DOS 内核的奥秘

张昆藏 著

北京科海培训中心

DOS内核的奥秘

张昆藏 著

北京科海培训中心

发　　行：北京科海培训中心资料组
地　　址：北京海淀路82号科海培训中心
邮　　码：100080
联系电话：2562449　2562954
乘　　车：320、332、302路车至海淀黄庄站下车文化馆北平房
印　　刷：河北省蔚县印刷厂

前　言

MS-DOS是Microsoft公司八十年代产品。IBM公司购买使用权后，将其改名为PC-DOS。随着IBM PC系列微型计算机的普及，已使MS-DOS成为当今的主流操作系统之一。

MS-DOS自1981年诞生以来，随着PC、PC/XT、PC/AT的发展，七年里推出了由1.00到3.30的八个版本，表现出MS-DOS的活跃生命力。但直到3.30版，MS-DOS本质上仍是单用户单任务操作系统。1986年12月推出的4.0版增加了前后台管理和多任务（进程）管理，才成为多任务操作系统。

为配合MS-DOS的使用，已有不少用户手册、技术手册之类的书籍出版。本人也曾在1988年编译了《DOS（3.10～3.30）磁盘操作系统》一书。

但在实际应用中，常遇到手册未公布的DOS保留信息和保留的系统功能，给用户带来不便。近年来，内存驻留程序成为软件界广为推崇的一种编程模式。一些优秀的TSR软件，如Superkey、Pctools、Turbolightning等相继问世。编制TSR程序，就要解决它与DOS的共容性和与前台暂驻程序共容性等问题。此外，数据、文件、磁盘的加密与解密技术，计算机病毒的防治等问题，都需要对DOS内核的结构和内部运行机制有深入的了解。

为此，作者以PC-DOS 2.10为兰本，详尽剖析了有200多个子程序的、长度为16K字节的IBMDOS.COM模块，也附带剖解了IBMBIOS.COM模块和COMMAND.COM模块，在此基础上编写了本书。其目的在于，揭示DOS内核的结构，各种系统功能使用的数据结构和实现的机理，并全面公布DOS保留的系统功能和其它保留信息。

本书按操作系统的基本概念和主要功能来组织内容，共有七章：内核的结构与核心态进程；磁盘设备管理；树型目录管理；文件的控制与读写管理；输入输出管理；内存的分配与程序的加载；程序的退出。此外，还有三个附录。本书例举了内核模块的近百个子程序，都给予了详细的注释，并清楚地交待了子程序的入、出口参数。（顺便申明：本书所给出的子程序代码段地址是仅供参考用，随机型、配置状况不同它将会有所变动。）

读者可从本书得到下述问题的解答：

1. DOS内核的结构和三种进入方式；
2. 系统核心态（临界态）进程的标志；
3. 系统三个内部堆栈；
4. 未公布的部件参数块（UPB）结构；
5. 未公布的扇段缓冲区（SBF）及其调度方式；
6. 介质检查的意义及实现过程；
7. 盘簇空间的分配与释放策略；
8. 依路经检索或新建目录项的实现过程；
9. 未公布的从根目录至当前目录的路径保存区；
10. 未公布的系统FCB（FCB*）的结构；
11. 未公布的句柄式文件打开表（OFT）；

12. 句柄式与FCB式文件操作的功能差异；
13. 句柄式与FCB式文件操作的内在联系；
14. 输入输出的ASCII方式与BIN方式；
15. 未公布的设备FCB结构；
16. 内核对标准输入输出重定向的支持；
17. 内核中的标准输入输出设备被替换；
18. Ctrl-C、Ctrl-P、Ctrl-S的检查及处理过程；
19. 系统定时及其相关中断；
20. 后台打印进程的激活；
21. 输入输出控制（IOCTL）的意义及实现过程；
22. 内存空间的分配与释放策略；
23. 未公布的系统第一个环境块的结构；
24. 加载程序时环境块的继承；
25. 未公布的程序段前缀（PSP）中的保留信息；
26. .EXE文件的头部信息及重定位过程；
27. 未公布的DOS INT 22H、INT23H和INT 24H处理程序；
28. 内核的严重错误处理程序；
29. 未公布的DOS系统功能；
30. 未公布的INT 0、INT 28H、INT 29H。

