Mikrocomputer-Gerätegeneration

M C 80.3x

-B E T R I E B S S Y S T E M-

(EGOS 30.1 - Teil 1+2

Systembeschreibung

Z.-Nr. 50300-4012.00 BA

VEB Elektronik Gera
Ausgabe 08/85

Inhaltsverzeichnis

1. Speicherverwaltung aus Anwendersicht

ØØØØ-ØFFF	Betriebssystemkern und VIS-Treiber (Tastaturroutinen, Sprungverteiler, Anwenderschnittstellen, Zeichengenerator, BSYS-Initialisierungsprogramm)
1ØØØ-1FFF	Magnetbandsoftware (Treiber, Handler)
2ØØØ-23FF	Sprungprogramme und Ladeprogramme zur Segmentumschaltung, Relativlader Verschiebeadresstabellenerstellprogramm
24ØØ-27FF	Geräteverwaltungskommandos und -Programme, Ausschrift Initialisierungtext auf Display
4ØØØ-BFFF	RAM zur freien Verfügbarkeit; segmentierbar in bis zu 8 32k-Segmenten, wobei immer nur 1 Segment eingeschaltet ist Bei Verwendung von weiteren Leiterplatten SPE sind ab Segment 4 (Nr. 3) auch ROM-Segmente zu wahlweise 16k-Byte (2716) oder 8k-Byte (27Ø8) möglich.
CØØØ-FØØØ	ZVE-RAM zur freien Verfügbarkeit; nicht segmentierbar (Vorzugsplatz für BASIC-Interpreter)
FØØ1-FFFF	ZVE-RAM zur bedingten freien Verfügbarkeit; von Adr. F6ØØH nach oben als lokaler Stack, gezeigert durch IY-Register, nutzbar, ab F8ØØH CPU-Stack (bis F6FF) F7Ø1-FFFF Arbeitszellen BSYS, Magnetband FCØØ: 1k-Byte-Schreibschutz

2. Speicherplatzverwaltung ZVE-RAM ab F6ØØH

....-F6ØØH	lokaler Stack, durch IY-Register gezeigert
F6Ø1-F8ØØH	CPU-Stack
F88Ø-F8DBH	BZPU-BASIC -Zwischencode -Zielpuffer
F8DC-F8FFH	Bereich zum Ausführen der Segmentumschaltung
F9ØØ-FB3FH	Arbeitszellen Magnetband
FB4Ø-FB7FH	Arbeitszellen VIS und Tastatur
FB8Ø-FBDFH	Eingabepuffer für Tastatur (Kommandoeingabe)

FBEØ-FBFAH	Systemzellen für BASIC
FBFB-FBFFH	Arbeitszellen für Segmentumschaltung
FCØØ-FC6FH	schreibgeschützte Zellen für active device table
FC7Ø-FECFH	schreibgeschützte Zellen für Anwender
FEDØ-FEE7H	schreibgeschützte Aussprungstellen für Sondertasten
FEE8-FEFFH	schreibgeschützte NMI- und RST-Aussprünge
FFØØ-FFAFH	für Erweiterungen vorgesehen
FFBØ-FFFFH	INT-Vektoren

3. INT-Vektoren und I-O-Adressen

FFØØ-FF7FH	frei für Anwender INT-Vektoren für zusätzliche E/A-Platinen
FF8Ø-FFAFH	frei für Systemerweiterungen
FFBØ-FFBBH	AKB für 52ØØ (nur MC 8Ø.31)
FFBC-FFD7H	reserviert für AKM (MC 8Ø.31/3Ø)
FFD8-FFDFH	CTC ZVE
FFEØ-FFE3H	PIO ZVE
FFE4-FFE7H	PIO ASP
FFE8-FFEFH	CTC ASP
FFFØ-FFFFH	SIO ASP

Adr.	ØØ-7FH	frei für Anwender (außer 3Ø-3FH)
	8Ø-83H	CTC auf ZVE (Kanal Ø-3)
	84-86H	PIO ZVE Kanal A (Datenwort/Steuerwort)
	85-87H	PIO ZVE Kanal B (Datenwort/Steuerwort)
	8CH	Schreibschutz sperren/freigeben für ZVE RAM ab FCØØ
	9Ø-BFH	zur Zeit nicht verwendet
	CØ-CFH	VIS-Adressen
	DØ/D1H	Datenwort/Steuerwort SIO Kanal A (auf ASP)
	D2/D3H	Datenwort/Steuerwort SIO Kanal B
	D4-D7H	CTC auf ASP (Kanal Ø-3; davon Kanal Ø und 1 zur Taktgewinnung des SIO)
	D8-DAH	PIO Kanal A Datenwort/Steuerwort

D9-DBH PIO Kanal B Datenwort/Steuerwort
EØ-EFH EPROM-Programmierung
3Ø-3FH Massenspeicher

