     CMP     ES:[BP+0],AX                     ;当前动态堆栈处于正确的空闲状态吗?
8187:     JNE     Link_PreviousISR_4               ;若不是,则跳转
8188:     PUSH    BP                               ;┐
8189:     MOV     BP,SP                            ;│ AX←存放"原来的中断服务程序入口地址"的
8190:     MOV     AX,[BP+08]                       ;│ 双字单元的指针值
8191:     POP     BP                               ;┘
8192:     PUSH    ES                               ;┐
8193:     POP     SS                               ;│ 切换堆栈
8194:     MOV     SP,BP                            ;┘
8195:     MOV     BP,AX                            ;┐ CS:BP←存放"原来的中断服务程序入口地址"
8196:     MOV     BP,CS:[BP+0]                     ;┘ 的双字单元的地址
8197:     PUSHF                                    ;┐ 调用原来的中断服务程序(即链接原来的中断
8198:     CALL    DWORD PTR CS:[BP+0]              ;┘ 向量)
8199:     MOV     BP,SP                            ;┐ ES:BP 指向当前动态堆栈的控制块
8200:     MOV     BP,ES:[BP+0]                     ;┘
8201:     MOV     SS,ES:[BP].SCB_SSValue           ;┐ 恢复先前的堆栈指针寄存器值
8202:     MOV     SP,ES:[BP].SCB_SPValue           ;┘
8203:     MOV     ES:[BP].SCB_UseMark,0            ;设置动态堆栈的空闲标志
8204:     MOV     CS:OFS_AllocatedManageBlock,BP   ;修改下一个可供分配的动态堆栈控制块的起始地
                                                   ;址偏移值
8205:     POP     ES                               ;┐
8206:     POP     BP                               ;│ 恢复堆栈指针值
8207:     POP     AX                               ;┘
8208:     ADD     SP,2                             ;废弃返回地址          ┐ 中断返回
8209:     IRET                                     ;                      ┘
8210: Link_PreviousISR_2:
```

```
8211:    CMP     AL,1                          ;┐ 若当前动态堆栈已被占用,则跳转
8212:    JZ      Link_PreviousISR_3            ;┘
8213:    XCHG    AL,ES:[BP].SCB_UseMark        ;恢复当前动态堆栈的先前使用标志值
8214: Link_PreviousISR_3:
8215:    CALL    Allocate_Stack                ;请求分配一个空闲的动态堆栈给当前的中断服务
                                               ;程序
8216:    JMP     SHORT Link_PreviousISR_1
8217: Link_PreviousISR_4:
8218:    CMP     BP,CS:OFS_ManageBlockHead2    ;┐ 若 ES:BP 不是指向动态堆栈的控制块,则跳转
8219:    JB      Link_PreviousISR_3            ;┘
8220:    MOV     BP,AX                         ;┐ 设置堆栈区数据操作错的错误标志
8221:    MOV     ES:[BP].SCB_UseMark,3         ;┘
8222:    JMP     SHORT Link_PreviousISR_3
8223: Link_PreviousISR      ENDP
8224: ;=============================================================
8225: ;相对地址偏移:01CA
8226: ;功能:试图分配一个空闲的动态堆栈给当前中断服务程序,若当前没有空闲的动态堆栈供分配,则
        给出错误信息,同时系统进入死循环
8227: ;入口参数:无
8228: ;出口参数:ES:BP 指向刚分配的空闲动态堆栈的控制块
8229: ;=============================================================
8230: Allocate_Stack  PROC  NEAR
8231:    MOV     BP, CS:OFS_ManageBlockTail    ;BP←最后一个动态堆栈的控制块的起始地址
                                               ;偏移值
8232: AllocateS1:
8233:    CMP     BYTE PTR ES:[BP].SCB_UseMark,00   ;┐ 若当前动态堆栈不是空闲的,则跳转
8234:    JNZ     AllocateS2                        ;┘
8235:    MOV     AL, 01                        ;┐ 取出动态堆栈先前的使用标志,并设置动态堆
8236:    XCHG    AL, ES:[BP].SCB_UseMark       ;┘ 栈被占用的标志
8237:    CMP     AL, 00                        ;┐ 若当前动态堆栈先前是空闲的,则成功返回
8238:    JZ      AllocateS3                    ;┘
8239:    CMP     AL, 01                        ;┐ 若当前动态堆栈先前已被分配,则跳转
8240:    JZ      AllocateS2                    ;┘
8241:    MOV     ES:[BP].SCB_UseMark,AL        ;恢复当前动态堆栈的先前使用标志值
8242: AllocateS2:
8243:    CMP     BP, CS:OFS_ManageBlockHead2   ;┐ 若动态堆栈已全部用完,则转去给出错误信息
8244:    JZ      AllocateS4                    ;┘
8245:    SUB     BP, +08                       ;ES:BP←下一个动态堆栈的控制块
8246:    JMP     Short AllocateS1              ;转去继续搜索空闲的动态堆栈
8247: AllocateS3:
8248:    RET
8249: AllocateS4:                              ;┐
8250:    PUSH    DS                            ;┃
```

```
8251:    MOV    AX,ROM_SEG                    ;|
8252:    MOV    DS,AX                         ;| 若机器不是 PC Convertible,则跳转
8253:    CMP    BYTE PTR DS:ROMAddr_FFFE,0F9H;|
8254:    POP    DS                            ;|
8255:    JNZ    AllocateS5                    ;┘
8256:    MOV    AL,7                          ;┐
8257:    OUT    72H,AL                        ;┘
8258: AllocateS5:
8259:    CLI                                  ;
8260:    MOV    AL,-1                         ;┐ 设置8259-1      ┐ 禁止所有可屏蔽
8261:    OUT    21H,AL                        ;┘                 | 的中断
8262:    OUT    0A1H,AL                       ; 设置8259-2       ┘
8263:    MOV    SI,CS                         ;┐
8264:    MOV    DS,SI                         ;| DS:SI 指向错误信息串
8265:    MOV    SI,offset ErrMes              ;┘
8266:    PUSH   AX
8267:    PUSH   DS
8268:    MOV    AX,BIO_SEG
8269:    MOV    DS,AX
8270:    TEST   D_08D0,01
8271:    POP    DS
8272:    POP    AX
8273:    JE     AllocateS6
8274:    DB     9AH
8275:    DW     08D1H
8276:    DW     BIO_SEG
8277:    ;CALL  0070:08D1
8278: AllocateS6:                             ;┐
8279:    LODSB                                ;|
8280:    CMP    AL,' $'                       ;|
8281:    JE     AllocateS7                    ;| 以电传方式显示动态堆栈操作的错误信息
8282:    MOV    BL,7                          ;|
8283:    MOV    AH,0EH                        ;|
8284:    INT    10H                           ;|
8285:    JMP    SHORT AllocateS6              ;┘
8286: AllocateS7:                             ;┐ 系统进入死循环
8287:    JMP    SHORT AllocateS7              ;┘
8288: Allocate_Stack  ENDP
8289: ;动态堆栈分配、使用出错时显示的提示信息
8290: ErrMes  DB  CR, LF, 7, CR, LF
8291:    DB      ' Internal stack overflow' , CR, LF
8292:    DB      ' System halted' , CR, LF, ' $'
8293: ;==========================================
```

```
8294：；相对地址偏移：0267
8295：；功能：系统初始化 SysInt_II 的处理程序
8296：；=========================================
8297：SysInt_II_Entry1   PROC   FAR
8298：    JMP     SysII_E1_1
8299：；相对地址偏移：026A
8300：DOS_Install_Site      DB    0    ；DOS 安装控制标志(0：DOS 全部安装在常规内存低端；1：IO2模块
                                       ；和 MSDOS2模块已安装在 HMA 中；－1：IO2模块和 MSDOS2模块
                                       ；需要安装在 HMA)
8301：Address_ListOfList    DD    0    ；调用 MSDOS.SYS 系统文件的初始化程序之后，它存放多重表的
                                       ；起始地址
8302：Addr_MSDOS            DD    0    ；当前 MSDOS.SYS 系统文件在常规内存中的起始地址。在 SysInt
                                       ；－ I 执行时，它为 MSDOS.SYS 系统文件初始读入时在常规内
                                       ；存中的起始地址在 SysInt－ Ⅱ 执行时，它为 MSDOS.SYS 系统文
                                       ；件在常规内存高端的起始地址
8303：Ptr_DriveLinker       DD    0    ；指向由 IO1模块提供的设备驱动程序标题链的头结点(即 CON 设
                                       ；备驱动程序标题)
8304：Addr_ExtCIMB          DD    0    ；存放扩充国家信息管理块的起始地址
8305：SubRoutine1_MSDOS     DD    0    ；保存“安装 MSDOS2模块的子程序”的入口地址，该子程序由
                                       ；MSDOS2模块定义，其偏移地址为 BAB7H
8306：SizeOfMSDOS2InCMA     DW    0    ；保存 MSDOS2模块安装在常规内存中的长度(字节数，值为
                                       ；B3E0H－3DD0H)
8307：SizeOfMSDOS2InHMA     DW    0    ；保存 MSDOS2模块安装在高内存区(HMA)中的长度(字节数，
                                       ；值为 B76AH－3DD0H)
8308：Seg_FirstMCB1         DW    0    ；存放第一个内存块的段地址值
8309：SubRoutine_IO2        DW    32H  ；┐保存由 IO2模块定义的子程序的入口地址(当 IO2模块移动时，
                                       ；｜该子程序修改保存在 IO1模块里的但由 IO2模块定义的子程序
8310：                      DW    2C7H ；┘中断地址
8311：FDI_Flag_IO3          DB    0    ；软盘驱动器的安装标志(1：没有安装软盘驱动器)
8312：NumberOfStack         DW    9    ；存放 stacks 系统配置命令指定的动态堆栈个数
8313：SizeOfStack           DW    80H  ；存放 stacks 系统配置命令指定的动态堆栈尺寸(字节数)
8314：OFS_StackField        DW    0    ；┐存放 stack(堆栈)数据块的结束地址(处理 CONFIG.SYS 时，
                                       ；｜ OFS_StackField 作为 stack 系统配置命令是否出现的标志(0：
8315：SEG_StackField        DW    0    ；┘没有；－1：用户指定了))
8316：；相对地址偏移：0292
8317：RAMSize               DW    1    ；存放以节为单位的常规内存容量
8318：Unknown               DW    0
8319：DriveNumber_Boot      DB    0    ；保存引导盘对应的驱动器号(1＝A，3＝C)
8320：NumberOfBuffer        DW    －1   ；存放 buffers 系统配置命令指定的磁盘缓冲区数目(即 buffers＝x，
                                       ；y 的 x 值)
8321：NumberOf2ndBuffer     DW    0    ；存放 buffers 系统配置命令指定的第二个高速缓存区中的缓冲区
                                       ；数目(即 buffers＝x，y 的 y 值)
8322：SizePerBuffer         DW    0    ；保存每个盘缓冲区的尺寸(字节数，标准值为200H＋14H＝532)
```

```
8323: NumberOfHandle    DB    8    ;存放 files 系统配置命令指定的能同时打开的最大文件数
8324: NumberOfFCB       DB    4    ;存放 fcbs 系统配置命令指定的可被 DOS 同时打开的文件控制块
                                   ;(FCB)数目
8325: NumberOfKeepFCB   DB    0    ;存放 fcbs 系统配置命令指定的由 FCB 命令打开而 DOS 不能自动
                                   ;关闭的文件数(在 MS—DOS 5.00版本中,该值恒为0)
8326: LastDrive         DB    5    ;存放 lastdrive 系统配置命令指定的可以访问的最大逻辑驱动器数
8327: SEG_Config        DW    0    ;保存 CONFIG.SYS 文件内容在常规内存中的段地址值
8328: Seg_CDSArray      DW    0    ;保存块设备的 CDS 数组的段地址值
8329: ;相对地址偏移:02A5
8330: Rootpath_CDS      DB    'A:\',0    ;当前目录结构的缺省目录路径串(根目录)
8331: ;相对地址偏移:02A9
8332: CMDLine           DB    2          ;┐
8333:                   DB    0,'P'      ;│存放命令行参数串
8334:                   DB    7EH DUP(0);┘
8335: ;相对地址偏移:032A
8336: TerminatorOfData  DB    0    ;数据终止符
8337: LineCounter       DW    0    ;行号计数器(当前正在处理的系统配置命令在 CONFIG.SYS 文本
                                   ;文件中的行号)
8338: OutputBuffer      DB    '  '                    ;┐输出 CONFIG.SYS 命令行的行号时使用
8339:                   DB    ' ',0DH,0AH,'¥';┘ 的输出缓冲区
8340: LineNumOfLastBuffer DW  0    ;存放最后一个 buffers 系统配置命令在 CONFIG.SYS 文本文件
                                   ;中的行号
8341: ModelByte         DB    -1   ;机器型号字节(机器 ID 值)
8342: SubmodelByte      DB    0    ;机器子型号字节
8343: OFS PreviousBuffer DW   0    ;保存先前建立的盘缓冲区的起始地址偏移值
8344: ;执行命令处理器时使用的 EXEC 功能调用的参数块
8345: EXECPB1 EXECParaBlock <,CMDLine,SysInt_II_SEG,DriveNumber_Boot,SysInt_II_SEG,0329H,
      SysInt_II_SEG>
8346: ;相对地址偏移:0349
8347: ProcessConfigFlag  DB    0    ;处理 CONFIG.SYS 文件时使用的控制标志,其值的含义如下:
8348:                                ; 0:只识别 dos 系统配置命令,决定 MS—DOS 的安装位置
8349:                                ; 1:检查是否存 install 系统配置命令,并识别 buffers、break、
                                     ;multitrack、devicehigh、device、drivparm、stacks、shell、fcbs 命令;
8350:                                ; 2:不识别任何命令;
8351:                                ; 3:处理 install 命令;
8352:                                ; 0BH:未能打开 CONFIG.SYS 文件,即不存在 CONFIG.SYS 文件。
8353: Install_Flag       DW    0    ;install 系统配置命令使用的标志(位0=1,表示 CONFIG.SYS 文件
                                   ;中有 install 系统配置命令;位1=1,表示在常规内存中建立了一
                                   ;个临时子段,该子段用来存放控制 install 系统配置命令指定的
                                   ;DOS 内存驻留程序运行的子程序)
8354: LengthOfConfig    DW    0    ;保存 CONFIG.SYS 文件长度(字节数)
8355: SubRoutine_103    DD    0    ;保存由 103模块定义的子程序(ExecuteDOSFunction)的入口地址,
                                   ;此时该子程序已被安装在常规内存的一个临时子段中,它控制
```

```
                                        ;install 系统配置命令指定的 DOS 内存驻留程序的运行
8356: LabelAddr_IO3     DD    0         ;保存后续程序段(Process_Install_5)的入口地址,该程序段是子程
                                        ;序(ExecuteDOSFunction)的一个组成部分
8357: AccumulatedCode   DW    0         ;当 SysInt一Ⅱ执行时,它保存 CONFIG.SYS 文件内容和 IO3模块
                                        ;有效代码的累加和。当由 install 系统配置命令指定的 DOS 内存
                                        ;驻留程序执行后,它用来检测 CONFIG.SYS 文件内容和 IO3模块
                                        ;是否被破坏
8358: FirstFCB          EQU   This BYTE ;EXEC 功能参数块中的 FCB1
8359: SecondFCB         DB    20 Dup(' ');EXEC 功能参数块中的 FCB2
8360: ;相对地址偏移:036C
8361: LengthOfPara      DB    0         ;保存程序参数(命令行参数)的字符数
8362: TempParaBuf       DB    20H       ;┐程序参数的暂存区,它保存 install 系统配置命令指定
8363:                   DB    50H Dup(0) ;┘的程序参数
8364: ;相对地址偏移:03BE
8365: ;EXEC 功能调用的参数块,它用于装入并执行由 install 系统配置命令指定的 DOS 内存驻留程序
8366: EXECPB2 EXECParaBlock <,LengthOfPara,SysInt_II_SEG,FirstFCB,SysInt_II_SEG,SecondFCB,
      SysInt_II_SEG>
8367: ;相对地址偏移:03CC
8368: QuotationMark       DB    0       ;双引号作用标志(0:双引号已配对;1:双引号未配对)
8369: CharsOfPreface      DB    0       ;保存 comment 系统配置命令指定的注释前导符 ;┐
                                        ;的组成字符个数                          ;│为0时表示未
8370: FirstCharOfPreface  DB    0       ;保存 comment 系统配置命令指定的注释前导符 ;│定义注释前
                                        ;的第1个组成字符                         ;│导符
8371: SecondCharOfPreface DB    0       ;保存 comment 系统配置命令指定的注释前导符 ;│
                                        ;的第2个组成字符                         ;┘
8372: KeySign             DB    0       ;保存(暂存)当前匹配的系统配置命令的缩写关键字符(如 buffers
                                        ;系统配置命令的'B')
8373: ErrorCtrlFlag       DB    0       ;错误位置信息的输出控制标志(1:不输出错误系统配置命令在
                                        ;CONFIG.SYS 文本文件中的位置信息)
8374: ConfigCount         DW    0       ;CONFIG.SYS 待处理的字符计数器
8375: BytesOfConfig       DW    0       ;保存 CONFIG.SYS 文件内容转换后的字节数(即有效内容大小)
8376: Ptr_Config          DW    0       ;CONFIG.SYS 待处理内容的读指针
8377: FileHandle          DW    0       ;存放国家信息文件的文件句柄
8378: Seg_FirstMCB2       DW    0       ;存放第一个内存块的段地址值
8379: SizeOfDDB           DW    0       ;在第一个内存块中,DOS 的各种数据块(如 CDS 数组等)和设备
                                        ;驱动程序占用的内存大小(节数)
8380: OFS_FreeMemory      DW    0       ;┐当前常规自由内存区的可供使用的起始地址
8381: SEG_FreeMemory      DW    0       ;┘
8382: ;相对地址偏移:03E2
8383:                     DW    0       ;未使用
8384: SEG_AllocatedMemory DW    0       ;保存当前分配到的自由内存区的段地址值
8385: ;相对地址偏移:03E6                 ;设备驱动程序的初始化请求标题
8386: RequestForInit      DB    18H     ;初始化请求标题的长度
```

```
8387:                       DB    0    ;部件号
8388:                       DB    0    ;初始化的命令码
8389:                       DW    0    ;状态字
8390:                       DB    8  Dup(0)   ;保留区
8391:                       DB    0    ;返回的块设备部件数(对块设备)
8392:                       DD    0    ;结束地址
8393:                       DW    0    ;┐程序名/BPB指针数组的起始地址
8394:                       DW    0    ;┘
8395:                       DB    0    ;起始逻辑驱动器号(对块设备)
8396: DispLineNumFlag       DW    0    ;标志(≠0,给出错误的系统配置命令行在CONFIG.SYS文本文件
                                        ;中的行号)
8397: LogicDriveFlag        DB    0    ;逻辑驱动器标志(1:逻辑驱动器超过26个;0:逻辑驱动器数未超
                                        ;过26个)
8398: Seg_IO2               DW    2C7H       ;保存IO2模块在内存中的段地址值
8399: ValidIO3_IO1          DW    IO3_Flag   ;┐"IO3模块有效标志"的地址,该标志单元位于
8400:                       DW    BIO_SEG    ;┘ Z01模块中
8401:                       DB    128 DUP (0) ;未使用
8402: ;相对地址偏移:0486
8403: SysII_E1_1:
8404:     MOV     AH, 0C0H                    ;┐取出系统配置参数,若ROM BIOS支持该功能,
8405:     INT     15H                         ;┘则ES:BX指向系统描述表
8406:     JB      SysII_E1_2                  ;┐
8407:     CMP     AH, 00                      ;│若系统ROM BIOS不支持该功能,则跳转
8408:     JNZ     SysII_E1_2                  ;┘
8409:     MOV     AL, ES:[BX+02]              ;┐
8410:     MOV     CS:ModelByte, AL            ;│保存机器型号(即机器ID值)
8411:     MOV     AL, ES:[BX+03]              ;│
8412:     MOV     CS:SubmodelByte, AL         ;┘
8413:     JMP     Short SysII_E1_4
8414: SysII_E1_2:                            ;XT档次机器
8415:     MOV     AX,ROM_SEG                  ;┐
8416:     MOV     DS,AX                       ;│从ROM数据区中取出机器ID值,并保存
8417:     MOV     AL,DS:ROMAddr_FFFE          ;│该ID值
8418:     MOV     CS:ModelByte,AL             ;┘
8419: SysII_E1_3:
8420:     INT     11H                         ;取出系统配置的设备信息
8421:     TEST    AX,0001                     ;┐若安装了软盘驱动器,则跳转
8422:     JNE     SysII_E1_4                  ;┘
8423:     PUSH    ES                          ;┐
8424:     XOR     CL,CL                       ;│
8425:     MOV     AH,08                       ;│取软盘驱动器参数
8426:     MOV     DL,00                       ;│
8427:     INT     13H                         ;│
```

```
8428:   POP     ES                          ;┘
8429:   JB      SysII_E1_4                  ;┐
8430:   CMP     CL,0                        ; |
8431:   JE      SysII_E1_4                  ; | 若没有能工作的软盘驱动器,则跳转
8432:   OR      DL,DL                       ; |
8433:   JNE     SysII_E1_4                  ;┘
8434:   MOV     CS:FDI_Flag_IO3,01          ;置没有安装软盘驱动器的标志
8435: SysII_E1_4:
8436:   CLD
8437:   XOR     SI,SI
8438:   MOV     DI,SI
8439:   MOV     CX,CS:RAMSize               ;CX=系统常规内存容量(节单位)
8440:   PUSH    CS                          ;┐ DS←SysInt—Ⅱ所在的段地址值
8441:   POP     DS                          ;┘
8442:   DEC     CX
8443:   XOR     BX,BX                       ;┐
8444:   MOV     ES,BX                       ; | ES:BX=INT 2FH 的中断向量
8445:   MOV     BX,ES:Vector_2FH_OFS        ; |
8446:   MOV     ES,ES:Vector_2FH_SEG        ;┘
8447:
8448:   CMP     WORD PTR ES:[BX+3],' PR'
8449:   JNE     SysII_E1_5
8450:   CMP     BYTE PTR ES:[BX+5],' L'
8451:   JNE     SysII_E1_5
8452:   MOV     DX,CX
8453:   PUSH    DX
8454:   MOV     AX,4A06H
8455:   INT     2FH
8456:   POP     AX
8457:   MOV     CX,DX
8458:   CMP     DX,AX
8459:   JE      SysII_E1_5
8460:   MOV     CS:Unknown,DX
8461:   DEC     CX
8462: SysII_E1_5:
8463:   MOV     AX, offset End_SysInt_II    ;┐ 计算 SysInt—Ⅱ程序的节数
8464:   CALL    ConvertToParagraphs         ;┘
8465:   SUB     CX,AX                       ;在常规内存高端留出存放 ┐
                                            ;SysInt—Ⅱ的内存空间    |
8466:   SUB     CX,0A00H                    ;在常规内存高端留出     |
                                            ;MSDOS.SYS 系统文件    | 确定 SysInt—Ⅱ移
                                            ;所需的内存空间        | 至常规内存高端的
8467:   MOV     AX,1A60H                    ;┐ 计算 IO2模块的节数   | 段地址
```

```
8468:    CALL    ConvertToParagraphs             ;┘                    |
8469:    SUB     CX,AX                           ;在常规内存高端留出 IO2  |
                                                 ;模块(即设备驱动程序)所 |
                                                 ;需的内存空间          ┘
8470:    MOV     ES,CX                           ;┘
8471:    MOV     CX,offset End_SysInt_II         ;┐
8472:    SHR     CX,1                            ;| 将 SysInt－Ⅱ复制到常规内存高端
8473:    REP     MOVSW                           ;┘
8474:    PUSH    ES                              ;┐
8475:    MOV     AX,offset SysInt_II_Entry2      ;| 控制权转移到 SysInt－Ⅱ的后续部分接着执行
8476:    PUSH    AX                              ;|
8477:    RETF                                    ;┘
8478: SysInt_II_Entry1  ENDP
8479: ;=========================================================
8480: ;功能:系统初始化 SysInt_II 的处理程序
8481: ;=========================================================
8482: SysInt_II_Entry2 PROC  FAR
8483:    MOV     AX,BIO_SEG                      ;┐ DS←IO1模块在常规内存低端的段地址值
8484:    MOV     DS,AX                           ;┘
8485:    MOV     WORD PTR PreEntry7+2,CS         ;设置 IO3模块定义的子程序(CalledByIO2)的段地
                                                 ;址值
8486:    MOV     IO3_Flag,01                     ;设置 IO3模块正在执行的标志(它被 IO1模块使
                                                 ;用)
8487:    MOV     AX, offset End_SysInt_II        ;AX←SysInt－Ⅱ程序的字节数
8488:    CALL    ConvertToParagraphs             ;计算 SysInt－Ⅱ程序的节数
8489:    MOV     CX,CS                           ;┐
8490:    ADD     AX,CX                           ;| ES＝MSDOS.SYS 系统文件在常规内存高端的
8491:    MOV     ES,AX                           ;┘ 段地址值
8492:    XOR     SI,SI                             ;┐
8493:    MOV     DI,SI                             ;|
8494:    MOV     DS,WORD PTR CS:Addr_MSDOS+2       ;| 将 MSDOS.SYS 移到常规内存高端,以便执行
8495:    MOV     CX,5000H                          ;| 其初始化部分
8496:    REP     MOVSW                             ;┘
8497:    MOV     WORD PTR CS:Addr_MSDOS+2,ES       ;设置 MSDOS.SYS 系统文件在常规内存高端的
                                                   ;段地址值
8498:    MOV     AX,DS:[3]                         ;┐ 变换 MSDOS.SYS 系统文件在常规内存高端
8499:    MOV     WORD PTR CS:Addr_MSDOS,AX         ;| 的地址格式(这是因为 MSDOS.SYS 系统文件
8500:    CALL    ConvertToParagraphs               ;| 是按起始地址偏移3DD0H 编程的,请参阅
8501:    SUB     WORD PTR CS:Addr_MSDOS+2,AX       ;┘ MSDOS.SYS 的源程序 DOS1.ASM)
8502:    MOV     AX,ES                             ;┐
8503:    ADD     AX,0A00H                          ;|
8504:    MOV     ES,AX                             ;|
8505:    XCHG    AX,WORD PTR CS:SubRoutine_IO2+2   ;|
```

```
8506:   MOV     DS,AX                           ;| 将 DOS BIOS 模块(IO.SYS 系统文件)提供
8507:   MOV     SI,0030H                        ;| 的设备驱动程序(IO2模块)移到常规内存
8508:   MOV     DI,SI                           ;| 高端
8509:   MOV     CX,1A60H                        ;|
8510:   SUB     CX,SI                           ;|
8511:   SHR     CX,1                            ;|
8512:   REP     MOVSW                           ;┘
8513:   MOV     AX,ES                           ;┐ 修改被 IO1模块调用但由 IO2模块定义的子程
                                                ;| 序入口地址(只改段地址值),这些子程序的入
8514:   CALL    DWORD PTR CS:SubRoutine_IO2     ;┘ 口地址保存在 IO1模块里
8515:   LES     DI,DWORD PTR CS:ValidIO3_IO1    ;ES:DI 指向"IO3模块有效标志"(ES:DI=70H:
                                                ;8FDH)
8516:   LDS     SI, CS:Ptr_DriveLinker          ;DS:SI 指向设备驱动程序标题链的头结点(即 CON
                                                ;设备驱动程序标题)
8517:   MOV     DX, CS:RAMSize                  ;DX=系统的常规内存节数
8518:   CLI                                     ;┐
8519:   MOV     AX, CS                          ;|
8520:   MOV     SS, AX                          ;| 设定堆栈指针
8521: SysInt_II_E2_1:                          ;|
8522:   MOV     SP, offset SysInt_II_E2_1       ;|
8523:   STI                                     ;┘
8524:   CALL    DWORD PTR CS:Addr_MSDOS         ;调用 MSDOS.SYS 系统文件的初始化程序,返回
                                                ;时,DS=第一个内存块的段地址值,ES:DI 指向
                                                ;DOS 核心数据块指针数组(XXXX:0D28H)
8525:   MOV     CS:Seg_FirstMCB1,DS             ;保存第一个内存块的段地址值
8526:   MOV     CS:SizeOfMSDOS2InHMA,AX         ;保存 MSDOS2模块安装在 HMA 中的长度(字节
                                                ;数)
8527:   MOV     CS:SizeOfMSDOS2InCMA,CX         ;保存 MSDOS2模块安装在常规内存中的长度(字节
                                                ;数)
8528:   MOV     WORD PTR CS:SubRoutine1_MSDOS,DX    ;保存"安装 MSDOS2模块的子程序"的入
                                                    ;口地址偏移值
8529:   MOV     AX, ES:[DI]                     ;┐
8530:   MOV     WORD PTR CS:Address_ListOfList, AX  ;| 取出并保存多重表的起始地址
8531:   MOV     AX, ES:[DI+2]                   ;|
8532:   MOV     WORD PTR CS:Address_ListOfList+2, AX ;┘
8533:   MOV     AX, ES:[DI+4]                   ;┐
8534:   MOV     WORD PTR CS:Addr_ExtCIMB, AX    ;| 取出并保存扩充国家信息控制块的起
8535:   MOV     AX, ES:[DI+6]                   ;| 始地址
8536:   MOV     WORD PTR CS:Addr_ExtCIMB+2, AX  ;┘
8537:   MOV     ES,WORD PTR CS:Addr_MSDOS+2     ;┐ 设置"安装 MSDOS2模块的子程序"的
8538:   MOV     WORD PTR CS:SubRoutine1_MSDOS+2,ES ;┘ 入口段地址
8539:   CMP     CS:Unknown,0
8540:   JE      SysInt_II_E2_2
```

```
8541:    MOV     BX,0FFFFH
8542:    MOV     AH,48H
8543:    INT     21H
8544:    MOV     AH,48H
8545:    INT     21H
8546:    MOV     ES,AX
8547:    PUSH    ES
8548:    SUB     AX,CS:Unknown
8549:    NEG     AX
8550:    DEC     AX
8551:    MOV     BX,AX
8552:    MOV     AH,4AH
8553:    INT     21H
8554:    MOV     BX,0FFFFH
8555:    MOV     AH,48H
8556:    INT     21H
8557:    MOV     AH,48H
8558:    INT     21H
8559:    DEC     AX
8560:    MOV     ES,AX
8561:    MOV     ES:MCB_ProcessID,0008          ;设置内存块 PID 值(8表示内存块被 DOS 操作系统
                                                ;使用)
8562:    MOV     WORD PTR ES:[0008],5052H       ;┐
8563:    MOV     WORD PTR ES:[000AH],004CH      ;├ 设置内存控制块的"程序名域"(RPL)
8564:    MOV     WORD PTR ES:[000CH],0          ;├
8565:    MOV     WORD PTR ES:[000EH],0          ;┘
8566:    POP     ES
8567:    MOV     AH,49H
8568:    INT     21H
8569: SysInt_II_E2_2:
8570:    LES     DI,CS:Address_ListOfList       ;ES:DI 指向多重表
8571:    CLC
8572:    MOV     AH,88H                         ;┐ 取扩充内存大小(AX=以节为单位的扩充内存
8573:    INT     15H                            ;┘ 容量)
8574:    JB      SysInt_II_E2_3                 ;若 ROM BIOS 不支持该功能(如 XT 档次机器),则
                                                ;跳转
8575:    MOV     ES:[DI+45H],AX                 ;将"扩充内存大小"保存在 DOS 内核数据区中
8576:    OR      AX,AX                          ;┐ 若系统没有扩充内存,则跳转
8577:    JE      SysInt_II_E2_3                 ;┘
8578:    CALL    ClearHIMEMMark                 ;清除扩充内存的起始32个字节,即清除 HIMEM.
                                                ;SYS 的安装标志
8579: SysInt_II_E2_3:
8580:    MOV     AX,ES:[DI+10H]                 ;AX←块设备的扇区大小(字节数)
```

```
8581:    ADD     AX,14H                  ;加上盘缓冲区控制块所需的字节数
8582:    MOV     CS:SizePerBuffer,AX     ;保存每个盘缓冲区所需的字节数
8583:    MOV     AL,CS:DriveNumber_Boot  ;┐将"引导盘的驱动器号(1=A,3=C)"保存在
8584:    MOV     ES:[DI+43H],AL          ;┘ DOS 内核数据区中
8585:    PUSHF
8586:    PUSH    BX
8587:    XOR     BX,BX                   ;清 CPU 类型标志
8588:    XOR     AX,AX                   ;┐
8589:    PUSH    AX                      ;|
8590:    POPF                            ;|
8591:    PUSHF                           ;| 若机器的 CPU 为8086/8088或80186/80188,则
8592:    POP     AX                      ;| 跳转
8593:    AND     AX,0F000H               ;|
8594:    CMP     AX,0F000H               ;|
8595:    JE      SysInt_II_E2_5          ;┘
8596:    MOV     AX,0F000H               ;┐
8597:    PUSH    AX                      ;|
8598:    POPF                            ;|
8599:    PUSHF                           ;| 若机器的 CPU 为80286,则跳转
8600:    POP     AX                      ;|
8601:    AND     AX,0F000H               ;|
8602:    JZ      SysInt_II_E2_4          ;┘
8603:    INC     BX
8604: SysInt_II_E2_4:
8605:    INC     BX
8606: SysInt_II_E2_5:
8607:    MOV     AX,BX
8608:    POP     BX
8609:    POPF
8610:    CMP     AX,2                    ;┐若机器的 CPU 不为80386或80486,则跳转
8611:    JNE     SysInt_II_E2_6          ;┘
8612:    MOV     BYTE PTR ES:[DI+44H],1  ;设置 MSDOS1模块中的 CPU 标志变量
8613: SysInt_II_E2_6:
8614:    MOV     AL,ES:[DI+20H]          ;┐保存系统中的逻辑驱动器数
8615:    MOV     CS:RequestForInit+16H,AL;┘
8616:    MOV     AX,CS                   ;┐留出一个 PSP      ┐
8617:    SUB     AX,11H                  ;┘所需的内存       |
8618:    MOV     CX,CS:SizePerBuffer     ;┐                 |
8619:    SHR     CX,1                    ;| 留出一个盘缓冲   | 预确定存放 CONFIG.SYS
8620:    SHR     CX,1                    ;| 区所需的内存     | 文件内容的缓存区的段
8621:    SHR     CX,1                    ;|                 | 地址值
8622:    SHR     CX,1                    ;|                 |
8623:    INC     CX                      ;|                 |
```

```
8624:   SUB     AX,CX                   ;┘                    ┘
8625:   MOV     CS:SEG_Config,AX
8626:   PUSH    ES
8627:   PUSH    DI
8628:   LES     DI,ES:[DI+12H]          ;ES:DI←保存盘缓冲区链头结点指针值的双字单
                                        ;元地址
8629:   MOV     WORD PTR ES:[DI+4],0
8630:   MOV     WORD PTR ES:[DI],0      ;┐设置盘缓冲区双链表头结点的指针值
8631:   MOV     WORD PTR ES:[DI+2],AX   ;┘
8632:   MOV     ES,AX                   ;┐
8633:   XOR     AX,AX                   ;|
8634:   MOV     DI,AX                   ;|
8635:   MOV     ES:[DI],AX              ;| 初始化盘缓冲区双链表头结点的控制块(此时
8636:   MOV     WORD PTR ES:[DI+2],AX   ;| 盘缓冲区双链表只有一个结点,且它为空闲状
8637:   MOV     WORD PTR ES:[DI+4],0FFH ;| 态)
8638:   MOV     WORD PTR ES:[DI+6],0    ;|
8639:   MOV     WORD PTR ES:[DI+8],0    ;┘
8640:   POP     DI
8641:   POP     ES
8642:   PUSH    CS                      ;┐
8643:   POP     DS                      ;| 在常规内存高端为每个逻辑驱动器建立一个
8644:   CALL    Create_CDS              ;┘ CDS
8645:   MOV     DS,CS:Seg  FirstMCB1    ;DS←第一个内存块的段地址
8646:   DB      9AH                     ;┐
8647:   DW      089BH                   ;| 此指令系列无任何作用
8648:   DW      BIO_SEG                 ;|
8649:   ;CALL   FAR PTR 0070:089BH      ;┘
8650:   STI
8651:   CLD                             ;┐
8652:   MOV     BX, CS                  ;|
8653:   SUB     BX, +10H                ;|
8654:   MOV     ES, BX                  ;| 复制当前的程序段前缀 PSP 到常规内存高端
8655:   XOR     SI, SI                  ;|
8656:   MOV     DI, SI                  ;|
8657:   MOV     CX, 0080H               ;|
8658:   REP     MOVSW                   ;┘
8659:   MOV     WORD PTR ES:PSPAddr_36,ES ;修改 PSP 的"文件句柄表"的段地址值
8660:   MOV     AH, 50H                 ;┐ 设定当前活动进程的 PSP 段地址
8661:   INT     21H                     ;┘
8662:   PUSH    DS
8663:   PUSH    CS                      ;┐
8664:   POP     DS                      ;|
8665:   MOV     DX, offset INT24H_ISR   ;| 建立 INT 24H 的中断向量
```

```
8666:    MOV     AX, 2524H                    ;|
8667:    INT     21H                          ;┘
8668:    CMP     LogicDriveFlag,00            ;┐ 若逻辑驱动器未超过26个,则跳转
8669:    JE      SysInt_II_E2_7               ;┘
8670:    MOV     DX,offset Err_LogicalDrive   ;┐ 显示"逻辑驱动器超过26个"的错误提示信息
8671:    CALL    DisplayString                ;┘
8672: SysInt_II_E2_7:
8673:    POP     DS                           ;DS=第一个内存块的段地址值
8674:    MOV     DL,CS:DriveNumber_Boot       ;┐
8675:    OR      DL, DL                       ;| 若不是 DOS 操作系统启动系统而执行
8676:    JZ      SysInt_II_E2_8               ;┘ SysInt—Ⅱ,则跳转
8677:    DEC     DL                           ;┐ 选择 DL 指定的逻辑驱动器作为缺省(当前)的
8678:    MOV     AH, 0EH                      ;| 驱动器(即引导盘作为当前驱动器)
8679:    INT     21H                          ;┘
8680: SysInt_II_E2_8:
8681:    CALL    Process_Config               ;转换 CONFIG.SYS 的文件内容,并识别 dos 系统配
                                              ;置命令
8682:    CMP     CS:DOS_Install_Site,0        ;┐ 若 DOS 全部安装在常规内存中,则跳转
8683:    JE      SysInt_II_E2_9               ;┘
8684:    MOV     ES,WORD PTR CS:Addr_MSDOS+2  ;┐ 修改与 MSDOS2模块相关的变量,以及由
8685:    XOR     AX,AX                        ;| 由 MSDOS2模块定义的子程序和中断服
8686:    CALL    DWORD PTR CS:SubRoutine1_MSDOS ;┘ 服务程序的段地址值
8687:    JMP     SHORT SysInt_II_E2_10
8688: SysInt_II_E2_9:
8689:    XOR     BX,BX                        ;┐ 将 DOS 操作系统的 IO2模块和 MSDOS2模块安
8690:    CALL    Install_IO2_MSDOS2_InCMA     ;┘ 装在常规内存低端
8691:    MOV     AX,1                         ;┐ 修改与 MSDOS2模块相关的变量,以及由
8692:    MOV     ES,WORD PTR CS:Addr_MSDOS+2  ;| MSDOS2模块定义的中断服务程序的段地
8693:    CALL    DWORD PTR CS:SubRoutine1_MSDOS ;┘ 址值
8694: SysInt_II_E2_10:
8695:    CALL    ApplyAllAvailableMem         ;申请全部可用的常规内存,为建立 DOS 的各种数
                                              ;据结构和在常规内存安装设备驱动程序作准备工
                                              ;作
8696:    INC     CS:ProcessConfigFlag         ;┐ 识别(处理)buffers、break、multitrack、device、
                                              ;| devicehigh、country、files、lastdrive、drivparm、
                                              ;| stacks、shell、fcbs 等系统配置命令;并检查是
8697:    CALL    Pro_Config4                  ;┘ 否存在 install 命令
8698:    CALL    ModifyUMBSize                ;修改作为 DOS 数据段的 UMB 内存块的大小
8699:    CALL    SetEndFlagOfCMC              ;设置常规内存控制块链的结束标志
8700:    INC     CS:ProcessConfigFlag         ;┐ 不识别任何系统配置命令
8701:    CALL    Pro_Config4                  ;┘
8702:    CALL    Create_ControlBlock          ;在作为 DOS 数据段的第一个内存块中为 DOS 内
                                              ;核建立各种数据块(如 SFT 表、盘缓冲区循环双链
```

```
                                        ;表、CDS 数组等);建立动态堆栈;以及为 install 系
                                        ;统配置命令的执行作为准备
8703:   MOV     AX,BIO_SEG
8704:   MOV     ES,AX
8705:   MOV     BYTE PTR ES:IO3_Flag,0
8706:   TEST    WORD PTR CS:Install_Flag,1      ;┐若 CONFIG.SYS 文件中没有 install 命令,则跳转
8707:   JZ      SysInt_II_E2_11                 ;┘
8708:   INC     CS:ProcessConfigFlag            ;┐识别 install 命令
8709:   CALL    Pro_Config4                     ;┘
8710: SysInt_II_E2_11:
8711:   CMP     CS:DOS_Install_Site,0FFH        ;┐若 DOS 的 IO2模块和 MSDOS2模块已被安装在
8712:   JNE     SysInt_II_E2_12                 ;┘ HMA 中或常规内存中,则跳转
8713:   CALL    Install_IO2_MSDOS2              ;在 HMA 中安装 IO2模块和 MSDOS2模块
8714: SysInt_II_E2_12:
8715:   CMP     CS:DOS_Install_Site,0           ;┐若 DOS 的 IO2模块和 MSDOS2模块安装在常规
8716:   JE      SysInt_II_E2_13                 ;┘内存中,则跳转
8717:   CALL    CopyDOSFarCallToIO2             ;将 DOS 远调用指令从中断向量表中复制到位于
                                        ;HMA 的 IO2模块中
8718: SysInt_II_E2_13:
8719:   MOV     CS:ErrorCtrlFlag,1              ;置"错误位置信息的输出控制标志"(1:不输出错
                                        ;误位置信息)
8720:   MOV     ES,CS:SEG_AllocatedMemory       ;┐释放由 SysInt—Ⅱ初始化时临时占用的内存块,
8721:   MOV     AH,49H                          ;┃它保存了 CONFIG.SYS 文件内容、CDS 表、IO3
8722:   INT     21H                             ;┘和 IO2模块,以及 MSDOS.SYS 文件内容等
8723:   TEST    WORD PTR CS:Install_Flag,2      ;┐若没有安装支持 install 命令的临时程序段,则跳
8724:   JZ      SysInt_II_E2_14                 ;┘转
8725:   PUSH    ES
8726:   PUSH    BX
8727:   MOV     ES,CS:Seg_FirstMCB2             ;┐
8728:   MOV     BX,CS:SizeOfDDB                 ;┃释放支持 install 命令的临时程序段占用的内存
8729:   MOV     AH,4AH                          ;┃块
8730:   INT     21H                             ;┘
8731:   MOV     AX,ES                           ;┐
8732:   DEC     AX                              ;┃设置作为 DOS 数据段的内存控制块信息(PID
8733:   MOV     ES,AX                           ;┃值、文件名)
8734:   MOV     WORD PTR ES:MCB_ProcessID,8     ;┃
8735:   MOV     WORD PTR ES:[8],' DS'           ;┘
8736:   POP     BX
8737:   POP     ES
8738: SysInt_II_E2_14:
8739:   CMP     CS:DOS_Install_Site,0           ;┐若 DOS 的 IO2模块和 MSDOS2模块安装在常规
8740:   JE      SysInt_II_E2_15                 ;┘内存中,则跳转
8741:   CALL    SetHIMEMMark                    ;建立 HIMEM.SYS 的安装标志
```

```
8742: SysInt_II_E2_15:
8743:     PUSH    CS                      ;┐
8744:     POP     DS                      ; |
8745:     MOV     SI,offset CMDLine+1     ; |
8746:     PUSH    DS                      ; |
8747:     POP     ES                      ; |
8748:     MOV     DI,SI                   ; | 计算传给命令处理器(即 Shell 模块,缺省为
8749:     MOV     CL,-1                   ; | command.com)的命令行参数的字符串长度
8750: SysInt_II_E2_16:                    ; | →CL
8751:     INC     CL                      ; |
8752:     LODSB                           ; |
8753:     STOSB                           ; |
8754:     OR      AL,AL                   ; |
8755:     JNZ     SysInt_II_E2_16         ;┘
8756:     DEC     DI                      ;┐
8757:     MOV     AL,CR                   ; | 设置命令行参数的结束符
8758:     STOSB                           ;┘
8759:     MOV     CMDLine,CL              ;设置命令行参数的长度(字符数)
8760:     MOV     DX, offset Path_Shell   ;DS:DX 指向命令处理器对应的文件名说明串
8761:     PUSH    DX
8762:     MOV     BX,-1                   ;┐
8763:     MOV     AH,48H                  ; | 测试可用的常规内存大小(节单位)
8764:     INT     21H                     ;┘
8765:     MOV     AH,48H                  ;┐ 申请所有可用的常规内存
8766:     INT     21H                     ;┘
8767:     JC      SysInt_II_E2_17
8768:     MOV     ES,AX                   ;┐
8769:     MOV     AH,49H                  ; | 释放刚申请的全部可用的常规内存
8770:     INT     21H                     ;┘
8771:     MOV     BP,BX                   ;BP=当前可用的常规内存的大小(节单位)
8772:     MOV     BX,RAMSize              ;┐
8773:     MOV     AX,CS                   ; |
8774:     SUB     BX,AX                   ; | 还有空闲的常规内存供装入命令处理器?
8775:     ADD     BX,11H                  ; |
8776:     SUB     BP,BX                   ;┘
8777:     JC      SysInt_II_E2_17         ;若不能装入,则跳转
8778:     MOV     AX,3D00H                ;┐
8779:     STC                             ; | 以"读方式"打开命令处理器(缺省为 command.
8780:     INT     21H                     ;┘ com 文件)
8781:     JC      SysInt_II_E2_19
8782:     MOV     BX,AX                   ;┐
8783:     XOR     CX,CX                   ; |
8784:     XOR     DX,DX                   ; | 测试 command.com 文件的字节长度
```

```
8785:    MOV     AX,4202H                    ;|
8786:    STC                                 ;|
8787:    INT     21H                         ;┘
8788:    JC      SysInt_II_E2_19
8789:    ADD     AX,000FH                    ;┐
8790:    ADC     DX,0                        ;|
8791:    CALL    ConvertToParagraphs         ;|
8792:    MOV     CL,0CH                      ;| 计算 command.com 文件的节数
8793:    SHL     DX,CL                       ;|
8794:    OR      AX,DX                       ;┘
8795:    ADD     AX,0010H                    ;加上程序段前缀的节数
8796:    CMP     AX,BP                       ;自由内存能装入 command.com 文件吗?
8797:    JB      SysInt_II_E2_18             ;若能装入,则跳转
8798: SysInt_II_E2_17:
8799:    JMP     MemoryError                 ;转去显示错误信息"Configuration too large for
                                             ;memory"
8800: SysInt_II_E2_18:
8801:    MOV     AH,3EH                      ;┐ 关闭 command.com 文件
8802:    INT     21H                         ;┘
8803:    POP     DX
8804:    PUSH    CS                          ;┐
8805:    POP     ES                          ;|
8806:    MOV     BX, offset EXECPB1          ;| 设定 EXEC 功能调用的参数值
8807:    MOV     [BX].EXEC_CMDLine_SEG,CS    ;|
8808:    MOV     [BX].EXEC_FCB1_SEG,CS       ;|
8809:    MOV     [BX].EXEC_FCB2_SEG,CS       ;┘
8810:    XOR     AX,AX                       ;┐
8811:    MOV     AH,4BH                      ;| 装入并执行 command.com 文件
8812:    STC                                 ;|
8813:    INT     21H                         ;┘
8814: SysInt_II_E2_19:
8815:    MOV     DX, offset CMD_Inter        ;┐ 显示错误信息"Command Interpreter?"
8816:    CALL    Disp_ErrorInfo              ;┘
8817: SysInt_II_E2_20:
8818:    JMP     SHORT SysInt_II_E2_20       ;系统进入死循环
8819: SysInt_II_Entry2 ENDP
8820: ;=============================================================
8821: ;相对地址偏移:089B
8822: ;功能:申请全部可用的常规内存
8823: ;入口参数:无
8824: ;出口参数:无
8825: ;=============================================================
8826: ApplyAllAvailableMem    PROC    NEAR
```

```
8827:     MOV     BX,0FFFFH                    ;┐
8828:     MOV     AH,48H                       ;│ 测试可用的常规内存大小(节数)
8829:     INT     21H                          ;┘
8830:     MOV     AH,48H                       ;┐ 分配全部可用的常规内存
8831:     INT     21H                          ;┘
8832:     MOV     CS:SEG_AllocatedMemory,AX    ;保存作为 DOS 数据段的常规内存块的段地址值
8833:     MOV     CS:SEG_FreeMemory,AX         ;初始化自由内存块的段地址值
8834:     RET
8835: ApplyAllAvailableMem  ENDP
8836: ;
8837: HMANotAvailable   DB    ' HMA not available ; Loading DOS low' , 0DH, 0AH, ' $'
8838: CannotAllocateMem  DB    ' Fatal Error;Cannot allocate Memory for DOS' , 0DH, 0AH, ' $'
8839: ;
8840: ;═══════════════════════════════════════════════
8841: ;相对地址偏移:0902
8842: ;功能:依据先 HMA、后常规内存的原则安装 IO2模块和 MSDOS2模块
8843: ;入口参数:无
8844: ;出口参数:无
8845: ;═══════════════════════════════════════════════
8846: Install_IO2_MSDOS2  PROC  NEAR
8847:     CALL    Install1_IO2_MSDOS2_InHMA    ;试图在 HMA 中安装 IO2模块和 MSDOS2模块
8848:     JC      Install_IO2_MSDOS2_1         ;若不能安装在 HMA 中,则跳转
8849:     RET
8850: Install_IO2_MSDOS2_1:
8851:     PUSH    CS                           ;┐
8852:     POP     DS                           ;│
8853:     MOV     AH,9                         ;│ 显示提示信息"HMA not auailable;Loading DOS
8854:     MOV     DX,offset HMANotAvailable    ;│ low"
8855:     INT     21H                          ;┘
8856:     MOV     BX,1                         ;┐ 在常规内存中安装 IO2模块和 MSDOS2模块
8857:     CALL    Install_IO2_MSDOS2_InCMA     ;┘
8858:     MOV     ES,WORD PTR CS:Addr_MSDOS+2  ;┐ 修改与 MSDOS2模块相关的变量,以及由
8859:     XOR     AX,AX                        ;│ MSDOS2模块定义的子程序和中断服务程序
8860:     CALL    DWORD PTR CS:SubRoutine1_MSDOS;┘ 的段地址值
8861:     MOV     CS:DOS_Install_Site,0        ;清 DOS 安装控制标志(0:DOS 全部安装在常规内
                                               ;存中)
8862:     RET
8863: Install_IO2_MSDOS2      ENDP
8864: ;═══════════════════════════════════════════════
8865: ;相对地址偏移:092A
8866: ;功能:在高内存区(HMA)中安装 IO2模块和 MSDOS2模块,并确定 HMA 的自由内存区的起始地址
8867: ;入口参数:无
8868: ;出口参数:CF=0,成功;CF=1,失败
```

```
8869: ;=========================================================
8870: Install1_IO2_MSDOS2_InHMA        PROC    NEAR
8871:     CALL    Install2_IO2_MSDOS2_InHMA       ;在 HMA 中安装 IO2模块和 MSDOS2模块
8872:     JC      Install1_IO2_MSDOS2_InHMA_RET   ;若安装失败,则结束返回
8873:     MOV     ES,WORD PTR CS:Addr_MSDOS+2     ;┐ 修改与 MSDOS2模块相关的变量,以及由
8874:     XOR     AX,AX                           ;| MSDOS2模块定义的子程序和中断服务程
8875:     CALL    DWORD PTR CS:SubRoutine1_MSDOS  ;┘ 序的段地址值
8876:     MOV     CS:DOS_Install_Site,1           ;设置 IO2模块和 MSDOS2模块在 HMA 的安装标志
8877:     CLC                                     ;CF←0(成功)
8878: Install1_IO2_MSDOS2_InHMA_RET:
8879:     RET
8880: Install1_IO2_MSDOS2_InHMA    ENDP
8881: ;=========================================================
8882: ;相对地址偏移:0943
8883: ;功能:在高内存区(HMA)中安装 IO2模块和 MSDOS2模块,并确定 HMA 的自由内存区的起始地址
8884: ;入口参数:无
8885: ;出口参数:CF=0,成功;CF=1,失败
8886: ;=========================================================
8887: Install2_IO2_MSDOS2_InHMA        PROC    NEAR
8888:     CALL    ChkHMA                          ;检查 HMA 是否存在,以及是否允许存取它
8889:     JC      Install2_IO2_MSDOS2_InHMA_RET   ;若不允许存取 HMA,则直接返回
8890:     MOV     AX,0FFFFH                       ;┐
8891:     MOV     ES,AX                           ;| 将 IO2模块安装在 HMA 中(即扩充内存的起始
8892:     CALL    Install_IO2                     ;┘ 64KB)
8893:     MOV     CX,CS:SizeOfMSDOS2InHMA         ;┐ 将 MSDOS2模块安装在 HMA 中
8894:     CALL    Install_MSDOS2                  ;┘
8895:     CALL    Set_OFS_FreeHMA                 ;确定 HMA 的自由内存区的起始地址
8896:     CLC
8897: Install2_IO2_MSDOS2_InHMA_RET:
8898:     RET
8899: Install2_IO2_MSDOS2_InHMA        ENDP
8900: ;=========================================================
8901: ;相对地址偏移:095D
8902: ;功能:在常规内存中安装 IO2模块和 MSDOS2模块
8903: ;入口参数:BX=IO2模块和 MSDOS2模块的安装标志(0:MS-DOS 启动时安装;≠0:非 MS-DOS
                启动时安装)
8904: ;出口参数:无
8905: ;=========================================================
8906: Install_IO2_MSDOS2_InCMA        PROC    NEAR
8907:     CALL    Seg_IO2_MSDOS2_InLoad           ;┐ ES←IO2模块和 MSDOS 模块在常规内存中的
8908:     MOV     ES,AX                           ;┘ 装入段地址
8909:     CALL    Install_IO2                     ;在常规内存中安装 IO2模块
8910:     MOV     CX,CS:SizeOfMSDOS2InCMA         ;┐ 在常规内存中安装 MSDOS2模块
```

```
8911:     CALL    Install_MSDOS2                    ;┘
8912:     RET
8913: Install_IO2_MSDOS2_InCMA          ENDP
8914: ;=============================================================
8915: ;相对地址偏移:096E
8916: ;功能:安装 IO2模块
8917: ;入口参数:ES=IO2模块的装入段地址值
8918: ;出口参数:无
8919: ;=============================================================
8920: Install_IO2   PROC    NEAR
8921:     MOV     DS,WORD PTR CS:SubRoutine_IO2+2;┐ DS:SI 指向 IO2模块的有效内容的起始位置
8922:     MOV     SI,30H                          ;┘
8923:     MOV     DI,SI                           ;ES:DI 指向 IO2模块最终在内存中的装入位置
8924:     MOV     CX,1A60H                        ;┐
8925:     SUB     CX,SI                           ;| 将 IO2模块安装到最终位置
8926:     SHR     CX,1                            ;|
8927:     REP     MOVSW                           ;┘
8928:     PUSH    ES
8929:     PUSH    DI
8930:     MOV     AX,ES                           ;┐ 保存 IO2模块最终在内存中的段地址值
8931:     MOV     CS:Seg_IO2,AX                   ;┘
8932:     CALL    DWORD PTR CS:SubRoutine_IO2     ;修改由 IO2模块定义的子程序入口的段地址值(这
                                                  ;些子程序被 IO1模块调用)
8933:     POP     DI
8934:     POP     ES
8935:     RET
8936: Install_IO2   ENDP
8937: ;=============================================================
8938: ;相对地址偏移:0991
8939: ;功能:安装 MSDOS2模块
8940: ;入口参数:CX=MSDOS2模块的大小(字节数)
8941: ;          ES:DI 指向 MSDOS2模块的装入位置
8942: ;出口参数:无
8943: ;=============================================================
8944: Install_MSDOS2   PROC   NEAR
8945:     PUSH    ES
8946:     PUSH    DI
8947:     LDS     SI,DWORD PTR CS:Addr_MSDOS ;┐ 将 MSDOS2模块安装到最终位置
8948:     REP     MOVSB                      ;┘
8949:     POP     BX                          ;┐
8950:     MOV     AX,WORD PTR CS:Addr_MSDOS   ;|
8951:     SUB     AX,BX                       ;| 变换 MSDOS2模块的起始地址格式(这是因
8952:     CALL    ConvertToParagraphs         ;| 为 MSDOS2模块是按起始地址偏移3DD0H 编
```

```
8953:     POP     BX                                  ;| 程的)
8954:     SUB     BX,AX                               ;|
8955:     MOV     WORD PTR CS:Addr_MSDOS+2,BX ;┘
8956:     RET
8957: Install_MSDOS2  ENDP
8958: ;=============================================================
8959: ;相对地址偏移:09AD
8960: ;功能:确定 IO2模块和 MSDOS2模块在常规内存中的装入段地址。当内存不够时,给出错误信息并
          停机
8961: ;入口参数:BX=IO2模块和 MSDOS2模块的安装标志(0:MS-DOS 启动时安装,≠0:非 MS-DOS
          启动时安装)
8962: ;出口参数:AX=IO2模块和 MSDOS2模块的装入段地址
8963: ;=============================================================
8964: Seg_IO2_MSDOS2_InLoad  PROC    NEAR
8965:     MOV     AX,1A60H                            ;AX←IO2模块的长度(字节数)
8966:     SUB     AX,30H                              ;AX=IO2模块的有效内容的字节数
8967:     ADD     AX,CS:SizeOfMSDOS2InCMA             ;加上 MSDOS2模块的长度(字节数)
8968:     ADD     AX,0FH                              ;┐ 计算 IO2模块和 MSDOS2模块占用的总节数
8969:     CALL    ConvertToParagraphs                 ;┘ →AX
8970:     OR      BX,BX
8971:     MOV     BX,AX
8972:     JZ      Seg_IO2_MSDOS2_InLoad_1             ;若 IO2模块和 MSDOS2模块安装在常规内存低端,
                                                      ;则跳转
8973:     MOV     AH,48H                              ;┐ 申请安装 IO2模块和 MSDOS2模块所需的内存
8974:     INT     21H                                 ;┘ 空间
8975:     JC      Seg_IO2_MSDOS2_InLoad_2             ;若没有足够的内存,则跳转
8976:     SUB     AX,3                                ;┐ ES←IO2模块和 MSDOS2模块的装入段地址(减
                                                      ;| "3"是因为 IO2模块是从地址偏移30H 开始编
8977:     MOV     ES,AX                               ;┘ 程的)
8978:     MOV     WORD PTR ES:[0021H],8               ;┐ 重新设置内存控制块的参数值(PID 值、文件名)
8979:     MOV     WORD PTR ES:[0028H],' CS'           ;┘
8980:     RET
8981: Seg_IO2_MSDOS2_InLoad_1:                        ;BX=IO2模块和 MSDOS2模块占用的总节数
8982:     PUSH    DS
8983:     PUSH    DI
8984:     PUSH    CX
8985:     PUSH    DX
8986:     LDS     DI,CS:Address_ListOfList            ;┐
8987:     DEC     DI                                  ;| ES←第一个内存控制块的段地址值
8988:     DEC     DI                                  ;|
8989:     MOV     ES,[DI]                             ;┘
8990:     MOV     CX,WORD PTR ES:[0003]               ;CX←第一个内存块的大小(节数)
8991:     CMP     CX,BX                               ;┐ 若没有足够内存装入 DOS 操作系统,则跳转
```

```
8992:    JB      Seg_IO2_MSDOS2_InLoad_2          ;┘
8993:    MOV     DL,BYTE PTR ES:[0]               ;DL←第一个内存块的标志符
8994:    MOV     AX,ES                            ;┐ 修改第一个内存控制块的段地址,此时给 IO2模
8995:    ADD     AX,BX                            ;│ 块和 MSDOS2模块留出安装时所需的内存空间
8996:    MOV     [DI],AX                          ;┘
8997:    MOV     DS,AX                            ;┐
8998:    MOV     BYTE PTR DS:[0],DL               ;│
8999:    MOV     WORD PTR DS:[1],0                ;│ 重新设置第一个内存控制块的参数值
9000:    SUB     CX,BX                            ;│
9001:    MOV     WORD PTR DS:[3],CX               ;┘
9002:    MOV     AX,ES                            ;┐ AX←IO2模块和 MSDOS2模块安装在常规内存
                                                  ;│ 低端时的段地址值(减"3"是因为 IO2模块是从
9003:    SUB     AX,3                             ;┘ 地址偏移30H 开始编程的)
9004:    POP     DX
9005:    POP     CX
9006:    POP     DI
9007:    POP     DS
9008:    RET
9009: Seg_IO2_MSDOS2_InLoad_2:
9010:    PUSH    CS                               ;┐
9011:    POP     DS                               ;│ 显示错误信息"Fatal Error: Can not allocate
9012:    MOV     DX,offset CannotAllocateMem      ;│ Memory for DOS"
9013:    MOV     AH,9                             ;│
9014:    INT     21H                              ;┘
9015:    CLI                                      ;关中断
9016:    HLT                                      ;停机
9017: Seg_IO2_MSDOS2_InLoad  ENDP
9018: ;=============================================================
9019: ;相对地址偏移:0A26
9020: ;功能:检查高内存区是否存在,以及是否允许存取它
9021: ;入口参数:无
9022: ;出口参数:CF=0,允许存取 HMA;CF=1,不允许存取 HMA 或 HMA 根本就不存在
9023: ;=============================================================
9024: ChkHMA  PROC  NEAR
9025:    PUSH    DS
9026:    MOV     AX,BIO_SEG                       ;┐ DS←IO1模块在常规内存低端的段地址值
9027:    MOV     DS,AX                            ;┘
9028:    CALL    GetInstallStateOfHIMEM           ;取 HIMEM.SYS 的安装状态
9029:    JNZ     ChkHMA_2                         ;若 HIMEM.SYS 未安装,则跳转
9030:    MOV     AX,4310H                         ;┐ 取 HIMEM.SYS 提供的扩充内存管理程序的入
9031:    INT     2FH                              ;┘ 口地址
9032:    MOV     WORD PTR HIMEMAddr,BX            ;┐ 保存 HIMEM.SYS 提供的扩充内存管理程序的
9033:    MOV     WORD PTR HIMEMAddr+2,ES          ;┘ 入口地址
```

