```
; Ix.asm
; Intel-Hex Ladeprogramm unter CP/M
;
;#########################################################
; Quelle:
;   Programmentwicklung unter CP/M2.2 auf dem CPC464
;   Helmut Tischer
;   Markt und Technik-Verlag 1986
;   ISBN 3-89090-209-X
;#########################################################
;
; Stand:
;       15.01.2017      #V.I6 Recordzaehler anzeigen noch Fehler in der HEX-ASCII wandlung
;       14.01.2017      #V.I5 Daten von V24 direkt in Buffer schreiben
;                             und auf Diskette sichern
;       12.01.2017      #V.I3 Daten in Datei sichern
;       08.01.2017      #V.I2 Nur Daten entgegennehmen und auf Console ausgeben
;       02.08.2016      #V.I1 v24 uber RDR:
;       01.08.2016      #V.I0
;
boot    equ     0000h           ; Warmstart
bdos    equ     0005h           ; Bdoseinsprung
dma     equ     2000h           ;
buffer  equ     2000h           ; Zielbuffer
;
bios    equ     0ea00h          ; Start BIOS 60k
wboot   equ     bios + 3        ; Einsprung Warmstart
;
;
; BDOS Befehle
; Registerbelegung:
;       Befehl in (C)
;       Ausgabe in (E) 8Bit (DE) 16Bit
;       Eingabe in (A) 8Bit (DE) 16Bit
;       FCB-Adresse (DE)
;
;
reset   equ     00h     ; Warmstart
cicha   equ     01h     ; Zeichen von Tastatur lesen (A)
cocha   equ     02h     ; Zeichen auf Bildschirm ausgeben (E)
rdv24   equ     03H     ; Zeichen von RDR: (V24) holen (A)
wrv24   equ     04h     ; Zeichen an PUN: (v24) ausgeben (E)
rdio    equ     07h     ; IO-Byte holen (A)
wrio    equ     08h     ; IO-Byte setzen (E)
type    equ     09h     ; Zeichenkette auf Bildschirm (DE)
rdbuf   equ     0Ah     ; Zeichenkette von Tastatur (DE)
cists   equ     0Bh     ; Consolenstatus holen (A)
open    equ     0Fh     ; Datei Ã¶ffnen (A) (DE)
close   equ     10h     ; Datei schliessen (DE)
search  equ     11h     ; Datei in DIR suchen (A) (DE)
clear   equ     13h     ; datei loeschen (A) (DE)
read    equ     14h     ; Datei sequentiell lesen (A) (DE)
write   equ     15h     ; Datei sequentiell schreiben (A) (DE)
make    equ     16h     ; Datei erzeugen (A) (DE)
setdma  equ     1Ah     ; Buffer einstellen (DE)
rndrd   equ     21h     ; Datei wahlfrei lesen (A) (DE)
rndwr   equ     22h     ; Datei wahlfrei schreiben (A) (DE)
size    equ     23h     ; Dateigroesse berechnen (DE)
;
; Bioserweiterug
;
mv24bd  equ     wboot+30h       ; (DE) Baudrate einstellen
mv24ist equ     wboot+33h       ; Status (A)=00h kein Zeichen, FFh Zeichen
mv24rx  equ     wboot+36h       ; (A) Byte empfangen, wartet auf Zeichen
mv24rxz equ     wboot+39h       ; (A) Byte empfangen, abbruch nach 10s mit Control-Z (1ah)
mv24tx  equ     wboot+3ch       ; (C) Byte senden
mv24txz equ     wboot+3fh       ; (C) Byte senden, abbruch wenn Geraet nicht bereit
;
; V24 auf CF-Karte
;
cv24ist equ     wboot+42h       ; Status (A)=00h kein Zeichen, FFh Zeichen
cv24rx  equ     wboot+45h       ; (A) Byte empfangen
cv24tx  equ     wboot+48h       ; (C) Byte senden
cv24rxs equ     wboot+4bh       ; String empfangen bis CR
;
;-----------------------------------------------------------------------------

        org 0100h       ; Programmstart
;
start:  jmp     iread           ; Intel-Hex lesen
                                ;
ende:   lxi     d,meld_8        ; Fertigmeldung
        mvi     c,type          ;
        call    bdos            ;
        call    resio           ; IO-Byte wieder herstellen
```

```
        mvi     c,reset         ; Zurueck
        call    bdos
;
; Zeichen empfangen in Buffer ablegen und zeilenweise ausgeben
;
;
iread:  call    init            ; Vorbereitungen
        call    setio           ; IO-Byte fuer RDR:/PUN: setzen

        lxi     d,meld_2        ; Startmeldung
        mvi     c,type          ;
        call    bdos            ; ausgeben

iread_l: mvi    e,2ah           ; Sternchen
        mvi     c,cocha         ;
        call    bdos

        call    strrx           ; Daten empfangen bis Timeout
        ;
; schreibt z.Zt. direkt in Ausgabebuffer
; spaeter wird hier der Intel-Hex-Interpreter
; pro Zeile eingefuegt
        ;

iread_e: lxi    d,meld_3        ; Ende der Uebertragung
        mvi     c,type          ;
        call    bdos            ; ausgeben
        ;
        jmp     recanz          ; weiter mit Berechnung der
                                ; benoetigten Records

iread_er: lxi   d,err_0         ; Fehlermeldung
        mvi     c,type          ;
        call    bdos            ; ausgeben
        jmp     ende            ; Stop zurueck zum System
        ;
;
; Anzahl der Records berechnen
;       Quelle:
;               http://www.8085projects.info/Program18.html
;               Programmieren mit CP/M Alan R.Miller Sybex Verlag
;       modifiziert W.Römer 2017
; Recordanzahl = Bufferende - Bufferstart / 128 + 1(Rest)
;
; Records auf 255 begrenzt (=32kbyte)
; (HL)  = Bufferlaenge
; (C)   = Divisor = 80h (Recordlaenge)
; (B)   = Quotient (Recordzaehler)
;
; Differenz(Bufferlaenge) = Bufferende - Bufferstart
;
recanz: lhld    dstbeg          ; Record-Berechnung
        xchg                    ;
        lhld    dstend          ; sichern auf Diskette
        mov     a,l             ;
        sub     e               ;
