Helmut Drieger

Vielspurig

80-Spur-Laufwerke am mc-CP/M-Computer

Will man andere als 40-Spur-Laufwerke am mc-CP/M-Computer betreiben, hilft der Artikel „Mini-Floppy-Anschluß" von R.-D. Klein [1] nicht weiter. Die Diskdefinitionsmakros werden nicht beschrieben; ebensowenig die Routinen zur Pufferverwaltung. In Anlehnung an das oben erwähnte BIOS hier nun eine etwas genauere Beschreibung.

Das Listing in *Bild 1* enthält gegenüber dem Klein-BIOS im wesentlichen drei Änderungen: Die Routinen GETTRK und PUTTRK werden zu einer Routine zusammengefaßt. Sie unterscheiden sich genau durch ein Byte, und Platz ist immer knapp. Die Routinen WRITE und READ werden ebenfalls zusammengefaßt. Nur die wenigen unterschiedlichen Befehle werden extra programmiert. Drittens erfolgte eine Vergrößerung des Datenblocks von 1 KByte auf 2 KByte. Die ersten beiden Änderungen bedürfen keiner weiteren Beschreibung, sie sind aus dem Assemblerlisting ersichtlich. Auf die dritte Änderung möchte ich etwas genauer eingehen: Das veränderte BIOS bedient ein 35-Spur-Laufwerk als CP/M-Laufwerk A und C und ein 80-Spur-Laufwerk als B und D. Die Umstellung auf zwei 80-Spur-Laufwerke bereitet keine Schwierigkeiten: Das DISKDEF 1 Makro wird zu DISKDEF 0; die DISKDEF's 1, 2, 3 werden DISKDEF 0 gleichgesetzt.

Begriffsbestimmung

BLS (Block size) ist die Größe des physischen Datenblocks im BIOS, der bei einem Sektorzugriff von der Diskette zum Speicher oder umgekehrt transportiert wird. Von diesem Wert hängen alle anderen Parameter zur Definition eines Diskettenlaufwerkes im BIOS ab.
DKS (Disk storage) ist das Speichervermögen des jeweiligen Laufwerkes. Beim Systemlaufwerk ist zu berücksichtigen, daß in der Regel die ersten vier Spuren für das Betriebssystem reserviert werden und somit nicht in die Rechnung eingehen. Berechnet wird DKS in BLS-Einheiten. Um z. B. ein 80-Spur-Laufwerk zu bestimmen, geht man wie folgt vor: Anzahl Sektoren × Anzahl Bytes pro Sektor × Anzahl Spuren (ohne Systemspuren) dividiert durch BLS. Herr Klein zieht vom Ergebnis noch 1 ab, was aber das DISKDEF-Makro noch mal tut. Nehmen wir 16 Sektoren zu je 256 Byte und 80 − 4 Spuren bei BLS = 1024 Byte an, so erhalten wir:

16 × 256 × 76/1024 = 304 − 1 = 303;
also 303 BLS-Einheiten, in diesem Fall gleich 303 KByte. Im Klein-BIOS ist BLS = 1024.
Will man dieses 80-Spur-Laufwerk in das BIOS aufnehmen, darf man einen wichtigen Satz im CP/M-Handbuch nicht überlesen: Wird DKS größer als 255 (was hier der Fall ist), muß BLS größer als 1024 gewählt werden. Die nächstmögliche Größe ist 2048. Die Rechnung sieht dann so aus:
16 × 256 × 76/2048 = 152 − 1 = 151
also 151 BLS-Einheiten.
Folgerung: Das Klein-BIOS muß von BLS = 1024 auf BLS = 2048 umgestellt werden. Wenn man die READ- und WRITE-Routine zusammengefaßt hat, ist die Umstellung durch Ändern von drei Bytes erledigt.