```
9034:     MOV     AH,1                        ;┐
9035:     MOV     DX,0FFFFH                   ;├ 请求 HMA
9036:     CALL    DWORD PTR HIMEMAddr         ;┘
9037:     DEC     AX                          ;┐ 若允许存取 HMA,则跳转
9038:     JZ      ChkHMA_1                    ;┘
9039:     MOV     AH,88H                      ;┐ 取扩充内存大小(AX=以节为单位的扩充内存
9040:     INT     15H                         ;┘ 容量)
9041:     CMP     AX,40H                      ;┐ 若扩充内存<64KB,则跳转
9042:     JB      ChkHMA_2                    ;┘
9043: ChkHMA_1:
9044:     MOV     AH,5                        ;┐ 使能 A20地址线
9045:     CALL    DWORD PTR HIMEMAddr         ;┘
9046:     DEC     AX                          ;┐ 若使能 A20地址线失败,则跳转
9047:     JNZ     ChkHMA_2                    ;┘
9048:     CALL    ChkHIMEMMark                ;检查 HIMEM.SYS 是否安装
9049:     JZ      ChkHMA_2                    ;若已安装,则跳转
9050:     MOV     AX,0FFFFH                   ;┐ 通过对 HMA 进行存取访问来判定系统中是否
9051:     MOV     ES,AX                       ;├ 有扩充内存或 CPU 是否为80286、80386或80486
9052:     MOV     WORD PTR ES:[10H],1234H     ;├ 等高档芯片.若没有扩充内存或 CPU 为8086/
9053:     CMP     WORD PTR ES:[10H],1234H     ;├ 8088,则跳转
9054:     JNE     ChkHMA_2                    ;┘
9055:     CLC
9056:     POP     DS
9057:     RET
9058: ChkHMA_2:
9059:     STC
9060:     POP     DS
9061:     RET
9062: ChkHMA   ENDP
9063: ;=============================================================
9064: ;相对地址偏移:0A7C
9065: ;功能:取 HIMEM.SYS 的安装状态
9066: ;入口参数:无
9067: ;出口参数:ZF=0,HIMEM.SYS 未安装;CF=1,HIMEM.SYS 已安装
9068: ;=============================================================
9069: GetInstallStateOfHIMEM   PROC   NEAR
9070:     MOV     AX,4300H                    ;┐ 取 HIMEM.SYS 的安装状态
9071:     INT     2FH                         ;┘
9072:     CMP     AL,80H                      ;判定 HIMEM.SYS 是否已安装
9073:     RET
9074: GetInstallStateOfHIMEM   ENDP
9075: ;=============================================================
9076: ;相对地址偏移:0A84
```

```
9077:  ;功能:该子程序由 IO2模块调用,它试图在 HMA 中安装 IO2模块和 MSDOS2模块,然后确定 HMA
           的自由内存区的起始地址
9078:  ;入口参数:无
9079:  ;出口参数:无
9080:  ;=============================================================
9081:  CalledByIO2  PROC  FAR
9082:      PUSH    AX
9083:      PUSH    BX
9084:      PUSH    CX
9085:      PUSH    DX
9086:      PUSH    SI
9087:      PUSH    DI
9088:      PUSH    DS
9089:      PUSH    ES
9090:      CMP     CS:DOS_Install_Site,-1          ;┐若 DOS 的 IO2模块和 MSDOS2模块安装在常规
9091:      JNE     CalledByIO2_1                   ;┘内存中或已被安装在 HMA 中,则跳转
9092:      CALL    Install1_IO2_MSDOS2_InHMA       ;在 HMA 中安装 IO2模块和 MSDOS2模块,并确定
                                                   ;HMA 的自由内存区的起始地址
9093:  CalledByIO2_1:
9094:      POP     ES
9095:      POP     DS
9096:      POP     DI
9097:      POP     SI
9098:      POP     DX
9099:      POP     CX
9100:      POP     BX
9101:      POP     AX
9102:      RETF
9103:  CalledByIO2  ENDP
9104:  ;相对地址偏移:0AA0
9105:  ;功能:虚拟的 VDISK.SYS 设备驱动程序标题和内存使用的控制信息,它确保后续的 VDISK.SYS 能
           正常安装
9106:  VDisk_Mark    DD    0                       ;┐
9107:               DW    8000H                   ;├虚拟的设备驱动程序标题
9108:               DW    0                       ;│
9109:               DW    0                       ;│
9110:               DB    1, 0, 0, 0, 0, 0, 0, 0  ;┘
9111:               DB    ' VDISK V3.3'           ;虚拟磁盘管理程序(VDISK.SYS)的安装标志
9112:               DB    15 DUP (0)              ;
9113:               DB    0, 0, 11H               ;24位表示的终止地址(字节单位,110000H 表
                                                   ;示1.64MB)
9114:               DB    0EAH                    ;JMP XXXX:YYYY 的指令┐
                                                   ;机器码              ├链接先前的 INT
```

```
9115:                DW    0                          ;┐ 先前的 INT 19H 中断向 ┃ 13H 中断服务
9116:                DW    0                          ;┘ 量                  ┘ 程序
9117: ;相对地址偏移:0AD4
9118: ;功能:扩充内存管理程序(HIMEM.SYS)的安装标志和内存使用的控制信息
9119: HIMEM_Mark     DB    0, 0, 0
9120:                DB    ' VDISK3.3'                ;扩充内存管理程序(HIMEM.SYS)的安装标志
9121:                DB    80H, 0, 1, 1, 0, 1
9122:                DW    40H                        ;使用的扩充内存大小(以 KB 为单位)
9123:                DB    0, 2, 0FEH, 6
9124:                DB    0, 8, 0, 1, 0, 0, 0
9125:                DW    440H                       ;16位表示的终止地址(以 KB 为单位,440H 表
                                                      ;示1.64MB)
9126: ;=============================================================
9127: ;相对地址偏移:0AF4
9128: ;功能:建立 HIMEM.SYS 的安装标志(或 IO2模块和 MSDOS2模块安装在 HMA 中的标志)
9129: ;入口参数:无
9130: ;出口参数:无
9131: ;=============================================================
9132: SetHIMEMMark PROC  NEAR
9133:     XOR     AX,AX                              ;┐
9134:     MOV     DS,AX                              ;┃
9135:     MOV     AX,DS:Vector_19H_OFS               ;┃ 设置 INT 19H 中断向量
9136:     MOV     WORD PTR CS:VDisk_Mark+30H,AX      ;┃
9137:     MOV     AX,DS:Vector_19H_SEG               ;┃
9138:     MOV     WORD PTR CS:VDisk_Mark+32H,AX      ;┘
9139:     MOV     AH,48H                             ;┐
9140:     MOV     BX,4                               ;┃ 申请64个字节的内存块
9141:     INT     21H                                ;┘
9142:     DEC     AX                                 ;┐
9143:     MOV     ES,AX                              ;┃
9144:     MOV     WORD PTR ES:MCB_ProcessID,8        ;┃ 修改内存控制块的 PID 值(8表示 DOS 内
9145:     MOV     WORD PTR ES:[8],' CS'              ;┃ 核使用)并修改内存控制块的文件名
9146:     INC     AX                                 ;┃ (' SC' 表示系统代码)
9147:     MOV     ES,AX                              ;┘
9148:     CLI                                        ;关中断
9149:     MOV     WORD PTR DS:Vector_19H_OFS,2FH     ;┐ 重新设置 INT   ┐ 建立虚拟的 VDISK.
9150:     MOV     DS:Vector_19H_SEG,AX               ;┘ 19H 的中断向量 ┃ SYS 设备驱动程序
9151:     MOV     CX,34H                             ;                 ┃ 标题和内存使用的
9152:     MOV     SI,offset VDisk_Mark               ;                 ┃ 控制信息,以便确保
9153:     XOR     DI,DI                              ;                 ┃ 扩充内存的起始64KB
9154:     PUSH    CS                                 ;                 ┃ 内存不被 VDISK.SYS
9155:     POP     DS                                 ;                 ┃ 使用(因为这个64KB
9156:     CLD                                        ;                 ┃ 的高内存区由 DOS 内
```

```
9157:    REP     MOVSB                          ;                    ┘核使用)
9158:    STI
9159:    PUSH    DI                             ;┐
9160:    PUSH    ES                             ;|
9161:    MOV     AX,0FFFFH                      ;|
9162:    MOV     ES,AX                          ;|
9163:    MOV     DI,10H                         ;| 在 HMA 的起始位置设立 HIMEM.SYS 的
9164:    MOV     CX,20H                         ;| 安装标志
9165:    MOV     SI,offset HIMEM_Mark           ;|
9166:    REP     MOVSB                          ;|
9167:    POP     DI                             ;|
9168:    POP     ES                             ;┘
9169:    RET
9170: SetHIMEMMark  ENDP
9171: ;相对地址偏移:0B4E
9172: ;当通过 ROM BIOS 提供的 INT 15H 中断服务来使用扩充内存时,它必须工作在保护方式下,为此
      必须建立一个全局描述符表(GDT),它由6个描述符组成,每个描述符占用8个字节,如下所示:
9173: GDT     DB     8 DUP (0)                ;BIOS 要求的伪描述符(全0)
9174:    DB     8 DUP (0)                      ;BIOS 要求的伪描述符(全0)
9175:    DW     -1                             ;段内范围              ┐
9176: Source_Addr_LW  DW  0                    ;┐ 24位物理基地址       |
9177: Source_Addr_HB  DB  0                    ;┘                     | 源段描述符
9178:    DB     93H                            ;访问权字节            |
9179:    DW     0                              ;保留                  ┘
9180:    DW     -1                             ;段内范围              ┐
9181:    DW     0                              ;┐ 24位物理基地址(1MB)  |
9182:    DB     10H                            ;┘                     | 目的段描这符
9183:    DB     93H                            ;访问权字节            |
9184:    DW     0                              ;保留                  ┘
9185:    DB     8 Dup(0)                       ;BIOS 要求的伪描述符(全0)
9186:    DB     8 Dup(0)                       ;BIOS 要求的伪描述符(全0)
9187: SourceData  DW  16 Dup(0)                ;源段的数据缓冲区
9188: ;=============================================================
9189: ;相对地址偏移:0B9E
9190: ;功能:清除扩充内存的起始32个字节,即清除 HIMEM.SYS 的安装标志
9191: ;入口参数:无
9192: ;出口参数:无
9193: ;=============================================================
9194: ClearHIMEMMark  PROC  NEAR
9195:    PUSH    ES
9196:    MOV     AX,CS                          ;┐
9197:    MOV     DX,AX                          ;|
9198:    MOV     CL,0CH                         ;|
```

```
9199:    SHR     DX,CL                    ;| 计算"源段数据缓冲区"的24位物理基地址
9200:    MOV     CL,4                     ;| →DL:AX
9201:    SHL     AX,CL                    ;|
9202:    ADD     AX,offset SourceData     ;|
9203:    ADC     DL,0                     ;┘
9204:    MOV     CS:Source_Addr_LW,AX     ;┐ 设置"源段数据缓冲区"的24位物理基地址值
9205:    MOV     CS:Source_Addr_HB,DL     ;┘
9206:    MOV     CX,10H                   ;CX←传送数据的字数
9207:    PUSH    CS                       ;┐
9208:    POP     ES                       ;| ES:SI 指向全局描述符
9209:    MOV     SI,offset GDT            ;┘
9210:    MOV     AH,87H                   ;┐ 将常规内存中的32个字节数据传送到扩充内
9211:    INT     15H                      ;┘ 存的起始位置,清除 HIMEM.SYS 的安装状态
9212:    POP     ES
9213:    RET
9214: ClearHIMEMMark  ENDP
9215: ;=============================================================
9216: ;相对地址偏移:0BC8
9217: ;功能:确定 HMA 的自由内存区的起始地址
9218: ;入口参数:ES:DI 指向 HMA 的自由内存区
9219: ;出口参数:无
9220: ;=============================================================
9221: Set_OFS_FreeHMA PROC  NEAR
9222:    MOV     BX,ES                    ;┐
9223:    MOV     AX,0FFFFH                ;|
9224:    SUB     AX,BX                    ;|
9225:    ADD     DI,0FH                   ;| 计算 HMA 的自由内存区的起始地址偏移值
9226:    AND     DI,0FFF0H                ;|
9227:    MOV     CL,4                     ;|
9228:    SHL     AX,CL                    ;|
9229:    SUB     DI,AX                    ;┘
9230:    PUSH    DS
9231:    MOV     AX,BIO_SEG               ;┐ DS 指向 IO1模块
9232:    MOV     DS,AX                    ;┘
9233:    MOV     FreeMemOFSofHMA,DI       ;设置 HMA 的自由内存区的起始地址偏移值
9234:    MOV     AllowHMA,0FFH            ;设置 IO1模块中的变量(已确定 HMA 的自由内存
                                          ;区的起始地址偏移值)
9235:    POP     DS
9236:    RET
9237: Set_OFS_FreeHMA  ENDP
9238: ;=============================================================
9239: ;相对地址偏移:0BEC
9240: ;功能:检查扩充内存管理程序(HIMEM.SYS)的安装标志
```

```
;入口参数:无
;出口参数:ZF=0,HIMEM.SYS 未安装;ZF=1,HIMEM.SYS 已被安装
;=============================================================
ChkHIMEMMark   PROC   NEAR
    XOR     AX,AX                       ;┐
    MOV     DS,AX                       ;|
    MOV     DS,DS:Vector_13H_SEG        ;|
    MOV     SI,12H                      ;|
    MOV     CX,5                        ;| 通过 INT 13H 中断向量来检查扩充内存管理程
    PUSH    CS                          ;| 序的安装状态
    POP     ES                          ;|
    MOV     DI,offset VDisk_Mark + 18   ;|
    REPE    CMPSB                       ;┘
    JZ      ChkHIMEMMark_RET            ;若已安装,则跳转
    MOV     AX,0FFFFH                   ;┐
    MOV     DS,AX                       ;|
    MOV     SI,13H                      ;| 检查扩充内存管理程序的安装标志
    MOV     DI,offset HIMEM_Mark + 3    ;|
    MOV     CX,5                        ;|
    REPE    CMPSB                       ;┘
ChkHIMEMMark_RET:
    RET
ChkHIMEMMark   ENDP
;=============================================================
;相对地址偏移:0C14
;功能:将保存在中断向量表中(占用0:00C0H~0:00C3H)的 DOS 远调用指令(如 JMP 0129:109E)
      复制到 IO2模块中
;入口参数:无
;出口参数:无
;=============================================================
CopyDOSFarCallToIO2     PROC   NEAR
    PUSH    DS
    MOV     CX,0FFFFH
    MOV     ES,CX
    XOR     CX,CX
    MOV     DS,CX
    MOV     SI,Vector_30H_OFS
    MOV     DI,offset IO2_00D0
    MOV     CX,5
    CLD
    REP     MOVSB
    RET
CopyDOSFarCallToIO2   ENDP
```

```
9284: ;=============================================================
9285: ;相对地址偏移:0C2C
9286: ;功能:AX/16→AX
9287: ;入口参数:AX=地址偏移值
9288: ;出口参数:AX=相对段地址值
9289: ;=============================================================
9290: ConvertToParagraphs  PROC  NEAR
9291:     SHR     AX,1
9292:     SHR     AX,1
9293:     SHR     AX,1
9294:     SHR     AX,1
9295:     RET
9296: ConvertToParagraphs  ENDP
9297: ;=============================================================
9298: ;相对地址偏移:0C35
9299: ;功能:为系统中每个逻辑驱动器建立一个当前目录路径结构(CDS)
9300: ;入口参数:无
9301: ;出口参数:无
9302: ;=============================================================
9303: Create_CDS  PROC  NEAR
9304:     LES     DI,Address  ListOfList          ;ES:DI 指向位于 MSDOS1模块中的多重表
9305:     MOV     CL,ES:[DI+20H]                  ;┐ CX=系统中的逻辑驱动器数
9306:     XOR     CH,CH                           ;┘
9307:     MOV     ES:[DI+21H],CL                  ;补缺系统中允许存取的逻辑驱动器数
9308:     MOV     AL,CL                           ;┐
9309:     MOV     AH,58H                          ;│ AX=逻辑驱动器的 CDS 数组占用内存的字节
9310:     MUL     AH                              ;┘ 数
9311:     CALL    Convert_Addr                    ;计算 CDS 数组占用内存的节数
9312:     MOV     SI,SEG_Config                   ;┐
9313:     SUB     SI,AX                           ;│ 确定并保存 CDS 数组在常规内存高端的段地址
9314:     MOV     Seg_CDSArray,SI                 ;┘ 值
9315:     MOV     ES:[DI+18H],SI                  ;┐
9316:     MOV     AX,SI                           ;│ 设置多重表的参数值(即 CDS 数组的起始地址)
9317:     MOV     WORD PTR ES:[DI+16H],0          ;┘
9318:     LDS     SI,DWORD PTR ES:[DI]            ;DS:SI 指向第一个 DOS 驱动器参数块(DDPB),即
                                                  ;DDPB 链的头结点
9319:     MOV     ES,AX                           ;┐ ES:DI=CDS 数组的起始地址
9320:     XOR     DI,DI                           ;┘
9321: ;相对地址偏移:0C69
9322: Init_CDS    LABEL  NEAR
9323:     MOV     AX,WORD PTR CS:Rootpath_CDS     ;┐
9324:     STOSW                                   ;│ 填 CDS 中的"当前目录路径说明串"为根
9325:     MOV     AX,WORD PTR CS:Rootpath_CDS+2   ;│ 目录
```

```
9326:     STOSW                                    ;┘
9327:     INC     BYTE PTR CS:Rootpath_CDS         ;修改逻辑驱动器符
9328:     XOR     AX,AX                            ;┐
9329:     PUSH    CX                               ;│初始化子目录路径串为空
9330:     MOV     CX,003FH                         ;│
9331:     REP     STOSB                            ;┘
9332:     CMP     SI,-1                            ;┐若是建立最后一个逻辑驱动器的 CDS,则跳转
9333:     JE      CreateCDS1                       ;┘
9334:     CMP     CS:FDI_Flag_IO3,01               ;┐若有软盘驱动器,则跳转
9335:     JNE     CreateCDS2                       ;┘
9336:     CMP     BYTE PTR [SI],01                 ;┐若当前逻辑驱动器对应于硬盘,则跳转
9337:     JA      CreateCDS2                       ;┘
9338:     MOV     CL,3                             ;┐初始化 CDS 的"状态字、DDPB 指针域"
9339:     REP     STOSW                            ;┘
9340:     POP     CX
9341:     JMP     SHORT CreateCDS4
9342: CreateCDS1:
9343:     MOV     CL,3                             ;┐初始化 CDS 的"状态字、DDPB 指针域"
9344:     REP     STOSW                            ;┘
9345:     POP     CX
9346:     JMP     SHORT CreateCDS5
9347: CreateCDS2:
9348:     POP     CX
9349:     CMP     BYTE PTR [SI+8],0                ;┐若 DDPB 的"每个 FAT 表占用的扇区数"字段值
9350:     JE      CreateCDS3                       ;┘为0,则跳转
9351:     MOV     AX,4000H                         ;设置逻辑驱动器为本地设备
9352: CreateCDS3:
9353:     STOSW                                    ;状态字值→CDS
9354:     MOV     AX,SI                            ;┐
9355:     STOSW                                    ;│当前逻辑驱动器的 DDPB 的起始地址→CDS
9356:     MOV     AX,DS                            ;│
9357:     STOSW                                    ;┘
9358: CreateCDS4:
9359:     LDS     SI,DWORD PTR [SI+19H]            ;DS:SI 指向下一个逻辑驱动器的 DDPB
9360: CreateCDS5:
9361:     MOV     AX, -1
9362:     STOSW                                    ;┐设置 CDS 的"重定向记录"指针域
9363:     STOSW                                    ;┘
9364:     STOSW
9365:     MOV     AX,2                             ;┐逻辑驱动器符号名的字符个数→CDS
9366:     STOSW                                    ;┘
9367:     MOV     AL,0
9368:     STOSB
```

```
9369:    STOSW                                  ;┐ 清 CDS 的 IFS 驱动程序的入口地址
9370:    STOSW                                  ;┘
9371:    STOSW
9372:    LOOP    Init_CDS                       ;转去初始化其它逻辑驱动器的 CDS
9373:    MOV     BYTE PTR CS:Rootpath_CDS,'A';设定第一个逻辑驱动器符
9374:    RET
9375: Create_CDS  ENDP
9376: ;=============================================================
9377: ;相对地址偏移:00CCD
9378: ;功能:在作为 DOS 数据段的第一个内存块中为 DOS 内核建立各种数据块(如 SFT 表数组、盘缓冲
          区的循环双链表、CDS 数组、动态堆栈等);建立 DOS 保留的5个文件句柄;并为 install 系统配
          置命令的执行作为准备工作
9379: ;入口参数:无
9380: ;出口参数:无
9381: ;=============================================================
9382: Create_ControlBlock  PROC  NEAR
9383:    PUSH    DS
9384:    MOV     AX,BIO_SEG                     ;┐ DS←IO1模块在常规内存低端的段地址值
9385:    MOV     DS,AX                          ;┘
9386:    CMP     MultiTrack,0                   ;┐ 若 CONFIG.SYS 文件中有 multitrack 系统配置
9387:    JNE     Create_CB_1                    ;┘ 命令,则跳转
9388:    OR      MultiTrack,0080H               ;补缺 multitrack 检查状态为"ON"
9389: Create_CB_1:
9390:    POP     DS
9391:    MOV     AX,CS:SEG_Config               ;┐ 修改自由内存的最大段地址
9392:    MOV     CS:Seg_CDSArray,AX             ;┘
9393:    PUSH    CS                             ;┐
9394:    POP     DS                             ;├ 转换自由内存的起始地址值格式
9395:    CALL    Convert_AddrFMT                ;┘
9396:    MOV     AL,CS:NumberOfHandle           ;AL=files 系统配置命令设定的"能同时打开的最
                                                ;大文件数"(即对应于文件句柄的 SFT 表个数)
9397:    SUB     AL,5                           ;减去5个保留的文件数(SFT 表个数),此时 AL=
                                                ;还需要建立的 SFT 表个数
9398:    JBE     Create_CB_2                    ;若不增加 SFT 表,则跳转
9399:    PUSH    AX                             ;┐
9400:    MOV     AL,'F'                         ;├ 为新增加的 SFT 表数组建立对应的子段控制块
9401:    CALL    CreateSCB                      ;├
9402:    POP     AX                             ;┘
9403:    XOR     AH,AH                          ;AX=新增加的 SFT 表数目
9404:    MOV     BX,CS:OFS_FreeMemory           ;┐ DX:BX←新增加的 SFT 表数组在常规内存中
9405:    MOV     DX,CS:SEG_FreeMemory           ;┘ 的起始地址
9406:    LDS     DI,CS:Address_ListOfList       ;┐ DS:DI 指向原来(保留)的 SFT 表数组控制块的
9407:    LDS     DI,DWORD PTR [DI+4]            ;┘ 起始位置
```

```
9408:    MOV    [DI],BX                          ;┐ 将新增加的 SFT 表数组控制块链入 SFT 表数组
9409:    MOV    [DI+2],DX                        ;┘ 的控制块链中
9410:    PUSH   CS                                  ;┐
9411:    POP    DS                                  ;├ 设置尾结点的指针域值
9412:    LES    DI,DWORD PTR CS:OFS_FreeMemory      ;├
9413:    MOV    WORD PTR ES:[DI],-1                 ;┘
9414:    MOV    ES:[DI+4],AX                     ;SFT 表个数→SFT 表数组的控制块
9415:    MOV    BL,3BH                           ;┐
9416:    MUL    BL                               ;├
9417:    MOV    CX,AX                            ;├ 计算新增加的 SFT 表数组占用的字节数,并修
9418:    ADD    CS:OFS_FreeMemory,AX             ;├ 改自由内存的起始地址
9419:    MOV    AX,6                             ;├
9420:    ADD    CS:OFS_FreeMemory,AX             ;┘
9421:    OR     CS:FlagOfSCB,02                  ;┐ 转换自由内存的起始地址值格式,并设置子段
9422:    CALL   Convert_AddrFMT                  ;┘ 的大小
9423:    ADD    DI,AX                            ;┐
9424:    XOR    AX,AX                            ;├ 初始化所有新增加的 SFT 表
9425:    REP    STOSB                            ;┘
9426: Create_CB_2:
9427:    PUSH   CS                               ;┐
9428:    POP    DS                               ;├ 转换自由内存的起始地址值格式
9429:    CALL   Convert_AddrFMT                  ;┘
9430:    MOV    AL,' X'                          ;┐ 为 FCB 对应的 SFT 表数组建立对应的子段控制
9431:    CALL   CreateSCB                        ;┘ 块
9432:    MOV    AL,CS:NumberOfFCB                ;┐ AX 为 fcbs=m,n 的 m 参数值
9433:    XOR    AH,AH                            ;┘
9434:    MOV    BX,CS:OFS_FreeMemory             ;┐ DX:BX←FCB 对应的 SFT 表数组在常规内存中
9435:    MOV    DX,CS:SEG_FreeMemory             ;┘ 的起始地址
9436:    LDS    DI,CS:Address_ListOfList         ;DS:DI 指向多重表
9437:    MOV    [DI+1AH],BX                      ;┐ FCB 对应的 SFT 表数组的控制块链头结点的起
9438:    MOV    [DI+1CH],DX                      ;┘ 始地址→多重表
9439:    MOV    BL,CS:NumberOfKeepFCB            ;┐
9440:    XOR    BH,BH                            ;├ fcbs=m,n 的 n 参数值(在 MS—DOS 5.00中,该
9441:    MOV    [DI+1EH],BX                      ;┘ 值固定为0)→多重表
9442:    PUSH   CS                               ;┐
9443:    POP    DS                               ;├ 设置 FCB 对应的 SFT 表数组的控制块链尾结点
9444:    LES    DI,DWORD PTR OFS_FreeMemory      ;├ 指针域
9445:    MOV    WORD PTR ES:[DI],-1              ;┘
9446:    MOV    ES:[DI+4],AX                     ;FCB 对应的 SFT 表的个数→SFT 表数组的控制块
9447:    MOV    BL,3BH                           ;┐
9448:    MOV    CX,AX                            ;├
9449:    MUL    BL                               ;├ 计算 FCB 对应的 SFT 表数组占用的字节数,并
9450:    ADD    OFS_FreeMemory,AX                ;├ 修改自由内存的起始地址
```

```
9451:       MOV     AX,6                            ;|
9452:       ADD     OFS_FreeMemory,AX               ;┘
9453:       OR      FlagOfSCB,02                    ;┐ 转换自由内存的起始地址值格式,并设置子段
9454:       CALL    Convert_AddrFMT                 ;┘ 的大小
9455:       ADD     DI,AX                           ;┐
9456:       MOV     AL,' A'                         ;|
9457: Create_CB_3:                                  ;|
9458:       PUSH    CX                              ;|
9459:       MOV     CX,3BH                          ;|
9460:       CLD                                     ;| 初始化 FCB 对应的所有 SFT 表
9461:       REP     STOSB                           ;|
9462:       MOV     WORD PTR ES:[DI-3BH],0          ;|
9463:       MOV     WORD PTR ES:[DI-26H],0          ;|
9464:       MOV     WORD PTR ES:[DI-24H],0          ;|
9465:       POP     CX                              ;|
9466:       LOOP    Create_CB_3                     ;┘
9467:       CMP     WORD PTR NumberOfBuffer,-1      ;┐ 若没有指定盘缓冲区个数,则转去补缺盘缓冲
9468:       JE      Create_CB_4                     ;┘ 区个数
9469:       JMP     Create_CB_10
9470: Create_CB_4:                                  ;根据系统的配置情况来决定缺省的盘缓冲区个数
9471:       MOV     NumberOf2ndBuffer,0             ;被缺第二个高速缓存区中的缓冲区数目
9472:       MOV     WORD PTR NumberOfBuffer,2       ;缺省的盘缓冲区个数(2)
9473:       PUSH    AX
9474:       PUSH    DS
9475:       LES     BP,CS:Address_ListOfList        ;┐ ES:BP 指向 DDPB 链的头结点
9476:       LES     BP,DWORD PTR ES:[BP+0]          ;┘
9477:       PUSH    CS
9478:       POP     DS
9479: Create_CB_5:
9480:       MOV     BL,ES:[BP+0]                    ;BL←逻辑驱动器号(0=A,1=B,…)
9481:       INC     BL                              ;BL←驱动器号(0=当前的,1=A,2=B,…)
9482:       MOV     AX,4408H                        ;┐ 取出 BL 指定的驱动器是否支持可装卸的介质
9483:       INT     21H                             ;┘
9484:       OR      AX,AX                           ;┐ 若介质不能装卸或指定的块设备不存在,则跳
9485:       JNZ     Create_CB_7                     ;┘ 转
9486:       XOR     BX,BX                           ;┐
9487:       MOV     BL,ES:[BP+0]                    ;| BL←驱动器号(0=当前的,1=A,2=B,…)
9488:       INC     BL                              ;┘
9489:       MOV     DX, offset Packet               ;┐
9490:       MOV     AX,440DH                        ;| 取出设备参数→DS:DX 指定的参数块
9491:       MOV     CX,860H                         ;|
9492:       INT     21H                             ;┘
9493:       JC      Create_CB_7
```

```
9494:     MOV     BX,TotalSectors                 ;BX=驱动器的总扇区数
9495:     MOV     AX,BytesPerSector               ;AX=每个扇区的字节数
9496:     XOR     DX,DX                           ;┐
9497:     MOV     CX,512                          ;│
9498:     DIV     CX                              ;│
9499:     MUL     BX                              ;│
9500:     OR      DX,DX                           ;│校证缺省的盘缓冲区个数(若安装了360KB以
9501:     JNZ     Create_CB_6                     ;│上的软盘盘驱动器,则缺省值为3)
9502:     CMP     AX,720                          ;│
9503:     JBE     Create_CB_7                     ;│
9504: Create_CB_6:                                ;│
9505:     MOV     NumberOfBuffer,3                ;┘
9506:     JMP     SHORT Create_CB_8
9507: Create_CB_7:
9508:     CMP     WORD PTR ES:[BP+19H],-1         ;还有其它的块设备吗?
9509:     JE      Create_CB_8                     ;若没有,则跳转
9510:     LES     BP,DWORD PTR ES:[BP+19H]        ;ES:BP←下一个块设备对应的 DDPB 的起始地址
9511:     JMP     SHORT Create_CB_5
9512: Create_CB_8:
9513:     CMP     WORD PTR RAMSize,2000H          ;若常规内存≤128KB,则跳转
9514:     JBE     Create_CB_9
9515:     MOV     NumberOfBuffer,5                ;补缺盘缓冲区个数为5
9516:     CMP     RAMSize,4000H                   ;若满足条件128KB<常规内存≤256KB,则跳转
9517:     JBE     Create_CB_9
9518:     MOV     NumberOfBuffer,10               ;补缺盘缓冲区个数为10
9519:     CMP     RAMSize,8000H                   ;若满足条件256KB<常规内存≤512KB,则跳转
9520:     JBE     Create_CB_9
9521:     MOV     NumberOfBuffer,15               ;补缺盘缓冲区个数为15
9522: Create_CB_9:
9523:     POP     DS
9524:     POP     AX
9525: Create_CB_10:
9526:     LDS     BX,CS:Address_ListOfList        ;DS:BX 指向位于 MSDOS1模块中的多重表
9527:     MOV     AX,CS:NumberOfBuffer            ;┐盘缓冲区个数→多重表
9528:     MOV     [BX+3FH],AX                     ;┘
9529:     MOV     AX,CS:NumberOf2ndBuffer         ;┐第二个高速缓存区中的缓冲区个数→多重表
9530:     MOV     [BX+41H],AX                     ;┘
9531:     LDS     BX,[BX+12H]                     ;DS:BX←保存"盘缓冲区循环双链表的头结点指
                                                  ;针值"的双字单元地址
9532:     CALL    Convert_AddrFMT                 ;转换自由内存的起始地址值格式,并检查是否有
                                                  ;空闲内存
9533:     MOV     AL,' B'                         ;┐为盘缓冲区所占用的内存块建立对应的子段控
9534:     CALL    CreateSCB                       ;┘制块
```

```
9535:    PUSH    DS                          ;┐
9536:    PUSH    BX                          ;│
9537:    CALL    BulidBufferChain            ;│ 建立盘缓冲区的循环双链表
9538:    POP     BX                          ;│
9539:    POP     DS                          ;┘
9540:    CMP     CS:NumberOf2ndBuffer,00     ;┐ 若不建立第二个高速缓存区,则跳转
9541:    JE      Create_CB_11                ;┘
9542:    CALL    Convert_AddrFMT             ;转换自由内存的起始地址值格式,并检查是否有
                                             ;空闲内存
9543:    MOV     CX,CS:OFS_FreeMemory        ;┐
9544:    MOV     [BX+06],CX                  ;│ 第二个高速缓存区的起始地址→MSDOS1模块
9545:    MOV     CX,CS:SEG_FreeMemory        ;│
9546:    MOV     [BX+08],CX                  ;┘
9547:    MOV     CX,CS:NumberOf2ndBuffer     ;┐ 第二个高速缓存区的缓冲区个数→MSDOS1模
9548:    MOV     [BX+0AH],CX                 ;┘ 块
9549:    MOV     AX,200H                     ;┐ 计算第二个高速缓存区占用的字节数→AX
9550:    MUL     CX                          ;┘
9551:    MOV     CS:OFS_FreeMemory,AX        ;修改自由内存的起始地址
9552:    OR      CS:FlagOfSCB,02             ;┐ 转换自由内存的起始地址值格式,并设置子段
9553:    CALL    Convert_AddrFMT             ;┘ 的大小
9554: Create_CB_11:
9555:    CALL    Convert_AddrFMT             ;转换自由内存的起始地址值格式,并检查是否有
                                             ;空闲内存
9556:    PUSH    AX                          ;┐
9557:    MOV     AX,' L'                     ;│ 为 CDS 数组建立对应的子段控制块
9558:    CALL    CreateSCB                   ;│
9559:    POP     AX                          ;┘
9560:    LES     DI,CS:Address_ListOfList    ;ES:DI 指向位于 MSDOS1模块中的多重表
9561:    MOV     CL,ES:[DI+20H]              ;CL=逻辑驱动器数
9562:    CMP     CL,CS:LastDrive             ;┐
9563:    JAE     Create_CB_12                ;│
9564:    MOV     CL,CS:LastDrive             ;│ 校证系统允许访问的最大逻辑驱动器数
9565: Create_CB_12:                          ;│
9566:    XOR     CH,CH                       ;│
9567:    MOV     ES:[DI+21H],CL              ;┘
9568:    MOV     AX,CS:SEG_FreeMemory        ;┐
9569:    MOV     ES:[DI+18H],AX              ;│ CDS 数组的起始地址→多重表
9570:    MOV     AX,CS:OFS_FreeMemory        ;│
9571:    MOV     ES:[DI+16H],AX              ;┘
9572:    MOV     AL,CL                       ;┐
9573:    MOV     AH,58H                      ;│ 计算 CDS 数组占用的字节数→AX
9574:    MUL     AH                          ;┘
9575:    CALL    Convert_Addr                ;计算 CDS 数组占用的节数→AX
```

```
9576:    ADD     CS:SEG_FreeMemory,AX          ;修改自由内存的起始地址
9577:    OR      CS:FlagOfSCB,02               ;┐转换自由内存的起始地址值格式,并设置子段
9578:    CALL    Convert_AddrFMT               ;┘的大小
9579:    LDS     SI,DWORD PTR ES:[DI]          ;DS:SI 指向 DDPB 链的头结点
9580:    LES     DI,DWORD PTR ES:[DI+16H]      ;ES:DI←CDS 数组的起始地址
9581:    CALL    Init_CDS                      ;初始化 CDS 数组
9582:    PUSH    CS
9583:    POP     DS
9584:    CMP     WORD PTR OFS_StackField,-1    ;┐若 CONFIG.SYS 文件中有 stacks 系统配置命令,
9585:    JE      Create_CB_13                  ;┘则跳转
9586:    CMP     BYTE PTR SubmodelByte,0
9587:    JNE     Create_CB_13
9588:    CMP     BYTE PTR ModelByte,0FEH       ;┐若是 PC/XT 及兼容机,则跳转
9589:    JNB     Create_CB_14                  ;┘
9590: Create_CB_13:
9591:    MOV     AX,NumberOfStack              ;AX←动态堆栈数(即 stacks=n,s 的 n 参数值)
9592:    OR      AX,AX                         ;┐当 n 值为0时,DOS 没有接管任何中断,因而
9593:    JE      Create_CB_14                  ;┘DOS 不更换中断服务程序使用的堆栈
9594:    CALL    Convert_AddrFMT               ;转换自由内存的起始地址值格式,并检查是否有
                                               ;空闲内存
9595:    MOV     AL,'S'                        ;┐为动态堆栈占用的内存块建立子段控制块
9596:    CALL    CreateSCB                     ;┘
9597:    MOV     AX,SEG_FreeMemory             ;┐
9598:    MOV     ES,AX                         ;│
9599:    PUSH    CS                            ;│
9600:    POP     DS                            ;│安装 INT 02H、INT 08H～INT 0EH、INT 70H、
9601:    XOR     SI,SI                         ;│INT 72H～INT 74H、INT 76H～INT 77H 的中断
9602:    XOR     DI,DI                         ;│接管程序;转换自由内存的起始地址值格式,并
9603:    MOV     CX, offset SysInt_II_Entry1   ;│检查是否有空闲的内存
9604:    MOV     OFS_FreeMemory,CX             ;│
9605:    CALL    Convert_AddrFMT               ;│
9606:    REP     MOVSB                         ;┘
9607:    PUSH    DS
9608:    MOV     AX,BIO_SEG                    ;┐
9609:    MOV     DS,AX                         ;│下次可供分配的动态堆栈控制块的指针→IO1
9610:    MOV     WORD PTR Ptr_SearchSCB,0010H  ;│模块
9611:    MOV     WORD PTR Ptr_SearchSCB+2,ES   ;┘
9612:    MOV     AX,CS:OFS_FreeMemory          ;┐
9613:    MOV     CS:OFS_StackField,AX          ;│
9614:    MOV     WORD PTR Ptr_SCBArray,AX      ;│设置动态堆栈的控制块数组的起始地址
9615:    MOV     AX,CS:SEG_FreeMemory          ;│
9616:    MOV     CS:SEG_StackField,AX          ;│
9617:    MOV     WORD PTR Ptr_SCBArray+2,AX    ;┘
```

```
9618:     MOV     AX,8                        ;AX=每个动态堆栈控制块占用的字节数
9619:     ADD     AX,CS:SizeOfStack           ;加上每个动态堆栈占用的字节数,此时 AX=每个
                                              ;动态堆栈所需的字节数
9620:     MUL     CS:NumberOfStack            ;计算动态堆栈占用的总内存字节数→AX
9621:     MOV     SizeOfDynamicStk,AX         ;动态堆栈占用的总内存字节数→IO1模块
9622:     POP     DS
9623:     CALL    Convert_Addr                ;┐ 修改自由内存的起始地址值格式
9624:     ADD     CS:SEG_FreeMemory,AX        ;┘
9625:     OR      CS:FlagOfSCB,02             ;┐ 转换自由内存的起始地址值格式,并设置子段
9626:     CALL    Convert_AddrFMT             ;┘ 的大小
9627:     CALL    Process_Stack               ;建立动态堆栈的控制块,并安装 INT 02H、INT 08H
                                              ;～INT 0EH、INT 70H、INT 72H～INT 74H、INT
                                              ;76H～INT 77H 的中断接管程序

9628: Create_CB_14:
9629:     PUSH    CS
9630:     POP     DS
9631:     MOV     AL,NumberOfHandle           ;┐ AX←允许"能同时打   ┐
9632:     XOR     AH,AH                       ;┘ 开的最大文件数"   │
9633:     MOV     CX,AX                       ;CX←循环次数         │
9634:     XOR     BX,BX                       ;┐ 关闭0号文件句柄    │
9635:     MOV     AH,3EH                      ;│(它对应于标准输     │除标准输出设备对
9636:     INT     21H                         ;┘ 入设备)           │应的文件句柄以外,
9637:     MOV     BX,2                        ;┐                   │关闭其它所有文件
9638: Create_CB_15:                           ;│                   │句柄
9639:     MOV     AH,3EH                      ;│ 关闭所有大于或     │
9640:     INT     21H                         ;│ 等2的文件句柄      │
9641:     INC     BX                          ;│                   │
9642:     LOOP    Create_CB_15                ;┘                   ┘
9643:     MOV     DX,offset CON_Device        ;┐                   ┐
9644:     MOV     AL,2                        ;│                   │
9645:     MOV     AH,3DH                      ;│ 以"读/写"方式打开  │
9646:     STC                                 ;│ CON 设备           │
9647:     INT     21H                         ;┘                   │将0号、1号、2号文
9648:     JNC     Create_CB_16                ;                    │件句柄对应于 CON
9649:     CALL    Disp_ErrorInfo              ;                    │设备,使之分别成为
9650:     JMP     SHORT Create_CB_17          ;                    │标准输入设备、标准
9651: Create_CB_16:                           ;                    │输出设备和标准错
9652:     PUSH    AX                          ;┐                   │误设备的文件句柄
9653:     MOV     BX,1                        ;│                   │
9654:     MOV     AH,3EH                      ;│ 关闭1号文件句柄    │
9655:     INT     21H                         ;│                   │
9656:     POP     AX                          ;┘                   │
9657:     MOV     BX,AX                       ;┐                   ┘
```

```
9658:    MOV     AH,45H                          ;| 复制文件句柄,使1号和 |
9659:    INT     21H                             ;| 2号文件句柄对应于 CON|
9660:    MOV     AH,45H                          ;| 设备                 |
9661:    INT     21H                             ;┘                      ┘
9662: Create_CB_17:
9663:    MOV     DX,offset AUX_Device            ;┐ 将3号文件句柄对应于标准辅助设备
9664:    MOV     AL,2                            ;|
9665:    CALL    OpenDevice                      ;┘
9666:    MOV     DX,offset PRN_Device            ;┐ 将4号文件句柄对应于标准打印设备
9667:    MOV     AL,1                            ;|
9668:    CALL    OpenDevice                      ;┘
9669:    PUSH    AX
9670:    PUSH    BX
9671:    PUSH    DX
9672:    PUSH    ES
9673:    MOV     AL,-1
9674:    MOV     DX,2F2H
9675:    OUT     DX,AL
9676:    INC     DX
9677:    OUT     DX,AL
9678:    INC     DX
9679:    OUT     DX,AL
9680:    INC     DX
9681:    OUT     DX,AL
9682:    INC     DX
9683:    OUT     DX,AL
9684:    INC     DX
9685:    OUT     DX,AL
9686:    MOV     AX,ROM_SEG                          ;┐
9687:    MOV     ES,AX                               ;| 若机器是 PC/AT,则跳转
9688:    CMP     BYTE PTR ES:ROMAddr_FFFE,0FCH       ;|
9689:    JE      Create_CB_18                        ;┘
9690:    MOV     AH,0C0H                         ;┐ 取出系统配置参数
9691:    INT     15H                             ;┘
9692:    JC      Create_CB_19                    ;若 ROM BIOS 不支持该功能,则跳转
9693:    TEST    BYTE PTR ES:[BX+5],40H          ;┐ 若没有第二个中断芯片,则跳转
9694:    JZ      Create_CB_19                    ;┘
9695: Create_CB_18:
9696:    MOV     AL,0FFH
9697:    MOV     DX,6F2H
9698:    OUT     DX,AL
9699:    INC     DX
9700:    OUT     DX,AL
```

```
9701:   INC     DX
9702:   OUT     DX,AL
9703:   INC     DX
9704:   INC     DX
9705:   OUT     DX,AL
9706:   INC     DX
9707:   OUT     DX,AL
9708: Create_CB_19:
9709:   POP     ES
9710:   POP     DX
9711:   POP     BX
9712:   POP     AX
9713:   PUSH    AX
9714:   MOV     AX,SEG_FreeMemory               ;┐
9715:   SUB     AX,SEG_AllocatedMemory          ;│保存 DOS 的各种数据块和设备驱动程序占用的
9716:   MOV     SizeOfDDB,AX                    ;┘内存大小(节数)
9717:   MOV     AL,' T'                         ;┐建立临时程序段的子段控制块,这些临时程序
9718:   CALL    CreateSCB                       ;┘段支持 install 系统配置命令
9719:   POP     AX
9720:   MOV     DI,SEG_FreeMemory               ;┐
9721:   MOV     ES,DI                           ;│
9722:   MOV     WORD PTR SubRoutine_IO3+2,DI    ;│设置 ExecuteDOSFunction 子程序的入口地址
9723:   XOR     DI,DI                           ;│
9724:   MOV     WORD PTR SubRoutine_IO3,DI      ;┘
9725:   MOV     SI,offset ExecuteDOSFunction                                    ;┐
9726:   MOV     CX,offset BulidBufferChain - offset ExecuteDOSFunction          ;│将 ExecuteDOSFunction、
9727:   ADD     OFS_FreeMemory,CX                                               ;│CalculateAccumulateCode 子
9728:   OR      CS:FlagOfSCB,02                                                 ;│程序等安装在临时程序段的
9729:   CALL    Convert_AddrFMT                                                 ;│子段内存块中
9730:   REP     MOVSB                                                           ;┘
9731:   MOV     WORD PTR LabelAddr_IO3,offset Process_Install_5                 ;┐设置后续程序段 Process_
                                                                                ;│Install_5的入口地址,该程序
                                                                                ;│段是 Process_Install 的一个
9732:   MOV     WORD PTR LabelAddr_IO3+2,CS                                     ;┘组成部分
9733:   OR      WORD PTR Install_Flag,2         ;设置 install 系统配置命令的执行环境已建立的标
                                               ;志
9734:   CALL    Convert_AddrFMT                ;转换自由内存的起始地址值格式,并检查是否有
                                               ;空闲的内存
9735:   MOV     BX,SEG_FreeMemory              ;┐
9736:   MOV     AX,SEG_AllocatedMemory         ;│
9737:   MOV     Seg_FirstMCB2,AX               ;│
9738:   MOV     ES,AX                          ;│修改第一个内存块的大小
9739:   SUB     BX,AX                          ;│
```

```
9740:   MOV     AH,4AH                          ;|
9741:   INT     21H                             ;┘
9742:   PUSH    ES                                      ;┐
9743:   MOV     AX,ES                                   ;| PID 值(8表示 DOS 内核使用)→第一
9744:   DEC     AX                                      ;| 个内存控制块文件名("SD"表示系统
9745:   MOV     ES,AX                                   ;| 数据块)→第一个内存控制块
9746:   MOV     WORD PTR ES:MCB_ProcessID,8             ;|
9747:   MOV     WORD PTR ES:MCB_ProgramName,' DS'       ;|
9748:   POP     ES                                      ;┘
9749:   MOV     BX,0FFFFH                       ;┐
9750:   MOV     AH,48H                          ;|
9751:   INT     21H                             ;| 申请系统中全部可用的自由内存
9752:   MOV     AH,48H                          ;|
9753:   INT     21H                             ;┘
9754:   MOV     SEG_FreeMemory,AX               ;┐ 设置自由内存的起始地址
9755:   MOV     OFS_FreeMemory,0                ;┘
9756:   MOV     ES,AX                           ;┐
9757:   MOV     BX,SEG_Config                   ;| 修改自由内存块的大小,确保 CONFIG.SYS 文
9758:   SUB     BX,AX                           ;| 件内容的缓存区、IO3模块等 SysInt—Ⅱ执行的
9759:   DEC     BX                              ;| 环境信息不被破坏
9760:   DEC     BX                              ;|
9761:   MOV     AH,4AH                          ;|
9762:   INT     21H                             ;┘
9763:   MOV     BX,0FFFFH                       ;┐ 申请作为"CONFIG.SYS 文件内容的缓存区、
9764:   MOV     AH,48H                          ;| IO3模块等 SysInt—Ⅱ执行的环境信息缓存区"
9765:   INT     21H                             ;| 的内存块,确保它不被安装 Install 系统配置命
9766:   MOV     AH,48H                          ;| 令指定的内存驻留程序所占用
9767:   INT     21H                             ;┘
9768:   MOV     SEG_AllocatedMemory,AX          ;保存 SysInt—Ⅱ初始化时临时占用的内存块的段
                                                ;地址值
9769:   MOV     ES,SEG_FreeMemory               ;┐
9770:   MOV     AH,49H                          ;| 释放可自由使用的内存块
9771:   INT     21H                             ;┘
9772:   RET
9773: Create_ControlBlock  ENDP
9774: ;=============================================================
9775: ;相对地址偏移:10CF
9776: ;功能:处理 install 系统配置命令,安装 DOS 的内存驻留程序,并根据安装时的出错类型给出相应的
          错误信息
9777: ;入口参数:ES:SI 指向 install 系统配置命令指定的参数串,它由命令文件的文件名说明串和程序参
          数(或称为命令参数)组成
9778: ;出口参数:CF=0,成功;CF=1,失败
9779: ;=============================================================
```

```
9780: Process_Install  PROC  NEAR
9781:     PUSH    SI
9782:     PUSH    ES
9783:     PUSH    DS
9784:     POP     ES
9785:     POP     DS
9786:     MOV     DX,SI                    ;DS:DX 指向 install 系统配置命令' 指定的命令文
                                           ;件的文件名说明串
9787:     XOR     CX,CX                    ;清"程序参数"的字符数
9788:     CLD
9789:     MOV     CS:TempParaBuf,20H       ;┐ ES:DI 指向程序参数的暂存区
9790:     MOV     DI,offset TempParaBuf+1  ;┘
9791: Process_Install_1:                   ;┐
9792:     LODSB                            ;│
9793:     CMP     AL,0                     ;│ 跳过"命令文件的文件名说明串"
9794:     JE      Process_Install_2        ;│
9795:     JMP     SHORT Process_Install_1  ;┘
9796: Process_Install_2:                   ;┐
9797:     LODSB                            ;│
9798:     MOV     ES:[DI],AL               ;│
9799:     CMP     AL,0AH                   ;│ 保存"程序参数",并计算"程序参数"的字符数
9800:     JE      Process_Install_3        ;│ →CL
9801:     INC     CL                       ;│
9802:     INC     DI                       ;│
9803:     JMP     SHORT Process_Install_2  ;┘
9804: Process_Install_3:
9805:     MOV     CS:LengthOfPara,CL       ;设置"程序参数"的字符数
9806:     CMP     CL,0                     ;┐ 若"程序参数"有内容,则跳转
9807:     JNE     Process_Install_4        ;┘
9808:     MOV     CS:TempParaBuf,0DH       ;设置程序参数的结束符(此时不存在"程序参数"
9809: Process_Install_4:
9810:     MOV     WORD PTR CS:[0],0
9811:     MOV     AX,CS                    ;                          ┐
9812:     MOV     CS:EXECPB2.EXEC_Envioronment,AX ;设置环境块的段地址  │ 建立功能调用
9813:     MOV     CS:EXECPB2.EXEC_CMDLine_SEG,AX;设置命令行的段地址  │ 4B00H 的参数块
9814:     MOV     CS:EXECPB2.EXEC_FCB1_SEG,AX   ;设置 FCB1的段地址   │
9815:     MOV     CS:EXECPB2.EXEC_FCB2_SEG,AX   ;设置 FCB2的段地址   ┘
9816:     CALL    CalculateAccumulateCode  ;┐ 计算并保存 CONFIG.SYS 文件内容和 IO3模块
9817:     MOV     ES:AccumulatedCode,AX    ;┘ 有效代码的累加和
9818:     XOR     AX,AX                    ;┐ AX←DOS 功能调用码(装入并执行)
9819:     MOV     AH,4BH                   ;┘
9820:     MOV     BX,offset EXECPB2        ;ES:BX 指向程序执行的参数块
9821:     PUSH    ES
```

```
9822:     PUSH    DS
9823:     JMP     DWORD PTR CS:SubRoutine_IO3   ;控制权转入 ExecuteDOSFunction,以便装入并执行
                                                ;由 install 系统配置命令指定的 DOS 外部命令文件
9824: Process_Install_5:
9825:     POP     SI                            ;┐
9826:     PUSH    ES                            ;│
9827:     PUSH    DS                            ;│ DS:SI 指向 install 系统配置命令指定的参数串
9828:     POP     ES                            ;│
9829:     POP     DS                            ;┘
9830:     JNB     Process_Install_6             ;若 install 系统配置命令指定的程序执行正确,则跳
                                                ;转
9831:     PUSH    SI                            ;┐
9832:     CALL    Disp_ErrorStr                 ;│ 显示错误信息"Bad or missing"和 install 系统配
9833:     POP     SI                            ;┘ 置命令指定的参数串
9834:     JMP     SHORT Process_Install_RET
9835: Process_Install_6:
9836:     MOV     AH,4DH                        ;┐ 取返回码
9837:     INT     21H                           ;┘
9838:     CMP     AH,3                          ;┐ 若由 install 系统配置命令安装的程序是正常的
9839:     JZ      Process_Install_RET           ;┘ 驻留退出,则结束返回
9840:     CALL    OutputErrSiteInfo             ;输出错误系统配置命令在 CONFIG.SYS 文本文件
                                                ;中的位置信息
9841:     STC
9842: Process_Install_RET:
9843:     RET
9844: Process_Install  ENDP
9845: ;=============================================================
9846: ;相对地址偏移:1150
9847: ;功能:将地址偏移值转换成相对段地址值
9848: ;入口参数:AX=地址偏移量
9849: ;出口参数:AX=相对段地址值
9850: ;=============================================================
9851: Convert_Addr  PROC  NEAR
9852:     ADD     AX,0FH
9853:     RCR     AX,1
9854:     SHR     AX,1
9855:     SHR     AX,1
9856:     SHR     AX,1
9857:     RET
9858: Convert_Addr  ENDP
9859: ;=============================================================
9860: ;相对地址偏移:115C
9861: ;功能:执行 DOS 功能调用(该子程序是为了完成 install 系统配置命令而设置的)
```

```
9862: ;入口参数:AX=DOS 功能调用码(4B00H)
9863: ;          DS:DI 指向文件名说明串
9864: ;          ES:BX 指向程序执行的参数块
9865: ;出口参数:无
9866: ;=========================================================
9867: ExecuteDOSFunction  PROC  NEAR
9868:     MOV     WORD PTR CS:[0061H],SS      ;0061H=11BD - 115C ┐ 保存堆栈指针寄存器值
9869:     MOV     WORD PTR CS:[0063H],SP      ;0063H=11BE - 115C ┘
9870:     INT     21H                         ;请求 DOS 功能调用
9871:     MOV     SS,WORD PTR CS:[0061H]      ;┐ 恢复堆栈指针寄存器值
9872:     MOV     SP,WORD PTR CS:[0063H]      ;┘
9873:     POP     DS
9874:     POP     ES
9875:     JB      ExecDOSFunc_2               ;若装入并执行由 install 系统配置命令指定的程序
                                              ;失败,则跳转
9876:     CALL    CalculateAccumulateCode     ;┐ 若装入并执行由 install 系统配置命令指定的程
9877:     CMP     ES:AccumulatedCode,AX       ;│ 序之后,SysInt— I 的执行环境未被破坏,则跳
9878:     JZ      ExecDOSFunc_2               ;┘ 转
9879:     MOV     AH,9                                    ;┐
9880:     PUSH    CS                                      ;│
9881:     POP     DS                                      ;│ 显示错误信息"Memory
9882:     MOV     DX,offset Install_MemErr - offset ExecuteDOSFunction;│ allocation error"
9883:     INT     21H                                     ;┘
9884: ExecDOSFunc_1:                          ;┐ 系统进入死循环
9885:     JMP     SHORT ExecDOSFunc_1         ;┘
9886: ExecDOSFunc_2:
9887:     JMP     DWORD PTR ES:LabelAddr_IO3  ;控制权转入 Process_Install_5处接着执行
9888: ExecuteDOSFunction  ENDP
9889: ;=========================================================
9890: ;相对地址偏移:1190
9891: ;功能:计算 CONFIG.SYS 文件内容和 IO3模块有效代码的累加和
9892: ;入口参数:无
9893: ;出口参数:AX=CONFIG.SYS 文件内容和 IO3模块有效代码的累加和
9894: ;=========================================================
9895: CalculateAccumulateCode  PROC  NEAR
9896:     PUSH    DS
9897:     MOV     AX,ES:SEG_Config            ;┐
9898:     MOV     DS,AX                       ;│ DS:SI 指向存放 CONFIG.SYS 文件内容的缓存
9899:     XOR     SI,SI                       ;┘ 区
9900:     XOR     AX,AX                       ;清 CONFIG.SYS 文件内容和 IO3模块有效代码的
                                              ;累加和
9901:     MOV     CX,ES:LengthOfConfig        ;┐
9902:     SHR     CX,1                        ;│
```

```
9903:     JZ      CalcuAccC_2                    ;|
9904: CalcuAccC_1:                               ;|
9905:     ADD     AX,[SI]                        ;| 计算 CONFIG.SYS 文件内容的累加和
9906:     INC     SI                             ;|
9907:     INC     SI                             ;|
9908:     LOOP    CalcuAccC_1                    ;|
9909: CalcuAccC_2:                               ;┘
9910:     MOV     SI,offset SysInt_II_E2_1       ;┐
9911:     MOV     CX,offset End_SysInt_II        ;|
9912:     SUB     CX,SI                          ;|
9913:     SHR     CX,1                           ;|
9914: CalcuAccC_3:                               ;| 计算 IO3模块有效代码的累加和
9915:     ADD     AX,ES:[SI]                     ;|
9916:     INC     SI                             ;|
9917:     INC     SI                             ;|
9918:     LOOP    CalcuAccC_3                    ;┘
9919:     POP     DS
9920:     RET
9921: CalculateAccumulateCode  ENDP
9922: ;相对地址偏移:11BD
9923:     DW      0                              ;保存堆栈指针寄存器 SS 的值
9924:     DW      0                              ;保存堆栈指针寄存器 SP 的值
9925: Install_MemErr  DB  0DH, 0AH, ' Memory allocation error ¥'   ;错误信息
9926: ;
9927: ;=============================================================
9928: ;相对地址偏移:11DC
9929: ;功能:建立盘缓冲区的循环双链表,并设置 MSDOS1模块中的对应变量
9930: ;入口参数:DS:BX=保存"盘缓冲区循环双链表的头结点指针值"的双字单元地址
9931: ;出口参数:无
9932: ;=============================================================
9933: BulidBufferChain  PROC  NEAR
9934:     XOR     DL,DL                          ;清盘缓冲区位置标志(0:表示盘缓冲区建在常规
                                                 ;内存中)
9935:     CALL    AllocateBufferMem              ;分配作为盘缓冲区的内存块
9936:     JZ      BulidBufChain_1                ;若在常规内存中建立盘缓冲区,则跳转
9937:     MOV     DL,1                           ;设置盘缓冲区位置标志(1:表示盘缓冲区建在
                                                 ;HMA 中)
9938: BulidBufChain_1:
9939:     MOV     [BX],DI                        ;┐ 盘缓冲区的循环双链表的头结点地址→
9940:     MOV     WORD PTR [BX+2],ES             ;┘ MSDOS1模块
9941:     MOV     WORD PTR [BX+4],0
9942:     MOV     AX,DI                          ;┐
9943:     MOV     CX,CS:NumberOfBuffer           ;|
```

```
9944:     PUSH    DI                              ;|
9945: BulidBufChain_2:                            ;|
9946:     CALL    Create_A_Buffer                 ;|
9947:     MOV     DI,AX                           ;| 建立盘缓冲区的循环双链表
9948:     LOOP    BulidBufChain_2                 ;|
9949:     SUB     DI,CS:SizePerBuffer             ;|
9950:     POP     CX                              ;|
9951:     MOV     ES:[DI],CX                      ;|
9952:     XCHG    CX,DI                           ;|
9953:     MOV     WORD PTR ES:[DI+2],CX           ;┘
9954:     OR      DL,DL                           ;┐ 若盘缓冲区是建在常规内存中,则跳转
9955:     JZ      BulidBufChain_3                 ;┘
9956: ;盘缓冲区建在 HMA 中
9957:     MOV     BYTE PTR [BX+0CH],1             ;盘缓冲区位置标志(1:表示盘缓冲区建在 HMA
                                                  ;中)→MSDOS1模块
9958:     MOV     AX,CS:SEG_FreeMemory            ;┐
9959:     MOV     WORD PTR [BX+0DH],0             ;| "建在常规内存中的一个盘缓存区"的起始地址
9960:     MOV     WORD PTR [BX+0FH],AX            ;┘ →MSDOS1模块
9961:     MOV     AX,CS:SizePerBuffer             ;┐ AX=盘缓存区占用的字节数
9962:     SUB     AX,14H                          ;┘
9963: BulidBufChain_3:
9964:     ADD     CS:OFS_FreeMemory,AX            ;修改自由内存块的起始地址
9965:     OR      CS:FlagOfSCB,02                 ;┐ 转换自由内存块的起始地址值格式,并设置子
9966:     CALL    Convert_AddrFMT                 ;┘ 段的大小
9967:     RET
9968: BulidBufferChain  ENDP
9969: ;=============================================================
9970: ;相对地址偏移:1237
9971: ;功能:分配作为盘缓冲区的常规内存块或高内存区中的内存块
9972: ;入口参数:无
9973: ;出口参数:ES:DI=作为盘缓冲区的内存块的起始地址
9974: ;          ZF=0,盘缓冲区建立在 HMA 中;ZF=1,盘缓冲区建立在常规内存中
9975: ;=============================================================
9976: AllocateBufferMem  PROC  NEAR
9977:     PUSH    BX
9978:     PUSH    DX
9979:     MOV     AX,CS:SizePerBuffer             ;┐ 计算盘缓冲区所需的字节数
9980:     MUL     CS:NumberOfBuffer               ;┘
9981:     ADD     AX,0FH                          ;┐
9982:     AND     AX,0FFF0H                       ;| BX=在 HMA 中建立盘缓冲区时所需的字节数
9983:     MOV     BX,AX                           ;┘
9984:     MOV     AX,4A02H                        ;┐ 申请建立盘缓冲区所需的 HMA 内存
9985:     INT     2FH                             ;┘
```

```
9986:     CMP     DI,-1                       ;┐ 若已分配到可用的 HMA 内存,则跳转
9987:     JNE     AllocateBufMem_1            ;┘
9988:     MOV     DI,0                        ;┐ ES:DI←作为盘缓冲区的常规内存块的起始地
9989:     MOV     ES,CS:SEG_FreeMemory        ;┘ 址
9990: AllocateBufMem_1:
9991:     POP     DX
9992:     POP     BX
9993:     RET
9994: AllocateBufferMem   ENDP
9995: ;=============================================================
9996: ;相对地址偏移:125F
9997: ;功能:建立一个盘缓冲区
9998: ;入口参数:AX=当前盘缓冲区的起始地址偏移值
9999: ;            ES:DI 指向当前盘缓冲区
10000: ;出口参数:AX=下一个盘缓冲区的起始地址偏移值
10001: ;=============================================================
10002: Create_A_Buffer   PROC   NEAR
10003:     PUSH    CS:OFS_PreviousBuffer       ;┐ 设置当前盘缓冲区的反向指针
10004:     POP     WORD PTR ES:[DI+2]          ;┘
10005:     MOV     CS:OFS_PreviousBuffer,AX    ;保存当盘缓冲区的起始地址偏移值
10006:     ADD     AX,CS:SizePerBuffer         ;┐ 设置当前盘缓冲区的正向指针
10007:     MOV     ES:[DI],AX                  ;┘
10008:     MOV     WORD PTR ES:[DI+4],0FFH     ;设置当前盘缓冲区的使用标志(-1:表示未被使
                                               ;用)
10009:     MOV     WORD PTR ES:[DI+6],0        ;┐ 清当前盘缓冲区的逻辑扇区号
10010:     MOV     WORD PTR ES:[DI+8],0        ;┘
10011:     RET
10012: Create_A_Buffer   ENDP
10013: ;=============================================================
10014: ;相对地址偏移:1287
10015: ;功能:建立动态堆栈的控制块,并安装 INT 02H、INT 08H~INT 0EH、INT 70H、INT 72H~INT 74H、
           INT 76H~INT 77H 的中断接管程序
10016: ;入口参数:ES=接管中断的中断接管程序和动态堆栈所占用的内存块的段地址值
10017: ;出口参数:无
10018: ;=============================================================
10019: Process_Stack   PROC   NEAR
10020:     PUSH    AX
10021:     PUSH    DS
10022:     PUSH    ES
10023:     PUSH    BX
10024:     PUSH    CX
10025:     PUSH    DX
```

