MICROSOFT ist eingetragenes Warenzeichen der Microsoft Corporation

SVI ist eingetragenes Warenzeichen der Spectravideo International Ltd.

Einleitung

Diese Zusammenfassung richtet sich an Alle,die mit der Maschinensprache der Z80-CPU vertraut sind und diese auf dem SVI-Computer anwenden wollen.

Diese Zusammenfassung sei als Anregung und erste Hilfe fuer den unersaettlichen Systembastler gedacht. Es wird noch genug Freiraum geboten, eigene Nachforschungen im ROM anzustellen. In der einschlaegigen Literatur sowie in den bekanntesten Zeitschriften finden sich heufig Artikel, die sich auf das MICROSOFT-BASIC eines anderen Rechners beziehen. In der Regel koennen die dort beschriebenen Dinge (bis auf die unterschiedlichen Adressen) auch auf die SVI-Rechner uebertragen werden.

Da es zu aufwendig ist,jede der beschriebenen Routinen oder System-Variablen auch mit Beispielen zur Benutzung zu versehen, wird auf die Cross-Referenz und das ROM-Listing verwiesen.

1. Basis-Routinen

Einige grundlegende Routinen des BASIC-ROMs werden hier mit Eingangs- und Ausgangsparametern beschrieben. Wenn nicht anders erwaehnt, werden die Register nicht veraendert.

1.1 Wichtige CPU-Einspruenge

Einige dieser Routinen koennen dazu dienen, die CPU-Instruktionen auf Basis der SVI-Hardware zu erweitern.

Adresse Beschreibung

0 Dieses ist die Kaltstart-Adresse fuer den Rechner

20 Simulation von CP HL,DE. Die Flags sind wie bei HL-DE gesetzt und A ist veraendert.

Z	bei HL = DE
NZ	HL <> DE
C	HL < DE
NC	HL >= DE

38 Interrupt-Einsprung fuer die Z80 im IM 1. Hier wird die Tastatur zeilenweise auf Tastendruck geprueft. Falls eine Taste gedrueckt ist, wird das Zeichen berechnet und in die Tastaturschlange geschrieben.

Zu den folgenden vier Routinen mit Bank-Umschaltung sei der Vollstaendigkeit halber erwaehnt, dass natuerlich nur x2 Speicherbanken angesprochen werden koennen, da das ROM sich in der unteren Speicherhaelfte befindet.

5F JMPBNK In eine neue Speicherbank springen. B enthaelt das Byte fuer Port 8C, HL die Adresse in der Bank. Die Interrupts sind in der neuen Bank freigegeben. Nur A ist veraendert.

62 CALBNK Aufrufen einer Routine in einer anderen Bank. Parameter wie 5F. Nach der Umschaltung, aber noch vor dem CALL, wird BC auf den Stack geschrieben und nach dem RETurn aus der Routine wieder restauriert.

126 PUTBNK LD (HL),A aber ueber verschiedene Baenke. Port 88 muss OFH enthalten und die Interrupts muessen abgeschaltet sein. Parameter: HL Zieladresse in der anderen Bank, A Byte, B enthaelt die gewuenschte, C die eigene Bank.

129 GETBNK LD A,(HL) sonst wie 126/PUTBNK

66 NMI Einsprung. Dieser Einsprung ist nicht benutzt.

399F INDJMP Indirekter Sprung ueber Tabelle. HL Tabellenadresse-2, A gesuchtes Byte, C Anzahl Tabelleneintraege. Tabellenadresse wird angesprungen, wenn A gleich dem Tabelleneintrag ist. Die Tabellenadressen sind Unterprogramme (d.h. Beenden mit RET). Ruecksprung zum Aufrufer von INDJMP mit gesetztem Zeroflag, wenn der

Eintrag gefunden wurde.
Tabellenaufbau:
Byte
Routinenadresse
... (C mal)

3EB8 Sprung ueber Tabelle. Tabellenaufbau wie oben. Ansprung der Routine, wenn Tabelleneintrag groesser als A ist. A darf nicht OFFH sein, sonst kein Abbruch moeglich. HL enthaelt Tabellenadresse.

1.2 Tastatur-Routinen

3B CHSNS Tastaturabfrage NZ Taste wurde gedrueckt. Nur A und F werden veraendert. Diese Routine zeigt abhaengig von SHIFT die Funktionstastenbelegung an. Die eigentliche Abfrageroutine ist auf Adresse 3DC2. Nach dieser Routine zeigt HL auf das eingelesene Zeichen, falls eines vorhanden ist, sonst auf die Position, auf der das naechste Zeichen abgelegt wird.

3E CHGET Zeichen von der Tastatur einlesen. Ist noch keine Taste gedrueckt, wird darauf gewartet. Das Zeichen steht in A. Nur AF wird veraendert.

5C BREAKX CTRL-STOP auf Tastatur testen. CARRY wenn gedrueckt. Falls die Tasten gedrueckt, sind wird der Tastaturpuffer geloescht.

16E CKCNTC prueft CTRL-STOP und STOP. Falls CTRL-STOP wird das Programm angehalten und SAVTXT auf 0 gesetzt.

1.3 Videospeicher-Routinen

3526 VDPWRT Byte B ins VDP-Register C schreiben

372A WRTVDP Byte A auf Adresse HL in den Videospeicher schreiben.

3734 RDVDP Ein Byte nach A aus dem Videospeicher von Adresse HL lesen.

373C SETWRT HL als Schreibadresse fuer den Videospeicher setzen.

3747 SETRD HL als Leseadresse fuer den Videospeicher setzen.

47 INITXT SCREEN 0 initialisieren.

4A INIGRP SCREEN 1 initialisieren.

4D INIMLT SCREEN 2 initialisieren.

37D9 CHGMOD In SCREEN steht der gewuenschte Bildschirm-Modus.

Entsprechend diesem Modus wird der Videospeicher initialisiert.

1.4 Bildschirm-Routinen

35C5 GETPAT Zeichengeneratormaske aus der Tabelle fuer ASCII-Zeichen A expandieren. Die expandierte Maske (8 Bytes) steht in PATWRK.

394D CHPUT Ausgabe eines ASCIIZeichens in A auf den Bildschirm in jedem SCREEN-Modus mit Interpretation der ESC-Folgen.

147 BEEP BASIC BEEP Anweisung

14A CNVCOD ASCII-Zeichen in A in Videocode wandeln. Videocode steht in A, C und F werden benutzt.

153 GETCOD ASCII-Zeichen vom 40-Zeichen- Bildschirm lesen. HL Spalten- und Zeilenadresse des gewuenschten Zeichens. Die Invers-Information geht verloren.

14D GETLEN Zeilenzahl des Bildschirms nach A laden (24 oder 23)

18 OUTCHAR Ausgabe eines Zeichens auf dem aktuellen Geraet bzw. Datei. Das auszugebende Zeichen steht in A, es werden keine CPU- Register veraendert. Das aktuelle Geraet ist eine Datei wenn PTRFIL <> 0 ist (d.h. eine Dateipufferadresse enthaelt) oder der Drucker wenn ODEVLINK<>0 ist.

6D13 PINLIN Einlesen einer Zeile vom Bildschirm wenn PTRFIL=0, sonst aus dem angegebenen Dateipuffer.

7B CRDO CR LF mit RST 18H ausgeben. Falls der Drucker das aktuelle Geraet war, wird LPOS zurueckgesetzt.

7E CRDONZ Falls der Cursor des Ausgabegeraetes nicht am Zeilenanfang steht, wird CR LF mit der 7B Routine ausgegeben.

DB LINPRT Ausgabe von HL als INTEGER-Zahl mit RST 18H

1B05 LISPRT Ausgabe eines Textes bis zum Zeichen 0. Die Adresse des ersten auszugebenden Zeichens steht in HL.

393E POSIT Cursor auf dem Bildschirm positionieren. HL enthaelt die Cursor-Adresse. L die Zeile (1-24), H die Spalte (1-39/40/80). Veraendert wird nur F.

A8 FINLPT Der Bildschirm wird zum Ausgabegeraet. Falls LPOS <> 0, wird CR LF ausgegeben.

15C FINPRT Bildschirm als Ausgabegeraet setzen.

150 GETTRM Begrenzung zur Zeile L nach A holen. 0 wenn Zeile auf dem Bildschirm laenger als 39/40 oder 80 Zeichen.

156 SETTRM Begrenzung A zur Zeile L setzen.

159 TERMIN Begrenzung zur Zeile L auf OAFH setzen.

1.5 Kasettenrecorderroutinen

6C CASIN Byte von Kasette nach A lesen. Ist CTRL-STOP gedrueckt, oder der Kasettenrecorder abgeschaltet, gibt es DEVICE IO ERROR.

75 CASOUT Byte aus A auf Kasette schreiben. CTRL-STOP Behandlung wie 6C.

72 CWRTON Kasettenrecorder einschalten zum Schreiben mit Text "Press play and.."

78 CTWOFF Kasettenrecoreder abschalten nach Schreiben von Bytes. Hier wird ein 0 Byte zusaetzlich geschrieben.

69 CSROON Kasettenrecorder zum Lesen einschalten mit Text "Press play on tape"

6F CTOFF Kasettenrecorder abschalten nach Lesen.

1.6 Drucker-Routinen

41 CHPSTT Test, ob Drucker fuer naechstes Zeichen bereit ist. NZ Drucker ist bereit. Nur AF werden veraendert.

44 CHPLPT Ausgabe eines Zeichens auf den Drucker. Ist dieser noch nicht bereit, wird gewartet. Auch CTRL-STOP wird geprueft. 3921 gibt ein Zeichen in A ohne Bereittest auf den Drucker.

81 OUTDLP Ausgabe eines Zeichens auf den Drucker mit Interpretation der Steuerzeichen.
TAB wird mit Leerzeichen auf 8ter Tabulation expandiert.
CR setzt LPOS auf 0. US erhoeht LPOS um 1. Alle anderen Steuerzeichen haben keine weitere Bedeutung und werden unveraendert ausgegeben.

1.7 Funktionstasten

59 RSTFNK Funktionstasten initialisieren mit STD-Belegung.

50 FNKSB Funktionstasten anzeigen, wenn SCREEN 0 und CNSDFG<>0. Die Anzeigeroutine ist 56.

53 ERAFNK Funktionstastenzeile loeschen. Bildschirm hat dann 24 Zeilen.

56 DSPFNK Funktionstasten abhaengig von SHIFT anzeigen.

1.8 Traps

6653 INITRP Initialisieren aller Traps

65EB ONTRP trap ON ausfuehren. HL enthaelt Adresse des Trap-Blocks. Zu jedem TrapBlock gehoeren 3 Bytes. Das erste Byte ist das Trap-Flag, Byte 2 und 3 werden als Wort interpretiert und geben die binaere Zeilennummer fuer GOTRP an. (Trap-Bloecke: TRPTBL fuer Funktionstasten ..._OOS fuer den Rest)

65FB OFFTRP trap OFF ausfuehren. HL wie ONTRP.

6601 STPTRP trap STOP ausfuehren. HL wie ONTRP.

660E RSTTRP trap wieder einschalten nach STPTRP.

6618 REQTRP trap request. HL wie ONTRP. Bei naechster Gelegenheit wird der Trap angesprungen.

666E GOTRP Trap suchen, der ausgefuehrt werden muss.
Der Trap wird freigegeben und ein STPTRP ausgefuehrt.
Danach wird die Routine angesprungen.

1.9 Grafik

48A1 SCALXY Prueft, ob die Koordinaten x in BC und y in DE innerhalb des darstellbaren Bereichs liegen. HL wird nicht veraendert. Sind die Koordinaten kleiner 0, ist das Ergebnis 0. Sind die Koordinaten > maxx(255) bzw. maxy(191), dann werden diese Werte ausgegeben.

48E9 MAPXYC x in BC und y in DE werden in Zeichenkoordinaten gewandelt. Das Ergebnis ist eine Maske in CMASK und eine Zeichenadresse in CLOC. Diese beiden Speicherzellen bilden den Grafik-Cursor.

4943 FETCHC holt die Zeichenkoordinaten CLOC nach HL und CMASK nach A.

494A STOREC speichert A und HL als Zeichenkoordinaten.

4951 READC Farbe des Bildpunktes lesen, auf dem der Grafik-Cursor steht. Die Farbe steht in A.

4980 SETATR Farbe fuer die folgenden SETC- Operationen setzen. Die Farbe steht in A. Ist A>15 RETURN mit CARRYflag, sonst Abspeichern in ATRBYT

4988 SETC Farbe eines Bildpunktes setzen. Die Farbe steht in ATRBYT, die Koordinaten in CLOC/CMASK.

49CF RIGHTC Grafik-Cursor CLOC/CMASK ein Pixel nach rechts bewegen.

49F8 LEFTC Grafik-Cursor ein Pixel nach links bewegen.

4A14 TDOWNC Grafik-Cursor ein Pixel nach unten mit Bereichstest

4A2D DOWNC wie TDOWNC, jedoch ohne Bereichstest

4A3F TUPC wie TDOWNC, jedoch nach oben

4A59 UPC wie DOWNC, jedoch nach oben

243C Ziehen einer Linie von (BC,DE)(GXPOS,GYPOS). Beides sind normale Koordinatenangaben (0255,0191).

Die restlichen Routinen fuer Grafik eignen sich nicht fuer AssemblerProgrammierung, da bei allen die Interpretation zwischen den Ausfuehrungsteilen fortgefuehrt wird. Es sei hier auf die BASIC-Schluesselwortliste bzw. auf das ROMListing ab Adresse 2320 verwiesen.

1.10 Sprites

36BE CLRSPR Alle Sprites loeschen.

Fuer weitere Routinen gilt das unter 1.9 Grafik Gesagte. Sprite-Routinen befinden sich im ROM ab Adresse 45D0.

1.11 Arithmetik

1.11.1 Doppelt genaue Funktionen

4D86 DECSUB FACCU:=FACCU-ARG

4D94 DECADD FACCU:=FACCU+ARG

4EFE DECMUL FACCU:=FACCU*ARG

4FB7 DECDIV FACCU:=FACCU/ARG

5F05 DBLEXP FACCU:=FACCU^ARG

50B8 COSinus(FACCU)

50D1 SINus(FACCU)

5120 TANgens(FACCU)

5139 ATN arcus tangens (FACCU)

5197 LOG natuerlicher Logarithmus von FACCU

5222 SQR Wurzel aus FACCU

526B EXP e potenziert mit FACCU

5300 RND(FACCU)

56AE DCOMP Vergleich SGN(FACCU-ARG)

FACCU > ARG : A=255
FACCU < ARG : A=1
FACCU = ARG : A=0

5376 MAF LD ARG,FACCU

5379 MAM LD ARG,(HL)

5382 MFA LD FACCU,ARG

5385 MFM LD FACCU,(HL)

538A MMA LD (HL),ARG

538F MMF LD (HL),FACCU

5397 XTF EX (SP),FACCU

53F1 PHA PUSH ARG

53F6 PHF PUSH FACCU

5406 PPA POP ARG

540C PPF POP FACCU

1.11.2 Einfach genaue Funktionen

R bezeichnet bei den einfach genauen Funktionen die Register B, C, D und E. Diese Register werden neben dem FACCU als Operandenregister benutzt. Die einfach genauen Zahlen haben nur den Vorteil des geringeren Speicherverbrauches gegenueber den doppelt genauen Zahlen. Es verbergen sich hinter den genannten Adressen jeweils eine Umwandlung in doppelt genaue Zahlen und der Einsprung in die entsprechende Routine fuer doppelte Genauigkeit.

5989 FSUB FACCU:=FACCU-R FACCU : R - FACCU !!

5980 FADD FACCU:=FACCU+R

598E FMULT FACCU:=FACCU*R

5999 FDIV FACCU:=FACCU/R FACCU R / FACCU !!

5650 Vergleich R gegen FACCU SGN(FACCU-R)

5EF6 SNGEXP ~~FACCU:=FACCU^R~~ FACCU := R ^ FACCU

55E0 PUSH FACCU

55ED LD FACCU,(HL)

55F0 LD FACCU,R

55FB LD R,FACCU

560E LD R,(HL)

5617 LD (HL),FACCU

ARG:
C ≙ Exp
B ≙ MSB
E
D ≙ LSB !!

1.11.3 Ganz zahlige Funktionen

5890 ISUB FACCU:=DE-HL

589B IADD FACCU:=DE+HL

58BC IMULT FACCU:=DE*HL

590F IDIV FACCU:=DE/HL

567A ICOMP Vergleich DE HL SGN(HL-DE)

5F6D INTEXP FACCU:=DE^HL

1.11.4 Funktionen fuer alle Datentypen

28 SIGN Vorzeichentest der Zahl in FACCU
A = 0 wenn FACCU=0
A = -1 wenn FACCU negativ
A = 1 wenn FACCU positiv

561E LD ARG,(HL)

5634 LD FACCU,ARG

5637 LD FACCU,(HL)

563C LD ARG,FACCU

563F LD (HL),FACCU

1.11.5 Wandlung der Zahlen in einen anderen Typ

56B5 FRCINT FACCU:=CINT(FACCU)

56DD FRCSNG FACCU:=CSNG(FACCU)

5765 FRCDBL FACCU:=CDBL(FACCU)

1.12 Tongenerator und Joysticks

Fuer den Tongenerator-Chip gibt es eine besondere Routine auf Adresse 40B6. Mit dieser Routine kann ein Byte in E in das Register A des AY38910 geschrieben werden. Fuer alle weiteren Funktionen sind bei den entsprechenden BasicAnweisungen ausreichend Beispiele. Dies gilt auch fuer die JoystickPorts.

2. BASICRoutinen

Unter diesen Punkt fallen Routinen, die fuer den Programmablauf gebraucht werden. So sind Routinen enthalten, die Zeilennummern

suchen, Zeilenpointer restaurieren, TextTokenWandlung ausfuehren und Strings erzeugen. Fuer diese Routine ist eine Beschreibung der Ein und Ausgabeparameter schwieriger als bei den Basis-Routinen. Aus diesem Grund ist es ratsam, fuer den Einzelfall in das ROM-Listing zu sehen.

2.1 Programmtext

171 SCRTCH ist NEW ohne Syntaxpruefung

9C READY Warmstarteinsprung. Textmodus wird eingeschaltet, evtl. zum Drucker und/oder Bildschirm ein Zeilenvorschub gegeben, die Programmausfuehrung beendet und der Prompt ausgegeben.

E1 RUNC BASIC-Variablenbereich initialisieren fuer RUN.

90 READYR Einsprung nach Fehler zur READY-Routine. Der Stack wird restauriert (STKERR), das Programm gestoppt und ein Sprung zur Warmstartadresse erfolgt.

C0 CRUNCH Umschluesseln einer Eingabezeile in Token. Eingabezeile steht ab HL, die Tokenzeile in KBUF.

120 MAKUPL LD A,(HL) und Wandeln in Grossbuchstaben, falls es ein Kleinbuchstabe war. Ergebnis in A.

123 OBCAH HL muss Adresse der ALPTAB enthalten und A den ersten Buchstaben der Anweisung. Auf dem Stack muss der Zeiger auf den ersten Buchstaben sein. Es wird wie bei CRUNCH das Token fuer diese Anweisung eingesetzt, wenn sie in der Tabelle gefunden wurde.

A2 NEWSTT Ausfuehren der naechsten BASIC-Anweisung. HL muss auf ':' oder EOS (0 Byte) zeigen, da es vorher erhoeht wird.

E82 GONE Ausfuehren einer BASIC-Anweisung. HL zeigt auf ein Zeichen vor dem ersten Token.

165 INILIN Interpretation einer Tokenzeile (HL+1)

1B0E BUFLIN expandiert eine Tokenzeile in den Zwischenspeicher BUF. HL zeigt auf den Programmtext hinter der Zeilennummer.

8 Es wird ein Byte (HL) mit ((SP)) verglichen und (SP)=(SP)+1 auf Wortbasis ausgefuehrt. Sind die Bytes ungleich, wird Syntax-Error ausgegeben, sonst wird 10H angesprungen. Der sinnvollste Aufruf erfolgt durch:

```
RST 8H
DEFB 'A'        wenn (HL) ungleich
                A ist Syntax-Error
```

Sind die Bytes identisch, erfolgt der Ruecksprung auf das auf 'A' folgende Byte.

10 CHRGTR entspricht 0EADH.
Pruefung des naechsten Zeichens, HL wird um 1 erhoeht

und (HL) nach A geladen und getestet. CARRY ist geloescht, falls das Zeichen groesser ':' ist, Z ist gesetzt, falls es ein 0 Byte ist. Es werden einige Zeichen ab der Adresse OEB2H interpretiert.

26 Basis-Adresse der Hook-Jumps. (FE79)

30 GETYPR Test des Datentyps in FACCU. VALTYP muss richtig gesetzt sein. Jeweils ein Bit aus F ist eindeutig einem Datentyp zugeordnet.

FACCU	VALTYP	A	F
INTEGER	2	-1	NZ C M PE
STRING	3	0	Z C P PE
SINGLE	4	1	NZ C P PO
DOUBLE	8	5	NZ NC P PE

2.2 Strings

84 STRINI String der Laenge A erzeugen. Ausgabe Parameter sind:
DE enthaelt Stringadresse, HL enthaelt die temporaere Descriptor-Adresse DSCTMP.

87 PUTNEW Temporaeren Stringdescriptor (DSCTMP) in den Stringbereich kopieren. Ausgabe Parameter: HL = DSCTMP, A Stringlaenge

8A FRESTR Der durch den String, dessen Descriptor in FACCU steht, belegte Speicher wird freigegeben, wenn es der zuletzt eingefuegte String war. (Sonst muss auf Garbage-Collection gewartet werden)

8D STROUT Stringkonstante mit 18H ausgeben. HL zeigt auf das erste Zeichen nach der Quote.

16B GETSPA versucht A Bytes im Stringspeicher zu reservieren; sind sie nicht vorhanden, wird die Garbage-Collection angestossen und der Versuch wiederholt. Tritt erneut der Fehler auf, wird die Fehlerroutine angesprungen. HL enthaelt bei Erfolg die Stringadresse

168 CHKSTR Pruefen ob in FACCU String ist, sonst Type mismatch

2.3 Zeilennummern

9F LINKER Restaurieren der Zeiger auf die naechste Programmzeile, nachdem eine Zeile eingefuegt wurde.

C3 CHEAD Neue Zeiger auf die Programmzeilen ab der Zeile, deren Adresse in DE steht, erzeugen.

B27 FNDLIN sucht Zeilennummer DE im Programmtext, C Zeilennummer gefunden. BC enthaelt Zeiger auf die gefundene Zeile, HL Zeiger auf naechste Zeile.

2.4 Fehler

96 ERROR Fehlerroutine. Fehlernummer muss in E stehen.

99 0981H Fehlertext zur Nummer in E suchen. Fehlertext-Basisadresse in HL.

93 SNERR Syntax Error-Einsprung

A5 FCERR Illegal Function-call-Fehler.

CF DIOERR Erzeugt Device IO-Error

DE OMERR Out of Memory-Error

13F NOROOM Loeschen des Programms und OM-Error

132 CHKTOP falls STKTOP >, HL return, sonst NOROOM

10E DERBFN Bad Filename-Error

111 DEFFAD File already open

114 DERFNF File not found

117 DERFND File not open

11A DERIER Internal Error

11D DERRPE Input past end

138 DERFOV Field Overflow

13E DERSAP Sequential after PUT

12C GETDEV PTRFIL gehoerende Geraete-Nummer nach A holen

135 GETBF1 DE:=9, HL:=HL+DE (Datenpuffer zu Datei-Zeiger in HL aufsetzen)

144 GETBUF Datenpufferadresse zu PTRFIL nach HL laden

177 INDSKE POP DE, POP HL, POP BC, RET

17A PTRGET Variablenzeiger finden. HL zeigt auf Tokenzeile mit dem Variablennamen.

2.5 Ausdrucksberechnung

AB EVAL Argument aus Tokenanweisung nach FACCU bringen. HL zeigt auf Argumentstart-1.

AE FRMEVL Ausdruck aus Tokenanweisung berechnen. HL zeigt auf das erste Byte des Ausdrucks.

B1 GETBYT Byte aus Tokenzeile. Ausdruck mit OAE berechnen und das Byte mit OB7H erzeugen. Das Byte steht in A,

E und FACCU (evt. FCerror)

15F GTBYTC Byte aus Token-Ausdruck (HL) berechnen, vorher RST 10H

B4 FRMQNT INTEGER aus Tokenzeile mit OAEH berechnen. FACCU und HL enthalten die Zahl.

B7 CONINT FACCU nach INTEGER wandeln. INTEGER in DE, A=E. (evt. FCerror) HL wird auf naechstes Token gesetzt.

BD GETIN2 INTEGER-Ausdruck aus Tokenzeile (HL) berechnen. NZ wenn kein INTEGER, FACCU und DE enthalten INTEGER.

162 INTIDX Positive INTEGER-Zahl aus Token-Ausdruck (HL) berechnen sonst Illegal Function-call.

BA SNGFLT A als INTEGER nach FACCU bringen.

174 FRCINT FACCU nach INTEGER wandeln und nach HL laden.

C6 CONIA Erzeugt aus A eine INTEGER-Zahl in FACCU. HL enthaelt die Zahl.

C9 INEG2 Wandelt HL in eine SINGLE-Zahl nach FACCU.

CC MAKINT Erzeugt aus HL eine INTEGER-Zahl in FACCU.

2.6 Dateien

E4 NAMSCN Dateinamen aus Tokenzeile (HL) berechnen. Name in FILNAM, Geraetenummer in A.

13B NAMSC1 wie NAMSCN, jedoch wird als Dateiname der zuletzt erzeugte String benutzt.

141 FILSCN Dateinummer aus Tokenzeile (HL) behandeln, HL zeigt auf # oder auf die Zahl (mit/ohne vorlaufenden Leerzeichen). Danach enthalten HL und PTRFIL die Dateipufferadresse.

EA GETFLP FACCU nach INTEGER wandeln. Low Byte nach A weiter mit OEDH.

ED GETPTR Zu der Dateinummer in A die Dateipufferadresse nach HL und die Geraetenummer nach A laden.

F0 SETFIL wie OEDH. Zusaetzlich wird die Pufferadresse nach PTRFIL gespeichert.

F3 NULOPN oeffnen der Datei, deren Nummer in A steht. Geraetenummer in D, Dateiname in FILNAM.

F6 CLSFIL Datei schliessen, deren Nummer in A steht.

FC CLSALL Schliessen aller Dateien, falls NLONLY nicht negativ (nicht Programmlademodus).

D5 EOF Berechnet EOF zur Dateinummer in FACCU.

F9 NOCLSB Puffer loeschen, Modus auf 0, Flag auf 0 PTRFIL auf 0. Parameter nur PTRFIL.
Aufrufen durch CALL xx, xx: PUSH HL, JP 0F9H.

105 CLRBUF Den zu PTRFIL gehoerenden Datenpuffer loeschen.

108 DOCLR B Bytes ab HL (einschliesslich) auf 0 setzen.

77A8 GENDSP Dateibehandlung HL Dateipufferadresse, A Funktion E,C Parameter.

A=0 OPEN E=1 INPUT
2 OUTPUT
4 RANDOM I/O
8 APPEND

2 CLOSE
4 PUT wenn auf Stack AF=NZ, GET wenn auf Stack AF=Z
6 Put CHR
8 Get CHR
A LOC
C LOF
E EOF
10 FPOS
12 Puffer fuer letztes Zeichen mit C setzen

FF FILOU1 Schreiben von A in die durch PTRFIL bezeichnete Datei.

102 INDSKC Einlesen eines Bytes nach A von der durch PTRFIL bezeichneten Datei.

3. Tabellen

```
185     Tabelle Ausfuehrungsadresse fuer Token
 .              Adresse fuer Token 81H
 .                  "   "    "    82H
294
295     Tabelle Eingangsadresse fuer den ersten Buchstaben der
        Schluesselworte in die naechste Tabelle
 .              Adresse fuer 'A'
 .                  "   "  'B'
2C7                 "   "  'Z'
2C9     Tabelle der Schluesselworte mit zugehoerigem Token
 .              Schluesselworte ohne erstes Zeichen
 .              Beim letzten Zeichen ist das 7. Bit gesetzt
 .              Das erste Zeichen ist 0, wenn dieser Anfangsbuchstabe
 .              beendet ist. Nach dem letzten Zeichen des
 .              Schluesselwortes steht das Token
586
587     Tabelle mit Operatoren    + - * / ^ \ ' > = <
 .
59B

59C     Operator-Praezedenz-Tabelle

5B2     Tabelle mit Ausfuehrungsadressen der Arithmetik-Routinen

5D7     Tabelle mit Fehlermeldungen
 .            Textende mit 0
84B
-


4198    Zeichengenerator-Maske komprimiert    6 Byte pro Zeichen
 .                  20H - 7EH ascii
 .       43D2       A0H - DFH grafik
4551

541C    Gleitpunkt-Konstanten

8000    BASIC Programmtexttabelle
        beginnend mit 0
        Folge von Programmzeilen
        beendet mit zwei 0 Bytes

        Programmzeilenaufbau:
              Adresse der folgenden Zeile
              Zeilennummer (INTEGER)
              Tokenzeile
              0 Byte  (End Of Statement)
```

4. BASIC-System-RAM

F500-F54D initialisiert durch 84C-899

```
F500    RAMLOW                                    * D3 0 C9 0
F504    RNDCNT                                    * 0 0
F506    RNDTAB  Tabelle mit Zufallszahlenparametern
F52B    USRTAB  Adressen der USR Routinen         * 10* FCERR
F53F    ERRFLG  speichert Fehlernummer
```



```
F540    LPTLST
F541    LPOS      Zeichenposition des Druckers
F542    ODEVLINK  0 Ausgabe auf Bildschirm sonst Drucker
F543    LINLEN    aktuelle Zeilenlaenge 39/40/80
F545    RUBSW
F546    STKTOP    Adresse des BASIC-Stacks geaendert durch CLEAR
F548    CURLIN    Zeilennummer der ausgefuehrten Zeile
        -1 wenn im direkten Modus
F54A    TXTTAB    Adresse der ersten Programmzeile
F54C    VLZADR    Adresse des durch VAL ersetzten Zeichens * 0 0

F54E    3A ... 00 00 00
F54F    KBUF      Puffer fuer CRUNCH. Hier wird die codierte
        Anweisung (mit Tokens) gespeichert. = 139 Byte
F68D    BUFMIN    ","
F68E    BUF       Eingabezeilenpuffer
F790    ENDBUF
F791    TTYPOS    POS      Cursorposition auf dem Bildschirm
F792    DIMFLG    <>0 wenn PTRGET fuer DIM Anweisung
F793    VALTYP    Datentyp fuer BASIC FACCU

F798    CONSAV    Speicher fuer Konstanten-Token nach CHRGTR
F7A2    MEMSIZ    Groesste, durch BASIC benutzbare Speicheradresse
                  = Adresse des Dateipuffer 0
F7A4    TEMPPT    Adresse des letzten temporueren String-Descriptors
F7A6    TEMPST    Speicher fuer temporaere String-Descriptoren
 ...

F7C4    DSCTMP    String-Descriptor fuer Stringfunktionen ( Laenge)
F7C5    DSCPTR      "       "              (Adresse des Strings)
F7C7    FRETOP    Adresse des freien Stringspeichers
F7C9    TEMP3     speichert Stringfeldende bei Garbage Collection
F7CB    TEMP8
F7CD    ENDFOR    speichert Programmtextzeiger am Ende der FOR-Schleife
F7D1    SUBFLG    <>0 PTRGET findet keine ARRAYs
F7D2    USFLG
F7D3    TEMP
F7D5    PTRFLG    0 wenn keine Zeilennummer in Zeiger umgewandelt ist.
F7D6    AUTFLG    <>0 AUTO-Modus
F7D7    AUTLIN    Zeilennummer fuer AUTO Modus
F7D9    AUTINC    AUTO-Increment
F7DB    SAVTXT    Zeiger auf Anweisung fuer RESUME durch NEWSTT
F7DD    SAVSTK    NEWSTT sichert Stackpointer fuer Fehlerroutine
F7DF    ERL       Zeilennummer des Fehlers
F7E1    DOT       enthaelt die durch . angesprochene Zeilennummer
F7E5    ONELIN    ON ERROR GOTO Zeilennummer
F7E7    ONEFLG    <>0 ERROR Trap wird ausgefuehrt
F7E8    TEMP2     Speicher fuer Formelberechnung
F7EA    OLDLIN    Zeilennummer, durch ^C STOP oder END gesetzt
F7EC    OLDTXT    Zeiger auf Anweisung, die auf STOP BREAK oder END folgt
F7EE    VARTAB    Zeiger auf Speicher fuer einfache Variable.
                  (TXTTAB)+2 durch NEW.
F7F0    ARYTAB    Zeiger auf ARRAY-Tabelle durch CLEARC oder Einfuegen
                  einer einfachen Variable
F7F2    STREND    Zeiger auf das Ende des benutzten Speichers durch CLEARC
                  oder Einfuegen einer Variable
F7F4    DATPTR    Adresse des naechsten DATA-Ausdrucks durch
                  RESTORE auf (TXTTAB)-1 geaendert durch READ
F7F6    DEFTBL    Voreingestellter Typ fuer jeweiligen Anfangsbuchstaben
                  Durch DEF INT/SNG/DBL/STR aenderbar. Durch PTRGET
                  benutzt, wenn kein $, %, !.oder # dem Variablennamen folgt
```

```
F810    PRMSTK  Definitionsblock fuer Garbage-Collection
F812    PRMLEN  Laenge des Blocks
F814    PARM1   Parameter-Definitionstabelle

F878    PARMPRV
F87A    PARMLN2 Groesse des Parameterblocks
                Parameter Speicher
F8E0    PARMFLG
F8E1    ARYTA2  Haltepunkt fuer einfache Suche
F8E3    NOFUNS  <>0 wenn FN-Funktionen definiert sind
F8E4    TEMP9
F8E6    FUNACT  Anzahl der aktiven Funktionen
F8E8    VLZDAT  Zeichen, das von VAL durch 0 ersetzt wurde
F8E9    SWPTMP  Speicher fuer erste SWAP-Variable
SETUP
F8F1    TRCFLG  <>0 bedeutet, Trace ist eingeschaltet



F8F2    FBUFFR  Puffer fuer FOUT

F91A    FMLTT1
F91B    FMLTT2
F91D    DECTMP  Speicher fuer Arithmetik-Routinen
F91F    DECTM2
F921    DECCNT
F923    FACCU   BASIC-Akkumulator
F925    FACLOW

F92A
F933    HOLDB   fuer Dezimal Multiplikation  x*8
F948    HOLD5                                x*5
F963    HOLD2                                x*2
F968    HOLD                                 x*1
F974    ARG     Argument-Akkumulator
F984    RNDX    die zuletzt generierte Zufallszahl
F98C    MAXDRV  Anzahl der Floppy-Laufwerke
F98D    MAXFILES  Anzahl der Dateien
F98E    FILTAB  Adresse der Tabelle mit Dateipufferadressen
F990    DRVTAB  Adresse der Tabelle mit Laufwerksparametern
F992    NULBUF  Adresse des Dateipuffers 0
F994    CURDRV  zuletzt benutzte Laufwerksnummer
F995    DRVPTR
F997    PTRFIL  <>0 Adresse des Dateipuffers, an den die Ausgabe
                gehen soll, bzw. von dem gelesen werden soll
F999    FREPLC
F99B    LISTFRE
F99D    FILMOD/RUNFLG  <>0 wenn Programm nach Laden gestartet werden soll
F99E    FILNAM  Dateiname fuer DIRSRC oder OPEN von NAMSCN

F9A7    FILNM2  2. Dateiname fuer NAME

F9B0    LASTTRK zuletzt benutzter Track
F9B1    LISTSCT
F9B2    NLONLY  <>0 wenn Programm geladen wird
F9B3    SAVFLG
F9B4    SAVEND  Endadresse fuer BSAVE
F9B6    DSKBSY
F9B7    ERRCNT
F9B8    ERRCN1
F9B9    RAWFLG
```



```
F9BA    EBCFLG
F9BB    SAVBC
F9BC    STAT0
F9BD    STAT1
F9BE    TSTACK


FA00-FADD     initialisiert durch 7A66-7B43      ***************

FA00    FRSTID  INIRAM                                      * "JS"
FA02    CLICK   <>0 Tastatur-CLICK ist eingeschaltet        * 1
FA03    CSRY    Cursor Zeilenposition  CSRLIN               * 1
FA04    CSRX    Cursor Spaltenposition                      * 1
FA05    CSRSW   <>0 Cursor anzeigen                         * 1
FA06    CNSDFG  <>0 Funktionstastenbelegung anzeigen        * 0FFH
FA07    RG1SAV  VDP Register 1 Spritegroesse Screenmode     * 0E0H
FA08    TRGFLG  Speicher fuer Port 98H Joystick-Trigger     * 30H
FA09    SPCFLG  0 wenn Space-Taste gedrueckt ist            * 1
FA0A    front COLOR     Vordergrundfarbe                    * 15
FA0B    back COLOR      Hintergrundfarbe                    * 4
FA0C    BORCLR          VDP-Register 7 Randfarbe            * 7
FA0D    MAXUPD  JP $code                                    * JP 0
FA10    MINUPD  JP $code                                    * JP 0
FA13    ATRBYT  aktuelle Farbe                              * 15
FA14    PUTFN   JP $code        set by pginit               * JP 0
FA17    QUEUES  Adresse der Musik-Queues                    * QUETAB
FA19    REPCNT  auto repeat-Zeit fuer die Tastatur          * 032H
FA1A    PUTPNT  Zeiger auf naechstes Eingabezeichen in
                Tastaturpuffer                              * KEYBUF
FA1C    GETPNT  Zeiger auf naechstes Zeichen, das aus dem
                Tastaturpuffer gelesen werden kann          * KEYBUF
FA1E    FNKSTR  Funktionstastenbelegung                     * ...

FABE    XOOFLG
FABF    CDMMSK
FAC0    CHKROM/CLOC     CHECK-ROM-ROUTINE oder
                Zeichenadresse des Grafik-Cursors
FAC2    CMASK   Bitmaske des Grafik-Cursors

                CIRCLE

FAC3    ASPECT  Seitenverhaeltnis des Kreises
FAC5    CENCNT  Zaehler fuer Beendigung
FAC7    CLINEF  <>0 Linie zum Zentrum ziehen
FAC8    CNPNTS  Anzahl der zu setzenden Punkte
FACA    CPLDTF  Zeichenrichtung
FACB    CPCND   Anzahl der Punkt im Kreis/8
FACD    CPCNTB  Anzahl der Punkt im Kreis
FACF    CRCSUM  Kreissumme
FAD1    CSTCNT  Anfangspunkt
FAD3    CSCLXY
FAD4    CSAVEA  ADVGRP-Zwischenspeicher
FAD6    CSAVEM    "
FAD7    CXOFF   x offset vom Zentrum
FAD9    CYOFF   Y  "

                PAINT

FADB    LOHMSK  Zwischenspeicher
FADC    LOHDIR
```

```
FADD    LOHADR
FADF    LOHCNT
FAE1    SKPCNT  Anzahl zu ueberspringende Pixel
FAE3    MOVCNT  Bewegungszaehler
FAE5    POIREC  Zeichenrichtung
FAE6    LFPROG
FAE7    RTPROG

                PUT/GET

FAE8    PUTFLG  <>0 PUT
FAE9    MINDEL
FAEB    MAXDEL
FAED    ARYPTR  Adresse des Feldes

                Macrosprache
FAEF    MCLTAB  Adresse der Tabelle der Macrosprache
FAF1    MCLFLG  PLAY oder DRAW

                Queues fuer PLAY und Modem

FAF2    QUETAB  4 Queues

FB0A    QUEBAK  fuer BCKQ

FB0E    VOICAQ  Tonkanal A  128 Zeichen

FB8E    VOICBQ  Tonkanal B  128 Zeichen

FC0E    VOICCQ  Tonkanal C  128 Zeichen

FC8E    RS2IQ   Modem Queue 64 Zeichen



FCCE    PRSCNT  D1-D0   Anzahl der verarbeiteten Strings
                D7=0 falls erste Verarbeitung
FCCF    SAVSP   BASIC Stackpointer waehrend PLAY
FCD1    VOICEN  Nummer des aktuell bearbeiteten Tonkanals
FCD2    SAVVOL  Zwischenspeicher der Lautstaerke fuer Pause
FCD4    MCLLEN  Laenge des Strings
FCD5    MCLPTR  Stringadresse
FCD7    QUEUEN
FCD8    MUSIKF  Musik-Interrupt-Flag
FCD9    PLAYCNT Anzahl der PLAY-Anweisungen, die im Hintergrund
                laufen
FCDA    VCBA
        METREX  2       Uhr countdown
        VCXLEN  1       MCLLEN fuer diese Stimme
        VCXPTR  2       MCLPTR  "
        VCXSPT  2       Stackpointer
        QLENGX  1       Anzahl der Bytes in der Queue
        NTICSX  2       neuer Countdown
        TONPRX  2       Tonperiode
        AMPLTX  1       Amplitudenform
        ENVPRX  2       Huellkurvenperiode
        OCTAVX  1       Oktave
        NOTELX  1       Notenlaenge
        TEMPCX  1       Tempo
        VOLUMX  1       Lautstaerke
        ENVLPX          Huellkurvenform
```