Der Datenverkehr

Das BDOS verkehrt mit dem BIOS nur in 128-Byte-Blöcken. Eine andere Größe kennt das BDOS nicht. Die kleinste adressierbare Einheit auf der angenommenen Diskette ist der Sektor (hier mit 256 Byte). Eine Spur enthält 16 Sektoren; die Spurkapazität ist also 4096 Byte. Aus der BDOS-Sicht sind dies 32 Sektoren zu je 128 Byte. Wie wir festgestellt haben, muß BLS = 2048 sein. Eine Spur kann also genau zwei BIOS-Datenblöcke aufnehmen.
Adressiert das BDOS einen Sektor, muß das BIOS dafür sorgen, daß der richtige 2048-Byte-Block auf der Diskette angesprochen wird und innerhalb dieses Blocks der richtige 128-Byte-Abschnitt. Das BIOS muß also eine Rechenroutine enthalten, die die logische Sektornummer (0...31) in eine physische Sektornummer umrechnet an der der 2048-Byte-Block beginnt. Da eine Spur zwei solcher Blocks faßt, liefert die Routine nur zwei unterschiedliche physische Sektornummern. Der erste Block liegt auf den Sektoren 1 bis 8 und enthält die logischen Sektoren 0 bis 15. Der zweite Block einer Spur liegt physisch auf 9 bis 16 und enthält die logischen Sektoren 16 bis 31. Die Routine CALC erfüllt also folgende Funktionen: BSN = 1 für $0 \leq$ LSN ≤ 15 und BSN = 9 für $16 \leq$ LSN ≤ 31, wobei BSN die Blocksektornummer ist, also der physische Sektor, an dem der Datenblock beginnt; LSN ist die logische Sektornummer des BDOS. Damit die CALC-Routine auch bei BLS = 2048 arbeitet, muß lediglich die Maske der UND-Verknüpfung von 12 auf 8 verändert werden. Das war das erste zu ändernde Byte (Adresse EBF5H in Bild 1).
Die Routinen GETTRK und PUTTRK werden von READ und WRITE aufgerufen. GETTRK und PUTTRK werden zu GETPUT zusammengefaßt. GETPUT liest, bzw. schreibt einen 2048-Byte-Block, was acht physikalischen Sektoren entspricht. Die neu eingeführte Konstante RPT (repeat) errechnet sich aus BLS dividiert durch 256 und ist in unserem Falle acht. Das war das zweite zu verändernde Byte (Adresse EC98H in Bild 1).
Die Routinen READ und WRITE besorgen den Transport zwischen Speicher (DMA-Adresse) und BIOS-Datenblock. Beide Routinen wurden bis auf die letzten Befehle zur Routine RW (read-write) zusammengefaßt. RW enthält die Berechnung der Adresse des logischen Sektors innerhalb des Datenblocks nach folgender Funktion:
LSN' = 0...15 für
LSN = 0...15 oder 16...31
LSNB = LSN' × 128 + BBA.
Dabei ist LSN die logische Sektornummer. LSN' kann nur Werte zwischen 0 und 15 annehmen, während LSN zwischen 0 und 31 liegt. BBA ist die Pufferbasisadresse, hier konstant F800H.
LSNB ist die Adresse des logischen Sektors im Datenpuffer.
Vor der Pufferadressierung stellt RW sicher, daß der richtige Datenblock geladen ist. Jeder Block enthält 16 logische Sektoren (gegenüber früher 8); die Sektornummer wird also mit 15 UND-maskiert (früher 7). Das war das dritte zu ändernde Byte (Adresse EC31H in Bild 1).

```
                ;
                ;*******************************************************
                ;
                ;       B I O S   FUER MC-CP/M COMPUTER
                ;       -------------------------------
                ;
                ;       DRIVE A: UND C: 35 TRACK 16 SEKTOREN JE 256 BYTE
                ;       DRIVE B: UND D: 80 TRACK 16 SEKTOREN JE 256 BYTE
                ;
                ;       16 PHYSISCHE SEKTOREN = 32 LOGISCHE SEKTOREN
                ;
                ;       BLOCKGROESSE = 2048 BYTE
                ;
                ;*******************************************************
                ;
                ;
                ;       H. DRIEGER   7.8.1984
                ;
                ;
EA00                    ORG     0EA00H      ;60K CP/M BIOS
D400 =          CPMB    EQU     0D400H      ; CP/M-BASISADRESSE
DC06 =          BDOS    EQU     0DC06H      ; BDOS-BASISADRESSE
1600 =          CPML    EQU     $-CPMB      ; CP/M-LAENGE
EA00 =          BIOS    EQU     $           ; BIOS-BASISADRESSE
0000 =          BIAS    EQU     0           ; DISPLACEMENT
                ;
002C =          NSECTS  EQU     CPML/128    ; SEKTOREN FUER CP/M
0004 =          OFFSET  EQU     4           ; START BEI TRACK 4
                ;
0004 =          CDISK   EQU     4           ; DRIVE-NR (ADRESSE 4)
0080 =          BUFF    EQU     80H         ; BUFFER ADRESSE
0100 =          CPMSEC  EQU     256         ; 1 SECTOR = 256 BYTE
