

```

; *****
;
; UNIX.ASM (RETRO UNIX 8086 Kernel - Only for 1.44 MB floppy disks)
; -----
; U7.ASM (include u7.asm) //// UNIX v1 -> u7.s

; 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
; (11/03/2013)
;
; [ Last Modification: 13/07/2014 ] ;;; completed ;;;
;
; 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>
;
; *****

; 13/07/2014 ottyp
; 12/07/2014 ottyp
; 15/04/2014 ottyp
; 26/01/2014 otty, ottyp, cttty, cttty
; 17/01/2014 otty, ottyp, ottys, cttty, cttty
; 13/01/2014 otty, ocvty, ottys, cttty, ccvt, ottyp, cttty
; 12/01/2014 iclose
; 06/12/2013 otty, ocvty, cttty, ccvt (major modification: p.ttyp, u.ttyp)
; 04/12/2013 (getc, putc procedures have been moved to U9.ASM)
; 03/12/2013 putc (write_tty, beep, waitf)
; 30/11/2013 putc
; 04/11/2013 putc, sysmount, sysumount
; 30/10/2013 putc
; 20/10/2013 getc
; 10/10/2013 getc
; 05/10/2013 getc
; 24/09/2013 getc, otty, ocvty, cttty, ccvt, putc (consistency check)
; 20/09/2013 putc, getc
; 17/09/2013 otty (ottys), cttty, ccvt
; 16/09/2013 ocvty, cttty
; 13/09/2013 otty
; 03/09/2013 otty, ocvty, cttty, ccvt
; 27/08/2013 iopen, iclose, ocvty, ccvt
; 26/08/2013 putc
; 16/08/2013 iopen, iclose, otty, cttty
; 13/08/2013 cttty (cttys)
; 05/08/2013 cttty
; 30/07/2013 iclose, cttty, ccvt
; 29/07/2013
; 28/07/2013
; 16/07/2013 iopen, otty, ocvty, cttty, ccvt, getc, iclose modifications
; 15/07/2013
; 09/07/2013 - sysmount, sysumount

sysmount: ; / mount file system; args special; name
; 04/11/2013
; 09/07/2013
; 'sysmount' announces to the system that a removable
; file system has been mounted on a special file.
; The device number of the special file is obtained via
; a call to 'getspl'. It is put in the I/O queue entry for
; dismountable file system (sbl) and the I/O queue entry is
; set up to read (bit 10 is set). 'ppoke' is then called to
; to read file system into core, i.e. the first block on the
; mountable file system is read in. This block is super block
; for the file system. This call is super user restricted.
;
; Calling sequence:
;     sysmount; special; name
; Arguments:
;     special - pointer to name of special file (device)
;     name - pointer to name of the root directory of the
;           newly mounted file system. 'name' should
;           always be a directory.
; Inputs: -
; Outputs: -
; .....
;

```

```

; Retro UNIX 8086 v1 modification:
;   'sysmount' system call has two arguments; so,
;   Retro UNIX 8086 v1 argument transfer method 2 is used
;   to get sysmount system call arguments from the user;
;   * 1st argument, special is pointed to by BX register
;   * 2nd argument, name is in CX register
;
;   NOTE1: Retro UNIX 8086 v1 'arg2' routine gets these
;   arguments which were in these registers;
;   but, it returns by putting the 1st argument
;   in 'u.namep' and the 2nd argument
;   on top of stack. (1st argument is offset of the
;   file/path name in the user's program segment.
;   NOTE2: Device numbers, names and related procedures are
;   already modified for IBM PC compatibility and
;   Retro UNIX 8086 v1 device configuration.
;call    arg2
;   jsr r0,arg2 / get arguments special and name
mov     word ptr [u.namep], bx
push    cx
cmp     word ptr [mnti], 0
;   tst mnti / is the i-number of the cross device file
;   ; / zero?
ja      error
;   ; bne errora / no, error
call    getspl
;   jsr r0,getspl / get special files device number in r1
; 04/11/2013
;pop     cx ; file name pointer
mov     bx, ax ; ; Retro UNIX 8086 v1 device number (0 to 5)
cmp     byte ptr [BX]+drv.err, 0
ja      error
;mov     word ptr [u.namep], cx
pop     word ptr [u.namep]
;   mov (sp)+,u.namep / put the name of file to be placed
;   ; / on the device
push    ax ; push bx
;   mov r1,-(sp) / save the device number
;
call    namei
;or      ax, ax ; Retro UNIX 8086 v1 modification !