4. VIS-Arbeitszellen und Tastaturzellen

FB4ØH-41H	Zeichengenerator (bei RESET ØCØØH)
FB42H	internes Statusbyte (Bit 4 = Ø: einfache Zeichenhöhe, Bit 4 = 1: doppelte Zeichenhöhe)
FB43H	Zeilenhöhe (ØAH=1Ø PIXEL normal, 14H=2Ø Pixel doppel)
FB44H/45H	CTC-Adresse Kanal 3 ASP (D7) TST-Zeichencode
FB46H/47H	Bildfenster und Bildfensterhintergrund
FB48H/49H	z. Z. frei
FB4AH	Rollzähler
FB4BH	Bildhintergrund
FB4CH/4DH	Zeiger auf lokaler Stack, Grundzustand
FB4EH/4FH	IY-Adresse für RIO-Vektor, Grundzustand
FB5ØH/51H	Textpufferanfang
FB52H/53H	maximales Textpufferende (Folgebyte)
FB54H/55H	Adresse Textpuffer Kursor
FB56H/57H	aktuelles Textpufferende
FB58H-5BH	Bildfensteranfang/Bildfensterende
FB5CH/5DH	aktueller Kursor in Pixelkoordinaten
FB5EH/5FH	aktuelles Bildfensterende
FB6ØH	Statusbyte Tastatur
FB61H	z. Z. frei
FB62H	letztes Zeichen von der Tastatur
FB63H/64H	aktueller Kursor (Spalte/Zeile)
FB65H/7BH	Tabulatorzellen in Pixelkoordinaten
FB7CH/7FH	Tastaturarbeitszellen

5. RAM-Kettung

Alle Dateien beliebigen Types sind in eine RAM-Kettung einzubinden. Die RAM-Kettung hat folgenden Aufbau:
Kennbyte: Distanz L | Distanz H | Name |(n Byte)| ØFFH |Beginn Datei

Mit dem Kennbyte wird der Typ der entsprechenden Datei gekennzeichnet (siehe Punkt 6). Die Adresse des Kennbytes erscheint als Programmadresse bei Anzeige des RAM-Inhaltes mittels der Kommandos CATA oder CAT bzw. CAT n, auch bei Umschaltung in ein anderes Segment.
Die Distanz wird gerechnet ab dem 1. Byte des Namens bzw. bei Dateien ohne Namen ab Trennbyte und zeigt auf das Kennbyte der folgenden Datei. Das heißt, von der tatsächlichen Distanz zwischen 2 Dateien sind zur Ermittlung der einzutragenden Distanz 3 zu subtrahieren.
Um die Adresse des tatsächlichen Dateibeginns zu ermitteln, ist das Unterprogramm RSU zu benutzen (siehe Punkt Softwareschnittstellen).
Die RAM-Kettung wird bei Magnetbandarbeit mit aufgezeichnet und bietet den Vorteil eines frei verschieblichen Ladens von Dateien unter Verwendung eines Relativladers und einer Verschiebeadresstabelle. Bei der Initialisierung (Kommando INIT) werden alle Segmente mit dem Kennbyte für freien RAM durchgekettet und vom letzten RAM-Segment auf die Adresse CØØØ (nicht segmentierter Bereich) geführt. Vom Betriebssystem aus wird die RAM-Kettung auf Adresse 4ØØØH im Segment 1 geführt.
Zur Segmentschaltung hat die RAM-Kettung folgenden Aufbau:

Kennbyte: ØFEH	Adresse L	Adresse H	Segmentnummer H	ØFFH

Als Segment ist einzutragen:
Segment 1 ---> 1
Segment 2 ---> 2
Segment 3 ---> 4
Segment 4 ---> 8
.
.
Segment 8 ---> 8ØH (letztes mögliches Segment)

Als Adresse ist die Adresse einzutragen, auf der das Kennbyte im neuen Segment stehen soll.

6. Tabelle der möglichen Kennbytes in der RAM-Kettung

ØØ freier RAM

Ø1 Assemblerprogramm ohne Parameterliste, über Namen aus dem Kommandoeingabemodus des BSYS startbar

Ø2 Treiberroutine mit IY-Schnittstelle

Ø3 namenlose Routine, die bei jedem RESET abgearbeitet wird

Ø4 Directory

Ø5 RAM-Teil von Ø1-Programm (z. B. Arbeitszellen)

Ø6 Binärdatei

Ø7 ASCII-Datei (z.B. Quelltext)

Ø8 BASIC-Prozedur, Zwischencode-Parameterliste oder Assemblerprogramm mit Parameterliste

Ø9 wie Ø8 aber als INTEGER-Prozedur

ØAH SHORT

ØBH REAL

ØCH LONG

ØDH BOOLEAN

ØEH POINTER

ØFH STRING

1ØH Verschiebeadresstabelle

11H RAM-Bereich für lokalen Stack

12H Markentabelle

13H Arbeitsbereich für Objektcodeeditor

14H numerisches Feld

15H boolean Feld

16H pointer Feld

17H string Feld

18H-1AH zur Zeit frei

1BH gesperrter Bereich für feste Programme

1CH BASIC-Bereich

1DH-1FH zur Zeit frei

2ØH-ØFDH nicht erlaubt (RAM-Kettungsfehler)

ØFEH Segmentumschaltung

ØFFH Ende RAM-Bereich

7. Betriebssystemkommandos

Das Betriebssystem meldet sich im Kommandoeingabemodus mit

OK
>_

in der letzten und vorletzten Displayzeile. Die Eingabe folgender Kommandos ist möglich:

INIT

Syntax: INIT ENTER / INIT "letztes zu init. Segment" ENTER

Aktion: löscht den gesamten RAM, schaltet alle Segmentmerkzellen auf Ø, kettet den RAM mit KBØ durch, wobei die Kettung vom letzten zu initialisierenden RAM-Segment auf Adresse CØØØH geführt wird; wenn nur 1 Segment vorhanden ist, dann sind keine Parameter erforderlich. Die Active Device Table wird angelegt. Nach INIT wird außerdem eine Initialisierung, wie nach RESET durchgeführt.