应当强调指出，调用DOS保留的系统功能时要特别小心。因为有些系统功能之所以不予公布，确是只宜系统内部使用而不宜应用程序中调用，请见附录Ⅱ的说明。此外，DOS未公布的数据结构和某些保留信息可能随版本的升迁而变动。例如，每个磁盘当前目录的自身绝对路径名是存放在每个磁盘的UPB中，这是DOS2.1版的情况；到了3.10版，为了支持网络上文件、目录的共享，此路径名又移至它处保存了。

IBMDOS.COM模块除一小段初始化程序代码外，绝大部分为内核模块。本书仅以内核模块为分析对象。但在内核模块与IBMBIO.COM模块和COMMAND.COM模块有着紧密的联系，故根据问题论述的需要，也介绍了这两个模块的部分内容。

MS-DOS版本2在CP/M操作系统基础上，引入了许多UNIX操作系统的特色，而成为一个全新的操作系统。虽然以后的各版本在支持磁盘升级和支持网络等方面做了不少改进，但MS-DOS的架框结构已在版本2定型了。故本书以PC-DOS 2.10版为兰本。另外，从版本2到版本3，DOS的保留信息变动较大。例如，3.10版为支持网络功能，仅INT 2FH就有38个子功能是未公布的。准确地描述它们需要不少篇幅，所以本书这次未兼顾3.X版。

由于本人水平有限，书中论点虽经仔细推敲和反复论证，但不妥之处甚至有误之处仍难免，还望读者批评指正。

张昆藏　1990年10月

目　录

第一章　内核结构与核心态进程

§1.1　内核的位置与结构

1.1.1　DOS组成的概况

MS-DOS（PC-DOS）由四部分组成：

①自举记录块（Boot Record）。系统启动时，由ROM-BIOS的引导程序（INT 19H）把系统盘上的自举记录块读入0:7C00H为开始的内存区。控制权移交给自举记录块，由它把DOS的主体装入内存。若系统盘为软盘，则软盘第一扇（0面0道1扇）为有自举记录块的DOS引导扇。若系统盘为硬盘，则硬盘第一扇为主引导扇，内有分区表和主自举程序，而DOS分区的第一扇才为DOS引导扇，因而有两次引导过程。

②IBMBIO.COM文件。该文件有两大模块；一是DOSBIOS模块，它在ROM-BIOS的基础上提供了一组设备驱动程序，从而构成了DOS的基本输入输出系统，而使DOS的内核与机器硬件相隔离；二是SYSINT模块，在自举记录（或称DOS引导程序）将IBMBIO.COM文件装入内存后，经BIO的初始化代码后由SYSINT模块承接了系统控制权，SYSINT模块结合DOS.COM的初始化程序完成BIO、DOS、COMMAND三个主模块的内存定位和初始化工作，最后将控制权交给COMMAND，SYSINT消失。

③IBMDOS.COM文件。它除了一个初始化程序外（初始化后被覆盖），主要是DOS的Kernel模块。Kernel模块是DOS的核心，它负责磁盘与其它系统资源的管理；尤其是它以软件中断INT 21H提供了88种系统功能调用，可使用户程序或DOS的系统软件方便地享用系统资源。此模块还给出INT 25H、INT 26H（绝对磁盘读、写），INT 20H（结束退出）、INT27H（驻留退出）INT0（除法溢出）等中断处理程序。

