HOPE COMPUTER COMPANY LTD.

汉字2.13H源程序详解

鲍岳锋 编写
郑国荣

北京希望电脑公司

■ 北京市新闻出版局
准印证号：891168
■ 订购单位：北京 8721信箱资料部
■ 电　　话：2562329
■ 电　　传：01—2561057
■ 电　　挂：0755
■ 地　　址：海淀影剧院北侧
■ 乘　　车：320、332、302路海淀黄庄下车
■ 办公地点：公司大楼 101房间

汉字 2.13H 源程序详解

鲍岳桥
郑国荣　　编写

北京希望电脑公司

一九九二年三月

内 容 提 要

计算机在我国的普及应用，首先就要解决好汉字信息的计算机处理问题。2.13H是我国微机用户最常使用的普及型汉字操作系统，它集中体现了当前我国众多汉字操作系统的许多优点，并形成了自己的特色。了解和研究2.13H的内部细节，对每一个与汉字操作系统打交道的人来说都将大有裨益。

本书针对2.13H（CC版）在微机上实现的每一步骤进行了系统而全面的解释和分析，揭示了其内部的全部秘密，甚至还提到了若干设计错误。全书共分为七章，第一章介绍一些基础知识，第二～七章分别对显示字库读取、打印字库读取、键盘管理、显示、打印驱动、特殊显示、光标闪烁、文件打印等模块程序作了详尽解释。为了方便起见，程序解释中用到的一些常用资料和数据均可在本书有关章节内找到，使读者不必钻入堆积如山的资料之中就能通读本书。

本书的主要读者对象是从事与汉字有关的计算机科研、设计和应用的工作人员，尤其是计算机系统软件的设计人员，也可作为大专院校师生计算机课程的教学参考书。对于已初步了解计算机系统和初步掌握8086汇编语言的计算机学习者，本书也可起到进一步引导和深化的作用。

前　言

2.13 系列汉字操作系统自 1986 年问世以来，经过不断完善和发展，获得了很大的成功．由于它具有功能强大、使用简便、适应面广等特点，到目前为止，已成为拥有最广泛用户的汉字操作系统。然而，要想真正发挥 2.13 系列汉字操作系统的全部优越性能，做到左右逢源、运用自如，甚至根据用户自身的特点加以适当改造，就必须深入了解其内部的技术特点．为此，我们对该系统在 PC 机上实现的全部过程进行了分析和整理，与广大用户和读者共勉．

2.13H 是 2.13 系列汉字操作系统中的最高软件版．根据机器显示方式的不同分为两种子版本，即 CC 版和 GW 版．CC 版主要用于原装机（没有配汉卡的机器），GW 版主要用于国产机（带汉卡的机器）．本书是针对 2.13H 的 CC 版编写的．

本书提供了 2.13H（CC 版）中主要文件的汇编源程序清单，并把这些程序按其功能结构进行了划分．各程序中每个子程序前都给出了功能和输入输出参数的说明．对程序内的几乎每条指令都作了较详细的解析．为了便于读者阅读时查找，程序中所有标号均用加载后内存绝对地址的形式给出．由于许多驻留程序的变量地址直接放在 PSP 中，这些单元在程序中无法找到，所以在源程序清单前还列出了本程序所使用变量的地址及相应的功能说明．另外，为节省篇幅，对那些较大的数据区，只给出其地址，而省略其内容．

本书共分为七章，第一章简要地介绍了一些与 2.13H 密切相关的基础知识，提供了以后需要使用的一些常用表格．第二-三章分别介绍了各种驻留方式显示字库的读取程序及打印字库读取程序，读者可根据所使用的机器特点和需要选择阅读．第四-六章分别详细介绍了键盘管理模块、显示模块、打印驱动模块程序．第七章介绍了特殊显示、光标闪烁、文件打印及其它一些有用的程序，供读者参考．以上所有程序整理完成后经汇编连接均能正确运行，如有必要，读者可直接按书中绝对地址加以修改。

在本书编写过程中，曾得到陈红雨等同志的大力支持，在此真诚地向他们表示谢意．

由于时间较紧，加之作者水平有限，书中难免会有错漏之处，恳请广大计算机同行指正。

作　者

一九九二年元月　　于杭州

目　录

第一章 基 本 知 识

本章将简要介绍一些与汉字操作系统2．13H密切相关的基础知识，尤其是由MS－DOS约定的磁盘和存储器的数据组织布局或格式。

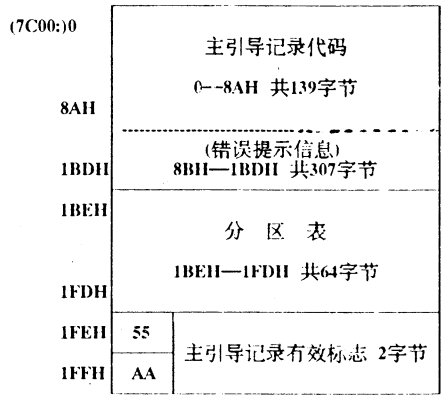
第一节 磁盘布局

由于2．13H必须在带有硬盘的计算机上运行，故本节仅限于针对硬盘介绍其布局。

PC的硬盘体系结构允许用户将一个物理硬盘划分为四个分区，安装多个操作系统共享资源，同时还可选择启动时所需的操作系统。由于16位DOS管理硬盘的最大容量为32MB，因此划分分区的做法就为DOS管理大容量提供了条件。当PC上电在硬盘上引导系统时它首先从系统中第一个硬盘的第一个物理扇区读入硬盘主引导记录，并将控制转给该记录。

一、主引导扇区（分区扇区）

主引导扇区就是硬盘的第一个物理扇区，即0头0柱面1扇区，它在磁盘（或内存）中的映象如下图所示：



可见，主引导扇区主要由两部分内容组成。

第一部分是主引导记录代码，它负责检查所选择分区（称为“活动分区”或“可自举分区”）是否唯一存在，若是则把相应操作系统引导记录（即该分区上的引导记录）装入内存，并把控制转给它。否则给出出错提示信息。

第二部分是分区表，它位于主引导扇区的后部。由于硬盘最多可能存在四个分区，每个分区表项的长度为16字节，这样，四个分区表项共占64字节。

每个分区表项布局如下：

偏移量	长 度	名 字	内 容 说 明
0(0)	1 字节	分区状态	0=不活动分区 80=活动分区,可引导的
1(1)	1 字节	起始磁头	本分区起始磁头号
2(2)	1 字	起始扇区和起始柱面	用位编码标志来存储柱面号和扇区号: 字节 n　字节 n+1 CC S S S S S SS C C C C C C C CC 字节 n的最前两位在前,字节 n+1 的八位在后,组成共十位的柱面号CYLINDER,字节 n的最后六位构成扇区号 SECTOR

4(4)	1 字节	分区类型	01=12位FAT的DOS 64=NOVELL 02=XENIX 75=PCIX 04=16位FAT的DOS DB=CP/M 06=保留DOS FF=BBT
5(5)	1 字节	终止磁头	本分区终止磁头号
6(6)	1 字	终止扇区和终止柱面	同起始扇区和起始柱面
8(8)	双字	起始绝对扇区	注意字节交换
C(12)	双字	扇区数	注意字节交换

二、分区引导扇区(BOOT)

分区引导记录位于分区第一逻辑扇区，其布局如下:

偏移量	长 度	内 容 说 明
0(0)	3字节	跳转到引导记录代码
3(3)	8字节	版本号。也是OEM(原厂委托制造)名字或标志
B(11)	字	每扇区字节数
D(13)	1字节	每簇扇区数(必须是 2 的幂)
E(14)	字	保留扇区数(用于根目录、FAT等等)
10(16)	1字节	文件定位表(FAT)的个数(拷贝)
11(17)	字	根目录中的最大目录项数
13(19)	字	总扇区数
15(21)	1字节	介质描述字节
16(22)	字	每个FAT占用的扇区数
18(24)	字	每道扇区数
1A(26)	字	磁头数
1C(28)	字	隐含扇区数
1E(30)	224字节	其它：包括程序、数据或保留未用区
FE(254)	2字节	引导扇区数有效标志：55 AA

三、文件定位表(FAT)

文件定位表(File Allocation Table)简称 FAT，它是记录分区中文件占用簇号链的一张表，其中的项用于存放簇号。12 位 FAT 每项占 1.5 字节，16 位 FAT 每项占 2 字节。FAT 表(一般有两个拷贝)位于紧接 BOOT 扇区之后的连续扇区中。

(一) 12 位 FAT

项	例 值	用 途	说 明
0	FF8	磁盘标识字节	1.第0,1两项保留给DOS 2.第0项 FF8 表示硬盘
1	FFF	填充符	
2	003	指向第 3 簇	磁盘簇的值: 000=可用簇 001-FEF=下一簇号 FF0-FF6=保留簇 FF7=坏簇 FF8-FFF=文件的最后簇
3	004	指向第 4 簇	
4	FFF	文件结束簇	
5	000	可用簇(空簇)	

(二)　16 位 FAT

项	例 值	用　途	说　明
0	FFF8	磁盘标识字节	1.第0,1两项保留给DOS 2.第0项 FFF8表示硬盘
1	FFFF	填充符	
2	0003	指向第 3 簇	1.磁盘簇的值: 0000=可用簇 0001-FFEF=下一簇号 FFF0-FFF6=保留簇 FFF7=坏簇 FFF8-FFFF=文件最后簇 2.FAT项是字节交换的
3	0004	指向第 4 簇	
4	FFFF	文件结束簇	
5	0000	可用簇(空簇)	

四、文件目录表(FDT)

文件目录表(File Directory Table)简称 FDT，它是记录分区中文件(或子目录)目录及其属性的一张表。根目录 FDT 位于紧接 FAT 表之后的盘簇中，子目录 FDT 位于其父目录指定的盘簇中。FDT 中的目录项每项 32 字节，顺序存放。其布局如下:

偏移量	长　度	描　述	格　式	说　明
0(0)	8字节	文件名	ASCII字符	文件名第一字节指出目录项状态: 00H=目录尚未使用过 05H=文件名的第一个字节实际为E5H E5H=文件被使用过但已被删除 2EH=是目录（若下一字节也为2EH,则簇域中包含父目录簇号）
8(8)	3字节	扩展名	ASCII字符	
B(11)	1字节	属性字节	位代码: 位0=只读　位4=子目录 位1=隐含　位5=修改位 位2=系统　位6=保留 位3=卷标　位6=保留	
C(12)	10字节	保留区		
16(22)	字	最后修改时间	日期/时间 格式	
18(24)	字	最后修改时间	日期/时间 格式	
1A(26)	字	起始簇号	二进制整数字	
1C(28)	双字	文件字节长度	二进制整数双字	

第二节　程序段前缀和文件控制块

一、程序段前缀（PSP）

当 DOS 运行一个程序时，首先为暂驻程序留出足够的空间，该空间称为程序段，用于存放从磁盘装入的程序。在 DOS 装载程序前，首先在程序段的前 100H 即 256 字节建立一个控制块，这个控制就是程序段前缀（Program Segment Profoix）。其布局如下:

偏移量	长　度	通常用法	描　述	说　明
0(0)	字	CD20H	INT20H终止地址	

2(2)	字		内存分配块结束地址	是段地址
4(4)	字节	00H	保留	
5(5)	5字节		DOS系统功能远调用人口	06H:起后用作段大小
A(10)	双字		INT22H中断向量原内容	程序结束处理
E(14)	双字		INT23H中断向量原内容	Ctrl-Break 处理
12(18)	双字		INT24H中断向量原内容	严重错误处理
16(22)	字		父进程的PSP	是段地址
18(24)	20字节	FF=可用	文件DOS内部数码表	每个文件占一字节，位7=1 表示不继承
2C(44)	字		环境块地址	是般地址
2E(46)	字节		保留	
32(50)	字	14H,00H	向量表大小	DOS允许更大的表
34(52)	双字	12H,00H	向量表地址	DOS3.3允许有向量表地址
38(56)	字节		保留	
50(80)	字	CD21H	INT21H DOS 调用	
52(82)	字节	CBH	远返回	
53(83)	9字节		保留	
5C(92)	36字节		缺省未打开文件控制块 1	
6C(108)	20字节		缺省未打开文件控制块 2	覆盖第一个FCB
80(128)	字节		命令行参数长度	也是缺省DTA的首址
81(129)	127字节		命令行参数	空格开始,回车结束

DOS设置程序段前缀的目的有三：一是用于存放DOS为了管理进程所需的数据；二是作为用户程序在运行时要求DOS提供信息的缓冲区；三是用作文件管理系统中输入／输出及交换文件数据的区域。

对用户来说，以下几个区域显得特别重要：

1．内存顶部地址（即偏移 02H:内存分配块结束地址）；

2．段大小（即偏移 06H:此域包含了本程序段的最大可用字节数）；

3．DOS内部数码表（018H-2BH:不同于文件句柄）；

4．环境块段地址（即偏移 2CH 处）；

5．文件控制块（FCB）；

6．命令行／默认的磁盘传输区（DTA）

二、文件控制块（FCB）

文件控制块（File Control Block）是传统的DOS文件管理机制中，用户程序和操作系统之间对话的接口格式。程序欲进行文件操作前，初始化FCB，按其标准格式给出驱动器代码、文件名、扩展名，构成“未打开的FCB”，然后将其地址传送给DOS以打开或生成文件。如果文件被成功地打开或建立，DOS就根据磁盘当前文件登记项填写FCB的某些域，形成“打开的FCB”，供用户程序进一步使用。在FCB的保留区中，有时也存放某些其他信息，这些信息是用于操作系统自身目的的，且随不同的DOS版本而异，用户程序不能加以修改。

（一）未打开的FCB

偏移量	长　度	名　字	内　　容
0(0)	1字节	驱动器号	逻辑驱动器号:0=缺省,1=A,2=B,3=C等等
1(1)	8字节	文件名	ASCII码字符,不足时用空格填充

9(9)	3字节	扩展名	ASCII码字符,不足时用空格填充
C(12)	25字节	保留	置为 0

（二）打开的ＦＣＢ

偏移量	长　度	名　字	内　　容
0(0)	1字节	驱动器号	逻辑驱动器号:0＝缺省,1＝A,2＝B,3＝C等等
1(1)	8字节	文件名	ASCII码字符,不足时用空格填充
9(9)	3字节	扩展名	ASCII码字符,不足时用空格填充
C(12)	字	当前块号	二进制值表示当前记录块(文件打开时置为 0)
E(14)	字	记录大小	每个记录的字节数(缺省＝128)
10(16)	双字	文件大小	二进制值表示文件大小(以字节计)
14(20)	字	文件日期	含有文件上次修改日期的紧缩字
16(22)	字	文件时间	含有文件上次修改时间的紧缩字
18(24)	8字节	保留	供 DOS 内部使用
20(32)	1字节	当前记录号	二进制值,表示当前记录(文件打开时为 0)
21(33)	双字	随机记录号	二进制值,表示下一个将要读写的随机块

第三节　内存控制块

内存控制块是 DOS 为了有效管理内存资源而设置的数据区，DOS 为每一个分配的内存区建立一个内存控制块，长度为 16 字节(一个段落)，直接处于所控制的内存区前部。其布局如下:

偏移量	长　度	名　字	内　　容
0	字节	位置	如不是最后一块为 4DH，如是则为 5AH
1	字	进程ID	PSP 段地址
3	字	分配总数	分配的段数
5	11字节	保留	保留

只要程序对内存块有任何分配、修改、释放的请求，或启动 EXEC 功能调用或结束一个程序,DOS 均检查内存控制链,如果有一个块表现出已被破坏或控制链被破坏,DOS 就显示出错信息: Memory allocation error，且系统死机。

第四节　本书引用的ＤＯＳ和ＢＩＯＳ中断调用

在本书引用的ＤＯＳ和ＢＩＯＳ中断调用中，INT 05H、INT 10H、INT 16H、INT 17H 等类型将在以后章节具体涉及的程序中详细说明，在此不再赘述。为节省篇幅，以下列举的中断类型及各子中断调用严格限于本书程序的使用范围。

一、ＩＮＴ　１３Ｈ　ＢＩＯＳ磁盘Ｉ／Ｏ

AH	功　　能	输 入 参 数	输 出 参 数
	把在AL中说明的扇区数从在DL中指定	AH＝02H AL＝要读的扇区数	AH＝00H　无错 ＝01-0FFH　错误代码

2	的驱动器读到ES:BX指定的缓冲区中。	CH=柱面号(作为低8位,0为基数) CL=柱面/扇区号,其中: 7-6位=柱面号(高2位) 5-0位=扇区数 DH=磁头号(0为基数) DL=驱动器号 =80H 硬盘1 =81H 硬盘2 ES:BX=缓冲区指针	=硬盘状态(40:74H) AL=所传送数据的扇区数 CF=0 无错 =1 有错
3	从由ES:BX定义的缓冲区把在AL中指家的扇区数写入由DL指定的驱动器中。	AH=03H AL=要写的扇区数 CH=柱面号(作为低8位,0为基数) CL=柱面/扇区号,其中: 7-6位=柱面号(高2位) 5-0位=扇区数 DH=磁头号(0为基数) DL=驱动器号 =80H 硬盘1 =81H 硬盘2 ES:BX=缓冲区指针	AH=00H 无错 =01-0FFH 错误代码 =硬盘状态(40:74H) AL=所传送数据的扇区数 CF=0 无错 =1 有错

二、INT 15H BIOS系统服务

AH	功 能	输入参数	输出参数
87	在任意地址空间(0-16MB)之间传送数据。	AH=87H CX=要移动的16位字的数量(0-8000H个字,64K) ES:SI=指向由调用者分配的30H字节的表(描述符表GDT,见下表)。	AH=00H 移动完成 =01H RAM奇偶错误 =02H 其他例外中断错 CF=0 无错 =1 有错 ZF=0 移动未完成 =1 移动完成

全局描述符表(GDT)结构

局部描述符序号	偏 移	描 述 符 名	填 写
1	00(0)	空描述符	用户
2	08(8)	GDT位置	BIOS
3	10(16)	源数据块描述符(DS)	用户
4	18(24)	目标数据块描述符(ES)	用户
5	20(32)	本功能的代码段描述符(CS)	BIOS
6	28(40)	本功能使用的用户堆栈段描述符(SS)	BIOS

局部描述符表(LDT)的内容

位 长	内 容
16	段 长
16	段地址低位
8	段地址高位
8	访问权
16	保留

三、INT 18H ROM-BASIC解释程序

功　能	输入参数	输出参数
进入 ROM-BASIC 解释器	/	进入ROM-BASIC的控制

四、INT 20H DOS终止程序

功　能	输入参数	输出参数
释放占用的内存并把结束地址、Ctrl-Break处理程序地址以及严重错误处理中断地址从PSP中得到恢复。	CS=当前程序的PSP段地址	控制转到结束地址

五、INT 21H DOS功能调用

AH	功　能	输入参数	输出参数
02	显示输出	DL=输出ASCII字符	
05	打印机输出 (DOS 自动检查打印机并给出适当信息)	DL=输出ASCII字符	
09	显示字符串	DS:DX=字符串首址 以'$'为字符串结束标志	
0C	清除输入缓冲区并请求指定的输入功能	AL=输入功能号 =01H读键盘字节并回响 =06H直接控制台 I/O =07H直接控制台输入但不回响 =08H读键盘字符位不回显 =0AH键盘字符存入缓冲区	
11	查找第一个目录项	DS:DX=未打开的 FCB首址	AL=0　找到 AL=FF 未找到 DTA DS:DX=FCB 首址
1A	置 DTA 地址	DS:DX=DTA 地址	
25	设置中断向量	DS:DX=中断向量 AL=中断类型号	
30	取 DOS 版本号		AH=子版本号 AL=主版本号 BX和CX使用后被置为 0
31	程序结束并驻留	AL=返回码 DX=驻留区大小	AL=终止码
35	取中断向量	AL=中断类型	ES:BX=中断向量(CS:IP)
36	取空闲磁盘空间	DL=驱动器号 =0　缺省 =1　A 盘 =2　B 盘	成功:AX=每簇扇区数 BX=有效簇数 CX=每扇区字节数 DX=总簇数 失败:AX=0FFFFH
3D	打开文件	DS:DX=ASCIIZ串首址 AL=0　读 =1　写 =2　读/写	成功:AX=文件句柄 失败:AX=错误码

3E	关闭文件	BX=文件句柄	失败:AX=错误码
3F	读文件或设备	DS:DX=数据缓冲区地址 BX=文件句柄 CX=读取的字节数	成功:AX=实际读入字节数 AX=0 已到文件尾 出错:AX=错误码 DS:DX=信息块地址
40	写文件或设备	DS:DX=数据缓冲区地址 BX=文件句柄 CX=写入的字节数	成功:AX=实际写入的字节数 出错:AX=错误码
42	移动文件指针	BX=文件句柄 CX:DX=位移量 AL=移动方式(0,1,2)	成功:DX:AX=新指针位置 失败:AX=错误码
43	置/取文件属性	DS:DX=ASCIIZ串地址 AL=0 取文件属性 =1 置文件属性 CX=文件属性	成功:AX=文件属性 CX=新的文件属性 失败:AX=错误码
4C	带返回码结束	AL=返回码	
57	置/取文件日期和时间	BX=文件句柄 AL=0 读取 =1 设置(DX:CX)	DX:CX=日期和时间 失败:AX=错误码

六、INT 25H DOS绝对磁盘读

功 能	输入参数	输出参数
读磁盘扇区，而与磁盘的文件结构无关。	AL=驱动器号 =0 A盘 =1 B盘 =2 C盘等等 CX=所读连续逻辑扇区数 DX=所读第1个逻辑扇区号 DS:BX=磁盘传输区(DTA)地址	DS:BX=指向传输区域的首址 注: 1.标志寄存器被置入栈中; 2.除般寄存器 (CS,DS,ES,SS)以外的所有寄存器被破坏.

七、INT 27H DOS程序驻留退出

功 能	输入参数	输出参数
促使DOS终止当前的程序，驻留于内存，而不被以后装入的其他程序所覆盖，仅对.COM程序有效	CS=程序的 PSP 首址 DX=欲驻留部分程序的字节数加 1	

第二章　显示字库读取

第一节　读硬盘字库　FILE0A.COM

一、使用说明

FILE0A　ab←┘

其中：a表示常用字区大小，以50个汉字为单位。

b表示内部词组区大小，以k为单位。

例如：FILE0A　82表示保留400个常用字的缓冲区，内部词组区为2k字节。

当计算机的内存小于或等于640K，同时又需较大的内存自由空间时，可选择使用本程序。因使用本程序时显示字库不驻留内存，故可为用户保留较大的内存空间。但又由于本程序直接使用INT　13H从硬盘上读取汉字的点阵数据，磁盘操作过于频繁，既影响硬盘寿命，也使汉字显示速度受到很大的影响。

二、程序功能

本程序主要包括两个部分，即初始化代码和INT　7FH代码。

初始化代码主要完成以下工作：

1）建立显示字库文件（HZK16）的扇区链表。

2）初始化常用字区。

3）初始化内部词组区。

4）设置INT　7FH。

5）驻留退出。

程序运行后，便可用INT　7FH读取汉字的点阵数据（16x16点阵）。

INT　7FH的运行过程如下：

1）检索常用字区，若所需汉字在常用字区中，那么直接返回该汉字点阵数据的地址。

2）否则，计算该汉字点阵数据在字库文件中的相对位移，查链表得到包含该汉字扇区的逻辑扇区号，换算为磁盘物理地址后，用INT　13H将该扇区读入内存。

3）将该汉字及其点阵数据保存在常用字区中，使该汉字成为常用字。

4）返回所需汉字点阵数据的地址。

三、变量名表

地址	长度	意义
0082h	1	命令行参数1，指定常用字区大小
0083h	1	命令行参数2，指定内部词组大小
00F0h	2	每簇扇区数
00F4h	2	磁盘数据区起始扇区的逻辑扇区号
00F6h	2	暂存连续簇数
01E0h	26h	字库文件(HZK16)的文件控制块(FCB)
0200h	2	内部词组区首址即常用字区尾址
0202h	2	内部词组区尾址
0204h	2	常用字区首址
0206h	2	常用字已使用区尾址
0208h	2	常用字区更新地址
020Ah	2	每扇区字节数
020Ch	2	每道扇区数
020Eh	2	磁头数
02C2h	1Eh	字库文件(HZK16)的扇区链表
02E0h	8	错误信息，'ERROR!',7,'$'

四、程序清单

```
;FILE0A.ASM
CODE            SEGMENT
                ASSUME  CS:CODE,DS:CODE
                ORG     100H
L_0100:
;读入硬盘主引导扇区，目的是为了查找 DOS 分区的起始物理地址
                MOV     AX,201H             ;AH=2:调用读扇区功能,AL=1:读
                                            ;入1个扇区
                MOV     BX,800H             ;读入数据存放地址
                MOV     CX,1
                MOV     DX,80H              ;物理地址为:C盘0道0面1扇区
                INT     13H                 ;读入主引导扇区

;查找 DOS 分区
                MOV     SI,OFFSET DS:[1BEH] ;分区表起始位移
L_0111:
                CMP     BYTE PTR [BX+SI+4],1
                JE      L_0128              ;是12 bits FAT 的DOS分区
                CMP     BYTE PTR [BX+SI+4],4
                JE      L_0128              ;是16 bits FAT 的DOS分区
                CMP     BYTE PTR [BX+SI+4],6
                JE      L_0128              ;是保留的DOS分区

当前分区项不是 DOS 分区项
                ADD     SI,10H              ;SI=下一分区项位移
                JMP     SHORT L_0111        ;检查下一分区项

L_0128:
;读入分区引导扇区(BOOT)，目的是为了知道当前分区的基本信息
                MOV     DH,[BX+SI+1]        ;起始磁头号
                MOV     CX,[BX+SI+2]        ;起始扇区号和磁道号
                MOV     AX,201H             ;读入该分区的第一个扇区,即该
                                            ;分区的引导扇区
                INT     13H

;读入文件定位表(FAT)
                MOV     AL,[BX+16H]         ;一个FAT占用扇区数
                INC     AX                  ;加上BOOT扇区
;此时，CX,DX 还是当前分区的起始物理地址，由于 FAT 正好紧接在 BOOT 扇区的后面，故这
; 里为避免复杂的物理地址计算，将 BOOT 扇区和 FTA 一起读入内存.
                MOV     AH,2
                INT     13H                 ;读入 BOOT 扇区和 FAT,数据存放地
                                            ;址仍为 800H
;计算硬盘数据区的起始逻辑扇区号
                MOV     AX,[BX+0BH]         ;每扇区的字节数
                MOV     D_020A,AX
                MOV     AX,[BX+18H]         ;每道扇区数
                MOV     D_020C,AX
                MOV     AX,[BX+1AH]         ;磁头数
                MOV     D_020E,AX
                MOV     AL,[BX+0DH]         ;每簇扇区数
                XOR     AH,AH
                MOV     DS:[0F0H],AX
                MOV     AX,[BX+16H]         ;一个FAT占用扇区数
                MUL     BYTE PTR [BX+10H]   ;到FAT数目=FAT所占总扇区数
```

```
                ADD     AX,[BX+0EH]             ;加保留扇区数
                ADD     AX,[BX+1CH]             ;加隐含扇区数=根目录起始逻辑
                                                ;扇区号
                MOV     DS:[0F4H],AX
                MOV     AX,20H                  ;每个目录项长度=32字节
                MUL     WORD PTR [BX+11H]       ;剩以根目录项数=根目录长度
                DIV     D_020A                  ;除以每扇区字节数=根目录所占
                                                ;扇区数
                ADD     DS:[0F4H],AX            ;加根目录起始逻辑扇区号=数据
                                                ;区起始逻辑扇区号

;建立 HZK16 的扇区链表
                MOV     BX,0A00H                ;FAT首址
                MOV     BP,2C2H                 ;链表首址
                CALL    L_0300                  ;调用建立链表子程序

;初始化常用字区
                MOV     AX,BP                   ;L_0300返回时，BP为可用内存首
                                                ;址
                MOV     D_0204,AX               ;常用字区首址
                MOV     D_0206,AX               ;常用字已使用区尾址
                MOV     D_0208,AX               ;常用字区更新地址,此时常用字
                                                ;区中没有任何常用字
                MOV     AL,DS:[82H]             ;取命令行参数1
                CMP     AL,41H
                JB      L_018F                  ;是数字(<'A')转
                SUB     AL,7                    ;'A'-'F'->4a-4f
L_018F:
                AND     AL,0FH                  ;30-4f->0-f
                XOR     AH,AH                   ;AXx50=常用字数
                MOV     DX,6A4H                 ;每50个字的缓冲区使用量
;每个汉字本身占 2 个字节，加上 32 字节的点阵数据，为 34 字节．50x(32+2)=6A4H

                MUL     DX                      ;AX=常用字区长度

;初始化内部词组区，内部词组区紧接在常用字区之后
                ADD     AX,BP                   ;AX=内部词组区首址
                                                ;或常用字区尾址
                MOV     D_0200,AX
                MOV     AL,2                    ;缺省内部词组区为2K字节
                CMP     BYTE PTR DS:[80H],3     ;有命令参数2吗?
                JB      L_01AB                  ;没有，按缺省值设置
                MOV     AL,DS:[83H]             ;取命令行参数2
                AND     AL,0FH                  ;将数字转换为相应的数值，AL=
                                                ;内部词组区长度(单位:KB)
L_01AB:
                XOR     AH,AH
                MOV     DX,400H                 ;1K=1024字节
                MUL     DX                      ;AX=内部词组长度(单位:字节)
                ADD     AX,D_0200               ;AX=内部词组区尾址
                MOV     D_0202,AX

;设置 INT 7FH,并驻留退出
                INC     AX                      ;留点余量
                PUSH    AX                      ;保存程序尾址
```

```
                MOV     DX,OFFSET L_0211        ;INT 7FH的入口地址

                MOV     AX,257FH
                INT     21H                     ;设置INT 7FH
                POP     DX                      ;DX=程序尾址
                INT     27H                     ;驻留退出

                DB      26 DUP (0)              ;没有用

D_01E0          DB      3                       ;HZK16的文件控制块(FCB)
                DB      'HZK16      '           ;文件名
                DB      20 DUP (0)

D_0200          DW      0                       ;内部词组区首址
D_0202          DW      0                       ;内部词组区尾址
D_0204          DW      0                       ;常用字区首址
D_0206          DW      0                       ;常用字已使用区尾址
D_0208          DW      0                       ;常用字更新地址

D_020A          DW      0                       ;每扇区字节数
D_020C          DW      0                       ;每道扇区数
D_020E          DW      0                       ;磁头数

                DB      0                       ;没有用

;INT 7FH
;输入: DX = 汉字
;输出: DX:0 = 汉字点阵数据地址
L_0211:
                PUSH    DS
                PUSH    ES
                PUSH    AX
                PUSH    BX
                PUSH    CX
                PUSH    SI
                PUSH    DI                      ;保存寄存器

                PUSH    CS
                POP     DS                      ;DS=CS
                PUSH    CS
                POP     ES                      ;ES=CS

                MOV     SI,D_0204               ;SI=常用字区首址
                JMP     SHORT L_0226
                NOP
L_0223:
                ADD     SI,20H                  ;SI=下一个常用字地址
L_0226:
                CMP     SI,D_0206
                JE      L_0242                  ;常用字区全部查完,没有找到指
                                                ;定汉字,转从磁盘上读入
                LODSW                           ;取汉字
                CMP     AX,DX                   ;是指定汉字吗?
                JNE     L_0223                  ;不是
```

```
;指定的汉字在常用字区中，不用再次从磁盘读入
                XOR     DI,DI                   ;点阵数据存放地址
                MOV     CX,20H                  ;共32个字节
                REP     MOVSB                   ;传送数据
                PUSH    CS
                POP     DX                      ;DX=CS,即点阵数据的段址
L_023A:
                POP     DI
                POP     SI
                POP     CX
                POP     BX
                POP     AX
                POP     ES
                POP     DS                      ;恢复寄存器

                IRET                            ;中断返回

;从磁盘上读入该汉字的点阵数据，同时，将该汉字及其点阵数据保存在常用字区中
L_0242:
                MOV     DI,D_0208               ;常用字更新地址
                MOV     [DI],DX                 ;保存汉字
                INC     DI
                INC     DI                      ;DI=保存点阵数据地址

;计算该汉字在字库中的顺序号
                AND     DX,7F7FH                ;屏蔽高位
                SUB     DX,2121H
;汉字机内码是从 ASCII 码的 0A1H(161)开始的，以上两条指令相当于指令: SUB DX,0A1A1H

                MOV     AL,5EH                  ;每区有94个汉字
                MUL     DH
                XOR     DH,DH
                ADD     AX,DX                   ;AX=汉字顺序号

;计算包含该汉字点阵数据的扇区在 HZK16 中的相对扇区号
                XOR     DX,DX
                MOV     CX,10H                  ;一个扇区可存放16个汉字的点阵
                                                ;数据
                DIV     CX                      ;AX=相对扇区号
                PUSH    DX                      ;DX=在扇区中的相对编号

;查链表，取包含该汉字点阵数据的逻辑扇区号
                MOV     BX,OFFSET D_02C2        ;链表首址
L_0265:
                CMP     AX,[BX+2]               ;在当前链项中吗?
                JB      L_0272                  ;是
                SUB     AX,[BX+2]               ;不是
;AX=从下一链开始的相对扇区号

                ADD     BX,4                    ;BX=下一链项地址
                JMP     SHORT L_0265            ;继续查找

L_0272:
                ADD     AX,[BX]                 ;AX=点阵所在扇区的逻辑扇区号
;根据逻辑扇区号计算物理地址
```

```
                XOR     DX,DX
                DIV     D_020C          ;除以每道扇区数
                MOV     CX,DX
                INC     CL              ;CL=绝对扇区号
                XOR     DX,DX
                DIV     D_020E          ;除以磁头数
                MOV     CH,AL           ;CH=磁道号，磁道号共10位,高2
                                        ;位数据还在AH中
                MOV     DH,DL           ;DH=磁头号,
                MOV     AL,40H
                MUL     AH              ;相当于SHL AH,6
                ADD     CL,AL           ;将磁道号的高2位放入CL中
;由于硬盘的磁道数可能超过 255，需要 10 位数据，因此不能用 CH 完全来表示. 同时，由
;于每道扇区数不会超过 64，因此，可以将磁道数的高 2 位数据保存在 CL 的高 2 位中

;读入包含指定汉字点阵数据的扇区
                MOV     DL,80H          ;C盘
                XOR     BX,BX           ;读入数据存放地址
                MOV     AX,201H         ;读一个扇区
                INT     13H

;将点阵数据保存到常用字区中
                POP     AX              ;AX=在扇区中的相对编号
                MOV     DX,CS
                ADD     DX,AX
                ADD     DX,AX           ;DX:0=点阵数据地址
;由于 DX 是段址，因此是以 16 字节为单位的，而每个汉字的点阵数据占 32 字节，这里两
;次 ADD DX,AX 使得 DX 指向所需汉字点阵数据的段址

                MOV     CX,20H          ;32个字节
                MUL     CL
                MOV     SI,AX           ;SI=点阵数据首址
                REP     MOVSB           ;保存到常用字区中

;修改常用字区内部指针
                CMP     DI,D_0206       ;是增加常用字吗?
                JBE     L_02B1          ;不是
                MOV     D_0206,DI       ;增加常用字，修改常用字已使用
                                        ;区尾址
L_02B1:
                CMP     DI,D_0200       ;常用字区已用完了吗?
                JNE     L_02BB          ;没有
                MOV     DI,D_0204       ;修改更新指针，指向常用字区首址
;这样，以后增加新的常用字时，就会淘汰一个已有的常用字

L_02BB:
                MOV     D_0208,DI       ;修改更新地址
                JMP     L_023A

D_02C2          DW      15 DUP (0)      ;HZK16的链表区

D_02E0          DB      'ERROR!', 7, '$'    ;错误信息
                DB      8 DUP (0)       ;没有用

;字库文件名(HZK16)没有找到，结束运行，INT 7FH 没有设置
```

```
L_02F0:
                POPF
                MOV     DX,OFFSET D_02E0
                MOV     AH,9
                INT     21H                     ;显示错误信息
                INT     20H                     ;结束运行

                DB      6 DUP (0)               ;没有用

;子程序: 建立字库文件(HZK16)的扇区链表
;输   入: BP=链表首址
;         BX=FAT 首址
;输   出: BP=链表尾址, 也即可用内存空间首址
L_0300:
;修改磁盘数据传送区地址(DTA), 因为缺省的 DTA 在 80H 处, 它刚好和命令行内容区重叠,
;如果不修改 DTA, 执行以下程序将会丢失命令行内容
                MOV     DX,400H                 ;新DTA地址
                MOV     DI,DX
                MOV     AH,1AH                  ;设置DTA功能
                INT     21H

                MOV     DX,OFFSET D_01E0        ;文件控制块(FCB)地址
                MOV     AH,11H
                INT     21H                     ;寻找匹配文件名(HZK16)
                OR      AL,AL                   ;找到?
                JNZ     L_02F0                  ;没有, 转非正常结束
                MOV     AX,[DI+1BH]             ;取文件起始簇号
                PUSH    AX
                SUB     AX,2                    ;减去两个保留簇号=实际簇号
;FAT 的前两项为保留簇号, 这两种簇号仅说明磁盘类型等, 不指向实际的磁盘数据区

                MOV     DX,DS:[0F0H]            ;每簇扇区数
                MUL     DX
                ADD     AX,DS:[0F4H]            ;AX=文件起始扇区的逻辑扇区号
                MOV     [BP],AX                 ;写入链表
                POP     AX                      ;恢复起始簇号
                MOV     WORD PTR DS:[0F6H],0    ;连续簇数=0,以后每当发现一连
                                                ;续簇时,该单元加1
L_032F:
                PUSH    AX
                INC     WORD PTR DS:[0F6H]      ;连续簇数加1
                MOV     SI,AX                   ;SI=当前簇号
                TEST    BYTE PTR [BX+4],40H
;对于 12 bits FAT, 每个簇号用 12 位表示. FAT 中, 前 3 字节为保留簇号占用. 第 4 字节的
;低 4 位和第 3 字节组成簇号 2, 第 5 字节和第 4 字节的高 4 位组成簇号 3. 一般情况下, 硬盘最
;前面的数据为两个系统文件占用, 而且必须连续存放, 这样簇号 2 的数值应为 3, 簇号 3 的
;数值应为 4, 即第 4 字节数值为 40H.

                JNZ     L_0345                  ;12 bits FAT 转
;处理 16 bits FAT
                ADD     SI,AX
                MOV     AX,[BX+SI]              ;AX=下一簇号
                CMP     AX,0FFF8H               ;是文件最后一簇吗?
                JMP     SHORT L_035B
```

```
;处理 12 bits FAT
L_0345:
                SHR     AX,1
                PUSHF                           ;保存CF
                ADD     SI,AX                   ;SI = AX x 1.5
                MOV     AX,[BX+SI]              ;取簇号(仅12位有效)
                POPF                            ;恢复CF
                JNC     L_0355                  ;CF=0, 表示是偶数簇
;当簇号为奇数时, 表示 AX 的高 12 位是下一簇号值, 反之, 表示 AX 的低 12 位是下一簇号值

                MOV     CL,4                    ;奇数簇
                SHR     AX,CL                   ;右移4位, 去掉低4位,AX=下一簇
                                                ;号
                JMP     SHORT L_0358
L_0355:
                AND      AX,0FFFH               ;偶数簇,屏蔽高 4 位,AX=下一簇号
L_0358:
                CMP      AX,0FF8H               ;是文件最后一簇吗?
L_035B:
                POP     SI                      ;恢复当前簇号
                JC      L_0368                  ;不是文件最后一簇

;处理文件的最后一簇
                MOV     AX,0FFFFH               ;最后簇标记
                MOV     [BP+2],AX               ;写入链表结束标记
                ADD     BP,4                    ;BP=内存可用空间首址
                RETN

L_0368:
                INC     SI                      ;当前簇号加 1
;如果是连续簇, 那么下一簇号应该=SI
                CMP     AX,SI                   ;是连续簇吗?
                JE      L_032F                  ;是
;非连续簇, 增加链项
                PUSH    AX                      ;保存下一簇号
                MOV     AX,DS:[0F0H]            ;AX=每簇扇区数
                MUL     WORD PTR DS:[0F6H]      ;剩以每簇扇区数=连续扇区数
                MOV     [BP+2],AX               ;写入当前链的连续扇区数
                XOR     AX,AX
                MOV     DS:[0F6H],AX            ;连续簇数=0
                MOV     AX,DS:[0F0H]            ;每簇扇区数
                POP     SI                      ;SI=下一簇号
                PUSH    SI                      ;保存下一簇号
                SUB     SI,2                    ;SI=实际簇号
                MUL     SI
                ADD     AX,DS:[0F4H]            ;AX=逻辑扇区号
                ADD     BP,4
                MOV     [BP],AX                 ;保存下一链项的起始逻辑扇区号
                POP     AX                      ;AX=下一簇号
                JMP     SHORT L_032F
CODE            ENDS
                END     L_0100
```

第二节　一级字库驻留内存　FILE1A.COM

一、使用说明

FILE1A a←┘

其中：a表示内部词组区大小，以k为单位。

例如：FILE1A 2表示设置内部词组区为2k字节。

当计算机的内存小于或等于640K时，若将汉字库内容全部读入内存，势必占用很大的内存空间（约256K），使用户可用内存减少，以至于不能运行许多常用的应用软件（如：Foxbase）。但若采用直接从硬盘读取汉字点阵数据的方法，又会影响汉字的显示速度。本程序提供了一种折中的处理方法，即仅将一级汉字的点阵数据读入内存，其它不常用汉字的点阵数据则直接从硬盘上读取。

二、程序功能

本程序主要包括两个部分，即初始化代码和INT 7FH代码。

初始化代码主要完成以下工作：

1）建立字库文件（HZK16）的扇区链表。

2）初始化内部词组区。

3）读入一级汉字（包括部分常用的汉字符号）的点阵数据。

4）设置INT 7FH。

5）驻留退出。

程序运行后，使可用INT 7FH读取汉字的点阵数据（16x16点阵）。

INT 7FH的运行过程如下：

1）判断指定汉字是否为一级汉字，若是，则直接从内存中读取该汉字的点阵数据，返回点阵数据的地址。

2）否则，计算该汉字点阵数据在字库文件中的相对位移，查链表得到包含该汉字的扇区的逻辑扇区号，换算为磁盘物理地址后，用INT 13H将该扇区读入内存。

3）返回指定汉字的点阵数据地址。

三、变量名表

地址	长度	意义
0082h	1	命令行参数1，指定内部词组大小
00F0h	2	每簇扇区数
00F4h	2	磁盘数据区起始逻辑扇区号
00F6h	2	暂存连续簇数
01D8h	6	字库文件名,'HZK16',0
01E0h	26h	HZK16的文件控制块(FCB)
0200h	2	内部词组区首址
0202h	2	内部词组区尾址
0204h	2	上次从磁盘取点阵数据汉字的顺序号
0206h	2	字库常驻内存区起始段址
020Ah	2	每扇区字节数
020Ch	2	每道扇区数
020Eh	2	磁头数
02C0h	20h	字库文件(HZK16)的链表区
02E0h	8	错误信息,'ERROR!',7,'$'

四、程序清单

```
;FILE1A.ASM
CODE            SEGMENT
                ASSUME CS:CODE,DS:CODE
                ORG     100H

L_0100:
                MOV     SP,0E0H                 ;修改堆栈指针
;由于要读入一级汉字的点阵数据(约需 130K 内存空间), 所以, 本程序占用内存空间将
;超过 64K, 因此必须将堆栈指针切换到安全的地方.
```

```
;读入硬盘主引导扇区，目的是为了查找 DOS 分区的起始物理地址
                MOV     AX,201H                 ;AH=2:调用读扇区功能,AL=1:读
                                                ;入1个扇区
                MOV     BX,800H                 ;BX=读入数据存放地址
                MOV     CX,1
                MOV     DX,80H                  ;物理地址为:C盘0道0面1扇区
                INT     13H                     ;读入主引导扇区

;查找 DOS 分区
                MOV     SI,OFFSET DS:[1BEH]     ;SI=分区表起始位移
L_0114:
                CMP     BYTE PTR [BX+SI+4],1
                JE      L_012B                  ;是12 bits FAT 的DOS分区
                CMP     BYTE PTR [BX+SI+4],4
                JE      L_012B                  ;是16 bits FAT 的DOS分区
                CMP     BYTE PTR [BX+SI+4],6
                JE      L_012B                  ;是保留的DOS分区

;当前分区项不是 DOS 分区项
                ADD     SI,10H                  ;SI=下一分区项位移
                JMP     SHORT L_0114            ;检查下一分区项

L_012B:
;读入分区引导扇区(BOOT)，目的是为了知道当前分区的基本信息
                MOV     DH,[BX+SI+1]            ;起始磁头号
                MOV     CX,[BX+SI+2]            ;起始扇区号和磁道号
                MOV     AX,201H                 ;读入该分区的第一个扇区，即该
                                                ;分区的引导扇区
                INT     13H

;读入文件定位表(FAT)
                MOV     AL,[BX+16H]             ;一个FAT占用扇区数
                INC     AX                      ;加上BOOT扇区
;此时，CX,DX 还是当前分区的起始物理地址，而 FAT 正好紧接在 BOOT 扇区的后面，为避免
;复杂的物理地址的计算，这里将 BOOT 扇区和 FAT 一起读入内存.
                MOV     AH,2
                INT     13H                     ;读入BOOT扇区和FAT
                                                ;存放地址仍为800H

;建立 HZK16 的扇区链表
                CALL    L_0300                  ;调用子程序建立扇区链表

;初始化内部词组区
                MOV     D_0200,BP               ;内部词组区首址
;调用 L_0300 返回时，BP=可用内存首址

                MOV     AL,DS:[82H]             ;取命令行参数1
                AND     AL,0FH                  ;屏蔽高4位，将数字转换为数值
                XOR     AH,AH
                MOV     DX,400H                 ;1K=1024字节
                MUL     DX                      ;AX=内部词组区长度(单位:字节)
                ADD     AX,BP                   ;AX=内部词组区尾址
                MOV     D_0202,AX
```

```
;读入一级汉字（1－3、9、16－55区）的点阵数据，此时AX为内存空闲区首址
        MOV     CL,4
        SHR     AX,CL                   ;AX=AX/16,转换为节地址
        INC     AX                      ;因为AX不一定刚好被16整除,故
                                        ;节地址要加1
        MOV     DX,CS                   ;DX=当前段址
        ADD     AX,DX                   ;AX=内存空闲区段址，也即字库
                                        ;常驻区的起始段址
        MOV     D_0206,AX
        PUSH    AX

        MOV     DX,OFFSET D_01D8        ;HZK16文件名地址
        MOV     AX,3D00H
        INT     21H                     ;打开HZK16
        MOV     BX,AX                   ;BX=文件句柄号

;读入第1-3区汉字
        POP     DS                      ;DS=字库常驻区起始段址
        XOR     DX,DX                   ;DS:DX=读入数据存放地址
        MOV     CX,2340H                ;2340H/32/94=3区汉字
;每个汉字点阵数据占32字节，每区共94个汉字.

        MOV     AH,3FH
        INT     21H                     ;读入第1-3区汉字

;越过第4-8区汉字
        MOV     DX,3AC0H                ;3AC0H/32/94=5个区
        XOR     CX,CX                   ;CX=0
        MOV     AX,4201H                ;移动文件指针，从当前位置往
                                        ;下移动
        INT     21H                     ;忽略第4-8区汉字，此时文件指
                                        ;针已指向第9区汉字
;读入第9区汉字
        MOV     AX,DS
        ADD     AX,234H                 ;上次读入数据的节长度
        MOV     DS,AX                   ;修改数据段址，指向内存空闲区
        XOR     DX,DX                   ;DS:DX=读入数据存放地址
        MOV     CX,0BC0H                ;0BC0H/32/94=1
        MOV     AH,3FH
        INT     21H                     ;读入第9区汉字(制表符区)

;越过第10-15区汉字
        MOV     DX,4680H                ;4680H/32/94=6个区
        XOR     CX,CX
        MOV     AX,4201H                ;移动文件指针，从当前
                                        ;往下移动
        INT     21H                     ;忽略第10-15区汉字，此时文件
                                        ;指针已指向第16区汉字
;读入第16-35区汉字
        MOV     AX,DS
        ADD     AX,0BCH                 ;上次读入数据的节长度
        MOV     DS,AX                   ;修改数据段址，指向内存空闲区
        XOR     DX,DX
        MOV     CX,0EB00H               ;0EB00H/32/94=20个区
        MOV     AH,3FH
```

```
                INT     21H                     ;读入16-35区汉字
;读入第 36-55 区汉字
                MOV     AX,DS
                ADD     AX,0EB0H                ;上次读入数据的节长度
                MOV     DS,AX                   ;修改数据段址
                MOV     AH,3FH
                INT     21H                     ;读入36-55区汉字

;至此，所有常驻内存汉字的点阵数据已全部读入，下面关闭 HZK16 文件
                MOV     AH,3EH
                INT     21H                     ;关闭文件

;计算程序驻留长度(单位:节)
                MOV     AX,DS
                ADD     AX,0EB0H                ;AX=内存空闲区段址
                MOV     DX,CS
                SUB     AX,DX                   ;减去起始段址=驻留长度
                ADD     AX,10H                  ;留点余量
                PUSH    AX                      ;保存驻留长度

                PUSH    CS
                POP     DS                      ;恢复DS=CS
                MOV     DX,OFFSET L_0211        ;INT 7FH入口地址
                MOV     AX,257FH
                INT     21H                     ;设置INT 7FH

                POP     DX                      ;DX=驻留长度(单位:节)
                MOV     AX,3100H
                INT     21H                     ;驻留退出

                DB      0

D_01D8          DB      'HZK16',0               ;HZK16的文件名
                DB      0,0
D_01E0          DB      3                       ;HZK16的文件控制块(FCB)
                DB      'HZK16                  ;文件名
                DB      20 DUP (0)

D_0200          DW      0                       ;内部词组区首址
D_0202          DW      0                       ;内部词组区尾址
D_0204          DW      0                       ;上次从磁盘取点阵数据汉字的顺
                                                ;序号
;当连续读取同一汉字的点阵数据时，因为第 1 次读入的点阵数据并没有丢失，因此可以
;重用.

D_0206          DW      0                       ;字库常驻内存区起始段址
                DB      0, 0                    ;没有用
D_020A          DW      0                       ;每扇区字节数
D_020C          DW      0                       ;每道扇区数
D_020E          DW      0                       ;磁头数
                DB      2                       ;没有用

;INT 7FH
;入口: DX = 汉字
;出口: DX:0 = 汉字点阵数据地址
```

```
L_0211:
                PUSH    AX
                PUSH    DS                      ;保存寄存器

                PUSH    CS
                POP     DS                      ;DS=CS

;判断指定汉字的点阵数据是否在内存中
                AND     DX,7F7FH                ;屏蔽高位
                CMP     DH,24H
                JB      L_0234                  ;是第1-3区汉字，转内存处理
                CMP     DH,29H
                JNE     L_0227                  ;不是第9区汉字
                MOV     DH,24H                  ;第9区汉字在内存中的相对区号
                                                ;为4，故要修改DH的值
                JMP     SHORT L_0234
L_0227:
                CMP     DH,30H
                JB      L_024B                  ;是第16区以下的汉字
                CMP     DH,57H
                JA      L_024B                  ;是二级汉字
                SUB     DH,0BH                  ;DH=一级汉字在内存中的相对区号
;一级汉字在内存中从相对区号 5 开始存放

L_0234:
;下面处理常驻内存汉字，先计算该汉字在内存中的顺序号
                SUB     DX,2121H
                MOV     AL,5EH                  ;每区=94个汉字
                MUL     DH                      ;区号 x 94
                XOR     DH,DH
                ADD     AX,DX                   ;AX=顺序号
                SHL     AX,1                    ;AX=该汉字在内存中的位移(单
                                                ;位:节)
;每个汉字占 32 字节，因此，顺序号为 AX 的汉字的相对位移为 AXx32(字节)=AXx2(节)

                ADD     AX,D_0206               ;加点阵数据区起始段址
                MOV     DX,AX                   ;DX=所求汉字点阵数据的段址

                POP     DS
                POP     AX                      ;恢复寄存器

                IRET                            ;中断返回

;以下处理非常驻内存汉字
L_024B:
;计算该汉字在字库中的顺序号
                SUB     DX,2121H
                MOV     AL,5EH                  ;每区为94个汉字
                MUL     DH
                XOR     DH,DH
                ADD     AX,DX                   ;AX=顺序号

                CMP     AX,D_0204               ;和上次读的是同一个汉字吗?
                JE      L_02B9                  ;是，不用再从磁盘读取了
```

```
        MOV     D_0204,AX               ;保存该汉字顺序号，下次如果仍
                                        ;读该汉字的点阵数据,则可直接
                                        ;从内存读取
        PUSH    CX
        PUSH    BX
        PUSH    DI
        PUSH    SI
        PUSH    ES                      ;进一步保存寄存器

        PUSH    CS
        POP     ES                      ;ES=CS

;计算包含该汉字点阵数据的扇区在 HZK16 中的相对扇区号
        XOR     DX,DX
        MOV     CX,10H                  ;一个扇区可以存放16个汉字的点
                                        ;阵数据
        DIV     CX                      ;AX=相对扇区号
        PUSH    DX                      ;DX=在扇区中的相对编号

;查链表，取包含该汉字点阵数据扇区的逻辑扇区号
        MOV     BX,OFFSET D_02C0        ;链表首址
L_0272:
        CMP     AX,[BX+2]               ;在当前链项中吗?
        JB      L_027F                  ;是
        SUB     AX,[BX+2]               ;不是
;AX=从下一链开始的相对扇区号

        ADD     BX,4                    ;BX=下一链项地址
        JMP     SHORT L_0272            ;继续查找

L_027F:
        ADD     AX,[BX]                 ;AX=点阵数据所在扇区的逻辑扇
                                        ;区号

;根据逻辑扇区号计算物理地址
        XOR     DX,DX
        DIV     D_020C                  ;除以每道扇区数
        MOV     CX,DX
        INC     CL                      ;CL=绝对扇区号
        XOR     DX,DX
        DIV     D_020E                  ;除以磁头数
        MOV     CH,AL                   ;CH=磁道号，高2位数据还在AH中
        MOV     DH,DL                   ;DH=磁头号
        MOV     AL,40H
        MUL     AH                      ;相当于SHL AH,6
        ADD     CL,AL                   ;将磁道号的高2位放入CL中
;由于硬盘的磁道数可能超过 255，需要 10 位数据，因此不能用 CH 来完全表示. 同时，由
;于每道扇区数不会超过 64，因此，可以将磁道数的高 2 位数据保存在 CL 的高 2 位中

;读入包含该汉字点阵数据的扇区
        MOV     DL,80H                  ;C盘
        XOR     BX,BX                   ;读入数据存放地址
        MOV     AX,201H                 ;读一个扇区
        INT     13H
```

```
                POP     AX                      ;AX=在扇区中的相对编号
                OR      AX,AX                   ;是该扇区的第一汉字吗?
                JZ      L_02B4                  ;是
                MOV     CX,20H                  ;32个字节
                MUL     CX
                MOV     SI,AX                   ;SI=点阵数据首址
                XOR     DI,DI
                REP     MOVSB                   ;传送到CS:0
L_02B4:
                POP     ES
                POP     SI
                POP     DI
                POP     BX
                POP     CX                      ;恢复寄存器
L_02B9:
                PUSH    CS
                POP     DX                      ;DX=CS=汉字点阵数据的段址

                POP     DS
                POP     AX                      ;恢复寄存器

                IRET                            ;中断返回

                DB      0, 0                    ;没有用
D_02C0          DB      32 DUP (0)              ;HZK16的链表区
D_02E0          DB      'ERROR!',7,'$'          ;错误信息
                DB      8 DUP (0)               ;没有用

;字库文件名(HZK16)没有找到, 结束运行, INT 7FH 没有设置
L_02F0:
                POPF
                MOV     DX,OFFSET D_02E0
                MOV     AH,9
                INT     21H                     ;显示错误信息

                INT     20H                     ;结束运行

                DB      6 DUP (0)               ;没有用

;子程序: 建立字库文件(HZK16)的扇区链表
;输   入: BP=链表首址
;         BX=FAT 首址
;输   出: BP=链表尾址, 也即可用内存空间首址
L_0300:
                MOV     AX,[BX+0BH]             ;每扇区字节数
                MOV     D_020A,AX
                MOV     AX,[BX+18H]             ;每道扇区数
                MOV     D_020C,AX
                MOV     AX,[BX+1AH]             ;磁头数
                MOV     D_020E,AX
                MOV     AL,[BX+0DH]             ;每簇扇区数
                XOR     AH,AH
                MOV     DS:[0F0H],AX
L_031A:
                MOV     AX,[BX+16H]             ;一个FAT占用扇区数
```

```
MUL     BYTE PTR [BX+10H]       ;剩以FAT数目=FAT所占总扇区数
ADD     AX,[BX+0EH]             ;加保留扇区数
ADD     AX,[BX+1CH]             ;加隐含扇区数=根目录起始逻辑
                                ;扇区号
MOV     DS:[0F4H],AX
MOV     AX,20H                  ;每个目录项长度=32字节
MUL     WORD PTR [BX+11H]       ;剩以根目录项数=根目录长度
DIV     D_020A                  ;除以每扇区字节数=根目录所占
                                ;扇区数
ADD     DS:[0F4H],AX            ;加根目录起始逻辑扇区号=数据
                                ;区起始逻辑扇区号

MOV     BX,0A00H                ;BX=FAT首址
MOV     BP,2C0H                 ;BP=链表首址

;修改磁盘数据传送区地址(DTA), 因为缺省的 DTA 在 80H 处, 它刚好和命令行内容区重叠,
;如果不修改 DTA, 执行以下程序将会丢失命令行内容
MOV     DX,400H                 ;新DTA地址
MOV     DI,DX
MOV     AH,1AH                  ;设置DTA功能
INT     21H

MOV     DX,OFFSET D_01E0        ;文件控制块(FCB)地址
MOV     AH,11H
INT     21H                     ;寻找匹配文件名(HZK16)
OR      AL,AL                   ;找到?
JNZ     L_2F0                   ;没有, 转非正常结束
MOV     AX,[DI+1BH]             ;取文件起始簇号
PUSH    AX
SUB     AX,2                    ;减去两个保留簇号=实际簇号
;FAT 的前两项为保留簇号

MOV     DX,DS:[0F0H]            ;每簇扇区数
MUL     DX
ADD     AX,DS:[0F4H]            ;AX=文件起始扇区的逻辑扇区号
MOV     [BP],AX                 ;写入链表
POP     AX                      ;恢复起始簇号
MOV     WORD PTR DS:[0F6H],0    ;连续簇数=0,以后每当发现一连
                                ;续簇时,该单元加1
L_036C:
PUSH    AX
INC     WORD PTR DS:[0F6H]      ;连续簇数加1
MOV     SI,AX                   ;SI=当前簇号
TEST    BYTE PTR [BX+4],40H
;对于 12 bits FAT, 每个簇号用 12 位表示. FAT 中, 前 3 字节为保留簇号占用. 第 4 字节
;的低 4 位和第 3 字节组成簇号 2, 第 5 字节和第 4 字节的高 4 位组成簇号 3. 一般情况下, 硬
;盘最前面的数据为两个系统文件占用, 而且必须连续存放, 这样簇号 2 的数值应为 3,
;簇号 3 的数值应为 4, 即第 4 字节数值为 40H

JNZ     L_0382                  ;12 bits FAT转
                                ;处理16 bits FAT
ADD     SI,AX
MOV     AX,[BX+SI]              ;AX=下一簇号
CMP     AX,0FFF8H               ;是文件最后一簇吗?
JMP     SHORT L_0398
```

```
                                        ;处理12 bits FAT
L_0382:
        SHR     AX,1
        PUSHF                           ;保存CF
        ADD     SI,AX                   ;SI = AX x 1.5
        MOV     AX,[BX+SI]              ;取簇号(仅12位有效)
        POPF                            ;恢复CF
        JNC     L_0392                  ;CF=0, 表示是偶数簇
;当簇号为奇数时, 表示 AX 的高 12 位是下一簇号值, 反之, 表示 AX 的的低 12 位是
;下一簇号值

        MOV     CL,4                    ;奇数簇
        SHR     AX,CL                   ;右移4位, 去掉低4位
                                        ;AX=下一簇号
        JMP     SHORT L_0395
L_0392:
        AND     AX,0FFFH                ;偶数簇, 屏蔽高4位
                                        ;AX=下一簇号
L_0395:
        CMP     AX,0FF8H                ;是文件最后一簇吗?
L_0398:
        POP     SI                      ;恢复当前簇号
        JC      L_03A5                  ;不是最后一簇

;处理文件的最后一簇
        MOV     AX,0FFFFH               ;最后簇标记
        MOV     [BP+2],AX               ;写入链表结束标记
        ADD     BP,4                    ;BP=内存可用空间首址
        RETN

L_03A5:
        INC     SI                      ;当前簇号加1
;如果是连续簇, 那么下一簇号应该=SI
        CMP     AX,SI                   ;是连续簇吗?
        JE      L_036C                  ;是
;非连续簇, 增加链项
        PUSH    AX                      ;保存下一簇号
        MOV     AX,DS:[0F0H]            ;AX=每簇扇区数
        MUL     WORD PTR DS:[0F6H]      ;剩以每簇扇区数=连续扇区数
        MOV     [BP+2],AX               ;写入当前链的连续扇区数
        XOR     AX,AX
        MOV     DS:[0F6H],AX            ;连续簇数=0
        MOV     AX,DS:[0F0H]            ;每簇扇区数
        POP     SI                      ;SI=下一簇号
        PUSH    SI                      ;保存下一簇号
        SUB     SI,2                    ;SI=实际簇号
        MUL     SI
        ADD     AX,DS:[0F4H]            ;AX=逻辑扇区号
        ADD     BP,4
        MOV     [BP],AX                 ;保存下一链项的起始逻辑扇区号
        POP     AX                      ;AX=下一簇号
        JMP     SHORT L_036C
CODE    ENDS
        END     L_0100
```

第三节　全部字库驻留内存　FILE2.COM

一、使用说明

FILE2←┘

本程序在启动时，将整个显示字库读入内存，这样可以避免直接从硬盘读取汉字点阵数据，明显地提高了汉字显示速度。另一方面，由于它将显示字库全部读入内存，占用了大量内存空间（约256K），使得计算机不能运行一些需较大内存空间的程序。通常本程序在进行文字处理时使用。

注意：本程序不能指定内部词组区大小。因此，在使用本程序启动后，系统内部词组区空间为0，不能进行内部词组定义。（参见CCCC.ASM）

二、程序功能

本程序主要包括两个部分，即初始化代码和INT　7FH代码。

初始化代码主要完成以下工作：

1）将显示字库全部读入内存。

2）设置INT　7FH。

3）驻留退出。

程序运行后，便可用INT　7FH读取汉字的点阵数据（16x16点阵）。

INT　7FH的运行过程如下：

1）计算指定汉字的顺序号。

2）从内存中读取该汉字的点阵数据。

3）返回该汉字点阵数据的地址。

三、程序清单

```
;FILE2.ASM
CODE            SEGMENT
                ASSUME CS:CODE,DS:CODE
                ORG     100H

L_0100:
                MOV     SP,100H                 ;修改堆栈指针
;由于要读入全部汉字的点阵数据(约需 256K 内存空间), 所以, 本程序占用内存空间将
;超过 64K, 因此必须将堆栈指针切换到安全的地方.

                MOV     DX,D_0180               ;DX=字库文件名(HZK16)地址
                MOV     AX,3D00H
                INT     21H                     ;打开HZK16
                JNC     L_0119                  ;文件打开成功

;字库文件打开错误, 非正常结束

L_010D:
                MOV     DX,158H                 ;错误信息地址
                MOV     AH,9
                INT     21H                     ;显示错误信息

                MOV     AX,4CFFH                ;AL=0FFH=返回错误代码
                INT     21H                     ;结束运行

;字库文件打开成功
L_0119:
                MOV     BX,AX                   ;BX=文件句柄号
```

```
                MOV     AX,CS
                ADD     AX,20H                  ;AX=CS+20h(单位:节), 相当于
                                                ;CS:200H, 指向空闲内存区起始
                                                ;段址
                MOV     WORDPTR DS:[175H],AX    ;字库内容起始段址,修改指令中
                                                ;的数据地址
                MOV     DS,AX
L_0125:
                XOR     DX,DX                   ;DS:DX=读入数据存放地址
                MOV     CX,8000H                ;每次读入32K字节
                MOV     AH,3FH                  ;读文件功能
                INT     21H
                JC      L_010D                  ;读文件失败, 转非正常结束
                OR      AX,AX                   ;已读完了吗?
                JZ      L_013D                  ;是
;调用读文件功能后, AX 返回读入字节数, 若 AX=0, 即表示文件读完

;修改 DS, 继续读入字库内容
                MOV     AX,DS
                ADD     AX,800H                 ;800H节=8000H字节
                MOV     DS,AX                   ;DS=空闲内存区段址
;由前述可知, 最后一次读入的文件字节数一般总是小于预定的字节数(这里为 32K), 显
;然, 最后一次读操作后, DS 的调整很可能是错误的, 也即调整后的 DS 不是实际字库内容
;的结束地址. 接下来我们将看到, 本程序在处理时直接将 DS 当作字库内容的结束地址了.
;因此, 浪费了不少内存空间
                JMP     SHORT L_0125            ;继续读

;文件读入完毕
L_013D:
                MOV     AX,DS                   ;计算驻留长度(单位:节)
                MOV     BX,CS
                SUB     AX,BX                   ;减去基地址=驻留长度

                PUSH    AX                      ;保存驻留长度

;设置INT 7FH, 并驻留退出
                PUSH    CS
                POP     DS                      ;恢复DS=CS
                MOV     DX,OFFSET L_0160        ;INT 7FH的入口地址
                MOV     AX,257FH
                INT     21H                     ;设置INT 7FH

                POP     DX                      ;DX=驻留的长度(单位:节)
                MOV     AX,3100H
                INT     21H                     ;驻留退出

                DB      0, 0, 0, 0              ;没有用
D_0158          DB      'ERROR!', 7, '$'        ;错误信息

;INT 7FH:
;入口: DX = 汉字
;出口: DX:0 = 汉字点阵数据地址
L_0160:
                PUSH    AX                      ;保存寄存器
```

```
;计算汉字顺序号
                AND     DX,7F7FH                ;屏蔽高位
                SUB     DX,2121H
;汉字机内码是从 ASCII 码的 0A1H(161)开始的，以上两条指令相当于指令 SUB DX,0A1A1H
                MOV     AL,5EH                  ;每区有94个汉字
                MUL     DH
                XOR     DH,DH
                ADD     AX,DX                   ;AX=汉字顺序号
                SHL     AX,1                    ;AX=该汉字在内存中的位移(单
                                                ;位:节)
;每个汉字占 32 字节，因此，顺序号为 AX 的汉字点阵数据在内存中的相对位移为 AXx32
;(字节)=AXx2(节)

                ADD     AX,0                    ;加上字库区起始段址
;指令 ADD AX,0 中的数据 0，在初始化时已被修改为具体的段址了

                MOV     DX,AX                   ;DX=该汉字点阵数据的段址

                POP     AX                      ;恢复寄存器

                IRET                            ;中断返回
CODE            ENDS
                END     L_0100
```

第四节　读虚拟盘字库　ＦＩＬＥ３．ＣＯＭ

一、使用说明

ＦＩＬＥ３　ａｂ←┘

其中：ａ表示虚拟盘名称，必须是第一个虚拟盘。

ｂ表示内部词组区大小，以５０个汉字为单位。

例如：ＦＩＬＥ３　Ｅ２表示虚拟盘名称为Ｅ盘，Ｄ盘必须是硬盘，系统启动时自动把ＨＺＫ１６拷贝到Ｅ盘，以后从Ｅ盘读取汉字的点阵数据。内部词组区为２ｋ字节。

目前国内使用的许多２８６、３８６微机，其内存都在１Ｍ或１Ｍ以上，但由于目前的ＤＯＳ只能管理６４０Ｋ内存空间（加上显示缓冲区和ＲＯＭ区共为１Ｍ）。其它１Ｍ以外的内存空间（又称扩展内存）一般只能通过建立虚拟盘，暂时保存数据。当系统重新启动时，保存的数据也将丢失，因此，扩展内存通常都闲置着不用。２．１３Ｈ汉字系统可将显示字库以文件的方式拷入虚拟盘，汉字显示时，由ＦＩＬＥ３．ＣＯＭ直接从扩展内存中读取汉字的点阵数据，这样既节省了大量的内存空间，又可保证汉字的显示速度，还可以充分使用硬件资源，虚拟盘的剩余空间仍可随意使用，可谓是三全其美的好方法。

二、程序功能

本程序主要包括两个部分，即初始化代码和ＩＮＴ　７ＦＨ代码。

初始化代码主要完成以下工作：

１）在指定的虚拟盘上查找字库文件（ＨＺＫ１６），并计算出ＨＺＫ１６的首址。

２）初始化内部词组区。

３）设置ＩＮＴ　７ＦＨ。

４）驻留退出。

程序运行后，使可用ＩＮＴ　７ＦＨ读取汉字的点阵数据（１６ｘ１６点阵）。

ＩＮＴ　７ＦＨ的运行过程如下：

１）根据指定汉字的顺序号，计算该汉字的点阵数据地址。

２）设置ＧＤＴ，通过ＩＮＴ　１５Ｈ读取扩展内存中的点阵数据。

３）返回所需汉字的点阵数据地址。

三、变量名表

地址	长度	意义
0082h	1	命令行参数1，指定虚拟盘名称
0083h	1	命令行参数2，指定内部词组大小
0180h	40h	描述符表地址
0192h	2	源数据地址(低16位)
0194h	1	源数据地址(高8位)
019Ah	2	目标数据地址(低16位)
019Ch	1	目标数据地址(高8位)
01D0h	Ah	错误信息,'ERROR!',7,0DH,0AH,'$'
01E0h	11	字库文件名,'HZK16
0200h	2	内部词组区首址
0202h	2	内部词组区尾址