CAT

Syntax: "CAT" ENTER Anzeige aller Dateien mit Kennbyte Ø1 in der RAM-Kettung
"CATA" ENTER oder "CAT Ø" ENTER Anzeige aller Dateien in der RAM-Kettung
"CAT" "Kennbyte" ENTER Anzeige aller-Dateien mit angegebenen Kennbyte, die im Speicher stehen

Aktion: Die CAT-Kommandos besitzen rein informativen Charakter. Es erfolgt eine tabellarische Auflistung aller Dateien in der Reihenfolge ihrer Kettung. (Dies muß nicht mit der adressmäßigen Reihenfolge übereinstimmen, da eine Distanz auch scheinbar zurückführen kann.)
Sind mehr als 23 Dateien mit entsprechenden KB im Speicher enthalten, muß ENTER betätigt werden, um die nächsten Dateien zur Anzeige zu bringen. Eine Tabellenzeile enthält folgende Daten:

"Name" ....% "Kennbyte" % "Segment" . "Adr. des Kennbytes"
Bei"CATA"und "CAT Ø" werden auch die Kennbytes ØØ und ØFFH mit zur Anzeige gebracht.

RELAD

Syntax: "RELAD" "Name" ENTER

Aktion: Nachdem ein auf Ø gebundenes Programm mit dazugehöriger Verschiebeadresstabelle und eventuell vorhandenem RAM-Teil auf die gewünschte "ROM-Adresse" geladen wurde, werden bei Aufruf dieses Kommandos alle nicht verschieblichen Adressen (Direktsprünge, Unterprogrammaufrufe) gemäß der Verschiebeadresstabelle auf der geladenen Adresse lauffähig gemacht. Alle erforderlichen Parameter werden aus dem Namen bezogen (jeweils mit verschiedenen KB: Ø1 oder Ø2 für ROM-BASIC; Ø5 für RAM-BASIC, wenn vorhanden; 1ØH für Verschiebadresstabelle).

LADT

Syntax: "LADT" ENTER

Aktion: Das Kommando gibt in tabellarischer Form Auskunft über die in der Active Device Table stehenden Treiberroutinen. Eine Zeile enthält dabei folgende Informationen:

"Name" ...%"Segment" %"Adresse" "log. Nummer"

Die Informationen Segment, Adresse und Name sind identisch mit den in CAT verwendeten Begriffen. Die logische Gerätenummer kann durch DEFNE zugeordnet werden. Nach INIT stehen folgende Treiber in der ADT:

CON Konsolentreiber logische Nummern 1, 2, 3
KLOG Kassettentreiber logische Nummer 4

DEFINE

Syntax: "DEFINE" ENTER

Aktion: Rechner meldet sich "Name" ....> "alte log. Gerätenummer"
Eingabe der neu vereinbarten log. Gerätenummer, Bestätigung der alten log. Nummer mit ENTER oder Verlassen des Programmes mit OFF möglich. Nach jeder Eingabe wird die jeweils nächste Treiberroutine aus der ADT angeboten.

ACTIVATE

Syntax: "ACT" "Name 1", "Name 2", ... ENTER

Aktion: Die jeweils gewählten Treiber mit Kennbyte Ø2 in der RAM-Kettung werden in die ADT eingetragen und erhalten dabei die log.Nummer Ø. Ein Treiber kann auch mehrmals aktiviert werden. Anschließend wird LADT angesprungen.

DEACTIVATE

Syntax: "DEACT" "Name 1", "Name 2", ... ENTER oder
"DEACT" ENTER

Aktion: Ein Treiber mit dem angegebenen Namen wird aus der ADT gestrichen und das Tabellenende verschoben. Wird kein Name angegeben, bleiben nur die ersten 4 bei INIT eingetragenen Treiber erhalten. Danach erfolgt eine Anzeige der ADT.

DIRECTORY

Syntax: "DIR" ENTER

Aktion: Anzeige des Directorys (Kennbyte Ø4); bei mehr als 17 Positionen erneute Betätigung von ENTER.

Das Directory wird bei INIT auf die Adresse ØBAF8H gelegt.
Eine Zeile enthält folgende Informationen:
Name ... "Dateinummer auf Kassette"

RKET

Syntax: "RKET", "Segmentnummer", "Ausgangsadresse", "Zieladresse"

Aktion: Die RAM-Kettung wird von der Ausgangsadresse, die in der Kettung liegen muß, auf die Zieladresse geführt. Hat die Ausgangsadresse das Kennbyte ØFFH, wird dort freier RAM geschaffen. Das Programm ist nur innerhalb eines Segmentes anwendbar.

