So programmiert man einen Floppy-Controller

Es gibt unter CP/M für den Zugriff auf Disketten verschiedene Programmiertechniken. Der einfachste Weg führt über eine höhere Programmiersprache, der schnellste und komplizierteste über die direkte Programmierung des Floppy-Controllers.

Ein großer Teil des CP/M-Betriebssystems befaßt sich ausschließlich mit der Steuerung und Kontrolle der angeschlossenen Diskettenstationen. Die hardwareabhängigen Teile des Betriebssystems sind im BIOS (Basic Input Output System) abgelegt und werden vom BDOS verarbeitet. Man kann aber auch noch tiefer gehen und den Floppy-Controller direkt ansteuern. Das ist zwar mit einer Menge Arbeit verbunden und eine genaue Kenntnis der Hardware des Computers ist Voraussetzung. Das Programm »MON.COM« wurde auf einem ITT 3030 entwickelt, der mit einem SY 1791 Floppy-Controller ausgerüstet ist. Man muß den Assembler-Source-Code mit einem Editor (ED.COM oder Wordstar) erfassen. Anschließend kann das Source-File mit ASM.COM (gehört zum Lieferumfang von CP/M) assembliert und mit LOAD.COM gelinkt werden. MON.COM ist ein Full-Screen-Disk-Editor mit Cursor-Steuerung und Hardcopy-Funktion. Eine Übertragung auf andere CP/M-Systeme ist nur bei der Verwendung des gleichen Floppy-Controllers möglich. Die Portadressen des Floppy-Controllers sind dann entsprechend zu modifizieren. ITT 3030-Besitzer, denen das Eintippen zu mühsam ist, können sich an den Autor wenden (Tel. 08161/83941). (G. Biebl/bo)

```
org     100h
jmp start
bdos:   equ 05                  ;bdos entry point
conin:  equ 01                  ;console input function
conout: equ 02                  ;console output function
lstout: equ 05                  ;lst output function
print:  equ 09                  ;print string function
cr:     equ 0dh                 ;carriage return character
lf:     equ 0ah                 ;line feed character
ff:     equ 0ch                 ;form feed character
esc:    equ 1bh                 ;escape character
home:   db esc,11h,'$'          ;home cursor function
left:   equ 08                  ;cursor left
right:  db esc,1ah,'$'          ;cursor right
up:     db esc,1ch,'$'          ;cursor up
bell:   equ 07                  ;sound output
invon:  db esc,28h,'$'          ;inverse screen on
invoff: db esc,18h,'$'          ;inverse screen off
pos:    db esc,16h              ;position cursor
line    ds 1
col     ds 1
delim:  db '$'
trackhi ds 1                    ;track counter, hi
tracklo ds 1                    ;track counter, low
trackbyte ds 1                  ;track counter, hex
secthi  ds 1                    ;sector counter, hi
sectlo  ds 1                    ;sector counter, low
sectbyte ds 1                   ;sector counter, hex
side:   ds 1                    ;side flag
selbyte: ds 1
art:    ds 1
drivebyte: ds 1
bytehi: ds 1
bytelo: ds 1
bytehex: ds 1
rescol: ds 1
resline: ds 1
mess0:  db esc,16h,24,01,esc,18h,'$'
mess1:  db esc,11h,' Laufwerk: $'
mess2:  db esc,16h,1,24,' Spur: $'
mess3:  db esc,16h,1,33,' Sektor: $'
mess4:  db esc,16h,24,01,bell,' Laufwerk schreibbereit !    $'
mess5:  db esc,16h,24,01,bell,' Laufwerk lesebereit !       $'
mess6:  db esc,16h,24,01,bell,' Schreibfehler aufgetreten ! $'
mess7:  db esc,16h,24,01,bell,' Lesefehler aufgetreten !    $'
mess8:  db ff,esc,16h,12,20,'*** Disk Editor CP/M 2.2 *** $'
mess9:  db esc,16h,24,01,bell,' Spur nicht gefunden !       $'
messa:  db esc,16h,24,01,bell,' Sektor nicht gefunden !     $'
messb:  db esc,16h,24,01,bell,' Laufwerk nicht bereit !     $'
messc:  db esc,16h,24,01,bell,' Laufwerk schreibgeschützt ! $'
messd:  db esc,16h,1,14,' Seite: $'
cardcom: equ 54h                ;fdc card control register
cardstat: equ cardcom           ;fdc card status register
fdccom: equ 58h                 ;fdc comand register
fdcstat: equ fdccom             ;fdc status register
datareg: equ 53h                ;fdc data register
trackreg: equ 51h               ;fdc track register
sectreg: equ 52h                ;fdc sector register
restore: equ 00001100b          ;restore comand fdc
busy:   equ 00000001b           ;drive busy status
select: equ 00011000b           ;motor on, door lock
