IBM PC 机内存驻留技术

昆明市北京路 333 号自动化站 李晓华

摘要：本文详细分析了实现常驻内存程序的一般方法，讨论了中断向量与常驻内存之间的关系。阐述了设置中断向量的几种基本方法，说明了 *COM* 文件的基本格式。为读者编写常驻程序起到抛砖引玉的作用。

目前大多数软件都具有同时运行多道程序的能力。这些软件一般都有计算器、帮助信息等辅助功能。例如当你在文字编辑过程中，需要对数据进行计算，那么就启动计算器，此时计算机就会出现一个窗口。完成计算后，返回到你的主程序中，这样的软件有许多。如 Turbo C 语言，，当需要帮助时，敲 F1 键，则会出现一个帮助窗口；而在使用 Word Start 过程中按 Ccrl +J 时，会出现帮助菜单等。按行家的话来说：这些软件都具有并发功能。然而，在 IBM PC 机上实现真正并发是很难的，这主要是因为 PC-DOS 是单用户的操作系统，某一时刻只能有一个任务处于活动态中。当然，在 IBM PC 机上可以模拟并发。在这些软件中，它们都提供了一个驻留程序，需要时敲一个特殊的键便能激活它。实质上它是产生了一个中断，由中断服务程序去完成需要的操作。此时正在执行程序被挂起。当中断服务程序结束后，被挂起的任务重新被激活。本文的目的，就是分析如何使一个程序驻留在机器的内存中。

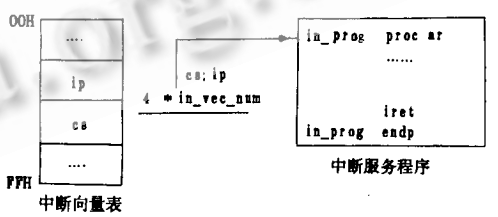
一、内存驻留与中断向量

1.中断向量

内存驻留程序与中断是紧密相联的。PC-DOS 是一个中断驱动型的操作系统。顾名思义，中断是让计算机暂时停此当前的流程，稍后再回过头来继续执行。CPU 接受到中断指令后，就把控制权转移到一个叫做“中断处理程序”(interrupt handler)去执行。当执行完后，原先被中断的程序重新继续执行。

PC 机操作系统提供 2 种类型的中断：ROM BIOS 和 DOS。其基本形式有 2 种：一种是硬件中断(hardware interrupt)，它是由外部设备所产生的；另一种叫软件中断(Software interrupt)，它是由程序产生的。PC 机所提供的有关中断号(interrupt number)和中断向量(interrupt vector)请参考有关资料。这里要注意的是中断向量，每个中断向量长度为 4 字节，它包括中断服务程序的起始地址值。其中前两个字节包含食地址值的偏移(offset)部分，后两字节则为段地址(segment)部分。通过修改中断向量值，就可以使指针指向所需要的中断服务程序。中断向量的地址值＝4 * 中断号码。有关中断向量的设置有几种方法：

(1)直接设置中断向量：由于中断向量表是存放中断报务程序入口地址值的存储位置，因此可以直接把中断报务程序的入口地址值存放到中断向量表中：




```
mov   ax,0
mov   es,ax ;段值为 0
mov   word ptr es: 4 * in-vec -num ,offset in-prog
mov   word ptr es: 4 * in-vec -num [2 ],seg in-prog
```

基中 in-vec -num 是中断向量号码，in-prog 是中断服务程序。

但在这种情况下有可能造成系统死机，主要是因为当执行和 3 条指令后，而第 4 条指令还没有执行时。此

时，若产生了优先级高于该中断号的新的中断，那么该中断的地址值便不完整了。有可能使得系统瘫痪。这样必须考虑增加以下 2 条指令，以便在修改中断向量时禁止其他中断发生。

```
…
cli ；禁止中断
mov   word ptes：4 * in-vec -num , offset in-prog
mov   word ptr es：4 * in-vecnum[2 ], seg in-prog
sti；允许中断
```