④COMMAND.COM文件。它是DOS顶层的缺省的命令解释程序，用于支持用户的键盘命令。它除了主要用于处理AUTOEXEC.BAT文件的初始化程序外，其余为COMMAND.COM模块。该模块分为常驻内存部分和暂驻内存部分。暂驻内存部分位于内存高地址端，按其功能可分为暂驻部分1和暂驻部分2。暂驻部分1给出DOS提示，接收命令，并负责执行各种内部命令。暂驻部分2是exec子程序，它负责加载并执行外部命令程序和.EXE或.COM文件形式的用户应用程序。常驻部分中，有4BH号（EXEC）系统功能的主流程，INT 22H结束出口，INT 23H、INT 24H中断处理程序和保留的INT 2EH中断处理程序；此部分还负责检查暂驻部分的完整性，若被破坏则要从系统盘重新读入暂驻部分。系统启动之后，COMMAND.COM的初始化程序即可被应用程序覆盖。

DOS（2.1版）占用内存不足40KB，其常驻区仅24KB左右。

DOS加载毕的内存映象如图1-1所示。

1.1.2　DOS-Kernel模块结构

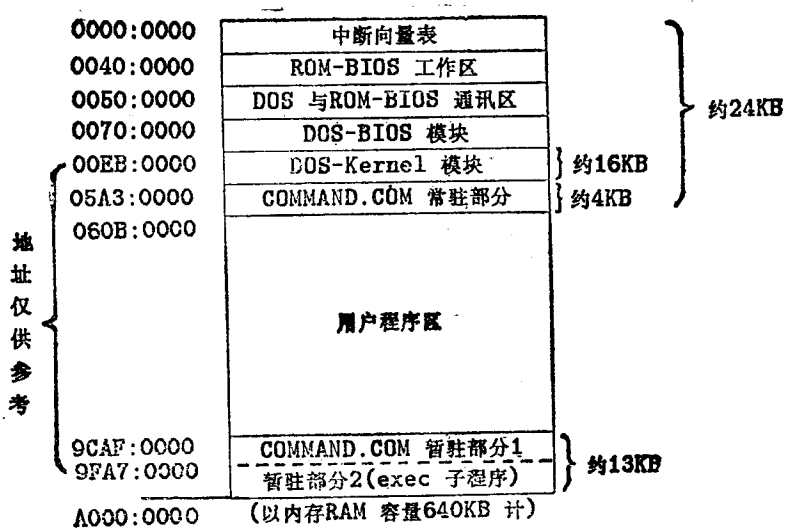

图 1-1 PC-DOS2.1的内存映象

PC-DOS 2.1版的Kernel模块提供了软件中断INT 21H的除4BH号外的87种系统调用的功能子程序。此外，还给出了INT0、INT 20H、INT 25H、INT 26H、INT 27H的中断处理程序。其结构如图1-2所示。