        mov     l,a             ;
        mov     a,h             ;
        sbb     d               ;
        mov     h,a             ; Differenz in (HL)
        shld    buflaeng        ;
        ;
;
; Records = Bufferlaenge / 128 (bei Rest + 1)
;
        LHLD    buflaeng        ; Dividend
        mvi     c,80h           ; Divisor
        mvi     b,00h           ; Quotient = 0
rec_l:  MOV     A,L             ;
        SUB     C               ; Subtract divisor
        MOV     L,A             ; Save partial result
        JNC     rec_j           ; if CY  1 jump
        DCR     H               ; Subtract borrow of previous subtraction
rec_j:  inr     b               ; Increment quotient
        MOV     A,H             ;
        CPI     00h             ; Check if dividend < divisor
        JNZ     rec_l           ; if no repeat
        MOV     A,L             ;
        CMP     C               ;
        JNC     rec_l           ;
        cpi     00h             ;
        jz      rec_nr          ; Rest vorhanden
        inr     b               ; Recordzaehler + 1
rec_nr: mov     a,b             ;
        sta     recz            ; Store the quotient
```

```
;
; Recordanzahl in ASCII wandeln
;
save:   mov     b,a             ;
        rlc                     ; obersten Wert wandeln
        rlc                     ;
        rlc                     ;
        rlc                     ;
        adi     0fh             ; Highnibel ausblenden
        adi     90h             ;
        daa                     ;
        aci     40h             ;
        daa                     ;
        sta     asciih          ; sichern
        mov     a,b             ; untersten Wert wandeln
        adi     0fh             ;
        adi     90h             ;
        daa                     ;
        aci     40h             ;
        daa                     ;
        sta     asciil          ; sichern
;
        lxi     d,meld_6        ; Anzahl Records
        mvi     c,type          ;
        call    bdos            ; ausgeben
;
        lda     asciih          ;
        mov     e,a             ;
        mvi     c,cocha         ;
        call    bdos            ;
        lda     asciil          ;
        mov     e,a             ;
        mvi     c,cocha         ;
        call    bdos            ;
        ;

;
;-------------------------------------------------------------------------------
; Datei erzeugen und Buffer sichern
;
dats:   lxi     d,efcb          ; Datei suchen
        mvi     c,search        ;
        call    bdos            ;
        inr     a               ;
        jz      create          ;

found:  lxi     d,efcb          ; Datei schon vorhanden
        mvi     c,open          ; loeschen
        call    bdos            ;
        ana     a               ;
        jz      nodat           ; Nicht gefunden
        jmp     work            ;
        ;

        ;
create: lxi     d,efcb          ; nicht vorhanden
        mvi     c,make          ; Datei erzeugen
        call    bdos            ;
        inr     a               ;
        jz      full            ; Diskette voll

work:   lxi     d,dma           ; DMA auf Buffer setzen
        mvi     c,setdma        ;
        call    bdos            ;

        lda     recz            ; Zaehler fuer Records
wloop:  push    psw             ;
        lxi     d,efcb          ; Datenbuffer auf Diskette schreiben
        mvi     c,write         ;
        call    bdos            ;
        ana     a               ;
        jnz     full            ; Fehler Diskette voll
        ;
        ; DMA fuer naechsten Record Setzen
        ;
        lhld    dstbeg          ;
        lxi     d,0080h         ;
        dad     d               ;
        shld    dstbeg          ;
        xchg                    ;
        mvi     c,setdma        ;
        call    bdos            ;
        ;
        pop     psw             ; alle Records gefuellt
        dcr     a
```

```
        jnz     wloop           ; nein

        lxi     d,efcb          ; Datei schliessen
        mvi     c,close         ;
        call    bdos            ;
        inr     a               ;
        jz      nodat           ; Fehler
        lxi     d,meld_7        ; alles OK
        jmp     meld            ; Ende

nodat:  lxi     d,err_4         ; Fehlermeldung
        jmp     meld            ;

full:   lxi     d,err_5         ; Diskette voll
meld:   mvi     c,type          ;
        call    bdos            ;
        jmp     ende            ;
;
;
;-----------------------------------------------------------------
; Unterprogramme
;
init:   lxi     h,0             ; alten Stack
        dad     sp              ;
        shld    oldstk          ; sichern
        mvi     a,01            ;
        sta     recz            ; Recordzaehler auf 1 setzen
        lxi     h,buffer        ;
        shld    dstbeg          ; Zeiger für Puffer einrichten
        shld    dstend          ;
        ret
;
setio:  mvi     c,rdio          ; IO-Byte holen
        call    bdos            ;
        sta     oldio           ; sichern
        adi     11011000b       ; RDR:/PUN: auf Moppel-V24,
                                ; RDR mit Zeitueberwachung (A)= 1Ah nach 10s.
