

```

;*****;
;*              S O U N D A              *;
;*-----*
;*   Task           : Plays a scale between octaves 3 and 5 of the *;
;*                   PC's musical range. This routine can be used *;
;*                   for other applications                         *;
;*-----*
;*   Author          : MICHAEL TISCHER                             *;
;*   Developed on     : 08/06/1987                                  *;
;*   Last update      : 04/12/95                                    *;
;*-----*
;*   Assembly        : MASM SOUNDA;                                *;
;*                   LINK SOUNDA;                                  *;
;*                   EXE2BIN SOUNDA SOUNDA.COM                     *;
;*-----*
;*   Call from DOS    : SOUNDA                                     *;
;*****;

```

```
code      segment para 'CODE'      ;Definition of CODE segments
```

```
org 100h      ;Starts at address 100H
              ;directly following PSP
```

```
assume cs:code, ds:code, es:code, ss:code
```

```
== Program ==
```

```
sound      proc near
```

```
    ;-- Display message -----
```

```
    mov ah,9      ;Function number for displaying string
```

```

mov  dx,offset initm      ;String's offset address
int  21h                  ;Call DOS interrupt 21H

;-- Play scale -----

xor  bl,bl                ;Start at C of octave 3
mov  dl,9                 ;for duration of 1/2 second
nextune: call play_tune    ;Play note
inc  bl                  ;Next note
cmp  bl,36               ;All notes in this octave played?
jne  nextune             ;NO --> Play next note

;-- Display end message -----

mov  ah,9                ;Function number for string display
mov  dx,offset endmes     ;String's offset address
int  21h                 ;Call DOS interrupt 21H

mov  ax,4C00h            ;Program ends when call to a DOS
int  21h                 ;function results in an error code
                        ;of 0

sound    endp

;== Main program data =====
initm    db 13,10,"SOUND (c) 1987 by Michael Tischer",13,10,13,10
          db "Your PC should now be playing a chromatic scale in the "
          db "3rd and 5th      ",13,10,"octaves of its range, if    "
          db "your PC speaker works.",13,10,"$"

endmes    db 13,10,"End",13,10,"$"

```

```

;-- PLAY_TUNE: Play a note -----
;-- Input      : BL = Note number (relative to C of the 3rd octave)
;--           : DL = Duration of note in 1/18 second increments
;-- Output     : none
;-- Register   : AX, CX, ES and FLAGS are changed
;-- Info       : Immediately after the tones, control returns to the
;--           : calling routine

```

play_tune proc near

```

    push dx                ;Push DX and BX onto the stack
    push bx

    ;-- Adapt timer interrupt to user program -----
    push dx                ;Push DX and BX onto stack
    push bx
    mov ax,351ch           ;Get address of time interrupt
    int 21h               ;Call DOS interrupt
    mov old_time,bx        ;Offset address of old interrupt
    mov old_time+2,es      ;and note segment address

    mov dx,offset sound_ti ;Offset address of new timer routine
    mov ax,251ch           ;Set new timer routine
    int 21h               ;Call DOS interrupt
    pop bx                 ;Pop BX and DX off of stack
    pop dx

    mov al,182             ;Prepare to play note
    out 43h,al             ;Send value to time command register
    xor bh,bh              ;BH for addressing note table = 0
    shl bx,1               ;Double note number (fr. word table)

```

```

    mov ax,[note+bx]      ;Get tone value
    out 42h,al             ;LO-byte on timer counter register
    mov al,ah             ;Transfer HI-byte to AL
    out 42h,al            ;and to timer counter register
    in al,61h             ;Read speaker control bit
    or al,11b             ;Lowest two bits enable speaker
    mov s_ende,1          ;Note still has to be played
    mov s_counter,d1      ;Play note for duration
    out 61h,al            ;Disable speaker

play:  cmp s_ende,0        ;Note finished?
       jne play           ;N) --> Wait

       in al,61h          ;Read speaker control bit
       and al,11111100b   ;Clear lowest two bits
       out 61h,al         ;Disable speaker

;-- Reactivate old timer interrupt -----

    mov cx,ds             ;Note DS
    mov ax,251ch          ;Set function no. for intrrpt vector
    lds dx,dword ptr old_time ;Load old address into DS:DX
    int 21h               ;Call DOS interrupt
    mov ds,cx             ;Return DS

    pop bx                ;Pop BX and DX off of stack
    pop dx
    ret                   ;Return to calling program

play_tune endp

;-- new timer interrupt -----

```

```

sound_ti  proc far                                ;Call 18 times per second

            dec  cs:s_counter                      ;Decrement counter
            jne  st_ende                          ;If still >0, end
            mov  cs:s_ende,0                      ;Signal note end
st_ende:   jmp  dword ptr cs:[old_time] ;Goto old timer interrupt

sound_ti  endp

;== Variable set needed by the routines =====

old_time  dw  (?), (?)                          ;Address of old timer interrupt
s_counter db  (?)                              ;counter for note duration in 1/18
                                                ;second increments
s_ende    db  (?)                              ;Displays whether note already played
note      dw  9121,8609,8126,7670 ;Note values for octave 3
            dw  7239,6833,6449,6087
            dw  5746,5423,5119,4831
            dw  4560,4304,4063,3834 ;Note values for octave 4
            dw  3619,3416,3224,3043
            dw  2873,2711,2559,2415
            dw  2280,2152,2031,1917 ;Note values for octave 5
            dw  1809,1715,1612,1521
            dw  1436,1355,1292,1207

;== End =====

code      ends                                  ;End of CODE segment
            end  sound                          ;End of assembler program

```