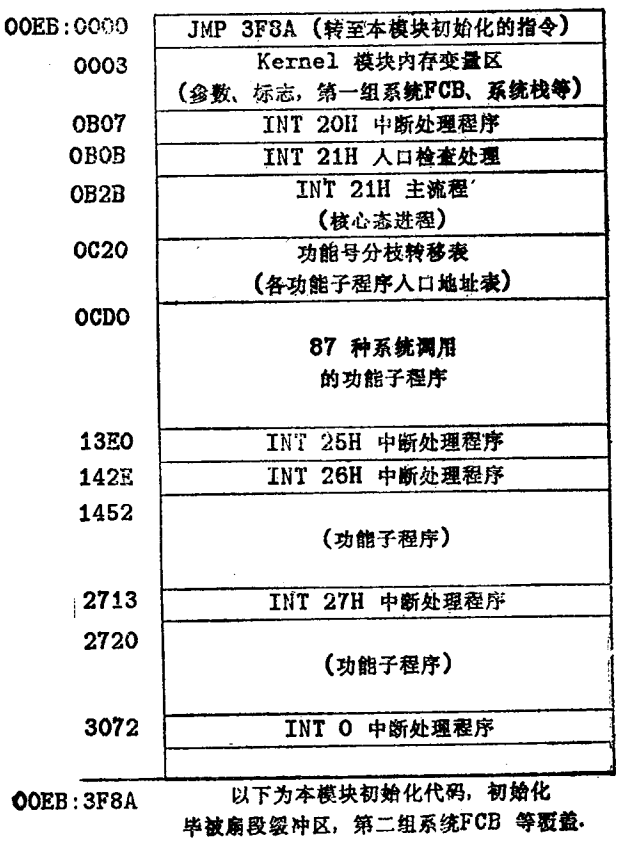

图 1-2 PC-DOS 2.1版内核模块结构图

PC-DOS 2.10未把4BH功能子模块列入Kernel模块中，而是位于COMMAND.COM中。4BH功能子模块主流程位于COMMAND.COM的常驻部分，其EXEC功能子程序位于COMMAND.COM的暂驻部分2。

系统启动后，中断向量表中INT 21H中断入口地址置为05A3:0180，即指向COMMAND.COM常驻部分的代码开始位置。程序判测AH功能号是否为4BH，若是，则往下执行4BH功能子模块；若否，则远转移到00EB:0B0B而进入Kernel模块，执行INT 21H入口检查，若功能号合法，继而进入INT 21H主流程。

1.1.3 INT21H的三种调用方式

用户程序可用表1-1所示的三种方式调用INT21H。

表 1-1 INT 21H的三种调用方式

类 型	调 用 格 式	说 明
中断指令方式 （最常用）	MOV AH，功能号（0～57H） 设置其它入口参数 INT 21H	
远过程调用方式 （很少用）	MOV AH，功能号（0～57H） 设置其它入口参数 CALL FAR PSP+50H	程序段前缀PSP的0050H～0052H 偏移处为一过程体： INT 21H RETF
近过程调用方式 （为与DOS1.0版兼容而保留的老式方式）	MOV CL，功能号（0～24H） 设置其它入口参数 CALL NEAR PSP+05H （此时应保证代码段寄存器CS为当前PSP的段）	程序段前缀PSP的0005H～0009H 偏移处为一条指令： CALL FAR 0000:00C0H， 而中断向量表0000:00C0处为 JMP 00EB:0B14 而进入INT 21H老式入口。

Kernel模块中INT 21H入口检查及处理的程序段如下：

```
00EB:0B0B  80FC57      CMP   AH, 57          ; 功能号小于等于57H,正确
00EB:0E0E  80FC57      NOP
00EB:0B0F  791A        JBE   0B2B
00EB:0B11  B000        MOV   AL, 00          ; 否则以AL=0中断返回
00EB:0B13  CF          IRET
00EB:0B14  58          POP   AX              ; INT 21H老式入口
00EB:0B15  58          POP   AX
00EB:0B16  2E          CS:                   ; 为以IRET指令返回，修改用户栈

00EB:0B17  8F06D102    POP   [02D1]
```

```
00EB:0B1B  9C          PUSHF                    ; 02D1字单元为暂存单元
00EB:0B1C  FA          CLI
00EB:0B1D  50          PUSH   AX
00EB:0B1E  2E          CS:
00EB:0B1F  FF36D102    PUSH   [ 02D1 ]
00EB:0B23  80F924      CMP    CL, 24            ; 老式调用，功能号要小于等
                                                 于24H
00EB:0B26  90          NOP
00EB:0B27  77EB        JA     0B11              ; 不对，以AL=0中断返回
00EB:0B29  8AE1        MOV    AH, CL            ; 正确，将功能号置入AH
; 以下为INT 21H的主流程
```

§1.2 核心态进程

1.2.1 核心态进程标志

当应用软件或系统上层软件以INT 21H发出（除4BH号外）系统功能调用时，即进入INT 21H的主流程。INT 21H中断处理进程是系统的核心态进程，应保证其执行的完整性并尽量避免重入。为此，系统以00EB:012D单元为核心态标志单元。此单元置为1，表示核心态（或称系统态）进程正在进行中；退出核心态进程时，此单元被清为0。此单元地址可用34H号系统功调返回的ES:BX值得到。

