chapter14

1. 访问CMOS RAM

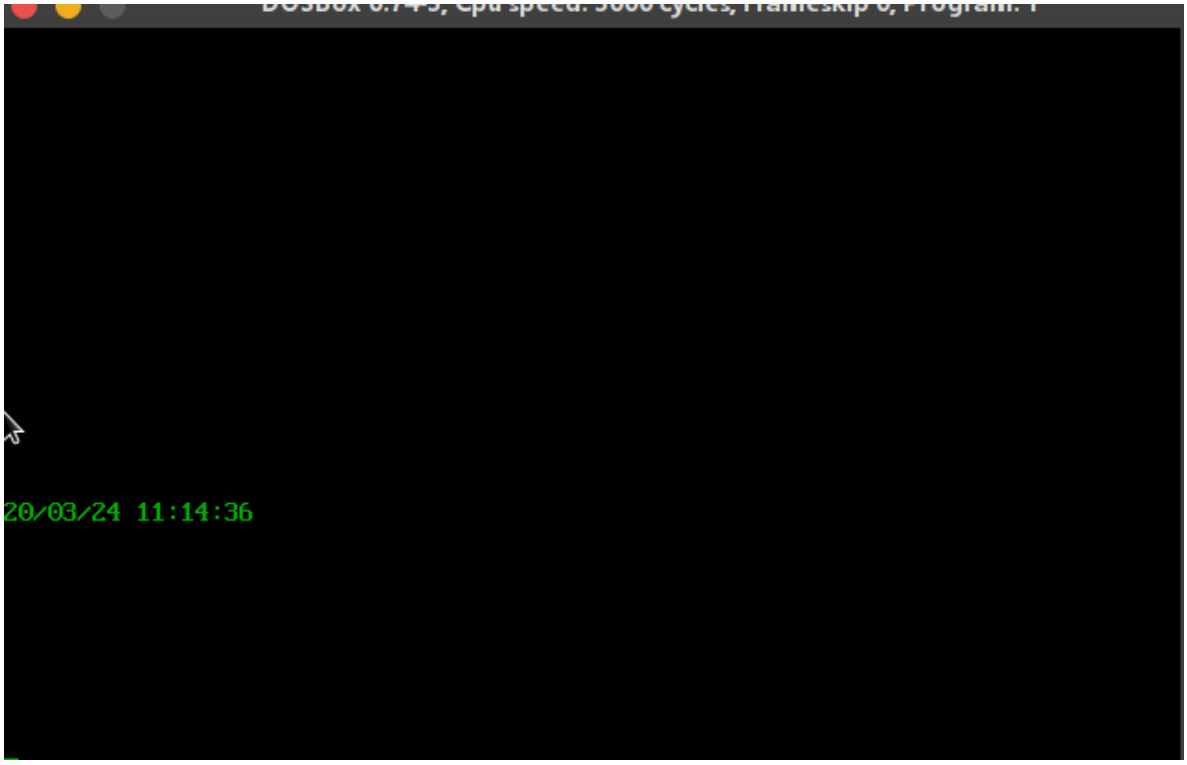
code

```
assume cs:code,ds:data,ss:stack

stack segment
    db 128 dup(0)
stack ends

data segment
    db 'YY/MM/DD hh:mm:ss',0
data ends


code segment

TIME_CMOS:  db  9,8,7,4,2,0

start:
    mov ax,stack
    mov ss,ax
    mov sp,128

    call clear_screen
    call show_init
    stime:
    call show_time
    jmp stime
    mov ax,4c00h
    int 21h

;============================
show_time:
    push ax
    push bx
    push cx
    push dx
    push ds
    push es
    push si
    push di
    show_time_bg:
        mov ax,0b800h
        mov es,ax
        mov di,16*160

        mov ax,cs
        mov ds,ax
        mov si,offset TIME_CMOS
```

```
47
48          mov cx,6
49          s1:
50              push cx
51
52              mov al,ds:[si]
53              out 70h,al
54              in al,71h
55
56              mov ah,al
57              mov cl,4
58              shr ah,cl
59              and al,00001111b
60
61              add ah,30h
62              add al,30h
63
64              mov byte ptr es:[di],ah
65              mov byte ptr es:[di+2],al
66
67              inc si
68              add di,6
69
70              pop cx
71              loop s1
72
73      show_time_end:
74          pop di
75          pop si
76          pop es
77          pop ds
78          pop dx
79          pop cx
80          pop bx
81          pop ax
82          ret
83  ;=============================
84  show_init:
85      push ax
86      push bx
87      push cx
88      push dx
89      push ds
90      push es
91      push si
92      push di
93      show_init_bg:
94          mov ax,data
95          mov ds,ax
96          mov si,0
97
98          mov ax,0b800h
99          mov es,ax
100         mov di,16*160
101
102         s:
103             mov al,ds:[si]
104             cmp al,0
```

```
            je show_init_end
            mov ah,2
            mov es:[di],ax
            inc si
            inc di
            inc di
            jmp s
    show_init_end:
        pop di
        pop si
        pop es
        pop ds
        pop dx
        pop cx
        pop bx
        pop ax
        ret
;=============================
clear_screen:
    push ax
    push bx
    push cx
    push dx
    push ds
    push es
    push si
    push di
    clear_screen_bg:
            mov bx,0b800h
            mov es,bx

            mov bx,0
            mov dl,0
            mov dh,00000010b
            mov cx,2000

    clearScreen:
                    mov es:[bx],dx
                    add bx,2

                    loop clearScreen

    clear_screen_end:
        pop di
        pop si
        pop es
        pop ds
        pop dx
        pop cx
        pop bx
        pop ax
        ret
code ends
end start
```

截屏


20/03/24 11:14:36