增加 cli 、sti 后，似乎是正确了，但请注意一点，即 cli 不能禁止 NMI（不可屏弊中断）。因此当 NMI 产生时，上述的程序就可不能正确执行了。这样不得不考虑在修改中断向量时，不准产生 NMI 中断。在汇编语言中，NMI 中断不会发生在一个完整的指令中，而用 REP MOVSW 修改中断向量时，就可避免上述不足之处。

```
inter -addr dd ?
        …
        mov   word   ptr   inter   -addr, offset in-prog
        mov   word ptr   inter-addr[2 ], seg in-prog
        mov   di, 0
        mov   es, di
        mov   di, 4 * in-vec -num     ；中断号码在中断向量表中
                                      的位置
        mov   si, offset   inter -addr;   中断服务程序的入口地址
        mov   cx, 2
        cld
        cli;关中断
rpe     movsw;                        把中断服务程序的入口地
                                      址（包括段和偏移量）移到
                                      中断向量表中（4 中断向
                                      量号码）
        sti;                          开中断          …
```

(2)使用 DOS 系统调用来设置中断向量：利用现有 DOS 的 21H 系统调用功能号 25H 和 35H，就能设置和读取中断向量。下面是设置中断向量的例子：

```
new-interrupt   proc   far
……
new-interrupt   endp
……
mov   ax, cs
mov   ds, ax
mov   dx, offset   new -interrupt ；新中断服务程序的偏移地址
mov   al, interrupt -number;      中断号
mov   ah, 25h   ;                 设置中断向量功能号
int   21h
下面是读取中断向量的例子：
old -interrupt   dd   ?
……
mov   al, interrupt -number          ；中断号
mov   ah, 35h ;                      取中断向量功能号
int   21h
mov   word   ptr   old -interrupt , bx
mov   word   ptr   old -interrupt [2 ], es
……
```

读取的中断向量地址在 old -interrupt 中。利用 DOS 来设置中断向量是比较好的方法。它既安全又方便，建义读者采用。

2.常驻程序基本结构

(1)**COM 文件**：首先应明确常驻程序必须是 COM 文件。即程序只能加载到一个段内。这就是说程序大小必须小于 64K 字节。下面就是 COM 文件的基本格式：

```
code  segment                          第一部份
      assume   cs: code, ds: code
      org   100h
start :                                第二部份
      …
      ret
code  ends                             第三部份
 end  start
```

程序分为 3 个部分：第 1 部分定义了数据段和程序段在程序中的位置，以及程序的开始地址；第 2 部分是可执行程序；第 3 部分是程序和段在结束。注意程序的执行是从地址 100 H 开始。

(2)**内存常驻程序**：用于常驻内存结束中断程序的有 2 种方法：

(A)使用 INT27H 结束中断，并驻留：该中断主要用

来终止当前程序的执行，但保留分或全部它的代码在内存。使得它不被另一装入的程度覆盖。入口参数为：

DX＝要保留程序的最后一字节加 1 的位移（包括程序段前缀）

CS＝程序段前缀 INT 27H 能保留的最大数目的为 64K 字节；由于执行 INT 27H 的结果，改变了 INT 22H、INT 23H INT24H 的向量，所以它不能用于永久安装用户写的严重错误处理程序；下面是一个例子，说明如何使程序常驻内存。

```
code  segment                          ;第一部份
      assume  cs:code, ds:  code
      org  100h
start:                                 ;  第二部份
      jmp  initi
      ......
init:                                  ;  第三部份
      mov  dx,  offset  initi
      int 27h
code  ends:                            ;  第四部份
      end  start
```

在上述程序中，比 COM 程序多增加了一个部分，即第 3 部分。这部分使 start 和 initi 之间的程序留在内存中。

(B)使用 INT 21H 系统调用功能号 31H 使程序驻留结束进程入口参数：

AH＝31H

AL－返回代码

DX＝保留内存大小(1 段数＝16 字节)

该中断普遍用于驻留驱动程序或子程序；它在 DOS 命令级被调用一次后，随后以软中断方式提供服务；在调用前，应分配内存块的长度；程序装入时，它仅仅处理片蚋存分配；该功能优先于 INT 27H 调用，因它允许传送回一返回码，并且可驻留大于 64K 字节的程序。例如下面程序为带返回码 1 结束，但驻留骨存，保留从程序段前缀开始的 16K 内存。

```
mov  ah,31h   ;功能号
mov  al,1     ;返回码
mov  dx,0400h ;保留长度为节数
int  21h      ;调DOS
```