0800 =          BLS     EQU     2048        ; BLOCKSIZE
0008 =          RPT     EQU     BLS/CPMSEC  ; LOG SEKTOREN / BLOCK
F800 =          BUFFER  EQU     0F800H      ; DATENBLOCK = FFFFH - BLS
                ;
                ; BIOS - SPRUNGVERTEILER
                ; ----------------------
                ;
EA00 C3FCEA             JMP     BOOT        ; KALTSTART
EA03 C31EEB     WBOOTE: JMP     WBOOT       ; WARMSTART
EA06 C312F0             JMP     CONST       ; KONSOLSTATUS
EA09 C3F6EA             JMP     CONIN       ; KONSOLEINGABE
EA0C C309F0             JMP     CONOUT      ; KONSOLAUSGABE
EA0F C30FF0             JMP     LIST        ; DRUCKAUSGABE
EA12 C30CF0             JMP     PUNCH       ; STANZAUSGABE
EA15 C306F0             JMP     READER      ; LOCHSTREIFENLESER
EA18 C3A2EB             JMP     HOME        ; TRACK 0
EA1B C3A7EB             JMP     SELDSK      ; DRIVE AUSWAEHLEN
EA1E C3BDEB             JMP     SETTRK      ; TRACK AUSWAEHLEN
EA21 C3C2EB             JMP     SETSEC      ; SEKTOR AUSWAEHLEN
EA24 C3DEEB             JMP     SETDMA      ; ZIEL- BZW. QUELLADRESSE
                                            ; IM ANWENDERPROGRAMM
EA27 C365EC             JMP     READ        ; AUSGEWAEHLTEN SEKTOR LESEN
EA2A C370EC             JMP     WRITE       ; AUSGEWAEHLTEN SEKTOR SCHREIBEN
EA2D C3A0EB             JMP     LISTST      ; DRUCKERSTATUS
EA30 C3C7EB             JMP     SECTRAN     ; SEKTORNR. UEBERSETZEN
                ;
                ;       LAUFWERKS-KAPAZITAETEN FESTLEGEN
                ;
003D =          DISKK35 EQU      61         ; (16*256*31/BLS)-1
0097 =          DISKK80 EQU     151         ; (16*256*76/BLS)-1
                ;
                ;       LAUFWERKE MIT MAKROS AUS DISKDEF.LIB DEFINIEREN
                ;
                ;
                ;       A: UND C:  16 SEKTOREN ZU 256 BYTE =
                ;                  32 SEKTOREN ZU 128 BYTE
                ;                  35 SPUREN
                ;       B: UND D:  DTO. 80 SPUREN
                ;
                        DISKS   4           ; 2 LAUFWERKE, 4 SEITEN
EA33+=          DPBASEEQU$;BASE OF DISK PARAMETER BLOCKS
EA33+00000000   DPE0:DWXLT0,0000H;TRANSLATE TABLE
EA37+00000000   DW0000H,0000H;SCRATCH AREA
EA3B+E3EC73EA   DWDIRBUF,DPB0;DIR BUFF,PARM BLOCK
EA3F+6BED63ED   DWCSV0,ALV0;CHECK, ALLOC VECTORS
EA43+00000000   DPE1:DWXLT1,0000H;TRANSLATE TABLE
EA47+00000000   DW0000H,0000H;SCRATCH AREA
EA4B+E3EC82EA   DWDIRBUF,DPB1;DIR BUFF,PARM BLOCK
EA4F+8EED7BED   DWCSV1,ALV1;CHECK, ALLOC VECTORS
EA53+00000000   DPE2:DWXLT2,0000H;TRANSLATE TABLE
EA57+00000000   DW0000H,0000H;SCRATCH AREA
EA5B+E3EC73EA   DWDIRBUF,DPB2;DIR BUFF,PARM BLOCK
EA5F+B6EDAEED   DWCSV2,ALV2;CHECK, ALLOC VECTORS
EA63+00000000   DPE3:DWXLT3,0000H;TRANSLATE TABLE
EA67+00000000   DW0000H,0000H;SCRATCH AREA
EA6B+E3EC82EA   DWDIRBUF,DPB3;DIR BUFF,PARM BLOCK
EA6F+D9EDC6ED   DWCSV3,ALV3;CHECK, ALLOC VECTORS
                ;
                ;       35-TRACK-DRIVE 4 RESEVIERTE TRACKS
                ;
                        DISKDEF 0,0,31,0,BLS,DISKK35,64,64,OFFSET
EA73+=          DPB0EQU$;DISK PARM BLOCK
EA73+2000       DW32;SEC PER TRACK
EA75+04         DB4;BLOCK SHIFT
EA76+0F         DB15;BLOCK MASK
EA77+01         DB1;EXTNT MASK
EA78+3C00       DW60;DISK SIZE-1
EA7A+3F00       DW63;DIRECTORY MAX
EA7C+80         DB128;ALLOC0
EA7D+00         DB0;ALLOC1
EA7E+1000       DW16;CHECK SIZE
EA80+0400       DW4;OFFSET
0000+=          XLT0EQU0;NO XLATE TABLE
                ;
                ;       80-TRACK-DRIVE 4 RESEVIERTE TRACKS
                ;
                        DISKDEF 1,0,31,0,BLS,DISKK80,128,128,OFFSET
EA82+=          DPB1EQU$;DISK PARM BLOCK
EA82+2000       DW32;SEC PER TRACK
EA84+04         DB4;BLOCK SHIFT
EA85+0F         DB15;BLOCK MASK
EA86+01         DB1;EXTNT MASK
EA87+9600       DW150;DISK SIZE-1
EA89+7F00       DW127;DIRECTORY MAX
EA8B+C0         DB192;ALLOC0
EA8C+00         DB0;ALLOC1
EA8D+2000       DW32;CHECK SIZE
EA8F+0400       DW4;OFFSET
0000+=          XLT1EQU0;NO XLATE TABLE
                ;
                ;       DEFINITION WIE DRIVE 0
                ;
                        DISKDEF 2,0
EA73+=          DPB2EQUDPB0;EQUIVALENT PARAMETERS
0008+=          ALS2EQUALS0;SAME ALLOCATION VECTOR SIZE
0010+=          CSS2EQUCSS0;SAME CHECKSUM VECTOR SIZE
0000+=          XLT2EQUXLT0;SAME TRANSLATE TABLE
```