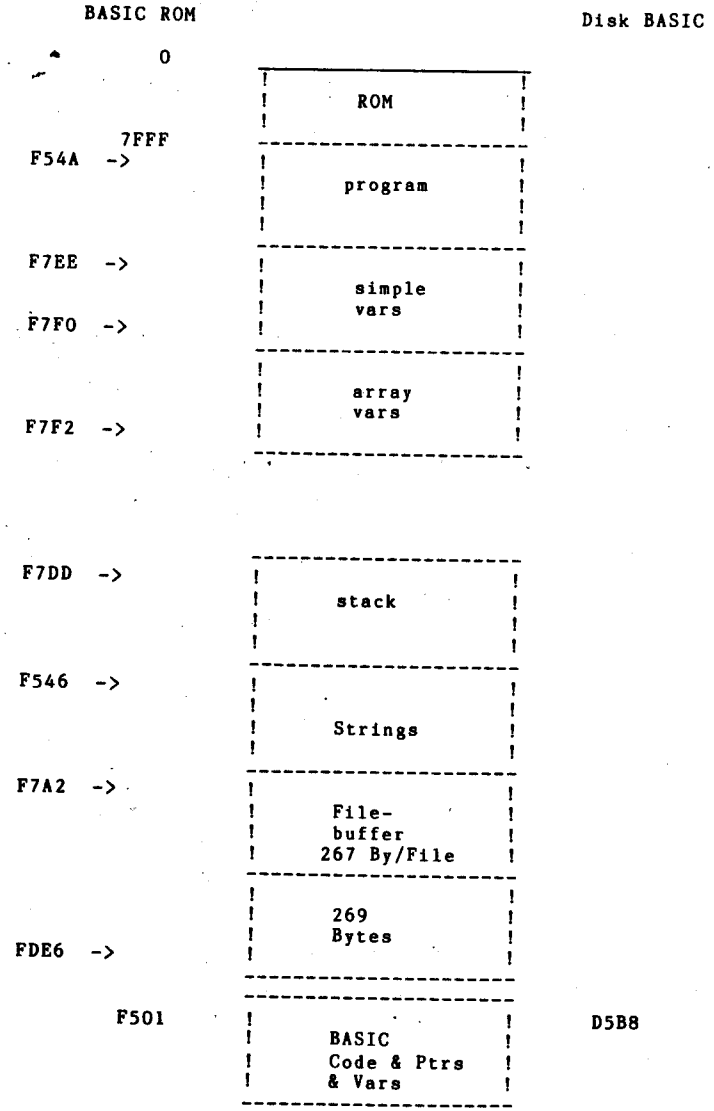
```
        MCLSTX          Stack-Zwischenspeicher
        MCLSEX          initialisierter Stack

FCE9-FCED   initialisiert durch 40B1-40B5      ***************

FCFF    VCBB

FD24    VCCC
++++++++++++++++++++++++++++++++++++++++++++++++++++++++
FD49    MDMFLG  NZ= Modem aktiv
FD4A    STPOPT  Flag fuer SWITCH STOP
FD4B    FRCNEW  Flag um Speicher zu initialisieren
FD4C    POLRTY  Polaritaet des Kasettensignals
FD4D    LINTTB  Zeilenbegrenzertabelle ein Byte pro Zeile.
                <>0, Zeile geht nicht ueber Bildschirmrand hinaus

FD65    FSTPOS  Cursorposition beim Eintritt in INLIN
FD67    CODSAV  Code des Zeichens, auf dem der Cursor steht
FD68    FNKSWI  0 oder 1 bestimmt die angezeigte Funktionstastenreihe
FD69    FNKFLG  <>0, Funktionstaste ist in ON KEY GOSUB definiert
.
FD72
FD73    ONGSBF  Ereignisflag <>0, ein Ereignis ist eingetreten
FD74    CLIKFL  Auszeit fuer CLICK
FD75    OLDKEYS Vorhergehende Tastaturbytes
.
FD7F
FD80    ACTKEYS aktuelle Tastaturbytes  Port 99H Puffer
        Bits Zero = Taste gedrueckt
        BIT     7     6     5     4     3     2     1     0
.       ROW 0   7&    6^    5%    4$    3#    2@    1!    0)
.               /?    .>    =+    ,<    '"    :;    9(    8*
.               G     F     E     D     C     B     A     _
.               O     N     M     L     K     J     I     H
.               W     V     U     T     S     R     Q     P
.               UP    BS    ]}    \     [{    Z     Y     X
FD86    SHCTRL  <-    ENTER stop  ESC RighGR LeftGR ctrl shift
.               DO    INS   CLS   F5    F4    F3    F2    F1
.               right       print sel   caps  DEL   TAB   SP
.               7     6     5     4     3     2     1     0
FD8A    ROW 10  ,     .     /     *     -     +     9     8

FD8B    KEYBUF  Tastaturpuffer fuer ASCII-Codes
.
FDB2
FDB3    BUFEND
FDB4    LINWRK  Zeilen-Zwischenspeicher fuer Bildschirm-Editor
FDDC    PATWRK  Speicher der expandierten Zeichengeneratormaske
.
FDE3
FDE4    BOTTOM  Niedrigste RAM-Adresse C000 oder 8000
FDE6    HIMEM   Hoechste verfuegbaare Speicheradresse fuer BASIC
                (2. parm in CLEAR stmt)
FDE8    TXPSAV  Programmtext Zeiger Zwischenspeicher
FDEA    CASATR  Kassettenattribut der gelesenen Datei
FDEB    TRPTBL  Trap Tabelle    3 Byte/Trap
                1. Byte   Bit 0  trap ist eingeschaltet
                          Bit 1  Trap-Stop wurde ausgefuehrt
                          Bit 2  Trap-Request existiert
                2.3.Byte  Zeilennummer des Traps
```

```
FE06    KEY_OOS          Flag fuer KEYS Adressen in TRPTBL
FE09    STOP_OOS         Trap fuer CTRL-STOP
FE0C    SPRITE_OOS       Trap fuer SPRITE-]berlappung
FE0F    STRIG_OOS fuer Joystick 1
FE12              "     "       2
FE15              "   Space-Taste
FE18    INTERVAL_OOS     Trap fuer Interval-bedingung
FE1B    MDM_OOS          Trap fuer Modemdaten

FE2A    RTYCNT   Anzahl der Leseversuche beim Booten
FE2B    INTFLG   3=CTRL-STOP gedrueckt  4=STOP gedrueckt
FE2C    PADY     Y Koordinate des Tablet
FE2D    PADX     X Koordinate des Tablet
FE2E    TIME     Interne Uhrzeit
FE30    INTVAL   Wert der bei ON INTERVAL angegeben wurde
FE32    INTCNT   aktueller INTERVAL-Zaehler (zaehlt bis 0, dann Interrupt)
FE34    ESCCNT   ESC wurde ausgegeben
FE35    REVFLG   <>0, alle Zeichen invers ausgeben
FE36    INSFLG   <>0, Einfuegemodus aktiv
FE37    CSTYLE   <>0, Cursor halb hoch
FE38    CAPST    <>0, nur Grossschrift
FE39    FLBMEM   0, wenn BASIC-Programm geladen wird
FE3A    SCREEN Modus    0 text, 1 hires, 2 multi
FE3B    SPRSIZ   SCREEN option (spritegroesse)
FE3C    VDP Register 0
FE3D    STATFL
FE3E    KBDPRV   zuletzt gelesenes Zeichen von der Tastatur
FE3F    CASPRV   ... Casette
FE40    MDMPRV   ... Modem
FE41    BRDATR   Randfarbe fuer PAINT
FE42    GXPOS    Grafik x-Koordinate
FE44    GYPOS    Grafik Y-Koordinate
FE46    GRPACX   Grafik Akku
FE48    GRPACY
FE4A    DRWFLG   DRAW Flag
FE4B    DRWSCL   DRAW Skalierungs-Faktor    0 keine Skalierung
FE4C    DRWANG   DRAW Winkel (0-3)

                 fuer das Modem

FE4D    DATCNT   <>0, Interrupt steht an
FE51    SIOFLG   SI/SO Control Logic
FE52    RCVXOF   xoff empfangen
FE53    SNTXOF   xoff bereits gesendet
FE54    RCVSFT   shift-Status beim Empfang
FE55    SNDSFT   shift-Status beim Senden
FE56    ADDPM    Waehlfrequenz

                 BLOAD / BSAVE

FE57    RUNBNF
FE58    SAVENT

                 Monitor

FE5A    REGPC
FE5C    REGSP
FE5E    REGHL
FE60    REGDE
FE62    REGBC
FE64    REGA
```



```
FE65    REGF

FE71    REGFT
FE72    MONFLG
FE73    SAVESP

FE75    SWIFLG   aktuelle Speicherbank
FE76    SPSAVE   Stack-Pointer-Speicher fuer SWITCH

FE78    SCNCNT   Zaehler fuer Interrupts; bei jedem dritten
                 Interrupt wird die Tastatur geprueft

FE79-FFA6   HOOK Jump Table   ****************************
```

4.1 Speicheraufteilung



5. I/O Portadressen

Portadresse (HEX) Funktion

80 write Data into VRAM.

VideoRAM Adresse — Text Modus

0..	Videocodes fuer die Zeichen
800H..	Zeichengeneratormasken

Grafik Modus

0..	Musterbelegung
1800H	Namenstabelle
1B00H	Spriteebenen-Attribute
2000H	Farbbelegung, obere 4 Bits Farbe fuer gesetzte Bits, untere 4 fuer 0 Bits
3800H	Spritemuster

Spriteebenenattribute

1. Byte Y-Position 255 oben am Schirm
2. " X-Position 0 links am Schirm
3. " Spritenummer fuer Darstellung
4. " untere 4 Bits Farbe des Spritemusters fuer gesetzte Bits. Ist Bit 7=1 wird die Horizontal-Darstellung um 32 Pixels nach links verschoben

81 write into VDPregister

VDPregister

0 000000 M3 0

1 1 BLK IE M1 M2 0 SIZE MAG
BLK=0 nur Randfarbe auf dem Schirm
IE=1 Interrupts enabled
M1 M2 M3 =1 SCREEN 1
=2 SCREEN 2
=4 SCREEN 0
SIZE=1 16*16 Pixel fuer Sprites
MAG=1 Sprites doppelt gross

7 obere 4 Bits Farbe fuer gesetzte Bits im Text-Modus.
untere 4 Bits Randfarbe

84 read VRAM

85 read VDP-Status sollte nur gelesen werden wenn VDP Interrupt anliegt
F 5S C N4 N3 N2 N1 N0
F=1 wenn ein Bild angezeigt ist
C=1 wenn sich zwei oder mehr Sprites ueberlappen
5S=1 wenn mehr als 4 Sprites in einer horizontalen Linie sind;
die Nummer des 5. Sprites steht dann in den unteren 5 Bit.

AY-3-8910

88 registeraddress Latch

8C write register

```
90              read register

                  R14 Port A
                        Joystick 1
                    A0  forward
                    A1  backward
                    A2  left
                    A3  right
                        Joystick 2
                    A4  forward
                    A5  backward
                    A6  left
                    A7  right

                  R15 Port B
                    B0  -CART
                    B1  -BK21
                    B2  -BK22
                    B3  -BK31
                    B4  -BK32
                    B5  CAPS
                    B6  -ROMENO
                    B7  -ROMEnO
        8255
94              write port A
95              write port B
96              write port C
97              write control
98              read port A
99              read port B
9A              read port C

                        Port C
                  C0-C3     Out Keyboard row 0 - 10 BCD
                  C4        CAS ON
                  C5        CAS WR
                  C6        CAS Audio
                  C7        Sound
                      Port B
                  B0-B7     Keyboard column input
                      Port A
                  A0        TA
                  A1        TB
                  A2        TC
                  A3        TD
                  A4        Trigger Joystick 1
                  A5        Trigger Joystick 2
                  A6        READY
                  A7        CAS RD


    P1 Connector Pins
-CNTRL1   disable RAM
-CNTRL2   disable IO-decode
-EXCSR    enable READ TMS 9929
-EXCSW    enable write TMS 9929
-RAMDIS   disable RAM
-ROMDIS   disable ROM
```



```
                Memory
0-3FFF    BASIC ROM     bei   -BK21 High  -ROMDIS high  -CART high
          -CCS1                -CART low
4000-7FFF BASIC ROM           -BK21 High  -ROMDIS high  -CART high
          -CCS2               -CART low
8000-BFFF RAM                 -CCS3 high  -CCS4 high
          -CCS3               -CART low   -ROMENO low
C000-FFFF RAM                 -CCS3 high  -CCS4 high
          -CCS4               -CART low   -ROMEN1 low

        bei CCS. ist -IORQ high hinreichend zum aktivieren,
        -MREQ wurde nicht dekodiert


    Externe Ports

     30         Floppy Status/Command (SY-1793)
                 0x      Restore                 STATUS
                 1x      Seek                    B7  Not Ready
                 2x      Step                    B6  Write protect
                 4x      Step in                 B5  deleted data
                 6x      Step out                    write fault
                                                 B4  Seek error
                 8x      Read Sector             B3  CRC error
                 Ax      Write Sector            B2  Track 00
                 Cx      Read Address                lost data
                 Ex      Read Track              B1  Index pulse
                 Fx      Write Track                 Data request
                 Dx      Force Interrupt         B0  Busy

     31         Track
     32         Sector
     33         Data

     34-37      B0   drive sel 0 aktiv high      WR-only
                B1   drive sel 1   "   "            "
                B2   Motor on 0    "   "            "
                B3   Motor on 1    "   "            "
                B6   1793 DMARQ  aktiv high      RD-only
                B7   1793 INTRQ    "   "            "

     38-3B      B0   high Single Density
                     low  Double Density

          Floppy SA 200                          SV
                2   reserved                     9  -
                4     "                          11   Index
                6   Drive 4                      13   Disksel
                8   Index                        15   -
               10   Drive 1                      17   -
               12   Drive 2                      19   Motor on
               13   Drive 3                      21   Direction
               16   Motor on                     23   Step
               18   Direction                    25   Write Data
               20   Step                         27   Write Gate
               22   Write Data                   29   Track 00
               24   Write Gate                   31   Write Protect
               26   Track 00                     33   Read Data
               28   Write Protected
               30   Read Data                 1,2,16,18,20      +12V
```

32	reserved	3-6,22,24,26,28,30,32,34	GND
34	reserved	7,8,10,12	+5V

alle Ungeraden Masse

1	+12V	1.1A
2	Masse	
3	Masse	
4	+5V	0.5A

Centronics

10	Data out		
11	BO	-STB	active low
12	BO	BUSY	active high

80 column (6845)

50	address register		
51	Data register		
58-59	BO	high Bank enable	internes RAM disabled
		low Bank disable	

80-Zeichen-RAM-Adresse F000-F7FF

Modem (8250)

20	Receiver Buffer register
21	divisor latch MSD
22	interrupt identification register
23	line control
24	line status
25	Modem control register
26	Modem status register

RS 232 (8250)

28	receiver buffer register
29	divisor latch MSD
2A	Interrupt identification register
2B	line control
2C	Modem control
2D	line status
2E	Modem status

Divisor latch

Baudrate	Wert
19200	10
9600	20
7200	27
4800	40
3600	53
2400	80
2000	96
1800	107
1200	160
600	320
300	640
150	1280
134.5	1428
110	1745

75	2560
50	3840

lines		
1,7	GND	
2	Data out	
3	Data in	
4	-RTS out	
5	-CTS in	
6	-DSR in	
8	-RLSD in	receive line signal defect
20	-DTR out	

Anhang A.1

1	'	ABS	6	55B1
2	'	AND	F8	*5765*
3	'	ASC	15	6B10
4	'	ATN	E	5139
5	'	ATTR$	E9	34D3
6	'	AUTO	A9	11F5
7	'	BEEP	C0	40BE
8	'	BIN$	1D	6904
9	'	BLOAD	CD	7684
10	'	BSAVE	CE	7624
11	'	CDBL	20	5765
12	'	CHR$	16	6B20
13	'	CINT	1E	56B5
14	'	CIRCLE	BC	2652
15	'	CLEAR	92	67A6
16	'	CLICK	C8	31AF
17	'	CLOAD	9B	1EAA
18	'	CLOSE	B4	7375
19	'	CLS	9F	3777
20	'	CMD	D7	34C4
21	'	COLOR	BD	4552
22	'	CONT	99	671B
23	'	COPY	D6	34BF
24	'	COS	C	50B8
25	'	CSAVE	9A	1E15
26	'	CSNG	1F	56DD
27	'	CSRLIN	E8	31C7
28	'	CVD	2A	7331
29	'	CVI	28	732B
30	'	CVS	29	732E
31	'	DATA	84	109B
32	'	DEF	97	188A
33	'	DEFDBL	AE	F65
34	'	DEFINT	AC	F5F
35	'	DEFSNG	AD	F62
36	'	DEFSTR	AB	F5C
37	'	DELETE	A8	1C6C
38	'	DIAL	D0	79C2
39	'	DIM	86	6061
40	'	DRAW	BE	29DA
41	'	DSKF	26	34C9
42	'	DSKI$	EA	34CE
43	'	DSKO$	D1	34A6
44	'	ELSE	A1	109D
45	'	END	81	66CF
46	'	EOF	2B	74B0
47	'	EQV	FB	*3263*
48	'	ERASE	A5	676E
49	'	ERL	E1	1671
50	'	ERR	E2	1663
51	'	ERROR	A6	11EA
52	'	EXP	B	526B
53	'	FIELD	B1	72CD
54	'	FILES	B7	73B2
55	'	FIX	21	57E9
56	'	FN	DE	18AD
57	'	FOR	82	D65
58	'	FPOS	27	74C6

59	'	FRE	F	6CF7
60	'	GET	B2	2FB4
61	'	GO TO	89	1028
62	'	GOSUB	8D	FF6
63	'	GOTO	89	1028
64	'	HEX$	1B	68FF
65	'	IF	8B	1225
66	'	IMP	FC	*3280*
67	'	INKEY$	EC	64F3
68	'	INP	10	1A37
69	'	INPUT	85	13D2 from File 7406 fkt input
70	'	INSTR	E5	6BF0
71	'	INT	5	57F8 3048 SOL(FF 85) 3050 INTERVAL
72	'	IPL	D5	34BA
73	'	KEY	C7	3120 3124 key list 3070 key (...
74	'	KILL	D4	34B5
75	'	LEFT$	1	6B66
76	'	LEN	12	6B04
77	'	LET	88	10C0
78	'	LFILES	BB	73AD
79	'	LINE	AF	1374
80	'	LIST	93	1AB8
81	'	LLIST	9E	1AB3
82	'	LOAD	B5	7121
83	'	LOC	2C	7484
84	'	LOCATE	D8	2FD1
85	'	LOF	2D	749A
86	'	LOG	A	5197
87	'	LPOS	1C	1834
88	'	LPRINT	9D	125D
89	'	LSET	B8	7228
90	'	MAX	CA	7CBA
91	'	MDM	CF	3036
92	'	MERGE	B6	7122
93	'	MID$	3	6B9F 6C73 SOL (FF 83)
94	'	MKD$	30	7318
95	'	MKI$	2E	7312
96	'	MKS$	2F	7315
97	'	MOD	FD	*32BD*
98	'	MON	CB	7B44
99	'	MOTOR	CC	2BE5
100	'	NAME	D3	34B0
101	'	NEW	94	6556
102	'	NEXT	83	6821
103	'	NOT	E0	*5300*
104	'	OCT$	1A	68FA
105	'	OFF	EB	6909
106	'	ON	95	1124
107	'	OPEN	B0	7080
108	'	OR	F9	*57E9*
109	'	OUT	9C	1A4C
110	'	PAD	25	32BD
111	'	PAINT	BF	24FC
112	'	PDL	24	3280
113	'	PEEK	17	1CA6
114	'	PLAY	C1	2C24 31DE
115	'	POINT	ED	2346
116	'	POKE	98	1CAD
117	'	POS	11	1839
118	'	PRESET	C3	2328
119	'	PRINT	91	1265

```
120   '     PSET        C2    232D
121   '     PUT         B3    2FB1
122   '     READ        87    1405
123   '     REM         8F    109D
124   '     RENUM       AA    1CF0
125   '     RESTORE     8C    66AE
126   '     RESUME      A7    119D
127   '     RETURN      8E    1061
128   '     RIGHT$       2    6B96
129   '     RND          8    5300
130   '     RSET        B9    7227
131   '     RUN         8A     FE2
132   '     SAVE        BA    7167
133   '     SCREEN      C5    459A
134   '     SET         D2    34AB
135   '     SGN          4    55C6
136   '     SIN          9    50D1
137   '     SOUND       C4    2BFD
138   '     SPACE$      19    6B4D
139   '     SPC(        DF   *5222*
140   '     SPRITE      EE    6B20  45D2 SOL  4604 fkt 45D8 sprt$= 3042 on..
141   '     SQR          7    5222
142   '     STEP        DC    55C6
143   '     STICK       22    3206
144   '     STOP        90    66C8
145   '     STR$        13    6909
146   '     STRIG       23    3263  3056 SOL (FF A3)
147   '     STRING$     E3    6B2E
148   '     SWAP        A4    6735
149   '     SWITCH      C9    337F        3377 fkt
150   '     TAB(        DB   *6B9F*
151   '     TAN          D    5120
152   '     THEN        DA    6B96
153   '     TIME        EF    31D3 SOL      31BD fkt
154   '     TO          D9    6B66
155   '     TROFF       A3    6730
156   '     TRON        A2    672F
157   '     USING       E4    50B8
158   '     USR         DD    1842
159   '     VAL         14    6BC0
160   '     VARPTR      E7    16A2
161   '     VPEEK       18    46F2
162   '     VPOKE       C6    46D8
163   '     WAIT        96    1A52
164   '     WIDTH       A0    1A6C
165   '     XOR         FA   *3206*
166   '     >           F0    0000
167   '     =           F1    0000
168   '     <           F2    0000
169   '     +           F3    0000
170   '     -           F4    0000
171   '     *           F5    0000
172   '     /           F6    0000
173   '     ^           F7    0000
174   '     \           FE    0000
```


Anhang A.2

```
1     '     LEFT$       1    6B66
2     '     RIGHT$      2    6B96
3     '     MID$        3    6B9F        6C73 SOL (FF 83)
4     '     SGN         4    55C6
5     '     INT         5    57F8     3048 SOL(FF 85)    3050 INTERVAL
6     '     SQR         7    5222
7     '     RND         8    5300
8     '     SIN         9    50D1
9     '     LOG         A    5197
10    '     EXP         B    526B
11    '     COS         C    50B8
12    '     TAN         D    5120
13    '     ATN         E    5139
14    '     FRE         F    6CF7
15    '     INP        10    1A37
16    '     POS        11    1839
17    '     LEN        12    6B04
18    '     STR$       13    6909
19    '     VAL        14    6BC0
20    '     ASC        15    6B10
21    '     CHR$       16    6B20
22    '     PEEK       17    1CA6
23    '     VPEEK      18    46F2
24    '     SPACE$     19    6B4D
25    '     OCT$       1A    68FA
26    '     HEX$       1B    68FF
27    '     LPOS       1C    1834
28    '     BIN$       1D    6904
29    '     CINT       1E    56B5
30    '     CSNG       1F    56DD
31    '     CDBL       20    5765
32    '     FIX        21    57E9
33    '     STICK      22    3206
34    '     STRIG      23    3263  3056 SOL (FF A3)
35    '     PDL        24    3280
36    '     PAD        25    32BD
37    '     DSKF       26    34C9
38    '     FPOS       27    74C6
39    '     CVI        28    732B
40    '     CVS        29    732E
41    '     CVD        2A    7331
42    '     EOF        2B    74B0
43    '     LOC        2C    7484
44    '     LOF        2D    749A
45    '     MKI$       2E    7312
46    '     MKS$       2F    7315
47    '     MKD$       30    7318
48    '     END        81    66CF
49    '     FOR        82     D65
50    '     NEXT       83    6821
51    '     DATA       84    109B
52    '     INPUT      85    13D2 from File   7406 fkt input
53    '     DIM        86    6061
54    '     READ       87    1405
55    '     LET        88    10C0
56    '     GO TO      89    1028
57    '     GOTO       89    1028
58    '     RUN        8A     FE2
```

59	IF	8B	1225	
60	RESTORE	8C	66AE	
61	GOSUB	8D	FF6	
62	RETURN	8E	1061	
63	REM	8F	109D	
64	STOP	90	66C8	
65	PRINT	91	1265	
66	CLEAR	92	67A6	
67	LIST	93	1AB8	
68	NEW	94	6556	
69	ON	95	1124	
70	WAIT	96	1A52	
71	DEF	97	188A	
72	POKE	98	1CAD	
73	CONT	99	671B	
74	CSAVE	9A	1E15	
75	CLOAD	9B	1EAA	
76	OUT	9C	1A4C	
77	LPRINT	9D	125D	
78	LLIST	9E	1AB3	
79	CLS	9F	3777	
80	WIDTH	A0	1A6C	
81	ELSE	A1	109D	
82	TRON	A2	672F	
83	TROFF	A3	6730	
84	SWAP	A4	6735	
85	ERASE	A5	676E	
86	ERROR	A6	11EA	
87	RESUME	A7	119D	
88	DELETE	A8	1C6C	
89	AUTO	A9	11F5	
90	RENUM	AA	1CF0	
91	DEFSTR	AB	F5C	
92	DEFINT	AC	F5F	
93	DEFSNG	AD	F62	
94	DEFDBL	AE	F65	
95	LINE	AF	1374	
96	OPEN	B0	7080	
97	FIELD	B1	72CD	
98	GET	B2	2FB4	
99	PUT	B3	2FB1	
100	CLOSE	B4	7375	
101	LOAD	B5	7121	
102	MERGE	B6	7122	
103	FILES	B7	73B2	
104	LSET	B8	7228	
105	RSET	B9	7227	
106	SAVE	BA	7167	
107	LFILES	BB	73AD	
108	CIRCLE	BC	2652	
109	COLOR	BD	4552	
110	DRAW	BE	29DA	
111	PAINT	BF	24FC	
112	BEEP	C0	40BE	
113	PLAY	C1	2C24	31DE
114	PSET	C2	232D	
115	PRESET	C3	2328	
116	SOUND	C4	2BFD	
117	SCREEN	C5	459A	
118	VPOKE	C6	46D8	
119	KEY	C7	3120	3124 key list 3070 key (...

120	CLICK	C8	31AF	
121	SWITCH	C9	337F	3377 fkt
122	MAX	CA	7CBA	
123	MON	CB	7B44	
124	MOTOR	CC	2BE5	
125	BLOAD	CD	7684	
126	BSAVE	CE	7624	
127	MDM	CF	3036	
128	DIAL	D0	79C2	
129	DSKO$	D1	34A6	
130	SET	D2	34AB	
131	NAME	D3	34B0	
132	KILL	D4	34B5	
133	IPL	D5	34BA	
134	COPY	D6	34BF	
135	CMD	D7	34C4	
136	LOCATE	D8	2FD1	
137	TO	D9	6B66	
138	THEN	DA	6B96	
139	TAB(	DB	*6B9F*	
140	STEP	DC	55C6	
141	USR	DD	1842	
142	FN	DE	18AD	
143	SPC(	DF	*5222*	
144	NOT	E0	*5300*	
145	ERL	E1	1671	
146	ERR	E2	1663	
147	STRING$	E3	6B2E	
148	USING	E4	50B8	
149	INSTR	E5	6BF0	
150	VARPTR	E7	16A2	
151	CSRLIN	E8	31C7	
152	ATTR$	E9	34D3	
153	DSKI$	EA	34CE	
154	OFF	EB	6909	
155	INKEY$	EC	64F3	
156	POINT	ED	2346	
157	SPRITE	EE	6B20	45D2 SOL 4604 fkt 45D8 sprt$= 3042 on..
158	TIME	EF	31D3	SOL 31BD fkt
159	>	F0	0000	
160	=	F1	0000	
161	<	F2	0000	
162	+	F3	0000	
163	-	F4	0000	
164	*	F5	0000	
165	/	F6	0000	
166	^	F7	0000	
167	AND	F8	*5765*	
168	OR	F9	*57E9*	
169	XOR	FA	*3206*	
170	EQV	FB	*3263*	
171	IMP	FC	*3280*	
172	MOD	FD	*32BD*	
173	\	FE	0000	

Anhang B

Alfabethische Liste der Marken

ABSFN	55B1
ACTKEYS	FD80
ADDPM	FE56
ADVCUR	3AD0
ALPTAB	295
AMPLTX	FCE6
ARG	F974
ARYPTR	FAED
ARYTA2	F8E1
ARYTAB	F7F0
ASC	6B10
ASC2	6B14
ASPECT	FAC3
ATN	5139
ATNFIX	24F
ATRBYT	FA13
ATRSCN	2390
ATTR$	34D3
AUTFLG	F7D6
AUTLIN	F7D7
AUTOP	F7D9
BCXQ	2B9E
BEEP	40BE
BLOAD	7684
BLTU	6520
BLTUC	6523
BOOT	79DC
BORCLR	FA0C
BOTTOM	FDE4
BRDATR	FE41
BREAKX	3512
BRKTXT	8A4
BS	3AC1
BSAVE	7624
BSERR	61DF
BUF	F68E
BUFEND	FDB3
BUFLIN	1B0E
BUFMIN	F68D
CALBNK	3480
CAPST	FE38
CASATR	FDEA
CASBNH	1FC6
CASBNR	1FE4
CASDSP	7841
CASIN	2016
CASOPW	1FA7
CASOUT	2026
CASPRV	FE3F
CAT	6A8C
CBLOAD	1E7F
CBSAVE	1E43
CDMMSK	FABF
CENCNT	FAC5
CGTABL	4198
CHEAD	AE9

CHGCLR	3750
CHGET/TRYIN	403D
CHGMOD	37D9
CHKBNK	3420
CHKBRN	76EA
CHKMDM	798E
CHKMOD	48E1
CHKROM/CLOC	FAC0
CHKSTR/FRCSTR	5783
CHKTOP	7209
CHPLPT	3915
CHPSTT	3938
CHPUT	394D
CHRCON	EB2
CHRGT2	EAE
CHRGTR	EAD oder 10
CHSNS	3DCA
CIRCLE	2652
CKCNTC	405D
CKERCS	3AA7
CKSTTP	64DF
CLEAR	67A6
CLEARO	6577
CLEARC	6571
CLICK	31AF
CLICK	FA02
CLIKFL	FD74
CLINEF	FAC7
CLOAD	1EAA
CLOSE	7375
CLRBUF	7469
CLRSPR	36BE
CLS	3777
CLSALL	737D
CLSCLR	738B
CLSFIL	70EA
CLSHRS	378B
CMASK	FAC2
CMD	34C4
CNPNTS	FAC8
CNSDFG	FA06
CNSGET	1715
CNVCOD	3C5A
CODSAV	FD67
COLOR	4552
CONASD	59B2
CONDS	576D
CONIA	55C9
CONINT	1AA9
CONIS	56BD
CONIS2	56CD
CONSAV	F798
CONSD	56E5
CONSI	56F3
CONSIH	56F6
CONSTP	66D9
CONSTR	84C
CONT	671B
COS	50B8
COSFIX	24B
CPCND	FACB

CPCNT	FACD
CPLDTF	FACA
CRCSU	FACF
CRDO	6474
CRDONZ	6463
CRFIN	647D
CRFIND	647C
CRTDSP	7817
CRUNCH	B44
CSAVE	1E15
CSAVE	FAD4
CSAVEM	FAD6
CSBSAV	1E3E
CSCLX	FAD3
CSHOME	3AF4
CSROON	203A
CSRSW	FA05
CSRX	FA04
CSRY	FA03
CSTCN	FAD1
CSTYLE	FE37
CTOFF	207C
CTRLPT	670B
CTROPT	6709
CTWOFF	206C
CURDRV	F994
CURLIN	F548
CVD	7331
CVI	732B
CVS	732E
CWRTON	2059
CXDPCS	3A6C
CXOFF	FAD7
CYOFF	FAD9
DADO/DECADD	4D94
DADOS	4D91
DATAW	20E3
DATCNT	FE4D
DATPTR	F7F4
DBLEXP	5F05
DBLZER	543E
DCOMP	56AE
DCRART	59C5
DCXBRT	57E7
DCXHRT	59C7
DDERR	8F6
DDIV/DECDIV	4FB7
DECCNT	F921
DECFET	225F
DECMRN	4F46
DECMUL/DMULT	4EFE
DECNRM	4DF6
DECROA	4E44
DECROB	4E3D
DECROU	4E38
DECSRD	4EF3
DECSUB/DSUB	4D86
DECTM2	F91F
DECTMP	F91D
DEFILE	7CDA
DEFTBL	F7F6

DEL	1C8F
DELLNO	3AFD
DEPTR	1E0D
DERBFN	75FA
DERFAD	75FD
DERFDR	7600
DERFND	7606
DERFNF	7603
DERFOV	7609
DERIER	760F
DERIFN	760C
DERRPE	7612
DERSAP	7615
DERSOO	7618
DEVTBL	7788
DGET	73B9
DIAL	79C2
DIM	6061
DIMFLG	F792
DIOERR	204D
DIROG	74D9
DIVMSG	684
DKCOPY	34BF
DLINE	7520
DMULTO	5364
DOASIG	1457
DOCLR	7474
DOGRP2	2488
DOGRPH	247C
DOT	F7E1
DOWNC	4A2D
DPUT	73B8
DRAW	29DA
DRVLEN	4C
DRVPTR	F995
DRVTAB	F990
DRWANG	FE4C
DRWFLG	FE4A
DRWSCL	FE4B
DSCPTR	F7C5
DSCTMP	F7C4
DSKBSY	F9B6
DSKF	34C9
DSKI$	34CE
DSKO$	34A6
DSPCSR	3A71
DSPFNK	3B9F
DVOERR	8F0
DVERR	8FF
EBCFLG	F9BA
ECL	3B60
EDENT	A28
ENDBUF	F790
ENDCON	66E6
ENDFOR	F7CD
ENDST	66CF
ENVLPX	FCEF
ENVPRX	FCE7
EOF	74B0
ERACSR	3AAC
ERAFNK	3B86

ERASE	676E
ERESET	92E
ERL	F7DF
ERRCN1	F9B8
ERRCNT	F9B7
ERRFIN	98D
ERRFLG	F53F
ERROR	907
ERSFIN	61AA
ESCCNT	FE34
EVAL	162D
EXP	526B
FACCU	F923
FACLOW	F925
FADD	5980
FADDS	597D
FBUFFR	F8F2
FCERR	F9E
FCOMP	5650
FDIV	5999
FDIVT	5997
FETCHC	4943
FETCHR	2242
FETCHZ	223C
FIELD	72CD
FILES	73B2
FILGET	74E8
FILIND	7514
FILINP	74E6
FILLEN	109
FILMOD/RUNFLG	F99D
FILNAM	F99E
FILNM2	F9A7
FILOU1	73CA
FILOUT	73C9
FILSCN	7067
FILTAB	F98E
FINBCK	6B2A
FINDBL	59CB
FINI	AC9
FININL	646A
FINLPT	643D
FINPRT	1365
FIXER	57E9
FIXINP	7406
FLBMEM	FE39
FMLTT1	F91A
FMLTT2	F91B
FMULT	598E
FNDFOR	8AA
FNDLIN	B27
FNKFLG	FD69
FNKROM	7A84
FNKSB	3B95
FNKSTR	FA1E
FNKSWI	FD68
FORSZC	19
FOUT	5B57
FOUTB	5E48
FOUTD	5E4C
FOUTH	5E50

FPOS	74C6
FRCDBL	5765
FRCINT	56B5
FRCNEW	FD4B
FRCSNG	56DD
FRE	6CF7
FREFAC	6AD8
FREPLC	F999
FRESTR	6AD5
FRETM2	6ADB
FRETMP	6ADC
FRETMS	6AF3
FRETOP	F7C7
FRETRP	663D
FRMCHK	14CB
FRMEQL	14C5
FRMEVL	14CA
FRMPRN	14C8
FRMQNT	1CB9
FRQINT	1CC3
FRSTID	FA00
FSTPOS	FD65
FSUB	5989
FUNACT	F8E6
GARBA2	69BB
GENDSP	77A8
GET	2FB4
GETBCD	5610
GETBF1	747D
GETBNK	3463
GETBUF	747A
GETBYT	1AA6
GETCOD	3C39
GETDEV	721A
GETFLP	7033
GETIN2	1A99
GETINT	1A98
GETLEN	3CBC
GETLIN	31C7
GETPAT	35C5
GETPNT	FA1C
GETPTR	7036
GETQ	2B60
GETSPA	6993
GETSTK	652E
GETTRM	3CA7
GETVC1	2D4D
GETVCP	2D46
GETVRM	3C4C
GETYPR	30
GICINI	4066
GIVDBL	182E
GIVINT	183E
GLINE	23E7
GONE	E82
GOSUB	FF6
GOSUB2	1013
GOTO2	102B
GOTRP	666E
GPUTG	28AB
GRPACX	FE46