```
10026:      PUSH    DI
10027:      PUSH    SI
10028:      PUSH    BP
10029:      MOV     AX,CS:NumberOfStack             ;┐ 设置动态堆栈的数目
10030:      MOV     WORD PTR ES:n_Stack,AX          ;┘
10031:      MOV     AX,SizeOfStack                  ;┐ 保存动态堆栈的尺寸(字节数)
10032:      MOV     ES:s_Stack,AX                   ;┘
10033:      MOV     AX,CS:OFS_StackField            ;┐
10034:      MOV     ES:OFS_ManageBlockHead,AX       ;├ 设置动态堆栈的控制块数组的起始地址
10035:      MOV     AX,CS:SEG_StackField            ;│
10036:      MOV     ES:SEG_ManageBlockHead,AX       ;┘
10037:      MOV     BP,ES:OFS_ManageBlockHead       ;┐ 设置动态堆栈的控制块数组的起始地址偏移
10038:      MOV     ES:OFS_ManageBlockHead2,BP      ;┘
10039:      MOV     AX,  8                          ;┐
10040:      MOV     CX,WORD PTR ES:n_Stack          ;│
10041:      MUL     CX                              ;├ 计算并保存动态堆栈区的起始地址偏移
10042:      ADD     AX,BP                           ;│
10043:      MOV     ES:OFS_Stack,AX                 ;┘
10044:      MOV     BX,AX
10045:      SUB     BX,2
10046:      MOV     DI,ES:OFS_Stack                 ;                          ┐
10047:      MOV     AX,ES:s_Stack                   ;┐                         │
10048:      MUL     CX                              ;├ 计算动态堆栈区占用的    │
10049:      MOV     CX,AX                           ;┘ 字节数→CX               │
10050:      XOR     AX,AX                           ;┐                         ├ 初始化动态堆栈区
10051:      PUSH    ES                              ;│                         │
10052:      POP     DS                              ;├ 初始化作为动态堆栈区    │
10053:      MOV     ES,DS:SEG_ManageBlockHead       ;├ 的内存块                │
10054:      CLD                                     ;│                         │
10055:      REP     STOSB                           ;┘                         ┘
10056:      MOV     CX,WORD PTR DS:n_Stack          ;CX←指定的动态堆栈个数
10057: Pro_Stack1:
10058:      MOV     ES:[BP].SCB_UseMark,0           ;清动态堆栈的状态标志      ┐
                                                    ;(0:空闲)                  │
10059:      MOV     ES:[BP].SCB_Reserved,AL         ;清保留字节                │
10060:      MOV     ES:[BP].SCB_SPValue,AX          ;┐ 清先前 SP、SS 寄存器值  │
10061:      MOV     ES:[BP].SCB_SSValue,AX          ;┘                         ├ 初始化各个动态堆
10062:      ADD     BX,DS:s_Stack                   ;┐ 设置对应的堆栈顶的指    │ 栈的控制块
10063:      MOV     ES:[BP].SCB_AddrOfStack,BX      ;┘ 针值                    │
10064:      MOV     ES:[BX],BP                      ;在堆栈顶预先存入对应的    │
                                                    ;控制块的地址偏移值        │
10065:      ADD     BP,8                            ;ES:BP 指向下一个动态堆    │
                                                    ;栈的控制块                │
```

```
10066:      LOOP    Pro_Stack1                      ;                    ┘
10067:      SUB     BP,8                            ;ES:BP 指向最后一个动态堆栈的控制块
10068:      MOV     DS:OFS_ManageBlockTail,BP       ;保存最后一个动态堆栈的控制块
10069:      MOV     DS:OFS_AllocatedManageBlock,BP  ;设置下次可供分配的动态堆栈对应的控制块的起
                                                    ;始地址偏移值
10070:      PUSH    DS                                  ;┐
10071:      MOV     AX,ROM_SEG                          ;|
10072:      MOV     DS,AX                               ;| 是 PC 兼容机吗?
10073:      CMP     BYTE PTR DS:ROMAddr_FFFE,0F9H       ;|
10074:      POP     DS                                  ;┘
10075:      JNZ     Pro_Stack2                      ;若不是,则跳转
10076:      MOV     AL,7
10077:      OUT     72H,AL
10078:  Pro_Stack2:
10079:      XOR     AX,AX
10080:      MOV     ES,AX
10081:      CLI
10082:      MOV     SI,Vector_02H_OFS               ;┐
10083:      MOV     DI, offset Old_INT02H_Vector1   ;|
10084:      MOV     BX, offset Old_INT02H_Vector2   ;| 安装 INT 02H 中断接管程序(为了切换堆栈)
10085:      MOV     DX, offset INT02H_Entry         ;|
10086:      CALL    Modify_Vector                   ;┘
10087:      MOV     SI,Vector_08H_OFS               ;┐
10088:      MOV     DI, offset Old_INT08H_Vector1   ;|
10089:      MOV     BX, offset Old_INT08H_Vector2   ;| 安装 INT 08H 中断接管程序(为了切换堆栈)
10090:      MOV     DX, offset INT08H_Entry         ;|
10091:      CALL    Modify_Vector                   ;┘
10092:      MOV     SI,Vector_09H_OFS               ;┐
10093:      MOV     DI, offset Old_INT09H_Vector1   ;|
10094:      MOV     BX, offset Old_INT09H_Vector2   ;| 安装 INT 09H 中断接管程序(为了切换堆栈)
10095:      MOV     DX, offset INT09H_Entry         ;|
10096:      CALL    Modify_Vector                   ;┘
10097:      MOV     SI,Vector_70H_OFS               ;┐
10098:      MOV     DI, offset Old_INT70H_Vector1   ;|
10099:      MOV     BX, offset Old_INT70H_Vector2   ;| 安装 INT 70H 中断接管程序(为了切换堆栈)
10100:      MOV     DX, offset INT70H_Entry         ;|
10101:      CALL    Modify_Vector                   ;┘
10102:      MOV     SI,Vector_0AH_OFS               ;┐
10103:      PUSH    DS                              ;|
10104:      LDS     BX,DWORD PTR ES:[SI]            ;|
10105:      PUSH    DS                              ;|
10106:      POP     DX                              ;|
10107:      CMP     DX,0                            ;|
```

```
10108:      JE      Pro_Stack4                  ;|
10109:      CMP     BYTE PTR [BX],0CFH          ;|
10110:      JE      Pro_Stack4                  ;|
10111:      CMP     WORD PTR [BX+6],' BK'       ;| 判定是否可以接管 INT 0AH 中断服务程序
10112:      JE      Pro_Stack3                  ;| 若不可以,则跳转
10113:      CMP     DX,ROM_SEG                  ;|
10114:      JNE     Pro_Stack3                  ;|
10115:      PUSH    ES                          ;|
10116:      PUSH    DX                          ;|
10117:      MOV     DX,ROM_SEG                  ;|
10118:      MOV     ES,DX                       ;|
10119:      CMP     BX,ES:ROMAddr_FF01          ;|
10120:      POP     DX                          ;|
10121:      POP     ES                          ;|
10122:      JE      Pro_Stack4                  ;|
10123: Pro_Stack3:                              ;┘
10124:      POP     DS
10125:      MOV     DI, offset Old_INT0AH_Vector1   ;┐
10126:      MOV     BX, offset Old_INT0AH_Vector2   ;| 安装 INT 0AH 中断接管程序(为了切换堆栈)
10127:      MOV     DX, offset INT0AH_Entry     ;|
10128:      CALL    Modify_Vector               ;┘
10129:      JMP     SHORT Pro_Stack5
10130: Pro_Stack4:
10131:      POP     DS
10132: Pro_Stack5:
10133:      MOV     SI,Vector_0BH_OFS           ;┐
10134:      PUSH    DS                          ;|
10135:      LDS     BX,DWORD PTR ES:[SI]        ;|
10136:      PUSH    DS                          ;|
10137:      POP     DX                          ;|
10138:      CMP     DX,0                        ;|
10139:      JE      Pro_Stack7                  ;|
10140:      CMP     BYTE PTR [BX],0CFH          ;|
10141:      JE      Pro_Stack7                  ;|
10142:      CMP     WORD PTR [BX+6],' BK'       ;|
10143:      JE      Pro_Stack6                  ;| 判定是否可以接管 INT 0BH 中断服务程序,
10144:      CMP     DX,ROM_SEG                  ;| 若不可以,则跳转
10145:      JNE     Pro_Stack6                  ;|
10146:      PUSH    ES                          ;|
10147:      PUSH    DX                          ;|
10148:      MOV     DX,ROM_SEG                  ;|
10149:      MOV     ES,DX                       ;|
10150:      CMP     BX,ES:ROMAddr_FF01          ;|
```

```
10151:      POP     DX                          ;|
10152:      POP     ES                          ;|
10153:      JE      Pro_Stack7                  ;|
10154: Pro_Stack6:                              ;┘
10155:      POP     DS
10156:      MOV     DI, offset Old_INT0BH_Vector1   ;┐
10157:      MOV     BX, offset Old_INT0BH_Vector2   ;|
10158:      MOV     DX, offset INT0BH_Entry         ;| 安装 INT 0BH 中断接管程序(为了切换堆栈)
10159:      CALL    Modify_Vector               ;|
10160:      JMP     SHORT Pro_Stack8            ;┘
10161: Pro_Stack7:
10162:      POP     DS
10163: Pro_Stack8:
10164:      MOV     SI,Vector_0CH_OFS           ;┐
10165:      PUSH    DS                          ;|
10166:      LDS     BX,DWORD PTR ES:[SI]        ;|
10167:      PUSH    DS                          ;|
10168:      POP     DX                          ;|
10169:      CMP     DX,0                        ;|
10170:      JE      Pro_Stack10                 ;|
10171:      CMP     BYTE PTR [BX],0CFH          ;|
10172:      JE      Pro_Stack10                 ;|
10173:      CMP     WORD PTR [BX+6],' BK'       ;|
10174:      JE      Pro_Stack9                  ;| 判定是否可以接管 INT 0CH 中断服务程序
10175:      CMP     DX,ROM_SEG                  ;| 若不可以,则跳转
10176:      JNE     Pro_Stack9                  ;|
10177:      PUSH    ES                          ;|
10178:      PUSH    DX                          ;|
10179:      MOV     DX,ROM_SEG                  ;|
10180:      MOV     ES,DX                       ;|
10181:      CMP     BX,ES:ROMAddr_FF01          ;|
10182:      POP     DX                          ;|
10183:      POP     ES                          ;|
10184:      JE      Pro_Stack10                 ;|
10185: Pro_Stack9:                              ;┘
10186:      POP     DS
10187:      MOV     DI, offset Old_INT0CH_Vector1   ;┐
10188:      MOV     BX, offset Old_INT0CH_Vector2   ;| 安装 INT 0CH 中断接管程序(为了切换堆栈)
10189:      MOV     DX, offset INT0CH_Entry         ;|
10190:      CALL    Modify_Vector               ;┘
10191:      JMP     SHORT Pro_Stack11
10192: Pro_Stack10:
10193:      POP     DS
```

```
10194: Pro_Stack11:
10195:     MOV     SI,Vector_0DH_OFS              ;┐
10196:     PUSH    DS                             ;|
10197:     LDS     BX,DWORD PTR ES:[SI]           ;|
10198:     PUSH    DS                             ;|
10199:     POP     DX                             ;|
10200:     CMP     DX,0                           ;|
10201:     JE      Pro_Stack13                    ;|
10202:     CMP     BYTE PTR [BX],0CFH             ;|
10203:     JE      Pro_Stack13                    ;|
10204:     CMP     WORD PTR [BX+6],' BK'          ;|
10205:     JE      Pro_Stack12                    ;| 判定是否可以接管 INT 0DH 中断服务程序
10206:     CMP     DX,ROM_SEG                     ;| 若不可以,则跳转
10207:     JNE     Pro_Stack12                    ;|
10208:     PUSH    ES                             ;|
10209:     PUSH    DX                             ;|
10210:     MOV     DX,ROM_SEG                     ;|
10211:     MOV     ES,DX                          ;|
10212:     CMP     BX,ES:ROMAddr_FF01             ;|
10213:     POP     DX                             ;|
10214:     POP     ES                             ;|
10215:     JE      Pro_Stack13                    ;|
10216: Pro_Stack12:                               ;┘
10217:     POP     DS
10218:     MOV     DI, offset Old_INT0DH_Vector1  ;┐
10219:     MOV     BX, offset Old_INT0DH_Vector2  ;|
10220:     MOV     DX, offset INT0DH_Entry        ;| 安装 INT 0DH 中断接管程序(为了切换堆栈)
10221:     CALL    Modify_Vector                  ;|
10222:     JMP     SHORT Pro_Stack14              ;┘
10223: Pro_Stack13:
10224:     POP     DS
10225: Pro_Stack14:
10226:     MOV     SI,Vector_0EH_OFS              ;┐
10227:     PUSH    DS                             ;|
10228:     LDS     BX,DWORD PTR ES:[SI]           ;|
10229:     PUSH    DS                             ;|
10230:     POP     DX                             ;|
10231:     CMP     DX,0                           ;|
10232:     JE      Pro_Stack16                    ;|
10233:     CMP     BYTE PTR [BX],0CFH             ;|
10234:     JE      Pro_Stack16                    ;|
10235:     CMP     WORD PTR [BX+6],' BK'          ;|
10236:     JE      Pro_Stack15                    ;| 判定是否可以接管 INT 0EH 中断服务程序
```

```
10237:    CMP     DX,ROM_SEG                    ;| 若不可以,则跳转
10238:    JNE     Pro_Stack15                   ;|
10239:    PUSH    ES                            ;|
10240:    PUSH    DX                            ;|
10241:    MOV     DX,ROM_SEG                    ;|
10242:    MOV     ES,DX                         ;|
10243:    CMP     BX,ES:ROMAddr_FF01            ;|
10244:    POP     DX                            ;|
10245:    POP     ES                            ;|
10246:    JE      Pro_Stack16                   ;|
10247: Pro_Stack15:                             ;┘
10248:    POP     DS
10249:    MOV     DI, offset Old_INT0EH_Vector1 ;┐
10250:    MOV     BX, offset Old_INT0EH_Vector2 ;| 安装 INT 0EH 中断接管程序(为了切换堆栈)
10251:    MOV     DX, offset INT0EH_Entry       ;|
10252:    CALL    Modify_Vector                 ;┘
10253:    JMP     SHORT Pro_Stack17
10254: Pro_Stack16:
10255:    POP     DS
10256: Pro_Stack17:
10257:    MOV     SI,Vector_72H_OFS             ;┐
10258:    PUSH    DS                            ;|
10259:    LDS     BX,DWORD PTR ES:[SI]          ;|
10260:    PUSH    DS                            ;|
10261:    POP     DX                            ;|
10262:    CMP     DX,0                          ;|
10263:    JE      Pro_Stack19                   ;|
10264:    CMP     BYTE PTR [BX],0CFH            ;|
10265:    JE      Pro_Stack19                   ;|
10266:    CMP     WORD PTR [BX+6],' BK'         ;|
10267:    JE      Pro_Stack18                   ;| 判定是否可以接管 INT 72H 中断服务程序
10268:    CMP     DX,ROM_SEG                    ;| 若不可以,则跳转
10269:    JNE     Pro_Stack18                   ;|
10270:    PUSH    ES                            ;|
10271:    PUSH    DX                            ;|
10272:    MOV     DX,ROM_SEG                    ;|
10273:    MOV     ES,DX                         ;|
10274:    CMP     BX,ES:ROMAddr_FF01            ;|
10275:    POP     DX                            ;|
10276:    POP     ES                            ;|
10277:    JE      Pro_Stack19                   ;|
10278: Pro_Stack18:                             ;┘
10279:    POP     DS
```

```
10280:      MOV     DI, offset Old_INT72H_Vector1     ;┐
10281:      MOV     BX, offset Old_INT72H_Vector2     ; | 安装 INT 72H 中断接管程序(为了切换堆栈)
10282:      MOV     DX, offset INT72H_Entry           ; |
10283:      CALL    Modify_Vector                     ;┘
10284:      JMP     SHORT Pro_Stack20
10285: Pro_Stack19:
10286:      POP     DS
10287: Pro_Stack20:
10288:      MOV     SI,Vector_73H_OFS                 ;┐
10289:      PUSH    DS                                ; |
10290:      LDS     BX,DWORD PTR ES:[SI]              ; |
10291:      PUSH    DS                                ; |
10292:      POP     DX                                ; |
10293:      CMP     DX,0                              ; |
10294:      JE      Pro_Stack22                       ; |
10295:      CMP     BYTE PTR [BX],0CFH                ; |
10296:      JE      Pro_Stack22                       ; |
10297:      CMP     WORD PTR [BX+6],' BK'             ; |
10298:      JE      Pro_Stack21                       ; | 判定是否可以接管 INT 73H 中断服务程序
10299:      CMP     DX,ROM_SEG                        ; | 若不可以,则跳转
10300:      JNE     Pro_Stack21                       ; |
10301:      PUSH    ES                                ; |
10302:      PUSH    DX                                ; |
10303:      MOV     DX,ROM_SEG                        ; |
10304:      MOV     ES,DX                             ; |
10305:      CMP     BX,ES:ROMAddr_FF01                ; |
10306:      POP     DX                                ; |
10307:      POP     ES                                ; |
10308:      JE      Pro_Stack22                       ; |
10309: Pro_Stack21:                                   ;┘
10310:      POP     DS
10311:      MOV     DI, offset Old_INT73H_Vector1     ;┐
10312:      MOV     BX, offset Old_INT73H_Vector2     ; | 安装 INT 73H 中断接管程序(为了切换堆栈)
10313:      MOV     DX, offset INT73H_Entry           ; |
10314:      CALL    Modify_Vector                     ;┘
10315:      JMP     SHORT Pro_Stack23
10316: Pro_Stack22:
10317:      POP     DS
10318: Pro_Stack23:
10319:      MOV     SI,Vector_74H_OFS                 ;┐
10320:      PUSH    DS                                ; |
10321:      LDS     BX,DWORD PTR ES:[SI]              ; |
10322:      PUSH    DS                                ; |
```

```
10323:    POP     DX                          ;|
10324:    CMP     DX,0                        ;|
10325:    JE      Pro_Stack25                 ;|
10326:    CMP     BYTE PTR [BX],0CFH          ;|
10327:    JE      Pro_Stack25                 ;|
10328:    CMP     WORD PTR [BX+6],' BK'       ;|
10329:    JE      Pro_Stack24                 ;| 判定是否可以接管 INT 74H 中断服务程序
10330:    CMP     DX,ROM_SEG                  ;| 若不可以,则跳转
10331:    JNE     Pro_Stack24                 ;|
10332:    PUSH    ES                          ;|
10333:    PUSH    DX                          ;|
10334:    MOV     DX,ROM_SEG                  ;|
10335:    MOV     ES,DX                       ;|
10336:    CMP     BX,ES:ROMAddr_FF01          ;|
10337:    POP     DX                          ;|
10338:    POP     ES                          ;|
10339:    JE      Pro_Stack25                 ;|
10340: Pro_Stack24:                           ;┘
10341:    POP     DS
10342:    MOV     DI, offset Old_INT74H_Vector1   ;┐
10343:    MOV     BX, offset Old_INT74H_Vector2   ;| 安装 INT 74H 中断接管程序(为了切换堆栈)
10344:    MOV     DX, offset INT74H_Entry     ;|
10345:    CALL    Modify_Vector               ;┘
10346:    JMP     SHORT Pro_Stack26
10347: Pro_Stack25:
10348:    POP     DS
10349: Pro_Stack26:
10350:    MOV     SI,Vector_76H_OFS           ;┐
10351:    PUSH    DS                          ;|
10352:    LDS     BX,DWORD PTR ES:[SI]        ;|
10353:    PUSH    DS                          ;|
10354:    POP     DX                          ;|
10355:    CMP     DX,0                        ;|
10356:    JE      Pro_Stack28                 ;|
10357:    CMP     BYTE PTR [BX],0CFH          ;|
10358:    JE      Pro_Stack28                 ;|
10359:    CMP     WORD PTR [BX+6],' BK'       ;|
10360:    JE      Pro_Stack27                 ;| 判定是否可以接管 INT 76H 中断服务程序
10361:    CMP     DX,ROM_SEG                  ;| 若不可以,则跳转
10362:    JNE     Pro_Stack27                 ;|
10363:    PUSH    ES                          ;|
10364:    PUSH    DX                          ;|
10365:    MOV     DX,ROM_SEG                  ;|
```

```
10366:      MOV     ES,DX                       ;|
10367:      CMP     BX,ES:ROMAddr_FF01          ;|
10368:      POP     DX                          ;|
10369:      POP     ES                          ;|
10370:      JE      Pro_Stack28                 ;|
10371:  Pro_Stack27:                            ;┘
10372:      POP     DS
10373:      MOV     DI, offset Old_INT76H_Vector1   ;┐
10374:      MOV     BX, offset Old_INT76H_Vector2   ;| 安装 INT 76H 中断接管程序(为了切换堆栈)
10375:      MOV     DX, offset INT76H_Entry     ;|
10376:      CALL    Modify_Vector               ;┘
10377:      JMP     SHORT Pro_Stack29
10378:  Pro_Stack28:
10379:      POP     DS
10380:  Pro_Stack29:
10381:      MOV     SI,Vector_77H_OFS           ;┐
10382:      PUSH    DS                          ;|
10383:      LDS     BX,DWORD PTR ES:[SI]        ;|
10384:      PUSH    DS                          ;|
10385:      POP     DX                          ;|
10386:      CMP     DX,0                        ;|
10387:      JE      Pro_Stack31                 ;|
10388:      CMP     BYTE PTR [BX],0CFH          ;|
10389:      JE      Pro_Stack31                 ;|
10390:      CMP     WORD PTR [BX+6],' BK'       ;|
10391:      JE      Pro_Stack30                 ;| 判定是否可以接管 INT 77H 中断服务程序
10392:      CMP     DX,ROM_SEG                  ;| 若不可以,则跳转
10393:      JNE     Pro_Stack30                 ;|
10394:      PUSH    ES                          ;|
10395:      PUSH    DX                          ;|
10396:      MOV     DX,ROM_SEG                  ;|
10397:      MOV     ES,DX                       ;|
10398:      CMP     BX,ES:ROMAddr_FF01          ;|
10399:      POP     DX                          ;|
10400:      POP     ES                          ;|
10401:      JE      Pro_Stack31                 ;|
10402:  Pro_Stack30:                            ;┘
10403:      POP     DS
10404:      MOV     DI, offset Old_INT77H_Vector1   ;┐
10405:      MOV     BX, offset Old_INT77H_Vector2   ;| 安装 INT 77H 中断接管程序(为了切换堆栈)
10406:      MOV     DX, offset INT77H_Entry     ;|
10407:      CALL    Modify_Vector               ;┘
10408:      JMP     SHORT Pro_Stack32
```

```
Pro_Stack31:
        POP     DS
Pro_Stack32:
        PUSH    DS                              ;┐
        MOV     AX,ROM_SEG                      ;│
        MOV     DS,AX                           ;│是PC兼容机吗?
        CMP     BYTE PTR DS:ROMAddr_FFFE,0F9H   ;│
        POP     DS                              ;┘
        JNZ     Pro_Stack33                     ;若不是,则跳转
        MOV     AL,27H
        OUT     72H,AL
Pro_Stack33:
        STI
        MOV     AX, BIO_SEG                     ;┐
        MOV     DS, AX                          ;│设置IO1模块中的标志变量(指定了动态堆栈)
        MOV     BYTE PTR StackFlag,1            ;┘
        POP     BP
        POP     SI
        POP     DI
        POP     DX
        POP     CX
        POP     BX
        POP     ES
        POP     DS
        POP     AX
        RET
Process_Stack   ENDP
;=============================================================
;相对地址偏移:1610
;功能:修改中断向量
;入口参数:DS:BX=存放原来中断向量的地址;
;         DS:DX=新的中断向量;
;         ES:SI=存放指定中断向量的地址;
;         70H:DI指向在IO1模块中存放原来中断向量的地址
;出口参数:无
;=============================================================
Modify_Vector   PROC    NEAR
        MOV     AX,ES:[SI]                      ;┐
        MOV     [BX],AX                         ;│取出ES:SI指定的中断向量,存入DS:BX指定
        MOV     AX,ES:[SI+2]                    ;│的地址
        MOV     [BX+2],AX                       ;┘
        PUSH    DS                              ;┐
        MOV     AX, BIO_SEG                     ;│
```

```
10452:    MOV    DS,AX                          ;|
10453:    MOV    AX,ES:[SI]                     ;| 取出 ES:SI 指定的中断向量,存入70H:DI 指定
10454:    MOV    [DI],AX                        ;| 的地址
10455:    MOV    AX,ES:[SI+2]                   ;|
10456:    MOV    [DI+2],AX                      ;|
10457:    POP    DS                             ;┘
10458:    MOV    ES:[SI],DX                     ;┐ 修改中断向量
10459:    MOV    ES:[SI+2],DS                   ;┘
10460:    RET
10461: Modify_Vector  ENDP
10462: ;=============================================
10463: ;相对地址偏移:1637
10464: ;功能:在 DOS 数据段中为 DOS 使用的各种数据块建立对应的子段控制块
10465: ;入口参数:AL=子段用法的标志符
10466: ;出口参数:无
10467: ;=============================================
10468: CreateSCB  PROC  NEAR
10469:    PUSH   ES
10470:    PUSH   CX
10471:    MOV    CX,CS:SEG_FreeMemory           ;┐ 保存当前子段控制块的段地址值
10472:    MOV    CS:Seg_CurrSCB,CX              ;┘
10473:    MOV    ES,CX                          ;ES 指向当前子段控制块
10474:    MOV    BYTE PTR ES:[0],AL             ;"表示子段用法"的标志符→子段控制块
10475:    INC    CX                             ;┐ "子段内存块"的段地址→子段控制块
10476:    MOV    WORD PTR ES:[1],CX             ;┘
10477:    POP    CX
10478:    POP    ES
10479:    INC    CS:SEG_FreeMemory              ;修改自由内存的段地址值
10480:    RET
10481: CreateSCB  ENDP
10482: ;相对地址偏移:1657
10483:                   DB    0               ;未使用
10484: DriverCounter     DB    0               ;设备驱动程序计数器,它记录了当前正在安装的
                                               ;设备驱动程序文件支持的设备驱动程序数
10485: Seg_CurrSCB       DW    0               ;存放当前子段控制块的段地址值
10486: FlagOfSCB         DB    0               ;设置子段控制块大小的标志(bit1=1,设置,bit0=
                                               ;0,不设置)
10487:                   DB    0               ;未使用
10488: Ptr_ProcessConfig DD    0               ;保存系统配置命令字符串的待处理内容的起始地
                                               ;址。当正在处理的系统配置命令格式错时,通过它
                                               ;能输出不能识别的字符串
10489: ParaNumber        DW    0               ;参数项的序号,它是当前正在取出的参数项的序
                                               ;号
```

```
10490: GetParaRetCode       DW   0          ;取参数值的返回码(0:正常;-1:命令正常结束;
                                            ;1:有多余的参数项;2:缺少参数项;3:命令参数中
                                            ;有非法的开关参数;6:参数值越界;8:非法的开关
                                            ;状态关键字,即参数值不是 ON 或 OFF;9:参数值
                                            ;错)
10491: EndAddrOfParaItem    DW   0          ;存放参数项字符串在命令串中的结束地址
10492: StartAddrOfPVIB      DW   0          ;存放参数项的参数值信息块的起始地址
10493: ParaSeparator        DB   0          ;存放当前使用的参数分隔符或系统保留字符
10494: Ptr_DBCS             DD   0          ;保存双字节字符集(DBCS)的引导字节表的起始地
                                            ;址
10495: NoUse                DB   0          ;该变量无实际意义
10496: ParaFeatureFlag      DB   0          ;参数项字符串的特征标志,其各位的意义如下:
10497:                                      ;7 6 5 4 3 2 1 0
10498:                                      ;              └─-1:参数项中含有
                                            ;                  "="字符
10499:                                      ;            └──参数值正负标志(0:正;
                                            ;                  1:负)
10500:                                      ;          └──── 1:参数项为"关键字=…"格
                                            ;                式
10501:                                      ;        └────1:开关参数未匹配
10502:                                      ;      └──1:参数分隔符",",标志,它表明当前参
                                            ;          数项结束,通常它用来表明一个参
                                            ;          数项被缺省
10503:                                      ;    └──1:参数项为开关参数
10504:                                      ;  └──1:参数值为整数(正负均可)
10505: StartAddrOfPara      DW   0          ;存放参数项字符串在命令串中的起始地址
10506: StartAddrOfKeyword   DW   0          ;保存关键字或开关参数说明串的后续参数字符串
                                            ;的起始地址
10507: EndAddrOfTempBuffer  DW   0          ;存放参数项暂存区缓存内容的结束地址(无实际
                                            ;意义)
10508: StartAddrOfSwitchPara DW  0          ;存放开关参数对应的字符串的起始地址
10509: ParaStrBuf           DB   128 DUP (0);参数项字符串缓冲区
10510: ;扩充国家信息的缓存区
10511: ExtCountryInfoBuf    DB   0FFH       ;扩充国家信息的 ID 值(参见功能调用65H 的入口
                                            ;参数 AL 值)
10512:                      DD   0          ;存放扩充国家信息(如大写表的起始地址等)
10513: ReservedChars        DB   ' []|<>+=;,';系统保留字符表
10514: ParaDefaultFlag      DB   0          ;参数项的缺省标志(0:指定了参数项;1:参数项被
                                            ;缺省)
10515: ;=============================================================
10516: ;相对地址偏移:1707
10517: ;功能:取出指定参数项的参数值
10518: ;入口参数:CX=先前已取出的参数项数
```

```
10519: ;          DS:SI 指向当前参数项的字符串(即待处理的命令说明串)
10520: ;          ES:DI 指向命令的参数信息块
10521: ;出口参数:无
10522: ;=============================================================
10523: GetIndicateParaValue  PROC  NEAR
10524:     MOV    WORD PTR CS:NoUse,0
10525:     CLD
10526:     MOV    CS:ParaNumber,CX          ;保存先前已取出的参数项数(即设置当前参数项
                                           ;序号)
10527:     MOV    CS:GetParaRetCode,0       ;清"取参数值的返回码"
10528:     MOV    CS:StartAddrOfSwitchPara,0 ;清开关参数对应的字符串的起始地址
10529:     MOV    CS:StartAddrOfPV1B,0      ;清参数项的参数值信息块的起始地址
10530:     MOV    WORD PTR CS:ReservedChars,' ]['      ;┐
10531:     MOV    WORD PTR CS:ReservedChars+2,' <|'    ;| 设置"系统保留字符表"
10532:     MOV    WORD PTR CS:ReservedChars+4,' +>'    ;|
10533:     MOV    WORD PTR CS:ReservedChars+6,' ;='    ;┘
10534:     CALL   OmitSeparators            ;跳过所有参数分隔符,并检查命令说明串是否已
                                           ;结束
10535:     JNC    GetIndParaV_2             ;若命令说明串未结束,则跳转
10536:     MOV    AX,-1                     ;AX←-1(命令已结束,参数项属于正常缺省)
10537:     PUSH   BX
10538:     MOV    BX,ES:[DI]                ;┐ 若参数属于正常缺省(即当前欲取出的参数项
10539:     CMP    CL,ES:[BX]                ;| 属于可选项),则跳转
10540:     JAE    GetIndParaV_1             ;┘
10541:     MOV    AX,2                      ;AX←2(命令已结束,参数项属于错误缺省,即参
                                           ;数项不够数)
10542: GetIndParaV_1:
10543:     POP    BX
10544:     JMP    GetIndParaV_RET           ;转去直接返回
10545: GetIndParaV_2:
10546:     MOV    CS:StartAddrOfPara,SI     ;保存参数项字符串在命令字符串中的起始地址
10547:     PUSH   BX
10548:     PUSH   DI
10549:     PUSH   BP
10550:     LEA    BX,CS:ParaStrBuf          ;BX←保存参数项字符串的暂存缓冲区的写指针值
10551:     TEST   BYTE PTR CS:ParaFeatureFlag,20H ;┐ 若当前欲取出的参数项被缺省,则跳转
10552:     JNZ    GetIndParaV_8             ;┘
10553: GetIndParaV_3:                       ;                                    ┐
10554:     LODSB                            ;AL←参数说明串的当前字符             |
10555:     CALL   TestSwitchPreface         ;┐ 若开关参数格式错,则跳转           |
10556:     JC     GetIndParaV_7             ;┘                                   |
10557:     CALL   ChkTerminator             ;检查命令说明串是否结束              |
10558:     JZ     GetIndParaV_7             ;若命令结束,则跳转                   |
```

```
10559:      CALL    TestSeparator                   ;┐ 若 AL 指定的字符不是参数分隔符,则 |
10560:      JNZ     GetIndParaV_5                   ;┘ 跳转                              |
10561:      TEST    BYTE PTR CS:ParaFeatureFlag,20H ;┐ 若参数分隔符是",",则跳转          |
10562:      JNZ     GetIndParaV_4                   ;┘                                   |
10563:      CALL    OmitSeparators                  ;跳过所有后续的参数分隔符            | 取
10564:      JMP     SHORT GetIndParaV_8             ;                                    | 出
10565: GetIndParaV_4:                               ;                                    | 参
10566:      TEST    BYTE PTR CS:ParaFeatureFlag,41H ;┐                                   | 数
10567:      JZ      GetIndParaV_8                   ;| 恢复先前取出的一个字符,它是       | 项
10568:      DEC     SI                              ;| 属于下一个参数项的                | 的
10569:      JMP     SHORT GetIndParaV_8             ;┘                                   | 组
10570: GetIndParaV_5:                               ;                                    | 成
10571:      MOV     CS:[BX],AL                      ;保存当前合法的参数项字符            | 字
10572:      CMP     AL,' ='                         ;┐ 测试参数项字符串中是否有"="。    | 符
10573:      JNE     GetIndParaV_6                   ;| 若有,置相应的标志位              |
10574:      OR      BYTE PTR CS:ParaFeatureFlag,1   ;┘                                   |
10575: GetIndParaV_6:                               ;                                    |
10576:      INC     BX                              ;修改参数项字符串的暂存缓冲区的写指  |
                                                    ;针值                                |
10577:      CALL    TestDBCSBootByte                ;┐ 若 AL 指定的字符是单字节字符,则跳 |
10578:      JNC     GetIndParaV_3                   ;┘ 转                                |
10579:      LODSB                                   ;┐                                   |
10580:      MOV     CS:[BX],AL                      ;| 取出并保存双字节字符的高字节值   |
10581:      INC     BX                              ;┘                                   |
10582:      JMP     SHORT GetIndParaV_3             ;转去继续取参数项的后续组成字符      |
10583: GetIndParaV_7:                               ;                                    |
10584:      DEC     SI                              ;恢复刚才取出的不属于当前参数项的字符|
10585: GetIndParaV_8:                               ;                                    ┘
10586:      MOV     CS:EndAddrOfParaItem,SI         ;保存参数项字符串在命令字符串中的结束地址
10587:      MOV     BYTE PTR CS:[BX],0              ;设置参数项字符串的结束符
10588:      MOV     CS:EndAddrOfTempBuffer,BX       ;保存参数项暂存区的写访问指针值
10589:      MOV     BX,ES:[DI]                      ;ES:BX 指向命令参数的组成控制信息块
10590:      LEA     SI,CS:ParaStrBuf                ;CS:SI 指向参数项字符串
10591:      CMP     BYTE PTR CS:[SI],' /'           ;┐ 若参数是开关参数,则跳转
10592:      JE      GetIndParaV_11                  ;┘
10593:      CMP     BYTE PTR CS:[SI],' "'
10594:      JE      GetIndParaV_9
10595:      TEST    BYTE PTR CS:ParaFeatureFlag,1   ;┐ 若参数项含有"="字符,则跳转
10596:      JNZ     GetIndParaV_14                  ;┘
10597: GetIndParaV_9:                               ;处理一般的参数项,如整数型、字符串型
10598:      MOV     AL,BYTE PTR ES:[BX+1]           ;┐ AX←命令允许的最大参数项个数
10599:      XOR     AH,AH                           ;┘
10600:      CMP     CS:ParaNumber,AX                ;┐ 若当前处理的参数项是多余的命令参数,则跳
```

```
10601:      JAE     GetIndParaV_10              ;┘ 转
10602:      MOV     AX,CS:ParaNumber            ;┐
10603:      SHL     AX,1                        ;|
10604:      INC     BX                          ;| ES:BX 指向当前参数项的参数项信息块
10605:      INC     BX                          ;|
10606:      ADD     BX,AX                       ;|
10607:      MOV     BX,ES:[BX]                  ;┘
10608:      CALL    GetNotSwitchParaValue1      ;取出当前参数项的参数值
10609:      JMP     SHORT GetIndParaV_17
10610: GetIndParaV_10:
10611:      MOV     CS:GetParaRetCode,1         ;设置"取参数值的返回码"(1:表示有多余的参数
                                                ;项)
10612:      JMP     SHORT GetIndParaV_17
10613: GetIndParaV_11:                          ;处理开关参数(如"/X"、"/K"等)
10614:      MOV     AL,ES:[BX+1]                ;┐
10615:      XOR     AH,AH                       ;|
10616:      INC     AX                          ;|
10617:      SHL     AX,1                        ;| CX←命令允许的开关参数个数
10618:      ADD     BX,AX                       ;|
10619:      MOV     CL,ES:[BX]                  ;|
10620:      XOR     CH,CH                       ;┘
10621:      OR      CX,CX                       ;┐ 若命令不允许有开关参数,则跳转
10622:      JZ      GetIndParaV_13              ;┘
10623:      INC     BX
10624: GetIndParaV_12:                          ;┐
10625:      PUSH    BX                          ;|
10626:      MOV     BX,ES:[BX]                  ;|
10627:      CALL    GetSwitchParaValue          ;| 取关键字参数值
10628:      POP     BX                          ;|
10629:      JNC     GetIndParaV_17              ;|
10630:      INC     BX                          ;|
10631:      INC     BX                          ;|
10632:      LOOP    GetIndParaV_12              ;┘
10633: GetIndParaV_13:
10634:      MOV     CS:GetParaRetCode,3         ;设置"取参数值的返回码"(3:表示非法开关参数)
10635:      JMP     SHORT GetIndParaV_17
10636: GetIndParaV_14:                          ;处理"关键字=…"格式的参数项(注意 MS—DOS
                                                ;5.00不支持该类参数)
10637:      MOV     AL,ES:[BX+1]                ;AL←命令允许的非开关参数个数
10638:      XOR     AH,AH                       ;┐
10639:      INC     AX                          ;| 计算存放"命令允许的开关参数个数"的单元地
10640:      SHL     AX,1                        ;| 址→ES:BX
10641:      ADD     BX,AX                       ;┘
```

```
10642:    MOV     AL,ES:[BX]                ;AX←命令允许的开关参数个数
10643:    XOR     AH,AH                     ;┐
10644:    SHL     AX,1                      ;| 计算存放"命令允许的含有' ='字符的参数
10645:    INC     AX                        ;| 个数"的单元地址→ES:BX
10646:    ADD     BX,AX                     ;┘
10647:    MOV     CL,ES:[BX]                ;┐ CX←命令允许的含有"="字符的参数个数
10648:    XOR     CH,CH                     ;┘
10649:    OR      CX,CX                     ;┐ 若命令不允许含有"="字符的参数,则跳转
10650:    JZ      GetIndParaV_16            ;┘
10651:    INC     BX
10652: GetIndParaV_15:                      ;┐
10653:    PUSH    BX                        ;|
10654:    MOV     BX,ES:[BX]                ;|
10655:    CALL    IllegalPara               ;| 取"关键字=…"格式的参数值
10656:    POP     BX                        ;|
10657:    JNC     GetIndParaV_17            ;|
10658:    INC     BX                        ;|
10659:    INC     BX                        ;|
10660:    LOOP    GetIndParaV_15            ;┘
10661: GetIndParaV_16:
10662:    MOV     CS:GetParaRetCode,4       ;设置"取参数值的返回码"(4:表示非法的"含有
                                            ;' ='字符的参数")
10663: GetIndParaV_17:
10664:    POP     BP
10665:    POP     DI
10666:    POP     BX
10667:    MOV     CX,CS:ParaNumber          ;CX←当前取出的参数项序号
10668:    MOV     AX,CS:GetParaRetCode      ;AX←取参数值的返回码
10669:    MOV     SI,CS:EndAddrOfParaItem   ;SI←下一个参数项字符串的起始地址
10670:    MOV     DX,CS:StartAddrOfPVIB     ;DX←当前参数项的参数值信息块的起始地址
10671:    MOV     BL,CS:ParaSeparator       ;BL←当前使用的参数分隔符或系统保留字符
10672: GetIndParaV_RET:
10673:    CLC
10674:    RET
10675: GetIndicateParaValue  ENDP
10676: ;=============================================================
10677: ;相对地址偏移:1882
10678: ;功能:取非开关参数项的参数值
10679: ;入口参数:CS:SI 指向参数项字符串;
10680: ;          ES:BX 指向参数项信息块
10681: ;出口参数:无
10682: ;=============================================================
10683: GetNotSwitchParaValue1  PROC  NEAR
```

```
10684:    PUSH    AX
10685:    MOV     AX,ES:[BX]                ;AX←参数项控制字
10686:    TEST    AX,2                      ;┐若本参数项不影响参数项编号,则跳转
10687:    JNZ     GetNotSwitchParav1_1      ;┘
10688:    INC     CS:ParaNumber             ;已取出的参数项计数值加1(即设定下一个参数项
                                            ;的序号)
10689: GetNotSwitchParav1_1:
10690:    CMP     BYTE PTR CS:[SI],0        ;┐若当前参数项有内容,则跳转
10691:    JNE     GetNotSwitchParav1_3      ;┘
10692:    TEST    AX,1                      ;┐若当前参数项允许缺省,则跳转
10693:    JNZ     GetNotSwitchParav1_2      ;┘
10694:    MOV     CS:GetParaRetCode,2       ;设置"取参数的返回码"(2:表示"参数项不够数")
10695:    JMP     SHORT GetNotSwitchParav1_4
10696: GetNotSwitchParav1_2:
10697:    PUSH    AX                        ;┐
10698:    MOV     AL,3                      ;│设置参数值信息块(此时 AL=3表示参数项被
10699:    MOV     AH,0FFH                   ;│正常缺省)
10700:    CALL    SetParaInfoBlk            ;│
10701:    POP     AX                        ;┘
10702:    JMP     SHORT GetNotSwitchParav1_4
10703: GetNotSwitchParav1_3:
10704:    CALL    GetNotSwitchParaValue2    ;取出参数值
10705: GetNotSwitchParav1_4:
10706:    POP     AX
10707:    RET
10708: GetNotSwitchParaValue1  ENDP
10709: ;=============================================================
10710: ;相对地址偏移:18B4
10711: ;功能:取"关键字=…"格式的参数值(注意:由于 MS-DOS 5.00不支持该类参数,因而此程序仅作
           为一个虚拟扩充的接口程序)
10712: ;入口参数:无
10713: ;出口参数:CF=1
10714: ;=============================================================
10715: IllegalPara  PROC  NEAR
10716:    STC
10717:    RET
10718: IllegalPara  ENDP
10719: ;=============================================================
10720: ;相对地址偏移:18B6
10721: ;功能:判定开关参数是否合法
10722: ;入口参数:CS:SI 指向开关参数的字符串
10723: ;          ES:BX 指向开关参数的参数项信息块
10724: ;出口参数:CF=0,开关参数已匹配;CF=1,开关参数未匹配
```

```
10725: ;=========================================================
10726: TestParaLegal  PROC  NEAR
10727:     PUSH    BP
10728:     PUSH    CX
10729:     MOV     CL,ES:[BX+8]          ;┐ CX←命令允许的开关参数说明串的个数
10730:     XOR     CH,CH                 ;┘
10731:     OR      CX,CX                 ;┐ 若没有开关参数说明串,则跳转
10732:     JZ      TestParaLegal_2       ;┘
10733:     LEA     BP,[BX+9]             ;ES:BP 指向第一个开关参数说明串
10734: TestParaLegal_1:
10735:     CALL    TestKeywordMatch      ;┐ 若开关参数说明串匹配,则跳转
10736:     JNC     TestParaLegal_3       ;┘
10737:     CALL    PointToNext           ;将 ES:BP 指向下一个命令允许的开关参数说明串
10738:     LOOP    TestParaLegal_1
10739: TestParaLegal_2:
10740:     STC
10741:     JMP     SHORT TestParaLegal_4
10742: TestParaLegal_3:
10743:     MOV     CS:StartAddrOfSwitchPara,BP
10744:     CLC
10745: TestParaLegal_4:
10746:     POP     CX
10747:     POP     BP
10748:     RET
10749: TestParaLegal  ENDP
10750: ;=========================================================
10751: ;相对地址偏移:18DB
10752: ;功能:将 ES:BP 指向下一个命令允许的开关参数说明串
10753: ;入口参数:ES:BP 指向当前命令允许的开关参数说明串
10754: ;出口参数:ES:BP 指向下一个命令允许的开关参数说明串
10755: ;=========================================================
10756: PointToNext  PROC  NEAR
10757: PointToNext_1:
10758:     CMP     BYTE PTR ES:[BP+0],0
10759:     JE      PointToNext_2
10760:     INC     BP
10761:     JMP     SHORT PointToNext_1
10762: PointToNext_2:
10763:     INC     BP
10764:     RET
10765: PointToNext  ENDP
10766: ;=========================================================
10767: ;相对地址偏移:18E7
```

```
;功能:取开关参数值
;入口参数:CS:SI 指向开关参数的字符串
;          ES:BX 指向开关参数的参数项信息块
;出口参数:CF=0,开关参数已匹配;CF=1,开关参数未匹配
;=========================================================
GetSwitchParaValue  PROC  NEAR
    OR      CS:ParaFeatureFlag,10H          ;置开关参数未匹配的标志位
    CALL    TestParaLegal                   ;┐若开关参数未匹配,则跳转
    JC      GetSwitchParaV_3                ;┘
    AND     CS:ParaFeatureFlag,0EFH         ;清开关参数未匹配的标志位
    PUSH    AX                              ;┐
    MOV     AX,CS:StartAddrOfKeyword        ;│修改参数项字符串在命令串中的起始地址(即
    SUB     AX,SI                           ;│设置开关参数的后续参数字符串的起始地址)
    ADD     CS:StartAddrOfPara,AX           ;│
    POP     AX                              ;┘
    MOV     SI,CS:StartAddrOfKeyword        ;┐
    CMP     BYTE PTR CS:[SI],0              ;│若开关参数还有后续参数,则跳转
    JNE     GetSwitchParaV_2                ;┘
    CMP     BYTE PTR CS:[SI-1],' :'         ;┐若开关参数的后续参数格式正确,则跳转
    JNE     GetSwitchParaV_1                ;┘
    MOV     CS:GetParaRetCode,9             ;置"取参数值的返回码"(9:参数值错)
    JMP     SHORT GetSwitchParaV_4
GetSwitchParaV_1:
    CMP     WORD PTR ES:[BX],0              ;┐
    JE      GetSwitchParaV_4                ;│
    TEST    WORD PTR ES:[BX],1              ;│若开关参数不允许有后续参数,则跳转
    JNZ     GetSwitchParaV_4                ;│
    MOV     CS:GetParaRetCode,2             ;│
    JMP     SHORT GetSwitchParaV_4          ;┘
GetSwitchParaV_2:
    CALL    GetNotSwitchParaValue2          ;取出开关参数的后续参数值
    CLC
    JMP     SHORT GetSwitchParaV_RET
GetSwitchParaV_3:
    STC
    JMP     SHORT GetSwitchParaV_RET
GetSwitchParaV_4:
    PUSH    AX                              ;┐
    MOV     AL,3                            ;│设置参数值信息块(此时 AL=3表示参数项无
    MOV     AH,0FFH                         ;│效)
    CALL    SetParaInfoBlk                  ;│
    POP     AX                              ;┘
    CLC
```

```
10811: GetSwitchParaV_RET:
10812:     RET
10813: GetSwitchParaValue   ENDP
10814: ;=============================================================
10815: ;相对地址偏移:194A
10816: ;功能:设置参数值信息块
10817: ;入口参数:AL=参数值类型码
10818: ;              1:参数值为整型数。此时,CX:DX=整型数值
10819: ;              2:参数值为关键字(如 ON/OFF,或者 HIGH/LOW、UMB/NOUMB)。此时,AH=关键
                       字的编号;DX=关键字字符串的起始地址
10820: ;              3:参数值无效或参数值为开关参数。此时,CS:SI 指向参数项字符串
10821: ;              4:MS-DOS5.00未使用
10822: ;              5:参数值为 ASCIIZ 字符串(如文件名)。此时,CS:SI 指向 ASCIIZ 字符串
10823: ;              6:参数值为最大逻辑驱动器数。此时,DL=最大逻辑驱动器数
10824: ;          ES:BX 指向参数项信息块
10825: ;出口参数:无
10826: ;=============================================================
10827: SetParaInfoBlk   PROC   NEAR
10828:     PUSH    DI
10829:     MOV     DI,ES:[BX+4]                ;┐ 设置"参数值信息块"的起始地址
10830:     MOV     CS:StartAddrOfPVIB,DI       ;┘
10831:     MOV     ES:[DI],AL                  ;设置参数值类型码
10832:     MOV     ES:[DI+1],AH                ;设置关键字的编号(只对 AL=2时有效)
10833:     PUSH    AX                          ;┐
10834:     MOV     AX,CS:StartAddrOfSwitchPara ;│ 保存开关参数对应的字符串的起始地址
10835:     MOV     ES:[DI+2],AX                ;│
10836:     POP     AX                          ;┘
10837:     CMP     AL,1
10838:     JNE     SetParaInfoBlk_2
10839: SetParaInfoBlk_1:
10840:     MOV     ES:[DI+4],DX                ;┐ 保存整型参数值
10841:     MOV     ES:[DI+6],CX                ;┘
10842:     JMP     SHORT SetParaInfoBlk_9
10843: SetParaInfoBlk_2:
10844:     CMP     AL,2                        ;┐
10845:     JNE     SetParaInfoBlk_3            ;│
10846:     MOV     ES:[DI+4],DX                ;│ 设置关键字字符串的起始地址
10847:     JMP     SHORT SetParaInfoBlk_9      ;│
10848: SetParaInfoBlk_3:                       ;┘
10849:     CMP     AL,7
10850:     JE      SetParaInfoBlk_1
10851:     CMP     AL,8
10852:     JE      SetParaInfoBlk_1
```

```
10853:    CMP     AL,6                          ;┐
10854:    JNE     SetParaInfoBlk_4              ;|
10855:    MOV     ES:[DI+4],DL                  ;| 保存允许访问的最大逻辑驱动器数
10856:    JMP     SHORT SetParaInfoBlk_9        ;|
10857: SetParaInfoBlk_4:                        ;┘
10858:    CMP     AL,4
10859:    JNE     SetParaInfoBlk_5
10860:    MOV     AX,CS:StartAddrOfPara
10861:    INC     AX
10862:    MOV     ES:[DI+4],AX
10863:    MOV     ES:[DI+6],DS
10864:    JMP     SHORT SetParaInfoBlk_9
10865: SetParaInfoBlk_5:
10866:    MOV     ES:[DI+4],SI                  ;┐ 保存参数项字符串(AL=3)或 ASCIIZ 字符串
10867:    MOV     ES:[DI+6],CS                  ;┘ (AL=5)的起始地址
10868:    PUSH    AX
10869:    TEST    BYTE PTR ES:[BX+2],1
10870:    JZ      SetParaInfoBlk_6
10871:    MOV     AL,4
10872:    JMP     SHORT SetParaInfoBlk_7
10873: SetParaInfoBlk_6:
10874:    TEST    BYTE PTR ES:[BX+2],2
10875:    JZ      SetParaInfoBlk_8
10876:    MOV     AL,2
10877: SetParaInfoBlk_7:
10878:    CALL    ChangeStrToCaps               ;将 CS:SI 指定的参数字符串转换成大写
10879: SetParaInfoBlk_8:
10880:    POP     AX
10881:    TEST    BYTE PTR ES:[BX+2],10H        ;┐
10882:    JZ      SetParaInfoBlk_9              ;| 丢掉设备名的结尾符":"
10883:    CALL    CutTailChar                   ;|
10884: SetParaInfoBlk_9:                        ;┘
10885:    POP     DI
10886:    RET
10887: SetParaInfoBlk  ENDP
10888: ;=============================================================
10889: ;相对地址偏移:19CF
10890: ;功能:取非开关参数项的参数值
10891: ;入口参数:CS:SI 指向参数项字符串
10892: ;          ES:BX 指向参数项信息块
10893: ;出口参数:无
10894: ;=============================================================
10895: GetNotSwitchParaValue2  PROC  NEAR
```

```
10896:    MOV     CS:ParaDefaultFlag,0          ;清参数项的缺省标志(0:指定了参数项)
10897:    PUSH    AX
10898:    MOV     AX,ES:[BX]                    ;AX←参数项控制字
10899:    OR      AX,AX                         ;┐若参数项的控制字值有效,则跳转
10900:    JNZ     GetNotSwitchParaV2_1          ;┘
10901:    PUSH    AX
10902:    PUSH    BX
10903:    PUSH    DX
10904:    PUSH    DI
10905:    MOV     CS:GetParaRetCode,9           ;设置"取参数值的返回码"(9:表示参数值有错)
10906:    MOV     AH,0FFH                       ;┐设置参数值信息块(此时 AL=3表示参数项信
10907:    MOV     AL,3                          ;│息块错)
10908:    CALL    SetParaInfoBlk                ;┘
10909:    POP     DI
10910:    POP     DX
10911:    POP     BX
10912:    POP     AX
10913:    JMP     SHORT GetNotSwitchParaV2_2
10914: GetNotSwitchParaV2_1:
10915:    JMP     SHORT GetNotSwitchParaV2_3
10916: GetNotSwitchParaV2_2:
10917:    JMP     GetNotSwitchParaV2_8
10918: GetNotSwitchParaV2_3:
10919:    TEST    AX,8000H                      ;┐若参数值不是正整数,则跳转
10920:    JZ      GetNotSwitchParaV2_4          ;┘
10921:    MOV     CS:GetParaRetCode,0           ;清"取参数值的返回码"(0:正常)
10922:    CALL    GetPlusIntegerValue           ;取出正整数参数值
10923:    CMP     CS:GetParaRetCode,9           ;┐若不是参数值错,则跳转
10924:    JNE     GetNotSwitchParaV2_8          ;┘
10925: GetNotSwitchParaV2_4:
10926:    TEST    AX,4000H                      ;┐若参数值不是整数,则跳转
10927:    JZ      GetNotSwitchParaV2_5          ;┘
10928:    MOV     CS:GetParaRetCode,0           ;清"取参数值的返回码"(0:正常)
10929:    CALL    GetIntegerParaValue           ;取出整数参数值
10930:    CMP     CS:GetParaRetCode,9           ;┐若不是参数值错,则跳转
10931:    JNE     GetNotSwitchParaV2_8          ;┘
10932: GetNotSwitchParaV2_5:
10933:    TEST    AX,100H                       ;┐若参数值不是逻辑驱动器名符,则跳转
10934:    JZ      GetNotSwitchParaV2_6          ;┘
10935:    MOV     CS:GetParaRetCode,0           ;清"取参数值的返回码"(0:正常)
10936:    CALL    VerifyParaStr                 ;校证 ASCIIZ 字符串
10937:    CALL    GetMaxLogicDrivePara          ;取出"允许访问的最大逻辑驱动器数"的参数值
10938:    CMP     CS:GetParaRetCode,9           ;┐若不是参数值错,则跳转
```

```
10939:    JNE     GetNotSwitchParaV2_8          ;┘
10940: GetNotSwitchParaV2_6:
10941:    TEST    AX,200H                       ;┐ 若参数值不是文件名 ASCIIZ 字符串,则跳转
10942:    JZ      GetNotSwitchParaV2_7          ;┘
10943:    MOV     CS:GetParaRetCode,0           ;清"取参数值的返回码"(0:正常)
10944:    CALL    VerifyParaStr                 ;校证文件名的 ASCIIZ 字符串,并保存文件名参数
                                                ;值
10945:    CMP     CS:GetParaRetCode,9           ;┐ 若不是参数值错,则跳转
10946:    JNE     GetNotSwitchParaV2_8          ;┘
10947: GetNotSwitchParaV2_7:
10948:    TEST    AX,2000H                      ;┐ 若参数值不是开关状态(ON/OFF),则跳转
10949:    JZ      GetNotSwitchParaV2_8          ;┘
10950:    MOV     CS:GetParaRetCode,0           ;清"取参数值的返回码"(0:正常)
10951:    CALL    GetSwitchStatusPara           ;取出开关状态(ON/OFF)或关键字(如 HIGH、UMB
                                                ;等)的参数值
10952: GetNotSwitchParaV2_8:
10953:    CMP     CS:ParaDefaultFlag,1          ;┐ 若参数项非空,则跳转
10954:    JNE     GetNotSwitchParaV2_9          ;┘
10955:    CMP     CS:GetParaRetCode,0           ;┐ 设置"取参数值的返回码"(9:表示参数值错)
10956:    JNE     GetNotSwitchParaV2_9          ;│
10957:    MOV     CS:GetParaRetCode,9           ;┘
10958: GetNotSwitchParaV2_9:
10959:    POP     AX
10960:    RET
10961: GetNotSwitchParaValue2  ENDP
10962: ;=============================================================
10963: ;相对地址偏移:1A81
10964: ;功能:丢掉设备名的结尾符":",如将"lpt1:"转换成"lpt1"
10965: ;入口参数:CS:SI 指向源 ASCIIZ 字符串
10966: ;出口参数:CS:SI 指向转换后的 ASCIIZ 字符串
10967: ;=============================================================
10968: CutTailChar  PROC  NEAR
10969:    PUSH    AX
10970:    PUSH    SI
10971: CutTailChar_1:
10972:    MOV     AL,CS:[SI]                    ;┐
10973:    OR      AL,AL                         ;│ 若字符串已结束,则跳转
10974:    JZ      CutTailChar_4                 ;┘
10975:    CMP     AL,':'                        ;┐
10976:    JNE     CutTailChar_2                 ;│
10977:    CMP     BYTE PTR CS:[SI+1],0          ;│ 丢掉设备名的结尾符":"
10978:    JNE     CutTailChar_2                 ;│
10979:    MOV     BYTE PTR CS:[SI],0            ;│
```

```
10980:    JMP     SHORT CutTailChar_4                ;┘
10981: CutTailChar_2:
10982:    CALL    TestDBCSBootByte                   ;┐ 若 AL 指定的字符为单字节字符,则跳转
10983:    JNC     CutTailChar_3                      ;┘
10984:    INC     SI                                 ;跳过双字节字符的第二个字节
10985: CutTailChar_3:
10986:    INC     SI                                 ;CS:SI 指向下一个字符
10987:    JMP     SHORT CutTailChar_1
10988: CutTailChar_4:
10989:    POP     SI
10990:    POP     AX
10991:    RET
10992: CutTailChar  ENDP
10993: ;=============================================================
10994: ;相对地址偏移:1AA7
10995: ;功能:将 CS:SI 指定的 ASCIIZ 字符串转换成大写
10996: ;入口参数:CS:SI 指向源 ASCIIZ 字符串
10997: ;出口参数:CS:SI 指向转换后的大写 ASCIIZ 字符串
10998: ;=============================================================
10999: ChangeStrToCaps  PROC  NEAR
11000:    PUSH    SI
11001:    PUSH    DX
11002:    MOV     DL,AL
11003: ChangeStrToCaps_1:
11004:    MOV     AL,CS:[SI]                         ;┐
11005:    CALL    TestDBCSBootByte                   ;| 若当前字符为双字节字符的引导字节,则跳转
11006:    JC      ChangeStrToCaps_2                  ;┘
11007:    OR      AL,AL                              ;┐ 若字符串结束了,则结束返回
11008:    JZ      ChangeStrToCaps_4                  ;┘
11009:    CALL    ChangeCharToCap                    ;┐ 将当前字符转换成大写
11010:    MOV     CS:[SI],AL                         ;┘
11011:    JMP     SHORT ChangeStrToCaps_3
11012: ChangeStrToCaps_2:
11013:    INC     SI                                 ;跳过双字节字符的第二个字节
11014: ChangeStrToCaps_3:
11015:    INC     SI                                 ;CS:SI 指向下一个字符
11016:    JMP     SHORT ChangeStrToCaps_1
11017: ChangeStrToCaps_4:
11018:    POP     DX
11019:    POP     SI
11020:    RET
11021: ChangeStrToCaps  ENDP
11022: ;=============================================================
```

```
11023: ;相对地址偏移:1AC6
11024: ;功能:将 AL 指定的字符转换成大写
11025: ;入口参数:AL=字符的 ASCII 码;
11026: ;          DL=2(大写表 ID 值,参见功能调用 65H 的入口参数 AL 值)
11027: ;出口参数:AL=大写字符的 ASCII 码
11028: ;=============================================================
11029: ChangeCharToCap  PROC  NEAR
11030:     CMP     AL,80H                  ;┐若 AL 指定的字符是扩展 ASCII 码,则跳转
11031:     JAE     ChangeCharToCap_1       ;┘
11032:     CMP     AL,'a'                  ;┐
11033:     JB      ChangeCharToCap_3       ;│若 AL 指定的字符不是英文字母,则跳转
11034:     CMP     AL,'z'                  ;│
11035:     JA      ChangeCharToCap_3       ;┘
11036:     AND     AL,0DFH                 ;将 AL 指定的英文字母转换成大写
11037:     JMP     SHORT ChangeCharToCap_3
11038: ChangeCharToCap_1:
11039:     PUSH    BX
11040:     PUSH    ES
11041:     PUSH    DI
11042:     LEA     DI,CS:ExtCountryInfoBuf ;CS:DI 指向存放扩充国家信息的缓冲区
11043:     CMP     CS:[DI],DL              ;┐若 DL 指定的扩充国家信息(即大写表的起始地
11044:     JE      ChangeCharToCap_2       ;┘址)已被取出,则跳转
11045:     PUSH    AX                      ;┐
11046:     PUSH    CX                      ;│
11047:     PUSH    DX                      ;│
11048:     PUSH    CS                      ;│
11049:     POP     ES                      ;│
11050:     MOV     AH,65H                  ;│取出 DL 指定的
11051:     MOV     AL,DL                   ;AL←扩充国家信息 ID 值(2)  │扩充国家信息
11052:     MOV     BX,0FFFFH               ;BX←-1(当前代码页)         │(即大写表的起
11053:     MOV     CX,5                    ;CX←返回的数据长度         │始地址)
11054:     MOV     DX,0FFFFH               ;DX←-1(当前国家信息)       │
11055:     INT     21H                     ;│
11056:     POP     DX                      ;│
11057:     POP     CX                      ;│
11058:     POP     AX                      ;┘
11059: ChangeCharToCap_2:
11060:     MOV     BX,WORD PTR CS:[DI+1]   ;┐ES:BX 指向大写表
11061:     MOV     ES,WORD PTR CS:[DI+3]   ;┘
11062:     INC     BX                      ;┐跳过大写表的字符数,ES:BX 指向大写表的字
11063:     INC     BX                      ;┘符表
11064:     SUB     AL,80H                  ;┐AL←大写字符的 ASCII 码
11065:     XLAT ES:[BX]                    ;┘
```