;   ; ax = 0 -> file not found
;jz      error
jc      error
;   jsr r0,namei / get the i-number of the file
;   ; br errora
mov     word ptr [mnti], ax
;   mov r1,mnti / put it in mnti
; 04/11/2013
mov     bx, offset sb1 ; super block buffer (of mounted disk)
@@: ;1:
cmp     byte ptr [BX]+1, 0
;   tstb sb1+1 / is 15th bit of I/O queue entry for
;   ; / dismountable device set?
jna     short @f
;   ; bne 1b / (inhibit bit) yes, skip writing
call    idle ; 04/11/2013 (wait for hardware interrupt)
jmp     short @b
@@:
pop     ax ; Retro UNIX 8086 v1 device number/ID (0 to 5)
mov     byte ptr [mdev], al
;   mov (sp),mntd / no, put the device number in mntd
; 04/11/2013
mov     byte ptr [BX], al
;   movb (sp),sb1 / put the device number in the lower byte
;   ; / of the I/O queue entry
;mov     byte ptr [cdev], 1 ; mounted device/drive
;   mov (sp)+,cdev / put device number in cdev
or      word ptr [BX], 400h ; Bit 10, 'read' flag/bit
;   bis $2000,sb1 / set the read bit
mov     byte ptr [BX]+2, 1 ; physical block number = 1
call    diskio
jnc     short @f
xor     ax, ax
mov     word ptr [mnti], ax ; 0
mov     byte ptr [mdev], al ; 0
;mov     byte ptr [cdev], al ; 0
mov     word ptr [BX], ax ; 0
jmp     error

```

```

@@:
    mov     byte ptr [BX]+1, 0 ; 18/07/2013
    ;;call  ppoke
    ; jsr r0,ppoke / read in entire file system
;@@: ;1:
    ;;cmp   byte ptr [sbl]+1, 0
    ;      ; tstb  sbl+1 / done reading?
    ;;jna   sysret
    ;,call  idle ; 04/11/2013 (wait for hardware interrupt)
    ;;jmp   short @b
    ;      ;bne lb / no, wait
    ;      ;br sysreta / yes
    jmp     sysret

sysumount: ; / special dismount file system
    ; 04/11/2013
    ; 09/07/2013
    ; 'sysmount' announces to the system that the special file,
    ; indicated as an argument is no longer contain a removable
    ; file system. 'getspl' gets the device number of the special
    ; file. If no file system was mounted on that device an error
    ; occurs. 'mntd' and 'mnti' are cleared and control is passed
    ; to 'sysret'.
    ;
    ; Calling sequence:
    ;     sysmount; special
    ; Arguments:
    ;     special - special file to dismount (device)
    ;
    ; Inputs: -
    ; Outputs: -
    ; .....
    ;
    ; Retro UNIX 8086 v1 modification:
    ;     'sysumount' system call has one argument; so,
    ;     Retro UNIX 8086 v1 argument transfer method 1 is used
    ;     to get sysmount system call argument from the user;
    ;     * Single argument, special is pointed to by BX register
    ;
    ;mov     ax, 1 ; one/single argument, put argument in BX
    ;call    arg
    ;      ; jsr r0,arg; u.namep / point u.namep to special
    mov     word ptr [u.namep], bx
    call    getspl
    ;      ; jsr r0,getspl / get the device number in r1
    cmp     al, byte ptr [mdev]
    ;      ; cmp r1,mntd / is it equal to the last device mounted?
    jne     error
    ;      ; bne errora / no error
    xor     al, al ; ah = 0
@@: ;1:
    cmp     byte ptr [sbl]+1, al ; 0
    ;      ; tstb sbl+1 / yes, is the device still doing I/O
    ;      ; / (inhibit bit set)?
    jna     short @f
    ;      ; bne lb / yes, wait
    call    idle ; 04/11/2013 (wait for hardware interrupt)
    jmp     short @b
@@:
    mov     byte ptr [mdev], al
    ;      ; clr mntd / no, clear these
    mov     word ptr [mnti], ax
    ;      ; clr mnti
    jmp     sysret
    ;      ; br sysreta / return

getspl: ; / get device number from a special file name
    ; 09/07/2013
    call    namei
    ;or      ax, ax ; Retro UNIX 8086 v1 modification !