四、程序清单

```
;FILE3.ASM
CODE            SEGMENT
                ASSUME  CS:CODE,DS:CODE
                ORG     100H

L_0100:
                JMP     L_0204                  ;转初始化程序

;INT 7FH
;入口: DX = 汉字
;出口: DX:0 = 汉字点阵数据地址

L_0103:
                PUSH    DS
                PUSH    ES
                PUSH    AX
                PUSH    CX
                PUSH    SI                      ;保存寄存器

                PUSH    CS
                POP     DS                      ;DS=CS
                PUSH    CS
                POP     ES                      ;ES=CS

;计算汉字顺序号
                AND     DX,7F7FH                ;屏蔽高位
                SUB     DX,2121H
                MOV     AL,5EH                  ;每区为94个汉字
                MUL     DH
                XOR     DH,DH
                ADD     AX,DX                   ;AX=顺序号

;计算指定汉字的点阵数据地址(在扩展内存中，其值必定大于1M)
                MOV     DL,20H                  ;每个汉字占32字节
                MUL     DX                      ;DX:AX=该汉字的点阵数据在字库
                                                ;中的相对位移
                ADD     DL,0                    ;加高8位的起始值
                ADD     AX,0                    ;加上低16位的起始值
;上面两条指令中的0已在初始化时，被修改为实际HZK16的首址。由于HZK16在扩展内存
```

```
;中，因此，其首址必定大于 1M，在此要用 24 位数来表示(DL 表示高 8 位，AX 表示低 16 位).
                JNC     L_0129                  ;AX没有进位转
                INC     DX                      ;AX有进位，应该将该数加到DX中

L_0129:
;建立描述符表的源数据地址指针
                MOV     D_0194,DL               ;源数据地址的高8位
                MOV     D_0192,AX               ;源数据地址的低16位

;建立描述符表的目标数据地址指针，地址为 CS:0
                MOV     DX,CS                   ;DX=CS=目标数据段址
                AND     DH,0F0H                 ;保留高4位数据
                MOV     CL,4
                SHR     DH,CL                   ;DH=目标数据地址的位16-19
                MOV     AX,CS
                AND     AH,0FH                  ;去掉高4位
                SHL     AX,CL                   ;AX=目标数据地址的低16位
                MOV     D_019A,AX
                MOV     D_019C,DH
;DH 中实际上只有 4 位数据有效，故目标数据地址是 20 位的，小于 1M

                MOV     SI,180H                 ;ES:DI指向全程描述符表(GDT)
                MOV     CX,10H                  ;CX=传送数据长度(单位:字)
                MOV     AH,87H                  ;数据传送功能
                INT     15H                     ;从扩展内存中取指定汉字的点阵
                                                ;数据到CS:0
                MOV     DX,CS                   ;DX=汉字点阵数据起始段址

                POP     SI
                POP     CX
                POP     AX
                POP     ES
                POP     DS                      ;恢复寄存器

                IRET                            ;中断返回

                DB      39 DUP (0)              ;没有用

;在调用 INT 15H 进行数据传送前，程序必须先建立一个称为描述符表(描述符由称做局
;部描述符表(LDT)或全程描述符表(GDT)的结构组成)，并且提供传送数据的源地址和目
;标地址，这些地址均为 24 位的.
D_0180
            DW      0                       ;空描述符，均应为0
                DW      0
                DB      0
                DB      0
                DW      0

                DW      0                       ;GDT 描述符，由BIOS处理
                DW      0
                DB      0
                DB      0
                DW      0

;源数据描述符，由程序填写
```

```
                DW      0020H                   ;数据段长度=32字节
D_0192          DW      0                       ;地址的低16位
D_0194          DB      0                       ;地址的高8位
                DB      93H                     ;存取权限，93H表示可写的数据
                                                ;段
                DW      0                       ;保留
;目标数据描述符，由程序填写
                DW      0020H                   ;数据段长度=32字节
D_019A          DW      0                       ;地址的低16位
D_019C          DB      0                       ;地址的高8位
                DB      93H                     ;存取权限，93H表示可写的数据
                                                ;段
                DW      0                       ;保留

                DW      0                       ;BIOS代码段描述符，由BIOS处理
                DW      0
                DB      0
                DB      0
                DW      0

                DW      0                       ;堆栈段描述符，由BIOS处理
                DW      0
                DB      0
                DB      0
                DW      0

                DB      32 DUP (0)              ;没有用

D_01D0          DB      'ERROR!', 7, 0DH, 0AH, '$' ;错误信息
                DB      6 DUP (0)               ;没有用
D_01E0          DB      'HZK16                  ;文件名

                DB      21 DUP (0)              ;没有用

D_0200          DW      204H                    ;内部词组区首址
D_0202          DW      0                       ;内部词组区尾址

L_0204:
;初始化程序，在系统启动后，被用作内部词组区

;读入指定虚拟盘的引导扇区(BOOT)
                MOV     AL,DS:[82H]             ;取命令行参数1,AL=虚拟盘名称
                AND     AL,5FH                  ;将小写改为大写
                SUB     AL,41H                  ;AL=虚拟盘号
                MOV     DS:[82H],AL             ;保存虚拟盘号

                MOV     BX,300H                 ;读入数据存放地址
                MOV     CX,1                    ;读一个扇区
                XOR     DX,DX                   ;逻辑扇区号为0
                INT     25H                     ;读入该虚拟盘的引导扇区(BOOT)

;计算虚拟盘中数据区的首址
                MOV     AL,DS:[316H]            ;一个FAT占用扇区数
                MUL     BYTE PTR DS:[310H]      ;乘以FAT数目=FAT所占总扇区数
                ADD     AX,DS:[30EH]            ;加保留扇区数=根目录起始逻辑
```

```
                                        ;扇区号
        PUSH    AX                      ;保存根目录起始逻辑扇区号
        MUL     WORD PTR DS:[30BH]      ;剩每扇区的字节数
        MOV     CX,AX
        MOV     AX,20H                  ;每个目录项长度=32字节
        MUL     WORD PTR DS:[311H]      ;剩以根目录项数=根目录长度
        ADD     AX,CX                   ;AX=数据区首址(单位:字节)
        MOV     WORD PTR DS:[124H],AX   ;修改指令中的数据,先保存数据
                                        ;区首址

;读入虚拟盘的根目录
        POP     DX                      ;DX=根目录起始逻辑扇区号
        MOV     AX,DS:[311H]            ;根目录项数
        MOV     CL,4
        SHR     AX,CL                   ;根目录所占扇区数
;每个目录项=32 字节, 每扇区为 512 字节, 因为 32x16=512, 故 AX 右移 4 位即为根目录所
;占扇区数

        MOV     CX,AX                   ;读入扇区数
        MOV     BX,400H                 ;读入数据存放地址
        MOV     AL,DS:[82H]             ;虚拟盘号
        INT     25H                     ;读入虚拟盘根目录区

        MOV     SI,BX                   ;根目录首址
L_024A:
        MOV     DI,OFFSET D_01E0        ;HZK16文件名地址
        MOV     CX,0BH                  ;文件名长度=11字节(包括空格)
        MOV     AL,[SI]                 ;AL=文件名第一字节,若为0则表
                                        ;示是没有使用的目录项
        OR      AL,AL                   ;已经是没有使用的目录项了吗?
        JZ      L_02A0                  ;是, 不用再检查以下的目录项了,
                                        ;表示指定虚拟盘中没有HZK16文件

        PUSH    SI
        REPE    CMPSB                   ;当前目录项是HZK16吗?
        POP     SI
        JZ      L_0261                  ;是
        ADD     SI,20H                  ;SI=下一个目录项地址
        JMP     SHORT L_024A            ;查找下一目录项

;已找到 HZK16 目录项
L_0261:
        MOV     AX,[SI+1AH]             ;取HZK16的首簇号
        SUB     AX,2                    ;减去两个保留簇号=实际簇号
        MOV     DL,DS:[30DH]            ;每簇扇区数
        XOR     DH,DH
        MUL     DX                      ;DX=从数据区开始的相对扇区号
        MOV     DX,200H                 ;每扇区长度=512字节
        MUL     DX                      ;DX:AX=起始字节长度
        ADD     WORD PTR DS:[124H],AX   ;计算该文件起始地址的低16位
;因为上面已将数据区首址保存在 124H 中, 因此, 此时仅需将从数据区开始的低 16 位加
;到 124H 中, 即得 HZK16 首址的低 16 位
        ADC     DX,10H                  ;加基数1M, 若低16位有进位
                                        ;则高8位加1
;DL 和 AX 组成 24 位地址, 因此, DL=10H 即表示 1M
```

```
                MOV     BYTE PTR DS:[122H],DL   ;写入高8位数据

                MOV     DX,OFFSET L_0103        ;INT 7FH的入口地址
                MOV     AX,257FH
                INT     21H                     ;设置INT 7FH
;初始化内部词组区
                MOV     AL,DS:[83H]             ;取命令行参数2
                AND     AL,0FH                  ;将数字转换为数值
                XOR     AH,AH
                MOV     DX,400H                 ;1K=1024字节
                MUL     DX                      ;AX=内部词组长度
                ADD     AX,204H                 ;加内部词组区首址=内部词组区
                                                ;尾址
                MOV     D_0202,AX
                ADD     AX,104H                 ;留点余量
                MOV     DX,AX                   ;驻留长度
                INT     27H                     ;驻留退出

;虚拟盘中没有 HZK16 文件，结束运行，INT 7FH 没有设置

L_02A0:
                MOV     DX,1D0H                 ;错误信息地址
                MOV     AH,9
                INT     21H                     ;显示错误信息

                INT     20H                     ;结束运行
CODE            ENDS
                END     L_0100
```

第三章　打　印　字　库　读　取

第一节　读16点阵字库　FILE16B.COM

一、使用说明

FILE16B↵

汉字的16点阵打印字库的点阵数据与显示字库内容实质是一样，但由于用于打印的点阵数据必须是纵向排列的，而显示字库点阵数据是横向排列的。因此首先要将显示字库点阵数据转换为纵向排列的点阵数据，同时还要根据字型进行扩大和缩小处理。除了汉字以外，本程序还可读取ASCII字符的点阵数据（16 x 8点阵），字符点阵数据全部包含在本程序内部，而且这些点阵数据已经是按纵向排列的。

二、程序功能

本程序主要包括两个部分，即初始化代码和INT　7AH代码。

初始化代码主要完成以下工作：

1）设置INT　7AH。

2）驻留退出。

程序运行后，便可调用INT　7AH读取汉字的打印点阵数据（16点阵，可以进行相应缩放）。INT　7AH运行过程如下：

1）若DX是字符，那么从字符点阵数据区读取点阵数据，然后，根据需要转换为上标或下标字符。

2）若DX是汉字，那么调用INT　7FH读入汉字显示点阵数据，按需要进行左旋或右旋处理，若不需旋转，则将原点阵转换为纵向排列点阵数据。

3）按需要对字符或汉字放大至24点阵。

4）返回处理后点阵数据的地址。

三、变量说明

地址	长度	意义
0100h	1	字型字节
0101h	1	修饰字节
0102h	1	点阵数据列数
0260h	800h	16x8点阵ASCII字符的点阵数据

四、程序清单

```
;FILE16B.ASM
CODE            SEGMENT
                ASSUME CS:CODE, DS:CODE
                ORG     100H

L_0100:
                JMP     L_0A60                  ;转初始化程序
                NOP
;上面三字节，在启动后又被作为变量单元存放数据

;INT 7AH
;输入: AH, 位5=1, 将16点阵数据放大至24点阵
;       BL, 位3=1, 将汉字点阵数据左旋90度(字符无效)
```

```
;              位4=1, 将汉字点阵数据右旋90度(字符无效)
;              位5=1, 取上标字符(汉字无效)点阵数据
;              位6=1, 取下标字符(汉字无效)点阵数据
;         DX, 若DH=0,则DL=ASCII字符, 否则DX=汉字
;输出: DS:SI=读出点阵数据首址
;         CX=点阵数据列数, 每排24点(3字节).
L_0104:
                PUSH    DI
                PUSH    AX
                PUSH    BP
                PUSH    ES                      ;保存寄存器

                PUSH    CS
                POP     DS                      ;DS=CS
                PUSH    CS
                POP     ES                      ;ES=CS

                MOV     BYTE PTR DS:[100H],AH   ;保存字型字节
                MOV     BYTE PTR DS:[101H],BL   ;保存修饰字节

                MOV     AX,8                    ;设点阵数据列数为8,即按ASCII
                                                ;字符设置
                OR      DH,DH                   ;DH=0吗?(当DH=0时, 表示读
ASCII
                                                ;字符点阵数据
                JNZ     L_0189                  ;DX为汉字转

;处理 ASCII 字符, DL=ASCII 字符
                MOV     WORD PTR DS:[102H],AX   ;保存点阵数据的列数
                MOV     CX,AX                   ;CX=点阵数据列数

;读取字符的点阵数据(16x8 点阵)
                MOV      AL,10H
;ASCII 字符是 16x8 点阵的, 因此, 每个 ASCII 字符占用 16 个字节
                MUL     DL                      ;AX=指定字符在点阵数据区中的
                                                ;相对位移
                ADD     AX,OFFSET D_0260        ;加点阵数据区的首址=指定汉字
                                                ;点阵数据地址
                MOV     SI,AX                   ;SI=点阵数据地址
                XOR     DI,DI                   ;DI=0, 传送目标地址

                PUSH    CX                      ;保存点阵数据列数
L_012C:
;对于 24 针打印机, 打印图形时, 每次送一列数据, 也即 24 位. 这里 ASCII 字符是 16x8 点
;阵的, 因此, 在一列中仅有 16 位有效数据, 所以下面先将前 8 位数据填 0, 然后再取后 16
;位数据, 组成 24 位
                MOV     AL,0                    ;前8位数据(全0)
                STOSB
                LODSW                           ;取后16位数据
                STOSW                           ;一列共24位(3字节)
                LOOP    L_012C                  ;取CX次

                POP     CX                      ;恢复CX=点阵数据列数

                XOR     DI,DI                   ;DI=0
```

```
                TEST    BYTE PTR DS:[101H],20H  ;是读上标字符点阵数据吗?
                JZ      L_0160                  ;不是

;修饰字节的位 5=1, 表示要读上标字符, 下面将正常字符点阵数据转换为上标字符点阵
;数据, 即将原点阵数据压缩一半, 放置于 24 位数据的中间 8 位

L_013D:
                INC     DI                      ;前8位数据不变(全为0)
                MOV     AX,[DI]                 ;取16位数据
                XCHG    AL,AH                   ;交换AH和AL的值
;因为点阵数据是按字节顺序存放的, MOV AX,[DI]后, AH 是后 8 位的数据, AL 是前 8 位的
;数据, 这样 AX 中的位 15 至位 0 不是原点阵排列的顺序, 交换 AL 和 AH 后, 即可使 AX 的 16 位(
;从高位到低位)与 16 点一一对应
                XOR     DX,DX
                MOV     BL,8                    ;16/2=8, 即要压缩 8 次
L_0146:
;下面将 AX 中的 16 点压缩为 8 点, 处理顺序是从高位到低位, 连续 2 点为一个单位, 两点
;中只要有一点的值为 1, 结果便为 1
                SHL     AX,1                    ;将前一点移入CF
                JNC     L_0150                  ;前一点的值不是1, 转检查后一
                                                ;点

;前一点是 1, 不用再检查后一点了, 但仍要将后一点移出 AX
                PUSHF                           ;保存CF, 此时CF=1
                SHL     AX,1                    ;将后一点从AX中移出
                POPF                            ;恢复CF=1
                JMP     SHORT L_0152

;前一点不是 1, 不管后一点是否为 1, 均移至 CF. 即当两点均为 0 时, CF=0
L_0150:
                SHL     AX,1                    ;将后一点移入 CF
;此时 CF 已是压缩后的数据
L_0152:
                RCL     DL,1                    ;将CF移入DL
                DEC     BL                      ;计数减1
                JNZ     L_0146                  ;共进行8次

;至此, DL 中已包含了压缩后的 8 位数据, 结合 DH(=0), DX 是原 16 点数据经上标处理后的
;值
                MOV     AX,DX
                STOSW                           ;写入压缩后的数据(覆盖原数据)
                LOOP    L_013D                  ;共CX列

                JMP     SHORT L_01DB
                NOP

L_0160:
                TEST    BYTE PTR DS:[101H],40H  ;是读下标字符点阵数据吗?
                JZ      L_0187                  ;不是
;修饰字节的位 4=1, 表示要读下标字符, 下面将正常字符点阵数据转换为下标字符点阵
;数据, 即将原点阵数据压缩一半, 放置于 24 位数据的最后 8 位

L_0167:
                INC     DI                      ;前8位0不变(全为0)
                MOV     AX,[DI]                 ;取16位数据
```

```
                XCHG    AL,AH                   ;交换AH和AL的值(字节交换)
                XOR     DX,DX
                MOV     BL,8                    ;压缩次数=8
L_0170:
                SHR     AX,1                    ;将前一点移入CF
                JNC     L_017A                  ;前一点不为1，转检查后一点

;前一点是1，不用再检查后一点了，但仍要将后一点移出AX
                PUSHF                           ;保存CF，此时CF=1
                SHR     AX,1                    ;将后一点从AX中移出
                POPF                            ;恢复CF=1
                JMP     SHORT L_017C

;前一点不是1，不管后一点是否为1，均移至CF．即当两点均为0时，CF=0
L_017A:
                SHR     AX,1                    ;将后一点移入CF
L_017C:
                RCR     DL,1                    ;将CF移入DL
;上面这条指令RCR DL,1是错误的，很显然如果这样压缩，那么压缩后的8位数据仍在DL
;中，也即是24位的中间8位，结果是上标字符．正确的指令应是:
;RCR DH,1或RCR DX,1

                DEC     BL                      ;计数减1
                JNZ     L_0170                  ;共压缩8次
                MOV     AX,DX
                STOSW                           ;写入压缩后的数据
                LOOP    L_0167                  ;共CX列
L_0187:
                JMP     SHORT L_01DB

;处理汉字，DX=汉字
L_0189:
                MOV     AL,10H                  ;点阵数据的列数=16
                MOV     WORD PTR DS:[102H],AX   ;保存列数
                MOV     CX,AX                   ;CX=点阵数据列数
                INT     7FH                     ;调用INT 7FH读取汉字显示点阵
                                                ;数据(是按横向排列的)，返回点
                                                ;阵数据地址为DX:0
                MOV     DS,DX                   ;DS=DX=点阵数据段址
                XOR     DI,DI                   ;目标地址

                TEST    BYTE PTR CS:[101H],8    ;是左旋90度吗?
                JZ      L_01B3                  ;不是

;处理左旋90度，只要将原水平的点阵数据左右翻转一下即可．例如：符号"┘"左旋后
;的字形应为"┐"，若直接用显示点阵数据打印，其字形为"┘"，显然不对．将显示点阵
;数据左右翻转后，其字形为"└"，这时的打印结果为"┐"，这正是我们需要的
                XOR     SI,SI                   ;SI=点阵数据首址
L_01A0:
                MOV     AL,0                    ;前8位仍为0
                STOSB
                MOV     AL,[SI+1]               ;取后8位数据．注意两字节的位
                                                ;置也要颠倒
                CALL    L_0253                  ;颠倒AL的位顺序
                STOSB                           ;写入中间8位数据
```

```
                LODSW                           ;取前8位数据
;AH 中的数据是没有用的，但反正 SI 要加 1，所以直接用 LODSW

                CALL    L_0253                  ;颠倒AL的位顺序
                STOSB                           ;写入后8位数据
                LOOP    L_01A0                  ;循环16次

                JMP     SHORT L_01D9
L_01B3:
                TEST    BYTE PTR CS:[101H],10H  ;是右旋90度吗?
                JZ      L_01CE                  ;不是

;处理右旋 90 度，只要将原水平的点阵数据上下翻转一下即可，例如：符号"┘"右旋后
;的字形应为"└"，若直接用显示点阵数据打印，其字形为"┘"，显然不对．将显示点阵
;数据上下翻转后，其字形为"┐"，这时的打印结果为"└"，这正是我们需要的
                MOV     SI,0FH                  ;SI 指向当前点阵数据的最后一排
;当前汉字的点阵是 16x16 点阵的，因此，每个汉字占 20H 字节，故最后一排的地址应为
;1EH，这里的指令 MOV SI,0FH 显然是错误的，应改为 MOV SI,1EH

L_01BE:
                MOV     AL,0                    ;前8位仍为0
                STOSB
                MOV     AL,[SI]
                STOSB                           ;写入中间8位数据
                MOV     AL,[SI+1]
                STOSB                           ;写入后8位数据
                DEC     SI
                DEC     SI                      ;每一排16点即两字节,SI指向上
                                                ;一排点阵数据地址
                LOOP    L_01BE                  ;共进行16次

                JMP     SHORT L_01D9

;不进行左旋和右旋，还须将水平的点阵数据转换为纵向的点阵数据
L_01CE:
                XOR     SI,SI                   ;SI=点阵数据首址
                CALL    L_0233                  ;转置前8位(即24点的中间8位)
                MOV     SI,1
                CALL    L_0233                  ;转置后8位(即24点的后面8位)
L_01D9:
                PUSH    CS
                POP     DS                      ;恢复DS=CS

;至此，字符和汉字的点阵数据全部具备，下面统一进行放大处理
L_01DB:
                XOR     SI,SI
                XOR     DI,DI                   ;SI=DI=0=点阵数据首址

                MOV     CX,WORD PTR DS:[102H]   ;CX=当前点阵数据的列数
                TEST    BYTE PTR DS:[100H],20H  ;要放大至24点阵吗?
                JZ      L_0219                  ;不要

;将 16(16x8 或 16x16)点阵数据放大至 24 点阵数据(隔点放大)，即第 0 位放大，第 1 位原样
;复制，依次进行
L_01EA:
```

```
                PUSH    CX                      ;保存CX
                INC     SI                      ;SI=16位数据地址
                MOV     CX,5                    ;循环计数=5
                MOV     BH,[SI]
                MOV     BL,[SI+1]               ;BX=16位数据，已作字节交换
                CALL    L_0224                  ;先将前10位数据放大到15位,保
                                                ;存于AX
;此时 BX 的高位已是第 10 位数据，该位数据要进行放大，但 AX 中已有 15 位数据，所以只
;能先将第 10 位复制一位到 AX，得前 16 位数据，保存前 16 数据后，再计算后 8 位数据
                PUSH    BX                      ;保存BX值，因为高位数据以后还
                                                ;要用
                RCL     BX,1                    ;将BX的高位(也即原数据的第10
                                                ;位)移入CF
                RCL     AX,1                    ;取得放大后的第16位数据
                PUSH    AX                      ;保存AL
                MOV     AL,AH                   ;因为点阵数据是按字节顺序存放
                                                ;的，所以应先写AH
                STOSB                           ;写入24位的前8位数据
                POP     AX                      ;恢复AL
                STOSB                           ;写入24点的中间8位数据

;下面计算后 8 位数据
                POP     BX                      ;恢复BX值
                XOR     AX,AX
                RCL     BX,1                    ;原第10位数据再次移入CF
                RCL     AL,1                    ;放大第10位数据
                RCL     BX,1                    ;原第11位数据移入CF
                RCL     AL,1                    ;复制原第11位数据
                MOV     CX,2                    ;还有4位数据在BX中
                CALL    L_0224                  ;将后4位放大到6位保存于AX中
                STOSB                           ;写入24点的后8位数据
                INC     SI
                INC     SI                      ;SI=下一列地址
                POP     CX                      ;恢复计数
                LOOP    L_01EA                  ;进行CX次

;数据已全部加工完毕
L_0219:
                MOV     CX,WORD PTR DS:[102H]   ;CX=点阵数据的列数
                XOR     SI,SI                   ;DS:SI指向加工好的点阵数据的
                                                ;地址
                POP     ES
                POP     BP
                POP     AX
                POP     DI                      ;恢复寄存器

                IRET                            ;中断返回

;子程序：隔点放大，每次将前 1 点放大为 2 点，复制后一点
;输    入：CX=放大次数
;          BX=源点阵数据(从高位开始)
;输    出：AX=放大后的点阵数据
L_0224:
                RCL     BX,1                    ;将前一点值移入CF
                PUSHF                           ;保存前一点的值
```

```
                RCL     AX,1                    ;复制一次
                POPF                            ;恢复前一点的值
                RCL     AX,1                    ;最复制一份(即放大)
                RCL     BX,1
                RCL     AX,1                    ;复制后一点
                LOOP    L_0224                  ;进行CX次

                RETN
;子程序: 将 8 列横向点阵数据转换纵向点阵数据, 原数据为 16 位, 转换后为 24 位, 其中
;           前8位为0
;输    入: SI = 0, 转换横向点阵数据的前8列
;           SI = 1: 转换横向点阵数据的后8列
;           DI:     数据存放地址
;输    出: 无
L_0233:
                MOV     CL,8                    ;共8列
L_0235:
                PUSH    SI                      ;保存SI
                MOV     AL,0                    ;前8位数据为0
                STOSB
                CALL    L_0243                  ;转换前8位数据
                CALL    L_0243                  ;转换后8位数据
                POP     SI                      ;恢复SI
                LOOP    L_0235                  ;进行CX次

                RETN

;子程序: 将横向点阵数据转化为纵向点阵数据
;入口: SI = 源数据地址
;       DI = 目标数据地址
;       CL = 位序号
;出口: 无
L_0243:
                MOV     DL,8                    ;共 8 点
;从原点阵数据中每排取一点

L_0245:
                LODSB                           ;取横向点阵的8位数据, 仅需取
                                                ;其中的一位
                SHR     AL,CL                   ;将第CL位移入CF
                RCL     BL,1                    ;将CF移入BL
                INC     SI                      ;SI指向下一排
                DEC     DL                      ;计数减1
                JNZ     L_0245                  ;进行8次

                MOV     AL,BL                   ;AL = 转换后的点阵数据
                STOSB
                RETN

;子程序: 颠倒 AL 的位顺序
;输    入: AL = 要颠倒的数据
;输    出: AL = 颠倒后的数据
L_0253:
                MOV     AH,AL                   ;AH = AL
                MOV     DL,8                    ;共8位
```

```
L_0257:
                SHL     AH,1                    ;将AH的最高位移入CF
                RCR     AL,1                    ;将CF移入AL的最高位
                DEC     DL                      ;计数减1
                JNZ     L_0257                  ;进行8次
                RETN
;以下是 16x8 点阵 ASCII 字符的点阵数据，每个 ASCII 字符占 16 字节
L_0260          DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
                .....

L_0A60:
;初始化程序
                MOV     DX,OFFSET L_0104        ;INT 7AH的入口地址
                MOV     AX,257AH
                INT     21H                     ;设置INT 7AH

                MOV     DX,0A60H                ;程序占用长度
                ADD     DX,104H                 ;留点余量
                INT     27H                     ;驻留退出
CODE            ENDS
                END     L_0100
```

第二节　读24点阵字库　FILE24A.COM

一、使用说明

FILE24A　abcde←┘

其中：a表示常用字区大小，以50个汉字为单位。

b～e为字库标识符。b对应于设置宋体字型时使用的字库，c对应于设置仿宋体字型时使用的字库，d对应于设置黑体字型时使用的字库，e对应于设置楷体字型时使用的字库。其值为“SFHK”（必须为大写，对应的字库文件名分别为HZK24S、HZK24F、HZK24H和HZK24K）之一。

例如：FILE24A　1SFHS表示保留50个常用字的缓冲区，设置字型和实际打印的字型对照如下：

设置字型：	宋体	仿宋体	黑体	楷体
打印字型：	宋体	仿宋体	黑体	宋体

由于24点阵字库文件都很大，若将它们全部拷入硬盘，将占用很多的硬盘空间。如用户硬盘空间较紧张且仅需使用一种字体打印时，可以删除其它24点阵字库文件（HZK24T除外），以节省硬盘空间。

24点阵ASCII字符的点阵数据空间由本程序保留，其点阵数据在文件ZF24.COM中，运行ZF24.COM即可将点阵数据传送到本程序保留的空间中。

二、程序功能

本程序主要包括两个部分，即初始化代码和INT　7BH代码。

初始化代码主要完成以下工作：

1）建立HZK16T和命令行指定的四个字库文件的扇区链表。

2）初始化常用字区。

3）设置INT　7BH和INT　7DH（与INT　7BH相同）。

4）驻留退出。

程序运行后，便可用INT　7BH或INT　7DH读取字符或汉字的24点阵数据（可根据要求作缩放处理）。INT　7BH的运行过程如下：

1）若DX是字符（DH＝0），那么从字符点阵数据区读取点阵数据，然后，根据需要转换为上标或下标字符的点阵数据。

2）若DX是汉字，且在常用字区中，则直接返回该汉字点阵数据的地址。

3）若DX是汉字，且该汉字不在常用字区中，则直接从指定的字库文件中将该汉字的点阵数据读入内存，并将该汉字及其点阵数据保存在常用字区中，使该汉字成为常用字。

4）若DX是汉字，则按需要进行左旋或右旋处理。

3）按需要对字符或汉字作缩小和放大处理。

4）返回处理后点阵数据的地址。

三、变量名表

地址	长度	意义
0082h	1	命令行参数1，设置内部词组大小
0083h	1	命令行参数2，指定宋体字库文件名
0084h	1	命令行参数3，指定仿宋体字库文件名
0085h	1	命令行参数4，指定黑体字库文件名
0086h	1	命令行参数5，指定楷体字库文件名
00F0h	2	每簇扇区数
00F4h	2	磁盘数据区起始扇区逻辑扇区号
01E0h	2	字库文件名的文件控制块(FCB)
0200h	2	常用字区首址
0202h	2	常用字区尾址
0204h	2	常用字已使用区尾址
0206h	2	常用字区更新地址
0208h	2	当前汉字点阵数据在字库文件中的起始扇区
020Ah	2	当前字型字库扇区链表首址
020Ch	2	ASCII字符点阵数据区首址(24x12)
020Eh	2	字库文件HZK24T的链表首址
020Eh	2	字符点阵数据尾址，也即链表首址
0210h	2	每扇区字节数
0212h	2	每扇区字节数减72
0214h	1	字型字节
0215h	1	指定汉字的前一字节
0216h	1	字体字节
0218h	2	宋体字库文件链表首址
021Ah	2	仿宋体字库文件链表首址
021Ch	2	黑体字库文件名链表首址
021Eh	2	楷体字库文件名链表首址
0221h	2	每道扇区数
0223h	2	磁头数
0520h	800h	ASCII字符点阵数据区

四、程序清单

```
;FILE24A.ASM
CODE            SEGMENT
                ASSUME  CS:CODE, DS:CODE
                ORG     100H

L_0100:
;读入硬盘主引导扇区，目的是为了查找 DOS 分区的起始物理地址
                MOV     AX,201H                 ;AH=2: 调用读扇区功能,AL=1:
                                                ;读入1个扇区
                MOV     BX,800H                 ;读入数据存放地址
                MOV     CX,1
                MOV     DX,80H                  ;物理地址为:C盘0道0面1扇区
                INT     13H

;查找 DOS 分区
                MOV     SI,OFFSET DS:[1BEH]     ;分区表起始位移
L_0111:
```

```
                CMP     BYTE PTR [BX+SI+4],1
                JE      L_0128                  ;是12 bits FAT 的DOS分区
                CMP     BYTE PTR [BX+SI+4],4
                JE      L_0128                  ;是16 bits FAT 的DOS分区
                CMP     BYTE PTR [BX+SI+4],6
                JE      L_0128                  ;是保留的DOS分区
;当前分区项不是 DOS 分区项
                ADD     SI,10H                  ;SI=下一分区项位移
                JMP     SHORT L_0111            ;检查下一分区项

L_0128:
;读入分区引导扇区(BOOT), 目的是为了知道当前分区的基本信息
                MOV     DH,[BX+SI+1]            ;起始磁头号
                MOV     CX,[BX+SI+2]            ;起始扇区号和磁道号
                MOV     AX,201H                 ;读入该分区的第一个扇区, 即该
                                                ;分区的引导扇区
                INT     13H

;读入文件定位表(FAT)
                MOV     AL,[BX+16H]             ;一个FAT占用扇区数
                INC     AX                      ;加上BOOT扇区
;此时, CX,DX 还是当前分区的起始物理地址, 而 FAT 正好紧接在 BOOT 扇区的后面, 为避免
;进行复杂的物理地址计算, 这里将 BOOT 扇区和 FAT 都读入内存
                MOV     AH,2
                INT     13H                     ;读入BOOT扇区和FAT,存放地址仍
                                                ;为800H

;计算数据区起始逻辑扇区号
                MOV     AX,[BX+0BH]             ;每扇区字节数
                MOV     D_0210,AX
                SUB     AX,48H                  ;每个汉字的点阵数据占72字节,
                                                ;AX=除去一个汉字点阵数据后扇
                                                ;区的长度
                MOV     D_0212,AX
                MOV     AX,[BX+18H]             ;每道扇区数
                MOV     D_0221,AX
                MOV     AX,[BX+1AH]             ;磁头个数
                MOV     D_0223,AX
                MOV     AL,[BX+0DH]             ;每簇扇区数
                XOR     AH,AH
                MOV     WORD PTR DS:[0F0H],AX
                MOV     AX,[BX+16H]             ;一个FAT占用扇区数
                MUL     BYTE PTR [BX+10H]       ;剩以FAT数目=FAT所占总扇区数
                ADD     AX,[BX+0EH]             ;加保留扇区数
                ADD     AX,[BX+1CH]             ;加隐含扇区数=根目录起始逻辑
                                                ;扇区号
                MOV     WORD PTR DS:[0F4H],AX   ;根目录起始逻辑扇区号
                MOV     AX,20H                  ;每个目录项长度=32字节
                MUL     WORD PTR [BX+11H]       ;剩以根目录项数目=根目录长度
                DIV     D_0210                  ;除以每扇区字节数=根目录所占
                                                ;扇区数
                ADD     WORD PTR DS:[0F4H],AX   ;加根目录起始逻辑扇区号=数据
                                                ;区起始逻辑扇区号

;下面开始建立打印字库文件的扇区链表, 共要建立 5 个字库文件的扇区链表
```

```
;建立 HZK24T 的扇区链表
                MOV     BX,0A00H                ;FAT首址
                MOV     BP,D_020E               ;链表首址
                CALL    L_0523                  ;建立HZK24T的扇区链表
;建立指定的第一个字库文件的扇区链表(通常为 HZK24S)
                MOV     D_0218,BP               ;链表首址
                MOV     AL,DS:[83H]             ;取命令行参数2
                CALL    L_0520                  ;建立指定字库1的扇区链表
;建立指定的第二个字库文件的扇区链表(通常为 HZK24F)
                MOV     D_021A,BP               ;链表首址
                MOV     AL,DS:[84H]             ;取命令行参数3
                CALL    L_0520                  ;建立指定字库2的扇区链表
;建立指定的第三个字库文件的扇区链表(通常为 HZK24H)
                MOV     D_021C,BP               ;链表首址
                MOV     AL,DS:[85H]             ;取命令行参数4
                CALL    L_0520                  ;建立指定字库3的扇区链表
;建立指定的第四个字库文件的扇区链表(通常为 HZK24K)
                MOV     D_021E,BP               ;链表首址
                MOV     AL,DS:[86H]             ;取命令行参数5
                CALL    L_0520                  ;建立指定字库4的扇区链表

;初始化常用字区
                MOV     AX,BP                   ;由L_0520返回时，BP为空闲内存
                                                ;区首址
                MOV     D_0200,AX               ;常用字区首址
                MOV     D_0204,AX               ;常用字已使用区尾址
                MOV     D_0206,AX               ;常用字区更新地址
;此时常用字区中没有任何常用字

                MOV     AL,DS:[82H]             ;取命令行参数1
                CMP     AL,41H
                JB      L_01BE                  ;数字转(<'A')转
                SUB     AL,7                    ;'A'-'F'->4a-4f
L_01BE:
                AND     AL,0FH                  ;30-4f->0-f
                XOR     AH,AH                   ;AXx50=常用字数
                MOV     DX,0E74H                ;每50个常用字的缓冲区使用量
;每个汉字本身占 2 个字节，加 24x24 点阵的数据 72 字节，为 74 字节. 50x74=0E74H

                MUL     DX                      ;AX=常用字区长度
                ADD     AX,D_0200               ;加常用字区首址=常用字区尾址
                MOV     D_0202,AX
                INC     AX                      ;留点余量

                PUSH    AX                      ;保存程序尾址
                MOV     DX,OFFSET L_0225        ;中断7BH的入口地址
                MOV     AX,257BH
                INT     21H                     ;设置INT 7BH
                MOV     AX,257DH
                INT     21H                     ;设置INT 7DH,其地址与INT 7BH
                                                ;相同

                POP     DX                      ;恢复程序尾址
                INT     27H                     ;驻留退出
```

```
D_01E0          DB      3                       ;文件控制块(FCB), 起始时是
                                                ;HZK24T的FCB. 当单元1E6H的值被
                                                ;修改后, 便成为其它字库文件的
                DB      'HZK24'                 ;FCB
D_01E6          DB      'T'
;该单元在建立指定字库扇区链表时, 被修改为命令行中的相应内容
                DB      '      '
                DB      20 DUP (0)

D_0200          DW      0                       ;常用字区首址
D_0202          DW      0                       ;常用字区尾址
D_0204          DW      0                       ;常用字已使用区尾址
D_0206          DW      0                       ;常用字区更新地址

D_0208          DW      0                       ;当前汉字点阵数据所在逻辑扇区
                                                ;号
D_020A          DW      0                       ;当前字型字库扇区链表首址
D_020C          DW      520H                    ;字符点阵数据首址(24x12)
D_020E          DW      1720H                   ;字符点阵数据尾址,也即链表首址
D_0210          DW      0                       ;每扇区字节数
D_0212          DW      0                       ;每扇区字节数减去72
D_0214          DB      0                       ;字型字节
D_0215          DB      0                       ;指定汉字的前一字节
D_0216          DW      0                       ;字体字节
D_0218          DW      0                       ;指定字库1的链表首址
D_021A          DW      0                       ;指定字库2的链表首址
D_021C          DW      0                       ;指定字库3的链表首址
D_021E          DW      0                       ;指定字库4的链表首址
                DB      0
D_0221          DW      0                       ;每道扇区数
D_0223          DW      0                       ;磁头数

;INT 7BH 和 INT 7DH
;输入: AH, 字型字节
;       位 0=1, 若位 5=1, 隔点横扩
;       位 1=1, 若位 5=1, 隔点纵扩
;       位 23, 00 宋体,01 仿宋体,10 黑体,11 楷体
;       位 5=1, 若位 0 和位 1 都为 0, 隔点压缩, 否则作隔点扩大
;       BL, 修饰字节
;       位 1=1, 上划线
;       位 2=1, 下划线
;       位 3=1, 左旋(字符无效)
;       位 4=1, 右旋(字符无效)
;       位 5=1, 上标(汉字无效)
;       位 6=1, 下标(汉字无效)
;       BH, 上下行标志(位 0=1 上半行,位 1=1 下半行)
;       若 DH=0, DL=ASCII 字符, 否则 DX=汉字
;输出: DS:SI=点阵数据地址, CX=点阵数据列数

L_0225:
                PUSH    ES
                PUSH    DI                      ;保存寄存器

                PUSH    CS
```

```
                POP     DS                      ;DS=CS
                PUSH    CS
                POP     ES                      ;ES=CS

                MOV     D_0214,AH               ;保存字型字节
                MOV     D_0215,DH             ;保存汉字的前一字节
                MOV     D_0216,BX               ;保存修饰字节

                OR      DH,DH                   ;DH=0吗?(当DH=0时，表示读ASCII
                                                ;字符点阵数据
                JNZ     L_02A4                  ;DX为汉字转

;处理 ASCII 字符，DL=ASCII 字符
                AND     DL,7FH                  ;屏蔽高位
;字符点阵数据中不包括大于 80H 且小于 0A1H 的扩展字符的点阵数据

                MOV     AL,24H                  ;每个24x12点阵的ASCII字符的点
                                                ;阵数据要占用36个字节
                MUL     DL
                ADD     AX,D_020C               ;AX=指定字符点阵数据首址
                MOV     SI,AX
                MOV     CX,0CH                  ;点阵数据列数=12

                TEST    BYTE PTR D_0216,20H     ;是读上标字符吗?
                JZ      L_0284                  ;不是

;修饰字节的位 5=1，表示要读上标字符，下面将正常字符的点阵数据转换为上标字符点
;阵数据，即将原点阵数据压缩一半，放置 24 位数据的前 12 位中
                XOR     DI,DI                   ;目标地址=0
L_0254:
                LODSW                           ;取16位数据
                CALL    L_026A                  ;压缩成8位，作为24位数据的前8
                                                ;位数据
                LODSB                           ;取后8位数据
                MOV     AH,0                    ;AH=0
                CALL    L_026A                  ;将AX压缩成8位数据，由于AH=0,
                                                ;故仅低4位有效
                MOV     AL,0
                STOSB                           ;24数据的后8位为0,这样24位数
                                                ;据有效位数为12位
                LOOP    L_0254                  ;压缩12次

                MOV     CL,0CH                  ;CX=点阵数据列数
                XOR     SI,SI                   ;SI=加工后点阵数据首址
                JMP     L_0361

;子程序：将 AX 中的 16 位数据压缩为 8 位数据
;输    入：AX=原数据
;          DI=压缩数据存放地址
;输    出：DI=DI+1
L_026A:
;因为点阵数据是按字节存放的，AH 是后 8 位数据，AL 是前 8 位数据，这样 AX 中的位 15 至位
;0 不是实际点阵数据的排列顺序. 交换 AL 和 AH 后，即可使 AX 的 16 位(从高位到低位)的排
;列顺序与实际排列顺序相同
                XCHG    AL,AH                   ;字节交换
```

```
L_026C:
                MOV     DL,8                    ;压缩次数=8

;下面将 AX 中的 16 点压缩为 8 点，处理顺序是从高位到低位，连续 2 点为一个单位，两点中
;只要有一点的值为 1，结果便为 1
L_026E:
                SHL     AX,1                    ;将前一点移入CF
                JNC     L_0278                  ;前一点的值不是1，转检查后一
                                                ;点
;前一点是 1，不用再检查后一点了，但仍要将后一点移出 AX
                RCL     BL,1                    ;将CF(前一点的值)移入BL
                SHL     AX,1                    ;将后一点从AX中移出
                JMP     SHORT L_027C
L_0278:
;前一点不是 1,不管后一点是否为 1，均移至 BL，所以当两点均为 0 时，结果为 0，否则，
;结果为 1
                SHL     AX,1                    ;将后一点移入CF
                RCL     BL,1                    ;将CF移入BL

L_027C:
                DEC     DL                      ;计数减1
                JNZ     L_026E                  ;压缩8次

                MOV     AL,BL                   ;AL=压缩数据
                STOSB                           ;保存压缩数据
                RETN

L_0284:
                TEST    BYTE PTR D_0216,40H     ;是读下标字符吗?
                JZ      L_02A0                  ;不是

;修饰字节的位 4=1，表示要读下标字符，下面将正常字符点阵数据转换为下标字符点阵
;数据，即将原点阵数据压缩一半，放置于 24 位数据的后 12 位
                XOR     DI,DI                   ;DI=0,数据写入地址
L_028D:
                MOV     AL,0                    ;24位的前8位=0
                STOSB
                LODSB                           ;取前8位数据
                MOV     AH,0                    ;AH=0
                CALL    L_026C                  ;将前8位数据压缩为4位，即生成
                                                ;24位的中间8位，该8位数据的高
                                                ;4位为0
                LODSW                           ;取后16位数据
                CALL    L_026A                  ;将AX压缩为8位数据，作为24位
                                                ;的最后8位
                LOOP    L_028D                  ;进行12次

                XOR     SI,SI                   ;SI=加工后点阵数据的首址
                MOV     CL,0CH                  ;点阵数据列数
L_02A0:
                JMP     L_0361
                NOP

;处理汉字，DX=汉字
```

```
L_02A4:
;对于同一汉字，若其字型不同，则其点阵数据也不同．在常用字区保存的常用字实际上
;是某种字型的汉字．因此，在验证 DX 是否为常用字时，必须同时检查字型是否匹配．字
;型选择数据在 AH 的第 2 位和第 3 位中．另一方面，为区别一般的 ASCII 字符，汉字机内码
;的第 7 位总是 1，实际这些位在这里是没有用的．因此，这里将两位字型数据分别保存于
;DH 和 DL 的第 7 位中．

;下面将 AH 的第 3 位保存于 DH 的第 7 位(已为 1)
                TEST    AH,8                    ;AH的第3位数据为1吗?
                JNZ     L_02AC                  ;是，和DH中的数据相同
                AND     DH,7FH                  ;将DH的第7位置0

;下面将 AH 的第 2 位保存于 DL 的第 7 位(已为 1)
L_02AC:
                TEST    AH,4                    ;AH的第2位数据为1吗?
                JNZ     L_02B4                  ;是，和DL中的数据相同
                AND     DL,7FH                  ;将DL的第7位置0

L_02B4:
                MOV     SI,D_0200               ;SI=常用字区首址
                SUB     SI,48H
;由于在循环中首先要执行 ADD  SI,48H，为了使第一次执行时 SI 指向常用字区首址，所以
;这里先执行 SUB  SI,48H

L_02BB:
                ADD     SI,48H                  ;SI=下一常用字地址
                CMP     SI,D_0204               ;已到常用字已使用区尾址了吗?
                JNE     L_02E1                  ;还没有

;常用字区中没有指定的字型汉字，下面直接从磁盘上读取指定字型汉字的点阵数据，并
;将其保存于常用字区中．
                CALL    L_0467                  ;调用子程序，从磁盘上读取汉字
                                                ;点阵数据
                CMP     DI,D_0204               ;是增加常用字吗?
                JBE     L_02D1                  ;不是，是覆盖常用字
                MOV     D_0204,DI               ;修改已使用区尾址
L_02D1:
                CMP     DI,D_0202               ;已没有剩余的常用字区了吗?
                JNE     L_02DB                  ;还有剩余空间

;已没有剩余的常用字空间了，将常用字区更新指针设置到缓冲区首
                MOV     DI,D_0200               ;将指针指向缓冲区首
L_02DB:
                MOV     D_0206,DI               ;修改常用字区更新指针
                JMP     SHORT L_02E6

L_02E1:
                LODSW                           ;取一个常用字
                CMP     AX,DX                   ;是当前指定汉字吗?(包括字型)
                JNE     L_02BB                  ;不是

L_02E6:
                MOV     CX,18H                  ;点阵数据的列数为24
                CMP     D_0215,0A9H             ;是制表符区汉字吗?
                JNE     L_02F3                  ;不是
```

```
                JMP     SHORT L_0361            ;制表符区汉字不用旋转
                NOP
L_02F3:
                TEST    BYTE PTR D_0216,8       ;是左旋90度吗?
                JZ      L_032B                  ;不是
;处理左旋 90 度. 由于每列有 24 点(3 字节), 因此分 3 次处理. 每次处理每列中的 8 点(即 8
;排数据), 处理时从当前点阵数据的最右列开始, 依次取 8 点作为新点阵数据.
;例如: 符号" ┌", 左旋后为" └".
                XOR     DI,DI                   ;DI=点阵数据存放地址
                ADD     SI,45H                  ;SI=原点阵数据最后列首址
                MOV     CL,3                    ;共进行3次

L_0301:
                PUSH    CX                      ;保存当前字节号
                MOV     CL,8                    ;每次3字节, 共8次

L_0304:
                PUSH    SI                      ;保存原点阵数据当前列地址
                MOV     DH,3                    ;一排共24点(3字节), 每次计算
                                                ;一排
L_0307:
                MOV     DL,8                    ;每一字节8点
L_0309:
                MOV     AL,[SI]                 ;取原点阵数据
                SHR     AL,CL                   ;将AL中第CL位移入CF
                RCL     BL,1                    ;将CF移入BL
                SUB     SI,3                    ;SI指向左一列地址
                DEC     DL                      ;计数减1
                JNZ     L_0309                  ;取纵向的8点->BL, 生成一字节
                MOV     AL,BL
                STOSB                           ;写入一字节
                DEC     DH                      ;计数减1
                JNZ     L_0307                  ;取一排的数据

                POP     SI                      ;恢复SI, 指向点阵数据最后一列
                                                ;的地址
                LOOP    L_0304                  ;每次3字节, 共8次

                INC     SI                      ;SI指向最后列下一字节
                POP     CX                      ;恢复CX=字节计数
                LOOP    L_0301                  ;每列共3字节

                XOR     SI,SI                   ;SI=0, 指向新点阵数据首址
                MOV     CL,18H                  ;CX=点阵数据列数
                JMP     SHORT L_0373
                NOP

L_032B:
                TEST    BYTE PTR D_0216,10H     ;是右旋90度吗?
                JZ      L_0361                  ;不是

;处理右旋 90 度. 由于每列有 24 点(3 字节), 因此分 3 次处理. 从第一列开始, 每列从最后
;一点开始取数据, 每次处理每列中的 8 点(即 8 排数据).
;例如: 符号" ┌", 右旋后为"┐". 处理方法是, 从第 1 列开始, 每列从下往上取点, 取
;出后成为新点阵数据的一排数据.
```

```
                XOR     DI,DI                   ;目标地址
                INC     SI
                INC     SI                      ;SI=当前列最后一字节地址
                MOV     CL,3                    ;将原点阵分为3大列，每大列8小
                                                ;列
L_0338:
                PUSH    CX                      ;保存大列号
                MOV     CL,8                    ;每大列取8次，每次3字节
L_033B:
                PUSH    SI
                MOV     DH,3                    ;每次3字节
L_033E:
                MOV     DL,8                    ;每字节8个点
L_0340:
                MOV     AL,[SI]                 ;取原点阵数据
                SHL     AL,CL                   ;将AL中的第CL位移入CF
                RCL     BL,1                    ;将CF移入BL
                ADD     SI,3                    ;SI=下一排地址
                DEC     DL                      ;计数减1
                JNZ     L_0340                  ;取1字节

                MOV     AL,BL
                STOSB                           ;写入1字节
                DEC     DH
                JNZ     L_033E                  ;取3字节
                POP     SI
                LOOP    L_033B                  ;取1大列

                DEC     SI                      ;SI=上一大列地址
                POP     CX
                LOOP    L_0338                  ;共3大列

                XOR     SI,SI                   ;SI=0，指向新点阵数据首址
                MOV     CL,18H                  ;CL=24，点阵数据列数
                JMP     SHORT L_0373

L_0361:
;此时若 SI<>0，表示不进行上标下标(字符)和旋转(汉字)处理，原点阵数据尚未读入
                OR      SI,SI                   ;SI=0?
                JZ      L_0373                  ;是，表示点阵数据已读入

;不进行上标下标和旋转处理，直接读原点阵
                PUSH    CX                      ;保存点阵数据列数
                MOV     AX,CX
                SHL     AX,1
                ADD     CX,AX                   ;CX=CX x 3,将列数化为字节数，
                                                ;每列24点(3字节)
                XOR     DI,DI
                REP     MOVSB                   ;取点阵数据
                XOR     SI,SI                   ;SI=0，指向新点阵数据首址

                POP     CX                      ;恢复点阵数据列数
L_0373:
;点阵数据已全部读入，下面进行其它(如:放大、缩小)处理
                TEST    BYTE PTR D_0216,2       ;要加上划线吗?
```

```
                JZ      L_0388                  ;不要

;加上划线
                PUSH    CX                      ;保存点阵数据列数
                XOR     BX,BX                   ;BX=第一排数据首址
L_037D:
                OR      BYTE PTR [BX],80H       ;给当前排数据加上划线
                ADD     BX,3                    ;BX=下一排首址
                LOOP    L_037D                  ;共CX列

                POP     CX                      ;恢复列数
                NOP
                NOP

L_0388:
                TEST    BYTE PTR D_0216,4       ;要加下划线吗?
                JZ      L_039C                  ;不要

;加下划线
                PUSH    CX                      ;保存点阵数据列数
                MOV     BX,2                    ;BX=第一排尾址(第三字节)
L_0393:
                OR      BYTE PTR [BX],1         ;给当前排数据加下划线
                ADD     BX,3                    ;BX=下一排尾址
                LOOP    L_0393                  ;共CX列

                POP     CX                      ;恢复列数
L_039C:
                TEST    D_0214,20H              ;要进行扩大或缩小吗?
                JZ      L_03C9                  ;不要

                TEST    D_0214,3                ;是进行横扩或纵扩吗?
                JNZ     L_03CC                  ;是
;当字型字节的位5=1时，表示要进行扩大或缩小．此时，横扩位(0)和纵扩位(1)均为0时，
;表示进行缩小．

;下面进行隔点压缩，即将原字符水平12点压缩为8点或将原汉字水平24点压缩为16点，
;压缩时以3列为单位，前一列照样复制，将后两列压缩成1列
                XOR     BX,BX                   ;列数计数
                XOR     DI,DI                   ;目标地址
L_03AE:
                LODSB
                STOSB
                LODSW
                STOSW                           ;复制一列
                INC     BX                      ;列数加1
                LODSB                           ;取当列前1字节
                OR      AL,[SI+2]               ;和下一列的前1字节相或，即将
                                                ;原两列的数据压缩为1列
                STOSB                           ;写入1字节
                LODSW                           ;取当前列的后2字节
                OR      AX,[SI+1]               ;和下一列的后2字节相或
                STOSW                           ;写入后2字节
                INC     BX                      ;列数加1
                ADD     SI,3                    ;在取数据后，SI已指向下一列地
```

```
                                        ;址，由于压缩，所以要越过一列
                DEC     CX
                DEC     CX
                LOOP    L_03AE          ;结合上面两条指令，相当于CX=
                                        ;CX-3

                MOV     CX,BX           ;修改点阵数据列数
                XOR     SI,SI           ;SI=0，指向新点阵数据首址
L_03C9:
                JMP     L_0460          ;处理结束

L_03CC:
                TEST    D_0214,2        ;要进行隔点纵扩吗?
                JZ      L_0433          ;不要

                AND     D_0214,0FDH     ;清除纵扩位
                XOR     DI,DI           ;DI=目标数据存放首址

;下面进行隔点纵扩，原来点阵数据的都为 24 排(包括字符和汉字)，扩大后变为 36 排，由
;于打印机一次只能打印 24 排，所以打印时要分两次，即分为上半行和下半行，上半行实
;际为 12 排，下半行实际为 24 排
                TEST    BYTE PTR D_0216+1,2     ;是取上半行点阵数据吗?
                JNZ     L_0426          ;不是

;取上半行点阵数据
                PUSH    CX              ;保存点阵数据列数
L_03E2:
                LODSB                   ;取8位
                MOV     AH,0            ;AH=0
                CALL    L_03F1          ;将AX中的16位数据扩大为24位

                                        ;前12位数据全为0
                INC     SI
                INC     SI              ;SI=下一排地址
                LOOP    L_03E2          ;取CX排

                POP     CX              ;恢复点阵数据列数
                XOR     SI,SI           ;SI=0，指向点阵数据首址
                JMP     SHORT L_0433

;子程序：将 16 位数据隔点扩为 24 位
;输   入：AX=原数据
;         DI=扩大后的数据存放地址
;输   出：无
L_03F1:
                PUSH    CX              ;保存点阵数据列数

;由于原数据为 16 位，扩大后数据为 24 位，扩大时，以两排为单位，前一排扩大为 2 排，
;后一排复制，这样，当进行第 6 次处理后，目标数据变为 18 位，超过了一字 16 位的限制，
;所以当目标数据达到 16 位时，必须先保存目标数据，再进行处理
                MOV     BX,AX           ;BX=原数据
                MOV     CL,5            ;CL=5次，即取15位数据
                CALL    L_0417          ;先将前10位数据扩为15位

;对于原数据的第 10 位要扩大为 2 倍，而目标数据中只能存放 1 位数据了.
```

```
                PUSH    BX              ;保存BX
                RCL     BX,1            ;将第10位数据移入CF
                RCL     AX,1            ;生成第15位数据

;写已生成的 16 位数据
                PUSH    AX              ;保存AX
                MOV     AL,AH
                STOSB                   ;写前8位
                POP     AX              ;恢复AX
                STOSB                   ;写第二个8位
                POP     BX              ;恢复BX
                XOR     AX,AX           ;目标数据置为0
                RCL     BX,1            ;将第10位数据再次移入CF
                RCL     AL,1            ;生成第16位数据
                RCL     BX,1
                RCL     AL,1            ;复制第11位数据

                MOV     CL,2            ;已生成18位数据，还有6位数据
                CALL    L_0417          ;再将后4位扩为6位
                STOSB                   ;写入第三个8位

                POP     CX              ;恢复点阵数据列数
                RETN

;子程序: 隔点扩大 BX 中的数据(从高位到低位)CX 次，每次将 2 位数据扩大至 3 位
;输   入: BX=原数据
;         CX=次数
;输   出: AX=扩大后的数据
L_0417:
                RCL     BX,1            ;将BX最高位移入CF
                PUSHF                   ;保存CF
                RCL     AX,1            ;复制一份
                POPF                    ;恢复CF
                RCL     AX,1            ;再复制一份
                RCL     BX,1
                RCL     AX,1            ;直接复制一份
                LOOP    L_0417          ;扩大CX次

                RETN

L_0426:
;取下半行点阵数据
                PUSH    CX              ;保存点阵数据列数
L_0427:
                INC     SI              ;第1字节扩大后全部在前半行中，
                                        ;需再计算
                LODSW                   ;取后二字节
                XCHG    AL,AH           ;进行字节交换
                CALL    L_03F1          ;将AX的16位扩大为24位
                LOOP    L_0427

                POP     CX              ;恢复点阵数据列数
                XOR     SI,SI           ;SI=0，指向点阵数据首址

L_0433:
```

```
                TEST    D_0214,1                ;要进行隔点横扩吗?
                JZ      L_0460                  ;不要
                AND     D_0214,0FEH             ;清除横扩位

                XOR     BX,BX                   ;列数计数
                MOV     DI,80H                  ;存放新点阵数据首址
L_0444:
                LODSB
                STOSB
                LODSW
                STOSW                           ;复制一排
                INC     BX                      ;列数加1
                LODSB
                STOSB
                MOV     [DI+2],AL               ;放大一排的前一字节
                LODSW
                STOSW
                MOV     [DI+1],AX               ;放大一排的后二字节
                INC     BX
                INC     BX                      ;列数加2
                ADD     DI,3                    ;DI=下一列地址
                DEC     CX
                LOOP    L_0444

                MOV     SI,80H                  ;新点阵数据首址=80H
                MOV     CX,BX                   ;修改点阵数据列数

L_0460:
                MOV     AH,D_0214               ;AH=字型数据(已被修改)

                POP     DI
                POP     ES                      ;恢复寄存器

                IRET                            ;中断返回

;子程序: 从磁盘上读取汉字的 24x24 点阵数据
;输   入: DX=汉字(DH 和 DL 的高位已为字型数据)
;输   出: DI=点阵数据尾址
;         SI=点阵数据首址
L_0467:
                MOV     DI,D_0206               ;DI=常用字区更新指针
                MOV     AX,DX                   ;AX=汉字(已带字型数据)
                STOSW                           ;作为常用字存入常用字区
                PUSH    DI                      ;保存首址
                PUSH    AX                      ;保存汉字
                AND     AH,7FH                  ;屏敝高位
                CMP     AH,30H                  ;是1-15区(符号汉字)吗?
                POP     AX                      ;恢复汉字
                JNC     L_047E                  ;不是符号汉字
                MOV     AX,D_020E               ;AX=HZK24T的链表起始地址
                JMP     SHORT L_0491

L_047E:
                SUB     DH,0FH                  ;减15,这样第一区就是汉字区了
                XOR     BX,BX                   ;BX=0
```

```
                SHL     AH,1                ;将AH高位(字型数据)移入CF
                RCL     BX,1                ;将CF移入BX
                SHL     AL,1                ;将AL高位(字型数据)移入CF
                RCL     BX,1                ;将CF移入BX
                SHL     BX,1                ;BX=BX x 2
                MOV     AX,D_0218[BX]       ;取对应字型汉字库链表首址
L_0491:
                MOV     D_020A,AX           ;保存当前汉字库链表首址
                AND     DX,7F7FH            ;屏敝汉字高位(原为字型数据)

;计算汉字顺序号
                SUB     DX,2121H
                MOV     AL,5EH              ;每区为94个汉字
                MUL     DH                  ;剩以区号
                XOR     DH,DH
                ADD     AX,DX               ;AX=汉字顺序号

;计算包含当前汉字点阵数据的扇区在当前汉字库中的相对扇区号
                MOV     DX,48H              ;每个汉字占72字节
                MUL     DX                  ;得在字库中的的偏移地址
                DIV     D_0210              ;除以每扇区字节数
                PUSH    DX                  ;保存该汉字在该扇区中的位移
                MOV     D_0208,AX           ;保存该汉字在字库中的相对扇区
                                            ;号
                CALL    L_04E5              ;读入该扇区
                POP     SI                  ;恢复在扇区中的位移
                MOV     CX,48H              ;长度=72
                CMP     SI,D_0212           ;该汉字的点阵数据在两个扇区中
                                            ;吗?
                JB      L_04C4              ;不是
                MOV     CX,D_0210           ;每扇区字节数
                SUB     CX,SI               ;该汉字在该扇区中的字节数
L_04C4:
                POP     DI                  ;恢复点阵数据首址
                PUSH    DI                  ;继续保存点阵数据首址
                PUSH    CX                  ;保存在该扇区中的字节数
                REP     MOVSB               ;取CX个字节
                POP     CX

                CMP     CX,48H              ;当前汉字点阵数据在同一扇区中
                                            ;吗?
                JE      L_04E3              ;是，不用再读下一扇区了

;当前汉字的点阵数据在两个扇区中，还须读入该字库文件的下一扇区.
                PUSH    CX                  ;保存已取得的字节数
                PUSH    DI                  ;保存当前写入地址
                MOV     AX,D_0208           ;当前扇区号
                INC     AX                  ;下一扇区号
                CALL    L_04E5              ;读入下一扇区
                POP     DI                  ;恢复写入地址
                POP     AX                  ;恢复已取得的字节数
                XOR     SI,SI               ;SI=0
                MOV     CX,48H
                SUB     CX,AX
                REP     MOVSB               ;取得剩余的点阵数据
```

```
L_04E3:
                POP     SI                      ;SI=点阵数据首址
                RETN
;子程序: 读入字库中指定扇区
;输   入: [208H]=相对扇区号
;输   出: 无

L_04E5:
                MOV     BX,D_020A               ;链表首址
L_04E9:
                CMP     AX,[BX+2]               ;在这一连续扇区群中吗?
                JB      L_04F6                  ;是
                SUB     AX,[BX+2]               ;不是
                ADD     BX,4                    ;指向下一链
                JMP     SHORT L_04E9

L_04F6:
                ADD     AX,[BX]                 ;得到逻辑扇区号
                XOR     DX,DX
                DIV     D_0221                  ;除以每道扇区数
                MOV     CX,DX
                INC     CL                      ;得绝对扇区号
                XOR     DX,DX
                DIV     D_0223                  ;除以磁头数目
                MOV     CH,AL                   ;得磁道号(低8位)
                MOV     DH,DL                   ;磁头号
                MOV     AL,40H
                MUL     AH
                ADD     CL,AL                   ;取磁道号的高2位
                MOV     DL,80H                  ;C盘
                XOR     BX,BX                   ;读入数据存放地址=0
                MOV     AX,201H                 ;读一个扇区
                INT     13H
                RETN

;字库文件名没有找到, 结束运行, INT 7AH 和 INT 7BH 没有设置
L_051C:
                POPF
                INT     20H                     ;结束运行
                NOP

;子程序: 建立文件扇区链表
;输   入: BX=FAT 首址
;         BP=链表首址
;         AL=SFHK, 分别表示建立宋体、仿宋体、黑体、楷体字库文件的扇区链表
;输   出: 无
L_0520:
                MOV     D_01E6,AL               ;按 AL 修改 FCB 中的字库文件名
L_0523:
;修改磁盘数据传送区地址(DTA), 因为缺省的 DTA 在 80H 处, 它刚好和命令行内容区重叠,
;如果不修改 DTA, 执行以下程序将会丢失命令行内容
                MOV     DX,5C0H                 ;新DTA地址
                MOV     DI,DX
                MOV     AH,1AH                  ;设置DTA功能
                INT     21H
```

```
                MOV     DX,OFFSET D_01E0        ;文件控制块(FCB)地址
                MOV     AH,11H
                INT     21H                     ;寻找匹配文件名
                OR      AL,AL                   ;找到?
                JNZ     L_051C                  ;没有，转非正常结束
                MOV     AX,[DI+1BH]             ;取文件起始簇号
                PUSH    AX
                SUB     AX,2                    ;减去2个保留簇号=实际簇号
;FAT 的前两项为系统保留簇号

                MOV     DX,WORD PTR DS:[0F0H]   ;每簇扇区数
                MUL     DX
                ADD     AX,WORD PTR DS:[0F4H]   ;AX=文件起始扇区的逻辑扇区号
                MOV     [BP],AX                 ;写入链表
                POP     AX                      ;恢复起始簇号
                MOV     WORD PTR DS:[0F6H],0    ;连续簇数=0,以后每当发现一连
                                                ;续簇时,该单元加1
L_0552:
                PUSH    AX
                INC     WORD PTR DS:[0F6H]      ;连续簇数加1
                MOV     SI,AX                   ;SI=当前簇号
                TEST    BYTE PTR [BX+4],40H
;对于 12 bits FAT, 每个簇号用 12 位表示. FAT 中, 前 3 字节为保留簇号占用. 第 4 字节
;的低 4 位和第 3 字节组成簇号 2, 第 5 字节和第 4 字节的高 4 位组成簇号 3. 一般情况下, 硬
;盘最前面的数据为两个系统文件占用, 而且必须连续存;放, 这样簇号 2 的数值应为 3,
;簇号 3 的数值应为 4, 即第 4 字节数值为 40H

                JNZ     L_0568                  ;12 bits FAT 转
;处理 16 bits FAT
                ADD     SI,AX
                MOV     AX,[BX+SI]              ;AX=下一簇号
                CMP     AX,0FFF8H               ;是文件最后一簇吗?
                JMP     SHORT L_057E

;处理 12 bits FAT
L_0568:
                SHR     AX,1
                PUSHF                           ;保存CF
                ADD     SI,AX                   ;SI = AX x 1.5
                MOV     AX,[BX+SI]              ;取簇号, 仅12位有效
                POPF                            ;恢复CF
                JNC     L_0578                  ;CF=0, 表示是偶数簇
;当簇号为奇数时, 表示 AX 的高 12 位是下一簇号值, 反之, 表示 AX 的低 12 位是下一簇号值
                MOV     CL,4                    ;奇数簇
                SHR     AX,CL                   ;右移4位, 去掉低4位, AX=下一
                                                ;簇号
                JMP     SHORT L_057B
L_0578:
                AND     AX,0FFFH                ;偶数簇, 屏蔽高4位, AX=下一簇
                                                ;号
L_057B:
                CMP     AX,0FF8H                ;是文件最后一簇吗?
L_057E:
                POP     SI                      ;恢复当前簇号
```

```
                JB      L_0588                  ;不是文件最后一簇

;处理文件的最后一簇
                MOV     AX,0FFFFH               ;最后簇标志
                MOV     [BP+2],AX               ;写入链表结束标记
                ADD     BP,4                    ;BP=内存可用空间首址
                RETN

L_0588:
                INC     SI                      ;当前簇号加1
;如果是连续，那么下一簇号应该=SI
                CMP     AX,SI                   ;是连续簇吗?
                JE      L_0552                  ;是

;非连续簇，增加链项
                PUSH    AX                      ;保存下一簇号
                MOV     AX,WORD PTR DS:[0F0H]   ;AX=每簇扇区数
                MUL     WORD PTR DS:[0F6H]      ;剩以每簇扇区数=连续扇区数
                MOV     [BP+2],AX               ;写入当前链的连续扇区数
                XOR     AX,AX
                MOV     WORD PTR DS:[0F6H],AX   ;连续簇数=0
                MOV     AX,WORD PTR DS:[0F0H]   ;每簇扇区数
                POP     SI                      ;SI=下一簇号
                PUSH    SI                      ;保存下一簇号
                SUB     SI,2                    ;SI=实际簇号
                MUL     SI
                ADD     AX,WORD PTR DS:[0F4H]   ;AX=逻辑扇区号
                ADD     BP,4
                MOV     [BP],AX                 ;保存下一链项的起始逻辑扇区号
                POP     AX                      ;AX=下一簇号
                JMP     SHORT L_0552
CODE            ENDS
                END     L_0100
```

第四章　键盘管理模块程序

第一节　使用说明

键盘管理模块主要是CCCC．COM程序，它包含对所有汉字输入方式的管理、各功能键的处理。

一、功能键简介

ALT_F1：　进入区位码输入方式。
ALT_F2：　进入首尾码输入方式。
ALT_F3：　进入拼音码输入方式。
ALT_F4：　进入快速码输入方式。
ALT_F6：　进入纯西文输入式。
ALT_F9：　内部词组管理。
ALT_F10：设置或取消联想词组输入状态。
CTRL_F1：进入预选字输入方式。
CTRL_F4：打印字符串。
CTRL_F5：清理内存。
CTRL_F6：设置字符显示颜色。
CTRL_F7：切换当前显示方式（西文或中文）。
CTRL_F8：建立或取消自动光标。
CTRL_F9：进入或退出纯中文输入方式。
CTRL_F10：设置打印字体和打印行距。

二、基本输入方式

同码字一页显示11个，首字（或词）用空格键选取，重选首字（或词）用ALT_' '。

区位码输入方式下，输入两位编码后可按空格进入翻页状态，此时既可上下翻页查找，也可直接选择输入。

拼音码长为三位，若不足三位的用‘[’补齐，在拼音码输入方式时，第四位编码是该汉字首尾码的第一码。

首尾码长为二位，在首尾码输入方式时，第三、四位编码是该汉字拼音码的前二码。

拼音码第二键是I或U时，提示行只显示以这两键为全拼音的汉字。

二、其它输入方式

内部词组可用ALT_F9定义，定义后可用外部命令CN．COM存盘，使内部词组变为外部词组。

要使用联想词组和外部词组均要事先在DOS提示符下运行相应的词组文件。

内部词组输入时编码以符号“'”结束，外部词组以符号“"”结束。

第二节　CCCC．COM程序解析

一、程序功能

本程序主要包括两个部分，即初始化代码和INT　16H代码。

初始化代码主要完成以下工作：

1．恢复经加密处理的系统初始化子程序，然后执行系统初始化子程序。

2．检查由HHDOS．COM设置的内部标志，若发现错误则重新启动或死机。

3．保存2．13H要使用的中断向量，设置其它中断（例如：INT　16H）。

4．初始化内部词组和外部词组。

5．驻留退出。

程序运行后，还不能使用汉字系统，只有运行显示模块程序后才能使用汉字系统（例如：CV26．COM）。注意，在运行显示模块程序前，也可以使用功能键，但由于还没有运行显示模块程序，因此有可能