load:   equ 00001000b           ;head load comand
hld:    equ 00100000b           ;head loaded status
seek:   equ 00011100b           ;seek track comand
seekerr: equ 00010000b          ;seek error status
drq:    equ 00000010b           ;data requested status
datalost: equ 00000100b         ;data lost status
readsec: equ 10000110b          ;read sector comand
readse1: equ 10001110b          ;read sector, side = 1
wrsec1: equ 10100110b           ;write sector side=0
wrsec2: equ 10101110b           ;write sector, side=1
intreq: equ 01000000b           ;interrupt requested
ready:  equ 00011100b           ;drive 1-3 ready
writeprot: equ 00000010b        ;disk write protected
retry:  equ 76                  ;# of retrys
secbuf: ds 100h                 ;buffer for sector read
xlim:   db '$'


        ;
start:  lxi d,mess0
        call prtmess
        call get
        mvi e,ff
        call put
        mvi a,0
        sta art
screen: lxi d,mess1
        call prtmess
        call get
        cpi esc
        jz ausgang
        call capitals
        jc screen
        sui 40h
        sta drivebyte
        rrc
        rrc
        xri 0c0h
        ori select
        sta selbyte
sidesel: call askside
        jc screen
        lda side
        cpi 0
        jz side0
        lda selbyte
        ori 00000100b
        jmp sideab
side0:  lda selbyte
sideab: out cardcom
        mvi b,busy
selloop: in fdcstat
        ana b
        jnz selloop
        in cardstat
        mov c,a
        ani ready
        cz drivefault
        jc screen
        call track0
        lxi d,mess0
        call prtmess
        ana a
        mov a,c
        ani writeprot
        cnz protect
ask:    call track0
        call asktrack
        jc sidesel
        lxi h,trackhi
        call dezhex
        call posit
        cc trackmeld
```

Listing zum Programm »MON.ASM«

```
retour:     call asksect
            jc ask
            lxi h,secthi
            call dezhex
            lxi d,mess5
            call prtmess
            call get
            cpi esc
            jz retour
            call lesen
            call buffread
            call edit
            jmp screen

asktrack:   lxi d,mess2
            call prtmess
tr2left:    ana a
            call get
            cpi esc
            jz escask
track:      cpi 30h
            jz trackok
            jnc trackok
            jmp asktrack
trackok:    cpi 37h
            jnc asktrack
            sta trackhi
track2:     call get
            cpi left
            jz tr2left
            cpi 30h
            jz wordok
            jnc wordok
            mvi e,left
            call put
            jmp track2
wordok:     mov e,a
            lxi h,trackhi
            mov a,m
            mov a,e
            cpi 39h
trtest:     jz allok
            jc allok
            mvi e,left
            call put
            jmp track2
allok:      sta tracklo
            ana a
            ret
escask:     stc
            ret
;
ausgang:    mvi e,ff
            call put
            mvi c,0
            call bdos
;
prtmess:    push psw
            push b
            push d
            push h
            mvi c,print
            call bdos
            pop h
            pop d
            pop b
            pop psw
            ret
;
get:        push b
            push d
            push h
            mvi c,conin
            call bdos
            pop h
            pop d
            pop b
            ret
;
put:        push psw
            push b
            push d
            push h
            mvi c,conout
            call bdos
            pop h
            pop d
            pop b
            pop psw
            ret
;
capitals:   ana a
            cpi 'A'
            jz cap
            jnc cap
false:      stc
            jmp cap2
cap:        cpi 'C'
            jnc false
            ana a
cap2:       ret
;
dezhex:     mov a,m
            ani 00001111b
            mov m,a
            inx h
            mov a,m
            ani 00001111b
            mov m,a
            dcx h
            mov a,m
            add a
            mov b,a
            add a
            add a
            add b
            inx h