Bild 1. Das geänderte BIOS im Listing, dazu muß noch mit „MACLIB“ die Bibliothek „DISKDEF“ eingebunden werden

```
                        ;
                        ;       DEFINITION WIE DRIVE 1
                        ;
                                DISKDEF 3,1
EA82+=                  DPB3EQUDPB1;EQUIVALENT PARAMETERS
0013+=                  ALS3EQUALS1;SAME ALLOCATION VECTOR SIZE
0020+=                  CSS3EQUCSS1;SAME CHECKSUM VECTOR SIZE
0000+=                  XLT3EQUXLT1;SAME TRANSLATE TABLE
                        ;
                        ;
F01E =                  MON80   EQU     0F01EH  ; MONITOR ADRESSE
                        ;
000D =                  CR      EQU     0DH     ; CARRIAGE RETURN
000A =                  LF      EQU     0AH     ; LINE FEED
                        ;
EA91 1A                 SIGNON: DB 26
EA92 3630204B20                 DB '60 K CP/M  MINIBIOS (DRIEGER) BLS=2048KB'
EABA 20372E382E                 DB ' 7.8.84'
EAC1 0D0A413A20                 DB CR,LF,'A: C:  35 TRACK 16 SECT'
EADA 0D0A423A20                 DB CR,LF,'B: D:  80 TRACK 16 SECT'
EAF3 0D0A00                     DB CR,LF,0
F012 =                  CONST   EQU     0F012H  ; KONSOLSTATUS
EAF6 CD03F0             CONIN:  CALL    0F003H  ; KONSOLEINGABE
EAF9 E67F               ;       ANI     7FH     ; MSB=0
EAFB C9                 ;       RET
F009 =                  CONOUT: EQU     0F009H  ; KONSOLAUSGABE
                        ;
F00F =                  LIST    EQU     0F00FH  ; DRUCKERAUSGABE
F00C =                  PUNCH   EQU     0F00CH  ; SIO-B OUT
F006 =                  READER  EQU     0F006H  ; SIO-B IN
                        ;
EAFC 310001             BOOT:   LXI     SP,BUFF+80H
EAFF 2191EA             ;       LXI     H,SIGNON ; SYSTEM-MELDUNG
EB02 CDE4EB             ;       CALL    PRMSG
                        ;
                        ;       MONITOR KURZSCHLIESSEN, WRITEFLAG RESET
                        ;       DRIVE UNDEFINIERT
                        ;
EB05 AF                 ;       XRA     A
EB06 32CEEC             ;       STA     WRTFLG
EB09 3EFF               ;       MVI     A,0FFH
EB0B 32CFEC             ;       STA     DRVAKT
EB0E 211EEB             ;       LXI     H,WBOOT
EB11 2234F0             ;       SHLD    0F033H+1
EB14 2237F0             ;       SHLD    0F036H+1
                        ;
EB17 AF                 ;       XRA     A
EB18 320400             ;       STA     CDISK
EB1B C35EEB             ;       JMP     G@CPM
                        ;
                        ;       WBOOT LAEDT BDOS+CCP, ABER NICHT DAS BIOS
                        ;
EB1E 318000             WBOOT:  LXI     SP,BUFF   ; SP:=BUFF
EB21 C5                 ;       PUSH    B
                        ;
                        ;       ALTEN TRACK ZURUECKSCHREIBEN,
                        ;       WENN WRITEFLAG = SET
                        ;
EB22 3ACEEC             WBOOT0: LDA     WRTFLG
EB25 B7                 ;       ORA     A
EB26 CA2CEB             ;       JZ      NOTBAC
EB29 CDC2EC             ;       CALL    PUTTRK
                        ;
EB2C 3EFF               NOTBAC: MVI     A,0FFH    ; DRIVE UNDEFINIERT FUER
EB2E 32CFEC             ;       STA     DRVAKT    ; DISKWECHSEL
EB31 2100D4             ;       LXI     H,CPMB    ; START ADRESSE
EB34 0616               ;       MVI     B,NSECTS/2
EB36 1601               ;       MVI     D,1       ; START TRACK 1 DA DD
EB38 1E01               ;       MVI     E,1       ; SEKTOR 1..16
EB3A E5                 RDSEC:  PUSH    H         ; SAVE REGISTER
EB3B D5                 ;       PUSH    D
EB3C C5                 ;       PUSH    B
                        ;       DRIVE 0, DD
EB3D 0ED0               ;       MVI     C,11010000B
EB3F 0601               ;       MVI     B,1       ; 1=READ
EB41 CDCBEC             ;       CALL    EXEC
EB44 C1                 ;       POP     B         ; REGISTER RELOAD
EB45 D1                 ;       POP     D
EB46 E1                 ;       POP     H
EB47 DA88EB             ;       JC      BOOTERR   ; BOOT-FEHLER
EB4A D5                 ;       PUSH    D         ; NEXT ADRESSE
EB4B 110001             ;       LXI     D,256     ; DOUBLE DENSE BOOT
EB4E 19                 ;       DAD     D
EB4F D1                 ;       POP     D
EB50 1C                 ;       INR     E         ; SEC:=SEC+1
EB51 7B                 ;       MOV     A,E
EB52 FE11               ;       CPI     16+1      ; ALLE SEKTOREN?
EB54 DA5AEB             ;       JC      RD1
EB57 1E01               ;       MVI     E,1       ; SEC:=1
EB59 14                 ;       INR     D         ; TRACK:=TRACK+1
EB5A 05                 RD1:    DCR     B         ; ZAEHLER:=ZAEHLER-1
EB5B C23AEB             ;       JNZ     RDSEC
                        ;
EB5E 018000             GOCPM:  LXI     B,BUFF
EB61 CDDEEB             ;       CALL    SETDMA    ; BUFFERADRESSE DEFINIEREN
EB64 3EC3               ;       MVI     A,JMP     ; WARMSTARTADRESSE DEFINIEREN
