Buletinul Ştiinţific al Universităţii din Baia Mare
Seria B,Matematică-Informatică,vol.XI(1995),121-128

Dedicated to the Centenary of "Gazeta Matematică"

SCREEN SAVERS FOR MS DOS

Ovidiu COSMA

The MS DOS operating system does not provide a screen saving mechanism. Yet the problem is solved at BIOS level for the new models of "Green PCs". Their BIOS is capable to clear the screen and to command the monitor passing in a wait state, with little power consumption.

This article presents a method for developing TSR programs to solve the screen saving problem. A screen saver automatically displays a dynamic image that uniformly covers the whole surface of the screen, in the inactivity periods, and then the system "wakes up" at request. This mechanism must not disturb the other programs in the system. Thus the screen saver must not reprogram the interrupts, and it's length must be as little as possible.

Activating / deactivating the saving mechanism

The method presented below overrides the keyboard and the timer interrupts. The timer interrupt increments a counter that is cleared by the keyboard interrupt. Therefore the dynamic image can

be activated by the timer interrupt when this counter reaches a certain value. If the saver image is currently on the screen, the keyboard interrupt has to restore the original image.

The keyboard interrupt

```
tas:                            ;The new keyboard interrupt
     call salv_context          ;Save the CPU registers
     mov  con_tast,0            ;con_tast = counter incremented by the new
;timer interrupt. If it's value reaches a certain value, the screen
;saver is activated. The keyboard interrupt signal operator
;activity, and must reset this counter in order to delay the saver
;activation.
     cmp  st_serp,1 ;Is the image of the screen saver active?
     jnz  nu_sunt   ;If not, continue with the original keyboard
;routine if yes, this is the command to resume activity, and the
;following section must prevent the key code from being passed to
;the other running programs.
     in   al,60h    ;An interrupt is generated for pressing and
                    ;for releasing of keys
     test al,10000000b
     jnz  apas      ;The key was pressed, the screen saver will
                    ;be deactivated at the releasing of the key.
     mov  st_serp,0 ;The key was released, clear the "screen
     call ref_ec    ;saver active" flag and restore the image on
                    ;the screen.
apas:
     in   al,61h    ;bit 7, pot B of the 8255 chip is used to send
;the acknowledge signal to the keyboard microprocessor. These port
;addresses apply to the AT as well, even though it does not have an
;8255 chip.
     or   al,10000000b ;The acknowledge signal contains 2 steps
     out  61h,al    ;Step 1: set
     and  al,01111111b
     out  61h,al    ;Step 2: reset, ready the acknowledge signal
```

```
     mov  al,20h    ;Signal end of hardware interrupt
     out  20h,al    ;to the 8259 interrupt controller
     call ref_context  ;Restore the CPU registers
     iret
nu_sunt:
     call ref_context
     jmp  DWORD PTR cs:[tasta] ;to the original interrupt 9h
                               ;service routine
```