造成死机。

二、变量名表

地址	长度	意义
00B0h	7D0h	当前屏幕字符ASCII值缓冲区,每一字符对应一字节
0880h	7D0h	当前屏幕字符属性值缓冲区，每一字节对应一字节
1050h	7D0h	当前屏幕字符类别值缓冲区，0:普通ASCII字符，1:汉字的前一字节，2:汉字的后一字节. 大于80H的表示是未匹配的汉字
1820h	2	0AA55H，内部标志，不能修改，否则运行中有可能死机
1822h	0F70h	INT 10H程序区，由显示模块程序传送过来
2793h	2	当前磁盘总簇数
2795h	2	213Eh，CCCC.COM系统正常初始化标志，不能修改，否则在运行中可能死机
2797h	1	=0CH,表示当前显示卡为CGA类型，=0Eh表示显示卡为EGA类型
2798h	400h	8x8点阵ASCII字符(0-127)点阵数据，每个字符占8字节
2B98h	2	词组检索时使用，表示当前使用词组段址
2B9Ah	2	词组检索时使用，表示当前检索地址(词组区)
2B9Ch	2	上页尾址(词组区)
2B9Eh	2	下页尾址(词组区)
2BA0h	2	当前使用词组编码区尾址
2BA2h	2	当前使用词组编码区首址
2BA4h	1	单字缓冲区格式标志，=0FFH:每字符仅占一字节. =1:每字符占二字节，且带有扫描码，=0: 表示仅有一个ASCII码
2BA5h	1	词组检索方向标志，=0:正向，=1:反向
2BA6h	2	词组检索前设置为3AH，以后没有使用
2BA9h	1	若输入字符是由INT 16H修改或生成的，该字节=0FFH，并且，第一次测试键盘时，返回没有字符可读
2BAAh	2	单字缓冲区指针
2BACh	1	纯中文状态字节，=0FFh: 表示在纯中文状态
2BADh	2	检索单字或词组时使用，表示当前检索地址(编码区)
2BAFh	2	检索单字或词组时使用，表示下页首址(编码区)
2BB1h	8	输入编码存放区
2BB9h	1	检索单字或词组时使用，位5=0表示未满一页，=1表示已满一页
2BBAh	22h	同码字保存区，最多可保存11个汉字
2BBCh	2	保存的第2个同码字的地址
2BCEh	2	保存的第11个同码字的地址
2BD0h	1	=0BH，每页最多同码字数
2BD2h	1	当前页同码字数
2BD6h	69C0h	单字的首尾码和拼音码编码表，每个汉字占4位，共6768个汉字
959Bh	1	是否正在输入编码，=0hFF:没有编码输入，此时可以直接输入非编码字符
959Ch	1	编码长度(选择输入后也不变)
959Dh	38h	单字缓冲区，最多可存放56个字符
95D5h	1	单字缓冲区中字符个数
95D7h	11h	CTRL_F6提示信息
95E8h	5	ALT_F2:"首尾:"
95EDh	5	ALT_F1:"区位:"
95F2h	5	ALT_F3:"拼音:"
95F7h	5	ALT_F4:"快速:"
95FCh	5	外部词组提示信息
9601h	0Eh	CTRL_F9:进入纯中文方式
960Fh	0Eh	CTRL_F9:退出纯中文方式
9620h	C	CTRL_F8:建立自动光标
962Dh	C	CTRL_F8:取消自动光标
963Ah	1	当前输入方式标志字节 =1:区位输入方式 =2:首尾输入方式

		=4:拼音输入方式
		=8:快速输入方式
		=10h:ASCII输入方式
		=20h:联想输入方式，可以和其它输入方式同时设置
		=40h:预选字输入方式
963Bh	6	ALT_F6:"ASCII:"
9646h	2	检索单字或词组时使用，表示上页尾址(编码区)
9648h	21h	当前页词组地址和长度索引表，每条词组占3字节，前一字为;地址，后一字节为词组长度．最多可存放11条词组的信息
9669h	2	当前输入词组地址
966Bh	1	词组缓冲区长度
966Ch	1	当前使用词组编号，0:外部词组，1:联想词组，2:内部词组
966Dh	1	内部标志
966Fh	1	内部词组管理标志，0FFh:正在进行内部词组管理
9670h	10h	程序自带外部词组区，其内容为"zg{ bj{ 中国北京"
9680h	2	外部词组段址
9682h	2	外部词组编码区尾址，也即词组区首址
9684h	2	外部词组词组区尾址
9686h	2	联想词组段址
9688h	2	联想词组编码区尾址，也即词组区首址
968Ah	2	联想词组词组区尾址
968Ch	2	内部词组段址，也即INT 7FH段址
968Eh	2	内部词组编码区尾址，也即词组区首址
9690h	2	内部词组词组区尾址
9692h	2	内部词组区更新地址
9694h	2	内部词组尾址
969Bh	5	内部词组提示信息
96A0h	2Eh	内部词组管理菜单
96D0h	20	增加内部词组菜单
96F0h	10	显示内部词组剩余空间字符串
9870h	0Fh	CTRL_F10h:打印字号
9880h	0Dh	CTRL_F10h:行距
988Eh	2	计算机内存总量
9890h	2	原INT 10h偏移地址
9892h	2	原INT 10h段地址
9894h	2	原INT 16h偏移地址
9896h	2	原INT 16h段地址
9898h	2	原INT 17h偏移地址
989Ah	2	原INT 17h段地址
98BEh	2	9907h, INT 16h功能0
98C0h	2	99BDh, INT 16h功能1
98C2h	2	98D7h, INT 16h功能2
98C4h	2	98DDh, INT 16h功能3
98C6h	2	98EDh, INT 16h功能4
98C8h	2	9993h, INT 16h功能5
98CAh	2	98CEh, INT 16h功能6
98CCh	2	99EBh, INT 16h功能7
9A20h	37h	CTRL_F5:清理内存
9A58h	2	1号INT 10h偏移地址
9A5Ah	2	1号INT 10h段地址
9A5Ch	2	2号INT 10h偏移地址
9A5Eh	2	2号INT 10h段地址
9A60h	5	ALT_F9:"联想:"
9A65h	5	ALT_F9:" :"
9A6Ah	2	预选字段地址

9A6Ch	2	与预选字类似的段地址，没有使用
A210h	1Eh	特殊字符纯中文表
A248h	1	出错标志,当其值累加进位时进入ROM BASIC
A4AEh	5	CTRL_F4:"打印:"
A95Eh	-3	存放剩余重码数字串，从个位数开始往上存放