EB66 320000             ;       STA     0
EB69 2103EA             ;       LXI     H,WBOOTE
EB6C 220100             ;       SHLD    1
EB6F 320500             ;       STA     5
EB72 2106DC             ;       LXI     H,BDOS    ; BDOS-ADRESSE
EB75 220600             ;       SHLD    6
EB78 323800             ;       STA     7*8       ; MONITORADRESSE FUER RST 38H
EB7B 211EF0             ;       LXI     H,MON80
EB7E 223900             ;       SHLD    7*8+1
EB81 3A0400             ;       LDA     CDISK     ; ZULETZT BEDIENTES LAUFWERK
EB84 4F                 ;       MOV     C,A
EB85 C300D4             ;       JMP     CPMB      ; BDOS STARTEN
                        ;
EB88 C1                 BOOTERR:POP     B
EB89 0D                 ;       DCR     C         ; BOOT DRIVE 0 ??
EB8A CA91EB             ;       JZ      BOOTER0
EB8D C5                 ;       PUSH    B
EB8E C322EB             ;       JMP     WBOOT0
                        ;
EB91 219AEB             BOOTER0:LXI     H,BOOTMSG ; BOOT NICHT MOEGLICH
EB94 CDE4EB             ;       CALL    PRMSG
EB97 C31EF0             ;       JMP     MON80     ; ZUM MONITOR
                        ;
EB9A 3F424F4F54         BOOTMSG:DB      '?BOOT',0
                        ;
EBA0 AF                 LISTST: XRA     A         ; IMMER OK
EBA1 C9                 ;       RET
                        ;
EBA2 0E00               HOME:   MVI     C,0       ; AKTUELLER TRACK=0
EBA4 C3BDEB             ;       JMP     SETTRK
                        ;
                        ;       ADRESSE DES DISKPARAMETERHEADERS (DPH) FUER
                        ;       EIN BESTIMMTES LAUFWERK BESTIMMEN
                        ;       C  =    ANGEFORDERTE DRIVE-NR VOM BDOS
                        ;       HL =    ADRESSE DES DPH BEI RUECKKEHR
                        ;
```

```
EBA7 210000     SELDSK: LXI     H,0
EBAA 79                 MOV     A,C
EBAB FE04               CPI     NDISKS
EBAD D0                 RNC
EBAE 32D3EC             STA     DBANK     ;BEREICH 0..3
EBB1 69                 MOV     L,C
EBB2 2600               MVI     H,0
EBB4 29                 DAD     H
EBB5 29                 DAD     H
EBB6 29                 DAD     H
EBB7 29                 DAD     H
EBB8 1133EA             LXI     D,DPBASE
EBBB 19                 DAD     D
EBBC C9                 RET
                ;
EBBD 21D6EC     SETTRK: LXI     H,IOT     ; AKTUELLEN TRACK MERKEN
EBC0 71                 MOV     M,C
EBC1 C9                 RET
                ;
EBC2 21D7EC     SETSEC: LXI     H,IOS     ; AKTUELLEN SEKTOR MERKEN
EBC5 71                 MOV     M,C
EBC6 C9                 RET
                ;
                ;       SEKTORNUMMER UEBERSETZEN
                ;       BC = BDOS-SEKTORNUMMER
                ;       DE = ADRESSE DER UMWANDLUNGSTABELLE
                ;       DE = 0, WENN KEINE TABELLE VORHANDEN IST
                ;       HL = UEBERSETZTE SEKTORNUMMER
                ;            (HL=BC, WENN INTERLEAVINGFAKTOR=1)
                ;
EBC7 7A         SECTRAN:MOV     A,D
EBC8 B3                 ORA     E
EBC9 CAD6EB             JZ      SE1
EBCC 0600               MVI     B,0
EBCE EB                 XCHG
EBCF 09                 DAD     B
EBD0 7E                 MOV     A,M
EBD1 32D7EC             STA     IOS
EBD4 6F                 MOV     L,A
EBD5 C9                 RET
EBD6 69         SE1:    MOV     L,C       ; KEINE TABELLE
EBD7 79                 MOV     A,C
EBD8 32D7EC             STA     IOS
EBDB 2600               MVI     H,0
EBDD C9                 RET
                ;
EBDE 69         SETDMA: MOV     L,C       ; ZIEL- BZW. QUELLADRESSE
EBDF 60                 MOV     H,B
EBE0 22D8EC             SHLD    IOD
EBE3 C9                 RET
                ;
EBE4 7E         PRMSG:  MOV     A,M       ; SYSTEMMELDUNG AUSGEBEN
EBE5 B7                 ORA     A
EBE6 C8                 RZ
EBE7 E5                 PUSH    H
EBE8 4F                 MOV     C,A
EBE9 CD09F0             CALL    CONOUT
EBEC E1                 POP     H
EBED 23                 INX     H
EBEE C3E4EB             JMP     PRMSG
                ;
                ; READ UND WRITE BENUTZEN EXEC
                ; HL = ZIEL- BZW. QUELLADRESSE
                ; DE = TRACK/SECTOR
                ; B  = 0 RSTORE, 1 READ, 2 WRITE
                ; C  = DRIVE 0...3
```

```
                ;       10H,11H,12H,13H  DOUBMIN 0D0H,0D1H,0D2H,0D3H
                ;                                A    C    B    D
                ; 2K BUFFER IN MONITORGEBIET (F800H - FFFFH)
                ; DIE MONITORROUTINEN WERDEN UNTER CP/M NICHT BENOETIGT
                ; ---------------------------------------------------
                ;
                ; CALC RECHNET DBANK IN PHYS LAUFWERK UM,
                ;       RECHNET IOS IN SEKTORBUFFERNR UM
                ;       (SEKTORBUFFERNR IST VON BLS ABHAENGIG, SEKTOR-
                ;       BUFFERNR IST DIE SEKTORNUMMER, AN DER DER
                ;       BUFFER BEGINNT)