GRPACY	FE48
GRPCCL	2000
GRPNAM	1800
GRPPRT	4702
GTASPC	4BC3
GTBYTC	1AA5
GTMPRT	65E7
GXPOS	FE42
GYPOS	FE44
HIMEM	FDE6
HLFDE	24F4
HOLD	F968
HOLD2	F963
HOLD5	F948
HOLDB	F933
IADD	589B
ICOMP	567A
IDIV	590F
IMOD	596C
IMULT	58BC
INDJMP	399F
INDSKC	73F1
INDSKE	7402
INEG	595D
INEG2	5968
INEGHL	5953
INIDAT	7A66
INIENT	7B9F
INIGRP	3610
INILIN	9FB
INIMLT	3665
INIT	7B50
INITIO	34D9
INITQ	2B8D
INITRP	6653
INITXT	3541
INKEY	64F3
INLIN	6D2F
INPRT	5B3C
INRART	55AA
INSFLG	FE36
INSLNO	3B2C
INSTR	6BF0
INT	5804
INTCNT	FE32
INTERVAL_OOS	FE18
INTEXP	5F6D
INTFLG	FE2B
INTFR2	1A9C
INTID2	F9A
INTIDX	F99
INTRET	3D80
INTTRP	3048
INTVAL	FE30
INTXT	89A
INXHRT	5615
IOGOR	1C1
IPL	34BA
ISCNTC	6495
ISFLIO	68C2
ISIGN	55DA

ISLET	679E
ISLET2	679F
ISUB	5890
ISVAR	16FD
JMPBNK	3476
KBDDSP	77CC
KBDPRV	FE3E
KBUF	F54F
KBUF-1	F54E
KEY	3120
KEYBUF	FD8B
KEYCHR	3144
KEYINT	3CC2
KEY_OOS	FE06
KILL	34B5
LASTTRK	F9B0
LASTWR	7DB0
LBOERR	D27
LEFT$	6B66
LEFTC	49F8
LEFTUS	6B6D
LEN	6B04
LETCON	10BB
LFILES	73AD
LFPROG	FAE6
LFTQ	2BC0
LHSMIO	6C73
LINGET	FAD
LINKER	AE5
LINLEN	F543
LINOUT	5B48
LINPRT	5B44
LINPT1	6456
LINPT3	12CA
LINPT4	131E
LINSPC	FA3
LINTTB	FD4D
LINWRK	FDB4
LISPRT	1B05
LIST	1AB8
LISTFRE	F99B
LISTSCT	F9B1
LOAD	7121
LOC	7484
LOCATE	2FD1
LODSCN	3831
LOF	749A
LOG	5197
LOHADR	FADD
LOHCNT	FADF
LOHDIR	FADC
LOHMSK	FADB
LPOS	F541
LPTCHR	643A
LPTDSP	7953
LPTLST	F540
LRUN	711F
LSET	722B
MACLNG	21C0
MAF	5376
MAIN	9C4

MAKINT	56C4
MAKUPL	170B
MAKUPS	170C
MAM	5379
MAPXYC	48E9
MAXDEL	FAEB
MAXDRV	F98C
MAXFILES	F98D
MAXUPD	FA0D
MCLEND	2CD0
MCLFLG	FAF1
MCLLEN	FCD4
MCLPTR	FCD5
MCLSCN	21F6
MCLSTX	FCF0
MCLTAB	FAEF
MCLXEQ	22CC
MDM	3036
MDMDSP	791F
MDMFLG	FD49
MDMPRV	FE40
MDM_OOS	FE1B
MEMSIZ	F7A2
MERGE	7122
MFA	5382
MFM	5385
MID$	6B9F
MINDEL	FAE9
MINPLS	17A9
MINUPD	FA10
MKD$	7318
MKI$	7312
MKS$	7315
MLTNAM/TXTCGP	800
MMA	538A
MMF	538F
MOERR	902
MON	7B44
MONERR	7B4A
MONFLG	FE72
MOTOR	2BE5
MOVCNT	FAE3
MOVE	561A
MOVE1	5626
MOVE1R	562D
MOVFM	55ED
MOVFR	55F0
MOVMF	5617
MOVRF	55FB
MOVRM	560E
MOVRMI	5605
MUSIKF	FCD8
NAME	34B0
NAMSC1	6FD6
NAMSCN	6FD3
NEG	55BC
NEGD/SCAN1	22DF
NEGDE	264C
NEGHL	23BB
NEWSTT	E3E
NEXT	6821

NFERR	8F3
NLONLY	F9B2
NOARYS	60CA
NOCLSB	710A
NOFUNS	F8E3
NOROOM	7210
NOSKCR	75D0
NOTELX	FCEA
NOTRFN	D2F
NREAD	4D01
NSETCX	4AF9
NTICSX	FCE2
NTONG1	1186
NULBUF	F992
NULOPN	70C3
NUMLEN	5E80
NUMQ	2BAE
NWRITE	4D21
NWSTRT	E51
NXTCON	E3A
OCTAVX	FCE9
OCTCNS	171A
ODEVLINK	F542
OFFDIO	204A
OFFTRP	65FB
OLDKEYS	FD75
OLDLIN	F7EA
OLDTXT	F7EC
OMERR	6545
OMERRR	6550
ONE	5446
ONEFLG	F7E7
ONELIN	F7E5
ONGOTP	30BB
ONGSBF	FD73
ONTRP	65EB
OOCNVF	19E7
OPEN	7080
OUTCH1	6513
OUTCHAR	18
OUTCON	6407
OUTDLP	6415
OVRMSG	62C
PAD	32BD
PADX	FE2D
PADY	FE2C
PAINT	24FC
PARCHK	16E9
PARDEV.	770B
PARM1	F814
PARMFLG	F8E0
PARMLN2	F87A
PARMPRV	F878
PATWRK	FDDC
PBDHRT	3967
PDL	3280
PGINIT	4CCF
PHA	53F1
PHF	53F6
PINLIN	6D13
PIXSIZ	4CC9

PLAY	2C24
PLAYCNT	FCD9
PLAYF	31DE
PLYTAB	2D87
PNTINI	4BC9
POINT	2346
POIREC	FAE5
POLRTY	FD4C
POPAHT	679B
POPALL	3966
POPHRT	59C9
POSIT	393E
PPA	5406
PPF	540C
PPSWRT	69AC
PRESET	2328
PRGFIN	750C
PRINTW	6446
PRINUS	6273
PRLOGO	4782
PRMLEN	F812
PRMSTK	F810
PRSCNT	FCCE
PSET	232D
PTRFIL	F997
PTRFLG	F7D5
PTRGET	6066
PTRGT2	606B
PTRGTN	611F
PTRGTR	6122
PUFOUT	5B58
PUSHF	55E0
PUT	2FB1
PUTBNK	346A
PUTCOD	3C7B
PUTDEI	6934
PUTFLG	FAE8
PUTFN	FA14
PUTNEW	6959
PUTPNT	FA1A
PUTQ	2B45
PUTSPR	464C
PUTTMP	695D
PUTVRM	3C7E
QINLIN	6D26
QINTA	5788
QLENGX	FCE1
QUEBAK	FB0A
QUETAB	FAF2
QUEUEN	FCD7
QUEUES	FA17
RAMLOW	F500
RAWFLG	F9B9
RCVSFT	FE54
RCVX	79D3
RCVXOF	FE52
RDVDP	3734
READC	4951
READY	9AF
READYR	8C9
REASON	6537

REDDY	89F
REGA	FE64
REGBC	FE62
REGDE	FE60
REGF	FE65
REGFT	FE71
REGHL	FE5E
REGPC	FE5A
REGSP	FE5C
RENCRN	BFB
REPCNT	FA19
REPINI	9C1
REQTRP	6618
RESFIN	66C3
RESTOR	66AE
RETSPR	4606
RETSWI	3377
RETVAR	1700
REVFLG	FE35
RG1SAV	FA07
RIGHT$	6B96
RIGHTC	49CF
RND	5300
RNDCNT	F504
RNDINI	5343
RNDMN2	534C
RNDTAB	F506
RNDX	F984
RS2INT	79D8
RS2IQ	FC8E
RSET	7227
RSTFNK	3498
RSTTRP	660E
RTPROG	FAE7
RTYCNT	FE2A
RUBSW	F545
RUNBNF	FE57
RUNC	656A
SAVBC	F9BB
SAVE	7167
SAVEND	F9B4
SAVENT	FE58
SAVESP	FE73
SAVFLG	F9B3
SAVSCN	37E6
SAVSP	FCCF
SAVSTK	F7DD
SAVTXT	F7DB
SAVVOL	FCD2
SCALXY	48A1
SCAND	22EE
SCANL	4C66
SCANR	4BD0
SCCPTR	1D78
SCMTRP	79C8
SCNBLK	702F
SCNCNT	FE78
SCRATH	6556
SCREEN	459A
SCREEN	FE3A
SCRTCH	6557

SETATR	4980
SETC	4988
SETFIL	7073
SETGSB	3110
SETMAX	7CBA
SETRD	3747
SETS	34AB
SETSTR	6B20
SETTRM	3CB5
SETTRP	6633
SETWRT	373C
SGN	55C6
SHCTRL	FD86
SIGN	28
SIGNC	55A1
SIGNS	55A8
SIN	50D1
SINFIX	245
SIOFLG	FE51
SKPCNT	FAE1
SNDSFT	FE55
SNERR	8ED
SNGEXP	5EF6
SNGFLT	183C
SNTXOF	FE53
SOUND	2BFD
SPACE$	6B4D
SPCFLG	FA09
SPRATR	1B00
SPRITE	45D2
SPRITE_OOS	FE0C
SPRPAT	3800
SPRSIZ	FE3B
SPRTTP	3042
SPSAVE	FE76
SPSVEX	71AB
SQR	5222
SRCCAS	1F34
STAT0	F9BC
STAT1	F9BD
STATFL	FE3D
STICK	3206
STKERR	65C0
STKINI	65B5
STKTOP	F546
STOP	66CB
STOPP	66C8
STOPRG	66CC
STOPTP	303C
STOP_OOS	FE09
STOREC	494A
STPEND	66E3
STPOPT	FD4A
STPRDY	9AE
STPTRP	6601
STR$	6909
STRAD1	6932
STRB$	6904
STRCMP	68CD
STRCPY	6916
STREND	F7F2

STRH$	68FF
STRIG	3056
STRIG_OOS	FE0F
STRINI	692A
STRINI	692C
STRLIT	693A
STRLT2	693E
STRLT3	693D
STRLTI	693B
STRNG$	6B2E
STRO$	68FA
STROUI	697C
STROUT	697D
STRPRT	6980
STRTMS	416E
STTIME	31D3
SUBDE	681A
SUBFLG	F7D1
SWAP	6735
SWIFLG	FE75
SWITCH	337F
SWPTMP	F8E9
TAN	5120
TANFIX	24D
TDOWNC	4A14
TEMP	F7D3
TEMP2	F7E8
TEMP3	F7C9
TEMP8	F7CB
TEMP9	F8E4
TEMPCX	FCEB
TEMPPT	F7A4
TEMPST	F7A6
TERMIN	3CB3
TIME	31BD
TIMEVAR	FE2E
TMERR	905
TOFF	6730
TON	672F
TONPRX	FCE4
TOTEXT	3768
TRCFLG	F8F1
TRGFLG	FA08
TRIGF	3263
TRPTBL	FDEB
TSTACK	F9BE
TSTOP	14D9
TTYCHR	6455
TTYPOS	F791
TUPC	4A3F
TXPSAV	FDE8
TXTTAB	F54A
UMULT	5873
UNTERM	3CB4
UPC	4A59
USERR	105C
USFLG	F7D2
USRTAB	F52B
VAL	6BC0
VALDBL	577A
VALINT	56C7

VALSC2	2279
VALSCN	226D
VALSNG	577E
VALTYP	F793
VARGET	22C4
VARTAB	F7EE
VCBA/METREX	FCDA
VCBB	FCFF
VCCC	FD24
VCXLEN	FCDC
VCXPTR	FCDD
VCXSPT	FCDF
VDFACS	5643
VDPRegO	FE3C
VDPWRT	3536
VINT	57F8
VLZADR	F54C
VLZDAT	F8E8
VMOVAF	563C
VMOVAM	561E
VMOVE	5622
VMOVFA	5634
VMOVFM	5637
VMOVMF	563F
VNEG	55B5
VOICAQ	FB0E
VOICBQ	FB8E
VOICCQ	FC0E
VOICEN	FCD1
VOLUMX	FCEC
VPEEK	46F2
VPOKE	46D8
VRFSCN	3887
VSIGN	55D0
WRTGIC	4086
WRTVDP	372A
XCHGX	23D8
XCHGY	23CE
XDCOMP	5689
XDELT	23B1
XOOFLG	FABE
XTF	5397
YDELT	23C3
ZERO	55AC
back/COLOR	FA0B
front/COLOR	FA0A

Anhang C

Crossreferenz der ROM-Marken

18	OUTCHAR	0575 3BB0 6400
19	FORSZC	542B
28	SIGN	47E0
30	GETYPR	
4C	DRVLEN	4CF4
109	FILLEN	2BEF 3F72 7D2F 7F7F
1C1	IOGOR	30E7
245	SINFIX	
24B	COSFIX	
24D	TANFIX	
24F	ATNFIX	
295	ALPTAB	0BC8
62C	OVRMSG	6397
684	DIVMSG	
800	MLTNAM/TXTCGP	
		350A 3588 36A2 36D7 381C 3870 38BD
		3A83 4300 432F 439C 43A2 43A8 43AE
		4DFA
84C	CONSTR	7BD0
89A	INTXT	5B3E
89F	REDDY	09BF 1C85 1EEE
8A4	BRKTXT	6701
8AA	FNDFOR	106A 682B
8C9	READYR	0091 6D7A 73AB
8ED	SNERR	000E 0094 0A0A 0B21 0E5F 0F6B 1035
		115B 120F 155B 1735 19AF 1A1D 1A35
		1D14 2BEC 2C85 30B9 31B6 6072 619C
		67C6 70C8 7CC2
8F0	DVOERR	4FBC 5912 5F80
8F3	NFERR	682E
8F6	DDERR	367E 61D3
8FF	DVERR	174F 1CE5 4DE8 4E35 4E4F 4F22 4FE1
		5137 528F 52FE 56C1 5A7F 602D 687D
902	MOERR	162F
905	TMERR	0DAA 1522 15B2 55BA 55D2 56BB 56E3
		5768 5786 5802 6750
907	ERROR	0097 0D2A 0FA1 105F 1078 11F3 14B3
		1A0C 1F06 2057 3903 61E3 6554 6724
		697A 69B3 6AA4 7622
92E	ERESET	08CD
98D	ERRFIN	6704
9AE	STPRDY	08CA 1AE2 1D73 6707 6FCB
9AF	READY	009D 1ACF 7207 7C29
9C1	REPINI	626C
9C4	MAIN	09F9 0AE3 1223 7165
9FB	INILIN	0166 2B89 2BAA
A28	EDENT	0910 0A20 0C61 27A4 301C 3E35 4B9D
		57D5 59FA 642A 6E90 74FE 763F
AC9	FINI	0A70 0A83 1C8C 1EF8 4845
AE5	LINKER	00A0 29C9 6546 71EF
AE9	CHEAD	00C4 0ACE
B27	FNDLIN	09E3 0A5E 104B 1C78 1D20 1D26 1DBA
		66BA
B44	CRUNCH	00C1 0A2A
BFB	RENCRN	
D27	LBOERR	

```
D2F     NOTRFN
E3A     NXTCON      109C 689D
E3E     NEWSTT      00A3 0C7E 0E84 0FF2 100A 1087 1DFB
                    321E 329C 32E2 34D9 402F 6818 68BC
                    6FD1 7204 74BA 7A49 7CD8
E51     NWSTRT      3082
E82     GONE        74DD
EAD     CHRGTR      0F22 0FAB 0FBA 122C 22E3 22F3 2FBE
                    6950 706C 70B6 72D2 7368
EAE     CHRGT2      0AFF 23AF
EB2     CHRCON      0016
F99     INTIDX      0163 6185
F9A     INTID2      67AD
F9E     FCERR       00A6 0877 0879 087B 087D 087F 0881
                    0883 0885 0887 0889 11F0 1215 16BF
                    1940 1A79 1AAD 1C82 1D19 1D2E 1D39
                    1D43 2203 22BE 23A9 250C 2520 2884
                    2918 291D 2980 29BF 2A59 2AF6 2AFB
                    2B39 2B41 2C0B 2D44 2E39 2EDE 305E
                    3077 3101 3171 319A 31E5 320C 3288
                    32C3 33A5 34D7 4553 45A4 45C0 464E
                    465A 46A4 46CC 46E4 46FA 48E6 5199
                    519C 5225 61EF 6295 637E 676C 67CD
                    67D5 6B18 6BA7 6C00 6CA1 6CC1 733A
                    73B6 743B 76C0 77D2 77D6 77D8 77DC
                    781F 7821 7823 7825 7827 7829 784B
                    784D 7851 7929 792B 792F 7951 795B
                    795D 795F 7961 7963 7965 79C6 7B48
                    7CC7
FA3     LINSPC      0B13 0B1E 11FC 1CFE 1D08
FAD     LINGET      0A02 0C85 0FFC 1029 112C 115F 11B6
                    120C 1718 1D11 66B6
FF6     GOSUB       019D
1013    GOSUB2      1F6E 35C0 45F8 4E26 66AC
102B    GOTO2       11BE 2947 3A99 4E48 4E63 4E81 4E98
                    4EA8 4FFD 5630 5EF0
105C    USERR       0A6C 1139 116C 66C0
10BB    LETCON      752F
1186    NTONG1      6A9F
12CA    LINPT3
131E    LINPT4      2054
1365    FINPRT      015D 1270 1493 63C4 65CE 7081 7521
1457    DOASIG      1465 7516
14C5    FRMEQL      19AA 45E2 6CAA
14C8    FRMPRN      16EA 176E 17AF 6093 6097 6BF2
14CA    FRMEVL      00AF 0DAE 0DE6 10CF 1226 128D 18E6
                    1A9A 1AA7 1CBA 1F15 21C5 26C9 2859
                    2C3A 3184 46D9 6372 63D4 6B38 6C06
                    6C13 6FD4 7234 7702
14CB    FRMCHK      0E1B 3C94 3C97 6274
14D9    TSTOP       6ABD
162D    EVAL        00AC 14D7 6A93
16E9    PARCHK      1784 1847
16FD    ISVAR       1638 2148 22C2
1700    RETVAR      4A20 6132
170B    MAKUPL      0121 0BBB 0BD8 0D36
170C    MAKUPS      0CFE 171F 173B
1715    CNSGET
171A    OCTCNS      0D19 165D 59D9
17A9    MINPLS      5A30
1800    GRPNAM      1C5B 21A0 2A1A 2A24 3617 3792 37F6
```



```
                    37FF 384A 3853 3897 38A0 3FD5 48AC
                    48C0 49DF 4A3C 4AD6 65FE 6D23 7901
182E    GIVDBL      3B8E 3BF5 6D01 6D11
183C    SNGFLT      00BB 1180 1669 1A3F 1CAB 3236 32BB
                    32CE 32D4 4700 66CC 6B05 6B11 6BC4
                    6C25
183E    GIVINT      17F7
19E7    OOCNVF      10DC 18F1 731C
1A98    GETINT      0DC2
1A99    GETIN2      00BE 0F9B 22F8 22FE 2658
1A9C    INTFR2      1AAA
1AA5    GTBYTC      0160 132E 16AA 72EE 74ED
1AA6    GETBYT      00B2 1184 11EB 177D 1A48 1A60 1A6D
                    1CB4 23A1 2519 2C06 2C11 2FE3 2FF4
                    3001 301F 302B 3059 3071 316B 31E0
                    4560 4575 4584 459F 45B4 4628 4655
                    469F 46BF 46EA 6B32 6C9D 6CF2 706F
                    70B9 72D5 736B 7419 7CBF
1AA9    CONINT      00B8 3207 3264 3281 32BE 6B24 6B42
                    6B4E 6BFC 7034
1AB8    LIST        01A9 7195
1B00    SPRATR      36E6 3810 3864 38B1 3E89 4663
1B05    LISPRT      1AFE
1B0E    BUFLIN      1AF8
1C8F    DEL         0A91
1CB9    FRMQNT      00B5 1882 1A42 1CAE 30FC 31D7 67C1
1CC3    FRQINT      1A38 1CA7 1CBE 5E54 7706
1D78    SCCPTR      1E13 7199
1E0D    DEPTR       0A8A 1C71 1EDF 1FC8
1E15    CSAVE       01B7
1E3E    CSBSAV
1E43    CBSAVE      7653
1E7F    CBLOAD      76AD
1EAA    CLOAD       01B9
1F34    SRCCAS      1E82 1EBE 7878
1FA7    CASOPW      1E2C 1E46 786F
1FC6    CASBNH      1E3F 3945 3949 4869
1FE4    CASBNR      1EE6
2000    GRPCCL      0899 2B64 36E9 3796 37FC 3850 389D
                    3F7C 419D 43C2 451B 4962 4B34 4B62
                    5B0E 7A93 7AA3 7AB3 7AC3 7AD3 7AE3
                    7AF3 7B03 7D8D 7FDD
2016    CASIN.      006D 1E77 1E7B 1E9B 1F3A 1F48 1F4F
                    1FED 38D3 38F2 78BD
2026    CASOUT      0076 1E61 1E70 1E74 1FAD 1FB9 1FC1
                    1FD4 1FDC 2070 38E3 78EE
203A    CSROON      006A 1B39 1E85 1F35 1FE5 383B 3888
                    78B7
204A    OFFDIO      0B86 201C 202D 21B7
204D    DIOERR      00D0 3936 6256 7B4E
2059    CWRTON      0073 1E49 1FA8 1FCB 37E7 78E8
206C    CTWOFF      0079 061C 069B 07A6 081E 0833 1E6A
                    1FC4 1FE2 382D 78F4
207C    CTOFF       0070 1EA5 2004 2014 204B 3881 38FE
                    787B 78C4
20E3    DATAW       2032
21C0    MACLNG      29E5 4038 4172 56D6
21F6    MCLSCN      220E 2CC8 2FAB 37A4
223C    FETCHZ      226E 22A5 22B9 2A3D 2A54
2242    FETCHR      21F7 221C 223D 229F 2E9E 2EEE 2F22
225F    DECFET      2231 2A4D 2EAF 2F3A
```

```
226D    VALSCN      2A50 2A5C
2279    VALSC2      2EF3
22C4    VARGET      2273
22CC    MCLXEQ      2A0C 2DB5
22DF    NEGD/SCAN1  23E8 24FD 2653 28B0 2B28 466E
22EE    SCAND       2332 2360 23EF 28BF
2328    PRESET      0209
232D    PSET        0207 0B7A
2346    POINT       08B5 1683 1ACA 1ADA 2BD2 2F9B 318A
                    5607 5614 686B 6B7F 6C3C
2390    ATRSCN      23F2 2502 267D 63A1
23B1    XDELT       2414 2491 28D2
23BB    NEGHL       24C2 25E6 264E 2776 2AB6
23C3    YDELT       240C 2489 28C8
23CE    XCHGY       23D9 240F 28CB
23D8    XCHGX       2406 2483 248C
23E7    GLINE       1377
247C    DOGRPH      2440 2812
2488    DOGRP2
24F4    HLFDE       24CD 2701 2706 28FC 28FF 2902 2964
                    2B16 2B19
24FC    PAINT       0201
264C    NEGDE       275B 277E 27EE 27FE 2A16 2A1E 2A28
                    2A31 2A36 2AAF 2AB9 2B2C
2652    CIRCLE      01FB
28AB    GPUTG       2FC8
29DA    DRAW        01FF 5B7E
2B45    PUTQ        2D71
2B60    GETQ        4193
2B8D    INITQ       4081
2B9E    BCXQ
2BAE    NUMQ
2BC0    LFTQ        2D80 5BCF
2BE5    MOTOR       021B 1E04 2B4E 6A71
2BFD    SOUND       020B
2C24    PLAY        0205
2CD0    MCLEND      21F1
2D46    GETVCP      2C52 2C96 2CD7 40E7
2D4D    GETVC1      4091
2D87    PLYTAB      2C26
2FB1    PUT         01E9
2FB4    GET         01E7
2FD1    LOCATE      0233
3036    MDM         0221 5B74 5BC2 5BF2 5D97 5DA2 5DC6
303C    STOPTP      66C9
3042    SPRTTP      45D6
3048    INTTRP      1A2F 233B 27D9 471F
3056    STRIG       1A2A
30BB    ONGOTP      114E
3110    SETGSB      1174
3120    KEY         0211 7ADA 7D60 7D6F 7FB0 7FBF
3144    KEYCHR      313C
31AF    CLICK       0213
31BD    TIME        167E
31C7    GETLIN      16E2
31D3    STTIME      0E93 7A18
31DE    PLAYF       168D
3206    STICK       0277 4760
3263    TRIGF       0279
3280    PDL         027B 18D7 1A11 5736 6FBF
32BD    PAD         027D
```



```
3377    RETSWI      1692
337F    SWITCH      0215
3420    CHKBNK      33A2 7C4B
3463    GETBNK      012A 33CD 33DB
346A    PUTBNK      0127
3476    JMPBNK      0060
3480    CALBNK      0063
3498    RSTFNK      005A
34A6    DSKO$       0225
34AB    SETS        0227
34B0    NAME        0229
34B5    KILL        022B
34BA    IPL         022D
34BF    DKCOPY      022F
34C4    CMD         0231
34C9    DSKF        027F
34CE    DSKI$       1697
34D3    ATTR$       169C 7A46 7A50
34D9    INITIO      7B65
3512    BREAKX      005D 2094 20EE 2167 2197 3917
3536    VDPWRT      354D 355D 3563 3569 3635 3643 3649
                    364F 3655 365B 3661 3677 3687 368D
                    3693 3699 369F 36CE 3766
3541    INITXT      0048 3775 37DE 7C3F 7C52
35C5    GETPAT      35B2 470D
3610    INIGRP      004B 1400 37E1 4783 4786
3665    INIMLT      004E 37E4
36BE    CLRSPR      45C7
372A    WRTVDP      36EE 36F4 3704 4673 4682 468D 46B1
                    46D4 46EE 49B1 4B31 4B3B 4B8D 4B9A
                    4BB5
3734    RDVDP       4687 46AB 46FD 495D 4966 4977 499C
                    4B5E 4B66
373C    SETWRT      358B 361A 36A5 36D4 370E 372C 3799
                    37AA 37C4 3B68 3C2C 3C83 45DE
3747    SETRD       3735 3C0F 3C51 460C
3750    CHGCLR      4596
3768    TOTEXT      0992 09B0 13D9 670E
3777    CLS         01C1 797B
378B    CLSHRS      3627
37D9    CHGMOD      3838 45AB
37E6    SAVSCN      1E33
3800    SPRPAT      1C2A 36D1 3819 386D 38BA 4293 4521
                    463B
3831    LODSCN      1EC9
3887    VRFSCN      1ECC 2593 25EB 271B 2AC1 2ACD 4B04
                    4F70 7A2A
3915    CHPLPT      0045 643B
3938    CHPSTT      0042 391C 3EA6
393E    POSIT       3014 6DE7
394D    CHPUT       6458
3966    POPALL      00D3 206A 40AF 4765 78A0 796D
3967    PBDHRT      3E03 405B 4196 783F
399F    INDJMP      3A27 6D4A
3A6C    CXDPCS      3964 6DD8 6E50 6E76 6EC1 6EF1
3A71    DSPCSR      76E9
3AA7    CKERCS      3753 395C 6DA9 6E1E 6E86 6ED2 6EDE
                    6EFB 6F1C 6F2F
3AAC    ERACSR
3AC1    BS          39B9 39D1 3B56 601D 6E30 6F58
3AD0    ADVCUR      39CE 6F17 6F40 6F48
```

```
3AF4    CSHOME      371C 39C2 39FE
3AFD    DELLNO      3998 3BCE 6E6D
3B2C    INSLNO      6E5F
3B60    ECL         39E0 3B78 6EE6
3B86    ERAFNK      0054 45CD
3B95    FNKSB       0051 31AB 33E6 34A4 3728
3B9F    DSPFNK      0057 3DE0 45D0
3C39    GETCOD      0154 6DB6 6F10 6F67
3C4C    GETVRM      3A77 3C3A 6E28 6EA8 6EC8
3C5A    CNVCOD      014B 3BE3 3C7C
3C7B    PUTCOD      3982
3C7E    PUTVRM      1BA4 3AB5 3B5C 6E2E 6EAC 6EBB 6ECC
3CA7    GETTRM      0151 3B0B 3B3C 3CB7 6DC8 6DF6 6E3B
                    6E97 6EBE 6EE2 6F03 6F85
3CB3    TERMIN      015A 3B61 6D37 6DE4
3CB4    UNTERM      6E53
3CB5    SETTRM      0157 398A
3CBC    GETLEN      014E 3A3C 3AD7 3AFE 3B2D 3B7C 6E59
                    6F4B
3CC2    KEYINT      0039 66C8
3D80    INTRET      3CD9
3DCA    CHSNS       003C 4041 64BC 64F6
403D    CHGET/TRYIN 003F 64BF 64FB 6D42 7448 77FB
405D    CKCNTC      016F 254C 26F6 744C
4066    GICINI      2D3E 6E05 7BE0
4086    WRTGIC
40BE    BEEP        0148 0203 39B6 7BE3
416E    STRTMS      2D2C
4198    CGTABL      5552
4552    COLOR       01FD 04A0 04ED 05F5 0739 0740
459A    SCREEN      020D
45D2    SPRITE      0E8E 67FA 7D03
4606    RETSPR      1688
464C    PUTSPR      2FB9
46D8    VPOKE       020F
46F2    VPEEK       0263
4702    GRPPRT      029A 1BBC 3956
4782    PRLOGO      5D61 7C4F
48A1    SCALXY      233A 2364 2403 2409 2480 2486 27D8
                    2B3E 471E
48E1    CHKMOD      2505 4651 4750 476C 48CF 48EB 4958
                    498B 49B8 49D1 49E4 49FA 4A1A 4A33
                    4A45 4A5F 4AFA 4BD5 4C6B 4CCA
48E9    MAPXYC      233F 236C 241C 24C8 2529 27DD 28DC
                    2931 4723 47E6 4837
4943    FETCHC      2349 2423 259B 2611 262F 2992 472D
                    47EF 4954 498E 49BE 49D7 49EA 4A00
                    4A76 4A88 4A99 4AA9 4B01 4BFD 4C1B
                    4C50
494A    STOREC      238C 242E 2565 25B9 263A 29A7 4744
                    480E 4C26 4C36
4951    READC       236F 4BE6 4C0F 4C7D 4CB4 4D08 4D2A
4980    SETATR      23A6 2B36
4988    SETC        2342 24DB 27E0 4735 481B 4821 4827
                    482D 4BB9 4CC3 4D3F
49CF    RIGHTC      2495 47FF 4802 481E 483E 4841 4BBC
                    4C89 4CAE 4D0C 4D42
49F8    LEFTC       249A 25F1 482A
4A14    TDOWNC      2577 4747
4A2D    DOWNC       2431 24A8 24B5 29AA 4811 4814 4824
4A3F    TUPC        2572
```



```
4A59    UPC         4830
4AF9    NSETCX      2429 4C31 4C92
4BC3    GTASPC      2600 26B8 2A7C
4BC9    PNTINI      251D
4BD0    SCANR       261E
4C66    SCANL       2645
4CC9    PIXSIZ      28E7 295B
4CCF    PGINIT      298D
4D01    NREAD       299D 33F6
4D21    NWRITE      29A2 371E 4B46 7B99 7DEC
4D86    DECSUB/DSUB 05B4 50EB 510C 5118 515F 529E 52CD
                    535F
4D91    DADOS       68A0
4D94    DADO/DECADD 05B2 521A 5254 52DC 5359 53EC 5871
                    5987
4DF6    DECNRM      4FA0 5341
4E38    DECROU      4DE1 5A7A
4E3D    DECROB      56F1 5EC5
4E44    DECROA
4EF3    DECSRD      4DEB 5D4F 5EBA
4EFE    DECMUL/DMULT
                    05B6 51F6 5202 5368 53CB 5995 5F68
4F46    DECMRN      05E4 5322
4FB7    DDIV/DECDIV 05B8 2F1A 5134 5150 5186 51CF 51F0
                    524E 52E5 536E 59B0 6039
50B8    COS         024B 5124
50D1    SIN         0245 512A
5120    TAN         024D
5139    ATN         024F
5197    LOG         0247 5F62
5222    SQR         0241
526B    EXP         0249 5F6B
5300    RND         0243 02DE 0559 05E7 0658 06C5 06D5
                    0814 0829 1F87
5343    RNDINI      6588
534C    RNDMN2      0386
5364    DMULTO      50BC 50D5 517A 5220 525A 526F 52F0
                    53E4 5FD1 5FE6
5376    MAF         50E5 5103 5147 5156 5228 525D 5298
                    5FF7 6030
5379    MAM         5115 52E2 5316 5356 535C 5365 536B
                    5371 53E8 77A3
5382    MFA         5269 53A2 5F38 5F5C 6007 6049
5385    MFM         5109 514D 515C 528C 531C 5335 53D9
                    5E9E 5F44 5FE1 6036
538A    MMA
538F    MMF         52D3 5307 5353 53D5 5F3E 5FB9 5FEA
                    6004
5397    XTF         0373 5127 5174 51C3 51E4 52BB 5F00
                    5F5F
53F1    PHA         524B 52C7 5EFD 604C
53F6    PHF         50DF 5121 516B 51BA 51D2 51D8 51DB
                    5205 5248 5272 52B2 52CA 539F 53BF
                    5F15 5FDA
5406    PPA         02D2 512D 5183 51CC 51ED 51F3 51FF
                    5217 5251 52C4 52D6 539C 53C8 5F03
                    5F24 5F65 6052
540C    PPF         50E8 5260 5286 529B 52D9 5FED
543E    DBLZER      5289 5E97
5446    ONE         0438 0DDE 5112 514A 517D 51BD 51C6
                    5F41 6033
```

```
55A1    SIGNC       002D
55A8    SIGNS       5687 68F8
55AA    INRART
55AC    ZERO        4E06 4F05 4FC5 5096 5815
55B1    ABSFN       023F
55B5    VNEG        16F9 57F6 5A69 5B6E
55BC    NEG         50CD 53A9 53AC 5759 57F0 590D 598A
55C6    SGN         023B
55C9    CONIA       00C7 17C1 31FA 3277 332C 7985
55D0    VSIGN       1237 55B2 55C7 5B63
55DA    ISIGN       0DC8
55E0    PUSHF       160C 1625 2887 58B2 58F7
55ED    MOVFM       2F17
55F0    MOVFR       15D7
55FB    MOVRF       0E05 0E22 1612
5605    MOVRMI      6852
560E    MOVRM       21D1 2C42 55EE 597E 685D 6892 6A2A
5610    GETBCD      6984
5615    INXHRT      0A1B 0A26 1C37 5BF5 5C03 5D7D
5617    MOVMF       6859
561A    MOVE        01AA
561E    VMOVAM      4D92 68A8
5622    VMOVE       0DFB 111F 5640 673E 6760 6767 696A
                    6C96
5626    MOVE1       0F11 532D 5395
562D    MOVE1R      4F92 50B3
5634    VMOVFA      4DA3
5637    VMOVFM      0F5A 1707 1926 6849 7346
563C    VMOVAF      15CA 15F0 2F11
563F    VMOVMF      0DD7 18FE 68A4 7326
5643    VDFACS
5650    FCOMP       05C8 1CD8 1CE2 28A7 56D4 6861
567A    ICOMP       05D4 688C
5689    XDCOMP      5374 56AF 6055 68AC
56AE    DCOMP       05BC
56B5    FRCINT      0003 0175 026F 05AC 0DB8 1565 17CB
                    17E8 1A9E 1CC4 22C8 266F 26E5 289B
                    2F1D 46DD 46F3 5275 5A9C
56BD    CONIS
56C4    MAKINT      0005 00CD 16C2 175A 1840 2375 2660
                    2890 55CE 5833 58BF 5905 5951 595B
                    59D2 5B49 5F86
56C7    VALINT      5978
56CD    CONIS2      5A76 783B
56DD    FRCSNG      0271 05B0 0E02 0E1F 1601 1CCF 2663
                    26CD 285E 2893 5AAE
56E5    CONSD
56F3    CONSI       56E0 576B 5B01 5D22
56F6    CONSIH      160F 1621 1628 2F0E 5211 534D 58AD
                    58B5 58F3 58FA 5909 5FBD 5FFB 6018
5765    FRCDBL      0273 05A8 0DCD 0DEA 15C7 15DA 1796
                    5295 5AB2 5E94 5FF4
576D    CONDS       5214 5984 5992 59AD 5EFA 5FC0 5FFE
                    601B
577A    VALDBL      5283 5F21 5F2A 5F59
577E    VALSNG      56E6 56FD 583E
5783    CHKSTR/FRCSTR
                    0169 05AE 1387 1773 6277 63D7 6A97
                    6AD6 6C09 6C79 722E 72F9 752B
5788    QINTA       56BE 605A
57E7    DCXBRT
```



```
57E9    FIXER       0275
57F8    VINT        023D 50E2 57ED 57F3 604F
5804    INT         504E
5873    UMULT       290B 2F0A 6212 6250
5890    ISUB        05CC
589B    IADD        05CA 6875
58BC    IMULT       05CE 603E
590F    IDIV        17F4 596E
5953    INEGHL      56FA 57DC 5895 591A 5961 5F90
595D    INEG        55B7
5968    INEG2       00CA 0F44 1677 1832 31C3
596C    IMOD        17EF
597D    FADDS
5980    FADD        05BE 1CEE 585F 58BA 6855
5989    FSUB        05C0
598E    FMULT       05C2 266C 26E2 287E 2898 58FF
5997    FDIVT       162B
5999    FDIV        05C4 26D9
59B2    CONASD      5981 598F 59AA 5EF7
59C5    DCRART      5CDA 5D36
59C7    DCXHRT      5C48 5D77
59C9    POPHRT      00D9 184E 5A72 5A98 61D9 6C21 6CB8
59CB    FINDBL      0C9C 1469 1632 6BDC 75E8 75EC
5B3C    INPRT       09AB 1DD4
5B44    LINPRT      00DC 09DE 0E7B 1AEC 1DCD 7C9F
5B48    LINOUT
5B57    FOUT        1294 1C06 5CAB 690A
5B58    PUFOUT      6384
5E48    FOUTB       6905
5E4C    FOUTD       1BFB 68FB
5E50    FOUTH       1C00 6900
5E80    NUMLEN      56E9 5BD5 5C9E 5D26 5DE5 5EB6 5ED3
                    5EE4
5EF6    SNGEXP      05C6
5F05    DBLEXP      05BA
5F6D    INTEXP      05D2
6061    DIM         018F
6066    PTRGET      017B 10C1 1384 1413 16FE 18A1 18DC
                    29BC 4867 6120 6736 6749 6774 6825
                    6C76 722B 72F6 7528
606B    PTRGT2      1A18
60CA    NOARYS      1288 2581 60BD
611F    PTRGTN      16B6
6122    PTRGTR      612C
61AA    ERSFIN      60BA
61DF    BSERR       587F 5888 624D
6273    PRINUS      1275
6407    OUTCON      001D
6415    OUTDLP      0082 641D 796A
643A    LPTCHR      6449 644E
643D    FINLPT      00A9 09B3 65CB 66FA
6446    PRINTW
6455    TTYCHR      6412
6456    LINPTI      783C
6463    CRDONZ      007F 0967 09BC 1DC2 66FD
646A    FININL
6474    CRDO        007C 126D 12D1 1320 1B01 1FA5 3132
                    63BC 6719
647C    CRFIND
647D    CRFIN       12CE 651B 698E
6495    ISCNTC      0E42 1AD5 4062
```

```
64DF    CKSTTP    64B7 6E08
64F3    INKEY     16D3
6513    OUTCH1    1B09
6520    BLTU      05EE 0722 072F 074E 0760 07A7 07E9
                  0AA8 614D 7D49 7F99
6523    BLTUC     2CBB 2D06 6A80
652E    GETSTK    0D9A 0FF9 14D1 18F6 196E 22DA 261B
                  61F7 66A4
6537    REASON    1FFE 6221 6521
6545    OMERR     00DF 1CDC 1FFB 621E 67ED 67FB 7218
                  7D04
6550    OMERRR
6556    SCRATH    01AB 0272 0729
6557    SCRTCH    0172 1ED3 715E 71DA 7211 739F 7C1A
656A    RUNC      00E2 0AD7 0FE3 6FCE 71F6
6571    CLEARC    0FEF 67A7 680C
6577    CLEARO    7CD5
65B5    STKINI    7C0E
65C0    STKERR    0932
65E7    GTMPRT    750D
65EB    ONTRP     30AC
65FB    OFFTRP    30B1
6601    STPTRP    30B6 669F
660E    RSTTRP    1083
6618    REQTRP    3CE5 3CF9 3D01 3D1A 3D28 3FC9 64C6
                  79CE
6633    SETTRP    4866
663D    FRETRP    669C
6653    INITRP    6578
666E    GOTRP     0E4F
66AE    RESTOR    019B 659D
66C3    RESFIN    00D2 017A 1481 2069 40AE 4764 789F
                  796C
66C8    STOPP     01A3
66CB    STOP      64DD
66CC    STOPRG
66CF    ENDST     0185
66D9    CONSTP
66E3    STPEND    1391 13F7 142C
66E6    ENDCON    08E3
6709    CTROPT
670B    CTRLPT    64D5
671B    CONT      01B5
672F    TON       01C7 17D1 3442 3456
6730    TOFF      01C9 655B
6735    SWAP      01CB
676E    ERASE     01CD 066C 06BD 06CA 06D3 06DA 076C
                  7DA4 7FF4
679B    POPAHT
679E    ISLET     0F68 0F7A 606F
679F    ISLET2    0BBE 1635 2225 22AE 520E 55CB 607A
                  6083
67A6    CLEAR     01A7
681A    SUBDE
6821    NEXT      0189 3095
68C2    ISFLIO    0A07 12A3 12E9 1341 1438 1485 1AD2
                  6408 646D 647E 6D17 74DA
68CD    STRCMP    0991 09AF 13D8 1584 1676 31C2 5662
                  670D
68FA    STRO$     0267
68FF    STRH$     0269
```



```
6904    STRB$     026D
6909    STR$      0259
6916    STRCPY    1118 19BE 6C91
692A    STRIN1    64FF 6B21
692C    STRINI    0085 4613 6AA7 6B55 72C0 7320 743F
6932    STRAD1
6934    PUTDEI
693A    STRLIT    1297 690D 697E
693B    STRLTI    13E7 1653
693D    STRLT3    1398 75DE
693E    STRLT2    1455
6959    PUTNEW    0088 4620 6AC2 6B2C 6B94 72C3 745F
695D    PUTTMP    19C1
697C    STROUI    5B45
697D    STROUT    008E 09A4 09C2 13C2 148F 1C88 1DC9
                  1EF1 1F98 208D 5B41 6387 7C58 7CA5
6980    STRPRT    12D4 12D8 13ED 63E8
6993    GETSPA    016C 691A 692D 6B79 726A
69AC    PPSWRT    1BE1 570C
69BB    GARBA2    6D07
6A8C    CAT       14F7
6AD5    FRESTR    008B 21CE 2C3F 3188 45E6 68CE 6B09
                  6C19 6CAE 6FD8 723B 7272 7335
6AD8    FREFAC    1859 6910 6981 6D04
6ADB    FRETM2    63C0 6AAF 6C2B
6ADC    FRETMP    68D9 6AAB 6B91
6AF3    FRETMS    1114 111B 6ADD
6B04    LEN       0257
6B10    ASC       025D
6B14    ASC2      1F19 6B47
6B20    SETSTR    025F
6B2A    FINBCK    6913 7329
6B2E    STRNG$    16D8
6B4D    SPACE$    0265
6B66    LEFT$     0235 6CD0 6FE1
6B6D    LEFTUS    63E5
6B96    RIGHT$    0237
6B9F    MID$      0239
6BC0    VAL       025B
6BF0    INSTR     16CE
6C73    LHSMIO    1A25
6CF7    FRE       0251
6D13    PINLIN    09F0
6D26    QINLIN    13F3 1427
6D2F    INLIN     138C
6FD1    .C33
6FD3    NAMSCN    00E5 7085 7125 7168 7625 7685
6FD6    NAMSC1    013C
702F    SCNBLK    00E8 7711 771E
7033    GETFLP    00EB 7488 749E 74B4 74CA
7036    GETPTR    00EE 16AE 7074 70CF 70F7 71E1 72D9
                  738F
7067    FILSCN    0142 73BC 7423
7073    SETFIL    00F1 06C0 7162 74F4
7080    OPEN      01E3 428D 4305
70C3    NULOPN    00F4 713F 718D 765E 76B3
70EA    CLSFIL    00F7 71AE 71FD 7376 7383 74E1 7512
                  76F2 76FF
710A    NOCLSB    00FA 77CE 7819 7895 7955
711F    LRUN      0FEC
7121    LOAD      01ED
```

```
7122    MERGE       01EF
7167    SAVE        01F7
71AB    SPSVEX      017E 767A
7209    CHKTOP      0133 2F48 70E2 71C6
7210    NOROOM      0130
721A    GETDEV      012D
7227    RSET        01F5
722B    LSET
72CD    FIELD       01E5 79D8
7312    MKI$        028F
7315    MKS$        0291
7318    MKD$        0293
732B    CVI         0283 6237 6882
732E    CVS         0285
7331    CVD         0287
7375    CLOSE       01EB 070A
737D    CLSALL      00FD 338B 65A9 66D6 71DD 7CCE
738B    CLSCLR
73AD    LFILES      01F9
73B2    FILES       01F1
73B8    DPUT        2FCF
73B9    DGET        2FCC
73C9    FILOUT      640B
73CA    FILOU1      0100 1274 1A24 71B9 767E 7682
73F1    INDSKC      0103 6F9E 71CA 7462 7538 7553 7576
                    759A 75A8 75B9 76C4 76C8 76CE 76D2
                    76DA 76DE
7402    INDSKE      0178 77F2 78A6
7406    FIXINP      16DD
7469    CLRBUF      0106 05DD 05FD 0748 710B 7D9C 7FEC
7474    DOCLR       0109 7470
747A    GETBUF      0145 746A
747D    GETBF1      0136 72DC
7484    LOC         028B
749A    LOF         028D
74B0    EOF         00D6 0289
74C6    FPOS        0281
74D9    DIROG       0A3B
74E6    FILINP      13C9 7525
74E8    FILGET      1268
750C    PRGFIN      09B6 6FC4
7514    FILIND      143B
7520    DLINE       0709 137E
75D0    NOSKCR      010C 78E5
75FA    DERBFN      010F 700D 71A9 71EC 73C7 73DA 7400
                    7498 74AE 74C4 74D7 75CE 772D 773A
                    773F 7786 77EA 7831 7866 7937 798B
75FD    DERFAD      0112 70D2
7600    DERFDR      74E4
7603    DERFNF      0115
7606    DERFND      0118 7077 748B 74A1 74B7
7609    DERFOV      0139 7304
760C    DERIFN      703C 70BD 7508
760F    DERIER      011B 7060 70DC 7108
7612    DERRPE      011E 742D 7465 753B
7615    DERSAP      013F
7618    DERSOO      71D7 77D0 781B 7845 7923 7957
7624    BSAVE       021F
7684    BLOAD       021D
76EA    CHKBRN      1EA8
770B    PARDEV      6FE5
```



```
7788    DEVTBL      7747
77A8    GENDSP      70E8 7102 73C1 73EF 7491 74A7 74BD
                    74D0
77CC    KBDDSP      779E
7817    CRTDSP      77A6
7841    CASDSP      77A0
791F    MDMDSP      5098 5B30 77A2
7953    LPTDSP      05E8 77A4
798E    CHKMDM      7C23
79C2    DIAL        0223
79C8    SCMTRP      0E48 3C75 5854
79D3    RCVX        0E45
79D8    RS2INT      3CD3
79DC    BOOT        7A43 7C20
7A66    INIDAT      7BDB
7A84    FNKROM      349F
7B44    MON         0219
7B4A    MONERR      2052
7B50    INIT        0001
7B9F    INIENT      33B6
7CBA    SETMAX      0217
7CDA    DEFILE      6812 7C0B 7CD2
```