二、程序清单

```
;CCCC.ASM
CODE            SEGMENT
                ASSUME CS:CODE,DS:CODE
                ORG    100H

L_0100:
;0200H-0400H 的程序是经过变换的，不能直接运行．下面将其恢复:
                MOV    SI,200H                  ;密文首址=0200h
                MOV    CX,200H                  ;密文长度为0200h字节
L_0106:
                LODSB                           ;取一字节
                NOT    AL                       ;复原(取反)
                MOV    [SI-1],AL                ;写回
                LOOP   L_0106                   ;恢复0200H-0400H的程序

L_010E:
                CALL   L_0200                   ;调用恢复后的初始化子程序

;设置外部词组，其中外部词组的编码区尾址和词组区尾址已在程序中直接设定，词组也
;只有两条，即"中国"和"北京".
                MOV    D_9680,CS                ;设置外部词组段址

;设置内部词组，内部词组区均由 FILE 系列文件设置(参见第二章)，单元 0200-0206 为内
;部词组变量地址．值对注意的是，在 FILE2.COM 程序中没有设置内部词组地址．该程序
;从偏移 0200H 处开始存放显示字库内容，因此，其内部词组地址是不正确的．但由于 16
;点阵汉字库的前几字节均为 0，故刚好使内部词组的首指针和尾指针相同，相当于没有
;保留内部词组空间.

                MOV    AX,357FH
                INT    21H                      ;取 INT 7FH 地址
;INT 7FH 是由 FILE 系列文件设置，因此，其段地址也为内部词组段址
                MOV    D_968C,ES                ;保存内部词组区段址
                MOV    AX,ES:[200H]             ;取内部词组区首址
                MOV    D_968E,AX                ;置内部词组编码区尾址
                MOV    D_9690,AX                ;置内部词组词组区尾址
                MOV    D_9692,AX                ;置添加内部词组地址
;由于系统启动时没有任何内部词组，所以编码区尾址、词组区尾址和添加地址均与内部
;词组首址相同

                MOV    AX,ES:[202H]             ;取内部词组尾址
                MOV    D_9694,AX                ;置内部词组尾址

                PUSH   CS
                POP    ES                       ;ES=CS

                CMP    ES:D_2795H,213EH         ;有系统正常标志吗?
```

```
                JZ      L_013F                  ;有
                INT     20H                     ;没有系统正常标志，直接退出L_013F:
                MOV     DX,0AA06H               ;程序驻留长度
                INT     27H                     ;驻留退出

                DB      188 DUP (0)

;以下程序均是由开始执行时复原得到的

;子程序: 系统初始化
;输   入: 无
;输   出: 无
L_0200:
;读硬盘主引导扇区，目的是为了取得 DOS 分区的起始物理地址
                MOV     AX,201H                 ;读一个扇区
                MOV     BX,1622H                ;读入内容存放地址
                MOV     CX,1
                MOV     DX,80H                  ;物理地址为: 0面0道1扇区
                INT     13H                     ;读入C盘主引导扇区

                MOV     D_2795,213EH            ;设置系统正常初始化标志

;查找DOS分区
                MOV     SI,01BEH                ;分区表起始位移
L_0217:
                CMP     BYTE PTR [BX+SI+4],1
                JE      L_022E                  ;是12 bits 的DOS分区
                CMP     BYTE PTR [BX+SI+4],4
                JE      L_022E                  ;是16 bits 的DOS分区
                CMP     BYTE PTR [BX+SI+4],6
                JE      L_022E                  ;是保留的DOS分区

;当前分区项不是 DOS 分区
                ADD     SI,10H                  ;SI=下一分区项位移
                JMP     L_0217                  ;检查下一分区项

L_022E:
;读分区引导扇区(BOOT)，目的是为了知道当前分区的基本信息
                MOV     DX,[BX+SI]
                MOV     CX,[BX+SI+2]            ;取分区起始物理地址
                MOV     BX,0B000H               ;读入内容存放地址
                MOV     AX,201H                 ;读一个扇区
                INT     13H

;读入根目录内容
                MOV     AX,[BX+16H]             ;一个FAT占用的扇区数
                MUL     BYTE PTR [BX+10H]       ;剩以FAT数目=FAT所占总扇区数
                ADD     AX,[BX+0EH]             ;加保留扇区数
                ADD     AX,[BX+1CH]             ;加隐含扇区数
                XOR     DX,DX
                DIV     WORD PTR [BX+18H]       ;除以每道扇区数
                MOV     CX,DX
                INC     CL                      ;CL=绝对扇区号
                XOR     DX,DX
                DIV     WORD PTR [BX+1AH]       ;除以磁头数
```

```
                MOV     CH,AL                   ;CH=磁道号的低8位数据
                MOV     DH,DL                   ;DH=磁头号
                MOV     AL,40H
                MUL     AH                      ;即左移6位
                ADD     CL,AL                   ;将磁道号的高2位->CL
                MOV     DL,80H                  ;C盘
                MOV     BX,0B000H               ;读入内容存放地址
                MOV     AX,220H                 ;读20H个扇区，相当于512个目录
                                                ;项
                INT     13H                     ;读入整个根目录内容

;检查由 HHDOS.COM 设置的内部标志
                MOV     SI,DS:[0B032H]          ;取出由HHDOS.COM保存的磁盘总
                                                ;簇数(参见HHDOS.ASM)
                MOV     DI,DS:[0B034H]          ;取出由HHDOS.COM保存的213子目
                                                ;录的起始簇号(取反后的值)
                NOT     DI                      ;DI=213子目录起始簇号
                MOV     DL,3                    ;逻辑盘号，3表示C盘
                MOV     AH,36H                  ;取磁盘参数功能号
                INT     21H                     ;DX返回该磁盘总簇数
                CMP     DX,SI                   ;与保存的相同吗?
                JE      L_0282                  ;是

L_027D:
;如果没有运行过 HHDOS.COM 则重新启动

;*              jmp     far ptr l_F000_FFF0     ;*
                DB      0EAH,0F0H,0FFH, 00H,0F0H ;重新启动

L_0282:
                MOV     D_2793,DX               ;保存磁盘总簇数

;查找 213 子目录的目录项
                MOV     BX,[0B000H]             ;目录项首址
L_0289:
                CMP     BYTE PTR [BX],0         ;0表示未使用的目录项，即表示
                                                ;所有目录项均已查过，没有发现
                                                ;213子目录
                JE      L_027D                  ;没有213子目录，转重新启动

                CMP     WORD PTR [BX],3132H     ;比较前两个字节
                JNE     L_029B                  ;不相同
                CMP     WORD PTR [BX+2],2033H   ;比较3,4两个字节
                JE      L_02A0                  ;是213子目录

;当前目录项不是 213 子目录,继续检查
L_029B:
                ADD     BX,20H                  ;BX=下一目录项地址
                JMP     SHORT L_0289            ;继续查找

L_02A0:
                CMP     DI,[BX+1AH]             ;子目录起始簇与由HHDOS.COM设
                                                ;置的相同吗?
                JNE     027EH                   ;不同，死机
;这条指令将直接转移到前面一条指令的中间部分，显然要死机
```

```
;保存和设置系统中断向量
                MOV     AX,3517
                INT     21H                     ;取原打印中断程序(INT 17H)地址
                MOV     WORD PTR DS:[9898H],BX
                MOV     WORD PTR DS:[989AH],ES ;保存INT 17h地址

                MOV     AX,3516H
                INT     21H                     ;取原键盘中断程序(INT 16H)地址
                MOV     WORD PTR DS:[9894H],BX
                MOV     WORD PTR DS:[9896H],ES ;保存INT 16h地址

                PUSH    ES
                POP     DS                      ;DS=ES=原INT 16H的段地址
                MOV     DX,BX                   ;DX=原INT 16H的偏移地址
                MOV     AX,257EH
                INT     21H                     ;设置INT 7FH, 使之指向原INT 16H

                MOV     AX,3510H
                INT     21H                     ;取原显示中断程序(INT 10H)地址
                MOV     WORD PTR CS:[9890H],BX
                MOV     WORD PTR CS:[9892H],ES ;保存INT 10h地址
                PUSH    ES
                POP     DS                      ;DS=ES=原INT 10H的段地址
                MOV     DX,BX                   ;BX=原INT 10H的偏移地址
                MOV     AX,2578H
                INT     21H                     ;设置INT 78H, 使之指向原INT 10H

                PUSH    CS
                POP     DS                      ;恢复DS

;检查当前显示卡是否为高分辨率彩显卡
                MOV     AX,ES                   ;原INT 10H的段地址
                CMP     AX,0F000H               ;是其它卡吗?
                JE      L_02F4                  ;是
                CMP     AX,0CC00H               ;是其它卡吗?
                JE      L_02F4                  ;是

                MOV     BYTE PTR CS:D_2797,0EH  ;置EGA标志
;2792H 原值为 0CH, 表示 CGA 方式. 这里用原 INT 10H 的段地址来判断是否为高分辨率彩显
;显然过于草率.

L_02F4:
                MOV     DX,OFFSET L_98A0        ;键盘中断程序入口地址
                MOV     AX,2516H
                INT     21H                     ;设置新的键盘中断程序

                MOV     DX,OFFSET L_2798        ;INT 1FH的偏移地址
                MOV     AX,251FH
                INT     21H                     ;设置新的INT 1FH
;该中断所指的并不是程序, 而是 ASCII 字符的点阵数据. 通常 INT 1FH 用于指出扩展 ASCII
;字符点阵数据的地址. 而这里指向 ASCII 码从 0 到 127 字符的点阵数据, 显然也是不合理的.
;而且, 从 CV26.COM 中, 我们还将看到它引用 INT 1FH 取得扩展 ASCII 字符的点阵数据.

                MOV     AX,CS                   ;AX=当前内存块段址
```

```
                DEC     AX
                MOV     ES,AX                   ;ES=当前内存控制块段址
                MOV     AX,ES:[1]               ;取当前内存块的段址
                ADD     AX,ES:[3]               ;加当前内存块长度(以节为单位)
;由于 DOS 在加载本程序时，已将所有剩余内存分配给本程序，所以此时 AX 值相当于系统
;内存总量.
                MOV     WORD PTR DS:[988EH],AX  ;保存系统内存总量
                RETN

                ORG     400H
;下面是系统启动时显示的"CCBIOS 2.13H..."信息的图形映象数据，长度 1040H 字节
                DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
                ...                             ;以下省略

;注: 从 00B0H-0880H 在系统启动后又被用于存放当前屏幕字符 ASCII 码. 每一光标位置对
;应 1 字节. 例如: 0B0H 处的值对应于坐标(0,0)，表示该位置字符的 ASCII 码.
;       从 0800H-1050H 在系统启动后又被用于存放当前屏幕字符属性.
;       从 1050H-1820H 在系统启动后又被用于存放当前屏幕字符类别. 0 表示为一般 ASCII
;字符，1 表示是汉字的前 1 个字节，2 表示是汉字的后一字节. 大于 80H 表示是该汉字的另
;一字节没有显示.

D_1820          DW      0AA55H
;标志字节，该字节不能修改，系统启动后仍将频繁地检查该数据，若有变动，将死机

;以下是显示中断程序区，它是由显示模块(例如:CV26.COM)传送过来的
D_1822          DB      0F70H DUP (0)

                DB      0                       ;没有用
D_2793          DW      0                       ;当前磁盘总簇数
D_2795          DW      0                       ;启动时设置为213Eh，是标志字节
                                                ;与1820H一样，该字节也不能修改

D_2797          DB      0CH                     ;若原显示卡为CGA方式=0CH
                                                ;否则=0EH，表示为EGA方式

;INT 1FH
;ASCII 字符(0-127)点阵数据，每个字符共 8x8 点，即 8 字节，其总长度为 400H 字节
D_2798          DB      0,0,0,0,0,0,0,0
                DB      0,0,0,1FH,10H,10H,10H   ;字符点阵数据
                ......                          ;以下省略

D_2B98          DW      0                       ;词组检索时使用，表示当前检索
                                                ;词组段址
D_2B9A          DW      0                       ;词组检索时使用，表示当前检索
                                                ;地址
D_2B9C          DW      0                       ;词组检索时使用，上页词组区首
                                                ;址
D_2B9E          DW      0                       ;词组检索时使用，下页词组区首
                                                ;址
D_2BA0          DW      0                       ;词组检索时使用，表示当前词组
                                                ;编码区尾址
D_2BA2          DW      0                       ;当前词组编码区首址
D_2BA4          DB      0                       ;单字缓冲区格式
;=FF: 表示缓冲区中仅有一个字符，并且该字符由二个字节组成，第一字节为 ASCII 码，
;第二字节为扫描码
```

```
;=00: 表示缓冲区中字符全部以 ASCII 码存放，每个字符仅占 1 字节
D_2BA5          DB      0                       ;词组检索时方向标志
;=0:表示进行正向检索，=1:表示进行反向检索

D_2BA6          DW      0                       ;在词组检索时使用，初始化时置
                                                ;为3AH，以后没有使用
D_2BA9          DB      0
;测试键盘时使用，=FF 时，不管缓冲区中有无字符，均返回无字符. 若输入的字符是由
;INT 16H 修改或生成的，那么该单元被设置成 FF.

D_2BAA          DW      0                       ;当前单字缓冲区指针
D_2BAC          DB      0                       ;纯中文状态字节
;=FF:表示已进入纯中文状态，=0:表示未进入纯中文状态

D_2BAD          DW      2BD6H                   ;单字和词组检索时使用，表示当
                                                ;前检索单字或词组编码地址
D_2BAF          DW      2BD6H                   ;检索单字或词组时使用，上页编
                                                ;码区尾址
D_2BB1          DB      8 DUP (0)               ;输入编码内容存放区
D_2BB9          DB      0                       ;检索单字或词组时使用，满页标
                                                ;志，位5=0:未满页，=1:已满页
D_2BBA          DB      22H DUP (0)             ;同码字存放区，最多可存放11个
                                                ;汉字
D_2BD0          DB      0BH                     ;最多同码字个数，等于11
                DB      0                       ;没有用
D_2BD2          DB      0                       ;当前页同码字数目

;以下是汉字编码表，每个汉字占 2 字，前一字为首尾码，后一字为拼音码
;格式如下:
;          首尾码字                拼音码字
;   {----------------}  {----------------}
;   XXXXXX XXXXX XXXXX  XXXXXX XXXXX XXXXX
;          -----  -----  |-----  -----  -----
;           码2    码1   |码3    码2    码1
;                        V
;                     高频标志位
;每一码由 5 位组成，每一码的实际编码均要加基数 64，例如:'a'的码值为 1，'['的码值
;为 27，另外，若该汉字为高频汉字，则第二字的高位=1

D_2BD6          DW      146H,6F61H              ;汉字'啊'的编码
;格式说明:
;          146H              6f61h
;000000 01010 00110      0 11011 11011 00001
;          ---  ---        ---  ---  ---
;          10(o)  6(f)      27([)  27([)   1(a)
;          首尾码为:'fo'     拼音码为:'a[[', 该汉字不是高频字
                ......                          ;以下省略

                DB      0                       ;没有用
D_9597          DW      0                       ;从当前页开始，到最后或最前的
                                                ;总同码字数，主要是为了显示剩
                                                ;余的同码字数
                DW      0                       ;没有用
D_959B          DB      0FFH                    ;是否已经有编码输入
;=FF: 没有编码输入，此时可以直接输入非编码字符
```

```
;=00: 已经有编码输入，不能直接输入非编码字符(扩展字符除外)
D_959C          DB      0                       ;当前已输入编码长度，在选择输
                                                ;入后仍保留
D_959D          DB      38H DUP (0)             ;单字缓冲区，当输入词组后，词
                                                ;组内容也将被传送到该缓冲区中
D_95D5          DB      0                       ;单字缓冲区长度
                DB      0                       ;没有用
D_95D7          DB      '键入颜色号(1-15):'     ;CTRL_F6提示信息
D_95E8          DB      '首尾:'                 ;ALT_F2提示信息
D_95ED          DB      '区位:'                 ;ALT_F1提示信息
D_95F2          DB      '拼音:'                 ;ALT_F3提示信息
D_95F7          DB      '快速:'                 ;ALT_F4提示信息
D_95FC          DB      '词组:'                 ;外部词组提示信息
D_9601          DB      '建立纯中文方式'        ;CTRL_F9提示信息
D_960F          DB      '取消纯中文方式'        ;CTRL_F9提示信息
                DB      0,0,0                   ;没有用
D_9620          DB      '建立自动光标'          ;CTRL_F8提示信息
D_962C          DB      0FFH                    ;当前光标状态
;=FF:表示有光标，  =00:表示没有光标

D_962D          DB      '取消自动光标'          ;CTRL_F8提示信息
D_963A          DB      0                       ;当前输入方式字节
;=01:区位码输入方式
;=02:首尾码输入方式
;=04:拼音码输入方式
;=08:快速码输入方式
;=10:ASCII 输入方式
;=20:联想输入方式
;=40:预选字输入方式

D_963B          DB      'ASCII:'                ;ALT_F6提示信息
                DB      0,0,0,0,0               ;没有用

D_9646          DB      2BD6H
;检索单字或词组时使用，表示上页编码区尾址．当按下往上翻页键时，以该单元的值作
;为首址反向检索．

D_9648          DB      21H DUP (0)             ;当前页词组的地址和长度索引表
;每条词组三字节组成，其中前一字为词组起始地址，后一字节为词组长度．共 33 字节，
;最多能保存 11 个词组的地址的长度．

D_9669          DW      0                       ;当前词组地址
D_966B          DB      0                       ;剩余词组长度
;当前选择词组的长度，若词组长度大于 56 字节，则要分几次才能将词组全部传送到单字
;缓冲区中，此时，该单元记录剩余词组长度．

D_966C          DB      0                       ;词组编号，在词组检索时使用
;=0:外部词组，=1:联想词组，=2:内部词组

D_966F          DB      0                       ;内部词组管理标志
;=FF:表示正在进行内部词组管理，=00:表示没有进行内部词组管理．在进入内部词组管
;理时可以输入功能键．

D_9670          DB      'zg{',4,'bj{',4,'中国','北京'
;程序固定外部词组，仅两条词组。系统启动时，外部词组指向这里，如果，运行了外部
```

```
;词组文件，则外部词组指向装载的外部词组.

D_9680          DW      0                       ;外部词组段地址
D_9682          DW      9670H                   ;外部词组编码区首址
D_9684          DW      9678H                   ;外部词组词组区首址
D_9686          DW      0                       ;联想词组段地址
D_9688          DW      0                       ;联想词组编码区首址
D_968A          DW      0                       ;联想词组词组区首址
D_968C          DW      0                       ;内部词组段地址
D_968E          DW      0                       ;内部词组编码区首址
D_9690          DW      0                       ;内部词组词组区首址
D_9692          DW      0                       ;内部词组词组区尾址
D_9694          DW      0                       ;内部词组区尾址
;由于内部词组是动态管理的，因此，需要更多的变量

                DB      0,0,0,0,0               ;没有用

D_969B          DB      '词组"                  ;内部词组提示信息
D_96A0          DB      '内部词组管理：A-添加，X-显示，Q-清除，←- 退出'
                DB      0,0                     ;没有用
D_96D0          DB      '编码:{{{     内容: 从光标起字符数
D_96F0          DB      '空间 00000 字节.'

;ALT_F9
;内部词组管理
L_9700:
                MOV     AX,1000H
                INT     10H                     ;清提示行，光标移至行首
L_9705:
                MOV     DL,0                    ;光标位置
                MOV     BX,96A0H                ;字符串地址
                MOV     CX,2EH                  ;长度
                CALL    L_A249                  ;在提示行显示内部词组管理菜单
                MOV     D_966F,0FFH             ;置正进行内部词组管理标志
L_9715:
                JMP     L_9DC5

L_9718:
                MOV     D_966F,0                ;清除内部词组管理标志

                CMP     AL,0DH                  ;是回车吗?
                JNE     L_9728                  ;不是

;内部词组管理，回车-->退出内部词组管理
                MOV     AX,1000H
                INT     10H                     ;清除提示行内容
                JMP     SHORT L_9715

L_9728:
                AND     AL,5FH                  ;将小写字符转换为大写字符

                CMP     AL,51H                  ;是'Q'吗?
                JNE     L_9739                  ;不是

;内部词组管理,'Q'-->清除内部词组
```

```
                MOV     AX,D_968E               ;取内部词组区首址
                MOV     D_9690,AX               ;置内部词组编码区尾址
                MOV     D_9692,AX               ;置内部词组词组区尾址
                JMP     SHORT L_9700

L_9739:
                CMP     AL,58H                  ;是'X'吗?
                JNE     L_9765                  ;不是

;内部词组管理,'X'-->显示剩余空间
                MOV     BX,0AH                  ;BX=10
                MOV     CX,5                    ;数字串长度=5
                MOV     DI,OFFSET D_96F9        ;数字串存放地址
                MOV     AX,D_9694               ;AX=内部词组区尾址
                SUB     AX,D_9692               ;减内部词组词组区尾址=内部词
                                                ;组区剩余空间
L_974D:
                XOR     DX,DX                   ;DX=0
                DIV     BX                      ;除以10
                OR      DL,30H                  ;加'0',转换为数字
                MOV     [DI],DL                 ;保存数字
                DEC     DI                      ;上一数字位置
;由个位数往上计算
                LOOP    L_974D

                MOV     BX,96F0H                ;剩余空间提示信息地址
                MOV     CL,10H                  ;显示长度=16
                MOV     DL,30H                  ;显示位置
                CALL    L_A249                  ;显示剩余空间大小
                JMP     SHORT L_9705

L_9765:
                CMP     AL,41H                  ;是'A'吗?
                JNE     L_9700                  ;不是

;内部词组管理,'A'-->添加内部词组
                MOV     AX,1000H
                INT     10H                     ;清除提示行
                MOV     AX,7B7BH                ;'{{'
                MOV     D_96D5,AX
                MOV     D_96D7,AL               ;将编码内容设置为'{{{'
                MOV     DL,0                    ;光标位置
                MOV     BX,96D0H                ;增加内部词组提示信息地址
                MOV     CX,20H                  ;字符串长度
                CALL    L_A249                  ;显示
                MOV     DL,5                    ;光标位置
                MOV     AX,1002H
                INT     10H                     ;在提示行设置光标位置

                MOV     DI,96D5H                ;存放输入的内部词组编码地址
                MOV     CL,3                    ;长度最多为3
L_978E:
                MOV     AH,0
                INT     7EH                     ;调用原键盘管理程序读一字符
                CMP     AL,8                    ;是删除键吗?
```

```
                JNE     L_97A3                  ;不是

;增加内部词组,删除键处理
                CMP     CL,3                    ;还没有输入过吗?
                JE      L_978E                  ;是,不能删除

                INC     CX
                INC     CX                      ;CX要增加两次
                MOV     BYTE PTR [DI-1],7BH     ;='}',删除编码
                JMP     SHORT L_97AC

L_97A3:
                CMP     AL,0DH                  ;是回车吗?
                JE      L_97B3                  ;是

;接收编码
                OR      AL,20H                  ;将大写转换为小写
                MOV     [DI],AL                 ;保存编码内容
                INC     DI                      ;编码地址加1
L_97AC:
                MOV     DL,AL                   ;取输入字符
                CALL    L_A6FB                  ;显示该字符, 当为删除字符时,
                                                ;删除上一字符
                LOOP    L_978E

;回车, 编码输入结束

L_97B3:
                MOV     DL,20H                  ;光标位置
                MOV     AX,1002H
                INT     10H                     ;设置提示行光标位置

                MOV     CL,0                    ;词组长度=0
L_97BC:
                MOV     AH,0
                INT     7EH                     ;调用原键盘中断程序读一字符

                CMP     AL,0DH                  ;是回车吗?
                JE      L_97E1                  ;是

                CMP     AL,30H                  ;>='0'吗?
                JB      L_97BC                  ;不是
                CMP     AL,39H                  ;<='9'吗
                JA      L_97BC                  ;不是

;输入的是数字
                MOV     DL,AL                   ;DL=输入数字
                PUSH    AX                      ;保存AX
                CALL    L_A6FB                  ;显示该数字
                POP     AX                      ;恢复AX
                AND     AL,0FH                  ;将数字转换为相应数值
                MOV     BL,AL                   ;保存于BL
                MOV     AL,0AH                  ;AL=10
                MUL     CL                      ;将原值剩以10
                ADD     AL,BL                   ;加上输入的数值
```

```
                MOV CL,AL                       ;CL=长度
                JMP     SHORT L_97BC

;已将词组长度输入，开始定义内部词组
L_97E1:
                OR      CX,CX                   ;长度=0吗?
                JNZ     L_97ED                  ;不是

;长度=0,无效的内部词组定义
L_97E5:
                MOV     DL,7                    ;响玲字符
                CALL    L_A6FB                  ;响玲
                JMP     L_9700

L_97ED:
                MOV     AX,D_9692               ;AX=内部词组词组区尾址，也就
                                                ;是内部词组剩余空间首址
                ADD     AX,4                    ;加当前词组编码长度
                ADD     AX,CX                   ;加当前词组长度
                CMP     AX,D_9694               ;已超过内部词组区尾址了吗?
                JA      L_97E5                  ;是，空间不够，定义无效

;将内部词组词组区内容往后移4字节，空出4字节用于存放新词组的编码
                PUSH    ES                      ;保存ES
                MOV     ES,D_968C               ;ES=内部词组段址

                PUSH    CX                      ;保存词组长度
                STD                             ;设置递减的字符串操作方式
                MOV     SI,D_9692               ;内部词组词组区尾址
                MOV     CX,SI
                SUB     CX,D_9690               ;减编码区尾址=词组区长度
                DEC     SI                      ;SI=内部词组词组区首址
                MOV     DI,SI
                ADD     DI,4                    ;新词组区首址
                PUSH    DS                      ;保存DS
                PUSH    ES
                POP     DS                      ;ES=DS=内部词组段址
                REP     MOVSB                   ;移动内部词组词组区内容
                POP     DS                      ;恢复DS

;保存新词组的编码
                CLD                             ;设置递增的字符串操作方式
                MOV     DI,D_9690               ;编码区结束地址
                MOV     AX,D_96D5               ;编码的前两位
                STOSW                           ;保存新词组编码的前二码
                MOV     AL,D_96D7               ;取编码的第三码
                STOSB                           ;保存新词组编码的第三码
                POP     CX                      ;恢复词组长度
                MOV     AL,CL
                STOSB                           ;保存新词组长度

;读取新词组的内容，并保存于内部词组词组区中
                MOV     DX,CX                   ;DX=词组长度
                MOV     AX,DS:[50H]             ;AX=当前光标在屏幕上的位置
                ADD     AH,DS:[0A5H]            ;加当前屏幕第一行在总25行中的
```

```
                                                ;行号. AX=当前光标的内部坐标
                MOV     BX,AX
                MOV     AL,AH                   ;AL=光标的行号
                MUL     BYTE PTR DS:[4AH]       ;剩以每行列数
                MOV     BH,0
                ADD     AX,BX                   ;AX=当前光标位置在内部缓冲区
                                                ;中的相对位移
                MOV     BX,AX
                MOV     DI,D_9692
                ADD     DI,4                    ;DI=新词组首址
                MOV     ES,D_968C               ;ES=内部词组段址

L_984B:
                MOV     AL,DS:[BX+0B0H]         ;取词组内容
                CMP     AL,20H                  ;<空格吗?
                JB      L_9854                  ;是,不能作为词组保存
                STOSB                           ;保存词组
L_9854:
                INC     BX                      ;BX=下一存放地址
                LOOP    L_984B                  ;词组长度为CX

                POP     ES                      ;恢复ES

;修改内部词组指针
                MOV     AX,D_9692               ;AX=内部词组词组区尾址
                ADD     AX,4                    ;加编码长度
                ADD     AX,DX                   ;加词组长度
                MOV     D_9692,AX               ;新的词组区尾址
                MOV     AX,D_9690               ;取编码区尾址
                ADD     AX,4                    ;加新词组编码长度
                MOV     D_9690,AX               ;新的编码区尾址，也即新的词组
                                                ;区首址
                JMP     L_9700

                DB      0                       ;没有用
D_9870          DB      '打印字号(A-x):         ;CTRL_F10提示信息
D_9880          DB      '行距(1-255):           ;CTRL_F10提示信息
                DB      0                       ;没有用
D_988E          DW      0                       ;系统内存总容量

D_9890          DW      0                       ;原INT 10H的偏移地址
D_9892          DW      0                       ;原INT 10H的段地址
D_9894          DW      0                       ;原INT 16H的偏移地址
D_9896          DW      0                       ;原INT 16H的段地址
D_9898          DW      0                       ;原INT 17H的偏移地址
D_989A          DW      0                       ;原INT 17H的段地址

;INT 16H
;1.键盘输入
;入口:AH=0
;出口:AL=输入字符
;2.检查键盘缓冲区
;入口:AH=1
;出口:ZF=0, 有字符, AL=输入字符
;       ZF=1, 无字符;3.取当前控制键状态
```

```
;入口:AH=2
;出口:AL 位 0-右 SHIF, 1-左 SHIFT, 2-CTRL, 3-ALT, 4-SCROLLLOCK, 5-NUMLOCK
;          6-CAPSLOCK, 7-INS
;4.装入预选字
;入口:AH=3
;      DX=段址, AL=0
;出口:无
;5.装载词组、联想词库
;入口:AH=4
;      AL=0 外部词组, =1 联想词库, =2 内部词组
;      BP=段址, DX=编码区起始地址, CX=词组区起始地址
;出口:无
;6.显示中断管理
;入口:AH=5
;      AL=0 保存新的 INT 10H 地址于单元 1, BP=段址, DX=偏移
;      AL=1 设置 INT 10H 中断向量为保存的地址
;      AL=2 设置 INT 10H 中断向量为保存的地址
;      AL=3 保存新的 INT 10H 地址于单元 2, BP=段址, DX=偏移
;出口:无
;7.模拟功能键
;入口:AH=6
;      AL=扩展 ASCII 码
;出口:无
;8.调用显示和键盘模块中的子程序
;入口:AH=7
;      BP=子程序偏移地址

L_98A0:
                STI                             ;开中断

                PUSH    ES
                PUSH    DS
                PUSH    BX
                PUSH    SI
                PUSH    DI
                PUSH    CX
                PUSH    BP
                PUSH    DX                      ;保存寄存器

                MOV     BX,40H
                MOV     ES,BX                   ;ES=ROM BIOS数据段址
                PUSH    CS
                POP     DS                      ;DS=CS

                CMP     AH,7                    ;功能号大于7吗?
                JA      L_98FE                  ;是,无效的功能号,直接结束中断
                MOV     BL,AH                   ;BL=功能号
                SHL     BX,1                    ;BX=BXx2
                JMP     WORD PTR D_98BE[BX]     ;转相应的功能入口地址

;INT 16H 功能入口地址表
D_98BE          DW      9907H                   ;0号功能,读键盘
                DW      99BDH                   ;1号功能,测试键盘缓冲区
                DW      98D7H                   ;2号功能,取控制键状态
                DW      98DDH                   ;3号功能,装入预选字
```

```
                DW      98EDH           ;4号功能,装载词组、联想词库
                DW      9993H           ;5号功能,显示中断管理
                DW      98CEH           ;6号功能,模拟功能键输入
                DW      99EBH           ;7号功能,调用显示和键盘模块中
                                        ;的子程序

;6 号功能,模拟功能键输入
L_98CE:
                MOV     AH,AL           ;AH=扩展ASCII码
                XOR     AL,AL           ;AL=0,表示是扩展字符
                CALL    L_9A70          ;伪输入
                JMP     SHORT L_98FE

;2 号功能,取控制键状态
L_98D7:
                MOV     AL,ES:[17H]     ;取ROM BIOS数据区控制键状态字
                                        ;节
                JMP     SHORT L_98FE

;3 号功能,装入预选字
L_98DD:
                OR      AL,AL           ;AL=0吗?
                JNZ     L_98E7          ;不是
                MOV     D_9A6A,DX       ;保存预选字段址
                JMP     SHORT L_98FE
L_98E7:
                MOV     D_9A6C,DX       ;保存段址
                JMP     SHORT L_98FE

;4 号功能,装载词组、联想词库
L_98ED:
                MOV     BL,6            ;每种词组占6字节
                MUL     BL              ;剩以BL
                ADD     AX,9680H        ;得词组信息保存区地址
                MOV     BX,AX
                MOV     [BX],BP         ;保存词组区段址
                MOV     [BX+2],DX       ;保存词组编码区首址
                MOV     [BX+4],CX       ;保存词组词组区首址
L_98FE:

                POP     DX
                POP     BP
                POP     CX
                POP     DI
                POP     SI
                POP     BX
                POP     DS
                POP     ES              ;恢复寄存器

                IRET                    ;中断结束

;0 号功能,读入字符
L_9907:
                CMP     D_95D5,0        ;单字缓冲区中有字符吗?
                JNE     L_9957          ;有
```

```
;单字缓冲区中无字符,还要等待用户输入
L_990E:
                XOR     AH,AH                   ;0号功能
                INT     7EH                     ;调用原INT 16H读入字符
                                                ;AL=ASCII码, AH=扫描码
                CMP     AL,80H                  ;>80H(扩展ASCII字符)吗?
                JB      L_9918                  ;不是
                XOR     AL,AL                   ;将原INT 10H输入的扩展字符改
                                                ;为无效字符
L_9918:
                PUSH    AX                      ;保存读入的键值

                STI                             ;开中断
                CALL    L_9A70                  ;对AX进行处理
                CLI                             ;关中断
                POP     AX                      ;恢复输入的AX值
                CMP     D_95D5,1                ;单字缓冲区中有字符了吗?
                JB      L_990E                  ;仍然没有
                JNZ     L_9942                  ;缓冲区字符个数大于1

;处理后单字缓冲区中只有 1 个字符, 表示读入的字符长度仅 1, 而且不是汉字
                CMP     AX,0E7FH                ;是CTRL_BACKSPACE吗?
                JE      L_9933                  ;是

;其它字符直接返回. 因为只有一个字符, 所以缓冲区中已没有字符了.
                DEC     D_95D5                  ;清除缓冲区长度
                JMP     SHORT L_98FE            ;AX是读入的字符

;处理 CTRL_BACKSPACE, 将该字符转换为两个 BACKSPACE, 一个直接返回, 另一个保存在
;单字缓冲区中
L_9933:
                MOV     AL,0FFH
                MOV     D_2BA4,AL               ;设置缓冲区格式. 仅一个字符,
                                                ;且带有扫描码
                MOV     D_2BA9,AL
;该字符是由 INT 16H 修改过的, 因此, 这里将 D_2BA9 置为 0FFH, 目的是使下一次来测试
;键盘时, 返回无字符可读

                MOV     AL,8                    ;改为BACKSPACE
                MOV     WORD PTR DS:[959DH],AX;保存到单字缓冲区中
                JMP     SHORT L_98FE            ;此时,AX中已有字符

L_9942:
                MOV     D_2BA9,0FFH             ;置字符修改标志(见上)
                DEC     D_95D5                  ;单字缓冲区中字符数减1
                MOV     SI,OFFSET DS:[959DH]    ;取单字缓冲区首址
                LODSB                           ;取字符
                MOV     D_2BAA,SI               ;修改单字缓冲区首址
                XOR     AH,AH                   ;AH=0, AL=ASCII码
                JMP     SHORT L_98FE

;缓冲区中仍有字符, 不用从键盘读入, 直接取出即可
L_9957:
                CMP     D_2BA4,0FFH             ;缓冲区仅一个字符, 且带扫描码
```

```
                                                    ;吗?
                JNE     L_996D                      ;不是
                MOV     D_2BA4,0                    ;清2BA4标志
                MOV     AX,WORD PTR DS:[959DH]      ;取字符(AH=扫描码,AL=ASCII码)
                MOV     D_95D5,0                    ;清除单字缓冲区
L_996B:
                JMP     SHORT L_98FE

L_996D:
                MOV     SI,D_2BAA                   ;SI=单字缓冲区首址
                LODSB                               ;取字符,仅一个,且没有扫描码
                MOV     D_2BAA,SI                   ;修改单字缓冲区首址
                XOR     AH,AH                       ;没有扫描码
                DEC     D_95D5                      ;单字缓冲区长度减1
                JNZ     L_996B                      ;缓冲区中还有字符

;单字缓冲区中已没有字符了，还要检查词组有否词组可读
                MOV     D_2BAA,959DH                ;复位单字缓冲区首址

                CMP     D_966B,0                    ;有词组可读吗?
                JE      L_996B                      ;没有

;取词组内容到单字缓冲区
                PUSH    AX                          ;保存AX
                CALL    L_A701                      ;从词组缓冲区中取字符->内部缓
                                                    ;冲区，一次最多56个字符
                POP     AX                          ;恢复AX
                JMP     L_98FE

;5号功能,显示中断管理
L_9993:
                MOV     BX,9A58H                    ;显示中断地址存放地址1
                CMP     AL,0                        ;AL=0吗?
                JE      L_99AC                      ;是
                CMP     AL,1                        ;=1吗?
                JE      L_99B4                      ;是
                MOV     BX,OFFSET D_9A5C            ;显示中断地址存放地址2
                CMP     AL,2                        ;是2吗?
                JE      L_99B4                      ;是
                CMP     AL,3                        ;是3吗?
                JE      L_99AC                      ;是
                JMP     L_98FE                      ;无效的功能

;保存 INT 10H 中断向量
L_99AC:
                MOV     [BX],DX                     ;保存偏移地址
                MOV     [BX+2],BP                   ;保存段地址
L_99B1:
                JMP     L_98FE

;重置 INT 10H 中断向量
L_99B4:
                LDS     DX,DWORD PTR [BX]           ;取保存的显示中断地址
                MOV     AX,2510H
                INT     21H                         ;设置INT 10H
```

```
                JMP     SHORT L_99B1

;1 号功能,测试键盘
L_99BD:
                CLI                             ;关中断

                MOV     AH,1
                INT     7EH                     ;ROM BIOS键盘缓冲区中有字符吗?
                JNZ     L_99DF                  ;有,不用再检查缓冲区了
                XOR     D_2BA9,0FFH
                TEST    D_2BA9,80H              ;是改过的字符吗?
                JZ      L_99DF                  ;是的,第一调用返回没有字符可
                                                ;读
                CMP     D_95D5,0                ;缓冲区中有字符吗?
                JE      L_99DF                  ;没有
                MOV     DI,D_2BAA               ;DI=缓冲区指针
                MOV     AH,0
                MOV     AL,[DI]                 ;取字符
L_99DF:
                STI                             ;开中断
                POP     DX
                POP     BP
                POP     CX
                POP     DI
                POP     SI
                POP     BX
                POP     DS
                POP     ES                      ;恢复寄存器

                RETF    2                       ;中断返回. 为了保持ZF标志位,
                                                ;因此这里用RETF 2实现中断返回

;7 号功能,执行子程序
L_99EB:
                CALL    BP                      ;执行子程序
                JMP     L_98FE

;下面这段程序没有使用
                MOV     AX,40H
                MOV     DS,AX
                XOR     SI,SI
L_99F7:
                LODSW
                CMP     AX,3E80H
                JNE     L_99F7
                CMP     WORD PTR [SI],102H
                JNE     L_99F7
                MOV     [SI+2],DL
                DEC     DL
                MOV     [SI+9],DL
                RETN

                DB      20 DUP (0)              ;没有用
D_9A20          DB      '清理内存: 1-退出汉字, 5-驱动程序, 9-外加模块
```

```
D_9A58          DW      0                       ;1号INT 10H偏移地址
D_9A5A          DW      0                       ;1号INT 10H段地址
D_9A5C          DW      0                       ;2号INT 10H偏移地址
D_9A5E          DW      0                       ;2号INT 10H段地址
D_9A60          DB      '联想:'                 ;ALT_F10提示信息
D_9A65          DB      '      :'               ;ALT_F10提示信息

D_9A6A          DW      0                       ;预选字段地址
D_9A6C          DW      0                       ;与预选字类似,但没有使用
D_9A6E          DW      0                       ;同上

;子程序: 字符综合处理
;输   入: AH=扫描码
;         AL=ASCII 码
;输   出: 无
L_9A70:
                CMP     AX,6400H                ;是CTRL_F7吗?
                JNE     L_9A91                  ;不是

;处理 CTRL_F7
                MOV     AL,ES:[49H]             ;取当前显示方式
                CMP     AL,3                    ;在文本方式吗?
                JBE     L_9A81                  ;是

;在汉字输入方式下,按 CTRL_F7 转换为西文方式
                MOV     AL,3                    ;设置为 3 号显示方式
                JMP     SHORT L_9A83

;在西文方式下,按 CTRL_F7 进入汉字显示方式
L_9A81:
                MOV     AL,6                    ;设置为6号显示方式
L_9A83:
                XOR     AH,AH
                INT     10H                     ;设置显示方式
L_9A87:
;伪输入一个回车
                MOV     BP,SP
                MOV     WORD PTR [BP+2],1C0DH   ;修改堆栈内容
;在恢复寄存器时,使得 AX 等于回车字符

L_9A8E:
                JMP     L_9BB9

L_9A91:
                CMP     BYTE PTR ES:[49H],4     ;在汉字显示方式中吗?
                JB      L_9A8E                  ;不是
                CMP     D_966F,0FFH             ;在内部词组管理状态吗?
                JNE     L_9AA7                  ;不是
                OR      AL,AL                   ;输入的是扩展字符吗?
                JZ      L_9A8E                  ;是
                JMP     L_9718                  ;转内部词组管理

L_9AA7:         OR      AL,AL
                JZ      L_9AAE                  ;扩展字符
                JMP     L_9C86                  ;一般ASCII字符处理
```

```
L_9AAE:
                JMP     SHORT L_9AF7

;下面这段程序没有使用，可能用于直接通过 INT 16H 的 7 号子功能执行的子程序.
                NOP
                NOP
                OR      CX,CX
                JZ      L_9AE7
                MOV     DS,CX
                MOV     SI,[112H]
                MOV     CX,DS:[110H]
                XOR     BH,BH

L_9AC1:
                LODSB
                CMP     AH,AL
                JE      L_9AD3
                LODSB
                MOV     BL,AL
                ADD     SI,BX
                LOOP    L_9AC1

                PUSH    CS
                POP     DS
                XOR     AL,AL
                JMP     SHORT L_9AE7
L_9AD3:
                LODSB
                MOV     CL,AL
                MOV     CS:D_95D5,AL
                MOV     DI,OFFSET DS:[959DH]
                PUSH    ES
                PUSH    CS
                POP     ES
                REP     MOVSB
                POP     ES
                PUSH    CS
                POP     DS
                RETN
                NOP
L_9AE7:
                CMP     AH,60H
                JNE     L_9AF7
                MOV     AX,D_9A6C
                XCHG    AX,D_9A6E
                MOV     D_9A6C,AX
                RETN

;扩展字符处理
L_9AF7:
                CMP     AH,5EH                  ;是CTRL_F1吗?
                JNE     L_9B31                  ;不是
;CTRL_F1处理
                MOV     AX,D_9A6A               ;取预选字段地址
                OR      AX,AX                   ;装了预选字吗?
```

```
                JNZ     L_9B04                  ;是
                RETN                            ;没有装预选,直接返回
L_9B04:
                MOV     D_963A,40H              ;置预选字输入方式
                MOV     DS,AX                   ;DS=预选字段地址

;显示预选字提示行
L_9B0B:
                XOR     DL,DL                   ;光标位置
                CALL    L_9F30                  ;设置提示行光标
                MOV     DL,20H                  ;空格
                CALL    L_A6FB                  ;显示一个空格
                MOV     SI,WORD PTR DS:[120H]   ;取预选字地址
                MOV     DL,61H                  ;编号=a
L_9B1B:
                PUSH    DX                      ;保存DX
                CALL    L_A6FB                  ;显示编号
                LODSB                           ;取汉字的前一个字节
                CALL    L_A6F9                  ;显示
                LODSB                           ;取汉字的后一个字节
                CALL    L_A6F9                  ;显示
                POP     DX                      ;恢复DX
                INC     DX                      ;编号加1
                CMP     DL,7BH                  ;显示完26个汉字了吗?
                JB      L_9B1B                  ;没有,继续显示

                PUSH    CS
                POP     DS                      ;恢复DS

                RETN

L_9B31:
                CMP     AH,61H                  ;是CTRL_F4吗?
                JNE     L_9B39                  ;不是
                JMP     L_A4B3                  ;处理CTRL_F4

L_9B39:
                CMP     AH,62H                  ;是CTRL_F5吗?
                JNE     L_9B41                  ;不是
                JMP     L_A410                  ;处理CTRL_F5

L_9B41:
                CMP     AH,63H                  ;是CTRL_F6吗?
                JNE     L_9B80                  ;不是

;处理CTRL_F6
                MOV     AH,10H                  ;AL=0,清提示行
                INT     10H
                MOV     BX,95D7H                ;设置颜色信息地址
                MOV     CX,11H                  ;字符串长度
                XOR     DL,DL                   ;光标位置
                CALL    L_A249                  ;显示
                XOR     BL,BL                   ;颜色=0
L_9B57:
                XOR     AH,AH
```

```
                INT     7EH             ;读一字符
                CMP     AL,0DH          ;是回车吗?
                JE      L_9B70          ;是
                CALL    L_A54F          ;显示输入字符
                AND     AL,0FH          ;将数字转换为数值
                MOV     BH,AL           ;保存于BH中
                MOV     AL,0AH          ;AL=10
                MUL     BL              ;剩以原数值
                ADD     AL,BH           ;加当前数字值
                MOV     BL,AL           ;保存颜色值
                JMP     SHORT L_9B57
L_9B70:
                OR      BL,BL           ;颜色值为0吗?
                JZ      L_9B78          ;是
                XOR     BH,BH           ;页号=0
                MOV     AH,0BH          ;设置调色板
L_9B78:
                INT     10H
                MOV     AX,1000H
                INT     10H             ;清提示行
                RETN

L_9B80:
                CMP     AH,65H          ;是CTRL_F8吗?
                JNE     L_9B9E          ;不是

;处理CTRL_F8
                MOV     BX,962DH        ;取消光标信息的地址
                XOR     D_962C,0FFH     ;取反光标状态
                JZ      L_9B92          ;有光标转为没有光标
                MOV     BL,20H          ;BX=建立光标信息的地址
                INC     AX              ;建立光标
L_9B92:
                PUSH    BX              ;保存BX
                MOV     AH,13H          ;取消/建立光标(AL=0:取消,AL=1
                                        ;建立)
                INT     10H
                POP     BX              ;恢复BX
                MOV     CX,0CH          ;字符串长度
                JMP     L_9CDE          ;显示

L_9B9E:
                CMP     AH,67H          ;是CTRL_F10吗?
                JNE     L_9BBF          ;不是
                JMP     L_A95F          ;处理CTRL_F10

L_9BA6:
                CMP     D_2BAC,0FFH     ;在纯中文方式吗?
                JNE     L_9BB9          ;不是
                MOV     DI,959DH        ;DI=单字缓冲区首址
                MOV     D_95D5,1        ;缓冲区长度=1
                JMP     L_A1E0          ;转处理纯中文
                NOP
L_9BB9:
                MOV     D_95D5,1        ;置缓冲区长度=1
```

```
                RETN

L_9BBF:
                CMP     AH,66H                  ;是CTRL_F9吗?
                JNE     L_9BD7                  ;不是
                MOV     BX,960FH                ;取建立纯中文信息地址
                XOR     D_2BAC,0FFH             ;取反纯中文标志
                JZ      L_9BD0                  ;建立纯中文
                MOV     BL,1                    ;取取消纯中文信息地址
L_9BD0:
                MOV     CX,0EH                  ;字符串长度
                JMP     L_9CDE                  ;显示
                NOP

L_9BD7:
                CMP     AH,68H                  ;是ALT_F1吗?
                JNE     L_9BF2                  ;不是

;处理ALT_F1
L_9BDC:
                MOV     BX,95EDH                ;取进入区位码提示信息地址
                MOV     AL,1                    ;区位码输入方式标志

;设置输入方式
L_9BE1:
                MOV     D_963A,AL               ;置输入方式标志
                CALL    L_9CDB                  ;显示提示信息
                MOV     BYTE PTR D_959C,0       ;当前码长=0
                MOV     D_959B,0FFH             ;设置当前没有编码输入标志，可
                                                ;以直接输入非编码字符
                RETN

L_9BF2:
                CMP     AH,69H                  ;是ALT_F2吗?
                JNE     L_9BFE                  ;不是

;处理ALT_F2
L_9BF7:
                MOV     BX,95E8H                ;取首尾码提示信息地址
                MOV     AL,2                    ;首尾码输入方式标志
                JMP     SHORT L_9BE1

L_9BFE:
                CMP     AH,6AH                  ;是ALT_F3吗?
                JNE     L_9C0A                  ;不是

;处理ALT_F3
L_9C03:
                MOV     BX,95F2H                ;取拼音提示信息地址
                MOV     AL,4                    ;拼音输入方式标志
                JMP     SHORT L_9BE1
L_9C0A:
                CMP     AH,6BH                  ;是ALT_F4吗?
                JNE     L_9C16                  ;不是
```

```
;处理ALT_F4
L_9C0F:
                MOV     BX,95F7H            ;取快速提示信息地址
                MOV     AL,8                ;快速输入方式标志
                JMP     SHORT L_9BE1

L_9C16:
                CMP     AH,6DH              ;是ALT_F6吗?
                JNE     L_9C22              ;不是
                MOV     BX,963BH            ;取ASCII提示信息地址
                MOV     AL,10H              ;ASCII输入方式标志
                JMP     SHORT L_9BE1

L_9C22:
                CMP     AH,70H              ;是ALT_F9吗?
                JNE     L_9C2A              ;不是
                JMP     L_9700              ;内部词组管理

L_9C2A:
                CMP     AH,71H              ;是ALT_F10吗?
                JNE     L_9C90              ;不是

;处理ALT_F10
                MOV     BX,9A65H            ;取设置联想提示信息地址
                XOR     D_9639,0FFH         ;联想标志取反
                JZ      L_9C3B              ;进入联想状态
                MOV     BL,60H              ;取取消联想提示信息地址
L_9C3B:
                JMP     L_9CDB

;区位码处理，已按过空格
L_9C3E:
                CMP     AL,2CH              ;是上翻一页键吗?
                JNE     L_9C54              ;不是

;区位码，上翻一页处理
                MOV     AX,D_2BBA           ;取当前页首汉字
                SUB     AH,0BH              ;位号减11
                CMP     AH,0A1H             ;当前区汉字(从当前页首汉字往
                                            ;数)已不满一页了吗?
                JAE     L_9C51              ;不是
;当前区已不满一页了，要取上一区的汉字了
                DEC     AX                  ;AL减1，即转上一区
                ADD     AH,5EH              ;AH=上一页第一个汉字的位号
L_9C51:
                JMP     L_9E0D

L_9C54:
                CMP     AL,2EH              ;是下翻一页键吗?
                JNE     L_9C60              ;不是
;区位码，下翻一页处理
                MOV     AX,[2BCEH]          ;取当前提示行的最后一个汉字
                INC     AH                  ;AH=下一页首汉字位号
                JMP     L_9E0D
```

```
L_9C60:
                CMP     AL,20H                  ;是空格吗?
                JNE     L_9C68                  ;不是
;区位码，已按空格后，再按空格相对于选择 0
                MOV     AL,30H                  ;将AL改为'0'
                JMP     SHORT L_9C74
L_9C68:
                CMP     AL,39H                  ;AL>'9'吗?
                JA      L_9C77                  ;是
                CMP     AL,30H                  ;AL<'0'吗?
                JB      L_9C77                  ;是
                JNZ     L_9C74                  ;是'1'-'9'

;0 是最后一个选择号，实际对应于提示行第 11 个同码字
                MOV     AL,3AH                  ;AL=':'
L_9C74:
                JMP     L_9D0A
L_9C77:
                JMP     L_9DC0

                DB      12 DUP (0)              ;没有用

;处理一般 ASCII 字符
L_9C86:
                TEST    D_963A,40H              ;是在预选字输人方式吗?
                JZ      L_9C90                  ;不是
                JMP     L_A4F0                  ;预选字输人

L_9C90:
                CMP     D_959B,0FFH             ;是否已经有编码输人?
                JNE     L_9C9A                  ;有
                JMP     L_9E68                  ;转处理首笔
L_9C9A:
                MOV     BX,OFFSET D_963A        ;BX=输人方式字节
                TEST    BYTE PTR [BX],20H       ;在联想状态吗?
                JZ      L_9CA5                  ;不是
                JMP     L_A798                  ;处理联想输人
L_9CA5:
                CMP     AL,3BH                  ;是';'吗?
                JNE     L_9CAC                  ;不是
                JMP     L_A5D7                  ;转处理外部词组
L_9CAC:
                CMP     AL,27H                  ;是"'"
                JNE     L_9CB3                  ;不是
                JMP     L_A5CD                  ;转处理内部词组
L_9CB3:
                TEST    BYTE PTR [BX],1         ;在区位码输人方式吗?
                JZ      L_9CBB                  ;不是
                JMP     L_9DCB                  ;转处理区位码输人L_9CBB:
                TEST    BYTE PTR [BX],2         ;在首尾码输人方式吗?
                JZ      L_9CC7                  ;不是

;首尾码处理
                CMP     AL,0DH                  ;是回车吗?
                JNE     L_9CF2                  ;不是
```

```
                JMP     L_9BF7                  ;相对于初始化首尾码输入方式
L_9CC7:
                TEST    BYTE PTR [BX],4         ;在拼音码输入方式吗?
                JNZ     L_9CEB                  ;是
                JMP     L_A081                  ;转快速处理

                CMP     D_95D5,3
                JE      L_9CD7
                RETN
L_9CD7:
                JMP     L_A13F
                NOP

;子程序: 显示提示信息, 长度为 5 个
;输   入: BX=字符串首址
;输   出: 无
L_9CDB:
                MOV     CX,5                    ;输入方式提示信息均为5个字节
                                                ;例如:'拼音:'、'联想:'等
;此时, CX 已为显示字符串的长度

L_9CDE:
                PUSH    AX                      ;保存AX
                MOV     AX,1000H
                INT     10H                     ;清提示行
                XOR     DL,DL
                CALL    L_A249                  ;显示提示信息
                POP     AX                      ;恢复AX
                RETN

;拼音处理
L_9CEB:
                CMP     AL,0DH                  ;是回车吗?
                JNE     L_9CF2                  ;不是
                JMP     L_9C03                  ;相对于初始化拼音码输入方式

;拼音码和首尾码输入方式共用
L_9CF2:
                CMP     AL,8                    ;是删除字符吗?
                JE      L_9D2A                  ;是
                CMP     AL,20H                  ;是空格吗?
                JNE     L_9CFE                  ;不是

;处理空格
                MOV     AL,30H                  ;将空格改为0
                JMP     SHORT L_9D0A
L_9CFE:
                CMP     AL,39H                  ;>'9'吗?
                JG      L_9D51                  ;是
                CMP     AL,30H                  ;<'0'吗?
                JB      L_9D51                  ;是
                JNZ     L_9D0A                  ;< >'0'
                MOV     AL,3AH                  ;='0',将'0'改为':',相当于10
L_9D0A:
                AND     AL,0FH                  ;将数字转换相应数值
```

```
L_9D0C:
                CMP     AL,BYTE PTR D_2BD2      ;选择数>同码字数吗?
                JG      L_9D27                  ;是,选择无效

;选择输人
                PUSH    BX                      ;保存BX
                MOV     BH,0
                MOV     BL,AL                   ;BX=选择号
                SHL     BX,1                    ;BX=BXx2
                ADD     BX,OFFSET D_2BBA        ;BX=相应汉字的地址
                MOV     BX,[BX]                 ;取汉字
                MOV     D_959D,BX               ;将汉字放人单字缓冲区
                POP     BX                      ;恢复BX
                JMP     L_A559

;选择无效，返回
L_9D27:
                JMP     L_9DC0

;处理删除字符
L_9D2A:
                CALL    L_9D7D                  ;设置提示光标位置
                CALL    L_9F35                  ;删除上一编码
                DEC     BYTE PTR D_959C         ;编码长度减1
                JNZ     L_9D4B                  ;还有编码

;输人的编码已全部删除
                MOV     BYTE PTR D_959C,1       ;再删除一个
                CALL    L_9D7D                  ;设置提示行光标位置
                MOV     BYTE PTR D_959C,0       ;置编码长度=0
                OR      D_959B,0FFH             ;没有编码已经输人
                JMP     SHORT L_9DC5
                NOP
L_9D4B:
                CALL    L_9FC0                  ;重新初始化检索地址
                JMP     L_9FA0                  ;重新按上一编码检索
L_9D51:
                CMP     AL,2CH                  ;是上翻一页吗?
                JE      L_9D8B                  ;是
                CMP     AL,2EH                  ;是下翻一页吗?
                JE      L_9DAE                  ;是
                CMP     AL,61H                  ;<'a'吗?
                JB      L_9D76                  ;是
                CMP     AL,7AH                  ;>'z'吗?
                JA      L_9D7A                  ;是

;处理编码
L_9D61:
                CMP     BYTE PTR D_959C,3       ;编码长度已大于3了吗?
                JA      L_9DC0                  ;是
                ADD     BYTE PTR D_959C,1       ;码长加1
                CALL    L_9D7D                  ;清除提示行编码后的字符
                CALL    L_9F89                  ;显示当前输人的编码
                JMP     L_9F9D
L_9D76:
```

```
                CMP     AL,5BH                  ;是'['吗?
                JE      L_9D61                  ;'['也是编码,转编码处理
L_9D7A:
                JMP     L_9BA6

;子程序: 将提示行光标位置设置于当前编码之后
;输  入: 无
;输  出: 无
L_9D7D:
                MOV     DL,4                    ;编码起始位置

L_9D7F:
                ADD     DL,BYTE PTR D_959C      ;加编码长度
                PUSH    AX                      ;保存AX
                MOV     AX,1002H
                INT     10H                     ;设置提示行光标位置
                POP     AX                      ;恢复AX
                RETN

;处理上翻一页
L_9D8B:
                CALL    L_A001                  ;单字检索初始化设置
                CALL    L_A011                  ;检索匹配汉字
                CALL    L_9FED                  ;为翻页设置上页尾址和下页首址
                CALL    L_A164                  ;显示同码字

L_9D97:
                CMP     BYTE PTR D_2BD2,0       ;同码字数为0吗?
                JE      L_9DC0                  ;是
                JMP     SHORT L_9DC5

                DB      14 DUP (0)              ;没有用

;处理下翻一页
L_9DAE:
                CALL    L_9FD0                  ;单字检索初始化设置
                CALL    L_A017                  ;反向检索匹配汉字
                CALL    L_9FD9                  ;为翻页设置上页尾址和下页首址
                CALL    L_A164                  ;显示同码字
                JMP     SHORT L_9D97

                DB      0, 0, 0, 0              ;没有用

L_9DC0:
;没有匹配汉字,响玲后返回
                MOV     DL,7                    ;响玲字符
                CALL    L_A6FB                  ;显示响玲字符
L_9DC5:
                MOV     D_95D5,0                ;清单字缓冲区
                RETN

;区位码输入
L_9DCB:
                CMP     AL,0DH                  ;是回车吗?
                JNE     L_9DD2                  ;不是
```

```
                JMP     L_9BDC                  ;相当于直接按ALT_F1，转初始化
                                                ;区位码输入方式
L_9DD2:
                CMP     BYTE PTR D_959C,5       ;已按过空格了吗?
;在区位码输入时，按过空格后，其码长设置为 5
                JNE     L_9DDC                  ;没有
                JMP     L_9C3E                  ;转按过空格后的区位码处理
L_9DDC:
                CMP     AL,8                    ;是删除键吗?
                JNE     L_9DF7                  ;不是

;区位码，处理删除键
L_9DE0:
                MOV     DL,8
                CALL    L_A6FB                  ;删除上一编码
                DEC     BYTE PTR D_959C         ;码长减1
                CMP     BYTE PTR D_959C,0       ;已全部删除吗?
                JNE     L_9DC5                  ;没有
                OR      D_959B,0FFH             ;置没有编码输入标志
                JMP     SHORT L_9DC5

L_9DF7:
                CMP     AL,20H                  ;是空格吗?
                JNE     L_9E2A                  ;不是

;区位码，处理空格，进入翻页状态
                MOV     BYTE PTR D_959C,5       ;置按过空格标志
                MOV     AX,D_2BB1               ;取前两位编码
                XCHG    AL,AH                   ;AH=编码1,AL=编码2
                AND     AX,0F0FH                ;转换为数值
                AAD                             ;AHx10+AL->AL,即AL=区号
                ADD     AX,0A1A0H               ;AX+1=该区第一个汉字
L_9E0D:
                MOV     SI,OFFSET D_2BBA        ;同码字存放区首址
                MOV     CX,0BH                  ;共11个汉字
                MOV     D_2BD2,CX               ;置同码字数目

L_9E17:
                CMP     AH,0FFH                 ;已到该区最后一个汉字了吗?
                JNE     L_9E1F                  ;没有
                MOV     AH,0A1H
                INC     AX                      ;AX=下一区第一个汉字
L_9E1F:
                MOV     [SI],AX                 ;保存汉字
                INC     SI
                INC     SI                      ;调整地址
                INC     AH                      ;增加位号
                LOOP    L_9E17
                JMP     L_A172

L_9E2A:
                CMP     AL,30H                  ;AL<'0'吗?
                JB      L_9E6F                  ;是
                CMP     AL,39H                  ;AL>'9'吗?
                JA      L_9E6F                  ;是
```

```
;区位码编码输入
                INC     BYTE PTR D_959C         ;编码长度加1
                CALL    L_9F89                  ;显示当前输入编码
                CMP     BYTE PTR D_959C,4       ;已输满4位了吗?
                JB      L_9DC5                  ;没有

;区位码，已输入4位(即已得到完整的区位码)，根据输入的区位码计算相应的汉字
                MOV     SI,OFFSET D_2BB1        ;编码区地址
                LODSW                           ;取1,2两位编码
                XCHG    AL,AH                   ;AH=编码1,AL=编码2
                AND     AX,0F0FH                ;将数字转换为数值
                AAD                             ;AL=区号
                MOV     BL,AL                   ;BL=区号
                LODSW                           ;取3,4位编码
                XCHG    AL,AH                   ;AH=编码3,AL=编码4
                AND     AX,0F0FH                ;将数字转换为数值
                AAD                             ;AL=位号
                MOV     BH,AL                   ;BH=位号
                ADD     BX,0A0A0H               ;加基数,BX=汉字
                MOV     WORD PTR DS:[959DH],BX;保存汉字
                MOV     BYTE PTR D_959C,0
                JMP     L_A559                  ;转联想判断
                NOP

L_9E68:
                CMP     D_963A,10H              ;是ASCII输入吗?
                JNE     L_9E72                  ;不是
L_9E6F:
                JMP     L_9BA6                  ;处理ASCII输入
L_9E72:
                TEST    D_963A,20H              ;是联想输入吗?
                JZ      L_9E7C                  ;不是
                JMP     L_A73B                  ;联想处理
L_9E7C:
                TEST    D_963A,8                ;是快速输入方式吗?
                JZ      L_9E86                  ;是
                JMP     L_A039                  ;转快速处理
L_9E86:
                CMP     BYTE PTR D_959C,0       ;码长为0吗?
                JE      L_9EC0                  ;是，不可以用ALT_-和ALT_=进行
                                                ;翻页
                CMP     AH,82H                  ;>=ALT_-吗?
                JAE     L_9EAB                  ;是
                CMP     AH,78H                  ;<ALT_0吗?
                JB      L_9E9E                  ;是
;处理ALT_0-ALT_9
                MOV     AL,AH
                SUB     AL,77H                  ;将ALT_0-ALT-9转换为'0'-'9'
                JMP     L_9D0C                  ;转选择输入

L_9E9E:
                CALL    L_9F66                  ;测试ALT_' ', 若为ALT+' ', 将
                                                ;AL改为0
                JNC     L_9EC0                  ;不是ALT_' '
```

```
                JMP     L_9D0C                  ;是，此时AL已为0，相当于选择
                JMP     L_9D0C                  ;是，此时AL已为0，相当于选择
                NOP                     ;了第一个同码字,转选择输人

                DB      4 DUP (0)               ;没有用

L_9EAB:
                JNZ     L_9EB4                  ;不是ALT_-转
;处理ALT_-
                NOT     D_959B                  ;恢复已经有编码输人标志
                JMP     L_9D8B                  ;转上翻一页

L_9EB4:
                CMP     AH,83H                  ;<>ALT_=吗?
                JNE     L_9EC0                  ;是

;处理ALT_=
                NOT     D_959B                  ;恢复已经有编码输人标志
                JMP     L_9DAE                  ;转下翻一页

L_9EC0:
                TEST    D_963A,1                ;在区位码输人方式吗?
                JZ      L_9F09                  ;不是

;处理区位码
                CMP     BYTE PTR D_959C,5       ;已有空格输人了吗?
                JNE     L_9ED1                  ;没有
                JMP     L_9DCB                  ;转按过空格后的区位码处理

;区位码首笔处理
L_9ED1:
;非数字直接输人
                CMP     AL,30H                  ;<'0'吗?
                JB      L_9F06                  ;是
                CMP     AL,39H                  ;>'9'吗?
                JA      L_9F06                  ;是

;是'0'-'9', 作为第一编码处理
                MOV     BX,95EDH                ;ALT_F1提示信息地址
                CALL    L_9CDB                  ;显示提示信息
                MOV     BYTE PTR D_2BB1,AL      ;保存编码
                CALL    L_A6F9                  ;显示编码
                MOV     BYTE PTR D_959C,1       ;码长设置为1
                JMP     SHORT L_9EFA

                DB      14 DUP (0)              ;没有用
L_9EFA:
                AND     D_963A,0DFH             ;去除联想标志
                NOT     D_959B                  ;设置已经有编码输人标志
                JMP     L_9DC5
L_9F06:
                JMP     L_9BA6

;首尾码和拼音码的第一编码处理
L_9F09:
```

```
                CMP     AL,61H                  ;<'a'吗?
                JL      L_9F06                  ;是
                CMP     AL,7AH                  ;>'z'吗?
                JG      L_9F06                  ;是
                MOV     BX,95E8H                ;ALT_F2的提示信息
                TEST    D_963A,2                ;在首尾码输入方式吗?
                JNZ     L_9F1E                  ;是
                MOV     BX,95F2H                ;取ALT_F3的提示信息
L_9F1E:
                CALL    L_9CDB                  ;显示提示信息
                MOV     BYTE PTR D_959C,1       ;码长为1
                CALL    L_9D7D                  ;设置提示行光标
                JMP     SHORT L_9F50            ;转继续处理

                DB      0,0                     ;没有用
                NOP

;子程序: 设置提示行光标并将光标后的字符全部清除, 光标位置为 5
;输   入: 无
;输   出: 无
L_9F2E:
                MOV     DL,5                    ;光标位置

;此时, 光标位置已在 DL 中
L_9F30:
                MOV     AX,1002H
                INT     10H                     ;设置提示行光标位置
L_9F35:
                MOV     CX,7FH                  ;总字符数
                SUB     CL,DL                   ;减光标位置=要清除的字符数
                PUSH    DX                      ;保存DX
                MOV     DL,20H                  ;用空格清除
L_9F3D:
                CALL    L_A6FB                  ;显示空格即清除
                LOOP    L_9F3D                  ;显示CX次

                POP     DX                      ;恢复DX

;此时, 仅设置提示行光标位置

L_9F43:
                MOV     AX,1002H
                INT     10H                     ;设置提示行光标位置
                RETN

                NOP
                DB      6 DUP (0)               ;没有用
;首尾码和拼音码的第一编码处理
L_9F50:
                MOV     D_959B,0FFH             ;设置已经有编码输入标志
                CALL    L_9F89                  ;显示输入的编码
                CALL    L_9FC0                  ;单字检索初始化设置
                CALL    L_A017                  ;检索匹配汉字
                CALL    L_9FD9                  ;为翻页设置上页尾址和下页首址
                CALL    L_A164                  ;显示同码字
```

```
                JMP     SHORT L_9EFA

;子程序: 测试当前字符是否为 ALT_' ', 若是则将它转换为 0
;输   入: AH=扫描码, AL=ASCII 码
;输   出: 若是 ALT_' ', 则 CF=1, AL=0
;         否则,CF=0
L_9F66:
                CMP     AH,39H                  ;扫描码是' '吗?
                JNE     L_9F77                  ;不是
                TEST    BYTE PTR ES:[17H],8     ;ALT键按下了吗?
                JZ      L_9F77                  ;没有
                XOR     AL,AL                   ;是ALT_' ',设置AL=0
                STC                             ;CF=1
                RETN
L_9F77:
                CLC                             ;CF=0
                RETN

                DB      6 DUP (0)               ;没有用

L_9F7F:
                MOV     AX,D_2BB1+2             ;取同码字区中的第二个汉字
;这里实际上应该取同码字区的第一个汉字
                MOV      WORD PTR DS:[959DH],AX  ;保存在单字缓冲区中
                JMP      L_A559
                NOP

;子程序: 显示当前输入编码
;输   入: 无
;输   出: 无
L_9F89:
                PUSH    AX                      ;保存AX
                XOR     BX,BX                   ;BX=0
                MOV     BL,BYTE PTR D_959C      ;取编码长度
                DEC     BL                      ;减1
                MOV     BYTE PTR D_2BB1[BX],AL ;取当前编码
                MOV     DL,AL
                CALL    L_A6FB                  ;显示编码
                POP     AX                      ;恢复AX
                RETN

;编码输入处理
L_9F9D:
                CALL     L_9FD0                  ;取当前页编码区首址
;这里是第一次查找, 实际上还没有保存下页首址
L_9FA0:
                MOV     D_2BAD,2BD6H            ;设置当前检索编码首址为单字编
                                                ;码表首址, 即从头开始检索
                CALL    L_A017                  ;检索匹配汉字
                CALL    L_9FD9                  ;为翻页保存上页尾址和下页首址
                CALL    L_A164                  ;显示同码字
                JMP     L_9DC5
                DB      14 DUP (0)              ;没有用

L_9FC0:
```

```
                LEA     SI,CS:[2BD6H]           ;SI=单字编码表首址
L_9FC4:
                MOV     D_2BAD,SI               ;设置当前编码区检索地址
                ADD     SI,4
                MOV     D_2BAF,SI               ;设置下一单字或词组的编码地址
                RETN

;子程序: 正向检索时, 取下页首址
;输   入: 无
;输   出: 无
L_9FD0:
                MOV     SI,D_2BAF               ;取保存的下页首址
L_9FD4:
                MOV     D_2BAD,SI               ;置当前检索地址
                RETN

;子程序: 正向检索结束后, 设置上页尾址和下页首址
;输   入: 无
;输   出: 无
L_9FD9:
                MOV     SI,D_2BAF               ;取当前页第二个编码地址, 该地
                                                ;址由初始化变量时设置
                SUB     SI,4                    ;SI=当前页第一个编码地址
                MOV     D_9646,SI               ;9646H=上页尾址
                MOV     SI,D_2BAD               ;当前页尾址
                MOV     D_2BAF,SI               ;2BAFH=下页首址
                RETN

;子程序: 反向检索结束后, 设置上页尾址和下页首址
;输   入: 无
;输   出: 无
L_9FED:
                MOV     SI,D_9646               ;取当前页尾址
                ADD     SI,4                    ;SI=下页首址
                MOV     D_2BAF,SI               ;2BAFH=下页首址
                MOV     SI,D_2BAD               ;当前页首址
                MOV     D_9646,SI               ;9646H=上页尾址
                RETN

L_A001:
                MOV     SI,D_9646               ;取上页尾址
                JMP     SHORT L_9FD4

;首尾码和拼音码按输入编码次数检索匹配汉字程序入口地址
D_A007          DW      OFFSET L_A28E           ;第一键
D_A009          DW OFFSET L_A2F0                ;第二键
D_A00B          DW      OFFSET L_A340           ;第三键
D_A00D          DW      OFFSET L_A393           ;第四键
D_A00F          DW      OFFSET L_9DC5           ;第五键

;反向检索匹配单字
L_A011:
                MOV     DI,2BD2H                ;DI=反向检索时单字编码表尾址
;单字编码表从 2BD6H 开始, 每个单字编码长度为 4 字节, 因此反向查找结束地址因为
2BD2H
```

```
                STD                                 ;设置反向检索标志
                JMP     SHORT L_A01A

;正向检索匹配单字
L_A017:
                MOV     DI,9596H                    ;DI=正向检索时单字编码表尾址

;单字检索总入口
L_A01A:
                MOV     BL,BYTE PTR D_959C          ;取当前码长
                XOR     BH,BH
                SHL     BX,1                        ;BX=码长x2
                MOV     SI,D_2BAD                   ;SI=当前页编码区首址
                AND     D_2BB9,0DFH                 ;置未满页标志
                XOR     AX,AX
                MOV     BYTE PTR D_2BD2,AL          ;当前页同码字数=0
                MOV     D_9597,AX                   ;总同码字数=0
                MOV     CX,AX                       ;CX=总同码字计数
                JMP     WORD PTR DS:[0A005H][BX] ;按码长转相应处理程序. 注意
                                                    ;这里没有第0笔

;快速首笔
L_A039:
                CMP     AH,82H                      ;>=ALT_-
                JAE     L_A046                      ;是
                CMP     AH,78H                      ;<ALT_0
                JB      L_A046                      ;是
                JMP     L_9E86                      ;转ALT_0-ALT_9处理

L_A046:
                CALL    L_A051                      ;处理
                JNC     L_A04E                      ;是一般字符
                JMP     L_9DC5                      ;已按指定功能处理
L_A04E:
                JMP     L_9BA6                      ;作为一般字符输入

;子程序: 处理快速首笔
;输   入: AX=输入字符
;输   出: CF=1, 表示已处理
;         CF=0, 表示未处理
L_A051:
                CMP     AL,61H                      ;<'a'
                JB      L_A07F                      ;是
                CMP     AL,7AH                      ;>'z'
                JA      L_A07F                      ;是
;输入的是编码键
                MOV     BX,95F7H                    ;ALT_F4提示信息
                CALL    L_9CDB                      ;显示提示信息
                MOV     BYTE PTR D_2BB1,AL          ;保存编码
                AND     D_963A,0DFH                 ;清除联想标志
                MOV     BYTE PTR D_959C,1           ;码长=1
                MOV     D_959B,CL                   ;置已有编码输入标志
                MOV     BYTE PTR D_2BD2,CL          ;同码字数=0
                PUSH    AX                          ;保存AX
```

```
                CALL    L_9F2E                  ;从当前光标开始清除提示行
                POP     AX                      ;恢复AX
                CALL    L_9F89                  ;显示当前编码
                STC                             ;CF=1,置已处理标志
                RETN
                NOP
L_A07F:
                CLC                             ;置未处理标志
                RETN

;快速处理
L_A081:
                CMP     AL,20H                  ;是空格吗?
                JNE     L_A0A4                  ;不是

;快速, 处理空格
                CMP     BYTE PTR D_959C,4       ;码长为4了吗?
                JE      L_A0A4                  ;是
                CALL    L_9FC0                  ;设置单字编码区检索首址
                CALL    L_A017                  ;检索匹配汉字
                CMP     BYTE PTR D_2BD2,0       ;没有匹配汉字吗?
                JNE     L_A09C                  ;不是, 有同码字
                JMP     L_A15C                  ;无同码字
L_A09C:
                MOV     BYTE PTR D_2BD2,1       ;同码字数置1
                JMP     L_9F7F                  ;取汉字结束

L_A0A4:
                CMP     AL,8                    ;是删除键吗?
                JNE     L_A0B6                  ;不是

;快速, 处理删除键
                INC     BYTE PTR D_959C
                CALL    L_9D7D                  ;将提示行光标设置在编码内容之
                                                ;后
                DEC     BYTE PTR D_959C         ;恢复码长
                JMP     L_9DE0                  ;转处理区位码删除键

L_A0B6:
                CMP     AL,0DH                  ;是回车吗?
                JNE     L_A0BD                  ;不是
                JMP     L_9C0F                  ;转处理回车

L_A0BD:
                CMP     BYTE PTR D_959C,0       ;码长=0?
                JE      L_A0C8                  ;是
                CMP     AL,5BH                  ;是'['吗?
                JE      L_A0D0                  ;也是编码
L_A0C8:
                CMP     AL,61H                  ;<'a'吗?
                JB      L_A108                  ;是
                CMP     AL,7AH                  ;>'z'吗?
                JA      L_A108                  ;是

;快速, 输入编码
```

```
L_A0D0:
                CMP     BYTE PTR D_959C,4       ;码长已为4?
                JNE     L_A0DA                  ;不是
                JMP     SHORT L_A124            ;转处理4笔快速
                NOP
L_A0DA:
                INC     BYTE PTR D_959C         ;码长加1
                CALL    L_9F89                  ;显示当前编码
                CMP     BYTE PTR D_959C,4       ;码长已为4
                JE      L_A0EB                  ;是,转处理4位
                JMP     L_9DC5                  ;结束

;快速，已输入4位编码，直接查找相应汉字
L_A0EB:
                CALL    L_9FC0                  ;初始化检索匹配汉字的变量
                CALL    L_A017                  ;查找匹配汉字
                CMP     BYTE PTR D_2BD2,0       ;有匹配的汉字吗?
                JNE     L_A0FB                  ;有
                JMP     L_9DC0                  ;没有同码字
L_A0FB:
                CMP     BYTE PTR D_2BD2,1       ;只有一个吗?
                JE      L_A09C                  ;是，直接输入
                CALL    L_A164                  ;显示提示行
                JMP     L_9DC5

L_A108:
                CMP     BYTE PTR D_959C,4       ;已有4位编码了吗?
                JNE     L_A121                  ;不是
                MOV     CL,BYTE PTR D_2BD2      ;CL=同码字数目
                ADD     CL,30H                  ;改为数字
                CMP     AL,30H                  ;输入字符<'0'吗?
                JB      L_A124                  ;是
                CMP     CL,AL                   ;大于同码字数吗?
                JB      L_A124                  ;是
                JMP     L_9D0A                  ;选择输入
L_A121:
                JMP     L_9DC0                  ;错误

L_A124:
                MOV     CX,D_2BBA+2             ;取第二个同码字
;应该是取第一个同码字

                MOV     D_959D,CX               ;保存汉字
                MOV     D_959B,0FFH             ;置没有编码输入标志
                CALL    L_A051                  ;再将当前输入字符作为首笔处理
                MOV     D_95D5,2                ;缓冲区长度=2
                JC      L_A154                  ;输入的也是编码转
                INC     D_95D5                  ;输入的是其它字符则将它输入
L_A13F:
                MOV     DI,OFFSET D_959D+2      ;DI=字符保存地址，因为已有一
                                                ;个汉字在单字缓冲区中，所以这
                                                ;里保存地址要从[959DH+2]开始
                CMP     D_2BAC,0FFH             ;在纯中文状态吗?
                JE      L_A151                  ;是，转纯中文处理
                MOV     [DI],AL                 ;不是，则直接将字符保存于单字
```

```
                                        ;缓冲区中
                RETN

                DB      00H, 00H, 00H, 00H,0C3H ;没有用

L_A151:
                JMP     L_A1E0                  ;转处理纯中文

L_A154:
                CMP     BYTE PTR D_2BD2,0       ;同码字=0吗?
                JE      L_A15C                  ;是
                RETN
L_A15C:
                JMP     L_9DC0

                NOP
                DB      4 DUP (0)               ;没有用

;子程序: 在提示行显示同码字内容, 若在显示过程中有键按下, 则停止显示返回
;输  入: 无
;输  出: 无
L_A164:
                MOV     AH,1
                INT     16H                     ;有键按下吗?
                JNZ     L_A1DE                  ;有,则结束显示

                MOV     DL,5                    ;光标位置
                CALL    L_9D7F                  ;设置提示行光标位置
                CALL    L_9F35                  ;清除提示行光标后的字符
L_A172:
                XOR     BX,BX                   ;BX=0
                MOV     CX,0BH                  ;最多可显示11个同码字
                MOV     DL,10H                  ;起始显示位置
                CMP     CX,D_9597               ;有剩余的同码字吗?
                JAE     L_A185                  ;没有
                SUB     D_9597,CX               ;取剩余同码字数, 有剩余同码字
                                                ;时, 当前页同码字数目必定为11
                JMP     SHORT L_A189
L_A185:
                MOV     D_9597,BX               ;剩余同码字数=0
L_A189:
                MOV     AX,1002H
                INT     10H                     ;设置提示行光标位置
                MOV     CL,BYTE PTR D_2BD2      ;取同码字个数
                OR      CL,CL                   ;有同码字吗?
                JNZ     L_A19D                  ;有
;没有匹配的汉字=0, 出错返回
                MOV     DL,7
                CALL    L_A6FB                  ;响玲
                JMP     SHORT L_A1CC

L_A19D:
                MOV     DL,30H                  ;从'0'开始
                PUSH    DX                      ;保存DX
                JMP     SHORT L_A1A6
```

```
L_A1A2:
                PUSH    DX                      ;保存DX
                CALL    L_A6FB                  ;显示编号
L_A1A6:
                MOV     DL,3AH
                CALL    L_A6FB                  ;显示':'
                MOV     DL,BYTE PTR D_2BBA[BX] ;取汉字前一字节
                CALL    L_A6FB                  ;显示
                MOV     DL,BYTE PTR D_2BBA+1[BX] ;取汉字后一字节
                CALL    L_A6FB                  ;显示
                MOV     DL,20H
                CALL    L_A6FB                  ;再加一个空格
                POP     DX                      ;恢复DX
                INC     DL                      ;编号加1
                CMP     DL,3AH                  ;第11个吗?
                JNE     L_A1C8                  ;不是
                MOV     DL,30H                  ;改为显示'0'
L_A1C8:
                INC     BX
                INC     BX                      ;BX=下一汉字位置
                LOOP    L_A1A2                  ;显示CX次

L_A1CC:
;显示剩余同码字数信息，包括两个中括号('[')
                MOV     DL,8                    ;删除字符
                CALL    L_A6FB                  ;显示
                MOV     DL,5BH                  ;'['
                CALL    L_A6FB                  ;显示
                CALL    L_A935                  ;显示剩余同码字数
                MOV     DL,5DH                  ;']'
                CALL    L_A6FB                  ;显示
L_A1DE:
                RETN
                NOP

L_A1E0:
                CMP     AL,20H                  ;小于空格吗?
                JAE     L_A1E7                  ;不是
                MOV     [DI],AL                 ;小于空格直接返回(不用进行纯
                                                ;中文扩展)
                RETN
L_A1E7:
                JA      L_A1F2                  ;大于空格，要作特殊处理
                MOV     AH,AL                   ;空格的纯中文就是两个空格L_A1EB:
                MOV     [DI],AX                 ;保存纯中文字符
                INC     D_95D5                  ;增加编码长度
                RETN

;处理一般纯中文字符
L_A1F2:
                MOV     SI,OFFSET D_A210        ;特殊纯中文字符表
                MOV     BL,AL                   ;BL=字符
                MOV     CX,0AH                  ;共10个
L_A1FA:
```

```
                LODSB                           ;取字符
                CMP     AL,BL                   ;是这个字符吗?
                JNE     L_A202                  ;不是
                LODSW                           ;是,取出这个字符对应的汉字
                JMP     SHORT L_A1EB
L_A202:
                LODSW                           ;SI=SI+2
                LOOP    L_A1FA                  ;再查

;不是特殊的纯中文字符,只要进行换算即可
                MOV     AH,BL                   ;AH=字符值
                MOV     AL,0A3H                 ;区号
                OR      AH,80H                  ;置高位即变为纯中文
                JMP     SHORT L_A1EB
                NOP
                NOP

;特殊字符纯中文表
D_A210          DB      '‘'
                DW      '、'
                DB      '.'
                DW      '。'
                DW      '’'
                DW      '‘'
                DB      '"'
                DW      '“'
                DB      '['
                DW      '〔'
                DB      ']'
                DW      '〕'
                DB      '\'
                DW      '／'
                DB      '|'
                DW      '｜'
                DB      '{'
                DW      '《'
                DB      '}'
                DW      '》'

                DB      10 DUP (0)              ;没有用

L_A238:
                CMP     BYTE PTR D_1820,55H     ;判断系统是否经正常初始化
                JNE     L_A240                  ;不是L_A23F:
                RETN                            ;子程序结束
L_A240:
;系统非经正常初始化, 每调用本子程序一次, 0A248H 要加 1, 当该单元数字溢;出时, 进
;入 ROM BASIC.
                INC     BYTE PTR DS:[0A248H]    ;每调用一次增加一次
                JNC     L_A23F                  ;没有溢出
                INT     18H                     ;溢出则进行ROM BASIC
                NOP

;子程序: 在提示行显示字符串
;输   入: DL=显示位置
```

```
;          BX=字符串首址
;          CX=字符串长度
;输   出: 无
L_A249:
                CALL    L_9F43                  ;设置提示行光标位置
L_A24C:
                MOV     DL,[BX]                 ;取一字符
                CALL    L_A6FB                  ;在提示行当前光标位置显示字符
                INC     BX                      ;下一字符地址
                LOOP    L_A24C                  ;显示CX个

                JMP     SHORT L_A238

                DB      0C3H                    ;没有用

;子程序:
;输   入: SI=当前汉字的编码内容地址
;输   出:
L_A257:
                PUSH    AX
                PUSH    DX
                PUSH    SI
                PUSH    BX                      ;保存寄存器

                MOV     BL,BYTE PTR D_2BD2      ;取同码字数
                CMP     BL,BYTE PTR D_2BD0      ;超过11个了吗?
                JAE     L_A289                  ;是, 不用再取汉字了, 此时CF=0

                MOV     AX,SI                   ;AX=当前汉字编码内容地址
                SUB     AX,2BD6H                ;AX=当前汉字的编码内容在编码
                                                ;表中的相对位移
                SHR     AX,1
                SHR     AX,1                    ;因为每个汉字的编码内容占4字
                                                ;节, 所以AX/4=当前汉字的顺序
                                                ;号(从汉字"啊"开始)
                MOV     DL,5EH                  ;每区有94个汉字
                DIV     DL                      ;AX除以94=在汉字区中的相对区号
                ADD     AX,2130H
                OR      AX,8080H                ;AX=汉字机内码
                XOR     BH,BH
                SHL     BL,1                    ;BX=BXx2
                MOV     D_2BBA[BX],AX           ;将取对的汉字保存于同码字区中
                SHR     BL,1
                INC     BL                      ;同码字数加1
                MOV     BYTE PTR D_2BD2,BL      ;修改同码字数
                STC                             ;CF=1
L_A289:
                POP     BX
                POP     SI
                POP     DX
                POP     AX                      ;恢复寄存器
                RETN

;首尾码和拼音码第一键检索程序, 此时检索变量均已设置好, SI=检索首址, DI=检索尾
;址, 检索方向也已设置好
```

```
L_A28E:
                MOV     BL,BYTE PTR D_2BB1      ;取编码，仅一字节
                AND     BL,1FH                  ;转换为数值，即将'a'-'z'->1-26
L_A295:
                CMP     SI,DI                   ;检索完了吗?
                JE      L_A2AF                  ;是

;查找匹配的编码
                MOV     AX,[SI]                 ;取当前汉字首尾码内容
                TEST    D_963A,4                ;当前是拼音输入方式吗?
                JZ      L_A2A5                  ;不是
                MOV     AX,[SI+2]               ;取拼音码内容
L_A2A5:
                AND     AL,1FH                  ;取编码1值
                CMP     AL,BL                   ;相同吗?
                JE      L_A2C7                  ;是
;不匹配
L_A2AB:
                LODSW
                LODSW                           ;使SI指向下一编码的地址，当
                                                ;DF=0时，SI=SI+4，DF=1时，SI
                                                ;=SI-4
                JMP     SHORT L_A295

;检索完毕
L_A2AF:
                OR      CX,CX                   ;有匹配的编码吗?
                JZ      L_A2C4                  ;没有
                MOV     D_9597,CX               ;保存总同码字数
                CMP     CX,D_2BD0               ;同码字数小于11吗?
                JA      L_A2C1                  ;不是，表示超过1页
;同码字未超过1页，还没有保存当前页尾址
                MOV     D_2BAD,DI               ;保存当前页尾地址
L_A2C1:
                STC                             ;CF=1，表示有匹配的编码，即有
                                                ;同码字
                CLD
                RETN

L_A2C4:
                CLC                             ;CF=0，表示没有匹配的编码，即
                                                ;没有同码字
                CLD                     RETN

;找到了匹配编码
L_A2C7:
                TEST    D_963A,4                ;当前是拼音输入方式吗?
                JZ      L_A2D4                  ;不是
;在拼音输入方式下，第一键显示的是高频汉字，因此判断当前汉字是否为高频字
                TEST    BYTE PTR [SI+3],80H     ;是高频字吗?
                JZ      L_A2AB                  ;不是，也不能当作同码字
L_A2D4:
                CALL    L_A2D9                  ;取当前汉字，保存于同码字区中
                JMP     SHORT L_A2AB
```

```
;子程序: 同码字处理
;输   入: CX=总同码字数
;输   出: CX=CX+1
L_A2D9:
                CALL    L_A257                  ;按当前编码地址计算汉字机内码,
                                                ;并保存于同码字区中
                JC      L_A2EE                  ;未满一页,表示有汉字取中,并
                                                ;保存在同码字区中

;已满页
                TEST    D_2BB9,20H              ;已有满页标志吗?
                JNZ     L_A2EE                  ;是
                MOV     D_2BAD,SI               ;保存当前页尾址
                OR      D_2BB9,20H              ;置满页标志
L_A2EE:
                INC     CX                      ;总同码字数加1
                RETN

;首尾码和拼音码第二键检索程序,此时检索变量均已设置好,SI=检索首址,DI=检索尾
;址,检索方向也已设置好
L_A2F0:
                TEST    D_963A,4                ;是拼音输入方式吗?
                JZ      L_A309                  ;不是

;拼音输入方式时要作一些特殊工作,即当第二编码为 i 或 u 时,由于同码字较多,所以将
;它扩展为 3 位编码. 第三位编码定为'[',表示没有第三笔编码.
                MOV     AL,BYTE PTR DS:[2BB2H]  ;取编码2
                CMP     AL,69H                  ;是i吗?
                JNE     L_A305                  ;不是
L_A2FE:
                MOV     BYTE PTR D_2BB3,5BH     ;置第三位为'['
                JMP     SHORT L_A340            ;转检索三笔
L_A305:
                CMP     AL,75H                  ;是u吗?
                JE      L_A2FE                  ;是,也转检索三笔

L_A309:
                MOV     BX,D_2BB1               ;BX=首尾码
                AND     BX,1F1FH                ;改为数值

;将两位编码转换为连续的 10 位数值(BL 中 8 位,BH 中 2 位)
                MOV     AH,BH                   ;AH=高5位
                XOR AL,AL                       ;AL=0
                SHR     AX,1
                SHR     AX,1
                SHR     AX,1                    ;将AH的低3位移入AL中的高3位
                MOV     BH,AH                   ;BH=10位数值的高2位
                OR      BL,AL                   ;将AL中的高3位复制到BL中,此
                                                ;时BX已是连续的10位数值
L_A31F:
                CMP     SI,DI                   ;检索完毕吗?
                JE      L_A2AF                  ;是

                MOV     AX,[SI]                 ;取当前汉字的首尾码
                TEST    D_963A,4                ;是拼音输入方式吗?
```

```
                JZ      L_A32F                  ;不是
                MOV     AX,[SI+2]               ;取当前汉字的拼音码
L_A32F:
                AND     AX,3FFH                 ;取10位数值
                CMP     AX,BX                   ;相同吗?
                JE      L_A33A                  ;相同
L_A336:
                LODSW
                LODSW                           ;使SI指向下一编码的地址，当
                                                ;DF=0时，SI=SI+4，DF=1时，SI
                                                ;=SI-4
                JMP     SHORT L_A31F
L_A33A:
                CALL    L_A2D9                  ;同码字处理
                JMP     SHORT L_A336

                NOP

;首尾码和拼音码第三键检索程序，此时检索变量均已设置好，SI=检索首址，DI=检索尾
;址，检索方向也已设置好.
L_A340:
                MOV     BX,D_2BB1               ;取编码1和编码2
                AND     BX,1F1FH                ;转换为数值
                MOV     AH,BH
                XOR     AL,AL
                SHR     AX,1
                SHR     AX,1
                SHR     AX,1
                MOV     BH,AH
                OR      BL,AL                   ;BX=10位数据
                MOV     AL,BYTE PTR D_2BB3      ;取编码3
                AND     AL,1FH                  ;转换为数值
                SHL     AL,1
                SHL     AL,1                    ;将AL的数据左移2位
                OR      BH,AL                   ;将第3码数值复制到BH中，这样
                                                ;BX=15位数据(3位编码)
L_A361:
                CMP     SI,DI                   ;检索完毕吗?
                JE      L_A38B                  ;是
                MOV     AX,[SI+2]               ;取拼音码
                AND     AX,7FFFH                ;屏幕高频位
                TEST    D_963A,4                ;是拼音输入方式吗?
                JNZ     L_A383                  ;是

;因为首尾码只有两位编码，所以第三码直接用拼音码的第一码取代
                MOV     AX,[SI]                 ;取首尾码，共10位有效数据
                AND     AX,3FFH                 ;得10位数据
                MOV     DL,[SI+2]               ;取拼音码的编码1
                AND     DL,1FH                  ;屏蔽
                SHL     DL,1
                SHL     DL,1
                OR      AH,DL                   ;AX=取得15位数据
L_A383:
                CMP     AX,BX                   ;是匹配的编码吗?
                JE      L_A38E                  ;是
```

```
L_A387:
                LODSW
                LODSW                           ;使SI指向下一编码的地址，当
                                                ;DF=0时，SI=SI+4，DF=1时，SI
                                                ;=SI-4
                JMP     SHORT L_A361
L_A38B:
                JMP     L_A2AF
L_A38E:
                CALL    L_A2D9                  ;同码字处理
                JMP     SHORT L_A387

;首尾码和拼音码第三键检索程序，此时检索变量均已设置好，SI=检索首址，DI=检索尾
;址，检索方向也已设置好．由于4位编码共20位数据，超过了16位(一字长度)．因此，
;处理时较复杂．若当前是拼音输入方式时，4位编码的前3位是拼音编码，最后一位是首
;尾码的第一码，其它在首尾和快速码输入方式时，前两位为首尾码，后两位为拼音码的
;前两位编码．
L_A393:
;4位编码共20位数据，下面将编码转换为连续的20位数据，分别存放在DH的低4位和BX中
                MOV     BX,D_2BB1               ;取前两位编码
                AND     BX,1F1FH                ;转换为数值
                MOV     AH,BH
                SHR     AX,1
                SHR     AX,1
                SHR     AX,1
                MOV     BH,AH
                OR      BL,AL                   ;BX=10数据
                MOV     AX,D_2BB3               ;取后两编码
                AND     AX,1F1FH                ;转换为数值
                SHL     AL,1
                SHL     AL,1                    ;AL空出两位
                OR      BH,AL                   ;BX=15位数据
                SHR     AH,1                    ;AH的低位有数据吗?
                JNC     L_A3BA                  ;没有
                OR      BH,80H                  ;取AH的低位数据,BX=16位数据
L_A3BA:
                MOV     DH,AH                   ;DH=高4位数据

L_A3BC:
                CMP     SI,DI                   ;检索完毕吗?
                JE      L_A401                  ;是
;取编码内容的20位数据，存放于DL的低4位和AX中
;先处理首尾和快速码输入方式
                MOV     AX,[SI]                 ;取首尾码
                AND     AX,3FFH                 ;得10位数据
                PUSH    BX                      ;保存BX
                MOV     BX,[SI+2]               ;取拼音码
                SHL     BL,1
                RCL     BH,1
                SHL     BL,1
                RCL     BH,1                    ;相当于SHL BX,2
                OR      AH,BL                   ;AX=16位数据
                AND     BH,0FH                  ;BH=高4位数据
                MOV     DL,BH                   ;保存于DL
                POP     BX                      ;恢复BX
```

```
                TEST    D_963A,4                ;是拼音输人方式吗?
                JZ      L_A3F5                  ;不是

;取拼音方式下的 20 位数据
                MOV     AX,[SI+2]               ;取拼音码(15位)
                AND     AX,7FFFH                ;拼音高频位
                TEST    BYTE PTR [SI],1         ;首尾码的第0位为1吗?
                JZ      L_A3EE                  ;不是
                OR      AH,80H                  ;首尾的第0位->AX的第15位, AX
                                                ;中已有16位数据
L_A3EE:
                MOV     DL,[SI]                 ;取首尾码的第1码
                AND     DL,1FH
                SHR     DL,1                    ;DL中只要4位即可, 最低位已放
                                                ;人AX上了

L_A3F5:
                CMP     AX,BX
                JNE     L_A3FD
                CMP     DH,DL                   ;是匹配的编码吗?
                JE      L_A404                  ;是
L_A3FD:
                LODSW
                LODSW                           ;使SI指向下一编码的地址, 当
                                                ;DF=0时, SI=SI+4, DF=1时, SI
                                                ;=SI-4
                JMP     SHORT L_A3BC
L_A401:
                JMP     L_A2AF
L_A404:
                CALL    L_A2D9                  ;同码字处理
                JMP     SHORT L_A3FD

                DB      0, 0, 0, 0              ;没有用

L_A40D:
                JMP     L_9BB9

;处理CTRL_F5
L_A410:         XOR     DL,DL                   ;光标位置
                MOV     BX,9A20H                ;CTRL_F5提示信息
                MOV     CX,37H                  ;字符串长度
                CALL    L_A249                  ;显示
                MOV     AH,0
                INT     7EH                     ;读字符
                PUSH    AX                      ;保存AX
                MOV     AX,1000H
                INT     10H                     ;清提示行
                POP     AX                      ;恢复提示行
                CMP     AL,39H                  ;='9'吗?
                JNE     L_A440                  ;不是

;清理内存, 9-->外加模块
                MOV     AX,357DH
```

```
                INT     21H                     ;取INT 7DH地址
                MOV     AX,ES                   ;AX=段址，即FILE24A.COM的段址
                DEC     AX                      ;AX=FILE24A内存控制块段址
                MOV     DS,AX                   ;DS=AX
                MOV     AX,DS:[1]               ;取段址
                ADD     AX,DS:[3]               ;加长度=下一内存控制块段地址
                CALL    L_A488                  ;释放FILE24A.COM以后的程序所
                                                ;占用的内存空间
                JMP     SHORT L_A483

L_A440:
                CMP     AL,35H                  ;='5'吗?
                JNE     L_A451                  ;不是

;清理内存，5-->驱动程序
                MOV     AX,3517H
                INT     21H                     ;取INT 17H地址
                MOV     AX,ES
                DEC     AX                      ;AX=打印驱动程序的内存控制块
                                                ;段址
                CALL    L_A488                  ;释放打印驱动程序占用的内存
                JMP     SHORT L_A479

L_A451:
                CMP     AL,31H                  ;='1'吗?
                JNE     L_A40D                  ;不是

;清理内存，1-->退出汉字
                MOV     AX,357FH
                INT     21H                     ;取INT 7FH地址
                MOV     AX,ES
                DEC     AX                      ;AX=读字库程序的内存控制块
                CALL    L_A488                  ;释放所有占用内存

;恢复系统启动时修改的中断向量
                LDS     DX,DWORD PTR CS:[9890H] ;取原INT 10H地址
                MOV     AX,2510H
                INT     21H                     ;恢复INT 10H地址
                MOV     AX,2
                INT     10H                     ;设置光标
                LDS     DX,DWORD PTR CS:[9894H] ;取原INT 16H中断向量
                MOV     AX,2516H
                INT     21H                     ;恢复原INT 16H中断向量
L_A479:
                LDS     DX,DWORD PTR CS:[9898H] ;取原INT 17H中断向量
                MOV     AX,2517H
                INT     21H                     ;恢复原INT 17H中断向量
;由于按 CTRL_F5 时可能正在 INT 21H 代码中，因此，这里直接调用 INT 21H 恢复中断向量
;有可能造成 DOS 代码重入，引起死机.

L_A483:
                PUSH    CS
                POP     DS                      ;DS=CS
                JMP     L_9A87                  ;用回车退出
```

```
;子程序: 释放内存
;输　入: AX=内存控制块段址
;输　出: 无
L_A488:
                XOR     SI,SI                   ;SI=0
L_A48A:
                MOV     DS,AX                   ;DS=AX
                CMP     BYTE PTR [SI],5AH       ;是内存控制块的最后一项吗?
                JE      L_A4AD                  ;是, 不用清理
                CMP     BYTE PTR [SI],4DH       ;是其它内存控制块吗?
                JE      L_A49B                  ;是, 转释放

;不是内存控制, 往上查找, 直到找到一内存控制块, 这一方法显然是不可靠的
                SUB     AX,10H                  ;指向上一节地址
                JMP     SHORT L_A48A

;释放当前内存块以下的内存空间, 方法是将当前内存块设置为系统最后的内存块
L_A49B:
                MOV     AX,WORD PTR CS:[988EH] ;取系统总内存量
                SUB     AX,DS:[1]               ;减当前内存控制块段址, 即为当
                                                ;前内存块后(包括当前内存块)所
                                                ;有空间长度
                MOV     DS:[3H],AX              ;设置长度
                MOV     DS:[1],SI               ;设置为空闲
                MOV     BYTE PTR [SI],5AH       ;置最后内存块标记

L_A4AD:
                RETN

D_A4AE          DB      '打印:'                 ;CTRL_F4提示信息

;处理CTRL_F4
L_A4B3:
                MOV     BX,0A4AEH               ;CTRL_F4提示信息
                CALL    L_9CDB                  ;显示
                XOR     SI,SI                   ;输入字符串存放地址
L_A4BB:
                MOV     AH,0
                INT     16H                     ;从键盘读一字符, 可以是汉字
                CMP     AL,8                    ;是删除键吗?
                JNE     L_A4C9                  ;不是
                DEC     SI                      ;存放地址减1
;这里, 当输入的第一个字符为删除字符时, 将出现错误. 因为, 此时 SI=0, SI-1=0FFFFH

L_A4C4:
                CALL    L_A6F9                  ;删除
                JMP     SHORT L_A4BB
L_A4C9:
                CMP     AL,0AH                  ;是CTRL_J吗?
                JE      L_A4DC                  ;是
                CMP     AL,0DH                  ;是回车吗?
                JE      L_A4D6                  ;是
                MOV     [SI],AL                 ;其它字符均作为字符串内容保存
                INC     SI                      ;存放地址加1
                JMP     SHORT L_A4C4
```

```
;CTRL_F4 回车处理，在打印时加回车
L_A4D6:
                MOV     WORD PTR [SI],0A0DH     ;添加回车
                INC     SI
                INC     SI

;CTRL_F4 CTRL_J 处理，在打印时不加回车，直接送打印机
L_A4DC:
                MOV     DI,SI
                XOR     SI,SI                   ;从地址0开始打印
L_A4E0:
                CMP     SI,DI                   ;结束了吗?
                JE      L_A4ED                  ;是
                LODSB                           ;取字符
                XOR     DX,DX
                MOV     AH,0
                INT     17H                     ;打印
                JMP     SHORT L_A4E0
L_A4ED:
                JMP     L_9DC5

;预选字输入
L_A4F0:
                MOV     SI,D_9A6A               ;取预选字段地址
                MOV     DS,SI
                CMP     AL,2CH                  ;是','吗?(上翻一页)
                JNE     L_A512                  ;不是

;预选字，上翻一页
                CMP     BYTE PTR DS:[122H],1    ;当前页号为1吗?
                JE      L_A50D                  ;是，不能翻页
                DEC     BYTE PTR DS:[122H]      ;页号减1
                SUB     WORD PTR DS:[120H],36H ;当前页首址
                JMP     L_9B0B

L_A50D:
                PUSH    CS
                POP     DS                      ;恢复DS
                JMP     L_9DC0                  ;出错

L_A512:
                CMP     AL,2EH                  ;是'.'吗?(下翻一页)
                JNE     L_A52C                  ;不是

;预选字，下翻一页
                MOV     SI,WORD PTR DS:[120H]   ;取当前页首址
                CMP     BYTE PTR [SI+36H],24H   ;是最后一页了吗?
                JE      L_A50D                  ;是，不能再往下翻页了
                INC     BYTE PTR DS:[122H]      ;页号加1
                ADD     WORD PTR DS:[120H],36H ;当前页首址
                JMP     L_9B0B

L_A52C:
                CMP     AL,61H                  ;>a吗?
```

```
                JAE     L_A535                  ;是

;小于 a, 作为一般字符直接输入
L_A530:
                PUSH    CS
                POP     DS                      ;恢复DS
                JMP     L_9C90

L_A535:
                CMP     AL,7AH                  ;大于z吗?
                JA      L_A530                  ;是, 也直接输入

;预选字选择输入
                SUB     AL,61H                  ;AL=AL-'a'=序号
                XOR     AH,AH                   ;AH=0
                SHL     AX,1                    ;AX=AXx2
                ADD     AX,WORD PTR DS:[120H]   ;当前所选择汉字地址
                MOV     SI,AX
                LODSW                           ;汉字->AX
                NOP
                NOP
                PUSH    CS
                POP     DS                      ;恢复DS
                MOV     WORD PTR DS:[959DH],AX;将汉字保存于单字缓冲区中
                JMP     SHORT L_A559

;子程序: 在提示行当前光标位置显示字符, 光标进一
;输  入: AL=字符
;输  出: 无
L_A54F:
                MOV     DL,AL                   ;DL=字符
                PUSH    AX                      ;保存AX
                MOV     AX,1004H
                INT     10H                     ;显示字符
                POP     AX                      ;恢复AX
                RETN

;按单字查找联想词组
L_A559:
                MOV     D_95D5,2                ;缓冲区长度=2
                MOV     D_959B,0FFH             ;置没有编码输入标志
                CMP     D_9639,0FFH             ;在联想词组状态吗?
                JNE     L_A58F                  ;不是
                CMP     D_9687,0                ;联想词组安装了吗?
;[9687]是联想词组段地址, 若为表示没有安装联想词组
                JE      L_A58F                  ;没有
                MOV     AX,WORD PTR DS:[959DH];取当前输入的汉字
L_A574:
                CMP     AL,0B0H                 ;是汉字吗?
                JB      L_A58F                  ;不是

;汉字联想处理
                MOV     D_2BB1,AX               ;当前输入的汉字作为编码保存在
                                                ;编码区中
                MOV     BYTE PTR D_959C,2       ;码长为2
```

```
                MOV     D_966C,1                ;词组编号=1, 表示是联想词组
                MOV     BYTE PTR D_966D,0FFH    ;置进入词组标志
                MOV     BX,9A60H                ;取联想词组提示信息
                JMP     SHORT L_A5DF

L_A58F:
                RETN                            ;不进行联想处理, 直接返回

;词组联想处理
L_A590:
                CMP     D_9639,0FFH             ;在联想词组状态吗?
                JE      L_A598                  ;是
                RETN                            ;不进行联想处理, 直接返回

L_A598:
                MOV     AX,[DI-2]               ;取当前输入词组的最后一个汉字
                MOV     D_959B,0                ;置已经有编码输入标志
                JMP     SHORT L_A574            ;检查是否可以进行联想处理
                NOP

                DB      42 DUP (0)              ;没有用

;内部词组处理
L_A5CD:
                MOV     BX,969BH                ;内部词组显示信息地址
                MOV     D_966C,2                ;词组编号=2, 表示内部词组
                JMP     SHORT L_A5DF

;外部词组处理
L_A5D7:
                MOV     BX,95FCH                ;外部词组显示信息
                MOV     D_966C,0                ;词组编号=0, 表示外部词组

;通用词组检索程序, 此时 BX 已为当前词组提示信息地址, 966C 为词组编号, 0 表示是外
;部词组, 1 表示是联想词组, 2 表示是内部词组
L_A5DF:
                PUSH    D_959C                  ;保存编码长度
                CALL    L_9CDB'                 ;显示提示信息, 实际上该子程序
                                                ;在执行过程中, 并未破坏959CH
                                                ;的内容
                POP     D_959C                  ;恢复编码长度

                CALL    L_A658                  ;显示词组编码内容, 当是联想词
                                                ;组时显示的是汉字
                CALL    L_A66A                  ;取当前词组的段址及其它信息
                MOV     SI,D_2BA2               ;SI=词组编码区首址
                CALL    L_A809                  ;取当前词组长度->AX
                ADD     AX,D_2B9A               ;加词组词组区首址=下一条词组
                                                ;(内容)地址
                MOV     D_2B9E,AX               ;下一条词组(内容)地址
                CALL    L_9FC4                  ;保存当前页编码区首址
                MOV     D_2BA5,0                ;设置检索方向, 0:正向,FF:反向
                MOV     DI,D_2BA0               ;DI=编码区尾址
                CALL    L_A85A                  ;查找匹配词组
                CALL    L_A816                  ;保存上页尾址和下页首址
```

```
L_A610:
                OR      D_963A,20H              ;置联想标志
                MOV     D_959B,0                ;置已有编码输入标志
                CALL    L_A694                  ;在提示行显示同码字

;下面这些程序是为了增加解读难度而设置，没有实质用处
                CMP     BYTE PTR D_966D,0FFH    ;有在词组初始化时设置的标志吗?
                JE      L_A627                  ;有
                JMP     L_9DC5
L_A627:
                CMP     BYTE PTR D_1820+1,0AAH  ;是正常启动的吗?
                JE      L_A630                  ;是
                INT     21H                     ;不是，由于此时AH的值不定，因
                                                ;此这一语句可能出现错误
L_A630:
                RETN

                NOP
                DB      0,0                     ;没有用

;词组选择输入，AL为选择号
L_A634:
                MOV     D_959B,0FFH             ;置没有编码输入标志
                MOV     BL,3                    ;为保存词组的地址和长度，每条
                                                ;词组占3字节
                MUL     BL
                MOV     BX,AX
                MOV     AX,D_9648[BX]           ;取词组首址
                MOV     D_9669,AX               ;保存当前词组首址
                MOV     AL,D_964A[BX]           ;取词组长度
                MOV     D_966B,AL               ;保存当前词组长度
                CALL    L_A701                  ;将当前词组传送到单字缓冲区中，
                                                ;若词组长度大于56，则在单字缓
                                                ;冲区为空时，还要将剩余的词组
                                                ;传送到单字缓冲区中
                JMP     L_A590

                NOP
                DB      4 DUP (0)               ;没有用
;子程序：在提示行当前光标处显示当前输入的编码内容
;输  入：无
;输  出：无
L_A658:
                XOR     BX,BX                   ;BX=0
L_A65A:
                MOV     DL,BYTE PTR D_2BB1[BX]  ;取编码
                CALL    L_A6FB                  ;显示一字符
                INC     BL                      ;指向下一编码
                CMP     BL,BYTE PTR D_959C      ;已将编码全部显示完毕了吗?
                JB      L_A65A                  ;还没有，继续显示下一编码
                RETN

;子程序：根据当前词组编号，设置词组的段址、编码区首址、词组区首址等
;输  入：无
;输  出：无
```

```
L_A66A:
        MOV     SI,OFFSET D_9680        ;词组地址表地址
        MOV     AL,6                    ;每种词组占用6字节
        MUL     BYTE PTR [SI-14H]       ;剩以词组编号, SI-14H=966CH
        ADD     SI,AX                   ;SI=当前词组变量区首址
        LODSW                           ;取当前词组段址
        MOV     WORD PTR D_2B98,AX      ;设置当前词组段址
        LODSW                           ;取当前词组编码区首址
        MOV     D_2BA2,AX               ;保存编码区首址
        LODSW                           ;取词组区首址
        MOV     D_2BA0,AX               ;保存词组区首址
        MOV     D_2B9A,AX               ;当前检索起始地址
        MOV     D_2BA6,3AH              ;=':', 以后没有使用

        CMP     WORD PTR DS:[2795H],213EH ;系统是正常启动的吗?
        JE      L_A692                  ;是
        INT     18H                     ;进入ROM BASIC
L_A692:
        RETN

        NOP

;子程序: 在提示行显示词组同码字内容
;入口: 无
;出口: 无
L_A694:
        MOV     DL,0BH                  ;光标位置
        CALL    L_9F30                  ;设置提示行光标位置
        MOV     DL,BYTE PTR D_2BD2      ;取同码字个数
        OR      DL,DL                   ;=0吗?
        JZ      L_A6F5                  ;是
        XOR     DH,DH
        SUB     D_9597,DX               ;DX=剩余同码字数
        CALL    L_A1CC                  ;显示剩余同码字数
        MOV     DL,20H                  ;显示空格,第一个选择号
        CALL    L_A6FB
        MOV     DL,30H                  ;DL='0'
        MOV     CL,BYTE PTR D_2BD2      ;CL=同码字数
        PUSH    DX
        JMP     SHORT L_A6BC

L_A6B8:
        PUSH    DX
        CALL    L_A6FB                  ;显示选择数字
L_A6BC:
        MOV     DL,3AH                  ;显示':'
        CALL    L_A6FB
        MOV     BL,BYTE PTR D_2BD2      ;同码字数
        SUB     BL,CL                   ;减当前编号
        MOV     AL,3                    ;每条词组占3字节
        MUL     BL
        MOV     BX,AX                   ;当前词组的地址和长度地址
        MOV     SI,D_9648[BX]           ;取当前词组地址
        PUSH    CX                      ;保存CX
        MOV     CL,D_964A[BX]           ;CL=当前词组长度
```

```
                MOV     AX,WORD PTR D_2B98
                MOV     DS,AX                   ;DS=词组段址
L_A6DB:
                LODSB                           ;取一字节
                CALL    L_A6F9                  ;显示
                LOOP    L_A6DB                  ;显示当前词组

                MOV     DL,20H                  ;最后加一个空格
                CALL    L_A6FB                  ;显示空格
                PUSH    CS
                POP     DS                      ;恢复DS
                POP     CX
                POP     DX                      ;恢复DX,CX
                INC     DL                      ;编号加1
                CMP     DL,3AH                  ;是第11个吗?
                JNE     L_A6F3                  ;不是
                MOV     DL,30H                  ;将10改为0
L_A6F3:
                LOOP    L_A6B8                  ;显示CX个同码字
L_A6F5:
                RETN

                NOP
                NOP
                NOP

;子程序: 在提示行光标位置显示字符
;输   入: AL=字符
;输   出: 无
L_A6F9:
                MOV     DL,AL                   ;字符->DL

;此时 DL=要显示的字符
L_A6FB:
                MOV     AX,1003H
                INT     10H                     ;显示字符
                RETN
;子程序: 将词组内容传送到单字缓冲区中, 一次最多可传送 38H 个字节
;输   入: 无
;输   出: 无
L_A701:
                MOV     AX,WORD PTR D_2B98      ;词组段址
                MOV     SI,D_9669               ;词组内容首址
                MOV     CL,D_966B               ;词组长度
                LEA     DI,CS:[959DH]           ;单字缓冲区首址
                CMP     CL,38H                  ;词组长度小于38H个字节吗?
                JB      L_A721                  ;是

;词组长度大于 56 字节, 一次不能全部传送完(因为, 单字缓冲区长度仅为 56 字节). 还要
;修改词组地址和长度, 以便下次继续传送
                SUB     CL,38H                  ;剩余字节数
                MOV     D_966B,CL               ;修改词组长度
                MOV     CL,38H                  ;传送38H个
                JMP     SHORT L_A726
                NOP
```

```
;词组长度小于56个字节，可以一次传送完
L_A721:
                MOV     D_966B,0                ;清词组长度

L_A726:
                MOV     D_95D5,CL               ;置单字缓冲区长度
                MOV     DS,AX                   ;DS=词组段址
                PUSH    ES                      ;保存ES
                PUSH    CS
                POP     ES                      ;ES=CS
                XOR     CH,CH
                REP     MOVSB                   ;传送
                POP     ES                      ;恢复ES
                PUSH    CS
                POP     DS                      ;恢复DS
                MOV     D_9669,SI               ;修改词组内容首址
                RETN

;联想词组处理
L_A73B:
                CMP     AH,82H                  ;>=ALT_-吗?
                JAE     L_A760                  ;是
                CMP     AH,78H                  ;<ALT_0吗?
                JB      L_A757                  ;是

;处理 ALT_0-ALT_9
                MOV     AL,AH
                SUB     AL,77H                  ;转换为0-9
L_A749:
                CMP     BYTE PTR D_2BD2,AL      ;选择项>同码字数吗?
                JAE     L_A752                  ;是,不能输入
                JMP     L_A15C                  ;能输入
L_A752:
                JMP     L_A634

                DB      0, 0                    ;没有用
L_A757:
                CALL    L_9F66                  ;是ALT_' '吗?
                JC      L_A749                  ;是
                JMP     SHORT L_A767
                NOP
                NOP
L_A760:
                JZ      L_A76A                  ;ALT_-转
                CMP     AH,83H                  ;是ALT_=吗?
                JE      L_A783                  ;是
L_A767:
                JMP     L_9E7C                  ;转回

;处理ALT_-(词组)
L_A76A:
                CALL    L_A831                  ;设置检索时编码区和词组区起始
                                                ;地址
                MOV     DI,D_2BA2               ;词组编码区首址
                SUB     DI,4                    ;DI=反向检索尾址
```

```
                STD                         ;设置反向的字符串操作方式
                MOV     D_2BA5,0FFH         ;设置反向检索方向
                CALL    L_A85A              ;检索匹配词组
                CALL    L_A83A              ;保存上页尾址和下页首址
                JMP     L_A610

;处理ALT_=(词组)
L_A783:
                CALL    L_A851              ;设置检索时编码区和词组区起始
                                            ;地址
                MOV     DI,D_2BA0           ;当前词组编码区尾址
                MOV     D_2BA5,0            ;设置正向检索方向
                CALL    L_A85A              ;检索匹配词组
                CALL    L_A816              ;保存上页尾址和下页首址
                JMP     L_A610

L_A798:
                CMP     AL,0DH              ;是回车吗?
                JNE     L_A7C0              ;不是

;词组，回车处理
                MOV     BYTE PTR D_959C,0   ;清除编码长度
                AND     D_963A,0DFH         ;清除联想输入方式
                CALL    L_9F2E              ;设置提示行光标于5
                MOV     D_959B,0FFH         ;设置没有编码输入标志
                JMP     L_9DC5
                NOP
                DB      14 DUP (0)          ;没有用
L_A7C0:
                CMP     AL,2CH              ;是','吗?
                JE      L_A76A              ;是转上翻一页
                CMP     AL,2EH              ;'.'吗?
                JE      L_A783              ;是转下翻一页
                CMP     AL,39H              ;>'9'吗?
                JA      L_A7E2              ;是，转一般字符处理
                CMP     AL,30H              ;<'0'吗?
                JB      L_A7D9              ;是，转一般字符处理

;词组，数字选择处理
                JNZ     L_A7D4              ;不是'0'
                MOV     AL,3AH              ;选择0，实际上是选择第11个重
                                            ;码，因此，要将AL改为':'
L_A7D4:
                SUB     AL,30H              ;AL=选择号
                JMP     L_A749
L_A7D9:
                CMP     AL,20H              ;AL是空格吗?
                JNE     L_A7E2              ;不是
                XOR     AL,AL               ;输入空格，相对于选择第一个重
                                            ;码
                JMP     L_A749

L_A7E2:
                CMP     D_1820,0AA55H       ;系统是正常启动的吗?
                JNE     L_A7ED              ;不是，死机
```

```
                JMP     L_9E7C
L_A7ED:
                DB      28 DUP (0)              ;没有用

;子程序: 取当前词组长度
;输   入: SI=当前词组的编码区地址
;输   出: AX=长度(以字节为单位)
L_A809:
                PUSH    DS                      ;保存DS

                MOV     AX,WORD PTR D_2B98      ;AX=当前词组段址
                MOV     DS,AX                   ;DS=当前词组段地址
                MOV     AL,[SI+3]               ;取当前词组长度
                XOR     AH,AH                   ;AX=长度

                POP     DS                      ;恢复DS
                RETN

;功能: 词组经正向检索后, 保存上页尾址和下页首址(包括编码区地址和词组区地址)
;入口: 无
;出口: 无
L_A816:
                CALL    L_9FD9                  ;保存编码区上页尾址和下页首址
                MOV     SI,D_9646               ;取编码区上页尾址, 即当前页首
                                                ;址
                CALL    L_A809                  ;取当前页第一条词组长度->AX
                SUB     D_2B9E,AX               ;2B9EH=上页词组区尾址
                MOV     AX,D_2B9E
                MOV     D_2B9C,AX               ;保存上页词组区尾址
                MOV     AX,D_2B9A               ;取当前页尾址
                MOV     D_2B9E,AX               ;保存下页首址
                RETN

;子程序: 设置检索时编码区和词组区起始地址(反向检索)
;输   入: 无
;输   出: 无 L_A831:
                MOV     AX,D_2B9C               ;取上页词组区尾址
                MOV     D_2B9A,AX               ;设置当前词组区检索地址
                JMP     L_A001                  ;继续

;功能: 词组经反向检索后, 保存上页尾址和下页首址(包括编码区地址和词组区地址)
;入口: 无
;出口: 无
L_A83A:
                MOV     SI,D_9646               ;SI=当前页首址
                CALL    L_A809                  ;取当前页第一条词组长度
                ADD     AX,D_2B9C
                MOV     D_2B9E,AX               ;保存下页词组区首址
                MOV     AX,D_2B9A
                MOV     D_2B9C,AX               ;保存上页词组区尾址
                JMP     L_9FED

;子程序: 设置检索时编码区和词组区起始地址(正向检索)
;输   入: 无
;输   出: 无
```

```
L_A851:
                MOV     AX,D_2B9E               ;AX=下页词组区首址
                MOV     D_2B9A,AX               ;设置当前检索词组区首址
                JMP     L_9FD0

;子程序: 检索匹配词组
;输   入: 无
;输   出: 无
L_A85A:
                AND     D_2BB9,0DFH             ;设置未满页标志
                XOR     DH,DH                   ;当前页长度=0
                MOV     BYTE PTR D_2BD2,DH      ;当前页同码字数=0
                MOV     AX,D_2BB1               ;取编码(联想词组时为汉字)
                MOV     DL,BYTE PTR D_2BB3      ;第三码
                MOV     SI,D_2BAD               ;检索首址(编码区)
                MOV     BP,D_2B9A               ;检索首址(词组区)
                MOV     CX,WORD PTR D_2B98
                MOV     DS,CX                   ;DS=词组段址
                XOR     CX,CX                   ;总同码字个数为0
L_A87C:
                CMP     SI,DI                   ;检索完毕了吗?
                JNE     L_A899                  ;还没有, 继续检索

;检索完毕
                PUSH    CS
                POP     DS                      ;恢复DS
                MOV     D_9597,CX               ;保存总同码字数
                MOV     AL,BYTE PTR D_2BD2
                XOR     AH,AH                   ;AX=当前页同码字数目
                CMP     CX,AX                   ;总同码字超过一页吗?
                JA      L_A897                  ;是
                MOV     D_2BAD,SI               ;保存检索尾址(编码区)
                MOV     D_2B9A,BP               ;保存检索尾址(词组区)
L_A897:
                CLD
                RETN

;比较第一码
L_A899:
                CMP     AL,[SI]                 ;第一码相同吗?
                JNE     L_A8BD                  ;不同
                CMP     BYTE PTR CS:D_959C,1    ;码长为1吗?
                JNE     L_A8A8                  ;不是
                JMP     SHORT L_A8DC            ;是匹配的词组
                NOP

;比较第二码
L_A8A8:
                CMP     AH,[SI+1]               ;第二码相同吗?
                JNE     L_A8BD                  ;不同
                CMP     BYTE PTR CS:D_959C,2    ;码长为2吗?
                JNE     L_A8B8                  ;不是
                JMP     SHORT L_A8DC            ;是匹配的词组
                NOP

;比较第三码
```

```
L_A8B8:
                CMP     DL,[SI+2]               ;第三码相同吗?
                JE      L_A8DC                  ;相同

;当前编码不匹配
L_A8BD:
                MOV     BL,[SI+3]               ;取当前词组长度
                MOV     BH,0
                PUSH    AX                      ;保存AX
                LODSW
                LODSW                           ;SI指向下一编码地址，由于已经
                                                ;按检索方向设置了DF标志，因此，
                                                ;SI可能是减4，也可能是加4
                POP     AX                      ;恢复AX
                CMP     CS:D_2BA5,0FFH          ;是反向检索吗?
                JE      L_A8D2                  ;是
                ADD     BP,BX                   ;正向查找，BP=下一词组地址
                JMP     SHORT L_A87C
L_A8D2:
                MOV     BL,[SI+3]
                SUB     BP,BX                   ;逆向查找，BP=下一词组地址
                JMP     SHORT L_A87C

                NOP
                DB      0,0                     ;没有用

;是匹配的编码
L_A8DC:
                INC     CX                      ;总同码字数加1
                ADD     DH,[SI+3]               ;当前页长度加词组长度
                ADD     DH,3                    ;还要加一条词组其它要显示的长
                                                ;度，包括编号和两个空格
                TEST    CS:D_2BB9,20H           ;已满页了吗?
                JNZ     L_A8BD                  ;是
                MOV     BL,BYTE PTR CS:D_2BD2 ;BL=已有同码字数目
                CMP     DH,40H                  ;提示只能同码字显示长度只能占
                                                ;64字节，当前超过46时表示当前
                                                ;页已太长要换页了
                JA      L_A8FA                  ;是
                CMP     BL,0BH                  ;同码字个数也不能超过11个
                JB      L_A8FF                  ;没有
L_A8FA:
                CMP     BL,0                    ;BL=0吗?
;BL=0 表示当前页仅一条词组，且其长度超过一页长度
                JNE     L_A91B                  ;不是
L_A8FF:
                PUSH    AX                      ;保存AX
                MOV     AL,3                    ;每个同码字占3字节
                MUL     BL                      ;剩以同码字数
                MOV     BX,AX
                MOV     AL,[SI+3]               ;取当前词组长度
                MOV     CS:D_964A[BX],AL        ;保存词组长度
                MOV     CS:D_9648[BX],BP        ;保存词组地址
                POP     AX                      ;恢复AX
                INC     BYTE PTR CS:D_2BD2      ;同码字数加1
```

```
                JMP     SHORT L_A8BD

L_A91B:
                TEST    CS:D_2BB9,20H           ;已满页了吗?
                JNZ     L_A933                  ;是

;设置满页标志
                MOV     CS:D_2BAD,SI            ;保存当前页尾址(编码区)
                MOV     CS:D_2B9A,BP            ;保存当前页尾址(词组区)
                OR      CS:D_2BB9,20H           ;置满页标志
L_A933:
                JMP     SHORT L_A8BD

;子程序: 显示剩余同码字数
;输  入: 无
;输  出: 无
L_A935:
                MOV     AX,D_9597               ;AX=剩余同码字数
                MOV     BX,OFFSET D_A95E        ;数字串存放尾址
                MOV     CX,30AH                 ;CH=3,CL=10
L_A93E:
                DIV     CL                      ;除以10, AH为AX/10后的余数,
                                                ;即最低位数字
                OR      AH,30H                  ;将数值转换为数字
                MOV     [BX],AH                 ;保存
                MOV     AH,0                    ;AH=0
                DEC     BX                      ;存放地址减1, 即存放高一位数字
                DEC     CH                      ;CH=CH-1
                JNZ     L_A93E                  ;共3位
                MOV     CL,3                    ;长度=3
L_A94E:
                INC     BX
                MOV     DL,[BX]                 ;取字符
                CALL    L_A6FB                  ;显示
                LOOP    L_A94E                  ;循环显示3个

                RETN

                DB      7 DUP (0)               ;没有用
D_A95E          DB      0                       ;剩余同码字数字串存放处

;CTRL_F10处理
L_A95F:
                MOV     AX,1000H
                INT     10H                     ;清提示行
                MOV     BX,9870H                ;取CTRL_F10提示信息地址
                MOV     CX,0FH                  ;长度
                XOR     DL,DL
                CALL    L_A249                  ;显示
                MOV     BX,9880H                ;第二个要显示的字符串地址
                MOV     CX,0DH
                MOV     DL,14H
                CALL    L_A249                  ;显示

;输入打印字号
```

```
                MOV     DL,10H
                MOV     AX,1002H
                INT     10H                     ;设置提示行光标
                XOR     BL,BL                   ;输入长度
L_A983:
                XOR     AH,AH
                INT     7EH                     ;读字符
                CMP     AL,8                    ;是删除键吗?
                JNE     L_A996                  ;不是
                OR      BL,BL                   ;已有字符输入吗?
                JZ      L_A983                  ;没有，不能删除
                XOR     BL,BL
                CALL    L_A6F9                  ;删除
                JMP     SHORT L_A983

L_A996:
                CMP     AL,0DH                  ;是回车吗?
                JE      L_A9A1                  ;是
                MOV     BL,AL                   ;保存字符->BL
                CALL    L_A6F9                  ;显示输入字符
                JMP     SHORT L_A983
L_A9A1:
                OR      BL,BL                   ;没有输入字符吗?
                JZ      L_A9B7                  ;是,不能设置

;设置打印字号，通过打印 ESC+'I'+字符号序列即可
                XOR     DX,DX
                MOV     AX,1BH
                INT     17H                     ;打印ESC
                MOV     AX,49H
                INT     17H                     ;打印'I'
                MOV     AL,BL
                XOR     AH,AH
                INT     17H                     ;打印字号
;输入打印行距
L_A9B7:
                MOV     DL,23H
                MOV     AX,1002H
                INT     10H                     ;设置提示行光标
                XOR     BX,BX                   ;数据长度
L_A9C0:
                XOR     AH,AH
                INT     7EH                     ;读数字
                CMP     AL,8                    ;是删除键吗?
                JNE     L_A9D2                  ;不是
                OR      BL,BL                   ;还没有数据输入吗?
                JZ      L_A9C0                  ;是,不能删除
                CALL    L_A6F9                  ;删除
                DEC     BX                      ;长度减1
                JMP     SHORT L_A9C0
L_A9D2:
                CMP     AL,0DH                  ;是回车吗?
                JE      L_A9E0                  ;是
                MOV     BYTE PTR D_2BB1[BX],AL  ;保存数字
                CALL    L_A6F9                  ;显示
```

```
                INC     BX                      ;下一个
                JMP     SHORT L_A9C0
L_A9E0:
                OR      BX,BX                   ;没有输入行距吗?
                JZ      L_AA00                  ;是

;将行距数字串换算为数值
                MOV     CX,BX
                MOV     DL,0AH                  ;10
                XOR     BX,BX
                XOR     AX,AX

L_A9EC:
                MUL     DL                      ;剩以10
                MOV     DH,BYTE PTR D_2BB1[BX] ;取数字
                AND     DH,0FH                  ;转换为数值
                ADD     AL,DH                   ;加当前数值
                INC     BX                      ;下一个数字
                LOOP    L_A9EC

                XOR     DX,DX
                MOV     AH,4                    ;设置行距功能号
                INT     17H                     ;设置行距
L_AA00:
                MOV     AX,1000H
                INT     10H                     ;清除提示行
                RETN
CODE            ENDS
                END     L_0100
```

第五章 显示管理模块

第一节 显示管理模块程序简介

汉字显示的屏幕特性取决于显示控制卡，主要是分辨率和彩色特性。不同的显示控制卡，其显示的屏幕特性也不同。２．１３Ｈ支持多种显示方式，不同显示方式的显示管理程序也不同。

一、各种显示卡对应的显示管理程序

下表描述了２．１３Ｈ支持的显示卡的屏幕特性及对应的显示管理程序：

显示卡	名　　称	分　辨　率	显示管理程序	颜色
CGA	Color Graphics Adapter 彩色图形适配器	640 x 200	CC11, CC16 CC25	2
EGA	Enhanced Graphics Adapter 增强型图形适配器	640 x 200 640 x 350	CC11, CC16 CC25 CE21, CE25 CE26	2 16
VGA	Video Graphics Array 视频图形阵列	640 x 200 640 x 350 640 x 480	CC11, CC16 CC25 CE21, CE25 CE26 CV26	2 16 16
CGE400	Color Gaphics Enhancer 提高型彩色图形卡	640 x 400	CL25	16
HGC	Hercules Graphics Card 大力神单色图形卡	720 x 350 640 x 400	CH21 CH25	2 2

由上表可知，同一种显示卡也可以有多种显示方式，即可以使用多种显示管理程序，例如：ＶＧＡ卡可以仿真ＣＧＡ和ＥＧＡ卡，所以也可以使用所有适用于ＣＧＡ和ＥＧＡ卡的显示管理程序。

二、使用说明

显示管理程序可以重复运行多次，首次运行时将显示版本信息，以后每次运行均重新设置显示方式。例如：首次运行ＣＣ１１．ＣＯＭ，显示版本信息，屏幕总行数为１１（包括提示行）。第二次运行ＣＶ２６．ＣＯＭ，仅设置显示方式为ＶＧＡ２６行显示方式（除提示行外还有２５行可供用户使用）。

运行显示管理程序时，若光标处于闪烁状态，则先停止光标闪烁（参见ＩＮＴ１Ｃ．ＡＳＭ）。

第二节 ＶＧＡ ２６行显示 ＣＶ２６．ＣＯＭ

２．１３Ｈ虽然有很多显示管理程序，但若从用户的角度来看，其不同之处仅包括两个部分，即显示行数和屏幕同时可显示的颜色数不同。从程序角度来看，各程序不仅内部结构基本相同，甚至连变量地址、子程序地址也大都相同。因此，本书仅介绍在目前流行的ＶＧＡ卡上使用的显示管理程序ＣＶ２６．ＣＯＭ。其它显示管理程序可以参照阅读。

一、程序功能

本程序主要包括两个部分，即初始化代码和ＩＮＴ　１０Ｈ代码。

初始化代码主要完成以下工作：

１．传送ＩＮＴ　１０Ｈ代码到ＣＣＣＣ．ＣＯＭ的保留空间中。该保留的内存空间是ＣＣＣＣ．ＣＯＭ在启动时为存放ＩＮＴ　１０Ｈ代码而设置的。

２）设置ＩＮＴ　１０Ｈ。

３）设置当前显示方式为汉字显示方式。

3）若是第一次运行显示管理程序，显示版本信息“CCBIOS 2.13H”。

4）返回DOS（不驻留）。

程序运行后，即可使用INT 10H的各功能，包括汉字显示、提示行管理等功能。这为使用CCCC.COM的各种功能提供了条件，也只有在运行显示管理模块后才能使用这些功能。

在西文方式下，屏幕显示行数为25行。由于显示分辨率的限制，在部分汉字显示方式下，屏幕不能同时显示25行。但为了尽可能与西文软件保持兼容，因此，在显示管理程序中有外部屏幕和内部屏幕之称。所谓内部屏幕是指与西文显示方式下相对应的整个屏幕，共25行。外部屏幕是指当前实际屏幕上显示的内容，它仅是内部屏幕的一个窗口，其行长与内部屏幕行长相同。同样屏幕行号也有内部行号和外部行号之称，内部行号是指在内部屏幕上的行号，外部行号是指在外部屏幕上的行号。外部行号加上当前屏幕首行的内部行号即为相应的内部行号。