8. Softwareschnittstellen

Adr.	Kurzzeichen	Erläuterung
ØBBBH	RLL	Löschen eines RAM-Objektes INPUT : HL - Zeiger auf Kennbyte der zu löschenden Datei OUTPUT: HL - unverändert BC - unbestimmt Cy=1 und A=48 - falsches Kennbyte
ØBBEH	OUT	Ausgabe von Zeichen an die VIS - innerhalb einer Zeile ab Position INPUT : BC - max. Zeichenzahl DE - D-Zeile/E-Spalte HL - Textpufferanfang OUTPUT: BC - tatsächlich geschriebene Zeichenzahl DE - Position nach letztem geschriebenen Zeichen HL - Textpufferende
ØBC1H	WBN	Ausgabe von Zeichen an die VIS ab aktuellem Kursor - Abarbeitung von Steuerzeichen möglich Ø7 - "Piep" an Tastatur ØD - carriage return 19 - Rest der Zeile löschen INPUT : BC - max. Zeichenzahl HL - Textanfang OUTPUT: BC - tatsächlich geschriebene Zeichenzahl HL - Textpufferende DE - unverändert

ØBC4H RSF Freibereich in der RAM-Kettung suchen

INPUT : BC - Länge des gewünschten Freiraumes

OUTPUT: BC - unverändert
DE - unbestimmt
Cy = 1 ---> Fehler
A. - Fehlercode: Ø1 kein Freiraum
48 RAM-Kettung
HL - Zeiger auf Fehlerstelle im Datensegment

ØBC7H RSU RAM-Objektsuche

INPUT : DE - Zeiger auf Name des zu suchenden RAM-Objektes
DE = Ø nur Typ wird gesucht, kein Namensvergleich
A - Typ des gesuchten RAM-Objektes

OUTPUT: Cy = Ø Objekt gefunden
HL - Zeiger auf 2. Kennbyte
BC - Zeiger auf 1. Kennbyte
DE - unverändert
Cy = 1 Objekt nicht gefunden
HL - Zeiger auf Fehlerstelle
A - Fehlercode: C7 Objekt nicht gefunden
48 RAM-Kettung fehlerhaft

ØBCAH RSØ RAM-Objektsuche
wie RSU außer INPUT HL - Zeiger auf Kennbyte, ab welchem gesucht werden soll

ØBCDH TST Tastaturabfrage (polling)

INPUT : keine

OUTPUT: A - Tastencode
A = Ø keine Taste betätigt

CY	Z	Erläuterung
Ø	1	keine Taste betätigt
1	Ø	Taste neu gedrückt
1	1	Taste repetierend

ØBDØH	ZLE	Zahl dezimal oder hexadezimal vom Textpuffer lesen INPUT : A - Bit Ø ... 4 = Anzahl der zu lesenden Stellen, Bit 7 = 1: hexadezimal lesen, Bit 7 = Ø: dezimal DE - Textpufferadresse OUTPUT:A - letzte gelesene Ziffer DE - Position nach letzter Ziffer HL - gelesene Zahl BC - unverändert
ØBD3H	WLN	wie WBN - ohne Steuerzeichen
ØBD6H	BFE	Bildfenster festlegen INPUT : DE - Bildfenster D:Zeile/E: Spalte OUTPUT: keiner
ØBD9H	CLB	Bildschirm dunkel löschen INPUT : keine OUTPUT: keine
ØBDCH	ROL	Bild um eine Zeile nach oben rollen INPUT : keine OUTPUT: keine
ØBDFH	ROU	Bild um eine Zeile nach unten rollen INPUT : keine OUTPUT: keine
ØBE8H	LIY	Laden des IY-Vektors INPUT : A - Anforderungscode BC - Datenlänge HL - Datenstartadresse OUTPUT:IY+Ø1 - Anforderungscode IY+Ø2/Ø3 - Datenstartadresse IY+Ø4/Ø5 - Datenlänge
ØBEBH	HBS	Hexabyte auf Textpuffer schreiben INPUT : A - Byte HL - Textpufferadresse OUTPUT: HL - Textpufferende
ØBEEH	CON	Consolentreiber mit RIO-Schnittstellen

		INPUT : IY+Ø1 - Anforderungscode IY+Ø2/Ø3 - Datenstartadresse IY+Ø4/Ø5 - Datenlänge OUTPUT: IY+Ø4/Ø5 - tatsächlich übertragene Datenlänge IY+1Ø - Fertigstellungscode IY+Ø1H - Fehlercode
ØBF1H	NIXE	Ansprung des RIO-Treibers mit entsprechender logischer Gerätenummer INPUT : wie CON, aber in (IY+Ø) logische Gerätenummer, ADT OUTPUT: wie CON
ØBF4H	LOUT	Ausgabe einer Zeile auf das logische Gerät Nr. 3 über NIXE INPUT : HL = Datenstartadresse (Textpufferende ØDH/FFH) OUTPUT: HL zeigt auf Textende, DE, BC verändert
ØBF7	LZOU	Leerzeilenausgabe an logisches Gerät Nr. 3 über NIXE INPUT : keiner OUTPUT: Register unverändert

9. Aufbau des IY-Vektors

Der IY-Vektor stellt die einheitliche Schnittstelle zu allen Treibern dar. Damit wird die Geräteverwaltung RIO-kompatibel. Der RIO-Vektor hat folgenden Aufbau:

IY	logische Gerätenummer (von 1 bis 14 H)
IY+1	Request-Code
IY+2/3	Datenstartadresse
IY+4/5	Datenlänge
IY+6/7	Rückkehradresse (vom Treiber verwendet)
IY+8/9	ERROR-RETURN-ADRESS (vom Treiber verwendet)
IY+ØAH	Fehler- oder Fertigstellungscode
IY+ØBH/ØCH	Folge-Vektor-Adresse bei benötigter Zusatzinformation