                ;
EBF1 3AD7EC     CALC:   LDA IOS           ; A:=IOS (0..31)
EBF4 0F                 RRC               ; A:=A/2
EBF5 E608               ANI 00001000B     ;12 BEI 1024 BYTE-BUFFER
                                          ; 8 BEI 2048 BYTE-BUFFER
                                          ; A:=A AND MASKE (8 ODER 12)
EBF7 3C                 INR A             ; A:=A+1
EBF8 5F                 MOV E,A           ; E:=A
                ;
EBF9 3AD3EC             LDA DBANK         ;DRIVE 0->0 1->2 2->1 3->3
EBFC FE02               CPI 2
EBFE C206EC             JNZ CAL1
EC01 3E01               MVI A,1
EC03 C30DEC             JMP CAL2
EC06 FE01       CAL1:   CPI 1
EC08 C20DEC             JNZ CAL2
EC0B 3E02               MVI A,2
EC0D 4F         CAL2:   MOV C,A
EC0E 3AD6EC             LDA IOT
EC11 57                 MOV D,A           ;TRACK=D SEKTOR=E DRIVE=C
EC12 C9                 RET
                ;
                ;
EC13 CDF1EB     RW:     CALL    CALC      ; ZUM VERGLEICHEN
EC16 3ACFEC             LDA     DRVAKT    ; DRIVE GLEICH ?
EC19 B9                 CMP     C
EC1A C240EC             JNZ     LOAD
EC1D 3AD0EC             LDA     TRKAKT    ; TRACK GLEICH ?
EC20 BA                 CMP     D
EC21 C240EC             JNZ     LOAD
EC24 3AD1EC             LDA     SEKAKT    ; SEKTOR GLEICH ?
EC27 BB                 CMP     E
EC28 C240EC             JNZ     LOAD
                ;
                ; RICHTIGER SEKTOR IST BEREITS
                ; IM AKTUELLEN BUFFER GELADEN
                ;
EC2B 2100F8     RWRD:   LXI     H,BUFFER  ; SEKTORADRESSE IM
                                          ; BUFFER BERECHNEN
EC2E 3AD7EC             LDA     IOS       ; ADR = 0..15  * 128 + BUFFERADR
EC31 E60F               ANI     00001111B ; 07H BEI BLS=1024
                                          ; 0FH BEI BLS=2048
EC33 57                 MOV     D,A
EC34 1E00               MVI     E,0       ; SCHIEBEN MIT Z80 BEFEHLEN
EC36 CB2A               DB      0CBH,2AH  ; SRA D
EC38 CB1B               DB      0CBH,1BH  ; RR E   = *256/2
EC3A 19                 DAD     D         ; +BUFFER
EC3B EB                 XCHG              ; DE IST ZIEL
EC3C 2AD8EC             LHLD    IOD       ; DMA ADRESSE IST QUELLE
EC3F C9                 RET
                ;
EC40 3ACEEC     LOAD:   LDA     WRTFLG    ; WRTFLG SET ?
EC43 B7                 ORA     A
EC44 CA4DEC             JZ      LOAD1
EC47 CDC2EC             CALL    PUTTRK    ; ALTEN SEKTOR ZURUECKSCHREIBEN
EC4A DA7FEC             JC      ERRIO
```

```
EC4D CDF1EB     LOAD1:  CALL    CALC        ; BUFFERNR BERECHNEN
EC50 79                 MOV     A,C
EC51 32CFEC             STA     DRVAKT      ; DRIVE MERKEN
EC54 7B                 MOV     A,E
EC55 32D1EC             STA     SEKAKT      ; SEKTOR MERKEN
EC58 7A                 MOV     A,D
EC59 32D0EC             STA     TRKAKT      ; TRACK MERKEN
EC5C CDB9EC             CALL    GETTRK      ; NEUEN BUFFER HOLEN
EC5F DA7FEC             JC      ERRIO
EC62 C32BEC             JMP     RWRD
                ;
                ;       SEKTOR AN ZIELADRESSE UEBERTRAGEN
                ;
EC65 CD13EC     READ:   CALL    RW
EC68 EB                 XCHG
EC69 018000             LXI     B,128
EC6C EDB0               DB      0EDH,0B0H   ; LDIR
EC6E AF                 XRA     A
EC6F C9                 RET
                ;
EC70 CD13EC     WRITE:  CALL    RW          ; SEKTOR IN BUFFER UEBERTRAGEN
EC73 018000             LXI     B,128
EC76 EDB0               DB      0EDH,0B0H   ; LDIR
EC78 3E01               MVI     A,1
EC7A 32CEEC             STA     WRTFLG      ; SET WRTFLG
EC7D AF                 XRA     A
EC7E C9                 RET
                ;
EC7F 3E01       ERRIO:  MVI     A,1
EC81 B7                 ORA     A
EC82 C9                 RET
                ;
                ; BUFFERVERWALTUNG
                ;
                ; TRKAKT,SEKAKT UND DRVAKT BESTIMMEN DIE NEUE
                ; BUFFERADRESSE
                ;
EC83 AF         GETPUT: XRA     A           ; WRTFLG RESET
EC84 32CEEC             STA     WRTFLG
EC87 2100F8             LXI     H,BUFFER    ; HL:=BUFFERADRESSE
EC8A 3AD1EC             LDA     SEKAKT
EC8D 5F                 MOV     E,A         ; E:=SEKTOR
EC8E 3ACFEC             LDA     DRVAKT
EC91 F6D0               ORI     11010000B   ; DRIVE PHYS 0,1,2,3 DD
EC93 4F                 MOV     C,A         ; C:=DRIVE
EC94 3AD2EC             LDA     MODE        ; MODE=1 READ, =2 WRITE
EC97 47                 MOV     B,A         ; B:=MODE
EC98 1608               MVI     D,RPT       ; RPT MAL
                                            ; RPT=ANZAHL DER 256BYTE-BLOECKE
                                            ; IM BUFFER
EC9A E5         GP1:    PUSH    H
EC9B D5                 PUSH    D
EC9C C5                 PUSH    B
EC9D 3AD0EC             LDA     TRKAKT      ; D:=TRACK
ECA0 57                 MOV     D,A
ECA1 CDCBEC             CALL    EXEC        ; FLOPPY-ZUGRIFF
ECA4 C1                 POP     B