1.2.2 INT21H中断处理主流程

INT 21H中断处理主流程大致分为三大部分：（1）在用户程序堆栈中保存现场AX～ES寄存器，然后保存用户堆栈指针；取三个系统内部栈之一为核心态进程使用的当前栈；建核心态进程标志，并完成一些初始设置和测试。（2）根据分枝转移表，调用与入口AH功能号相应的某一功能子程序，实现此系统功能。（3）撤除核心态进程标志；恢复用户堆栈指针，恢复AX～ES寄存器；中断返回。

此主流程代码及分枝转移表列于后：

```
00EB:0B2B  E8DC00      CALL   0C0A              ; 在用户栈中保存AX～ES寄
                                                 存器
00EB:0B2E  2E          CS:
00EB:0B2F  8C1E3403    MOV    [0334], DS        ; 为下面使用DS,BX寄存器
00EB:0B33  2E          CS:
00EB:0B34  891E3203    MOV    [0332], BX        ; 先保存
00EB:0B38  8CCB        MOV    BX, CS
00EB:0B3A  8EDB        MOV    DS, BX            ; CS→DS
00EB:0B3C  FF062D01    INC    BYTE PTR [012D]   ; 建核心态进程标志
00EB:0B40  A1D102      MOV    AX, [02D1]
00EB:0B43  A3DE0A      MOV    [0ADE], AX        ; 保存02D1双字单元
00EB:0B46  A1D303      MOV    AX, [02D3]
00EB40B49  A3DC0A      MOV    [0ADC], AX
```

```
00EB:0B4C  58          POP   AX              ; 使AX寄存器为入口值
00EB:0B4D  50          PUSH  AX
00EB:0B4E  8926D102    MOV   [02D1], SP      ; 将用户栈当前（双字）指针
                                               保存于02D1双字单元

00EB:0B52  8C16D302    MOV   [02D3], SS
00EB:0B56  8E1E9101    MOV   DS, [0191]      ; 0191字单元中当前PSP的段
                                               址

00EB:0B5A  89262E00    MOV   [002E], SP      ; 将用户栈当前（双字）指针
00EB:0B5E  8C163000    MOV   [0030], SS        保存于PSP+2E～31处

00EB:0B62  8CCB        MOV   BX, CS          ; 置SS:SP=00EB:099C第一
                                               系统栈

00EB90B64  8ED3        MOV   SS, BX
00EB:0B66  BC9C09      MOV   SP, 099C
00EB:0B69  FB          STI                   ; 开中
00EB:0B6A  0E          PUSH  CS
00EB:0B6B  1F          POP   DS
00EB:0B6C  32FF        XOR   BII, BH
00EB:0B6E  883E3001    MOV   [0130], BH      ; 清“CON被替换”标志
00EB:0B72  C606310101  MOV   BYTE PTR[0131], 01  ; 建“INT 28H 允许”标志
00EB:0B77  C606D50000  MOV   BYTE PTR[00D5], 00  ; 建“子目录项向下检索标”
                                                   志

00EB:0B7C  8ADC        MOV   BL, AH
00EB:0B7E  D1E3        SHL   BX, 1           ; 根据AH中功能号得到入口
                                               地址在转移表中的偏移→
                                               BX

00EB:0B80  FC          CLD
00EB:0B81  0AE4        OR    AH, AH
00EB:0B83  741B        JZ    0BA0
00EB:0B85  80FC0C      CMP   AH, 0C          ; 1～0CH、50H、51H号作
                                               为第一类系统功能调用
00EB:0B88  760A        JBE   0B94            ; 其它74种功能号作为第二类
                                               系统功能调用

00EB:0B8A  80FC51      CMP   AH, 51