二、变量名表

地址	长度	意义
0049h	1	当前屏幕显示方式
004Ah	1	屏幕列数
004Ch	2	当前显示页页长，设置后没有使用
004Eh	2	当前显示页地址，设置后没有使用
0050h	2	外部光标位置
0052h	2	原光标位置
0054h	1	屏幕有否光标标志，1:有，0:无
0055h	1	可否进行光标操作(0:不允许,FF:允许)
0056h	1	提示行光标位置
0060h	2	当前光标类型
0062h	1	当前显示页号，在汉字显示方式下=0
0065h	1	显示方式寄存器，设置后没有使用
0066h	1	调色板值，用于CGA
006Ch	2	暂时保存00A0值单元
006Eh	2	暂时保存00A2值单元
0071h	2	在显示汉字时，保存汉字显示位置
0073h	1	提示行是否已显示了前半个汉字
0077h	24h	汉字或字符点阵数据存放区
009Bh	1	设置后没有使用
00A1h	1	提示行显示汉字时，保存汉字前一字节
00A2h	2	设置后没有使用
00A4h	1	外部屏幕首行的内部行号
00A5h	1	外部屏幕尾行的内部行号
00A6h	2	保存读出的光标位置，以后没有使用
00AEh	1	当光标没有建立时，控制光标操作子程序执行 <=1不能执行．其它:奇数执行．偶数下次执行
00AFh	7D0h	屏幕字符缓冲区
087Fh	7D0h	屏幕字符属性缓冲区
104Fh	7D0h	屏幕字符类型缓冲区
181Fh	1	调色板值
1820h	2	AA55h,运行了CCCC.COM的标志字节
1822h	2	18E0h,INT 10H功能0
1824h	2	1A48h,INT 10H功能1
1826h	2	1A69h,INT 10H功能2
1828h	2	1C65h,INT 10H功能3
182Ah	2	24BFh,INT 10H功能4
182Ch	2	1C9Eh,INT 10H功能5
182Eh	2	1D49h,INT 10H功能6
1830h	2	1DF1h,INT 10H功能7
1832h	2	1E30h,INT 10H功能8
1834h	2	1E76h,INT 10H功能9
1836h	2	1EB5h,INT 10H功能0Ah

1838h	2	1CC2h,INT 10H功能0Bh
183Ah	2	1EF7h,INT 10H功能0Ch
183Ch	2	1EE6h,INT 10H功能0Dh
183Eh	2	2408h,INT 10H功能0Eh
1840h	2	1CE6h,INT 10H功能0Fh
1842h	2	24C4h,INT 10H功能10h
1844h	2	1A3Fh,INT 10H功能11h
1846h	2	25CCh,INT 10H功能12h
1848h	2	25DFh,INT 10H功能13h
1899h	10h	显示方式寄存器表
18CAh	1	当前显示前景颜色
18CBh	1	当前显示背景色

三、程序清单

```
;CV26.ASM
CODE            SEGMENT
                ASSUME  CS:CODE,DS:CODE
                ORG     100H

L_0100:
;2.13H 使用 INT1C.COM 程序使光标闪烁. INT1C.COM 本身并非驻留程序, 它是将 INT 1CH 的
;代码(包含在 INT1C.COM 中, 参见 INT1C.ASM)传送到显示模块的空闲区域, 然后, 修改
;INT 1CH 中断向量. 因此, 在运行显示模块时, 必须先检查当前光标是否闪烁, 若光标处
;于闪烁状态, 则必须停止光标闪烁(即恢复原 INT 1CH 中断向量), 否则, 新的显示模块程
;序(INT 10H)将破坏原有 INT 1CH, 当时钟硬中断(18.2 次/秒)发生时, 将引起死机.
                MOV     AX,351CH
                INT     21H                     ;取当前INT 1CH中断向量
                CMP     BX,1923H
;BX=1923H 表示已运行过 INT1C.COM, 也即当前光标处于闪烁状态

                JNE     L_0117                  ;没有运行过 INT1C.COM

;恢复原 INT 1CH
                LDS     DX,ES:[1968H]           ;取原INT 1CH中断向量->DS:DX
                MOV     AX,251CH
                INT     21H                     ;恢复INT 1CH
                PUSH    CS
                POP     DS                      ;恢复DS

;由于显示模块(如:CV26.COM)本身也不是驻留程序, 它们仅将 INT 10H 代码传送到 CCCC.COM
;为 INT 10H 保留的空间(从 1822H 开始, 长度 F70H 字节). 因此, 这里首先判断是否已运行
;了 CCCC.COM, 若没有运行过则也不能运行显示管理模块.
L_0117:
                MOV     AX,351FH
                INT     21H
;取 INT 1FH 中断向量. 若运行了 CCCC.COM, 那么其段址与 CCCC.COM 的段址相同

                CMP     ES:[1820H],0AA55H       ;运行了 CCCC.COM 吗? ;
[1820H]=0AA55H 是 CCCC.COM 设置的内部标志.
                JE      L_0128                  ;是
                JMP     0                       ;没有,则转0000
;在 PSP 位移 0 处是一条 INT 20H 指令, JMP 0 相当于执行 INT 20H, 该中断的作用是结束程序,
;返回 DOS
```

```
L_0128:
                CMP     BYTE PTR ES:[184AH],0    ;是第一次运行显示模块程序吗?
;因为若运行过显示模块程序, 则该处应是指令 STI, 其值为 0FBH<>0
                JNE     L_0143                   ;不是

;第一次运行显示模块程序, 保存系统启动时要显示的版本号的图形数据
                MOV     SI,OFFSET D_0400         ;图形数据首址
                MOV     CX,1040H                 ;长度
L_0136:
                MOV     AL,ES:[SI]               ;取数据
                MOV     DS:2C00H[SI],AL          ;保存数据
;实际数据保存地址从 3000H 开始

                INC     SI
                LOOP    L_0136

                MOV     BP,0FFFFH                ;设置首次运行标志

L_0143:
;将 INT 10H 代码传送到 CCCC.COM 的保留空间中
                MOV     SI,DS:[1822H]            ;首址
                MOV     DI,SI
                MOV     CX,0F70H                 ;长度
                REP     MOVSB                    ;传送

                PUSH    ES
                POP     DS                       ;DS=CCCC.COM程序的段地址

;下面设置 INT 10H 中断向量. 但若系统运行过 INT10F.COM, 则不能重新设置, 否则将不能
;使用由 INT10F.COM 提供的特殊显示功能, 同时, 由于各显示模块中显示中断程序的入口
;地址都相同, 故不重新设置 INT 10H 中断向量也可以实现显示方式的切换.
                MOV     AX,3510H
                INT     21H                      ;取INT 10H中断向量
                CMP     BX,18DH                  ;运行过INT10F.COM吗?
;因为 INT10F.COM 运行后,INT 10H 的入口地址为 018DH(见 INT10F.ASM)

                JE      L_0162                   ;是,不能重新设置INT 10H

;重新设置INT 10H
                MOV     DX,L_184A                ;取新INT 10H入口地址
                MOV     AX,2510H
                INT     21H                      ;设置INT 10H中断向量

L_0162:
                MOV     BYTE PTR DS:[9830H],0A4H ;修改CCCC.COM中的程序
;原指令是 ADD AH,[0A5H], 其中变量 0A5H 是屏幕当前首行的内部行号, 在 CV26.COM 中为
;A4H, 故改为 ADD AH,[A4H].

                MOV     BYTE PTR DS:[984DH],0AFH ;修改 CCCC.COM 中的程序
;原指令是 MOV AL,0B0H[BX], 0B0H 是当前屏幕 ASCII 字符表的首址, 在 CV26.COM 中为 0AFH,
;故改为 MOV AL,0AFH[BX], 这些程序可能是编者想增加程序的解读性而故意设置的.

                MOV     AX,6
                INT     10H                      ;设置成汉字显示方式
```

```
                MOV     AX,1301H
                INT     10H                     ;建立光标

                CMP     BP,0FFFFH               ;是首次运行显示管理程序吗?
                JNE     L_01B4                  ;不是

;首次运行显示模块, 要显示版本信息(CCBIOS 2.13H...)
                PUSH    CS
                POP     DS                      ;恢复DS
                MOV     DX,3C4H                 ;索引寄存器端口号
                MOV     AX,0A02H
;AL=选择寄存器号, AH=向所选择的寄存器输出的数据. AL 输出至 3C4H 端口, 而 AH 输出至
;3C5H 端口. AL=2 表示选择彩色页面写允许寄存器, AH=0AH 表示仅允许向页面 1 和页面 3 写
;数据, 即设置 10 号颜色
                OUT     DX,AX                   ;设置显示颜色

                MOV     SI,3000H                ;数据首址
                XOR     DI,DI                   ;DI=视频区中位移=0, 表示从屏
                                                ;幕左上角开始显示
                MOV     AX,0A000H
                MOV     ES,AX                   ;ES=视频区段址
;EGA和VGA的视频区位于段0A000H处
                MOV     CH,1AH                  ;共循环1AH次, 每次显示两条扫
                                                ;描线
L_0190:
                MOV     CL,50H                  ;每条扫描线共640点(50H个字节)
L_0192:
;每次分别在连续两条扫描线上各显示一字节
                LODSB
                STOSB                           ;显示一个字节(8点)
                MOV     AL,D_081F[SI]           ;取下一扫描线该位置的数据
                MOV     ES:[DI+4FH],AL          ;在下一扫描线上显示一字节
;显示数据区包括两个部分, 每个部分为 820H 个字节. 前一部分是偶数条扫描线的数据,
;后一部分是奇数条扫描线的数据. 这里每次向两条扫描线各显示一字节. 第一字节显示
;好后 DI 已加 1, 故下一条扫描线当前位置地址应为 DI+4FH. 数据按奇偶方式存放主要是由
;于在 CGA 系列显示上, 其视频区是按奇偶地址排列的.

                DEC     CL
                JNZ     L_0192                  ;显示二条扫描线
;此时 DI 已指向第二条扫描线首址, 由于第二条扫描线已显示过, 所以这里 DI 还要加 50H
                ADD     DI,50H                  ;DI=第三条扫描线首址
                DEC     CH
                JNZ     L_0190                  ;共循环1AH次, 即显示52条扫描
                                                ;线
                MOV     AX,0F02H                ;允许向所有页面写入数据
                OUT     DX,AX                   ;输出数据(此时DX仍为3C4H)
;VGA 的页面选择寄存器通常情况下均应设置为可以向所有页面写入数据

;由于显示了版本信息, 光标应位于第 4 行了, 下面设置光标位置
                XOR     BH,BH                   ;0页面
                MOV     DX,401H
                MOV     AH,2
                INT     10H                     ;设置光标

;2.13H 有多个显示管理程序, 可支持多种屏幕行数. 但由于使用了 ANSI.SYS, 缺省情况下
```

```
;均假定屏幕行数为25. 当当前显示方式下, 屏幕显示行数小于25行(不包括提示行)时,
;当前光标进入最后行后, 会出现屏幕不翻滚的现象. 因此下面修改ANSI.SYS中使用的行
;数, 以使DOS正常工作.
L_01B4:
                MOV     BP,99F0H            ;子程序地址
                MOV     DL,19H              ;CV26.COM支持的行数
                MOV     AH,7
                INT     16H                 ;执行CCCC.COM中的子程序, 修改
                                            ;ANSI.SYS的屏幕行数
                MOV     AX,4C00H
                INT     21H                 ;程序结束

                ORG     1822H
;为了与CCCC.COM中的地址统一, 这里空出许多空间不用

;下面是传送到CCCC.COM中的程序和数据

;显示模块各子模块入口地址
D_1822          DW      OFFSET L_18E0       ;功能0, 设置显示方式
D_1824          DW      OFFSET L_1A48       ;功能1, 设置光标类型
D_1826          DW      OFFSET L_1A69       ;功能2, 设置光标位置
D_1828          DW      OFFSET L_1C65       ;功能3, 读取光标位置的类型
D_182A          DW      OFFSET L_24BF       ;功能4, 读取光笔位置
D_182C          DW      OFFSET L_1C9E       ;功能5, 选择当前显示页
D_182E          DW      OFFSET L_1D49       ;功能6, 指定窗口上滚
D_1830          DW      OFFSET L_1DF1       ;功能7, 指定窗口下滚
D_1832          DW      OFFSET L_1E30       ;功能8, 读取当前光标位置字符
                                            ;和属性
D_1834          DW      OFFSET L_1E76       ;功能9, 在当前光标位置写字符
                                            ;和属性
D_1836          DW      OFFSET L_1EB5       ;功能0AH, 在当前光标位置写字
                                            ;符
D_1838          DW      OFFSET L_1CC2       ;功能0BH, 设置调色板(仅CGA有
                                            ;效)
D_183A          DW      OFFSET L_1EF7       ;功能0CH, 显示点
D_183C          DW      OFFSET L_1EE6       ;功能0DH, 读取点
D_183E          DW      OFFSET L_2408       ;功能0EH, 按TTY方式显示字符
D_1840          DW      OFFSET L_1CE6       ;功能0FH, 读取当前显示方式
D_1842          DW      OFFSET L_24C4       ;功能10H, 提示行操作
D_1844          DW      OFFSET L_1A3F       ;功能11H, 保留
D_1846          DW      OFFSET L_25CC       ;功能12H, 读取汉字的点阵数据
D_1848          DW      OFFSET L_25DF       ;功能13H, 建立或取消光标

;INT 10H
;功能0, 设置显示方式
;输入: AH=0, AL=显示方式
;输出: 无

;功能1, 设置光标类型, 仅西文方式下有效
;输入: AH=1, CH=光标起始扫描线, CL=光标结束扫描线;输出: 无
;
;功能2, 设置光标位置
;输入: AH=2, BH=页号(仅西文方式下有效), DH=字符行号, DL=字符列号
;
;功能3, 读取光标位置和光标类型
```

```
;输入: AH=3, BH=页号(仅西文方式下有效), DH=当前光标行号, DL=当前光标列号
;       CX=光标类型
;
;功能 4, 读出光标位置(仅西文方式下有效)
;输入: AH=4
;输出: AH=0 光笔开关未打开, =1 光笔有效
;       DX=光笔字符行列位置, CX=光栅线, BX=象素点列值
;
;功能 5, 选择当前显示页
;输入: AH=5, AL=页号
;输出: 无
;
;功能 6, 指定窗口上滚
;输入: AH=6, AL=滚动行数(若为 0 表示清除整个窗口)
;       CX=窗口左上角坐标, DX=窗口右下角坐标
;       BH=空行属性
;
;功能 7, 指定窗口下滚
;输入: AH=7, AL=滚动行数(若为 0 表示清除整个窗口)
;       CX=窗口左上角坐标, DX=窗口右下角坐标
;       BH=空行属性
;
;功能 8, 读取当前光标位置字符和属性
;输入: AH=8, BH=页号(仅西文方式下有效)
;输出: AL=当前光标位置字符 ASCII 码, AH=当前光标位置字符属性
;
;功能 9, 在当前光标位置写字符和属性
;输入: AH=9, BH=页号(仅西文方式下有效)
;       AL=字符 ASCII 码, CL=重复次数(汉字只能显示一个)
;
;功能 0AH, 在当前光标位置写字符, 使用当前位置原有属性
;输入: AH=0AH, BH=页号(仅西文方式下有效)
;       AL=字符 ASCII 码, CL=重复次数(汉字只能显示一个)
;
;功能 0BH, 设置调色板(仅 CGA 有效)
;输入: AH=0BH, BH=新颜色值, BL=调色板寄存器
;
;功能 0CH, 显示点
;输入: AH=0CH, CX=点的列位置, DX=点的行位置
;输出: 无
;
;功能 0DH, 读取点
;输入: AH=0DH, CX=点的列位置, DX=点的行位置
;输出: AL=点的值
;
;功能 0EH, TTY 方式的显示字符
;输入: AH=0EH, AL=字符 ASCII 码
;输出: 无
; ;功能 0FH, 取得当前显示方式
;输入: AH=0FH
;输出: AL=当前显示方式, AH=每行字符个数, BH=当前显示页号
;
;功能 10H, 提示行操作
;输入: AL=0, 清提示行, 提示行光标移至行首
;       AL=1, 在提示行当前位置显示字符, DL=字符 ASCII 码
```

```
;       AL=2, 设置提示行光标位置, DL=光标位置
;       AL=3, 在提示行模仿 TTY 显示字符, DL=字符
;输出: 无
;
;功能 11H, 保留
;
;功能 12H, 读取汉字点阵数据
;输入: AH=12H, DX=汉字
;输出: BP:DX 汉字的点阵数据地址
;
;功能 13H, 建立或取消光标
;输入: AH=13H, AL=0 关闭光标, AL=1 打开光标
;输出: 无
L_184A:
                STI                             ;开中断
                CLD                             ;设置递增的字符串操作方式

                PUSH    ES
                PUSH    DS
                PUSH    DX
                PUSH    CX
                PUSH    BX
                PUSH    SI
                PUSH    DI
                PUSH    BP                      ;保存寄存器

                CMP     AH,14H                  ;有效的功能号吗?
                JB      L_185C                  ;是
                JMP     L_1A3F                  ;无效功能号, 直接返回

L_185C:
                PUSH    AX                      ;保存AX
                MOV     AL,AH
                MOV     AH,0                    ;AX=功能号
                SHL     AX,1                    ;AX=AXx2
                MOV     SI,AX
                MOV     AX,CS
                MOV     DS,AX                   ;DS=CS
                MOV     AX,0A000H
                MOV     ES,AX                   ;ES=0A000H, 即视频区段址
                POP     AX                      ;恢复AX
                MOV     AH,DS:[49H]             ;AH=当前显示方式
                JMP     WORD PTR D_1822[SI]     ;转相应功能程序执行

                DB      33 DUP (0),28H          ;没有用

;下面是当前方式选择寄存器表, 这些数据在本程序中没有起任何作用
D_1899          DB      28H,50H,50H,28H,28H
                DB      11 DUP (50H)
                DB      1EH,2DH,0AH,7FH,06H,64H
                DB      70H,53H,01H,55H,07H,4EH
                DB      00H,00H,00H,61H,50H,52H
                DB      0FH,19H,06H,19H,19H,02H
                DB      0DH,0BH,0CH,00H,00H,00H
                DB      00H,00H,80H
```

```
D_18CA          DB      0AH                     ;当前显示前景色
D_18CB          DB      10H                     ;当前显示背景色

;以下数据为 CGA 显示方式使用数据，在 CV26.COM 中没有使用
                DB      00H,40H,00H,40H,28H,28H
                DB      50H,50H,28H,28H,50H,50H
                DB      2CH,28H,2DH,29H,2AH,2EH
                DB      1EH,29H

;功能 0，设置屏幕显示方式
L_18E0:
                MOV     DS:[49H],AL             ;保存显示方式

                CMP     AL,4                    ;是设置为西文显示方式吗?
                JB      L_18E9                  ;是
;当显示方式号大于 3 时，均设置为汉字显示方式. 这些显示方式都是 640x480 的 VGA
;图形显示方式，调用原 INT 10H 时，显示方式一例要改为 12H.
                MOV     AL,12H                  ;AL=12H，即真正的图形方式号
L_18E9:
                XOR     AH,AH
                INT     78H                     ;调用原INT 10H设置显示方式
                JMP     L_1970

                DB      126 DUP (0)             ;空闲区
;该空闲区用于存放 INT 1CH，它由 INT1C.COM 设置(参见 INT1C.ASM)

;变量初始化
L_1970:
                XOR     DI,DI                   ;DI=0
                MOV     DS:[4EH],DI             ;置当前显示页在视频区中的相对
                                                ;位移
                MOV     BYTE PTR DS:[62H],0     ;设置当前页号，在汉字显示方式
                                                ;下页号均为0，而且只有一个可
                                                ;用页
                NOP
                MOV     WORD PTR DS:[60H],67H   ;设置光标类型，在汉字显示方式
                                                ;下，光标类型不能设置，因此，
                                                ;该单元没有用
                MOV     AL,DS:[49H]             ;AL=要设置的显示方式
                XOR     AH,AH
                MOV     SI,AX
                NOP
                NOP
                NOP
                NOP
                MOV     AL,D_1899[SI]           ;取当前方式选择寄存器值
                NOP                             ;设置当前方式选择寄存器,在本
                MOV     DS:[65H],AL             ;程序中没有使用
                PUSH    DS                      ;保存DS
                MOV     DI,40H
                MOV     DS,DI                   ;DS=40H,指向ROM BIOS数据段
                MOV     DI,65H
                MOV     [DI],AL                 ;修改BIOS数据区中的方式选择寄
                                                ;存器单元值
```

```
        POP     DS                      ;恢复DS
        NOP
        NOP
        NOP
        MOV     AL,50H
        MOV     DS:[4AH],AL             ;屏幕可显示列数
        NOP
        NOP
        AND     SI,0EH
        NOP
        NOP
        MOV     CX,8000H                ;CX=当前显示页长度
        NOP
        MOV     DS:[4CH],CX             ;设置当前显示页长度
        MOV     DI,50H
        XOR     AX,AX
        MOV     [DI],AX                 ;设置当前外部光标位置=(0,0)
        MOV     [DI+5],AL               ;禁止执行光标操作子程序
        PUSH    DS                      ;保存DS
        MOV     AX,40H
        MOV     DS,AX                   ;DS=40H,指向ROM BIOS数据段
        MOV     WORD PTR [DI],0         ;修改BIOS数据区中当前光标位置
                                        ;单元值
        POP     DS                      ;恢复DS
        NOP
        MOV     AL,0AH                  ;调色板值
        CMP     BYTE PTR DS:[49H],10H   ;是设置为10H号显示方式吗?
        JNE     L_19DA                  ;不是
        MOV     AL,[181FH]              ;=0,
L_19DA:
        NOP
        MOV     DS:[66H],AL             ;置调色板值，在CV26.COM中没有
                                        ;实质作用
        CMP     BYTE PTR DS:[49H],4     ;是设置为西文方式吗?
        JB      L_1A3F                  ;是，转中断返回

;设置汉字显示方式，还要初始化其它一些数据
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOV     AX,0FFFFH
        MOV     DS:[71H],AX             ;71H在显示汉字时存放显示坐标
        MOV     DS:[0ACH],AX            ;以后没有使用
        NOT     AX
        MOV     DS:[0A6H],AX            ;提示行光标位置=0
        AND     BYTE PTR DS:[73H],30H   ;提示行已显示汉字前半字节标志
        MOV     CX,12H                  ;18字节
        PUSH    DS
        POP     ES                      ;ES=DS
        MOV     DI,77H                  ;当前显示字符或汉字的点阵数据
```

```
                                        ;保存区
        REP     STOSW                   ;全部清0
        MOV     DS:[9BH],AL             ;以后没有使用
        MOV     CX,4
        MOV     DI,0A0H
        REP     STOSB                   ;0A0H-0A3H设置为0
        MOV     BYTE PTR DS:[0A4H],0    ;设置外部屏幕首行内部行号
        MOV     BYTE PTR DS:[0A5H],18H  ;设置外部屏幕尾行内部行号
;在 CV26.COM 中, 外部屏幕实际上已包括了整个内部屏幕

        MOV     DS:[0AAH],AX            ;以后没有使用
        MOV     DS:[0A8H],AX            ;以后没有使用

;初始化屏幕缓冲区
        MOV     CX,7D0H                 ;7D0H=2000, 整个显示页字符数
        MOV     DI,0AFH
        REP     STOSB                   ;初始化屏幕字符缓冲区
        MOV     CX,7D0H
        MOV     DI,104FH
        REP     STOSB                   ;初始化屏幕字符类型缓冲区
        MOV     CX,7D0H
        MOV     DI,087FH
        MOV     AL,3
        REP     STOSB                   ;初始化屏幕字符属性缓冲区

;中断返回
L_1A3F:
        POP     BP
        POP     DI
        POP     SI
        POP     BX
L_1A43:
        POP     CX
        POP     DX
        POP     DS
        POP     ES                      ;恢复寄存器

        IRET                            ;中断返回

;功能 1, 设置光标类型
L_1A48:
        MOV     DS:[60H],CX             ;保存光标类型
        MOV     AH,1
        INT     78H                     ;调用原INT 10H, 仅西文方式有
                                        ;效
        JMP     SHORT L_1A3F
        NOP
        DB      22 DUP (0)              ;没有用

;功能 2, 设置光标位置, 若光标位置在内部屏幕外, 则不设置. DX=内部屏幕坐标
L_1A69:
        CMP     DH,19H                  ;行数>25吗?
        JAE     L_1A3F                  ;是, 中断返回
        CMP     DL,DS:[4AH]             ;列数>屏幕最大列数吗?
        JAE     L_1A3F                  ;是, 中断返回
```

```
;修改 ROM BIOS 数据区的光标位置单元
                PUSH    DS                      ;保存DS
                MOV     CX,40H
                MOV     DS,CX                   ;DS=40H,指向ROM BIOS数据段
                MOV     DS:[50H],DX             ;修改光标位置
                POP     DS                      ;恢复DS

                CMP     AH,4                    ;是西文方式吗?
                JB      L_1A87                  ;是
                JMP     L_1B33

;西文方式, 只要直接调用原 INT 10H 即可
L_1A87:
                MOV     AH,2
                INT     78H                     ;调用原INT 10H
                JMP     SHORT L_1A3F            ;中断返回

                DB      32 DUP (0)              ;没有用

;子程序: 光标操作, 其光标位置均指外部屏幕上的光标位置
;输   入: BP=0, 消除原光标, 设置新光标
;         BP=1, 消除原光标
;输   出: 无
L_1AAD:
                TEST    BYTE PTR DS:[55H],0FFH  ;允许执行光标操作子程序吗?
                JZ      L_1B32                  ;不能, 直接返回

                TEST    BYTE PTR DS:[54H],0FFH  ;当前屏幕上有光标吗?
                JNZ     L_1AD1                  ;有

;当前屏幕上没有光标时, 若[0AEH]的值为大于 1 的奇数, 也要执行. 若[0AEH]的值为大
;于 1 的偶数, 不执行, 但将[0AEH]设置为大于 1 的奇数, 当前再次调用时可以执行. 当
;[0AEH]的值小于 2 时, 都不执行. 这段程序对 CV26.COM 没有实质性的作用.
                TEST    BYTE PTR DS:[0AEH],0FEH ;[0AEH]<2吗?
                JZ      L_1B32                  ;是, 不执行
;[0AEH]>1
                TEST    BYTE PTR DS:[0AEH],1    ;[0AEH]是奇数吗?
                JNZ     L_1AD1                  ;是, 执行
                OR      BYTE PTR DS:[0AEH],1    ;置[0AEH]为偶数
                JMP     SHORT L_1B32            ;不执行
                NOP

;执行光标操作
L_1AD1:
                MOV     AX,DS:[50H]             ;取新光标位置
                PUSH    AX                      ;保存新光标位置
                MOV     AX,DS:[52H]             ;取原光标
                MOV     DS:[50H],AX             ;暂时改为当前光标
                CALL    L_23C6                  ;AX=当前光标位置第一条扫描线
                                                ;地址

                JMP     SHORT L_1AE7
                NOP
                DB      6 DUP (0)               ;没有用
```

```
L_1AE7:
                ADD     AX,500H                 ;AX=当前光标位置最后一条扫描
                                                ;线地址，即显示光标地址
                MOV     SI,AX
                TEST    BYTE PTR ES:[SI],0FFH   ;有光标吗?
                JZ      L_1B02                  ;没有，不用消除
                XOR     BYTE PTR ES:[SI],0FFH   ;消除原光标
                JMP     SHORT L_1B02

                DB      10 DUP (0)              ;没有用

L_1B02:
                POP     WORD PTR DS:[50H]       ;恢复新光标位置

                CMP     BP,1                    ;要设置新光标吗?
                JE      L_1B2C                  ;不要

                CALL    L_23C6                  ;AX=新光标位置第一条扫描线地
                                                ;址

                JMP     SHORT L_1B17
                NOP
                DB      6 DUP (0)               ;没有用

L_1B17:
                ADD     AX,500H                 ;AX=当前光标位置最后一条扫描
                                                ;线地址，即显示光标地址
                MOV     SI,AX
                XOR     BYTE PTR ES:[SI],0FFH   ;设置光标
                JMP     SHORT L_1B2C

                DB      10 DUP (0)              ;没有用

L_1B2C:
                MOV     AX,DS:[50H]             ;取新光标位置
                MOV     DS:[52H],AX             ;保存于[52H]中

L_1B32:
                RETN

;设置光标位置
L_1B33:
                MOV     BP,1
                CALL    L_1AAD                  ;消除原光标
                CMP     DH,18H                  ;是将光标设置到提示行吗?
                JNE     L_1B41                  ;不是
将光标设置于提示行，因为提示行肯定在当前屏幕中，所以，不用判断是否需要上滚或下滚
                JMP     SHORT L_1B55
                NOP

L_1B41:
                XOR     BH,BH                   ;BH=0
                MOV     AL,DS:[0A4H]            ;取外部屏幕起始内部行号
                CMP     DH,AL                   ;要将光标设置在当前屏幕上方吗?
```

```
                JB      L_1B66                  ;是
```
由于外部屏幕仅是内部屏幕的一个窗口，当前要设置的光标在外部屏幕之外时，首先要滚动外部屏幕，使得要设置的位置在外部屏幕内

```
                MOV     AL,DS:[0A5H]            ;取外部屏幕结束内部行号
                CMP     AL,DH                   ;要将光标设置在当前屏幕下方吗?
                JB      L_1BAC                  ;是

要设置的光标位置在外部屏幕内，不用翻滚
                SUB     DH,DS:[0A4H]            ;DH=外部行号，至此，DX已是外部
                                                ;坐标
L_1B55:
                MOV     DS:[50H],DX             ;保存光标位置
                MOV     BYTE PTR DS:[55H],0FFH  ;允许进行光标操作
                XOR     BP,BP
                CALL    L_1AAD                  ;设置新光标
                JMP     L_1A3F

;新光标位置在外部屏幕上方，首先要执行下滚操作，使得光标位置在外部屏幕内
L_1B66:
                PUSH    DX                      ;保存光标位置
                SUB     AL,DH                   ;AL=要翻滚的行数
                CMP     AL,18H                  ;大于外部屏幕行数吗?(不包括提
                                                ;行)
                JB      L_1B6F                  ;不是
                MOV     AL,18H                  ;翻滚行数最多为外部屏幕行数
L_1B6F:
;定义翻滚窗口为整个内部屏幕
                XOR     CX,CX                   ;窗口左上角坐标=内部屏幕左上
                                                ;角坐标
                MOV     DH,18H
                MOV     DL,DS:[4AH]
                DEC     DL                      ;窗口右下角坐标=内部屏幕右上
                                                ;角坐标

                PUSH    AX                      ;保存AX
                CALL    L_2149                  ;调用窗口下滚子程序
                POP     BX                      ;原AX->BX

;外部屏幕下滚后，翻滚出来的行都是空行.但实际上，这些行在内部屏幕中都是有内容
;的.所以下面还要显示这些空行
                MOV     BH,BL                   ;翻滚出来的空白行数
                DEC     BH                      ;BH=要显示的行数
                POP     DX                      ;新光标的坐标
                PUSH    DX                      ;再保存
                MOV     AX,DX
                MOV     DS:[0A4H],AH            ;设置外部屏幕首行内部行号
                ADD     AH,18H                  ;加外部屏幕行数(24)=外部屏幕
                                                ;尾行的内部行号
;实际上，在VGA显示方式中，由于显示分辨高，外部屏幕有26行，包括了整个内部屏幕.
;故不存在上下翻滚的问题.

                MOV     DS:[0A5H],AH            ;设置外部屏幕尾行内部行号
                ADD     DH,BH                   ;加显示行数=要显示的行号(从最
                                                ;下面一行开始显示)
```

```
L_1B93:
                XOR     DL,DL                   ;从第0列开始显示
                MOV     DI,DX                   ;DI=显示内容首坐标
                CALL    L_1BF7                  ;显示一行内容
                CMP     BH,0                    ;已显示完毕
                JE      L_1BA7                  ;是
                DEC     BH                      ;计数减1
                MOV     DX,DI
                DEC     DH                      ;再显示上面一行内容
                JMP     SHORT L_1B93

L_1BA7:
                POP     DX                      ;恢复新光标位置
                XOR     DH,DH                   ;新光标在外部屏幕的首行中,
                                                ;故其外部行号应为1
                JMP     SHORT L_1B55

;新光标位置在外部屏幕下方，首先要执行上滚操作，使得光标位置在外部屏幕内
L_1BAC:
                PUSH    DX                      ;保存光标位置
                SUB     DH,AL
                MOV     AL,DH                   ;AL=要翻滚的行数
                CMP     AL,18H                  ;大于外部屏幕行数吗?(不包括提
                                                ;行)
                JB      L_1BB7                  ;不是
                MOV     AL,18H                  ;翻滚行数最多为外部屏幕行数
L_1BB7:
;定义翻滚窗口为整个内部屏幕
                XOR     CX,CX                   ;窗口左上角坐标=内部屏幕左上
                                                ;角坐标
                MOV     DH,18H
                MOV     DL,DS:[4AH]
                DEC     DL                      ;窗口右下角坐标=内部屏幕右上
                                                ;角坐标
                PUSH    AX                      ;保存AX
                CALL    L_2057                  ;调用窗口上滚子程序
                POP     BX                      ;BX=原AX

;外部屏幕上滚后，翻滚出来的行都是空行. 但实际上，这些行在内部屏幕中都是有内容
;的，所以下面还要显示这些空行
                DEC     BL                      ;BL=要显示的行数
                MOV     BH,18H
                SUB     BH,BL                   ;从最下面一行开始显示
                POP     DX                      ;恢复光标位置
                MOV     AX,DX
                PUSH    DX                      ;再保存光标位置
                MOV     DS:[0A5H],AH            ;设置外部屏幕尾行的内部行号
                SUB     AH,18H                  ;减外部屏幕行数
                MOV     DS:[0A4H],AH            ;设置外部屏幕首行的内部行号
                SUB     DH,BL                   ;起始显示行号
L_1BDD:
                XOR     DL,DL
                MOV     DI,DX                   ;DI=显示内容首坐标
                CALL    L_1BF7                  ;显示一行内容
                CMP     BH,18H                  ;显示完毕
```

```
                JE      L_1BF1                  ;是
                INC     BH                      ;计数加1
                MOV     DX,DI
                INC     DH                      ;下一行坐标
                JMP     SHORT L_1BDD

L_1BF1:
                POP     DX                      ;恢复新光标坐标
                MOV     DH,18H                  ;新光标在外部屏幕的尾行中,
                                                ;故其外部行号应为18H
                JMP     L_1B55

;子程序: 显示一行内容, 本子程序调用其它功能显示字符, 故必须保存一部分变量
;输   入: DI=显示内容首坐标
;         BH=外部屏幕行号
;输   出: 无
L_1BF7:
;保存要被破坏的变量
                MOV     AX,DS:[55H]             ;取光标操作允许标志
                PUSH    AX                      ;保存
                MOV     BYTE PTR DS:[55H],0     ;不允许执行光标操作子程序
                MOV     AX,DS:[0A0H]
                MOV     DS:[6CH],AX
                MOV     AX,DS:[0A2H]
                MOV     DS:[6EH],AX             ;保存0A0H-0A2H的值
                MOV     AX,DS:[71H]
                PUSH    AX                      ;保存71H的值

                MOV     CX,1                    ;每次显示一个字符
                XOR     BL,BL                   ;从当前行第一列开始显示
                MOV     DS:[50H],BX             ;暂时设置当前光标位置, 实际上
                                                ;在显示时不显示光标的
L_1C19:
                MOV     AX,DI                   ;AX=显示内容首坐标
                CALL    L_1D11                  ;取坐标AX屏幕缓冲区中的相对位
                                                ;移(其值是按每个字符两字节计算
                                                ;的)
                SAR     AX,1                    ;AX=在屏幕缓冲区中的相对位移
                MOV     SI,AX
                MOV     AL,DS:0AFH[SI]          ;取字符
                MOV     BL,D_087F[SI]           ;取属性
                MOV     AH,0AH
                INT     10H                     ;调用功能10H显示一个字符
                INC     DI                      ;内容区坐标加1
                MOV     AX,DS:[50H]             ;AX=当前显示坐标
                INC     AL                      ;列号加1
                CMP     AL,DS:[4AH]             ;已显示完毕吗?
                JB      L_1C4F                  ;没有

;恢复保存的变量
                POP     AX
                MOV     DS:[71H],AX
                POP     AX
                MOV     DS:[55H],AX
                MOV     AX,DS:[6CH]
```

```
                MOV     DS:[0A0H],AX
                MOV     AX,DS:[6EH]
                MOV     DS:[0A2H],AX

                RETN

L_1C4F:
                MOV     DS:[50H],AX             ;修改显示位置
                JMP     SHORT L_1C19            ;显示下一字符

;下面这段程序没有使用
                CALL    1CFAH                   ;该子程序也没有内容
                MOV     CX,AX
                ADD     CX,DS:[4EH]
                SAR     CX,1
                MOV     AH,0EH
                CALL    L_1A53
                RETN

;功能 3，读当前光标位置和光标类型
L_1C65:
                CMP     AH,4                    ;在西文显示方式下吗?
                JB      L_1C91                  ;是，直接调用原INT 10H即可

                MOV     DX,DS:[50H]             ;取外部屏幕光标位置
                CMP     DH,17H                  ;是在提示行吗?
                JA      L_1C77                  ;是，不用换算成内部行号
                ADD     DH,DS:[0A4H]            ;加屏幕首行内部行号，DH=光标
                                                ;位置的内部行号
L_1C77:
                MOV     DS:[0A6H],DX            ;保存光标位置，以后没有使用
                MOV     CX,DS:[60H]             ;CX=当前光标类型
                JMP     SHORT L_1C95

                DB      16 DUP (0)              ;没有用

;功能 3，西文处理
L_1C91:
                MOV     AH,3                    ;读当前光标位置和光标类型
                INT     78H                     ;调用原INT 10H

;中断返回点，CX 和 DX 是返回的数据，所以这里不能恢复 CX 和 DX 的值
L_1C95:         POP     BP
                POP     DI
                POP     SI
                POP     BX
                POP     AX                      ;不恢复CX
                POP     AX                      ;不恢复DX
                POP     DS
                POP     ES                      ;恢复寄存器

                IRET                            ;中断返回

;功能 5，选择当前显示页，仅西文方式下有效，直接调用原 INT 10H
L_1C9E:
```

```
                MOV     DS:[62H],AL             ;保存当前显示页
                MOV     AH,5
                INT     78H                     ;调用原INT 10H
                JMP     L_1A3F

                DB      26 DUP (0)              ;没有用

;功能 0BH, 设置调色板
L_1CC2:
;以下均为设置 CGA 的调用板, 在 CV26.COM 中不适用
;正确的设置方法为:
;               mov     bh,bl                   ;新颜色值
;               mov     bl,7                    ;通常在DOS下显示的字符均用7号
;                                               ;调色板寄存器
;               mov     ax,1000h                ;设置EGA或VGA的调色板值, AL=0
;                                               ;表示设置单个寄存器值
;               int     78h                     ;调用原INT 10H
;               jmp     l_1A3F

                MOV     AL,DS:[66H]
                OR      BH,BH
                JNZ     L_1CDA
                AND     AL,0E0H
                AND     BL,1FH
                OR      AL,BL
                MOV     DS:[66H],AL
L_1CD3:
                MOV     CS:D_18CA,AL
                JMP     L_1A3F
L_1CDA:
                AND     AL,0DFH
                SHR     BL,1
                JNC     L_1CD3
                OR      AL,20H
                JMP     SHORT L_1CD3

                DB      0, 0

;功能 0FH, 取得当前显示方式
L_1CE6:
                MOV     AX,40H
                MOV     ES,AX                   ;ES指向ROM BIOS数据段
                MOV     AX,ES:[49H]             ;取当前显示方式->AL
                MOV     BH,DS:[62H]             ;取当前显示页号->BH
                POP     BP
                POP     DI
                POP     SI
                POP     CX                      ;恢复寄存器
                JMP     L_1A43

;子程序: 计算外部坐标 AX 在屏幕缓冲区的相对位移, 每个字符按两个字节计算
;输   入: AH=行号, AL=列号
;输   出: AX=相对位移
L_1CFA:
;由于 AX 中坐标为外部坐标, 所以还要将之换算为内部坐标.
```

```
                CMP     BYTE PTR DS:[49H],4     ;在西文方式吗?
                JB      L_1D11                  ;是, 不用转换
                CMP     BYTE PTR DS:[49H],7     ;是单显方式吗?
                JE      L_1D11                  ;是, 也不用转换
                CMP     AH,17H                  ;已是最后一行吗?
                JG      L_1D11                  ;是, 也不用转换
                ADD     AH,DS:[0A4H]            ;AH=内部行号

;此时, AX 已为内部坐标
L_1D11:
                PUSH    BX                      ;保存BX
                MOV     BX,AX                   ;BX=AX, 保存列号
                MOV     AL,AH                   ;AL=行号
                MUL     BYTE PTR DS:[4AH]       ;剩以屏幕列数
                XOR     BH,BH                   ;BX=列号
                ADD     AX,BX                   ;AX=每个字符按1字节计算时的位
                                                ;移
                SHL     AX,1                    ;剩以2, AX=每个字符按2字节计
                                                ;算时的位移
                POP     BX                      ;恢复BX
                RETN

;子程序: 检查定义的窗口, 若定义超过了内部屏幕, 则调整到边界
;输   入: CX=窗口左上角坐标
;         DX=窗口右下角坐标
;输   出: CX,DX 调整后的窗口位置
L_1D22:
                CMP     CH,17H
                JB      L_1D29                  ;窗口首行不能是内部屏幕的尾行
                MOV     CH,17H                  ;调整
L_1D29:
                CMP     DH,18H
                JB      L_1D30
                MOV     DH,18H                  ;窗口尾行在内部屏幕外面, 调整
                                                ;为内部屏幕的尾行
L_1D30:
                CMP     CL,DS:[4AH]
                JB      L_1D3C
                MOV     CL,DS:[4AH]
                DEC     CL                      ;调整左边列号
L_1D3C:
                CMP     DL,DS:[4AH]
                JB      L_1D48
                MOV     DL,DS:[4AH]
                DEC     DL                      ;调整右边列号
L_1D48:
                RETN
;功能6, 窗口上滚
L_1D49:
                CALL    L_1D22                  ;检查窗口定义是否正确
                MOV     BL,AL                   ;BL=要翻滚的行数
                CMP     AH,4                    ;在西文方式下吗?
                JB      L_1D56                  ;是, 直接调用原INT 10H即可
                JMP     L_1F5D
L_1D56:
```

```
                MOV     AH,6
                INT     78H                     ;调用原INT 10H实现上滚操作
                JMP     L_1A3F

                DB      148 DUP (0)             ;没有用

;功能7，窗口下滚
L_1DF1:
                CALL    L_1D22                  ;检查窗口定义是否正确
                MOV     BL,AL                   ;BL=要翻滚的行数
                CMP     AH,4                    ;在西文方式下吗?
                JB      L_1DFE                  ;是，直接调用原INT 10H即可
                JMP     L_20BD

L_1DFE:
                MOV     AH,7
                INT     78H                     ;调用原INT 10H实现下滚操作
                JMP     L_1A3F

                DB      43 DUP (0)              ;没有用

;功能 8，读取当前光标位置的字符和属性
L_1E30:
                CMP     AH,4                    ;是西文方式下吗?
                JB      L_1E38                  ;是，直接调用原INT 10H即可
                JMP     L_23F3
L_1E38:
                MOV     AH,8
                INT     78H                     ;调用原INT 10H
                JMP     L_1A3F

                DB      55 DUP (0)              ;没有用

;功能 9，在当前光标位置写字符和属性
L_1E76:
                CMP     AH,4                    ;是在西文方式下吗?
                JB      L_1E82                  ;是，直接调用原INT 10H即可
                MOV     D_18CA,BL               ;保存显示属性
                JMP     L_220C                  ;其它处理与功能0AH一样

L_1E82:
                MOV     AH,9
                INT     78H                     ;调用原INT 10H
                JMP     L_1A3F

                DB      44 DUP (0)              ;没有用

;功能 0AH，在当前光标位置写字符
L_1EB5:
                CMP     AH,4                    ;是在西文方式下吗?
                JB      L_1EBD                  ;是，直接调用原INT 10H即可
                JMP     L_220C                  ;转显示字符

L_1EBD:
                MOV     AH,0AH
```

```
                INT     78H                     ;调用原INT 10H
                JMP     L_1A3F

                DB      34 DUP (0)              ;没有用

;功能 0DH, 读点, 直接调用原 INT 10H 即可
L_1EE6:
                MOV     AH,0DH
                INT     78H                     ;调用原INT 10H
                JMP     L_1A3F

                DB      10 DUP (0)              ;没有用

;功能 0CH, 显示点, 直接调用原 INT 10H 即可
L_1EF7:
                MOV     AH,0CH
                INT     78H                     ;调用原INT 10H
                JMP     L_1A3F

                DB      95 DUP (0)              ;没有用

;屏幕上滚处理
L_1F5D:
                NOP
                NOP
                PUSH    ES
                PUSH    AX
                PUSH    BX
                PUSH    CX
                PUSH    DX                      ;保存寄存器

                PUSH    AX
                PUSH    BX
                PUSH    CX
                PUSH    DX                      ;再次保存寄存器
                MOV     BP,1
                CALL    L_1AAD                  ;消除光标
                POP     DX
                POP     CX
                POP     BX
                POP     AX                      ;恢复寄存器
                MOV     BL,AL                   ;BL=要上滚的行数
                PUSH    BX                      ;保存BX
                MOV     AX,CX                   ;AX=窗口左上角坐标
                CALL    L_1FEC                  ;数据初始化
;调用结果: DH=要窗口总行数, DL=窗口列数, SI=DI=窗口左上角缓冲区地址
          AX=行数x80, BP=列数, 若翻滚行数=0, ZF=1

                JZ      L_1FCA                  ;翻滚行数=0, 表示清除整个窗口
                ADD     SI,AX                   ;SI=窗口右下角地址
                MOV     AH,DH                   ;AH=窗口行数
                SUB     AH,BL                   ;减要翻滚的行数=空行数
                JNZ     L_1F88                  ;不是翻滚所有行

;这里应该处理翻滚窗口所有的行, 即相当于清除窗口. 实际处理方法是不合理的
```

```
                POP     AX                      ;恢复AX
                JMP     SHORT L_1F9F            ;不翻滚
                NOP

;整个窗口共 DH 行，翻滚后还剩下 AH 行. 这些行仅仅是向上移动，没有从窗口消失. 下面
;进行移动，此时 SI=窗口左上角地址，DI=要移动的首行地址
L_1F88:
                CALL    L_200D                  ;移动一行
                ADD     SI,BP                   ;SI指向下一行地址
                ADD     DI,BP                   ;DI指向下一行地址
                DEC     AH                      ;行数减一
                JNZ     L_1F88                  ;移动AH行

;此时 DI 已指向翻滚后窗口中第 1 个空行地址，共有 BL 个空行
L_1F93:
                POP     AX
                MOV     AL,0                    ;空行用字符0填充
L_1F96:
                CALL    L_203E                  ;填充一行
                ADD     DI,BP                   ;DI指向下一空行地址
                DEC     BL                      ;行数减1
                JNZ     L_1F96                  ;填充BL行
L_1F9F:
                POP     DX
                POP     CX
                POP     BX
                POP     AX
                POP     ES                      ;恢复寄存器

;上面的操作仅仅在屏幕缓冲区内实现了翻滚，实际屏幕上的内容还没有翻滚. 下面开始
;进行屏幕内容翻滚. 如果，在外部屏幕以外部分窗口内容不翻滚
                CMP     DS:[0A5H],CH            ;窗口首行在外部屏幕下面吗?
                JB      L_1FDF                  ;是，不用翻滚
                MOV     AH,DS:[0A4H]            ;AH=外部屏幕首行内部行号
                CMP     DH,AH                   ;窗口尾行在外部屏幕上面吗?
                JB      L_1FDF                  ;是，不用翻滚

;窗口全部或部分在外部屏幕中，要进行翻滚
                CMP     AH,CH                   ;窗口一部分在外部屏幕上面吗?
                JB      L_1FBB                  ;不是
;窗口的一部分在外部屏幕上面，那么将窗口缩小到外部屏幕内
                XOR     CH,CH                   ;窗口首行外部行号=0
                JMP     SHORT L_1FBD
                NOP
L_1FBB:
                SUB     CH,AH                   ;CH=窗口首行外部行号

L_1FBD:
                MOV     AH,DS:[0A5H]            ;AH=外部屏幕尾行内部行号
                CMP     DH,AH                   ;窗口一部分在外部屏幕下面吗?
                JB      L_1FCE                  ;不是
;窗口的一部分在外部屏幕下面，那么将窗口缩小到外部屏幕内
                MOV     DH,18H                  ;窗口尾行外部行号=外部屏幕尾行
                JMP     SHORT L_1FD2
                NOP
```

```
;处理翻滚行数为0的情况，将翻滚行数置为窗口行数后转一般处理
L_1FCA:
                MOV     BL,DH                   ;翻滚行数=窗口行数
                JMP     SHORT L_1F93            ;转翻滚

L_1FCE:
                SUB     DH,DS:[0A4H]            ;DH=窗口尾行外部行号
L_1FD2:
                CMP     AL,18H                  ;翻滚行数大于外部屏幕行数吗?
                JB      L_1FD8                  ;不是
                MOV     AL,18H                  ;改为外部屏幕行数
L_1FD8:
                CMP     CH,DH                   ;窗口仅1行吗?
                JE      L_1FDF                  ;是，不用翻滚
;按理一行的窗口也应该翻滚

                CALL    L_2057                  ;进入屏幕内容翻滚
L_1FDF:
                MOV     BYTE PTR DS:[55H],0FFH  ;允许执行光标操作子程序
                XOR     BP,BP
                CALL    L_1AAD                  ;恢复光标，光标位置没有变
                JMP     L_1A3F

;子程序：屏幕内容翻滚前初始化设置
;输  入：AX=窗口左上角位置
;       BL=翻滚行数
;       CX,DX=窗口坐标
;输  出：AX=从窗口首行到翻滚后内容仍要保留的行内容的长度
;       DH=窗口行数，DL=窗口列数
;       SI=DI=窗口左上角在屏幕缓冲区中的相对位移
;       ZF=0,表示翻滚行数=0
L_1FEC:
                CALL    L_1D11                  ;取坐标AX在屏幕缓冲区的相对位
                                                ;移->AX(每字符两字节)
                SAR     AX,1                    ;AX=每字符一字节的位移
                MOV     DI,AX
                MOV     SI,AX                   ;SI=DI=窗口左上角在屏幕缓冲区
                SUB     DX,CX                   ;中是相对位移
                INC     DH
                INC     DL                      ;DH=窗口行数，DL=窗口列数
                XOR     CH,CH                   ;CH=0
                MOV     BP,DS:[4AH]             ;BP=屏幕列数
                MOV     AL,BL                   ;AL=翻滚行数
                MUL     BYTE PTR DS:[4AH]       ;AX=翻滚行占屏幕缓冲区长度
                PUSH    DS
                POP     ES                      ;ES=DS
                CMP     BL,0                    ;翻滚行数=0?
                RETN

;子程序：在屏幕缓冲区中复制内容
;输  入：SI=源数据首址，DI=目标数据首址，DL=复制长度
;输  出：无
L_200D:
                MOV     CL,DL                   ;CL=复制长度
```

```
        PUSH    SI
        PUSH    DI
        ADD     SI,0AFH
        ADD     DI,0AFH
        REP     MOVSB                   ;复制屏幕字符缓冲区
        POP     DI
        POP     SI

        PUSH    SI
        PUSH    DI
        ADD     SI,104FH
        ADD     DI,104FH
        MOV     CL,DL
        REP     MOVSB                   ;复制屏幕字符类型缓冲区
        POP     DI
        POP     SI

        PUSH    SI
        PUSH    DI
        ADD     SI,087FH
        ADD     DI,087FH
        MOV     CL,DL
        REP     MOVSB                   ;复制屏幕字符属性缓冲区
        POP     DI
        POP     SI
        RETN

;子程序: 填充屏幕缓冲区, 该子程序没有填充屏幕字符属性缓冲区(应该也填充)
;输   入: AL=字符值, DI=填充首址, DL=长度
;输   出: 无
L_203E:
        MOV     CL,DL                   ;CL=长度

        PUSH    DI
        ADD     DI,0AFH
        REP     STOSB                   ;填充屏幕字符缓冲区
        POP     DI
        PUSH    DI
        ADD     DI,104FH
        PUSH    AX
        XOR     AL,AL                   ;AL=0
        MOV     CL,DL
        REP     STOSB                   ;用0填充屏幕字符类型缓冲区
        POP     AX
        POP     DI
        RETN

;子程序: 屏幕内容上滚
;输   入: CX,DX 窗口坐标, BH:空白行属性
;输   出: 无
L_2057:
        CMP     CH,DH                   ;窗口首行在尾行上方吗?
        JB      L_205C                  ;是
;窗口首行与尾行相同或在尾行下面, 不翻滚
```

```
                RETN

L_205C:
                PUSH    DS                      ;保存DS
```

```
                MOV     BL,AL           ;BL=翻滚行数
                MOV     AX,CX           ;AX=左上角坐标
                PUSH    BX
                CALL    L_23C9          ;取坐标AX在视频区的相对位移
                                        ->AX
                POP     BX
                MOV     DI,AX           ;DI=坐标在视频区首址
                SUB     DX,CX
                ADD     DX,101H         ;DH=窗口行数, DL=窗口列数
                PUSH    BP
                PUSH    AX
                MOV     AL,12H          ;每行18包括条扫描线
                MUL     DH              ;剩以总行数=总扫描线数
                MOV     BP,AX           ;保存于BP
                POP     AX
                PUSH    ES
                POP     DS              ;DS=ES=视频区段址
                SUB     CH,CH
                MOV     AL,12H
                MUL     BL              ;AX=要清除的总扫描线数
                PUSH    BX              ;保存BH, 属性字节
                CMP     AX,0            ;是清除整个窗口吗?
                JE      L_20AD          ;是
                MOV     BX,AX           ;保存于BX中
                PUSH    DX
                MOV     DX,50H          ;每条扫描线共640个点=80字节
                MUL     DX              ;AX=总要清除的长度=保留长度
                POP     DX
                MOV     SI,DI
                ADD     SI,AX           ;SI=源数据地址
                MOV     AX,BP           ;AX=窗口总线扫描数
                SUB     AX,BX           ;减清除线数=要移动的线数
L_2096:
                CALL    L_21BA          ;移动一条扫描线
                DEC     AX
                JNZ     L_2096          ;共 移动AX条扫描线
L_209C:
                POP     AX              ;AH=原BH=空白行属性
                MOV     CL,4
                SHR     AH,CL           ;取背景色
L_20A1:
                CALL    L_21E0          ;清除一条扫描线
                DEC     BX
                JNZ     L_20A1          ;清除BX条扫描线

                POP     BP
                POP     DS              ;恢复寄存器

                CALL    L_20B1          ;恢复页面选择寄存器
                RETN

;清除整个窗口
L_20AD:
                MOV     BX,BP           ;BX=窗口总线数
                JMP     SHORT L_209C
```

```
;子程序: 恢复页面寄存器
;输   入: 无
;输   出: 无
L_20B1:
                PUSH    DX
                PUSH    AX                      ;保存寄存器
                MOV     DX,3C4H                 ;索引寄存器端口
                MOV     AX,0F02H                ;允许向4个页面写数据
                OUT     DX,AX
                POP     AX                      ;恢复寄存器
                POP     DX
                RETN

;屏幕下滚处理
L_20BD:
                NOP
                NOP
                PUSH    ES
                PUSH    AX
                PUSH    BX
                PUSH    CX
                PUSH    DX                      ;保存寄存器

                PUSH    AX
                PUSH    BX
                PUSH    CX
                PUSH    DX                      ;再保存一次
                MOV     BP,1
                CALL    L_1AAD                  ;消除光标
                POP     DX
                POP     CX
                POP     BX
                POP     AX                      ;恢复寄存器
                STD                             ;设置递减的字符串操作方式
                MOV     BL,AL                   ;BL=翻滚行数
                PUSH    BX
                MOV     AX,DX                   ;AX=窗口右下角坐标
                CALL    L_1FEC                  ;数据初始化
;调用结果: DH=要窗口总行数, DL=窗口列数, SI=DI=窗口左上角缓冲区地址
;         AX=行数 x80, BP=列数, 若翻滚行数=0, ZF=1
                JZ      L_211E                  ;翻滚行数=0, 即清除整个窗口
                SUB     SI,AX                   ;SI=窗口左上角在屏幕缓冲区中
                                                ;的相对位移
                MOV     AH,DH                   ;AH=窗口总行数
                SUB     AH,BL                   ;AH=翻滚后仍存在的行数
                JNZ     L_20E9

;翻滚的行数=窗口的总行数, 此时应该全部翻滚, 实际处理方法是错误的
                POP     AX
                JMP     SHORT L_2100            ;不翻滚
                NOP

;翻滚屏幕缓冲区
L_20E9:
```

```
                CALL    L_200D                  ;移动一行
                SUB     SI,BP
                SUB     DI,BP
                DEC     AH
                JNZ     L_20E9                  ;共移动AH行，这些行仍要被保留
                                                ;在窗口中
L_20F4:
                POP     AX
                MOV     AL,0                    ;字符=0
L_20F7:
                CALL    L_203E                  ;清除一行
                SUB     DI,BP
                DEC     BL
                JNZ     L_20F7                  ;清除翻滚出来的空白行
L_2100:
                POP     DX
                POP     CX
                POP     BX
                POP     AX
                POP     ES                      ;恢复寄存器

                CMP     DS:[0A5H],CH            ;窗口在外部屏幕的下面吗?
                JB      L_213C                  ;是，不用翻滚
                CMP     DH,DS:[0A4H]            ;窗口在外部屏幕的上面吗?
                JB      L_213C                  ;是，不用翻滚

;窗口全部或部分在外部屏幕中，要进行翻滚
                MOV     AH,DS:[0A4H]            ;AH=外部屏幕首行内部行号
                CMP     AH,CH                   ;窗口部分在外部屏幕上面吗?
                JB      L_2122                  ;不是
                XOR     CH,CH                   ;窗口首行外部行号=0
                JMP     SHORT L_2124
                NOP
;处理清除整个窗口
L_211E:
                MOV     BL,DH                   ;清除行数=BL=窗口总行数
                JMP     SHORT L_20F4            ;转正常翻滚

L_2122:
                SUB     CH,AH                   ;内部行号转换为外部行号
L_2124:
                MOV     AH,DS:[0A5H]            ;AH=外部屏幕尾行内部行号
                CMP     DH,AH                   ;窗口部分在屏幕下面吗?
                JB      L_2131                  ;不是
                MOV     DH,17H                  ;窗口尾行外部行号=17H
;在CV26.COM中，因为外部屏幕总共有26行，这里DH应该=18H
                JMP     SHORT L_2133
                NOP
L_2131:
                SUB     AH,DH                   ;内部行号转换为外部行号
L_2133:
                CMP     AL,17H                  ;翻滚行数大于24吗?
;同样，上面这条指令应改为CMP AL,18H

                JB      L_2139
```

```
                MOV     AL,17H                  ;翻滚行数最大为24行(应为25)
L_2139:
                CALL    L_2149                  ;屏幕内容翻滚
L_213C:
                MOV     BYTE PTR DS:[55H],0FFH  ;允许进行光标操作
                XOR     BP,BP
                CALL    L_1AAD                  ;恢复光标
                JMP     L_1A3F

;子程序: 屏幕内容下滚
;输   入: CX,DX 窗口坐标, BH:空白行属性
;输   出: 无
L_2149:
                CMP     CH,DH
                JB      L_214E
                RETN
;窗口首行与尾行相同或在尾行后, 不翻滚

L_214E:
                PUSH    DS
                STD                             ;设置递减的字符串操作方式
                MOV     BL,AL                   ;BL=翻滚行数
                MOV     AX,DX                   ;AX=窗口右下角坐标
                PUSH    BX
                CALL    L_23C9                  ;取坐标AX在视频区的相对位移->AX
                POP     BX
                MOV     DI,AX
                SUB     DX,CX
                ADD     DX,101H                 ;DH=窗口行数, DL=窗口列数
                PUSH    BP
                PUSH    AX
                MOV     AL,12H                  ;每行共18条扫描线
                MUL     DH                      ;AX=窗口总扫描线数
                MOV     BP,AX                   ;保存于BP中
                POP     AX
                PUSH    ES
                POP     DS                      ;DS=ES=0A000H=视频区段址
                SUB     CH,CH
                ADD     DI,OFFSET D_04B0
                MOV     AL,12H
                MUL     BL                      ;空白行的总扫描线数
                PUSH    BX
                CMP     AX,0                    ;清除整个窗口吗?
                JE      L_21B1                  ;是

                MOV     BX,AX                   ;BX=空白行线数
                PUSH    DX
                MOV     DX,50H
                MUL     DX                      ;AX=空白行占缓冲区长度
                POP     DX
                MOV     SI,DI
                SUB     SI,AX                   ;SI=要保留行首址
                MOV     AX,BP                   ;AX=总线数
                SUB     AX,BX                   ;减空白行线数=移动线数
L_218D:
```

```
                CALL    L_21BA                  ;移动一条扫描线
                SUB     SI,0A0H                 ;指向上一条线地址，由于是反向
                                                ;操作，而SI和DI已在子程序中加
                                                ;50H了，因此，这里要减0A0H
                SUB     DI,0A0H
                DEC     AX
                JNZ     L_218D                  ;移动AX条扫描线
L_219B:
                POP     AX                      ;AH=原BH=空白行属性
                MOV     CL,4
                SHR     AH,CL                   ;取背景色
L_21A0:
                CALL    L_21E0                  ;清除一行空白行
                SUB     DI,0A0H                 ;指向上一行地址
                DEC     BX
                JNZ     L_21A0

                CLD                             ;恢复字符串操作方式
                POP     BP
                POP     DS
                CALL    L_20B1                  ;恢复寄存器设置
                RETN
L_21B1:

;清除整个窗口
                MOV     BX,BP                   ;BX=窗口总扫描行数
                JMP     SHORT L_219B

                NOP
                DB      4 DUP (0)               ;没有用

;子程序：复制视频区内容
;输  入：DL=长度，SI=源数据地址，DI=目标数据地址;输  出：无
L_21BA:
                MOV     CL,DL                   ;CL=长度
                PUSH    DX
                PUSH    AX                      ;保存寄存器
                MOV     DX,3CEH                 ;DX=显示寄存器端口地址
;3CEH 是图形控制寄存器中的索引寄存器，指定以下对 3CFH 寄存器操作的具体寄存器号
                MOV     AX,105H                 ;AL->3CEH，AH->3CFH
;AL=5，选择模式选择寄存器，AH=1，表示设置 1 号写入模式，即使用锁存器内容作为写
;入数据
                OUT     DX,AX
                PUSH    SI
                PUSH    DI
                REP     MOVSB
;读入数据后，存放于锁存器中，然后，再将数据写入目标地址，实现复制功能

                POP     DI
                POP     SI
                ADD     SI,50H                  ;SI=指向下一线地址
                ADD     DI,50H                  ;DI=指向下一线地址
                MOV     AX,5                    ;恢复0号写入模式
                OUT     DX,AX
                POP     AX
```

```
                POP     DX                      ;恢复寄存器
                RETN

                DB      0,0,88H,0,0,0,0,0       ;没有用

;子程序: 清除视频区
;输   入: DL=长度, DI=首址, AH=清除属性
;输   出: 无
L_21E0:
                MOV     CL,DL                   ;CL=长度
                CALL    L_20B1                  ;允许向4页面写数据
;先清除所有数据
                PUSH    DI
                MOV     AL,0
                REP     STOSB                   ;用0填充
                POP     DI
;再写入指定的清除颜色
                MOV     CL,DL
                CALL    L_21FA                  ;设置属性
                PUSH    DI
                MOV     AL,0FFH                 ;用全1写
                REP     STOSB                   ;用指定颜色填充
                POP     DI
                ADD     DI,50H                  ;DI指向下一线地址
                RETN

;子程序: 设置页面写允许寄存器
;输   入: AH=设置值
;输   出: 无
L_21FA:
                PUSH    DX
                MOV     DX,3C4H
                MOV     AL,2                    ;选择页面写允许寄存器
                OUT     DX,AX                   ;设置
                POP     DX
                RETN

                DB      9 DUP (0)               ;没有用

;显示字符, AL=字符
L_220C:
                PUSH    AX                      ;保存字符
                CMP     AL,80H                  ;是扩展字符吗?
                JB      L_2225                  ;仅是一般<128的ASCII字符
                CMP     AL,0A0H                 ;是汉字吗?
                JG      L_2225                  ;是

;显示大于等于 128 的非汉字字符
L_2215:
                AND     BYTE PTR DS:[0AEH],0FEH ;不能执行光标操作子程序
                MOV     DI,DS:[50H]             ;取光标位置
                CALL    L_22AE                  ;显示字符
                POP     AX
                JMP     L_1A3F
```

```
L_2225:
                CMP     AL,0FFH                 ;是255号字符吗?
                JE      L_2215                  ;是, 255号字符也不是汉字

                CALL    L_25EB                  ;综合处理
;返回 AH=0: 汉字的前一字节, 不用显示
;       AH=1: 一般 ASCII 字符, 显示字符
;       AH=2: 汉字的第 2 个字节, 显示汉字

                MOV     BP,AX
                TEST    AH,7                    ;不用显示吗?
                JNZ     L_2237                  ;要显示

;汉字的前一个字节, 不用显示
                POP     AX
                JMP     L_1A3F
L_2237:
                TEST    AH,1                    ;是一般ASCII字符吗?
                JNZ     L_223F                  ;是
                JMP     SHORT L_2245            ;转显示汉字
                NOP

L_223F:
                CALL    L_22AE                  ;显示一般ASCII字符
                JMP     SHORT L_224E
                NOP

L_2245:
                MOV     SI,DX                   ;SI=汉字点阵数据段址
                MOV     DI,DS:[71H]             ;DI=显示的位置
                CALL    L_2257                  ;显示汉字
L_224E:         XOR     BP,BP
                CALL    L_1AAD                  ;设置光标
                POP     AX
                JMP     L_1A3F

;子程序: 显示汉字
;输   入: SI=汉字点阵数据段址, DI=显示位置
;输   出: 无
L_2257:
                PUSH    ES
                PUSH    DI
                PUSH    DS
                MOV     DI,77H                  ;汉字点阵数据存放区地址
                POP     ES                      ;ES=DS
                PUSH    ES
                MOV     DS,SI                   ;DS=汉字点阵数据段址
                XOR     SI,SI                   ;偏移=0
                MOV     CX,10H                  ;共16个字
L_2266:
                LODSW                           ;取16点
                STOSB                           ;保存前8点
                MOV     ES:[DI+11H],AH          ;保存后8点
;显示一个汉字相当于显示两个字符, 因此, 这里将一个汉字的左半边的点阵数据和右半
;边的点阵数据分开, 目的便是为了分两次显示
```

```
                LOOP    L_2266

                JMP     SHORT L_227F

                NOP
                DB      14 DUP (0)              ;没有用

L_227F:
                POP     DS                      ;恢复DS

                POP     DI                      ;恢复坐标
                POP     ES
                MOV     SI,77H                  ;点阵数据存放区地址
                PUSH    DI
                MOV     AX,DI                   ;AX=坐标
                CALL    L_23C9                  ;取坐标AX在视频区的位移->AX
                MOV     DI,AX
                MOV     CL,1                    ;显示次数
                CALL    L_2319                  ;显示左半边的汉字点阵
                ADD     SI,12H                  ;SI=汉字右半边点阵数据地址
                POP     AX                      ;恢复坐标
                INC     AL                      ;指向下一列
                CMP     AL,DS:[4AH]             ;已为当前行最后一行?
                JB      L_22A2                  ;没有
                XOR     AL,AL                   ;列号=0
                INC     AH                      ;行号加1
L_22A2:
                CALL    L_23C9                  ;取坐标AX在视频区的位移->AX
                MOV     DI,AX
                MOV     CX,1                    ;显示次数
                CALL    L_2319                  ;显示右半边的汉字点阵
RETN

;子程序: 显示字符
;输  入: AL=字符
;输  出: 无
L_22AE:
                XOR     AH,AH
                PUSH    ES
                PUSH    DS
                CMP     AL,80H                  ;显示扩展字符吗?
                JAE     L_22BC                  ;是
;扩展字符的点阵数据地址由 INT 1FH 提供, 小于 128 的字符点阵数据就在 CCCC.COM 中

;处理小于 128 的字符
                MOV     SI,2798H                ;<128的ASCII字符点阵数据首址
                PUSH    CS
                JMP     SHORT L_22C7

;处理扩展字符
L_22BC:
                SUB     AL,80H                  ;减去128
                SUB     SI,SI
                MOV     DS,SI                   ;DS=0
                LDS     SI,DWORD PTR DS:[7CH]   ;取INT 1FH中断向量->DS:SI
```

```
                PUSH    DS
L_22C7:
                SHL     AX,1
                SHL     AX,1
                SHL     AX,1                    ;AX=AXx8=当前字符在点阵数据区
                                                ;的相对位移
                ADD     SI,AX                   ;SI=当前字符点阵数据首址
                POP     DS
                POP     ES
                PUSH    DI
                PUSH    CX
                MOV     DI,77H                  ;点阵数据存放区地址
                MOV     CX,8                    ;共8字节
L_22D9:
                LODSB                           ;取8点
                STOSB
                STOSB                           ;连写两次
;用于显示的汉字点阵是 16x16 的，当前取得的字符点阵数据是 8x8 的，为了让显示的字符
;与汉字一样高，要将点阵数据扩大为 16x8
                LOOP    L_22D9

                JMP     SHORT L_22FF

                NOP
                DB      30 DUP (0)              ;没有用

L_22FF:
                POP     CX
                POP     DI
                PUSH    ES
                POP     DS                      POP     ES
                MOV     SI,77H                  ;SI=点阵起始地址
                MOV     AX,DI                   ;AX=坐标

                CMP     AH,18H
                JNE     L_2310
                INC     BH
;上面三条指令在 CV26.COM 中没有作用，以后 BH 根本没有用

L_2310:
                CALL    L_23C9                  ;取坐标AX在视频区的相对位移
                MOV     DI,AX
                CALL    L_2319                  ;显示字符
                RETN

;子程序：显示字符
;输　入：SI=点阵数据首址，DI=在视频区显示首址，BL=属性，CX=显示次数
;输　出：无
L_2319:
                PUSH    DX
                PUSH    AX
                PUSH    BX
                CALL    L_2350                  ;清除当前位置内容
L_231F:
                PUSH    DI
```

```
                PUSH    SI
                MOV     BH,10H                  ;显示16条扫描线
L_2323:
                LODSB                           ;取8点数据
                TEST    BL,80H                  ;覆盖原数据?
                JZ      L_232D                  ;是
                XOR     AL,ES:[DI]              ;与原数据异或，相当于保存了原
                                                ;数据
                NOP
L_232D:
                CALL    L_23A0                  ;显示一排数据
                ADD     DI,4FH                  ;DI=下一排显示地址
                DEC     BH
                JNZ     L_2323                  ;显示16条线
                POP     SI
                POP     DI
                INC     DI
                LOOP    L_231F                  ;显示CX次

                MOV     AX,0F02H
                OUT     DX,AX                   ;恢复页面写允许寄存器值
                POP     BX
                POP     AX
                POP     DX                      ;恢复寄存器
                RETN

                DB      12 DUP (0)              ;没有用

;子程序: 清除当前位置处原有图形
;输    入: 无;输    出: 无
L_2350:
                MOV     AH,D_18CA               ;取属性字节
                MOV     DX,3C4H
                OR      AH,AH                   ;要保存原数据吗?
                JGE     L_2363                  ;不要

                MOV     AX,1803H                ;字符集选择寄存器
                OUT     DX,AX
;上面两条语句似乎没有用

                JMP     SHORT L_2375
                NOP
                NOP

L_2363:
                MOV     AX,0F02H                ;允许写入4个页面
                OUT     DX,AX
                PUSH    DI
                XOR     AL,AL                   ;用0清除
;将 4 个页面全部清 0，即抹去原图形
                MOV     BH,10H                  ;16条线
L_236C:
                STOSB                           ;抹一条线
                ADD     DI,4FH                  ;下一条线
                DEC     BH
```

```
                JNZ     L_236C

                POP     DI
L_2375:
                MOV     AL,D_18CA               ;取属性字节
                SHR     AL,1
                SHR     AL,1
                SHR     AL,1
                SHR     AL,1                    ;取高4位值
                AND     AL,7                    ;清除最高位数据，得背景颜色
;最高位不是图形颜色数据，仅指明是否覆盖原图形
                MOV     D_18CB,AL               ;保存
                RETN

                NOP
                DB      14 DUP (0)
                DB      53H,52H,0BBH,50H,00H,0F7H
                DB      0E3H,5AH,5BH,0C3H,90H   ;没有用

;子程序：显示 8 点
;输   入：AL=图形数据
;输   出：无
L_23A0:
                PUSH    AX                      ;保存AX
                MOV     AH,D_18CA               ;取前景颜色
                MOV     AL,2
                OUT     DX,AX                   ;设置颜色屏幕寄存器
                POP     AX
                MOV     ES:[DI],AL              ;显示前景点阵
                NOT     AL                      ;得背景点阵
                PUSH    AX
                MOV     AH,D_18CB               ;取背景颜色
                MOV     AL,2
                OUT     DX,AX                   ;设置为背景颜色
                POP     AX
                STOSB                           ;显示背景点阵
                RETN

                NOP
                DB      12 DUP (0)              ;没有用

;子程序：计算当前光标位置在视频区中的相对位移
;输   入：无
;输   出：AX=相对位移
L_23C6:
                MOV     AX,DS:[50H]             ;取光标位置

;此时，AX 已为光标位置
L_23C9:
                PUSH    BX
                PUSH    CX
                PUSH    AX                      ;保存寄存器
                CMP     AH,18H
                JNE     L_23D3
                MOV     AH,18H
```

```
L_23D3:
                MOV     CL,12H                  ;每行18条线
                XOR     CH,CH
                MOV     BX,AX
                MOV     AL,AH
                MUL     BYTE PTR DS:[4AH]       ;行号x列长->AX
                PUSH    DX
                MUL     CX                      ;再剩以每行18线
                POP     DX
                SUB     BH,BH
                ADD     AX,BX                   ;加当前行位移
                POP     CX

                CMP     CH,18H                  ;光标在提示行吗?
                JB      L_23F0                  ;不是
                ADD     AX,50H                  ;加一条扫描线
L_23F0:
                POP     CX
                POP     BX                      ;恢复寄存器
                RETN

;读当前光标处的字符及其属性
L_23F3:
                MOV     AX,DS:[50H]             ;取光标位置
                CALL    L_1CFA                  ;计算当前光标位置在屏幕缓冲区
                                                ;中的相对位移(每字符两字节)
                SAR     AX,1                    ;AX=内部相对位移(每字符一字节)
                MOV     SI,AX
                MOV     AL,DS:0AFH[SI]          ;取字符
                MOV     AH,D_087F[SI]           ;取属性
                JMP     L_1A3F

;在当前光标位置以 TTY 方式显示字符
L_2408:
                OR      BYTE PTR DS:[0AEH],0FEH ;当没有建立光标时，允许执行光
                                                ;标操作子程序
                PUSH    AX
                PUSH    AX
                MOV     AH,3
                INT     10H                     ;读当前光标位置
                POP     AX

                CMP     AL,8                    ;是退格字符吗?
                JNE     L_241B                  ;不是

                JMP     SHORT L_2482            ;转处理退格字符
                NOP
L_241B:
                CMP     AL,0DH                  ;是回车字符吗?
                JNE     L_2422                  ;不是

                JMP     SHORT L_248B            ;转处理回车
                NOP
L_2422:
                CMP     AL,0AH                  ;是换行符吗?
```

```
                JNE     L_2429                  ;不是

                JMP     SHORT L_248F            ;转处理换行符
                NOP

L_2429:
                CMP     AL,7                    ;是响玲字符吗?
                JNE     L_2430                  ;不是

                JMP     SHORT L_2496            ;转处理响玲字符
                NOP
L_2430:
                MOV     BH,DS:[62H]             ;取当前页号
                MOV     AH,0AH                  ;显示字符
                MOV     CX,1                    ;一个
                INT     10H                     ;调用10号功能显示字符
;显示后光标要前进一字符位置，若已在行尾，则要换行，若是外部屏幕的最后行，还要
;翻滚.
                INC     DL                      ;光标列数加1
                CMP     DL,DS:[4AH]             ;到行尾了吗?
                JNE     L_247E                  ;没有
                MOV     DL,0                    ;列号改为0
                CMP     DH,18H                  ;到最后一行了吗?
                JB      L_247C                  ;没有
L_244A:
                MOV     AH,2
                MOV     BH,0
                INT     10H                     ;设置光标，翻滚后光标位置不变
                MOV     AL,DS:[49H]             ;取当前显示方式
                CMP     AL,4                    ;是西文方式吗?
                JB      L_245D                  ;是
                CMP     AL,7                    ;是单色显示方式吗?
                MOV     BH,0                    ;页号均为0
                JNZ     L_2463                  ;不是单色显示方式，即汉字方式

;西文方式要取得翻滚行的属性
L_245D:
                MOV     AH,8
                INT     10H
                MOV     BH,AH                   ;保存属性
L_2463:
                MOV     AX,601H                 ;上滚一行
                MOV     CX,0                    ;窗口左上角坐标为(0,0)
                MOV     DH,18H
                MOV     DL,DS:[4AH]
                DEC     DL                      ;窗口右下角为内部屏幕右下角,
                                                ;相当于定义整个内部屏幕为窗口
L_2471:
                INT     10H                     ;翻滚一行
L_2473:
                AND     BYTE PTR DS:[0AEH],1    ;不能执行光标操作子程序
                POP     AX
                JMP     L_1A3F

L_247C:
```

```
                INC     DH                      ;换行至下一行首
L_247E:
                MOV     AH,2
                JMP     SHORT L_2471            ;设置光标位置

;处理退格字符
L_2482:
                CMP     DL,0                    ;当前光标位置已是行首吗?
                JE      L_247E                  ;是, 不用退格了
                DEC     DL                      ;光标列位置减1
                JMP     SHORT L_247E

;处理回车字符
L_248B:
                MOV     DL,0                    ;光标列号=0
                JMP     SHORT L_247E

;换行处理
L_248F:
                CMP     DH,18H                  ;已是屏幕最后行吗?
                JNE     L_247C                  ;不是
                JMP     SHORT L_244A            ;是, 要翻滚

;响玲字符处理
L_2496:
                MOV     BL,2                    ;循环计数次数
                CALL    L_249D                  ;调用响玲子程序
                JMP     SHORT L_2473
;子程序: 响玲
;输   入: BL=大循环次数
;输   出: 无
L_249D:
                MOV     AL,0B6H                 ;设置定时器2工作于方式3
                OUT     43H,AL                  ;输入命令

                MOV     AX,533H                 ;频率数据
                OUT     42H,AL                  ;输出低8位
                MOV     AL,AH
                OUT     42H,AL                  ;输出高8位
                IN      AL,61H                  ;取原端口B数据
                MOV     AH,AL                   ;保存于AH中
                OR      AL,3                    ;启动扬声器
                OUT     61H,AL                  ;输出命令
                SUB     CX,CX                   ;设置循环计数次数
                                                ;0相当于65536
L_24B4:
                LOOP    L_24B4

                DEC     BL                      ;大循环
                JNZ     L_24B4

                MOV     AL,AH
                OUT     61H,AL                  ;恢复端口B数据
                RETN
```

```
;功能 11H, 直接返回
L_24BF:
                XOR     AX,AX                   ;AX=0
                JMP     L_1A3F                  ;返回

;功能 10H, 提示行管理
L_24C4:
                OR      AL,AL                   ;子功能号=0?
                JZ      L_24D2                  ;是, 转清除提示行
                CMP     AL,1                    ;子功能号=1?
                JE      L_2507                  ;是, 转在提示行显示字符
                CMP     AL,2                    ;子功能号=2?
                JE      L_24F6                  ;是, 转设置提示行光标
                JMP     SHORT L_250D            ;转在提示行仿TTY显示字符

;清除提示行, 光标移至行首
L_24D2:
                MOV     BYTE PTR DS:[56H],AL    ;AL=0, 提示行光标位置置0
                MOV     DS:[73H],AL             ;清除汉字前一字节标志
;当显示汉字前一字节时, 实际并不显示, 仅将 73H 置 1, 并将其值保存在单元 0A1H 中

;清除提示行内容, 这里假定页面写允许寄存器已设置为可以向 4 个页面写数据(这是缺省
;的设置)
                MOV     DI,8CF0H                ;视频缓冲区中提示行首址
                MOV     CX,550H                 ;要清除的字节数, 共17条线
                PUSH    DI                      ;保存提示行首址
                PUSH    CX                      ;保存长度
                REP     STOSB                   ;清除原数据
                POP     CX                      ;恢复长度
                POP     DI                      ;恢复提示行首址

                MOV     DX,3C4H
                MOV     AX,102H                 ;设置页面选择寄存器, 设置蓝色
                OUT     DX,AX
                MOV     AL,0FFH                 ;用全1写
                REP     STOSB                   ;将整个提示行用写入蓝色点阵
                MOV     AX,0F02H
                OUT     DX,AX                   ;恢复面允许寄存器
                JMP     L_1A3F

;设置提示行光标位置, 清除当前光标位置字符. 提示行光标实际上是字符'_'.
L_24F6:
                PUSH    DX                      ;保存光标位置
                MOV     DL,20H                  ;空格
                CALL    L_254A                  ;用空格清除当前位置字符
                POP     DX                      ;恢复光标位置
                MOV     BYTE PTR DS:[56H],DL    ;保存光标位置
                CALL    L_2548                  ;显示'_'字符, 表示提示行光标
                JMP     L_1A3F

;在提示行显示字符(DL)
L_2507:
                CALL    L_254A                  ;调用提示行显示字符子程序
                JMP     L_1A3F
```

```
;在提示行以 TTY 方式显示字符
L_250D:
                CMP     DL,8                    ;是退格字符吗?
                JE      L_252C                  ;是
                CMP     DL,7                    ;是响玲字符吗?
                JE      L_2540                  ;是
                CALL    L_254A                  ;直接显示字符
                MOV     AL,BYTE PTR DS:[56H]    ;取当前光标位置
                CMP     AL,4FH                  ;到行尾了吗?
                JAE     L_2523                  ;是, 不能再增加了
                INC     AL                      ;光标位置加1
L_2523:
                MOV     BYTE PTR DS:[56H],AL    ;保存光标位置
L_2526:
                CALL    L_2548                  ;显示光标('_')
                JMP     L_1A3F

;处理退格字符
L_252C:
                MOV     DL,20H
                CALL    L_254A                  ;用空格清除当前光标位置字符
                MOV     AL,BYTE PTR DS:[56H]    ;取光标位置
                CMP     AL,0                    ;已到第0列了吗?
                JE      L_253A                  ;是, 不能退格了
                DEC     AL                      ;光标位置减1
L_253A:
                MOV     BYTE PTR DS:[56H],AL    ;保存光标位置
                JMP     SHORT L_2526
                NOP

;处理响玲字符
L_2540:
                MOV     BL,2                    ;大循环次数
                CALL    L_249D                  ;响玲
                JMP     L_1A3F

;子程序: 显示提示行光标, 实际上相对于显示字符'_'
;输   入: 无
;输   出: 无
L_2548:
                MOV     DL,5FH                  ;显示'_'字符

;此时 DL 已是要显示的字符了
L_254A:
                MOV     DI,8D3FH                ;DI=提示行视频区首址
                MOV     BL,BYTE PTR DS:[56H]    ;取提示行光标位置
                MOV     BH,0
                ADD     DI,BX                   ;DI=视频区光标位置地址
                MOV     AL,DL                   ;AL=字符
                CMP     AL,0A0H                 ;是汉字吗?
                JAE     L_2594                  ;是, 转处理汉字

;取字符点阵数据
                MOV     BL,8                    ;每个字符8字节
                MUL     BL
```

```
                ADD     AX,2798H
                MOV     SI,AX                   ;SI=字符点阵数据首址
                INC     DI
                MOV     DX,3C4H
                MOV     AX,0A02H                ;设置显示颜色
                OUT     DX,AX
                MOV     CX,8                    ;显示8字节
L_256F:
                LODSB
                STOSB                           ;显示
                MOV     ES:[DI+4FH],AL          ;放大，即将原8x8点阵图形放大
为16x8点阵
                                                ;图形
                ADD     DI,9FH                  ;修改显示地址
                LOOP    L_256F

                MOV     AX,0F02H
                OUT     DX,AX                   ;恢复页面写允许寄存器
                RETN

                DB      20 DUP (0)              ;没有用

;处理在提示行显示汉字
L_2594:
                MOV     BX,73H
                TEST    BYTE PTR [BX],1         ;已显示了前半个汉字了吗?
                JNZ     L_25A2                  ;是
;当前显示的是汉字的前一字节
                INC     BYTE PTR [BX]           ;置标志
                MOV     BYTE PTR DS:[0A1H],AL   ;保存汉字的前一字节值
                RETN

L_25A2:
                AND     BYTE PTR [BX],0F0H      ;清除汉字前一字节标志
                MOV     DH,BYTE PTR DS:[0A1H]   ;取汉字的前一字节
                CALL    L_2778                  ;读汉字点阵
                PUSH    DX
                MOV     DX,3C4H
                MOV     AX,0A02H                ;设置显示颜色
                OUT     DX,AX
                POP     DS
                XOR     SI,SI                   ;DS:SI指向汉字点阵地址
                MOV     CX,10H                  ;共16条线
L_25BA:
                LODSW
                STOSW                           ;显示
                ADD     DI,4EH                  ;下一排
                LOOP    L_25BA

                PUSH    CS
                POP     DS
                MOV     AX,0F02H
                OUT     DX,AX                   ;恢复寄存器设置
                RETN
```

```
                DB      0, 0, 0, 0                      ;没有用

;功能 12H, 读汉字点阵数据
L_25CC:
                CALL    L_2778                  ;读汉字(DX)的点阵数据
                MOV     DS,DX                   ;DS=点阵数据段
                MOV     ES,BP                   ;ES=目标数据段
                MOV     DI,BX                   ;DI=目标偏移地址
                XOR     SI,SI
                MOV     CX,10H
                REP     MOVSW                   ;传送数据
                JMP     L_1A3F

;功能 13H, 设置或取消光标
L_25DF:
                MOV     DS:[54H],AL             ;保存光标状态
                MOV     BP,1
                CALL    L_1AAD                  ;关闭光标
;无论是否建立还是取消光标, 都将光标消除, 当前最显示字符或汉字时, 若光标建立了
;那么又会显示出来
                JMP     L_1A3F

;子程序: 将显示字符写入屏幕缓冲区, 并判断是显示字符还是显示汉字或不显示
;输   入: AL:字符, BL=属性
;输   出: AH=0, 汉字的前一字节, 不用显示
;        AH=1, 一般 ASCII 字符, 显示字符
;        AH=2, 汉字的第 2 个字节, 显示汉字

L_25EB:
                XOR     DX,DX                   ;DX=0
                PUSH    CX
                PUSH    BX
                MOV     BP,DS:[50H]             ;BP=当前光标位置
                TEST    AL,80H                  ;字符是汉字吗?
                JNZ     L_2612                  ;是

;字符处理
                PUSH    AX
                MOV     AX,BP                   ;AX=坐标位置
                CALL    L_2766                  ;取坐标AX在屏幕缓冲区中的相对
                                                ;位移(已转换为每字符一字节)
                POP     AX
                POP     BX
                POP     CX
                MOV     DS:0AFH[DI],AL          ;写屏幕字符缓冲区
                MOV     D_087F[DI],BL           ;写屏幕字符属性缓冲区
                MOV     BYTE PTR D_104F[DI],0   ;写屏幕字符类型缓冲区
                MOV     DI,BP
                MOV     AH,1                    ;AH=1, 表示显示字符
                RETN

;汉字处理
L_2612:
                PUSH    AX
                CALL    L_26F2                  ;对当前列号进行综合判断
```

```
;返回:DL=0 在第 0 列, DL=1 在最后列, DL=2 其它

                POP     AX
                OR      DL,DL
                JZ      L_2633                  ;处理第0列
                DEC     DL
                JZ      L_268A                  ;处理最后列
                CALL    L_26D1                  ;取上列字符
                JNZ     L_262E                  ;不是汉字的前一字节

;上一字节是汉字的前一字节, 结合当前字节刚好组成一个汉字
                MOV     DH,81H                   ;设置上一字符为汉字前一字节标
                                                ;志

;此时, DH=00H 表示当前字符是汉字的前一字节, 不用显示
;       DH=81H 表示上一字符为汉字的前一字节, 当前字符为汉字的后一字节, 要显示
;       DH=82H 表示下一字符为汉字的前一字节, 当前字符为汉字的后一字节, 要显示
L_2626:
                MOV     CX,AX
                POP     BX
                CALL    L_2709                  ;汉字处理
                POP     CX
                RETN

;上一字符非汉字的前一字节, 检查下一字符
L_262E:
                CALL    L_26DD                  ;检查
                JMP     SHORT L_2626
;当前光标位置第 0 列
L_2633:
                CMP     BYTE PTR DS:[50H]+1,0   ;当前屏幕行号=0吗?
                JNE     L_2641                  ;不是
                CMP     BYTE PTR DS:[0A4H],0    ;内部行号=0吗?
                JE      L_2626                  ;是, 作为汉字的前一字节处理

;当前列号=0, 但当前行不是第一行, 还要将上行的最后一字符取出
L_2641:
                PUSH    AX
                MOV     AX,BP
                DEC     AH                      ;行号减1
                MOV     AL,DS:[4AH]
                DEC     AL                      ;AX=上行最后一列
                CALL    L_276E                  ;判断是前半个汉字吗?
                JZ      L_2653                  ;是
                POP     AX
                JMP     SHORT L_2626            ;不是前半个汉字, 直接作前半个
                                                ;汉字处理
;前一字符是前半个汉字
L_2653:
                MOV     AX,BP
                DEC     AH
                MOV     AL,DS:[4AH]
                DEC     AL
                MOV     DS:[71H],AX             ;保存汉字显示坐标
                CALL    L_2766                  ;取坐标在屏幕中的位移
```

```
                POP     CX
                POP     BX
                MOV     DS:0B0H[DI],CL          ;写字符(后半个汉字)
                MOV     D_104F[DI],201H         ;写字符类型(01表示前半个
                                                ;02表示后半个)
                MOV     DS:880H[DI],BL          ;置属性
                MOV     DL,CL
                MOV     DH,DS:0AFH[DI]          ;DX=汉字
                MOV     AX,BP
                CMP     AH,0
                JE      L_2686
                CALL    L_2778                  ;取汉字点阵
                MOV     AH,2                    ;要显示汉字
                POP     CX
                RETN

L_2686:
                XOR     AH,AH                   ;不用显示汉字
                POP     CX
                RETN

;处理最后一列情况
L_268A:
                PUSH    AX
                CALL    L_26D1                  ;读上字符
                JNZ     L_269B                  ;不是前半个汉字
;上一字符是前半个汉字
                MOV     AX,BP
                MOV     DH,81H                  ;置标志
                POP     CX
                POP     BX
                CALL    L_2709                  ;汉字处理
                POP     CX
                RETN

;检查下一行第一个字符是否是汉字的前半个字节
L_269B:
                MOV     AX,BP                   ;AX=光标位置
                CMP     AH,9                    ;当前行是屏幕的最后一行吗?
;这里的处理方式实际上是 CC11 中的处理方法，在 CV26.COM 中，应改为 CMP AH,18H
                JNE     L_26B2                  ;不是
                CMP     BYTE PTR DS:[0A5H],17H  ;当前行>=24行吗?
                JB      L_26B2                  ;不是

;当前坐标是整个显示页的最后一个字符，不用再检查下一字符了
L_26A9:
                POP     CX
                POP     BX
                XOR     DH,DH                   ;不用显示
                CALL    L_2709                  ;汉字处理
                POP     CX
                RETN

L_26B2:
                MOV     AX,BP
```

```
                XOR     AL,AL
                INC     AH                      ;下一行第一列
                CALL    L_276E
                JZ      L_26BF                  ;是汉字前半个字节转
                JMP     SHORT L_26A9
L_26BF:
                POP     CX
                POP     BX
                MOV     DH,82H                  ;显示汉字
                CALL    L_2709                  ;汉字处理
                MOV     CX,BP
                CMP     CH,9                    ;是当前屏幕的最后一行吗?
                JNE     L_26CF                  ;不是
                XOR     AH,AH                   ;最后一个汉字，不用显示
L_26CF:
                POP     CX
                RETN

;子程序: 取当前坐标前一字符，判断是否前半个汉字
;输   入: BP=当前光标坐标
;输   出: ZF=1:是前半个汉字
L_26D1:
                PUSH    BX
                PUSH    AX                      ;保存寄存器
                MOV     AX,BP
                DEC     AL                      ;上一坐标
                CALL    L_276E                  ;判断是前半个汉字吗?
                POP     AX
                POP     BX                      ;恢复寄存器
                RETN

;子程序: 取当前坐标下一字符，判断是否前半个汉字
;输   入: BP=当前光标坐标
;输   出: ZF=1:是前半个汉字
L_26DD:
                PUSH    BX
                PUSH    AX                      ;保存寄存器
                MOV     AX,BP
                INC     AL                      ;下一坐标
                CALL    L_276E                  ;判断是前半个汉字吗?
                JNZ     L_26ED                  ;不是
                MOV     DH,82H                  ;置标志
                JMP     SHORT L_26EF
                NOP
L_26ED:
                XOR     DH,DH
L_26EF:
                POP     AX
                POP     BX                      ;恢复寄存器
                RETN

;子程序: 按光标列号设置 DL 值
;输   入: BP=光标坐标
;         DL=0
;输   出: DL=0:光标在当前行第 0 列
```

```
;           DL=1:光标在当前行最后列
;           DL=2:其它
;           DH=0
L_26F2:
                MOV     AX,BP
                CMP     AL,0                    ;是第0列吗?
                JE      L_2706                  ;是

                MOV     DL,1                    ;DL=1
                MOV     DH,DS:[4AH]
                DEC     DH
                CMP     AL,DH                   ;是最后一列吗?
                JE      L_2706                  ;是

                INC     DL                      ;DL=2
L_2706:
                XOR     DH,DH                   ;DH=0
                RETN

;子程序: 汉字处理
;输    入: BP=坐标位置, DX<80H, 前半个汉字,不用显示. DX=81H,和上一字符组成汉字.
;           DX=82H,和后一字符组成一个汉字
;输    出: AH=0, 未匹配汉字
L_2709:
                MOV     AX,BP                   ;AX=坐标位置
                TEST    DH,80H                  ;要显示吗?
                JNZ     L_2723                  ;是

;未匹配汉字, 不用显示
                CALL    L_2766                  ;取坐标在屏幕缓冲区中的相对位移
                MOV     DS:0AFH[DI],CL          ;将字符保存在屏幕缓冲区
                MOV     D_087F[DI],BL           ;将字符属性保存在屏幕缓冲区
                MOV     BYTE PTR D_104F[DI],80H ;设置字符类型, 80H:未匹配汉字
                XOR     AH,AH
                RETN
L_2723:
                TEST    DH,1                    ;是和上一字符组成一个汉字吗?
                JNZ     L_2748                  ;是

;和下一字符组成汉字
                MOV     DS:[71H],AX             ;保存坐标位置
                CALL    L_2766                  ;取坐标在屏幕缓冲区的相对位移
                MOV     DS:0AFH[DI],CL          ;保存字符
                MOV     D_104F[DI],201H         ;连续一字节,1表示为汉字的前一
                                                ;字节, 2表示为汉字的后一字节
                MOV     D_087F[DI],BL           ;写属性
                MOV     DH,CL
                MOV     DL,DS:0B0H[DI]          ;DX=汉字
L_2742:
                CALL    L_2778                  ;取当前汉字的点阵数据
                MOV     AH,2                    ;要显示汉字
                RETN

;和上一字符组成汉字
L_2748:
```

```
                DEC     AL
                MOV     DS:[71H],AX             ;汉字显示坐标
                CALL    L_2766                  ;取坐标在屏幕缓冲区中的相对位移
                MOV     DS:0B0H[DI],CL          ;保存字符. 当前汉字的后一字节,
                                                ;即当前显示字符
                MOV     D_104F[DI],201H         ;字符类型
                MOV     D_0880[DI],BL           ;属性
                MOV     DL,CL
                MOV     DH,DS:0AFH[DI]          ;DX=汉字
                JMP     SHORT L_2742

;子程序: 计算外部坐标 AX 在屏幕缓冲区的相对位移, 每个字符按一个字节计算
;输   入: AH=行号, AL=列号
;输   出: DI=相对位移
L_2766:
                CALL    L_1CFA                  ;AX=以每字符两字节计算的位移
                SAR     AX,1                    ;除以2, 换算为一字节位移
                MOV     DI,AX                   ;DI=地址
                RETN

;子程序: 当前坐标位置的字符是否为未配对的汉字
;输   入: AH=行号, AL=列号
;输   出: ZF=1:是
L_276E:
                CALL    L_2766                  ;取相对位移
                MOV     AL,BYTE PTR D_104F[DI]  ;取字符类型
                CMP     AL,80H                  ;是未配对的汉字吗?
                RETN

;子程序: 读汉字点阵数据
;输   入: DX=汉字
;输   出: DX:0 为汉字点阵数据首址
L_2778:
                INT     7FH                     ;取点阵
                RETN
CODE            ENDS
                END     L_0100
```

