

```
; *****
;
; SCLOCK.ASM - print current date & time on serial port 1 or 2 (continuously)
; -----
;
; RETRO UNIX 8086 (Retro Unix == Turkish Rational Unix)
; Operating System Project (v0.1) by ERDOGAN TAN (Beginning: 11/07/2012)
; 1.44 MB Floppy Disk
;
; [ Last Modification: 20/01/2014 ]
;
; Derivation from UNIX Operating System (v1.0 for PDP-11)
; (Original) Source Code by Ken Thompson (1971-1972)
; <Bell Laboratories (17/3/1972)>
; <Preliminary Release of UNIX Implementation Document> (Section E.12)
;
; *****
; SCLOCK0.ASM (13/12/2013) <<modified from CLOCK0.ASM (12/12/2013)>>
;
```

.8086

```
; UNIX v1 system calls
```

```
_rele equ 0
_exit equ 1
_fork equ 2
_read equ 3
_write equ 4
_open equ 5
_close equ 6
_wait equ 7
_creat equ 8
_link equ 9
_unlinkequ 10
_exec equ 11
_chdir equ 12
_time equ 13
_mkdir equ 14
_chmod equ 15
_chown equ 16
_break equ 17
_stat equ 18
_seek equ 19
_tell equ 20
_mount equ 21
_umountequ 22
_setuidequ 23
_getuidequ 24
_stime equ 25
_quit equ 26
_intr equ 27
_fstat equ 28
_emt equ 29
_mdate equ 30
_stty equ 31
_gtty equ 32
_ilginsequ 33
```

```
sys macro syscallnumber, arg1, arg2, arg3
```

```
; Retro UNIX 8086 v1 system call.
```

```
ifnb <arg1>
```

```
mov bx, arg1
```

```
endif
```

```
ifnb <arg2>
```

```
mov cx, arg2
```

```
endif
```

```
ifnb <arg3>
```

```
mov dx, arg3
```

```
endif
```

```
mov ax, syscallnumber
```

```
int 20h
```

```
endm
```

```
; Retro UNIX 8086 v1 system call format:
```

```
; sys syscall (ax) <arg1 (bx)>, <arg2 (cx)>, <arg3 (dx)>
```

```

code    SEGMENT PUBLIC 'CODE'
        assume cs:code,ds:code,es:code,ss:code

start_code:
        mov     al, '1'
        pop     bx ; argument count
        cmp     bx, 1
        jna     short @f
        pop     bx
        pop     bx
        mov     ah, byte ptr [BX]
        cmp     ah, '1'
        je      short @f
        cmp     ah, '2'
        jne     short error
        inc     al

@@:
        mov     byte ptr [COMPORT], al
        sys     _open, device, 1
        jnc     short @f

error:
        mov     byte ptr [chr], 07h
        sys     _write, 1, chr, 1
        jmp     short _xit

@@:
        mov     word ptr [dnum], ax

        xor     bx, bx
        xor     cx, cx

clk0:
        push    bx ; 0, hw
        push    cx ; 0, lw
        sys     _write, ax, dt_txt, dt_size
        sys     _time
        ; DX:AX = Unix epoch time
        pop     cx ; previous time, lw
        pop     bx ; previous time, hw
        cmp     ax, cx
        jne     short clk1
        cmp     dx, bx
        je      short clk4

clk1:
        push    dx ; current time, hw
        push    ax ; current time, lw
        call    ctime
        mov     bx, word ptr [dnum]
        mov     cx, offset cbuf
        mov     dx, 25
        sys     _write
        mov     cx, offset nl
        mov     dl, 2
        sys     _write
        sys     _gtty, 0, 0
        ;cmp     bl, 1Bh ; ESC key
        ;je      short clk3 ; exit
        or      bx, bx
        jz      short clk2
        sys     _read, 0, chr, 1
        cmp     byte ptr [chr], 1Bh ; ESC key
        je      short clk3 ; exit

clk2:
        pop     cx ; current -> previous time, lw
        pop     bx ; current -> previous time, hw
        jmp     short clk0

clk3:
        pop     ax
        pop     ax
        mov     bx, word ptr [dnum]
        sys     _close

_xit:
        sys     _exit

clk4:
        nop
        nop
        nop
        nop
        jmp     short clk0

```

```
include ctime.inc

dnum:  dw 0

device:
        db '/dev/COM'
COMPORT:
        db 0
        db 0

dt_txt:
        db 0Dh, 0Ah
        db 'Current Date & Time : '
dt_size equ $ - offset dt_txt
chr:    db 0

nl:     db 0Dh, 0Ah, 0

code    ends

        end  start_code
```