Anhang D

Crossreferenzliste des System-RAMs

F500	RAMLOW	2830 7BBE 7BCD 7E11 7E20
F504	RNDCNT	1969 2781 7C3C 7E8C
F506	RNDTAB	
F52B	USRTAB	0C14 1876
F53F	ERRFLG	0938 1147 11AE 1665
F540	LPTLST	64FC 7449
F541	LPOS	12B7 12F5 134E 1835 33F1 392E 6420
		6433 6437 6442 6452 648C
F542	ODEVLINK	1260 12B1 12EF 1347 136A 1AB6 640E
		643F 6485 73B0
F543	LINLEN	12BF 1A7C 1A85 33FA 3A32 3ABA 3AC6
		3B70 3BD0 3C0B 3C28 3CA0 6DC1 6E32
		6E49 6E93 6EA1 6EB1 6F08 6F4F
F545	RUBSW	
F546	STKTOP	4610 6549 65B7 67FF 6998 6D0B 720B
		725C 7BF9 7C5F 7CE8 7D1C 7E4C 7EAF
		7F38 7F6C
F548	CURLIN	08D3 08EB 091E 09CB 0DA3 0E6E 1002
		1016 1043 1096 11D6 1A02 1AC4 1F91
		3FBB 64EB 6675 66E7 672C 6898
F54A	TXTTAB	0AE6 0B28 1D7D 1E3C 1EE3 1EF4 6558
		656E 66B0 6C89 71A2 724D 73A2 7C17
		7E67
F54C	VLZADR	090C 091A 6BD0 6BE2
F54E	KBUF-1	0B62 10F9 4EB8 7BF0 7E43
F54F	KBUF	0ABA 0B52
F68D	BUFMIN	13CD 6470 6D54 6FB5 75D4
F68E	BUF	1AFB 1B0F 22A8 6DAD 6DFF 6F9B 7558
F790	ENDBUF	7BE8 7E3B
F791	TTYPOS	12C6 1303 1318 1353 183A 33EE 645F
		6464 6491
F792	DIMFLG	6068 617E 61A4 61CF 6201
F793	VALTYP	0031 0CA5 0CCB 0F4A 10CB 10D5 10DF
		14F1 151D 158E 15F4 1853 18B1 18E1
		1901 191F 1934 5623 5647 56CA 5B7C
		6010 6041 60B2 60EE 613C 61C0 61E6
		625A 650F 6846 6878 6967 7343
F798	CONSAV	0EC3 0ECE 0F2D 0FBE 102C 1C0B
F7A2	MEMSIZ	6597 67B8 6808 69BC 7C00 7CED 7D12
		7E53 7F3D 7F62
F7A4	TEMPPT	65C8 66E0 695F 6971 69CE 6AF7 6B01
F7A6	TEMPST	65C5 66DD 69CA
F7C4	DSCTMP	19B5 6930 695A
F7C5	DSCPTR	1CC6 4616 6AB3 6B27 6B5E 7323
F7C7	FRETOP	659A 696D 699C 69A8 69BF 6A53 6A7D
		6AE7 6AEF 6D0E 7261
F7C9	TEMP3	14E2 1545 155E 18CA 18ED 1910 195C
		1963 19A7 51B7 5208 5245 5263 5292
		52F3 56B4 57E8 5B4D 5B77 5B96 5DB5
		5E7F 5E8E 61FC 622F 75E1
F7CB	TEMP8	5F1E 5F2D 5F89 6025 6A33 6A3E
F7CD	ENDFOR	0D73 0D88 0D9E 2D9C 656A 7E16 7ECD
F7D1	SUBFLG	0D68 18D9 1A13 29B9 29C2 60B6 60CC
		65E3 6771 6778
F7D2	USFLG	13B7 140C 141D 1449 147C 6282 628C

		6390
F7D3	TEMP	0E37 1065 108C 10C7 6575 65E8 6815
		6828 68B6 7CCA 7F1A
F7D5	PTRFLG	1053 1D7A 1D8F 1E0E 6561
F7D6	AUTFLG	09D4 09ED 09F6 0A10 0A38 0A5B 0A68
		0A7B 121B 655E 6D1D 6DA0
F7D7	AUTLIN	09DA 0A55 121F
F7D9	AUTOP	0A47 0DA7 1207 1218
F7DB	SAVTXT	093B 09D1 0A2F 0E52 1021 13C5 66DA
		66F4
F7DD	SAVSTK	092F 0D92 0E56 1073 64D9 654E 65BC
		68B2
F7DF	ERL	0921 0942 098A 11D3 1674
F7E1	DOT	0929 0A44 0FA8 1AE9
F7E5	ONELIN	0951 0A9D 113D 11A4 128F 157A 6591
		684D 6B3D
F7E7	ONEFLG	08DB 0957 1141 119E 11C2 11CB 11E5
		56DC 658C 666F 66D2
F7E8	TEMP2	0AD4 0ADD 14DA 14DD 16F5 18B6 18CE
		18E9 1905 1957 5C5E 5D14 5D80 5DC1
		5DD4 61A0 6270
F7EA	OLDLIN	094A 66F1 6728
F7EC	OLDTXT	094E 6594 66F7 671C
F7EE	VARTAB	0AA0 0AAC 1C91 1C9D 1E36 1EEB 1EFD
		611B 6568 65A0 67F1 69E4 719C 71F3
		7253 7C5B 7CFF 7EAB 7F4F
F7F0	ARYTAB	1CA0 55B4 6115 6156 61AD 65A3 6741
		6756 69DE 6A08
F7F2	STREND	10FF 1CA3 6146 6151 61B3 6224 6530
		65A6 6784 6791 69C6 6A21 6C83 6CF8
F7F4	DATPTR	1407 66C4
F7F6	DEFTBL	0F87 657D
F810	PRMSTK	197B 1982 19C4 19D2 1F3F 1F60 1F6F
		3DA5 4E64 4FB1 4FFE 56A7 65E0 665F
		6CE6 7BF3 7E46
F812	PRMLEN	1966 1988 60D9 65D4
F814	PARM1	198D 60DC
F878	PARMPRV	69D8 7BF6 7E49
F87A	PARMLN2	1918 1939 1985 1998 65DA
F8E0	PARMFLG	60D4 610B 6112
F8E1	ARYTA2	60E0 60FE 6118 69E1 69E8 6A19
F8E3	NOFUNS	19A4 19E3 60D0 65D7
F8E4	TEMP9	69DB 6A00 6A0E
F8E6	FUNACT	199B 199F 19DA 19DE 4904 492B 49C4
		49F0 4A7F 4AA2 65DD
F8E8	VLZDAT	0913 6BD4
F8E9	SWPTMP	673B 6764
F8F1	TRCFLG	0E71 6732
F8F2	FBUFFR	5399 53A5 53B1 53B7 53BC 53C2 53D2
		53E1 57EC 5C5A 5CAE 5F3B 5FB6 5FCE
		5FDD 6001
F91A	FMLTT1	
F91B	FMLTT2	
F91D	DECTMP	5005 5017 57B6 57C9
F91F	DECTM2	5001 5013
F921	DECCNT	1D3D 5009 501A 5039
F923	FACCU	0029 0CC8 0DEE 10E2 152D 1616 185E
		2862 4D9E 4DD1 4E2E 4F09 4FC0 506B
		50BF 50C4 50D8 50EE 513A 519F 51B2
		522B 52F6 5338 5362 5377 5386 5390
		540D 559D 55A2 55AE 55BD 55E7 55F7

```
                    5600 5618 5644 565D 569A 5704 5738
                    578D 57D2 580A 59A2 59A6 5A55 5A60
                    5A8B 5B0A 5B1A 5BDD 5CA2 5D52 5D84
                    5E9A 5ED6 5EDC 5F0E 601F 616A
F925    FACLOW      0CAC 0CD2 0F54 10E9 10F1 129C 12AE
                    151A 1526 1580 15C2 15FC 161A 1703
                    1777 17D4 19B8 1ED0 1ED6 55D8 55E2
                    55F2 55FC 564D 56B7 56C5 56F4 5707
                    595E 599A 599E 5ADF 5AF6 5E1E 5E2D
                    616F 6178 627D 63DD 63EB 650A 6962
                    6A8F 6A9C 6AD9 6C0F
F933    HOLDB
F948    HOLD5
F963    HOLD2
F968    HOLD
F974    ARG         4D87 4D95 4DCE 4F01 4FB8 5130 5232
                    537A 5383 538B 5407 561F 5635 563D
                    568A 586E 59B9 5F06 5F33
F984    RNDX        5302 530D 5330 5347 5350
F98C    MAXDRV
F98D    MAXFILES    67DD 680F 7038 7379 7387 7394 7D08
                    7D24 7F58 7F74
F98E    FILTAB      7043 7D0D 7F5D
F990    DRVTAB
F992    NULBUF      7D44 7F94
F994    CURDRV      41AC
F995    DRVPTR
F997    PTRFIL      0AD1 0AE0 12E5 133D 1370 68C7 707A
                    70FC 7117 7142 71E6 721D 73CF 73F5
                    740D 7414 745B 747B 75C2 76B6 7834
F999    FREPLC
F99B    LISTFRE
F99D    FILMOD/RUNFLG
                    6FC7 7155 7200
F99E    FILNAM      1F0E 1F21 1F56 1F65 1FB4 4BDB 4C2D
                    6FE9 7004
F9A7    FILNM2      1F42 1F9B
F9B0    LASTTRK
F9B1    LISTSCT
F9B2    NLONLY      0909 65AC 65B3 6FBC 6FC1 704B 70EF
                    714F 71FA 7215 737E 739C 761C 7BEB
                    7E3E
F9B3    SAVFLG
F9B4    SAVEND      1E39 1E51 1FCF 719F 71B2 71BF 763A
                    7666 76D7
F9B6    DSKBSY
F9B7    ERRCNT
F9B8    ERRCN1
F9B9    RAWFLG
F9BA    EBCFLG
F9BB    SAVBC
F9BC    STAT0
F9BD    STAT1
F9BE    TSTACK
FA00    FRSTID      33AD 4215 4221 4275 7BA8 7BD8 7C7B
                    7DFB 7E2B 7ECB
FA02    CLICK       31B9 3406 401A
FA03    CSRY        2FD3 3032 31CF 33EB 3572 3614 3669
                    397A 39AC 3A73 3AAE 3ADF 3BA9 4716
                    476F 477E 6D33 6E21 6E64 6E82 6E9E
                    6ED8 6EFE 6F45 6F55 6F5E
```



```
FA04    CSRX        3A35 3AEC 4753 4762 4769 645B
FA05    CSRSW       33F4 3A63 3A6D 3AA8 464C
FA06    CNSDFG      33FD 3B8B 3B96 3BA6 3CBD 3DDB
FA07    RG1SAV      3418 3550 3557 3638 363D 367A 3681
                    36BF 36C8
FA08    TRGFLG      3CE8
FA09    SPCFLG      3D11
FA0A    front/COLOR 232E 2391 36E2 375A 4558 4593 4710
FA0B    back/COLOR  2329 379C 37C7
FA0C    BORCLR      3755 378C 37B8 458B
FA0D    MAXUPD      24D8
FA10    MINUPD      1751 24F0 2948 3A9A 3C18 3C35 3F8D
                    4602 461D 4EA9 4EEA 562A 5631
FA13    ATRBYT      250F 4713 478F 47C4 4833 4985 49A4
                    4B38 4B74 4BB2 4C9C 4CBE 4D3C
FA14    PUTFN       4D37
FA17    QUEUES      2BE1
FA19    REPCNT      3409 3532 3D69 3DAC
FA1A    PUTPNT      3521 3DC6 400B 4017 64CC
FA1C    GETPNT      3524 3DC3 4012 4050 4057 64CF
FA1E    FNKSTR      3127 317C 349C 3BBF
FABE    XOOFLG
FABF    CDMMSK
FAC0    CHKROM/CLOC 4911 493F 4947 494E 4A0B 4A17 4A30
                    4A42 4A5C 4A6F 4AB9 4AF3 7C1D 7E6D
FAC2    CMASK       29D8 48FA 4926 4944 494B 4A0F 4ABD
FAC3    ASPECT      010E 26E8 2824 700C 71A8 71EB 73C6
                    73D9 73FF 7497 74AD 74C3 74D6 75CD
                    772C 7739 7785 7936
FAC5    CENCNT      26AF 27A1
FAC7    CLINEF      2676 269C 26A4 27B1 27B9 2872
FAC8    CNPNTS      2672 2763 278C 288A 5223
FACA    CPLDTF      26A8 27A9 27C9 4EB6 77E9 7830 7865
                    798A
FACB    CPCND       2740 276A 2770 2794
FACD    CPCNT       163F 2747 2766 2789 2791 62B4 62BB
                    63B2 63CB
FACF    CRCSU       26EF 2715 272B
FAD1    CSTCN       0C73 26AC 2799
FAD3    CSCLX       2679 26C1 26D6 2738 2A70 2A85 2A8B
                    2AA8
FAD4    CSAVE       25B3 25D6 2634 263F 4C00 4C20 4C53
                    4CEC 4D7A
FAD6    CSAVEM      25B6 25D9 2637 2642 4C03 4C23 4C56
                    4D56 4D60 4D68 4D6E 4D75 4D81
FAD7    CXOFF       274D 2773 2779 27EB 27F1
FAD9    CYOFF       2757 27F7 27FB
FADB    LOHMSK      2559 25A1
FADC    LOHDIR      254F 2586 25A8
FADD    LOHADR      2555 259E
FADF    LOHCNT      255D 2598
FAE1    SKPCNT      25BC 2622
FAE3    MOVCNT      25AB 25C0 2625
FAE5    POIREC      1528 2569 25A4 25DD 25FA 260D
FAE6    LFPROG      2604 2649
FAE7    RTPROG      2608 262B
FAE8    PUTFLG      28AC 2997
FAE9    MINDEL      24A1 24BC 24E0 28CF 2905 2925
FAEB    MAXDEL      24C5 24E5 28D9 28EB
FAED    ARYPTR      4CD0 4D4D 4D7D
FAEF    MCLTAB      21C2 21FE 2C29 5197
```

```
FAF1    MCLFLG      21EA 29E2 2CC5
FAF2    QUETAB      7A7D
FB0A    QUEBAK
FB0E    VOICAQ      4075
FB8E    VOICBQ
FC0E    VOICCQ
FC8E    RS2IQ
FCCE    PRSCNT      2C2E 2D22 2D30 2D61 2D65
FCCF    SAVSP       2C35 2CB5 2CEB
FCD1    VOICEN      2C77 2C8B 2CD4 2D4B 2D6D 2D7B
FCD2    SAVVOL      2E72 2F90
FCD4    MCLLEN      21E2 2244 2261 22D0 2C9E 2CCB 2CDA
FCD5    MCLPTR      21F4 224C 2251 2265 2269 22D3 2CA7
                    2CE0
FCD7    QUEUEN      40F6 418D
FCD8    MUSIKF      31E9 3D45 406B 4164 416F 4189
FCD9    PLAYCNT     2D28 4174
FCDA    VCBA/METREX 163C 2D4F 417E
FCFF    VCBB        4181 44EE
FD24    VCCC        4184
FD49    MDMFLG      3CCA 7991 79A1
FD4A    STPOPT      3388 33D8
FD4B    FRCNEW      1EDB
FD4C    POLRTY      2147 216B 3CA9
FD4D    LINTTB      33F7 371F 4CF6 7793
FD65    FSTPOS      3B13 6D3B 6F8B 6F92
FD67    CODSAV      3A7A 3AB1
FD68    FNKSWI      3096 3400 3BCA 3DD4
FD69    FNKFLG      6662
FD73    ONGSBF      0E4B 6626 662A 6635 6639 6648 664F
                    666B
FD74    CLIKFL      3D61 401F 4026
FD75    OLDKEYS     340C 3D71 3DB1
FD80    ACTKEYS     3D8E 3D9B 3DB4
FD86    SHCTRL      3BBB 3DD7 3EC7 3EE4 3F17 3F2A 3F82
                    3F93 3FCF 3FFC
FD8B    KEYBUF      403A 7A80 7A82
FDB3    BUFEND
FDB4    LINWRK      3592 35A4 35BB 3BCD 3C12 3C2F
FDDC    PATWRK      35B5 35E9 4726
FDE4    BOTTOM      7BC7 7C11 7C85 7E1A 7E61 7ED5
FDE6    HIMEM       1B43 67B3 6804 7BC1 7CDC 7E14 7F2C
FDE8    TXPSAV      1EB9 309B 3884
FDEA    CASATR      1E26 1E2F 1EC1 1F52 1FBE 3832
FDEB    TRPTBL      6654 667D
FE06    KEY_OOS     0977 0B84 0D43 0ED7 1740 1C12 3ED9
                    59FF 635F 6DAF 774F
FE09    STOP_OOS    303E 35D3 64C2 64E0 6A2D
FE0C    SPRITE_OOS  0C67 2F6D 3044 3CE2 5BB4
FE0F    STRIG_OOS   0CB7 1197 3066 35CF 3D17
FE18    INTERVAL_OOS
                    16CA 3052 3D25
FE1B    MDM_OOS     1B29 1B9E 3038 79CA
FE2A    RTYCNT      7562 7A3C 7B67 7DBA
FE2B    INTFLG      0BAD 2D1B 33E3 4006 4046 6497 64A7
                    6D3F
FE2C    PADY        239B 2853 2FEE 32D1 3319 4668 4699
                    7694
FE2D    PADX        32CB
FE2E    TIMEVAR     31C0 31DB 3D31 3D35
FE30    INTVAL      1A2B 3105 315A 315F 3164 3D2B 5E71
```



```
FE32    INTCNT      3108 3D1D 3D2E
FE34    ESCCNT      1698 169D 396D 3A18
FE35    REVFLG      3A69 3BB3 3C72
FE36    INSFLG      6D5A 6DEE 6E11 6EF5
FE37    CSTYLE      3A5C 3A8F
FE38    CAPST       3403 3EFC 3F78 3FEC 505F
FE39    FLBMEM      1FF1 761F 7658 7677
FE3A    SCREEN      12A8 12FD 1E23 2FD7 31CA 340F 3751
                    3769 376F 377F 37DA 37EA 3835 383E
                    388B 3952 3B9B 3DCE 45A7 45B7 478B
                    47DC 48E2
FE3B    SPRSIZ      3412 36C4 36F9 45C3 4642 46C2
FE3C    VDPReg0     2134 2D13 3308 3312 3415 3542 3547
                    362A 362F 366C 3671 3C3B 3D52 40A2
                    6F11 6F68
FE3D    STATFL      1646 3073 316D 3284 341B 3CDD 7731
FE3E    KBDPRV      7726 77E2 77F5 7803 780C 7812
FE3F    CASPRV      785E 7892 78AA 78D2 78DB 78E1
FE40    MDMPRV      62DA 77FC
FE41    BRDATR      40FA 4BCD 4BE2 4C74 4CB8
FE42    GXPOS       2312 2353 2382 23B2 23DE 23E2 2446
                    2469 2473 265D 26F2 2809 296C
FE44    GYPOS       2323 234F 2386 23C4 23D0 23D4 244C
                    2454 245E 2466 280F 297B
FE46    GRPACX      1689 2306 230F 235B 237A 2443 2806
                    2817 2AD7 2FBA
FE48    GRPACY      2317 2320 2357 237E 2449 280C 281E
                    2AD2
FE4A    DRWFLG      29DF 2ABF 2ADB 2AE5
FE4B    DRWSCL      2AFF 2B03
FE4C    DRWANG      2A73 2AF2
FE4D    DATCNT      05B5 79D4
FE51    SIOFLG      50FC
FE52    RCVXOF
FE53    SNTXOF      5140 778F
FE54    RCVSFT
FE55    SNDSFT
FE56    ADDPM
FE57    RUNBNF      768A 769A 76EB
FE58    SAVENT      1285 1E58 1E96 762F 7648 766C 76E3
                    76F9
FE5A    REGPC       59E8 59ED 5A14 5A19 5A1E
FE5C    REGSP       5B83
FE5E    REGHL       1BFC
FE60    REGDE       1A19 3C41
FE62    REGBC       1276
FE64    REGA        16D4
FE65    REGF        0FE4 30AD 30B2
FE71    REGFT
FE72    MONFLG      204E
FE73    SAVESP
FE75    SWIFLG      337A 773B 7867 7C2E 7C48 7E7E 7E98
FE76    SPSAVE      3391 33BA 753C
FE78    SCNCNT      156E 159F 15A9 2C67 2C7D 2EB4 3D58
                    62EB 7C08 7E07
```