第六章 打 印 驱 动 程 序

第一节 打印机驱动 PRTA.COM

一、使用说明

2.13H支持多种24针打印机，不同的打印机都使用同一个汉字打印驱动程序PRTA.COM，默认的打印机型号也通过运行PRTA.COM进行设置。因此，使用前应先运行一遍PRTA.COM，屏幕显示当前默认的打印机名称。若所显示的打印机与正在使用的打印机不同，则还要运行PRTA.COM设置默认的打印机。

1.设置打印机型号

在DOS提示符下键入：

PRTA ←┘

注意在PRTA后必须有空格。屏幕显示如下：

PRT.COM 汉字打印驱动程序打印机选择：

1 - P1351(P1350)	2 - M2024(M1724)
3 - TH3070(3080)	4 - AR2463
5 - LQ1500(NEC P7)	6 - OKI8320(5320)
7 - M1570	8 - NEC3824
9 - NM9400	

请键入选择号：_

用户按实际使用的打印机进行选择。当选择不同的打印机型号时，程序将选择号写入PRTA.COM中，成为今后默认的打印机型号，直至重新选择设置。

如果PRTA.COM被设置为只读，上述设置无效。

2.装载汉字打印模块

格式[1]

在DOS提示符下键入：

PRTA←┘

屏幕显示默认的打印机名称等信息。

要改变使用的打印机型号，必须先设置好（见1）默认的打印机型号，再重新启动2.13H汉字系统。

格式[2]

PRTA.COM可打印多种字型，每一种字型都有一个字母与之对应，如字母A对应于宋体24x24点阵汉字，也即A型字。字母A－X都对应一种字型，而且，其对应的字型是可以设置的。例如可以设置字母E对应于宋体24x24点阵汉字。设置方法如下：

在DOS提示符下键入：

PRTA xxxx...←┘

x表示当前位置（第一个x表示字母A，依次为B，C...）实际打印时使用的内部字型号，可以是字母A－X、a－p，即当字型设置为A时，实际打印的字型为第一个x所对应的内部字型。

例如：PRTA EDCBa←┘

打印时字型对照如下：

设置字型	实际打印字型
A	E（24x24点阵仿宋体字型）
B	D（48x48点阵宋体字型）
C	C（48x24点阵宋体字型）

D	B（24x48点阵宋体字型）
E	a（24x16点阵宋体字型，由24x24点阵宋体字型压缩得到）
F	F（24x48点阵仿宋体字型）
...	...

这些设置不保存在PRTA.COM中，必须在每次启动时设置。

二、程序功能

本程序主要包括三个部分，即默认打印机型号设置、加载驱动程序和INT 17H代码。

打印机型号设置工作过程如下：

1．在屏幕上显示显示各打印机名称及其代号（1－9）。

2．等待用户输入选择的打印机型号。

3．若输入的是数字1－9，将PRTA.COM文件读入内存，修改打印机型号字节后，存盘。

4．结束运行，返回DOS。

驱动程序加载过程如下：

1．设置INT 17H和INT 7DH。

2．设置INT 7AH、INT 7BH和INT 7CH，指向安全的空中断程序区。

3．按打印机型号设置与打印机硬件有关的几个子程序和一些数据。

4．程序驻留，返回DOS。

驱动程序加载后，还必须运行FILE系列程序（例如FILE16B.COM），这样便可用INT 17H打印汉字和字符了。

三、变量名表

地址	长度	意义
0103h	0Eh	'\213\PRTA.COM',0, 本程序的路径文件名
0112h	2Bh	'打印机驱动程序',0Dh,0Ah
01E0h	1Ah	用户定义的字型表，可以在启动时设置
0200h	1	页长
0201h	1	页间空行
0205h	2	当前行长度(字符数)
0207h	2	前一字节=打印的字符，后一字节打印机状态
0209h	2	当前行列数
020Bh	2	修饰字，上行结束时保存
020Dh	2	修饰字，前一字节:值0-7分别对应8种背景，位4=1表示抽点打印. 后一字节(20E):
		位0:反白打印
		位1:上划线
		位2:下划线
		位3:左旋90度
		位4:右旋90度
		位5:上标字符
		位6:下标字符
		位7:高点阵字(40点阵)
020Fh	1	位0:当前打印行是否要分两次打印
		位1:打印后半行标志
		位6:@控制状态
		位7:ESC控制状态
0210h	1	当前页未打印行数
0211h	2	前一字节=字型(上行打印结束保存值),后一字节=字型汉字的列数
0213h	2	当前字型和列数
0215h	1	行距
0216h	1	两次打印行的打印方向，默认为单向

0217h	2	字符间距
0219h	2	字符间距(上行结束时保存)
021Dh	2	左空列数
021Fh	1	保存前半个汉字，=0表示没有
0220h	2	当前行宽，默认为最大行宽
0222h	2	保存打印字符或汉字
0226h	1	保存修饰字
0229h	1	横向放大(TH3070用)
022Ah	2	控制状态
022Eh	2	暂存数值
0230h	38h	背景数据
0270h	2	02BC,功能0入口地址
0272h	2	01A2,功能1入口地址
0274h	2	01A2,功能2入口地址
0276h	2	02AB,功能3入口地址
0278h	2	027C,功能4入口地址
09FFh	1	打印方向
0A43h	1	默认行距
0A44h	1	16，半行长度，应为24，但由于在换行时进行了计算(x3/2),所以这里是16
0A45h	1	默认打印方向
0A46h	2	最大行宽
0A48h	8	打印机名称
0A80h		设置打印机型号时的菜单

四、程序清单

```
;PRTA.ASM
CODE            SEGMENT
                ASSUME CS:CODE,DS:CODE
                ORG     100H

L_0100:
                JMP     L_1088                  ;转起始程序

D_0103          DB      '\213\PRTA.COM',0       ;本程序的路径文件名
                DB      0
D_0112          DB      '打印机驱动程序',0DH,0Ah
                DB      '研制人:吴晓军    1989.12.8',0DH,0Ah
                DB      '$'                     ;启动时显示的提示信息
                DB      0, 0, 0
D_0140          DB      0
;在程序运行后，0000H-0140H 又作为当前行打印内容存放区，最多可存放 320 个字符(包
;括控制字符).

;子程序: 选择打印颜色，仅对彩色打印机有效
;输    入: AL=彩色号
;输    出: 无
L_0141:
                NOP
                PUSH    AX                      ;保存彩色号
                MOV     AL,1BH
                CALL    L_01A0                  ;向打印机输出字符ESC
                MOV     AL,72H                  ;ESC+72H是设置打印颜色的打印机
                                                ;控制码(对LQ1500系列打印机)
```

```
                CALL    L_01A0                  ;向打印机输出字符72H
                 POP     AX                      ;恢复AL=彩色号
                AND     AL,0FFH
                JMP     SHORT L_01A0            ;转1A0H, 打印AL值, 并直接返回

L_0152:
                INC     BP                      ;修饰符数据位移加1
;打印一列(24点, 分3次)
                CALL    L_0159                  ;打印一字节
                CALL    L_0159                  ;再打印一字节

L_0159:
                LODSB                           ;取一字节
L_015A:
                TEST    BYTE PTR CS:D_020D,7
                JZ      L_0179                  ;无背景
;有背景打印, 取背景数据
                PUSH    AX
                MOV     AL,BYTE PTR CS:D_020D   ;AL=背景号
                AND     AL,7                    ;共1-7种背景
                MOV     AH,8                    ;每种背景有8字节数据, 分别对应
                                                ;字符的第1列到第8列
                MUL     AH                      ;AX=当前背景数据首址
                AND     BP,7                    ;BP=当前列号(除8后的余数)
                NOP
                ADD     BP,AX                   ;BP=当前背景数据位移
                POP     AX
                OR      AL,CS:D_0228[BP]        ;取背景数据.
;因为没有背景号0, 因此, 实际背景数据首址为228H+8=230H.
L_0179:
                TEST    BYTE PTR CS:D_020D+1,1  ;反白打印吗?
                JZ      L_0183                  ;不是
                NOT     AL                      ;反白数据
L_0183:
                TEST    BYTE PTR CS:D_020D,8    ;抽点打印吗?
                JZ      L_0193                  ;不是
                TEST    BP,1                    ;是奇数列吗?
                JZ      L_0193                  ;不是
                MOV     AL,0                    ;不打印数据
;所谓抽点打印, 实际中就是隔点打印, 即打印偶数列, 不打印奇数列
L_0193:
                JMP     SHORT L_01A0            ;转打印

                PUSH    CX
                MOV     CX,8
L_0199:
                SHL     AH,1
                RCR     AL,1
                LOOP    L_0199

                POP     CX

;子程序: 打印一字节内容
;输   入: AL=打印数据
;输   出: AH=[208H]=打印机状态字节
```

```
L_01A0:                 MOV     AH,0
L_01A2:
                        PUSH    DX                      ;保存DX值
                        XOR     DX,DX                   ;DX=0, 表示输出至0号打印机
                        PUSHF
;                       call    far ptr s_F000_EFD2     ;调用原INT 17H打印
;这里的子程序地址, 已在启动时被修改为原 INT 17H 地址
                        DB      9AH,0D2H,0EFH, 00H,0F0H
;这里使用 ROM BIOS 的 INT 17H 打印, 目的是为了使本程序适应各种的计算机.

                        POP     DX                      ;恢复DX值
                        MOV     BYTE PTR CS:D_0208,AH   ;保存打印机状态字节
                        RETN

                        DB      10 DUP (0)              ;没有用

;子程序: 将数字串转化为相应的数值, 每调用一次计算一位, [022EH]是已计算好的数值
;输   入: AL=当前数字
;输   出: CX=计算后的数值
;         CF=1 数字串尚未结束
;         CF=0 数字串已结束
L_01BC:
;判断 AL 是否是数字
                        CMP     AL,30H                  ;小于'0'吗?
                        JB      L_01DA
                        CMP     AL,39H                  ;大于'9'吗?
                        JA      L_01DA

;AL 是数字, 可以计算
                        PUSH    AX                      ;保存数字
                        AND     AL,0FH                  ;将数字转换为数值
                        MOV     CL,AL
                        MOV     CH,0                    ;CX=数值
                        MOV     AX,0AH                  ;AX=10
                        MUL     D_022E                  ;原有的数x10
                        ADD     AX,CX                   ;加当前数值
                        MOV     D_022E,AX               ;保存于022E中
                        POP     AX                      ;恢复AX
                        STC                             ;CF=1,表示数字串没有结束
                        RETN
L_01DA:
                        MOV     CX,D_022E               ;CX=数字串的数值
                        CLC                             ;CF=0,表示数字串已结束, CX为值
                        RETN

D_01E0          DB      'ABCDEFGHIJKLMNOPQRSTUVWXYZ'用户定义的字型表,可以在
                                                        ;启动时修改

                        DB      0, 0, 0, 0, 0, 0

D_0200          DB      42H                             ;页长
D_0201          DB      0                               ;页间空行
                DB      0,0,0
D_0205          DW      0                               ;当前行长度(字符数)
D_0207          DW      0                               ;[207H]=打印的字符, [208H]=
```

```
                                        ;打印机状态
D_0209          DW      0               ;当前行列数

D_020B          DW      0               ;修饰字，上行结束时保存
D_020D          DW      0               ;修饰字，前一字节:值0-7分别对
                                        ;应8种背景，位4=1表示抽点打印.
                                        ;后一字节(20E):
                                        ;位0:反白打印
                                        ;位1:上划线
                                        ;位2:下划线
                                        ;位3:左旋90度
                                        ;位4:右旋90度
                                        ;位5:上标字符
                                        ;位6:下标字符
                                        ;位7:高点阵字(40)
D_020F          DB      0               ;位0:当前打印行是否要分两次打印
                                        ;位1:打印后半行标志
                                        ;位6:@控制状态
                                        ;位7:ESC控制状态
D_0210          DB      0               ;当前页未打印行数
D_0211          DW      1800H           ;[211H]=字型(上行打印结束保存值)
                                        ;[212H]=[211H]字型汉字的列数
D_0213          DW      1800H           ;当前字型和列数
D_0215          DB      14H             ;行距
D_0216          DB      3CH             ;两次打印行的打印方向，默认为
                                        ;单向
D_0217          DW      0               ;字符间距
D_0219          DW      0               ;字符间距(上行结束时保存)
                DB      0, 0
D_021D          DW      0               ;左空列数
D_021F          DB      0               ;保存前半个汉字，=0表示没有
D_0220          DW      990H            ;当前行宽，默认为最大行宽
D_0222          DW      0               ;保存打印字符或汉字
                DB      0, 0
D_0226          DB      0               ;保存修饰字
                DB      0,0
D_0229          DB      0               ;横向放大(TH3070用)
D_022A          DW      0               ;控制状态
                DB      0, 0
D_022E          DW      0               ;暂存数值

;以下是背景数据，每种背景 8 字节

D_0230          DB      88H,0,0,0,88H,0,0,0
                DB      0AAH,0,80H,0,80H,0,80H,0
                DB      80H,0,80H,0,80H,0,80H,0
                DB      0AAH,0,0,0,0,0,0,0
                DB      10H,20H,40H,80H,1,2,4,8
                DB      80H,40H,20H,10H,8H,4,2,1
                DB      90H,60H,60H,90H,9,6,6,9
                DB      0,0,6CH,2,0,0,0,0

;各功能入口地址表
D_0270          DW      OFFSET L_02BC   ;功能0入口地址
D_0272          DW      OFFSET L_01A2   ;功能1入口地址
```

```
D_0274          DW      OFFSET L_01A2           ;功能2入口地址
D_0276          DW      OFFSET L_02AB           ;功能3入口地址
D_0278          DW      OFFSET L_027C           ;功能4入口地址

                DB      0, 0                    ;没有用

;子程序: 设置行距
;输   入: AL=行距
;输   出: 无
L_027C:
                MOV     D_0215,AL               ;设置行距
                RETN

;INT 17H
;AH=0, 打印字符
;输入: AL=打印字符
;输出: AH=打印机状态字节

;AH=1, 初始化打印机
;输入: DX=打印机号
;输出: AH=打印机状态字节

;AH=2, 取打印机状态字节
;输入: DX=打印机号
;输出: AH=打印机状态字

;AH=3, 设置打印行宽
;输入: AL=字符数, 每字符 12 列
;输出: 无

;AH=4, 设置打印行距
;输入: AL=行距
L_0280:
                STI                             ;允许中断

                PUSH    DS
                PUSH    ES
                PUSH    BP
                PUSH    DI
                PUSH    SI
                PUSH    DX
                PUSH    CX
                PUSH    BX                      ;保存寄存器

                PUSH    CS
                POP     DS                      ;DS=CS
                PUSH    CS
                POP     ES                      ;ES=CS

                CMP     AH,4                    ;功能号>4吗?
                JA      L_02A2                  ;是, 无效的功能号

                MOV     D_0207,AX               ;保存输入数据
                MOV     BL,AH
                MOV     BH,0                    ;BX=功能号
```

```
                SHL     BX,1                    ;BX乘以2, 入口表中的相对位移
                CALL    WORD PTR D_0270[ BX]    ;执行相应功能
                MOV     AX,D_0207               ;取打印字符和状态字节

L_02A2:
                POP     BX
                POP     CX
                POP     DX
                POP     SI
                POP     DI
                POP     BP
                POP     ES
                POP     DS                      ;恢复寄存器

                IRET                            ;中断返回

;子程序: 功能 3, 设置打印行宽
;输   入: AX=字符数
;输   出: 无
L_02AB:
                MOV     CL,0CH                  ;CL=12, 每字符列数
                MUL     CL                      ;乘以总字符数=行宽(单位:点)
                CMP     AX,D_0A46               ;超过最大行宽吗?
                JBE     L_02B8                  ;没有
                MOV     AX,D_0A46               ;AX=最大行宽
L_02B8:
                MOV     D_0220,AX               ;设置行宽
                RETN

;子程序: 功能 0, 打印字符
;输   入: AL=字符
;输   出: AH=打印机状态字节
L_02BC:
                CMP     D_021F,0                ;已打印过前半个汉字吗?
                JE      L_02E5                  ;没有

;已打印了前半个汉字, 下面应该是后半个汉字
                MOV     AH,D_021F               ;AH=前半个汉字
                MOV     D_021F,0                ;清前半个汉字
L_02CC:
                MOV     DI,D_0205               ;取当前行长度
                CMP     DI,OFFSET D_0140        ;超过320个了
                JA      L_02E4                  ;是
                CMP     AL,0A0H                 ;当前字节是汉字(应该是)
                JA      L_02DD                  ;是

;后一字节为一般的 ASCII 字符
                MOV     AX,2020H                ;不能确定汉字, 用空格代替
L_02DD:
                XCHG    AL,AH                   ;交换AL和AH值, 即高字节在前,
                                                ;低字节在后
                STOSW                           ;保存
                MOV     D_0205,DI               ;修改当前行长度
L_02E4:
                RETN
```

```
L_02E5:
                MOV     BX,OFFSET D_020F        ;取控制状态字节
                TEST    BYTE PTR [BX],40H       ;已进入@控制状态吗?('@'或ESC I)
                JZ      L_033F                  ;没有

;已进入@控制状态
                CMP     AL,3EH                  ;是设置双向打印吗?
                JNE     L_0306                  ;不是
L_02F1:
                CALL    L_0300                  ;设置打印机方向
L_02F4:
                DEC     BYTE PTR D_0205         ;删除无用的控制字符
L_02F8:
                DEC     BYTE PTR D_0205         ;删除无用的控制字符
L_02FC:
                AND     BYTE PTR [BX],3         ;退出控制状态
                RETN

;子程序: 设置打印方向
;输   入: AL='>'双向, AL='<'单向
;输   出: 无
L_0300:
                MOV     D_0216,AL               ;修改打印方向单元
                JMP     L_0A27

L_0306:
                CMP     AL,3CH                  ;是'<'吗?
                JE      L_02F1                  ;是转设置打印方向

                CMP     AL,60H                  ;是'`'吗?
                JA      L_0322                  ;>, 可能是小写字母
                JNZ     L_031A                  ;不是

                MOV     AL,3                    ;ESC+'I', 恢复'功能
L_0312:
                MOV     BYTE PTR DS:[4E9H],AL   ;修改程序
                MOV     BYTE PTR DS:[7C3H],AL   ;修改程序
;当 AL=3 时, 使得对'`'判断有效, 即开启'功能. 当 AL=0 时, 使得对'`'判断无效, 即取
;消'功能.
                JMP     SHORT L_02F4
L_031A:
                CMP     AL,27H                  ;是'"'吗?
                JNE     L_0328                  ;不是

                MOV     AL,0                    ;ESC+"I", 取消'`'功能
                JMP     SHORT L_0312

;选择字型
L_0322:
                CMP     AL,7AH                  ;>'z'吗?
                JA      L_02F4                  ;是, 无效的字型指定
                JMP     SHORT L_0334            ;小写字母
L_0328:
                CMP     AL,41H                  ;<'A'吗?
                JB      L_02F4                  ;是, 无效的字型指定
```

```
                CMP     AL,5AH                  ;>'Z'吗?
                JA      L_02F4                  ;是, 无效的字型指定

                MOV     BX,19FH                 ;BX至少是字符A, 所以表起始地址
                                                ;应为BX+41H=1E0H, 指向字型表
                XLAT                            ;查找对应的内部字型, AL=内部
                                                ;字型
L_0334:
                CALL    L_03D7                  ;保存AL到行缓冲区
                CALL    L_0908                  ;设置字型
                MOV     BX,OFFSET D_020F        ;BX=控制状态字节地址
                JMP     SHORT L_02FC

L_033F:
                TEST    BYTE PTR [BX],80H       ;已有ESC控制字符输入吗?
                JNZ     L_0347                  ;是
                JMP     L_04DC                  ;没有, 转其它处理

;已有 ESC 控制字符输入

L_0347:
                AND     AL,5FH                  ;AL转换为大写字符
                CMP     AL,49H                  ;是'I'吗?(判断是否ESC+'I'控制
                                                ;序列
                JNE     L_0354                  ;不是
                CALL    L_03D7                  ;保存'I'
L_0350:
                OR      BYTE PTR [BX],40H       ;设置@控制状态
;ESC I相当于'@
                RETN

L_0354:
                CMP     AL,57H                  ;是'W'吗?
                JNE     L_02F8                  ;不是

;ESC+'W', 屏幕打印
                INT     5                       ;屏幕打印
                JMP     SHORT L_02F8

L_035C:
                CMP     AL,1BH                  ;是ESC吗?
                JNE     L_0368                  ;不是

;进入 ESC 控制状态
                OR      D_020F,80H              ;设置ESC控制状态
                JMP     SHORT L_03D7
                NOP

L_0368:
                CMP     AL,0A0H                 ;是汉字吗?
                JA      L_03E7                  ;是, 转处理汉字
                AND     AL,7FH                  ;屏幕高位, 80H-A0H改为0-20H

                CMP     AL,0AH                  ;是换行字符吗?
                JNE     L_0382                  ;不是
```

```
;换行处理
L_0372:
                MOV     BYTE PTR D_0207,AL      ;保存换行或回车字符
                CMP     D_0205,0                ;缓冲区中有字符吗?
                JE      L_037F                  ;没有，直接打印换行或回车
                JMP     L_0453                  ;转处理换行或回车(行长>0)
L_037F:
                JMP     L_0986                  ;转处理换行或回车(行长=0)

L_0382:
                CMP     AL,0DH                  ;是回车吗?
                JE      L_0372                  ;转回车处理

                CMP     AL,0CH                  ;是换页字符吗?
                JNE     L_03AD                  ;不是

;处理换页
                MOV     AL,D_0210               ;取当前页未打印行数
                CMP     AL,0                    ;已打印完?
                JE      L_03A1                  ;是，不用换页了

;当前页尚未打印完，输出回车直到打印完毕

                MOV     AH,0
                MOV     CX,AX                   ;共打印CX行
L_0395:
                MOV     AL,0DH                  ;回车字符
                CALL    L_01A0                  ;打印
                MOV     AL,0AH                  ;换行字符
                CALL    L_01A0                  ;打印
                LOOP    L_0395                  ;打印剩余空行

L_03A1:
                MOV     AL,D_0200               ;AL=页长
                MOV     D_0210,AL               ;置新页中未打印行数
                RETN

                DB      00H, 00H, 00H, 00H      ;没有用
                NOP

L_03AD:
                CMP     AL,9                    ;是TAB字符吗?
                JNE     L_03C4                  ;不是

;处理 TAB 字符，用'空格'打印，直到到制表位置
                MOV     AL,20H                  ;AL=' '
L_03B3:
                TEST    D_0205,7                ;已到制表位置了吗?
                JZ      L_03C3                  ;是
                CALL    L_03D7                  ;输入空格
                CALL    L_03EA                  ;调整行列数
                JMP     SHORT L_03B3
L_03C3:
                RETN
```

```
L_03C4:                 TEST    BYTE PTR D_020E,80H     ;在全角字符打印状态吗?
                JZ      L_03D4                          ;不是

;将半角字符转换为全角字符
                MOV     AH,0A3H                         ;全角字符区号
                OR      AL,80H                          ;加80H, 转换为相应的汉字符号
                CALL    L_02CC                          ;保存汉字
                JMP     SHORT L_03EA

L_03D4:
                CALL    L_03EA                          ;调整行列数

;子程序: 将字符保存到行缓冲区
;输   入: AL=字符
;输   出: 无
L_03D7:
                MOV     DI,D_0205                       ;取行长
                CMP     DI,OFFSET D_0140                ;缓冲区已满?
                JA      L_03E6                          ;是
                STOSB                                   ;保存字符
                MOV     D_0205,DI                       ;修改缓冲区长度

L_03E6:
                RETN

;AL 是汉字的前一字节, 以整个汉字调整当前行列数, 打印后半个汉字时不再调整
L_03E7:
                MOV     D_021F,AL                       ;保存前半个汉字

;子程序: 根据当前字符占用列数, 调整当前行列数
;输   入: 无
;输   出: 无
L_03EA:
                MOV     CX,D_0209                       ;取当前行列数
                MOV     BL,BYTE PTR D_0214              ;取当前字型汉字的列数
                MOV     BH,0                            ;BX=列数
                CMP     AL,0A0H                         ;是汉字吗?
                JA      L_0412                          ;是
                SHR     BX,1                            ;西文字符的列数是汉字的一半
                JMP     SHORT L_0416

                DB      22 DUP (0)

L_0412:
                ADD     BX,D_0217                       ;加字间距
;由于汉字的字间距等于字符的的两倍, 因此这里先加一次, 下面还要加一次

L_0416:
                ADD     BX,D_0217                       ;加字间距
                TEST    BYTE PTR D_0213,1               ;是横扩字型吗?
                JZ      L_0423                          ;不是
                ADD     BX,BX                           ;横扩字型, 再加一倍. 实际上有
                                                        ;些字型并非倍扩
L_0423:
                ADD     CX,BX                           ;CX=调整后的行列数
```

```
                CMP     CX,D_0220               ;超过最大行宽了吗?
                JA      L_042F                  ;是
                MOV     BX,CX
                JMP     SHORT L_0436

;行太长
L_042F:
                PUSH    BX
                PUSH    AX
                CALL    L_0453                  ;先将缓冲区的内容打印
                POP     AX
                POP     BX
L_0436:
                MOV     D_0209,BX               ;修改当前行列数单元
                RETN


;子程序: 当前行打印前, 设置基本参数. 这些数据是在上行打印结束时保存的. 因为在
;       没有接收到回车或换行时, 打印的字符仅保存在行缓冲区中, 不被直接打印,
;       在这过程中, 以下这些数据将被修改, 所以这里要恢复上行结束值.
;输   入: 无
;输   出: 无
L_043B:
                MOV     BYTE PTR D_022A,0       ;清除'标志
                MOV     AX,D_020B               ;取修饰字. 由上行打印结束时保
                                                ;存, 即继承字型修饰字
                MOV     D_020D,AX               ;设置起始修饰字
                MOV     AX,D_0211               ;取字型, 由上行打印结束时保存
                MOV     D_0213,AX               ;设置起始字型
                MOV     AX,D_0219               ;取字符间距. 由上行打印结束时
                                                ;保存
                MOV     D_0217,AX               ;设置起始字符间距
                RETN

;子程序: 回车或换行, 即打印行缓冲区中的字符(必定存在), 然后回车或换行
;输   入: AL=回车或换行
;输   出: 无
L_0453:
                CALL    L_043B                  ;恢复上行打印结束时的基本参数
                MOV     DI,D_0205               ;DI=当前行长度
                TEST    D_020F,1                ;是打印上半行吗?
                JZ      L_0486                  ;不是, 打印下半行或单行字

;单行打印指当前行中所有字符的垂直点数 24 都小于或等于 24, 因此, 一次即可打印. 超
;过 24 的必须分两次打印, 即分前半行和后半行打印
                CALL    L_0963                  ;设置图形打印方式并打印左边空
                                                ;格
                XOR     BP,BP                   ;背景修饰数据位移=0
L_0466:
                MOV     AH,BYTE PTR D_0213      ;AH=取字型
                LODSB                           ;取字符
                CMP     AL,0A0H                 ;是汉字吗?
                JAE     L_0474                  ;是
                CALL    L_07B9                  ;打印字符
                JMP     SHORT L_047C L_0474:
```

```
                MOV     DH,AL                   ;DH=汉字的前一字节
                LODSB                           ;取后一字节
                MOV     DL,AL                   ;DX=汉字
                CALL    L_06B3                  ;打印汉字
L_047C:
                CMP     SI,DI                   ;打印完毕?
                JB      L_0466                  ;没有
                CALL    L_097E                  ;设置打印机为单向打印，打印机
                                                ;走纸半行(指向下半行位置)
;为了提高打印质量，需两次才能打印汉字或字符均单向打印. 也可以用'@<'设置双向
                CALL    L_043B                  ;恢复上行打印结束时的基本参数

L_0486:
                CALL    L_0963                  ;设置图形打印方式并打印左边空
                                                ;格
                XOR     BP,BP                   ;背景修饰数据位移=0
                OR      D_020F,2                ;设置下半行标志
L_0490:
                MOV     AH,BYTE PTR D_0213      ;AH=当前字型汉字的列数
                LODSB
                CMP     AL,0A0H                 ;是汉字吗?
                JAE     L_049E                  ;是
                CALL    L_07B9                  ;打印字符
                JMP     SHORT L_04A6
L_049E:
                MOV     DH,AL
                LODSB
                MOV     DL,AL                   ;DX=汉字
                CALL    L_06B8                  ;打印汉字
L_04A6:
                CMP     SI,DI                   ;打印完毕?
                JB      L_0490                  ;没有

;新行设置和保存基本参数
                XOR     AX,AX                   ;AX=0
                MOV     D_0205,AX               ;行长度=0
                MOV     D_0209,AX               ;当前行列数=0
                MOV     D_022A,AX               ;清除'功能状态
                MOV     AX,D_020D
                MOV     D_020B,AX               ;保存修饰字
                MOV     AX,D_0213
                MOV     D_0211,AX               ;保存当前字型汉字的列数
                MOV     AX,D_0217
                MOV     D_0219,AX               ;保存字符间距
                CALL    L_0986                  ;打印回车或换行

                TEST    BYTE PTR D_0213,82H     ;是高点阵字型吗?
                JNZ     L_04D6                  ;是，如果是40点阵字型，不进行
                                                ;纵扩每行也要打印两次，所以要
                                                ;保留打印两次标志，因为下一行
                                                ;可能不再设置字型
                AND     D_020F,0FEH             ;清打印两次标志
L_04D6:
                AND     D_020F,0FDH             ;清后打印半行标志
                RETN
```

```
L_04DC:
                MOV     SI,22BH

;22BH 是控制序列引导字节，如打印'@'后(在'控制字符后)，22BH='@'. 检查该字节即可
;知道当前是否在进行相应的功能设置

                CMP     BYTE PTR D_022A,60H     ;已输入过控制字符'了吗?
                JE      L_04F3                  ;是
                CMP     AL,60H                  ;是'"吗?
                JE      L_04ED                  ;是
                JMP     L_035C

;进入'状态
L_04ED:
                MOV     BYTE PTR D_022A,AL      ;22A='"
                JMP     L_03D7

;已进入'状态
L_04F3:
                CMP     BYTE PTR [SI],40H       ;是在'@'状态下吗?
                JNE     L_0509                  ;不是

;设置字型
                CALL    L_03D7                  ;保存AL
                CMP     AL,3AH                  ;是数字吗?
                JAE     L_0502                  ;不是

;ESC+'0'-'9', 设置打印颜色
                CALL    L_0141                  ;选择打印颜色
L_0502:
                CALL    L_0908                  ;设置字型
L_0505:
                MOV     BYTE PTR [SI],0         ;当前控制序列结束
L_0508:
                RETN

L_0509:
                CMP     BYTE PTR [SI],23H       ;是在'#'状态吗?
                JNE     L_0527                  ;不是

;#, 置页长
                CALL    L_01BC                  ;取页长
;该子程序将一数字串转换为相应的数值，返回 CF=0，表示数字串已结束，CX 便是数字串
;的数值. 返回 CF=0，表示数字串尚未结束，还要继续计算. 返回的数值也是没有用的.

                JC      L_0508                  ;数字尚未结束
                MOV     D_0200,CL               ;设置页长
                MOV     D_0210,CL               ;设置本页未打印的行数
                NOP
L_051C:
                MOV     WORD PTR [SI+3],0
                MOV     BYTE PTR [SI],0         ;当前控制序列结束
                JMP     L_05F1
L_0527:
```

```
                CMP     BYTE PTR [SI],2AH       ;是在'*'状态吗?
                JNE     L_0537                  ;不是

;*, 置页间空行
                CALL    L_01BC                  ;取数值
                JC      L_0508                  ;尚未结束
                MOV     BYTE PTR D_0201,CL      ;保存页间空行
                JMP     SHORT L_051C

L_0537:
                CMP     BYTE PTR [SI],26H       ;是在'&'状态吗?
                JNE     L_0547                  ;不是

;&, 置行距
                CALL    L_01BC                  ;取行距
                JC      L_0508                  ;尚未结束
                MOV     D_0215,CL               ;保存行距
                JMP     SHORT L_051C

L_0547:
                CMP     BYTE PTR [SI],5BH       ;是在'['状态吗?
                JNE     L_055C                  ;不是

;[, 置左边空格数
                CALL    L_01BC                  ;取空格数
                JC      L_0508                  ;尚未结束
                PUSH    AX                      ;保存AX
                MOV     AL,0CH                  ;每个空格占12列
                MUL     CL                      ;剩以空格数
                MOV     D_021D,AX               ;保存左空列数
                POP     AX                      ;恢复AX
                JMP     SHORT L_051C

L_055C:
                CMP     BYTE PTR [SI],5DH       ;是在']'状态吗?
                JNE     L_056F                  ;不是

;], 置行宽(即右边界)
                CALL    L_01BC                  ;取行宽
                JC      L_0508                  ;尚未结束
                PUSH    AX                      ;保存AX
                MOV     AL,CL                   ;AL=行宽
                CALL    L_02AB                  ;调用子程序设置行宽
                POP     AX                      ;恢复AX
                JMP     SHORT L_051C

L_056F:
                CMP     BYTE PTR [SI],24H       ;是在'$'状态吗?
                JNE     L_057B                  ;不是

;$, 将$后的字符直接输出到打印机, 直到再打印$为止
                CMP     AL,24H                  ;是结束的'$'吗?
                JE      L_0505                  ;是, 结束
                JMP     L_01A0                  ;直接将AL输出到打印机
L_057B:
```

```
                CMP     BYTE PTR [SI],5EH       ;是在'^'状态吗?
                JNE     L_058E                  ;不是
                CALL    L_01BC                  ;取字距
                JNC     L_0588                  ;取到转
                JMP     L_03D7                  ;尚未结束
L_0588:
                MOV     D_0217,CX               ;保存字距
                JMP     SHORT L_051C

L_058E:
                CMP     BYTE PTR [SI],7EH       ;是在'~'状态吗?
                JNE     L_05A2                  ;不是

;~, 输出空列
                CALL    L_01BC                  ;取空列数
                JC      L_059F                  ;尚未结束
                ADD     D_0209,CX               ;直接调整当前行列数
                JMP     L_051C
L_059F:
                JMP     L_03D7

L_05A2:
                CMP     BYTE PTR [SI],7CH       ;是在'|'状态吗?
                JNE     L_05E6                  ;不是

;|, 水平定位
                CALL    L_01BC                  ;取坐标
                JNC     L_05AD                  ;已在CX中
                RETN

L_05AD:
                CMP     CX,D_0209               ;在当前行位置之前吗?
                JBE     L_05E3                  ;是, 无效
                PUSH    AX                      ;保存AX
                MOV     AL,7EH                  ;转换为'~'控制序列
                CALL    L_03D7                  ;保存~
                MOV     AX,CX                   ;AX=坐标
                SUB     AX,D_0209               ;减当前坐标=输出空列数
                MOV     D_0209,CX               ;修改当前行列数
;下面将空列数转换为数字输入到行缓冲区中, 即仿造~控制序列

                MOV     CX,40AH                 ;最多4位
                MOV     BX,D_0205               ;当前行长度
                MOV     DI,3                    ;从个位数开始
L_05CD:
                DIV     CL                      ;除以10
                OR      AH,30H                  ;AH转换为数字
                MOV     [BX+DI],AH              ;写入缓冲区
                MOV     AH,0
                DEC     DI
                DEC     CH
                JNZ     L_05CD                  ;上一个
                POP     AX                      ;恢复AX
                ADD     BX,4                    ;当前行长度加4
                MOV     D_0205,BX               ;修改当前行长度
```

```
L_05E3:
                JMP     L_051C

L_05E6:
                CMP     BYTE PTR [SI],25H       ;是在'%'状态吗?
                JNE     L_05F1                  ;不是

;%, 置打印背景
                CALL    L_03D7                  ;直接保存在缓冲区中即可
                JMP     L_0505

;控制序列仅输入了'字符, 还没有得到具体的控制命令, 下面继续判断命令
L_05F1:
                CMP     AL,60H                  ;是'`'吗?
                JNE     L_05FD                  ;不是

;结束'状态
                MOV     BYTE PTR D_022A,0       ;清除'状态字

;将当前字符保存于行缓冲区中, 以便于打印时再度判断
L_05FA:
                JMP     L_03D7

L_05FD:
                CMP     AL,3BH                  ;是';'吗?
                JE      L_05FA                  ;是
                CMP     AL,7BH                  ;是'{'吗?
                JNE     L_060C                  ;不是

;{, 设置横向放大
                MOV     AL,0EH                  ;14即CTRL-N
                CALL    L_01A0                  ;打印控制字符
;该命令仅对 TH3070 有效, 实现横向放大
                JMP     SHORT L_0612
L_060C:
                CMP     AL,7DH                  ;是'}'吗?
                JNE     L_0616                  ;不是

;}, 结束横向放大
                MOV     AL,0FH
L_0612:
                MOV     D_0229,AL
                RETN

L_0616:
                CMP     AL,23H                  ;是'#'吗?
                JNE     L_061D                  ;不是
L_061A:
                MOV     [SI],AL                 ;保存控制命令, 但不保存于行缓
                                                ;冲区中
                RETN
L_061D:
                CMP     AL,24H                  ;'$',将两$之间字符直接输出到
                                                ;打印机
                JE      L_061A
```

```
                CMP     AL,26H          ;'&',置行距
                JE      L_061A
                CMP     AL,2AH          ;'*',日页间空行
                JE      L_061A
                CMP     AL,5BH          ;'[',置左边空格
                JE      L_061A
                CMP     AL,5DH          ;']',置行宽
                JE      L_061A
                CMP     AL,7Ch          ;'|',水平定位
                JE      L_061A
                CMP     AL,25H          ;'%',置打印背景
                JE      L_0649
                CMP     AL,3BH          ;';',置抽点打印
                JE      L_0649
                CMP     AL,40H          ;'@',设置字型或打印方向
                JE      L_0649
                CMP     AL,5EH          ;'^',置字符间距
                JE      L_0649
                CMP     AL,7EH          ;'~',输出空列
                JNE     L_064E
L_0649:
                MOV     [SI],AL         ;保存控制命令
                JMP     L_03D7          ;同时保存于行缓冲区中
L_064E:
                CMP     AL,3EH          ;'>',置双向打印
L_0650:
                JNE     L_0656          ;不是
L_0652:
                CALL    L_0300          ;设置打印方向
                RETN
L_0656:
                CMP     AL,3CH          ;'<',置单向打印
                JE      L_0652
                JMP     SHORT L_0665

                DB      0, 0

L_065E:
                JMP     L_03D7

                DB      0, 0, 0, 0

L_0665:
                CMP     AL,5CH          ;'\',置反白打印
                JE      L_065E
                CMP     AL,2FH          ;'/',置上划线打印
                JE      L_065E
                CMP     AL,5FH          ;'_',置下划线打印
                JE      L_065E
                CMP     AL,3DH          ;'=',置正常打印
                JE      L_065E
                CMP     AL,28H          ;'(',置左旋打印
                JE      L_065E
                CMP     AL,29H          ;')',置右旋打印
                JE L_065E
```

```
                CMP     AL,3FH                  ;'?',结束旋转打印
                JE      L_065E
                CMP     AL,2BH                  ;'+',置上标字符
                JE      L_0699
                CMP     AL,2DH                  ;'-',置下标字符
                JE      L_0699
                CMP     AL,3AH                  ;':',置全角字符打印
                JNE     L_0695
                OR      BYTE PTR D_020E,80H     ;置全角字符打印标志
                JMP     L_03D7
L_0695:
                CMP     AL,21H                  ;'!',置正常字符打印(取消上标
                                                ;或下标打印)
                JNE     L_06A1
L_0699:
                AND     BYTE PTR D_020E,1FH
                JMP     L_03D7
L_06A1:
                CMP     AL,2EH                  ;'.',暂停打印
                JNE     L_06B0
                MOV     AX,0E07H
                INT     10H                     ;响玲
                MOV     AH,0
                INT     16H                     ;等待用户按键
                RETN
                NOP
L_06B0:
                JMP     L_0368

;子程序: 打印前半行字符
;输   入: DX=汉字(若 DH=0,DL=字符), AH=字型
;输   出: 无
L_06B3:
                TEST    AH,2                    ;是纵扩字吗?
                JZ      L_0735                  ;不是

;打印半行或单行内容
L_06B8:
                PUSH    SI                      ;保存SI
                MOV     D_0222,DX               ;保存字符或汉字
                MOV     BX,WORD PTR D_020D+1    ;BX=修饰字
                MOV     D_0226,AH               ;保存字型
                TEST    AH,80H                  ;是高点阵字(40x40)
                JZ      L_06E5                  ;不是
                OR      DH,DH                   ;是汉字吗?
                JNZ     L_06D3                  ;是
;高点阵下不能打印 ASCII 字符, 所有 ASCII 字符都要转换为相应的全角字符
                MOV     DH,0AAH                 ;全角字符区号
L_06D0:
                OR      DL,80H                  ;DX=相应的全角字符
L_06D3:
                INT     7CH                     ;取高点阵汉字的点阵数据
                JMP     SHORT L_06F0
                DB      14 DUP (0)              ;没有用
```

```
L_06E5:
                TEST    AH,10H                  ;是16点阵字型
                JNZ     L_06EE                  ;是
                INT     7BH                     ;取24点阵数据
                JMP     SHORT L_06F0
L_06EE:
                INT     7AH                     ;取16点阵数据

;此时 DS:SI 已是点阵数据首址
L_06F0:
                MOV     BX,AX                   ;BH=字型
                TEST    AH,2                    ;是纵扩字吗?
;注意，如果是隔点扩，则中断返回时，纵扩或横扩标志位均已被清除，因为这些取得的
;数据已扩充好，不用再扩了
                JZ      L_070F                  ;不是
;纵扩打印
L_06F7:
                PUSH    SI                      ;保存点阵数据地址
                CALL    L_0794                  ;打印，要进行纵扩
                POP     SI                      ;恢复点阵数据地址
                TEST    BH,1                    ;要横扩吗?
                JNZ     L_0708                  ;是
                ADD     SI,3                    ;SI=下列数据地址
                LOOP    L_06F7                  ;打印CX列
                JMP     SHORT L_0725
L_0708:
                CALL    L_0794                  ;再打印一次，此时SI没有保存，
                                                ;返回时SI已指向下列数据地址
                LOOP    L_06F7                  ;打印CX次
                JMP     SHORT L_0725

;非纵扩打印
L_070F:
                PUSH    SI                      ;保存点阵数据地址
                CALL    L_0152                  ;打印一列
                POP     SI                      ;恢复点阵数据地址
                TEST    BH,1                    ;要横扩吗?
                JNZ     L_0720                  ;要
                ADD     SI,3                    ;SI=下一列点阵数据地址
                LOOP    L_070F                  ;打印CX列
                JMP     SHORT L_0725
L_0720:
                CALL    L_0152                  ;再打印一次，横扩
                LOOP    L_070F                  ;打印CX次

L_0725:
                MOV     CX,CS:D_0217            ;CX=字间距
                OR      CX,CX                   ;没有字间距吗?
                JZ      L_0731                  ;是
                CALL    L_08B8                  ;打印字间距
L_0731:
                POP     SI                      ;恢复SI
                PUSH    CS
                POP     DS                      ;恢复DS=CS
                RETN
```

```
L_0735:
                TEST    AH,80H                  ;是高点阵数据
                JZ      L_073D                  ;不是
                JMP     L_06B8
;不是高点阵字型，也不是纵扩字，一行不可能分两次打印. 也即是错误的调用，这里打
;印空白数据
L_073D:
                MOV     CL,BYTE PTR D_0213+1    ;取当前字型汉字的列数
                MOV     CH,0
                SHR     CX,1                    ;列数/2=字符列数
                OR      DH,DH
                JZ      L_074F                  ;非汉字
                SHL     CX,1                    ;汉字列数x2
                ADD     CX,D_0217               ;字间距加倍
L_074F:
                ADD     CX,D_0217               ;加字间距
                NOP
                TEST    AH,1                    ;要横扩吗?
                JZ      L_075B                  ;不要
                ADD     CX,CX                   ;列数加倍

;打印字间距
L_075B:
                MOV     AL,0
                CALL    L_015A
                MOV     AL,0
                CALL    L_015A
                MOV     AL,0
                CALL    L_015A                  ;打印24个空白点
                INC     BP                      ;修饰符数据位移加1
                LOOP    L_075B                  ;打印CX次
                RETN

;子程序: 纵扩时，打印 8 点
;输   入: AL=原数据(高 4 位)
;输   出: 无
L_076E:
                MOV     AH,AL                   ;AH=数据
                MOV     DH,4                    ;4点
L_0772:
                RCL     AH,1                    ;AH的最高位->CF
                PUSHF                           ;保存CF
                RCL     AL,1                    ;将CF移入AL
                POPF                            ;恢复CF
                RCL     AL,1                    ;再次将CF移入AL，扩大
                DEC     DH                      ;计数减1
                JNZ     L_0772
                JMP     L_015A                  ;转打印AL

;子程序: 纵扩时，打印 8 点
;输   入: AL=原数据(低 4 位)
;输   出: 无 L_0781:
                MOV     AH,AL                   ;AH=数据
                MOV     DH,4                    ;4点
```

```
L_0785:
                RCR     AH,1                    ;将AH的最低位->CF
                PUSHF                           ;保存CF
                RCR     AL,1                    ;将CF移入AL
                POPF                            ;恢复CF
                RCR     AL,1                    ;再次将CF移入AL，扩大
                DEC     DH                      ;计数减1
                JNZ     L_0785
                JMP     L_015A

;子程序：纵扩时，打印一列数据
;输    入：SI=点阵数据地址
;输    出：无
L_0794:
                INC     BP                      ;修饰符数据位移加1

                LODSB                           ;取第一字节
                TEST    CS:D_020F,2             ;是打印后半行吗?
                JNZ     L_07AC                  ;是
;打印前半行，共打印24点
                PUSH    AX
                CALL    L_076E                  ;将AL高4位放大打印8点
                POP     AX
                CALL    L_0781                  ;将AL的低4位放大打印8点
                LODSB
                CALL    L_076E                  ;将AL的高4位放大打印8点
                LODSB
                RETN

;打印后半行，共打印24点
L_07AC:
                LODSB                           ;前12点数据已被打印，这里要打
                                                ;印后12点(放大后为24点)
                CALL    L_0781                  ;将AL的低4位放大打印8点
                LODSB
                PUSH    AX
                CALL    L_076E                  ;将AL的高4位放大打印8点
                POP     AX
                JMP     SHORT L_0781            ;再打印后8点

                NOP

;子程序：打印字符，并处理控制字符
;输    入：AL=字符
;输    出：无
L_07B9:
                CMP     BYTE PTR D_022A,60H     ;在"'状态吗?
                JE      L_07CB                  ;是

                CMP     AL,60H                  ;是"'吗?
                JE      L_07C7                  ;是
                JMP     L_089A L_07C7:
                MOV     BYTE PTR D_022A,AL      ;进入'功能状态
                RETN
```

```
L_07CB:
                CMP     AL,60H                  ;是"`"吗?
                JNE     L_07D3                  ;不是
                MOV     AL,0                    ;结束'功能状态
                JMP     SHORT L_07C7
L_07D3:
                CMP     AL,5EH                  ;是"^"吗?
                JNE     L_07E8                  ;不是
;^, 置字符间距
L_07D7:
                LODSB
                CALL    L_01BC                  ;取间距
                JC      L_07D7
                MOV     D_0217,CX               ;修改间距
L_07E1:
                XOR     AX,AX
                MOV     D_022E,AX               ;清除中间数据
                DEC     SI
                RETN

L_07E8:
                CMP     AL,7EH                  ;是'~'吗?
                JNE     L_07F7                  ;不是
;~, 输出空列
L_07EC:
                LODSB
                CALL    L_01BC                  ;取空列数
                JC      L_07EC
                CALL    L_075B                  ;打印空列
                JMP     SHORT L_07E1

L_07F7:
                CMP     AL,40H                  ;是'@'吗?
                JNE     L_0810                  ;不是
;@, 设置打印方向或字型
                LODSB                           ;取字符
                JMP     L_08FC
                NOP
                DB      16 DUP (0)              ;没有用

L_0810:
                CMP     AL,25H                  ;是'%'吗?
                JNE     L_081B                  ;不是
;%, 置打印背景
                LODSB                           ;取背景
                AND     AL,7                    ;只能从0-7
                MOV     BYTE PTR D_020D,AL      ;修改背景号, 0表示没有
                RETN

L_081B:
                MOV     BX,OFFSET D_020D+1      ;BX=修饰字地址
                CMP     AL,3BH                  ;是';'吗?
                JNE     L_082E                  ;不是
                OR      BYTE PTR D_020D,8       ;';', 置抽点打印
                RETN
```

```
                DB      0, 0, 0, 0, 0, 0        ;没有用

L_082E:
                CMP     AL,5CH                  ;是'\'吗?
                JNE     L_0836                  ;不是
                OR      BYTE PTR [BX],1         ;\, 置反白打印
                RETN
L_0836:
                CMP     AL,2FH                  ;是'/'吗?
                JNE     L_083E                  ;不是
                OR      BYTE PTR [BX],2         ;置上划线打印
                RETN
L_083E:
                CMP     AL,5FH                  ;是'_'吗?
                JNE     L_0846                  ;不是
                OR      BYTE PTR [BX],4         ;置下划线打印
                RETN
L_0846:
                CMP     AL,3DH                  ;是'='吗?
                JNE     L_0853                  ;不是
                AND     BYTE PTR [BX],0F8H      ;清上下划线、反白打印
                AND     BYTE PTR D_020D,7       ;清抽点打印
                RETN
L_0853:
                CMP     AL,28H                  ;是'('吗?
                JNE     L_085E                  ;不是
                AND     BYTE PTR [BX],0E7H      ;清右旋90度打印
                OR      BYTE PTR [BX],8         ;置左旋90度打印
                RETN
L_085E:
                CMP     AL,29H                  ;是')'吗?
                JNE     L_0869                  ;不是
                AND     BYTE PTR [BX],0E7H      ;清左旋90度打印
                OR      BYTE PTR [BX],10H       ;置右旋90度打印
                RETN
L_0869:
                CMP     AL,3FH                  ;是'?'吗?
                JNE     L_0871                  ;不是
                AND     BYTE PTR [BX],0E7H      ;清旋转打印
                RETN
L_0871:
                CMP     AL,2BH                  ;是'+'吗?
                JNE     L_087C                  ;不是
                AND     BYTE PTR [BX],1FH       ;清下标打印
                OR      BYTE PTR [BX],20H       ;置上标打印
                RETN
L_087C:
                CMP     AL,2DH                  ;是'-'吗?
                JNE     L_0887                  ;不是
                AND     BYTE PTR [BX],1FH       ;清上标打印
                OR      BYTE PTR [BX],40H       ;置下标打印
                RETN
L_0887:
                CMP     AL,3AH                  ;是':'吗?
```

```
                JNE     L_0892                  ;不是
                AND     BYTE PTR [BX],1FH       ;清上下标打印
                OR      BYTE PTR [BX],80H       ;置全角字符打印
                RETN
L_0892:
                CMP     AL,21H                  ;是'!'吗?
                JNE     L_089A                  ;不是
                AND     BYTE PTR [BX],1FH       ;清上下标打印
                RETN
L_089A:
                CMP     AL,1BH                  ;是ESC吗?
                JNE     L_08A2                  ;不是
;必定为 ESC I 序列
                LODSB
                LODSB                           ;取字符
                JMP     SHORT L_0908
;非控制字符
L_08A2:
                MOV     DH,0
                MOV     DL,AL                   ;DH=0, DL=字符
                TEST    D_020F,2                ;要打印两次吗?
                JNZ     L_08B5                  ;不是
                TEST    AH,2                    ;还要纵扩吗?
                JNZ     L_08B5                  ;不是
                JMP     L_0735                  ;高点阵字符不能纵扩
L_08B5:
                JMP     L_06B8                  ;转字符打印

;子程序: 打印字间距
;输   入: 无
;输   出: 无
L_08B8:
                TEST    CS:D_0226,1             ;是横扩吗?(横扩字间距加倍)
                JZ      L_08C2                  ;不是
                ADD     CX,CX                   ;字间距加倍
L_08C2:
                MOV     DX,CS:D_0222            ;DX=字符或汉字
                CMP     DH,0A1H                 ;是汉字吗?
                JB      L_08E8                  ;不是
                ADD     CX,CX                   ;汉字字间距加倍
                CMP     DH,0A9H                 ;是制表符吗?
                JNE     L_08E8                  ;不是
                SUB     SI,6                    ;SI=最后二列数据地址
;制表符用符号的最后两列数据打印字间距

L_08D6:
                PUSH    SI                      ;保存SI
                TEST    BH,2                    ;要纵扩吗?
                JZ      L_08E1                  ;不用
                CALL    L_0794                  ;纵扩打印
                JMP     SHORT L_08E4
L_08E1:
                CALL    L_0152                  ;不纵扩打印
L_08E4:
                POP     SI                      ;恢复SI
```

```
                LOOP    L_08D6
                RETN
L_08E8:
                JMP     L_075B                  ;打印空列

L_08EB:
                TEST    D_020F,2                ;要两次打印吗?
                JZ      L_0907                  ;不是
;下面程序仅对 TH3070 有效, 实现单行放大
                MOV     AL,D_0229
                CMP     AL,0FH                  ;是结束放大吗?
                JNE     L_0907                  ;不是
                JMP     L_01A0                  ;结束单行放大

;设置打印方向或字型
L_08FC:
                CMP     AL,3EH                  ;是'>'吗?
                JE      L_0904                  ;是, 设置打印方向
                CMP     AL,3CH                  ;是'<'吗?
                JNE     L_0908                  ;不是设置打印方向
L_0904:
                MOV     BYTE PTR DS:[0A26H],AL  ;设置打印机方向
L_0907:
                RETN

;子程序: 设置打印字型, 计算当前字型汉字的列数
;输   入: AL=字型
;输   出: 无
L_0908:
                MOV     DL,AL                   ;DL=AL=字型
                AND     AL,3FH                  ;'A'-'Z'->1-26, 'a'-'z'->17-32
                MOV     DH,18H                  ;列数=24
                AND     DL,5FH                  ;小写改为大写
                CMP     DL,41H                  ;<'A'吗?
                JB      L_0962                  ;是, 无效的字型
                CMP     DL,58H                  ;>'X'吗?
                JA      L_0962                  ;是, 无效的字型
                DEC     AX                      ;AX=AX-1
                CMP     DL,55H                  ;小于'U'吗?
                JB      L_0931                  ;是, 非高点阵字型
;高点阵字型
                AND     AL,3                    ;仅4种字型
                SHL     AL,1
                SHL     AL,1
                OR      AL,80H                  ;置高点阵字型标志
                MOV     DH,28H                  ;列数=40
                NOP
                NOP
                NOP
                NOP             NOP
                NOP
L_0931:
                TEST    AL,10H                  ;是16点阵字型吗?
                JZ      L_0939                  ;不是
                MOV     DH,10H                  ;列数=16
```

```
                JMP     SHORT L_094D
L_0939:
                TEST    AL,20H                  ;是小写字型吗?
                JZ      L_094D                  ;不是
                MOV     DH,12H                  ;=18. 应该是36即22H
;应改为 MOV  DH,22H
                TEST    AL,3
                JNZ     L_0947
                MOV     DH,10H                  ;a,e...字型
                JMP     SHORT L_094D
L_0947:
                TEST    AL,1
                JNZ     L_094D                  ;d,f...字型
                MOV     DH,18H                  ;c,g...字型
L_094D:
                CMP     DL,55H                  ;是高点阵字型吗?
                JAE     L_0956                  ;是
                TEST    AL,2                    ;纵扩吗?
                JZ      L_095B                  ;不
L_0956:
                OR      D_020F,1                ;置纵扩标志
L_095B:
                MOV     BYTE PTR D_0213+1,DH    ;保存当前字型汉字的列数
                MOV     BYTE PTR D_0213,AL      ;保存字型
L_0962:
                RETN

;子程序: 设置图形打印方式, 并打印左边空白
;输   入: 无
;输   出: SI=当前行数据首址
L_0963:
                MOV     BH,0FFH
L_0965:
                MOV     CX,D_0209               ;行列数
                ADD     CX,D_021D               ;加左空列数
                CALL    L_09D0                  ;设置图形打印方式
                MOV     CX,D_021D               ;取左空列数
                OR      CX,CX                   ;=0?
                JZ      L_097B                  ;是, 不用打印左空
                CALL    L_075P                  ;打印左空
L_097B:
                XOR     SI,SI                   ;SI=当前行数据首址
                RETN

;子程序: 打印纵扩字时, 设置打印方向, 并走纸半行
;输   入: 无
;输   出: 无
L_097E:
                CALL    L_0A25                  ;设置打印方向
                MOV     AL,D_0A44
                JMP     SHORT L_099B            ;走纸半行

;子程序: 打印回车或换行
;输   入: 无
;输   出: 无
```

```
L_0986:
                MOV     AL,BYTE PTR D_0207      ;取字符
                CMP     AL,0DH                  ;是回车吗?
                JNE     L_0992                  ;不是
                CALL    L_01A0                  ;打印回车
                JMP     SHORT L_09C7
;打印换行
L_0992:
                MOV     AL,D_0216               ;取打印方向
                CALL    L_0A27                  ;设置打印方向
                MOV     AL,D_0215               ;取行距
L_099B:
                MOV     AH,0
                CALL    L_0A00                  ;换行
                MOV     AL,D_0210               ;AL=当前页剩余行数
                CMP     AL,0                    ;=0?
                JE      L_09AB                  ;是
                DEC     AL                      ;剩余行数减1
                JNZ     L_09C4
L_09AB:
                MOV     CL,BYTE PTR D_0201      ;CL=页间空行
                MOV     CH,0
                OR      CX,CX                   ;有页间空行吗?
                JZ      L_09C1                  ;没有
;打印页间空行
L_09B5:
                MOV     AL,0DH
                CALL    L_01A0                  ;打印回车
                MOV     AL,0AH
                CALL    L_01A0                  ;打印换行
                LOOP    L_09B5                  ;打印CX次
L_09C1:
                MOV     AL,D_0200               ;取页长
L_09C4:
                MOV     D_0210,AL               ;置当页未打印行数
L_09C7:
                JMP     L_08EB

                DB      0, 0, 0, 0, 0, 0        ;没有用
```