10. Arbeit mit dem zentralen Treiberverwaltungsprogramm "NIXE"

Das Treiberverwaltungsprogramm arbeitet mit dem RIO-Vektor. Es wird nur (IY+Ø) ausgewertet. Aus der logischen Gerätenummer und den in der ADT enthaltenen Informationen über Segment und Adresse des Treibers wird der Treiber aufgerufen, der der entsprechenden logischen Nummer mit DEFINE zugeordnet wurde. Die möglichen Request-Codes in IY+1 werden weiter gegeben, unabhängig davon, ob der Request erlaubt ist, oder nicht. Diesesmuß im Treiber selbst entschieden werden. Ist der gewünschte Request im gewählten Treiber abgearbeitet, erfolgt eine Rückkehr über das Treiberverwaltungsprogramm in den Handler oder ein übergeordnetes Hauptprogramm. Diese muß die Fehlerauswertung durchführen. Die Fehlerauswertung geschieht nach folgendem Schema:

```
        LD    A, (IY+ØAH)      ; Rückmeldung Fehlercode von
                               ; Treiber
        CMP   8ØH
        JRZ   WEIT             ; Fertigstellungscode, fehler-
                               ; frei mit gesetztem CY und
        SCF                    ; sauberen Stack
        :                      ; Rückkehr in das BSYS
        RET                    ; Fehlercode in A wird ange-
                               ; zeigt
WEIT:   :
        :
```

Es sind folgende Request's möglich:

CON:	Ø	Initialize Request
	ØAH	READ BINARY
	ØCH	READ LINE
	ØEH	WRITE BINARY
	1ØH	WRITE LINE
	4ØH	READ STATUS
	42H	WRITE STATUS
	48H	INPUT

KLOG: ØAH READ BINARY
ØEH WRITE BINARY

Grundlage für das Auffinden des Treibers mit der gewünschten log. Nummer ist die Active Device Table. Sie ist als 4-Byte-Tabelle aufgebaut und steht fest im schreibgeschützten, unsegmentierten ZVE-RAM auf der Adresse FCØØH.

Byte 1 : Bit 7 = Aktivbit (immer gesetzt)
Bit 6, 5 frei
Bit 4..Ø log. Gerätenummer
Byte 2 : Segmentnummer des Treibers
Byte 3/4 : Adresse des Treibers (zeigt auf 1 Kennbyte)

Als Tabellenende steht eine Ø.

11. Arbeit mit der Segmentierung und den Segmentumschaltprogrammen

Bei Verwendung mehrerer Leiterplatten vom Typ "SPE" ist eine Segmentierung des verwendeten Speicherbereiches möglich, die das BSYS unterstützt.

Der segmentierbare Speicher hat die Adresse 4ØØØ..BFFFH (32 kByte). Neben dem immer vorhandenen nichtsegmentierten ZVE-RAM kann nur jeweils 1 Segment eingeschaltet werden. Das Schalten der Segmente erfolgt über den Befehl OUT 85H, wobei in A die Segmentnummer steht. In A darf dabei nur 1 Bit gesetzt sein (siehe auch RAM-Kettung).
Bei der Arbeit im segmentierten RAM ist darauf zu achten, daß zum Schluß das Ausgangssegment wieder eingeschalten wird. Diese wird vorher auf der Merkzelle "Maschinencodesegment" SEGC FBFDH abgespeichert. Für die Arbeit im segmentierten Bereich stehen spezielle Segmentbefehle zur Verfügung, die besonders vom BASIC benutzt werden, aber auch für Anwender-

programme nutzbar sind.

a) Sprungbefehle in den segmentierten Speicherbereich

```
2ØØ4H JHL    JMP(HL) mit Segmentnummer in A (SEGC) wird A
2ØØ7H JHD    JMP(HL) mit Segmentnummer in (SEGD), (SEGC)
             wird (SEGD)
2ØØAH JPD    JMP (SP) mit Segmentnummer in (SEGD)
```

b) CALL-Befehle in den segmentierten Speicherbereich

```
im Stack steht:   SP      Adresse RET-Programm
                  SP +2   frei
                  SP +3   Rückkehrsegmentnummer --->(SEGC)
                  SP +4/5 Rückkehradresse
2ØØDH CHL         CALL(HL) mit Segmentnummer in A; (SEGC)
                           wird A
2Ø1ØH CHD         CALL(HL) in (SEGD)
2Ø13H CAD         CALL(SP) in (SEGD)
```

c) Ladebefehle (Blockbefehle) aus Datensegment (SEGD) in Externsegment (SEGE); E/A wie LDIR/LDDR

```
2Ø37H LDIR
2Ø3AH LDDR
```

d) Akku-Ladebefehle aus- und in den segmentierten Bereich

```
2Ø3DH LØL   LD A,M      }
2Ø4ØH LØE   LD A,(DE)   }
2Ø43H LØX   LD A,(IX)   }   Segmentnummer vorher im Akku
2Ø46H LØY   LD A,(IY)   }
```

```
2Ø49H L1L   LD A,M      }
2Ø4CH L1E   LD A,(DE)   }
2Ø4FH L1X   LD A,(IX)   }   Segmentnummer steht in (SEGD)
2Ø52H L1Y   LD A,(IY)   }
```

```
2Ø55H   L2L    LD A, M      }
2Ø58H   L2E    LD A, (DE)   }
2Ø5BH   L2X    LD A, (IX)   }   Segmentnummer in (SEGE)
2Ø5EH   L2Y    LD A, (IY)   }

2Ø61H   S1L    LD M,A       }   in SEGD zu laden
2Ø64H   S1X    LD (IX), A   }
2Ø6AH   S2L    LD M, A      }   in (SEGE) zu laden
2Ø6DH   S2X    LD (IX),A    }
```