Anhang E

Crossreferenzliste der Hook-Einspr)nge

```
FE79    0026 1768 215C 3974 3CC7 3EAE 5855
        7504 77E6 782D 7862 7BB4
FE7C    12A0 7B3A
FE7F    5791 73C4
FE82    73FD

FE85    137A 1D96
FE88    0E11
FE8B    74D4
FE8E    09B9

FE91    0BEE
FE94    09C5
FE97    7458
FE9A    7495

FE9D    75CB
FEA0    65C1
FEA3    770C
FEA6    6AF4

FEA9    7105
FEAC    0C17
FEAF    6572 67AA
FEB2    7485 749B 74B1 74C7

FEB5    716B
FEB8    73B3
FEBB    74AB
FEBE    150D

FEC1    1BD3 2D83 5C7D 7729
FEC4    247D
FEC7    7128
FECA    74C1

FECD    21A2 51F5 5201 5F67 606C
FED0    0EB1 6F6C 6F73 6F7A 70C0
FED3    08D0
FED6    1BB8

FED9    6475
FEDC    0591 178F
FEDF    227F 6513 698A 705D
FEE2    164A 6580

FEE5    10EB 1281 759E 7737
FEE8    707D
FEEB    70D9
FEEE    1062

FEF1    0C46 2C13 4DBA 73A8
FEF4    1AB9 5C67 694C
FEF7    0032 0AF9 0CA6 0CCC 0F2E 0F4B 102D
        1319 14E3 14F2 15F5 1C26 1C44 5648
```



```
        6011 6042 6391 656B 6879 75AF
FEFA    0893 0D5A 1640 31EC 397B 3BAA 5AF2
        5BFF

FEFD    1A32
FF00    12DF 4446 7B23
FF03    1CCA 364C
FF06    0A32 130A 4EBD 69A0 6F98 755A 7B76
        7B8D 7DC9 7DE0

FF09    001A 6405
FF0C    0D2D
FF0F    77A9
FF12    73D7

FF15    68C3
FF18    096D
FF1B    098E
FF1E    13B4 2D67 34DB 6CE9

FF21    09C7 0BC5 1AC0 2366 582F
FF24    0B4C
FF27    1366
FF2A    14D4

FF2D    71A6
FF30    0ACA
FF33    71E9
FF36    0ADA 3D79

FF39    17A1
FF3C
FF3F    1A70 3E28
FF42    6D14

FF45    6D27
FF48    6D30
FF4B    4474 6F96
FF4E    3B87

FF51    3BA0
FF54    0E3F
FF57    0E89
FF5A    3D38

FF5D    7934
FF60    793A
FF63    793F
FF66    7944

FF69    7949
FF6C    794E
FF6F    79C3
FF72    79D9

FF75    30BC
FF78    3FCC
FF7B    3FD9
FF7E    0181

FF81    3EB6
```

```
FF84            7B45
FF87
FF8A            34A7

FF8D            34AC
FF90            34B1
FF93            34B6
FF96            34BB

FF99            34C0
FF9C            34C5 575F
FF9F            34CA
FFA2            34CF

FFA5            34D4
FFA8            7B4B
FFAB            3585
FFAE            3959

FFB1            3772
```



```
L0:
        JP INIT /7B5Ø

        DEFB OB5H,056H,0C4H,056H,0
L8:
        LD A,(HL)       ; Vergleich mit dem
        EX (SP),HL      ; auf RST 8H folgenden
        CP (HL)         ; Byte
        INC HL          ; falls ungleich
        EX (SP),HL      ; SYNTAX ERROR
        JP NZ,SNERR     ; 8ED
L10:                    ; naechstes Zeichen des
        INC HL          ; BASIC-Programms
        LD A,(HL)
        CP 03AH
        RET NC
        JP CHRCON       ; EB2

OUTCHAR:/ 18            ;18
        PUSH AF         ; Ausgabe eines Zeichens
FORSZC:/ 19             ; auf dem aktuellen Geraet
        CALL LFF09
        JP OUTCON       ; 64Ø7
        NOP
L20:                    ; ZERO falls  HL=DE
        LD A,H
        SUB D
        RET NZ
        LD A,L
        SUB E
        RET
L26:
        DEFW  OFE79H    ; Basis-Adresse der Einspruenge
                        ; im RAM
SIGN: /  ;28H
        LD A,(FACCU)    ; Vorzeichentest im BASIC AKKU
        OR A            ; ZERO falls Zahl=0
        JP NZ,SIGNC     ; 55A1
        RET

GETYPR:/;30H   F793H
30H     LD  A,(VALTYP)  ; Test Datentyp
33H     CP 08H          ; AKKU     VALTYP FLAGS          A
35H     JP L17DA        ; INTEGER     2    NZ  C M PE    -1
                        ; STRING      3     Z  C P PE     0
                        ; SINGLE      4    NZ  C P PO     1
                        ; DOUBLE      8    NZ NC P PE     5
L38:
        JP KEYINT       ; Timer Interrupt Einsprung /17DA
        JP CHSNS        ; 3CC2
        JP CHGET/TRYIN  ; 3DCA
        JP CHPSTT       ; 4Ø3D
        JP CHPLPT       ; 3938
        JP INITXT       ; 3915
        JP INIGRP       ; 3541
        JP INIMLT       ; 3610
        JP FNKSB        ; 3665
        JP ERAFNK       ; 3695
        JP DSPFNK       ; 3888
        JP RSTFNK       ; 3B9F
```

```
          JP BREAKX     ;3498
          JP JMPBNK     ;3512
          JP CALBNK     ;3476
JP 3480 > NOP
   L66:
          JP L180/180
          JP CSROON/203A
          JP CASIN/2016
          JP CTOFF/207C
          JP CWRTON/2059
          JP CASOUT/2026
          JP CTWOFF/206C
          JP CRDO/6474
          JP CRDONZ/6463
          JP OUTDLP/6415
          JP STRINI/692C
          JP PUTNEW/6959
          JP FRESTR/6A05
          JP STROUT/6970
          JP READYR/8C9
          JP SNERR/8ED
          JP ERROR/907
          JP L981/981
          JP READY/9AF
          JP LINKER/AE5
          JP NEWSTT/E3E
          JP FCERR/F9E
          JP FINLPT/643D
          JP EVAL/162D
          JP FRMEVL/14CA
          JP GETBYT/1AA6
          JP FRMQNT/1CB9
          JP CONINT/1AA9
          JP SNGFLT/183E
          JP GETIN2/1AA9
          JP CRUNCH/0B44
          JP CHEAD/0AE9
          JP CONIA/56C9
          JP INEG2/5968
          JP MAKINT/56C4
          JP DIOERR/204D
          JP POPALL/3966
          JP EOF/34B0
          JP POPHRT/59C9
          JP LINPRT/5B44
          JP OMERR/6545
          JP RUNC/6566
          JP NAMSCN/6FD3
          JP SCNBLK/702F
          JP GETFLP/7033
          JP GETPTR/7036
          JP SETFIL/7078
          JP NULOPN/70C3
          JP CLSFIL/70EA
          JP NOCLSB/710A
          JP CLSALL/7370
          JP FILOU1/73CA
          JP INDSKC/73F1
          JP CLRBUF/7469
          JP DOCLR/7474
          JP NOSKCR/75D0
```



```
          JP DERBFN/75FA
          JP DERFAD/75FD
          JP DERFNF/7603
          JP DERFND/7606
          JP DERIER/760F
          JP DERRPE/7612
          JP MAKUPL/170B
          JP LBCA/0BCA
          JP PUTBNK/396A
          JP GETBNK/3463
          JP GETDEV/721A
          JP NOROOM/7210
          JP CHKTOP/7209
          JP GETBF1/747D
          JP DERFOV/7609
          JP NAMSC1/6FD6
          JP DERSAP/7615
          JP FILSCN/7067
          JP GETBUF/747A
          JP BEEP/40BE
          JP CNVCOD/365A
          JP GETLEN/3CBC
          JP GETTRM/3CA7
          JP GETCOD/3C39
          JP SETTRM/3CB5
          JP TERMIN/3CB3
          JP FINPRT/1365
          JP GTBYTC/1AA5
          JP INTIDX/0F80
          JP INILIN/6993
          JP CHKSTR/FRCSTR/5783
          JP GETSPA/6993
          JP CKCNTC/406D
          JP SCRTCH/6557
          JP FRCINT/5685
          JP INDSKE/7402
          JP PTRGET/6066
          JP SPSVEX/71A8

L180:
          CALL LFF7E        ; NMI Routine
          EI
          RET

L185:                       ; Ausfuehrungs- Tabelle fuer
          DEFW ENDST        ; TOKEN  81H    END
          DEFW LOD65        ;        82H    FOR
          DEFW NEXT
          DEFW L109B        ; DATA
          DEFW L13D2        ; INPUT
          DEFW DIM
          DEFW L1405        ; READ
          DEFW L10CO        ; LET
          DEFW L1028        ; GOTO
          DEFW LOFE2        ; RUN
          DEFW L1225        ; IF
          DEFW RESTOR
          DEFW GOSUB
          DEFW L1061        ; RETURN
          DEFW L109D        ; REM
          DEFW STOPP
          DEFW L1265        ; PRINT
```

```
        DEFW CLEAR
        DEFW LIST
        DEFW SCRATH     ; NEW
        DEFW L1124      ; ON
        DEFW L1A52      ; WAIT
        DEFW L188A      ; DEF
        DEFW L1CAD      ; POKE
        DEFW CONT
        DEFW CSAVE
        DEFW CLOAD
        DEFW L1A4C      ; OUT
        DEFW L125D      ; LPRINT
        DEFW L1AB3      ; LLIST
IOGOR:/1C1
        DEFW CLS
        DEFW L1A6C      ; WIDTH
        DEFW L109D      ; ELSE
        DEFW TON
        DEFW TOFF
        DEFW SWAP
        DEFW ERASE
        DEFW L11EA      ; ERROR
        DEFW L119D      ; RESUME
        DEFW L1C6C      ; DELETE
        DEFW L11F5      ; AUTO
        DEFW L1CF0      ; RENUM
        DEFW L0F5C      ; DEFSTR
        DEFW L0F5F      ; DEFINT
        DEFW L0F62      ; DEFSNG
        DEFW L0F65      ; DEFDBL
        DEFW L1374      ; LINE
        DEFW OPEN
        DEFW FIELD
        DEFW GET
        DEFW PUT
        DEFW CLOSE
        DEFW LOAD
        DEFW MERGE
        DEFW FILES
        DEFW LSET
        DEFW RSET
        DEFW SAVE
        DEFW LFILES
        DEFW CIRCLE
        DEFW COLOR
        DEFW DRAW
        DEFW PAINT
        DEFW BEEP
        DEFW PLAY
        DEFW PSET
        DEFW PRESET
        DEFW SOUND
        DEFW SCREEN
        DEFW VPOKE
        DEFW KEY
        DEFW CLICK
        DEFW SWITCH
        DEFW SETMAX
        DEFW MON
        DEFW MOTOR
        DEFW BLOAD
```



```
        DEFW BSAVE
        DEFW MDM
        DEFW DIAL
        DEFW DSKO$
        DEFW SETS
        DEFW NAME
        DEFW KILL
        DEFW IPL
        DEFW DKCOPY
        DEFW CMD
        DEFW LOCATE

L235:                   ; Tabelle mit Ausfuehrungs-
                        ; adressen  TOKEN OFFH XX  <80H
        DEFW LEFT$
        DEFW RIGHT$
        DEFW MID$
        DEFW SGN
        DEFW VINT
        DEFW ABSFN
        DEFW SQR
        DEFW RND
SINFIX:/245
        DEFW SIN
        DEFW LOG
        DEFW EXP
COSFIX:/248
        DEFW COS
TANFIX:/24D
        DEFW TAN
ATNFIX:/24F
        DEFW ATN
        DEFW FRE
        DEFW L1A37      ; INP
        DEFW L1839      ; POS
        DEFW LEN
        DEFW STR$
        DEFW VAL
        DEFW ASC
        DEFW SETSTR
        DEFW L1CA6      ; PEEK
        DEFW VPEEK
        DEFW SPACE$
        DEFW STRO$      ; OCT$
        DEFW STRH$      ; HEX$
        DEFW L1834      ; LPOS
        DEFW STRB$      ; BIN$
        DEFW FRCINT     ; CINT
        DEFW FRCSNG     ; CSNG
        DEFW FRCDBL     ; CDBL
        DEFW FIXER      ; FIX
        DEFW STICK
        DEFW TRIGF      ; STRIG
        DEFW PDL
        DEFW PAD
        DEFW DSKF
        DEFW FPOS
        DEFW CVI
        DEFW CVS
        DEFW CVD
        DEFW EOF
```

```
        DEFW LOC
        DEFW LOF
        DEFW MKI$
        DEFW MKS$
        DEFW MKD$

ALPTAB: 295                     ; L295
        DEFW T_BST_A
        DEFW T_BST_B
        DEFW T_BST_C
        DEFW T_BST_D
        DEFW T_BST_E
        DEFW T_BST_F
        DEFW T_BST_G
        DEFW T_BST_H
        DEFW T_BST_I
        DEFW T_BST_J
        DEFW T_BST_K
        DEFW T_BST_L
        DEFW T_BST_M
        DEFW T_BST_N
        DEFW T_BST_O
        DEFW T_BST_P
        DEFW T_BST_Q
        DEFW T_BST_R
        DEFW T_BST_S
        DEFW T_BST_T
        DEFW T_BST_U
        DEFW T_BST_V
        DEFW T_BST_W
        DEFW T_BST_X
        DEFW T_BST_Y
        DEFW T_BST_Z


T_BST_A:                        ;* 2C9
        DC 'UTO'
        DEFB 0A9H
        DC 'ND'
        DEFB 0F8H
        DC 'BS'
        DEFB 06H
        DC 'TN'
        DEFB 0EH
        DC 'SC'
        DEFB 015H
        DC 'TTR$'
        DEFB 0E9H
        DEFB 0

T_BST_B:
        DC 'SAVE'
        DEFB 0CEH
        DC 'LOAD'
        DEFB 0CDH
        DC 'EEP'
        DEFB 0C0H
        DC 'IN$'
        DEFB 01DH
        DEFB 0
```



```
T_BST_C:
        DC 'LICK'
        DEFB 0C8H
        DC 'LOSE'
        DEFB 0B4H
        DC 'OPY'
        DEFB 0D6H
        DC 'ONT'
        DEFB 099H
        DC 'LEAR'
        DEFB 092H
        DC 'LOAD'
        DEFB 09BH
        DC 'SAVE'
        DEFB 09AH
        DC 'SRLIN'
        DEFB 0E8H
        DC 'INT'
        DEFB 01EH
        DC 'SNG'
        DEFB 01FH
        DC 'DBL'
        DEFB 020H
        DC 'VI'
        DEFB SIGN
        DC 'VS'
        DEFB 029H
        DC 'VD'
        DEFB 02AH
        DC 'OS'
        DEFB 0CH
        DC 'HR$'
        DEFB 016H
        DC 'IRCLE'
        DEFB 0BCH
        DC 'OLOR'
        DEFB 0BDH
        DC 'LS'
        DEFB 09FH
        DC 'MD'
        DEFB 0D7H
        DEFB 0

T_BST_D:
        DC 'ELETE'
        DEFB 0A8H
        DC 'ATA'
        DEFB 084H
        DC 'IM'
        DEFB 086H
        DC 'EFSTR'
        DEFB 0ABH
        DC 'EFINT'
        DEFB 0ACH
        DC 'EFSNG'
        DEFB 0ADH
        DC 'EFDBL'
        DEFB 0AEH
        DC 'SKO$'
        DEFB 0D1H
        DC 'EF'
```

```
        DEFB 097H
        DC 'SKI$'
        DEFB 0EAH
        DC 'SKF'
        DEFB 026H
        DC 'RAW'
        DEFB 0BEH
        DC 'IAL'
        DEFB 0D0H
        DEFB 0

T_BST_E:/
        DC 'LSE'
        DEFB 0A1H
        DC 'ND'
        DEFB 081H
        DC 'RASE'
        DEFB 0A5H
        DC 'RROR'
        DEFB 0A6H
        DC 'RL'
        DEFB 0E1H
        DC 'RR'
        DEFB 0A6H
        DC 'XP'
        DEFB 0BH
        DC 'OF'
        DEFB 02BH
        DC 'QV'
        DEFB 0FBH
        DEFB 0

T_BST_F:/
        DC 'OR'
        DEFB 082H
        DC 'IELD'
        DEFB 0B1H
        DC 'ILES'
        DEFB 0B7H
        DC 'N'
        DEFB 0DEH
        DC 'RE'
        DEFB 0FH
        DC 'IX'
        DEFB 021H
        DC 'POS'
        DEFB 027H
        DEFB 0

T_BST_G:/
        DC 'OTO'
        DEFB 089H
        DC 'O TO'
        DEFB 089H
        DC 'OSUB'
        DEFB 08DH
        DC 'ET'
        DEFB 0B2H
        DEFB 0
        DC 'EX$'
        DEFB 01BH
```



```
        DEFB 0

T_BST_H:/
        DC 'NPUT'
        DEFB 085H
        DC 'F'
        DEFB 08BH
        DC 'NSTR'
        DEFB 0E5H
        DC 'NT'
        DEFB 05H
        DC 'NP'
        DEFB 010H
        DC 'MP'
        DEFB 0FCH
        DC 'NKEY$'
        DEFB 0ECH
        DC 'PL'
        DEFB 0D5H
        DEFB 0

T_BST_I:/
T_BST_J:/
        DEFB 0

T_BST_K:/
        DC 'ILL'
        DEFB 0D4H
        DC 'EY'
        DEFB 0C7H
        DEFB 0

T_BST_L:/
        DC 'PRINT'
        DEFB 09DH
        DC 'LIST'
        DEFB 09EH
        DC 'POS'
        DEFB 01CH
        DC 'ET'
        DEFB 088H
        DC 'OCATE'
        DEFB 0D8H
        DC 'INE'
        DEFB 0AFH
        DC 'OAD'
        DEFB 0B5H
        DC 'SET'
        DEFB 0B8H
        DC 'IST'
        DEFB 093H
        DC 'FILES'
        DEFB 0BBH
        DC 'OG'
        DEFB 0AH
        DC 'OC'
        DEFB 02CH
        DC 'EN'
        DEFB 012H
        DC 'EFT$'
        DEFB 01H
```

```
        DC 'OF'
        DEFB 02DH
        DEFB 0

T_BST_M:/
        DC 'OTOR'
        DEFB 0CCH
        DC 'ERGE'
        DEFB 0B6H
        DC 'OD'
        DEFB 0FDH
        DC 'KI$'
        DEFB 02EH
        DC 'KS$'
        DEFB 02FH
        DC 'KD$'
        DEFB GETYPR
        DC 'ID$'
        DEFB 03H
        DC 'ON'
        DEFB 0CBH
        DC 'AX'
        DEFB 0CAH
        DC 'DM'
        DEFB 0CFH
        DEFB 0

T_BST_N:/
        DC 'EXT'
        DEFB 083H
        DC 'AME'
        DEFB 0D3H
        DC 'EW'
        DEFB 094H
        DC 'OT'
        DEFB 0E0H
        DEFB 0

T_BST_O:/
        DC 'PEN'
        DEFB 0B0H
        DC 'UT'
        DEFB 09CH
        DC 'N'
        DEFB 095H
        DC 'R'
        DEFB 0F9H
        DC 'CT$'
        DEFB 01AH
        DC 'FF'
        DEFB 0EBH
        DEFB 0

T_BST_P:/
        DC 'RINT'
        DEFB 091H
        DC 'UT'
        DEFB 0B3H
        DC 'OKE'
        DEFB 098H
        DC 'OS'
```



```
        DEFB 011H
        DC 'EEK'
        DEFB 017H
        DC 'SET'
        DEFB 0C2H
        DC 'RESET'
        DEFB 0C3H
        DC 'OINT'
        DEFB 0EDH
        DC 'AINT'
        DEFB 0BFH
        DC 'DL'
        DEFB 024H
        DC 'AD'
        DEFB 025H
        DC 'LAY'
        DEFB 0C1H
        DEFB 0

T_BST_Q:/
        DEFB 0

T_BST_R:/
        DC 'ETURN'
        DEFB 08EH
        DC 'EAD'
        DEFB 087H
        DC 'UN'
        DEFB 08AH
        DC 'ESTORE'
        DEFB 08CH
        DC 'EM'
        DEFB 08FH
        DC 'ESUME'
        DEFB 0A7H
        DC 'SET'
        DEFB 0B9H
        DC 'IGHT$
        DEFB 02H
        DC 'ND'
        DEFB 008H
        DC 'ENUM'
        DEFB 0AAH
        DEFB 0

T_BST_S:/
        DC 'CREEN'
        DEFB 0C5H
        DC 'PRITE'
        DEFB 0EEH
        DC 'WITCH'
        DEFB 0C9H
        DC 'TOP'
        DEFB 090H
        DC 'WAP'
        DEFB 0A4H
        DC 'ET'
        DEFB 0D2H
        DC 'AVE'
        DEFB 0BAH
        DC 'PC('
```

```
        DEFB 0DFH
        DC 'TEP'
        DEFB 0DCH
        DC 'GN'
        DEFB 04H
        DC 'QR'
        DEFB 07H
        DC 'IN'
        DEFB 09H
        DC 'TR$'
        DEFB 013H
        DC 'TRING$'
        DEFB 0E3H
        DC 'PACE$'
        DEFB FORSZC
        DC 'OUND'
        DEFB 0C4H
        DC 'TICK'
        DEFB 022H
        DC 'TRIG'
        DEFB 023H
        DEFB 0

T_BST_T:/
        DC 'HEN'
        DEFB 0DAH
        DC 'RON'
        DEFB 0A2H
        DC 'ROFF'
        DEFB 0A3H
        DC 'AB('
        DEFB 0DBH
        DC 'O'
        DEFB 0D9H
        DC 'IME'
        DEFB 0EFH
        DC 'AN'
        DEFB 0DH
        DEFB 0

T_BST_U:/
        DC 'SING'
        DEFB 0E4H
        DC 'SR'
        DEFB 0DDH
        DEFB 0

T_BST_V:/
        DC 'AL'
        DEFB 014H
        DC 'ARPTR'
        DEFB 0E7H
        DC 'POKE'
        DEFB 0C6H
        DC 'PEEK'
        DEFB 18H
        DEFB 0

T_BST_W:/
        DC 'IDTH'
        DEFB 0A0H
```



```
        DC 'AIT'
        DEFB 096H
        DEFB 0

T_BST_X:/
        DC 'OR'
        DEFB 0FAH
        DEFB 0

T_BST_Y:/
        DEFB 0

T_BST_Z:/
        DEFB 0


L587:                   ; OPERATOR / TOKEN TABELLE
        DEFB '+'+128,0F3H
        DEFB '-'+128,0F4H
        DEFB '*'+128,0F5H
        DEFB '/'+128,0F6H
        DEFB '^'+128,0F7H
        DEFB '\'+128,0FEH
        DEFB ''''+128,0E6H
        DEFB '>'+128,0F0H
        DEFB '='+128,0F1H
        DEFB '<'+128,0F2H
        DEFB 0


L59C:   ; Operator Praezedenz Tabelle
        DEFB 79H,79H,7CH,7CH,7FH,50H,46H,3CH,32H,28H,7AH,7BH

L5A8:                   ; fuer Datenumwandlung
        DEFW FRCDBL
        DEFW 0
        DEFW FRCINT
        DEFW FRCSTR
        DEFW FRCSNG

L5B2:                   ; Block mit double Arithmetik
        DEFW DECADD     ; Adressen
        DEFW DECSUB
        DEFW DECMUL
        DEFW DECDIV
        DEFW DBLEXP
        DEFW DCOMP

L5BE:                   ; wie oben fuer Single
        DEFW FADD
        DEFW FSUB
        DEFW FMULT
        DEFW FDIV
        DEFW SNGEXP
        DEFW FCOMP

L5CA:                   ; und fuer INTEGER
        DEFW IADD
        DEFW ISUB
        DEFW IMULT
        DEFW L161E
```

```
        DEFW INTEXP
        DEFW ICOMP



L5D6:                   ; Fehlertexte
        DEFB 0
        DEFB 'NEXT without FOR',0
        DEFB 'Syntax error',0
        DEFB 'RETURN without GOSUB',0
        DEFB 'Out of DATA',0
        DEFB 'Illegal function call',0
OVRMSG: /62C
        DEFB 'Overflow',0
        DEFB 'Out of memory',0
        DEFB 'Undefined line number',0
        DEFB 'Subscript out of range',0
        DEFB 'Redimensioned array',0
DIVMSG: /684
        DEFB 'Division by zero',0
        DEFB 'Illegal direct',0
        DEFB 'Type mismatch',0
        DEFB 'Out of string space',0
        DEFB 'String too long',0
        DEFB 'String formula too complex',0
        DEFB 'Can''t continue',0
        DEFB 'Undefined user function',0
        DEFB 'Device I/O error',0
        DEFB 'Verify error',0
        DEFB 'No RESUME',0
        DEFB 'RESUME without error',0
        DEFB 'Unprintable error',0
        DEFB 'Missing operand',0
        DEFB 'Line buffer overflow',0
        DEFB '?',0
        DEFB '?',0
        DEFB 'FIELD overflow',0
        DEFB 'Internal error',0
        DEFB 'Bad file number',0
        DEFB 'File not found',0
        DEFB 'File already open',0
        DEFB 'Input past end',0
        DEFB 'Bad file name',0
        DEFB 'Direct statement in file',0
        DEFB 'Sequential after PUT',0
        DEFB 'Sequential I/O only',0
        DEFB 'File not OPEN',0
CONSTR: /84C                    ;*84C  BEI 7B00H
        DEFW  00D3H             ;* RAM Initialisierungsbereich
        DEFW  00C9H             ;* 0F500H-0F54DH
        DEFW  0
        DEFW  04A35H
        DEFW  099CAH
        DEFW  01C39H
        DEFW  09876H
        DEFW  09522H
        DEFW  098B3H
        DEFW  0DD0AH
        DEFW  09847H
        DEFW  0D153H
```



```
        DEFW  09999H
        DEFW  01A0AH
        DEFW  0989FH
        DEFW  0BC65H
        DEFW  098CDH
        DEFW  077D6H
        DEFW  0983EH
        DEFW  0003AH
        DEFW  0
        DEFW  09E00H
        DEFW  09E0FH
        DEFW  09E0FH
        DEFW  09E0FH
        DEFW  09E0FH
        DEFW  09E0FH
        DEFW  09E0FH
        DEFW  09E0FH
        DEFW  09E0FH
        DEFW  09E0FH
        DEFW  09E0FH
        DEFW  0000FH
        DEFW  0
        DEFW  02700H
        DEFW  000E0H
        DEFW  0FA22H
        DEFW  0FFFEH
        DEFW  0BFFFH
        DEFW  000F9H
        DEFB  0
INTXT:  /89A
        DEFB ' in ',0
REDDY:  /89F
        DEFB 'Ok',13,10,0
BRKTXT: /8A4
        DEFB 'Break',0


FNDFOR: /8AA            ; testet ob FOR-Konstrukt auf
        LD  HL,04H      ; dem Stack
        ADD HL,SP
L8AE:
        LD A,(HL)
        INC HL
        CP 082H
        RET NZ
        LD C,(HL)       ; FOR gefunden
        INC HL
        LD B,(HL)
        INC HL
        PUSH HL
        LD H,B
        LD L,C
        LD A,D
        OR E
        EX DE,HL
        JR Z,L8C1
        EX DE,HL
        RST 20H
```

```
L8C1:
        LD  BC,016H
        POP HL
        RET Z               ; Return falls FOR mit gleichem Index
        ADD HL,BC           ; gefunden
        JR L8AE
READYR: 8C9
        LD  BC,STPRDY
        JP ERESET           ; 92E
L8CF:
        CALL LFED3
        LD HL,(CURLIN)
        LD A,H
        AND L
        INC A
        JR Z,L8E2
        LD  A,(ONEFLG)
        OR A
        LD  E,015H
        JR NZ,ERROR         ; 907
L8E2:
        JP ENDCON           ; 66E6
L8E5:
        JR ERROR            ; 907
L8E7:                       ; div. Fehler Einspruenge
        LD HL,(OF7CFH)      ;
        LD (CURLIN),HL      ; F548
SNERR:  8ED
        LD  E,02H
        DEFB 1
DVOERR: 8F0
        LD E,11
        DEFB 1
NFERR:  8F3
        LD E,1
        DEFB 1
DDERR:  8F6
        LD E,10
        DEFB 1
L8F9:
        LD E,18
        DEFB 1
L8FC:
        LD E,22
        DEFB 1
DVERR:  8FF
        LD E,6
        DEFB 1
MOERR:  902
        LD E,24
        DEFB 1
TMERR:  905
        LD E,13             TYPE MISMATCH
ERROR:  907
        XOR A
        LD (NLONLY),A       ; F982
        LD HL,(VLZADR)      ; F54C
        LD A,H
        OR L
        JR Z,L91C
        LD  A,(VLZDAT)      ; F8E8
```



```
        LD (HL),A
        LD  HL,00H
        LD (VLZADR),HL      ; F54C
L91C:
        EI
        LD HL,(CURLIN)      ; F548
        LD (ERL),HL         ; F7DF
        LD A,H
        AND L
        INC A
        JR Z,L92B
        LD (DOT),HL         ; F7E1
L92B:
        LD  BC,L934
ERESET: 92E
        LD HL,(SAVSTK)      ; F7DD
        JP STKERR           ; 66C0
L934:
        POP BC
        LD A,E              ; ERROR #
        LD C,E
        LD (ERRFLG),A       ; Sichern der Fehlernummer / F53F
        LD HL,(SAVTXT)      ; Letztes ausgefuehrtes Byte/ F7D8
        LD (OF7E3H),HL
        EX DE,HL
        LD HL,(ERL)         ; Zeile  F7DF
        LD A,H
        AND L
        INC A
        JR Z,L950           ; JP, falls Eingabephase
        LD (OLDLIN),HL      ; F7EA
        EX DE,HL
        LD (OLDTXT),HL      ; Byte / F7EC
L950:
        LD HL,(ONELIN)      ; ON ERROR Zeilennummer /F7E5
        LD A,H
        OR L
        EX DE,HL
        LD  HL,ONEFLG       ; F7E7
        JR Z,L963           ; JP, falls nicht angegeben
L95B:
        AND (HL)
        JR NZ,L963          ; JP, falls geschachtelte ON ERRORS
        DEC (HL)
        EX DE,HL
        JP LE62             ; ON ERROR ausfuehren
L963:
        XOR A
        LD (HL),A           ; Loesche ON ERROR FLAG
        LD E,C
        CALL CRDONZ         ; evtl. neue Zeile/ 6463
        LD  HL,05D6H        ; Fehlertext-Adresse
        CALL OFF18H
        LD A,E              ; linearisierte Fehlernummer
        CP 03DH
        JR NC,L97C
        CP 032H
        JR NC,L97E
        CP 01CH
        JR C,L981
L97C:
```

```
        LD  A,02DH
L97E:
        SUB 016H
        LD E,A          ; suche zugehoerigen Fehlertext
L981:
        CALL L109D
        INC HL
        DEC E
        JR NZ,L981
        PUSH HL         ; Fehlertext-Adresse in HL
        LD HL,(ERL)
        EX (SP),HL      ; speichere Fehler Zeile
ERRFIN: /98D
        CALL OFF1BH
        PUSH HL
        CALL TOTEXT     ; Text- Modus einschalten /3768
        POP HL
        LD A,(HL)
        CP 03FH
        JR NZ,L9A0
        POP HL
        LD  HL,05D6H
        JR L97C
L9A0:
        LD  A,07H
        RST 18H         ; BEEP
        CALL STROUT     ; Fehlertext ausgeben /697D
        POP HL
        LD A,H
        AND L
        INC A
        CALL NZ,INPRT   ; Zeilennummer, falls im RUN Modus / 5B3C
READY:  /9AD  CDA,C4H
        DEFB 03EH               ;*      LD  A,0C1H
STPRDY: /9AE                    ;*      3EC1        >A
        POP BC [C5]
!!  9AF: CALL 3768 <- CALL TOTEXT   ; Text Modus
        CALL FINLPT     ; Ausgabe Printer Puffer/ 643D
        CALL PRGFIN     ; 750C
        CALL OFE8EH
        CALL CRDONZ     ; evtl. neue Zeile/6463
        LD  HL,REDDY
REPINI: /9C1
        CALL STROUT     ; Ausgabe OK / 697D
MAIN:   /9C4
        CALL OFE94H
        LD  HL,0FFFFH   ; FFFF als Zeilennummer zeigt
        LD (CURLIN),HL  ; direkten Modus an / F548
        LD  HL,0F526H
        LD (SAVTXT),HL  ; F7D8
        LD  A,(AUTFLG)  ; F7D6
        OR A
        JR Z,L9EF       ; ZERO= kein AUTO Modus
        LD HL,(AUTLIN)  ; F7D7
        PUSH HL
        CALL LINPRT     ; Ausgabe Zeilennummer/ 5B44
        POP DE
        PUSH DE
        CALL FNDLIN     ; Test, ob Zeile vorhanden/ B27
        LD  A,'*'
        JR C,L9EB       ; C, falls vorhanden
```



```
        LD  A,' '
L9EB:
        RST 18H
        LD (AUTFLG),A   ; F7D6
L9EF:
        CALL PINLIN     ; Zeile einlesen / 6D13
        JR NC,INILIN    ; NC = kein CTRL-C / 9F8
        XOR A
        LD (AUTFLG),A   ; AUTO Modus loeschen/F7D6
        JP MAIN         ; 9C4
INILIN: /9F8    ;*** Interpretation der Eingabe-Zeile
        RST 10H         ; erstes Zeichen holen
        INC A
        DEC A
        JR Z,MAIN       ; JP, falls leere Zeile /9C4
        PUSH AF
        CALL LINGET     ; FAD
        JR NC,LAOC      ; Zeilennummer gefunden
        CALL ISFLIO     ; 68C2
        JP Z,SNERR      ; 8ED
LAOC:
        CALL LD55       ; ueberspringen der Fuellzeichen
        LD  A,(AUTFLG)  ; F7D6
        OR A
        JR Z,LA1D       ; JP, wenn kein AUTO
        CP '*'
        JR NZ,LA1D      ; Zeile nicht loeschen
        CP (HL)
        CALL Z,INXHRT   ; 5615
LA1D:
        LD A,D
        OR E
        JP Z,EDENT      ; wenn Zeilennummer=0 / A28
        LD A,(HL)
        CP ' '
        CALL Z,INXHRT   ; 5615
EDENT:  /A28
        PUSH DE         ; Zeilennummer
        CALL CRUNCH     ; Eingabe --> Token/ B44
        POP DE
        POP AF
        LD (SAVTXT),HL  ; speichere Zeilenpointer/ F7D8
        CALL OFF06H
        JR C,LA3D       ; wenn Ziffern am Anfang der
        XOR A           ; Eingabezeile
        LD (AUTFLG),A   ; F7D6
        JP DIROG        ; ausfuehren im direkten Modus/74D9
LA3D:
        PUSH DE         ; Zeilennummer
        PUSH BC         ; Laenge der Anweisung
        RST 10H         ; erstes Token
        OR A
        PUSH AF
        LD (DOT),DE     ; F7E1
        LD HL,(AUTOP)   ; F7D9
        ADD HL,DE
        JR C,LA59
        PUSH DE
        LD  DE,0FFFAH   ; groesste Zeilennummer
        RST 20H         ; Vergleich gegen Aktuelle
```

```
        POP DE
        JR NC,LA59      ; JP, falls groesser
        LD (AUTLIN),HL  ; F7D7
        JR LA5D
LA59:
        XOR A
        LD (AUTFLG),A   ; F7D6
LA5D:
        CALL FNDLIN     ; 827
        JR C,LA75
        POP AF
        PUSH AF
        JP NZ,LA72
        LD  A,(AUTFLG)  ; F7D6
        OR A
        JP Z,USERR      ; 105C
        PUSH BC
        JP FINI         ; A86
LA72:
        OR A
        JR LA86
LA75:
        POP AF
        PUSH AF
        JP NZ,LA85
        LD  A,(AUTFLG)  ; F7D6
        OR A
        JP Z,LA85
        PUSH BC
        JP FINI         ; A86
LA85:
        SCF             ; Zeile loeschen
LA86:
        PUSH BC         ; Zeilenpointer
        PUSH AF         ; 1. Token
        PUSH HL         ; Next Pointer
        CALL DEPTR      ; 1E0D
        POP HL
        POP AF
        POP BC
        PUSH BC
        CALL C,DEL      ; Zeilenbereich bis zum Ende/ AC8F
        POP DE          ; aufruecken lassen
        POP AF
        PUSH DE
        JR Z,FINI       ; wenn leere Zeile /A86
        POP DE
        LD  HL,00H
        LD (ONELIN),HL  ; F7E5
        LD HL,(VARTAB)  ; F7EE
        EX (SP),HL
        POP BC
        PUSH HL
        ADD HL,BC
        PUSH HL
        CALL BLTU       ; verschieben der folgenden Zeilen/ 6520
        POP HL
        LD (VARTAB),HL  ; F7EE
        EX DE,HL
        LD (HL),H
        POP BC
```



```
        POP DE
        PUSH HL
        INC HL
        INC HL
        LD (HL),E       ; Zeilennummer speichern
        INC HL
        LD (HL),D
        INC HL
        LD  DE,KBUF     ; F54F
        DEC BC
        DEC BC
        DEC BC
        DEC BC
LACO:
        LD  A,(DE)      ; speichere Programmzeile
        LD (HL),A
        INC HL
        INC DE
        DEC BC
        LD A,C
        OR B
        JR NZ,LACO
FINI: /AC9
        CALL OFF30H
        POP DE
        CALL CHEAD      ; neue Zeilen Pointer erzeugen/AE9
        LD HL,(PTRFIL)  ; F997
        LD (TEMP2),HL   ; F7E8
        CALL RUNC       ; 656A
        CALL OFF36H     ; FF36
        LD HL,(TEMP2)   ; F7E8
        LD (PTRFIL),HL  ; F997
        JP MAIN


LINKER:/ AE5            ; neue Programm Pointer ab Programm-
        LD HL,(TXTTAB)  ; anfang/F54A
        EX DE,HL
CHEAD:/ AE9             ; neue Zeilen Pointer fuer
        LD H,D          ; alle Zeilen nach der aktuellen
        LD L,E          ; DE ADR der aktuellen Zeile
        LD A,(HL)
        INC HL
        OR (HL)
        RET Z           ; Ende Programmtexttabelle
        INC HL
        INC HL
LAF1:
        INC HL
        LD A,(HL)
LAF3:
        OR A
        JR Z,LB04       ; Ende einer Anweisung
        CP 020H
        JR NC,LAF1      ; AF1
        CP 0BH
        JR C,LAF1       ; AF1
        CALL CHRGT2     ; EAE
        RST 10H
        JR LAF3         ; AF3
```

```
LB04:
        INC HL
        EX DE,HL
        LD (HL),E
        INC HL
        LD (HL),D
        JR CHEAD        ;AE9


LB0B:                   ; holt Zeilennummern-Bereich,
        LD  DE,00H      ; falls nicht angegeben, 0-65529
        PUSH DE
        JR Z,LB1A
        POP DE
        CALL LINSPC     ;FA3
        PUSH DE
        JR Z,LB23
        RST 8H
        DEFB 0F4H       ; - Token
LB1A:
        LD DE,0FFFAH
        CALL NZ,LINSPC  ;FA3
        JP NZ,SNERR     ;8ED
LB23:
        EX DE,HL
        POP DE
LB25:
        EX (SP),HL
        PUSH HL

FNDLIN:/B27             ; sucht gleiche Zeilennummer
        LD HL,(TXTTAB)  ; wie in DE/F54A
LB2A:                   ; C Z, falls gefunden
        LD B,H          ; NC, nicht gefunden
        LD C,L
        LD A,(HL)
        INC HL
        OR (HL)
        DEC HL
        RET Z
        INC HL
        INC HL
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        RST 20H
        LD H,B
        LD L,C
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        CCF
        RET Z
        CCF
        RET NC
        JR LB2A
```



```
CRUNCH:/B44             ; umschluesseln der Eingabe-Zeile
        XOR A           ; in Token
        LD (0F795H),A
        LD (0F794H),A
        CALL 0FF24H
        LD  BC,013BH
        LD  DE,KBUF     ;F54B
LB54:
        LD A,(HL)       ; Zeichen aus Eingabe
        OR A
        JR NZ,LB6B
LB58:
        LD  HL,0140H
        LD A,L
        SUB C
        LD C,A
        LD A,H
        SBC A,B
        LD B,A
        LD  HL,KBUF-1   ;F54E
        XOR A
        LD (DE),A       ; 3*0 hinter haengen
        INC DE
        LD (DE),A
        INC DE
        LD (DE),A
        RET
LB6B:
        CP '"'
        JP Z,LBA0       ; String Test
        CP ' '
        JR Z,LB7B
        LD  A,(0F794H)
        OR A
        LD A,(HL)
        JR Z,LBA8
LB7B:                   ; Zeichen speichern
        INC HL
        PUSH AF
        CALL LD21
        POP AF
        SUB ':'
        JR Z,LB8B
        CP 04AH
        JR NZ,LB91
        LD  A,01H
LB8B:
        LD (0F794H),A
        LD (0F795H),A
LB91:
        SUB 055H
        JR NZ,LB54
        PUSH AF
LB96:
        LD A,(HL)
        OR A
        EX (SP),HL
        LD A,H
        POP HL
```

```
        JR Z,LB58
        CP (HL)
        JR Z,LB7B
LBAO:
        PUSH AF
        LD A,(HL)
LBA2:
        INC HL
        CALL LD21
        JR LB96
LBA8:
        INC HL
        OR A
        JP M,LB54
        DEC HL
        CP '?'
        LD  A,091H      ; Print Token
        PUSH DE
        PUSH BC
        JP Z,LC15
        LD  DE,0587H    ; Token Tabelle fuer Operatoren
        CALL MAKUPL     ; 1708
        CALL ISLET2     ; 679F
        JP C,LC5E       ; falls Zuweisung
        PUSH HL
        CALL OFF21H
        LD  HL,ALPTAB   ; 295
LBCA:
        SUB 'A'
        ADD A,A
        LD C,A
        LD B,00H
        ADD HL,BC
        LD E,(HL)
        INC HL
        LD D,(HL)       ; ADR der Token-Tabelle fuer
        POP HL          ; diesen Anfangsbuchstaben
        INC HL
LBD6:
        PUSH HL
LBD7:
        CALL MAKUPL     ; 1708
        LD C,A
        LD  A,(DE)
        AND 07FH
        JP Z,LD2C       ; falls Buchstabe zu Ende
        INC HL
        CP C
        JR NZ,LC09      ; ungleich der Eingabe
        LD  A,(DE)
        INC DE
        OR A
        JP P,LBD7       ; falls Schluesselwort nicht zu Ende
        POP AF          ; Schluesselwort gefunden
        LD  A,(DE)      ; Token holen
        CALL OFE91H
        OR A
        JP M,LC14       ; falls Token >7FH
        POP BC
        POP DE
        OR 080H
```



```
        PUSH AF
        LD  A,0FFH      ; Token OFFH einsetzen
RENCRN: / BF8
        CALL LD21
        XOR A
        LD (0F795H),A
        POP AF
        CALL LD21       ; Token setzen
        JP LB54

LC09:
        POP HL
LC0A:
        LD  A,(DE)
        INC DE
        OR A
        JP P,LC0A       ; noch nicht zu Ende
        INC DE
        JP LBD6         ; weiter mit naechstem Schl.wort


LC14:           ; TOKEN >7FH
        DEC HL
LC15:
        PUSH AF
        CALL OFEACH
        LD  DE,LINTOKENS        ; TOKENS mit Zeilennummern / C27
        LD C,A
LC1D:
        LD  A,(DE)
        OR A
        JR Z,LC36       ; Tabellen Ende
        INC DE
        CP C
        JR NZ,LC1D
        JR LC38         ; Eins aus der Tabelle

LINTOKENS: / C27        ; 0C27H
        DEFB 08CH,0A9H,0AAH,0A8H,0A7H,0E1H
        DEFB 0A1H,08AH,093H,09EH,089H,08EH,0DAH,08DH,0

LC36:                   ; Loeschen Zeilennummern- Flag
        XOR A
        DEFB 0C2H       ;*      JP NZ,L13E
LC38:
        LD A,1          ;*      Setzen des Flags
LC3A:
        LD (0F795H),A   ; Zeilennummern -Flag
        POP AF
LC3E:
        POP BC
        POP DE
        CP 0A1H         ; ELSE TOKEN
        PUSH AF
        CALL Z,0D1FH
        POP AF
        CP 0E6H
        JP NZ,LCF5
        PUSH AF
        CALL LD1F
        LD  A,08FH      ; REM TOKEN
```

```
        CALL LD21
        POP AF
        PUSH HL
        LD  HL,00H
        EX (SP),HL