    ;      ; ax = 0 -> file not found
    ;jz      error
    jc      error
    ;      ; jsr r0,namei / get the i-number of the special file
    ;      ; br errora / no such file
    sub     ax, 3 ; Retro UNIX 8086 v1 modification !
    ;      ; i-number-3, 0 = fd0, 5 = hd3
    ;      ; sub $4,r1 / i-number-4 rk=1,tap=2+n
    jc      error

```

```

        ; ble errora / less than 0? yes, error
    cmp    ax, 5 ;
        ; cmp r1,$9. / greater than 9 tap 7
    ja     error
        ; bgt errora / yes, error
        ; AX = Retro UNIX 8086 v1 Device Number (0 to 5)
@@:
    retn

        ; rts    r0 / return with device number in r1

iopen:
    ;27/08/2013
    ;16/08/2013
    ;16/07/2013
    ;21/05/2013
    ;
    ; open file whose i-number is in r1
    ;
    ; INPUTS ->
    ;     r1 - inode number
    ; OUTPUTS ->
    ;     file's inode in core
    ;     r1 - inode number (positive)
    ;
    ; ((AX = R1))
    ;     ((Modified registers: DX, BX, CX, SI, DI, BP))
    ;
    ; / open file whose i-number is in r1
    test    ah, 80h ; Bit 15 of AX
    ;tst r1 / write or read access?
    jnz     short iopen_2
    ;blt 2f / write, go to 2f
    mov     dl, 2 ; read access
    call    access
    ; jsr r0,access; 2
    ; / get inode into core with read access

    ; DL=2
iopen_0:
    cmp     ax, 40
    ; cmp r1,$40. / is it a special file
    ;ja     short @f
    ;bgt 3f / no. 3f
    ja      short @b ; 16/08/2013
    push    ax
    ; mov r1,-(sp) / yes, figure out
    mov     bx, ax
    shl     bx, 1
    ; asl r1
    add     bx, offset iopen_1 - 2
    jmp     word ptr [BX]
    ; jmp *1f-2(r1) / which one and transfer to it
iopen_1: ; 1:
    dw      offset otty ; tty, AX = 1 (runix)
    ;otty / tty ; r1=2
    ;oppt / ppt ; r1=4
    dw      offset sret ; mem, AX = 2 (runix)
    ;sret / mem ; r1=6
    ;sret / rf0
    ;sret / rk0
    ;sret / tap0
    ;sret / tap1
    ;sret / tap2
    ;sret / tap3
    ;sret / tap4
    ;sret / tap5
    ;sret / tap6
    ;sret / tap7
    dw      offset sret ; fd0, AX = 3 (runix only)
    dw      offset sret ; fd1, AX = 4 (runix only)
    dw      offset sret ; hd0, AX = 5 (runix only)
    dw      offset sret ; hd1, AX = 6 (runix only)
    dw      offset sret ; hd2, AX = 7 (runix only)
    dw      offset sret ; hd3, AX = 8 (runix only)
    ;dw      offset error ; lpr, AX = 9 (error !)
    dw      offset sret ; lpr, AX = 9 (runix)
    dw      offset ocvt ; tty0, AX = 10 (runix)
    ;ocvt / tty0
    dw      offset ocvt ; tty1, AX = 11 (runix)
    ;ocvt / tty1

```

```

dw      offset ocvt ; tty2, AX = 12 (runix)
        ;ocvt / tty2
dw      offset ocvt ; tty3, AX = 13 (runix)
        ;ocvt / tty3
dw      offset ocvt ; tty4, AX = 14 (runix)
        ;ocvt / tty4
dw      offset ocvt ; tty5, AX = 15 (runix)
        ;ocvt / tty5
dw      offset ocvt ; tty6, AX = 16 (runix)
        ;ocvt / tty6
dw      offset ocvt ; tty7, AX = 17 (runix)
        ;ocvt / tty7
dw      offset ocvt ; COM1, AX = 18 (runix only)
        ;error / crd
dw      offset ocvt ; COM2, AX = 19 (runix only)
;@@:
        ;retn

iopen_2: ; 2: / check open write access
        neg     ax
        ;neg r1 / make inode number positive
        mov     dl, 1 ; write access
        call    access
        ;jsr r0,access; 1 / get inode in core

        ; DL=1
        test    word ptr [i.flgs], 4000h ; Bit 14 : Directory flag
        ;bit $40000,i.flgs / is it a directory?
        jnz     error
        ; bne 2f / yes, transfer (error)
        jmp     short iopen_0
;cmp     ax, 40
        ; cmp r1,$40. / no, is it a special file?