00EB:0B8D  7405        JZ    0B94
00EB:0B8F  80FC50      CMP   AH, 50
00EB:0B92  750C        JNZ   0BA0
00EB:0B94  803E2E0100  CMP   BYTE PTR [012E], 00  ; 第一类系统功能调用处理：
00EB:0B99  751E        JNZ   0BB9            ; 若“INT 24H执行”标志
                                               为0
00EB:0B9B  BCDC0A      MOV   SP, 0ADC        ; 则改置SS:SP=00EB:0A
00EB:0B9E  EB19        JMP   0BB9              DC为第三系统栈

00EB:0BA0  C6062E0100  MOV   BYTE PTR [012E], 00  ; 第二类系统功能调用处理：
00EB:0BA5  C6062F01FF  MOV   BYTE PTR [012F], FF  ; 清“INT 24H执行”标
00EB:0BAA  BC3CDA      MOV   SP, 0A3C            志，建“写禁止部件单元”
```

```
                                                    为-1
00EB:0BAD  F6063201FF    TEST    BYTE PTR [0132], FF    ; 改置SS:SP=00EB:0A3C
                                                        为第二系统栈
00EB:0BB2  7405          JZ      0BB9                   ; "Ctrl-Break检查开关"为
00EB:0BB4  50            PUSH    AX                      0否

00EB:0BB5  E88323        CALL    2F3B                   ; 若不为0，则打开，调2F38
                                                        子程序进 行Ctrl-Break检查
00EB:0BB8  58            POP     AX
00EB:0BB9  FF369401      PUSH    [0194]                 ; 两大类系统功能调 用预处理
                                                        毕，又归至此
00EB:0BBD  2E            CS:                            ; 0194字单元 中为0BCB，即
00EB:0BBE  FFB7200C      PUSH    [BX+0C20]               功能子程序返回主流程地址

00EB:0BC2  8B1E3203      MOV     BX, [0332]             ; 恢复入口时的DS、BX寄存
                                                        器
00EB:0BC6  8E1E3403      MOV     DS, [0334]
00EB:0BCA  C3            RET                            ; 进入相应功能子程 序入口
00EB:0BCB  FA            CLI                            ; 主流程公共 出口处理
00EB:0BCC  2E            CS:                            ; 关中
00EB:0BCD  FE0E2D01      DEC     BYTE PTR [012D]        ; 清"核心态进程"标志
00EB:0BD1  2E            CS:
00EB:0BD2  8B26D102      MOV     SP, [02D1]             ; 由02D1双字单元，恢 复用
                                                        户栈指针→SS:SP
00EB:0BD6  2E            CS:
00EB:0BD7  8E16D302      MOV     SS, [02D3]
00EB:0BDB  8BEC          MOV     BP, SP

00EB:0BDD  884600        MOV     [BP+00], AL            ; 将功能子程 序返 回的AL值
00EB:0BE0  2E            CS:                            置入用户栈中，以便带出
00EB:0BE1  A1DE0A        MOV     AX, [0ADE]             ; 恢复02D1双字单元
00EB:0BE4  2E            CS:
00EB:0BE5  A3D102        MOV     [02D1], AX
00EB:0BE8  2E            CS:
00EB:0BE9  A1DC02        MOV     AX, [0ADC]