```
11066:    POP     DI
11067:    POP     ES
11068:    POP     BX
11069: ChangeCharToCap_3:
11070:    RET
11071: ChangeCharToCap  ENDP
11072: ;=============================================================
11073: ;相对地址偏移:1B0B
11074: ;功能:取出整数参数值
11075: ;入口参数:CS:SI 指向参数项字符串
11076: ;          ES:BX 指向参数项信息块
11077: ;出口参数:无
11078: ;=============================================================
11079: GetIntegerParaValue  PROC  NEAR
11080:    PUSH    AX
11081:    OR      CS:ParaFeatureFlag,80H          ;设置"参数值为整数"的标志位
11082:    AND     CS:ParaFeatureFlag,0FDH         ;清"参数值正负"的标志位,表示参数值为正整数
11083:    MOV     AL,CS:[SI]                      ;AL←参数项字符串的第一个字符
11084:    CMP     AL,'+'                          ;┐若 AL 指定的字符为"+",则跳转
11085:    JE      GetIntParaV_1                   ;┘
11086:    CMP     AL,'-'                          ;┐若 AL 指定的字符不为"-",则跳转
11087:    JNE     GetIntParaV_2                   ;┘
11088:    OR      CS:ParaFeatureFlag,02           ;置"参数值正负"的标志位,表示参数值为负整数
11089: GetIntParaV_1:
11090:    INC     SI                              ;跳过参数值的正负符号
11091: GetIntParaV_2:
11092:    CALL    GetPlusIntegerValue             ;读取整数参数值
11093:    POP     AX
11094:    RET
11095: GetIntegerParaValue  ENDP
11096: ;=============================================================
11097: ;相对地址偏移:1B2F
11098: ;功能:取出正整数参数值
11099: ;入口参数:CS:SI 指向参数项字符串
11100: ;          ES:BX 指向参数项信息块
11101: ;出口参数:无
11102: ;=============================================================
11103: GetPlusIntegerValue  PROC  NEAR
11104:    PUSH    AX
11105:    PUSH    CX
11106:    PUSH    DX
11107:    PUSH    SI
11108:    XOR     CX,CX                           ;┐CX:DX←0(清参数值)
```

```
11109:      XOR     DX,DX                   ;┘
11110:      PUSH    BX
11111: GetPlusIntV_1:                       ;                          ┐
11112:      MOV     AL,CS:[SI]              ;AL←当前数字符             |
11113:      OR      AL,AL                   ;┐ 若参数项已结束,则跳转    |
11114:      JZ      GetPlusIntV_3           ;┘                         |
11115:      CALL    CalculateDecimalValue   ;计算 AL 指定的十进制      |
                                            ;数字符的值→AL             |
11116:      JC      GetPlusIntV_2           ;若 AL 指定的字符不是数    |
                                            ;字符,则跳转               |
11117:      XOR     AH,AH                   ;┐ BP←数字值               |
11118:      MOV     BP,AX                   ;┘                         |
11119:      SHL     DX,1                    ;┐                         |
11120:      RCL     CX,1                    ; |                        |
11121:      CALL    TestParaOverflow        ; |                        |
11122:      JC      GetPlusIntV_2           ; |                        |
11123:      MOV     BX,DX                   ; |                        |
11124:      MOV     AX,CX                   ; |                        |
11125:      SHL     DX,1                    ; |                        |
11126:      RCL     CX,1                    ; |                        |
11127:      CALL    TestParaOverflow        ; | CX:DX * 10→CX:DX       |
11128:      JC      GetPlusIntV_2           ; |                        |
11129:      SHL     DX,1                    ; |                        | 计算参数值→
11130:      RCL     CX,1                    ; |                        | CX:DX
11131:      CALL    TestParaOverflow        ; |                        |
11132:      JC      GetPlusIntV_2           ; |                        |
11133:      ADD     DX,BX                   ; |                        |
11134:      ADC     CX,AX                   ; |                        |
11135:      CALL    TestParaOverflow        ; |                        |
11136:      JC      GetPlusIntV_2           ;┘                         |
11137:      ADD     DX,BP                   ;┐ CX:DX+当前数字值        |
11138:      ADC     CX,0                    ;┘ →CX:DX                  |
11139:      CALL    TestParaOverflow        ;┐ 若数据溢出,则跳转       |
11140:      JC      GetPlusIntV_2           ;┘                         |
11141:      INC     SI                      ;SI 指向下一个待处理的数字符 |
11142:      JMP     SHORT GetPlusIntV_1     ;                          |
11143: GetPlusIntV_2:                       ;                          |
11144:      POP     BX                      ;                          |
11145:      JMP     GetPlusIntV_12          ;                          |
11146: GetPlusIntV_3:                       ;                          ┘
11147:      POP     BX
11148:      TEST    CS:ParaFeatureFlag,2    ;┐ 若参数值为正,则跳转
11149:      JZ      GetPlusIntV_4           ;┘
```

```
11150:    NOT     CX                          ;┐
11151:    NOT     DX                          ;├ 参数值取补→CX:DX
11152:    ADD     DX,1                        ;│
11153:    ADC     CX,0                        ;┘
11154: GetPlusIntV_4:
11155:    MOV     SI,ES:[BX+6]
11156:    MOV     AL,ES:[SI]
11157:    CMP     AL,0
11158:    JNE     GetPlusIntV_5
11159:    MOV     AL,1
11160:    MOV     AH,0FFH
11161:    JMP     GetPlusIntV_13
11162: GetPlusIntV_5:
11163:    INC     SI                          ;┐ AL←参数值范围的个数
11164:    MOV     AL,ES:[SI]                  ;┘
11165:    CMP     AL,0                        ;┐ 若参数值大小无规定的范围,则跳转
11166:    JE      GetPlusIntV_12              ;┘
11167:    INC     SI
11168: GetPlusIntV_6:
11169:    TEST    CS:ParaFeatureFlag,80H      ;┐ 若参数值为整数,则跳转
11170:    JNZ     GetPlusIntV_8               ;┘
11171: ;参数值为正整数
11172:    CMP     CX,ES:[SI+3]                ;┐
11173:    JB      GetPlusIntV_10              ;│
11174:    JA      GetPlusIntV_7               ;├ 若参数值小于最小值,则跳转
11175:    CMP     DX,WORD PTR ES:[SI+1]       ;│
11176:    JB      GetPlusIntV_10              ;│
11177: GetPlusIntV_7:                         ;┘
11178:    CMP     CX,WORD PTR ES:[SI+7]       ;┐
11179:    JA      GetPlusIntV_10              ;│
11180:    JC      GetPlusIntV_11              ;├ 若参数值≤最大值,则转到 GetPlusIntV_11
11181:    CMP     DX,WORD PTR ES:[SI+5]       ;├ 若参数值>最大值,则转到 GetPlusIntV_10
11182:    JA      GetPlusIntV_10              ;│
11183:    JMP     SHORT GetPlusIntV_11        ;┘
11184: GetPlusIntV_8:
11185:    CMP     CX,WORD PTR ES:[SI+3]       ;┐
11186:    JL      GetPlusIntV_10              ;│
11187:    JG      GetPlusIntV_9               ;│
11188:    CMP     DX,WORD PTR ES:[SI+1]       ;│
11189:    JL      GetPlusIntV_10              ;├ 若参数符合最大值、最小值的要求(方法同上),
11190: GetPlusIntV_9:                         ;├ 则跳转
11191:    CMP     CX,WORD PTR ES:[SI+7]       ;│
11192:    JG      GetPlusIntV_10              ;│
```

```
11193:      JL      GetPlusIntV_11                  ;|
11194:      CMP     DX,WORD PTR ES:[SI+5]           ;|
11195:      JG      GetPlusIntV_10                  ;|
11196:      JMP     SHORT GetPlusIntV_11            ;|
11197: GetPlusIntV_10:                              ;┘
11198:      ADD     SI,9                            ;ES:SI 指向下一个"参数值范围"
11199:      DEC     AL                              ;修改未检查的"参数值范围"的个数
11200:      JNZ     GetPlusIntV_6                   ;若还有参数值范围未检查,则跳转
11201:      MOV     CS:GetParaRetCode,6             ;设置"取参数值的返回码"(6:表示参数值越界)
11202:      MOV     AL,1
11203:      MOV     AH,0FFH
11204:      JMP     SHORT GetPlusIntV_13
11205: GetPlusIntV_11:
11206:      MOV     AL,1
11207:      MOV     AH,ES:[SI]
11208:      JMP     SHORT GetPlusIntV_13
11209: GetPlusIntV_12:
11210:      MOV     CS:GetParaRetCode,9             ;设置"取参数值的返回码"(9:表示参数值错)
11211:      MOV     AL,3
11212:      MOV     AH,0FFH
11213: GetPlusIntV_13:
11214:      CALL    SetParaInfoBlk                  ;保存参数值
11215:      POP     SI
11216:      POP     DX
11217:      POP     CX
11218:      POP     AX
11219:      RET
11220: GetPlusIntegerValue  ENDP
11221: ;=============================================================
11222: ;相对地址偏移:1C21
11223: ;功能:判定参数值是否溢出
11224: ;入口参数:无
11225: ;出口参数:CF=0,未溢出;CF=1,溢出
11226: ;=============================================================
11227: TestParaOverflow  PROC  NEAR
11228:      PUSHF
11229:      TEST    CS:ParaFeatureFlag,2            ;┐若参数值为负,则跳转
11230:      JNZ     TestParaOverflow_1              ;┘
11231:      POPF
11232:      RET
11233: TestParaOverflow_1:
11234:      POPF
11235:      JO      TestParaOverflow_2
```

```
11236:      CLC
11237:      RET
11238: TestParaOverflow_2:
11239:      STC
11240:      RET
11241: TestParaOverflow  ENDP
11242: ;=============================================================
11243: ;相对地址偏移:1C33
11244: ;功能:计算 AL 指定的十进制数字符的数字值
11245: ;入口参数:AL=字符的 ASCII 码
11246: ;出口参数:AL=数字值
11247: ;=============================================================
11248: CalculateDecimalValue  PROC  NEAR
11249:      CMP     AL,' 0'
11250:      JB      CalcuDeciV_1
11251:      CMP     AL,' 9'
11252:      JA      CalcuDeciV_1
11253:      SUB     AL,' 0'
11254:      CLC
11255:      RET
11256: CalcuDeciV_1:
11257:      STC
11258:      RET
11259: CalculateDecimalValue  ENDP
11260: ;=============================================================
11261: ;相对地址偏移:1C41
11262: ;功能:取开关状态的参数值(ON/OFF)或关键字的参数值(如 HIGH、LOW、UMB、NOUMB)
11263: ;入口参数:CS:SI 指向参数项字符串
11264: ;          ES:BX 指向参数项信息块
11265: ;出口参数:无
11266: ;=============================================================
11267: GetSwitchStatusPara  PROC  NEAR
11268:      PUSH    AX
11269:      PUSH    BX
11270:      PUSH    DX
11271:      PUSH    DI
11272:      MOV     DI,ES:[BX+6]              ;ES:DI 指向关键字的参数值信息块
11273:      MOV     AL,ES:[DI]
11274:      OR      AL,AL
11275:      JNZ     GetSwitchStatusP_1
11276:      MOV     AH,0FFH
11277:      JMP     SHORT GetSwitchStatusP_5
11278: GetSwitchStatusP_1:
```

```
11279:      CMP     AL,3
11280:      JNE     GetSwitchStatusP_4
11281:      INC     DI                          ;┐
11282:      MOV     AL,ES:[DI]                  ;│
11283:      MOV     AH,9                        ;│
11284:      MUL     AH                          ;│
11285:      INC     AX                          ;│ 计算存放"开关状态关键字个数"的单元地址→
11286:      ADD     DI,AX                       ;│ ES:DI
11287:      MOV     AL,ES:[DI]                  ;│
11288:      MOV     AH,5                        ;│
11289:      MUL     AH                          ;│
11290:      INC     AX                          ;│
11291:      ADD     DI,AX                       ;┘
11292:      MOV     AL,ES:[DI]                  ;AL←命令允许的关键字个数
11293:      INC     DI                          ;┐ ES:DI←存放第一个关键字字符串指针的单元
11294:      INC     DI                          ;┘ 地址
11295: GetSwitchStatusP_2:
11296:      MOV     BP,ES:[DI]                  ;ES:BP 指向命令允许的关键字字符串
11297:      CALL    TestKeywordMatch            ;┐ 若指定的关键字合法,则跳转
11298:      JNC     GetSwitchStatusP_3          ;┘
11299:      ADD     DI,3                        ;ES:DI←存放下一个关键字字符串指针的单元
                                                ;地址
11300:      DEC     AL                          ;┐ 若还有关键字字符串未比较,则跳转
11301:      JNZ     GetSwitchStatusP_2          ;┘
11302:      MOV     CS:GetParaRetCode,8         ;设置"取参数值的返回码"(8:表示非法的开关
                                                ;状态关键字,即不是 ON 或 OFF)
11303:      MOV     AH,0FFH
11304:      JMP     SHORT GetSwitchStatusP_5
11305: GetSwitchStatusP_3:
11306:      MOV     AH,ES:[DI-1]                ;AH←关键字的序号
11307:      MOV     AL,2                        ;AL←参数值类型码(2:关键字)
11308:      MOV     DX,ES:[DI]                  ;DX←关键字字符串的起始地址
11309:      JMP     SHORT GetSwitchStatusP_6
11310: GetSwitchStatusP_4:
11311:      MOV     CS:GetParaRetCode,9         ;设置"取参数值的返回"(9:表示参数值错)
11312:      MOV     AH,0FFH
11313: GetSwitchStatusP_5:
11314:      MOV     AL,3                        ;AL←参数值类型码(3:参数值无效)
11315: GetSwitchStatusP_6:
11316:      CALL    SetParaInfoBlk              ;保存参数值(即开关状态值)
11317:      POP     DI
11318:      POP     DX
11319:      POP     BX
```

```
11320:      POP     AX
11321:      RET
11322: GetSwitchStatusPara  ENDP
11323: ;=============================================================
11324: ;相对地址偏移:1CAA
11325: ;功能:判定关键字是否匹配,或者判定开关参数说明串是否匹配
11326: ;入口参数:CS:SI 指向用户指定的参数项字符串
11327: ;          ES:BX 指向参数项信息块
11328: ;          ES:BP 指向命令允许的关键字字符串或开关参数说明串
11329: ;出口参数:CF=0,相同,CF=1,不相同
11330: ;=============================================================
11331: TestKeywordMatch  PROC  NEAR
11332:      PUSH    AX
11333:      PUSH    BP
11334:      PUSH    DX
11335:      PUSH    SI
11336:      MOV     DL,2                    ;DL←2(大写表的 ID 值,参见功能调用 65H 的
                                            ;入口参数 AL 值)
11337: TestKeywordM_1:
11338:      MOV     AL,CS:[SI]              ;AL←参数项的当前字符
11339:      CALL    TestDBCSBootByte        ;┐若 AL 指定的字符是双字节字符的引导字节,
11340:      JC      TestKeywordM_5          ;┘则跳转
11341:      CALL    ChangeCharToCap         ;将 AL 指定的字符转换成大写
11342:      TEST    CS:ParaFeatureFlag,08   ;┐若参数项不是"关键字…"格式,则跳转
11343:      JZ      TestKeywordM_2          ;┘
11344:      CMP     AL,'='                  ;┐若关键字或开关参数说明串未匹配完,则跳转
11345:      JNE     TestKeywordM_4          ;┘
11346:      CMP     BYTE PTR ES:[BP+1],0    ;┐若关键字或开关参数说明串不相同,则跳转
11347:      JNE     TestKeywordM_10         ;┘
11348:      JMP     SHORT TestKeywordM_3
11349: TestKeywordM_2:
11350:      TEST    CS:ParaFeatureFlag,10H  ;┐若参数项不是"关键字:…"格式,则跳转
11351:      JZ      TestKeywordM_4          ;┘
11352:      CMP     AL,':'                  ;┐若关键字或开关参数说明串未匹配完,则跳转
11353:      JNE     TestKeywordM_4          ;┘
11354:      CMP     BYTE PTR ES:[BP+0],0    ;┐若关键字或开关参数说明串不相同,则跳转
11355:      JNE     TestKeywordM_10         ;┘
11356: TestKeywordM_3:
11357:      INC     SI                      ;CS:SI 指向关键字或开关参数说明串的后续参数
                                            ;字符串
11358:      JMP     SHORT TestKeywordM_11
11359: TestKeywordM_4:
11360:      CMP     AL,ES:[BP+0]            ;┐若当前关键字符不同,则跳转
```

```
11361:      JNE     TestKeywordM_7                  ;┘
11362:      OR      AL,AL                           ;┐ 若参数项字符串结束,则跳转
11363:      JZ      TestKeywordM_11                 ;┘
11364:      INC     SI
11365:      INC     BP
11366:      JMP     SHORT TestKeywordM_6
11367: TestKeywordM_5:                              ;比较双字节字符
11368:      CMP     AL,ES:[BP+0]                    ;┐ 若双字节字符的引导字节不相同,表明不是
11369:      JNE     TestKeywordM_10                 ;┘ 当前关键字,则跳转
11370:      INC     SI                              ;┐
11371:      MOV     AL,CS:[SI]                      ;│ 若双字节字符的第二个字节不相同,表明不是
11372:      INC     BP                              ;│ 当前关键字,则跳转
11373:      CMP     AL,ES:[BP+0]                    ;│
11374:      JNE     TestKeywordM_10                 ;┘
11375:      INC     SI
11376:      INC     BP
11377: TestKeywordM_6:
11378:      JMP     SHORT TestKeywordM_1            ;转去比较后续的字符
11379: TestKeywordM_7:
11380:      TEST    CS:ParaFeatureFlag,40H          ;┐ 若参数项不是开关参数,则跳转
11381:      JZ      TestKeywordM_8                  ;┘
11382:      TEST    WORD PTR ES:[BX+2],20H          ;┐
11383:      JZ      TestKeywordM_8                  ;│ 若开关参数只要求关键字或开关参数说明串
11384:      CMP     BYTE PTR ES:[BP+0],0            ;│ 局部匹配,则跳转
11385:      JE      TestKeywordM_11                 ;┘
11386: TestKeywordM_8:
11387:      TEST    WORD PTR ES:[BX],10H            ;┐
11388:      JZ      TestKeywordM_10                 ;│
11389:      CMP     AL,' :'                         ;│
11390:      JNE     TestKeywordM_9                  ;│ 若关键字的结尾符":"不要求匹配,则转到
11391:      CMP     BYTE PTR ES:[BP+0],0            ;│ TestKeywordM_11,否则转到 TestKeywordM_10
11392:      JNE     TestKeywordM_10                 ;│
11393:      JMP     SHORT TestKeywordM_11           ;│
11394: TestKeywordM_9:                              ;┘
11395:      CMP     AL,0                            ;┐
11396:      JNE     TestKeywordM_10                 ;│ 若关键字或开关参数说明串匹配,则跳转
11397:      CMP     BYTE PTR ES:[BP+0],' :'         ;│
11398:      JE      TestKeywordM_11                 ;┘
11399: TestKeywordM_10:
11400:      STC
11401:      JMP     SHORT TestKeywordM_12
11402: TestKeywordM_11:
11403:      MOV     CS:StartAddrOfKeyword,SI        ;保存关键字或开关参数说明串的后续参数字符
```

```
                                                    ;串的起始地址
11404:      CLC
11405: TestKeywordM_12:
11406:      POP     SI
11407:      POP     DX
11408:      POP     BP
11409:      POP     AX
11410:      RET
11411: TestKeywordMatch  ENDP
11412: ;=============================================================
11413: ;相对地址偏移:1D4D
11414: ;功能:根据参数项字符串中是否含有系统保留字符来校正参数项字符串,同时,若是文件名对应的
           参数项,则还保存参数值
11415: ;入口参数:AX=参数项的控制字
11416: ;          CS:SI 指向参数项字符串
11417: ;          ES:BX 指向参数项信息块
11418: ;出口参数:无
11419: ;=============================================================
11420: VerifyParaStr  PROC  NEAR
11421:      PUSH    AX
11422:      PUSH    DI
11423:      PUSH    SI
11424:      MOV     DI,CS:StartAddrOfPara       ;DI←参数项字符串在命令字符串中的起始地址
11425:      MOV     AL,CS:[SI]                  ;┐
11426:      OR      AL,AL                       ;├ 若参数项字符串为空串,则跳转
11427:      JZ      VerifyParaStr_1             ;┘
11428:      CALL    TestReservedChar            ;┐ 若 AL 指定的字符不是系统保留字符,则跳转
11429:      JNZ     VerifyParaStr_2             ;┘
11430:      MOV     CS:ParaDefaultFlag,1        ;设置参数项为空字符串的标志
11431:      POP     SI                          ;┐ 置"参数项字符串"为空字符串
11432:      MOV     BYTE PTR CS:[SI],0          ;┘
11433:      POP     DI
11434:      JMP     SHORT VerifyParaStr_7
11435: VerifyParaStr_1:
11436:      POP     SI                          ;┐ 重新设置"参数项字符串"的结束符(此工作
11437:      MOV     BYTE PTR CS:[SI],0          ;┘ 无意义)
11438:      POP     DI
11439:      TEST    WORD PTR ES:[BX],1          ;┐ 若此参数项允许缺省,则跳转
11440:      JNZ     VerifyParaStr_7             ;┘
11441:      MOV     CS:GetParaRetCode,2         ;设置"取参数值的返回码"(2:表示"参数项
                                                ;不够数")
11442:      JMP     SHORT VerifyParaStr_7
11443: VerifyParaStr_2:
```

```
11444:    POP     AX                          ;┐(此工作无意义)
11445:    PUSH    SI                          ;┘
11446: VerifyParaStr_3:
11447:    MOV     AL,CS:[SI]                  ;AL←参数项字符串的当前字符
11448:    OR      AL,AL                       ;┐若参数项字符串已结束,则跳转
11449:    JZ      VerifyParaStr_6             ;┘
11450:    CALL    TestReservedChar            ;┐若 AL 指定的字符是系统保留字符,则跳转
11451:    JZ      VerifyParaStr_5             ;┘
11452:    CALL    TestDBCSBootByte            ;┐若 AL 指定的字符是单字节字符,则跳转
11453:    JNC     VerifyParaStr_4             ;┘
11454:    INC     DI                          ;┐跳过双字节字符的高字节
11455:    INC     SI                          ;┘
11456: VerifyParaStr_4:
11457:    INC     DI                          ;┐
11458:    INC     SI                          ;├转去检查下一个字符是否是系统保留字符
11459:    JMP     SHORT VerifyParaStr_3       ;┘
11460: VerifyParaStr_5:
11461:    MOV     CS:ParaSeparator,AL         ;保存当前使用的系统保留字符
11462:    MOV     BYTE PTR CS:[SI],0          ;重新设置"参数项字符串"的结束符,以便截掉从
                                              ;系统保留字符开始的所有后续字符
11463:    INC     DI                          ;┐修改参数项字符串在命令字符串中的结束地址
11464:    MOV     CS:EndAddrOfParaItem,DI     ;┘
11465: VerifyParaStr_6:
11466:    POP     SI
11467:    POP     DI
11468: VerifyParaStr_7:
11469:    POP     AX                          ;┐若参数值不是作为文件名的 ASCIIZ 字符串,则
11470:    TEST    AX,200H                     ;├直接返回
11471:    JZ      VerifyParaStr_RET           ;┘
11472:    PUSH    AX                          ;┐
11473:    MOV     AH,0FFH                     ;├设置"文件名 ASCIIZ 字符串"的参数值(此时
11474:    MOV     AL,5                        ;├ AL=5 表示参数是 ASCIIZ 字符串)
11475:    CALL    SetParaInfoBlk              ;├
11476:    POP     AX                          ;┘
11477: VerifyParaStr_RET:
11478:    RET
11479: VerifyParaStr  ENDP
11480: ;=============================================================
11481: ;相对地址偏移:1DBE
11482: ;功能:判断 AL 指定的字符是否是系统保留字符
11483: ;入口参数:AL=字符 ASCII 码
11484: ;出口参数:ZF=0,AL 指定的字符不是系统保留字符;ZF=1,AL 指定的字符是系统保留字符
11485: ;=============================================================
```

```
11486: TestReservedChar  PROC  NEAR
11487:     PUSH    BX
11488:     PUSH    CX
11489:     LEA     BX,CS:ReservedChars          ;CS:BX 指向系统保留字符表
11490:     MOV     CX,9                         ;CX←系统保留字符个数
11491: TestReservedChar_1:
11492:     CMP     AL,CS:[BX]                   ;┐若 AL 指定的字符是系统保留字符,则跳转
11493:     JE      TestReservedChar_2           ;┘
11494:     INC     BX                           ;┐继续与其它系统保留字符相比较
11495:     LOOP    TestReservedChar_1           ;┘
11496:     INC     CX                           ;设置返回标志值(ZF=0,AL 指定的字符不是系统
                                                ;保留字符)
11497: TestReservedChar_2:
11498:     POP     CX
11499:     POP     BX
11500:     RET
11501: TestReservedChar  ENDP
11502: ;=============================================================
11503: ;相对地址偏移:1DD3
11504: ;功能:取出"允许访问的最大逻辑驱动器数"的参数值
11505: ;入口参数:CS:SI 指向参数项字符串
11506: ;出口参数:无
11507: ;=============================================================
11508: GetMaxLogicDrivePara  PROC  NEAR
11509:     PUSH    AX
11510:     PUSH    DX
11511:     MOV     AL,CS:[SI]                   ;AL←参数项字符串的第一个字符
11512:     OR      AL,AL                        ;┐若参数项字符串为空,则结束返回
11513:     JZ      GetMaxLogicDP_3              ;┘
11514:     CALL    TestDBCSBootByte             ;┐若 AL 指定的字符是双字节字符的引导字节,则
11515:     JC      GetMaxLogicDP_2              ;┘跳转
11516:     CMP     WORD PTR CS:[SI+1],3AH       ;┐若符合逻辑驱动器名符格式(如 B:),则跳转
11517:     JE      GetMaxLogicDP_1              ;┘
11518:     TEST    WORD PTR ES:[BX],10H         ;┐
11519:     JZ      GetMaxLogicDP_2              ;├若逻辑驱动器符非法,则跳转
11520:     CMP     BYTE PTR CS:[SI+1],0         ;│
11521:     JNE     GetMaxLogicDP_2              ;┘
11522: GetMaxLogicDP_1:
11523:     OR      AL,'a'-'A'                   ;┐
11524:     CMP     AL,'a'                       ;│
11525:     JB      GetMaxLogicDP_2              ;├若逻辑驱动器符非法,则跳转
11526:     CMP     AL,'z'                       ;│
11527:     JA      GetMaxLogicDP_2              ;┘
```

```
11528:      SUB     AL,60H                  ;┐ DL←允许访问的最大逻辑驱动器数
11529:      MOV     DL,AL                   ;┘
11530:      MOV     AH,0FFH                 ;┐ 设置"允许访问的最大逻辑驱动器数"的参数值
11531:      MOV     AL,6                    ;│ (此时 AL=6 表示参数值是最大逻辑驱动器数)
11532:      CALL    SetParaInfoBlk          ;┘
11533:      JMP     SHORT GetMaxLogicDP_3
11534: GetMaxLogicDP_2:
11535:      MOV     CS:GetParaRetCode,9     ;设置"取参数值的返回码"(9:表示参数错)
11536: GetMaxLogicDP_3:
11537:      POP     DX
11538:      POP     AX
11539:      RET
11540: GetMaxLogicDrivePara  ENDP
11541: ;=============================================================
11542: ;相对地址偏移:1E17
11543: ;功能:跳过所有参数分隔符,并检查命令说明串是否已结束
11544: ;入口参数:DS:SI 指向待处理的命令说明串;
11545: ;          ES:DI 指向命令的参数信息块
11546: ;出口参数:DS:SI 指向新的待处理的命令说明串
11547: ;          CF=0,命令未结束;CF=1,命令已结束
11548: ;=============================================================
11549: OmitSeparators  PROC  NEAR
11550: OmitSeparators_1:
11551:      LODSB                           ;AL←命令说明串的当前待处理字符
11552:      CALL    ChkTerminator           ;┐ 若命令已结束,则转去置返回标志
11553:      JZ      OmitSeparators_2        ;┘
11554:      CALL    TestSeparator           ;┐ 若 AL 指定的字符是一个参数项的组成字符,则
11555:      JNZ     OmitSeparators_3        ;┘ 跳转
11556:      TEST    CS:ParaFeatureFlag,20H  ;┐ 若当前使用的参数分隔符不是",",则跳转(因
11557:      JZ      OmitSeparators_1        ;┘ 为","常用来表示一个参数项被缺省)
11558:      TEST    CS:ParaFeatureFlag,41H  ;┐ 若参数项的组成字符中没有"/"或"=",则结束
11559:      JZ      OmitSeparators_5        ;┘ 返回
11560:      DEC     SI                      ;恢复先前取出的一个字符(它是参数项的一个组
                                            ;成字符)
11561:      JMP     SHORT OmitSeparators_5
11562: OmitSeparators_2:
11563:      STC                             ;置返回标志(CF=1,命令已结束)
11564:      JMP     SHORT OmitSeparators_4
11565: OmitSeparators_3:
11566:      CLC                             ;置返回标志(CF=0,命令未结束)
11567: OmitSeparators_4:
11568:      DEC     SI                      ;恢复先前取出的命令说明串的字符(它是参数项
                                            ;的一个组成字符)
```

```
11569:      RET
11570: OmitSeparators_5:
11571:      CLC                                    ;置返回标志(CF=0,命令未结束)
11572:      RET
11573: OmitSeparators  ENDP
11574: ;=============================================================
11575: ;相对地址偏移:1E3D
11576: ;功能:检查命令说明串是否结束
11577: ;入口参数:AL=命令说明串的当前字符
11578: ; ES:DI 指向命令的参数信息块
11579: ;出口参数:ZF=0,命令未结束;ZF=1,命令结束
11580: ;=============================================================
11581: ChkTerminator  PROC  NEAR
11582:      PUSH    BX
11583:      PUSH    CX
11584:      CMP     AL,0DH                         ;┐若当前字符是回车符,则结束返回
11585:      JE      ChkTerminator_3                ;┘
11586:      CMP     AL,0                           ;┐若当前字符是命令字符串结束符,则结束返回
11587:      JE      ChkTerminator_3                ;┘
11588:      CMP     AL,0AH                         ;┐若当前字符是换行符,则结束返回
11589:      JE      ChkTerminator_3                ;┘
11590:      CMP     BYTE PTR ES:[DI+2],2           ;┐若没有指定命令结束符表,则结束返回
11591:      JB      ChkTerminator_3                ;┘
11592:      XOR     BX,BX                          ;┐BX←参数分隔符表的大小(字节数)
11593:      MOV     BL,ES:[DI+3]                   ;┘
11594:      ADD     BX,4                           ;ES:BX+DI 指向命令结束符表
11595:      CMP     BYTE PTR ES:[BX+DI],0          ;┐若命令结束符表为空表,则转去置返回标志
11596:      JE      ChkTerminator_2                ;┘
11597:      XOR     CX,CX                          ;┐CX←扩充的命令结束符个数
11598:      MOV     CL,ES:[BX+DI]                  ;┘
11599: ChkTerminator_1:                            ;┐
11600:      INC     BX                             ;│将 AL 指定的字符依次与命令结束符表中的结
11601:      CMP     AL,ES:[BX+DI]                  ;│束符相比较,以判定 AL 指定的字符是不是命令
11602:      JE      ChkTerminator_3                ;│结束符
11603:      LOOP    ChkTerminator_1                ;┘
11604: ChkTerminator_2:
11605:      CMP     AL,0DH                         ;设置返回标志(ZF=0,命令未结束)
11606: ChkTerminator_3:
11607:      POP     CX
11608:      POP     BX
11609:      RET
11610: ChkTerminator  ENDP
11611: ;=============================================================
```

```
11612: ;相对地址偏移:1E73
11613: ;功能:检查 AL 指定的字符是否是参数分隔符
11614: ;入口参数:AL=字符的 ASCII 码;
11615: ;          DS:SI 指向待处理的命令说明串
11616: ;出口参数:AL=字符的 ASCII 码;
11617: ;          DS:SI 指向待处理的命令说明串(SI 值可能被修改)
11618: ;          ZF=0,AL 指定的字符不是参数分隔数;ZF=1,AL 指定的字符是参数分隔符
11619: ;==========================================================
11620: TestSeparator  PROC  NEAR
11621:     PUSH    BX
11622:     PUSH    CX
11623:     MOV     CS:ParaSeparator,' '          ;补缺当前使用的参数分隔符
11624:     AND     CS:ParaFeatureFlag,0DFH       ;清参数分隔符","标志,表明未出现","
11625:     CMP     AL,' '                        ;┐
11626:     JE      TestSeparator_4               ;| 若 AL 指定的字符是通常使用的参数分隔符(空
11627:     CMP     AL,9                          ;| 格或 Tab),则结束返回
11628:     JE      TestSeparator_4               ;┘
11629:     CMP     AL,','                        ;┐ 若 AL 指定的字符是","(它表明有一个参数项
                                                 ;| 结束,通常它用来表示一个参数项被缺省),则
11630:     JE      TestSeparator_5               ;┘ 跳转
11631:     CMP     AL,' '                        ;┐
11632:     JNE     TestSeparator_1               ;|
11633:     CMP     BYTE PTR [SI],' '             ;|
11634:     JNE     TestSeparator_1               ;| 当两个空格符相邻时,跳过前面的空格符
11635:     MOV     AL,' '                        ;|
11636:     INC     SI                            ;|
11637:     CMP     AL,AL                         ;┘
11638:     JMP     SHORT TestSeparator_4
11639: TestSeparator_1:
11640:     CMP     BYTE PTR ES:[DI+2],1          ;┐ 若没有指定参数分隔符表,则结束返回
11641:     JB      TestSeparator_4               ;┘
11642:     XOR     CX,CX                         ;┐ CX←扩充的参数分隔符个数
11643:     MOV     CL,ES:[DI+3]                  ;┘
11644:     OR      CX,CX                         ;┐ 若参数分隔符表为空,则跳转
11645:     JZ      TestSeparator_3               ;┘
11646:     MOV     BX,3                          ;BX←参数分隔符表的的相对偏移(此时 ES:BX+
                                                 ;DI 指向参数分隔符表)
11647: TestSeparator_2:                          ;┐
11648:     INC     BX                            ;| 将 AL 指定的字符依次与参数分隔符表中的分
11649:     CMP     AL,ES:[BX+DI]                 ;| 隔符相比较,以便判定 AL 指定的字符是否是参
11650:     JE      TestSeparator_5               ;| 数分隔符
11651:     LOOP    TestSeparator_2               ;┘
11652: TestSeparator_3:
```

```
11653:    CMP     AL,' '                    ;设置返回标志(ZF=0,AL 指定的字符不是参数分
                                            ;隔符)
11654: TestSeparator_4:
11655:    POP     CX
11656:    POP     BX
11657:    RET
11658: TestSeparator_5:
11659:    MOV     CS:ParaSeparator,AL       ;保存当前使用的扩充的参数分隔符
11660:    TEST    CS:ParaFeatureFlag,01     ;┐若参数项中含有"="字符,则跳转
11661:    JNZ     TestSeparator_6           ;┘
11662:    OR      CS:ParaFeatureFlag,20H    ;置参数分隔符","标志,表明出现了","
11663: TestSeparator_6:
11664:    CMP     AL,AL                     ;设置返回标志值(ZF=1,AL 指定的字符是参数分
                                            ;隔符)
11665:    JMP     SHORT TestSeparator_4
11666: TestSeparator  ENDP
11667: ;=============================================================
11668: ;相对地址偏移:1ED4
11669: ;功能:检查 AL 指定的字符是否是开关参数前导符(或称为开关符),若是开关参数前导符,并且开
          关参数格式正确,则设置对应的标志位
11670: ;入口参数:AL=字符的 ASCII 码;
11671: ;          BX=保存参数项字符串的暂存缓冲区的写指针值
11672: ;出口参数:CF=0,AL 指定的字符不是开关参数前导符或开关参数格式正确;CF=1,开关参数格式
          错
11673: ;=============================================================
11674: TestSwitchPreface  PROC  NEAR
11675:    LEA     BP,CS:ParaStrBuf          ;┐
11676:    CMP     BX,BP                     ;├若 AL 指定的字符是参数项的第一个字符,则跳
11677:    JE      TestSwitchP_2             ;┘转
11678:    CMP     AL,' /'                   ;┐若 AL 指定的字符不是开关参数前导符,则跳转
11679:    JNE     TestSwitchP_1             ;┘
11680:    STC                               ;CF←1(开关参数格式错)
11681:    JMP     SHORT TestSwitchP_RET
11682: TestSwitchP_1:
11683:    CLC                               ;CF←0
11684:    JMP     SHORT TestSwitchP_RET
11685: TestSwitchP_2:
11686:    CMP     AL,' /'                   ;┐若 AL 指定的字符不是开关参数前导符,则跳转
11687:    JNE     TestSwitchP_3             ;┘
11688:    OR      CS:ParaFeatureFlag,40H    ;置"参数项为开关参数"的标志位
11689: TestSwitchP_3:
11690:    CLC
11691: TestSwitchP_RET:
```

```
11692:     RET
11693: TestSwitchPreface  ENDP
11694: ;=============================================================
11695: ;相对地址偏移:1EF2
11696: ;功能:判定 AL 指定的字符是否是双字节字符(DBCS)的引导字节
11697: ;入口参数:AL=字符的 ASCII 码
11698: ;出口参数:CF=0,AL 指定的字符不是双字节字符的引导字节;CF=1,AL 指定的字符是双字节字
                符的引导字节
11699: ;=============================================================
11700: TestDBCSBootByte  PROC  NEAR
11701:     PUSH    DS
11702:     PUSH    SI
11703:     PUSH    BX
11704:     CMP     WORD PTR CS:Ptr_DBCS+2,0     ;┐若已取出双字节字符集的引导字节表,则跳转
11705:     JNE     TestDBCSBootByte_1           ;┘
11706:     PUSH    AX                           ;┐
11707:     PUSH    DS                           ;|
11708:     PUSH    CX                           ;|
11709:     PUSH    DX                           ;|
11710:     PUSH    DI                           ;|
11711:     PUSH    BP                           ;|
11712:     PUSH    ES                           ;|
11713:     XOR     SI,SI                        ;|
11714:     MOV     DS,SI                        ;|
11715:     MOV     AX,6300H                     ;|取双字节字符集的引导字节表
11716:     INT     21H                          ;|
11717:     MOV     BX,DS                        ;|
11718:     OR      BX,BX                        ;|
11719:     POP     ES                           ;|
11720:     POP     BP                           ;|
11721:     POP     DI                           ;|
11722:     POP     DX                           ;|
11723:     POP     CX                           ;|
11724:     POP     DS                           ;|
11725:     POP     AX                           ;┘
11726:     JZ      TestDBCSBootByte_4           ;若 DOS 不支持该功能调用,则结束返回
11727:     MOV     WORD PTR CS:Ptr_DBCS,SI      ;┐保存双字节字符集的引导字节表的起始地址
11728:     MOV     WORD PTR CS:Ptr_DBCS+2,BX    ;┘
11729: TestDBCSBootByte_1:
11730:     MOV     SI,WORD PTR CS:Ptr_DBCS      ;┐ DS:SI 指向双字节字符集的引导字节表
11731:     MOV     DS,WORD PTR CS:Ptr_DBCS+2    ;┘
11732: TestDBCSBootByte_2:
11733:     CMP     WORD PTR [SI],0              ;┐若双字节字符集的引导字节表为空,则结束返
```

```
11734:      JE      TestDBCSBootByte_4          ;┘ 回
11735:      CMP     AL,[SI]                     ;┐
11736:      JB      TestDBCSBootByte_3          ;│ 若 AL 指定的字符不在当前引导字节上下限之
11737:      CMP     AL,[SI+1]                   ;│ 内,则跳转
11738:      JA      TestDBCSBootByte_3          ;┘
11739:      STC                                 ;CF←1(AL 指定的字符是双字节字符的引导字节)
11740:      JMP     SHORT TestDBCSBootByte_5
11741: TestDBCSBootByte_3:
11742:      INC     SI                          ;┐ DS:SI 指向下一个引导字节上下限
11743:      INC     SI                          ;┘
11744:      JMP     SHORT TestDBCSBootByte_2    ;转去继续检查 AL 指定的字符是否是双字节字符
                                                ;的引导字节
11745: TestDBCSBootByte_4:
11746:      CLC                                 ;CF←0(AL 指定的字符不是双字节字符的引导字
                                                ;节)
11747: TestDBCSBootByte_5:
11748:      POP     BX
11749:      POP     SI
11750:      POP     DS
11751:      RET
11752: TestDBCSBootByte  ENDP
11753: ;相对地址偏移:1F48
11754: ;1F48 Buffers=x [,y][/X]                 ;buffers 系统配置命令的参数信息块
11755: CPIB_B       DW      CCIB_B              ;命令参数的组成控制信息块 ┐
                                                ;的地址                   │
11756:              DB      1                   ;参数分隔符表和命令结束符 │ 命令参数的格
                                                ;表的组成标志(1:只有参数  │ 式控制信息块
                                                ;分隔符表)                │
11757:              DB      1                   ;参数分隔符个数           │
11758:              DB      ' ;'                ;参数分隔符表             ┘
11759: CCIB_B       DB      1                   ;命令允许缺省的参数项序号  ┐
11760:              DB      2                   ;命令允许的非开关参数个数  │
11761:              DW      PIB_X_B             ;参数 X 的参数项信息块的地址 │ 命令参数的组
11762:              DW      PIB_Y_B             ;参数 Y 的参数项信息块的地址 │ 成控制信息块
11763:              DB      1                   ;命令允许的开关参数个数    │
11764:              DW      PIB_SX_B            ;开关参数(/X)的参数项信息块 │
11765:              DB      0                   ;命令允许的含有"="字符的参 │
                                                ;数个数                   ┘
11766: PIB_X_B      DW      8000H               ;                         ┐
11767:              DB      0                   ;                         │
11768:              DB      0                   ;                         │ X 参数的参数项
11769:              DW      ParaValueTypeCode   ;参数值信息块的地址       │ 信息块
11770:              DW      PSIB_X_B            ;参数值范围信息块的地址   │
```

```
11771:                  DB      0                       ;                       ┘
11772: PSIB_X_B         DB      1                       ;                       ┐
11773:                  DB      1                       ;参数值范围个数        │
11774:                  DB      1                       ;                       │ X 参数的参数值
11775:                  DD      1                       ;X 参数的最小值        │ 范围信息块
11776:                  DD      63H                     ;X 参数的最大值        ┘
11777: PIB_Y_B          DW      8001H                   ;                       ┐
11778:                  DB      0                       ;                       │
11779:                  DB      0                       ;                       │ Y 参数的参数项
11780:                  DW      ParaValueTypeCode       ;参数值信息块的地址    │ 信息块
11781:                  DW      PSIB_Y_B                ;参数值范围信息块的地址 │
11782:                  DB      0                       ;                       ┘
11783: PSIB_Y_B         DB      1                       ;                       ┐
11784:                  DB      1                       ;参数值范围个数        │ Y 参数的参数值
11785:                  DB      1                       ;                       │ 范围信息块
11786:                  DD      0                       ;Y 参数的最小值        │
11787:                  DD      8                       ;Y 参数的最大值        ┘
11788: PIB_SX_B         DW      0                       ;                       ┐
11789:                  DB      0                       ;                       │
11790:                  DB      0                       ;                       │ X 开关参数的参
11791:                  DW      ParaValueTypeCode       ;参数值信息块的地址    │ 数项信息块
11792:                  DW      SIB_Pub                 ;开关参数的参数值范围信息 │
                                                        ;块的地址              │
11793:                  DB  1                           ;                       │
11794: SPS_SX_B                 DB      '/X',0          ;开关参数的说明字符串  ┘
11795: X_Buffers                DW      0               ;存放 X 参数值(即盘缓冲区个数)
11796: Y_Buffers                DW      0               ;存放 Y 参数值(即第二个高速缓存区的缓冲区个
                                                        ;数)
11797: SX_Buffers               DB      0               ;存放"/X"开关参数标志(MS—DOS 5.00 已废弃
                                                        ;了该开关参数)
11798: SIB_Pub                  DB      0               ;字符、文件名或开关参数的范围信息块
11799: ;相对地址偏移:1F91                               ;参数值信息块
11800: ParaValueTypeCode        DB      0               ;参数值类型码(1:整数值;2:开关状态,如 ON 或
                                                        ;OFF;3:没有参数;5:ASCIIZ 字符串,如文件名;
                                                        ;6:逻辑驱动器数,1=A,…,26=Z)
11801: KeywordNumber            DB      0               ;关键字的编号。对 dos 系统配置命令,0=HIGH,
                                                        ;1=LOW、2=UMB、3=NOUMB;对 break 系统配置
                                                        ;命令,0=OFF,1=ON 等
11802: StartAddrOfSwitchStr     DW      0               ;开关参数对应的字符串的起始地址
11803: ParaValue                DD      0               ;参数值或 ASCIIZ 字符串的起始地址
11804: ;1F99            Break=ON/OFF                    ;break 系统配置命令的参数信息块
11805: CPIB_C           DW      CCIB_C                  ;命令参数的组成控制信息块 ┐
                                                        ;的地址                   │
```

```
11806:                DB      1                ;参数分隔符表和命令结束符 ┐命令参数的格
                                               ;表的组成标志             ┘式控制信息块
11807:                DB      1                ;参数分隔符个数           ┐
11808:                DB      ' ;'             ;参数分隔符表             ┘
11809: CCIB_C         DB      1                ;命令允许缺省的参数项序号 ┐
11810:                DB      1                ;命令允许的非开关参数个数 │
11811:                DW      PIB_C            ;参数项信息块的地址       ┘命令参数的组成
11812:                DB      0                ;命令允许的开关参数个数   ┐控制信息块
11813:                DB      0                ;命令允许的含有"="字符的  │
                                               ;参数个数                 ┘
11814: PIB_C          DW      2000H            ;                         ┐
11815:                DB      0                ;                         │
11816:                DB      0                ;                         ┘参数项信息块
11817:                DW      ParaValueTypeCode;参数值信息块的地址       ┐
11818:                DW      PSIB_C           ;参数值范围信息块的地址   ┘
11819:                DB      0                ;                         ┘
11820: PSIB_C         DB      3                ;                         ┐
11821:                DB      0                ;                         │
11822:                DB      0                ;                         ┘
11823:                DB      2                ;关键字个数               ┐
11824:                DB      1                ;关键字序号               ┘参数值范围信息
11825:                DW      On_Keyword       ;关键字字符串的地址       ┐块
11826:                DB      2                ;关键字序号               ┘
11827:                DW      OFF_Keyword      ;关键字字符串的地址       ┐
11828: On_Keyword     DB      ' ON' ,0         ;┐关键字字符串             ┘
11829: OFF_Keyword    DB      ' OFF' ,0        ;┘                        ┘
11830: Status_Break   DB      0                ;存放 break 系统配置命令的开关状态(0:OFF;
                                               ;1:ON)
11831: ;1FBF Country=XXX,[YYY,]Filename        ;country 系统配置命令的参数信息块
11832: CPIB_Q         DW      CCIB_Q           ;命令参数的组成控制信息块 ┐
                                               ;的地址                   │
11833:                DB      1                ;参数分隔符表和命令结束符 ┘命令参数的格
                                               ;表的组成标志             ┐式控制信息块
11834:                DB      1                ;参数分隔符个数           ┘
11835:                DB      ' ;'             ;参数分隔符表             ┘
11836: CCIB_Q         DB      1                ;命令允许缺省的参数项序号 ┐
11837:                DB      3                ;命令允许的非开关参数个数 │
11838:                DW      PIB_X_Q          ;参数 XXX 的参数项信息块  │
                                               ;的地址                   │
11839:                DW      PIB_Y_Q          ;参数 YYY 的参数项信息块  │
                                               ;的地址                   ┘命令参数的组
                                                                         ┘成控制信息块
11840:                DW      PIB_F_Q          ;文件名参数的参数项信息   ┘
```

```
                                        ;块的地址                    |
11841:                  DB      0       ;命令允许的开关参数个数      |
11842:                  DB      0       ;命令允许的含有"="字符的    |
                                        ;参数个数                    ┘
11843: PIB_X_Q          DW      8000H   ;                            ┐
11844:                  DB      0       ;                            |
11845:                  DB      0       ;                            | XXX 参数的参数
11846:                  DW      ParaValueTypeCode ;参数值信息块的地址 | 项信息块
11847:                  DW      PSIB_XY_Q ;参数值范围信息块的地址    |
11848:                  DB      0       ;                            ┘
11849: PSIB_XY_Q        DB      1       ;                              ┐
11850:                  DB      1       ;参数值范围个数                | XXX 参数和
11851:                  DB      1       ;                              | YYY 参数的参
11852:                  DD      1       ;XXX 参数或 YYY 参数的最小值   | 数值范围信息
11853:                  DD      999     ;XXX 参数或 YYY 参数的最大值   ┘ 块
11854: PIB_Y_Q          DW      8001H   ;                            ┐
11855:                  DB      0       ;                            |
11856:                  DB      0       ;                            | YYY 参数的参数
11857:                  DW      ParaValueTypeCode ;参数值信息块的地址 | 项信息块
11858:                  DW      PSIB_XY_Q ;参数值范围信息块的地址    |
11859:                  DB      0       ;                            ┘
11860: PIB_F_Q          DW      0201H   ;                            ┐
11861:                  DB      0       ;                            |
11862:                  DB      0       ;                            | 文件名参数的
11863:                  DW      ParaValueTypeCode ;参数值信息块的地址 | 参数项信息块
11864:                  DW      SIB_Pub ;文件名参数的参数项范围      |
                                        ;信息块的地址                |
11865:                  DB      0       ;                            ┘
11866: CountryCode      DW      0       ;存放 XXX 参数值(即国家代码)
11867: CodePageNumber   DW      0       ;存放 YYY 参数值(即代码页号)
11868: ;1FF8            Files=n         ;files 系统配置命令的参数信息块
11869: CPIB_F           DW      CCIB_F  ;命令参数的组成控制信息块    ┐
                                        ;的地址                      |
11870:                  DB      1       ;参数分隔符表和命令结束符    | 命令参数的格
                                        ;表的组成标志                | 式控制信息块
11871:                  DB      1       ;参数分隔符个数              |
11872:                  DB      ' ;'    ;参数分隔符表                ┘
11873: CCIB_F           DB      1       ;命令允许缺省的参数项序号    ┐
11874:                  DB      1       ;命令允许的非开关参数个数    |
11875:                  DW      PIB_n_F ;n 参数的参数项信息块的地址  | 命令参数的组
11876:                  DB      0       ;命令允许的开关参数个数      | 成控制信息块
11877:                  DB      0       ;命令允许的含有"="字符的    |
                                        ;参数个数                    |
```

```
11878: PIB_n_F      DW    8000H                ;                          ┐
11879:              DB    0                    ;                          ┐
11880:              DB    0                    ;                          |
11881:              DW    ParaValueTypeCode    ;参数值信息块的地址        | n参数的参
11882:              DW    PSIB_n_F             ;参数值范围信息块的地址    | 数项信息块
11883:              DB    0                    ;                          ┘
11884: PSIB_n_F     DB    1                    ;                          ┐
11885:              DB    1                    ;参数值范围个数            |
11886:              DB    1                    ;                          | n参数的参数
11887:              DD    8                    ;n参数的最小值             | 值范围信息块
11888:              DD    255                  ;n参数的最大值             ┘
11889: NumberOfSFT  DB    0                    ;存放n参数值(同时存取的文件数)
11890: ;2018        FCBs=N [,S]                ;fcbs系统配置命令的参数信息块
11891: CPIB_X       DW    CCIB_X               ;命令参数的组成控制信息块  ┐
                                               ;的地址                    |
11892:              DB    1                    ;参数分隔符表和命令结束符  | 命令参数的格
                                               ;表的组成标志              | 式控制信息块
11893:              DB    1                    ;参数分隔符个数            |
11894:              DB    ' ;'                 ;参数分隔符表              ┘
11895: CCIB_X       DB    1                    ;命令允许缺省的参数项序号  ┐
11896:              DB    2                    ;命令允许的非开关参数个数  |
11897:              DW    PIB_N_X              ;N参数的参数项信息块的地址 | 命令参数的
11898:              DW    PIB_S_X              ;S参数的参数项信息块的地址 | 组成控制信
11899:              DB    0                    ;命令允许的开关参数个数    | 息块
11900:              DB    0                    ;命令允许的含有"="字符的   |
                                               ;参数个数                  ┘
11901: PIB_N_X      DW    8000H                ;                          ┐
11902:              DB    0                    ;                          |
11903:              DB    0                    ;                          | N参数的参数项
11904:              DW    ParaValueTypeCode    ;参数值信息块的地址        | 信息块
11905:              DW    PSIB_N_X             ;参数值范围信息块的地址    |
11906:              DB    0                    ;                          ┘
11907: PSIB_N_X     DB    1                    ;                          ┐
11908:              DB    1                    ;参数值范围个数            | N参数的参数
11909:              DB    1                    ;                          | 值范围信息块
11910:              DD    1                    ;N参数的最小值             |
11911:              DD    255                  ;N参数的最大值             ┘
11912: PIB_S_X      DW    8000H                ;                          ┐
11913:              DB    0                    ;                          |
11914:              DB    0                    ;                          | S参数的参
11915:              DW    ParaValueTypeCode    ;参数值信息块的地址        | 数项信息块
11916:              DW    PSIB_S_X             ;参数值范围信息块的地址    |
11917:              DB    0                    ;                          ┘
```

```
11918: PSIB_S_X     DB      1                ;                          ┐
11919:              DB      1                ;参数值范围个数           | S参数的参数
11920:              DB      1                ;                          | 值范围信息块
11921:              DD      0                ;S参数的最小值             |
11922:              DD      255              ;S参数的最大值             ┘
11923: N_FCBS       DB      0                ;存放N参数值(允许同时打开的FCB数)
11924: S_FCBS       DB      0                ;存放S参数值(MS-DOS 5.00已废弃该参数,它
                                             ;的值恒为0)
11925: ;204F        Lastdrive=S              ;lastdrive系统配置命令的参数信息块
11926: CPIB_L       DW      CCIB_L           ;命令参数的组成控制信息块 ┐
                                             ;的地址                   |
11927:              DB      1                ;参数分隔符表和命令结束符 | 命令参数的格
                                             ;表的组成标志             | 式控制信息块
11928:              DB      1                ;参数分隔符个数           |
11929:              DB      ' ;'             ;参数分隔符表             ┘
11930: CCIB_L       DB      1                ;命令允许缺省的参数项序号 ┐
11931:              DB      1                ;命令允许的非开关参数个数 |
11932:              DW      PIB_S_L          ;S参数的参数项信息块的地址| 命令参数的组
11933:              DB      0                ;命令允许的开关参数个数   | 成控制信息块
11934:              DB      0                ;命令允许的"含有="字符的  |
                                             ;参数个数                 ┘
11935: PIB_S_L      DW      0110H            ;                          ┐
11936:              DB      10H              ;                          |
11937:              DB      0                ;                          | S参数的参数
11938:              DW      ParaValueTypeCode;参数值信息块的地址        | 项信息块
11939:              DW      SIB_Pub          ;驱动器名符的参数项范围信  |
                                             ;息块的地址                |
11940:              DB      0                ;                          ┘
11941: LastDrive_L  DB      0                ;存放S参数值(1=A,…,26=Z)
11942: ;2064        Stacks=N,S               ;stacks系统配置命令的参数信息块
11943: CPIB_K       DW      CCIB_K           ;命令参数的组成控制信息块 ┐
                                             ;的地址                   |
11944:              DB      1                ;参数分隔符表和命令结束符 | 命令参数的格
                                             ;表的组成标志             | 式控制信息块
11945:              DB      1                ;参数分隔符个数           |
11946:              DB      ' ;'             ;参数分隔符表             ┘
11947: CCIB_K       DB      2                ;命令允许缺省的参数项序号 ┐
11948:              DB      2                ;命令允许的非开关参数个数 |
11949:              DW      PIB_N_K          ;N参数的参数项信息块的地址 | 命令参数的
11950:              DW      PIB_S_K          ;S参数的参数项信息块的地址 | 组成控制信
11951:              DB      0                ;命令允许的开关参数个数   | 息块
11952:              DB      0                ;命令允许的含有"="字符的  |
                                             ;参数个数                 ┘
```

```
11953: PIB_N_K      DW      8000H           ;                       ┐
11954:              DB      0               ;                       │
11955:              DB      0               ;                       │ N参数的参数
11956:              DW      ParaValueTypeCode ;参数值信息块的地址    │ 项信息块
11957:              DW      PSIB_N_K        ;参数值范围信息块的地址  ┘
11958:              DB      0               ;                       ┐
11959: PSIB_N_K     DB      1               ;参数值范围个数         │
11960:              DB      1               ;                       │ N参数的参数
11961:              DB      1               ;                       │ 值范围信息块
11962:              DD      0               ;N参数的最小值          │
11963:              DD      64              ;N参数的最大值          ┘
11964: PIB_S_K      DW      8000H           ;                       ┐
11965:              DB      0               ;                       │
11966:              DB      0               ;                       │ S参数的参数
11967:              DW      ParaValueTypeCode ;参数值信息块的地址    │ 项信息块
11968:              DW      PSIB_S_K        ;参数值范围信息块的地址  │
11969:              DB      0               ;                       ┘
11970: PSIB_S_K     DB      1               ;                       ┐
11971:              DB      1               ;参数值范围个数         │ S参数的参数
11972:              DB      1               ;                       │ 值范围信息块
11973:              DD      0               ;S参数的最小值          │
11974:              DD      512             ;S参数的最大值          ┘
11975: N_Stacks     DW      0               ;存放N参数值(动态堆栈个数)
11976: S_Stacks     DW      0               ;存放S参数值(每个堆栈的大小)
11977: ;209D        MultiTrack=ON/OFF       ;multitrack系统配置命令的参数信息块
11978: CPIB_M       DW      CCIB_M          ;命令参数的组成控制信息块 ┐
                                            ;的地址                 │
11979:              DB      1               ;参数分隔符表和命令结束符 │ 命令参数的格
                                            ;表的组成标志           │ 式控制信息块
11980:              DB      1               ;参数分隔符个数         │
11981:              DB      ' ;'            ;参数分隔符表           ┘
11982: CCIB_M       DB      1               ;命令允许缺省的参数项序号 ┐
11983:              DB      1               ;命令允许的非开关参数个数 │ 命令参数的组
11984:              DW      PIB_M           ;参数项信息块的地址      │ 成控制信息块
11985:              DB      0               ;命令允许的开关参数个数  │
11986:              DB      0               ;命令允许的含有"="字符的 │
                                            ;参数个数               ┘
11987: PIB_M        DW      2000H           ;                       ┐
11988:              DB      0               ;                       │
11989:              DB      0               ;                       │ 参数项信息块
11990:              DW      ParaValueTypeCode ;参数值信息块的地址    │
11991:              DW      PSIB_C          ;参数值范围信息块的地址  │
11992:              DB      0               ;                       ┘
```

```
11993: SwitchStatus_M DB      0                ;存放 multitrack 系统配置命令的开关状态(0:OFF;
                                               ;1:ON)
11994: ;20B2          Switches=S               ;switches 系统配置命令的参数信息块
11995: CPIB_1         DW      CCIB_1           ;命令参数的组成控制信息块 ┐
                                               ;的地址                   |
11996:                DB      1                ;参数分隔符表和命令结束符 | 命令参数的格
                                               ;表的组成标志             | 式控制信息块
11997:                DB      1                ;参数分隔符个数           |
11998:                DB      ' ;'             ;参数分隔符表             ┘
11999: CCIB_1         DB      0                ;命令允许缺省的参数项序号       ┐ 命令
12000:                DB      0                ;命令允许的非开关参数个数       | 参数
12001:                DB      3                ;命令允许的开关参数个数         | 的组
12002:                DW      PIB_SK_1         ;"/K"开关参数的参数项信息块的地址 | 成控
12003:                DW      PIB_ST_1         ;"/T"开关参数的参数项信息块的地址 | 制信
12004:                DW      PIB_SW_1         ;"/W"开关参数的参数项信息块的地址 | 息块
12005:                DB      0                ;命令允许的含有"="字符的参数个数 ┘
12006: PIB_SK_1       DW      0                ;                         ┐
12007:                DB      0                ;                         |
12008:                DB      0                ;                         |
12009:                DW      ParaValueTypeCode;参数值信息块的地址       | "/K"开关参数的
12010:                DW      SIB_Pub          ;开关参数的参数值范围信息 | 参数项信息块
                                               ;块的地址                 |
12011:                DB      1                ;                         |
12012: SPS_SK_1       DB      ' /K' , 0        ;"/K"开关参数的说明字符串 ┘
12013: PIB_ST_1       DW      0                ;                         ┐
12014:                DB      0                ;                         |
12015:                DB      0                ;                         | "/T"开关参数的
12016:                DW      ParaValueTypeCode;参数值信息块的地址       | 参数项信息块
12017:                DW      SIB_Pub          ;开关参数的参数值范围信息 |
                                               ;块的地址                 |
12018:                DB      1                ;                         |
12019: SPS_ST_1       DB      ' /T' , 0        ;"/T"开关参数的说明字符串 ┘
12020: PIB_SW_1       DW      0                ;                         ┐
12021:                DB      0                ;                         |
12022:                DB      0                ;                         | "/W"开关参数的
12023:                DW      ParaValueTypeCode;参数值信息块的地址       | 参数项信息块
12024:                DW      SIB_Pub          ;开关参数的参数值范围信息 |
                                               ;块的地址                 |
12025:                DB      1                ;                         |
12026: SPS_SW_1       DB      ' /W' , 0        ;"/W"开关参数的说明字符串 ┘
12027: SK_Switches    DB      0                ;存放"/K"开关参数的标志(1:指定了"/K"开关)
12028: ST_Switches    DB      0                ;存放"/T"开关参数的标志(1:指定了"/T"开关)
12029: SW_Switches    DB      0                ;存放"/W"开关参数的标志(1:指定了"/W"开关)
```