        ;ja     short @b
        ;bgt 3f / no, return
;push     ax
        ;mov r1,-(sp) / yes
;mov     bx, ax
;shl     bx, 1
        ; asl r1
;add     bx, offset ipen_3 - 2
;jmp     word ptr [BX]
        ; jmp *1f-2(r1) / figure out
        ; / which special file it is and transfer

;iopen_3: ; 1:
;        dw      offset otty ; tty, AX = 1 (runix)
        ;otty / tty ; r1=2
        ;leadr / ppt ; r1=4
;        dw      offset sret ; mem, AX = 2 (runix)
        ;sret / mem ; r1=6
        ;sret / rf0
        ;sret / rk0
        ;sret / tap0
        ;sret / tap1
        ;sret / tap2
        ;sret / tap3
        ;sret / tap4
        ;sret / tap5
        ;sret / tap6
        ;sret / tap7
;        dw      offset sret ; fd0, AX = 3 (runix only)
;        dw      offset sret ; fd1, AX = 4 (runix only)
;        dw      offset sret ; hd0, AX = 5 (runix only)
;        dw      offset sret ; hd1, AX = 6 (runix only)
;        dw      offset sret ; hd2, AX = 7 (runix only)
;        dw      offset sret ; hd3, AX = 8 (runix only)
;        dw      offset sret ; lpr, AX = 9 (runix)
;dw      offset ejec ; lpr, AX = 9 (runix)
;        dw      offset sret ; tty0, AX = 10 (runix)
        ;ocvt / tty0
;        dw      offset sret ; tty1, AX = 11 (runix)
        ;ocvt / tty1
;        dw      offset sret ; tty2, AX = 12 (runix)
        ;ocvt / tty2
;        dw      offset sret ; tty3, AX = 13 (runix)
        ;ocvt / tty3
;        dw      offset sret ; tty4, AX = 14 (runix)
        ;ocvt / tty4
;        dw      offset sret ; tty5, AX = 15 (runix)
        ;ocvt / tty5

```

```

;      dw      offset sret ; tty6, AX = 16 (runix)
;           ;ocvt / tty6
;      dw      offset sret ; tty7, AX = 17 (runix)
;           ;ocvt / tty7
;      dw      offset ocvt ; COM1, AX = 18 (runix only)
;           ;// ejec / lpr
;      dw      offset ocvt ; COM2, AX = 19 (runix only)

otty: ;/ open console tty for reading or writing
; 13/07/2014
; 12/07/2014
; 15/04/2014 (modification for serial ports)
; 26/01/2014
; 17/01/2014
; 13/01/2014
; 06/12/2013 (major modification: p.ttyc, u.ttyp)
; 24/09/2013 consistency check -> ok
; 17/09/2013
; 16/09/2013
; 13/09/2013
; 03/09/2013
; 16/08/2013
; 16/07/2013
; 15/07/2013
; 27/05/2013
; 21/05/2013
; Retro UNIX 8086 v1 modification !
;
; 16/07/2013
; Retro UNIX 8086 v1 modification:
; If a tty is open for read or write by
;   a process (u.uno), only same process can open
;   same tty to write or read (R->R&W or W->W&R).
