

```
; *****
; GETTY.ASM (Retro Unix 8086 v1 - /etc/getty)
; -----
;
; RETRO UNIX 8086 (Retro Unix == Turkish Rational Unix)
; Operating System Project (v0.1) by ERDOGAN TAN (Beginning: 11/07/2012)
; Retro UNIX 8086 v1 - /etc/getty file
;
; [ Last Modification: 26/06/2014 ]
;
; Derivation from UNIX Operating System (v1.0 for PDP-11)
; (Original) Source Code by Ken Thompson (Bell Laboratories, 1971-1972)
;
; *****
; Derived from 'getty.s' file of original UNIX v1
;
; GETTY07.ASM, 22/05/2014 --> serial port modifications
; GETTY06.ASM, 17/01/2014
; GETTY05.ASM, 06/11/2013, 06/12/2013

.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
_unlink equ 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
_umount equ 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
_sleep equ 34 ; 11/06/2014 (Retro UNIX 8086 v1 Feature Only!)

ENTERKEY equ 0Dh
NEXTLINE equ 0Ah
BACKSPACE equ 08h
; 22/05/2014
EOT equ 04h ; 'End Of Transfer' for serial ports

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)>
```

```
UNIX    SEGMENT PUBLIC 'CODE'
        assume cs:UNIX,ds:UNIX,es:UNIX,ss:UNIX
```

```
START_CODE:
    sys _quit, 0
    sys _intr, 0

    sys _gtty, 0, 1 ; get status of console tty (w)
    jc terminate
    ;
    mov byte ptr [console], al
    add al, '0'
    mov byte ptr [ttynum], al
@@:
    sys _write, 1, msglogin, ml_size
    ;
    mov al, byte ptr [console]
    ;
    jnc short @f

    cmp al, 8
    jb terminate

    sys _sleep ; 11/06/2014
    jmp short @b
@@:
    ; mov word ptr [cursorpos], 0FF00h
    ; mov byte ptr [cposll], 0
    cmp al, 7
    ja short @f
    ;
    sys _gtty, 0, 1 ; get status of console tty (w)
    jc terminate

    mov word ptr [cursorpos], bx
    mov byte ptr [cposll], bl
    mov byte ptr [chr], 07h ; bell/beep
@@:
    sys _write, 1, chr, 1
    jnc short @f

    cmp byte ptr [console], 8
    jb terminate

    sys _sleep ; 11/06/2014
    jmp short @b
@@:
    mov di, offset uname
    mov si, di ; 26/06/2014
getc:
    sys _read, 0, chr, 1
    jnc short @f

    cmp byte ptr [console], 8
    jb terminate
    sys _sleep ; 11/06/2014
    jmp short getc
@@:
    mov al, byte ptr [chr]

    or al, al ; EOT for Retro UNIX 8086 v1
    jz short g5 ; (login via serial ports)

    cmp al, 20h
    jb short g1

    cmp al, 127
    je short g2

    cmp di, offset uname + 16
    jnb short g3
```

```

putc:
    stosb
    inc byte ptr [cursorpos]
    ; 26/06/2014
    cmp si, di
    jnb short g0
    mov si, di
g0:
    sys _write, 1, chr, 1
    jnc shortgetc

    cmp byte ptr [console], 8
    jb short terminate
    sys _sleep
    jmp short g0
g1:
    cmp al, ENTERKEY ; \r (carriage return)
    je short g5
@@:
    ; cmp al, NEXTLINE ; \n (next line)
    ; je short g5

    cmp al, BACKSPACE ; \b (back space)
    jne short g3 ; 19/06/2014
g2:
    mov dx, word ptr [cursorpos]
    ; dh = FFh for serial ports
    cmp dl, byte ptr [cposll] ; left limit
    ja short g4
g3:
    mov byte ptr [chr], 07h
    ; sys _write, 1, chr, 1
    ; jc short terminate
    ; jmp shortgetc
    jmp short g0
g4:
    dec dl
    mov byte ptr [cursorpos], dl

    mov cl, byte ptr [console]

    ; 26/06/2014
    cmp cl, 8
    jb short @f
    dec di
    mov byte ptr [chr], al ; BACKSPACE
    jmp short g0
@@:
    mov ch, 20h ; ch < FFh & ch > 0 -> write 20h
    ; (space) at requested cursor position
    xor bx, bx ; 0
    ; dh = FFh for serial ports = do not set comm. params.)
    sys _stty ; set cursor pos. for console tty
    ; (back space)
    jc short terminate
    dec di
    jmp shortgetc
g5:
    ; 26/06/2014
    mov byte ptr [SI], 0 ; ASCII string
    cmp byte ptr [SI]-1, 20h
    jne short GO
    mov byte ptr [SI]-1, 0
GO:
    sys _exec, login, loginp
terminate:
    sys _exit
here:
    hlt
    jmp short here

EVEN
loginp: dw login
        dw uname
        dw 0
EVEN
chr:    db 0
;EVEN
console: db 0 ; console tty

