

汉字 FoxBASE 下屏幕的随机保存与恢复

林荣庆 (浙江省台州地区信托投资公司)

汉字 FoxBASE 提供了屏幕存取的二条命令,其中 Save Screen 命令把当前屏幕图象存储到内存缓冲区,restore screen 命令则从内存缓冲区把图象恢复到屏幕,利用该功能,不必重画而重现原来已保存的屏幕,既方便程序编制,又可提高显示速度,对于在程序中随时提供帮助功能及保存屏幕重要信息以作其它处理特别有用。

但是这二条命令只能在西文方式下有效,在汉字方式下恢复的屏幕会面目全非,无法使用。其根本原因在于 save screen 和 restore screen 命令只是保存和恢复了字符显示缓冲区的内容,而对字符的属性缓冲区内容未作处理。以东海机为例,汉字显示缓冲区有 64K 字节,分成字符数据区和字符属性区,各占 32K 字节,数据区起始段址是 0B000H,偏址是 0H,属性区起始段址是 0B000H,偏址是 8000,显示缓冲区分八页,每页包括 4K 字节数据区和 4K 字节属性区,要存取汉字必须同时存取数据区和属性区的信息。

FoxBASE 提供了调用汇编语言程序的功能,又为我们用汇编语言程序扩充 FoxBASE 的功能提供了极大的方便,笔者用汇编语言编制汉字屏幕随机存取程序,可以取代 save screen 和 restore screen 命令。

FoxBASE 调用汇编程序有二种途径,第一种是命令 Run(文件名),执行操作系统的文件,然后返回到 FoxBASE,其中文件名是带有扩展名.EXE 的运行文件,汇编语言执行程序生成步骤如下:

```
MASM pmsa0;
```

```
LINK pmsa0;
```

为了使用 RUN 命令,DOS 的 Command.com 文件必须存于当前目录,也可以用 DOS 的 path 参数设置一个 Command.com 文件查找的路径。

第二种是命令 Call(文件名)[with(参数)],执行内存中带有扩展名.BIN 的文件,然后返回到 FoxBASE,汇编语言的.BIN 文件生成步骤如下:

```
MASM pmsa1;
```

```
LINK pmsa1;
```

```
EXE2BINL pmsa1 pmsa1;
```

使用 call 使用时,应先用 LOAC(文件名)命令,把带有.BIN 扩展名的二进制程序装入内存。

为随机执行民间的保存和恢复程序,可使用 FoxBASE 提供的按键处理命令,动态地控制程序的流程。

```
on key = 315 DO pmsare with sys(16)
```

命令中设置 F1 为热键,符合国际通用规定,在应用程序执行的任何时候,随时可按下热键 F1,则执行 pmsare 过程,实现屏幕的保存和恢复功能,pmsare 过程处理结束后,由 RETURN 返回到应用程序中断处继续执行。

例子中 SCREEN.PRG 是 FoxBASE 的主程序,pmsare 是 FoxBASE 的过程,pmsa0.ASM 把屏幕信息保存到文件,pmre0.ASM 把文件信息恢复到屏幕,pmas1.ASM 把当前屏幕信息暂存到显示缓冲区的第一页,pmre1.ASM 则把第一的信息恢复到当前屏幕。

本文所列程序均在东海机上通过。

```
B> type screen.prg
set procedure to screen
set talk off
on key = 315 do pmsare with sys(16)
do while .t.
clear
z = 0
@ 6,20 say " * * * * * "
@ 7,20 say " 1.功能选择一  "
@ 8,20 say " 2.功能选择二  "
@ 9,20 say " 3.退出      "
@ 10,20 say " * * * * * "
@ 11,20 say "请选择:" get z
read
do case
```

```

case z=0
close all
return
case z=1
do gn1
case z=2
do gn2
endcase
enddo
procedure gn1
clear
@ 2,22 say"执行功能一,按任一键返回。"
wait
return
procedure gn2
clear
@ 3,33 say"执行功能二,按任一键返回。"
wait
return
procedure pmsave
parameter a
clear typehead
@ 13,33 say"按任一键执行保存屏幕"
wait
run pmsa0.exe
@ 14,33 say"按任一键执行清除屏幕"
wait
clear
@ 15,33 say"按任一键执行恢复屏幕"
wait
run pmre0.exe
@ 16,33 say"按任一键再执行保存屏幕"wait
load pmsal
call pmsal
@ 17,33 say"按任一键再执行清除屏幕"
wait
clear
@ 18,33 say"按任一键再执行恢复屏幕"wait
load pmrel
call pmrel
return
B> type pmsa0.asm
data segment
    handle dw 0
    file db 'b:\pmsave.dat',0
    buff db 8192 dup(0)
data ends
code segment
    assume cs:code,ds:data
pm proc far
start:push ds
    push es
    push ax
    mov ah,3dh
    mov al,0h
    mov dx,data

```

```

push es
push ax
mov dx,0b000h
mov ds,dx
mov bx,data
mov es,bx
mov si,offset 0000h
mov di offset buff
mov cx,1000h
repz movsb
mov si,8000h
mov di,offset buff+4096
mov cx,1000h
repz movsb
mov ah,3eh
mov cx,0
mov dx,data
mov ds,dx
mov dx,offset file
int 21h
mov handle,ax
mov ah,40h
mov bx,handle
mov cx,2000h
mov dx,data
mov ds,dx
mov dx,offset buff
int 21h
mov ah,3eh
mov bx,handle
int 21h
ret
pm endp
code ends
end start
B>

```

```

B> type pmre0.asm
data segment
    handle dw 0
    file db 'b:\pmsave.dat',0
    buff db 8192 dup(0)
data ends
code segment
    assume cs:code ,ds:data
pm proc far
start:push ds
    push es
    push ax
    mov ah,3dh
    mov al,0h
    mov dx,data

```

```

mov ds,dx
mov dx,offset file
int 21h
mov handle,ax
mov ah,3fh
mov bx,handle
mov dx,data
mov ds,dx
mov dx,offset buff
mov cx,2000h
int 21h
mov dx,data
mov ds,dx
mov bx,0b000h
mov es,bx
mov si,offset buff
mov di,offset 0000h
mov cx,1000h
repz movsb
mov si,offset buff+4096
mov di,8000h
mov cx,1000h
repz movsb
mov ah,05h
mov al,0
int 10h
mov ah,3eh
mov bx,handle
int 21h
ret
pm endp
code ends
end start

```

```

B> type pmsal.asm
code segment word
assume cs:code
org 0h
i0[pm ] proc far
push ax
push bx
push cx
push dx
push si
push di
push ds
mov cx,0b000h
mov ds,cx
mov es,cx
mov si,offset 0000h
mov di,offset 1000h

```

```

mov cx,1000h
repz movsb
mov si,offset 8000h
mov di,offset 09000h
mov cx,1000h
repz movsb
pop ds
pop di
pop si
pop dx
pop cx
pop bx
pop ax
ret
pm endp
code ends
end pm

```

```

B> type pmrel.asm
code segment word
assume cs:code
org 0h
pm proc far
push ax
push bx
push cx
push dx
push si
push di
push ds
mov cx,0b000h
mov ds,cx
mov es,cx
mov si,offset 0000h
mov di,offset 1000h
mov cx,1000h
repz movsb
mov si,offset 8000h
mov di,offset 09000h
mov cx,1000h
repz movsb
pop ds
pop di
pop si
pop dx
pop cx
pop bx
pop ax
ret
pm endp
code ends
end pm

```