;
; (INPUT: DL=2 for Read, DL=1 for Write, DL=0 for sysstty)
; ah = 0
; 06/12/2013
mov     bl, byte ptr [u.uno] ; process number
xor     bh, bh
mov     al, byte ptr [BX]+p.ttyc-1 ; current/console tty
; 13/01/2014
jmp     short ottyp

ocvt:
sub     al, 10

ottyp:
; 13/07/2014
; 12/07/2014
; 15/04/2014 (modification for serial ports)
; 26/01/2014
; 13/01/2014
; 06/12/2013
mov     dh, al ; tty number
; 16/08/2013
mov     bx, ax ; AL = tty number (0 to 9), AH = 0
shl     bl, 1 ; aligned to word
;26/01/2014
add     bx, offset ttyl
mov     cx, word ptr [BX]
; CL = lock value (0 or process number)
; CH = open count
and     cl, cl
; 13/01/2014
jz      short otty_ret
;
cmp     cl, byte ptr [u.uno]
je      short otty_ret
;
mov     bl, cl ; the process which has locked the tty
shl     bl, 1
xor     bh, bh
mov     ax, word ptr [BX]+p.pid-2
mov     bl, byte ptr [u.uno]
shl     bl, 1
cmp     ax, word ptr [BX]+p.ppid-2
je      short otty_ret
;jne    short otty_err
; the tty is locked by another process
; except the parent process (p.ppid)

```

```

;;otty_err: ; 13/01/2014
        or     dl, dl ; DL = 0 -> called by sysstty
        jnz    error
        stc
        retn
otty_ret:
        ; 13/01/2014
        cmp    dh, 7
        jna    short ottys_ret
ottyys:
        ; 17/01/2013
        push   dx ; *
        mov    ah, dl ; open mode
        mov    dl, dh
        xor    dh, dh
        sub    dl, 8
        ;
        and    ah, ah ; sysstty system call check
        jz     short com_port_init
        ;
        and    cx, cx
        jz     short @f ; unlocked/free tty (serial port)
        ;
        ; 13/01/2014
        ; DX = port number (COM1=0, COM2=1)
        mov    ah, 3
        int    14h ; Get serial port status
        ; 13/07/2014
        pop    dx ; *
        test   ah, 80h
        jz     short ottys_rtn
;;otty_err: ; 13/01/2014
        or     dl, dl ; DL = 0 -> called by sysstty
        jnz    error
        stc
        retn
@@:
        xor    ah, ah ; 0
com_port_init:
        mov    si, offset comlp
        or     dl, dl ; COM1 ?
        jz     short @f ; yes, it is COM1
        inc    si ; no, it is COM2
@@:
        mov    al, byte ptr [SI] ; comm. parameters
        ;
        ; Initializing serial port parameters
        xor    ah, ah ; 0
        ; AL = Communication parameters
        ; DX = Serial port number (COM1 = 0, COM2 = 1)
        int    14h ; Initialize serial port parameters
        ;
        ; (Note: Serial port interrupts
        ;       will be disabled here...)
        ; (INT 14h initialization code
        ;       disables interrupts.)
        ; 13/07/2014
        and    dl, dl
        jz     short comlp_eirq
        ;
        ;; COM2 - enabling IRQ 3
        mov    dx, 2FCh ;modem control register
        in     al, dx ;read register
        or     al, 8 ;enable bit 3 (OUT2)
        out    dx, al ;write back to register
        mov    dx, 2F9h ;interrupt enable register
        in     al, dx ;read register
        or     al, 1 ;receiver data interrupt enable
        out    dx, al ;write back to register
        in     al, 21h ;read interrupt mask register
        and    al, 0F7h ;enable IRQ 3 (COM2)
        out    21h, al ;write back to register
        mov    dx, 1
        jmp    short comp_get_stat
comlp_eirq:
        ;; COM1 - enabling IRQ 4
        mov    dx, 3FCh ;modem control register
        in     al, dx ;read register
        or     al, 8 ;enable bit 3 (OUT2)

```

```

    out    dx, al    ;write back to register
    mov    dx, 3F9h  ;interrupt enable register
    in     al, dx    ;read register
    or     al, 1     ;receiver data interrupt enable
    out    dx, al    ;write back to register
    in     al, 21h   ;read interrupt mask register
    and    al, 0EFh  ;enable IRQ 4 (COM1)
    out    21h, al   ;write back to register
    xor    dx, dx
comp_get_stat:
    mov    ah, 3
    int    14h      ; Get serial port status
    ;
    test   ah, 80h
    jz     short comp_init_ok ; successfully initialized
    ; Initialization ERROR !