00EB:0BEC  2E            CS
00EB:0BED  A3D30A        MOV     [02D3], AX
00EB:0BF0  E80300        CALL    0BF6                   ; 从用户 栈 中 恢复AX～ES
                                                        寄存器
00EB:0BF3  CF            IRET                           ; 中断返回，INT 21H 主 流
                                                        程结束
00EB:0BF4  F30B          DW      0BF3                   ; 暂存单元
00EB:0BF6  2E8F06F40B    POP     CS:[0BF4]              ; 0BF6子程序，恢复现场
00EB:0BFB  58            POP     AX
00EB:0BFC  5B            POP     BX
```

```
00EB:0BFD  59              POP     CX
00EB:0BFE  5A              POP     DX
00EB:0BFF  5E              POP     SI
00EB:0C00  5F              POP     DI
00EB:0C01  5D              POP     BP
00EB:0C02  1F              POP     DS
00EB:0C03  07              POP     ES
00EB:0C04  2E              CS:
00EB:0C05  FF36F40B        PUSH    [0BF4]
00EB:0C09  C3              RET
00EB:0C0A  2E              CS:                          ; 0C0A子程序，保护现场
00EB:0C0B  0F03F40B        POP     [03F4]
00EB:0C0F  06              PUSH    ES
00EB:0C10  1E              PUSH    DS
00EB:0C11  55              PUSH    BP
00EB:0C12  57              PUSH    DI
00EB:0C13  56              PUSH    SI
00EB:0C14  52              PUSH    DX
00EB:0C15  51              PUSH    CX
00EB:0C16  53              FUSH    BX
00EB:0C17  50              PUSH    AX
00EB:0C18  EBEA            JMP     0C04
00EB:0C1A  2E              CS:                          ; 0C1A子程序，取用户栈指
                                                        针至DS:SI
00EB:0C1B  C536D102        LDS     SI, [02D1]
00EB:0C1F  C3              RET
00EB:0C20  BA 0D 1B 34 24  34 B7 34-D4 34 DA 34 49 0F 75 0F   ; 分枝转移表
00EB:0C30  EE 34 FD 34 0D  35 CF 36-D9 36 08 0F 3B 0F A9 2D   0号功能分程序入口=
00EB:0C40  0F 2E E4 0D 45  0E 9F 2C-2F 2C 37 2C 81 2E 1D 2D   0DBA
                                                              ⋮
00EB:0C50  D0 0C 36 0F E0  0E D3 0C-D5 0C D0 0C D0 0C EB 0E   ⋮
00EB:0C60  D0 0C 47 2C 4F  2C 7B 0E-B9 0E 0B 10 27 10 57 2C   ⋮
00EB:0C70  5F 2C F0 0F 9A  2B B7 2B-D7 2B E8 2B 3D 0D CC 0E   ⋮
00BE:0C80  05 0D EA 26 ED  0E 44 0D-63 0D FB 0F 7B 0D 3F 13
00BE:0C90  1E 0D 09 11 96  12 B9 11-A7 3A 50 39 2D 3B 42 3B   57H功能子程序入口=
                                                              3DA9
00EB:0CA0  6F 3B 71 3A D5  3B 23 3C-72 3C 98 3A BF 3B 5E 3D
00EB:0CB0  C0 14 3C 15 51  15 E5 26-20 27 D7 26 94 3E 44 3F
00EB:0CC0  D9 10 DC 10 6F  0D 71 13-38 0D 21 10 E2 3D A9 3D
```