The timer interrupt

```
timp:                          ;The new timer interrupt routine
     pushf
     call DWORD PTR cs:[inceput]     ;Call to the original int 8h
                                     ;service routine
     call salv_context               ;Save CPU registers
     cmp  st_serp,1 ;Screen saver active?
     jnz  temporiz  ;If not, just increment the delay counter
     int  61h       ;This  software  interrupt  is  used  as  an
;interface with other programs that display moving images on the
;screen. This programs can be much simpler because they do not need
;to the save / restore the screen image, monitor the keyboard, keep
;delay counters, etc. Thus the image of the screen saver can be
;improved with little memory consumption. Normally the software
;interrupt 61h is not used by the operating system.
     inc  contor          ;The next counter controls the speed of
     mov  al,vit_ser      ;the screen saver image.
     cmp  contor,al
     jl   mai_astept
     mov  contor,0
     call afis_tmp        ;Call to the display routine
mai_astept:
     call ref_context     ;Restore the CPU registers
     iret                 ;Ends the timer interrupt and restores
                          ;the control to the interrupted program.
```

```
temporiz:
     inc  con_tast  ;This counter determines the waiting period
     cmp  con_tast,6000  ;before activating the screen saver
     jl   mai_astept
;The waiting period is over, activate the screen saver
     mov  ah,0fh    ;The mechanism works only in character modes.
                    ;Usually the graphic interfaces have there
                    ;own screen savers.
     int  10h       ;al-video mode
     cmp  al,2
     jz   mod_bun
     cmp  al,3
     jz   mod_bun
     mov  con_tast,0     ;The screen is not in one of the usual
                         ;character modes
     jmp  mai_astept     ;Just clear the counter and do not
                         ;activate the screen saver image
mod_bun:
     inc  st_serp        ;Set the 'screen saver active' flag
     call salv_ec        ;Saves the screen image
     call sterg_ec       ;Clears the screen
     jmp  mai_astept     ;Terminates the interrupt
int61:    ;The software int 61h terminates immediately. It can be
     iret ;used as a clock by other programs that display moving
          ;images on the screen.
adr_retur dw   ?    ;Keep the return address
salv_context:       ;Save the general registers
     cli            ;Ensures that no more than one interrupt
     pop  adr_retur ;service routine accesses this section at the
                    ;same time.
     push ax   ;The return address is on the top of the stack
     push bx
     push cx
     push dx
     push si
```

```
        push di
        push ds
        push es
        push adr_retur ;Prepares the return address
        sti
        ret
ref_context:           ;Restore the general registers
        cli            ;This routine works likewise the previous one
        pop  adr_retur
        pop  es
        pop  ds
        pop  di
        pop  si
        pop  dx
        pop  cx
        pop  bx
        pop  ax
        push adr_retur
        sti
        ret
```

Saving /clearing / restoring the screen image

For smooth operation, the image can be saved in the internal memory. It is not always possible to use the video memory for this purpose (eventually just switch to another video page) because the monochrome video adapter that is currently in use does not have more than one video page. If we do not consider this aspect, our screen saver will not work on monochrome video adapters.

```
salv_ec:               ;Saves the screen image
        mov  es,st_buf ;Address of the beginning of the video
memory. This
        push cs        ;depends on the video mode.
        push es
        pop  ds
```

```
     pop  es
     mov  si,0       ;ds:si=beginning of video memory
     mov  di,offset ds:buf_ec ;es:di-address of the buffer
     cld
     mov  cx,80*25   ;Number of bytes to transfer = number of
rep  movsw           ;characters on the screen
     ret
sterg_ec:            ;Clears the screen
     mov  es,st_buf
     mov  di,0       ;es:di=beginning of the video memory
     cld
     mov  cx,80*25   ;Number of characters on the screen
     mov  ax,0       ;Character used to clear the screen
rep  stosw           ;Fills the video memory with 0
     ret
ref_ec:                   ;Restores the screen image
     push cs
     pop  ds              ;ds=cs
     mov  es,st_buf
     mov  di,0            ;es:di=beginning of the video memory
     mov  si,offset ds:buf_ec ;ds:si=address of the buffer
     cld
     mov  cx,80*25        ;Number of bytes to transfer
rep  movsw                ;Transfers the contents of the buffer in
     ret                  ;the video memory
st_buf label WORD dw 0b800h      ;Address of the beginning of
                          ;video memory
buf_ec label WORD dw 80*25 dup(?) ;Buffer for saving the video
;memory. The length of the buffer - number of characters on the
;screen*2. Each character on the screen occupies 2 bytes in the
;video memory. The first byte holds the ASCII code of the character
;and the second the display attributes.
```

The image routine

The main task of the screen saver image routine is to

uniformly cover the whole screen. There are 2 ways to follow for developing such a routine: one of using a random number generator and the other one of systematically going through the whole screen. The second way seems to be more efficient.

This routine has to allow the other programs in the system to function normally. Thus at every appellation, after moving the screen image with a step, the image routine ends and frees the system resources. This approach allows the system to continue the current operations after the activation of the screen saver image.

The appearance of the screen saver image can be improved with little TSR's that use the software interrupt 61h, generated in the timer interrupt.

REFERENCES

1.INTEL, "MCS-86 Macro Assembly Language Reference Manual", 1979.

2.PETER NORTON, MICROSOFT PRESS "PC Programmer's Bible", 1993.

3.MICROSOFT CORPORATION, "MS-DOS Programmer's Reference 6.0", 93.

University of Baia Mare
Victoriei 76,4800 Baia Mare
ROMANIA