```
12030: ;20E8  DOS=High/Low UMB/NOUMB      ;dos 系统配置命令的参数信息块
12031: CPIB_H       DW      CCIB_H        ;命令参数的组成控制信息块 ┐
                                          ;的地址                   |
12032:              DB      1             ;参数分隔符表和命令结束符 | 命令参数的格
                                          ;表的组成标志             | 式控制信息块
12033:              DB      1             ;参数分隔符个数           |
12034:              DB      ' ;'          ;参数分隔符表             ┘
12035: CCIB_H       DB      1             ;命令允许缺省的参数项序号 ┐
12036:              DB      2             ;命令允许的非开关参数个数 |
12037:              DW      PIB_H         ;参数项信息块的地址       | 命令参数的组
12038:              DW      PIB_H         ;参数项信息块的地址       | 成控制信息块
12039:              DB      0             ;命令允许的开关参数个数   |
12040:              DB      0             ;命令允许的含有"="字符的  |
                                          ;参数个数                 ┘
12041: PIB_H        DW      2000H         ;                         ┐
12042:              DB      0             ;                         |
12043:              DB      0             ;                         | 参数项信息块
12044:              DW      ParaValueTypeCode ;参数值信息块的地址   |
12045:              DW      PSIB_H        ;参数值范围信息块的地址   |
12046:              DB      0             ;                         ┘
12047: ;相对地址偏移:20FE                 ;                         ┐
12048:              DW      2000H         ;                         |
12049:              DB      0             ;                         |
12050:              DB      0             ;                         | 未使用
12051:              DW      ParaValueTypeCode ;                     |
12052:              DW      PSIB_H        ;                         |
12053:              DB      0             ;                         ┘
12054: PSIB_H       DB      3             ;                         ┐
12055:              DB      0             ;                         |
12056:              DB      0             ;                         |
12057:              DB      4             ;关键字个数               |
12058:              DB      1             ;关键字序号               |
12059:              DW      HIGH_Keyword  ;关键字说明串的地址       | 参数值范
12060:              DB      2             ;关键字序号               | 围信息块
12061:              DW      LOW_Keyword   ;关键字说明串的地址       |
12062:              DB      3             ;关键字序号               |
12063:              DW      UMB_Keyword   ;关键字说明串的地址       |
12064:              DB      4             ;关键字序号               |
12065:              DW      NOUMB_Keyword ;关键字说明串的地址       ┘
12066: HIGH_Keyword     DB   ' HIGH' , 0  ;                         ┐
12067: LOW_Keyword      DB   ' LOW' , 0   ;                         | 关键字说明串
12068: UMB_Keyword      DB   ' UMB' , 0   ;                         |
12069: NOUMB_Keyword    DB   ' NOUMB' , 0 ;                         ┘
```

```
12070:      DB      0                                   ;未使用
12071:  ;相对地址偏移:212B
12072:  CurrDriverParagraphs  DW    0                   ;存放当前正在安装的设备驱动程序所需的内存节
                                                        ;数
12073:  LoadSeg_CurrDriver    DW    0                   ;存放当前正在安装的设备驱动程序的装入段地址
                                                        ;值
12074:  MaxSeg_CurrDriver     DW    0                   ;存放当前允许设备驱动程序使用的最大段地址值
12075:  DriverLoadAddr        DD    0                   ;在系统配置时,它存放当前正在安装的设备驱动
                                                        ;程序在内存中的起始地址
12076:  EndAddrOfDriver       DD    0                   ;存放设备驱动程序初始化后返回的结束地址
12077:  UMB_Flag              DB    0                   ;DOS 在常规内存和 UMB 之间保持链接的标志
                                                        (0:noumb;1:umb)
12078:  UMB_StartSeg          DW    0                   ;在 UMB 中安装设备驱动程序时,它存放 UMB 内
                                                        ;存块的起始段地址值
12079:  UMB_MB_Size           DW    0                   ;在 UMB 中安装设备驱动程序时,它存放 UMB 内
                                                        ;存块的大小(节数)
12080:  Seg_FreeUMB           DW    0                   ;在 UMB 中安装设备驱动程序时,它存放 UMB 内
                                                        ;存块的自由内存区的起始段地址值,也就是设备
                                                        ;驱动程序的装入段地址值
12081:  UMB_ManageProgramEntry DD     0                 ;保存 EMM386.EXE 提供的 UMB 管理程序的入
                                                        ;口地址
12082:  ;相对地址偏移:2144
12083:  OverlayPPB    LABEL   WORD                      ;以"覆盖方式"装入程序的参数块
12084:      DW      0                                   ;装入段地址
12085:      DW      0                                   ;重定位因子
12086:  DriverLoadFlag        DB    0                   ;设备驱动程序的安装标志(0:安装在常规内存中;
                                                        ;1:安装在 UMB 中)
12087:  Hexsize               DW    0                   ;存放"devicehigh size=hexsize…"命令的 hexsize 参
                                                        ;数值(即可用的最小内存量)
12088:  FlagOfMemory          DB    0                   ;常规内存大小的修改标志(0:未修改;-1:已修
                                                        ;改)
12089:  SizeOfCMA             DW    0                   ;保存实际的常规内存尺寸(节数),它的值为实际
                                                        ;值-1KB
12090:  DriverName            DB    'PROTMAN$'          ;设备驱动程序名
12091:  UMBChain_Flag         DB    0                   ;UMB 内存控制块链的建立标志(0:未建立;-1:已
                                                        ;建立)
12092:  Seg_MSDOS1            DW    0                   ;保存 MSDOS1 模块的段地址值,即多重表的段地
                                                        ;址值
12093:  Ptr_DriverNameStr     DD    0                   ;保存正在安装的设备驱动程序的文件名说明串
                                                        ;的起始地址
12094:  Terminator            DB    0                   ;存放参数分隔符或设备驱动程序的文件名说明串
                                                        ;的结束符
12095:  ;
```

```
12096: ;=============================================================
12097: ;相对地址偏移:215E
12098: ;功能:处理 CONFIG.SYS 系统配置文件
12099: ;入口参数:无
12100: ;出口参数:无
12101: ;=============================================================
12102: Process_Config  PROC  NEAR
12103:     PUSH    CS
12104:     POP     DS
12105:     MOV     AX,3700H                ;┐取出开关符(即开关参数前导符)
12106:     INT     21H                     ;┘
12107:     MOV     CMDLine+1,DL            ;保存开关符
12108:     MOV     DX, offset ConfigFile   ;┐
12109:     MOV     AX,3D00H                ;├以"读"方式打开 CONFIG.SYS 文件
12110:     STC                             ;│
12111:     INT     21H                     ;┘
12112:     JNC     Pro_Config1             ;若 CONFIG.SYS 存在,则跳转
12113:     MOV     ProcessConfigFlag,0BH   ;设置"CONFIG.SYS 文件不存在"的标志值
12114:     RET
12115: Pro_Config1:
12116:     MOV     BX,AX                   ;┐
12117:     XOR     CX,CX                   ;│
12118:     XOR     DX,DX                   ;├测试 CONFIG.SYS 的文件长度
12119:     MOV     AX,4202H                ;│
12120:     INT     21H                     ;┘
12121:     MOV     ConfigCount,AX          ;预置 CONFIG.SYS 文件的有效内容长度(字节数)
12122:     XOR     DX,DX                   ;┐
12123:     MOV     AX,4200H                ;├移动文件读写指针到 CONFIG.SYS 文件头
12124:     INT     21H                     ;┘
12125:     MOV     DX,SEG_Config           ;                          ┐
12126:     MOV     AX,ConfigCount          ;┐保存 CONFIG.SYS 的文      │
12127:     MOV     LengthOfConfig,AX       ;┘件长度                   ├确定 CONFIG.SYS
12128:     CALL    Convert_Addr            ;                          │文件内容在常规内
12129:     SUB     DX,AX                   ;                          │存中的段地址
12130:     SUB     DX,11H                  ;                          │
12131:     MOV     SEG_Config,DX           ;                          ┘
12132:     CALL    Create_CDS              ;为系统中每个逻辑驱动器建立一个 CDS
12133:     MOV     DX,CS:SEG_Config        ;┐
12134:     MOV     DS,DX                   ;│
12135:     MOV     ES,DX                   ;│
12136:     XOR     DX,DX                   ;├读取整个 CONFIG.SYS 的文件内容到指定的内
12137:     MOV     CX,CS:ConfigCount       ;│存中
12138:     MOV     AH,3FH                  ;│
```

```
12139:    STC                              ;|
12140:    INT     21H                      ;┘
12141:    PUSHF
12142:    PUSH    AX
12143:    PUSH    DI
12144:    PUSH    CX
12145:    MOV     AL,1AH                   ;┐
12146:    MOV     DI,DX                    ;| 通过搜索 CONFIG.SYS 的文件结束符(1AH)来
12147:    JCXZ    Pro_Config2              ;| 确定 CONFIG.SYS 文件的有效内容大小(字节
12148:    REPNE   SCASB                    ;| 数)
12149:    JNZ     Pro_Config2              ;|
12150:    DEC     DI                       ;┘
12151: Pro_Config2:
12152:    MOV     AL,CR                    ;┐ 填回车符
12153:    STOSB                            ;┘
12154:    MOV     AL,LF                    ;┐ 填换行符
12155:    STOSB                            ;┘
12156:    SUB     DI,DX                    ;┐ 保存 CONFIG.SYS 的实际有效内容的大小(字
12157:    MOV     CS:ConfigCount,DI        ;┘ 节数)
12158:    POP     CX
12159:    POP     DI
12160:    POP     AX
12161:    PUSH    CS
12162:    POP     DS
12163:    PUSH    AX                       ;┐
12164:    MOV     AH,3EH                   ;| 关闭 CONFIG.SYS 文件
12165:    INT     21H                      ;|
12166:    POP     AX                       ;┘
12167:    POPF
12168:    JC      Pro_Config3              ;若读 CONFIG.SYS 文件时出错,则跳转
12169:    CMP     CX,AX                    ;┐ 若已全部读入 CONFIG.SYS 文件内容,则跳转
12170:    JE      Pro_Config5              ;┘
12171: Pro_Config3:
12172:    MOV     DX, offset ConfigFile    ;┐ 给出读 CONFIG.SYS 文件时出错的错误信息,
12173:    CALL    Disp_ErrorInfo           ;┘ 并结束 CONFIG.SYS 处理
12174: Pro_Config_RET:
12175:    RET
12176: ;=============================================================
12177: ;相对地址偏移:21F2
12178: ;功能:识别(处理)CONFIG.SYS 文件中的系统配置命令
12179: ;入口参数:无
12180: ;出口参数:无
12181: ;=============================================================
```

```
12182: Pro_Config4   LABEL   NEAR
12183:      PUSH    CS
12184:      POP     DS
12185:      CMP     ProcessConfigFlag,0AH       ;┐ 若不存在 CONFIG.SYS 文件,则直接返回
12186:      JAE     Pro_Config_RET              ;┘
12187:      PUSH    SEG_Config                  ;┐ ES←存放 CONFIG.SYS 文件内容的缓冲区的
12188:      POP     ES                          ;┘ 段地址值
12189:      MOV     SI,BytesOfConfig            ;┐ 设置 CONFIG.SYS 待处理的字符数
12190:      MOV     ConfigCount,SI              ;┘
12191:      XOR     SI,SI                       ;┐ 设置 CONFIG.SYS 待处理内容的读指针值
12192:      MOV     Ptr_Config,SI               ;┘
12193:      MOV     LineCounter,SI              ;清行号计数器
12194:      CALL    Get_CharOfConfig            ;取出一个待处理的字符→AL
12195:      JMP     SHORT Pro_Config6
12196: Pro_Config5:
12197:      CALL    ConvertConfig               ;转换 CONFIG.SYS 文件内容(即系统配置命令用
                                                ;对应的缩写关键字符代替等工作)
12198:      CALL    Get_CharOfConfig            ;取出一个待处理的字符→AL
12199: Pro_Config6:                             ;开始识别一个系统配置命令
12200:      JC      Pro_Config_RET              ;若 CONFIG.SYS 文件已处理完,则直接返回
12201:      INC     LineCounter                 ;累计行号(它等于当前被识别的命令行在
                                                ;CONFIG.SYS 文本文件中的行号)
12202:      MOV     DriverCounter,0             ;清设备驱动程序计数器
12203:      MOV     FlagOfSCB,0                 ;清"设置子段控制块大小"的标志值
12204:      CMP     AL,0AH                      ;┐ 若当前系统配置命令行已结束,则跳转
12205:      JE      Pro_Config8                 ;┘
12206:      MOV     AH,AL                       ;AH←当前待处理的字符
12207:      CALL    Get_CharOfConfig            ;取出一个后续待处理的字符→AL
12208:      JNC     Pro_Config9                 ;若 CONFIG.SYS 文件未处理完,则跳转
12209:      CMP     ProcessConfigFlag,2         ;┐ 若是处理 install 系统配置命令,则直接返回
12210:      JAE     Pro_Config_RET              ;┘
12211:      JMP     Pro_Config155
12212: Pro_Config7:                             ;为识别下一个命令行作准备工作
12213:      PUSH    CS                          ;┐ 跳过当前系统配置命令行的所有未处理的字
12214:      POP     DS                          ;┃ 符,并取出下一个命令行的第一个字符
12215:      CALL    Omit_A_ConfigCMD            ;┘
12216:      JMP     SHORT Pro_Config6
12217: Pro_Config8:
12218:      CALL    Get_CharOfConfig            ;取出一个待处理的字符→AL
12219:      JMP     SHORT Pro_Config6
12220:      PUSH    CS
12221:      POP     DS
12222: Pro_Config9:
```

```
12223:    CMP     ProcessConfigFlag,0          ;┐
12224:    JNE     Pro_Config10                 ;│若本次只识别 dos 系统配置命令,以决定 DOS
12225:    JMP     Pro_Config40                 ;│决定本身的一部分程序(IO2 模块和 MSDOS2 模
12226: Pro_Config10:                           ;┘块)是否装在 HMA,则跳转
12227:    CMP     ProcessConfigFlag,2          ;┐若本次不识别任何系统配置命令,则跳转
12228:    JE      Pro_Config14                 ;┘
12229:    CMP     ProcessConfigFlag,3          ;┐若本次只识别 install 系统配置命令,则跳转
12230:    JE      Pro_Config11                 ;┘
12231:    CMP     AH,'H'                       ;┐若是 dos 系统配置命令,则跳转
12232:    JE      Pro_Config14                 ;┘
12233:    CMP     AH,'I'                       ;┐若不是 install 系统配置  ┐
12234:    JNE     Pro_Config15                 ;┘命令,则跳转             │检查 CONFIG.
12235:    OR      WORD PTR Install_Flag,1      ;置 CONFIG.SYS 文件中有   │SYS 文件中是否
                                               ;install 系统配置命令的标志│有 install 系统配
12236:    JMP     SHORT Pro_Config7            ;转去为识别下一个命令行   │置命令
                                               ;作准备                   ┘
12237: Pro_Config11:
12238:    CMP     AH,'I'                       ;┐若不是 install 系统配置命令,则跳转
12239:    JNE     Pro_Config12                 ;┘
12240:    CALL    Process_Install              ;处理 install 系统配置命令,安装 DOS 的内存驻留
                                               ;程序
12241:    JMP     SHORT Pro_Config7            ;转去为识别下一个命令行作准备
12242: Pro_Config12:
12243:    CMP     AH,'Y'                       ;┐若是 comment 系统配置命令,则跳转
12244:    JE      Pro_Config13                 ;┘
12245:    CMP     AH,'Z'                       ;┐若是非法的系统配置命令,则跳转
12246:    JE      Pro_Config13                 ;┘
12247:    CMP     AH,'0'                       ;┐若不是 rem 系统配置命令,则跳转
12248:    JNE     Pro_Config14                 ;┘
12249: Pro_Config13:
12250:    DEC     Ptr_Config                   ;┐恢复先前取出的一个字符
12251:    INC     ConfigCount                  ;┘
12252: Pro_Config14:
12253:    JMP     SHORT Pro_Config7            ;转去为识别下一个命令行作准备
12254: Pro_Config15:
12255:    CMP     AH,'B'                       ;┐若不是 buffers 系统配置命令,则跳转
12256:    JNE     Pro_Config24                 ;┘
12257: ;处理 CONFIG.SYS 的 buffers 系统配置命令
12258:    MOV     SX_Buffers,0                 ;清"/X"开关标志(0:表示没有"/X"开关)
12259:    MOV     DI,offset CP1B_B             ;DI 指向 buffers 命令的参数信息块
12260:    XOR     CX,CX                        ;清已取出的参数项数
12261:    MOV     DX,CX
12262: Pro_Config16:
```

```
12263:    CALL    GetCurrParaValue                ;取出当前参数项的参数值
12264:    JNC     Pro_Config17                    ;若参数项合法,则跳转
12265:    CALL    OutputErrInfo                   ;输出 buffers 系统配置命令格式错的错误信息
12266:    JMP     SHORT Pro_Config23
12267: Pro_Config17:
12268:    CMP     AX,-1                           ;┐若 buffers 系统配置命令正常结束,则跳转
12269:    JE      Pro_Config21                    ;┘
12270:    CMP     StartAddrOfSwitchStr,offset SPS_SX_B  ;┐若当前参数项不是"/X"开关,则跳转
12271:    JNE     Pro_Config18                    ;┘
12272:    JMP     SHORT Pro_Config20
12273: Pro_Config18:
12274:    MOV     AX,WORD PTR ParaValue           ;AX←参数值
12275:    CMP     CX,1                            ;┐若当前参数值是指定第二个高速缓存区的缓冲
12276:    JNE     Pro_Config19                    ;┘区数,则跳转
12277:    MOV     X_Buffers,AX                    ;保存盘缓冲区数(即 buffers=n,m 的 n 参数值)
12278:    JMP     SHORT Pro_Config20
12279: Pro_Config19:
12280:    MOV     Y_Buffers,AX                    ;保存第二个高速缓存区的缓冲区数(即 buffers=
                                                  ;n,m 的 m 参数值)
12281: Pro_Config20:
12282:    JMP     SHORT Pro_Config16              ;转去继续取后续参数项
12283: Pro_Config21:
12284:    CMP     X_Buffers,99                    ;┐若 n 参数值未超过最大值,则跳转
12285:    JBE     Pro_Config22                    ;┘
12286:    CALL    OutputErrInfo                   ;输出 buffers 系统配置命令格式错的错误信息
12287:    MOV     Y_Buffers,0                     ;清第二个高速缓存区的缓冲区数
12288:    JMP     SHORT Pro_Config23
12289: Pro_Config22:
12290:    MOV     AX,X_Buffers                    ;┐设置盘缓冲区数
12291:    MOV     NumberOfBuffer,AX               ;┘
12292:    MOV     AX,Y_Buffers                    ;┐设置第二个高速缓存区的缓冲区数
12293:    MOV     NumberOf2ndBuffer,AX            ;┘
12294:    MOV     AX,LineCounter                  ;┐保存最后一个 buffers 系统配置命令在
12295:    MOV     LineNumOfLastBuffer,AX          ;┘CONFIG.SYS 文本文件中的行号
12296: Pro_Config23:
12297:    JMP     Pro_Config7                     ;转去为识别下一个命令行作准备
12298: Pro_Config24:
12299:    CMP     AH,'C'                          ;┐若不是 break 系统配置命令,则跳转
12300:    JNE     Pro_Config31                    ;┘
12301: ;处理 CONFIG.SYS 的 break 系统配置命令
12302:    MOV     DI,offset CPIB_C                ;DI 指向 break 系统配置命令的参数信息块
12303:    XOR     CX,CX                           ;清已取出的参数项数
12304:    MOV     DX,CX
```

```
12305: Pro_Config25:
12306:     CALL    GetCurrParaValue          ;取出当前参数项的参数值
12307:     JNC     Pro_Config26              ;若参数项合法,则跳转
12308:     CALL    OutputErrInfo             ;输出 break 系统配置命令格式错的错误信息
12309:     JMP     SHORT Pro_Config30
12310: Pro_Config26:
12311:     CMP     AX,-1                     ;┐若 break 系统配置命令正常结束,则转去设置
12312:     JE      Pro_Config29              ;┘ Ctrl+Break 检查状态
12313:     CMP     KeywordNumber,1           ;┐若 Ctrl+Break 检查状态为"OFF",则跳转
12314:     JNE     Pro_Config27              ;┘
12315:     MOV     Status_Break,1            ;保存 Ctrl+Break 检查状态为"ON"的参数值
12316:     JMP     SHORT Pro_Config28
12317: Pro_Config27:
12318:     MOV     Status_Break,0            ;保存 Ctrl+Break 检查状态为"OFF"的参数值
12319: Pro_Config28:
12320:     JMP     SHORT Pro_Config25        ;转去继续取后续参数项,以便检查 break 系统配
                                             ;置命令格式的正确性
12321: Pro_Config29:
12322:     MOV     AH,33H                    ;┐
12323:     MOV     AL,1                      ;├设置 Ctrl+Break 检查状态
12324:     MOV     DL,Status_Break           ;│
12325:     INT     21H                       ;┘
12326: Pro_Config30:
12327:     JMP     Pro_Config7               ;转去为识别下一个命令行作准备
12328: Pro_Config31:
12329:     CMP     AH,'M'                    ;┐若不是 multitrack 系统配置命令,则跳转
12330:     JNE     Pro_Config45              ;┘
12331: ;处理 CONFIG.SYS 的 multitrack 系统配置命令
12332:     MOV     DI,offset CPIB_M          ;DI 指向 multitrack 系统配置命令的参数信息块
12333:     XOR     CX,CX                     ;清已取出的参数项数
12334:     MOV     DX,CX
12335: Pro_Config32:
12336:     CALL    GetCurrParaValue          ;取出当前参数项的参数值
12337:     JNC     Pro_Config33              ;若参数项合法,则跳转
12338:     CALL    OutputErrInfo             ;输出 multitrack 系统配置命令格式错的错误信息
12339:     JMP     SHORT Pro_Config39
12340: Pro_Config33:
12341:     CMP     AX,-1                     ;┐若 multitrack 系统配置命令正常结束,则转去置
12342:     JE      Pro_Config36              ;┘ IO1 模块的变量值
12343:     CMP     KeywordNumber,1           ;┐若 multitrack 检查状态为"OFF",则跳转
12344:     JNE     Pro_Config34              ;┘
12345:     MOV     SwitchStatus_M,1          ;保存 multitrack 检查状态为"ON"的参数值
12346:     JMP     SHORT Pro_Config35        ;
```

```
12347: Pro_Config34:
12348:     MOV     SwitchStatus_M,0            ;保存 multitrack 检查状态为"OFF"的参数值
12349: Pro_Config35:
12350:     JMP     SHORT Pro_Config32          ;转去继续取后续参数项,以便检查 multitrack 系统
                                               ;配置命令格式的正确性
12351: Pro_Config36:
12352:     PUSH    DS                          ;┐
12353:     MOV     AX,BIO_SEG                  ;|
12354:     MOV     DS,AX                       ;|
12355:     CMP     CS:SwitchStatus_M,0         ;|
12356:     JNE     Pro_Config37                ;|
12357:     MOV     MultiTrack,1                ;| 设置 IO1 模块变量值(1:OFF;80:ON)
12358:     JMP     SHORT Pro_Config38          ;|
12359: Pro_Config37:                           ;|
12360:     MOV     MultiTrack,80H              ;|
12361: Pro_Config38:                           ;|
12362:     POP     DS                          ;┘
12363: Pro_Config39:
12364:     JMP     Pro_Config7                 ;转去为识别下一个命令行作准备
12365: Pro_Config40:
12366:     CMP     AH,' H'                     ;┐
12367:     JE      Pro_Config41                ;| 若不是 dos 系统配置命令,则跳转
12368:     JMP     Pro_Config12                ;┘
12369: Pro_Config41:                           ;处理 CONFIG.SYS 的 dos 系统配置命令
12370:     MOV     DI,offset CPIB_H            ;DI 指向 dos 系统配置命令的参数信息块
12371:     XOR     CX,CX                       ;清已取出的参数项数
12372:     MOV     DX,CX
12373: Pro_Config42:
12374:     CALL    GetCurrParaValue            ;取当前参数项的参数值
12375:     JNC     Pro_Config43                ;若参数项合法,则跳转
12376:     CALL    OutputErrInfo               ;输出 dos 系统配置命令格式错的错误信息
12377:     JMP     SHORT Pro_Config44          ;
12378: Pro_Config43:
12379:     CMP     AX,-1                       ;┐ 若 dos 系统配置命令正常结束,则跳转
12380:     JE      Pro_Config44                ;┘
12381:     CALL    Save_dos_Para               ;保存当前参数项的参数值
12382:     JMP     SHORT Pro_Config42          ;转去继续取后续参数项
12383: Pro_Config44:
12384:     JMP     Pro_Config7                 ;转去为识别下一个命令行作准备
12385: Pro_Config45:
12386:     CMP     AH,' U'                     ;┐ 若不是 devicehigh 或 devicehigh size 系统配置命
12387:     JNE     Pro_Config49                ;┘ 令,则跳转
12388: ;处理 CONFIG.SYS 的 devicehigh 或 devicehigh size 系统配置命令
```

```
12389:    MOV    WORD PTR CS:Ptr_ProcessConfig,SI    ;┐ 保存装入 UMB 的设备驱动程序的文件名
12390:    MOV    WORD PTR CS:Ptr_ProcessConfig+2,ES;┘ 说明串的起始地址
12391:    CALL   ProcessHexsizePara                  ;处理 devicehigh size 系统配置命令的 hexsize 参数
12392:    JNC    Pro_Config46                        ;若 hexsize 参数格式正确,则跳转
12393:    CALL   OutputErrInfo                       ;输出 devicehigh 系统配置命令格式错的错误信息
12394:    JMP    Pro_Config7                         ;转去为识别下一个命令行作准备
12395: Pro_Config46:                                 ;                              ┐
12396:    PUSH   SI                                  ;┐ ES:SI 指向设备驱动程序      |
12397:    PUSH   ES                                  ;┘ 的文件名说明串              |
12398: Pro_Config47:                                 ;┐                             |
12399:    MOV    AL,ES:[SI]                          ;|                             |
12400:    CMP    AL,0DH                              ;|                             | 更换设备驱动
12401:    JE     Pro_Config48                        ;|                             | 程序的文件名
12402:    CMP    AL,0AH                              ;|                             | 说明串的结束
12403:    JE     Pro_Config48                        ;|                             | 符,以便通过
12404:    CALL   TestTerminatorOrSeparator           ;| 保存设备驱动程序的文件     | 调用装入设备
12405:    JZ     Pro_Config48                        ;| 名说明串的结束符           | 请求 DOS 功能
12406:    INC    SI                                  ;|                             | 驱动程序
12407:    JMP    SHORT Pro_Config47                  ;|                             |
12408: Pro_Config48:                                 ;|                             |
12409:    MOV    CS:Terminator,AL                    ;┘                             |
12410:    MOV    BYTE PTR ES:[SI],0                  ;更换文件说明串的结束符        |
12411:    POP    ES                                  ;                              |
12412:    POP    SI                                  ;                              ┘
12413:    MOV    CS:DriverLoadFlag,0                 ;清设备驱动程序的安装标志(0:设备驱动程序
                                                     ;安装在常规内存中)
12414:    CMP    CS:UMB_Flag,0                       ;┐ 若 MS—DOS 没有提供 UMB 的链接(noumb),则
12415:    JE     Pro_Config51                        ;┘ 跳转
12416:    MOV    CS:DriverLoadFlag,1                 ;设置设备驱动程序的安装标志(1:设备驱动程序
                                                     ;安装在 UMB 中)
12417:    JMP    SHORT Pro_Config51
12418: Pro_Config49:
12419:    CMP    AH,'D'                              ;┐
12420:    JE     Pro_Config50                        ;| 若不是 device 系统配置命令,则跳转
12421:    JMP    Pro_Config77                        ;┘
12422: Pro_Config50:                                 ;处理 CONFIG.SYS 的 device 系统配置命令
12423:    MOV    CS:DriverLoadFlag,0                 ;清设备驱动程序的安装标志(0:设备驱动程序安
                                                     ;装在常规内存中)
12424:    MOV    CS:Hexsize,0                        ;清 hexsize(可用的最小内存量)参数值
12425:    MOV    CS:Terminator,' '                   ;设置参数分隔符
12426: Pro_Config51:
12427:    MOV    BX,CS
12428:    MOV    DS,BX
```

```
12429:    MOV     WORD PTR CS:RequestForInit+12H,SI ;┐ 设备驱动程序的文件名说明串的起始地址
12430:    MOV     WORD PTR CS:RequestForInit+14H,ES ;┘ →初始化请求标题
12431:    MOV     WORD PTR CS:Ptr_DriverNameStr,SI    ;┐ 保存正在安装的设备驱动程序的文件名
12432:    MOV     WORD PTR CS:Ptr_DriverNameStr+2,ES ;┘ 说明串的起始地址
12433:    CALL    Convert_AddrFMT                   ;转换自由内存的起始地址值格式,并检查是否有
                                                    ;空闲内存
12434:    CALL    DetermineDriverParagraphs         ;确定正在安装的设备驱动程序所需的内存节数
12435:    JC      Pro_Config53                      ;若指定的设备驱动程序有错,则跳转
12436:    CALL    CreateDriverSCB1                  ;建立设备驱动程序的子段控制块,确定设备驱动
                                                    ;程序的装入地址
12437:    MOV     AX,CS:LoadSeg_CurrDriver          ;┐
12438:    ADD     AX,CS:CurrDriverParagraphs        ;├ 若有足够的内存空间装入设备驱动程序,则跳
12439:    JC      Pro_Config52                      ;├ 转
12440:    CMP     CS:MaxSeg_CurrDriver,AX           ;├
12441:    JAE     Pro_Config55                      ;┘
12442: Pro_Config52:
12443:    JMP     MemoryError                       ;转去显示"Configuration too large for memory"
12444: Pro_Config53:
12445:    CMP     BYTE PTR ES:[SI],0DH              ;┐ 若没有指定设备驱动程序的文件名,则跳转
12446:    JNE     Pro_Config54                      ;┘
12447:    JMP     Pro_Config155                     ;转去显示错误信息"Unrecognized command in
                                                    ;CONFIG.SYS"
12448: Pro_Config54:
12449:    CALL    Disp_ErrorStr                     ;输出"Bad or Missing"及设备驱动程序的文件名说
                                                    ;明串等信息
12450:    JMP     Pro_Config7                       ;转去为识别下一个命令行作准备
12451: Pro_Config55:
12452:    PUSH    ES                                ;┐
12453:    POP     DS                                ;├
12454:    MOV     DX,SI                             ;├
12455:    CALL    LoadDriver                        ;├ 装入设备驱动程序
12456:    PUSH    DS                                ;├
12457:    POP     ES                                ;├
12458:    PUSH    CS                                ;├
12459:    POP     DS                                ;┘
12460:    JC      Pro_Config53                      ;若装入失败,则跳转
12461: Pro_Config56:
12462:    PUSH    ES                                ;┐ ES:SI 指向设备驱动程序的文件名说明串
12463:    PUSH    SI                                ;┘
12464:    CALL    ChangedTerminator                 ;将文件名说明串的结束符改为参数分隔符
12465:    PUSH    ES                                ;┐ ES:SI 指向设备驱动程序文件名后面的参数串
12466:    PUSH    SI                                ;┘
12467:    PUSH    CS
```

```
12468:    POP     ES
12469:    PUSH    DS
12470:    PUSH    SI
12471:    LDS     SI,CS:DriverLoadAddr              ;┐
12472:    TEST    WORD PTR [SI+4],8000H             ;├ 若是安装字符设备驱动程序,则跳转
12473:    JNZ     Pro_Config57                      ;┘
12474:    LDS     SI,CS:Address_ListOfList          ;┐
12475:    CMP     BYTE PTR [SI+20H],1AH             ;├ 若已安装的逻辑驱动器数<26,则跳转
12476:    JB      Pro_Config57                      ;┘
12477:    POP     SI
12478:    POP     DS
12479:    POP     SI
12480:    POP     ES
12481:    JMP     SHORT Pro_Config61                ;转去显示错误信息"Too many block devices"
12482: Pro_Config57:
12483:    POP     SI
12484:    POP     DS
12485:    CALL    CurrAvailableMem                  ;确定当前允许使用的常规内 ┐
                                                    ;存大小                   │
12486:    CALL    SetAvailableMemSeg                ;设置当前活动进程可使用的 │
                                                    ;最大内存段地址           │
12487:    CMP     CS:DriverCounter,0                    ;┐                    │
12488:    JNE     Pro_Config58                          ;│                    │
12489:    MOV     WORD PTR CS:RequestForInit+0EH,0      ;├ 预置设备驱动程序的 │
12490:    MOV     BX,CS:MaxSeg_CurrDriver               ;├ 结束地址           ├ 初始化设备
12491:    MOV     WORD PTR CS:RequestForInit+10H,BX;    ├                     ├ 驱动程序
12492: Pro_Config58:                                    ;┘                    │
12493:    MOV     BX,6                              ;┐ 执行设备驱动程序的策略 │
12494:    CALL    RequestDrive                      ;┘ 过程                   │
12495:    MOV     BX,8                              ;┐ 执行设备驱动程序的中断 │
12496:    CALL    RequestDrive                      ;┘ 过程                   │
12497:    CALL    RecoverMemSize                    ;恢复系统的常规内存大小值 ┘
12498:    MOV     AX,WORD PTR CS:RequestForInit+0EH ;┐
12499:    MOV     WORD PTR CS:EndAddrOfDriver,AX    ;├ 设置自由内存的起始地址
12500:    MOV     AX,WORD PTR CS:RequestForInit+10H ;│
12501:    MOV     WORD PTR CS:EndAddrOfDriver+2,AX  ;┘
12502:    CMP     CS:UMB_Flag,0                     ;┐ 若 DOS 不提供与 UMB 的链接(即设备驱动程序
12503:    JE      Pro_Config59                      ;┘ 安装在常规内存中),则跳转
12504:    CALL    CreateEntireUMBChain              ;合并所有相邻的自由 UMB 内存块
12505: Pro_Config59:
12506:    CMP     CS:DOS_Install_Site,0FFH          ;┐ 若 IO2 模块和 MSDOS2 模块安装在常规内存中
12507:    JNE     Pro_Config60                      ;┘ 或已安装在 HMA 中,则跳转
12508:    CALL    Install1_IO2_MSDOS2_InHMA         ;将 IO2 模块和 MSDOS2 模块安装在 HMA 中
```

```
12509: Pro_Config60:
12510:     POP     SI
12511:     POP     DS
12512:     MOV     BYTE PTR [SI],0                 ;更换文件名说明串的结束符
12513:     PUSH    CS
12514:     POP     DS
12515:     JMP     SHORT Pro_Config64
12516: Pro_Config61:
12517:     PUSH    CS                              ;┐
12518:     POP     DS                              ;├ 显示错误信息"Too many block devices"
12519:     MOV     DX,offset Err_BlockDev          ;├
12520:     CALL    DisplayString                   ;┘
12521: Pro_Config62:
12522:     POP     SI
12523:     POP     ES
12524:     PUSH    CS
12525:     POP     DS
12526:     CMP     CS:DispLineNumFlag,0            ;┐ 若不显示错误系统配置命令行的行号,则跳转
12527:     JE      Pro_Config63                    ;┘
12528:     CALL    OutputErrSiteInfo               ;显示错误信息"Error in CONFIG.SYS Line
                                                   ;XXXXX"
12529:     MOV     CS:DispLineNumFlag,0            ;清"允许显示错误系统配置命令行的行号"标志
12530: Pro_Config63:
12531:     JMP     Pro_Config7                     ;转去为识别下一个命令行作准备
12532: Pro_Config64:
12533:     MOV     AX,WORD PTR CS:EndAddrOfDriver+2  ;┐
12534:     CMP     AX,CS:MaxSeg_CurrDriver           ;├ 若有足够内存安装设备驱动程序,则跳
12535:     JBE     Pro_Config65                      ;┘ 转
12536:     POP     SI
12537:     POP     ES
12538:     JMP     Pro_Config53                    ;当内存不够时,转去进行出错处理
12539: Pro_Config65:
12540:     LDS     DX,CS:DriverLoadAddr            ;┐ DS:DX(SI)指向当前正在安装的设备驱动程序
12541:     MOV     SI,DX                           ;┘
12542:     LES     DI,CS:Address_ListOfList        ;ES:DI 指向多重表
12543:     MOV     AX,[SI+4]                       ;AX←设备属性字
12544:     TEST    AX,8000H                        ;┐ 若正在安装块设备驱动程序,则跳转
12545:     JZ      Pro_Config69                    ;┘
12546:     OR      CS:FlagOfSCB,2                  ;置"设置子段控制块大小"的标志位
12547:     CALL    ChkDriver                       ;转换自由内存的起始地址值格式,并检查内存状
                                                   ;态
12548: Pro_Config66:
12549:     JC      Pro_Config62                    ;若没有足够内存,则跳转
```

```
12550:    TEST    AX,1                          ;┐ 若不是安装替代标准输入设备的设备驱动程
12551:    JZ      Pro_Config67                  ;┘ 序,则跳转
12552:    MOV     ES:[DI+0CH],DX                ;┐ 保存标准输入设备的设备驱动程序标题的起始
12553:    MOV     ES:[DI+0EH],DS                ;┘ 地址
12554: Pro_Config67:
12555:    TEST    AX,8                          ;┐ 若不是安装替代时钟设备的设备驱动程序,则
12556:    JZ      Pro_Config68                  ;┘ 跳转
12557:    MOV     ES:[DI+8],DX                  ;┐ 保存时钟设备的设备驱动程序标题的起始地址
12558:    MOV     ES:[DI+0AH],DS                ;┘
12559: Pro_Config68:
12560:    JMP     Pro_Config74
12561: Pro_Config69:
12562:    MOV     AL,CS:RequestForInit+0DH      ;┐ 若块设备驱动程序初始化返回的部件数
12563:    OR      AL,AL                         ;│ (即逻辑驱动器数)为 0,则跳转
12564:    JZ      Pro_Config62                  ;┘
12565:    MOV     [SI+0AH],AL                   ;保存块设备驱动程序支持的部件数
12566:    CBW                                   ;┐ CX←块设备驱动程序支持的部件数
12567:    MOV     CX,AX                         ;┘
12568:    MOV     DH,AH                         ;DH←部件号(0,即相对逻辑驱动器号)
12569:    MOV     DL,ES:[DI+20H]                ;DL←起始逻辑驱动器号
12570:    MOV     AH,DL                         ;┐
12571:    ADD     AH,AL                         ;│
12572:    CMP     AH,1AH                        ;│ 若当前逻辑驱动器总数超过 26 个,则转去显示
12573:    JBE     Pro_Config70                  ;│ 错误信息"Too many block devices"
12574:    JMP     Pro_Config61                  ;│
12575: Pro_Config70:                            ;┘
12576:    OR      CS:FlagOfSCB,2                ;置"设置子段控制块大小"的标志位
12577:    CALL    ChkDriver                     ;转换自由内存的起始地址值格式,并检查内存状
                                                ;态
12578:    JC      Pro_Config66                  ;若没有足够内存,则跳转
12579:    ADD     ES:[DI+20H],AL                ;修改系统中允许访问的逻辑驱动器数(MSDOS1
                                                ;模块的变量)
12580:    ADD     CS:RequestForInit+16H,AL      ;指定下一个块设备驱动程序的起始逻辑驱动器号
                                                ;(SysInit—Ⅱ变量)
12581:    LDS     BX,DWORD PTR CS:RequestForInit+12H ;块设备的 BPB 指针数组的起始地址
                                                ;→DS:BX
12582: Pro_Config71:
12583:    LES     BP,CS:Address_ListOfList      ;┐ ES:BP 指向 DDPB 链的头结点
12584:    LES     BP,DWORD PTR ES:[BP+0]        ;┘
12585: Pro_Config72:                            ;┐
12586:    CMP     WORD PTR ES:[BP+19H],0FFFFH   ;│
12587:    JE      Pro_Config73                  ;│ 搜索 DDPB 链的尾结点
12588:    LES     BP,DWORD PTR ES:[BP+19H]      ;│
```

```
12589:    JMP     SHORT Pro_Config72                ;|
12590: Pro_Config73:                                ;┘
12591:    MOV     AX,WORD PTR CS:EndAddrOfDriver    ;┐
12592:    MOV     ES:[BP+19H],AX                    ;| 在 DDPb 链尾追加一个新的 DDPB 结点
12593:    MOV     AX,WORD PTR CS:EndAddrOfDriver+2  ;|
12594:    MOV     ES:[BP+1BH],AX                    ;┘
12595:    LES     BP,DWORD PTR CS:EndAddrOfDriver   ;ES:BP 指向新的 DDPB
12596:    ADD     WORD PTR CS:EndAddrOfDriver,21H   ;修改自由内存的起始地址,留出 DDPB 所需
                                                    ;的内存空间
12597:    CALL    ConvertAddrFMT                    ;转换自由内存的起始地址值格式
12598:    MOV     WORD PTR ES:[BP+19H],0FFFFH       ;置尾结点的指针域
12599:    MOV     BYTE PTR ES:[BP+18H],0FFH         ;置介质更换标志
12600:    MOV     SI,[BX]                           ;DS:SI 指向当前部件的 BPB 参数块
12601:    INC     BX                          ;┐ 修改 BPB 指针数组的访问指针值
12602:    INC     BX                          ;┘
12603:    MOV     ES:[BP],DX                  ;设置逻辑驱动器号和部件号
12604:    MOV     AH,53H                      ;┐ 依据 BPB 参数块设置当前部件的 DDPB 的参数
12605:    INT     21H                         ;┘ 值
12606:    MOV     AX,ES:[BP+2]                ;┐
12607:    PUSH    ES                          ;|
12608:    LES     DI,CS:Address_ListOfList    ;| 若块设备的扇区尺寸大于系统允许的扇区大
12609:    CMP     AX,ES:[DI+10H]              ;| 小,则转去显示错误信息"Sector size too large in
12610:    POP     ES                          ;| file"
12611:    JA      Pro_Config76                ;┘
12612:    PUSH    DS                          ;┐
12613:    PUSH    DX                          ;|
12614:    LDS     DX,CS:DriverLoadAddr        ;|
12615:    MOV     ES:[BP+13H],DX              ;| 设备驱动程序标题的起始地址→DDPB
12616:    MOV     ES:[BP+15H],DS              ;|
12617:    POP     DX                          ;|
12618:    POP     DS                          ;┘
12619:    INC     DX                          ;修改逻辑驱动器号
12620:    INC     DH                          ;修改部件号
12621:    LOOP    Pro_Config71                ;转去建立其它部件的 DDPB
12622:    PUSH    CS                          ;┐
12623:    POP     DS                          ;| 重新为系统中每个逻辑驱动器各自建立一个
12624:    CALL    Create_CDS                  ;┘ CDS
12625: Pro_Config74:
12626:    LES     DI,CS:Address_ListOfList    ;ES:DI 指向多重表
12627:    MOV     CX,ES:[DI+22H]              ;┐
12628:    MOV     DX,ES:[DI+24H]              ;|
12629:    LDS     SI,CS:DriverLoadAddr        ;|
12630:    MOV     ES:[DI+22H],SI              ;| 将设备驱动程序标题插入到设备驱动程序标题
```

```
12631:    MOV     ES:[DI+24H],DS                ;|链中,同时它作为新的头结点
12632:    MOV     AX,[SI]                       ;|
12633:    MOV     WORD PTR CS:DriverLoadAddr,AX;|
12634:    MOV     [SI],CX                       ;|
12635:    MOV     WORD PTR [SI+2],DX            ;┘
12636:    POP     SI
12637:    POP     ES
12638:    INC     AX                            ;┐若当前装入的设备驱动程序文件中只有一个设
12639:    JZ      Pro_Config75                  ;┘备驱动程序,则跳转
12640:    INC     CS:DriverCounter              ;设备驱动程序文件支持的设备驱动程序计数器
                                                ;加 1
12641:    CALL    SetSizeOfSCB                  ;设置设备驱动程序对应的子段内存块的大小,并
                                                ;确定下次使用的自由内存的起始地址
12642:    JMP     Pro_Config56                  ;转去初始化其它设备驱动程序
12643: Pro_Config75:
12644:    MOV     CS:DriverCounter,0            ;清设备驱动程序文件支持的设备驱动程序计数器
12645:    CALL    SetSizeOfSCB                  ;设置设备驱动程序对应的子段内存块的大小,
                                                ;并确定下次使用的自由内存的起始地址
12646:    JMP     Pro_Config7                   ;转去为识别下一个命令行作准备
12647: Pro_Config76:
12648:    POP     SI
12649:    POP     ES
12650:    MOV     DX,offset BigSectorSize       ;┐
12651:    MOV     BX,offset OutputCRLF          ;|显示错误信息"Sector size too large in file"
12652:    CALL    Disp_CharStr1                 ;┘
12653:    JMP     Pro_Config7                   ;转去为识别下一个命令行作准备
12654: Pro_Config77:
12655:    CMP     AH,'Q'                        ;┐
12656:    JE      Pro_Config78                  ;|若不是 country 系统配置命令,则跳转
12657:    JMP     Pro_Config100                 ;┘
12658: Pro_Config78:                            ;处理 CONFIG.SYS 的 country 系统配置命令
12659:    MOV     BYTE PTR CS:CountryFile,0     ;清文件名说明串
12660:    MOV     CS:CodePageNumber,0
12661:    MOV     DI,offset CPIB_Q              ;DI 指向 country 系统配置命令的参数信息块
12662:    XOR     CX,CX                         ;清已取出的参数项数
12663:    MOV     DX,CX
12664: Pro_Config79:
12665:    CALL    GetCurrParaValue              ;取出当前参数项的参数值
12666:    JNC     Pro_Config80                  ;若参数项合法,则跳转
12667:    CALL    Pro_Config97                  ;输出 country 系统配置命令格式错的错误信息
12668:    MOV     CS:CountryCode,0FFFFH         ;设置 country 系统配置命令格式错的标志
12669:    JMP     SHORT Pro_Config85
12670: Pro_Config80:
```

```
12671:    CMP     AX,-1                       ;┐若 country 系统配置命令正常结束,则跳转
12672:    JE      Pro_Config85                ;┘
12673:    CMP     CS:ParaValueTypeCode,1      ;┐若当前参数项是指定文件名说明串,则跳转
12674:    JNE     Pro_Config83                ;┘
12675:    MOV     AX,WORD PTR CS:ParaValue    ;AX←参数值
12676:    CMP     CX,1                        ;┐若当前参数项是指定代码页号,则跳转
12677:    JNE     Pro_Config81                ;┘
12678:    MOV     CS:CountryCode,AX           ;保存国家代码
12679:    JMP     SHORT Pro_Config82
12680: Pro_Config81:
12681:    MOV     CS:CodePageNumber,AX        ;保存代码页号
12682: Pro_Config82:
12683:    JMP     SHORT Pro_Config84
12684: Pro_Config83:
12685:    PUSH    DS                          ;┐
12686:    PUSH    ES                          ;│
12687:    PUSH    SI                          ;│
12688:    PUSH    DI                          ;│
12689:    PUSH    CS                          ;│
12690:    POP     ES                          ;│
12691:    LDS     SI,CS:ParaValue             ;│更换国家信息文件的文件名说明串
12692:    MOV     DI,offset CountryFile       ;│
12693:    CALL    Copy_String                 ;│
12694:    POP     DI                          ;│
12695:    POP     SI                          ;│
12696:    POP     ES                          ;│
12697:    POP     DS                          ;┘
12698: Pro_Config84:
12699:    JMP     SHORT Pro_Config79          ;转去继续取后续参数项
12700: Pro_Config85:
12701:    CMP     CS:CountryCode,0FFFFH       ;┐若 country 系统配置命令格式错,则转去为识别
12702:    JNE     Pro_Config87                ;│下一个命令行作准备
12703:    JMP     Pro_Config7                 ;┘
12704: Pro_Config86:
12705:    STC
12706:    MOV     DX,offset InvalidCodePage   ;DX 指向错误信息"Invalid country code or code
                                              ;page"
12707:    JMP     Pro_Config95
12708: Pro_Config87:
12709:    CMP     BYTE PTR CS:CountryFile,0   ;┐若未指定国家信息文件,则跳转
12710:    JE      Pro_Config88                ;┘
12711:    MOV     DX,offset CountryFile       ;DX 指向由 country 系统配置命令指定的文件名说
                                              ;明串
```

```
12712:      JMP     SHORT Pro_Config89
12713: Pro_Config88:
12714:      MOV     DX,offset CountryFile + 2       ;DX 指向缺省的国家信息文件的文件名说明串
12715: Pro_Config89:
12716:      MOV     AX,3D00H                        ;┐
12717:      STC                                     ;│以"读"方式打开国家信息文件
12718:      INT     21H                             ;┘
12719:      JC      Pro_Config91                    ;当打开失败时,转去给出错误信息
12720:      MOV     CS:FileHandle,AX                ;保存国家信息文件的文件句柄
12721:      MOV     BX,AX                           ;BX←国家信息文件的文件名柄
12722:      MOV     AX,CS:CountryCode               ;AX←国家代码
12723:      MOV     DX,CS:CodePageNumber            ;DX←代码页号
12724:      MOV     CX,CS:SEG_FreeMemory            ;┐
12725:      ADD     CX,180H                         ;│没有足够的自由内存用于处理国家信息文件,
12726:      CMP     CX,CS:Seg_CDSArray              ;│则跳转
12727:      JA      Pro_Config94                    ;┘
12728:      MOV     SI,offset CountryFile           ;┐
12729:      CMP     BYTE PTR [SI],0                 ;│
12730:      JNE     Pro_Config90                    ;│DS:SI 指向当前使用的国家信息文件的文件名
12731:      INC     SI                              ;│说明串
12732:      INC     SI                              ;│
12733: Pro_Config90:                                ;┘
12734:      LES     DI,CS:Addr_ExtCIMB              ;ES:DI 指向位于 MSDOS1 模块中的扩充国家信息
                                                    ;管理块
12735:      PUSH    DI                              ;┐
12736:      ADD     DI,8                            ;│修改扩充国家信息管理块中的国家信息文件名
12737:      CALL    Copy_String                     ;│
12738:      POP     DI                              ;┘
12739:      MOV     CX,CS:SEG_FreeMemory            ;┐
12740:      MOV     DS,CX                           ;│DS:SI←当前自由内存的起始地址
12741:      XOR     SI,SI                           ;┘
12742:      CALL    Process_Country                 ;根据 country 系统配置命令指定的国家信息文件,
                                                    ;设置扩充国家信息管理块中的数据
12743:      JNC     Pro_Config95                    ;若成功,则跳转
12744:      CMP     CX,-1                           ;┐若指定的国家代码或代码页号非法,则转去显
12745:      JE      Pro_Config86                    ;┘示错误信息
12746: Pro_Config91:
12747:      PUSH    CS                              ;┐
12748:      POP     ES                              ;│
12749:      CMP     BYTE PTR CS:CountryFile,0       ;│
12750:      JE      Pro_Config92                    ;│
12751:      MOV     SI,offset CountryFile           ;│
12752:      JMP     SHORT Pro_Config93              ;│输出错误信息:"Bad or missing"、文件名说明串
```

```
12753: Pro_Config92:                          ;| 和错误的 country 系统配置命令的位置信息
12754:     MOV     SI,offset CountryFile+2    ;|
12755: Pro_Config93:                          ;|
12756:     CALL    Disp_ErrorStr              ;|
12757:     MOV     CX,CS:SEG_Config           ;|
12758:     MOV     ES,CX                      ;┘
12759:     JMP     SHORT Pro_Config96
12760: Pro_Config94:
12761:     MOV     DX,offset Err_CountryMem   ;DX 指向错误信息"Insufficient memory for
                                              ;COUNTRY.SYS file"
12762: Pro_Config95:
12763:     MOV     CX,CS:SEG_Config           ;┐ ES←存放 CONFIG.SYS 文件内容的缓冲区的
12764:     MOV     ES,CX                      ;┘ 段地址值
12765:     PUSH    CS
12766:     POP     DS
12767:     JNC     Pro_Config96               ;若处理指定国家信息文件成功,则跳转
12768:     CALL    DisplayString              ;显示错误信息
12769:     CALL    OutputErrSiteInfo          ;显示错误 country 系统配置命令在 CONFIG.SYS
                                              ;文本文件中的位置信息
12770: Pro_Config96:
12771:     MOV     BX,CS:FileHandle           ;┐
12772:     MOV     AH,3EH                     ;| 关闭指定的国家信息文件
12773:     INT     21H                        ;┘
12774:     JMP     Pro_Config7                ;转去为识别下一个命令行作准备
12775: ;=============================================================
12776: ;相对地址偏移:277A
12777: ;功能:输出 country 系统配置命令格式错的错误信息
12778: ;入口参数:AX=取参数值的返回码
12779: ;出口参数:无
12780: ;=============================================================
12781: Pro_Config97  LABEL  NEAR
12782:     CMP     AX,6                       ;┐ 若不是"参数值越界"错,则跳转
12783:     JNE     Pro_Config98               ;┘
12784:     MOV     DX,offset InvalidCodePage  ;DX 指向错误信息"Invalid country code or code
                                              ;page"
12785:     JMP     SHORT Pro_Config99
12786: Pro_Config98:
12787:     MOV     DX,offset Err_CountryCMD   ;DX 指向错误信息"Error in COUNTRY command"
12788: Pro_Config99:
12789:     CALL    DisplayString              ;显示 DX 指定的错误信息
12790:     CALL    OutputErrSiteInfo          ;输出错误的 country 系统配置命令在 CONFIG.SYS
                                              ;文本文件中的位置信息
12791:     RET
```

```
12792: Pro_Config100:
12793:     CMP     AH,' F'                 ;┐若不是 files 系统配置命令,则跳转
12794:     JNE     Pro_Config105           ;┘
12795: ;处理 CONFIG.SYS 的 files 系统配置命令
12796:     MOV     DI,offset CPIB_F        ;DI 指向 files 系统配置命令的参数信息块
12797:     XOR     CX,CX                   ;清已取出的参数项数
12798:     MOV     DX,CX
12799: Pro_Config101:
12800:     CALL    GetCurrParaValue        ;取出当前参数项的参数值
12801:     JNC     Pro_Config102           ;若参数项合法,则跳转
12802:     CALL    OutputErrInfo           ;输出 files 系统配置命令格式错的错误信息
12803:     JMP     SHORT Pro_Config104
12804: Pro_Config102:
12805:     CMP     AX,-1                   ;┐若 files 系统配置命令正常结束,则跳转
12806:     JE      Pro_Config103           ;┘
12807:     MOV     AL,BYTE PTR CS:ParaValue ;┐取出并保存"DOS 可以同时存取的文件数"
12808:     MOV     CS:NumberOfSFT,AL       ;┘
12809:     JMP     SHORT Pro_Config101     ;转去继续取后续参数项
12810: Pro_Config103:
12811:     MOV     AL,CS:NumberOfSFT       ;┐设置"DOS 可以同时存取的文件数"
12812:     MOV     CS:NumberOfHandle,AL    ;┘
12813: Pro_Config104:
12814:     JMP     Pro_Config7             ;转去为识别下一个命令行作准备
12815: Pro_Config105:
12816:     CMP     AH,' L'                 ;┐若不是 lastdrive 系统配置命令,则跳转
12817:     JNE     Pro_Config110           ;┘
12818: ;处理 CONFIG.SYS 的 lastdrive 系统配置命令
12819:     MOV     DI,offset CPIB_L        ;DI 指向 lastdrive 系统配置命令的参数信息块
12820:     XOR     CX,CX                   ;清已取出的参数项数
12821:     MOV     DX,CX
12822: Pro_Config106:
12823:     CALL    GetCurrParaValue        ;取出当前参数项的参数值
12824:     JNC     Pro_Config107           ;若参数值合法,则跳转
12825:     CALL    OutputErrInfo           ;输出 lastdrive 系统配置命令格式错的错误信息
12826:     JMP     SHORT Pro_Config109
12827: Pro_Config107:
12828:     CMP     AX,-1                   ;┐若 lastdrive 系统配置命令正常结束,则跳转
12829:     JE      Pro_Config108           ;┘
12830:     MOV     AL,BYTE PTR CS:ParaValue ;┐取出并保存"允许访问的最大逻辑驱动器数"
12831:     MOV     CS:LastDrive_L,AL       ;┘
12832:     JMP     SHORT Pro_Config106
12833: Pro_Config108:
12834:     MOV     AL,CS:LastDrive_L       ;┐设置"允许访问的最大逻辑驱动器数"
```

```
12835:    MOV     CS:LastDrive,AL                ;┘
12836: Pro_Config109:
12837:    JMP     Pro_Config7                    ;转去为识别下一个命令行作准备
12838: Pro_Config110:
12839:    CMP     AH,' P'                        ;┐若不是 drivparm 系统配置命令,则跳转
12840:    JNE     Pro_Config112                  ;┘
12841: ;处理 CONFIG.SYS 的 drivparm 系统配置命令
12842:    CALL    Process_DrivParm               ;取出 drivparm 系统配置命令参数,并参考 drivparm
                                                 ;系统配置命令指定的开关参数设置对应的设备参
                                                 ;数块 Packet
12843:    JC      Pro_Config111                  ;若 drivparm 系统配置命令错,则转去显示错误信
                                                 ;息
12844:    CALL    Set_DevicePara                 ;设置设备参数
12845:    CALL    Set_Packet                     ;修改"设备参数块(Packet)"的部分参数值
12846:    JC      Pro_Config111                  ;若"设置设备参数"操作失败,则转去显示错误信
                                                 ;息
12847:    JMP     Pro_Config7                    ;转去为识别下一个命令行作准备
12848: Pro_Config111:
12849:    JMP     Pro_Config155
12850: Pro_Config112:
12851:    CMP     AH,' K'                        ;┐
12852:    JE      Pro_Config113                  ;├若不是 stacks 系统配置命令,则跳转
12853:    JMP     Pro_Config125                  ;┘
12854: Pro_Config113:                            ;处理 CONFIG.SYS 的 stacks 系统配置命令
12855:    MOV     DI,offset CPIB_K               ;DI 指向 stacks 系统配置命令的参数信息块
12856:    XOR     CX,CX                          ;清已取出的参数项数
12857:    MOV     DX,CX
12858: Pro_Config114:
12859:    CALL    GetCurrParaValue               ;取出当前参数项的参数值
12860:    JNC     Pro_Config115                  ;若参数项合法,则跳转
12861:    MOV     DX,offset InvalidSTPara        ;┐显示错误信息"Invalid STACK parameters"
12862:    CALL    DisplayString                  ;┘
12863:    CALL    OutputErrSiteInfo              ;输出错误 stacks 系统配置命令在 CONFIG.SYS 文
                                                 ;本文件中的位置信息
12864:    JMP     Pro_Config124
12865: Pro_Config115:
12866:    CMP     AX,-1                          ;┐若 stacks 系统配置命令正常结束,则跳转
12867:    JE      Pro_Config118                  ;┘
12868:    MOV     AX,WORD PTR CS:ParaValue       ;AX←当前参数项的参数值
12869:    CMP     CX,1                           ;┐若当前参数项是指定"堆栈尺寸",则跳转
12870:    JNE     Pro_Config116                  ;┘
12871:    MOV     CS:N_Stacks,AX                 ;保存"动态堆栈数"
12872:    JMP     SHORT Pro_Config117
```

```
12873: Pro_Config116:
12874:     MOV     CS:S_Stacks,AX              ;保存"堆栈尺寸"(字节数)
12875: Pro_Config117:
12876:     JMP     SHORT Pro_Config114         ;转去继续取后续参数项
12877: Pro_Config118:
12878:     CMP     CS:N_Stacks,0               ;┐
12879:     JE      Pro_Config121               ;|
12880:     CMP     CS:N_Stacks,8               ;|
12881:     JB      Pro_Config119               ;|
12882:     CMP     CS:S_Stacks,32              ;|
12883:     JAE     Pro_Config120               ;|
12884: Pro_Config119:                          ;| 检查"stacks=n,s"的 n、s 参数值是否满足对最
12885:     MOV     CS:N_Stacks,0FFFFH          ;| 小值的规定
12886: Pro_Config120:                          ;|
12887:     JMP     SHORT Pro_Config122         ;|
12888: Pro_Config121:                          ;|
12889:     CMP     CS:S_Stacks,0               ;|
12890:     JE      Pro_Config122               ;|
12891:     MOV     CS:N_Stacks,0FFFFH          ;|
12892: Pro_Config122:                          ;┘
12893:     CMP     CS:N_Stacks,0FFFFH          ;┐ 若 n 参数值和 s 参数值合法,则跳转
12894:     JNE     Pro_Config123               ;┘
12895:     MOV     CS:NumberOfStack,9          ;设置缺省的"动态堆栈数"
12896:     MOV     CS:SizeOfStack,80H          ;设置缺省的"堆栈尺寸"
12897:     MOV     CS:OFS_StackField,0
12898:     MOV     DX,offset InvalidSTPara     ;┐ 显示错误信息"Invalid STACK parameters"
12899:     CALL    DisplayString               ;┘
12900:     CALL    OutputErrSiteInfo           ;输出错误 stacks 系统配置命令在 CONFIG.SYS 文
                                               ;本文件中的位置信息
12901:     JMP     SHORT Pro_Config124
12902: Pro_Config123:
12903:     MOV     AX,CS:N_Stacks              ;┐ 设置"动态堆栈数"
12904:     MOV     CS:NumberOfStack,AX         ;┘
12905:     MOV     AX,CS:S_Stacks              ;┐ 设置"堆栈尺寸"
12906:     MOV     CS:SizeOfStack,AX           ;┘
12907:     MOV     CS:OFS_StackField,0FFFFH    ;设置 CONFIG.SYS 文件中有"stacks 系统配置命
                                               ;令"的标志
12908: Pro_Config124:
12909:     JMP     Pro_Config7                 ;转去为识别下一个命令行作准备
12910: Pro_Config125:
12911:     CMP     AH,'S'                      ;┐ 若不是 shell 系统配置命令,则跳转
12912:     JNE     Pro_Config131               ;┘
12913: ;处理 CONFIG.SYS 的 shell 系统配置命令
```