ECA5 D1                 POP     D
ECA6 E1                 POP     H
ECA7 DAB7EC             JC      ERRX
ECAA D5                 PUSH    D
ECAB 110001             LXI     D,256
ECAE 19                 DAD     D           ; HL:=HL+256
ECAF D1                 POP     D
ECB0 1C                 INR     E           ; SEKTOR:=SEKTOR+1
```

```
ECB1 15                 DCR     D
ECB2 C29AEC             JNZ     GP1         ; ALLE SEKTOREN ?
ECB5 AF                 XRA     A
ECB6 C9                 RET
                ;
ECB7 37         ERRX:   STC                 ; CARRY SETZEN
ECB8 C9                 RET
                ;
ECB9 3E01       GETTRK: MVI     A,1         ; READ MODE
ECBB 32D2EC             STA     MODE
ECBE CD83EC             CALL    GETPUT
ECC1 C9                 RET
                ;
ECC2 3E02       PUTTRK: MVI     A,2         ; WRITE MODE
ECC4 32D2EC             STA     MODE
ECC7 CD83EC             CALL    GETPUT
ECCA C9                 RET
                ;
                ;
                ; EXEC SYSTEM
                ;
                ; FLOPPY-ROUTINEN FUER 5 1/4 ZOLL IM MONITOR
                ;
ECCB C327F0     EXEC:   JMP     0F027H
                ;
                ; BIOS RAM-BEREICH
                ;
ECCE 00         WRTFLG: DB      0       ; <>0 SET
ECCF 00         DRVAKT: DB      0       ; DRIVE LOADED
ECD0 00         TRKAKT: DB      0       ; TRACK
ECD1 00         SEKAKT: DB      0       ; SEKTOR
                ;
ECD2 00         MODE:   DB      0       ; 1=READ, 2=WRITE
ECD3 00         DBANK:  DB      0
ECD4 80         IOPB:   DB      80H     ; NORM IO
ECD5 01         ION:    DB      1       ; SECTOR NR
ECD6 04         IOT:    DB      OFFSET ; TRK
ECD7 01         IOS:    DB      1
ECD8 8000       IOD:    DW      BUFF
ECDA 00         TRKBUF: DB      0
ECDB 00                 DB      0
ECDC 00                 DB      0
ECDD 00                 DB      0
ECDE 01         ALTDRV: DB      1
ECDF 0000       INDADR: DW      0
ECE1 0000       INDADR2:DW      0
                ;
                ;       DEFINITION DES DIRECTORY-BUFFERS UND DER
                ;       BDOS-SCRATCHPAD-AREAS FUER JEDES LAUFWERK
                ;
                ENDEF
ECE3+=          BEGDATEQU$
ECE3+           DIRBUF:DS128;DIRECTORY ACCESS BUFFER
ED63+           ALV0:DS8
ED6B+           CSV0:DS16
ED7B+           ALV1:DS19
ED8E+           CSV1:DS32
EDAE+           ALV2:DS8
EDB6+           CSV2:DS16
EDC6+           ALV3:DS19
EDD9+           CSV3:DS32
EDF9+=          ENDDATEQU$
0116+=          DATSIZEQU$-BEGDAT
                ;
EDF9            END
```

```
EA00 C3 FC EA C3 1E EB C3 12 F0 C3 F6 EA C3 09 F0 C3
EA10 0F F0 C3 0C F0 C3 06 F0 C3 A2 EB C3 A7 EB C3 BD
EA20 EB C3 C2 EB C3 DE EB C3 65 EC C3 70 EC C3 A0 EB
EA30 C3 C7 EB 00 00 00 00 00 00 00 E3 EC 73 EA 6B
EA40 ED 63 ED 00 00 00 00 00 00 00 E3 EC 82 EA 8E
EA50 ED 7B ED 00 00 00 00 00 00 00 E3 EC 73 EA B6
EA60 ED AE ED 00 00 00 00 00 00 00 E3 EC 82 EA D9
EA70 ED C6 ED 20 00 04 0F 01 3C 00 3F 00 80 00 10 00
EA80 04 00 20 00 04 0F 01 96 00 7F 00 C0 00 20 00 04
EA90 00 1A 36 30 20 4B 20 43 50 2F 4D 20 20 4D 49 4E
EAA0 49 42 49 4F 53 20 28 44 52 49 45 47 45 52 29 20
EAB0 42 4C 53 3D 32 30 34 38 4B 42 20 37 2E 38 2E 38
EAC0 34 0D 0A 41 3A 20 43 3A 20 20 33 35 20 54 52 41
EAD0 43 4B 20 31 36 20 53 45 43 54 0D 0A 42 3A 20 44
EAE0 3A 20 20 38 30 20 54 52 41 43 4B 20 31 36 20 53
EAF0 45 43 54 0D 0A 00 CD 03 F0 E6 7F C9 31 00 01 21
EB00 91 EA CD E4 EB AF 32 CE EC 3E FF 32 CF EC 21 1E
EB10 EB 22 34 F0 22 37 F0 AF 32 04 00 C3 5E EB 31 80
EB20 00 C5 3A CE EC B7 CA 2C EB CD C2 EC 3E FF 32 CF
EB30 EC 21 00 D4 06 16 16 01 1E 01 E5 D5 C5 0E D0 06
EB40 01 CD CB EC C1 D1 E1 DA 88 EB D5 11 00 01 19 D1
EB50 1C 7B FE 11 DA 5A EB 1E 01 14 05 C2 3A EB 01 80
EB60 00 CD DE EB 3E C3 32 00 00 21 03 EA 22 01 00 32
EB70 05 00 21 06 DC 22 06 00 32 38 00 21 1E F0 22 39
EB80 00 3A 04 00 4F C3 00 D4 C1 0D CA 91 EB C5 C3 22
EB90 EB 21 9A EB CD E4 EB C3 1E F0 3F 42 4F 4F 54 00
EBA0 AF C9 0E 00 C3 BD EB 21 00 00 79 FE 04 D0 32 D3
EBB0 EC 69 26 00 29 29 29 29 11 33 EA 19 C9 21 D6 EC
EBC0 71 C9 21 D7 EC 71 C9 7A B3 CA D6 EB 06 00 EB 09
EBD0 7E 32 D7 EC 6F C9 69 79 32 D7 EC 26 00 C9 69 60
EBE0 22 D8 EC C9 7E B7 C8 E5 4F CD 09 F0 E1 23 C3 E4
EBF0 EB 3A D7 EC 0F E6 08 3C 5F 3A D3 EC FE 02 C2 06
EC00 EC 3E 01 C3 0D EC FE 01 C2 0D EC 3E 02 4F 3A D6
EC10 EC 57 C9 CD F1 EB 3A CF EC B9 C2 40 EC 3A D0 EC
EC20 BA C2 40 EC 3A D1 EC BB C2 40 EC 21 00 F8 3A D7
EC30 EC E6 0F 57 1E 00 CB 2A CB 1B 19 EB 2A D8 EC C9
EC40 3A CE EC B7 CA 4D EC CD C2 EC DA 7F EC CD F1 EB
EC50 79 32 CF EC 7B 32 D1 EC 7A 32 D0 EC CD B9 EC DA
EC60 7F EC C3 2B EC CD 13 EC EB 01 80 00 ED B0 AF C9
EC70 CD 13 EC 01 80 00 ED B0 3E 01 32 CE EC AF C9 3E
EC80 01 B7 C9 AF 32 CE EC 21 00 F8 3A D1 EC 5F 3A CF
EC90 EC F6 D0 4F 3A D2 EC 47 16 08 E5 D5 C5 3A D0 EC
ECA0 57 CD CB EC C1 D1 E1 DA B7 EC D5 11 00 01 19 D1
ECB0 1C 15 C2 9A EC AF C9 37 C9 3E 01 32 D2 EC CD 83
ECC0 EC C9 3E 02 32 D2 EC CD 83 EC C9 C3 27 F0 00 ..
```