    ; 11100011b ; E3h
    ; (111) Baud rate: 9600, (00) parity: none,
    ; (0) stop bits: 1, (11) word length: 8 bits
    ; 15/04/2014
    cmp    byte ptr [SI], 0E3h
    je     short @f
    ;
    mov    byte ptr [SI], 0E3h ; Reset comm. parameters
    xor    ah, ah
    jmp    short @b
@@:
    ; 12/07/2014
    pop    dx ; *
    stc
    retn
comp_init_ok:
    ; 12/07/2014
    pop    dx ; *
ottys_ret:
    or     cl, cl    ; cl = lock/owner, ch = open count
    jnz    short @f
    mov    cl, byte ptr [u.uno]
ottys_rtn:
@@:
    inc    ch
    mov    word ptr [BX], cx ; set tty lock again
    ; 06/12/2013
    inc    dh ; tty number + 1
    mov    bx, offset u.ttyp
    ; 13/01/2014
    test   dl, 2 ; open for read sign
    jnz    short @f
    inc    bx
@@:
    ; Set 'u.ttyp' ('the recent TTY') value
    mov    byte ptr [BX], dh ; tty number + 1
sret:
    or     dl, dl ; sysstty system call check (DL=0)
    jz     short @f
    pop    ax
@@:
    retn
    ;
    ; Original UNIX v1 'otty' routine:
    ;
    ;mov    $100,*$tk$ / set interrupt enable bit (zero others) in
    ;                / reader status reg
    ;mov    $100,*$tps / set interrupt enable bit (zero others) in
    ;                / punch status reg
    ;mov    tty+[ntty*8]-8+6,r5 / r5 points to the header of the
    ;                / console tty buffer
    ;incb   (r5) / increment the count of processes that opened the
    ;                / console tty
    ;tst    u.ttyp / is there a process control tty (i.e., has a tty
    ;                / buffer header
    ;bne    sret / address been loaded into u.ttyp yet)? yes, branch
    ;mov    r5,u.ttyp / no, make the console tty the process control
    ;                / tty
    ;br     sret / ?
;sret:
    ;clr    *$ps / set processor priority to zero
    ;
    ; pop    ax
    ;mov    (sp)+,r1 / pop stack to r1

```



```

;3:
;      retn
;               ;rts r0

;ocvt: ; < open tty >
;      13/01/2014
;      06/12/2013 (major modification: p.ttyc, u.ttyp)
;      24/09/2013 consistency check -> ok
;      16/09/2013
;      03/09/2013
;      27/08/2013
;      16/08/2013
;      16/07/2013
;      27/05/2013
;      21/05/2013
;
;      Retro UNIX 8086 v1 modification !
;
;      In original UNIX v1, 'ocvt' routine
;      (exactly different than this one)
;      was in 'u9.s' file.
;
;      16/07/2013
;      Retro UNIX 8086 v1 modification:
;      If a tty is open for read or write by
;      a process (u.uno), only same process can open
;      same tty to write or read (R->R&W or W->W&R).
;
;      INPUT: DL=2 for Read DL=1 for Write

;      16/09/2013
;      sub    al, 10
;      06/12/2013
;      cmp     al, 7
;      jna     short ottyp
;      13/01/2014
;      jmp     short ottyp

;oppt: / open paper tape for reading or writing
;      mov     $100,$prs / set reader interrupt enable bit
;      tstb    pptiflg / is file already open
;      bne     2f / yes, branch
;1:
;      mov     $240,$ps / no, set processor priority to 5
;      jsr     r0,getc; 2 / remove all entries in clist
;      br      .+4 / for paper tape input and place in free list
;      br      1b
;      movb    $2,pptiflg / set pptiflg to indicate file just open
;      movb    $10.,toutt+1 / place 10 in paper tape input tout entry
;      br      sret
;2:
;      jmp     error / file already open

;close:
;      13/01/2014
;      12/01/2014
;      27/08/2013
;      16/08/2013
;      30/07/2013
;      16/07/2013
;      21/05/2013
;
;      close file whose i-number is in r1
;
;      INPUTS ->
;      r1 - inode number
;      OUTPUTS ->
;      file's inode in core
;      r1 - inode number (positive)
;
;      ; ((AX = R1))
;      ; ((Modified registers: -BX-, DX))
;/ close file whose i-number is in r1
mov     dl, 2 ; 12/01/2014
test     ah, 80h ; Bit 15 of AX
;      tst r1 / test i-number
;      jnz     short iclose_2
;      blt 2f / if neg., branch
jz       short iclose_0 ; 30/07/2013

```

```

; 16/07/2013
neg    ax ; make it positive
; 12/01/2014
dec    dl ; dl = 1 (open for write)
iclose_0:
    cmp    ax, 40
        ; cmp r1,$40. / is it a special file
    ja     short @b ; 13/01/2014
        ; bgt 3b / no, return
; 12/01/2014
; DL=2 -> special file was opened for reading
; DL=1 -> special file was opened for writing
push    ax
        ; mov r1,-(sp) / yes, save r1 on stack
mov     bx, ax
shl     bx, 1
        ; asl r1
add     bx, offset iclose_1 - 2
jmp     word ptr [BX]
        ; jmp *1f-2(r1) / compute jump address and transfer
iclose_1 :
dw      offset cttty ; tty, AX = 1 (runix)
dw      offset cret ; mem, AX = 2 (runix)
dw      offset cret ; fd0, AX = 3 (runix only)
dw      offset cret ; fd1, AX = 4 (runix only)
dw      offset cret ; hd0, AX = 5 (runix only)
dw      offset cret ; hd1, AX = 6 (runix only)
dw      offset cret ; hd2, AX = 7 (runix only)
dw      offset cret ; hd3, AX = 8 (runix only)
dw      offset cret ; lpr, AX = 9 (runix)
;dw      offset error; lpr, AX = 9 (error !)