1.2.3 系统三个内部栈及重入性讨论

由上述INT 21H中断主流程可以看出：INT 21H在把AX～ES九个寄存器的内容保存到用户堆栈中，并将用户堆栈当前双字指针（SS:SP）保存在CS:02D1双字单元后，就不再使用用户堆栈。而以新建立的系统内部堆栈来实现各功能子程序的调用，调用完毕后再舍弃系统内部栈而恢复用户栈的使用。

系统三个内部栈使用情况如下：

（1）若功能调用号为01～0CH、50H、51H之一，而且前无严重错误发生（即INT 24H执行标志（00EB:012E）=0），则使用第三个系统内部栈，即置SS:SP=00EB:0ADC。

（2）若功能调用号为01～0CH、50H、51H之一，但前已有严重错误发生（即INT 24H执行标志（00EB:012E）=1），则使用第一个系统内部栈，即置SS:SP=00EB:099C。

（3）若功能调用号不为01～0CH、50H、51H的话，则使用第二个系统内部栈，即置SS:SP=00EB:0A3C。

上述第（2）种情况是，在系统功能调用发生严重设备错误而进入INT 24H中断处理时，此时需显示设备错误信息而发生字符设备I/O功能调用，从而再次进入INT 21H。为避免破坏上次功能调用所使用的系统栈状况，显然应另设一系统栈。除此特殊情况外，以01～0CH、50H、51H功能号为一大类（不涉及磁盘操作），其它74种功能号为另一大类，两类功能调用各使用不同的系统栈。

用户栈和系统栈状况分别于图1-3的（a）和（b）。

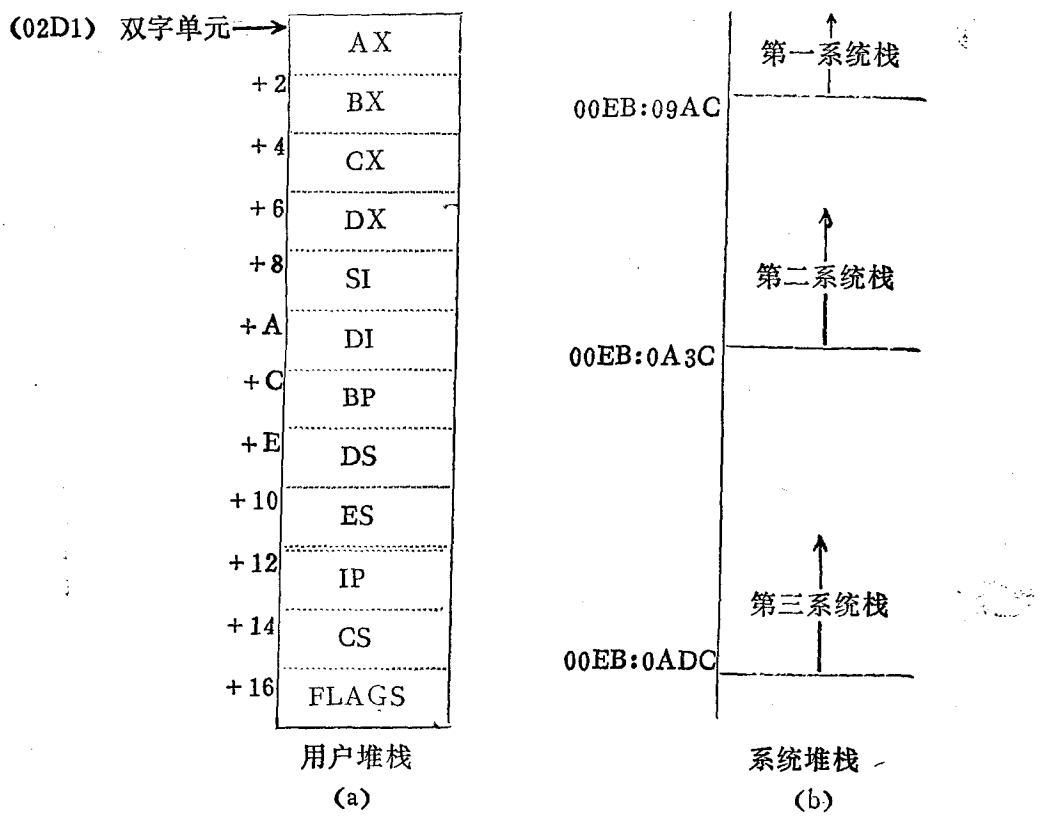

图 1-3 核心态进程中的堆栈

两大类功能调用分别使用不同的系统堆栈，以及用02D1双字单元保存当前堆栈指针，用0ADC双字单元保存原02D1双字单元内存，这一结构原则上允许核心态进程的一次性重入，但先后两次的进入必须是分属不同大类的功能调用。

核心态进程的重入不仅应关心堆栈的使用情况，而且应关心其它的DOS参量，如程序段前缀（PSP）和磁盘传输区（DTA）的地址。因为它们同堆栈指针一样，系统任何时刻