;以下程序在 PRTA.COM 中是没有的(全为 0). 它是在启动时设置的. 为了说明起见, 特将
;LQ1500 系列的子程序反汇编后, 供大家参阅

;子程序: 设置图形打印方式, ESC * 为 LQ1500 系列打印机设置图形打印控制码
;输　入: CX=行宽
;输　出: 无

```
L_09D0:
                MOV     AL,1BH
                CALL    L_01A0                  ;打印ESC
                MOV     AL,2AH
                CALL    L_01A0                  ;打印*
                MOV     AL,27H                  ;=39, 即设置为3倍密度打印方式
                CALL    L_01A0
                MOV     AL,CL
                CALL    L_01A0                  ;行宽后低字节
```

```
                MOV     AL,CH
                JMP     L_01A0                  ;行宽的高字节

                DB      0,0,0,0,0,0
                NOP

;为两次打印设置打印方向
L_09F0:
                CMP     BYTE PTR D_09FF,3EH     ;是全部单向打印吗?
;9FFH 控制全部字符的打印方向，当前为>时，所有字符均单向打印. 为<时，对于要两次
;才能打完的行，还要进一步判断 216H 是否为双向打印
                JE      L_0A25                  ;是
                MOV     AL,D_0216               ;取纵扩字打印方向
                JMP     SHORT L_0A27

                DB      0, 0, 0

D_09FF          DB      0                       ;打印方向

;子程序: 换行
;输   入: AL=行距
;输   出: 无
L_0A00:
                MOV     CL,3
                MUL     CL
                SHR     AX,1                    ;行距x3/2, 化为1/180英寸数
                PUSH    AX
                MOV     AL,1BH                  ;ESC
                CALL    L_01A0
                MOV     AL,33H                  ;'3', ESC 3设置行距控制码
                CALL    L_01A0
                POP     AX
                CALL    L_01A0                  ;设置行距
                MOV     AL,0AH                  ;换行字符
                CALL    L_01A0                  ;换行
                JMP     SHORT L_09F0            ;转为两次打印的行设置打印方向

D_0A1C          NOP
                DB      8 DUP (0)

L_0A25:
                MOV     AL,3EH                  ;设置单向打印

;此时 AL 已为打印方向
L_0A27:
                MOV     D_09FF,AL               ;保存打印方向
                CMP     AL,3EH                  ;是'>'吗?
                JE      L_0A32                  ;是
                MOV     AL,1                    ;置双向打印
                JMP SHORT L_0A34
L_0A32:
                MOV     AL,0                    ;置单向打印
L_0A34:
                PUSH    AX
                MOV     AL,1BH                  ;ESC
```

```
                CALL    L_01A0
                MOV     AL,55H                  ;'U', ESC U为设置打印方向的控
                                                ;制码
                CALL    L_01A0
                POP     AX
                JMP     L_01A0

D_0A43          DB      14H                     ;默认行距
D_0A44          DB      10H                     ;半行长度，应为24，但由于在换
                                                ;行时进行了计算(x3/2)，所以这
                                                ;里是16
                DB      3CH                     ;默认打印方向
D_0A46          DW      990H                    ;最大行宽
D_0A48          DB      ' AR3240$'              ;打印机名称

;INT 7AH
L_0A50:
                MOV     CX,10H                  ;列数=16
                JMP     SHORT L_0A62

;INT 7BH
L_0A55:
                MOV     CX,18H                  ;列数=24
                JMP     SHORT L_0A62

;INT 7CH
L_0A5A:
                MOV     CX,28H                  ;列数=40
                CMP     DH,0AAH                 ;是字符吗?
                JMP     SHORT L_0A65
L_0A62:
                CMP     DH,0                    ;是字符吗?
L_0A65:
                JNE     L_0A69                  ;不是
                SHR     CX,1                    ;字符列数除2
L_0A69:
                IRET                            ;中断返回

                DB      0, 0, 0, 0, 0, 0        ;没有用

D_0A70          DB      0
                DB      15 DUP (0)

;打印机型号选择菜单
D_0A80          DB      9,'PRT.COM 汉字打印驱动程序打印选择：',0DH,0Ah
                DB      0DH, 0AH
                DB      '     1 - P1351(P1350)          2 - M2024(M1724)',0DH,0Ah
                DB      '     3 - TH3070(3080)          4 - AR2463',0DH,0Ah
                DB      '     5 - AR3240(NEC P7)        6 - OKI8320(5320)',0DH,0Ah
                DB      '     7 - M1570                 8 - NEC3824',0DH,0AH,
                DB      '     9 - NM9400',0DH,0AH,0DH,0Ah
                DB      9,'请键入选择号：$'
                DB      0
                DB      10 DUP (0)
```

```
;以下是各打印机特有的子程序，从1号打印机开始，每种打印机占80H字节，共9种
D_0B80:
                MOV     AX,CX
                MOV     CX,40AH
                ..... .                                 ;以下省略

;P1351打印机的额外程序，46个字节
D_1000          EQU     $
                PUSH    AX
                PUSH    BX
                PUSH    CX
                PUSH    DS
                PUSH    CS
                POP     DS
                MOV     BX,[26AH]
                MOV     [BX],AL
                INC     WORD PTR [26AH]
                INC     WORD PTR [26FH]
                CMP     BYTE PTR [26FH],3
                JNZ     L_1041
                MOV     WORD PTR [26AH],26
                MOV     BYTE PTR [26FH],0
                MOV     CX,4
                PUSH    CX
                MOV     CX,6
                MOV     AL,0

L_102F:
                SHL     BYTE PTR [BX],1
                RCL     BYTE PTR [BX-1],1
                RCL     BYTE PTR [BX-2],1
                RCL     AL,1
                LOOP    L_102F

;               call    s_0750
                DB      0E8H, 12H,0F7H
                POP     CX
                LOOP    L_1029

L_1041:
                POP     DS
                POP     CX
                POP     BX
                POP     AX
                RETN

;设置打印机型号
L_1046:
                MOV     DX,OFFSET D_0A80        ;DX=菜单地址
                MOV     AH,9
                INT     21H                     ;显示菜单
                MOV     AX,0C01H
                INT     21H                     ;读数字
                CMP     AL,31H                  ;<'1'吗?
                JB      L_1085                  ;是，无效的选择
```

```
                CMP     AL,39H                  ;>'9'吗?
                JA      L_1085                  ;是, 无效的选择
                AND     AL,0FH                  ;转换为数值
                MOV     BYTE PTR DS:[10E6H],AL  ;保存选择号

                MOV     DX,OFFSET D_0103        ;文件名地址
                MOV     AX,3D01H
                INT     21H                     ;打开文件PRTA.COM
                MOV     BX,AX
                MOV     AX,5700H
                INT     21H                     ;取文件日期
                PUSH    CX
                PUSH    DX                      ;保存日期
                MOV     DX,OFFSET DS:[100H]
                MOV     CX,1083H
                MOV     AH,40H
                INT     21H                     ;写文件内容
                POP     DX
                POP     CX                      ;恢复日期
                MOV     AX,5701H
                INT     21H                     ;恢复日期
                MOV     AH,3EH                  ;关闭文件
                INT     21H
L_1085:
                INT     20H                     ;返回DOS
                NOP

;初始化程序
L_1088:
                MOV     AL,DS:[80H]             ;取命令行参数数目, 即PRTA后的
                                                ;字符数目
                OR      AL,AL                   ;有参数吗?
                JZ      L_10A1                  ;没有
                CMP     AL,5                    ;参数数大于4吗?
                JB      L_1046                  ;不是
;命令行参数大于 4 个, 表示命令行中有字型表. 缺省的字型表与设置的相同.
                MOV     SI,82H                  ;字型表首址
L_1096:
                LODSB                           ;取一字符
                CMP     AL,0DH                  ;命令行内容已结束吗?
                JE      L_10A1                  ;是
                MOV     BYTE PTR DS:[15DH][SI],AL ;修改字型表
                JMP     SHORT L_1096
L_10A1:
                MOV     AX,3517H
                INT     21H                     ;取原INT 17H地址
                CMP     BX,280H                 ;已运行过PRTA.COM吗?
                JNE     L_10B6                  ;没有
;已运行过 PRTA.COM, 取得的中断向量是出 PRTA.COM 自己设置的
                MOV     BX,WORD PTR ES:[1A7H]
                MOV     ES,WORD PTR ES:[1A9H]   ;取上次运行时保存的原 INT 17H
                                                ;中断向量, 这样 ES:BX 便为原 INT
                                                ;17H 中断向量.
L_10B6:
                MOV     WORD PTR DS:[1A7H],BX
```

```
MOV     WORD PTR DS:[1A9H],ES   ;保存原 INT 17H 中断向量
MOV     DX,OFFSET L_0280        ;新 INT 17H 入口地址
MOV     AX,2517H
INT     21H                     ;重置 INT 17H
MOV     AX,257DH
INT     21H                     ;设置 INT 7DH 与 INT 17H 相同,这
                                ;中断向量在清除外加模块时使用
;INT 7AH, INT 7BH, INT 7CH是DOS为用户保留的中断程序. DOS启动时, 这些中断向量
;都被设置为0000:0000, 该地址是中断向量表首址, 固执行这些中断程序都将引起死机.
;在2.13H中, 这些中断向量均由FILE系列程序设置(参见第一章). 但由于在运行PRTA.COM
;时这些程序可能还未运行或者有些高点阵字库没有安装, 使得打印时可能会死机. 为安
;全起见, 这里特意设置了这些中断向量, 这些中断程序都没有实质性的内容, 但执行这
;些中断程序不会引起死机. 另一方面, 这一方法使得PRTA.COM不能两次运行, 因为第二
;次运行后, 字符和汉字的点阵数据就无法取得.
MOV     DX,OFFSET L_0A50        ;空的INT 7AH入口地址
MOV     AX,257AH
INT     21H                     ;设置 INT 7AH
MOV     DX,OFFSET L_0A55        ;空的 INT 7BH 入口地址
MOV     AX,257BH
INT     21H                     ;设置 INT 7BH
MOV     DX,OFFSET L_0A5A        ;空的 INT 7CH 入口地址
MOV     AX,257CH
INT     21H                     ;设置 INT 7CH

PUSH    CS
POP     ES                      ;恢复 ES=CS
MOV     AL,5                    ;AL=打印机型号, 5 表示 LQ1500 系
                                ;列打印机.
;在选择打印机型号时, PRTA.COM实质上仅修改了上面指令中的数据字节(这里为5).
;PRTA.COM是个通用的打印驱动程序, 它可以支持多种不同的打印机. 每种打印机的打印
;方法基本相同, 只是几个低层的子程序和一些打印机特性数据不同而已, 在启动时只要
;按默认的打印机型号设置这些子程序和数据即可.
PUSH    AX                      ;保存打印机型号
MOV     CL,80H                  ;子程序长度
MUL     CL                      ;剩打印机型号
ADD     AX,0B00H                ;AX=默认打印机子程序首址
MOV     SI,AX
MOV     DI,OFFSET DS:[9D0H]     ;实际使用时子程序的首址
REP     MOVSB                   ;设置子程序
MOV     SI,OFFSET D_0A43
LODSB
MOV     D_0215,AL               ;设置行距
LODSB
LODSB
MOV     D_0216,AL               ;设置打印方向
LODSW                           ;AX=[0A46H]=行宽(以点为单位)
MOV     D_0220,AX               ;设置左边界
MOV     DX,OFFSET D_0A48        ;打印机名称地址
MOV     AH,9
INT     21H                     ;显示打印机名称
MOV     DX,OFFSET D_0112        ;"打印机驱动程序"信息地址
MOV     AH,9
INT     21H
POP     AX                      ;恢复 AL=打印机型号
CMP     AL,1                    ;是 1 号打印机吗?
```

```
                JNE     L_1134                  ;不是
;P1351打印机初始化
                MOV     SI,OFFSET D_1000
                MOV     DI,OFFSET D_0A70
                MOV     CX,46H
                REP     MOVSB                   ;其它程序(超过 80H 字节的部分)
                MOV     WORD PTR DS:[193H],0BAE9H
                MOV     D_0195,8                ;修改指令, JMP 0A50H
                MOV     DX,0AB6H                ;驻留长度
                JMP     SHORT L_1181

L_1134:
                CMP     AL,2                    ;是 2 号打印机吗?
                JNE     L_114F                  ;不是
;M2024 打印初始化
                MOV     AX,9090H                ;指令 NOP, NOP 的值
                MOV     WORD PTR DS:[397H],AX
                MOV     BYTE PTR DS:[399H],AL
                MOV     WORD PTR DS:[98DH],AX
                MOV     BYTE PTR DS:[98FH],AL
                MOV     WORD PTR DS:[9B7H],AX
                MOV     BYTE PTR DS:[9B9H],AL   ;修改指令
                JMP     SHORT L_117E

L_114F:
                CMP     AL,4                    ;是 4 号打印机吗?
                JE      L_1178                  ;是
                CMP     AL,5                    ;是 5 号打印机吗?
                JNE     L_1163                  ;不是
;LQ1500打印机初始化
                MOV     BYTE PTR DS:[149H],72H  ;修改控制码, 缺省为43H
                MOV     BYTE PTR DS:[141H],90H  ;NOP
                JMP     SHORT L_117E
L_1163:
                CMP     AL,7
                JB      L_117E                  ;3,6 号打印机
                JNZ     L_1178                  ;8,9 号打印机
;M1570 彩色打印机
                MOV     BYTE PTR DS:[149H],43H  ;修改控制码
                MOV     BYTE PTR DS:[14FH],0FFH
                MOV     BYTE PTR DS:[141H],90H
L_1178:
                MOV     WORD PTR DS:[193H],0C488H ;MOV AH,AL
L_117E:
                MOV     DX,0A70H                ;缺省的驻留长度
L_1181:
                INT     27H                     ;驻留退出
CODE            ENDS
                END     L_0100
```

第二节　屏幕拷贝　SEGP.COM

屏幕打印就是将当前屏幕内容输出到打印机。屏幕打印是由INT　5H中断程序处理的，当用户按下屏幕打印键SHIFT_PRTSC时，系统便调用INT　5H中断程序，实现屏幕打印。

在DOS启动后，INT 5H指向ROM BIOS的屏幕打印程序，该程序仅打印当前屏幕上的字符，不能打印屏幕上的图形信息。如果要打印屏幕上的图形信息，必须另行设计屏幕打印程序。

一、屏幕打印原理

屏幕打印主要有两种类型，即按字符方式打印和按图形方式打印。

按字符方式打印的屏幕打印程序，主要利用显示中断程序（INT 10H）读取屏幕各位置的字符，并直接将这些字符输出到打印机，每个屏幕行结束时，增加输出一个回车换行符。这种类型的屏幕打印程序比较容易设计，但往往不能满足用户的需要。ROM BIOS中的屏幕打印程序就是按字符方式打印的。

按图形方式打印的屏幕打印程序，直接读取视频缓冲区的内容，转换为打印数据，在图形打印方式下实现屏幕打印。在图形显示方式下，视频缓冲区的内容都是以点为单位存放的，数据读取比较复杂，同时，24针打印机又要求打印图形时，以列为单位传送数据，每列为垂直24点（3字节）的数据，因此，按图形方式打印的屏幕打印程序的设计是比较复杂的。

2.13H的屏幕打印程序都是按图形方式打印的。在CC版中，屏幕打印程序为SGP.COM和SEGP.COM。SGP.COM用于CGA显示方式（在EGA或VGA上，也可以使用CGA的显示方式），SEGP.COM用于EGA和VGA显示方式。

由于图形屏幕打印程序的实现方法类似，所以本书只介绍使用在EGA和VGA显示方式中的屏幕打印程序——SEGP.COM。

二、SEGP.COM（EGA和VGA屏幕打印模块）

1.使用说明

SEGP ab←┘

其中a为打印机型号，可以是数字2－9，对应的打印机是：

2－M2024　　3－TH3070　　4－AR2463

5－LQ1500　　6－OKI8324　　7－M1570

8－NEC3824　　9－NM9400

b放大倍数，可以是数值1－3。

例如：SEGP 53表示采用LQ1500系列打印机，打印图形放大3倍。

该程序只能在EGA或VGA显示方式下使用。运行该程序后，即可按SHIFT_PRTSC键进行屏幕打印，也可以直接执行INT 5H实现屏幕打印。在打印过程中，可以按CTRL_BREAK中止打印。

2.程序功能

本程序主要包括两个部分，即初始化代码和INT 5H代码。

初始化代码主要完成以下工作：

1）检查命令行中的打印机型号，设置指定的打印机的控制码数据。

2）检查放大倍数，若小于1或大于3，则修改为2。

3）设置INT 5H。

4）驻留退出。

3.变量名表

地址	长度	意义
0101	3	存放打印数据，列共24点，分3字节
0103	2	原INT 5H的偏移地址
0105	2	原INT 5H的段址
0107	1	屏幕打印时的放大倍数，可以是1,2或3
02E4		程序初始化失败时的提示信息
0330		各打印机的控制码

4.程序清单

```
;SEGP.ASM
CODE            SEGMENT
```

```
                ASSUME  CS:CODE, DS:CODE
                ORG     100H

L_0100:
                JMP     L_026C                  ;转初始化程序
                NOP

                DB      0,0,0,0                 ;没有用

;INT 05H
;屏幕打印
L_0108:
                PUSH    DS
                PUSH    BX                      ;保存寄存器

                MOV     BX,40H
                MOV     DS,BX                   ;DS=40H=BIOS 数据段址

                MOV     BX,49H                  ;当前显示方式单元
                CMP     BYTE PTR [BX],10H       ;是 EGA 显示方式吗?
                JE      L_0125                  ;是
                CMP     BYTE PTR [BX],12H       ;是 VGA 显示方式吗?
                JE      L_012A                  ;是

;不是EGA和VGA显示方式, 调用原INT 5H实现屏幕打印
                PUSH    CS
                POP     DS                      ;DS=CS

                PUSHF
                CALL    DWORD PTR DS:[103H]     ;调用原 INT 05H

                POP     BX
                POP     DS                      ;恢复寄存器

                IRET                            ;中断返回

;EGA 显示方式
L_0125:
                MOV     BX,15EH                 ;BX=当前屏幕高度(350 线)
                JMP     SHORT L_012D

;VGA 显示方式
L_012A:
                MOV     BX,1E0H                 ;BX=当前屏幕高度(480 线)

L_012D:
                PUSH    ES
                PUSH    SI
                PUSH    DI
                PUSH    AX
                PUSH    CX
                PUSH    DX                      ;保存寄存器
                PUSH    CS
                POP     DS                      ;DS=CS
```

```
                MOV     AX,0A000H
                MOV     ES,AX                   ;ES=视频区段址

                MOV     AL,0AH                  ;换行符
                CALL    L_0237                  ;使打印机换行
                CALL    L_020D                  ;置单向打印
                MOV     AL,BYTE PTR DS:[107H]   ;取放大倍数
                XOR     AH,AH
                MUL     BX                      ;剩以屏幕扫描线行数=要打印的
                                                ;总线数
                MOV     DL,18H                  ;每行打印 24 线
                DIV     DL                      ;除 24
                OR      AH,AH                   ;有余数吗?
                JZ      L_0154                  ;没有
                INC     AX                      ;有余数,增加一行
                XOR     AH,AH
L_0154:
                MOV     CX,AX                   ;CX=要打印的行数
                XOR     DI,DI                   ;屏幕内容(在视频区中)首址

L_0158:
                MOV     AH,1
                INT     16H                     ;有键按下吗?
                JZ      L_0166                  ;没有
;检测 CTRL_BREAK
                XOR     AH,AH
                INT     16H                     ;读入按下的键
                OR      AX,AX                   ;是 CTRL_BREAK 吗?
                JZ      L_016E                  ;是,打印中止
L_0166:
                CALL    L_0217                  ;设置图形打印方式
                CALL    L_017A                  ;打印一行
                LOOP    L_0158                  ;共要打印 CX 行

L_016E:
                CALL    L_0212                  ;置双向打印

                POP     DX
                POP     CX
                POP     AX
                POP     DI
                POP     SI
                POP     ES
                POP     BX
                POP     DS                      ;恢复寄存器

                IRET                            ;中断返回
;子程序: 打印一行, 每行垂直24点, 水平640点. 打印时要按放大倍数放大, 如:放大倍
;       数为2, 相当于打印两行, 水平点阵数为640x2=1280点
;输   入: DI=当前行(以垂直24点为单位)内容在视频区首址
;输   出: 无
L_017A:
                PUSH    CX                      ;保存当前行号
                PUSH    DI                      ;保存当前行首址
```

```
                MOV     CX,50H                  ;每行 640 点即 80 个字节
L_017F:
                PUSH    CX                      ;保存未打印字节数
                MOV     CL,8                    ;每一字节有 8 点

L_0182:
                MOV     DX,18H                  ;每次打印 24 点(与 24 针对应)
                PUSH    DI                      ;保存当前字节地址(当前行第一
                                                ;条扫描线地址)
L_0186:
                CALL    L_01C6                  ;取一点(已按倍数放大)
                ADD     DI,50H                  ;下一条扫描线
                OR      DL,DL                   ;已取足 24 点了吗?
                JNZ     L_0186                  ;还没有

;已取到垂直24点数据(放大后的数据)
                PUSH    CX                      ;保存当前打印位数(CL)
                MOV     CL,BYTE PTR DS:[107H]   ;取倍数
                XOR     CH,CH

L_0197:
                MOV     AL,BYTE PTR DS:[100H]   ;第一字节内容
                CALL    L_022A                  ;打印前 8 点
                MOV     AL,BYTE PTR DS:[101H]   ;第二字节内容
                CALL    L_022A                  ;打印中间 8 点
                MOV     AL,BYTE PTR DS:[102H]   ;第三字节内容
                CALL    L_022A                  ;打印后 8 点
                LOOP    L_0197                  ;按倍数在水平方向重复打印

                POP     CX                      ;恢复当前位号
                POP     DI                      ;恢复当前字节地址
                LOOP    L_0182                  ;将屏幕上水平 8 列数据放大打印

                INC     DI                      ;一下字节位置
                POP     CX                      ;恢复打印字节数
                LOOP    L_017F                  ;打印一行

                POP     DI                      ;恢复当前行首址
                CALL    L_0208                  ;换行
                MOV     AX,18H                  ;每次打印 24 点
                DIV     BYTE PTR DS:[107H]      ;除以倍数,得一行打印的扫描线数
                MOV     DL,50H
                MUL     DL                      ;每行 80 字节(80x8 点)
                ADD     DI,AX                   ;指向未打印的扫描线
                POP     CX                      ;恢复当前打印行号
                RETN
;子程序: 读视频区一点,放大后送入打印缓冲区
;输    入: DI = 当前视频区地址
;          CL = 扫描线位号
;          DL = 剩余位数
;输    出: 无
L_01C6:
                MOV     BL,BYTE PTR DS:[107H]   ;倍数数->BL

                PUSH    DX
```

```
                PUSH    AX                      ;保存寄存器
                XOR     AX,AX                   ;AL=数据, AH=当前页面
L_01CE:
                PUSH    AX                      ;保存数据
                MOV     DX,3CEH                 ;图形控制器索引寄存器
                MOV     AL,4                    ;选择读映象选择寄存器
                                                ;即页面选择
                OUT     DX,AX                   ;输入
                POP     AX                      ;恢复数据
                OR      AL,ES:[DI]              ;读数据
                INC     AH                      ;下一页面
                CMP     AH,4                    ;已读完4个页面了吗?
                JB      L_01CE                  ;还没有

                MOV     BH,AL                   ;数据->BH

                POP     AX
                POP     DX                      ;恢复寄存器
L_01E4:
                MOV     AH,BH                   ;数据->AH
                SHR     AH,CL                   ;第CL位->CF
                RCL     AL,1                    ;CF移入AL中
                DEC     DL                      ;减去一点
                TEST    DL,7                    ;读满入8点数据了吗?
                JNZ     L_01FF                  ;没有
;已读入8点数据
                PUSH    AX                      ;保存数据
                MOV     AL,DH                   ;字节计数
                XOR     AH,AH
                MOV     SI,AX                   ;写入地址
                POP     AX                      ;恢复数据
                MOV     BYTE PTR DS:[100H][SI],AL ;存入打印缓冲区
                INC     DH                      ;下一字节
L_01FF:
                DEC     BL
                JNZ     L_01E4                  ;按倍数重复
                RETN

                DB      0, 0, 0, 0              ;没有用

;子程序: 打印机设置
;输    入: 无
;输    出: 无
L_0208:
;1.图形换行
                MOV     SI,90H                  ;换行控制码地址
                JMP     SHORT L_021A

;2.置单向打印
L_020D:
                MOV     SI,98H                  ;单向打印控制码地址
                JMP     SHORT L_021A

;3.置双向打印
L_0212:
```

```
                MOV     SI,9CH                  ;双向打印控制码地址
                JMP     SHORT L_021A

;4.置图形打印方式
L_0217:
                MOV     SI,0                    ;图形打印控制码地址
;这里的数据0已在初始化时被修改为具体控制码地址

;设置打印机
L_021A:
                LODSB                           ;取控制码长度字节
                OR      AL,AL
                JZ      L_0229                  ;长度=0,不设置

                MOV     DL,AL                   ;DL=长度
L_0221:
                LODSB                           ;取控制码
                CALL    L_0237                  ;送打印机
                DEC     DL
                JNZ     L_0221                  ;下一个控制码

L_0229:
                RETN

;子程序: 打印AL中的数据
;输    入: AL=要打印的数据
;输    出: 无
L_022A:
                JMP     SHORT L_0237            ;若是AR2463或M1570,这条指令
                                                ;将改为MOV AH,AL
;对于AR2463和M1575打印机,打印数据的位顺序与其它打印机是颠倒的,下面将AL中的位
;顺序颠倒一下
                PUSH    CX
                MOV     CX,8                    ;计数
L_0230:
                RCR     AH,1                    ;AH的最低位->CF
                RCL     AL,1                    ;CF->AL的最低位
                LOOP    L_0230                  ;实现位颠倒

                POP     CX                      ;恢复CX
L_0237:
                PUSH    DX
                PUSH    CX
                PUSH    AX                      ;保存寄存器

                MOV     DX,378H                 ;打印机数据端口
                OUT     DX,AL                   ;将数据送数据锁存器中
                INC     DX                      ;DX指向状态寄存器端口(379H)
                MOV     AH,0AH                  ;等待计数值
L_0241:
                XOR     CX,CX                   ;等待计数值
L_0243:
                IN      AL,DX                   ;读状态寄存器
                OR      AL,AL                   ;打印机忙吗?
                JS      L_0256                  ;不忙
```

```
                LOOP    L_0243                  ;循环等待

                DEC     AH                      ;大循环
                JNZ     L_0241
;打印机出错
                OR      AL,1                    ;置错误标志
                AND     AL,0F9H                 ;屏蔽无用位
                MOV     AH,AL
                JMP     SHORT L_0263

L_0256:
;此时打印数据已送锁存器中,但必须等选通信号为1时,打印机才能将它打印
                MOV     AL,0DH                  ;令选通信号为1
                INC     DX                      ;控制锁存器端口地址=37AH
                OUT     DX,AL                   ;输出
                MOV     AL,0CH                  ;再清选通信号
                OUT     DX,AL
                DEC     DX                      ;状态寄存器
                IN      AL,DX                   ;读状态寄存器值
                AND     AL,0F8H                 ;屏蔽无用位
                MOV     AH,AL                   ;状态送 AH
L_0263:
                POP     DX                      ;恢复数据
                MOV     AL,DL                   ;AL=要打印机的数据
                XOR     AH,48H                  ;加工状态信息

                POP     CX
                POP     DX                      ;恢复寄存器

                RETN

;初始化程序
L_026C:
                CMP     BYTE PTR DS:80H,3       ;参数个数少于 3 吗?
                JB      L_02DB                  ;参数太少, 错误终止

                MOV     AX,3505H
                INT     21H                     ;取原 INT 05H 中断向量
                MOV     WORD PTR DS:[103H],BX   ;保存原 INT 05H 偏移地址
                MOV     WORD PTR DS:[105H],ES   ;保存原 INT 05H 段地址
                PUSH    CS
                POP     ES                      ;恢复 ES

                MOV     AL,DS:82H               ;取第一个参数(打印机代号)
                AND     AL,0FH                  ;屏蔽高位
                CMP     AL,1
                JBE     L_02DB                  ;打印机代号不能小于 1

                CMP     AL,4                    ;是 AR2463 吗?
                JE      L_0293                  ;是
                CMP     AL,7                    ;是 M1570 吗?
                JB      L_0299                  ;不是
L_0293:
                MOV     WORD PTR DS:[22AH],0C488H ;修改 22A 处指令的 JMP !_0237
                                                  ;改为 MOV AH,AL
```

```
L_0299:
                MOV     CL,40H                  ;CX=40H,注意:程序直接运行时
                                                ;DOS 将 CX 置 0 若 DEBUG 跟踪则此时
                                                ;CH<>0
                MUL     CL
                ADD     AX,2B0H                 ;打印机控制命令集地址
                MOV     SI,AX
                MOV     DI,90H
                REP     MOVSB                   ;保存在 90H-D0H

                MOV     AL,DS:83H               ;取倍数
                CMP     AL,31H                  ;小于 1 倍
                JB      L_02B2                  ;是
                CMP     AL,33H                  ;小于 3 倍
                JBE     L_02B4
L_02B2:
                MOV     AL,32H                  ;改为 2 倍
L_02B4:
                AND     AL,0FH                  ;屏蔽高位
                MOV     BYTE PTR DS:[107H],AL   ;保存放大倍数
                MOV     DL,10H                  ;剩 16
                MUL     DL
                ADD     AL,90H                  ;按放大倍数计算设置图形方式控
                                                ;制码的地址
                MOV     WORD PTR DS:[218H],AX   ;修改指令

                MOV     DX,OFFSET L_0108        ;INT 05H 的偏移地址
                MOV     AX,2505H
                INT     21H                     ;修改 INT 05H

                MOV     AX,40H
                MOV     DS,AX                   ;DS=BIOS 数据段址
                MOV     AX,DS:8                 ;打印机端口地址
                MOV     WORD PTR CS:[23BH],AX

                MOV     DX,26CH                 ;驻留长度
                INT     27H                     ;驻留退出

L_02DB:
                MOV     DX,OFFSET D_02E4
                MOV     AH,9
                INT     21H                     ;显示错误信息

                INT     20H                     ;结束运行, INT 5H 没有设置

D_02E4          DB      '格式:SEGP ab'
                DB      '其中:a 为打印机代号(2~9)',0DH,0Ah
                DB      '      b 为放大倍数(1~3)',07H,'$'
                DB      29H,07H,24H,0,0,0,0,0,0,0       ;没有用

D_0330:
                DB      04H,1BH,4AH,12H,0AH             ;打印机控制码
                ......                                  ;以下省略
CODE            ENDS
                END     L_0100
```

```
L_0299:
                MOV     CL,40H                  ;CX=40H,注意:程序直接运行时
                                                ;DOS 将 CX 置 0 若 DEBUG 跟踪则此时
                                                ;CH<>0
                MUL     CL
                ADD     AX,2B0H                 ;打印机控制命令集地址
                MOV     SI,AX
                MOV     DI,90H
                REP     MOVSB                   ;保存在 90H-D0H

                MOV     AL,DS:83H               ;取倍数
                CMP     AL,31H                  ;小于 1 倍
                JB      L_02B2                  ;是
                CMP     AL,33H                  ;小于 3 倍
                JBE     L_02B4
L_02B2:
                MOV     AL,32H                  ;改为 2 倍
L_02B4:
                AND     AL,0FH                  ;屏蔽高位
                MOV     BYTE PTR DS:[107H],AL   ;保存放大倍数
                MOV     DL,10H                  ;剩 16
                MUL     DL
                ADD     AL,90H                  ;按放大倍数计算设置图形方式控
                                                ;制码的地址
                MOV     WORD PTR DS:[218H],AX   ;修改指令

                MOV     DX,OFFSET L_0108        ;INT 05H 的偏移地址
                MOV     AX,2505H
                INT     21H                     ;修改 INT 05H

                MOV     AX,40H
                MOV     DS,AX                   ;DS=BIOS 数据段址
                MOV     AX,DS:8                 ;打印机端口地址
                MOV     WORD PTR CS:[23BH],AX

                MOV     DX,26CH                 ;驻留长度
                INT     27H                     ;驻留退出

L_02DB:
                MOV     DX,OFFSET D_02E4
                MOV     AH,9
                INT     21H                     ;显示错误信息

                INT     20H                     ;结束运行, INT 5H 没有设置

D_02E4          DB      '格式:SEGP ab'          DB      '其中:a 为打印机代号(2~9)',0DH,0Ah
                DB      '       b 为放大倍数(1~3)',07H,'$'

                DB      29H,07H,24H,0,0,0,0,0,0         ;没有用

D_0330:
                DB      04H,1BH,4AH,12H,0AH             ;打印机控制码
                ......                                  ;以下省略
CODE            ENDS
                END     L_0100
```

第七章 其　　它

第一节 安装工作参数 HHDOS.COM

一、程序功能

本程序主要是为保护编者权益而设计的，它保存当前硬盘的总可用簇数和213子目录的起始簇号。这些数据在系统启动时（运行CCCC.COM时），要与实际的数据核对，若发现不同则重新启动或死机。

二、程序清单

```
;HHDOS.ASM
CODE            SEGMENT
                ASSUME  CS:CODE,DS:CODE
                ORG     100H

L_0100:
                MOV     AX,0FEE8H
                NOT     AX                      ;AX=117H
                JMP     AX                      ;转 117H 执行

                MOV     AL,2
                MOV     CX,1
                XOR     DX,DX
                INT     25H
                CMP     [202H],46H
                JNE     L_0125                  ;这段程序没有执行

L_0117:
;读硬盘主引导扇区
                MOV     AX,201H                 ;读取一个扇区
                MOV     BX,300H                 ;读取数据保存地址
                MOV     CX,1
                MOV     DX,80H                  ;物理地址为:C 盘 0 道 0 面 1 扇区
                INT     13H                     ;读入硬盘主引导扇区
L_0125:
                JNC     L_0132                  ;操作成功
L_0127:
;错误终止
                MOV     DL,7
                MOV     AH,2
                INT     21H                     ;响玲
                MOV     DL,1                    ;返回的错误码
                JMP     L_01E3                  ;错误终止

L_0132:
;查找 DOS 分区
                MOV     SI,OFFSET DS:[1BEH]     ;分区表起始位移
L_0135:
                CMP     BYTE PTR [BX+SI+4],1
                JE      L_014C                  ;是 12 bits 的 DOS 分区
                CMP     BYTE PTR [BX+SI+4],4
```

```
                JE      L_014C                  ;是 16 bits 的 DOS 分区
                CMP     BYTE PTR [BX+SI+4],6
                JE      L_014C                  ;是保留的DOS分区
;当前分区项不是DOS分区
                ADD     SI,10H                  ;SI=下一分区项位移
                JMP     SHORT L_0135            ;检查下一分区项

;读入分区引导扇区(BOOT)
L_014C:
                MOV     DX,[BX+SI]
                MOV     CX,[BX+SI+2]            ;起始物理地址
                MOV     BX,300H
                MOV     AX,201H
                INT     13H                     ;读入分区 BOOT 扇区
                JC      L_0127                  ;操作失败
                MOV     BYTE PTR [BX+2],48H     ;='H',设置 213H 标志
                MOV     AX,301H
                INT     13H                     ;写回 BOOT 扇区
                JC      L_0127                  ;操作失败

                MOV     AX,[BX+16H]             ;FAT 数目
                MUL     BYTE PTR [BX+10H]       ;剩以每个 FAT 占用扇区数
                ADD     AX,[BX+0EH]             ;加保留扇区
                ADD     AX,[BX+1CH]             ;加隐含扇区=根目录区起始逻辑
                                                ;扇区号
                XOR     DX,DX
                DIV     WORD PTR [BX+18H]       ;除每道扇区数
                MOV     CX,DX
                INC     CL                      ;CL=绝对扇区数
                XOR     DX,DX
                DIV     WORD PTR [BX+1AH]       ;除磁头个数
                MOV     CH,AL                   ;磁头号
                MOV     DH,DL                   ;磁道号的低 8 位数据
                MOV     AL,40H
                MUL     AH
                ADD     CL,AL                   ;磁道号的高 2 位数据放入 CL 中
                MOV     DL,80H                  ;C 盘
                MOV     BX,300H                 ;读入数据存放地址
                MOV     AX,220H                 ;读 20H 个扇区,相当于 512 个目录项
                                                ;即将硬盘的根目录区全部读入内存
                INT     13H
                JC      L_0127                  ;操作失败

                PUSH    DX
                PUSH    CX
                MOV     DL,3
                MOV     AH,36H                  ;取磁盘总簇数
                INT     21H
                MOV     DS:332H,DX              ;磁盘总簇数->第二个系统文件的
                                                ;目录项中,该单元一般不会改变
                MOV     BX,300H                 ;目录项首地址
L_01A5:
;查找213子目录
                CMP     BYTE PTR [BX],0
                JE      L_01C0                  ;没有 213 子目录
```

```
                CMP     WORD PTR [BX],3132H
                JNE     L_01B7
                CMP     WORD PTR [BX+2],2033H
                JE      L_01C3                  ;是 213 子目录
L_01B7:
                ADD     BX,20H
                CMP     BX,4300H
                JB      L_01A5                  ;查下一个目录项
L_01C0:
                JMP     L_0127                  ;错误结束

L_01C3:
                MOV     AX,[BX+1AH]             ;取 213 子目录的起始簇号
                NOT     AX                      ;取反
                MOV     DS:334H,AX              ;也保存在第二个系统文件的目录
                                                ;项中
                POP     CX
                POP     DX                      ;恢复根目录区物理地址
                MOV     BX,300H
                MOV     AX,301H
                INT     13H                     ;将修改后的数据写回，即保存数据
                JNC     L_01DA
                JMP     L_0127                  ;操作失败

;操作成功
L_01DA:
                MOV     DX,OFFSET D_01F4
                MOV     AH,9
                INT     21H                     ;显示操作成功信息

                XOR     DL,DL                   ;错误码=0，表示成功
L_01E3:
                CLD
                MOV     DI,OFFSET DS:[100H]
                MOV     CX,0DDH
                XOR     AL,AL
                REP     STOSB                   ;清除程序内容
                MOV     AL,DL
                MOV     AH,4CH
                INT     21H                     ;程序结束

D_01F4          DB      '2.13H 工作参数已安装好!'
                DB      7, 7,7,'$'              ;操作成功后的显示信息
CODE            ENDS
                END     L_0100
```

第二节　汉化C盘DOS系统　HHCDOS．COM

一、程序功能

查找当前硬盘DOS的系统文件，将一些屏蔽汉字的指令改掉，使得硬盘的DOS支持汉字文件名。

二、程序清单

```
;HHCDOS.ASM
```

```
CODE            SEGMENT
                ASSUME CS:CODE,DS:CODE
                ORG     100H
L_0100:
                JMP     SHORT L_0150
;数据区
                DB      0
D_0103          DB      'C:\GWDOS.COM',0        ;GWDOS的系统文件名
D_0110          DB      'C:\IBMDOS.COM',0       ;PC DOS的系统文件名
                DB      0,0
D_0120          DB      'C:\MSDOS.SYS',0        ;MS DOS的系统文件名
                DB      0,0,0
D_0130          DB      'DOS读出错!',7,'$'      ;错误信息
D_013C          DB      '汉字DOS已经生成!',0DH,0AH,'$' ;操作成功信息
                DB      0

L_0150:
                MOV     DX,OFFSET D_0103        ;GWDOS地址
                MOV     CX,20H
                MOV     AX,4301H                ;设置文件为一般属性
                INT     21H
                JNC     L_017C                  ;操作成功
;不是GWDOS
                MOV     DX,OFFSET D_0110        ;取IBMDOS地址
                MOV     AX,4301H
                INT     21H
                JNC     L_017C                  ;操作成功
;不是PC DOS
                MOV     DX,OFFSET D_0120        ;MSDOS地址
                MOV     AX,4301H
                INT     21H
                JNC     L_017C                  ;操作成功
;没有操作系统在硬盘上
L_0171:
                MOV     DX,OFFSET D_0130
                MOV     AH,9
                INT     21H                     ;显示错误信息
                MOV     AL,1                    ;返回错误码
                JMP     SHORT L_01EF            ;错误结束

L_017C:
                MOV     AX,3D02H                ;以读写方式打开文件
                INT     21H
                JC      L_0171                  ;操作失败
                MOV     BX,AX                   ;BX=文件句柄
                PUSH    DX                      ;保存文件名地址
                MOV     CX,0FFFFH               ;64K
                MOV     DX,300H                 ;缓冲区地址
                MOV     AH,3FH                  ;读文件功能
                INT     21H                     ;读入打开的系统文件
                JNC     L_0195
                POP     DX
                JMP     SHORT L_0171            ;操作失败

L_0195:
```

```
                PUSH    BX              ;保存文件句柄
                PUSH    AX              ;保存读入的长度
                MOV     AH,30H          ;取DOS版本号
                INT     21H             ;AL=主版本号,AH=次版本号
                POP     CX              ;CX=读入的字节数
                POP     BX              ;恢复文件句柄
                CMP     AL,2
                JA      L_01B5          ;DOS版本号>2

                PUSH    CX              ;保存读入的字节数
                DEC     CX
                MOV     SI,300H         ;文件内容的起始地址
L_01A6:
                MOV     AX,[SI]
                CMP     AX,7F24H        ;7F24H 即为指令 AND AL,7FH的
                                        ;二进制码. 这条程序将屏蔽字
                                        ;符(AL)的高位值, 使得系统不支
                                        ;汉字文件名等. 因此必须改掉
                JNE     L_01B1
                MOV     AH,0FFH
                MOV     [SI],AX         ;改为AND AL,0FFH, 这样就不会
                                        ;汉字屏敝了
L_01B1:
                INC     SI
                LOOP    L_01A6          ;继续修改

                POP     CX              ;恢复文件长度

L_01B5:
                PUSH    CX              ;保存文件长度
                DEC     CX
                MOV     SI,300H         ;缓冲区首址

L_01BA:
                MOV     AX,[SI]
                CMP     AX,0A73CH       ;0A73CH 即指令 CMP AL,0A7H
                JNE     L_01C5
                MOV     AH,0A0H         ;改为CMP AL,0A0H.(汉字的机内
                                        ;码是从0A1H开始的)
                MOV     [SI],AX
L_01C5:
                INC     SI
                LOOP    L_01BA          ;继续查找

                XOR     CX,CX
                XOR     DX,DX
                MOV     AX,4200H
                INT     21H             ;将文件指针移到文件头
                POP     CX              ;恢复文件长度
                MOV     DX,300H         ;修改后的文件内容地址
                MOV     AH,40H          ;写文件功能
                INT     21H             ;修改系统文件
                MOV     AH,3EH
                INT     21H             ;关闭文件
                POP     DX              ;文件名地址
```

```
                MOV     CX,27H                  ;加系统隐含只读属性
                MOV     AX,4301H
                INT     21H                     ;设置文件属性

                MOV     DX,OFFSET D_013C
                MOV     AH,9
                INT     21H                     ;显示操作成功信息

                XOR     AL,AL                   ;程序返回码=0
L_01EF:
                MOV     AH,4CH
                INT     21H                     ;返回DOS
CODE            ENDS
                END     L_0100
```

第三节　菜单选择　MENUHH. COM

一、程序功能

显示字库选择菜单，将用户输入的选择号作为错误码返回，使系统按用户的意图进行启动。

二、程序清单

```
;MENUHH.ASM
CODE            SEGMENT
                ASSUME CS:CODE,DS:CODE
                ORG     100H

L_0100:
                LEA     DX,OFFSET D_0111        ;DX=提示菜单地址
                MOV     AH,9
                INT     21H                     ;显示菜单
                MOV     AX,0C01H
                INT     21H                     ;等待输入一个字符
                                                ;输入字符在AL中
                MOV     AH,4CH
                INT     21H                     ;程序结束，返回码为AL
                                                ;中的值，即选择项

D_0111          DB      '1 __ 2.13H __ ALL HZK16 ON HARDRIVE',0DH,0Ah
                DB      '2 __ DOS X.XX',0DH,0Ah
                DB      '3 __ 2.13H __ ALL HZK16 ON VDISK',0DH,0Ah
                DB      'CR _ 2.13H __ 1 HALF HZK16 TO MEMORY',0DH,0Ah
                DB      '       Qing Xuanze : $'        ;选择菜单
CODE            ENDS
                END     L_0100
```

第四节　特殊显示　INT10F. COM

一、程序功能

2．13H汉字系统的特殊显示功能是指在按行显示16×16点阵汉字和字符之外，还可以显示24×24点阵汉字及字符，以及在屏幕上画点、线、矩形及图形填充等，从而改善了一些应用程序的屏幕显示效果。这一功能是2．13H所特有的。

INT10F. COM通过修改显示中断程序INT　10H，使得当用户调用9号功能进行显示时，

首先检查显示字符是否为特殊显示控制字符，如果不是，则调用原ＩＮＴ１０Ｈ完成显示功能。否则，按控制码执行相应的控制程序，实现特殊显示的功能。

使用特殊显示功能实际上就是显示特殊显示控制字符串的过程，同时，必须保证显示的字符是调用ＩＮＴ　１０Ｈ的９号功能实现的（一般高级语言中的字符显示都是调用该功能实现的）。

特殊显示的控制码必须以字符ＣＨＲ（１４）（该字符的ＡＳＣＩＩ码为１４）作为引导字符，紧跟命令字符和该命令的参数，最后以］结尾。

２．１３ＨＣＣ版的特殊显示功能说明如下：

```
CHR(14)+"A扩展ASCII码]"                    模拟功能键
CHR(14)+"B宽,高]"                          画矩形(当前点为左下角)
CHR(14)+"C色号]"                           设置图形颜色
CHR(14)+"D点X,Y]"                          画点
CHR(14)+"E长度]"                           向右上画线
CHR(14)+"F长度]"                           向右下画线
CHR(14)+"G长度]"                           向左下画线
CHR(14)+"H长度]"                           向左上画线
CHR(14)+"I寄存器参数串]"                   执行INT10中断
  其中:寄存器参数串为 AH,AL,BH,BL,CH,CL,DH,DL
    AH=0,AL=显示方式(参见十六)                    改变显示方式
    AH=1,CH=光标起始线,CL=结束线                  设定光标大小
    AH=2,BH=页号(字符方式有效),DH=行号,DL=列号    设定光标位置
    AH=6,AL=行数(=0 全滚),CH=滚动窗口左上角行号,CL=列号
        DH=滚动窗口右下角行号,DL=列号             上滚当前页
    AH=7(其它参数同上)                            下滚当前页
    AH=9,AL=ASCII 码,BH=页号(字符方式有效),BL=属性
        CX=字符个数(汉字无效)                     在当前光标位置显示字符
    AH=10(其它参数同上)                           同　　上
    AH=11,BH=0 置色(保留高亮属性);非 0 取消高亮属性
        BL=色号(BH=0 时)                          置屏幕彩色
    AH=12,AL=色号,DX=Y 座标(0~199),CX=X 座标(0~639)  在指定座标处写点
    AH=14,AL=ASCII 码,BH=页号(字符方式有效),BL=属性  以 TTY 方式显示字符
    AH=16,AL=0                                    清提示行
        AL=1,DL=ASCII码                           在提示行光标处显示字符
        AL=2,DL=位置(0~79)                        提示行光标定位
        AL=3,DL=ASCII码                           以TTY方式显示字符
        AL=4(同上,反相显示字符)
    AH=19,AL=0 取消;非 0 建立                     建立／取消光标
CHR(14)+"J]"                               执行命令串
CHR(14)+"K比例因子]"                       设置比例因子(1~255)
CHR(14)+"L终点X,Y]"                        画斜线(若X有符号则为相对座标)
CHR(14)+"M...]"                            前缀：移当前点但不画线
CHR(14)+"N...]"                            前缀：画线但不移当前点
CHR(14)+"O]"                               画边框线
CHR(14)+"P内色号,边界色号]"                填充
CHR(14)+"Q]"                               清屏,初始化工作参数
CHR(14)+"R控制字]"                         光标控制(奇数建立,偶数取消)
CHR(14)+"S前景色号,背景色号]"              设置字符和汉字颜色
CHR(14)+"T调色板号]"                       设置调色板
CHR(14)+"U长度]"                           向上画线
CHR(14)+"V 色号,左下角 X,Y,宽,高,类型,间隔,线宽]"  矩形区填线
    其中:类型=1 为横线,=2 为竖线,=3 为右斜线,=4 为左斜线
CHR(14)+"W...]"                            定义命令串(到]为止)
CHR(14)+"X长度]"                           向下画线
CHR(14)+"Y长度]"                           向右画线
CHR(14)+"Z长度]"                           向左画线
CHR(14)+"[功能符、汉字或字符]"             显示24×24点阵汉字或字符
```

其中:功能符为 @字型　指定字型(A-P 同 24 点阵字型,a~p 为 A~P 横纵均扩大二倍,
t 为宋体 192*192,u~x 为 16 点阵.开机字型为 A)

^间距　指定字符间距(0-255,汉字间距加倍,开机为 0)

&间距　指定行间距(0-255,开机为1)

_点数　水平定位(0-639,开机为0)

|点数　垂直定位(0-199,开机为0)

*色号　指定前景颜色(0-7,开机为1)

#色号　指定背景颜色(0-7,开机为0)

\0或1　0为汉字背景复盖原图形,1不复盖(开机为不复盖)

注：功能符(A~Z)大小写均可。参数可自动四舍五入为整数，用逗号分隔。0 可用(!)互换。色号 0~15(参看十六)，CGA方式中色号为奇数时按字符前景色显示图形，为偶数时显示黑色图形。划线长度除向下外均指水平宽度。

垂直定位范围:CC11-16-25 为 0~199,CE21-26 和 CH21 为 0~349,CV26 为 0~479,CL25 为 0~399

二、变量名表

地址	长度	意义
00D0h	2	最大颜色号
00D1h	2	当前列坐标
00D3h	2	当前行坐标
00D5h	2	屏幕宽度
00D7h	2	屏幕高度
00D9h	2	列步长
00DBh	2	行步长
00DDh	2	保存当前列坐标(在画线不移点状态下)
00DFh	2	保存当前行坐标(在画线不移点状态下)
00E0h	8	存放设置的寄存器值(AX-DX)
00E1h	2	保存列长度
00E1h	1	填充时，保存填充颜色
00E2h	1	保存边界颜色
00E3h	2	保存行长度
00E5h	2	宽或高的一半
00E7h	2	行或列步长控制计数值
00E9h	2	画斜线时，循环计数值
00E1h	9	在填充时，用存放临时数据
00EBh	2	保存行步长
00EDh	1	当前显示颜色
00EEh	1	=1表示处于画线不移点状态
00EFh	1	=1表示处于移点不画线状态
00F0h	1	显示时，当前显示字型
00F1h	1	显示时，当前显示前景色
00F2h	1	显示时，当前显示背景色
00F3h	2	字间距
00F5h	2	行间距
00F7h	2	当前行末地址
00FBh	1	显示方式
00FDh	1	显示属性值，=0表示不控制
00FEh	1	命令串长度或命令串写入地址
00FFh	1	是否已经显示过控制字符CHR(14)，=0没有
0102h	1	比例因子(缺省为4)
09D0h	2	矩形填充时，颜色
09D2h	2	矩形填充时，左下角列坐标

09D4h	2	矩形填充时，左下角到坐标
09D6h	2	矩形填充时，矩形宽度
09D8h	2	矩形填充时，矩形高度
09DAh	2	矩形填充时，填充类型
09DCh	2	矩形填充时,间隔
09DEh	2	矩形填充时，线宽
09E0h	2	矩形填充时，存放临时数据
09E2h	2	矩形填充时，存放临时数据
0B80h		命令串保存地址(用于直接执行)