;dw      offset ejec ; lpr, AX = 9
dw      offset ccvt ; tty0, AX = 10 (runix)
dw      offset ccvt ; tty1, AX = 11 (runix)
dw      offset ccvt ; tty2, AX = 12 (runix)
dw      offset ccvt ; tty3, AX = 13 (runix)
dw      offset ccvt ; tty4, AX = 14 (runix)
dw      offset ccvt ; tty5, AX = 15 (runix)
dw      offset ccvt ; tty6, AX = 16 (runix)
dw      offset ccvt ; tty7, AX = 17 (runix)
dw      offset ccvt ; COM1, AX = 18 (runix only)
dw      offset ccvt ; COM2, AX = 19 (runix only)

; 1:
;      cttty / tty
;      cppt / ppt
;      sret / mem
;      sret / rf0
;      sret / rk0
;      sret / tap0
;      sret / tap1
;      sret / tap2
;      sret / tap3
;      sret / tap4
;      sret / tap5
;      sret / tap6
;      sret / tap7
;      ccvt / tty0
;      ccvt / tty1
;      ccvt / tty2
;      ccvt / tty3
;      ccvt / tty4
;      ccvt / tty5
;      ccvt / tty6
;      ccvt / tty7
;      error / crd

; iclose_2: ; 2: / negative i-number
; neg    ax
; neg r1 / make it positive
; cmp    ax, 40
; cmp r1,$40. / is it a special file?
; ja     short @b
; bgt    3b / no. return
; push    ax
; mov r1,-(sp)
; mov     bx, ax
; shl     bx, 1
; asl r1 / yes. compute jump address and transfer

```

```

;add    bx, offset iclose_3 - 2
;jmp    word ptr [BX]
;jmp    *1f-2(r1) / figure out
;iclose_3:
;dw      offset cttty ; tty, AX = 1 (runix)
;dw      offset sret ; mem, AX = 2 (runix)
;dw      offset sret ; fd0, AX = 3 (runix only)
;dw      offset sret ; fd1, AX = 4 (runix only)
;dw      offset sret ; hd0, AX = 5 (runix only)
;dw      offset sret ; hd1, AX = 6 (runix only)
;dw      offset sret ; hd2, AX = 7 (runix only)
;dw      offset sret ; hd3, AX = 8 (runix only)
;dw      offset sret ; lpr, AX = 9
;dw      offset ejec ; lpr, AX = 9 (runix)
;dw      offset ccvt ; tty0, AX = 10 (runix)
;dw      offset ccvt ; tty1, AX = 11 (runix)
;dw      offset ccvt ; tty2, AX = 12 (runix)
;dw      offset ccvt ; tty3, AX = 13 (runix)
;dw      offset ccvt ; tty4, AX = 14 (runix)
;dw      offset ccvt ; tty5, AX = 15 (runix)
;dw      offset ccvt ; tty6, AX = 16 (runix)
;dw      offset ccvt ; tty7, AX = 17 (runix)
;dw      offset ccvt ; COM1, AX = 18 (runix only)
;dw      offset ccvt ; COM2, AX = 19 (runix only)

;1:
;      cttty / tty
;      leadr / ppt
;      sret / mem
;      sret / rf0
;      sret / rk0
;      sret / tap0
;      sret / tap1
;      sret / tap2
;      sret / tap3
;      sret / tap4
;      sret / tap5
;      sret / tap6
;      sret / tap7
;      ccvt / tty0
;      ccvt / tty1
;      ccvt / tty2
;      ccvt / tty3
;      ccvt / tty4
;      ccvt / tty5
;      ccvt / tty6
;      ccvt / tty7
;/      ejec / lpr

ctty: ; / close console tty
; 26/01/2014
; 17/01/2014
; 13/01/2014
; 06/12/2013 (major modification: p.ttyc, u.ttyp)
; 24/09/2013 consistency check -> OK
; 17/09/2013
; 16/09/2013
; 03/09/2013
; 16/08/2013
; 13/08/2013
; 05/08/2013
; 30/07/2013
; 16/07/2013
; 27/05/2013
; 21/05/2013
; Retro UNIX 8086 v1 modification !