            add m
            inx h
            mov m,a
            ret
;
posit:      lxi h,trackbyte
            mvi e,retry
            mov a,m
            mov c,a
            out datareg
            mvi b,busy
trackseek:  mvi a,seek
            dcr e
            jz trackfehl
            out fdccom
testpos:    in fdcstat
            mov c,a
            ana b
            jz testpos
            mov a,c
            cpi seekerr
            jz trackseek
            ana a
trackret:   ret
trackfehl:  stc
            jmp trackret
;
trackmeld:  lxi d,mess9
            call prtmess
            ret
;
drivefault: lxi d,messb
            call prtmess
            stc
            ret
;
protect:    lxi d,messc
            call prtmess
            stc
            ret
;
asksect:    lxi d,mess3
            call prtmess
se2left:    ana a
            call get
            cpi esc
            jz escsect
sectcheck:  cpi 30h
            jz sectok
            jnc sectok
            jmp asksect
sectok:     cpi 32h
            jnc asksect
            sta secthi
sect2:      call get
            cpi left
            jz se2left
            cpi 30h
            jz byteok
            jnc byteok
            mvi e,left
            call put
            jmp sect2
byteok:     mov e,a
            lxi h,secthi
            mov a,m
            cpi 31h
            jz letztsec
            mov a,e
            cpi 3ah
sctest:     jc sectab
            mvi e,left
            call put
            jmp sect2
letztsec:   mov a,e
            cpi 37h
            jmp sctest
sectab:     sta sectlo
            ana a
            ret
escsect:    stc
            ret
;
prthex:     push psw
            rar
            rar
            rar
            rar
            call prtnbl
            pop psw
prtnbl:     ani 00001111b
            ori 30h
            cpi 3ah
            jc sml
            adi 07
sml:        mov e,a
            lda art
            cpi 1
            jnz prx
            call lst
            jmp prend
prx:        call put
prend:      ret
;
lesen:      lxi h,sectbyte
            mov a,m
            lxi h,secbuf
            mvi b,drq
            mvi c,busy
            mvi d,seekerr
            out sectreg
            lda side
            cpi 0
            jz readc
            mvi a,readse1
            jmp readout
readc:      mvi a,readsec
readout:    out fdccom
loop1:      in fdcstat
            mov e,a
            ana b
            jnz read
            mov a,e
            ana d
            jnz record
            jmp loop1
check:      in fdcstat
            mov e,a
            ana c
            jz fertig
            mov a,e
            ana b
            jz check
read:       in datareg
            mov m,a
            inx h
            jmp check
fertig:     ret
;
record:     in cardstat
            cpi intreq
            jz nochmal
            lxi d,messa
            call prtmess
            jmp fertig
;
nochmal:    in fdcstat
            ana b
            jnz nochmal
            jmp lesen
;
buffread:   lxi h,secbuf
            mvi b,00h
            mvi a,3dh
            sta line
carriage:   mvi a,01h
            sta col
buffloop:   lxi d,pos
            call prtmess
            mov a,m
            dcr b
            jz buffend
            call prthex
            inx h
            lda col
            adi 03
            sta col
            cpi 31h
            jz umbruch
            jc buffloop
umbruch:    mvi e,20h
            call put
            call ascii
            lda line
            inr a
store:      sta line
            jmp carriage
buffend:    mov a,m
            call prthex
            inx h
            mvi e,20h
            call put
            call ascii
            ret
;
ascii:      push b
            push h
            push d
            mvi a,16
ascan:      dcx h
            dcr a
            jnz ascan
            mvi c,0
asciiread:  mov a,m
            cpi ' '
            jc weiter
            cpi 07FH
            jnc weiter
            mov e,a
            lda art
            cpi 1
            jnz schirm
            call lst
            jmp next
schirm:     call put
            jmp next
weiter:     mvi e,'.'