        JP LBA2
LC5E:
        LD A,(HL)
        CP '.'
        JR Z,LC6D
        CP ':'
        JP NC,LCE3
        CP GETYPR       ; 30
        JP C,LCE3
LC6D:
        LD  A,(0F795H)
        OR A
        LD A,(HL)
        POP BC
        POP DE
        JP M,LB7B
        JR Z,LC98
        CP '.'
        JP Z,LB7B
        LD  A,0EH       ; Zeilennummern-Token
        CALL LD21
        PUSH DE
        CALL LINGET     ; Zeilennummer holen / FAD
        CALL LD55
LC8A:
        EX (SP),HL
        EX DE,HL
LC8C:
        LD A,L
        CALL LD21
        LD A,H
LC91:
        POP HL
        CALL LD21
        JP LB54
LC98:
        PUSH DE
        PUSH BC
        LD A,(HL)
        CALL FINDBL     ; 59CB
        CALL LD55
        POP BC
        POP DE
        PUSH HL
        LD  A,(VALTYP)  ; F793
        CP 02H
        JR NZ,LCC0
        LD HL,(FACLOW)  ; F925
        LD A,H
        OR A
        LD  A,02H
        JR NZ,LCC0
        LD A,L
        LD H,L
        LD  L,0FH
        CP 0AH
```



```
        JR NC,LC8C
        ADD A,011H
        JR LC91
LCC0:
        PUSH AF
        RRCA
        ADD A,01BH
        CALL LD21
        LD  HL,FACCU
        LD  A,(VALTYP)  ; F793
        CP 02H
        JR NZ,LCD4
        LD  HL,FACLOW   ; F925
LCD4:
        POP AF
LCD5:
        PUSH AF
        LD A,(HL)
        CALL LD21
        POP AF
        INC HL
        DEC A
        JR NZ,LCD5
        POP HL
        JP LB54
LCE3:
        LD  DE,0586H    ; Operator- Token-1
LCE6:
        INC DE
        LD  A,(DE)
        AND 07FH
        JP Z,LD3B
        INC DE
        CP (HL)
        LD  A,(DE)
        JR NZ,LCE6
        JP LD4A
LCF5:
        CP '&'
        JP NZ,LB7B
        PUSH HL
        RST 10H
        POP HL
        CALL MAKUPS     ; 170C
        CP 'H'
        JR Z,LD11
        CP 'O'
        JR Z,LDOD
        LD  A,'&'
        JP LB7B
LDOD:
        LD  A,0BH       ; Oktal-Konstante
        JR LD13
LD11:
        LD  A,0CH       ; Hex-Konstante
LD13:
        CALL LD21
        PUSH DE
        PUSH BC
        CALL OCTCNS     ; 171A
        POP BC
```

```
        JP LC8A
LD1F:
        LD  A,':'
LD21:
        LD (DE),A
        INC DE
        DEC BC
        LD A,C
        OR B
        RET NZ
LBOERR: LD  E,019H      ; LINE BUFFER OVERFLOW
        JP ERROR

LD2C:                   ; wenn das Schluesselwort nicht in
        CALL OFFOCH     ; der Tabelle enthalten ist
NOTRFN: POP HL
        DEC HL
        DEC A
        LD (0F795H),A
        CALL MAKUPL
        JP LC3E
LD3B:
        LD A,(HL)
        CP ' '
        JR NC,LD4A
        CP 09H          ; TAB
        JR Z,LD4A
        CP 0AH          ; LF
        JR Z,LD4A
        LD  A,' '
LD4A:
        PUSH AF
        LD  A,(0F795H)
        INC A
        JR Z,LD52
        DEC A
LD52:
        JP LC3A
LD55:
        DEC HL
        LD A,(HL)
        CP ' '
        JR Z,LD55
        CP 09H
        JR Z,LD55
        CP 0AH
        JR Z,LD55
        INC HL
        RET

        ;************************************

LD65:                   ; FOR STATEMENT
        LD  A,064H
        LD (SUBFLG),A
        CALL L10C0
        POP BC
        PUSH HL
        CALL L109B
```



```
        LD (ENDFOR),HL
        LD  HL,02H
        ADD HL,SP
LD79:
        CALL L8AE
        JR NZ,LD95
        ADD HL,BC
        PUSH DE
        DEC HL
        LD D,(HL)
        DEC HL
        LD E,(HL)
        INC HL
        INC HL
        PUSH HL
        LD HL,(ENDFOR)
        RST 20H
        POP HL
        POP DE
        JR NZ,LD79
        POP DE
        LD SP,HL
        LD (SAVSTK),HL
        DEFB 0EH                ;*      LD  C,0D1H
LD95:
        POP DE                  ;*
        EX DE,HL
        LD  C,0CH
        CALL GETSTK
        PUSH HL
        LD HL,(ENDFOR)
        EX (SP),HL
        PUSH HL
        LD HL,(CURLIN)
        EX (SP),HL
        RST 8H
        DEFB 0D9H       ; TO- Token
        RST 30H
        JP Z,TMERR
        PUSH AF
        CALL FRMEVL
        POP AF
        PUSH HL
        JR NC,LDCC
        JP P,LE01
        CALL FRCINT
        EX (SP),HL
        LD  DE,01H
        LD A,(HL)
        CP 0DCH         ; STEP- Token
        CALL Z,GETINT
        PUSH DE
        PUSH HL
        EX DE,HL
        CALL ISIGN
        JR LE25
LDCC:
        CALL FRCDBL
        POP DE
        LD  HL,0FFF8H
        ADD HL,SP
```

```
        LD SP,HL
        PUSH DE
        CALL VMOVMF
        POP HL
        LD A,(HL)
        CP ODCH          ; STEP- Token
        LD  DE,ONE
        LD  A,01H
        JR NZ,LDF1
        RST 10H
        CALL FRMEVL
        PUSH HL
        CALL FRCDBL
        RST 28H
        LD  DE,FACCU
        POP HL
LDF1:
        LD B,H
        LD C,L
        LD  HL,OFFF8H
        ADD HL,SP
        LD SP,HL
        PUSH AF
        PUSH BC
        CALL VMOVE
        POP HL
        POP AF
        JR LE2C
LE01:
        CALL FRCSNG
        CALL MOVRF
        POP HL
        PUSH BC
        PUSH DE
        LD  BC,01041H
        LD  DE,00H
        CALL LFE88
        LD A,(HL)
        CP ODCH
        LD  A,01H
        JR NZ,LE26
        CALL FRMCHK
        PUSH HL
        CALL FRCSNG
        CALL MOVRF
        RST 28H
LE25:
        POP HL
LE26:
        PUSH DE
        PUSH BC
        PUSH BC
        PUSH BC
        PUSH BC
        PUSH BC
LE2C:
        OR A
        JR NZ,LE31
        LD  A,02H
LE31:
        LD C,A
```



```
        RST 30H
        LD B,A
        PUSH BC
        PUSH HL
        LD HL,(TEMP)
        EX (SP),HL
NXTCON: E3A
        LD B,082H        ; speichern FOR Token fuer NEXT
        PUSH BC
        INC SP
NEWSTT: E3E              ; Eingang nach Ausfuehrung einer Prg-Zeile
        CALL OFF54H
        CALL ISCNTC      ; Test CTRL C
        CALL RCVX        ; Empfangsdaten vom Modem
        CALL NZ,SCMTRP
        LD  A,(ONGSBF)   ; Test ON .. GOSUB
        OR A
        CALL NZ,GOTRP    ; anspringen falls gesetzt
NWSTRT: E51
        LD (SAVTXT),HL
        LD (SAVSTK),SP
        LD A,(HL)
        CP 03AH
        JR Z,GONE        ; mehrfach Anweisung
        OR A
        JP NZ,SNERR      ; falls nicht zu Ende SYNTAX ERROR
        INC HL
LE62:
        LD A,(HL)
        INC HL
        OR (HL)
        JP Z,L8CF        ; Programmende
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        EX DE,HL
        LD (CURLIN),HL   ; neue Zeilennummer
        LD  A,(TRCFLG)
        OR A
        JR Z,LE81        ; falls nicht TRACE ON
        PUSH DE
        LD  A,'['
        RST 18H
        CALL LINPRT
        LD  A,']'
        RST 18H
        POP DE
LE81:
        EX DE,HL
GONE:   E82
        RST 10H
        LD  DE,NEWSTT
        PUSH DE
        RET Z
LE88:
        CALL OFF57H
        CP OEEH          ; SPRITE setzen
        JP Z,SPRITE
        CP OEFH          ; TIME setzten
        JP Z,STTIME
```

```
        SUB 081H
        JP C,L10C0      ; falls Zuweisung
        CP 058H
        JP NC,L1A1A
        RLCA            ; Token <=0D8H
        LD C,A
        LD B,00H
        EX DE,HL
        LD  HL,0185H    ; Token-Ausfuehrungs-Tabelle
        ADD HL,BC
        LD C,(HL)
        INC HL
        LD B,(HL)
        PUSH BC
        EX DE,HL
CHRGTR: /EAD
        INC HL
CHRGT2: /EAE
        LD A,(HL)
        CP ':'
        RET NC
CHRCON: /EB2
        CP ' '
        JR Z,CHRGTR
        JR NC,LF24
        OR A
        RET Z
        CP 0BH
        JR C,LF1F
        CP 01EH
        JR NZ,LEC7
        LD  A,(CONSAV)
        OR A
        RET
LEC7:
        CP 010H
        JR Z,LEFF
        PUSH AF
        INC HL
        LD (CONSAV),A
        SUB 01CH
        JR NC,LF04
        SUB 0F5H
        JR NC,LEDE
        CP 0FEH
        JR NZ,LEF2
        LD A,(HL)
        INC HL
LEDE:
        LD (0F796H),HL
        LD H,00H
LEE3:
        LD L,A
        LD (0F79AH),HL
        LD  A,02H
        LD (0F799H),A
        LD  HL,LF2A
        POP AF
        OR A
        RET
LEF2:
```



```
        LD A,(HL)
        INC HL
        INC HL
        LD (0F796H),HL
        DEC HL
        LD H,(HL)
        JR LEE3
LEFC:
        CALL LF2C
LEFF:
        LD HL,(0F796H)
        JR CHRGT2
LF04:
        INC A
        RLCA
        LD (0F799H),A
        PUSH DE
        PUSH BC
        LD  DE,0F79AH
        EX DE,HL
        LD B,A
        CALL MOVE1
        EX DE,HL
        POP BC
        POP DE
        LD (0F796H),HL
        POP AF
        LD  HL,0F2AH
        OR A
        RET
LF1F:
        CP 09H
        JP NC,CHRGTR
LF24:
        CP '0'
        CCF
        INC A
        DEC A
        RET
LF2A:
        LD  E,010H
LF2C:
        LD  A,(CONSAV)
        CP 0FH
        JR NC,LF46
        CP 0DH
        JR C,LF46
        LD HL,(0F79AH)
        JR NZ,LF43
        INC HL
        INC HL
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        EX DE,HL
LF43:
        JP INEG2
LF46:
        LD  A,(0F799H)
        LD (VALTYP),A
```

```
        CP 02H
        JR NZ,LF56
        LD HL,(0F79AH)
        LD (FACLOW),HL
LF56:
        LD  HL,0F79AH
        JP VMOVFM
LF5C:                   ;*** DEFSTR
        LD  E,03H
        DEFB 1                  ;*      LD  BC,021EH
LF5F:                   ;*** DEFINT
        LD E,2                  ;*
        DEFB 1                  ;*      LD  BC,041EH
LF62:                   ;*** DEFSNG
        LD E,4                  ;*
        DEFB 1                  ;*      LD  BC,081EH
LF65:                   ;*** DEFDBL
        LD E,8
LF67:
        CALL ISLET
        LD  BC,SNERR
        PUSH BC
        RET C
        SUB 'A'
        LD C,A
        LD B,A
        RST 10H
        CP 0F4H         ; - Token
        JR NZ,LF81
        RST 10H
        CALL ISLET
        RET C
        SUB 'A'
        LD B,A
        RST 10H
LF81:
        LD A,B
        SUB C
        RET C
        INC A
        EX (SP),HL
        LD  HL,DEFTBL
        LD B,00H
        ADD HL,BC
LF8C:
        LD (HL),E
        INC HL
        DEC A
        JR NZ,LF8C
        POP HL
        LD A,(HL)
        CP ','
        RET NZ
        RST 10H
        JR LF67

INTIDX: F99
        RST 10H
INTID2: F9A
        CALL GETIN2
```



```
        RET P
FCERR:  F9E
        LD  E,05H       ; ILLEGAL FUNCTION CALL
        JP ERROR
LINSPC: FA3             ; als Zeilennummer gilt auch
        LD A,(HL)
        CP '.'
        LD DE,(DOT)     ; letzte aktuelle Zeilennummer
        JP Z,CHRGTR
LINGET: FA0             ;******* Zeilennummer aus Token String
        DEC HL
LFAE:
        RST 10H
        CP 0EH
        JR Z,LFB5
        CP 0DH
LFB5:
        LD DE,(0F79AH)
        JP Z,CHRGTR
        XOR A
        LD (CONSAV),A
        LD  DE,00H
        DEC HL
LFC4:
        RST 10H
        RET NC
        PUSH HL
        PUSH AF
        LD  HL,01998H
        RST 20H
        JR C,LFDF
        LD H,D
        LD L,E
        ADD HL,DE
        ADD HL,HL
        ADD HL,DE
        ADD HL,HL
        POP AF
        SUB '0'
        LD E,A
        LD D,00H
        ADD HL,DE
        EX DE,HL
        POP HL
        JR LFC4
LFDF:
        POP AF
        POP HL
        RET

LFE2:                   ;************ RUN
        JP Z,RUNC
        CP 0EH
        JR Z,LFEE
        CP 0DH
        JP NZ,LRUN
LFEE:
        CALL CLEARC
        LD  BC,NEWSTT
        JR L1027
```

```
GOSUB:/ FF6             ;*************
        LD  C,03H
        CALL GETSTK
        CALL LINGET
        POP BC
        PUSH HL
        PUSH HL
        LD HL,(CURLIN)
        EX (SP),HL
        LD  BC,00H
        PUSH BC
        LD  BC,NEWSTT
        LD  A,08DH
        PUSH AF
        INC SP
        PUSH BC
        JR GOTO2
GOSUB2:/ 1013
        PUSH HL
        PUSH HL
        LD HL,(CURLIN)
        EX (SP),HL
        PUSH BC
        LD  A,08DH
        PUSH AF
        INC SP
        EX DE,HL
        DEC HL
        LD (SAVTXT),HL
        INC HL
        JP LE62
L1027:
        PUSH BC
```



```
L1028:                  ;************ GOTO
        CALL LINGET
GOTO2:/ 1028
        LD  A,(CONSAV)
        CP ODH
        EX DE,HL
        RET Z
        CP OEH
        JP NZ,SNERR
        EX DE,HL
        PUSH HL
        LD HL,(0F796H)
        EX (SP),HL
        CALL L109D
        INC HL
        PUSH HL
        LD HL,(CURLIN)
        RST 20H
        POP HL
        CALL C,0B2AH
        CALL NC,FNDLIN
        JR NC,USERR
        DEC BC
        LD  A,0DH
        LD (PTRFLG),A
        POP HL
        CALL L1E04
        LD H,B
        LD L,C
        RET
USERR:/ 105C
        LD  E,08H
        JP ERROR

L1061:                          ;********* RETURN
        CALL LFEEE
        LD (TEMP),HL
        LD D,0FFH
        CALL FNDFOR
        CP 08DH
        JR Z,L1071
        DEC HL
L1071:
        LD SP,HL
        LD (SAVSTK),HL
        LD  E,03H
        JP NZ,L907
        POP HL
        LD A,H
        OR L
        JR Z,L1085
        LD A,(HL)
        AND 01H
        CALL NZ,RSTTRP
L1085:
        POP BC
        LD  HL,NEWSTT
        EX (SP),HL
        EX DE,HL
        LD HL,(TEMP)
```

```
        DEC HL
        RST 10H
        JP NZ,L1028
        LD H,B
        LD L,C
        LD (CURLIN),HL
        EX DE,HL
        DEFB 03EH               ;*      LD  A,0E1H
L109A:
        POP HL                  ;*
L109B:                  ;***************** DATA
        DEFB 01H,03AH           ;*      LD  BC,0E3AH
L109D:
        LD C,0                  ;*
        LD B,00H
L10A1:
        LD A,C
        LD C,B
        LD B,A
L10A4:
        DEC HL
L10A5:
        RST 10H
        OR A
        RET Z
        CP B
        RET Z
        INC HL
        CP '"'
        JR Z,L10A1
        INC A
        JR Z,L10A5
        SUB 08CH
        JR NZ,L10A4
        CP B
        ADC A,D
        LD D,A
        JR L10A4
LETCON:                 ;1088
        POP AF
        ADD A,03H
        JR L10D2
L10C0:                          ;************ LET
        CALL PTRGET
        RST 8H
        DEFB 0F1H       ; = Token
        LD (TEMP),DE
        PUSH DE
        LD  A,(VALTYP)
        PUSH AF
        CALL FRMEVL     ; Ausdruck berechnen
        POP AF
L10D2:
        EX (SP),HL
L10D3:
        LD B,A
        LD  A,(VALTYP)
        CP B
        LD A,B
        JR Z,L10E1
        CALL DOCNVF
```



```
L10DE:
        LD  A,(VALTYP)
L10E1:
        LD  DE,FACCU    ; SINGLE/DOUBLE
        CP 02H
        JR NZ,L10EB
        LD  DE,FACLOW   ; INTEGER
L10EB:
        PUSH HL
        CP 03H
        JR NZ,L111E
        LD HL,(FACLOW)  ; STRING Ergebnis
        PUSH HL
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        LD  HL,KBUF-1
        RST 20H
        JR C,L1112
        LD HL,(STREND)
        RST 20H
        POP DE
        JR NC,L111A
        LD  HL,0F7C3H
        RST 20H
        JR C,L1111
        LD  HL,0F7A5H
        RST 20H
        JR C,L111A
L1111:
        DEFB 03EH               ;*      LD  A,0D1H
L1112:
        POP DE                  ;*
        CALL FRETMS
        EX DE,HL
        CALL STRCPY
L111A:
        CALL FRETMS
        EX (SP),HL
L111E:
        CALL VMOVE
        POP DE
        POP HL
        RET
L1124:                          ;************** ON ...
        CP 0A6H         ; ERROR Token
        JR NZ,L114D
        RST 10H         ; ON ERROR
        RST 8H
        DEFB 089H       ; GOTO
        CALL LINGET
        LD A,D
        OR E
        JR Z,L113B
        CALL LB25
        LD D,B
        LD E,C
        POP HL
        JP NC,USERR
L113B:
```

```
        LD (ONELIN),DE
        RET C
        LD  A,(ONEFLG)
        OR A
        LD A,E
        RET Z
        LD  A,(ERRFLG)
        LD E,A
        JP L92B
L114D:
        CALL ONGOTP
        JR C,L1183
        PUSH BC
        RST 10H
        RST 8H
        DEFB 08DH       ; GOSUB
        XOR A
L1157:
        POP BC
        PUSH BC
        CP C
        JP NC,SNERR
        PUSH AF
        CALL LINGET
        LD A,D
        OR E
        JR Z,L116E
        CALL LB25
        LD D,B
        LD E,C
        POP HL
        JP NC,USERR
L116E:
        POP AF
        POP BC
        PUSH AF
        ADD A,B
        PUSH BC
        CALL SETGSB
        DEC HL
        RST 10H
        POP BC
        POP DE
        RET Z
        PUSH BC
        PUSH DE
        RST 8H
        DEFB ','
        POP AF
        INC A
        JR L1157
L1183:
        CALL GETBYT
NTONG1: 1186
        LD A,(HL)
        LD B,A
        CP 08DH         ; GOSUB
        JR Z,L118F
        RST 8H
        DEFB 089H       ; GOTO
        DEC HL
```



```
L118F:
        LD C,E
L1190:
        DEC C
        LD A,B
        JP Z,LE88
        CALL LFAE
        CP 02CH
        RET NZ
        JR L1190
L119D:                  ; ************** RESUME
        LD  A,(ONEFLG)
        OR A
        JR NZ,L11AC
        LD (ONELIN),A
        LD (0F7E6H),A
        JP L8FC
L11AC:
        INC A
        LD (ERRFLG),A
        LD A,(HL)
        CP 083H         ; NEXT
        JR Z,L11C5
        CALL LINGET
        RET NZ
        LD A,D
        OR E
        JR Z,L11C9
        CALL GOTO2
        XOR A
        LD (ONEFLG),A
        RET
L11C5:
        RST 10H
        RET NZ
        JR L11CE
L11C9:
        XOR A
        LD (ONEFLG),A
        INC A
L11CE:
        LD HL,(0F7E3H)
        EX DE,HL
        LD HL,(ERL)
        LD (CURLIN),HL
        EX DE,HL
        RET NZ
        LD A,(HL)
        OR A
        JR NZ,L11E2
        INC HL
        INC HL
        INC HL
        INC HL
L11E2:
        INC HL
        XOR A
        LD (ONEFLG),A
        JP L109B

L11EA:                  ;************ ERROR
```

```
        CALL GETBYT
        RET NZ
        OR A
        JP Z,FCERR
        JP ERROR

L11F5:                  ;************ AUTO
        LD  DE,0AH
        PUSH DE
        JR Z,L1211
        CALL LINSPC
        EX DE,HL
        EX (SP),HL
        JR Z,L1212
        EX DE,HL
        RST 8H
        DEFB ','
        LD DE,(AUTOP)
        JR Z,L1211
        CALL LINGET
        JP NZ,SNERR
L1211:
        EX DE,HL
L1212:
        LD A,H
        OR L
        JP Z,FCERR
        LD (AUTOP),HL
        LD (AUTFLG),A
        POP HL
        LD (AUTLIN),HL
        POP BC
        JP MAIN


L1225:                  ;************ IF
        CALL FRMEVL
        LD A,(HL)
        CP 02CH
        CALL Z,CHRGTR
        CP 089H         ; GOTO
        JR Z,L1235
        RST 8H
        DEFB 0DAH       ; THEN
        DEC HL
L1235:
        PUSH HL
        CALL VSIGN
        POP HL
        JR Z,L124C
L123C:
        RST 10H
        RET Z
        CP 0EH
        JP Z,L1028
        CP 0DH
        JP NZ,LE88
        LD HL,(0F79AH)
        RET
L124C:
        LD D,01H
```



```
L124E:
        CALL L109B
        OR A
        RET Z
        RST 10H
        CP 0A1H         ; ELSE
        JR NZ,L124E
        DEC D
        JR NZ,L124E
        JR L123C


L125D:                  ;*************** LPRINT
        LD  A,01H
        LD (0DEVLINK),A
        JP L126A
L1265:                  ;*************** PRINT
        LD  C,02H
        CALL FILGET
L126A:
        DEC HL
        RST 10H
        CALL Z,CRDO
L126F:
        JP Z,FINPRT
        CP 0E4H         ; USING
        JP Z,PRINUS
        CP 0DBH         ; TAB(
        JP Z,L132C
        CP 0DFH         ; SPC(
        JP Z,L132C
        PUSH HL
        CP ','
        JR Z,L12DE
        CP ';'
        JP Z,L1360
        POP BC
        CALL FRMEVL
        PUSH HL
        RST 30H
        JR Z,L12D7
        CALL FOUT
        CALL STRLIT
        LD (HL),' '
        LD HL,(FACLOW)
        INC (HL)
        CALL LFE7C
        CALL ISFLIO
        JR NZ,L12D3
        LD  A,(SCREEN)
        AND A
        JR NZ,L12D3
        LD HL,(FACLOW)
        LD  A,(0DEVLINK)
        OR A
        JR Z,L12BE
        LD  A,(LPOS)
        ADD A,(HL)
        CP 0FFH
        JR L12CB
```

```
L12BE:
        LD  A,(LINLEN)
        LD B,A
        INC A
        JR Z,L12D3
        LD  A,(TTYPOS)
        ADD A,(HL)
        DEC A
LINPT3: 12CA
        CP B
L12CB:
        JR C,L12D3
        CALL Z,CRFIN
        CALL NZ,CRDO
L12D3:
        CALL STRPRT
        OR A
L12D7:
        CALL Z,STRPRT
        POP HL
        JP L126A
L12DE:
        CALL OFFOOH
        LD  BC,08H
        LD HL,(PTRFIL)
        ADD HL,BC
        CALL ISFLIO
        LD A,(HL)
        JR NZ,L1325
        LD  A,(ODEVLINK)
        OR A
        JR Z,L12FC
        LD  A,(LPOS)
        CP OEEH
        JP L131F
L12FC:
        LD  A,(SCREEN)
        AND A
        JR Z,L1313
        LD  A,(TTYPOS)
        CALL L130A
        JR L1325
L130A:
        LD B,OFFH
L130C:
        INC B
        SUB 06H
        JR NC,L130C
        LD A,B
        RET
L1313:
        LD  A,(0F544H)
        LD B,A
        LD  A,(TTYPOS)
        CP OFFH
        JR Z,L1325
LINPT4: 131E
        CP B
L131F:
        CALL NC,CRDO
        JP NC,L1360
```



```
L1325:
        SUB OEH
        JR NC,L1325
        CPL
        JR L1359
L132C:                  ; SPC() TAB()   Verarbeitung
        PUSH AF
        CALL GTBYTC
        RST 8H
        DEFB ')'
        DEC HL
        POP AF
        SUB ODFH
        PUSH HL
        JR Z,L1355
        LD  BC,08H
        LD HL,(PTRFIL)
        ADD HL,BC
        CALL ISFLIO
        LD A,(HL)
        JR NZ,L1355
        LD  A,(ODEVLINK)
        OR A
        JP Z,01352H
        LD  A,(LPOS)
        JR L1355
        LD  A,(TTYPOS)
L1355:
        CPL
        ADD A,E
        JR NC,L1360
L1359:
        INC A
        LD B,A
        LD  A,020H
L135D:
        RST 18H
        DJNZ L135D
L1360:
        POP HL
        RST 10H
        JP L126F
FINPRT: 1365
        CALL OFF27H
        XOR A
        LD (ODEVLINK),A
        PUSH HL
        LD H,A
        LD L,A
        LD (PTRFIL),HL
        POP HL
        RET


L1374:                  ;************** LINE
        CP 085H ; INPUT
        JP NZ,GLINE
        RST 8H
        DEFB 085H       ; LINE INPUT
        CP '#'
        JP Z,DLINE
```

```
        CALL L13E1
        CALL PTRGET
        CALL CHKSTR/FRCSTR
        PUSH DE
        PUSH HL
        CALL INLIN
        POP DE
        POP BC
        JP C,STPEND
        PUSH BC
        PUSH DE
        LD B,00H
        CALL STRLT3
        POP HL
        LD  A,03H
        JP L10D2
L13A0:
        DEFB '?Redo from start',13,10,0

L13B3:
        CALL 0FF1EH
        LD  A,(USFLG)
        OR A
        JP NZ,L8E7
        POP BC
        LD  HL,013A0H
        CALL STROUT
        LD HL,(SAVTXT)
        RET
L13C8:
        CALL FILINP
        PUSH HL
        LD  HL,BUFMIN
        JP L1401
L13D2:                  ;*************** INPUT
        CP '#'
        JR Z,L13C8
        PUSH HL
        PUSH AF
        CALL TOTEXT     ; SCREEN 0
        POP AF
        POP HL
        LD  BC,L13F1
        PUSH BC
L13E1:
        CP '"'
        LD  A,00H
        RET NZ
        CALL STRLTI
        RST 8H
        DEFB ';'
        PUSH HL
        CALL STRPRT
        POP HL
        RET
L13F1:
        PUSH HL
        CALL QINLIN
        POP BC
        JP C,STPEND
        INC HL
```



```
        LD A,(HL)
        OR A
        DEC HL
        PUSH BC
        JP Z,L109A
L1401:
        LD (HL),','
        JR L140A
L1405:                  ;***************** READ
        PUSH HL
        LD HL,(DATPTR)
        DEFB 0F6H               ;*      OR 0AFH
L140A:
        XOR A
        LD (USFLG),A
        EX (SP),HL
        DEFB 1                  ;*      LD  BC,02CCFH
L1410:
        RST 8H                  ;*
        DEFB ','                ;*
        CALL PTRGET
        EX (SP),HL
        PUSH DE
        LD A,(HL)
        CP ','
        JR Z,L1437
        LD  A,(USFLG)
        OR A
        JP NZ,L14A6
        LD  A,'?'
        RST 18H
        CALL QINLIN
        POP DE
        POP BC
        JP C,STPEND
        INC HL
        LD A,(HL)
        DEC HL
        OR A
        PUSH BC
        JP Z,L109A
        PUSH DE
L1437:
        CALL ISFLIO
        JP NZ,FILIND
        RST 30H
        PUSH AF
        JR NZ,L1463
        RST 10H
        LD D,A
        LD B,A
        CP '"'
        JR Z,L1454
        LD  A,(USFLG)
        OR A
        LD D,A
        JR Z,L1451
        LD D,':'
L1451:
        LD B,','
        DEC HL
```

```
L1454:
        CALL STRLT2
DOASIG:/1457
        POP AF
        ADD A,03H
        EX DE,HL
        LD  HL,0146BH
        EX (SP),HL
        PUSH DE
        JP L10D3
L1463:
        RST 10H
        LD  BC,DOASIG
        PUSH BC
        JP FINDBL
        DEC HL
        RST 10H
        JR Z,L1474
        CP ','
        JP NZ,L13B3
L1474:
        EX (SP),HL
        DEC HL
        RST 10H
        JP NZ,L1410
        POP DE
        LD  A,(USFLG)
        OR A
        EX DE,HL
        JP NZ,RESFIN
        PUSH DE
        CALL ISFLIO
        JR NZ,L1491
        LD A,(HL)
        OR A
        LD  HL,L1495
        CALL NZ,STROUT
L1491:
        POP HL
        JP FINPRT
L1495:
        DEFB '?Extra ignored',13,10,0

L14A6:
        CALL L109B
        OR A
        JR NZ,L14BD
        INC HL
        LD A,(HL)
        INC HL
        OR (HL)
        LD  E,04H
        JP Z,ERROR
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        LD (0F7CFH),DE
L14BD:
        RST 10H
        CP 084H
```



```
        JR NZ,L14A6
        JP L1437

FRMEQL:/14C5
        RST 8H
        DEFB 0F1H           ;=
        DEFB 1              ;**     LD  BC,028CFH
FRMPRN:/14C8
        RST 8H              ;**
        DEFB '('            ;**
FRMEVL:/14CA
        DEC HL
FRMCHK:/14CB                ; Ausdruck berechnen
        LD D,00H
L14CD:
        PUSH DE
        LD  C,01H
        CALL GETSTK 652E
        CALL 0FF2AH RET
        CALL EVAL  1629
TSTOP:/14D9
        LD (TEMP2),HL
L14DC:
        LD HL,(TEMP2)
        POP BC
        LD A,(HL)
        LD (TEMP3),HL
        CP 0F0H             ; kein Operator Token falls C
        RET C
        CP 0F3H
        JR C,L154A          ; JP falls < > =
        SUB 0F3H
        LD E,A
        JR NZ,L14F9         ; JP falls kein +
        LD  A,(VALTYP)
        CP 03H
        LD A,E
        JP Z,CAT            ; fuer Strings CONCAT
L14F9:
        CP 0CH
        RET NC
        LD  HL,059CH        ; Operatos-Praezedenz-Tabelle
        LD D,00H
        ADD HL,DE
        LD A,B
        LD D,(HL)           ; Vergleich mit letztem Operator
        CP D
        RET NC              ; RET falls dieser hoehere prioritaet hat
        PUSH BC
        LD  BC,L14DC        ; wieder Eintrittspunkt
        PUSH BC
        LD A,D
        CALL 0FEBEH
        CP 051H
        JR C,L1563          ; JP falls logische OPS
        AND 0FEH
        CP 07AH
        JR Z,L1563
L1519:
        LD  HL,FACLOW
```

```
        LD  A,(VALTYP)
        SUB 03H
        JP Z,TMERR      ; TYPE MISMATCH
        OR A
        LD HL,(FACLOW)
        PUSH HL
        JP M,L153B
        LD HL,(FACCU)
        PUSH HL
        JP PO,L153B
        LD HL,(OF929H)
        PUSH HL
        LD HL,(OF927H)
        PUSH HL
L153B:
        ADD A,03H
        LD C,E
        LD B,A
        PUSH BC
        LD  BC,L1588
L1543:
        PUSH BC
        LD HL,(TEMP3)
        JP L14CD
L154A:
        LD D,00H
L154C:
        SUB 0F0H
        JR C,L156E
        CP 03H
        JR NC,L156E
        CP 01H
        RLA
        XOR D
        CP D
        LD D,A
        JP C,SNERR
        LD (TEMP3),HL
        RST 10H
        JR L154C
L1563:                  ; logische Operatoren
        PUSH DE
        CALL FRCINT
        POP DE
        PUSH HL
        LD  BC,L17E5
        JR L1543
L156E:
        LD A,B
        CP 064H
        RET NC
        PUSH BC
        PUSH DE
        LD  DE,06405H
        LD  HL,017B9H
        PUSH HL
        RST 30H
        JP NZ,L1519
        LD HL,(FACLOW)
        PUSH HL
        LD  BC,STRCMP
```



```
        JR L1543
L1588:
        POP BC          ; B Datentyp   C Operator
        LD A,C
        LD (0F794H),A
        LD  A,(VALTYP)
        CP B
        JR NZ,L159E
        CP 02H
        JR Z,L15B6      ; INTEGER
        CP 04H
        JP Z,L1603      ; SINGLE
        JR NC,L15C9     ; DOUBLE
L159E:
        LD D,A
        LD A,B
        CP 08H
        JR Z,L15C6
        LD A,D
        CP 08H
        JR Z,L15ED
        LD A,B
        CP 04H
        JR Z,L1600
        LD A,D
        CP 03H
        JP Z,TMERR
        JR NC,L160A
L15B6:
        LD  HL,05CAH            ; Adressberechnung der
        LD B,00H                ; INTEGER Arithmetik Routine
        ADD HL,BC
        ADD HL,BC
        LD C,(HL)
        INC HL
        LD B,(HL)
        POP DE
        LD HL,(FACLOW)
        PUSH BC
        RET
L15C6:
        CALL FRCDBL
L15C9:                  ; DOUBLE PRECISION
        CALL VMOVAF
        POP HL
        LD (0F927H),HL
        POP HL
        LD (0F929H),HL
L15D4:
        POP BC
        POP DE
        CALL MOVFR
L15D9:
        CALL FRCDBL
        LD  HL,05B2H
L15DF:
        LD  A,(0F794H)
        RLCA
        ADD A,L
        LD L,A
        ADC A,H
```

```
        SUB L
        LD H,A          ; HL:=HL+2*(0F794H)
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        JP (HL)
L15ED:
        LD A,B
        PUSH AF
        CALL VMOVAF
        POP AF
        LD (VALTYP),A
        CP 04H
        JR Z,L15D4
        POP HL
        LD (FACLOW),HL
        JR L15D9
L1600:
        CALL FRCSNG
L1603:                  ; SINGLE PRECISION
        POP BC
        POP DE
L1605:
        LD  HL,05BEH
        JR L15DF

L160A:
        POP HL
        CALL PUSHF
        CALL CONSIH
        CALL MOVRF
        POP HL
        LD (FACCU),HL
        POP HL
        LD (FACLOW),HL
        JR L1605
L161E:
        PUSH HL
        EX DE,HL
        CALL CONSIH
        POP HL
        CALL PUSHF
        CALL CONSIH
        JP FDIVT
EVAL:   LD D,0          ;************ Argument -> FACCU
        RST 10H
        JP Z,MOERR
        JP C,FINDBL
        CALL ISLET2
        JP NC,ISVAR
        CP ' '
        JP C,LEFC
        CALL LFEFA
        INC A
        JP Z,L175E      ; jp falls 0FFH Token
        DEC A
        CP 0F3H         ; +
        JR Z,EVAL
        CP 0F4H         ; -
        JP Z,L16EF
```



```
        CP '"'
        JP Z,STRLTI
        CP 0E0H         ; NOT
        JP Z,L17C5
        CP '&'
        JP Z,OCTCNS
        CP 0E2H         ; ERR
        JR NZ,L166D
        RST 10H
        LD  A,(ERRFLG)
        PUSH HL
        CALL SNGFLT
        POP HL
        RET
L166D:
        CP 0E1H         ; ERL
        JR NZ,L167B
        RST 10H
        PUSH HL
        LD HL,(ERL)
        CALL INEG2
        POP HL
        RET
L167B:
        CP 0EFH         ; TIME
        JP Z,TIME
        CP 0EDH
        JP Z,POINT
        CP 0EEH         ; =SPRITE$
        JP Z,RETSPR
        CP 0C1H         ; PLAY
        JP Z,PLAYF
        CP 0C9H         ; SWITCH
        JP Z,RETSWI
        CP 0EAH
        JP Z,DSKI$
        CP 0E9H
        JP Z,ATTR$
        CP 0E7H         ; VARPTR
        JR NZ,L16C6
        RST 10H
        RST 8H
        DEFB '('
        CP '#'
        JR NZ,L16B5
        CALL GTBYTC     ; VARPTR auf Datei
        PUSH HL
        CALL GETPTR
        EX DE,HL
        POP HL
        JP L16B8
L16B5:
        CALL PTRGTN     ; VARPTR auf Variable
L16B8:
        RST 8H
        DEFB ')'
        PUSH HL
        EX DE,HL
        LD A,H
        OR L
        JP Z,FCERR
```

```
        CALL MAKINT
        POP HL
        RET
L16C6:
        CP 0DDH         ; USR
        JP Z,L1842
        CP 0E5H
        JP Z,INSTR
        CP 0ECH
        JP Z,INKEY
        CP 0E3H         ; STRING$
        JP Z,STRNG$
        CP 085H         ; INPUT
        JP Z,FIXINP
        CP 0E8H         ; CSRLIN
        JP Z,GETLIN
        CP 0DEH         ; FN
        JP Z,L18AD
PARCHK: 16E9
        CALL FRMPRN
        RST 8H
        DEFB ')'
        RET
L16EF:                  ; UNARY minus
        LD D,07DH
        CALL L14CD
        LD HL,(TEMP2)
        PUSH HL
        CALL VNEG
L16FB:
        POP HL
        RET
ISVAR:  16FD            ; Variable im Ausdruck
        CALL PTRGET
RETVAR: 1700            ; Wert liefern
        PUSH HL
        EX DE,HL
        LD (FACLOW),HL
        RST 30H
        CALL NZ,VMOVFM
        POP HL
        RET
MAKUPL: 1708
        LD A,(HL)
MAKUPS: 170C            ; Kleinbuchstaben --> Grossbuchstaben
        CP 061H         ; wandeln
        RET C
        CP 07BH
        RET NC
        AND 05FH
        RET
CNSGET: 1715            ; Konstante im Ausdruck
        CP '&'
        JP NZ,LINGET    ; Dezimalzahl
OCTCNS:
        LD DE,00H       ; Null zu Anfang
        RST 10H
        CALL MAKUPS
        LD BC,0102H     ; &B
        CP 'B'
        JR Z,L1737
```



```
        LD BC,0308H     ; &O
        CP 'O'
        JR Z,L1737
        LD BC,0410H     ; &H
        CP 'H'
        JP NZ,SNERR
L1737:                  ; Erzeugen der Konstante
        INC HL          ;   B Shiftfaktor       C Bereichsgrenze
        LD A,(HL)
        EX DE,HL
        CALL MAKUPS     ; naechster Grossbuchstabe
        CP 03AH
        JR C,L1747
        CP 041H
        JR C,L1759
        SUB 07H
L1747:
        SUB 030H        ; hexadezimal Zahl
        CP C
        JR NC,L1759     ; Bereichstest  C innerhalb
        PUSH BC         ; sichern
L174D:
        ADD HL,HL       ; bisherigen Inhalt shiften
        JP C,DVERR      ; Ueberlauf
        DJNZ L174D
        POP BC          ; alte Parameter
        OR L            ; Stelle hinzufuegen
        LD L,A
        EX DE,HL        ; retten nach DE
        JR L1737
L1759:
        CALL MAKINT
        EX DE,HL
        RET
L175E:                  ; ************ Token OFFH
        INC HL
        LD A,(HL)
        SUB 081H
        LD B,00H
        RLCA
        LD C,A
        PUSH BC         ; (Token-081H)*2
        RST 10H
        LD A,C
        CP 05H
        JR NC,L1783     ; NC weder LEFT$ noch RIGHT$
        CALL FRMPRN     ;******************* LEFT$ ** RIGHT$
        RST 8H
        DEFB ','
        CALL CHKSTR/FRCSTR
        EX DE,HL
        LD HL,(FACLOW)
        EX (SP),HL
        PUSH HL
        EX DE,HL
        CALL GETBYT
        EX DE,HL
        EX (SP),HL
        JR L179D
L1783:
        CALL PARCHK     ; Parameter Berechnung
```

```
        EX (SP),HL      ; letztes Token nach HL
        LD A,L
        CP 0CH
        JR C,L1799
        CP 01BH
        CALL LFEDC
        JR NC,L1799
        RST 30H         ; Test des Datentyps
        PUSH HL
        CALL C,FRCDBL   ; Argument --> doppelte Genauigkeit
        POP HL          ; fuer Tokens 6-13
L1799:
        LD  DE,L16FB
        PUSH DE
L179D:
        LD  BC,0235H    ; Tokentabelle ab 81H
        CALL LFF39
L17A3:
        ADD HL,BC
        LD C,(HL)
        INC HL
        LD H,(HL)
        LD L,C
        JP (HL)         ; Sprung zur Ausfuehrung
MINPLS: 17A9
        DEC D
        CP 0F4H         ; -
        RET Z
        CP '-'
        RET Z
        INC D
        CP '+'
        RET Z
        CP 0F3H         ; +
        RET Z
        DEC HL
        RET
L17B9:                  ; ???????????????????????? kein Einsprung
        INC A
        ADC A,A
        POP BC
        AND B
        ADD A,0FFH
        SBC A,A
        CALL CONIA
        JR L17D7
L17C5:                  ; ************** NOT
        LD D,05AH
        CALL L14CD
        CALL FRCINT     ; Argument --> INTEGER
        LD A,L          ; Negation
        CPL             ;*
        LD L,A          ;*
        LD A,H          ;*
        CPL             ;*
        LD H,A          ;*
        LD (FACLOW),HL  ; speichern
        POP BC
L17D7:
        JP L14DC
L17DA:                  ; Fortsetzen des Datentyptests
```