;
; (DL = 2 -> it is open for reading)
; (DL = 1 -> it is open for writing)
; (DL = 0 -> it is open for sysstty system call)
;
; 06/12/2013
mov     bl, byte ptr [u.uno] ; process number
xor     bh, bh
mov     al, byte ptr [BX]+p.ttyc-1
; 13/01/2014
jmp     short ctttyp

ccvt:
sub     al, 10

```

```

cttyp:
; 26/01/2014
; 13/01/2014
; 24/09/2013 consistency check -> ok
; 16/08/2013
; AH = 0
mov     bx, ax ; tty number (0 to 9)
shl     bl, 1 ; aligned to word
; 26/01/2014
add     bx, offset ttyl
mov     dh, al ; tty number
mov     ax, word ptr [BX]
; AL = lock value (0 or process number)
; AH = open count
and     ah, ah
;jz     short ctty_err ; open count = 0, it is not open !
jz      error
; 26/01/2014
ctty_ret:
dec     ah ; decrease open count
jnz     short @f
xor     al, al ; unlock/free tty
@@:
mov     word ptr [BX], ax ; close tty instance
;
mov     bx, offset u.ttyp
test    dl, 1 ; open for write sign
jz      short @f
inc     bx
@@:
inc     dh ; tty number + 1
cmp     dh, byte ptr [BX]
jne     short cret
; Reset/Clear 'u.ttyp' ('the recent TTY') value
mov     byte ptr [BX], 0
cret:
or      dl, dl ; sysstty system call check (DL=0)
jz      short @f
pop     ax
@@:
retn

;ctty_err: ; 13/01/2014
; or      dl, dl ; DL = 0 -> called by sysstty
; jnz     error
; stc
; retn

; Original UNIX v1 'ctty' routine:
;
;mov     tty+[ntty*8]-8+6,r5
;          ;// point r5 to the console tty buffer
;decb    (r5) / dec number of processes using console tty
;br      sret / return via sret

;ccvt: ; < close tty >
; 13/01/2014
; 06/12/2013 (major modification: p.ttyc, u.ttyp)
; 24/09/2013 consistency check -> ok
; 17/09/2013
; 03/09/2013
; 27/08/2013
; 16/08/2013
; 30/07/2013
; 16/07/2013
; 27/05/2013
; 21/05/2013
;
; Retro UNIX 8086 v1 modification !
;
; In original UNIX v1, 'ccvt' routine
; (exactly different than this one)
; was in 'u9.s' file.
;
; DL = 2 -> it is open for reading
; DL = 1 -> it is open for writing
;

```

```
; 17/09/2013
;sub    al, 10
;cmp    al, 7
;jna     short cttyp
; 13/01/2014
;jmp     short cttyp

;cppt: / close paper tape
;        clrb    pptiflg / set pptiflg to indicate file not open
;l:
;        mov     $240,*$ps /set process or priority to 5
;        jsr     r0,getc; 2 / remove all ppt input entries from clist
;                               / and assign to free list
;        br      sret
;        br      lb

;eject:
;        jmp     error
;/eject:
;/        mov     $100,*$lps / set line printer interrupt enable bit
;/        mov     $14,r1 / 'form feed' character in r1 (new page).
;/        jsr     r0,lptoc / space the printer to a new page
;/        br      sret / return to caller via 'sret'
```