三、程序清单

```
;INT10F.ASM
CODE            SEGMENT
                ASSUME CS:CODE, DS:CODE
                ORG    100H

L_0100:
                JMP    L_0B80                ;转初始化程序

;子程序: 将数字串转换为数值，并处理四舍五入
;输   入: SI=数字串首址
;输   出: AX=数值
L_0103:
                XOR    CX,CX                 ;数值=0
                XOR    DL,DL                 ;清小数点标志
L_0107:
                LODSB                        ;取字符
                CMP    AL,20H                ;为空格吗?
                JE     L_0107                ;忽略空格
                CMP    AL,2EH                ;是小数点吗?
                JNE    L_0119                ;不是

;处理小数点
                LODSB                        ;再取一个数字
                CMP    AL,35H                ;四舍五入处理
                JB     L_0116                ;小于5不用舍入
                INC    CX                    ;舍入,数值加1
L_0116:
                INC    DX                    ;置小数点标志，以下的数字不用
                                             ;计算了
                JMP    SHORT L_0107          ;继续处理

;处理其它数字和字符
L_0119:
                CMP    AL,30H                ;小于0吗?
                JB     L_0135                ;是,结束处理
                CMP    AL,3AH                ;大于0吗?
                JA     L_0135                ;是,结束处理

;计算数值
                OR     DL,DL                 ;是小数点后的数据吗?
                JNZ    L_0107                ;是,忽略
```

```
                AND     AL,0FH                  ;将数字转换为数值
                XOR     AH,AH                   ;AH=0
                MOV     BX,AX                   ;BX=当前数字的值
                MOV     AL,0AH                  ;AL=10
                MUL     CX                      ;将原数剩以10
                ADD     AX,BX                   ;加上当前数
                MOV     CX,AX                   ;将数值保存于CX中
                JMP     SHORT L_0107            ;继续处理

;处理结束
L_0135:
                MOV     AX,CX                   ;AX=数值
                DEC     SI                      ;SI指向最后的一个数字位置
                RETN

;子程序: 调用原 INT 10H
;输   入: 寄存器值已设置
;输   出: 原 INT 10H 返回值
L_0139:
                PUSHF
;               call    far ptr s_0000_0000
                DB      9AH,0,0,0,0             ;这里的子程序地址在初始化时已
                                                ;被修改为原INT 10H地址
                RETN

;子程序: 在当前坐标画点(若处于移点不画线状态时, 不画点)
;输   入: AL=颜色号
;输   出: 无
L_0140:
                CMP     BYTE PTR DS:[0EFH],0    ;是在移点不画线状态吗?
                JNE     L_015A                  ;是
L_0147:
                MOV     CX,DS:[0D1H]            ;取列坐标
                MOV     DX,DS:[0D3H]            ;取行坐标
L_014F:
                MOV     AL,DS:[0EDH]            ;当前显示颜色
L_0152:
                PUSH    BX                      ;保存BX
                XOR     BX,BX                   ;BH=页号=0
                MOV     AH,0CH                  ;画点子功能号
                INT     78H                     ;调用原显示中断程序

                POP     BX                      ;恢复BX
L_015A:
                RETN

;子程序: 在画线不移点状态下, 保存当前坐标位置, 以便在画完后恢复
;输   入: 无
;输   出: 无
L_015B:
                CMP     BYTE PTR DS:[0EEH],0    ;在画线不移点状态吗?
                JE      L_016E                  ;不是

;保存当前坐标位置
                MOV     AX,DS:[0D1H]            ;取列坐标
```

```
                MOV     DS:[0DDH],AX            ;保存列坐标
                MOV     AX,DS:[0D3H]            ;取行坐标
                MOV     DS:[0DFH],AX            ;保存行坐标

L_016E:
                RETN
;子程序: 清除移点不画线标志, 当在画线不移点状态下, 恢复当前坐标位置, 清除画线
;       不移点标志. 即恢复画线前状态
;输  入: 无
;输  出: 无
L_016F:
                MOV     BYTE PTR DS:[0EFH],0    ;清移点不画线标志

                CMP     BYTE PTR DS:[0EEH],0    ;在画线不移点状态吗?
                JE      L_018C                  ;不是

                MOV     AX,DS:[0DDH]
                MOV     DS:[0D1H],AX            ;恢复列坐标
                MOV     AX,DS:[0DFH]
                MOV     DS:[0D3H],AX            ;恢复行坐标

                MOV     BYTE PTR DS:[0EEH],0    ;清画线不移点标志
L_018C:
                RETN

;INT 10H
;控制字符串以 CHR(14)开始, 执行特殊功能

L_018D:
                PUSH    DS
                PUSH    AX                      ;保存寄存器

                PUSH    CS
                POP     DS                      ;DS=CS

                CMP     AH,9                    ;是显示字符吗?
                JNE     L_01AD                  ;不是

                CMP     BYTE PTR DS:[0FFH],0EH  ;已显示过CHR(14)字符吗?
                JE      L_01C2                  ;是

;没有显示过 CHR(14),仅判断当前显示字符是否为 CHR(14)
                CMP     AL,0EH                  ;是CHR(14)吗?
                JE      L_01B3                  ;是

;不是 CHR(14),进行正常显示,此时,若属性控制单元值不为 0,则修改显示属性
                MOV     AL,DS:[0FDH]            ;取显示属性值
                OR      AL,AL                   ;要修改显示属性吗?
                JZ      L_01AD                  ;不要,仍按原属性显示

;修改显示属性
                AND     BL,80H                  ;保留BL的高位属性
                OR      BL,AL                   ;修改属性字节
L_01AD:
                POP     AX
```

```
                POP     DS                      ;恢复寄存器

                CALL    L_0139                  ;调用原INT 10H

                IRET                            ;中断返回
;字符是 CHR(14),设置已输入 CHR(14)标志
L_01B3:
                MOV     BYTE PTR DS:[0FFH],AL   ;置控制字符串开始标志
                JMP     SHORT L_01BF            ;中断返回

L_01B8:
                POP     BP
                POP     DI
                POP     SI
                POP     DX
                POP     CX
                POP     ES
L_01BE:
                POP     BX
L_01BF:
                POP     AX
                POP     DS                      ;恢复寄存器

                IRET                            ;中断返回

;已输入CHR(14)
L_01C2:
                PUSH    BX                      ;保存BX

                MOV     BL,DS:[0FEH]            ;取当前命令串保存地址
                XOR     BH,BH
                MOV     [BX],AL                 ;保存字符
                INC     BYTE PTR DS:[0FEH]      ;修改保存地址

                CMP     AL,5DH                  ;']',是命令结束符吗?
                JNE     L_01BE                  ;不是,中断结束

;输入了']', 表示当前控制字符串已结束, 下面执行特殊显示功能
                PUSH    ES
                PUSH    CX
                PUSH    DX
                PUSH    SI
                PUSH    DI
                PUSH    BP                      ;进一步保存寄存器

                PUSH    CS
                POP     ES                      ;ES=CS

                XOR     AX,AX                   ;AX=0
                MOV     DS:[0FEH],AX            ;复位命令串保存地址置0

                MOV     SI,AX                   ;SI=命令串首址
                CALL    L_01E7                  ;执行特殊功能
                JMP     SHORT L_01B8
```

```
;子程序: 执行特殊显示功能, 可以递归执行
;输   入: SI=命令串首址
;输   出: 无
L_01E7:
                LODSB                   ;取功能符
                AND     AL,5FH          ;转换为大写字符

                CMP     AL,5BH          ;是[吗?
                JNE     L_01F1          ;不是
                JMP     L_071A          ;[,在屏幕上显示特殊字符和汉字

L_01F1:
                CMP     AL,5DH          ;是]吗?
                JNE     L_01F6          ;不是
                RETN                    ;命令串结束,执行完毕

L_01F6:
                CMP     AL,41H          ;是A吗?
                JNE     L_0203          ;不是

;模拟键盘输入功能键
;格式:CHR(14)+'A 扩展 ASCII 码]'
                CALL    L_0103          ;取扩展ASCII码->AL
                MOV     AH,6            ;模拟输入
                INT     16H             ;调用键盘管理中断程序
                JMP     SHORT L_01E7

L_0203:
                CMP     AL,42H          ;是B吗?
                JNE     L_0252          ;不是

;画矩形,当前点为矩形的左下角
;格式:CHR(14)+'B 宽,高]'
                CALL    L_0103          ;取宽
                CALL    L_02DE          ;进行比例转换
                MOV     DI,AX           ;DI=宽
                INC     SI              ;指向下一数字串地址
                CALL    L_0103          ;取高
                CALL    L_02DE          ;进行比例转换
                PUSH    SI              ;保存当前命令串地址

                MOV     SI,AX           ;SI=高
                MOV     CX,DS:[0D1H]    ;CX=当前列坐标
                MOV     DX,DS:[0D3H]    ;DX=当前行坐标
                SUB     DX,SI           ;CX,DX为矩形的左上角坐标
                CALL    L_022D          ;画矩形(CX,DX起始坐标,DI=宽,
                                        ;SI=高)
                POP     SI              ;恢复命令串地址
                JMP     SHORT L_01E7    ;当前命令执行结束

                DB      0, 0, 0, 0              ;没有用

;子程序: 画矩形
;输   入: CX=左上角列坐标
;         DX=左上角行坐标
```

```
;          SI=矩形高度, DI=矩形宽度
;输  出: 无
L_022D:
;显示上面一条横线
                MOV     BP,DI                   ;BP=宽L_022F:
                CALL    L_014F                  ;显示一点
                INC     CX                      ;指向下一坐标
                DEC     BP                      ;计数减1
                JNZ     L_022F                  ;连续显示BP个点

;显示右边一条竖线
                MOV     BP,SI                   ;BP=高
L_0238:
                CALL    L_014F                  ;显示一点
                INC     DX                      ;指向下一坐标
                DEC     BP                      ;计数减1
                JNZ     L_0238                  ;连续显示BP个点

;显示下面一条横线
                MOV     BP,DI                   ;BP=宽
L_0241:
                CALL    L_014F                  ;显示一点
                DEC     CX                      ;指向下一坐标
                DEC     BP                      ;计数减1
                JNZ     L_0241                  ;连续显示BP个点

;显示左边一条竖线
                MOV     BP,SI                   ;BP=高
L_024A:
                CALL    L_014F                  ;显示一点
                DEC     DX                      ;指向下一点坐标
                DEC     BP                      ;计数减1
                JNZ     L_024A                  ;连续显示BP个点
                RETN

L_0252:
                CMP     AL,43H                  ;是C吗?
                JNE     L_025E                  ;不是

;设置显示颜色
;格式:CHR(14)+'C 色号]'
                CALL    L_0103                  ;取色号
                MOV     DS:[0EDH],AL            ;修改色号
                JMP     SHORT L_01E7

L_025E:
                CMP     AL,44H                  ;是D吗?
                JNE     L_0275                  ;不是

;画点,也可以用来设置当前坐标
;格式:CHR(14)+'D 列,行]'
                CALL    L_0103                  ;取列坐标
                MOV     DS:[0D1H],AX            ;保存当前列坐标
                INC     SI                      ;指向下一数字串位置
                CALL    L_0103                  ;取行坐标
```

```
                MOV     DS:[0D3H],AX            ;保存行坐标
                CALL    L_0140                  ;显示点
                JMP     L_01E7
L_0275:
                CMP     AL,45H                  ;是E吗?
                JNE     L_027E                  ;不是

;向右上画线
;格式:CHR(14)+'E 长度]'
                MOV     AX,1                    ;列步长=1
                JMP     SHORT L_02A1            ;转2A1继续

L_027E:
                CMP     AL,46H                  ;是F吗?
                JNE     L_0287                  ;不是

;向右下画线
;格式:CHR(14)+'F 长度]'
                MOV     AX,1                    ;列步长=1
                JMP     SHORT L_0297

L_0287:
                CMP     AL,47H                  ;是G吗?
                JNE     L_0290                  ;不是

;向左下画线
;格式:CHR(14)+'G 长度]'
                MOV     AX,0FFFFH               ;列步长=-1
                JMP     SHORT L_02AC

L_0290:
                CMP     AL,48H                  ;是H吗?
                JNE     L_029B                  ;不是

;向左上画线
;格式:CHR(14)+'H 长度]'
                MOV     AX,0FFFFH               ;列步长=-1
L_0297:
                MOV     BX,AX                   ;行步长=列步长
                JMP     SHORT L_02C3

L_029B:
                CMP     AL,55H                  ;是U吗?
                JNE     L_02A6                  ;不是

;向上画线
;格式:CHR(14)+'U 长度]'
                XOR     AX,AX                   ;列步长=0
L_02A1:
                MOV     BX,0FFFFH               ;行步长=-1
                JMP     SHORT L_02C3

L_02A6:
                CMP     AL,58H                  ;是X吗?
                JNE     L_02B1                  ;不是
```

```
;向下画线
;格式:CHR(14)+'X 长度]'
                XOR     AX,AX                   ;列步长=0
L_02AC:
                MOV     BX,1                    ;行步长=1
                JMP     SHORT L_02C3

L_02B1:
                CMP     AL,5AH                  ;是Z吗?
                JNE     L_02BA                  ;不是

;向左画线
;格式:CHR(14)+'Z 长度]'
                MOV     AX,0FFFFH               ;列步长=-1
                JMP     SHORT L_02C1

L_02BA:
                CMP     AL,59H                  ;是Y吗?
                JNE     L_02F0                  ;不是

;向右画线
;格式:CHR(14)+'Y 长度]'
                MOV     AX,1                    ;列步长=1
L_02C1:
                XOR     BX,BX                   ;行步长=0

;列步长和行步长均已设置好,开始画线
L_02C3:
                MOV     DS:[0D9H],AX            ;保存列步长
                MOV     DS:[0DBH],BX            ;保存行步长
                CALL    L_0103                  ;取长度
                CALL    L_02DE                  ;进行比例调整
                MOV     BP,AX                   ;BP=长度
                CALL    L_015B                  ;若画线不移点,保存坐标
                CALL    L_0426                  ;画线
                CALL    L_016F                  ;若画线不移点,恢复坐标
                JMP     L_01E7

;子程序: 将 AX 作比例调整, 调整公式为: AX=(AX x 比例) / 4
;输   入: AX=源数值
;输   出: AX=调整后的数值
L_02DE:
                MOV     DL,BYTE PTR DS:[102H]   ;取比例因子
                CMP     DL,4                    ;等于4吗?
                JE      L_02EF                  ;是,不用调整了

                XOR     DH,DH
                MUL     DX                      ;数值剩以比例
                SHR     AX,1
                SHR     AX,1                    ;再除以4

L_02EF:
                RETN
```

```
L_02F0:
                CMP     AL,49H                  ;是I吗?
                JNE     L_032E                  ;不是
;执行 INT 10H 中断
;格式:CHR(14)+'IAH,AL,BH,BL,CH,CL,DH,DL]'
                MOV     DI,0E0H                 ;寄存器值临时存放地址
L_02F7:
                CALL    L_0103                  ;取寄存器值
                STOSB                           ;保存
                CMP     BYTE PTR [SI],2CH       ;是","吗?
                JNE     L_0303                  ;不是,表示参数已结束
                INC     SI                      ;继续读参数
                JMP     SHORT L_02F7

L_0303:
                XOR     AL,AL                   ;AL=0
L_0305:
                CMP     DI,0E8H                 ;是否已读到全部的寄存器
                JE      L_030E                  ;是
                STOSB                           ;没有,则余下的寄存器赋值0
                JMP     SHORT L_0305

L_030E:
                MOV     DI,0E0H                 ;寄存器表起始地址
                MOV     AX,[DI]                 ;取AX
                MOV     BX,[DI+2]               ;取BX
                MOV     CX,[DI+4]               ;取CX
                MOV     DX,[DI+6]               ;取DX
                XCHG    AL,AH                   ;交换AL,AH
                XCHG    BL,BH                   ;交换BL,BH
                XCHG    CL,CH                   ;交换CL,CH
                XCHG    DL,DH                   ;交换DL,DH
                CALL    L_0139                  ;执行INT 10H
                JMP     L_01E7

                DB      0, 0, 0, 0              ;没有用

L_032E:
                CMP     AL,4AH                  ;是J吗?
                JNE     L_0342                  ;不是

;执行命令串,命令串保存在 0B80H 起始的地方
;格式:CHR(14)+'J]'
                PUSH    SI                      ;保存当前命令串地址SI

                MOV     SI,OFFSET DS:[0B80H]    ;SI=命令串首址
                CMP     BYTE PTR [SI],0         ;有命令吗?
                JE      L_033E                  ;没有
                CALL    L_01E7                  ;递归执行命令
L_033E:
                POP     SI                      ;恢复当前命令串
                JMP     L_01E7

L_0342:
                CMP     AL,57H                  ;是W吗?
```

```
                JNE     L_0352                  ;不是
;定义命令串
;格式:CHR(14)+'W 命令字符串]'
                PUSH    CS
                POP     ES                      ;ES=CS
                MOV     DI,OFFSET DS:[0B80H]    ;DI=命令串保存地址
L_034B:
                LODSB
                STOSB                           ;保存一字节
                CMP     AL,5DH                  ;命令串结束了吗?
                JNE     L_034B                  ;没有
                RETN

L_0352:
                CMP     AL,4BH                  ;是K吗?
                JNE     L_035F                  ;不是

;定义比例因子
;格式:CHR(14)+'K 比例因子]'
                CALL    L_0103                  ;取比例因子
                MOV     BYTE PTR DS:[102H],AL   ;保存比例因子
                JMP     L_01E7

L_035F:
                CMP     AL,4CH                  ;是L吗?
                JNE     L_0369                  ;不是

;画斜线,X 表示列坐标,Y 表示行坐标,若有+号或-号,则表示相对当前坐标的位移值
;格式:CHR(14)+'LX,Y]'
                CALL    L_0457                  ;画斜线
                JMP     L_01E7

L_0369:
                CMP     AL,4DH                  ;是M吗?
                JNE     L_0375                  ;不是

;设置画线不移点标志,是前缀命令,即必须与其它命令结合使用
;格式:CHR(14)+'M...]'
                MOV     BYTE PTR DS:[0EEH],1    ;设置画线不移点标志
                JMP     L_01E7

L_0375:
                CMP     AL,4EH                  ;是N吗?
                JNE     L_0381                  ;不是

;设置移点不画线标志,是前缀命令,即必须与其它命令结合使用
;格式:CHR(14)+'N...]'
                MOV     BYTE PTR DS:[0EFH],1    ;设置移点不画线标志
                JMP     L_01E7

L_0381:
                CMP     AL,4FH                  ;是O吗?
                JNE     L_0399                  ;不是

;在屏幕四周画边框线
```

```
;格式:CHR(14)+'O]'
                XOR     CX,CX                   ;列坐标=0
                XOR     DX,DX                   ;行坐标=0
                PUSH    SI                      ;保存SI
                MOV     DI,DS:[0D5H]            ;宽=屏幕长度
                MOV     SI,DS:[0D7H]            ;高=屏幕高度
                CALL    L_022D                  ;执行画矩形子程序
                POP     SI                      ;恢复SI
                JMP     L_01E7

L_0399:
                CMP     AL,50H                  ;是P吗?
                JNE     L_03A3                  ;不是

;用指定颜色填充
;格式:CHR(14)+'P 填充颜色,边界颜色]'
                CALL    L_0588                  ;填充
                JMP     L_01E7

L_03A3:
                CMP     AL,51H                  ;是Q吗?
                JNE     L_03B6                  ;不是

;清屏,初始化工作参数
;格式:CHR(14)+'Q]'
                PUSH    SI                      ;保存SI
                CALL    L_0952                  ;初始化工作参数
                MOV     AL,DS:[0FBH]            ;取显示方式
                XOR     AH,AH
                INT     10H                     ;清屏
                POP     SI                      ;恢复SI
                JMP     L_01E7

L_03B6:
                CMP     AL,52H                  ;是R吗?
                JNE     L_03C5                  ;不是

;建立或取消光标,数值为奇数时建立光标,为偶数时取消光标
;格式:CHR(14)+'R 数值]'
                LODSB                           ;取数字
                AND     AL,1                    ;取低位
                MOV     AH,13H                  ;设置或取消光标子功能
                CALL    L_0139                  ;调用原显示中断程序
                JMP     L_01E7

L_03C5:
                CMP     AL,53H                  ;是S吗?
                JNE     L_03F2                  ;不是

;设置以后显示字符或汉字的前景和背景颜色
;格式:CHR(14)+'S 前景色,边界色]'
                MOV     DI,0FDH                 ;DI=显示属性单元
                CALL    L_0103                  ;取前景颜色
                AND     BYTE PTR [DI],0F0H      ;保存原背景颜色
                OR      [DI],AL                 ;置前景色
```

```
                MOV     BL,AL                   ;BL=前景色
                XOR     BH,BH                   ;调色板号
                MOV     AH,0BH
                CALL    L_0139                  ;设置调色板
                CMP     BYTE PTR [SI],2CH       ;有背景颜色吗?
                JNE     L_03EF                  ;没有
                INC     SI
                CALL    L_0103                  ;取背景颜色
                MOV     CL,4
                SHL     AL,CL                   ;移到高4位
                AND     BYTE PTR [DI],0FH       ;保存前景颜色
                OR      [DI],AL                 ;置背景颜色
L_03EF:
                JMP     L_01E7

L_03F2:
                CMP     AL,54H                  ;是T吗?
                JNE     L_0410                  ;不是

;设置调色板
;格式:CHR(14)+'T 调色板号,颜色]'
                CALL    L_0103                  ;读调色板号
                PUSH    AX                      ;保存AX
                CMP     BYTE PTR [SI],2CH       ;有彩色值吗?
                JNE     L_0405                  ;没有
                INC     SI
                CALL    L_0103                  ;取彩色号
                MOV     BL,AL
L_0405:
                POP     AX
                MOV     BH,AL                   ;BH=调色板号
                MOV     AH,0BH
                CALL    L_0139                  ;调用原显示中断程序
L_040D:
                JMP     L_01E7

L_0410:
                CMP     AL,56H                  ;是V吗?
                JNE     L_040D                  ;不是,则是无效字符
;矩形区域填充
;格式:CHR(14)+'V 颜色,左下角 X,Y,宽,高,类型,间隔长度,线宽]'
;其中:类型=1 为横线,=2 为竖线,=3 为右斜线,=4 为左斜线
                JMP     L_09F0                  ;填充

;子程序: 画线但不破坏寄存器 CX 和 DX 的内容
;输   入: BP=长度-1
;输   出: 无
L_0417:
                PUSH    CX
                PUSH    DX                      ;保存寄存器
                CALL    L_041F                  ;画线(将破坏CX和DX的值)
                POP     DX
                POP     CX                      ;恢复寄存器
                RETN
```

```
;子程序: 画线(包括当前点) ;输  入: BP=长度-1
;输  出: 无
L_041F:
                CALL    L_0140          ;显示第一点
                OR      BP,BP           ;BP=0吗?
                JZ      L_0456          ;是,则不用继续画

;子程序: 画线(不包括当前点)
;输  入: BP=长度
;输  出: 无
L_0426:
;计算列坐标值
                MOV     AX,DS:[0D1H]    ;AX=列坐标
                ADD     AX,DS:[0D9H]    ;加列步长
                CMP     AX,0
                JB      L_0456          ;小于0
                CMP     AX,DS:[0D5H]
                JA      L_0456          ;大于屏幕宽度
                MOV     DS:[0D1H],AX    ;修改列坐标
;计算行坐标
                MOV     AX,DS:[0D3H]    ;AX=行坐标
                ADD     AX,DS:[0DBH]    ;加行步长
                CMP     AX,0
                JB      L_0456          ;小于0
                CMP     AX,DS:[0D7H]
                JA      L_0456          ;大于屏幕高度
                MOV     DS:[0D3H],AX

                CALL    L_0140          ;显示一点
                DEC     BP              ;计数减1
                JNZ     L_0426          ;连续显示BP个点

L_0456:
                RETN

;子程序: 画斜线
;输  入: 无
;输  出: 无
L_0457:
                CALL    L_0569          ;取列坐标或长度,并检查有否正
                                        ;负号
                JNC     L_0488          ;无符号

;下面作有符号的数据处理
                MOV     CX,1            ;先设步长=1
                CMP     BP,2            ;是负号吗?
                JNE     L_0468          ;不是
                NEG     CX              ;负号,步长=-1
                NOP
                NOP
L_0468:
                MOV     DS:[0E1H],AX    ;保存列长度
                MOV     DS:[0D9H],CX    ;保存列步长
                INC     SI
                CALL    L_0569          ;取行坐标或长度,并检查有否正
```

```
                                        ;负号
                MOV     CX,1            ;先设步长=1
                CMP     BP,2            ;是负号吗?
                JNE     L_047F          ;不是
                NEG     CX              ;步长=-1
                NOP
                NOP
L_047F:
                MOV     DS:[0E3H],AX    ;保存行长度
                MOV     DS:[0DBH],CX    ;保存行步长
                JMP     SHORT L_04B4

L_0488:
;无符号数据,列和行是绝对坐标,要转换成相对位移
                MOV     CX,1            ;先设步长=1
                SUB     AX,DS:[0D1H]    ;减当前点列坐标=长度
                JNC     L_0495          ;是负的吗?
                NEG     CX              ;步长=-1
                NEG     AX              ;AX取绝对值
L_0495:
                MOV     DS:[0E1H],AX    ;保存列坐标
                MOV     DS:[0D9H],CX    ;保存列步长
                INC     SI
                CALL    L_0569          ;取行坐标
                SUB     AX,DS:[0D3H]    ;减当前点行坐标=长度
                MOV     CX,1            ;先设步长=1
                JNC     L_04AD          ;是负的吗?
                NEG     CX              ;步长=-1
                NEG     AX              ;AX取绝对值
L_04AD:
                MOV     DS:[0E3H],AX    ;保存行长度
                MOV     DS:[0DBH],CX    ;保存行步长
;步长和长度已设置好
L_04B4:
                CALL    L_0140          ;显示一点(置坐标)
                CALL    L_015B          ;若画线不移点,保存当前坐标

                MOV     AX,DS:[0E1H]    ;取列长度
                CMP     AX,DS:[0E3H]    ;列长度大于行长度吗?
                JB      L_04C8          ;不是
                CALL    L_04CF          ;列长度>=高长度,画斜线
                JMP     SHORT L_04CB
L_04C8:
                CALL    L_051C          ;列长度<高长度,画斜线
L_04CB:
                CALL    L_016F          ;若画线不移点,恢复当前坐标
                RETN

;子程序: 画斜线(列长度>=行长度)
;输  入: 无
;输  出: 无
L_04CF:
                MOV     AX,DS:[0E1H]    ;AX=列长度
                MOV     DS:[0F9H],AX    ;置循环计数单元
                INC     WORD PTR DS:[0E9H]  ;实际长度应加1
```

```
                SHR     AX,1                    ;AX=AX/2
                MOV     DS:[0E5H],AX            ;将宽的一半->[0E5H]
                MOV     WORD PTR DS:[0E7H],0    ;行步长控制计数值=0
                MOV     AX,DS:[0DBH]            ;取行步长
                MOV     DS:[0EBH],AX            ;保存行步长
                JMP     SHORT L_04F2
L_04EC:
                MOV     BP,1                    ;显示一点
                CALL    L_0426                  ;画线
L_04F2:
                MOV     WORD PTR DS:[0DBH],0    ;先设行步长=0
                MOV     AX,DS:[0E3H]            ;行长度->AX
                ADD     AX,DS:[0E7H]
                MOV     DS:[0E7H],AX            ;增加行步长控制计数值
                CMP     AX,DS:[0E5H]            ;超过列长度的一半了吗?
                JLE     L_0515                  ;没有

;当行步长控制计数值超过列长度的一半时,修正行坐标
                SUB     AX,DS:[0E1H]            ;减列长度
                MOV     DS:[0E7H],AX            ;设置新的行步长控制计数值
                MOV     AX,DS:[0EBH]            ;取行步长
                MOV     DS:[0DBH],AX            ;设置行步长
L_0515:
                DEC     WORD PTR DS:[0E9H]      ;计数减1
                JNZ     L_04EC
                RETN

;子程序: 画斜线(列长度<行长度)
;输   入: 无
;输   出: 无
L_051C:
                MOV     AX,DS:[0E3H]            ;取行长度
                MOV     DS:[0E9H],AX            ;送循环计数单元
                INC     WORD PTR DS:[0E9H]      ;实际长度应加1
                SHR     AX,1                    ;AX=AX/2
                MOV     DS:[0E7H],AX            ;将高的一半->[0E7H]
                MOV     WORD PTR DS:[0E5H],0    ;列步长控制计数值=0
                MOV     AX,DS:[0D9H]            ;取列步长
                MOV     DS:[0EBH],AX            ;保存列步长
                JMP     SHORT L_053F
L_0539:
                MOV     BP,1                    ;显示一点
                CALL    L_0426                  ;显示
L_053F:
                MOV     WORD PTR DS:[0D9H],0    ;先设列步长=0
                MOV     AX,DS:[0E1H]            ;列长度->AX
                ADD     AX,DS:[0E5H]
                MOV     DS:[0E5H],AX            ;增加列步长控制计数值
                CMP     AX,DS:[0E7H]            ;超过行长度的一半了吗?
                JLE     L_0562                  ;没有

;当列步长控制计数值超过行长度的一半时,改变列坐标
                SUB     AX,DS:[0E3H]            ;减行长度
                MOV     DS:[0E5H],AX            ;设置新的列步长控制计数值
                MOV     AX,DS:[0EBH]            ;取列步长
```

```
                MOV     DS:[0D9H],AX            ;设置列步长
L_0562:
                DEC     WORD PTR DS:[0E9H]      ;计数减1
                JNZ     L_0539
                RETN

;子程序: 将数字串转化为数值,并进行符号判断
;输   入: SI=数字串首址
;输   出: AX=读出数的绝对值
;         BP=0:没有符号
;         BP=1:有正号
;         BP=2:有负号
;         当 BP<>0 时,CF=1,否则 CF=0
L_0569:
                XOR     BP,BP                   ;BP=0
                CMP     BYTE PTR [SI],2BH       ;有"+"号吗?
                JNE     L_0574                  ;没有
                INC     SI
                INC     BP                      ;BP=1
                JMP     SHORT L_057D
L_0574:
                CMP     BYTE PTR [SI],2DH       ;有"-"号吗?
                JNE     L_057D                  ;没有
                INC     SI
                MOV     BP,2                    ;BP=2
L_057D:
                CALL    L_0103                  ;取数字->AX
                OR      BP,BP                   ;BP=0吗?
                JNZ     L_0586                  ;不是
                CLC                             ;清CF
                RETN
L_0586:
                STC                             ;置CF
                RETN

;子程序: 填色, 将区域划分为上下两部分, 递归执行
;输   入: 无
;输   出: 无
L_0588:
                CALL    L_0103                  ;取填充颜色
                CMP     AL,DS:[0D0H]            ;比最大允许色号还大吗?
                JA      L_060C                  ;是,命令无效
                MOV     DS:[0E1H],AL            ;保存填充颜色
                INC     SI
                CALL    L_0103                  ;取边界颜色
                CMP     AL,DS:[0D0H]            ;比最大允许色号还大吗?
                JA      L_060C                  ;是,命令无效
                MOV     BYTE PTR DS:[0E2H],AL   ;保存边界颜色
                MOV     DX,DS:[0D3H]            ;当前行坐标->DX
                MOV     CX,DS:[0D1H]            ;当前列坐标->CX
                CALL    L_06F6                  ;读当前点颜色
                JC      L_060C                  ;超出屏幕范围,命令无效
                CMP     AL,BYTE PTR DS:[0E2H]   ;当前点颜色和边界色相同
                JE      L_060C                  ;是,则填充结束
;从当前坐标列,向左找具有边界颜色的点,直到屏幕边界为止
```

```
                MOV     BX,CX                   ;BX=列坐标
L_05B6:
                DEC     CX                      ;列坐标减1
                CALL    L_06F6                  ;取点
                JC      L_05C6                  ;到屏幕边界
                CMP     AL,BYTE PTR DS:[0E2H]   ;是边界色吗?
                JE      L_05C6                  ;找到
                MOV     BX,CX                   ;继续找
                JMP     SHORT L_05B6
L_05C6:
                MOV     DS:[0E3H],BX            ;保存左边界列号
                PUSH    BX                      ;保存左边界列号
                MOV     CX,DS:[0D1H]            ;当前列坐标

;从当前坐标列,向右找具有边界颜色的点,直到屏幕边界为止
                MOV     BX,CX                   ;BX=列坐标
L_05D1:
                INC     CX                      ;列坐标加1
                CALL    L_06F6                  ;取点颜色
                JC      L_05E1                  ;到屏幕边界
                CMP     AL,BYTE PTR DS:[0E2H]   ;是边界色吗?
                JE      L_05E1                  ;是
                MOV     BX,CX
                JMP     SHORT L_05D1            ;继续找

L_05E1:
                MOV     DS:[0E5H],BX            ;保存右边界列号
                PUSH    BX                      ;保存右边界列号
                SUB     BX,DS:[0E3H]            ;长度
                INC     BX                      ;实际长度
                MOV     CX,DS:[0E3H]            ;左边界列号
L_05EF:
;画一条横线
                MOV     AL,DS:[0E1H]            ;AL=填充颜色
                CALL    L_0152                  ;写点
                NOP
                INC     CX
                DEC     BX
                JNZ     L_05EF                  ;下一个

                PUSH    DX                      ;保存行坐标
                DEC     DX                      ;DX指向上一扫描线
                CALL    L_060D                  ;填充上面半个区域
                POP     DX                      ;恢复DX
                INC     DX                      ;DX指向下一扫描线
                POP     AX                      ;恢复右边界列号
                MOV     DS:[0E5H],AX
                POP     AX                      ;恢复左边界
                MOV     DS:[0E3H],AX
                CALL    L_060D                  ;填充下面半个区域
L_060C:
                RETN

;子程序: 填充半边区域;输   入: [E3]=左边界列号
;         [E5]=右边界列号
```

```
;输  出: 无
L_060D:
                MOV     CX,DS:[0E3H]            ;取左边界列号
L_0611:
                MOV     DS:[0E7H],CX            ;保存左边界列号
                MOV     CX,DS:[0E7H]
                CALL    L_06F6                  ;取当前点颜色
                JNC     L_061F
                RETN                            ;超出屏幕边界

L_061F:
                CMP     AL,BYTE PTR DS:[0E2H]   ;是边界色吗?
                JE      L_0647                  ;是
                CMP     AL,DS:[0E1H]            ;是填充色吗?
                JE      L_0647                  ;是

;处理右边不连续边界
                MOV     DS:[0E9H],CX            ;将当前左边界作为右边界列号保存
;从当前左边界列号再向左寻找边界点(产生另一区间)
                MOV     BX,CX                   ;BX=右边界列号
L_0631:
                DEC     CX                      ;列号减1
                CALL    L_06F6                  ;取点
                JC      L_0641                  ;到屏幕边界了
                CMP     AL,BYTE PTR DS:[0E2H]   ;是边界色吗?
                JE      L_0641                  ;是
                MOV     BX,CX
                JMP     SHORT L_0631            ;不是,继续找
L_0641:
                MOV     DS:[0E7H],BX            ;保存新的左边界列号
                JMP     SHORT L_066A

;左边是连续边界,下面两个边界中间的点
;过滤中间的边界点和已填充色点,直到右边界点或屏幕边界点
L_0647:
                INC     CX                      ;增加列号
L_0648:
                CMP     CX,DS:[0E5H]            ;到右边界
                JG      L_0662                  ;到了
                CALL    L_06F6                  ;取点
                JC      L_0662                  ;到屏幕边界
                CMP     AL,DS:[0E1H]            ;是填充颜色吗?
                JE      L_065F                  ;是
                CMP     AL,BYTE PTR DS:[0E2H]   ;是边界颜色吗?
                JNE     L_0662                  ;不是
L_065F:
                INC     CX
                JMP     SHORT L_0648            ;继续找

L_0662:
                MOV     DS:[0E7H],CX            ;新的左边界点
                MOV     DS:[0E9H],CX            ;右边界点=左边界点
L_066A:         MOV     BX,DS:[0E7H]
                CMP     BX,DS:[0E5H]            ;已全部填满了吗?
                JLE     L_0675                  ;没有
```

```
                RETN
L_0675:
;寻找新的右边界点
                MOV     BX,DS:[0E9H]
                MOV     CX,BX                   ;右边界点
L_067B:
                INC     CX                      ;列号加1
                CALL    L_06F6                  ;取点
                JC      L_068B                  ;已到边界点
                CMP     AL,BYTE PTR DS:[0E2H]   ;是边界色吗?
                JE      L_068B                  ;是
                MOV     BX,CX
                JMP     SHORT L_067B            ;继续找
L_068B:
                MOV     DS:[0E9H],BX            ;保存新的右边界点列号
                MOV     CX,DS:[0E7H]            ;CX=新的左边界点列号
                SUB     BX,CX                   ;BX=长度
                INC     BX                      ;实际长度要加1
;画一条横线
L_0696:
                MOV     AL,DS:[0E1H]            ;当前显示颜色
                CALL    L_0152                  ;显示一点
                NOP
                INC     CX
                DEC     BX
                JNZ     L_0696                  ;画BX次

                PUSH    WORD PTR DS:[0E3H]      ;保存当前左边界点
                PUSH    WORD PTR DS:[0E5H]      ;保存当前右边界点
                MOV     CX,DS:[0E7H]            ;取新的左边界点
                PUSH    CX                      ;保存新的左边界点
                MOV     DS:[0E3H],CX            ;设置左边界点
                MOV     CX,DS:[0E9H]            ;取新的右边界点
                PUSH    CX                      ;保存新的右边界点
                MOV     DS:[0E5H],CX            ;设置右边界点
                PUSH    DX                      ;保存行坐标
                DEC     DX
                CALL    L_060D                  ;填上面半个区域
                POP     DX                      ;恢复右边界点
                POP     CX                      ;恢复左边界点
                MOV     DS:[0E5H],CX            ;恢复右边界点
                MOV     DS:[0E9H],CX            ;恢复新的右边界点
                POP     AX                      ;恢复左边界点
                MOV     DS:[0E3H],AX            ;恢复左边界点
                MOV     DS:[0E7H],AX            ;恢复新的左边界点
                PUSH    CX
                PUSH    DX                      ;保存行坐标
                INC     DX
                CALL    L_060D                  ;填下半个区域
                POP     DX                      ;恢复行坐标
                POP     CX                      ;恢复新的左边界点
                MOV     DS:[0E9H],CX            ;恢复新的左边界点
                ADD     CX,2                    ;起码已填满两点
                MOV     DS:[0E7H],CX            ;新的左边界点
                POP     WORD PTR DS:[0E5H]      ;恢复右边界点
```

```
                POP     WORD PTR DS:[0E3H]      ;恢复右边界点
                CMP     CX,DS:[0E5H]            ;已填满
                JG      L_06F5                  ;是
                JMP     L_0611                  ;没有,重复填色
L_06F5:
                RETN

;子程序: 读当前点颜色,并进行屏幕边界判断
;输   入: CX=列号,DX=行号
;输   出: 当坐标在屏幕中时,AL=颜色,CF=0
;         当坐标不在屏幕中时,AL=0FFH,CF=1
L_06F6:
                CMP     CX,0
                JB      L_0716                  ;边界
                CMP     CX,DS:[0D5H]
                JA      L_0716                  ;边界
                CMP     DX,0
                JB      L_0716                  ;边界
                CMP     DX,DS:[0D7H]
                JA      L_0716                  ;边界

                PUSH    BX                      ;保存BX
                XOR     BX,BX                   ;页号=0
                MOV     AH,0DH                  ;读点子功能
                INT     78H                     ;调原显示中断程序
                POP     BX                      ;恢复BX
                CLC                             ;清除CF
                RETN
L_0716:
;超出屏幕
                MOV     AL,0FFH                 ;AL=0FFH
                STC                             ;设置CF
                RETN

;在屏幕显示特殊字符或汉字
;格式:CHR(14)+'[功能符、字符或汉字]'
;其中:功能符为 @字型   指定字型(A-P 同 24 点阵字型,a-p 为 A-P 横纵均扩大二倍,
;                      t 为宋体 192x192,u-x 为 16 点阵(初始值为 A)
;              ^间距   指定字符间距(0-255,汉字间距加倍,初始值为 0)
;              &间距   指定行间距(0-255,初始值为 0)
;              _数值   开始列坐标(0-总显示列数,初始值为 0)
;              |数值   垂直定位(0-总显示行数,初始值为 0)
;              *颜色   指定前景颜色(0-7,初始值为 1)
;              #颜色   指定背景颜色(0-7,初始值为 0)
;              \0 或 1   0 为汉字背景复盖原图形,1 不复盖原图形(初始值为不复盖)
L_071A:
                MOV     DI,0F0H                 ;控制值存放地址
L_071D:
                LODSB                           ;取字符
                CMP     AL,40H                  ;是@吗?
                JNE     L_073A                  ;不是
;设置显示字型
                LODSB                           ;取字型
                DEC     AX                      ;字型号减1
                MOV     [DI],AL                 ;保存
```

```
                JMP     SHORT L_071D

                DB      18 DUP (0)              ;没有用

L_073A:
                CMP     AL,2AH                  ;是*吗?
                JNE     L_0746                  ;不是

;设置前景色
                CALL    L_0103                  ;取前景颜色
                MOV     DS:[0F1H],AL            ;保存
                JMP     SHORT L_071D

L_0746:
                CMP     AL,23H                  ;是#吗?
                JNE     L_0752                  ;不是

;设置背景色
                CALL    L_0103                  ;取背景颜色
                MOV     DS:[0F2H],AL            ;保存
                JMP     SHORT L_071D

L_0752:
                CMP     AL,5FH                  ;是_吗?
                JNE     L_075E                  ;不是

;设置起始列坐标
                CALL    L_0103                  ;取列坐标
                MOV     DS:[0D1H],AX            ;保存
                JMP     SHORT L_071D

L_075E:
                CMP     AL,7CH                  ;是|吗?
                JNE     L_076A                  ;不是

;设置起始行坐标
                CALL    L_0103                  ;取行坐标
                MOV     DS:[0D3H],AX            ;保存
                JMP     SHORT L_071D

L_076A:
                CMP     AL,5EH                  ;是^吗?
                JNE     L_0776                  ;不是

;指定字符间距
                CALL    L_0103                  ;取间距
                MOV     DS:[0F3H],AX            ;保存
                JMP     SHORT L_071D

L_0776:
                CMP     AL,26H                  ;是&吗?
                JNE     L_0782                  ;不是

;指定行间距
                CALL    L_0103                  ;取间距
```

```
                MOV     DS:[0F5H],AX            ;保存
                JMP     SHORT L_071D

L_0782:
                CMP     AL,5CH                  ;是\吗?
                JNE     L_0794                  ;不是

;设置复盖与不复盖
                CALL    L_0103                  ;取数值
                CMP     AL,0                    ;复盖吗?
                JE      L_078F                  ;是
                MOV     AL,3                    ;不复盖
L_078F:
                MOV     BYTE PTR DS:[94AH],AL   ;修改程序
                JMP     SHORT L_071D

L_0794:
                CMP     AL,5DH                  ;是]吗?
                JNE     L_07A8                  ;不是

;显示结束
L_0798:
                MOV     AX,DS:[0F7H]            ;当前行末线行坐标
                ADD     AX,DS:[0F5H]            ;加行间距
                MOV     DS:[0D3H],AX            ;修改行坐标
                XOR     AX,AX
                MOV     DS:[0D1H],AX            ;指定列坐标为0
                RETN

;不是功能符,要进行显示
L_07A8:
                CMP     AL,0A1H                 ;是汉字吗?
                JAE     L_07B0                  ;是
                XOR     AH,AH                   ;AH=0,AL=ASCII字符
                JMP     SHORT L_07B3

L_07B0:
                MOV     AH,AL                   ;AH=汉字的前一字节
                LODSB                           ;取汉字的后一字节
L_07B3:
                MOV     DX,AX                   ;DX=汉字
                PUSH    SI                      ;保存SI
                XOR     BX,BX                   ;没有修饰字
                MOV     AH,[DI]                 ;取字号
                CMP     AH,74H                  ;是t号字以上
                JAE     L_07CA                  ;是16点阵字型
                AND     AH,0FH                  ;屏蔽扩展位
                INT     7BH                     ;读24点阵汉字点阵
                PUSH    DS
                POP     ES                      ;ES=点阵数据段地址
                PUSH    CS
                POP     DS                      ;DS=CS
                JMP     SHORT L_082B
;取 16 点阵汉字或字符点阵
L_07CA:
```

```
                PUSH    DI                      ;保存DI
                MOV     DI,OFFSET D_09D0        ;字型单元地址
                OR      DH,DH                   ;是汉字吗?
                JNZ     L_0806                  ;是
;读 16 点阵字符点阵
                MOV     ES,WORD PTR DS:[13DH]   ;原显示中断程序段址->ES
                MOV     CL,3
                SHL     DX,CL                   ;DX=DXx8,因为每个字符占8个字节
                ADD     DX,2798H                ;加字符点阵起始地址
                MOV     SI,DX                   ;字符点阵起始地址
                MOV     CX,8                    ;8字节
L_07E3:
;旋转后读入字符点阵
                PUSH    SI                      ;保存点阵起始地址
                MOV     DL,8                    ;8位
L_07E6:
                LODS BYTE PTR ES:[SI]           ;取一排数据
                SHR     AL,CL                   ;第CL位->CF
                RCL     BL,1                    ;将CF移入BL
                DEC     DL                      ;下一点
                JNZ     L_07E6
                MOV     [DI],BL                 ;BL=旋转后的一排数据
                INC     DI                      ;DI=下一地址
                POP     SI                      ;恢复点阵起始地址
                LOOP    L_07E3                  ;取8字节

                MOV     CL,8                    ;列数=8
L_07F8:
                MOV     SI,9D0H                 ;字型单元地址
                POP     DI                      ;恢复DI
                PUSH    CS
                POP     ES                      ;ES=CS
                AND     BYTE PTR [DI],0FH       ;屏蔽扩展位
                OR      BYTE PTR [DI],80H       ;置高位
                JMP     SHORT L_082B

;读 16 点阵汉字点阵
L_0806:
                INT     7FH                     ;读汉字横向点阵
                MOV     DS,DX                   ;点阵段地址->DS
                XOR     SI,SI                   ;偏移地址=0
;将横向点阵转化为纵向点阵
                MOV     CX,10H                  ;共16排
L_080F:
                PUSH    SI                      ;保存点阵数据首址
                MOV     DL,10H                  ;共16位
L_0812:
                LODSW                           ;取一排
                XCHG    AL,AH                   ;在AX中按顺序排列
                SHR     AX,CL                   ;将第CL点->CF
                RCL     BX,1                    ;将CF移入BX
                DEC     DL                      ;下一个点
                JNZ     L_0812
                XCHG    BL,BH                   ;换回来
                MOV     AX,BX                   ;AX=旋转后的点阵
```

```
                STOSW                           ;写一排
                POP     SI                      ;恢复源点阵数据首址
                LOOP    L_080F

                PUSH    CS
                POP     DS                      ;DS=CS
                MOV     CL,10H                  ;列数=16
                JMP     SHORT L_07F8
;点阵已全部读入,CX=列数
L_082B:
                MOV     BX,CX                   ;BX=列数
                TEST    BYTE PTR [DI],1         ;要扩吗?
                JZ      L_0834                  ;不要
                SHL     BX,1                    ;扩大一倍
L_0834:
                CMP     BYTE PTR [DI],60H       ;小于a型字吗?
                JB      L_0840                  ;是
                CMP     BYTE PTR [DI],73H       ;大于t型字吗?
                JA      L_0840                  ;是
                SHL     BX,1                    ;对于a-t型字,再扩大一倍
L_0840:
                MOV     AX,DS:[0D5H]            ;屏幕最大列数->AX
                SUB     AX,DS:[0D1H]            ;减当前列坐标
                JC      L_084D                  ;超出屏幕边界
                CMP     AX,BX                   ;还有足够的宽度去显示吗?
                JAE     L_085C                  ;有
;换行
L_084D:
                XOR     AX,AX                   ;AX=0
                MOV     DS:[0D1H],AX            ;置列坐标=0
                MOV     AX,DS:[0F7H]            ;当前行末线坐标->AX
                ADD     AX,DS:[0F5H]            ;加行间距
                MOV     DS:[0D3H],AX            ;调整当前行坐标
L_085C:
                MOV     AX,DS:[0D3H]            ;AX=当前行坐标
                NOP
                NOP
                NOP
                NOP
                CMP     AX,DS:[0D7H]            ;小于等于屏幕最大行数吗?
                JBE     L_0888                  ;是
;已到屏幕最下面,不能显示了,程序即响玲并等待用户按键,若按 ESC 键,则中止显示,其它
;字符则继续显示
                MOV     DL,7                    ;CHR(7),响玲字符
                MOV     AX,1003H                ;提示显示字符
                CALL    L_0139                  ;响玲
                XOR     AH,AH
                INT     16H                     ;读入一个字符
                CMP     AL,1BH                  ;是ESC键吗?
                JNE     L_087D                  ;不是
                POP     SI                      ;恢复SI
                JMP     L_0798                  ;中止当前命令
;清屏后继续显示
L_087D:
                MOV     AX,DS:[0FBH]            ;取当前显示方式
```

```
                CALL    L_0139                  ;清屏
                XOR     AX,AX
                MOV     DS:[0D3H],AX            ;行坐标=0
L_0888:
;计算每一排占几个字节
                CMP     BYTE PTR [DI],74H       ;是16点阵字型吗?
                JB      L_08A0                  ;不是
                CMP     CL,8                    ;是字符吗?
                JNE     L_0899                  ;不是
                MOV     BYTE PTR DS:[8E5H],1    ;是16点阵字符,每排1字节修改程
                                                ;序
                JMP     SHORT L_08A5

L_0899:
                MOV     BYTE PTR DS:[8E5H],2    ;是16点阵汉字,每排2字节修改程
                                                ;序
                JMP     SHORT L_08A5
L_08A0:
                MOV     BYTE PTR DS:[8E5H],3    ;是24点阵汉字,每排3字节修改程
                                                ;序
L_08A5:
                PUSH    CX                      ;保存CX
                CALL    L_08E4                  ;显示一排
                TEST    BYTE PTR [DI],1         ;要进行横扩吗?
                JZ      L_08B1                  ;不要
                CALL    L_08E4                  ;扩一倍
L_08B1:
                TEST    BYTE PTR [DI],20H       ;是小写字型吗?
                JZ      L_08D2                  ;不是
                CALL    L_08E4                  ;再扩一倍
                TEST    BYTE PTR [DI],1         ;要横扩吗?
                JZ      L_08D2                  ;不要
                CALL    L_08E4                  ;再扩一倍
                TEST    BYTE PTR [DI],10H       ;是扩8倍吗?
                JZ      L_08D2                  ;不是
                CALL    L_08E4
                CALL    L_08E4
                CALL    L_08E4
                CALL    L_08E4                  ;再扩4倍
L_08D2:
                POP     CX                      ;恢复列计数
                ADD     SI,WORD PTR DS:[8E5H]   ;SI指向下一排点阵
                LOOP    L_08A5                  ;显示CX次

                MOV     AX,DS:[0F3H]            ;行间距->AX
                ADD     DS:[0D1H],AX            ;指向下一行首
                POP     SI                      ;恢复SI
                JMP     L_071D

;子程序: 显示一排点阵(在屏幕上为一列)
;输  入: 无;输  出: 无
L_08E4:
                MOV     BP,3                    ;这里是3已在前面被修改为相应
                                                ;的一排的字节数
                PUSH    SI                      ;保存起始地址
```

```
                PUSH    WORD PTR DS:[0D3H]      ;保存当前行坐标
L_08EC:
                LODS BYTE PTR ES:[SI]           ;取一字节
                MOV     BL,AL                   ;BL=点阵
                MOV     BH,8                    ;共8位
L_08F2:
                SHL     BL,1                    ;取一点->CF
                JC      L_08FB                  ;是1吗?
                MOV     AL,DS:[0F2H]            ;不是,取背景颜色
                JMP     SHORT L_08FE
L_08FB:
                MOV     AL,DS:[0F1H]            ;取前景颜色
L_08FE:
                MOV     DS:[0EDH],AL            ;保存颜色
                CALL    L_0944                  ;显示一点
                TEST    BYTE PTR [DI],2         ;要纵扩吗?
                JZ      L_090C                  ;不要
                CALL    L_0944                  ;再写一遍,相对于横扩
L_090C:
                TEST    BYTE PTR [DI],20H       ;是小写字型吗?
                JZ      L_092D                  ;不是
                CALL    L_0944                  ;再扩一次
                TEST    BYTE PTR [DI],2         ;是否纵扩
                JZ      L_092D                  ;不要
                CALL    L_0944                  ;再扩一次
                TEST    BYTE PTR [DI],10H       ;是扩8倍吗?
                JZ      L_092D                  ;不是
                CALL    L_0944
                CALL    L_0944
                CALL    L_0944
                CALL    L_0944                  ;再扩4倍
L_092D:
                DEC     BH                      ;下一位
                JNZ     L_08F2                  ;显示一字节
                DEC     BP                      ;减字节计数
                JNZ     L_08EC                  ;显示BP个字节

                MOV     AX,DS:[0D3H]            ;取行坐标
                MOV     DS:[0F7H],AX            ;设置行末线坐标
                POP     WORD PTR DS:[0D3H]      ;恢复行坐标
                POP     SI                      ;恢复起始地址
                INC     WORD PTR DS:[0D1H]      ;指向下一列
                RETN

;子程序: 显示一点
;输   入: 无
;输   出: 无
L_0944:
                CMP     BYTE PTR DS:[0EDH],0    ;若不复盖,这里的0也已被修改
                JE      L_094E                  ;不复盖
                CALL    L_0147                  ;显示
L_094E:
                INC     WORD PTR [DI-1DH]       ;修改[00D3]值指向下一行扫描线
                RETN
```

```
;子程序: 初始化工作参数
;输   入: 无
;输   出: CF=0:成功
;         CF=1:失败(因为不支持当前显示方式)
L_0952:
                XOR     AX,AX                   ;AX=0
                MOV     DI,0D0H                 ;数据区起始地址
                MOV     CX,33H                  ;长度=33H
                REP     STOSB                   ;清数据区
                PUSH    ES                      ;保存ES
                MOV     ES,AX                   ;ES=0
                MOV     DL,ES:449H              ;取当前显示方式
                POP     ES                      ;恢复ES
                MOV     SI,OFFSET D_09B0        ;取显示方式表地址
                MOV     CL,5                    ;共5种显示方式
L_096A:
                LODSB                           ;取显示方式
                CMP     AL,DL                   ;与当前显示方式相同吗?
                JE      L_0976                  ;是
                ADD     SI,5                    ;查下一种显示方式
                LOOP    L_096A
;不支持当前显示方式
                STC                             ;CF=1
                RETN
;找到当前显示方式
L_0976:
                MOV     ES:0FBH,AL              ;保存显示方式
                LODSB                           ;取最大颜色也即当前显示颜色
                MOV     ES:0D0H,AL              ;置当前显示颜色
                MOV     ES:0EDH,AL              ;置最大显示颜色
                MOV     ES:0F1H,AL              ;置前景颜色
                LODSW                           ;取当前显示方式屏幕最大列数
                MOV     ES:0D5H,AX              ;保存
                LODSW                           ;取当前显示方式屏幕最大行数
                MOV     ES:0D7H,AX              ;保存
                MOV     BYTE PTR ES:0F5H,1      ;置显示行间距=1
                MOV     BYTE PTR ES:[102H],4    ;置比例因子=4,即不放大缩小
                MOV     BYTE PTR ES:[0B80H],0   ;修改程序
                CLC                             ;CF=0
                RETN

                DB      11 DUP (0)              ;没有用

D_09B0          DB      4                       ;显示方式4
                DB      3                       ;3种颜色
                DW      27FH                    ;640列
                DW      C7H                     ;200行

                DB      6                       ;显示方式6
                DB      1                       ;1种颜色
                DW      27FH                    ;640
                DW      C7H                     ;200

                DB      10H                     ;EGA
                DB      0FH                     ;16种颜色
```

```
                DW      27FH                    ;640
                DW      15DH                    ;350

                DB      12H                     ;VGA
                DB      0FH                     ;16
                DW      27FH                    ;640
                DW      1DFH                    ;480

                DB      42H                     ;COLOR400
                DB      0FH                     ;16
                DW      27FH                    ;640
                DW      18F                     ;400

                DB      22H DUP (0)             ;没有用

;矩形填充
L_09F0:
                MOV     DI,9D0H                 ;参数保存区地址
L_09F3:
                CALL    L_0103                  ;取参数
                STOSW                           ;保存
                CMP     BYTE PTR [SI],2CH       ;还有吗?
                JNE     L_09FF                  ;没有了
                INC     SI
                JMP     SHORT L_09F3            ;取下一个
L_09FF:
                XOR     AX,AX                   ;AX=0
L_0A01:
                CMP     DI,OFFSET DS:[9E0H]     ;参数满了吗?
                JE      L_0A0A                  ;满了
                STOSW                           ;省略的参数置0
                JMP     SHORT L_0A01

L_0A0A:
                MOV     BX,OFFSET DS:[9D0H]
                MOV     AL,[BX]                 ;取颜色
                MOV     DS:[0EDH],AL            ;设置颜色
                INC     WORD PTR [BX+2]         ;列坐标加1
                DEC     WORD PTR [BX+4]         ;行坐标减1
                SUB     WORD PTR [BX+6],2       ;宽减2
                SUB     WORD PTR [BX+8],2       ;高减2
                MOV     AL,[BX+0AH]             ;取填充类型

                CMP     AL,1                    ;是1吗?
                JNE     L_0A60                  ;不是

;填充横线
                MOV     WORD PTR DS:[0D9H],1    ;列步长=1
                MOV     WORD PTR DS:[0DBH],0    ;行步长=0
                MOV     DI,[BX+4]               ;取行坐标
                SUB     DI,[BX+8]               ;减高
L_0A39:
                ADD     DI,[BX+0CH]             ;加间隔
                MOV     DX,1                    ;线宽计数
L_0A3F:
```

```
                MOV     BP,[BX+6]               ;BP=宽
                CMP     DI,[BX+4]               ;已填满了吗?
                JG      L_0A5D                  ;是
                MOV     DS:[0D3H],DI            ;置行坐标
                MOV     AX,[BX+2]               ;取列坐标
                MOV     DS:[0D1H],AX            ;置列坐标
                CALL    L_0417                  ;画线
                INC     DI                      ;下一线
                INC     DX                      ;线宽加1
                CMP     DX,[BX+0EH]             ;要间隔了吗?
                JG      L_0A39                  ;是
                JMP     SHORT L_0A3F
L_0A5D:
                JMP     L_01E7

L_0A60:
                CMP     AL,2                    ;是2吗?
                JNE     L_0A9B                  ;不是

;填充竖线
                MOV     WORD PTR DS:[0D9H],0    ;列步长=0
                MOV     WORD PTR DS:[0DBH],0FFFFH ;行步长=-1
                MOV     DI,[BX+2]               ;取列坐标
                MOV     CX,DI
                ADD     CX,[BX+6]               ;加宽=终点列坐标
L_0A78:
                ADD     DI,[BX+0CH]             ;加间隔
                MOV     DX,1                    ;线宽计数
L_0A7E:
                MOV     BP,[BX+8]               ;BP=高
                CMP     DI,CX                   ;已填满了吗?
                JG      L_0A5D                  ;是
                MOV     DS:[0D1H],DI            ;置列坐标
                MOV     AX,[BX+4]               ;取行坐标
                MOV     DS:[0D3H],AX            ;置行坐标
                CALL    L_0417                  ;画线
                INC     DI                      ;下一列
                INC     DX                      ;线宽加1
                CMP     DX,[BX+0EH]             ;要间隔了吗?
                JG      L_0A78                  ;是
                JMP     SHORT L_0A7E

L_0A9B:
                CMP     AL,3                    ;是3吗?
                JNE     L_0B0F                  ;不是

;填充右斜线
                PUSH    SI                      ;保存SI
                MOV     SI,[BX+2]               ;取列坐标
                MOV     DI,[BX+4]               ;取行坐标
                SUB     DI,[BX+8]               ;减高=左上角行坐标

;SI=列坐标, DI=行坐标, 即起始点坐标. 始点从左上角开始沿左竖线到底, 然而, 沿下
;面的横线向右到右下角. 终点从左上角开始沿上横线向右, 到右上角后再沿右竖线向下.
;因此, DI 初值为左上角的行坐标, 结束时应为右下角行坐标加矩形宽
```

```
                MOV     AX,DI                   ;AX=左上角行坐标
                ADD     AX,[BX+6]               ;加宽
                MOV     [BX+10H],AX
;[BX+10H]=左上角行坐标加宽，若 DI<该值，表示斜线的终点还在上横线上.

                MOV     AX,[BX+4]               ;取行坐标
                ADD     AX,[BX+6]               ;加宽
                MOV     [BX+12H],AX             ;保存
;[BX+12H]=左下角行坐标加宽，该值正好是 DI 填充结束时的值，下面用该值判断是否填
;充结束

                MOV     WORD PTR DS:[0D9H],1    ;列步长=1
                MOV     WORD PTR DS:[0DBH],0FFFFH ;行步长=-1
                                                ;向右上画线
L_0AC6:
                ADD     DI,[BX+0CH]             ;加间隔
                MOV     DX,1                    ;线宽计数
L_0ACC:
                CMP     DI,[BX+12H]             ;画完了吗?
                JG      L_0B0B                  ;是

                MOV     BP,[BX+6]               ;假设长度=宽度
                CMP     DI,[BX+10H]             ;终点还在上横线上吗?
                JGE     L_0AE1                  ;不是，已在右竖线上了

;终点上横线上，始点可能在左竖线上，也可能在下横线上
                MOV     BP,DI
                ADD     BP,[BX+8]               ;加高
                SUB     BP,[BX+4]               ;减行坐标

L_0AE1:
                MOV     DS:[0D1H],SI            ;置列坐标
                MOV     DS:[0D3H],DI            ;置行坐标
                CMP     DI,[BX+4]               ;始点在左竖线上吗?
                JLE     L_0AFF                  ;是，不用再调整了

;终点在下横线上
                MOV     AX,DI
                SUB     AX,[BX+4]               ;减左下角行坐标=在横线上的位移
                SUB     BP,AX
                ADD     DS:[0D1H],AX            ;修正列坐标
                MOV     AX,[BX+4]
                MOV     DS:[0D3H],AX            ;行坐标=左下角行坐标
L_0AFF:
                CALL    L_0417                  ;画线
                INC     DI                      ;下一条线
                INC     DX                      ;间隔加1
                CMP     DX,[BX+0EH]             ;要间隔了吗?
                JG      L_0AC6                  ;要
                JMP     SHORT L_0ACC
L_0B0B:         POP     SI                      ;恢复SI
                JMP     L_01E7

;填充左斜线
```

```
L_0B0F:
                PUSH    SI                      ;保存SI
                MOV     SI,[BX+2]               ;SI=列坐标
                ADD     SI,[BX+6]               ;加宽度
                MOV     DI,[BX+4]               ;DI=行坐标
                SUB     DI,[BX+8]               ;减高
;SI=右上角列坐标, DI=右上角行坐标

                MOV      AX,DI
                ADD      AX,[BX+6]
;DI=右上角行坐标加宽

                MOV      [BX+10H],AX
                MOV      AX,[BX+4]
                ADD      AX,[BX+6]
                MOV      [BX+12H],AX
;[BX+12H]=填充结束后的 DI 值

                MOV     WORD PTR DS:[0D9H],0FFFFH ;列步长=-1
                MOV     WORD PTR DS:[0DBH],0FFFFH ;行步长=-1
                                                  ;向左上画线
L_0B39:
                ADD     DI,[BX+0CH]             ;加间隔
                MOV     DX,1                    ;间隔计数
L_0B3F:         MOV     BP,[BX+6]               ;长度=宽
                CMP     DI,[BX+12H]             ;画完了吗?
                JG      L_0B0B                  ;是

                CMP     DI,[BX+10H]             ;终点在上横线上吗?
                JGE     L_0B54                  ;不是
;终点在上横线上
                MOV     BP,DI
                ADD     BP,[BX+8]
                SUB     BP,[BX+4]               ;调整长度
L_0B54:
                MOV     DS:[0D1H],SI
                MOV     DS:[0D3H],DI            ;置起始坐标

                CMP     DI,[BX+4]               ;起点在下横线上吗?
                JLE     L_0B72                  ;不是,已不用调整了

;两边都要调整
                MOV     AX,DI
                SUB     AX,[BX+4]
                SUB     BP,AX                   ;调整长度
                SUB     DS:[0D1H],AX
                MOV     AX,[BX+4]
                MOV     DS:[0D3H],AX            ;调整行坐标
L_0B72:
                CALL    L_0417                  ;画线
                INC     DI                      ;下一条线
                INC     DX                      ;间隔加1
                CMP     DX,[BX+0EH]             ;要间隔了吗?
                JG      L_0B39                  ;是
                JMP     SHORT L_0B3F
```

```
                DB      0, 0                    ;没有用

L_0B80:
;初始化
                MOV     AX,3510H
                INT     21H                     ;取INT 10H中断向量
                CMP     BX,18DH                 ;已装载过了吗?
                JE      L_0B8F                  ;是

                MOV     BP,ES                   ;保存原INT 10H的段地址->BP
                PUSH    DS
                POP     ES                      ;ES=DS
L_0B8F:
                CALL    L_0952                  ;初始化数据变量
                JNC     L_0B96                  ;操作成功
L_0B94:
                INT     20H                     ;不驻留退出

L_0B96:
                CMP     BX,18DH
                JE      L_0B94                  ;已装载过了,不驻留退出

                MOV     WORD PTR DS:[13BH],BX   ;修改指令
                MOV     WORD PTR DS:[13DH],BP   ;修改指令

                MOV     DX,OFFSET L_018D        ;取INT 10H偏移地址
                MOV     AX,2510H
                INT     21H                     ;修改INT 10H中断向量

                MOV     DX,0C50H                ;驻留长度
                INT     27H                     ;驻留退出
CODE            ENDS
                END     L_0100
```

第五节　光标闪烁控制　INT1C.COM

一、使用说明

1．进入光标闪烁状态

INT1C←┘

2．退出光标闪烁状态

INT1C　←┘

在光标闪烁时，不能用CTRL_F5退出系统，否则在运行其它程序时，有可能死机。这是因为，光标闪烁是利用时间中断INT　1CH来实现的，该中断和INT8H一样每秒要执行18．2次。光标闪烁时，INT　1CH程序被传送到显示模块中，当用CTRL_F5退出汉字系统后，显示模块占用的内存被释放了，但INT1CH中断向量没有恢复，所以当再次运行其它程序时，有可以覆盖现有INT　1CH代码造成死机现象。

在单色显示程序中（CH21．COM和CH25．COM），没有空闲的区域，所以不用用本程序使光标闪烁。若运行则会出现死机现象。

二、程序清单

```
;INT1C.ASM
```

```
CODE        SEGMENT
            ASSUME CS:CODE,DS:CODE
            ORG    100H

L_0100:
            MOV    AX,351FH
            INT    21H                  ;取INT 1FH中断向量
            CMP    BYTE PTR DS:80H,0    ;命令行有参数吗?
            JNE    L_0136               ;没有,转退出光标闪烁
;设置光标闪烁
            MOV    SI,OFFSET D_0142     ;光标闪烁程序地址
            MOV    DI,1920H             ;显示管理程序中空闲区地址，该
                                        ;空间也可以说是在键盘管理模块
                                        ;中，因为显示模块也包含在键盘
                                        ;管理模块中
            MOV    CX,46H               ;程序长度
            REP    MOVSB                ;传送至CCCC.COM模块

            PUSH   ES
            POP    DS                   ;DS=ES=INT 1FH段地址
            MOV    AX,351CH
            INT    21H                  ;取原INT 1CH中断向量
            CMP    BX,1923H             ;已经设置过了吗?
            JE     L_0134               ;是,不用再设置了
            MOV    DS:1968H,BX          ;保存原INT 1CH偏移值
            MOV    WORD PTR DS:1968H+2,ES;保存原INT 1CH段值

            MOV    DX,OFFSET L_1923
            MOV    AX,251CH
            INT    21H                  ;设置INT 1CH

L_0134:
            INT    20H                  ;程序结束

L_0136:
;取消光标闪烁
            LDS    DX,DWORD PTR ES:1968H ;取原INT 1CH中断向量
            MOV    AX,251CH
            INT    21H                  ;恢复原INT 1CH中断向量

            INT    20H                  ;程序结束

D_0142      DW     0                    ;保存当前光标位置
D_0143      DB     0                    ;时钟中断计数

;INT 1CH:
L_0145:
            PUSH   DS
            PUSH   ES
            PUSH   AX
            PUSH   SI
            PUSH   BP                   ;保存寄存器

            PUSH   CS
            POP    DS                   ;DS=CS
```

```
L_015E:
                INC     BYTE PTR DS:1922H       ;增加计数(1922H 即 d_0143)
                CMP     BYTE PTR DS:1922H,4     ;计数4次了吗?
                JNE     L_0171                  ;继续等待

                MOV     BP,1                    ;BP=1设置光标,见CV26.ASM
                CALL    2CFH                    ;即1AAD, 调用光标操作子程序
                JMP     SHORT L_0182

L_0171:
                CMP     BYTE PTR DS:1922H,8     ;计数到8了吗?
                JNE     L_0182                  ;还没有
                XOR     BP,BP                   ;清除光标
                CALL    02CFH                   ;调用光标操作子程序

L_017D:
                MOV     BYTE PTR DS:1922H,0     ;将计数器清0

L_0182:
                POP     BP
                POP     SI
                POP     AX
                POP     ES
                POP     DS                              ;恢复寄存器

                IRET                                    ;中断返回
CODE            ENDS
                END     L_0100
```

第六节　文件打印　LP.COM

一、使用说明

本打印程序主要有两种打印方法。

1．直接打印字符串。

LP　字符串←┘

例如：LP　'@K'中华人民共和国←┘

设置打印字型为K型字，然后打印字符串“中华人民共和国”

2．直接打印文本文件

LP　a　文件名←┘

其中a表示打印份数，文件名表示要打印的文件的名字。打印时屏幕显示“正在打印第xxx份...”，打印完每一份后提示“按回车键继续打印下一份!”，打印过程中，可按CTRL_BREAK中止打印。

例如：LP　2　2.13H←┘

表示打印2.13H文件，两份。

二、程序清单

```
;LP.ASM
CODE            SEGMENT
                ASSUME CS:CODE, DS:CODE
                ORG     100H
L_0100:
                MOV     SI,82H                  ;命令行参数首址
                MOV     BL,[SI-2]               ;命令行参数长度
```

```
                MOV     BH,0
                LEA     DI,[BX+SI]              ;命令行参数尾址
                MOV     BYTE PTR [DI-1],0       ;置结束标志(用于文件操作)
                CMP     SI,DI                   ;命令行有字符吗?
                JAE     L_0126                  ;没有
                LODSB                           ;取字符
                CMP     AL,30H
                JB      L_011B                  ;不是数字
                CMP     AL,3AH
                JB      L_0134                  ;是数字,表示打印文件
L_011B:
;打印字符串
                MOV     DL,AL
                MOV     AH,5
                INT     21H                     ;打印一个字符
                LODSB                           ;取下一个字符
                CMP     SI,DI
                JB      L_011B                  ;打下一个
L_0126:
                MOV     DL,0DH                  ;回车
                MOV     AH,5
                INT     21H
                MOV     DL,0AH                  ;换行
                MOV     AH,5
                INT     21H
                INT     20H                     ;字符串打印结束

L_0134:
;打印文件
                DEC     SI
                MOV     CX,0AH
L_0138:
;计算打印份数
                LODSB
                CMP     SI,DI
                JE      L_0199                  ;没有文件名
                CMP     AL,20H
                JE      L_014D                  ;找到分隔符了
                AND     AL,0FH
                XCHG    AL,CH
                MUL     CL                      ; x 10
                ADD     AL,CH                   ; + 数字
                MOV     CH,AL                   ;CH=份数
                JMP     SHORT L_0138

L_014D:
                MOV     CL,CH
                MOV     CH,0                    ;CX = 打印份数
                MOV     BP,CX                   ;保存于BP之中
L_0153:
                LODSB
                CMP     AL,20H
                JE      L_0153                  ;过虑空格

                DEC     SI
```

```
                MOV     DX,SI                   ;文件名首地址
                MOV     AX,3D00H
                INT     21H                     ;打开文件
                JNC     L_016B                  ;操作成功

                MOV     DX,OFFSET D_01DC        ;错误信息地址->DX
L_0165:
                MOV     AH,9
                INT     21H                     ;显示错误信息
                INT     20H                     ;错误结束

L_016B:
                MOV     BX,AX                   ;BX=AX=文件句柄
                MOV     DX,240H                 ;缓冲区地址
                MOV     CX,0FFFFH               ;读入最多64K(实际上此时由于
                                                ;DX<>0,因此不可能读入64K)
                MOV     AH,3FH
                INT     21H                     ;读入文件内容
                JNC     L_017E

                MOV     DX,1F0H                 ;错误信息地址->DX
                JMP     SHORT L_0165

L_017E:
                MOV     BX,AX                   ;BX=文件长度
                MOV     CX,BP                   ;CX=打印份数
                MOV     DI,210H                 ;修改份数显示信息的基地址
L_0185:
                MOV     DX,OFFSET D_0208        ;显示正在打印信息
                MOV     AH,9
                INT     21H

                PUSH    CX                      ;保存份数
                MOV     CX,BX                   ;CX=文件长度
                MOV     SI,240H                 ;文件内容起始地址
L_0192:
                LODSB                           ;取一个字符
                CMP     AL,1AH                  ;是文件结束字符吗?
                JE      L_019F                  ;一份打印完毕
                MOV     DL,AL
L_0199:
                MOV     AH,5
                INT     21H                     ;打印一个字符
                LOOP    L_0192                  ;最后打印机CX个(文件长度)

L_019F:
                POP     CX                      ;恢复份数
                CMP     CX,1
                JE      L_01DA                  ;打印完毕

                MOV     DX,OFFSET D_0220
                MOV     AH,9
                INT     21H                     ;显示按任意键继续打印
L_01AC:
                MOV     AX,0C01H
```

```
                INT     21H                     ;等用户按键
                CMP     AL,0DH
                JNE     L_01AC                  ;不是回车键,则继续等待
                MOV     DL,0AH
                MOV     AH,2
                INT     21H                     ;换行符
;修改份数字符串(是纯中文方式数字串)
                INC     BYTE PTR [DI+7]         ;增加个位数
                CMP     BYTE PTR [DI+7],0BAH    ;已到10
                JB      L_01D8
                MOV     BYTE PTR [DI+7],0B0H    ;个位=0
                INC     BYTE PTR [DI+5]         ;增加十位数
                CMP     BYTE PTR [DI+5],0BAH    ;已到10
                JB      L_01D8
                MOV     BYTE PTR [DI+5],0B0H    ;十位=0
                INC     BYTE PTR [DI+3]         ;增加百位
L_01D8:
                LOOP    L_0185                  ;继续打印下一份

L_01DA:
                INT     20H                     ;程序结束

D_01DC          DB      '指定文件未找到!', 0DH, 0AH, 7, 7,'$'
                DB      '指定文件读出错误!',0DH,0AH,7,7,7,$
                DB      0,0,0
D_0208          DB      '正在打印第００１份. . .$'
D_0220          DB      '按回车键继续打印下一份!', 7, 7, '$'
CODE            ENDS
                END     L_0100
```