Die Speicherzellen für die Segmentnummern liegen auf folgenden Adressen:

```
SEGC:   ØFBFBH   Maschinencodesegment
SEGD:   ØFBFEH   Datensegment
SEGE:   ØFBFFH   Externsegment
```

12. Erstellen einer Verschiebeadresstabelle und Arbeit mit dem Relativlader

Im Betriebssystem existiert ein Unterprogramm zur Erstellung von Verschiebeadresstabellen (VERA: 229ØH).
Das Programm, zu dem eine Verschiebeadresstabelle erstellt werden soll, muß zu diesem Zweck zweimal im Speicher vorliegen:

- ROM-Teil gebunden auf Ø auf Adresse 1 stehend
  RAM-Teil gebunden auf 8ØØØH auf Adresse 2 stehend } Programm1

- RAM- und ROM-Teil auf verschiedene, beliebige andere Adressen gebunden und auf Adr. 3 bzw. 4 stehend } Programm 2

Die Programme dürfen nur aus einem Programmblock (ROM-Teil) und einem davon abgesetzten RAM-Teil (z. B. Arbeitsbereich bei ROM-Programmen, Textpuffer, Datenspeicherbereiche usw.) bestehen. Eine Verbindung zu anderen Programmen ist nicht

möglich! Sollen die Programme nur im RAM lauffähig sein, kann auf einen RAM-Teil verzichtet werden, wenn sie mit dem Objektcodeditor unter Nutzung der Möglichkeit der Vereinbarung externer Marken erstellt wurden. Damit vereinfacht sich das Verfahren wesentlich. Die Verschiebeadresstabelle ist nach folgendem Muster zu erstellen:

```
PUSH    IY      ;(Rettung localer Stack)
LD      IX      ;Programm A (Adresse 1)
LD      IY      ;Programm B (Adresse 3)
LD      HL      ;Verschiebeadresstabelle-Zieladresse
PUSH    HL      ;(diese Adresse wird als (SP) übergeben)
LD      BC      ;Programmlänge
LD      HL      ;ROM-Bindedistanz
LD      DE      ;RAM-Bindedistanz
CALL    VERA
POP     HL      ;Endadresse Verschiebeadresstabelle
POP     IY      ;localen Stack gerettet
RET
```

Alle Adressen beziehen sich auf den Programmanfang, nicht auf die Position des 1. Kennbytes. Die Verschiebeadresstabelle wird ohne Kennbyte erstellt, sie muß das Kennbyte 1Ø, Distanz und den Namen des Programmes erhalten. Ausgangspunkt für das Kommando "RELAD" ist das auf Ø gebundene Programm mit Kopf und die Verschiebeadresstabelle mit Kopf. Nach dem Einlesen dieser beiden Komponenten vom Magnetband auf beliebigen freien RAM wird das Programm auf die Ladeadresse lauffähig, wenn RELAD "programmname" aufgerufen wurde.

13. Die RESET-Initialisierung mit Kennbyte Ø3

Bei Betätigung von RESET arbeitet das Betriebssystem nacheinander alle Routinen mit Kennbyte Ø3 , beginnend am Anfang der RAM-Kettung, ab. Die RESET-ROUTINEN müssen als Unterprogramm geschrieben sein. IY muß unbedingt gerettet sein.

Die Rückkehr muß mit gelöschtem CY-Flag erfolgen. RESET-Routinen können Anwendung finden für die Initialisierung von zusätzlichen Geräten oder für die Ausschrift eines eigenen Initialisierungtextes auf dem Display. Sie stehen in der RAM-Kettung mit Kennbyte Ø3 und dürfen keinen Namen erhalten. Augenmerk ist darauf zu richten, daß keine falsch gebundenen oder fehlerhaften [illegible]-Routinen in der RAM-Kettung stehen. In diesem Falle wäre nur durch Ausschalten des Rechners und erneutes Einschalten eine Fortsetzung der Arbeit möglich, da durch Betätigung von RESET diese fehlerhafte Routine immer wieder abgearbeitet würde. Dieser Effekt würde auch bei einem RST Ø oder JMP Ø am Ende der Routine auftreten.

14. Die Arbeit mit den Sondertasten (8-ter-Feld)

Das Tastaturprogramm ist so ausgelegt, daß der Anwender diese Sondertasten selbst mit Funktionen belegen kann. Die Tasten liefern die Codes zwischen Ø und 7. Bei Betätigung einer dieser Tasten wird von der SIO auf der ASP ein INT ausgelöst. In der INT-Routine wird getestet, ob auf der Ausprungstelle für Sondertasten ein C3 (JMP nn) steht. Ist das nicht der Fall, wird die Betätigung übergangen, anderenfalls die nachfolgend eingetragene Adresse angesprungen. Der Anwender hat die Fortführung der Tastaturinterruptroutine als Unterprogramm zu schreiben, d. h., mit RET abzuschließen. Die INT-Routine muß kurz (L 1Øms) sein, da während dieser Zeit alle weiteren INT gesperrt sind. Somit ist während der Abarbeitung dieser Routinen auch keine Tastaturbetätigung möglich. Für längere Anwenderprogramme ist es empfehlenswert, mittels dieser kurzen INT-Routinen niedriger priorisierte INT-Routinen (z. B. CTC) anzustoßen.