二、编写一个常驻内存的中断程序

前面，分析了设中断向量的几种方法和常驻程序的基本结构。接下来，通过一个例子说明如何编写一个实用的常驻内存程序。该程序通过修改 05H 中断（按 Print Screen 键产生 05H 中断）来显示一字符串。

```
;file name mem－resi·asm
;MEM－RESI·OBJ:masm mem－resi;
;MEM－RESI·COM:tlink /t mem－resi
dos prtsc in      equ  05h     ;屏幕打印中断
dos function      squ  21h     ;DOS 系统调用
dos term resid    equ  27h     ;结束内存驻留
get vector        equ  35h     ;取中断向量
set vector        equ  25h     ;置中断向量
dos bios          equ  10h     ;BIOS 调用
dos dchar         equ  0eh     ;DOS 调用显示一个字符
code segment
      assume cs:code,ds:code
      org 100h
start:
      jmp ini
old Print Screen in    dd?            ;保存老的05H中断
string db'This is a Print  Screen  memory resident program'
       db 0dh,0ah
       db'Strike ENTER return to DOS。'
       ,0dh,0ah,0
testmsg db'Print Screen memory resident
        alread  loaded。 $'
msg     db'Print Screen memory resident
        program!',0ah,0dh
       db'Copyright (C)Li Xiao Hua 1992。'
        0dh,0ah,'$'
msginst db'Print Screen memory resident is
        install OK!  $';
New－Print－Screen－in proc far
                                  ;新的05H中断服务程序
      jmp    new
testmake db 'prtsc          ;安装信息标志
new: sti
       push  ds
       push  ax
       push  si
       mov   ax,cs
       mov   ds,ax
```

```
        mov   si,offset  string
        call  disp-string          ;调显示字符串子程序
        pop   si
        pop   ax
        pop   ds
        iret
New-Print-Screen-in  endb
disp-strinproc  near               ;显示字符串子程序
        push  si
        push  ax
disp:
        mov   al,[si]
        cmp   al,0
        je    dispdon
        call  dchar                ;调显示一个字符子程序
        inc   si
        jmp   disp
dispdon:
        pop   ax
        pop   si
        ret
disp-string endp
dchar   proc  near                 ;显示一个字符子程序
        push  ax
        push  bx
        mov   bh,1
        mov   ah,dos-dchar
        int   dos-bios
        pop   bx
        pop   ax
        ret
dchar   endp
test proc  near                    ;测试是否已经安装?
    xor   ax,ax
    mov   ds,ax
    mov   bx,05h*4
    ids   si,[bx]
    add   si,3
    mov   cx,3
    push  cs
    pop   es
    mov   di,offset  testmake
    rep   cmpsb
    or    cx,cx
    jnz   testret
    push  cs
    pop   ds
    mov   dx,offset  testmsg       ;显示已安装信息
    mov   ah,9
    int   21h
    mov   ax,4c00h                 ;返回DOS
    int   21h
testret:ret
test  endp
ini   :                            ;初始化
      push  cs
      pop   ds
      mov   dx,offset  msg         ;显示版本信息
      mov   ah,09
      int   21h
      call  test                   ;测试是否已经安装?
      mov   ax,cs
      mov   ds,ax
      mov   al,dos-prtsc-ic
      mov   ah,get-vector
      int   dos-function           ;取05H中断向量
      mov   word ptr  old-Print-Screen-in,bx;保存05H
                                   的偏移
      mov   word ptr  old-Print-Screen-in[2],es; 保存
                                   05H的段址
      mov   ds,offset New-P
rint-Screen-in
      mov   al,dos-prtsc-in
      mov   ah,set-vector
      int   dos-function           ;置中断05H向量的新地
                                   址
      mov   dx,offset ini
      int   dos-term-resid         ;结束驻留
code ends
      end   start
```

▲