FORTSETZUNG VON 16ST 0H


```
        JR NC,L17E1
        SUB 03H
        OR A
        SCF
        RET
L17E1:
        SUB 03H
        OR A
        RET
L17E5:                  ; Ausfuehrung logischer Operatoren
        LD A,B
        PUSH AF
        CALL FRCINT     ; Integer Argument
        POP AF          ; Operator- Praezedenz
        POP DE
        CP 07AH         ; MOD
        JP Z,IMOD
        CP 07BH         ; DIV
        JP Z,IDIV
        LD  BC,GIVINT
        PUSH BC
        CP 046H
        JR NZ,L1804
        LD A,E          ; OR
        OR L
GRPNAM: 1800
        LD L,A
        LD A,H
        OR D
        RET
L1804:
        CP 050H
        JR NZ,L180E
        LD A,E          ; AND
        AND L
        LD L,A
        LD A,H
        AND D
        RET
L180E:
        CP 03CH
        JR NZ,L1818
        LD A,E          ; XOR
        XOR L
        LD L,A
        LD A,H
        XOR D
        RET
L1818:
        CP 032H
        JR NZ,L1824
        LD A,E          ; EQV
        XOR L
        CPL
        LD L,A
        LD A,H
        XOR D
        CPL
        RET
L1824:                  ; IMP
        LD A,L
```

```
        CPL
        AND E
        CPL
        LD L,A
        LD A,H
        CPL
        AND D
        CPL
        RET
GIVDBL: /182E
        OR A
        SBC HL,DE
        JP INEG2
L1834:                  ;****************** LPOS
        LD  A,(LPOS)
        JR SNGFLT
L1839:                  ; ****************** POS
        LD  A,(TTYPOS)
SNGFLT: /183C
        LD L,A
        XOR A
GIVINT: /183E
        LD H,A
        JP MAKINT
L1842:                  ; ****************** USR
        CALL L1861
        PUSH DE
        CALL PARCHK
        EX (SP),HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        LD  HL,POPHRT   ; RETURN Adresse fuer USR Routine
        PUSH HL
        PUSH DE         ; Adresse der USR Routine
        LD  A,(VALTYP)
        PUSH AF         ; Datentyp des Arguments
        CP 03H
        CALL Z,FREFAC
        POP AF
        EX DE,HL
        LD  HL,FACCU
        RET             ; RETURN zur USR Routine
L1861:
        RST 10H
        LD  BC,00H
        CP 01BH
        JR NC,L1874     ; >26
        CP 011H
        JR C,L1874      ; <17
        RST 10H
        LD  A,(0F79AH)
        OR A
        RLA
        LD C,A
L1874:
        EX DE,HL
        LD  HL,USRTAB
        ADD HL,BC
        EX DE,HL
        RET
```



```
L187B:                  ;****************** DEFUSR
        CALL L1861
        PUSH DE
        RST 8H
        DEFB 0F1H
        CALL FRMQNT
        EX (SP),HL
        LD (HL),E
        INC HL
        LD (HL),D
        POP HL
        RET
L188A:                  ;****************** DEF
        CP 0DDH
        JR Z,L187B
        CALL L1A0E      ; ****************** DEFFN
        CALL L1A00
        EX DE,HL
        LD (HL),E
        INC HL
        LD (HL),D
        EX DE,HL
        LD A,(HL)
        CP '('
        JP NZ,L109B
        RST 10H
L18A0:
        CALL PTRGET
        LD A,(HL)
        CP ')'
        JP Z,L109B
        RST 8H
        DEFB ','
        JR L18A0
L18AD:                  ; ****************** FN
        CALL L1A0E
        LD  A,(VALTYP)
        OR A
        PUSH AF
        LD (TEMP2),HL
        EX DE,HL
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        LD A,H
        OR L
        JP Z,L8F9
        LD A,(HL)
        CP '('
        JP NZ,L1961
        RST 10H
        LD (TEMP3),HL
        EX DE,HL
        LD HL,(TEMP2)
        RST 8H
        DEFB '('
        XOR A
        PUSH AF
        PUSH HL
        EX DE,HL
```

```
L18D6:
        LD  A,080H
        LD (SUBFLG),A
        CALL PTRGET
        EX DE,HL
        EX (SP),HL
        LD  A,(VALTYP)
        PUSH AF
        PUSH DE
        CALL FRMEVL
        LD (TEMP2),HL
        POP HL
        LD (TEMP3),HL
        POP AF
        CALL DOCNVF
        LD  C,04H
        CALL GETSTK
        LD  HL,0FFF8H
        ADD HL,SP
        LD SP,HL
        CALL VMOVMF
        LD  A,(VALTYP)
        PUSH AF
        LD HL,(TEMP2)
        LD A,(HL)
        CP ')'
        JR Z,L191A
        RST 8H
        DEFB ','
        PUSH HL
        LD HL,(TEMP3)
        RST 8H
        DEFB ','
        JR L18D6
        POP AF
        LD (PARMLN2),A
L191A:
        POP AF
        OR A
        JR Z,L1956
        LD (VALTYP),A
        LD  HL,00H
        ADD HL,SP
        CALL VMOVFM
        LD  HL,08H
        ADD HL,SP
        LD SP,HL
        POP DE
        LD  L,03H
        DEC DE
        DEC DE
        DEC DE
        LD  A,(VALTYP)
        ADD A,L
        LD B,A
        LD  A,(PARMLN2)
        LD C,A
        ADD A,B
        CP 064H
        JP NC,FCERR
        PUSH AF
```



```
        LD A,L
        LD B,00H
        LD  HL,0F87CH
        ADD HL,BC
        LD C,A
        CALL L19FB
        LD  BC,01916H
        PUSH BC
        PUSH BC
        JP L1ODE
L1956:
        LD HL,(TEMP2)
        RST 10H
        PUSH HL
        LD HL,(TEMP3)
        RST 8H
        DEFB ')'
        DEFB 03EH                ;*      LD  A,0D5H
L1961:
        PUSH DE                  ;*
        LD (TEMP3),HL
        LD  A,(PRMLEN)
        ADD A,04H
        PUSH AF
        RRCA
        LD C,A
        CALL GETSTK
        POP AF
        LD C,A
        CPL
        INC A
        LD L,A
        LD H,0FFH
        ADD HL,SP
        LD SP,HL
        PUSH HL
        LD  DE,PRMSTK
        CALL L19FB
        POP HL
        LD (PRMSTK),HL
        LD HL,(PARMLN2)
        LD (PRMLEN),HL
        LD B,H
        LD C,L
        LD  HL,PARM1
        LD  DE,0F87CH
        CALL L19FB
        LD H,A
        LD L,A
        LD (PARMLN2),HL
        LD HL,(FUNACT)
        INC HL
        LD (FUNACT),HL
        LD A,H
        OR L
        LD (NOFUNS),A
        LD HL,(TEMP3)
        CALL FRMEQL
        DEC HL
        RST 10H
        JP NZ,SNERR
```

```
        RST 30H
        JR NZ,L19C3
        LD  DE,DSCTMP
        LD HL,(FACLOW)
        RST 20H
        JR C,L19C3
        CALL STRCPY
        CALL PUTTMP
L19C3:
        LD HL,(PRMSTK)
        LD D,H
        LD E,L
        INC HL
        INC HL
        LD C,(HL)
        INC HL
        LD B,(HL)
        INC BC
        INC BC
        INC BC
        INC BC          ; BC:=BC+4
        LD  HL,PRMSTK
        CALL L19FB      ; Move BC Bytes   (DE) --> (HL)
        EX DE,HL
        LD SP,HL
        LD HL,(FUNACT)
        DEC HL
        LD (FUNACT),HL
        LD A,H
        OR L
        LD (NOFUNS),A
        POP HL
        POP AF
DOCNVF:
        PUSH HL
        AND 07H
        LD  HL,05A8H
        LD C,A
        LD B,00H
        ADD HL,BC
        CALL L17A3
        POP HL
        RET
L19F6:
        LD  A,(DE)
        LD (HL),A
        INC HL
        INC DE
        DEC BC
L19FB:
        LD A,B
        OR C
        JR NZ,L19F6
        RET
L1A00:
        PUSH HL
        LD HL,(CURLIN)
        INC HL
        LD A,H
        OR L
        POP HL
```



```
        RET NZ
        LD  E,0CH
        JP ERROR
L1A0E:
        RST 8H
        DEFB 0DEH       ; FN
        LD A,080H
        LD (SUBFLG),A
        OR (HL)
        LD C,A
        JP PTRGT2
L1A1A:                  ; fortsetzen von E9C
        CP 07EH         ; Token 0FFH am Anfang einer Programmzeile
        JP NZ,SNERR
        INC HL
        LD A,(HL)       ; folgendes Token
        INC HL
        CP 083H         ; MID$()=
        JP Z,LHSMID
        CP 0A3H         ; STRIG ...
        JP Z,STRIG
        CP 085H         ; INT(ERRUPT ...)
        JP Z,INTTRP
        CALL LFEFD
        JP SNERR
L1A37:                  ; ********************* INP
        CALL FRQINT
        LD B,H
        LD C,L
        IN A,(C)
        JP SNGFLT
L1A41:                  ;**   PP,P
        CALL FRMQNT
        PUSH DE         ; save Parameter
        RST 8H
        DEFB ','
        CALL GETBYT     ; naechster Parameter nach A
        POP BC          ; erster Parameter
        RET
L1A4C:                  ; ************************ OUT
        CALL L1A41
        OUT (C),A
        RET
L1A52:                  ; ************************ WAIT
        CALL L1A41
        PUSH BC
        PUSH AF
        LD  E,00H
        DEC HL
        RST 10H
        JR Z,L1A62
        RST 8H
        DEFB ','
        CALL GETBYT
L1A62:
        POP AF
        LD D,A
        POP BC
L1A65:
        IN A,(C)
        XOR E
```

```
        AND D
        JR Z,L1A65
        RET
L1A6C:                  ; ********************** WIDTH
        CALL GETBYT
        CALL LFF3F
        CP 40
        JR Z,L1A7B
        CP 39
        JP NZ,FCERR
L1A7B:
        LD  A,(LINLEN)
        CP E
        RET Z
        LD  A,0CH       ; Formfeed ausgeben
        RST 18H
        LD A,E
        LD (LINLEN),A
        LD  A,0CH
        RST 18H
        LD A,E
L1A8B:
        SUB 14
        JR NC,L1A8B
        ADD A,28
        CPL
        INC A
        ADD A,E
        LD (0F544H),A
        RET
GETINT: 1A98
        RST 10H
GETIN2: 1A99
        CALL FRMEVL
INTFR2: 1A9C
        PUSH HL
        CALL FRCINT
        EX DE,HL
        POP HL
        LD A,D
        OR A
        RET
GTBYTC: 1AA5
        RST 10H
GETBYT: 1AA6
        CALL FRMEVL
CONINT: 1AA9
        CALL INTFR2
        JP NZ,FCERR
        DEC HL
        RST 10H
        LD A,E
        RET
L1AB3:                  ;********************** LLIST
        LD  A,01H
        LD (0DEVLINK),A
LIST:   1AB8            ;********************** LIST
        CALL LFEF4
        POP BC
        CALL LBOB       ; Zeilenbereich holen
        PUSH BC         ; Zeiger auf Startzeile speichern
```



```
L1ACO:
        LD  HL,0FFFFH
        LD (CURLIN),HL  ; direkt Modus setzen
        POP HL          ; Zeiger auf aktuelle Zeile
        POP DE          ; Letzte Zeilennummer
        LD C,(HL)
        INC HL
        LD B,(HL)       ; Zeiger auf folgende Programmzeile
        INC HL
        LD A,B
        OR C            ; falls =0
        JP Z,READY      ; Ende des Programms
        CALL ISFLIO 6882
        CALL Z,ISCNTC   ; CTRL-STOP Test
        PUSH BC
        LD C,(HL)
        INC HL
        LD B,(HL)
        INC HL
        PUSH BC         ; aktuelle Zeilennummer
        EX (SP),HL
        EX DE,HL
        RST 20H         ; falls groesser der Letzten
        POP BC
        JP C,STPRDY     ; fertig mit LIST wieder zur Eingabe 9AE
        EX (SP),HL
        PUSH HL
        PUSH BC
        EX DE,HL
        LD (DOT),HL
        CALL LINPRT     ; Zeilennummer
        POP HL
        LD A,(HL)
        CP 09H
        JR Z,L1AF7
        LD  A,' '
        RST 18H
L1AF7:
        CALL BUFLIN     ; Prgzeile --> ASCII
        LD  HL,BUF
        CALL LISPRT
SPRATR: 1B00
        CALL CRDO       ; CR LF
        JR L1ACO        ; weiter LISTen
LISPRT: 1B06            ;****schickt einen Text bis zum Zeichen
        LD A,(HL)       ;      0 zum aktuellen Output Device
        OR A            ; HL  Zeiger auf erstes Textzeichen
        RET Z
        CALL OUTCH1 6513
        INC HL
        JR LISPRT
BUFLIN: 1B0E            ;*****  Expandiert eine Programmzeile
        LD  BC,BUF      ;       nach BUF
        LD D,0FFH       ; HL  Zeiger auf Programmtext nach
        XOR A           ;       zeilennummer
        LD (0F794H),A   ; Loeschen der Flags
        JR L1B1D
L1B19:
        INC BC
        INC HL
        DEC D
```

```
        RET Z
L1B1D:
        LD A,(HL)          ; Token holen
        OR A
        LD (BC),A
        RET Z
        CP 0BH
        JR C,L1B4A
        CP ' '
        JP C,L1BEB
        CP '"'
        JR NZ,L1B38
        LD  A,(0F794H)     ; String Anfang/Ende
        XOR 01H
        LD (0F794H),A
        LD  A,'"'
L1B38:
        CP ':'
        JR NZ,L1B4A
        LD  A,(0F794H)     ; Test ob ':' innerhalb des Strings war
        RRA
        JR C,L1B48         ; JP falls wahr
        RLA
        AND 0FDH           ; Statement Start setzen
        LD (0F794H),A
L1B48:
        LD  A,':'
L1B4A:
        OR A
        JP P,L1B19         ; JP falls Name oder Zeichen
        LD  A,(0F794H)
        RRA
        JR C,L1B82         ; JP falls Token im String
        RRA
        RRA
        JR NC,L1B96
        LD A,(HL)
        CP 0E6H
        PUSH HL
        PUSH BC
        LD  HL,L1B7F
        PUSH HL
        RET NZ             ; RET falls nicht ' Token
        DEC BC
        LD  A,(BC)
        CP 'M'
        RET NZ
        DEC BC
        LD  A,(BC)
        CP 'E'
        RET NZ
        DEC BC
        LD  A,(BC)
        CP 'R'
        RET NZ
        DEC BC
        LD  A,(BC)
        CP ':'
        RET NZ
        POP AF
        POP AF
```



```
        POP HL
        INC D
        INC D
        INC D
        INC D
        JR L1BA4
L1B7F:
        POP BC
        POP HL
        LD A,(HL)
L1B82:
        JP L1B19
        LD  A,(0F794H)
        OR 02H             ; Setzen des Start-Statements
L1B8A:
        LD (0F794H),A
        XOR A
        RET
L1B8F:
        LD  A,(0F794H)
        OR 04H             ; Setzen des Kommentar-Flags
        JR L1B8A
L1B96:                     ; Listen eines Token
        RLA
        JR C,L1B82
        LD A,(HL)
        CP 084H            ; DATA
        CALL Z,01B85H
        CP 08FH            ; REM
        CALL Z,01B8FH
L1BA4:
        LD A,(HL)
        INC A
        LD A,(HL)
        JR NZ,L1BAD        ; JP falls Token <> 0FFH
        INC HL
        LD A,(HL)
        AND 07FH           ; Listen von 0FFH ...
L1BAD:
        INC HL
        CP 0A1H
        JR NZ,L1BB4
        DEC BC             ; nur fuer ELSE-Token
        INC D
L1BB4:
        PUSH HL
        PUSH BC
        PUSH DE
        CALL LFED6         ; A  Token
        LD  HL,02C8H       ; HL Adresse der Statement Token-Tabelle
        LD B,A             ; B  Token
        LD  C,'A'-1        ; Anfangsbuchstabe des Wortes-1
L1BC0:
        INC C
L1BC1:
        INC HL
        LD D,H
        LD E,L
L1BC4:
        LD A,(HL)
        OR A
```

```
        JR Z,L1BC0      ; naechster Anfangsbuchstabe
        INC HL
        JP P,L1BC4      ; Wort noch nicht beendet
        LD A,(HL)
        CP B
        JR NZ,L1BC1     ; neues Wort
        EX DE,HL
        LD A,C
        POP DE
        POP BC
        CP 'Z'+1
        JR NZ,L1BDA     ; neue Wortreihe
L1BD8:
        LD A,(HL)
        INC HL
L1BDA:
        LD E,A
        AND 07FH
        LD (BC),A       ; Zeichen speichern
        INC BC
        DEC D
        JP Z,PPSWRT
        OR E
        JP P,L1BD8      ; naechster Buchstabe des Wortes
        POP HL
        JP L1B1D
L1BEB:                  ; listen der Token 0BH ... 1FH
        DEC HL
        RST 10H
        PUSH DE
        PUSH BC
        PUSH AF
        CALL LF2C
        POP AF
        LD  BC,L1C08
        PUSH BC
        CP 0BH
        JP Z,FOUTD
        CP 0CH
        JP Z,FOUTH
        LD HL,(0F79AH)
        JP FOUT
L1C08:
        POP BC
        POP DE
        LD  A,(CONSAV)
        LD  E,'0'
        CP 0BH
        JR Z,L1C19
        CP 0CH
        LD  E,'H'
        JR NZ,L1C24
L1C19:
        LD  A,'&'
        LD (BC),A
        INC BC
        DEC D
        RET Z
        LD A,E
        LD (BC),A
        INC BC
```



```
        DEC D
        RET Z
L1C24:
        LD  A,(0F799H)
        CP 04H
        LD  E,00H
        JR C,L1C33
        LD  E,'!'
        JR Z,L1C33
        LD  E,'#'
L1C33:
        LD A,(HL)
        CP ' '
        CALL Z,INXHRT
L1C39:
        LD A,(HL)
        INC HL
        OR A
        JR Z,L1C5E
        LD (BC),A
        INC BC
        DEC D
        RET Z
        LD  A,(0F799H)
        CP 04H
        JR C,L1C39
        DEC BC
        LD  A,(BC)
        INC BC
        JR NZ,L1C52
        CP '.'
        JR Z,L1C5A
L1C52:
        CP 'D'
        JR Z,L1C5A
        CP 'E'
        JR NZ,L1C39
L1C5A:
        LD  E,00H
        JR L1C39
L1C5E:
        LD A,E
        OR A
        JR Z,L1C66
        LD (BC),A
        INC BC
        DEC D
        RET Z
L1C66:
        LD HL,(0F796H)
        JP L1B1D
L1C6C:                  ; ************************ DELETE
        CALL LBOB
        PUSH BC
        CALL DEPTR
        POP BC
        POP DE
        PUSH BC
        PUSH BC
        CALL FNDLIN
```

```
        JR NC,L1C81
        LD D,H
        LD E,L
        EX (SP),HL
        PUSH HL
        RST 20H
L1C81:
        JP NC,FCERR
        LD  HL,REDDY
        CALL STROUT
        POP BC
        LD  HL,FINI
        EX (SP),HL
DEL:                    ; Verschieben der nach HL folgenden
        EX DE,HL        ; Zeilen bis zum Programmende -> BC
        LD HL,(VARTAB)
L1C93:
        LD  A,(DE)
        LD (BC),A
        INC BC
        INC DE
        RST 20H
        JR NZ,L1C93
        LD H,B
        LD L,C
        LD (VARTAB),HL  ; loeschen der Variablen und Felder
        LD (ARYTAB),HL
        LD (STREND),HL
        RET
L1CA6:                  ; ********************** PEEK
        CALL FRQINT
        LD A,(HL)
        JP SNGFLT
L1CAD:
        CALL FRMQNT
        PUSH DE
        RST 8H
        DEFB ','
        CALL GETBYT
        POP DE
        LD (DE),A
        RET
FRMQNT:
        CALL FRMEVL
        PUSH HL
        CALL FRQINT
        EX DE,HL
        POP HL
        RET
FRQINT:
        LD  BC,FRCINT
        PUSH BC
        RST 30H
        RET M
        CALL LFF03
        RST 28H
        RET M           ; ret falls INTEGER
        CALL FRCSNG     ; einfache Genauigkeit
        LD  BC,03245H
        LD  DE,08076H
        CALL FCOMP
```



```
        RET C
        LD  BC,06545H
        LD  DE,06053H
        CALL FCOMP
        JP NC,DVERR
        LD  BC,065C5H
        LD  DE,06053H
        JP FADD
L1CF0:                  ; ********************** RENUM
        LD  BC,0AH
        PUSH BC
        LD D,B
        LD E,B
        JR Z,L1D1E
        CP 02CH
        JR Z,L1D05
        PUSH DE
        CALL LINSPC
        LD B,D
        LD C,E
        POP DE
        JR Z,L1D1E
L1D05:
        RST 8H
        DEFB ','
        CALL LINSPC
        JR Z,L1D1E
        POP AF
        RST 8H
        DEFB ','
        PUSH DE
        CALL LINGET
        JP NZ,SNERR
        LD A,D
        OR E
        JP Z,FCERR
        EX DE,HL
        EX (SP),HL
        EX DE,HL
L1D1E:
        PUSH BC
        CALL FNDLIN
        POP DE
        PUSH DE
        PUSH BC
        CALL FNDLIN
        LD H,B
        LD L,C
        POP DE
        RST 20H
        EX DE,HL
        JP C,FCERR
        POP DE
        POP BC
        POP AF
        PUSH HL
        PUSH DE
        JR L1D45
L1D37:
        ADD HL,BC
        JP C,FCERR
```

```
        EX DE,HL
        PUSH HL
        LD  HL,0FFF9H
        RST 20H
        POP HL
        JP C,FCERR
L1D45:
        PUSH DE
        LD E,(HL)
        INC HL
        LD D,(HL)
        LD A,D
        OR E
        EX DE,HL
        POP DE
        JR Z,L1D56
        LD A,(HL)
        INC HL
        OR (HL)
        DEC HL
        EX DE,HL
        JR NZ,L1D37
L1D56:
        PUSH BC
        CALL L1D77
        POP BC
        POP DE
        POP HL
L1D5D:
        PUSH DE
        LD E,(HL)
        INC HL
        LD D,(HL)
        LD A,D
        OR E
        JR Z,L1D72
        EX DE,HL
        EX (SP),HL
        EX DE,HL
        INC HL
        LD (HL),E
        INC HL
        LD (HL),D
        EX DE,HL
        ADD HL,BC
        EX DE,HL
        POP HL
        JR L1D5D
L1D72:
        LD  BC,STPRDY
        PUSH BC
        DEFB 0FEH               ;*      CP 0F6H
L1D77:
        DEFB 0F6H               ;*          OR 0AFH
SCCPTR: /1D78
        XOR A                   ;**
        LD (PTRFLG),A   ; NZ bei RENUM , Z bei DEPTR,SCCPTR
        LD HL,(TXTTAB)
        DEC HL
L1D80:
        INC HL
```



```
        LD A,(HL)
        INC HL
        OR (HL)
        RET Z           ; bei Programmende
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)       ; Pointer zur naechsten Zeile
L1D89:
        RST 10H         ; 1. Token
L1D8A:
        OR A
        JR Z,L1D80      ; bei Anweisungsende
        LD C,A
        LD  A,(PTRFLG)
        OR A
        LD A,C
        JR Z,L1DEB      ; wenn nicht RENUMBER
        CALL LFE85
        CP 0A6H         ; ERROR Token
        JR NZ,L1DB0
        RST 10H
        CP 089H         ; GOTO Token
        JR NZ,L1D8A
        RST 10H
        CP 0EH          ; binaere Zeilennummer folgt
        JR NZ,L1D8A
        PUSH DE
        CALL LFB5
        LD A,D
        OR E
        JR NZ,L1DB8
        JR L1DD7
L1DB0:
        CP 0EH          ; Test binaere Zeilennummer
        JR NZ,L1D89     ; keine
        PUSH DE
        CALL LFB5
L1DB8:
        PUSH HL
        CALL FNDLIN
        DEC BC
        LD  A,0DH
        JR C,L1DFD
        CALL CRDONZ
        LD  HL,L1DDB
        PUSH DE
        CALL STROUT
        POP HL
        CALL LINPRT
        POP BC
        POP HL
        PUSH HL
        PUSH BC
        CALL INPRT
L1DD6:
        POP HL
L1DD7:
        POP DE
        DEC HL
L1DD9:
```

```
        JR L1D89
L1DDB:
        DEFB 'Undefined line ',0

L1DEB:
        CP ODH
        JR NZ,L1DD9     ; keine Zeilennummer
        PUSH DE
        CALL LFB5       ; Zeilennummer nach DE
        PUSH HL
        EX DE,HL
        INC HL
        INC HL
        INC HL
        LD C,(HL)
        INC HL
        LD B,(HL)
        LD  A,OEH       ; binaere Zeilennumemr setzen
L1DFD:
        LD  HL,L1DD6
        PUSH HL
        LD HL,(OF796H)
L1E04:
        PUSH HL
        DEC HL
        LD (HL),B
        DEC HL
        LD (HL),C
        DEC HL
        LD (HL),A
        POP HL
        RET
DEPTR:  1E0D
        LD  A,(PTRFLG)
        OR A
        RET Z
        JP SCCPTR
L1E15:                  ; ********************** CSAVE
        CALL L1F14      ; Filename
        DEC HL
        RST 10H
        LD  A,OFFH      ; Voreinstellung Programm
        JR Z,L1E25      ; keine weiteren Parameter
        RST 8H
        DEFB ','
        RST 8H
        DEFB 'S'        ; oder ,S
        LD  A,(SCREEN)
L1E25:
        LD (CASATR),A   ; Attribut fuer Datei Programm oder Screen
        PUSH HL
        LD  A,OD3H      ; Dateikennsatz schreiben mit D3
        CALL CASOPW
        LD  A,(CASATR)
        AND A
        JP P,SAVSCN     ; JP falls ,S
        LD HL,(VARTAB)
        LD (SAVEND),HL  ; sonst letzte Adresse fuer Programm-
        LD HL,(TXTTAB)  ;       speicherung
CSBSAV: 1E3E
        CALL CASBNH     ; Speichern des Bereichs
```



```
        POP HL
        RET
CBSAVE: 1E43            ; ************************ BSAVE "CAS:
        LD  A,ODOH      ; Vorspann  DO fuer Binaerprogramm
        CALL CASOPW     ; 1FA7 /Dateikennsatz (Typ, etc.) schreiben
        CALL CWRTON     ; 2069
        POP HL
        PUSH HL
        CALL L1E6E      ; Anfangsadresse
        LD HL,(SAVEND)  ; F658
        PUSH HL
        CALL L1E6E      ; Endadresse
        LD HL,(SAVENT)  ; F658
        CALL L1E6E      ; Startadresse
        POP DE
        POP HL
L1E5F:                  ; Bereich Byteweise speichern
        LD A,(HL)
        CALL CASOUT     ; 2026
        RST 20H
        JR NC,L1E69
        INC HL
        JR L1E5F
L1E69:
        CALL CTWOFF     ; 206C
        POP HL
        RET
L1E6E:                  ; Speichern von HL auf Cassette
        LD A,L
        CALL CASOUT     ; 2026 /
        LD A,H                  } A -> CAS
        JP CASOUT       ; 2026 /
L1E76:                  ; Lesen eines Wortes von Casette nach HL
        CALL CASIN      ; 2016
        LD L,A
        CALL CASIN      ; 2016
        LD H,A
        RET
CBLOAD: 1E7F            ; ************************ BLOAD "CAS:
        LD  C,ODOH
        CALL SRCCAS     ; Binaerprogramm suchen /1F34
        CALL CSROON     ; 203A
        POP BC
        CALL L1E76      ; Anfangsadresse / Wort einlesen
        ADD HL,BC
        EX DE,HL
        CALL L1E76      ; Endadresse / Wort einlesen
        ADD HL,BC
        PUSH HL
        CALL L1E76      ; Startadresse / Wort einlesen
        LD (SAVENT),HL
        EX DE,HL
        POP DE
L1E9A:                  ; Byteweise Bereich einlesen
        CALL CASIN      ; 2016
        LD (HL),A
        RST 20H
        JR Z,L1EA4
        INC HL
        JR L1E9A
L1EA4:
```

```
        CALL CTOFF
        JP CHKBRN
CLOAD:  15AA            ; ***********************************
        SUB 091H        ; CLOAD?
        JR Z,L1EBO
        XOR A
        DEFB 1                  ;*      LD  BC,0232FH
L1EBO:
        CPL                     ;*
        INC HL                  ;*
        CP 01H          ; -1 bei ?   0 ohne
        PUSH AF
        CALL L1F08      ; Filename
        LD (TXPSAV),HL
        LD  C,0D3H      ; Programm oder SCREEN Datei suchen
        CALL SRCCAS     ; Datei suchen /1F39
        LD  A,(CASATR)  ; Attribut der gelesenen Datei
        AND A
        JP M,L1ECE      ; JP falls Programm
        POP AF
        JP C,LODSCN     ; Schirm laden
        JP VRFSCN       ; Schirm testen
L1ECE:  Programm laden
        POP AF
        LD (FACLOW),A   ; F925
        CALL C,SCRTCH   ; Programm loeschen, falls vorhanden
        LD  A,(FACLOW)  ; F925
        CP 01H
        LD (FRCNEW),A   ; FD4B / Flag ums Speicher zu initialisieren
        PUSH AF
        CALL DEPTR      ; 1E0D
        POP AF
        LD HL,(TXTTAB)  ; F54A / Erste Programmzeile
        CALL CASBNR     ; Einlesen
        JR NZ,L1EFA
        LD (VARTAB),HL  ; F7EE / zeigt auf Speicher für einfache Variable
L1EED:
        LD  HL,REDDY    ; 89F / OK
        CALL STROUT     ; Drucken
        LD HL,(TXTTAB)  ; Zeiger biegen
        PUSH HL
        JP FINI         ; Pointer erneuern
L1EFA:
        INC HL
        EX DE,HL
        LD HL,(VARTAB)  ; speicher für variable suche
        RST 20H
        JP C,L1EED
        LD  E,014H
        JP ERROR        ; Verify error
L1F08:                  ; ******* Filename aus Programmzeile
        DEC HL          ; nach 0F99E-0F9A3 speichern
        RST 10H
        JR NZ,L1F14
        PUSH HL
        LD  HL,FILNAM
        LD B,06H
        JR L1F2D
L1F14:
        CALL FRMEVL     ; Ausdruck berechnen
        PUSH HL
```



```
        CALL ASC2       ; Stringpointer holen
        DEC HL
        DEC HL
        LD B,(HL)
        LD  C,06H
        LD  HL,FILNAM
L1F23:
        LD  A,(DE)
        LD (HL),A
        INC HL
        INC DE
        DEC C
        JR Z,L1F32      ; JP falls 6 Zeichen erreicht
        DJNZ L1F23      ; JP falls Stringlaenge noch nicht
        LD B,C          ;       erschoepft
L1F2D:
        LD (HL),' '     ; mit Blanks fuellen
        INC HL
        DJNZ L1F2D
L1F32:
        POP HL
        RET
SRCCAS: 1F34            ; ****** Anfang einer Cassetten Datei
        CALL CSROON     ;        suchen
        LD B,0AH        ; Dateityp in C
L1F39:
        CALL CASIN      ; 2016
        CP C
        JR NZ,SRCCAS
        DJNZ L1F39      ; 10 mal Dateityp
        LD  HL,FILNM2
        PUSH HL
        LD B,06H
L1F47:                  ; Dateinamen einlesen
        CALL CASIN
        LD (HL),A
        INC HL
        DJNZ L1F47
        CALL CASIN      ; Datei-Attribut einlesen
        LD (CASATR),A
        POP HL
        LD  DE,FILNAM
        LD B,06H
L1F5A:
        LD  A,(DE)      ; Skip Blanks im vorhandenen Dateinamen
        INC DE
        CP ' '
        JR NZ,L1F64
        DJNZ L1F5A
        JR L1F71
L1F64:
        LD  DE,FILNAM   ; Filename
        LD B,06H
L1F69:
        LD  A,(DE)
        CP (HL)
        JR NZ,L1F77     ; JP falls Dateinamen ungleich
        INC HL
        INC DE
        DJNZ L1F69
L1F71:                  ; found Datei
```

```
        LD  HL,L1F81
        JP L1F8F
L1F77:                  ; skip Datei
        PUSH BC
        LD  HL,L1F88
        CALL L1F8F
        POP BC
        JR SRCCAS

L1F81:
        DEFB 'Found:',0         ; 1F81
L1F88:
        DEFB :Skip :',0         ; 1F88

L1F8F:
        LD DE,(CURLIN)  ; F548/aktuelle Zeilennummer (-1 im Direktmodus)
        INC DE
        LD A,D
        OR E
        RET NZ          ; ret falls Programm laeuft
        CALL STROUT     ; Drucken
        LD  HL,FILNM2   ; F9A7/?. Dateiname für NAME
        LD B,06H
L1F9F:
        LD A,(HL)
        INC HL
        RST 18H
        DJNZ L1F9F
        JP CRDO
CASOPW: 1FA7            ; ***** Casetten Dateikennsatz schreiben
        CALL CWRTON     ; Vorspann schreiben
        LD B,0AH
L1FAC:
        CALL CASOUT
        DJNZ L1FAC      ; 10 mal Dateityp
        LD B,06H
        LD  HL,FILNAM
L1FB6:                  ; Dateiname
        LD A,(HL)
        INC HL
        CALL CASOUT
        DJNZ L1FB6
        LD  A,(CASATR)
        CALL CASOUT     ; Datei-Attribut
        JP CTWOFF
CASBNH: 1FC6            ; ****** Binaerbereich auf Kasette
        PUSH HL         ;        speichern
        CALL DEPTR
        CALL CWRTON
        POP DE
        LD HL,(SAVEND)
L1FD1:
        LD  A,(DE)
        INC DE
        CALL CASOUT
        RST 20H
        JR NZ,L1FD1
        LD  L,07H
L1FDB:                  ; 7 mal 00
        CALL CASOUT
        DEC L
```



```
        JR NZ,L1FDB
        JP CTWOFF
CASBNR: 1FE9            ; ******* Binaerbereich von Kasette
        CALL CSROON     ;         lesen
        SBC A,A
        CPL
        LD D,A
        LD B,0AH
        CALL CASIN
        LD E,A
        LD  A,(FLBMEM)
        OR A
        JR Z,L1FFD
        LD A,H
        OR L
        JR NZ,GRPCCL
        JP OMERR
L1FFD:
        CALL REASON     ; 6537 / Check auf Overflow = Stackbereich
GRPCCL: 2000
        LD A,E
        SUB (HL)
        AND D
        JP NZ,CTOFF
```

```
GRPCCL:/2060
        LD A,E
        SUB (HL)
        AND D
        JP NZ,CTOFF
        LD (HL),E
        LD A,(HL)
        OR A
        INC HL
        JR NZ,L1FEA
        DJNZ L1FEC
        LD  BC,OFFFAH
        ADD HL,BC
        XOR A
        JP CTOFF
CASIN:/2076             ; ***** Test CAS bereit?
        PUSH DE         ;       falls ja,9 bit Lesen 8 nach A
        PUSH BC
        IN A,(098H)
        ADD A,A
        JP M,OFFDIO     ; Abschalten falls not ready
        CALL L21A9      ; nach A&D 8 Bit einlesen
        JR C,OFFDIO     ; abschalten & Fehler ?
        POP BC
        POP DE
        RET
CASOUT:/2026            ; ***** Test CAS bereit?
        PUSH DE         ;   falls ja,9 bit Schreiben 8 aus A
        PUSH BC
        LD B,A
        IN A,(098H)
        ADD A,A
        JP M,OFFDIO     ; abschalten und Fehler ?
        LD A,B
        PUSH AF
        CALL DATAW      ; 20E3 / Bit auf CAS schreiben
        JR C,OFFDIO     ; abschalten und Fehler ?
        POP AF
        POP BC          ; Daten holen
        POP DE
        RET
CSROON:/2034            ; ***** falls cas nicht eingeschaltet
        PUSH HL         ;       Press play ... und warten auf
        PUSH DE         ;       fileheader
        PUSH BC
        PUSH AF
        LD  HL,L209A    ;       Text
        CALL L2084      ; CAS bereit ?
        DI              ; keine Unterbrechung mehr
        CALL L210A      ; Dateivorspann synchronisieren
        JR NC,L2069     ; Ende falls kein Ctrl-Stop
OFFDIO:/2049A
        CALL CTOFF      ; Abschalten
DIOERR:/2050
        LD  A,(MONFLG)  ; FE72
        AND A
        JP NZ,MONERR    ; Kaltstart
        LD  E,013H      ; Device I/O error falls nicht im
        JP ERROR        ;   Monitor
CWRTON:/2059            ; wie CASROON jedoch zum Schreiben
        PUSH HL
```



```
        PUSH DE
        PUSH BC
        PUSH AF
        LD  HL,L20AF
        CALL L2084
        DI
        CALL L20CF
        JR C,OFFDIO     ; abschalten & Fehler ?
L2069:
        JP POPALL
CTWOFF:/206C
        PUSH BC
        PUSH AF
        XOR A           ; 0 mit 9 bit schreiben
        CALL CASOUT     ; Dateiende
        LD  BC,00H
L2075:
        DEC BC
        LD A,B
        OR C
        JR NZ,L2075     ; warten
        POP AF          ; Daten holen
        POP BC
CTOFF:/207C
        PUSH AF
        LD  A,09H       ; OUT (97H),9 Recorder aus
        OUT (097H),A    ; abschalten des Recorders
        POP AF
        EI              ; Ende der Routine
        RET
L2084:/Bereitsetzen (Unterprogramm) ; ***** Return falls cas bereit, / OUT (97H),8 REC ON
        LD  A,08H       ;       sonst Textausgabe
        OUT (097H),A    ;       und weiter auf Bereit oder
        IN A,(098H)     ;       ctrl-stop warten
        ADD A,A
        RET P           ; ret falls bereit , sonst Text drucken
        CALL STROUT     ; 6970
L208F:
        IN A,(098H)     ; Taste am Recorder gedrückt ?