Tastencode	Adresse für Anwender
ØØ	ØFEDØH C3 nn nn
Ø1	ØFED3H C3 nn nn
Ø2	ØFED6H C3 nn nn
Ø3	ØFED9H C3 nn nn
Ø4	ØFEDCH C3 nn nn
Ø5	ØFEDFH C3 nn nn
Ø6	ØFEE2H C3 nn nn
Ø7	ØFEE5H C3 nn nn

Mikrocomputer-Gerätegeneration

M C 80.3x

-B E T R I E B S S Y S T E M-

Magnetbandtreiberroutine

(EGOS 30.1 - Teil 2)
Systembeschreibung

Z.-Nr. 50300.4012.00 BA

VEB Elektronik Gera
Ausgabe 08/85

Der Komplex Magnetbandprogramme beinhaltet 3 Komponenten:

- physische Treiberroutinen
- logische Treiberroutinen
- Bedienprogramm zum Schreiben und Lesen von Programmen.

1. Physische Treiberroutinen

Die physischen Treiberroutinen realisieren die elementaren Grundfunktionen, die für die Arbeit mit Magnetband notwendig sind. Diese sind:

- Schreiben eines Blockes auf Magnetband
- Lesen eines Blockes vom Magnetband
- Schreiben einer Bandmarke
- Schreiben einer Schlußlücke
- Suchen einer Bandmarke vorwärts
- Suchen einer Bandmarke rückwärts
- Umspulen an den Kassettenanfang

Genannte Treiber sind für die Laufwerkstypen LW 1200 und K 5200 realisiert. Zusätzlich existieren für das Laufwerk K 5200 noch folgende Funktionen:

- einen Block vorsetzen
- einen Block rücksetzen
- n-Blöcke vorsetzen
- n-Blöcke rücksetzen

Alle Treiber benutzen eine einheitliche Schnittstelle, deren Adresse sich im IX-Register befinden muß.

(IX)	- Status
(IX+1)	- Fehlercode, falls Fehlermeldung
(IX+2)	- Basisadresse
(IX+3	- Subadresse (1 oder 2)
(IX+4)	- Kommando
(IX+5,6)	- Adresse Eintrittspunkt
(IX+7,8)	- Blockadresse
(IX+9)	- H- Teil Blocklänge
(IX+1Ø)	- L- Teil Blocklänge oder n bei Bandmarken suchen oder n bei Blöcke vor-/rücksetzen

Kommando (IX+4)

%ØØ - Schreiben eines Blockes mit RAW
%Ø2 - Lesen eines Blockes
%11 - einen Block vorsetzen (K 5200)
%15 - einen Block rücksetzen (K 5200)
%21 - Rückspulen an Bandanfang
%31 - Reservieren ein
%41 - Reservieren aus
%51 - Schreiben einer Bandmarke
%61 - Schreiben einer Schlußlücke
%71 - Suchen n-te Bandmarke vorwärts
%75 - Suchen n-te Bandmarke rückwärts

Status (IX)
Bit Ø: Ø - Gerät frei
1 - Gerät besetzt

Bit 1: Ø -
1 - Datenträgerende
Bit 2: Ø - Schreiben erlaubt
1 - Schreiben verboten
Bit 3: Ø -
1 - Datenträgeranfang
Bit 4: Ø - Gerät bereit
1 - Gerät nicht bereit
Bit 5: Ø - Gerät nicht reserviert
1 - Gerät reserviert
Bit 6: Ø - 38 cm/s (nur K 5200)
1 - 19 cm/s (nur K 5200)
Bit 7: Ø - Kommando wurde fehlerfrei abgearbeitet
1 - Fehler (Fehlerart ⟶ (IX+1))

Fehlercodes

%1Ø - unerlaubter Kommandocode
%11 - Gerät nicht reserviert
%12 - unerlaubte Laufwerksnummer
%13 - unerlaubte Blocklänge
%14 - Aufzeichnungsende erreicht beim Lesen
%15 - Gerät besetzt - arbeitet noch
%16 - Aufzeichnungsende oder -anfang erreicht beim Bandmarken suchen
%17 - unerlaubte Blocklänge beim Lesen
%18 - Bandmarke erkannt aber nicht gefunden beim Kontrollesen
%19 - Aufzeichnen verboten
%1A - Gerät nicht mehr bereit
%21 - Fehler nach festgelegter Anzahl Lesewiederholungen
%22 - Fehler nach festgelegter Anzahl Schreibwiederholg.
%23 - kein Echo vom Eingabekanal RAW

2. Logische Treiberoutinen

Alle Treiberroutinen benutzen eine einheitliche Schnittstelle, deren Adresse im IY-Register steht. Alle notwendigen Daten werden über diesen Vektor übergeben, der folgenden Aufbau hat:

(IY) - Nummer der logischen Einheit
alle logischen Einheiten 1-2Ø können angegeben werden

(IY+1) - Anforderungscode
identifiziert die gewünschte I/O-Operation
Im MC 80.3x sind folgende I/O-Operationen realisiert:
- INITIALIZE (%ØØ)
- OPEN FOR NEW FILE (%Ø4)
- OPEN FOR INPUT (%Ø4) (Spezifizierung durch Folgevektor)

- CLOSE (%Ø6)
- READ BINARY (%ØA)
- WRITE BINARY (%ØE)

(IY+2) - Datenstartadresse
(IY+3) - Gibt die Startadresse der zu übertragenden Daten an, %ØØØØ, wenn keine Datenübertragung erforderlich ist.