setio1: mov     e,a             ;
        mvi     c,wrio          ;
        call    bdos            ; zurueck schreiben
        ret                     ;

resio:  lda     oldio           ; IO-Byte wieder herstellen
        jmp     setio1          ;
;
;
; z.Zt. dierekt in Ausgabebuffer schreiben bis Timeout
;
strrx:  lhld    dstend          ; Staradresse holen

str_l:  push    h               ;
        mvi     c,rdv24         ;
        call    bdos            ; Byte holen
        pop     h               ;
        cpi     1ah             ;
        jz      str_z           ; Wartezeit abgelaufen
        mov     m,a             ;      speichern
        inx     h               ;
        jmp     str_l

str_z:  shld    dstend          ; Adr. sichern
        ret                     ; Fehlercode 1Ah
;
;
;
;-----------------------------------------------------------------
; Meldetexte
;
meld_0: db 0dh          ; CR  Startmeldung
        db 0ah          ; LF
        db "Intel-Hex Lader" ;
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ; Ende

meld_2: db 0dh          ; CR
        db 0ah          ; LF
        db "Sender starten"
crlf:   db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;

meld_3: db 0dh          ; CR
```

```
        db 0ah          ; LF
        db "Empfang Ende"
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;

meld_4: db 0dh          ; CR
        db 0ah          ; LF
        db "Uebertragung"
        db " OK"
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;

meld_5: db 0dh          ; CR
        db 0ah          ; LF
        db "Hex-Interpreter"
        db " fertig"
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;

meld_6: db 0dh          ; CR
        db 0ah          ; LF
        db "Records = "
        db "$"          ;
        db 00h          ;

meld_7: db 0dh          ; CR
        db 0ah          ; LF
        db "Datei"
        db " I.TXT"
        db " erzeugt"
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;


meld_8: db 0dh          ; CR
        db 0ah          ; LF
        db "Ende"
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;  1
;
;-----------------------------------------------------
; Fehlermeldungen
;
err_0:  db 0dh          ;
        db 0ah          ;
        db " Uebertragungs"
        db "fehler"
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;

err_1: db 0dh           ; CR
        db 0ah          ; LF
        db " kein ASCII"
        db " Zeichen"
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;

err_2: db 0dh           ; CR
        db 0ah          ; LF
        db " kein Startmarke"
        db " gefunden"
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;

err_3: db 0dh           ; CR
        db 0ah          ; LF
        db " Pruefsumme"
        db " falsch"
        db 0dh          ; CR
```

```
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;


err_4:  db 0dh          ; CR
        db 0ah          ; LF
        db "Datei nicht"
        db " gefunden"
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;

err_5:  db 0dh          ; CR
        db 0ah          ; LF
        db "Diskette voll"
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;

err_6:  db 0dh          ; CR
        db 0ah          ; LF
        db "Zeitueber"  ;
        db "schreitung"
        db 0dh          ; CR
        db 0ah          ; LF
        db "$"          ;
        db 00h          ;
;-----------------------------------------------------------------
; Bufferbereich
;
oldio:  ds      1h              ; Sicherung Io-Byte
dstbeg  ds      2               ;
dstend  ds      2               ;
bzaehl  db      0               ; Sicherung fuer Zaehlregister (B)
buflaeng ds     2               ; Bufferlaenge
recz     ds     1               ; Recordzaehler
asciih  db      30h             ; ASCII "0"
asciil  db      30h             ; ASCII "0"
;
efcb:   db      0               ; Schreiben auf Bezugslaufwerk
        db      "I       HEX"   ; Dateiname
        db      0               ; erste Extentnummer
        db      0               ;
        db      0               ;
        ds      17              ; beliebiger Inhalt
        db      0               ; 1.Record im Extend
        ds      3               ; erwiterter FCB
;
;-----------------------------------------------------------------
; Lokaler Stack
;
        ds      64      ; Stack fuer 32 Eintraege
oldstk: ds      2       ; alten Stack sichern
ibuff:  ds      40h     ; Buffer fuer Intel-Hex Zeile
```