            lda art
            cpi 1
            jnz schirm2
            call lst
            jmp next
schirm2:    call put
next:       inx h
            inr c
            ana a
            mov a,c
            cpi 16
            jz asciiend
            jmp asciiread
asciiend:   pop d
            pop h
            pop b
            ret
;
track0:     mvi a,restore
            out fdccom
tr0loop:    in trackreg
            cpi 0
            jnz track0
            ret
;
askside:      lxi d,messd
              call prtmess
              ana a
              call get
              cpi esc
              jz escside
seite:        cpi 30h
              jz sideok
              jnc sideok
              jmp askside
sideok:       cpi 32h
              jnc askside
              ani 00001111b
              sta side
              ana a
              ret
escside:      stc
              ret
;
hardcopy:     mvi a,1
              sta art
              lxi h,messi+2
              call string
              lda drivebyte
              adi 40h
              mov e,a
              call lst
              mvi e,20h
              call lst
              lxi h,mess2+4
              call string
              lda trackbyte
              call prthex
              mvi e,20h
              call lst
              lxi h,mess3+4
              call string
              lda sectbyte
              call prthex
              mvi e,cr
              call lst
              mvi e,lf
              call lst
              call lst
              lxi h,secbuf
              mvi e,cr
              call lst
              mvi c,0
              mvi b,0
hardnext:     mov a,m
              call prthex
              inx h
              inr c
              ana a
              mov a,c
              cpi 16
              jz hardasc
              mvi e,20h
              call lst
              jmp hardnext
hardasc:      mvi e,20h
              call lst
              mvi c,0
              call ascii
              mvi e,cr
              call lst
              mvi e,lf
              call lst
              inr b
              ana a
              mov a,b
              cpi 16
              jnz hardnext
```

Listing zum Programm »MON.ASM« (Fortsetzung)

```
                mvi e,cr
                call lst
                mvi e,lf
                call lst
                call lst
                call lst
                mvi a,0
                sta art
hardend:        ret
;
lst:            push h
                push d
                push b
                mvi c,lstout
                call bdos
                pop b
                pop d
                pop h
                ret
;
string:         mov a,m
                cpi '$'
                jz strend
                mov e,a
                call lst
                inx h
                jmp string
strend:         ret
;
edit:           mvi a,3dh
                sta line
                mvi a,01
                sta col
highloop:       lxi d,pos
                call prtmess
                call get
                cpi esc
                jz highret
                cpi 10h
                jnz skipprt
                cz hardcopy
                jmp highloop
skipprt:        cpi left
                jz zurueck
                cpi 13h
                jz zurueck
                cpi 15h
                jz chome
                cpi 0ch
                jnz skip1
                lxi d,mess5
                call prtmess
                call get
                cpi esc
                jz highloop
                call lesen
                call buffread
                mvi a,3dh
                sta line
                mvi a,01
                sta col
                jmp highloop
skip1:          cpi 5
                jz lineup
                cpi 17h
                jnz skipa
                lxi d,mess4
                call prtmess
                call get
                cpi esc
                jz highloop
                call wrsec
                jmp highloop
skipa:          cpi 0eh
                jnz skip4
                call buffread
                jmp highloop
skip4:          cpi 18h
                jz linedown
                cpi 4
                jz rechts
                cpi 30h
                jc highloop
                cpi 3ah
                jc highok
                cpi 'A'
                jc highloop
                cpi 'F'
                jc highok
                jz highok
                jmp highloop
;
zurueck:        lda col
                cpi 1
                jz hoch
                sui 3
                sta col
                jmp highloop
;
rechts:         lda col
                cpi 2eh
                jz linenext
                adi 3
                sta col
                jmp highloop
;
chome:          mvi a,3dh
                sta line
                mvi a,01h
                sta col
                jmp highloop
;
lineup:         lda line
                cpi 3dh
                jz highloop
                dcr a
                sta line
                jmp highloop
;
linedown:       lda line
                cpi 4eh
                jz wraparound
                inr a
                sta line
                jmp highloop
;
back:           lda col
                dcr a
                sta col
                jmp highloop
;
hoch:           lda line
                cpi 3dh
                jz highloop
                dcr a
                sta line
                mvi a,2eh
                sta col
                jmp highloop
;
highok:         sta bytehi
                lda col
                inr a
                sta col
lowloop:        lxi d,pos
                call prtmess
                call get
                cpi esc
                jz highret
                cpi left