```
12914:      MOV     CS:CMDLine+1,0          ;清执行缺省的命令处理器(COMMAND.COM)时
                                            ;传给它的命令行参数
12915:      MOV     DI,offset Path_Shell+1  ;DS:DI 指向存放命令处理器文件名说明串的缓冲
                                            ;区
12916:      MOV     [DI-1],AL               ;保存属于命令处理器文件名说明串的字符
12917:  Pro_Config126:                      ;┐
12918:      CALL    Get_CharOfConfig        ;│
12919:      OR      AL,AL                   ;│
12920:      JZ      Pro_Config129           ;│
12921:      CMP     AL,' '                  ;├替换缺省的命令处理器的文件名说明串
12922:      JB      Pro_Config127           ;│
12923:      MOV     [DI],AL                 ;│
12924:      INC     DI                      ;│
12925:      JMP     SHORT Pro_Config126     ;┘
12926:  Pro_Config127:
12927:      MOV     BYTE PTR [DI],0         ;设置文件名说明串的结束符
12928:      CALL    Get_CharOfConfig        ;┐
12929:      CMP     AL,0AH                  ;│
12930:      JNE     Pro_Config128           ;├为处理下一个 CONFIG.SYS 命令作准备
12931:      CALL    Get_CharOfConfig        ;│
12932:  Pro_Config128:                      ;┘
12933:      JMP     Pro_Config6
12934:  Pro_Config129:
12935:      MOV     BYTE PTR [DI],0         ;设置文件名说明串的结束符
12936:      MOV     DI,offset CMDLine+1     ;┐
12937:  Pro_Config130:                      ;│
12938:      CALL    Get_CharOfConfig        ;│
12939:      CMP     AL,' '                  ;├设置执行指定的命令处理器时使用的命令行参
12940:      JB      Pro_Config127           ;│数
12941:      MOV     [DI],AL                 ;│
12942:      INC     DI                      ;│
12943:      JMP     SHORT Pro_Config130     ;┘
12944:  Pro_Config131:
12945:      CMP     AH,'X'                  ;┐若不是 fcbs 系统配置命令,则跳转
12946:      JNE     Pro_Config138           ;┘
12947:  ;处理 CONFIG.SYS 的 fcbs 系统配置命令
12948:      MOV     DI,offset CPIB_X        ;DI 指向 fcbs 系统配置命令的参数信息块
12949:      XOR     CX,CX                   ;清已取出的参数项数
12950:      MOV     DX,CX
12951:  Pro_Config132:
12952:      CALL    GetCurrParaValue        ;取出当前参数项的参数值
12953:      JNC     Pro_Config133           ;若参数项合法,则跳转
12954:      CALL    OutputErrInfo           ;输出 fcbs 系统配置命令格式错的错误信息
```

```
12955:    JMP     SHORT Pro_Config137
12956: Pro_Config133:
12957:    CMP     AX,-1                    ;┐若 fcbs 系统配置命令正常结束,则跳转
12958:    JE      Pro_Config136            ;┘
12959:    MOV     AL,BYTE PTR CS:ParaValue ;AL←当前参数项的参数值
12960:    CMP     CX,1                     ;┐若当前参数项是指定"fcbs=x,y 的 y 参数",则
12961:    JNE     Pro_Config134            ;┘跳转
12962:    MOV     CS:N_FCBS,AL             ;保存"fcbs=x,y"的 x 参数值
12963:    JMP     SHORT Pro_Config135
12964: Pro_Config134:
12965:    MOV     CS:S_FCBS,AL             ;保存"fcbs=x,y"的 y 参数值
12966: Pro_Config135:
12967:    JMP     SHORT Pro_Config132
12968: Pro_Config136:
12969:    MOV     AL,CS:N_FCBS             ;设置"DOS 可以同时打开的 FCB 数"
12970:    MOV     CS:NumberOfFCB,AL        ;清"由 FCB 打开而 DOS 不能自动关闭的文件
                                           ;数"(它表示 fcbs=x,y 的 y 参数已无意义)
12971:    MOV     CS:NumberOfKeepFCB,0
12972: Pro_Config137:
12973:    JMP     Pro_Config7              ;转去为识别下一个命令行作准备
12974: Pro_Config138:
12975:    CMP     AH,' Y'                  ;┐若不是 comment 系统配置命令,则跳转
12976:    JNE     Pro_Config140            ;┘
12977: Pro_Config139:                      ;处理 CONFIG.SYS 的 comment、rem 系统配置命令
12978:    DEC     CS:Ptr_Config            ;┐恢复先前取出的字符(它不属于 comment、rem
12979:    INC     CS:ConfigCount           ;┘系统配置命令)
12980:    JMP     Pro_Config7              ;转去为识别下一个命令行作准备
12981: Pro_Config140:
12982:    CMP     AH,' 0'                  ;┐若是 rem 系统配置命令,则跳转
12983:    JE      Pro_Config139            ;┘
12984:    CMP     AH,' 1'                  ;┐
12985:    JE      Pro_Config141            ;┘若不是 switches 系统配置命令,则跳转
12986:    JMP     Pro_Config150            ;┘
12987: Pro_Config141:                      ;处理 CONFIG.SYS 的 switches 系统配置命令
12988:    MOV     DI,offset CPIB_1         ;DI 指向 switches 系统配置命令的参数信息块
12989:    XOR     CX,CX                    ;CX←0(参数项序号)
12990:    MOV     DX,CX
12991: Pro_Config142:
12992:    CALL    GetCurrParaValue         ;取出当前参数项的参数值
12993:    JNC     Pro_Config143            ;若参数项合法,则跳转
12994:    CALL    OutputErrInfo            ;输出 switches 系统配置命令格式错的错误信息
12995:    JMP     SHORT Pro_Config149
12996: Pro_Config143:
```

```
12997:    CMP     AX,-1                          ;┐ 若 switches 系统配置命令正常结束,则跳转
12998:    JE      Pro_Config146                  ;┘
12999:    CMP     CS:StartAddrOfSwitchStr,offset SPS_SK_1 ;┐ 若当前参数项不是"/K"开关,则跳转
13000:    JNE     Pro_Config144                  ;┘
13001:    MOV     CS:SK_Switches,1               ;置"/K"开关标志
13002:    JMP     SHORT Pro_Config142            ;转去继续取后续开关参数
13003: Pro_Config144:
13004:    CMP     CS:StartAddrOfSwitchStr,offset SPS_ST_1 ;┐ 若当前参数项不是"/T"开关,则跳转
13005:    JNE     Pro_Config145                  ;┘
13006:    MOV     CS:ST_Switches,1               ;置"/T"开关标志
13007:    JMP     SHORT Pro_Config142            ;转去继续取后续开关参数
13008: Pro_Config145:
13009:    CMP     CS:StartAddrOfSwitchStr,offset SPS_SW_1 ;┐ 若当前参数项是非法开关,则转去继续
13010:    JNE     Pro_Config142                  ;┘ 取后续开关参数
13011:    MOV     CS:SW_Switches,1               ;置"/W"开关标志
13012:    JMP     SHORT Pro_Config142            ;转去继续取后续开关参数
13013: Pro_Config146:
13014:    CMP     CS:SK_Switches,1               ;┐
13015:    PUSH    DS                             ;|
13016:    MOV     AX,BIO_SEG                     ;| 若未指定"/K"开关,则跳转
13017:    MOV     DS,AX                          ;|
13018:    JNZ     Pro_Config147                  ;┘
13019:    MOV     ReadKeyCode,0                  ;"读常规键盘输入"的命令码┐ IO1 模块的变量
13020:    MOV     GetKeyStatus,1                 ;"取常规键盘状态"的命令码┘
13021: Pro_Config147:
13022:    MOV     AL,CS:ST_Switches              ;┐ 设置 IO1 模块中的变量值("/T"开关标志)
13023:    MOV     Switch_T,AL                    ;┘
13024:    CMP     CS:SW_Switches,0               ;┐ 若未指定"/W"开关,则跳转
13025:    JE      Pro_Config148                  ;┘
13026:    PUSH    ES                             ;┐
13027:    PUSH    BX                             ;|
13028:    MOV     AH,52H                         ;| 设置 MSDOS1 模块中的控制变量标志位
13029:    INT     21H                            ;| (WINA20.386 在指定的位置,而不使用
13030:    OR      BYTE PTR ES:[86H],2            ;| 根目录下的 WINA20.386 文件)
13031:    POP     BX                             ;|
13032:    POP     ES                             ;┘
13033: Pro_Config148:
13034:    POP     DS                             ;恢复 DS 寄存器值
13035: Pro_Config149:
13036:    JMP     Pro_Config7                    ;转去为识别下一个命令行作准备
13037: Pro_Config150:
13038:    CMP     AH,0FFH
13039:    JE      Pro_Config151
```

```
13040:    DEC     Ptr_Config                    ;┐ 恢复先前取出的字符
13041:    INC     ConfigCount                   ;┘
13042:    JMP     SHORT Pro_Config155           ;转去显示"非法系统配置命令"的错误信息
13043: Pro_Config151:
13044:    JMP     Pro_Config139
13045: ;=============================================================
13046: ;相对地址偏移:29E9
13047: ;功能:取出当前参数项的参数值
13048: ;入口参数:CX=已取出的参数项数
13049: ;          CS:DI 指向命令的参数信息块
13050: ;          ES:SI 指向系统配置命令字符串的待处理内容(即指向当前参数项字符串)
13051: ;出口参数:CF=0,参数项合法,AX=取参数的返回码;CF=1,参数项非法
13052: ;=============================================================
13053: GetCurrParaValue  LABEL  NEAR
13054:    PUSH    ES
13055:    PUSH    DS
13056:    PUSH    ES
13057:    POP     DS
13058:    PUSH    CS
13059:    POP     ES
13060:    MOV     WORD PTR CS:Ptr_ProcessConfig+2,DS  ;┐ 保存当前系统配置命令字符串的待处理
13061:    MOV     WORD PTR CS:Ptr_ProcessConfig,SI    ;┘ 内容的起始地址
13062:    MOV     DX,0                          ;┐ 取出当前参数项的参数值
13063:    CALL    GetIndicateParaValue          ;┘
13064:    CMP     AX,0                          ;┐ 若参数值已取出,并且合法,则跳转
13065:    JE      Pro_Config152                 ;┘
13066:    CMP     AX,-1                         ;┐ 若参数值无效,则跳转
13067:    JNE     Pro_Config153                 ;┘
13068: Pro_Config152:
13069:    CLC
13070:    JMP     SHORT Pro_Config154
13071: Pro_Config153:
13072:    STC
13073: Pro_Config154:
13074:    POP     DS
13075:    POP     ES
13076:    RET
13077: ;相对地址偏移:2A10
13078:    PUSH    CS                            ;┐
13079:    POP     DS                            ;|
13080:    MOV     DX,offset UnknownCMD          ;| 无用指令系列
13081:    CALL    DisplayString                 ;|
13082:    CALL    OutputErrSiteInfo             ;|
```

```
13083:    RET                                   ;┘
13084: Pro_Config155:
13085:    MOV     DX,offset UnknownCMD          ;┐ 显示错误信息"Unrecognized command in
13086:    CALL    DisplayString                 ;┘ CONFIG.SYS"
13087:    CALL    OutputErrSiteInfo             ;输出"非法系统配置命令"在 CONFIG.SYS 文本文
                                                ;件中的位置信息
13088:    JMP     Pro_Config7                   ;转去为识别下一个命令行作准备
13089: Process_Config  ENDP
13090: ;=============================================================
13091: ;相对地址偏移:2A28
13092: ;功能:输出系统配置命令格式错的错误信息
13093: ;入口参数:无
13094: ;出口参数:无
13095: ;=============================================================
13096: OutputErrInfo  PROC  NEAR
13097:    PUSH    DS
13098:    PUSH    DX
13099:    PUSH    SI
13100:    PUSH    CS                            ;┐
13101:    POP     DS                            ;│ 显示错误信息"Bad command or parameter — "
13102:    MOV     DX,offset InvalidPara         ;│
13103:    CALL    DisplayString                 ;┘
13104:    LDS     SI,DWORD PTR Ptr_ProcessConfig ;┐
13105: OutputErrInfo_1:                         ;│
13106:    MOV     DL,[SI]                       ;│
13107:    CMP     DL,0DH                        ;│
13108:    JE      OutputErrInfo_2               ;│ 显示系统配置命令行中不能识别的内容
13109:    MOV     AH,2                          ;│
13110:    INT     21H                           ;│
13111:    INC     SI                            ;│
13112:    JMP     SHORT OutputErrInfo_1         ;│
13113: OutputErrInfo_2:                         ;┘
13114:    PUSH    CS                            ;┐
13115:    POP     DS                            ;│ 输出回车换行,使光标移到下一行的起始位置
13116:    MOV     DX,offset OutputCRLF          ;│
13117:    CALL    DisplayString                 ;┘
13118:    CALL    OutputErrSiteInfo             ;输出错误系统配置命令在 CONFIG.SYS 文本文件
                                                ;中的位置信息
13119:    POP     SI
13120:    POP     DX
13121:    POP     DS
13122:    RET
13123: OutputErrInfo  ENDP
```

```
13124: ;=========== ==========  ==========  ==========  ==========
13125: ;相对地址偏移:2A54
13126: ;功能:从转换后的 CONFIG.SYS 文件内容中取出一个待处理的字符
13127: ;入口参数:无
13128: ;出口参数:CF=0,已取出一个字符,AL=取出的字符 ASCII 码;CF=1,未取出字符
13129: ;=========== ========== ========== ==========
13130: Get_CharOfConfig  PROC  NEAR
13131:     PUSH    CX
13132:     MOV     CX,ConfigCount              ;CX←CONFIG.SYS 待处理的字符数
13133:     JCXZ    Get_ChrOfConfig2            ;若 CONFIG.SYS 已全部处理完,则跳转
13134:     MOV     SI,Ptr_Config               ;┐取出一个字符→AL
13135:     MOV     AL,ES:[SI]                  ;┘
13136:     DEC     ConfigCount                 ;┐修改 CONFIG.SYS 待处理内容的管理变量
13137:     INC     Ptr_Config                  ;┘
13138:     CLC
13139: Get_ChrOfConfig1:
13140:     POP     CX
13141:     RET
13142: Get_ChrOfConfig2:                       ;CF←1(未取出字符)
13143:     STC
13144:     JMP     SHORT Get_ChrOfConfig1
13145: Get_CharOfConfig  ENDP
13146: ;相对地址偏移:2A70                      ;┐
13147:     MOV     DX,offset IncorrectOrder    ;|
13148:     CALL    DisplayString               ;| 无用指令系列
13149:     CALL    OutputErrLineNum            ;|
13150:     RETN                                ;┘
13151: ;=========== ========== ========== ==========
13152: ;相对地址偏移:2A7A
13153: ;功能:输出错误系统配置命令在 CONFIG.SYS 文本文件中的位置信息
13154: ;入口参数:无
13155: ;出口参数:无
13156: ;=========== ========== ========== ==========
13157: OutputErrSiteInfo  PROC  NEAR
13158:     PUSH    CS                          ;┐
13159:     POP     DS                          ;| 显示错误信息"Error in CONFIG.SYS line "
13160:     MOV     DX,offset Err_Config        ;|
13161:     CALL    DisplayString               ;┘
13162:     CALL    OutputErrLineNum            ;显示该系统配置命令的行号
13163:     RET
13164: OutputErrSiteInfo  ENDP
13165: ;=========== ========== ========== ==========
13166: ;相对地址偏移:2A86
```

```
13167: ;功能:显示错误系统配置命令在 CONFIG.SYS 文本文件中的行号
13168: ;入口参数:无
13169: ;出口参数:无
13170: ;=============================================
13171: OutputErrLineNum  PROC  NEAR
13172:     PUSH    ES
13173:     PUSH    DS
13174:     PUSH    DI
13175:     PUSH    CS
13176:     POP     ES
13177:     PUSH    CS                          ;┐ DS:DI 指向输出行号时使用的缓冲区
13178:     POP     DS                          ;┘
13179:     MOV     DI,offset OutputBuffer+4    ;CX←十进制的基数
13180:     MOV     CX,0AH                      ;AX←系统配置命令的行号
13181:     MOV     AX,CS:LineCounter
13182: OutputErrLineNum_1:
13183:     CMP     AX,0AH                      ;┐ 若行号<10,则跳转
13184:     JB      OutputErrLineNum_2          ;┘
13185:     XOR     DX,DX                       ;┐
13186:     DIV     CX                          ;| 保存当前数字位的数字符
13187:     OR      DL,'0'                      ;|
13188:     MOV     [DI],DL                     ;┘
13189:     DEC     DI                          ;修改缓存行号数字符的地址
13190:     JMP     SHORT OutputErrLineNum_1    ;转去继续计算行号的高位数字符
13191: OutputErrLineNum_2:
13192:     OR      AL,'0'                      ;┐ 存放高位数字符
13193:     MOV     [DI],AL                     ;┘
13194:     MOV     DX,DI                       ;┐ 显示系统配置命令行的行号
13195:     CALL    DisplayString               ;┘
13196:     POP     DI
13197:     POP     DS
13198:     POP     ES
13199:     RET
13200: OutputErrLineNum  ENDP
13201: ;=============================================
13202: ;相对地址偏移:2AB5
13203: ;功能:根据 dos 系统配置命令的参数值设置对应的控制变量值
13204: ;入口参数:无
13205: ;出口参数:无
13206: ;=============================================
13207: Save_dos_Para  PROC  NEAR
13208:     XOR     AH,AH                       ;┐ AX←dos 系统配置命令的参数值(关键字的编
13209:     MOV     AL,CS:KeywordNumber         ;┘ 码)
```

```
13210:      DEC     AX                          ;┐若是 HIGH 参数,则跳转
13211:      JZ      Save_dos_Para_3             ;┘
13212:      DEC     AX                          ;┐若是 LOW 参数,则跳转
13213:      JZ      Save_dos_Para_2             ;┘
13214:      DEC     AX                          ;┐若是 UMB 参数,则跳转
13215:      JZ      Save_dos_Para_1             ;┘
13216:      MOV     CS:UMB_Flag,0               ;清"MS—DOS 在常规内存和 UMB 之间保持链接"
                                                ;的标志(即 noumb)
13217:      RET
13218: Save_dos_Para_1:
13219:      MOV     CS:UMB_Flag,0FFH            ;设置"MS—DOS 在常规内存和 UMB 之间保持链
                                                ;接"的标志(umb)
13220:      RET
13221: Save_dos_Para_2:
13222:      MOV     CS:DOS_Install_Site,0       ;清 DOS 安装控制标志(0:DOS 安装在常规内存)
13223:      RET
13224: Save_dos_Para_3:
13225:      MOV     CS:DOS_Install_Site,0FFH    ;设置 DOS 安装控制标志(—1:DOS 的 IO2 模块和
                                                ;MSDOS2 模块安装在 HMA)
13226:      RET
13227: Save_dos_Para  ENDP
13228: ;=============================================================
13229: ;相对地址偏移:2AE0
13230: ;功能:确定当前允许使用的常规内存大小
13231: ;入口参数:无
13232: ;出口参数:无
13233: ;=============================================================
13234: CurrAvailableMem  PROC  NEAR
13235:      MOV     AX,CS:Seg_CDSArray          ;AX←当前最大可用的常规内存的段地址值
13236:      CALL    Chk_PROTMIN$                ;┐若正在安装"PROTMAN$"的设备驱动程序,则
13237:      JZ      CurrAvailMem_1              ;┘跳转
13238:      CMP     CS:DriverLoadFlag,0         ;┐若设备驱动程序安装在常规内存中,则跳转
13239:      JE      CurrAvailMem_2              ;┘
13240:      MOV     AX,CS:MaxSeg_CurrDriver     ;AX←当前自由内存的最大段地址值
13241: CurrAvailMem_1:
13242:      CALL    ModifyMemSize               ;设定 BIOS 通讯区中保存的"系统常规内存大小"
                                                ;值
13243: CurrAvailMem_2:
13244:      RET
13245: CurrAvailableMem  ENDP
13246: ;=============================================================
13247: ;相对地址偏移:2AF9
13248: ;功能:重新设置 BIOS 通讯区中保存的"系统常规内存大小"值
```

```
13249: ;入口参数:AX=新的"系统常规内存大小"值
13250: ;出口参数:无
13251: ;=============================================================
13252: ModifyMemSize  PROC  NEAR
13253:     PUSH    DS
13254:     MOV     BX,40H                  ;┐
13255:     MOV     DS,BX                   ;| BX←系统的常规内存大小(实际KB数-1)
13256:     MOV     BX,WORD PTR DS:Info_13H ;┘
13257:     MOV     CS:SizeOfCMA,BX         ;保存系统实际的常规内存大小
13258:     MOV     CL,6                    ;┐
13259:     SHR     AX,CL                   ;| 更改BIOS通讯区中保存的"系统常规内存大
13260:     MOV     WORD PTR DS:Info_13H,AX ;┘ 小"
13261:     MOV     CS:FlagOfMemory,0FFH    ;设置BIOS通讯区中保存的"系统常规内存大小"
                                           ;被修改的标志
13262:     POP     DS
13263:     RET
13264: ModifyMemSize  ENDP
13265: ;=============================================================
13266: ;相对地址偏移:2B17
13267: ;功能:恢复系统的常规内存大小值
13268: ;入口参数:无
13269: ;出口参数:无
13270: ;=============================================================
13271: RecoverMemSize  PROC  NEAR
13272:     CMP     CS:FlagOfMemory,0       ;BIOS通讯区中保存的"系统常规内存大小"值被修
                                           ;改?
13273:     MOV     CS:FlagOfMemory,0       ;清除"BIOS通讯区中保存的常规内存大小"值被修
                                           ;改的标志
13274:     JZ      RecoverMemSize_RET      ;若未被修改,则跳转
13275:     PUSH    DS                      ;┐
13276:     MOV     AX,40H                  ;|
13277:     MOV     DS,AX                   ;| 恢复系统实际配置的"常规内存大小"值
13278:     MOV     AX,CS:SizeOfCMA         ;|
13279:     MOV     WORD PTR DS:Info_13H,AX ;|
13280:     POP     DS                      ;┘
13281: RecoverMemSize_RET:
13282:     RET
13283: RecoverMemSize  ENDP
13284: ;=============================================================
13285: ;相对地址偏移:2B34
13286: ;功能:检测正在安装的设备驱动程序的设备名是否为"PROTMAN$"
13287: ;入口参数:无
13288: ;出口参数:ZF=1,是"PROTMAN$";ZF=0,不是"PROTMAN$"
```

```
13289: ;=========================================  ==========
13290: Chk_PROTMIN¥  PROC  NEAR
13291:     PUSH    DS
13292:     PUSH    ES
13293:     PUSH    SI
13294:     LDS     SI,CS:DriverLoadAddr          ;┐ DS:DI 指向设备名
13295:     ADD     SI,0AH                        ;┘
13296:     PUSH    CS                            ;┐
13297:     POP     ES                            ;|
13298:     MOV     DI,offset DriverName          ;| 判定设备名是否是"PROTMAN$"
13299:     MOV     CX,8                          ;|
13300:     REPE    CMPSB                         ;┘
13301:     POP     SI
13302:     POP     ES
13303:     POP     DS
13304:     RET
13305: Chk_PROTMIN¥  ENDP
13306: ;=============================================================
13307: ;相对地址偏移:2B4D
13308: ;功能:设置当前活动进程可使用的最大内存的段地址值
13309: ;入口参数:无
13310: ;出口参数:无
13311: ;=============================================================
13312: SetAvailableMemSeg  PROC  NEAR
13313:     PUSH    DS
13314:     MOV     AH,62H                        ;┐ 取出当前活动进程的 PSP 段地址值→BX
13315:     INT     21H                           ;┘
13316:     MOV     DS,BX                         ;┐ 设置当前活动进程可使用的最大内存的段地址
13317:     MOV     BX,CS:Seg_CDSArray            ;| 值
13318:     MOV     WORD PTR DS:[2],BX            ;┘
13319:     POP     DS
13320:     RET
13321: SetAvailableMemSeg  ENDP

13322: ;=============================================================
13323: ;相对地址偏移:2B5F
13324: ;功能:建立设备驱动程序的子段控制块,确定设备驱动程序的装入地址和允许设备驱动程序使用
           的内存空间大小
13325: ;入口参数:无
13326: ;出口参数:无
13327: ;=============================================================
13328: CreateDriverSCB1  PROC  NEAR
13329:     CMP     CS:DriverLoadFlag,0           ;┐ 若设备驱动程序是安装在常规内存中,则跳转
```

```
13330:    JE      CreateDriverSCB1_2                ;┘
13331:    CALL    TestLoadDriverInUMB               ;判定在当前 UMB 内存块中是否能安装设备驱动
                                                    ;程序
13332:    JNC     CreateDriverSCB1_1                ;若有足够的内存空间,可以膊装,则跳转
13333:    CALL    ModifyUMBSize                     ;修改当前 UMB 内存块的大小
13334:    CALL    ApplyMaxUMB                       ;申请一个最大的 UMB 内存块,以便安装设备驱动
                                                    ;程序
13335:    JC      CreateDriverSCB1_2                ;若没有可用的 UMB 内存,则转去在常规内存中安
                                                    ;装设备驱动程序
13336: CreateDriverSCB1_1:
13337:    MOV     AX,CS:Seg_FreeUMB                 ;AX←设备驱动程序的子段控制块的段地址值
13338:    MOV     DX,CS:UMB_StartSeg                ;┐ DX←允许设备驱动程序使用的最大 UMB 段地
13339:    ADD     DX,CS:UMB_MB_Size                 ;┘ 址值
13340:    JMP     SHORT CreateDriverSCB1_3
13341: CreateDriverSCB1_2:
13342:    MOV     CS:DriverLoadFlag,0               ;清设备驱动程序的安装标志(0:安装在常规内存
                                                    ;中)
13343:    MOV     AX,CS:SEG_FreeMemory              ;AX←设备驱动程序的子段控制块的段地址值
13344:    MOV     DX,CS:Seg_CDSArray                ;DX←允许使用的常规内存顶端的段地址值
13345: CreateDriverSCB1_3:
13346:    CALL    CreateDriverSCB2                  ;建立设备驱动程序的子段控制块
13347:    MOV     CS:LoadSeg_CurrDriver,AX          ;保存设备驱动程序的装入段地址值
13348:    MOV     CS:MaxSeg_CurrDriver,DX           ;保存当前允许设备驱动程序使用的内存最大段地
                                                    ;址值
13349:    MOV     WORD PTR CS:DriverLoadAddr,0      ;┐ 设置设备驱动程序的装入地址值
13350:    MOV     WORD PTR CS:DriverLoadAddr+2,AX   ;┘
13351:    RET
13352: CreateDriverSCB1  ENDP
13353: ;=============================================================
13354: ;相对地址偏移:2BAB
13355: ;功能:判定在当前 UMB 内存块中是否能安装设备驱动程序
13356: ;入口参数:无
13357: ;出口参数:CF=0,可以膊装;CF=1,没有足够的内存空间
13358: ;=============================================================
13359: TestLoadDriverInUMB  PROC  NEAR
13360:    MOV     AX,CS:UMB_MB_Size                 ;┐
13361:    ADD     AX,CS:UMB_StartSeg                ;├ 计算当前 UMB 内存块中的可用内存节数→AX
13362:    SUB     AX,CS:Seg_FreeUMB                 ;┘
13363:    OR      AX,AX                             ;┐ 若可用内存节数≠0,则跳转
13364:    JNZ     TestLoadDriverInUMB_1             ;┘
13365:    STC
13366:    RET
13367: TestLoadDriverInUMB_1:
```

```
13368:    DEC     AX                              ;AX=可供设备驱动程序使用的 UMB 内存空间节
                                                  ;数
13369:    CMP     AX,CS:CurrDriverParagraphs      ;判定是否有足够的 UMB 内存空间装入设备驱动
                                                  ;程序
13370:    RET
13371: TestLoadDriverInUMB  ENDP
13372: ;=============================================================
13373: ;相对地址偏移:2BC6
13374: ;功能:申请一个最大的 UMB 内存块,它作为 DOS 的数据段,被用来存放设备驱动程序
13375: ;入口参数:无
13376: ;出口参数:CF=0,有一个满足要求的 UMB 内存块;CF=1,没有符合要求的 UMB 内存块
13377: ;=============================================================
13378: ApplyMaxUMB  PROC  NEAR
13379:    MOV     BX,0FFFFH                       ;┐
13380:    MOV     AX,4800H                        ;│测试最大的自由 UMB 内存块的大小
13381:    INT     21H                             ;┘
13382:    OR      BX,BX                           ;┐若 UMB 内存块全部用完,则跳转
13383:    JZ      ApplyMaxUMB_1                   ;┘
13384:    DEC     BX                              ;┐
13385:    CMP     CS:CurrDriverParagraphs,BX      ;│若最大的自由 UMB 内存块不能满足设备驱动
13386:    JA      ApplyMaxUMB_1                   ;┘程序的需要,则跳转
13387:    INC     BX                              ;┐
13388:    MOV     AX,4800H                        ;│申请最大的自由 UMB 内存块
13389:    INT     21H                             ;┘
13390:    JC      ApplyMaxUMB_1
13391:    PUSH    DS                              ;┐
13392:    DEC     AX                              ;│
13393:    MOV     DS,AX                           ;│ PID 值(8:DOS 内核使用)→UMB 内存控制块
13394:    MOV     WORD PTR DS:MCB_ProcessID,8     ;│文件名("SD":表示系统数据段)→UMB 内存控
13395:    MOV     WORD PTR DS:[8],' DS'           ;│制块
13396:    INC     AX                              ;│
13397:    POP     DS                              ;┘
13398:    MOV     CS:UMB_MB_Size,BX               ;设置 UMB 内存块的大小(节数)
13399:    MOV     CS:UMB_StartSeg,AX              ;设置 UMB 内存块的起始段地址值
13400:    MOV     CS:Seg_FreeUMB,AX               ;设置设备驱动程序的装入段地址值
13401:    CLC
13402:    RET
13403: ApplyMaxUMB_1:
13404:    XOR     AX,AX                           ;┐
13405:    MOV     CS:UMB_MB_Size,AX               ;│清与 UMB 内存块相关的控制变量
13406:    MOV     CS:UMB_StartSeg,AX              ;│
13407:    MOV     CS:Seg_FreeUMB,AX               ;┘
13408:    STC
```

```
13409:      RET
13410: ApplyMaxUMB  ENDP
13411: ;=============================================================
13412: ;相对地址偏移:2C13
13413: ;功能:建立设备驱动程序对应的子段控制块
13414: ;入口参数:AX=设备驱动程序的子段控制块的段地址值
13415: ;出口参数:AX=设备驱动程序的装入段地址值
13416: ;=============================================================
13417: CreateDriverSCB2  PROC  NEAR
13418:      PUSH    ES
13419:      PUSH    DI
13420:      PUSH    DS
13421:      PUSH    SI
13422:      MOV     ES,AX                          ;ES←设备驱动程序的子段控制块的段地址值
13423:      MOV     BYTE PTR ES:[0],' D'           ;设备驱动程序的标志符→子段控制块
13424:      INC     AX                             ;┐子段的段地址值→子段控制块
13425:      MOV     WORD PTR ES:[1],AX             ;┘
13426:      PUSH    AX
13427:      LDS     SI,DWORD PTR CS:RequestForInit+12H  ;DS:SI 指向设备驱动程序的文件名说明串
13428:      MOV     DI,SI                          ;DI←设备驱动程序的文件名┐
                                                   ;说明串的地址偏移       |
13429:      CLD                                    ;                       |
13430: CreateDriverSCB2_1:                         ;AL←文件名说明串的当前字|
                                                   ;符                     |
13431:      LODSB                                  ;                       |
13432:      CMP     AL,' :'                        ;┐                      |
13433:      JNE     CreateDriverSCB2_2             ;| 跳过逻辑驱动器名      |
13434:      MOV     DI,SI                          ;|                      |
13435:      JMP     SHORT CreateDriverSCB2_1       ;┘                      | DS:SI 指向设备
13436: CreateDriverSCB2_2:                         ;                       | 驱动程序对应
13437:      CMP     AL,'\'                         ;┐                      | 的文件名
13438:      JNE     CreateDriverSCB2_3             ;| 跳过路径名说明串      |
13439:      MOV     DI,SI                          ;|                      |
13440:      JMP     SHORT CreateDriverSCB2_1       ;┘                      |
13441: CreateDriverSCB2_3:                         ;                       |
13442:      OR      AL,AL                          ;                       |
13443:      JNZ     CreateDriverSCB2_1             ;                       |
13444:      MOV     SI,DI                          ;                       ┘
13445:      MOV     DI,8                           ;ES:DI 指向子段控制块的文件名域
13446:      MOV     CX,8                           ;CX←文件名长度
13447: CreateDriverSCB2_4:                         ;┐
13448:      LODSB                                  ;|
13449:      OR      AL,AL                          ;|
```

```
13450:    JZ      CreateDriverSCB2_5              ;|
13451:    CMP     AL,'.'                          ;|
13452:    JE      CreateDriverSCB2_5              ;| 将文件名复制到子段控制块中,不足 8 个字
13453:    STOSB                                   ;| 符的文件名以空格补齐
13454:    LOOP    CreateDriverSCB2_4              ;|
13455: CreateDriverSCB2_5:                        ;|
13456:    JCXZ    CreateDriverSCB2_6              ;|
13457:    MOV     AL,' '                          ;|
13458:    REP     STOSB                           ;|
13459: CreateDriverSCB2_6:                        ;┘
13460:    POP     AX
13461:    POP     SI
13462:    POP     DS
13463:    POP     DI
13464:    POP     ES
13465:    RET
13466: CreateDriverSCB2  ENDP
13467: ;=============================================================
13468: ;相对地址偏移:2C62
13469: ;功能:确定当前正在安装的设备驱动程序所需的内存空间大小
13470: ;入口参数:ES:SI 指向正在安装的设备驱动程序的文件名说明串
13471: ;出口参数:CF=0,成功;CF=1,失败,设备驱动程序不存在
13472: ;=============================================================
13473: DetermineDriverParagraphs  PROC  NEAR
13474:    PUSH    ES                              ;┐
13475:    POP     DS                              ;| 以"读"方式打开当前欲装入的设备驱动程序的
13476:    MOV     DX,SI                           ;| 文件
13477:    MOV     AX,3D00H                        ;|
13478:    INT     21H                             ;┘
13479:    JC      DetermiDriverParas_RET          ;若打开文件失败,则直接返回
13480:    MOV     BX,AX                           ;┐
13481:    MOV     AX,4202H                        ;|
13482:    XOR     CX,CX                           ;| 测试文件大小
13483:    MOV     DX,CX                           ;|
13484:    INT     21H                             ;┘
13485:    JC      DetermiDriverParas_3            ;若测试文件大小失败,则跳转
13486:    ADD     AX,0FH                          ;┐ DX:AX←当前欲装入的设备驱动程序所需的
13487:    ADC     DX,0                            ;┘ 内存空间(字节数)
13488:    TEST    DX,0FFF0H                       ;┐ 若所需内存<64KB,则跳转
13489:    JZ      DetermiDriverParas_1            ;┘
13490:    MOV     CS:CurrDriverParagraphs,0FFFFH  ;设置当前欲装入的设备驱动程序所需的内存节数
13491:    JMP     SHORT DetermiDriverParas_3
13492: DetermiDriverParas_1:
```

```
13493:    MOV     CL,4                          ;┐
13494:    SHR     AX,CL                         ;| AX←当前欲装入的设备驱动程序所需的内存
13495:    MOV     CL,0CH                        ;| 节数
13496:    SHL     DX,CL                         ;|
13497:    OR      AX,DX                         ;┘
13498:    CMP     AX,CS:Hexsize                 ;┐
13499:    JA      DetermiDriverParas_2          ;| 设置当前欲装入的设备驱动程序所需的内存节
13500:    MOV     AX,CS:Hexsize                 ;| 数(当通过"devicehigh size="的系统配置命令装
13501: DetermiDriverParas_2:                   ;| 入设备驱动程序时,它确保了能留出不小于
13502:    MOV     CS:CurrDriverParagraphs,AX    ;┘ hexsize 值的内存空间)
13503:    CLC
13504: DetermiDriverParas_3:
13505:    PUSHF                                 ;┐
13506:    MOV     AX,3E00H                      ;| 关闭"当前欲装入的设备驱动程序"对应的文件
13507:    INT     21H                           ;|
13508:    POPF                                  ;┘
13509: DetermiDriverParas_RET:
13510:    RET
13511: DetermineDriverParagraphs  ENDP
13512: ;=============================================================
13513: ;相对地址偏移:2CB1
13514: ;功能:以"覆盖方式"装入设备驱动程序
13515: ;入口参数:DS:DX 指向设备驱动程序对应的文件名说明串
13516: ;出口参数:CF=0,装入成功;CF=1,装入失败,AL=错误码
13517: ;=============================================================
13518: LoadDriver  PROC  NEAR
13519:    MOV     BX,CS:LoadSeg_CurrDriver      ;BX←设备驱动程序的装入段地址值
13520:    MOV     CS:OverlayPPB,BX              ;设置装入的段地址
13521:    MOV     WORD PTR CS:OverlayPPB+2,BX   ;设置重定位因子
13522:    MOV     BX,CS                         ;┐
13523:    MOV     ES,BX                         ;|
13524:    MOV     BX,offset OverlayPPB          ;| 以"覆盖方式"装入设备驱动程序
13525:    MOV     AL,3                          ;|
13526:    MOV     AH,4BH                        ;|
13527:    INT     21H                           ;┘
13528:    RET
13529: LoadDriver  ENDP
13530: ;=============================================================
13531: ;相对地址偏移:2CCE
13532: ;功能:变换文件名说明串的结束符,从而使得文件名说明串与其后的参数串用参数分隔符(空格)
            相连接
13533: ;入口参数:ES:SI 指向"device="之后的字符串
13534: ;出口参数:ES:SI 指向参数串
```

```
13535: ;=========================================================
13536: ChangedTerminator  PROC  NEAR
13537: ChangeTerminator_1:                    ;┐
13538:     MOV     BL,ES:[SI]                 ;│
13539:     OR      BL,BL                      ;│
13540:     JZ      ChangeTerminator_2         ;├ 搜索文件名说明串的结束符
13541:     INC     SI                         ;│
13542:     JMP     SHORT ChangeTerminator_1   ;│
13543: ChangeTerminator_2:                    ;┘
13544:     MOV     BL,CS:Terminator           ;┐ 更换文件名说明串的结束符为参数分隔符
13545:     MOV     ES:[SI],BL                 ;┘
13546:     RET
13547: ChangedTerminator  ENDP
13548: ;=========================================================
13549: ;相对地址偏移:2CE1
13550: ;功能:转换自由内存的起始地址值格式,并进一步地检查内存状态
13551: ;入口参数:无
13552: ;出口参数:无
13553: ;=========================================================
13554: ConvertAddrFMT  PROC  NEAR
13555:     MOV     AX,WORD PTR CS:EndAddrOfDriver    ;┐
13556:     CALL    Convert_Addr                      ;├ 转换自由内存的起始地址
13557:     ADD     WORD PTR CS:EndAddrOfDriver+2,AX  ;│
13558:     MOV     WORD PTR CS:EndAddrOfDriver,0     ;┘
13559:     MOV     AX,CS:MaxSeg_CurrDriver           ;┐
13560:     CMP     WORD PTR CS:EndAddrOfDriver+2,AX  ;├ 若有足够的自由内存,则直接返回
13561:     JBE     ConvertAddrFMT_RET                ;┘
13562:     JMP     MemoryError                       ;转去显示错误信息"Configuration too
                                                    ;large for memory",并进入死循环
13563: ConvertAddrFMT_RET:
13564:     RET
13565: ConvertAddrFMT  ENDP
13566: ;=========================================================
13567: ;相对地址偏移:2D03
13568: ;功能:检查设备驱动程序的安装状态,并转换自由内存的起始地址值格式
13569: ;入口参数:无
13570: ;出口参数:CF=0,有足够的内存;CF=1,内存不够
13571: ;=========================================================
13572: ChkDriver  PROC  NEAR
13573:     PUSH    AX
13574:     MOV     AX,WORD PTR CS:EndAddrOfDriver+2  ;┐
13575:     CMP     CS:DriverCounter,0                ;├
13576:     JNE     ChkDriver_1                       ;┘
```

```
13577:     CMP     AX,CS:LoadSeg_CurrDriver                ;| 若正在安装的设备驱动程序占用的内存
13578:     JNE     ChkDriver_1                             ;| 量为 0,则跳转
13579:     CMP     WORD PTR CS:EndAddrOfDriver,0           ;|
13580:     JE      VhkDriver_2                             ;|
13581: ChkDriver_1:                                        ;┘
13582:     CALL    ConvertAddrFMT                          ;转换自由内存的起始地址
13583:     POP     AX
13584:     CLC
13585:     RET
13586: VhkDriver_2:
13587:     POP     AX
13588:     STC
13589:     RET
13590: ChkDriver  ENDP
13591: ;=============================================
13592: ;相对地址偏移:2D28
13593: ;功能:设置设备驱动程序占用的子段内存块大小,同时还设置可供下次使用的自由内存区的起始
            地址
13594: ;入口参数:无
13595: ;出口参数:无
13596: ;=============================================
13597: SetSizeOfSCB  PROC  NEAR
13598:     PUSH    DS
13599:     MOV     AX,CS:LoadSeg_CurrDriver                ;┐
13600:     MOV     BX,WORD PTR CS:EndAddrOfDriver+2        ;|
13601:     DEC     AX                                      ;|
13602:     MOV     DS,AX                                   ;| 子段内存块大小→子段控制块
13603:     INC     AX                                      ;|
13604:     SUB     AX,BX                                   ;|
13605:     NEG     AX                                      ;|
13606:     MOV     WORD PTR DS:[3],AX                      ;┘
13607:     CMP     CS:DriverLoadFlag,0             ;┐ 若设备驱动程序是安装在常规内存中,则跳转
13608:     JE      SetSizeOfSCB_1                  ;┘
13609:     MOV     CS:Seg_FreeUMB,BX               ;设置 UMB 内存块的自由内存区的起始段地址值
13610:     JMP     SHORT SetSizeOfSCB_2
13611: SetSizeOfSCB_1:
13612:     MOV     CS:SEG_FreeMemory,BX            ;┐ 设置常规内存的自由内存区的起始地址
13613:     MOV     CS:OFS_FreeMemory,0             ;┘
13614: SetSizeOfSCB_2:
13615:     POP     DS
13616:     RET
13617: SetSizeOfSCB  ENDP
13618: ;=============================================
```

```
13619: ;相对地址偏移:2D5A
13620: ;功能:处理"devicehigh size"系统配置命令的 hexsize 参数
13621: ;入口参数:ES:SI 指向系统配置命令行的待处理字符串
13622: ;出口参数:ES:SI 指向新的系统配置命令行的待处理字符串
13623: ;          CF=0,hexsize 参数格式正确;CF=1,hexsize 参数格式错
13624: ;=========================================================
13625: ProcessHexsizePara  PROC  NEAR
13626:     MOV    CS:Hexsize,0                    ;清 hexsize 参数值
13627:     MOV    WORD PTR CS:Ptr_DriverNameStr,SI    ;┐保存正在安装的设备驱动程序的文件名
13628:     MOV    WORD PTR CS:Ptr_DriverNameStr+2,ES ;┘说明串指针
13629:     CALL   OmitAllSeparator                ;跳过参数分隔符
13630:     CMP    WORD PTR ES:[SI],' IS'          ;┐
13631:     JNE    ProcessHexsizeP_1               ;|
13632:     CMP    WORD PTR ES:[SI+2],' EZ'        ;|
13633:     JNE    ProcessHexsizeP_1               ;|若不是"devicehigh size"系统配置命令,则结束返
13634:     MOV    AL,ES:[SI+4]                    ;|回
13635:     CALL   TestTerminatorOrSeparator       ;|
13636:     JNZ    ProcessHexsizeP_1               ;┘
13637:     ADD    SI,5                            ;跳过"size="关键字
13638:     CALL   GetHexsizePara                  ;取出"devicehigh size"系统配置命令的 hexsize 参数
                                                  值
13639:     JC     ProcessHexsizeP_2               ;若 hexsize 参数格式错,则跳转
13640:     MOV    CS:Hexsize,AX                   ;保存 hexsize 参数值
13641:     CALL   OmitAllSeparator                ;跳过 hexsize 参数后的所有参数分隔符,使得 ES:SI
                                                  ;指向设备驱动程序的文件名说明串
13642: ProcessHexsizeP_1:
13643:     CLC
13644:     RET
13645: ProcessHexsizeP_2:
13646:     STC
13647:     RET
13648: ProcessHexsizePara  ENDP
13649: ;=========================================================
13650: ;相对地址偏移:2D99
13651: ;功能:跳过所有的参数分隔符
13652: ;入口参数:ES:SI 指向系统配置命令行的待处理字符串
13653: ;出口参数:ES:SI 指向跳过所有参数分隔符后的待处理字符串
13654: ;=========================================================
13655: OmitAllSeparator  PROC  NEAR
13656: OmitAllSeparator_1:
13657:     MOV    AL,ES:[SI]                      ;AL←当前字符
13658:     CALL   TestTerminatorOrSeparator       ;┐若当前字符不是参数分隔符,则直接返回
13659:     JNZ    OmitAllSeparator_RET            ;┘
```

```
13660:    INC     SI                              ;跳过当前参数分隔符
13661:    JMP     SHORT OmitAllSeparator_1
13662: OmitAllSeparator_RET:
13663:    RET
13664: OmitAllSeparator  ENDP
13665: ;=============================================================
13666: ;相对地址偏移:2DA5
13667: ;功能:取出"devicehigh size"系统配置命令的 hexsize 参数值
13668: ;入口参数:ES:SI 指向系统配置命令行的待处理字符串
13669: ;出口参数:CF=0,DX:AX=以节为单位的 hexsize 参数值;CF=1,hexsize 参数格式错
13670: ;=============================================================
13671: GetHexsizePara  PROC  NEAR
13672:    XOR     AX,AX                           ;┐ 清"可用的最小内存量"(hexsize)参数值
13673:    XOR     DX,DX                           ;┘
13674: GetHexsizeP_1:
13675:    MOV     BL,ES:[SI]                      ;BL←系统配置命令行的当前字符
13676:    CMP     BL,0DH                          ;┐
13677:    JE      GetHexsizeP_5                   ;├ 若系统配置命令行已结束,则结束返回
13678:    CMP     BL,0AH                          ;│
13679:    JE      GetHexsizeP_5                   ;┘
13680:    PUSH    AX                              ;┐
13681:    MOV     AL,BL                           ;│
13682:    CALL    TestTerminatorOrSeparator       ;├ 若"hexsize"参数已结束,则跳转
13683:    POP     AX                              ;│
13684:    JZ      GetHexsizeP_3                   ;┘
13685:    CALL    CalculateHexValue               ;计算 BL 指定的数字符值
13686:    JC      GetHexsizeP_5                   ;若 BL 指定的字符不是十六进制数字符,则跳转
13687:    MOV     CX,4                            ;┐
13688: GetHexsizeP_2:                             ;│
13689:    SHL     AX,1                            ;├ 累加当前的十六进制数字值
13690:    RCL     DX,1                            ;│
13691:    LOOP    GetHexsizeP_2                   ;│
13692:    OR      AL,BL                           ;┘
13693:    INC     SI                              ;ES:SI 指向系统配置命令行的下一个字符
13694:    JMP     SHORT GetHexsizeP_1
13695: GetHexsizeP_3:
13696:    ADD     AX,0FH                          ;┐
13697:    ADC     DX,0                            ;├ 若"hexsize"参数值超过 1MB,则跳转
13698:    TEST    DX,0FFF0H                       ;│
13699:    JNZ     GetHexsizeP_5                   ;┘
13700:    MOV     CX,4                            ;┐
13701: GetHexsizeP_4:                             ;│
13702:    CLC                                     ;├ 将"hexsize"参数值化成节单位→DX:AX
```

```
13703:    RCR     DX,1                          ;|
13704:    RCR     AX,1                          ;|
13705:    LOOP    GetHexsizeP_4                 ;┘
13706:    CLC
13707:    RET
13708: GetHexsizeP_5:
13709:    STC
13710:    RET
13711: GetHexsizePara  ENDP
13712: ;=============================================================
13713: ;相对地址偏移:2DEC
13714: ;功能:计算一位十六进制数字值
13715: ;入口参数:BL=数字符
13716: ;出口参数:CF=0,BL=数字值;CF=1,BL 指的字符不是十六进制数字符
13717: ;=============================================================
13718: CalculateHexValue  PROC  NEAR
13719:    CMP     BL,' 0'
13720:    JB      CalcuHexValue_2
13721:    CMP     BL,' 9'
13722:    JA      CalcuHexValue_1
13723:    SUB     BL,' 0'
13724:    RET
13725: CalcuHexValue_1:
13726:    CMP     BL,' A'
13727:    JB      CalcuHexValue_2
13728:    CMP     BL,' F'
13729:    JA      CalcuHexValue_2
13730:    SUB     BL,' 7'
13731:    RET
13732: CalcuHexValue_2:
13733:    STC
13734:    RET
13735: CalculateHexValue  ENDP
13736: ;=============================================================
13737: ;相对地址偏移:2E0A
13738: ;功能:建立一个完整的 UMB 内存控制块链,并合并所有相邻的自由 UMB 内存块
13739: ;入口参数:无
13740: ;出口参数:无
13741: ;=============================================================
13742: CreateEntireUMBChain  PROC  NEAR
13743:    CALL    CreateUMBChain1               ;建立 UMB 的内存控制块链
13744:    JC      CreateEntireUMBCh_RET         ;若没有 UMB,则直接返回
13745: CreateEntireUMBCh_1:                     ;┐
```

```
13746:      CALL    AllocateMaxFreeUMB              ;┌ 建立一个完整的 UMB 内存控制块链,使得所有
13747:      JC      CreateEntireUMBCh_2             ;│ 的 UMB 都能被链接起来
13748:      CALL    JoinFreeUMBToUMBChain           ;│
13749:      JMP     SHORT CreateEntireUMBCh_1       ;┘
13750: CreateEntireUMBCh_2:
13751:      CALL    JoinAllNeighborFreeUMB          ;合并所有相邻的自由 UMB 内存块
13752: CreateEntireUMBCh_RET:
13753:      RET
13754: CreateEntireUMBChain  ENDP
13755: ;=============================================================
13756: ;相对地址偏移:2E1D
13757: ;功能:建立 UMB 的内存控制块链
13758: ;入口参数:无
13759: ;出口参数:CF=0,已建立 UMB 的内存控制块链;CF=1,没有 UMB
13760: ;=============================================================
13761: CreateUMBChain1  PROC  NEAR
13762:      CALL    GetInstallStateOfHIMEM          ;取 HIMEM.SYS 的安装状态
13763:      JNZ     CreateUMBCh1_2                  ;若未安装,则跳转
13764:      MOV     AH,52H                          ;┐ 取出多重表的地址→ES:BX
13765:      INT     21H                             ;┘
13766:      MOV     CS:Seg MSDOS1,ES                ;保存多重表的段地址值
13767:      MOV     AX,4310H                        ;┐ 取出 HIMEM.SYS 提供的扩充内存管理程序的
13768:      INT     2FH                             ;┘ 入口地址
13769:      MOV     WORD PTR CS:UMB_ManageProgramEntry,BX   ;┐ 保存 HIMEM.SYS 提供的扩
                                                            ;│ 充内存管理程序的入口地址
                                                            ;│ (EMM386.EXE 提供的程序
13770:      MOV     WORD PTR CS:UMB_ManageProgramEntry+2,ES;┘ 仍然是通过该接口)
13771:      CMP     CS:UMBChain_Flag,0              ;┐ 若已建立了 UMB 的内存控制块链,则跳转
13772:      JNE     CreateUMBCh1_1                  ;┘
13773:      CALL    CreateUMBChain2                 ;建立 UMB 内存控制块链
13774:      JC      CreateUMBCh1_2                  ;若没有 UMB,则跳转
13775:      MOV     CS:UMBChain_Flag,0FFH           ;设置 UMB 内存控制块链的建立标志(-1:已建立
                                                    ;UMB 内存控制块链)
13776: CreateUMBCh1_1:
13777:      CLC
13778:      RET
13779: CreateUMBCh1_2:
13780:      STC
13781:      RET
13782: CreateUMBChain1  ENDP
13783: ;=============================================================
13784: ;相对地址偏移:2E51
13785: ;功能:分配最大的自由 UMB
```

```
13786: ;入口参数:无
13787: ;出口参数:CF=0,成功,BX=UMB 的段地址值
13788: ;           DX=UMB 的大小(节单位)
13789: ;           CF=1,失败,BL=错误码
13790: ;=============================================================
13791: AllocateMaxFreeUMB  PROC  NEAR
13792:     PUSH    AX
13793:     MOV     AH,10H                                  ;┐
13794:     MOV     DX,0FFFFH                               ;| 测试 UMB 的大小
13795:     CALL    DWORD PTR CS:UMB_ManageProgramEntry     ;┘
13796:     OR      DX,DX                           ;┐ 若 UMB 全用完或者没有 UMB,则跳转
13797:     JZ      AllocateMaxFreeUMB_2            ;┘
13798:     MOV     AH,10H                                  ;┐ 申请(分配)最大可用的 UMB
13799:     CALL    DWORD PTR CS:UMB_ManageProgramEntry     ;┘
13800:     CMP     AX,1                            ;┐ 若分配失败,则跳转
13801:     JNE     AllocateMaxFreeUMB_2            ;┘
13802:     CLC
13803: AllocateMaxFreeUMB_1:
13804:     POP     AX
13805:     RET
13806: AllocateMaxFreeUMB_2:
13807:     STC
13808:     JMP     SHORT AllocateMaxFreeUMB_1
13809: AllocateMaxFreeUMB  ENDP
13810: ;=============================================================
13811: ;相对地址偏移:2E72
13812: ;功能:将一个自由的 UMB 内存块加入到 UMB 内存控制块链中
13813: ;入口参数:BX=自由 UMB 内存块的段地址值
13814: ;           DX=自由 UMB 内存块的大小(节单位)
13815: ;出口参数:无
13816: ;=============================================================
13817: JoinFreeUMBToUMBChain  PROC  NEAR
13818:     PUSH    DS
13819:     MOV     DS,WORD PTR CS:Seg_MSDOS1       ;┐
13820:     MOV     DS,DS:DOS_008C                  ;| DS(ES)=UMB 的接口内存控制块(或称为第一
13821:     MOV     AX,DS                           ;| 个 UMB 内存控制块)的段地址值
13822:     MOV     ES,AX                           ;┘
13823: JoinFreeUMBToCh_1:
13824:     CMP     AX,BX                           ;┐ 若当前的自由 UMB 内存块应插入到 UMB 内存
13825:     JA      JoinFreeUMBToCh_2               ;┘ 控制块链之中,则跳转
13826:     CMP     BYTE PTR ES:MCB_Location,' Z'   ;┐ 若当前的自由 UMB 内存块追加到 UMB 内存控
13827:     JE      JoinFreeUMBToCh_3               ;┘ 制块链尾,则跳转
13828:     MOV     DS,AX                           ;┐ 取出下一个 UMB 的内存控制块的段地址值→
```

```
13829:    CALL    GetNextUMBSeg                    ;┘ AX(ES)
13830:    JMP     SHORT JoinFreeUMBToCh_1
13831: JoinFreeUMBToCh_2:
13832:    MOV     CX,DS                            ;┐
13833:    INC     CX                               ;|
13834:    SUB     CX,BX                            ;|
13835:    NEG     CX                               ;|
13836:    MOV     DS:MCB_Location,' M'             ;|
13837:    MOV     DS:MCB_ProcessID,8               ;|
13838:    MOV     DS:MCB_AllocationAmount,CX       ;|
13839:    MOV     WORD PTR DS:[8],' CS'            ;|
13840:    MOV     ES,BX                            ;|
13841:    MOV     ES:MCB_Location,' M'             ;| 将当前的自由 UMB 内存块插入到 UMB 内存控
13842:    MOV     ES:MCB_ProcessID,0               ;| 制块链之中
13843:    SUB     DX,2                             ;|
13844:    MOV     ES:MCB_AllocationAmount,DX       ;|
13845:    ADD     BX,DX                            ;|
13846:    INC     BX                               ;|
13847:    MOV     ES,BX                            ;|
13848:    INC     BX                               ;|
13849:    SUB     AX,BX                            ;|
13850:    MOV     ES:MCB_Location,' M'             ;|
13851:    MOV     ES:MCB_ProcessID,8               ;|
13852:    MOV     ES:MCB_AllocationAmount,AX       ;|
13853:    MOV     WORD PTR ES:[8],' CS'            ;|
13854:    JMP     SHORT JoinFreeUMBToCh_4          ;┘
13855: JoinFreeUMBToCh_3:
13856:    ADD     AX,ES:MCB_AllocationAmount       ;┐
13857:    SUB     ES:MCB_AllocationAmount,1        ;|
13858:    MOV     ES:MCB_Location,' M'             ;|
13859:    MOV     CX,AX                            ;|
13860:    INC     AX                               ;| 建立一个虚拟的 UMB 内存控制块,使得当前的
13861:    SUB     AX,BX                            ;| 自由 UMB 内存控制块能追加到 UMB 内存控制
13862:    NEG     AX                               ;| 块链尾
13863:    MOV     ES,CX                            ;|
13864:    MOV     ES:MCB_Location,' M'             ;|
13865:    MOV     ES:MCB_ProcessID,8               ;|
13866:    MOV     ES:MCB_AllocationAmount,AX       ;|
13867:    MOV     WORD PTR ES:[8],' CS'            ;┘
13868:    MOV     ES,BX                            ;┐
13869:    MOV     ES:MCB_Location,' Z'             ;|
13870:    MOV     ES:MCB_ProcessID,0               ;| 建立 UMB 内存控制块链的一个尾结点
13871:    DEC     DX                               ;|
```

```
13872:    MOV     ES:MCB_AllocationAmount,DX      ;┘
13873: JoinFreeUMBToCh_4:
13874:    POP     DS
13875:    RET
13876: JoinFreeUMBToUMBChain  ENDP
13877: ;=============================================================
13878: ;相对地址偏移:2F31
13879: ;功能:合并所有相邻的自由 UMB 内存块,使得自由 UMB 内存块的尺寸尽可能地大
13880: ;入口参数:无
13881: ;出口参数:无
13882: ;=============================================================
13883: JoinAllNeighborFreeUMB  PROC  NEAR
13884:    XOR     DI,DI                           ;┐
13885:    MOV     ES,WORD PTR CS:Seg_MSDOS1       ;│ ES:DI 指向 UMB 内存控制块链的接口内存控
13886:    MOV     ES,ES:DOS_008C                  ;┘ 制块
13887: JoinAllNeighor_1:                          ;┐
13888:    MOV     AX,ES                           ;│
13889:    MOV     DS,AX                           ;│
13890:    CMP     WORD PTR ES:MCB_ProcessID,DI    ;│ 寻找自由的 UMB 内存块
13891:    JE      JoinAllNeighor_2                ;│
13892:    CALL    GetNextUMBSeg                   ;│
13893:    JC      JoinAllNeighor_RET              ;│
13894:    JMP     SHORT JoinAllNeighor_1          ;┘
13895: JoinAllNeighor_2:
13896:    CALL    GetNextUMBSeg                   ;取出下一个 UMB 的内存控制块的段地址值→ES
                                                  ;(AX)
13897:    JC      JoinAllNeighor_RET              ;若 UMB 内存控制块链已搜索完,则结束返回
13898:    CMP     WORD PTR ES:MCB_ProcessID,DI    ;┐ 若 ES 指定的 UMB 内存控制块已被分配,则跳
13899:    JNE     JoinAllNeighor_1                ;┘ 转
13900:    MOV     CX,ES:MCB_AllocationAmount          ;┐
13901:    INC     CX                                  ;│
13902:    ADD     WORD PTR DS:MCB_AllocationAmount,CX ;│ 合并两个相邻的自由 UMB 内存块
13903:    MOV     CL,ES:[DI]                          ;│
13904:    MOV     [DI],CL                             ;┘
13905:    JMP     SHORT JoinAllNeighor_2
13906: JoinAllNeighor_RET:
13907:    RET
13908: JoinAllNeighborFreeUMB  ENDP
13909: ;=============================================================
13910: ;相对地址偏移:2F6D
13911: ;功能:取出下一个 UMB 的内存控制块的段地址值
13912: ;入口参数:DS=当前 UMB 的内存控制块的段地址值
13913: ;出口参数:CF=0,成功,AX=下一个 UMB 的内存控制块的段地址值
```

```
13914: ;          ES=下一个 UMB 的内存控制块的段地址值
13915: ;          CF=1,DS=最后一个 UMB 的内存控制块的段地址值
13916: ;==========================================================
13917: GetNextUMBSeg  PROC  NEAR
13918:     CMP     BYTE PTR DS:MCB_Location,' Z'
13919:     JE      GetNextUMBSeg_1
13920:     MOV     AX,DS
13921:     ADD     AX,DS:MCB_AllocationAmount
13922:     INC     AX
13923:     MOV     ES,AX
13924:     CLC
13925:     RET
13926: GetNextUMBSeg_1:
13927:     STC
13928:     RET
13929: GetNextUMBSeg  ENDP
13930: ;==========================================================
13931: ;相对地址偏移:2F81
13932: ;功能:建立 UMB 的内存控制块链,并设置 MSDOS1 模块中的变量值
13933: ;入口参数:无
13934: ;出口参数:CF=0,成功;CF=1,失败
13935: ;==========================================================
13936: CreateUMBChain2  PROC  NEAR
13937:     CALL    AllocateMaxFreeUMB          ;分配最大的 UMB。当分配成功时,BX=UMB 的起
                                                ;始段地址值,DX=以节为单位的 UMB 大小
13938:     JC      CreateUMBCh2_3              ;若没有 UMB,则结束返回
13939:     INT     12H                         ;取出系统中的常规内存大小(以 KB 为单位)
13940:     MOV     CL,6                        ;┐
13941:     SHL     AX,CL                       ;| CX=以节为单位的常规内存大小
13942:     MOV     CX,AX                       ;┘
13943:     SUB     AX,BX                       ;┐ AX=UMB 的起始段地址值-以节为单位的常
13944:     NEG     AX                          ;┘ 规内存大小
13945:     SUB     CX,1                        ;┐
13946:     MOV     ES,CX                       ;|
13947:     MOV     ES:MCB_Location,' M'        ;| 在常规内存最高端建立一个 DOS 内核使用的内
13948:     MOV     ES:MCB_ProcessID,8          ;| 存控制块,它是为了访问 UMB 而虚设的
13949:     MOV     ES:MCB_AllocationAmount,AX  ;|
13950:     MOV     WORD PTR ES:[8],' CS'       ;┘
13951:     MOV     ES,BX                       ;┐
13952:     MOV     ES:MCB_Location,' Z'        ;| 将申请到的 UMB 作为一个自由内存块,它的位
13953:     MOV     ES:MCB_ProcessID,0          ;| 置标志为"Z",表明是最后一个内存块
13954:     DEC     DX                          ;|
13955:     MOV     ES:MCB_AllocationAmount,DX  ;┘
```