(IY+4) - Datenlänge
(IY+5) - Gibt die Byteanzahl der zu übertragenden Daten an, %ØØØØ, wenn keine Datenübertragung erforlich ist. Bei Rückkehr wird die Anzahl der tatsächlichen übertragenen Daten angegeben.

(IY+6) - Rückkehradresse
(IY+7) - wird verwendet, wenn Treiber im Interruptbetrieb arbeitet (BIT Ø, (IY+1) ist 1).

(IY+8) - Fehlerrückkehradresse
(IY+9) - Rechner springt nach Abarbeitung der I/O-Operation auf die angegebene Adresse. %ØØØØ, Fehlerrückkehradresse wird übergangen.

(IY+1Ø) - Fertigstellungscode
wird nach Beendigung der Operation mit einem Code geladen, der die Fertigstellung, bzw. Fehler anzeigt.

(IY+11) - Folgevektor
(IY+12) - 2 Byte Zusatzinformation oder Adresse, die auf Zusatzparameter zeigt.

3. Bedienprogramm

Das Bedienprogramm realisiert das Formatieren, Schreiben und Lesen aus dem Kommandomodus des Betriebssystems heraus. Für eine richtige Abarbeitung des entsprechenden Kommandos ist die Angabe von Parametern notwendig, die nach folgendem Schema eingegeben werden müssen.
Die Unterbrechung der Magnetbandarbeit mit "RESET" oder Abschalten des Gerätes kann wegen der hohen Bandgeschwindigkeiten zur Zerstörung des Magnetbandes der Kassette führen. Aus diesem Grunde sollte die Beendigung der Magnetbandarbeit abgewartet werden.

FORMAT lwnr
Formatieren einer leeren oder zu überschreibenden Kassette. Das Programm schreibt ein leeres Directory auf Kassette.

READ lwnr, Dateiname
Das Programm liest eine Datei von Kassette in den Arbeitsspeicher. Zur Identifizierung und zum Suchen der Datei auf der Kassette wird der Dateiname verwendet.

READ lwnr, Dateiname, R
Das Programm liest eine Prodzedur in den Arbeitsspeicher,die nur auf einer absoluten Adresse lauffähig ist. Die RAM-Kettung wird auf die Anfangsadresse der Prozedur geführt.

ACHTUNG!
Dateien, die den zu überschreibenden bzw. nachfolgenden Speicherbereich nutzen, können zerstört werden bzw. sind nicht mehr in der RAM-Kettung enthalten.

WRITE lwnr, Dateiname
Schreibt einen im Arbeitsspeicher vorhandenen, durch Kennbytes gekennzeichneten Arbeitsbereich auf Kassette.

ACHTUNG! Es darf nur ein Arbeitsbereich im RAM existieren.

WRITE lwnr, Dateiname, typ
Schreibt den Arbeitsbereich, der durch 'typ' spezifiziert ist auf Kassette.

WRITE lwnr, Dateiname, seg, rama, rame
Schreibt den Inhalt des angegebenen RAM-Bereiches auf Kassette.

ACHTUNG! Diese Dateien können nur auf der Adresse wieder eingelesen werden, ab der sie auch ausgelagert wurden.

lwnr - Laufwerksnummer (1 oder 2, kann weggelassen werden, dann gilt lwnr = 1)

Dateiname - dient zur Identifizierung der Datei auf Kassette

typ - Typ des auszulagernden Arbeitsbereiches
A - Assemblerarbeitsbereich (Kennbyte %13)
B - BASIC-Arbeitsbereich (Kennbyte %1C)

seg - RAM-Segmentnummer
rama - Datenstartadresse im RAM
rame - Datenendadresse im RAM

4. Fehlercodes

Code	Bedeutung
%4Ø	Treibername ungültig
%41	Gerät ungültig oder inaktiv
%42	logische Funktion ungültig
%43	Speicherschutz verletzt
%44	Operand fehlend oder ungültig
%45	Systemfehler
%46	Dateiname unerlaubt
%47	Kommando existiert nicht
%48	unerlaubter Dateityp
%49	Programmabbruch
%4A	Speicher nicht verfügbar
%4B	Dateieigenschaften fehlend oder ungültig
%4C	Ein-/Ausgabefehler

%8Ø	Operation fehlerfrei
%81	Formatfehler im Directory
%82	Scratch-datei eröffnet
%83	Dateiname zu lang, wird abgeschnitten
%84	Merkmalliste zu lang, wird abgeschnitten
%C1	Operation ungültig
%C2	Gerät nicht bereit
%C3	Datei ist schreib- und löschgeschützt
%C5	Suchfehler
%C6	Datenübertragungsfehler
%C7	Datei nicht gefunden
%C9	Dateischlußfehler (EOF)
%CB	Datei noch nicht eröffnet
%CC	logische Funktion ist bereits aktiv
%CD	Zuweisungsspeicherpuffer voll
%CE	Laufwerk ungültig
%CF	Tabelle der logischen Funktionen voll
%DØ	Dateiname doppelt
%D1	Identifizierungsfehler Kassette
%D2	Merkmale ungültig
%D3	Kassette voll
%D4	Datei nicht im Directory
%D5	Dateianfangsfehler
%D6	Datei von anderer log. Funktion schon eröffnet
%D7	Umbenennung der Scratch-Datei ungültig
%D8	Merkmalsänderung gesperrt
%D9	ungültiger OPEN-Aufruf
%DA	Speicherplatz für den Kassettenbelegungsplan reicht nicht aus.