```

```

; cursor position
cposll: db 0 ; left limit of cursor position
cursorpos: db 0 ; row (for backspace)
           db 0FFh ; column
           ; (FFh for serial ports, for sysstty)
;(cursorpos will set by return of sysgtty for psuedo ttys)

```

```

;EVEN
login: db '/bin/login', 0

```

```

EVEN
msglogin:
           db 0Dh, 0Ah
           db 'Retro Unix 8086 v1 (tty'
ttynum: db 'x'
           db ')'
           db 0Dh, 0Ah
           db 'login : '
ml_size equ $ - offset msglogin
           db 0

```

```

EVEN
uname: db 16 dup(0)

```

```

UNIX ends

```

```

; // getty -- get name and tty mode
; // for initialization
;
; // cycle through speeds and "login:" messages
; // summarized in itab
;
;stty = 31.
;
; sys quit; 0
; sys intr; 0
;0:
; jsr r5,nextspeed
;1:
; mov $name,r5
;2:
; jsr r5,getc
; cmp r0,$174
; beq 5f
; cmp r0,$176
; beq 5f
; cmp r0,$'\n
; beq 1f
; cmp r0,$'\r
; beq 4f
; cmp r0,$'@
; beq 1b
; cmp r0,$'#
; bne 3f
; cmp r5,$name
; blos 2b
; dec r5
; br 2b
;3:
; movb r0,(r5)+
; br 2b
;4:
; bis $20,flags /cr bit
; mov $1,r0
; sys write; nl; 1
; br 2f
;5:
; mov $tab2741,itabp
; inc nowr
; br 0b
;1:
; mov $1,r0
; sys write; cr; 1
;2:
; clrb (r5)+
;
; // determine whether terminal is upper-case only
;
; cmp r5,$name+1

```

```

;      bhi      1f
;      bic      $4,flags      /no data-assume lc
;1:
;      mov      $name,r5
;1:
;      movb     (r5)+,r0
;      beq      1f
;      cmp      r0,$'A
;      blo      2f
;      cmp      r0,$'Z
;      bhi      2f
;      add      $40,r0      / map to lc
;      movb     r0,-1(r5)
;      br       1b
;2:
;      cmp      r0,$'a
;      blo      1b
;      cmp      r0,$'z
;      bhi      1b
;      bic      $4,flags
;      br       1b
;1:
;      clr      r0
;      mov      fstate,r4
;      bis      flags,4(r4)
;      sys      stty; fstate: ..
;
;go:
;      sys      exec; login; loginp
;      sys      exit
;
;getc:
;      clr      r0
;      sys      read; ch; 1
;      tst      r0
;      beq      done
;      mov      ch,r2
;      beq      1f
;getcl:
;      cmp      r2,$174
;      bhis     3f
;      tst      nowr
;      bne      3f
;      mov      $1,r0
;      sys      write; ch; 1
;3:
;      mov      r2,r0
;      rts      r5
;1:
;      dec      $0      / wait a while
;      bne      1b
;      mov      $name,(sp)
;      jsr      r5,nextspeed
;2:
;      clr      r0      / flush nulls
;      sys      read; ch; 1
;      tst      r0
;      beq      done
;      movb     ch,r2
;      beq      2b
;      br       getcl
;
;done:
;      sys      exit
;
;nextspeed:
;      mov      itabp,r1
;      mov      (r1)+,0f
;      bne      1f
;      mov      $itab,itabp
;      br       nextspeed
;1:
;      clr      r0
;      sys      stty; 0:...
;      bes      go
;      mov      (r1)+,-(sp)
;      mov      (r1)+,fstate
;      mov      r1,itabp
;      mov      (sp)+,r1

```

```

;1:
;   movb    (r1)+,ch
;   beq     lf
;   mov     $1,r0
;   sys     write; ch; 1
;   br      lb
;1:
;   rts     r5
;
;itabp: itab
;loginp: login
;   name
;   0
;
;itab:
;   itty37; ttyes; tty37
;   itn300; tnmes; tn300
;tab2741:i2741; m2741; f2741
;   0
;
;itty37:511; 511; 340 / any parity, raw, 150 baud
;tty37: 511; 511; 210 / 37 parity, echo, 150 baud
;itn300:521; 521; 340 / any parity, raw, cr, 300 baud
;tn300: 521; 521; 310 / any parity, echo, 300 baud
;i2741:1501; 501; 100540 /134 bits, 2741, raw, first time
;f2741:1501; 501; 500 /134 bps, 2741
;
;   0
;m2741:<\nlogin: \0>
;
;ttyes:
;   <\n\r\p:\alogin: \0>
;tnmes:
;   <\n\r\p;login: \0>
;
;login: </bin/login\0>
;   .even
;
;nl:    <\n>
;cr:    <\r>
;
;flags: 004 / upper case map
;
;   .bss
;ch:    .+.2
;nowr:  .+.2
;name:  .+.32.

        end START_CODE

```