```
13956:    MOV     ES,WORD PTR CS:Seg_MSDOS1   ;┐ 设置 MSDOS1 模块的变量(UMB 内存控制块链
13957:    MOV     DI,DOS_008C                 ;│ 的头结点指针,即保存 UMB 的第一个内存控制
                                              ;│ 块的段地址值,该内存控制块实际建立在常规
13958:    MOV     ES:[DI],CX                  ;┘ 内存的最高端)
13959:    MOV     DI,DOS_0024                 ;┐
13960:    MOV     ES,ES:[DI]                  ;│ ES:DI 指向常规内存的第一个内存控制块
13961:    XOR     DI,DI                       ;┘
13962: CreateUMBCh2_1:                        ;┐
13963:    CMP     BYTE PTR ES:[DI],' Z'       ;│
13964:    JE      CreateUMBCh2_2              ;│
13965:    MOV     AX,ES                       ;│ 搜索常规内存的最后一个内存控制块
13966:    ADD     AX,ES:MCB_AllocationAmount  ;│
13967:    INC     AX                          ;│
13968:    MOV     ES,AX                       ;│
13969:    JMP     SHORT CreateUMBCh2_1        ;┘
13970: CreateUMBCh2_2:                        ;ES=常规内存的最后一个内存控制块的段地址值
13971:    SUB     ES:MCB_AllocationAmount,1   ;修改内存块的大小(因为在常规内存的最高端建
                                              ;立了一个 UMB 的接口内存控制块)
13972:    MOV     ES:MCB_Location,' M'        ;更改内存控制块的位置标志
13973:    CLC
13974:    RET
13975: CreateUMBCh2_3:
13976:    STC
13977:    RET
13978: CreateUMBChain2   ENDP
13979: ;=============================================================
13980: ;相对地址偏移:2FF9
13981: ;功能:修改 UMB 内存块的大小。该 UMB 内存块是作为 DOS 内核的数据段,它保存了设备驱动程序
          的驻留部分和相关的数据块(如块设备的 DDPB)
13982: ;入口参数:无
13983: ;出口参数:无
13984: ;=============================================================
13985: ModifyUMBSize   PROC   NEAR
13986:    CMP     CS:UMB_StartSeg,0           ;┐ 若未申请过 UMB 内存块,则跳转
13987:    JE      ModifyUMBSize_RET           ;┘
13988:    PUSH    ES
13989:    PUSH    BX
13990:    MOV     BX,CS:Seg_FreeUMB           ;┐ BX←新的 UMB 内存块的大小(节数)
13991:    SUB     BX,CS:UMB_StartSeg          ;┘
13992:    MOV     ES,CS:UMB_StartSeg          ;ES=UMB 内存块的起始段地址值
13993:    MOV     AX,4A00H                    ;┐ 修改 UMB 内存块的大小
13994:    INT     21H                         ;┘
13995:    MOV     AX,ES                       ;┐
```

```
13996:     DEC     AX                          ;| PID 值(8:DOS 内核使用)→UMB 内存控制块
13997:     MOV     ES,AX                       ;|
13998:     MOV     ES:MCB_ProcessID,8          ;┘
13999:     POP     BX
14000:     POP     ES
14001: ModifyUMBSize_RET:
14002:     RET
14003: ModifyUMBSize  ENDP
14004: ;=============================================
14005: ;相对地址偏移:3026
14006: ;功能:设置常规内存的内存控制块链的结束位置标志
14007: ;入口参数:无
14008: ;出口参数:无
14009: ;=============================================
14010: SetEndFlagOfCMC  PROC  NEAR
14011:     PUSH    DS
14012:     PUSH    ES
14013:     CMP     CS:UMBChain_Flag,0          ;┐若未建立 UMB 内存控制块链,则结束返回
14014:     JE      SetEndFlagOfCMC_3           ;┘
14015:     MOV     ES,WORD PTR CS:Seg_MSDOS1   ;ES=DOS 内核数据区的段地址值
14016:     MOV     DS,ES:DOS_0024              ;DS=第一个常规内存控制块的段地址值
14017:     MOV     DI,ES:DOS_008C              ;DI=UMB 内存控制块链的接口内存控制块的段
                                               ;地址值
14018: SetEndFlagOfCMC_1:                      ;┐
14019:     CALL    GetNextUMBSeg               ;|
14020:     JC      SetEndFlagOfCMC_3           ;|
14021:     CMP     DI,AX                       ;|最后内存控制块的位置标志("Z")→常规内存
14022:     JE      SetEndFlagOfCMC_2           ;|的最后一个内存控制块
14023:     MOV     DS,AX                       ;|
14024:     JMP     SHORT SetEndFlagOfCMC_1     ;|
14025: SetEndFlagOfCMC_2:                      ;|
14026:     MOV     DS:MCB_Location,' Z'        ;┘
14027: SetEndFlagOfCMC_3:
14028:     POP     ES
14029:     POP     DS
14030:     RET
14031: SetEndFlagOfCMC  ENDP
14032: ;相对地址偏移:3054
14033: DevicehighCtrlVar  DB  0                ;转换 devicehigh 命令行时使用的控制变量(1:表示
                                               ;已追加了一个空参数项;0:表示未追加一个空参
                                               ;数项)
14034: ;=============================================
14035: ;相对地址偏移:3055
```

```
14036: ;功 能:设置设备参数
14037: ;入口参数:无
14038: ;出口参数:无
14039: ;=========================================================
14040: Set_DevicePara  PROC  NEAR
14041:     PUSH    DS
14042:     PUSH    AX
14043:     PUSH    BX
14044:     PUSH    CX
14045:     PUSH    DX
14046:     PUSH    CS                      ;                       ┐
14047:     POP     DS                      ;                       |
14048:     XOR     BX,BX                   ;                       |
14049:     MOV     BL,DriveOfDrivParm      ;物理驱动器号(0=A,1=B,  |
                                           ;2=C,…)                 |
14050:     INC     BL                      ;BL←驱动器号(1=A,2=B,   |
                                           ;3=C,…)                 |
14051:     MOV     DX, offset Packet       ;DS:DX 指向设备参数块   | 设置设备参数
14052:     MOV     AH,44H                  ;设备输入/输出控制      |
14053:     MOV     AL,0DH                  ;一般设备 IOCTL         |
14054:     MOV     CH,8                    ;设备类型为"块设备"     |
14055:     MOV     CL,40H                  ;设置子功能为"设置设备参数" |
14056:     INT     21H                     ;请求 DOS 服务          ┘
14057:     TEST    MarkOfControlSym,0004   ;┐若 drivparm 系统配置命令未指定"/1"开关,则跳
14058:     JZ      Set_DevPara1            ;┘转
14059:     MOV     CL,DriveOfDrivParm      ;┐
14060:     MOV     AX,BIO_SEG              ;| 设置 IO1 模块中的控制变量值(即设置"指定软
14061:     MOV     DS,AX                   ;| 盘驱动器是一个兼容的 3.5 英寸软盘驱动器"
14062:     MOV     AL,1                    ;| 的标志位)
14063:     SHL     AL,CL                   ;|
14064:     OR      CompatibleFDiskFlag,AL  ;┘
14065: Set_DevPara1:
14066:     POP     DX
14067:     POP     CX
14068:     POP     BX
14069:     POP     AX
14070:     POP     DS
14071:     RET
14072: Set_DevicePara  ENDP
14073: ;=========================================================
14074: ;相对地址偏移:3090
14075: ;功能:修改"设备参数块(Packet)"的部分参数值
14076: ;入口参数:无
```

```
14077: ;出口参数:无
14078: ;=============================================================
14079: Set_Packet  PROC  NEAR
14080:     PUSH    DS
14081:     PUSH    CS
14082:     POP     DS
14083:     MOV     WORD PTR NumberOfCylinders,80 ;重新设置"圆柱面数"
14084:     MOV     BYTE PTR DeviceTypeByte, 2    ;重新设置"驱动器类型"(2:表示 3.5 英寸的 720KB
                                                 ;软盘驱动器)
14085:     MOV     WORD PTR DeviceAttributes,0   ;重新设置"设置属性"
14086:     MOV     WORD PTR MarkOfControlSym,0   ;清 drivparm 系统配置命令的开关参数标志字
14087:     POP     DS
14088:     RET
14089: Set_Packet  ENDP
14090: ;=============================================================
14091: ;相对地址偏移:30AC
14092: ;功能:取出 drivparm 系统配置命令的参数,并参考 drivparm 系统配置命令指定的开关参数设置对
          应的设备参数块 Packet
14093: ;入口参数:无
14094: ;出口参数:CF=0,drivparm 系统配置命令正确;CF=1,drivparm 系统配置命令错
14095: ;=============================================================
14096: Process_DrivParm  PROC  NEAR
14097:     PUSH    DS
14098:     PUSH    CS
14099:     POP     DS
14100: Pro_DrivP1:
14101:     CMP     AL, CR                        ;┐若 drivparm 系统配置命令已结束,则跳转
14102:     JE      Pro_DrivP5                    ;┘
14103:     CMP     AL, LF                        ;┐若为换行符,则转去恢复当前读取的字符
14104:     JE      Pro_DrivP8                    ;┘
14105:     CMP     AL, ' '                       ;┐若为控制符或空格符,则跳转
14106:     JBE     Pro_DrivP3                    ;┘
14107:     CMP     AL, '/'                       ;┐若是开关参数前导符,则跳转
14108:     JE      Pro_DrivP2                    ;┘
14109:     STC                                   ;┐出错返回
14110:     JMP     SHORT Pro_DrivP7              ;┘
14111: Pro_DrivP2:
14112:     CALL    Process_SwitchPara            ;处理 drivparm 系统配置命令的开关参数
14113:     MOV     MarkOfControlSym,BX           ;保存当前开关参数
14114:     JC      Pro_DrivP4                    ;若开关参数格式错,则结束返回
14115: Pro_DrivP3:
14116:     CALL    Get_CharOfConfig              ;取出一个字符
14117:     JC      Pro_DrivP5                    ;若 drivparm 系统配置命令已结束,则跳转
```

```
14118:      JMP     SHORT Pro_DrivP1                ;转去继续处理后续命令参数串
14119: Pro_DrivP4:
14120:      JMP     SHORT Pro_DrivP7
14121: Pro_DrivP5:                                  ;命令结束处理
14122:      TEST    MarkOfControlSym,0008           ;┐若 drivparm 系统配置命令有"/D"开关参数,则
14123:      JNZ     Pro_DrivP6                      ;┘跳转
14124:      STC                                     ;CF←1(drivparm 系统配置命令格式错)
14125:      JMP     SHORT Pro_DrivP7
14126: Pro_DrivP6:
14127:      MOV     AX,MarkOfControlSym             ;┐
14128:      AND     AX,3                            ;├保存"设备属性"字段值
14129:      MOV     DeviceAttributes,AX             ;┘
14130:      MOV     WORD PTR SectorCount,0          ;清磁道布局字段的总扇区数
14131:      CLC
14132:      CALL    Set_PacketPara                  ;参考 drivparm 系统配置命令指定的参数设置"设
                                                    ;备参数块 Packet"的参数值
14133: Pro_DrivP7:
14134:      POP     DS
14135:      RET
14136: Pro_DrivP8:
14137:      INC     ConfigCount                     ;┐恢复(保留)当前读取的 CONFIG.SYS 文件内容
14138:      DEC     Ptr_Config                      ;┘
14139:      JMP     SHORT Pro_DrivP5
14140: Process_DrivParm  ENDP
14141: ;=============================================================
14142: ;相对地址偏移:30FE
14143: ;功能:处理 drivparm 系统配置命令的开关参数
14144: ;入口参数:无
14145: ;出口参数:BX=开关参数符的标志,其格式如下:
14146: ;  7  6  5  4  3  2  1  0
14147: ;                       └──"/N"开关标志
14148: ;                    └─────"/C"开关标志
14149: ;                 └────────"/I"开关标志
14150: ;              └───────────"/D"开关标志
14151: ;           └──────────────"/T"开关标志
14152: ;        └─────────────────"/S"开关标志
14153: ;     └────────────────────"/H"开关标志
14154: ;  └───────────────────────"/F"开关标志
14155: ;          CF=0,开关参数格式正确;CF=1,开关参数格式不正确
14156: ;=============================================================
14157: Process_SwitchPara  PROC  NEAR
14158:      CALL    Get_CharOfConfig                ;取出开关参数符→AL
14159:      JC      Pro_CSym4                       ;┐
```

```
14160:    AND     AL,0DFH                    ;|
14161:    CMP     AL,'A'                     ;| 判定 AL 指定的字符是否是开关参数符,若不
14162:    JB      Pro_CSym4                  ;| 是,则跳转
14163: Pro_CSym1:                            ;|
14164:    CMP     AL,'Z'                     ;|
14165:    JA      Pro_CSym4                  ;┘
14166:    PUSH    ES                         ;                          ┐
14167:    PUSH    CS                         ;                          |
14168:    POP     ES                         ;                          |
14169:    MOV     CL,NumberOfSwitchSym       ;┐ CX←drivparm 允许的开关 | 检查 drivparm 的
14170:    MOV     CH,0                       ;┘ 参数符个数              | 开关参数符正确
14171:    MOV     DI,offset TableOfSwitchSym ;ES:DI 指向 drivparm 的开关 | 否
                                             ;参数符表                  |
14172:    REPNZ   SCASB                      ;                          |
14173:    POP     ES                         ;                          ┘
14174:    JNZ     Pro_CSym4                  ;若开关参数符非法,则跳转
14175:    MOV     AX,0001                    ;┐ 设置开关参数符对应的标志位
14176:    SHL     AX,CL                      ;┘
14177:    MOV     BX,MarkOfControlSym        ;┐ 与先前的开关参数符标志相或,此时 BX=当前
14178:    OR      BX,AX                      ;┘ 已识别的开关参数符标志
14179:    MOV     CX,AX
14180:    TEST    AX,00F8H                   ;┐ 若开关参数符为 N、I 或 C,则跳转
14181:    JZ      Pro_CSym2                  ;┘
14182:    CALL    Get_CharOfConfig           ;┐
14183:    JB      Pro_CSym3                  ;| 开关参数符 F、H、S、T、D 的后续符(:)对否?,若
14184:    CMP     AL,':'                     ;| 不对,则跳转
14185:    JNZ     Pro_CSym3                  ;┘
14186:    CALL    Get_CharOfConfig           ;取出参数的当前字符→AL
14187:    PUSH    BX                         ;┐
14188:    MOV BYTE PTR CS:TerminatorOfData,' ' ;|
14189:    CALL    Get_Data                   ;| 取出十进制的参数值→AX
14190:    MOV BYTE PTR CS:TerminatorOfData,0 ;|
14191:    POP     BX                         ;┘
14192:    CALL    Get_DrivParmPara           ;保存 drivparm 系统配置命令设定的块设备参数值
14193: Pro_CSym2:
14194:    CLC
14195:    RET
14196: Pro_CSym3:
14197:    XOR     BX,CX                      ;清除开关参数符对应的标志位
14198: Pro_CSym4:
14199:    STC
14200:    RET
14201: Process_SwitchPara  ENDP
```

```
14202: ;=========================================================
14203: ;相对地址偏移:3156
14204: ;功能: 保存 drivparm 系统配置命令的开关参数 F、H、S、T、D 设定的块设备参数
14205: ;入口参数:AX=由开关参数指定的参数值;
14206: ;          CX=当前的 drivparm 开关参数的标志
14207: ;出口参数:无
14208: ;=========================================================
14209: Get_DrivParmPara  PROC  NEAR
14210:     TEST    MarkOfControlSym, CX          ;┐ 若先前已指定了当前开关参数,则结束返回
14211:     JNZ     Get_DPP5                      ;┘
14212:     TEST    CX, 0008                      ;┐ 若当前开关参数不是"/D"开关,则跳转
14213:     JZ      Get_DPP1                      ;┘
14214:     MOV     DriveOfDrivParm, AL           ;保存物理驱动器号(/d:n 的 n 值,0=A,1=B,…)
14215:     JMP     SHORT Get_DPP5
14216: Get_DPP1:
14217:     TEST    CX,0080H                      ;┐ 若当前开关参数不是"/F"开关,则跳转
14218:     JZ      Get_DPP2                      ;┘
14219:     MOV     DeviceTypeByte,AL             ;保存块设备的设备类型(/f:n 的 n 值)
14220:     JMP     SHORT Get_DPP5
14221: Get_DPP2:
14222:     OR      AX,AX                         ;┐ 若由开关参数指定的参数值为 0,则结束返回
14223: JZ Get_DPP5                               ;┘
14224:     TEST    CX,0010H                      ;┐ 若当前开关参数不是"/T"开关,则跳转
14225:     JZ      Get_DPP3                      ;┘
14226:     MOV     NumberOfCylinders,AX          ;保存块设备支持的圆柱面数(/t:n 的 n 值)
14227:     JMP     SHORT Get_DPP5
14228: Get_DPP3:
14229:     TEST    CX,0020H                      ;┐ 若当前开关参数不是"/S"开关,则跳转
14230:     JZ      Get_DPP4                      ;┘
14231:     MOV     SecPerTrackOfDrivParm,AX      ;保存块设备支持的每道扇区数(/s:n 的 n 值)
14232:     JMP     SHORT Get_DPP5
14233: Get_DPP4:
14234:     MOV     HeadsOfDrivParm,AX            ;保存最大磁头数(即/h:n 的 n 值)
14235: Get_DPP5:
14236:     CLC
14237:     RET
14238: Get_DrivParmPara  ENDP
14239: ;=========================================================
14240: ;相对地址偏移:3191
14241: ;功能:参考 drivparm 系统配置命令指定的参数设置功能调用 44H(设置设备参数)的类属 IOCTL 请
           求的参数块(Packet)
14242: ;入口参数:无
14243: ;出口参数:无
```

```
14244: ;=============================================================
14245: Set_PacketPara  PROC  NEAR
14246:     PUSH    ES
14247:     PUSH    CS
14248:     POP     ES
14249:     XOR     BX,BX
14250:     MOV     BL,DeviceTypeByte              ;BL←块设备的设备类型
14251:     CMP     BL,0                           ;┐若驱动器不是 320/360KB 或 160/180KB、5.25
14252:     JNE     Set_Para1                      ;┘英寸的软盘驱动器,则跳转
14253:     MOV     WORD PTR NumberOfCylinders,40  ;设置圆柱面数
14254: Set_Para1:
14255:     SHL     BX,1                           ;┐SI 指向指定类型的软盘驱动器对应的 BPB 参
14256:     MOV     SI,AddrOfBPB[BX]               ;┘数块
14257:     MOV     DI, offset BytesPerSector      ;DI 指向"参数块 Packet"中的设备 BPB 字段
14258:     MOV     CX,1FH                         ;┐
14259:     CLD                                    ;├设置"参数块 Packet"中的设备 BPB 参数块
14260:     REP     MOVSB                          ;┘
14261:     POP     ES                             ;┘
14262:     TEST    MarkOfControlSym,0020H         ;┐若不修改"每道扇区数",则跳转
14263:     JZ      Set_Para2                      ;┘
14264:     MOV     AX,SecPerTrackOfDrivParm       ;┐修改"每道扇区数"
14265:     MOV     SectorsPerTrack,AX             ;┘
14266: Set_Para2:
14267:     TEST    MarkOfControlSym,0040H         ;┐若不修改"磁头数",则跳转
14268:     JZ      Set_Para3                      ;┘
14269:     MOV     AX,HeadsOfDrivParm             ;┐修改"磁头数"
14270:     MOV     Heads,AX                       ;┘
14271: Set_Para3:
14272:     MOV     BYTE PTR SectorsPerCluster,02  ;设置"每簇扇区数"          ┐
14273:     MOV     BL,0F0H                        ;BL←缺省的介质描述符       |
14274:     MOV     BH,MediaDescriptor             ;BH←介质描述符             |
14275:     CMP     Heads,2                        ;┐若"磁头数">2,则跳转       |
14276:     JA      Set_Para7                      ;┘                         |
14277:     JNE     Set_Para5                      ;若"磁头数"≠2,则跳转       |
14278:     MOV     BL,BH                          ;不更改"介质描述符"        |
14279:     CMP     SectorsPerTrack,18             ;┐                         |
14280:     JNE     Set_Para4                      ;├若驱动器类型不是 3.5 英  |
14281:     CMP     NumberOfCylinders,80           ;├寸 1.44MB,则跳转         |
14282:     JNE     Set_Para4                      ;┘                         |
14283:     JMP     SHORT Set_Para6                ;转去修改"每簇扇区数"      |
14284: Set_Para4:                                 ;                          |设置"介质描述
14285:     CMP     NumberOfCylinders,40           ;┐                         |符"和"每簇扇区
14286:     JNE     Set_Para7                      ;├若驱动器类型不是 360KB   |数"
```

```
14287:    CMP     SectorsPerTrack,8                ;|(或以下),则跳转          |
14288:    JNE     Set_Para7                        ;┘                          |
14289:    MOV     BL,0FCH                          ;修改"介质描述符"           |
14290:    JMP     SHORT Set_Para7                  ;                           |
14291: Set_Para5:                                  ;                           |
14292:    CMP     DeviceTypeByte,00                ;┐若驱动器类型不是 360KB     |
14293:    JNE     Set_Para6                        ;┘(或以下),则跳转           |
14294:    MOV     BL,0FCH                          ;┐                          |
14295:    CMP     SectorsPerTrack,8                ;|根据"每道扇区数"修改      |
14296:    JNE     Set_Para6                        ;|"介质描述符"               |
14297:    MOV     BL,0FEH                          ;┘                          |
14298: Set_Para6:                                  ;                           |
14299:    MOV     BYTE PTR SectorsPerCluster,01    ;修改"每簇扇区数"           ┘
14300: Set_Para7:
14301:    MOV     MediaDescriptor,BL               ;设置"介质描述符"
14302:    MOV     AX,NumberOfCylinders             ;┐
14303:    MUL     Heads                            ;|设置"总扇区数"
14304:    MUL     SectorsPerTrack                  ;|
14305:    MOV     TotalSectors,AX                  ;┘
14306:    CLC
14307:    RET
14308: Set_PacketPara  ENDP
14309: ;=============================================================
14310: ;相对地址偏移:3234
14311: ;功能:转换 CONFIG.SYS 的文件内容,具体的转换方法和所完成的工作如下:
14312: ;       1.小写字符转换成大写字符;
14313: ;       2.丢掉每个 CONFIG.SYS 命令行的所有起始参数分隔符
14314: ;       3.系统配置命令名用对应的缩写关键字符取代(如 buffers 用"B"表示,参见*****行程
                序);
14315: ;       4.识别 comment 系统配置命令,并丢掉在它之后的所有注释字符串;
14316: ;       5.命令行的结束符只使用换行符(0AH),而回车符(0DH)被丢掉;
14317: ;       6.用结束符(00H)代替文件名说明串后的参数分隔符;
14318: ;       7.用"Z"字符和换行符(0AH)代替整个非法的命令行;
14319: ;       8.给 devicehigh 命令行追加一个空参数项,以确保能正确地处理 devicehigh 系统配置命令的
                两种格式。
14320: ;入口参数:无
14321: ;出口参数:无
14322: ;=============================================================
14323: ConvertConfig  PROC  NEAR
14324:    MOV     CX,CS:ConfigCount                ;CX←CONFIG.SYS 文件内容的字节数
14325:    JCXZ    ConvertConfig3                   ;若 CONFIG.SYS 文件没有内容,则跳转
14326:    CALL    Capital                          ;将 CONGFIG.SYS 文件内容全部转换成大写字符
14327:    XOR     SI,SI                            ;┐ ES:DI(SI)指向存放 CONFIG.SYS 文件内容的
```

```
14328:      MOV     DI,SI                   ;┘ 缓冲区
14329:      XOR     AX,AX
14330:      MOV     CS:QuotationMark,00     ;清双引号作用标志,表示双引号已配对
14331: ConvertConfig1:                      ;                          ┐
14332:      CALL    OmitCommentStr          ;丢掉注释字符串            |
14333:      JE      ConvertConfig2          ;若命令行为一注释行,则跳转 | 丢掉命令行头的
14334:      CALL    GetConfigChar           ;取出 CONFIG.SYS 文件的当  | 所有无效(无作
                                            ;前字符→AL                 | 用)的字符(包括
14335:      CMP     AL,LF                   ;┐ 若当前字符是换行符,     | 丢掉注释行的全
14336:      JE      ConvertConfig2          ;┘ 则转去保留该字符        | 部内容,只留下命
14337:      CMP     AL,' '                  ;┐ 若当前字符是控制符,     | 令行的结束符,以
14338:      JBE     ConvertConfig1          ;┘ 则废弃该字符            | 便统计命令行的
14339:      JMP     SHORT ConvertConfig4    ;                          | 行号,供显示命令
14340: ConvertConfig2:                      ;                          | 行的位置信息时
14341:      STOSB                           ;保存命令行的结束符(0AH)   | 使用)
14342:      MOV     CS:QuotationMark,00     ;清双引号作用标志,表示双引 |
                                            ;号已配对                  |
14343:      JMP     SHORT ConvertConfig1    ;                          ┘
14344: ConvertConfig3:
14345:      STC
14346:      RET
14347: ConvertConfig4:
14348:      PUSH    CX
14349:      PUSH    SI
14350:      PUSH    DI
14351:      MOV     BP,SI                   ;BP←CONFIG.SYS 的处理指针值
14352:      DEC     BP                      ;BP 指向 CONFIG.SYS 的待处理内容
14353:      MOV     SI, offset TableOfConfigCMD ;DS:SI 指向 DOS 支持的系统配置命令表
14354:      MOV     CH,0
14355: ConvertConfig5:
14356:      MOV     DI,BP                   ;ES:DI 指向 CONFIG.SYS 的待转换内容
14357:      MOV     CL,[SI]                 ;CX←当前系统配置命令串的长度(字符个数)
14358:      INC     SI                      ;DS:SI 指向当前系统配置命令串
14359:      JCXZ    ConvertConfig7          ;若系统配置命令表已搜索完,则跳转
14360:      REPE    CMPSB                   ;┐
14361:      LAHF                            ;|
14362:      ADD     SI,CX                   ;| CONFIG.SYS 的待转换内容中是否含有当前系
14363:      SAHF                            ;| 统配置命令?若没有,则转去与后续系统配置命
14364:      LODSB                           ;| 令相比较
14365:      JNZ     ConvertConfig5          ;┘
14366:      CMP     BYTE PTR ES:[DI],CR     ;┐
14367:      JE      ConvertConfig6          ;| 若系统配置命令行结束,则跳转
14368:      CMP     BYTE PTR ES:[DI],LF     ;|
```

```
14369:    JE      ConvertConfig6                ;┘
14370:    PUSH    AX                            ;┐
14371:    MOV     AL,ES:[DI]                    ;│
14372:    CALL    TestTerminatorOrSeparator     ;│若系统配置命令不匹配,则转去继续匹配
14373:    POP     AX                            ;│
14374:    JNE     ConvertConfig5                ;┘
14375: ConvertConfig6:
14376:    POP     DI
14377:    POP     SI
14378:    POP     CX
14379:    JMP     SHORT ConvertConfig9
14380: ConvertConfig7:                          ;处理"非法的系统配置命令"
14381:    POP     DI
14382:    POP     SI
14383:    POP     CX
14384:    MOV     AL,' Z'                       ;┐存入代表"非法的系统配置命令"的关键字符
14385:    STOSB                                 ;┘
14386: ConvertConfig8:                          ;┐
14387:    CALL    GetConfigChar                 ;│跳过"非法的系统配置命令"的命令行,使 DS:SI
14388:    CMP     AL,0AH                        ;│指向下一个 CONFIG.SYS 命令行
14389:    JNE     ConvertConfig8                ;┘
14390:    JMP     SHORT ConvertConfig2          ;转去存入命令结束符,并开始转换下一个
                                                ;CONFIG.SYS 命令行
14391: ConvertConfig9:                          ;AL=系统配置命令的编写关键字符
14392:    STOSB                                 ;存入系统配置命令的缩写关 ┐
                                                ;键字符                   │
14393:    MOV     CS:KeySign,AL                 ;保存当前匹配的系统配置命 │
                                                ;令的缩写关键字符         │
14394: ConvertConfig10:                         ;                         │
14395:    CALL    GetConfigChar                 ;取出一个待转换的字符→AL  │系统配置命令名
14396:    CMP     AL,0AH                        ;┐                        │被对应的缩写关
14397:    JE      ConvertConfig11               ;│若 AL 指定的字符是命令行 │键字符取代
                                                ;│的结束符,则跳转         │
14398:    CMP     AL,0DH                        ;│                        │
14399:    JE      ConvertConfig11               ;┘                        │
14400:    CALL    TestTerminatorOrSeparator     ;┐若 AL 指定的字符不是参  │
                                                ;│数终止符或参数分隔符,   │
14401:    JNZ     ConvertConfig10               ;┘则跳转                  │
14402:    JMP     SHORT ConvertConfig12         ;                         │
14403: ConvertConfig11:                         ;                         │
14404:    DEC     SI                            ;                         │
14405:    INC     CX                            ;                         │
14406: ConvertConfig12:                         ;                         ┘
```

```
14407:    CMP     CS:KeySign,' Y'          ;┐ 若当前系统配置命令是"comment",则跳转
14408:    JE      ConvertConfig16          ;┘
14409:    CMP     CS:KeySign,' I'          ;┐ 若当前系统配置命令是"install",则跳转
14410:    JE      ConvertConfig14          ;┘
14411:    CMP     CS:KeySign,' D'          ;┐ 若当前系统配置命令是"device",则跳转
14412:    JE      ConvertConfig14          ;┘
14413:    CMP     CS:KeySign,' S'          ;┐ 若当前系统配置命令是"shell",则跳转
14414:    JE      ConvertConfig14          ;┘
14415:    CMP     CS:KeySign,' I'          ;┐ 若当前系统配置命令是"switches",则跳转
14416:    JE      ConvertConfig13          ;┘
14417:    JMP     ConvertConfig22
14418: ConvertConfig13:                     ;处理 switches 系统配置命令的参数
14419:    CALL    OmitCommentStr           ;丢掉注释字符串
14420:    JZ      ConvertConfig18          ;若 switches 系统配置命令中含有注释,则跳转(此
                                           ;时命令参数已结束)
14421:    CALL    GetConfigChar            ;┐
14422:    CALL    TestParaSeparator        ;| 跳过参数前的所有参数分隔符
14423:    JZ      ConvertConfig13          ;┘
14424:    STOSB                            ;保存当前有效的字符
14425:    JMP     ConvertConfig23
14426: ConvertConfig14:                     ;处理 install、device 和 shell 系统配置命令的参数
14427:    CALL    OmitCommentStr           ;丢掉注释字符串
14428:    JZ      ConvertConfig19          ;若命令行中有注释,则跳转(此时命令参数已结
                                           ;束)
14429:    CALL    GetConfigChar            ;┐
14430:    CALL    TestTerminatorOrSeparator ;| 跳过参数前的所有参数分隔符或终止符
14431:    JZ      ConvertConfig14          ;┘
14432:    STOSB                            ;保存当前有效的字符
14433: ConvertConfig15:
14434:    CALL    OmitCommentStr           ;丢掉注释字符串
14435:    JZ      ConvertConfig19          ;若命令行中有注释,则跳转(此时命令参数已结
                                           ;束)
14436:    CALL    GetConfigChar            ;取出一个待转换的字符→AL
14437:    CMP     AL,' /'                  ;┐ 若文件名说明串已结束,则跳转
14438:    JE      ConvertConfig20          ;┘
14439:    STOSB                            ;保存当前有效的字符
14440:    CALL    TestTerminatorOrSeparator ;┐ 若 AL 指定的字符是参数终止符或参数分隔符,
14441:    JZ      ConvertConfig21          ;┘ 则跳转
14442:    CMP     AL,' '                   ;┐ 若 AL 指定的字符对文件名说明串是合法的,则
14443:    JA      ConvertConfig15          ;┘ 跳转
14444:    JMP     SHORT ConvertConfig21
14445: ConvertConfig16:                     ;处理 comment 系统配置命令的参数
14446:    CALL    GetConfigChar            ;┐
```

```
14447:    CMP     AL,' '                  ;|
14448:    JE      ConvertConfig16         ;| 跳过参数前的所有参数分隔符
14449:    CMP     AL,Tab                  ;|
14450:    JE      ConvertConfig16         ;|
14451:    CMP     AL,'='                  ;|
14452:    JE      ConvertConfig16         ;┘
14453:    CMP     AL,CR                   ;┐
14454:    JE      ConvertConfig17         ;| 若 comment 系统配置命令结束,则跳转
14455:    CMP     AL,LF                   ;|
14456:    JE      ConvertConfig17         ;┘
14457:    MOV     CS:FirstCharOfPreface,AL ;保存由 comment 系统配置命令的指定的注释前导
                                          ;符的第 1 个组成字符
14458:    MOV     CS:CharsOfPreface,1     ;设置由 comment 系统配置命令指定的注释前导符
                                          ;的组成字符个数(1)
14459:    CALL    GetConfigChar           ;┐
14460:    CMP     AL,' '                  ;| 若 comment 系统配置命令的当前字符是参数分
14461:    JE      ConvertConfig17         ;| 隔符,则跳转
14462:    CMP     AL,Tab                  ;|
14463:    JE      ConvertConfig17         ;┘
14464:    CMP     AL,CR                   ;┐ 若 comment 系统配置命令即将结束,则跳转
14465:    JE      ConvertConfig17         ;┘
14466:    CMP     AL,LF                   ;┐ 若 comment 系统配置命令已结束,则转去保存
                                          ;| 命令行的结束符,并开始转换下一个 CONFIG.
14467:    JE      ConvertConfig18         ;┘ SYS 命令行
14468:    MOV     CS:SecondCharOfPreface,AL ;保存由 comment 系统配置命令指定的注释前导符
                                          ;的第 2 个组成字符
14469:    INC     CS:CharsOfPreface       ;累计由 comment 系统配置命令指定的注释前导符
                                          ;的组成字符个数(2)
14470: ConvertConfig17:                   ;┐
14471:    CALL    GetConfigChar           ;| 丢掉命令参数之后的所有无效字符
14472:    CMP     AL,LF                   ;|
14473:    JNE     ConvertConfig17         ;┘
14474: ConvertConfig18:
14475:    JMP     ConvertConfig2          ;转去保存命令行的结束符(0AH),并开始转换下
                                          ;一个 CONFIG.SYS 命令行

14476: ConvertConfig19:
14477:    MOV     BYTE PTR ES:[DI],0      ;┐ 设置 ASCIIZ 字符串的结束符
14478:    INC     DI                      ;┘
14479:    JMP     ConvertConfig2          ;转去保存命令行的结束符(0AH),并开始转换下
                                          ;一个 CONFIG.SYS 命令行

14480: ConvertConfig20:
14481:    MOV     BYTE PTR ES:[DI],0      ;┐ 设置文件名说明串的结束符
14482:    INC     DI                      ;┘
```

```
14483:    STOSB                                ;保存当前有效的字符
14484:    JMP     SHORT ConvertConfig23
14485: ConvertConfig21:
14486:    MOV     BYTE PTR ES:[DI-1],0         ;设置文件名说明串的结束符
14487:    CMP     AL,LF                        ;┐若命令行已结束,则跳转
14488:    JE      ConvertConfig18              ;┘
14489:    JMP     SHORT ConvertConfig23
14490: ConvertConfig22:
14491:    CALL    OmitCommentStr               ;丢掉注释字符串
14492:    JZ      ConvertConfig18              ;若命令行中有注释,则跳转(此时命令参数已结
                                               ;束)
14493:    CALL    GetConfigChar                ;┐
14494:    CALL    TestParaSeparator            ;├跳过参数前的所有参数分隔符
14495:    JZ      ConvertConfig22              ;┘
14496:    JMP     SHORT ConvertConfig24
14497: ConvertConfig23:
14498:    CALL    OmitCommentStr               ;丢掉注释字符串
14499:    JZ      ConvertConfig18              ;若命令行中有注释,则跳转(此时命令参数已结
                                               ;束)
14500:    CALL    GetConfigChar                ;取出一个待转换的字符→AL
14501: ConvertConfig24:
14502:    STOSB                                ;保存 AL 指定的当前字符
14503:    CMP     AL,'"'                       ;┐若 AL 指定的字符是双引号,则转去检查双引号
14504:    JE      ConvertConfig28              ;┘的配对情况
14505:    CMP     AL,' '                       ;┐若 AL 指定的字符不是参数分隔符或命令行结
14506:    JA      ConvertConfig23              ;┘束符,则跳转
14507:    CMP     CS:KeySign,'U'               ;┐若当前系统配置命令不是 devicehigh,则跳转
14508:    JNE     ConvertConfig25              ;┘
14509:    CMP     AL,LF                        ;┐若 AL 指定的字符是命令行的结束符,则跳转
14510:    JE      ConvertConfig26              ;┘
14511:    CMP     AL,CR                        ;┐若 AL 指定的字符不是回车符,则跳转
14512:    JNE     ConvertConfig23              ;┘
14513:    MOV     BYTE PTR ES:[DI-1],' '       ;┐在回车符之前插入一个空格符,即追加一个空
                                               ;├参数项(这是因为 devicehigh 系统配置命令的有
14514:    STOSB                                ;┘两种格式)
14515:    INC     CS:DevicehighCtrlVar         ;设置追加一个空参数项的标志
14516:    JMP     SHORT ConvertConfig23
14517: ConvertConfig25:
14518:    CMP     AL,LF                        ;┐若 AL 指定的字符是命令行的结束符,则转去转
14519:    JE      ConvertConfig27              ;┘换下一个 CONFIG.SYS 命令行
14520:    JMP     SHORT ConvertConfig23
14521: ConvertConfig26:
14522:    CMP     CS:DevicehighCtrlVar,1       ;┐
```

```
14523:    JE      ConvertConfig27                ;| 确保给 devicehigh 系统配置命令追加一个空
14524:    MOV     BYTE PTR ES:[DI-1],' '         ;| 参数项
14525:    STOSB                                  ;|
14526: ConvertConfig27:                          ;┘
14527:    MOV     CS:DevicehighCtrlVar,0         ;清标志,表示未给 devicehigh 系统配置命令追加空
                                                 ;参数项
14528:    JMP     ConvertConfig1                 ;转去转换下一个 CONFIG.SYS 命令行
14529: ConvertConfig28:                          ;检查双引号的配对情况
14530:    CMP     CS:QuotationMark,0             ;┐ 若先前未出现双引号,则跳转
14531:    JE      ConvertConfig29                ;┘
14532:    MOV     CS:QuotationMark,0             ;清双引号的作用标志(0:双引号已配对)
14533:    JMP     SHORT ConvertConfig23
14534: ConvertConfig29:
14535:    INC     CS:QuotationMark               ;置双引号的作用标志(1:双引号未配对)
14536:    JMP     SHORT ConvertConfig23
14537: ConvertConfig  ENDP
14538: ;=============================================================
14539: ;相对地址偏移:33FA
14540: ;功能:从 CONFIG.SYS 文件的原始数据中取出一个待转换的字符。当 CONFIG.SYS 的文件内容全
        部被转换完时,在设置与 CONFIG.SYS 处理相关的控制变量之后,结束 CONFIG.SYS 的文件
        内容转换
14541: ;入口参数:CX=CONFIG.SYS 待转换的字符数
14542: ;          ES:SI 指向 CONFIG.SYS 的待转换内容
14543: ;出口参数:AL=取出字符的 ASCII 码
14544: ;          CX=新的 CONFIG.SYS 待转换的字符数
14545: ;          ES:SI 指向新的 CONFIG.SYS 的待转换内容
14546: ;=============================================================
14547: GetConfigChar  PROC  NEAR
14548:    JCXZ    GetConfigChar_1                ;若 CONFIG.SYS 的文件内容已转换完,则转去初
                                                 ;始化 CONFIG.SYS 识别时使用的控制变量值
14549:    MOV     AL,ES:[SI]                     ;┐ 取出 CONFIG.SYS 文件的当前字符→AL
14550:    INC     SI                             ;┘
14551:    DEC     CX                             ;CX←CONFIG.SYS 待转换的字符数
14552:    RET
14553: GetConfigChar_1:
14554:    POP     CX                             ;去掉返回地址
14555:    MOV     CS:ConfigCount,DI              ;设置"CONFIG.SYS 待处理的字符计数器"的初始
                                                 ;值
14556:    MOV     CS:BytesOfConfig,DI            ;设置 CONFIG.SYS 有效内容的字节数
14557:    XOR     SI,SI                          ;┐ 初始化 CONFIG.SYS 待处理内容的读指针值
14558:    MOV     CS:Ptr_Config,SI               ;┘
14559:    RET                                    ;结束转换,相当于从 ConvertConfig 子程序返回
14560: GetConfigChar  ENDP
```

```
14561: ;=============================================================
14562: ;相对地址偏移:3415
14563: ;功能:丢掉(跳过)注释字符串
14564: ;入口参数:ES:SI 指向 CONFIG.SYS 待处理的内容
14565: ;出口参数:ES:SI 指向新的 CONFIG.SYS 待处理的内容
14566: ;          ZF=0,未发现注释;ZF=1,已丢掉注释字符串
14567: ;=============================================================
14568: OmitCommentStr  PROC  NEAR
14569:     JCXZ    GetConfigChar_1          ;若 CONFIG.SYS 的文件内容已转换完,则转去初
                                           ;始化 CONFIG.SYS 识别时使用的控制变量值
14570:     CMP     CS:QuotationMark,0       ;┐若双引号未配对,则直接返回
14571:     JNE     OmitCommentStr_RET       ;┘
14572:     CMP     CS:CharsOfPreface,1      ;┐若用户未指定注释前导符,则直接返回
14573:     JB      OmitCommentStr_RET       ;┘
14574:     MOV     AL,ES:[SI]               ;AL←当前待转换的字符      ┐
14575:     CMP     CS:FirstCharOfPreface,AL ;┐若 AL 指定的字符不是注释  |
14576:     JNE     OmitCommentStr_RET       ;┘前导符,则直接返回       |
14577:     CMP     CS:CharsOfPreface,2      ;┐若用户只指定一个注释     | 若当前未出现注
14578:     JNE     OmitCommentStr_1         ;┘前导符,则跳转           | 释前导符,则直
14579:     MOV     AL,ES:[SI+1]             ;┐若第二个字符不为注释前   | 接返回
14580:     CMP     CS:SecondCharOfPreface,AL ;| 导符的第二个字符,则直接 |
14581:     JNE     OmitCommentStr_RET       ;┘返回                    ┘
14582: OmitCommentStr_1:                    ;┐
14583:     JCXZ    GetConfigChar_1          ;|
14584:     MOV     AL,ES:[SI]               ;|
14585:     INC     SI                       ;| 丢掉注释字符串
14586:     DEC     CX                       ;|
14587:     CMP     AL,LF                    ;|
14588:     JNE     OmitCommentStr_1         ;┘
14589: OmitCommentStr_RET:
14590:     RET
14591: OmitCommentStr  ENDP
14592: ;=============================================================
14593: ;相对地址偏移:3450
14594: ;功能:判定 AL 指定的字符是否是参数终止符或参数分隔符
14595: ;入口参数:AL=字符的 ASCII 码
14596: ;出口参数:ZF=0,AL 指定的字符不是参数终止符或参数分隔符;ZF=1,AL 指定的字符是参数终
              止符或参数分隔符
14597: ;=============================================================
14598: TestTerminatorOrSeparator  PROC  NEAR
14599:     CMP     AL,' /'
14600:     JE      TestParaSeparator_RET
14601:     CMP     AL,0
```

```
14602:     JE        TestParaSeparator_RET
14603: ;=========================================================
14604: ;相对地址偏移:3458
14605: ;功能:判定 AL 指定的字符是否是参数分隔符
14606: ;入口参数:AL=字符的 ASCII 码
14607: ;出口参数:ZF=0,AL 指定的字符不是参数分隔符;ZF=1,AL 指定的字符是参数分隔符
14608: ;=========================================================
14609: TestParaSeparator  LABEL  NEAR
14610:     CMP       AL,' '
14611:     JE        TestParaSeparator_RET
14612:     CMP       AL,Tab
14613:     JE        TestParaSeparator_RET
14614:     CMP       AL,'='
14615:     JE        TestParaSeparator_RET
14616:     CMP       AL,','
14617:     JE        TestParaSeparator_RET
14618:     CMP       AL,';'
14619: TestParaSeparator_RET:
14620:     RET
14621: TestTerminatorOrSeparator  ENDP
14622: ;=========================================================
14623: ;相对地址偏移:346B
14624: ;功能:跳过一个 CONFIG.SYS 命令行
14625: ;入口参数:无
14626: ;出口参数:AL=取出的字符(待识别的字符)
14627: ;=========================================================
14628: Omit_A_ConfigCMD  PROC  NEAR
14629: Omit_A_ConfigCMD1:
14630:     CALL      Get_CharOfConfig
14631:     JC        Omit_A_ConfigCMD_RET
14632:     CMP       AL, LF
14633:     JNE       Omit_A_ConfigCMD1
14634:     CALL      Get_CharOfConfig
14635: Omit_A_ConfigCMD_RET:
14636:     RET
14637: Omit_A_ConfigCMD  ENDP
14638: ;=========================================================
14639: ;相对地址偏移:3478
14640: ;功能:将 ES:0 指定的 CX 个大小写混合的字符全部转换成大写,并存放在原来的位置
14641: ;入口参数:CX=字符个数;
14642: ;          ES:0=数据存放区的起始地址
14643: ;出口参数:无
14644: ;=========================================================
```

```
14645: Capital  PROC  NEAR
14646:     PUSH    CX
14647:     PUSH    SI
14648:     PUSH    DS
14649:     PUSH    ES
14650:     POP     DS
14651:     XOR     SI,SI
14652: Capital1:
14653:     LODSB
14654:     CMP     AL,' a'
14655:     JB      Capital2
14656:     CMP     AL,' z'
14657:     JA      Capital2
14658:     SUB     AL,' a' -' A'
14659:     MOV     [SI-1],AL
14660: Capital2:
14661:     LOOP    Capital1
14662:     POP     DS
14663:     POP     SI
14664:     POP     CX
14665:     RET
14666: Capital  ENDP
14667: ;=============================================
14668: ;相对地址偏移:3493
14669: ;功能:转换常规自由内存的起始地址值格式;检查是否有空闲内存;并设置子段的大小
14670: ;入口参数:无
14671: ;出口参数:无
14672: ;=============================================
14673: Convert_AddrFMT  PROC  NEAR
14674:     PUSH    AX
14675:     MOV     AX,CS:OFS_FreeMemory        ;┐
14676:     CALL    Convert_Addr                ;│转换常规自由内存的起始地址值格式,使得段
14677:     ADD     CS:SEG_FreeMemory,AX        ;│内偏移值为 0
14678:     MOV     CS:OFS_FreeMemory,0         ;┘
14679:     MOV     AX,CS:SEG_FreeMemory        ;┐若系统中没有空闲的常规内存,则转去给出错
14680:     CMP     AX,CS:Seg_CDSArray          ;│误信息
14681:     JAE     MemoryError                 ;┘
14682:     TEST    CS:FlagOfSCB,02             ;┐若不建立子段控制块,则结束返回
14683:     JE      Convert_AddrFMT1            ;┘
14684:     PUSH    ES                          ;在作为 DOS 数据区的第一个内存块内为各个 DOS
                                               ;数据结构(如设备驱动程序、CDS 数组等)建立子
                                               ;段控制块
14685:     PUSH    SI
```

```
14686:    MOV     SI,CS:Seg_CurrSCB                ;┐ ES←当前子段控制块的段地址值
14687:    MOV     ES,SI                            ;┘
14688:    SUB     AX,SI                            ;┐
14689:    DEC     AX                               ;├ 设置当前子段的大小(节数)
14690:    MOV     ES:MCB_AllocationAmount,AX       ;┘
14691:    AND     CS:FlagOfSCB,0FDH                ;清除"建立子段控制块"的标志位
14692:    POP     SI
14693:    POP     ES
14694: Convert_AddrFMT1:
14695:    POP     AX
14696:    CLC
14697:    RET
14698: MemoryError:
14699:    MOV     DX, offset Err_Memory            ;┐
14700:    PUSH    CS                               ;├ 显示错误信息"Configuration too large for
14701:    POP     DS                               ;├ memory"
14702:    CALL    DisplayString                    ;┘
14703:    JMP     SysInt_II_E2_20                  ;系统进入死循环
14704: Convert_AddrFMT  ENDP
14705: ;=============================================================
14706: ;相对地址偏移:34E0
14707: ;功能 :调用设备驱动程序的策略过程和中断过程
14708: ;入口参数:ES=CS
14709: ;出口参数:无
14710: ;=============================================================
14711: RequestDrive  PROC  NEAR
14712:    MOV     DS,WORD PTR CS:DriverLoadAddr+2   ;DS←设备驱动程序标题的段地址值
14713:    ADD     BX,WORD PTR CS:DriverLoadAddr;┐
14714:    MOV     AX,[BX]                          ;├ 设置要执行的程序地址
14715:    PUSH    WORD PTR CS:DriverLoadAddr       ;├
14716:    MOV     WORD PTR CS:DriverLoadAddr,AX;┘
14717:    MOV     BX, offset RequestForInit        ;ES:BX 指向请求调用表
14718:    CALL    DWORD PTR CS:DriverLoadAddr      ;调用设备驱动程序的策略过程或中断过程
14719:    POP     WORD PTR CS:DriverLoadAddr       ;恢复变量值
14720:    RET
14721: RequestDrive  ENDP
14722: ;=============================================================
14723: ;相对地址偏移:3503
14724: ;功能:判定 AL 指定的字符是否是数字符("0~9")
14725: ;入口参数:AL=字符的 ASCII 码
14726: ;出口参数:AL=十进制的参数值;
14727: ;          CF=0,是数字符,CF=1,不是数字符
14728: ;=============================================================
```

```
14729: Test_Numeric  PROC  NEAR
14730:     SUB     AL, ' 0'
14731:     JC      Test_Numeric1
14732:     CMP     AL, 9
14733:     JA      Test_Numeric1
14734:     CLC
14735:     RET
14736: Test_Numeric1:
14737:     STC
14738:     RET
14739: Test_Numeric  ENDP
14740: ;=============================================================
14741: ;相对地址偏移:350F
14742: ;功能:取出十进制的参数值
14743: ;入口参数:AL=参数项的第 1 个字符的 ASCII 码
14744: ;出口参数:AX=十进制的参数值
14745: ;=============================================================
14746: Get_Data  PROC  NEAR
14747:     PUSH    BX
14748:     XOR     BX,BX                   ;清累积和寄存器
14749: Get_Data1:
14750:     CALL    Test_Numeric            ;┐若 AL 指定的字符不是数字符,则跳转
14751:     JC      DataError               ;┘
14752:     XCHG    AX,BX                   ;┐
14753:     PUSH    BX                      ;|
14754:     MOV     BX,0AH                  ;|
14755:     MUL     BX                      ;|若参数值超过 65535,则跳转
14756:     POP     BX                      ;|
14757:     ADD     AL,BL                   ;|
14758:     ADC     AH,0                    ;|
14759:     JC      DataError               ;┘
14760:     XCHG    AX,BX
14761:     CALL    Get_CharOfConfig        ;取出下一个字符→AL
14762:     JC      Get_Data3               ;┐
14763:     CMP     AL, ' '                 ;|
14764:     JE      Get_Data2               ;|
14765:     CMP     AL, ' ,'                ;|
14766:     JE      Get_Data2               ;|
14767:     CMP     AL, Tab                 ;|
14768:     JE      Get_Data2               ;|
14769:     CMP     AL,CS:TerminatorOfData  ;|若参数数字串结束了,则跳转
14770:     JE      Get_Data2               ;|
14771:     CMP     AL, ' /'                ;|
```

```
14772:    NOP                                ;|
14773:    NOP                                ;|
14774:    JZ      Get_Data2                  ;|
14775:    CMP     AL, LF                     ;|
14776:    JE      Get_Data2                  ;|
14777:    CMP     AL, CR                     ;|
14778:    JE      Get_Data2                  ;┘
14779:    OR      AL,AL                      ;┐ 若命令字符串未结束,则继续累计后续的数字
14780:    JNZ     Get_Data1                  ;┘ 符
14781: Get_Data2:
14782:    INC     CS:ConfigCount             ;┐ 确保 CONFIG.SYS 待处理内容的指针指向下一
                                             ;| 个参数项或下一个 CONFIG.SYS 命令行的起
14783:    DEC     CS:Ptr_Config              ;┘ 始处
14784: Get Data3:
14785:    MOV     AX,BX
14786:    OR      AX,AX
14787:    POP     BX
14788:    RET
14789: Get_Data  ENDP
14790: ;=============================================================
14791: ;相对地址偏移:3561
14792: ;功能:设置数据出错标志
14793: ;入口参数:无
14794: ;出口参数:CF=1,AX=0
14795: ;=============================================================
14796: DataError  PROC  NEAR
14797:    MOV     CS:TerminatorOfData,0
14798:    XOR     AX,AX
14799:    POP     BX
14800:    STC
14801:    RET
14802: DataError  ENDP
14803: ;=============================================================
14804: ;相对地址偏移:356C
14805: ;功能:根据 country 系统配置命令指定的国家信息文件设置位于 MSDOS1 模块中的扩充国家信息
          管理块的数据
14806: ;入口参数:AX=国家代码;
14807: ;          BX=国家信息文件的文件句柄;
14808: ;          DX=代码页号;
14809: ;          DS:SI 指向自由内存区的起始地址;
14810: ;          ES:DI 指向位于 MSDOS1 模块中的扩充国家信息管理块
14811: ;出口参数:CF=0,成功;CF=1,CX=-1,失败,表明国家信息文件中没有指定的代码页号或国家
          代码对应的国家信息
```

```
14812: ;=============================================================
14813: Process_Country  PROC  NEAR
14814:     PUSH    DI
14815:     PUSH    AX
14816:     PUSH    DX
14817:     XOR     CX,CX                    ;┐
14818:     XOR     DX,DX                    ;| 从指定的国家信息文件头开始读取 200H 个字
14819:     MOV     AX,200H                  ;| 节的内容(读取国家信息文件的文件标题)
14820:     CALL    ReadFile                 ;┘
14821:     JC      Pro_Country4
14822:     PUSH    ES                       ;┐
14823:     PUSH    SI                       ;|
14824:     PUSH    CS                       ;|
14825:     POP     ES                       ;| 判断指定的文件格式是否是国家信息文件所要
14826:     MOV     DI,offset MarkOfCountry  ;| 求的格式
14827:     MOV     CX,8                     ;|
14828:     REPE    CMPSB                    ;|
14829:     POP     SI                       ;|
14830:     POP     ES                       ;┘
14831:     JNZ     Pro_Country4             ;若不是,则跳转
14832:     ADD     SI,12H                   ;┐
14833:     CMP     BYTE PTR [SI],1          ;| 若标志字节不对,则跳转
14834:     JNE     Pro_Country4             ;┘
14835:     INC     SI                       ;┐ 取出国家信息文件支持的扩充国家信息标题在
14836:     MOV     DX,[SI]                  ;| 文件中的起始位置
14837:     MOV     CX,[SI+2]                ;┘
14838:     MOV     AX,1800H                 ;┐ 读取扩充国家信息标题
14839:     CALL    ReadFile                 ;┘
14840:     JC      Pro_Country4
14841:     MOV     CX,[SI]                  ;CX←国家信息文件支持的国家代码和对应的代码
                                            ;页号的组合个数
14842:     CMP     CX,1B6H;48H              ;┐ 若超出了 438 个,则跳转
14843:     JA      Pro_Country4             ;┘
14844:     INC     SI
14845:     INC     SI
14846:     POP     DX                       ;DX=代码页号
14847:     POP     AX                       ;AX=国家代码
14848:     POP     DI                       ;ES:DI 指向位于 MSDOS1 模块中的扩充国家信息
                                            ;管理块(MSDOS:122A)
14849: Pro_Country1:
14850:     CMP     AX,[SI+2]                ;┐ 若国家代码不同,则跳转
14851:     JNE     Pro_Country2             ;┘
14852:     CMP     DX,0                     ;┐ 若是取缺省的代码页,则跳转
```

```
14853:    JE      Pro_Country5                ;┘
14854:    CMP     DX,[SI+4]                   ;┐ 若代码页号相符,则跳转
14855:    JE      Pro_Country6                ;┘
14856: Pro_Country2:
14857:    ADD     SI,[SI]                     ;加上当前信息控制块的长度,SI 指向下一个信息
                                              ;控制块
14858:    INC     SI
14859:    INC     SI
14860:    LOOP    Pro_Country1                ;继续检查
14861:    MOV     CX,-1                       ;设置未找到指定的代码页号或国家代码对应的国
                                              ;家信息的标志
14862: Pro_Country3:
14863:    STC
14864:    RET
14865: Pro_Country4:                          ;COUNTRY.SYS 文件格式不对
14866:    POP     SI
14867:    POP     CX
14868:    POP     DI
14869:    JMP     SHORT Pro_Country3
14870: Pro_Country5:
14871:    MOV     DX,[SI+4]                   ;DX←代码页号
14872: Pro_Country6:
14873:    MOV     CS:CodePage,DX              ;保存代码页号
14874:    MOV     DX,[SI+0AH]                 ;┐ 取出对应的扩充国家信息管理块在文件中的起
14875:    MOV     CX,[SI+0CH]                 ;┘ 始位置
14876:    MOV     AX,200H                     ;┐ 读取对应的扩充国家信息管理块
14877:    CALL    ReadFile                    ;┘
14878:    JC      Pro_Country3
14879:    MOV     CX,[SI]                     ;CX←支持的信息 ID 个数
14880:    INC     SI
14881:    INC     SI
14882: Pro_Country7:                          ;更换位于 MSDOS1 模块中的扩充国家信息
14883:    PUSH    DI
14884:    PUSH    CX
14885:    PUSH    SI
14886:    MOV     AL,[SI+2]                   ;AL←信息 ID 值
14887:    CALL    Get_AddrOfInfoID            ;取出存放 AL 指定的信息 ID 对应的数据信息的缓
                                              ;冲区地址→ES:DI
14888:    JC      Pro_Country9                ;若信息 ID 值无效,则跳转
14889:    MOV     DX,[SI+4]                   ;┐
14890:    MOV     CX,[SI+6]                   ;│ 移动文件读写指针到 AL 指定的信息 ID 对应的
14891:    MOV     AX,4200H                    ;│ 数据信息标题
14892:    STC                                 ;│
```

```
14893:  INT     21H                     ;┘
14894:  JC      Pro_Country4
14895:  MOV     DX,200H                 ;┐
14896:  MOV     CX,0014H                ;│
14897:  MOV     AH,3FH                  ;│ 读取 AL 指定的信息 ID 对应的数据信息标题
14898:  STC                             ;│
14899:  INT     21H                     ;┘
14900:  JC      Pro_Country4            ;┐
14901:  CMP     AX,CX                   ;│ 若数据错误,则跳转
14902:  JNE     Pro_Country4            ;┘
14903:  MOV     DX,[SI+4]               ;┐
14904:  MOV     CX,[SI+6]               ;│
14905:  MOV     AX,4200H                ;│ 移动文件读写指针到 AL 指定的信息 ID 对应的
14906:  STC                             ;│ 数据信息标题
14907:  INT     21H                     ;┘
14908:  JC      Pro_Country4
14909:  PUSH    SI                      ;┐ CX←AL 指定的信息 ID 对应的数据信息长度
14910:  MOV     SI,208H                 ;│ (字节数)
14911:  MOV     CX,[SI]                 ;┘
14912:  POP     SI
14913:  MOV     DX,200H                 ;┐
14914:  ADD     CX,0AH                  ;│ 读取 AL 指定的信息 ID 对应的数据信息
14915:  MOV     AH,3FH                  ;│
14916:  STC                             ;│
14917:  INT     21H                     ;┘
14918:  JC      Pro_Country4            ;┐
14919:  CMP     AX,CX                   ;│ 若数据错误,则跳转
14920:  JNE     Pro_Country4            ;┘
14921:  MOV     AL,[SI+2]               ;AL←信息 ID 值
14922:  MOV     SI,208H                 ;SI 指向 AL 指定的信息 ID 对应的数据信息块
14923:  MOV     CX,[SI]                 ;┐
14924:  INC     CX                      ;│ CX=数据信息块长度
14925:  INC     CX                      ;┘
14926:  CMP     CX,5F8H                 ;┐ 若长度超长,则跳转
14927:  JA      Pro_Country4            ;┘
14928:  CALL    Update_DBCSInMSDOS1     ;若需要更换 MSDOS1 模块中的 DBCS 表的内容,则
                                        ;清除先前设定的 DBCS 表的内容
14929:  CMP     AL,1                    ;┐ 若不是设置全国家信息表,则跳转
14930:  JNE     Pro_Country8            ;┘
14931:  PUSH    WORD PTR ES:[DI+18H]    ;┐
14932:  PUSH    WORD PTR ES:[DI+1AH]    ;│ 保存"小写字符到大写字符"的转换程序的入口
14933:  PUSH    DI                      ;┘ 地址
14934:  PUSH    AX                      ;┐
```

```
14935:    MOV     AX, WORD PTR CS:CodePage      ;| 修改代码页号
14936:    MOV     [SI+4],AX                     ;|
14937:    POP     AX                            ;┘
14938: Pro_Country8:
14939:    REP     MOVSB
14940:    CMP     AL,1                          ;┐ 若不是设置全国家信息表,则跳转
14941:    JNE     Pro_Country9                  ;┘
14942:    POP     DI                            ;┐
14943:    POP     WORD PTR ES:[DI+1AH]          ;| 恢复"小写字符到大写字符"的转换程序的入口
14944:    POP     WORD PTR ES:[DI+18H]          ;┘ 地址
14945: Pro_Country9:
14946:    POP     SI                            ;┐
14947:    POP     CX                            ;|
14948:    POP     DI                            ;|
14949:    ADD     SI,[SI]                       ;| DS:SI 指向下一个信息 ID 对应的数据信息标
14950:    INC     SI                            ;| 题
14951:    INC     SI                            ;┘
14952:    DEC     CX                            ;┐
14953:    CMP     CX,0                          ;| 若整个扩充国家信息管理块搜索完,则直接返
14954:    JE      Pro_Country_RET               ;┘ 回
14955:    JMP     Pro_Country7
14956: Pro_Country_RET:
14957:    RET
14958: Process_Country  ENDP
14959: ;=============================================================
14960: ;相对地址偏移:3680
14961: ;功能:若更换 MSDOS1 模块中的 DBCS 表的内容,则清除先前设定的 DBCS 表的内容
14962: ;入口参数:AL=信息 ID 值
14963: ;          ES:DI 指向存放信息 ID 对应的数据信息的缓冲区
14964: ;出口参数:无
14965: ;=============================================================
14966: Update_DBCSInMSDOS1  PROC  NEAR
14967:    CMP     AL,7                          ;┐ 若信息 ID 值≠7(取 DBCS 表的地址),则直接返
14968:    JNE     Update_DBCSInMSDOS1_RET       ;┘ 回
14969:    CMP     WORD PTR ES:[DI],0            ;┐ 若 DBCS 表为空,则直接返回
14970:    JE      Update_DBCSInMSDOS1_RET       ;┘
14971:    PUSH    DI
14972:    PUSH    AX
14973:    PUSH    CX
14974:    MOV     CX,ES:[DI]                    ;CX←DBCS 表的长度(字节数)
14975:    ADD     DI,2                          ;┐
14976:    XOR     AL,AL                         ;| 清除先前设定的 DBCS 表的内容
14977:    REP     STOSB                         ;┘
```