                jz back
                cpi 13h
                jz back
                cpi 15h
                jz chome
                cpi 0ch
                jnz skip2
                lxi d,mess5
                call prtmess
                call get
                cpi esc
                jz lowloop
                call lesen
                call buffread
                mvi a,3dh
                sta line
                mvi a,01
                sta col
                jmp lowloop
skip2:          cpi 5
                jz lineup
                cpi 17h
                jnz skipb
                lxi d,mess4
                call prtmess
                call put
                cpi esc
                jz lowloop
                call wrsec
                jmp lowloop
skipb:          cpi 0eh
                jnz skip3
                call buffread
                jmp lowloop
skip3:          cpi 18h
                jz linedown
                cpi 4
                jz rechts
                cpi 30h
                jc lowloop
                cpi 3ah
                jc lowok
                cpi 10h
                jnz lowskip
                call hardcopy
                jmp lowloop
lowskip:        cpi 30h
                jc lowloop
                cpi 3ah
                jc lowok
                cpi 'A'
                jc lowloop
                cpi 'F'
                jc lowok
                jz lowok
                jmp lowloop
lowok:          sta bytelo
                lda col
                adi 2
                sta col
                cpi 31h
                cz screenbuf
                jz linenext
                jnc linenext
                call screenbuf
                jmp highloop
linenext:       mvi a,01
                sta col
                lda line
                inr a
                cpi 4dh
                jz wraparound
                sta line
                call screenbuf
                call newline
                jmp highloop
highret:        ret
;
wraparound:     mvi a,3dh
                sta line
                call screenbuf
                call newline
                jmp highloop
;
screenbuf:      lxi h,secbuf
                lda line
                sui 3dh
                jz colad
                add a
                add a
                add a
                add a
colad:          add l
                mov l,a
                mvi a,0
                adc h
                mov h,a
                mvi c,0
                lda col
                sui 1
                jz coljmp
                mov b,a
                mvi a,3
comp:           cmp b
                jz coljmp
                inr c
                adi 3
                jmp comp
coljmp:         mov a,c
                add l
                mov l,a
                mvi a,0
                adc h
                mov h,a
                lda bytelo
                cpi 41h
                jnc alpha
                sui 30h
                jmp lostore
alpha:          ani 00001111b
                adi 9
lostore:        sta bytelo
                lda bytehi
                cpi 41h
                jnc alphax
                sui 30h
                rlc
                rlc
                rlc
                rlc
                jmp histore
alphax:         ani 00001111b
                adi 9
histore:        sta bytehi
                xchg
                lxi h,bytelo
                add m
                xchg
                sta bytehex
                mov m,a
                ret
;
newline:        lda col
                sta rescol
                lda line
                sta resline
                sui 1
                cpi 3ch
                jz tief
                sta line
                jmp newy
tief:           adi 16
                sta line
newy:           mvi a,16
                lxi h,secbuf
                add l
                mov l,a
                lda line
                sui 3dh
                jz newz
                add a
                add a
                add a
                add a
newz:           add l
                mov l,a
                mvi a,0
                adc h
                mov h,a
                mvi a,31h
                sta col
                lxi d,pos
                call prtmess
                call ascii
                lda rescol
                sta col
                lda resline
                sta line
                lxi d,pos
                call put
                ret
;
wrsec:          di
                lxi h,sectbyte
                mov a,m
                lxi h,secbuf
                mvi b,drq
                mvi c,busy
                mvi d,seekerr
                out sectreg
                lda side
                ori 0
                jc wrrec
                mvi a,wrsec1
                jmp wrout
wrrec:          mvi a,wrsec2
wrout:          out fdccom
rloop1:         in fdcstat
                mov e,a
                ana b
                jz ex1
                mvi e,0ffh
                jmp write
ex1:            mov a,e
                ana d
                jnz recwr
                jmp rloop1
wrcheck:        in fdcstat
                ana b
                jz wrcheck
                dcr e
                jz wrpre
write:          mov a,m
                out datareg
                inx h
                jmp wrcheck
wrpre:          mov a,m
                out datareg
wrfertig:       ei
                ret
;
wrnoch:         in fdcstat
                ana b
                jnz wrnoch
                jmp write
;
recwr:          in cardstat
                cpi intreq
                jz wrnoch
                lxi d,messa
                call prtmess
                jmp wrfertig
;
```

Listing zum Programm »MON.ASM« (Schluß)