Bild 2. Der Hex-Dump des neuen BIOS

Die neue Systemdiskette

Die BIOS-Quelle BIOS.ASM wird mit MAC assembliert. Die Makrobibliothek DISKDEF.LIB muß vorhanden sein.

MAC liefert die Objektdatei BIOS.HEX, sie enthält den Code im Intel-Hex-Format. Da das BIOS direkte Adressen mit der Basis EA00H enthält, kann nicht mit LOAD gearbeitet werden. Das resultierende COM-File würde den ganzen TPA-Bereich bis zum Ende des BIOS enthalten (100h bis ca. EFFFH). Das Dienstprogramm DDT ist aber in der Lage, HEX-Files zu laden. Nur muß man wissen, wie ein Objektfile mit einer Basisadresse größer als 100H bei 100H abgelegt werden kann. Dazu ruft man DDT zunächst alleine auf und gibt dann das Kommando IBIOS.HEX ein. Mit dem zweiten Kommando R1700 wird dann das BIOS ab 100H in den Arbeitsspeicher geladen. Mit CTRL-C beendet man den DDT und speichert mit SAVE 4 BIOS.COM das Objektfile ab. Die Information wieviele 256-KByte-Blöcke beim SAVE-Kommando anzugeben sind, liefert DDT. Der Versatz beim R-Kommando errechnet sich wie folgt:

FFFFH − EA00H = 15FFH + 1
= 1600H + 100H = 1700H.

Nun formatiert man eine neue Diskette und kopiert darauf mit SYSGEN das alte System. Dann lädt man mit dem alten System mittels DDT das Boot-Programm wie es in Bild 6 des eingangs erwähnten Artikels abgedruckt ist. Die Adresse A0H ist mit 2 zu besetzen (WRITE-Befehl). Danach holt man das neue BIOS-.COM mit dem DDT in die TPA und kopiert es von 100H an seine Bestimmungsadresse EA00H. Das Boot-Programm liegt also auf 80H und das BIOS auf EA00H. Nach CTRL-C legt man die formatierte Diskette mit dem alten System in das A-Laufwerk und drückt den RESET-Knopf. Mit dem Monitorbefehl G80 schreibt man das neue System auf die Diskette. Die alten Disketten sind natürlich nicht mehr lesbar, durch abwechselndes „Booten" von altem und neuem System sind die gewünschten Files über DDT und SAVE zu kopieren. Eine etwas mühsame Angelegenheit.

Literatur

[1] *Klein, Rolf-Dieter:* Mini-Floppy-Anschluß. mc 2/1983, S. 66 und im mc-CP/M-Sonderheft, S. 44.

HX-20-Dateitransfer per Modem

Will man mit dem Epson-Computer HX-20 ein Programm mit Hilfe eines Modems per Telefon übertragen, so gibt es dafür im Prinzip den Befehl SAVE-„COM0:NAME". Leider hilft er normalerweise aber wenig, weil vor der eigentlichen Übertragung ja ein Dialog mit der Gegenstation abgewickelt werden muß und das zu übertragende Programm gewöhnlich in einem anderen LOGIN-Bereich des HX-20 steht als das Kommunikations-Programm.
Das Mini-Programm im *Bild* zeigt einen einfachen Ausweg. Es stellt eine Erweiterung des bereits in unserem Modem-Sonderheft abgedruckten kleinen Dialogprogramms dar: Bei Eingabe eines Doppelkreuz-Zeichens (Shift-3) verlangt es nach dem Filenamen der auf Kassette befindlichen Datei, die gesendet werden soll. Dann liest das Programm diese Datei zeilenweise von der Kassette und sendet sie zur Gegenstation. Zeile 100 sorgt dafür, daß nach jedem Return-Zeichen eine kleine Pause entsteht, die u. U. für das korrekte Verarbeiten beim angerufenen Rechner erforderlich ist (z. B. bei TEDAS).
Möchte man nun ein Basic-Programm senden, so muß man es zuerst als ASCII-Datei auf Kassette abspeichern. Das geht ganz einfach mit dem Kommando LIST-„CAS0:NAME". Dann spult man die Kassette wieder zurück und startet das Kommunikationsprogramm. Fe.

```
10 WIDTH20,4:OPEN"O",#1,
"COM0:(28N2F)":OPEN"I",#
2,"COM0:(28N2F)"
20 IFLOF(2)>0THENPRINTIN
PUT$(LOF(2),#2);
30 B$=INKEY$:IFB$=""THEN
20
40 IFB$="#"THENPRINT:GOT
O60
50 PRINT#1,B$;:GOTO20
60 INPUT"FILENAME";N$:OP
EN"I",#3,"CAS0:"+N$
70 IFEOF(3)THENCLOSE3:GO
TO20
80 LINEINPUT#3,B$:PRINT#
1,B$
90 LINEINPUT#2,B$:PRINTB
$
100 FORI=0TO99:NEXT:GOTO
70
```

Absenden eines auf Kassette als ASCII-Text stehenden Programms und Datenbank-Dialog ermöglicht dieses kurze Programm