```
14978:      POP     CX
14979:      POP     AX
14980:      POP     DI
14981: Update_DBCSInMSDOS1_RET:
14982:      RET
14983: Update_DBCSInMSDOS1  ENDP
14984: ;=============================================================
14985: ;相对地址偏移:369B
14986: ;功能:确定 AL 指定的信息 ID 对应的数据块在 MSDOS1 模块中的存放位置(扩充国家信息管理块
            请参考 MSDOS.SYS 的源程序)
14987: ;入口参数:AL=信息 ID 值;
14988: ;           ES:DI 指向位于 MSDO1 模块中的扩充国家信息管理块
14989: ;出口参数:CF=0,成功;CF=1,信息 ID 值不匹配
14990: ;           ES:DI 指向存放信息 ID 对应的数据块的内存区的起始地址
14991: ;=============================================================
14992: Get_AddrOfInfoID  PROC  NEAR
14993:      PUSH    CX
14994:      ADD     DI,4AH                  ;┐ CX←扩充国家信息管理块中允许的信息 ID 值
14995:      MOV     CX,ES:[DI]              ;┘ 个数
14996:      INC     DI
14997:      INC     DI                      ;ES:DI 指向第一个信息 ID 对应的数据块
14998: Get_Addr1:
14999:      CMP     ES:[DI],AL              ;┐ 若信息 ID 值匹配,则跳转
15000:      JE      Get_Addr5               ;┘
15001:      CMP     BYTE PTR ES:[DI],1      ;┐
15002: Get_Addr2:                           ;│
15003:      JZ      Get_Addr3               ;│
15004:      ADD     DI, +05                 ;│ ES:DI 指向下一个信息 ID 对应的数据块
15005:      JMP     Short Get_Addr4         ;│
15006: Get_Addr3:                           ;│
15007:      ADD     DI, +29H                ;│
15008: Get_Addr4:                           ;┘
15009:      LOOP    Get_Addr1
15010:      STC                             ;置失败标志,表示无对应的信息
15011:      JMP     Short Get_Addr7
15012: Get_Addr5:
15013:      CMP     AL,1                    ;┐ 若信息 ID 值不是 1(全国家信息),则跳转
15014:      JNE     Get_Addr6               ;┘
15015:      INC     DI                      ;ES:DI 指向存放"全国家信息"的缓冲区
15016:      JMP     SHORT Get_Addr7
15017: Get_Addr6:
15018:      LES     DI,DWORD PTR ES:[DI+1]  ;ES:DI 指向存放字符表(如大写表)的缓冲区
15019: Get_Addr7:
```

```
15020:     POP     CX
15021:     RET
15022: Get_AddrOfInfoID  ENDP
15023: ;=========================================================
15024: ;相对地址偏移:36C9
15025: ;功能:将文件读写指针移到从文件头开始的 CX:DX 位置处,然后读取 AX 个字节的文件内容
15026: ;入口参数:AX=要读取的字节数
15027: ;          CX:DX=起始位置
15028: ;          DS:0=存放国家信息文件内容的缓冲区的起始地址
15029: ;出口参数:AX=实际完成的字节数
15030: ;          DS:SI 指向存放国家信息文件内容的缓冲区
15031: ;=========================================================
15032: ReadFile  PROC  NEAR
15033:     PUSH    AX                          ;┐
15034:     MOV     AX,4200H                    ;|
15035:     STC                                 ;| 移动文件读写指针到 CX:DX 指定的位置
15036:     INT     21H                         ;|
15037:     POP     CX                          ;┘
15038:     JC      ReadFile_RET                ;
15039:     XOR     DX,DX                       ;┐
15040:     XOR     SI,SI                       ;| 从当前位置(即从文件头开始的 CX:DX 处)读
15041:     MOV     AH,3FH                      ;| 取 CX 个字节的文件内容
15042:     STC                                 ;|
15043:     INT     21H                         ;┘
15044: ReadFile_RET:
15045:     RET
15046: ReadFile  ENDP
15047: ;=========================================================
15048: ;相对地址偏移:36DD
15049: ;功能:更改命令处理器(即 Shell 模块)对应的文件名说明串
15050: ;入口参数:ES:SI 指向命令处理器的文件名说明串
15051: ;出口参数:无
15052: ;=========================================================
15053: Modify_PathOfShell  PROC  NEAR
15054:     PUSH    SI
15055:     PUSH    DS
15056:     PUSH    ES
15057:     POP     DS
15058:     POP     ES
15059:     CALL    Test_DriveSym               ;┐ 若文件名说明串中没有逻辑驱动器符,则跳转
15060:     JB      Modify_Path1                ;┘
15061:     MOV     AL,[SI]                     ;┐
15062:     INC     SI                          ;| 跳过逻辑驱动器符
```

```
15063:    INC     SI                              ;┘
15064:    JMP     SHORT Modify_Path2              ;
15065: Modify_Path1:
15066:    MOV     AH, 19H                         ;┐ 取出缺省的逻辑驱动器号(0=A,1=B,…)
15067:    INT     21H                             ;┘
15068:    ADD AL, ' A'                            ;AL←缺省的逻辑驱动器符
15069: Modify_Path2:
15070:    MOV     BYTE PTR CS:CountryFile, AL     ;设置"逻辑驱动器符"
15071:    MOV     DI, offset CountryFile+3        ;┐
15072:    MOV     AL, [SI]                        ;|
15073:    CMP     AL, ' \'                        ;|
15074:    JZ      Modify_Path3                    ;|
15075:    CMP     AL, ' /'                        ;|
15076:    JZ      Modify_Path3                    ;| 复制命令处理器的文件名说明串
15077:    JMP     SHORT Modify_Path4              ;|
15078: Modify_Path3:                              ;|
15079:    DEC     DI                              ;|
15080: Modify_Path4:                              ;|
15081:    CALL    Copy_String                     ;┘
15082:    MOV     DI, offset CountryFile          ;
15083:    PUSH    DS
15084:    PUSH    ES
15085:    POP     DS
15086:    POP     ES
15087:    POP     SI
15088:    RET
15089: Modify_PathOfShell  ENDP
15090: ;=============================================================
15091: ;相对地址偏移:3712
15092: ;功能:检查 DS:SI 指定的文件名说明串是否包含逻辑驱动器符
15093: ;入口参数:DS:SI 指向文件名说明串
15094: ;出口参数:CF=0:包含逻辑驱动器符;CF=1:不包含逻辑驱动器符
15095: ;=============================================================
15096: Test_DriveSym  PROC NEAR
15097:    PUSH    AX
15098:    CMP     BYTE PTR [SI], ' A'
15099:    JB      Test_DriveSym1
15100:    CMP BYTE PTR [SI], ' Z'
15101:    JA      Test_DriveSym1
15102:    CMP BYTE PTR [SI+1], ' :'
15103:    JNZ     Test_DriveSym1
15104:    JMP     SHORT Test_DriveSym2
15105: Test_DriveSym1:
```

```
15106:    STC
15107: Test_DriveSym2:
15108:    POP     AX
15109:    RET
15110: Test_DriveSym  ENDP
15111: ;=============================================================
15112: ;相对地址偏移:3729
15113: ;功能:复制以 0 结尾的字符串
15114: ;入口参数:DS:SI 指向源字符串;
15115: ;          ES:DI 指向存放目的字符中的缓冲区
15116: ;出口参数:无
15117: ;=============================================================
15118: Copy_String  PROC  NEAR
15119: Copy_Str1:                          ;┐
15120:    MOVSB                            ;| 复制字符串
15121:    CMP     BYTE PTR [SI-1],0        ;|
15122:    JNE     Copy_Str1                ;┘
15123:    RET
15124: Copy_String  ENDP
15125: ;=============================================================
15126: ;相对地址偏移:3731
15127: ;功能:显示错误信息
15128: ;入口参数:DX 指向错误信息说明串
15129: ;出口参数:无
15130: ;=============================================================
15131: Disp_ErrorInfo  PROC  NEAR
15132:    PUSH    CS                       ;┐
15133:    POP     ES                       ;| ES:DX 指向指定的错误信息
15134:    MOV     SI,DX                    ;┘
15135: Disp_ErrorStr  LABEL  NEAR          ;输出错误提示信息
15136:    MOV     DX, offset BadMissing    ;DX 指向错误信息"Bad or missing"
15137:    MOV     BX, offset OutputCRLF    ;BX 指向"回车换行字符串"
15138: Disp_CharStr1  LABEL  NEAR          ;输出 CS:DX、ES:SI 和 CS:BX 指定的字符串
15139:    PUSH    CS                       ;┐
15140:    POP     DS                       ;| 输出 CS:DX 指定的字符串
15141:    CALL    DisplayString            ;┘
15142: Disp_Info1:                         ;输出 ES:SI 和 CS:BX 指定的字符串
15143:    MOV     DL,ES:[SI]               ;┐
15144:    OR      DL,DL                    ;| 若欲输出的字符串已结束,则跳转
15145:    JZ      Disp_Info2               ;┘
15146:    MOV     AH,2                     ;┐ 将 DL 指定的字符输出到标准输出设备
15147:    INT     21H                      ;┘
15148:    INC     SI                       ;修改输出字符串的指针
```

```
15149:    JMP     SHORT Disp_Info1
15150: Disp_Info2:                          ;输出 CS:BX 指定的字符串
15151:    MOV     DX,BX
15152: Disp_CharStr2:
15153:    CALL    DisplayString             ;输出 DS:DX 指定的字符串
15154:    CMP     CS:ErrorCtrlFlag,01       ;┐若不输出错误系统配置命令在 CONFIG.SYS 文
15155:    JE      Disp_CharStr_RET          ;┘本文件中的位置信息,则直接返回
15156:    CALL    OutputErrSiteInfo         ;输出错误系统配置命令在 CONFIG.SYS 文本文件
                                            ;中的位置信息
15157: Disp_CharStr_RET:
15158:    RET
15159: Disp_ErrorInfo  ENDP
15160: ;=============================================================
15161: ;相对地址偏移:375F
15162: ;功能:显示 DS:DX 指定的以 $ 结尾的字符串
15163: ;入口参数:DS:DX 指向以 $ 结尾的字符串
15164: ;出口参数:无
15165: ;=============================================================
15166: DisplayString  PROC  NEAR
15167:    MOV     AH,9
15168:    INT     21H
15169:    RET
15170: DisplayString  ENDP
15171: ;=============================================================
15172: ;相对地址偏移:3764
15173: ;功能:试图打开 DS:DX 指定的设备。若 DS:DX 指定一个磁盘文件,或者指定的设备/磁盘文件不
           存在,则打开 NUL 设备
15174: ;入口参数:DS:DX 指向设备名或磁盘文件的文件名说明串
15175: ;出口参数:BX=文件句柄
15176: ;=============================================================
15177: OpenDevice  PROC  NEAR
15178:    CALL    OpenFile                  ;┐打开 DS:DX 指定的文件,若打开成功,则跳转
15179:    JNC     OpenDevice2               ;┘
15180: OpenDevice1:
15181:    MOV     DX, offset NUL_Device     ;┐打开 NUL 设备
15182:    CALL    OpenFile                  ;┘
15183: OpenDevice_RET:
15184:    RET
15185: OpenDevice2:
15186:    MOV     BX,AX                     ;┐
15187:    XOR     AX,AX                     ;├取设备信息→DX
15188:    MOV     AH,44H                    ;│
15189:    INT     21H                       ;┘
```

```
15190:    TEST    DL,80H                          ;┐ 若打开一个设备,则直接返回
15191:    JNZ     OpenDevice_RET                  ;┘
15192:    MOV     AH,3EH                          ;┐ 关闭刚才打开的磁盘文件
15193:    INT     21H                             ;┘
15194:    JMP     ShortOpenDevice1                ;转去打开 NUL 设备
15195: OpenDevice  ENDP
15196: ;=============================================================
15197: ;相对地址偏移:3783
15198: ;功能:试图打开 DS:DX 指定的一个设备或一个磁盘文件
15199: ;入口参数:DS:DX 指向设备名或磁盘文件的文件名说明串
15200: ;出口参数:CF=0,打开成功,AX=文件句柄; CF=1,打开文件失败,AX=错误码
15201: ;=============================================================
15202: OpenFile  PROC  NEAR
15203:    MOV     AH,3DH                          ;┐
15204:    STC                                     ;| 打开 DS:DX 指定的设备或磁盘文件
15205:    INT     21H                             ;┘
15206:    RET
15207: OpenFile  ENDP
15208: ;=============================================================
15209: ;相对地址偏移:3789
15210: ;功能:INT 24H 中断服务程序
15211: ;入口参数:无
15212: ;出口参数:AL=返回的错误处理回答码 3(放弃出错处理)
15213: ;=============================================================
15214: INT24H_ISR  PROC  FAR
15215:    MOV     AL,3                            ;AL←返回的错误处理回答码
15216:    IRET
15217: INT24H_ISR  ENDP
15218: ;相对地址偏移:378C
15219:    DB      ' MS DOS Version 5.00 (C)Copyright 1981—1991 Microsoft '
15220:    DB      ' Corp Licensed Material — Property of Microsoft All rights '
15221:    DB      ' reserved '
15222: ;相对地址偏移:3804
15223: NUL_Device  DB  ' NUL' ,0                   ;┐
15224: CON_Device  DB  ' CON' ,0                   ;| DOS 操作系统使用的标准设备名字符串
15225: AUX_Device  DB  ' AUX' ,0                   ;|
15226: PRN_Device  DB  ' PRN' ,0                   ;┘
15227: ConfigFile    DB  ' \CONFIG.SYS' ,0         ;系统配置文件 CONFIG.SYS 的文件名说明串
15228: ;相对地址偏移:3820
15229: CountryFile  DB  ' A:\COUNTRY.SYS' ,0      ;┐ 存放 country 系统配置命令指定的国家信息文
15230:    DB      52 DUP (0)                      ;┘ 件的文件名说明串
15231: ;相对地址偏移:3863
15232: MarkOfCountry    DB  -1,' COUNTRY'         ;国家信息文件的标志字符串
```

```
15233: CodePage          DW 0                    ;存放当前代码页号
15234: ;相对地址偏移:386D
15235: Path_Shell        DB ' \COMMAND.COM'  ;┐ 存放 shell 系统配置命令指定的命令处理器的文
15236:     DB      116 Dup(0)   ;21 DUP (0)    ;┘ 件名说明串
15237: ;相对地址偏移:38ED
15238: ;DOS 操作系统支持的系统配置命令表,它由 17 个子项和一个结束标志组成,每个子项的格式如
       下:
15239: ;1. 系统配置命令名的字符长度                ;
15240: ;2. 系统配置命令名                        ;
15241: ;3. 系统配置命令名对应的缩写关键字符
15242: TableOfConfigCMD  Label  BYTE
15243:     DB      7
15244:     DB      ' BUFFERS'
15245:     DB      ' B'
15246:     DB      5
15247:     DB      ' BREAK'
15248:     DB      ' C'
15249:     DB      6
15250:     DB      ' DEVICE'
15251:     DB      ' D'
15252:     DB      0AH
15253:     DB      ' DEVICEHIGH'
15254:     DB      ' U'
15255:     DB      5
15256:     DB      ' FILES'
15257:     DB      ' F'
15258:     DB      4
15259:     DB      ' FCBS'
15260:     DB      ' X'
15261:     DB      9
15262:     DB      ' LASTDRIVE'
15263:     DB      ' L'
15264:     DB      0AH
15265:     DB      ' MULTITRACK'
15266:     DB      ' M'
15267:     DB      8
15268:     DB      ' DRIVPARM'
15269:     DB      ' P'
15270:     DB      6
15271:     DB      ' STACKS'
15272:     DB      ' K'
15273:     DB      7
15274:     DB      ' COUNTRY'
```

```
15275:    DB      ' Q'
15276:    DB      5
15277:    DB      ' SHELL'
15278:    DB      ' S'
15279:    DB      7
15280:    DB      ' INSTALL'
15281:    DB      ' I'
15282:    DB      7
15283:    DB      ' COMMENT'
15284:    DB      ' Y'
15285:    DB      3
15286:    DB      ' REM'
15287:    DB      ' 0'
15288:    DB      8
15289:    DB      ' SWITCHES'
15290:    DB      ' 1'
15291:    DB      3
15292:    DB      ' DOS'
15293:    DB      ' H'
15294:    DB      0                                   ;系统配置命令表的结束符
15295: ;相对地址偏移:397E
15296: Packet              Label   Byte
15297: SpecialFunctions    DB      0                   ;┐
15298: DeviceTypeByte      DB      2                   ; |
15299: DeviceAttributes    DW      0                   ; |
15300: NumberOfCylinders   DW      50H                 ; |
15301: MediaType           DB      0                   ; |
15302: ;Device BPB   ;设备 BPB 参数块的起始地址       ; |
15303: BytesPerSector      DW      0                   ; |
15304: SectorsPerCluster   DB      0                   ; |
15305: ReservedSectors     DW                          ; | 取置设备参数的参数块,它供 DOS 功能调用
15306: NumberOfFAT         DB      0                   ; | 44H(设备输入/输出控制)使用
15307: RootEntries         DW      0                   ; |
15308: TotalSectors        DW      0                   ; |
15309: MediaDescriptor     DB      0                   ; |
15310: SectorsPerFAT       DW      0                   ; |
15311: SectorsPerTrack     DW      0                   ; |
15312: Heads               DW      0                   ; |
15313:    DB      14    Dup(0)                         ; |
15314: ;Track Layout   ;设备的磁道布局表的起始地址    ; |
15315: SectorCount         DW      0                   ; |
15316:    DB      252    Dup(0)                       ;┘
15317: ;End of Packet
```

```
15318: ;相对地址偏移:3AA2
15319: HeadsOfDrivParm    DW    2              ;存放"drivparm 系统配置命令的"/H"开关指定的
                                               ;磁头数
15320: SecPerTrackOfDrivParm  DW  09          ;存放"drivparm 系统配置命令的"/S"开关指定的每
                                               ;道扇区数
15321: DriveOfDrivParm    DB      0           ;存放"drivparm 系统配置命令的"/D"开关指定的
                                               ;物理驱动器号(0=A,1=B,…)
15322: MarkOfControlSym   DW      0           ;"drivparm"系统配置命令的开关参数的标志位字
15323: ;相对地址偏移:3AA9
15324: DS_BPB             DW      512         ;┐
15325:     DB      2                          ;|
15326:     DW      1                          ;|
15327:     DB      2                          ;|
15328:     DW      112                        ;|
15329:     DW      2D0H                       ;| 单面/双面双密度(160KB/180KB 或 320KB/
15330:     DB      0FDH                       ;| 360KB、5.25 英寸)的软盘驱动器的 BPB 参数块
15331:     DW      2                          ;|
15332:     DW      9                          ;|
15333:     DW      2                          ;|
15334:     DD      0                          ;|
15335:     DD      0                          ;┘
15336: ;相对地址偏移:3AC2
15337: HD_BPB  DW      512                    ;┐
15338:     DB      1                          ;|
15339:     DW      1                          ;|
15340:     DB      2                          ;|
15341:     DW      224                        ;|
15342:     DW      960H                       ;| 1.2MB(5.25 英寸)双面四倍密度的软盘驱动器
15343:     DB      0F9H                       ;| 的 BPB 参数块
15344:     DW      7                          ;|
15345:     DW      15                         ;|
15346:     DW      2                          ;|
15347:     DD      0                          ;|
15348:     DD      0                          ;┘
15349: ;相对地址偏移:3ADB
15350: FD720_BPB    DW    512                 ;┐
15351:     DB      2                          ;|
15352:     DW      1                          ;|
15353:     DB      2                          ;|
15354:     DW      112                        ;| 720KB(3.5 英寸)双面双密度软盘驱动器的
15355:     DW      5A0H                       ;| BPB 参数块
15356:     DB      0F9H                       ;|
```

```
15357:      DW      3                               ;|
15358:      DW      9                               ;|
15359:      DW      2                               ;|
15360:      DD      0                               ;|
15361:      DD      0                               ;┘
15362:  ;相对地址偏移:3AF4
15363:  FD144_BPB   DW    512                       ;┐
15364:      DB      1                               ;|
15365:      DW      1                               ;|
15366:      DB      2                               ;|
15367:      DW      224                             ;|
15368:      DW      0B40H                           ;| 1.44MB(3.5英寸)双面四倍密度的软盘驱动器
15369:      DB      0F0H                            ;| 的BPB参数块
15370:      DW      9                               ;|
15371:      DW      18                              ;|
15372:      DW      2                               ;|
15373:      DD      0                               ;|
15374:      DD      0                               ;┘
15375:  ;相对地址偏移:3B0D
15376:  FD288_BPB   DW    512                       ;┐
15377:      DB      2                               ;|
15378:      DW      1                               ;|
15379:      DB      2                               ;|
15380:      DW      240                             ;|
15381:      DW      1680H                           ;| 2.88MB(3.5英寸)的软盘驱动器的BPB参数
15382:      DB      0F0H                            ;| 块
15383:      DW      9                               ;|
15384:      DW      36                              ;|
15385:      DW      2                               ;|
15386:      DD      0                               ;|
15387:      DD      0                               ;┘
15388:  ;相对地址偏移:3B26                           ;软盘驱动器BPB参数块起始地址表
15389:  AddrOfBPB     DW      DS_BPB, HD_BPB, FD720_BPB
15390:      DW      FD720_BPB, FD720_BPB, FD720_BPB
15391:      DW      FD720_BPB, FD144_BPB, FD720_BPB
15392:      DW      FD288_BPB
15393:  NumberOfSwitchSym     DB    8               ;为drivparm系统配置命令允许的开关参数符的个
                                                    ;数
15394:  TableOfSwitchSym      DB    ' FHSTDICN' ,0  ;drivparm系统配置命令允许的开关参数符表
15395:  UnknownCMD           DB    CR, LF,' Unrecognized command in CONFIG.SYS'
15396:  OutputCRLF           DB    CR, LF,' $'
15397:  InvalidPara          DB    0DH, 0AH, ' Bad command or parameters - $'
15398:  BigSectorSize        DB    CR, LF, ' Sector size too large in file $'
```

```
15399: BadMissing              DB    CR, LF, ' Bad or missing $'
15400: CMD_Inter               DB    ' Command Interpreter' ,0
15401: InvalidCodePage         DB    CR, LF, ' Invalid country code or code page' ,CR, LF,' $'
15402: Err_CountryCMD          DB    CR, LF, ' Error in COUNTRY command' , CR, LF, ' $'
15403: Err_CountryMem          DB    CR, LF, ' Insufficient memory for COUNTRY.SYS file' ,CR, LF,'
$'
15404: Err_Memory              DB    CR, LF, ' Configuration too large for memory' ,CR, LF,' $'
15405: Err_BlockDev            DB    CR, LF, ' Too many block devices' , CR, LF, ' $'
15406: InvalidSTPara           DB    CR, LF, ' Invalid STACK parameters' ,CR, LF, ' $'
15407: IncorrectOrder          DB    0DH, 0AH, ' Incorrect order in CONFIG.SYS line $'
15408: Err_Config              DB    ' Error in CONFIG.SYS line $'
15409: Err_LogicalDrive        DB    ' WARNING! Logical drives past Z: '
15410:     DB       ' exist and will be ignored' , 0DH, 0AH, ' $'
15411:     DB       0, 0, 0, 0
15412: End_SysInt_II       EQU   $
15413: ;
15414: IO3   ENDS
15415:       END
15416:
15417:
```

§10.5 IO.SYS 源程序索引

10.5.1 IO.SYS 源程序的过程名索引

过程名	定义行号	引用行号
AllocateBufferMem	9976	9935
AllocateMaxFreeUMB	13791	13746,13937
Allocate_Stack	8230	8215
AppendNewBDPB	7363	6749
ApplyAllAvailableMem	8826	8695
ApplyMaxUMB	13378	13334
App_New_Node	3481	2496,2517, 3422
AUX_Inp_Flush	4424	3985

过程名	定义行号	引用行号
AUX_Intr	973	618
AUX_ND_RD	4373	3883
AUX_Outp_Status	4396	3988
AUX_RD	4332	3982
AUX_WR	4438	3986
AUX_Write	4437	
AUX_WR_Vertify	4439	3987
BCDtoBIN	1848	1829,1832,1835,1838
BINtoBCD	1601	788, 2599, 2600
BlkDevIntrEntry	1248	6740
BuildCDB	1434	1409
Build_BDPB_DefaultBPB	3213	2689
Build_BPB	4781	4645
Build_ExtendedDisk_BDPB	3336	2502
BulidBufferChain	9933	9537, 9726
Calcuate_DayCount	1710	1692, 3537
CalculateAccumulateCode	9895	9816, 9876
CalculateClockCount	4472	
CalculateDecimalValue	11248	11115
CalculateHexValue	13718	13685
Calculate_Bytes	6193	6160

过程名	定义行号	引用行号
CallSubInIO2	2084	2640,2643,2657,3008,3069,3144,3731,3770,3792
CALL_INT13H	5654	5533,5578,5641
CALL_ROMBIOS	1517	
Capital	14645	14326
ChangeCharToCap	11029	11009,11341
ChangedTerminator	13536	12464
ChangeStrToCaps	10999	10878
CheckA20	1109	1042, 1065, 1084
CheckLogicDriveNum	3464	2494,2515,3383
Check_FMTStatus	6256	5942,5960,6009
ChkDriver	13572	12547,12577
ChkHIMEMMark	9244	9048
ChkHMA	9024	8888
ChkMediaRemove	7449	4711
ChkTerminator	11581	10557,11552
ChkWithMediaDescript	7520	7578
Chk_PROTMIN $	13290	13236
ClearHIMEMMark	9194	8578
ClearStartingHMA	1224	1188, 1213
Clear_Intrrupt	3611	3602,3639
Clear_ModifyBPBFlag	7693	5004,7721

过程名	定义行号	引用行号
CLOCK $ _Intr	990	634
CLOCK $ _RD	4574	3996
CLOCK $ _WR	4522	4000
CLOCK $ _Write	4521	
CLOCK $ _WR_Vertify	4523	4001
COM1_Intr	974	656
COM2_Intr	978	691
COM3_Intr	982	699
COM4_Intr	986	707
ConvertAddrFMT	13554	12597,13582
ConvertConfig	14323	12197
ConvertData	1672	786,2603,2604
ConvertToParagraphs	9290	8464,8468,8488,8500,8791,8952,8969
Convert_Addr	9851	9311,9575,9623,12128,13556,14676
Convert_AddrFMT	14673	9395,9422,9429,9454,9532,9542,9553,9555, 9578,9594,9605,9626,9729,9734,9966,12433
CON_Inp_Flush	4136	7421
CON_Intr	954	610
CON_ND_RD	4064	3940,4089
CON_ND_Read	4063	
CON_RD	4009	3939
CON_WR	4118	3943

过程名	定义行号	引用行号
CON_Write	4117	
CON_WR_Vertify	4119	3944
CopyDOSFarCallToIO2	9270	8717
Copy_String	15118	12693,12737,15081
CreateDriverSCB1	13328	12436
CreateDriverSCB2	13417	13346
CreateEntireUMBChain	13742	12504
CreateSCB	10468	9401,9431,9534,9558,9596,9718
CreateUMBChain1	13761	13743
CreateUMBChain2	13936	13773
Create_A_Buffer	10002	9946
Create_CDS	9303	8644,12132,12624
Create_ControlBlock	9382	8702
CurrAvailableMem	13234	12485
CutTailChar	10968	10883
DataError	14796	
DetermineDriverParagraphs	13473	12434
DisableA20	1132	
Disk_19H	6619	4668
Disk_Build_BPB	4784	4645
Disk_Dev_Close	5150	4657

过程名	定义行号	引用行号
Disk_Dev_Open	5135	4656
Disk_Generic_IOCTL	5766	4662
Disk_Get_Map	6398	4666
Disk_Init	7350	4643
Disk_Intr	993	642,1249
Disk_Map	6397	
Disk_Media_Check	4693	4644
Disk_ND_RD	5180	4648,5170
Disk_NOP_Sub	4736	4649,4650,4653,4654,4659,4660,4661,4663,4664,4665
Disk_Operate1	5455	5373
Disk_Operate2	5495	5462,5477,6138,6157
Disk_Operation	5301	5203,6548
Disk_Read	5290	3730,3791,5208
Disk_Rem_Media	5167	4658
Disk_Set_Map	6411	4667
Disk_WR	5198	4651
Disk_WR_RD	5196	
Disk_WR_Vertify	5197	4652
DisplayString	495	478,8671,12520,12768,12789,12862,12899,13081,13086,13103,13117,13148,13161,13195,14702,15141,15153

过程名	定义行号	引用行号
Disp_CharStr1	15138	12652
Disp_ErrorInfo	15131	8816,9649,12173
Disp_ErrorStr	15135	9832,12449,12756
Disp_Insert_FDisk	7393	5263
Do_INT13H	6354	5989,5999
DummyISR	937	2252,2321
Enable1_A20	1083	1012,1229
Enable2_A20	1094	1044,1067
ExecuteDOSFunction	9867	9725,9726,9882
FormatTrack	5934	5746
GetChar	4354	4383
GetChrBuf	4461	4334, 4374, 4425
GetChrStatus	4413	4356,4445
GetConfigChar	14547	14334, 14387, 14395, 14421, 14429, 14436, 14446,14459,14471,14493,14500
GetCurrParaValue	13053	12263, 12306, 12336, 12374, 12665, 12823, 12859,12952,12992
GetHexSizePara	13671	13638
GetIndicateParaValue	10523	13063
GetInstallStateOfHIMEM	9069	9028,13762
GetIntegerParaValue	11079	10929
GetMaxLogicDrivePara	11508	10937
GetNextUMBSeg	13917	13829,13892,13896,14019

过程名	定义行号	引用行号
GetNotSwitchParaValue1	10683	10608
GetNotSwitchParaValue2	10895	10704,10798
GetPlusIntegerValue	11103	10922,11092
GetPortStatus	4411	4378,4397
GetPRNStatus	4211	4171,4198,4255
GetSwitchParaValue	10773	10627
GetSwitchStatusPara	11267	10951
Get_68H	6649	5741
Get_AccessState	6586	5740
Get_AddrOfInfoID	14992	14887
Get_BDPB_Node	4675	2638,2655,3768,4694,4786,5153,5168,5304,5767,6399,6412
Get_BPB_Para	4988	4855
Get_Char	4026	4012
Get_CharOfConfig	13130	12194, 12198, 12207, 12218, 12918, 12928, 12931, 12938, 14116, 14158, 14182, 14186, 14630,14634,14761
Get_Data	14746	14189
Get_DevicePara	5803	5733
Get_DrivParmPara	14209	14192
Get_ExtMediaState	6463	5739
Get_NextCluster	369	250, 396, 403, 410
Get_OFS_FreeHMA	6829	6785, 6802

过程名	定义行号	引用行号
Get_RTCPara	3583	3568
Get_VolumeID	7741	7532, 7718
IllegalPara	10715	10655
Init_CDS	9322	9372,9581
Init_COM_Adapter	3322	2225,2227,2229,2231
Init_ExtendedDisk_BDPB	3373	3357
Init_FixedBDPB	3434	3419
Init_PRN_Adapter	3309	2234,2236,2238
Init_RTC	3555	3535
Install1_IO2_MSDOS2_InHMA	8870	8847,9092,12508
Install2_IO2_MSDOS2_InHMA	8887	8871
Install_IO2	8846	8892, 8909
Install_IO2_MSDOS2	8846	8713
Install_IO2_MSDOS2_InCMA	8906	8690,8857
Install_MSDOS2	8944	8894,8911
INT02H_Entry	7958	10085
INT08H_Entry	7981	10090
INT09H_Entry	7990	10095
INT0AH_Entry	8010	10127
INT0BH_Entry	8026	10158
INT0CH_Entry	8042	10189

过程名	定义行号	引用行号
INT0DH_Entry	8058	10220
INT0EH_Entry	8074	10251
INT13H_ILR	1371	
INT13H_ILR1	1371	1395, 2558
INT13H_ILR2	1572	2585
INT13H_ISR	1039	2133,6981,7039,7059,7099,7114,7146,7210
INT15H_ISR	1196	2135
INT19H_ISR	1145	2137
INT1BH_ISR	935	2247
INT24H_ISR	15214	8665
INT29H_ISR	1019	2249
INT6CH_ISR	1686	2622, 2623
INT70H_Entry	8002	10100
INT72H_Entry	8090	10282
INT73H_Entry	8106	10313
INT74H_Entry	8122	10344
INT76H_Entry	8138	10375
INT77H_Entry	8154	10406
InvalidCmd	3922	3909, 3981
InvalidSubCmd	5792	5736,5737,5738,5747,5748,5749
IO1_INT13H_ISR	1055	6981,7039,7059,7099,7114,7146,7210

过程名	定义行号	引用行号
IO1_Interrupt	953	
IO2_INT13H_ISR	7048	
IO2_INT2F_ISR	6690	
IO2_Interrupt	3848	
IO_Loader	37	140,147,235
JoinAllNeighborFreeUMB	13883	13751
JoinFreeUMBToUMBChain	13817	13748
Link_Old_INT13H	6966	7196,7259,7298,7311,7332
Link_PreviousISR	8170	7973,7982,7994,8003,8017,8033,8049,8065, 8081,8097,8113,8129,8145,8161
LoadDriver	13518	12455
LPT1_Intr	961	664
LPT2_Intr	965	672
LPT3_Intr	969	683
ModifyMemSize	13252	13242
ModifySegAddr	3833	
ModifyUMBSize	13985	8698,13333
Modify_PathOfShell	15053	
Modify_Vector	10445	10086, 10091, 10096, 10101, 10128, 10159, 10190,10314,10345,10376,10407
MultiDriveName	5217	5343,5820,5971,6108,6414,7450
OmitAllSeparator	13655	13629,13641
OmitCommentStr	14568	14332,14419,14427,14434,14491,14498

过程名	定义行号	引用行号
OmitSeparators	11549	10534,10563
Omit_A_ConfigCMD	14628	12215
OpenDevice	15177	9665,9668
OpenFile_	15202	15178,15182
OutputErrInfo	13096	12265, 12286, 12308, 12338, 12376, 12393, 12802,12825,12954,12994
OutputErrLineNum	13171	13149,13162
OutputErrSiteInfo	13157	9840, 12528, 12769, 12790, 12863, 12900, 13082,13087,13118,15156
PointToNext	10756	10737
PRN_19H	4309	3974
PRN_Generic_IOCTL	4279	3968
PRN_Intr	957	626
PRN_Output	4164	
PRN_Outp_Busy	4244	3965
PRN_Outp_Status	4197	3959
PRN_RD	4152	3953
PRN_Request	4213	4175,4263
PRN_WR	4165	3957
PRN_WR_Vertify	4166	3958
ProcessHexSizePara	13625	12391
Process_Config	12102	8681
Process_Country	14813	12742

过程名	定义行号	引用行号
Process_DevTrack	6079	
Process_DrivParm	14096	12842
Process_Install	9780	12240
Process_RemMedia	7571	5593
Process_RTC	3511	2336
Process_Stack	10019	9627
Process_SwitchPara	14157	14112
Pro_Config4	12182	8697,8701,8709
Pro_Config97	12781	12667
PublicIntrProcess	1001	955, 958, 962, 966, 970, 975, 979, 983, 987, 991, 994
PUB_ND_RD	4080	3954,3997,4202,4388
PUB_NOP_Sub	4106	3935, 3949, 3969, 3971, 3972, 3973, 3978, 3992, 3993, 3999, 4120
ReadASectorofFAT	3750	3674,3683,3706
ReadBootSector	6538	6467,6499,6523
ReadClockCount	4624	4575,4754,5700
ReadDisk	288	226,268,456
ReadFATTable	436	383,391,417
ReadFile	15032	14820,14839,14877
Read_BootstrapSector1	2759	2805,3355
Read_BootstrapSector2	4937	4851
Read_ClockCount	1795	

过程名	定义行号	引用行号
Read_DeviceTrack	6080	5734
Read_Disk	5067	4940,5051,7777
Read_MSDOS	3651	2680
Read_SecondSector	5048	4860
Rebuild_BPB	7493	5346
RecoverMemSize	13271	12497
Recover_INT1EH	6434	5891
RedefineDriveLogicName	2706	2530
RequestDrive	14711	12496
ResetDiskControler	5720	5081, 5600
Reset_Drive	6334	5584, 5587, 5721, 6004
RestartOS	474	
RetfEntry	1250	
Retry_INT13H	5631	5572
ROMBIOS1	1521	1457
ROMBIOS2	1529	1480
ROMBIOS3	1537	1484
ROMBIOS4	1545	1497
ROMBIOS5	1553	1506
ROMBIOS6	1561	1405
Save_dos_Para	13207	12381

过程名	定义行号	引用行号
Seek_BDPB	6948	6882,7238
Seg_IO2_MSDOS2_InLoad	8964	8907
SendCDB	1477	1412
SetAvailableMemSeg	13312	12486
SetEndFlagOfCMC	14010	8699
SetErrorDoneBit	3924	4153,5206
SetHIMEMMark	9132	8741
SetParaInfoBlk	10827	10700, 10808, 10908, 11214, 11316, 11475, 11532
SetSizeOfSCB	13597	12641,12645
Set_AccessState	6603	5751
Set_BDPBDefExtMediaState	4817	3142,4789
Set_BDPBExtMediaState	5015	3006, 3067, 4998, 6520
Set_BDPB_DevInfo	7661	6575,7005,7082
Set_BytesPerSector	6178	5917
Set_CodeOfDriver	5674	4741,4923, 5618, 7506
Set_DASDType	6209	5968,6012
Set_DevicePara	5843	5744,12844
Set_ExtMediaState	6493	5750
Set_LastVolumeID	7612	4733
Set_LastVolumeID_1	7614	7601
Set_MediaChangedCode	4752	4720, 7471

过程名	定义行号	引用行号
Set_OFS_FreeHMA	9221	8895
Set_Packet	14079	12845
Set_PacketPara	14245	14132
Set_RebuildBPBFlag	6560	6464, 6494
Set_Recover_Para	5422	5374,6167
Set_ReturnPara	7599	7504,7585
Set_RTCPara	3621	3566, 3572
ShiftInterface	1246	585
Strategy	944	609,617,625,633,641,655,663,671,682,690,698,706
SUB_08D1	1271	1272,7397
SysInt_II_Entry1	8297	9603
SysInt_II_Entry2	8482	8475
SysInt_I_Entry	2100	581
Temp1	5278	
TestDBCSBootByte	11700	10577,10982,11005,11339,11452,11514
TestKeywordMatch	11331	10735,11297
TestLoadDriverInUMB	13359	13331
TestParaLegal	10726	10775
TestParaOverflow	11227	11121,11127,11131,11135,11139
TestParaSeparator	14609	14422, 14494
TestReservedChar	11486	11428, 11450

过程名	定义行号	引用行号
TestSeparator	11620	10559, 11554
TestSwitchPreface	11674	10555
TestTerminatorOrSeparator	14598	12404, 13635, 13658, 13682, 14372, 14400, 14430,14440
Test_BCDData	1913	1734, 1808
Test_ChangeLineStatus	7703	4713, 6568, 7452, 7527, 7716
Test_DayData	1862	1737
Test_DevOpCount	7439	7494,7574
Test_DriveSym	15096	15059
Test_ModifyBPBFlag	7683	7454, 7496
Test_Numeric	14729	14750
Test_TimeData	1893	1810
Time_To_ClockCount	1827	1740,1812
Track_Operation	6106	6066
TransferData	6848	7159, 7228, 7290,7319
Update_BuildBPB	7631	4864
Update_ClockCount	5698	5124, 5425
Update_DBCSInMSDOS1	14966	14928
Update_FPT_Para	5385	5348, 6113
VerifyParaStr	11420	10936,10941
VerifyTotalSector	3175	2957
VerifyTrack	6035	5735

过程名	定义行号	引用行号
Verify_INT13H_Para	6877	7112,7269,7297,7310,7331
VolumeIDtoBDPB	7713	4798
Write_DeviceTrack	6083	5745

10.5.2 IO.SYS 源程序的变量名索引

变量名	定义行号	引用行号
AccumulatedCode	8357	9817,9877
AddressField	811	5908,5976,5977,5985,5986,5996,6047,6117
Address_ListOfList	8301	8530,8532,8570,8986,9304,9406,9436,9475, 9526,9560,12474,12542,12583,12608,12626
AddrOfBPB	15389	14256
AddrOfBuffer_SEG	63	139,371
Addr_ExtCIMB	8304	8534,8536,12734
Addr_MSDOS	8302	8494,8497,8499,8501,8524,8537,8684,8692, 8858,8873,8947,8950
AllowHMA	589	1002,1040,1063,1186,1211
AUX_Buffer	593	4463
AUX_CMDNum	3976	976,978,984,988
AUX_CMDTab	3978	
AUX_Device	15225	9663
AUX_DriverHeader	614	606
AXValue	599	7095,7102,7164
BadMissing	15399	15136

变量名	定义行号	引用行号
BDPB_Head	722	2354,2707,2723,3215,3484,4676,5228,6401,6755,6949,7664
BigSectorSize	15398	12650
BootDriveNo	58	68,275,325,338,2279,2637,2649,2733,2743,3722,3767,3783,
BootMediaDescriptor	1943	2648
BPB_Ptr_Array	894	2003,7352
BytesOfConfig	8375	12189,14556
BytesPerCluster	43	163,219,234
BytesPerSector	51	88,135,270,356,441,458,3281,9495,14257
BytesPerSector_Init	1950	2662,3758
CannotAllocateMem	8838	9012
CCIB_1	11999	11995
CCIB_B	11759	11755
CCIB_C	11809	11805
CCIB_H	12035	12031
CCIB_K	11947	11943
CCIB_L	11930	11926
CCIB_M	11982	11978
CCIB_Q	11836	11832
CCIB_X	11895	11891
Century	779	1605,1609,1653
ChangeLine	646	5856

变量名	定义行号	引用行号
CharsOfPreface	8369	14458,14469,14572,14577
CLOCK $ _CMDNum	3990	992
CLOCK $ _CMDTab	3992	
CLOCK $ _DriverHeader	630	622
ClockFlag	921	1723,1733,1736,1739,1754,1761,1767,1772,1780,1782
ClusterNo	46	387,394,395
ClusterNo_Init	1951	3679,3686,3687
CMDLine	8332	8745,12914,12936
CMD_Inter	15400	8815
CodePage	11867	14873,14935
COM1_DriverHeader	652	638
COM2_DriverHeader	687	679
COM3_DriverHeader	695	687
COM4_DriverHeader	703	695
CompatibleFDiskFlag	800	7017
ConfigCount	8374	12126, 12137, 12157, 12979, 13132, 14324, 14555,14782
ConfigFile	15227	12108,12172
CON_Buffer	588	936,4030,4065,4137
CON_CMDNum	3933	956
CON_CMDTab	3935	
CON_Device	15224	9643

变量名	定义行号	引用行号
CON_DriverHeader	606	2305
CountryCode	11866	12668,12678,12701,12722
CountryFile	15229	12659, 12692, 12709, 12711, 12714, 12728, 12749,12751,12754,15070,15071,15082
CPIB_1	11995	12988
CPIB_B	11755	12259
CPIB_C	11805	12302
CPIB_F	11869	12796
CPIB_H	12031	12370
CPIB_K	11943	12855
CPIB_L	11926	12819
CPIB_M	11978	12332
CPIB_Q	11832	12661
CPIB_X	11891	12948
CurrDriverParagraphs	12072	12438,13369,13385,13490,13502
CurrentCluster	53	194,231,252,374,400
Cylinder	742	5508,5560
Cylinders_Init	1955	2465
Cylinder_FMT	746	6379
DayPerMonth	781	1639
DefaultVolID	1307	7751

变量名	定义行号	引用行号
DefaultVolume	794	4823
DeviceAttributes	15299	14085
Device_No	602	4214,4414,4462
DiskBuffer	753	2910,2913,2923,2972,2974,2982,3014,3179,3186,4943,4945,4947,4949,4952,4956,4961,4963,4966,4989,4990,4991,4992,4993,4994,4995,5016,5019,5021,5026,5030,5071,6469,6471,6475,6502,6504,6510,6544,7141,7158,7208,7227,7287,7307,7318,7525,7530,7549,7552
DiskMedia	726	7875
DiskNumber	645	5524,5567
Disk_CMD0DH	4656	
Disk_CMDNum	4641	3870
Disk_CMDTab	4643	
Disk_DriverHeader	638	630
DispLineNumFlag	8396	12526,12529
DOS_Install_Site	8300	8682, 8711, 8715, 8739, 8861, 8876, 9090, 12506,13222,13225
DRH_Ptr	591	945, 946, 1006, 3865, 3894, 3925, 4073, 4104, 4280,4310,4726,4803,5769,5804,5844,5935, 6037,6087,6423,6473,6511,6587,6604,6621, 6650,6736,6737,7616
DriveNo_Init	1949	2848,2907
DriveNumber_Boot	8319	8583,8674
DriveNumber_Init	2001	3354,3394,3412
DriveOfDrivParm	15321	14049,14059

变量名	定义行号	引用行号
DriverCounter	10484	12487,12640,12644,13575
DriverLoadAddr	12075	12471, 12540, 12614, 12629, 12633, 13294, 13349, 13350, 14712, 14713, 14715, 14716, 14718,14719
DriverLoadFlag	12086	12413, 12416, 12423, 13238, 13329, 13342, 13607
DriverName	12090	13298
DriveType0	2010	2066
DriveType1	2024	2067
DriveType2	2038	2068,2069,2070,2071,2072,2073,2074
DriveType9	2052	2075
Drive_ErrCode	751	
Drive_Number	725	4699,4717,7465
DSValue	598	1046,1049,1056,1059
DS_BPB	15324	15389
D_089C	1254	6775
D_089E	1255	6773
D_08A0	1256	
D_08D0	1266	7394,8270
EnableWR_Flag		
EndAddrOfDriver	12076	12499, 12501, 12533, 12591, 12593, 12595, 12596, 13555, 13557, 13558, 13560, 13574, 13579,13600
EndAddrOfParaItem	10491	10586,10669,11464
EndAddrOfTempBuffer	10507	10588

变量名	定义行号	引用行号
EOF_IOSYS	61	253,373,423
ErrMes	8290	8265
ErrorCtrlFlag	8373	8719,15154
Err_BlockDev	15405	12519
Err_Config	15408	13160
Err_CountryCMD	15402	12787
Err_CountryMem	15403	12761
Err_LogicalDrive	15409	8670
Err_Memory	15404	14699
ESDSValue	804	5658,6384
EXECPB1	8345	8806
EXECPB2	8366	9812,9813,9814,9815,9820
ExtCountryInfoBuf	10511	11042
ExtendedBDPB1	1314	2004,2533
ExtendedBDPB10	1332	
ExtendedBDPB11	1334	
ExtendedBDPB12	1336	
ExtendedBDPB13	1338	
ExtendedBDPB14	1340	
ExtendedBDPB15	1342	
ExtendedBDPB16	1344	

变量名	定义行号	引用行号
ExtendedBDPB17	1346	
ExtendedBDPB18	1348	
ExtendedBDPB19	1350	
ExtendedBDPB2	1316	
ExtendedBDPB20	1352	
ExtendedBDPB21	1354	
ExtendedBDPB22	1356	
ExtendedBDPB23	1358	
ExtendedBDPB24	1360	
ExtendedBDPB25	1362	
ExtendedBDPB3	1318	
ExtendedBDPB4	1320	
ExtendedBDPB5	1322	
ExtendedBDPB6	1324	
ExtendedBDPB7	1326	
ExtendedBDPB8	1328	
ExtendedBDPB9	1330	
FATType_Init	1946	2664,2681,3101,3135,3159,3668
FAT_12Bits	792	4829
FAT_16Bits	793	4827
FD144_BPB	15363	15391

变量名	定义行号	引用行号
FD288_BPB	15376	15392
FD720_BPB	15350	15389,15390,15391
FDiskBDPB1	755	722,894
FDiskBDPB2	757	895
FDiskBDPB3	759	896
FDiskBDPB4	761	897
FDI_Flag_IO1	1956	2168,3255
FDI_Flag_IO3	8311	8434,9334
Febmary	782	1634,1650
FileHandle	8377	12720,12771
FirstCharOfPreface	8370	14457,14575
FirstDataSectorNo	55	66,105,202,204,260,261,276,277
FixedDiskNumber	2002	2483,2504,3337
FixedDiskPara	1972	3082
FlagOfMemory	12088	13261,13272,13273
FMTOpStatus	852	
FreeAddress1	1305	2538
FreeMemOFSofHMA	1300	6811,6831,6837
GapLength_FMT	747	6369
GDT	9173	9209
GenericIOCTLFuncCode	5754	6628

变量名	定义行号	引用行号
GetKeyStatus	771	2330,4068
GetParaRetCode	10490	10527, 10611, 10634, 10662, 10668, 10694, 10788, 10795, 10905, 10921, 10923, 10928, 10930, 10935, 10938, 10943, 10945, 10950, 10955, 10957, 11201, 11210, 11302, 11311, 11441,11535
Get_GenericIOCTL_CMDNum	5732	5775
HardIntVectorArray	864	1164
HDtoDD_BPB_Para	763	4871
HD_BPB	15337	15389
HeadLoadTime	733	5435
HeadLoadTime_BAK	736	5106
HeadNumber	741	5520,5555
HeadsOfFixed_Init	2005	3390
Heads_Init	1953	2457
Head_FMT	745	6382
Hexsize	12087	12424,13498,13500,13626,13640
HiddenSectors	50	100,107,448,449
HIGH_Keyword	12066	12059
HIMEMAddr	590	1098,1136,9032,9033,9036,9045
HIMEM_Mark	9119	9165,9258
HMAAddress	1073	1115
HMANotAvailable	8837	8854

变量名	定义行号	引用行号
IncorrectOrder	15407	13147
InsertFDiskSym	7432	7412
Install_Flag	8353	8706,8723,9733,12235
INT10H_Number	711	1150,2113
INT13H_CMDCode	727	5198,5201,5291,5505,5538,6081,6084,6466,6497,6522
INT13H_RetCode	749	5682
INT13H_Shift_Entry	676	1058,2130,2132,2589,2590
INT13H_TS	750	
InvalidCodePage	15401	12706,12784
InvalidPara	15397	13102
InvalidSTPara	15406	12861,12898
IO2_00D0	3915	3913,9277
IO3_Flag	1303	8399,8705
KeySign	8372	14393, 14407, 14409, 14411, 14413, 14415, 14507
KeywordNumber	11801	13209
LabelAddr_IO3	8356	9731,9732
LastDrive	8326	9562,9564,12835
LastDrive_L	11941	12831,12834
LengthOfConfig	8354	9901
LengthOfPara	8361	8366,9805
LineCounter	8337	12294,13181

变量名	定义行号	引用行号
LineNumOfLastBuffer	8340	12259
LoaderUsedAP	54	243,264,271
LoadSeg_CurrDriver	12073	12437,13347,13519,13577,13599
LogicDriveFlag	8397	8668
LOW_Keyword	12067	12061
LPT1_DriverHeader	660	652
LPT2_DriverHeader	668	660
LPT3_DriverHeader	679	668
MarkOfControlSym	15322	14057, 14086, 14122, 14127, 14177, 14210, 14262,14267
MarkOfCountry	15232	14826
MaxSectorsPerTrack	738	2390,2445,5390,6149
MaxSeg_CurrDriver	12074	12440,12490,12534,13240,13348,13559
MediaDescriptor	60	67,274,14274
MediaType	15301	
MediaTypeFlag	850	5869,6227
ModelByte	854	8410,8418,9588
ModelByte_Init	854	2201,2209,2213,2270,3557
ModifyFlag	851	6357,6438
MSDOS1_SEG	582	1005
MultiTrack	799	5460,5544,6059,6134
MultiTrack_OpFlag	803	

变量名	定义行号	引用行号
NewHeadLoadTime	734	5412,5636
NOUMB_Keyword	12069	12065
NUL_Device	15223	15181
NumberOf2NDBuffer	8321	9529,9540,9547,11229
NumberOfBuffer	8320	9467,9472,9527,9943,9980
NumberOfCylinders	15300	14083,14253,14302
NumberOfFCB	8324	9432,12970
NumberOfFDisk1	731	2725,2736
NumberOfFDisk2	2091	2153,2162,2164
NumberOfHandle	8323	9396,9631,12812
NumberOfHead	42	94,314,317
NumberOfKeepFCB	8325	9439,12971
NumberOfSFT	11889	12808,12811
NumberOfStack	8312	9591,9620,10029,12895,12904
NumberOfSwitchSym	15393	14169
N_FCBS	11923	12962,12969
N_Stacks	11975	12871, 12878, 12880, 12885, 12891, 12893, 12903
OFF_Keyword	11829	11827
OFS_AllocatedManageBlock	7951	8175,8180,8204,10069
OFS_FreeMemory	8380	9404,9412,9418,9420,9434,9444,9543,9551, 9570,9612,9964,13613,14675,14678
OFS_ManageBlockHead	7947	10034,10037

变量名	定义行号	引用行号
OFS_ManageBlockHead2	7949	8218,8243,10038
OFS_ManageBlockTail	7950	8231,10068
OFS_PreviousBuffer	8343	10003,10005
OFS_RetfIO2	2076	2085
OFS_Stack	7945	10043,10046
OFS_StackField	8314	9584,9613,10033,12897,12907
Old_INT02H_Vector1	866	10083
Old_INT02H_Vector2	7953	7970,10084
Old_INT08H_Vector1	868	10088
Old_INT08H_Vector2	7976	10089
Old_INT09H_Vector1	870	10093
Old_INT09H_Vector2	7985	10094
Old_INT0AH_Vector1	872	10125
Old_INT0AH_Vector2	8012	10126
Old_INT0BH_Vector1	874	10156
Old_INT0BH_Vector2	8028	10157
Old_INT0CH_Vector1	876	10187
Old_INT0CH_Vector2	8044	10188
Old_INT0DH_Vector1	878	10218
Old_INT0DH_Vector2	8060	10219
Old_INT0EH_Vector1	880	10249

变量名	定义行号	引用行号
Old_INT0EH_Vector2	8076	10250
Old_INT13H_Vector	714	1576,1586,2129,2131
Old_INT15H_Vector	716	1199,1218
Old_INT19H_Vector	718	
Old_INT1EH_Vector	62	69,71,483,5089,5108,5338,5339,5392,5400, 5432,5637,5643,6150,6270,6271,6365,6366
Old_INT1EH_Vector_FMT	853	6296,6297,6444
Old_INT70H_Vector1	882	10098
Old_INT70H_Vector2	7997	10099
Old_INT72H_Vector1	884	10280
Old_INT72H_Vector2	8092	10281
Old_INT73H_Vector1	886	10311
Old_INT73H_Vector2	8108	10312
Old_INT74H_Vector1	888	10342
Old_INT74H_Vector2	8124	10343
Old_INT76H_Vector1	890	10373
Old_INT76H_Vector2	8140	10374
Old_INT77H_Vector1	892	10404
Old_INT77H_Vector2	8156	10405
On_Keyword	11828	11825
Operate_Times	724	4762,4763,4765
OutputBuffer	8338	13179

变量名	定义行号	引用行号
OutputCRLF	15396	12651,13116,15137
OverlayPPB	12083	13520,13521,13524
ParaDefaultFlag	10514	10896,10953,11430
qParaFeatureFlag	10496	10551, 10561, 10566, 10574, 10595, 10774, 10777, 11081, 11082, 11088, 11148, 11169, 11229, 11342, 11350, 11380, 11556, 11558, 11624,11660,11662,11688
ParaSeparator	10493	10671,11461,11623,11659
ParaStrBuf	10509	10550,10590,11675
ParaValue	11800	12274, 12675, 12691, 12807, 12830, 12868, 12959
ParaValueTypeCode	11800	12673
Path_Shell	15235	8760,12915
PIB_C	11814	11811
PIB_F_Q	11860	11840
PIB_H	12041	12037,12038
PIB_M	11987	11984
PIB_n_F	11878	11875
PIB_N_K	11953	11949
PIB_N_X	11901	11897
PIB_ST_1	12013	12003
PIB_SW_1	12020	12004
PIB_SX_B	11788	
PIB_S_K	11964	11950

变量名	定义行号	引用行号
PIB_S_L	11935	11932
PIB_S_X	11912	11898
PIB_X_B	11766	
PIB_X_Q	11843	11838
PIB_Y_B	11777	
PIB_Y_Q	11854	11839
PreEntry1	786	1654,1656,1658,1660,4529,4532,4535
PreEntry2	788	4553
PreEntry3	923	2086,3835
PreEntry4	925	1817,4545
PreEntry5	927	1247
PreEntry6	929	1048
PreEntry7	1301	6836,8485
PRN_CMDNum	3947	
PRN_CMDTab	3949	
PRN_Device	15226	9666
PRN_DriverHeader	622	614
PRN_RepeatCount	774	4252,4291
ProcessConfigFlag	8347	8696,8700,8708
PSIB_C	11820	11818
PSIB_H	12054	12045,12052

变量名	定义行号	引用行号
PSIB_n_F	11884	11882
PSIB_N_K	11959	11957
PSIB_N_X	11907	11905
PSIB_S_K	11970	
PSIB_S_X	11918	11916
PSIB_XY_Q	11849	11847,11858
PSIB_X_B	11772	11770
PSIB_Y_B	11783	11781
Ptr_BPBArray	2003	3496
Ptr_BPB_Array	2066	3287
Ptr_Config	8376	12978,13134,14558,14783
Ptr_DriveLinker	8303	8516
Ptr_DriverNameStr	12093	12431,12432,13627,13628
Ptr_FreeBDPB	2004	2489,2511,2532,3385
Ptr_ProcessConfig	10488	12389,12390,13060,13061,13104
Ptr_SCBArray	1263	9614,9617
Ptr_SearchSCB	1261	9610,9611
RAMSize	8317	8439,8517,8772,9513
ReadingSectors	48	225,248,249,251,265,304,306,350,454
ReadKeyCode	770	2329,4028,4087
RealTimeRetPara1	902	1813,1828,1863,1866,1895,1915

变量名	定义行号	引用行号
RealTimeRetPara2	903	1741,1814,1831,1868,1871,1897
RealTimeRetPara3	904	1773,1815,1834,1873,1875,1899
RequestForInit	8386	8615, 12429, 12430, 12489, 12491, 12498, 12500,12562,12580,12581,13427,14717
ReservedSectors	52	98,167,450
RetriesForECC	802	5582
RetriesForUnECC	801	5588
ReturnAddr	900	1689,1690,1701
ROMBIOS_Version	2007	2551
Rootpath_CDS	8330	9323,9325,9327,9373
RTCFlag	778	3534,4526,4551
SecondCharOfPreface	8371	14468,14580
SecondFCB	8359	
SecPerTK_FMT	744	6367
SecPerTrackOfDrivParm	15320	14264
SectorNoOfFAT	47	443,445
SectorNoOfFAT_Init	1952	3776,3778
SectorNumber	740	5468,5517,5554
SectorNumber_Init	1944	2106,2107,2667,2669,3659,3660
SectorOfBootDirectory	7733	7760,7767,7792
SectorsOfRemain	729	5372,5621
SectorsPerCluster	64	90,196,206,213,224,247,258,14272,14299

变量名	定义行号	引用行号
SectorsPerFAT	49	96,169
SectorsPerTrack	57	92,297,300,301
SectorsPerTrackOfFixed	2006	3392
SectorsPerTrack_BAK	737	5433
SectorsPerTrack_Init	1954	2459
Sectors_Init	1945	3654,3716,3723,3735
SEG_AllocatedMemory	8384	8720,8832,9715,9736
Seg_CDSArray	8328	9392,12726,13235,13317,13344,14680
SEG_Config	8327	8625, 9312, 9391, 9757, 9897, 12125, 12187, 12757,12763
Seg_DPTBuffer	1948	2301,2760
Seg_FATBuffer	1947	2299,3665
Seg_FirstMCB1	8308	8525,8645
Seg_FirstMCB2	8378	8727
SEG_FreeMemory	8381	8833,9405,9435,9545,9568,9576,9597,9615, 9624, 9714, 9720, 9735, 9769, 9958, 9989, 10471, 10479, 12724, 12739, 13343, 13612, 14677,14679
Seg_FreeUMB	12080	13337,13362,13400,13407,13609,13990
Seg_IO2	8398	8931
SEG_ManageBlockHead	7948	8174,10036,10053
Seg_MSDOS1	12092	13766,13819,13885,13956,14015
SEG_StackField	8315	9616,10035
Set_GenericIOCTL_CMDNum	5743	5778

变量名	定义行号	引用行号
SingleFDisk	647	2179
SizeOfCMA	12089	13257,13278
SizeOfDDB	8379	8728
SizeOfDynamicStk	1264	9621
SizeOfMSDOS2InCMA	8306	8527,8910,8967
SizeOfMSDOS2InHMA	8307	8526,8893
SizeOfStack	8313	9619,10031,12896,12906
SizePerBuffer	8322	8582,8618,9949,9961,9979,10006
SK_Switches	12027	13001,13014
SourceData	9187	9202
Source_Addr_HB	9177	9205
Source_Addr_LW	9176	9204
SpecialFunctions	15297	
SPS_SK_1	12012	12999
SPS_ST_1	12019	13004
SPS_SW_1	12026	13009
SPS_SX_B	11794	12270
SPValue	743	5622
StackFlag	856	10424
StartAddrOfKeyword	10506	10779,10783,11403
StartAddrOfPara	10505	10546,10781,10860,11424

变量名	定义行号	引用行号
StartAddrOfSwitchPara	10508	10528,10743,10834
StartAddrOfSwitchStr	11802	12999,13004,13009
StartingSector_HW	797	3725,3780,5324,5331
StartingSector_LW	798	5354
StartSectorNo	44	203,205,207,208,215,216,262,263,292,293,352,353,452,453
StartupWaitTime	732	2269,5436
Status_Break	11830	12324
Str_InsertFDisk	7431	7413
ST_Switches	12028	13006,13022
SubmodelByte	855	8412,9586
SubmodelByte_Init	855	2203,3559,3561
SubRoutine1_MSDOS	8305	8528,8538,8686,8693,8860,8875
SubRoutine_IO2	8309	8505,8514,8921,8932
SubRoutine_IO3	8355	9722,9724
Support_15H	648	2615,4071
SwitchStatus_M	11993	12355
Switch_T	777	
SW_Switches	12029	13011,13024
SX_Buffers	11797	12258
S_FCBS	11924	12965
S_Stacks	11976	12874,12882,12889,12905

变量名	定义行号	引用行号
TableOfSwitchSym	15394	14171
TailOfINT13H_ILR2	1590	2584
TempParaBuf	8362	9789,9790,9808
TempWordVar	45	184,298,310,315
Temp_HW	796	2879,2888,2890,2893,3053,3056,3240,5361
Terminator	8336	12409,12425,13544
TerminatorOfData	8336	14188,14190,14769,14797
TotalOfDay_Table	908	1777
TotalOfSector	56	102,111,113,165,166
TotalSectors	15308	9494
TypeOfFAT	59	164,189,375
UMBChain_Flag	12091	13771,13775,14013
UMB_Flag	12077	12414,12502,13216,13219
UMB_Keyword	12068	
UMB_ManageProgramEntry	12081	13769,13770,13795,13799
UMB_MB_Size	12079	13339,13360,13398,13405
UMB_StartSeg	12078	13338, 13361, 13399, 13406, 13986, 13991, 13992
UncompletedSectors	600	7109,7125
Unknown	8318	8460,8539,8548
UnknownCMD	15395	13080,13085
Value_2	3203	3279

变量名	定义行号	引用行号
Value_3	3204	3278
VDisk Mark	9106	9136,9138,9152,9252
VectorAddress	1075	1116
VerifyFlag	728	
VolumeIDBuffer	1308	7750,7804,7838,7857
X Buffers	11797	12277,12284,12290
Year	780	1606,1610,1620,1624,1630,1652
ZeroValue	597	1148,5245,5256,5336,6234,6268,6294,6298,6363,6446