        ADD A,A
        RET P           ; Ja : Return
        CALL BREAKX     ; CTRL-STOP ? / 2512
        JR NC,L208F     ; nein : warte (= NZ)
        JR OFFDIO       ; abschalten & Error ? / 204A

L209A:
        DEFB 'Press play on tape',13,10,0

L20AF:
        DEFB 'Press play and record on tape',13,10,0

L20CF:                  ; ***** Dateivorspann schreiben
        LD  BC,0190H    ;       190H * 55H zur,Synchronisation
L20D2:                  ;       1    * 7F
        LD  A,055H
        PUSH BC
        CALL L20E6      ; 55H auf CAS schreiben
        POP BC
        RET C
        DEC BC
        LD A,B
```

```
        OR C
        JR NZ,L20D2
        LD  A,07FH          ; 7FH auf CAS schreiben
        JR L20E6

DATAW:/ 20E3                ; ***** A mit vorlaufendem 0 Bit zum
        CALL L20F6          ;   CAS
L20E6:
        LD B,08H            ; 8 Datenbits des A abspeichern
L20E8:
        CALL L20F0
        DJNZ L20E8
        JP BREAKX           ; Test Ctrl-Stop

L20F0:                      ;***** Bit 7 des A auf CAS speichern
        RLCA
        LD  DE,02428H       ; Zeitkonstante fuer Bitwert 1
        JR C,L20F9
L20F6:
        LD  DE,04D52H       ; Zeitkonstante fuer Bitwert 0
L20F9:
        DEC D
        JR NZ,L20F9         ; Vorlauf Wartezeit
        LD D,A
        LD  A,0BH           ;* Bit 5 port C   CASW setzen
        OUT (097H),A        ;*
L2101:
        DEC E
        JR NZ,L2101         ; Wartezeit fuer gesetztes Bit
        LD  A,0AH           ;* bit 5 zuruecksetzen
        OUT (097H),A        ;*
        LD A,D              ; A restaurieren
        RET


L210A:                      ; ***** synchronisieren auf Datei-
        LD B,064H           ;       vorspann  55H und 7FH
L210C:
        CALL L2196          ; Laenge der 01
        RET C
        CALL L215C          ; Bereich testen
        JR C,L210A          ; falls nicht im Bereich
        DJNZ L210C
L2117:
        LD  HL,00H
        LD B,064H
L211C:
        CALL L2196          ; Laenge der 01
        RET C
        CALL L215C          ; Bereich
        JR C,L2117
        LD D,C              ; Sichern der Halbwellenlaenge
        CALL L2196
        RET C
        CALL L215C
        JR C,L2117
        LD A,D
        SUB C               ; differenz der Sequenzen
        JR NC,L2135         ; JP falls positiv
        CPL                 ;* negieren
        INC A               ;*
```



```
L2135:
        CP 0EH
        JR C,L213B          ; JP falls 0 Bit zuerst
        INC H
        DEFB 03EH           ;*      LD  A,02CH
L213B:
        INC L               ;*
        DJNZ L211C
        LD  A,064H
        CP L
        JR Z,L2146          ; Polaritaet = 64H
        SUB H
        JR NZ,L2117         ; Polaritaet = 0
L2146:
        LD (POLRTY),A       ; 64H oder 0 je nach Polaritaet
        LD D,00H
L214B:                      ;***** warten auf 7FH oder Ctrl-Stop
        CALL L2164          ; bit  lesen
        RET C
        CALL L215C
        JR C,L2117
        CALL L21A3          ; Bit shiften
        CP 07FH
        JR NZ,L214B
        RET
L215C:                      ;*****  Bereich der Halbwelle testen
        LD A,C              ;       NC falls 18H<= C <=50H
        CP 18H
        RET C
        CP 051H
        CCF
        RET
L2164:                      ;****** Bit von Kassette lesen nach
        LD  C,00H           ;       Polaritaet
        CALL BREAKX
        RET C
        LD  A,(POLRTY)
        AND A
        JR NZ,L2181
L2170:                      ; erst warten auf 1
        INC C
        JR Z,L2193          ; JP falls keine 1 bis C auf 0
        IN A,(098H)
        RLCA
        JR NC,L2170
L2178:                      ; dann warten auf 0
        INC C
        JR Z,L2193
        IN A,(098H)
        RLCA
        JR C,L2178
        RET
L2181:                      ; erst 0
        INC C
        JR Z,L2193
        IN A,(098H)
        RLCA
        JR C,L2181
L2189:                      ; dann 1
        INC C
        JR Z,L2193
```

```
        IN A,(098H)
        RLCA
        JR NC,L2189
        AND A           ; NC fuer kein Ctrl-Stop
        RET
L2193:
        DEC C           ; C danach OFFH
        AND A           ; NC
        RET
L2196:                  ; ***** warten auf Sequenz 1]01[0
        CALL BREAKX     ;       Laenge der          .. in C
        RET C
        IN A,(098H)     ; Port lesen
        RLCA            ; bit nach Carry
        JR C,L2196      ; JP falls 1
        LD  C,00H       ; 0 gefunden!
        JR L2170
L21A3:                  ; ***** Bitfrequenz testen und Bit von
        CP 02EH         ;       rechts in D hineinschieben
        LD A,D          ;       Ergebnis auch in A
        RLA
        LD D,A
        RET
L21A9:                  ; ***** Byte als die rechten 8 von 9
        CALL L2164      ; Bit lesen Ergebnis in A und D
        RET C
        LD B,08H
L21AF:
        CALL L2164
        RET C
        CALL L215C
        JP C,0FFD10
        CALL L21A3
        DJNZ L21AF
        AND A
        RET



        ;************* Eingang fuer Makrosprache
        ;       DE Adresse der Sprachtabelle

MACLNG: 21C0
        LD (MCLTAB),DE
        CALL FRMEVL     ; Stringausdruck berechnen
        PUSH HL
        LD  DE,00H
        PUSH DE
        PUSH AF
L21CD:
        CALL FRESTR     ; evtl. Speicher freigeben
        CALL MOVRM      ; String-Parameter holen
        LD B,C
        LD C,D
        LD D,E
        LD A,B
        OR C
        JR Z,L21E0
        LD A,D
        OR A
        JR Z,L21E0
```



```
        PUSH BC
        PUSH DE         ; BC Adresse des Strings D=A Laenge
L21E0:
        POP AF
        LD (MCLLEN),A
        POP HL
        LD A,H
        OR L
        JR NZ,L21F3
        LD  A,(MCLFLG)
        OR A
        JP Z,L225D
        JP MCLEND
L21F3:
        LD (MCLPTR),HL  ; Adresse des Strings
MCLSCN: 21F6
        CALL FETCHR     ; Zeichen aus String
        JR Z,L21E0      ; JP falls Stringende
        ADD A,A         ; mal 2
        LD C,A          ; nach C sichern
        LD HL,(MCLTAB)
L2200:
        LD A,(HL)       ; Zeichen aus Tabelle
        ADD A,A
L2202:
        CALL Z,FCERR    ; falls Tabellenende
        CP C
        JR Z,L220D      ; JP falls Zeichen identisch
        INC HL
        INC HL
        INC HL
        JR L2200        ; naechster Tabelleneintrag
L220D:
        LD  BC,MCLSCN
        PUSH BC
        LD A,(HL)
        LD C,A
        ADD A,A
        JR NC,L2236     ; JP falls Tabellencode < 80H
        OR A            ; C loeschen
        RRA
        LD C,A          ; Tabellencode AND 7FH
        PUSH BC
        PUSH HL         ; Stringadresse
        CALL FETCHR
        LD  DE,01H
        JP Z,L2233      ; falls Stringende
        CALL ISLET2
        JP NC,L2230
        CALL L2270
        SCF
        JR L2234
L2230:
        CALL DECFET
L2233:
        OR A
L2234:
        POP HL
        POP BC
L2236:
        INC HL          ; Ausfuehrungsadresse holen
```

```
        LD A,(HL)       ;
        INC HL
        LD H,(HL)
        LD L,A
        JP (HL)
FETCHZ: 223C
        CALL FETCHR
        JR Z,L2202      ; Fehlereinsprung
        RET
FETCHR: 2242            ;***** Zeichen aus MCL string holen
        PUSH HL         ;  skip blanks wandeln von Kleinbuch-
L2243:                  ;  staben vorsicht bei Zeichen > z
        LD  HL,MCLLEN   ;   Z   falls String zu Ende
        LD A,(HL)
        OR A
        JR Z,L225D
        DEC (HL)
        LD HL,(MCLPTR)
        LD A,(HL)
        INC HL
        LD (MCLPTR),HL
        CP 020H
        JR Z,L2243
        CP 060H
        JR C,L225D
        SUB 020H
L225D:
        POP HL
        RET
DECFET: 225F            ; Zuruecksetzen des MCL Zeichen-Pointers
        PUSH HL         ; um ein Zeichen
        LD  HL,MCLLEN
        INC (HL)
        LD HL,(MCLPTR)
        DEC HL
        LD (MCLPTR),HL
        POP HL
        RET


VALSCN: 226C            ; Wert berechnen
        CALL FETCHZ
L2270:
        CP '='
        JP Z,VARGET
        CP ';'
        JR Z,VALSCN
VALSC2: 2279
        LD  DE,00H
L227C:
        CP ','
        JR Z,DECFET
        CP '='
        RET Z
        CP 03AH
        JR NC,DECFET
        CP 030H
        JR C,DECFET
        LD  HL,00H      ; HL:=10*DE
        LD B,0AH
L2290:
```



```
        ADD HL,DE
        JR C,L22BD
        DJNZ L2290
        SUB 030H
        LD E,A
        LD D,00H
        ADD HL,DE       ; HL:=HL+aktuelle Stelle
        JR C,L22BD      ; zu gross,
        EX DE,HL        ; sichern,
        CALL FETCHR
        JR NZ,L227C
        RET
L22A4:                  ; ***** move MCL-Einheit einschliesslich
        CALL FETCHZ     ;       ; nach BUF
        LD  DE,BUF
        PUSH DE
        LD B,028H
        CALL ISLET2
        JR C,L22BD      ; JP falls kein Grossbuchstabe
L22B2:
        LD (DE),A
        INC DE
        CP ';'
        JR Z,L22C0
        CALL FETCHZ
        DJNZ L22B2
L22BD:
        CALL FCERR
L22C0:
        POP HL
        JP ISVAR        ; Wert berechnen


VARGET: 22C4            ; Variablenwert holen
        CALL L22A4
        CALL FRCINT
        EX DE,HL
        RET

MCLXEQ: 22CC            ; ***** MCL X befehl
        CALL L22A4
        LD  A,(MCLLEN)
        LD HL,(MCLPTR)
        EX (SP),HL
        PUSH AF
        LD  C,02H
        CALL GETSTK
        JP L21CD

NEGD/SCAN1: 22DF        ; ***** Grafik SCAN  [-]|[[STEP] (p1,p2)]
        LD A,(HL)
        CP 040H
        CALL Z,CHRGTR   ; @ Ueberspringen
        LD  BC,00H      ; (0,0) Voreinstellung
        LD D,B
        LD E,C
        CP 0F4H         ; - Token
        JR Z,L2304
SCAND:  22EE            ; Grafik SCAN [STEP] (p1,p2)
        LD A,(HL)
        CP 0DCH         ; STEP TOKEN
```

```
        PUSH AF
        CALL Z,CHRGTR
        RST 8H
        DEFB '('
        CALL GETIN2
        PUSH DE
        RST 8H
        DEFB ','
        CALL GETIN2
        RST 8H
        DEFB ')'
        POP BC          ; 1. Par in BC   2. Par in DE
        POP AF
L2304:
        PUSH HL
        LD HL,(GRPACX)
        JR Z,L230D      ; JP falls STEP oder -
        LD  HL,00H
L230D:
        ADD HL,BC
        LD (GRPACX),HL  ; 1. Par speichern
        LD (GXPOS),HL
        LD B,H
        LD C,L
        LD HL,(GRPACY)
        JR Z,L231E
        LD  HL,00H
L231E:
        ADD HL,DE
        LD (GRPACY),HL  ; 2. Par speichern
        LD (GYPOS),HL
        EX DE,HL
        POP HL
        RET

PRESET: 2328 ;****************************************
        LD  A,(back/COLOR)
        JR L2330
PSET:   232D ;****************************************
        LD  A,(front/COLOR)
L2330:
        PUSH AF
        CALL SCAND      ; Koordinaten
        POP AF
        CALL L2393
        PUSH HL
        CALL SCALXY     ; Skalieren
        JR NC,L2344
        CALL MAPXYC     ; Umsetzen in CHRs
        CALL SETC       ; CHR Setzen
L2344:
        POP HL
        RET

POINT:  2346  ; ****************************************
        RST 10H
        PUSH HL
        CALL FETCHC
        POP DE
        PUSH HL
```



```
        PUSH AF
        LD HL,(GYPOS)
        PUSH HL
        LD HL,(GXPOS)
        PUSH HL
        LD HL,(GRPACY)
        PUSH HL
        LD HL,(GRPACX)
        PUSH HL         ; sichern der alten Koordinaten
        EX DE,HL
        CALL SCAND      ; Koordinaten
        PUSH HL
        CALL SCALXY
        LD  HL,0FFFFH
        JR NC,L2374
        CALL MAPXYC     ; Pixel in CHR Koordinaten
        CALL READC      ; CHR lesen
        LD L,A
        LD H,00H
L2374:
        CALL MAKINT
        POP DE
        POP HL
        LD (GRPACX),HL
        POP HL
        LD (GRPACY),HL
        POP HL
        LD (GXPOS),HL
        POP HL
        LD (GYPOS),HL   ; Restaurieren der alten Koordinaten
        POP AF
        POP HL
        PUSH DE
        CALL STOREC
        POP HL
        RET
ATRSCN: 2390            ; ***** Auswertung Farbe aus Statement
        LD  A,(front/COLOR)     ; Voreinstellung
L2393:
        PUSH BC
        PUSH DE
        LD E,A
        DEC HL
        RST 10H
        JR Z,L23A3      ; falls STmt Ende
        RST 8H
        DEFB ','        ; sonst ,
        CP ','          ; falls Farbe nicht angegeben
        JR Z,L23A3
        CALL GETBYT     ; Farbe holen
L23A3:
        LD A,E
        PUSH HL
        CALL SETATR     ; setzen
        JP C,FCERR
        POP HL
        POP DE
        POP BC
        JP CHRGT2
```

```
XDELT:/ 2381          ; ***** Berechnung Delta X
        LD HL,(GXPOS)
        LD A,L
        SUB C
        LD L,A
        LD A,H
        SBC A,B
        LD H,A        ; HL:=HL-BC
L23BA:
        RET NC
NEGHL:
        XOR A
        SUB L
        LD L,A
        SBC A,H
        SUB L
        LD H,A        ; HL:=0-HL
        SCF
        RET

YDELT:/ 23C3          ; Berechnung Delta Y
        LD HL,(GYPOS)
        LD A,L
        SUB E
        LD L,A
        LD A,H
        SBC A,D
        LD H,A
        JR L23BA

XCHGY:/ 23CE          ; Tauschen von X und Y Koordinate
        PUSH HL       ;   DE gegen (GYPOS)
        LD HL,(GYPOS)
        EX DE,HL
        LD (GYPOS),HL
        POP HL
        RET
XCHGX:/ 23D8          ; Tauschen DE gegen (GYPOS)
        CALL XCHGY    ; und BC gegen (GXPOS)
L23DB:
        PUSH HL
        PUSH BC
        LD HL,(GXPOS)
        EX (SP),HL
        LD (GXPOS),HL
        POP BC
        POP HL
        RET


GLINE:/ 23E7          ; ************************ LINE ()-()
        CALL NEGD/SCAN1 ; erste Koordinaten
        PUSH BC       ; x
        PUSH DE       ; y
        RST 8H
        DEFB 0F4H     ; - Token
        CALL SCAND    ; zweite Koordinaten
        CALL ATRSCN   ; Farbe
        POP DE
        POP BC
        JR Z,L243C    ; ende des Stmts
```



```
        RST 8H
        DEFB ','
        RST 8H
        DEFB 'B'      ; Box Parameter
        JP Z,02452H
L23FF:
        RST 8H
        DEFB 'F'      ; Fill Parameter
        PUSH HL
        CALL SCALXY
        CALL XCHGX
        CALL SCALXY
        CALL YDELT
        CALL C,XCHGY
        INC HL
        PUSH HL
        CALL XDELT
        CALL C,L23DB
        INC HL
        PUSH HL
        CALL MAPXYC
        POP DE
        POP BC
L2420:
        PUSH DE
        PUSH BC
        CALL FETCHC
        PUSH AF
        PUSH HL
        EX DE,HL
        CALL NSETCX
        POP HL
        POP AF
        CALL STOREC
        CALL DOWNC
        POP BC
        POP DE
        DEC BC
        LD A,B
        OR C
        JR NZ,L2420
        POP HL
        RET
L243C:                ; **** LINE allein (BC,DE)-(GXPOS,GYPOS)
        PUSH BC       ; start x
        PUSH DE       ; start y
        PUSH HL
        CALL DOGRPH   ; 247C
        LD HL,(GRPACX) ; F846
        LD (GXPOS),HL ; end x
        LD HL,(GRPACY) ; F848
        LD (GYPOS),HL ; end y
        POP HL
        POP DE
        POP BC
        RET
L2452:                ; ***** LINE BOX ohne fill
        PUSH HL
        LD HL,(GYPOS)
        PUSH HL
        PUSH DE
```

```
        EX DE,HL
        CALL L243C      ; line (x1,y2)-(x2,y2)
        POP HL          ; y1
        LD (GYPOS),HL   ; FE44
        EX DE,HL
        CALL L243C      ; line (x1,y1)-(x2,y1)
        POP HL          ; y2
        LD (GYPOS),HL   ; FE44
        LD HL,(GXPOS)   ; FE42
        PUSH BC
        LD B,H
        LD C,L
        CALL L243C      ; line (x2,y1)-(x2,y2)
        POP HL          ; x1
        LD (GXPOS),HL   ; FE42
        LD B,H
        LD C,L
        CALL L243C      ; line (x1,y1)-(x1,y2)
        POP HL
        RET
DOGRPH: 247C            ; ******* Ziehen der Linie
        CALL LFEC4
        CALL SCALXY     ; 48A1 / Bereichscheck
        CALL XCHGX      ; 2308 / Inhalt GXPOS & GYPOS gege BC,DE
        CALL SCALXY     ; 48A1 / Bereichscheck
DOGRP2: 2488
        CALL YDELT      ; 23C3
        CALL C,XCHGX    ; C wenn ausserhalb des Bereichs
        PUSH DE
        PUSH HL
        CALL XDELT      ; 23B1
        EX DE,HL
        LD  HL,RIGHTC
        JR NC,L249C
        LD  HL,LEFTC
L249C:
        EX (SP),HL
        RST 20H
        JR NC,L24BO
        LD (MINDEL),HL
        POP HL
        LD (OFAOEH),HL
        LD  HL,DOWNC
        LD (OFA11H),HL
        EX DE,HL
        JR L24BF
L24BO:
        EX (SP),HL
        LD (OFA11H),HL
        LD  HL,DOWNC
        LD (OFAOEH),HL
        EX DE,HL
        LD (MINDEL),HL
        POP HL
L24BF:
        POP DE
        PUSH HL
        CALL NEGHL
        LD (MAXDEL),HL
        CALL MAPXYC
        POP DE
```



```
        PUSH DE
        CALL HLFDE
        POP BC
        INC BC
        JR L24DA
L24D3:
        POP HL
        LD A,B
        OR C
        RET Z
L24D7:
        CALL MAXUPD
L24DA:
        CALL SETC
        DEC BC
        PUSH HL
        LD HL,(MINDEL)
        ADD HL,DE
        EX DE,HL
        LD HL,(MAXDEL)
        ADD HL,DE
        JR NC,L24D3
        EX DE,HL
        POP HL
        LD A,B
        OR C
        RET Z
        CALL MINUPD
        JR L24D7
HLFDE:  24F4            ;*****  DE:=DE\2
        LD A,D
        OR A
        RRA
        LD D,A
        LD A,E
        RRA
        LD E,A
        RET
PAINT:  24FC            ; **********************************
        CALL NEGD/SCAN1 ; Koordinaten / 220F
        PUSH BC
        PUSH DE
        CALL ATRSCN     ; Farbe /
        CALL CHKMOD     ; Grafik ?
        JR NZ,L250E     ; JP falls nicht SCREEN 1
        DEC HL
        RST 10H
        JP NZ,FCERR     ; SCREEN 1 dann kein weiterer Parameter
L250E:
        LD  A,(ATRBYT)  ; FA13
        LD E,A
        DEC HL
        RST 10H
        JR Z,L251B
        RST 8H
        DEFB ',',
        CALL GETBYT
L251B:
        LD A,E
        CALL PNTINI     ; Randfarbe setzen
        JP C,FCERR      ; AS
```

```
        POP DE
        POP BC
        PUSH HL
        CALL L2B3C
        CALL MAPXYC      ; CHR-Position ausrechnen
        LD  DE,01H
        LD B,00H
        CALL L261D
        JR Z,L2549
        PUSH HL
        CALL L262E
        POP DE
        ADD HL,DE
        EX DE,HL
        XOR A
        CALL L260F
        LD  A,040H
        CALL L260F
        LD B,0C0H
        JR L2567
L2549:
        POP HL
        RET
L254B:
        CALL CKCNTC
        LD  A,(LOHDIR)
        OR A
        JR Z,L2560
        LD HL,(LOHADR)
        PUSH HL
        LD HL,(LOHMSK)
        PUSH HL
        LD HL,(LOHCNT)
        PUSH HL
L2560:
        POP DE
        POP BC
        POP HL
        LD A,C
        CALL STOREC
L2567:
        LD A,B
        LD (POIREC),A
        ADD A,A
        JR Z,L2549
        PUSH DE
        JR NC,L2576
        CALL TUPC
        JR L2579
L2576:
        CALL TDOWNC
L2579:
        POP DE
        JR C,L2560
        LD B,00H
        CALL L261D
        JP Z,L2560
        XOR A
        LD (LOHDIR),A
        CALL L262E
        LD E,L
```



```
        LD D,H
        OR A
        JR Z,L25AA
        DEC HL
        DEC HL
        LD A,H
        ADD A,A
        JR C,L25AA
        LD (LOHCNT),DE
        CALL FETCHC
        LD (LOHADR),HL
        LD (LOHMSK),A
        LD  A,(POIREC)
        CPL
        LD (LOHDIR),A
L25AA:
        LD HL,(MOVCNT)
        ADD HL,DE
        EX DE,HL
        CALL L2603
        LD HL,(CSAVE)
        LD  A,(CSAVEM)
        CALL STOREC
L25BB:
        LD HL,(SKPCNT)
        LD DE,(MOVCNT)
        OR A
        SBC HL,DE
        JR Z,L2600
        JR C,L25E5
        EX DE,HL
        LD B,01H
        CALL L261D
        JR Z,L2600
        OR A
        JR Z,L25BB
        EX DE,HL
        LD HL,(CSAVE)
        LD  A,(CSAVEM)
        LD C,A
        LD  A,(POIREC)
        LD B,A
        CALL L2614
        JR L25BB
L25E5:
        CALL NEGHL
        DEC HL
        DEC HL
        LD A,H
        ADD A,A
        JR C,L2600
        INC HL
        PUSH HL
L25FO:
        CALL LEFTC
        DEC HL
        LD A,H
        OR L
        JR NZ,L25FO
        POP DE
        LD  A,(POIREC)
```

```
        CPL
        CALL L260F
L2600:
        JP L254B
L2603:
        LD  A,(LFPROG)
        LD C,A
        LD  A,(RTPROG)
        OR C
        RET Z
        LD  A,(POIREC)
L260F:
        LD B,A
        CALL FETCHC
        LD C,A
L2614:
        EX (SP),HL
        PUSH BC
        PUSH DE
        PUSH HL
        LD  C,02H
        JP GETSTK
L261D:
        CALL SCANR
        LD (SKPCNT),DE
        LD (MOVCNT),HL
        LD A,H
        OR L
        LD A,C
        LD (RTPROG),A
        RET
L262E:
        CALL FETCHC
        PUSH HL
        PUSH AF
        LD HL,(CSAVE)
        LD  A,(CSAVEM)
        CALL STOREC
        POP AF
        POP HL
        LD (CSAVE),HL
        LD (CSAVEM),A
        CALL SCANL
        LD A,C
        LD (LFPROG),A
        RET
NEGDE:  2640            ; **********    DE:=0-DE
        EX DE,HL
        CALL NEGHL
        EX DE,HL
        RET
CIRCLE: 2652            ; ************************************
        CALL NEGD/SCAN1
        RST 8H
        DEFB ','
        CALL GETIN2
        PUSH HL
        EX DE,HL
        LD (GXPOS),HL   ; Radius
        CALL MAKINT
        CALL FRCSNG
```



```
        LD  BC,07040H
        LD  DE,0771H
        CALL FMULT
        CALL FRCINT
        LD (CNPNTS),HL
        XOR A
        LD (CLINEF),A
        LD (CSCLX),A
        POP HL
        CALL ATRSCN     ; Farbe
        LD  C,01H
        LD  DE,00H
        CALL L284F      ; start Kreisbogen
        PUSH DE
        LD  C,080H
        LD  DE,0FFFFH
        CALL L284F      ; ende Kreisbogen
        EX (SP),HL
        XOR A
        EX DE,HL
        RST 20H
        LD  A,00H
        JR NC,L26A7
        DEC A
        EX DE,HL
        PUSH AF
        LD  A,(CLINEF)
        LD C,A
        RLCA
        RLCA
        OR C
        RRCA
        LD (CLINEF),A
        POP AF
L26A7:
        LD (CPLDTF),A
        LD (CSTCN),DE
        LD (CENCNT),HL
        POP HL
        DEC HL
        RST 10H
        JR NZ,L26C6
        PUSH HL
        CALL GTASPC     ; Seitenverhaeltnis
        LD A,H
        OR A
        JR Z,L26E7
        LD  A,01H
        LD (CSCLX),A
        EX DE,HL
        JR L26E7
L26C6:
        RST 8H
        DEFB ','
        CALL FRMEVL
        PUSH HL
        CALL FRCSNG
        CALL L28A0
        JR NZ,L26DB
        INC A
        LD (CSCLX),A
```

```
        CALL FDIV
L26DB:
        LD  BC,02543H
        LD  DE,060H
        CALL FMULT
        CALL FRCINT
L26E7:
        LD (ASPECT),HL
        LD  DE,00H
        LD (CRCSU),DE
        LD HL,(GXPOS)
        ADD HL,HL
L26F5:
        CALL CKCNTC
        LD A,E
        RRA
        JR C,L2712
        PUSH DE
        PUSH HL
        INC HL
        EX DE,HL
        CALL HLFDE
        EX DE,HL
        INC DE
        CALL HLFDE
        CALL L273E
        POP DE
        POP HL
        RST 20H
        JP NC,L2549
        EX DE,HL
L2712:
        LD B,H
        LD C,L
        LD HL,(CRCSU)
        INC HL
        ADD HL,DE
        ADD HL,DE
        LD A,H
        ADD A,A
        JR C,L272A
        PUSH DE
        EX DE,HL
        LD H,B
        LD L,C
        ADD HL,HL
        DEC HL
        EX DE,HL
        OR A
        SBC HL,DE
        DEC BC
        POP DE
L272A:
        LD (CRCSU),HL
        LD H,B
        LD L,C
        INC DE
        JR L26F5
L2732:
        PUSH DE
        CALL L2823
```



```
        POP HL
        LD  A,(CSCLX)
        OR A
        RET Z
        EX DE,HL
        RET
L273E:
        LD (CPCND),DE
        PUSH HL
        LD  HL,00H
        LD (CPCNT),HL
        CALL L2732
        LD (CXOFF),HL
        POP HL
        EX DE,HL
        PUSH HL
        CALL L2732
        LD (CYOFF),DE
        POP DE
        CALL NEGDE
        CALL L2780
        PUSH HL
        PUSH DE
        LD HL,(CNPNTS)
        LD (CPCNT),HL
        LD DE,(CPCND)
        OR A
        SBC HL,DE
        LD (CPCND),HL
        LD HL,(CXOFF)
        CALL NEGHL
        LD (CXOFF),HL
        POP DE
        POP HL
        CALL NEGDE
L2780:
        LD  A,04H
L2782:
        PUSH AF
        PUSH HL
        PUSH DE
        PUSH HL
        PUSH DE
        LD DE,(CPCNT)
        LD HL,(CNPNTS)
        ADD HL,HL
        ADD HL,DE
        LD (CPCNT),HL
        LD HL,(CPCND)
        ADD HL,DE
        EX DE,HL
        LD HL,(CSTCN)
        RST 20H
        JR Z,L27B8
        JR NC,L27A8
        LD HL,(CENCNT)
        RST 20H
        JR Z,L27B0
        JR NC,L27C8
L27A8:
        LD  A,(CPLDTF)
```

```
        OR A
        JR NZ,L27D2
        JR L27CE
L27B0:
        LD  A,(CLINEF)
        ADD A,A
        JR NC,L27D2
        JR L27BE
L27B8:
        LD  A,(CLINEF)
        RRA
        JR NC,L27D2
L27BE:
        POP DE
        POP HL
        CALL L2814
        CALL L2805
        JR L27E2
L27C8:
        LD  A,(CPLDTF)
        OR A
        JR Z,L27D2
L27CE:
        POP DE
        POP HL
        JR L27E2
L27D2:
        POP DE
        POP HL
        CALL L2814
        CALL SCALXY
        JR NC,L27E2
        CALL MAPXYC
        CALL SETC
L27E2:
        POP DE
        POP HL
        POP AF
        DEC A
        RET Z
        PUSH AF
        PUSH DE
        LD DE,(CXOFF)
        CALL NEGDE
        LD (CXOFF),HL
        EX DE,HL
        POP DE
        PUSH HL
        LD HL,(CYOFF)
        EX DE,HL
        LD (CYOFF),HL
        CALL NEGDE
        POP HL
        POP AF
        JP L2782
L2805:
        LD HL,(GRPACX)
        LD (GXPOS),HL
        LD HL,(GRPACY)
        LD (GYPOS),HL
        JP DOGRPH
```



```
L2814:
        PUSH DE
        LD DE,(GRPACX)
        ADD HL,DE
        LD B,H
        LD C,L
        POP DE
        LD HL,(GRPACY)
        ADD HL,DE
        EX DE,HL
        RET
L2823:
        LD HL,(ASPECT)
        LD A,L
L2827:
        OR A
        JR NZ,L282E
        OR H
        RET NZ
        EX DE,HL
        RET
L282E:
        LD C,D
        LD D,00H
        PUSH AF
        CALL L2842
        LD  E,080H
        ADD HL,DE
        LD E,C
        LD C,H
        POP AF
        CALL L2842
        LD E,C
        ADD HL,DE
        EX DE,HL
        RET
L2842:
        LD B,08H
        LD  HL,00H
L2847:
        ADD HL,HL
        ADD A,A
        JR NC,L284C
        ADD HL,DE
L284C:
        DJNZ L2847
        RET
L284F:                  ; **** Kreisbogenteile fuer CIRCLE holen
        DEC HL
        RST 10H
        RET Z
        RST 8H
        DEFB ','
        CP ','
        RET Z
        PUSH BC
        CALL FRMEVL
        EX (SP),HL
        PUSH HL
        CALL FRCSNG
        POP BC
```

```
        LD  HL,FACCU
        LD A,(HL)
        OR A
        JR Z,L2877
        DEC HL
        LD A,(HL)
        OR A
        JP P,L2877
        AND 07FH
        LD (HL),A
        LD  HL,CLINEF
        LD A,(HL)
        OR C
        LD (HL),A
L2877:
        LD  BC,01540H
        LD  DE,05591H
        CALL FMULT
        CALL L28AO
        JP Z,FCERR
        CALL PUSHF
        LD HL,(CNPNTS)
        ADD HL,HL
        ADD HL,HL
        ADD HL,HL
        CALL MAKINT
        CALL FRCSNG
        POP BC
        POP DE
        CALL FMULT
        CALL FRCINT
        POP DE
        EX DE,HL
        RET
L28AO:
        LD  BC,01041H
        LD  DE,00H
        CALL FCOMP
        DEC A
        RET

GPUTG:  28AB            ; ******************** Grafik PUT
        LD (PUTFLG),A
        PUSH AF
        CALL NEGD/SCAN1
        CALL L2B3C
        POP AF
        OR A
        JP NZ,L292F
        RST 8H
        DEFB 0F4H       ; - Token
        PUSH BC
        PUSH DE
        CALL SCAND
        CALL L2B3C
        POP DE
        POP BC
        PUSH HL
        CALL YDELT
        CALL C,XCHGY
```



```
        INC HL
        LD (MINDEL),HL
        CALL XDELT
        CALL C,L23DB
        INC HL
        LD (MAXDEL),HL
        CALL MAPXYC
        POP HL
        CALL L29B4
        PUSH HL
        PUSH DE
        PUSH BC
        PUSH DE
        CALL PIXSIZ
        LD DE,(MAXDEL)
        LD  HL,00H
L28F0:
        ADD HL,DE
        DEC A
        JR NZ,L28F0
        LD B,H
        LD C,L
        LD  DE,07H
        ADD HL,DE
        EX DE,HL
        CALL HLFDE
        CALL HLFDE
        CALL HLFDE
        LD HL,(MINDEL)
        PUSH BC
        LD B,H
        LD C,L
        CALL UMULT
        POP BC
        LD  HL,04H
        ADD HL,DE
        POP DE
        ADD HL,DE
        EX DE,HL
        POP HL
        RST 20H
        JP C,FCERR
        POP HL
        RST 20H
        JP NC,FCERR
        LD (HL),C
        INC HL
        LD (HL),B
        INC HL
        LD DE,(MINDEL)
        LD (HL),E
        INC HL
        LD (HL),D
        INC HL
        OR A
        JP L298B
L292F:
        PUSH HL
        CALL MAPXYC
        POP HL
        CALL L29B4
```

```
        PUSH DE
        DEC HL
        RST 10H
        LD B,05H
        JR Z,L294F
        RST 8H
        DEFB ','
        EX DE,HL
        LD  HL,L29D9    ; tabelle der moeglichen mischungen
L2944:                  ; FUER PUT
        CP (HL)
        JR Z,L294D
        DEC HL
        DJNZ L2944
        EX DE,HL
        POP DE
        RET
L294D:                  ; B=5   XOR
        EX DE,HL        ;   4   PSET
        RST 10H         ;   3   PRESET
L294F:                  ;   2   AND
        DEC B           ;   1   OR
        LD A,B
        EX (SP),HL
        PUSH AF
        LD E,(HL)
        INC HL
        LD D,(HL)
        INC HL
        PUSH DE
        PUSH HL
        EX DE,HL
        CALL PIXSIZ
        LD E,A
L295E:
        DEC E
        JP Z,L2969
        EX DE,HL
        CALL HLFDE
        EX DE,HL
        JR L295E
L2969:
        DEC HL
        LD DE,(GXPOS)
        ADD HL,DE
        JR C,L297F
        LD B,H
        LD C,L
        POP HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        INC HL
        PUSH DE
        PUSH HL
        LD HL,(GYPOS)
        DEC DE
        ADD HL,DE
L297F:
        JP C,FCERR
        EX DE,HL
```



```
        POP HL
        CALL L2B3C
        POP DE
        POP BC
        POP AF
        SCF
L298B:
        PUSH DE
        CALL PGINIT
        POP DE
L2990:
        PUSH DE
        CALL FETCHC
        PUSH HL
        PUSH AF
        LD  A,(PUTFLG)
        OR A
        JR NZ,L29A1
        CALL NREAD
        JR L29A4
L29A1:
        CALL NWRITE
L29A4:
        POP AF
        POP HL
        CALL STOREC
        CALL DOWNC
        POP DE
        DEC DE
        LD A,D
        OR E
        JR NZ,L2990
        POP HL
        RET
L29B4:
        RST 8H
        DEFB ','
        LD  A,01H
        LD (SUBFLG),A
        CALL PTRGET
        JP NZ,FCERR
        LD (SUBFLG),A
        PUSH HL
        LD H,B
        LD L,C
        EX DE,HL
        ADD HL,DE
        PUSH HL
        LD  A,(BC)
        ADD A,A
        LD L,A
        LD H,00H
        INC BC
        ADD HL,BC
        EX DE,HL
        POP BC
        POP HL
        RET

        DEFB 0F9H,0F8H,0C3H,0C2H
L29D9:
```

```
DRAW:   DEFB 0FAH
        29DA                ; ************************************
        LD  DE,L29E7
        XOR A
        LD (DRWFLG),A
        LD (MCLFLG),A
        JP MACLNG
L29E7:
                            ;*29E7
        DC 'U'
        DEFW L2A15
        DC 'D'
        DEFW L2A18
        DC 'L'
        DEFW L2A1D
        DC 'R'
        DEFW L2A20
        DEFB 'M'
        DEFW L2A3C
        DC 'E'
        DEFW L2A2E
        DC 'F'
        DEFW L2A2A
        DC 'G'
        DEFW L2A35
        DC 'H'
        DEFW L2A27
        DC 'A'
        DEFW L2AEA
        DEFB 'B'
        DEFW L2AE2
        DEFB 'N'
        DEFW L2ADE
        DEFB 'X'
        DEFW L22CC
        DC 'C'
        DEFW L2B32
        DC 'S'
        DEFW L2AF5
        DEFB 0

L2A15:                      ; ***** DRAW Up
        CALL NEGDE
L2A18:                      ; ***** DRAW Down
        LD BC,0
        JR L2A63
L2A1D:                      ; ***** DARW Left
        CALL NEGDE
L2A20:                      ; ***** DARW Right
        LD B,D
        LD C,E
        LD  DE,00H
        JR L2A63
L2A27:                      ; ***** DRAW H   U*L
        CALL NEGDE
L2A2A:                      ; ***** DRAW F   D*R
        LD B,D
        LD C,E
        JR L2A63
L2A2E:                      ; ***** DRAW E   U*R
        LD B,D
```



```
        LD C,E
L2A30:
        CALL NEGDE
        JR L2A63
L2A35:                      ; ***** DRAW G   D*L
        CALL NEGDE
        LD B,D
        LD C,E
        JR L2A30
L2A3C:                      ; ***** DRAW M
        CALL FETCHZ
        LD B,00H
        CP '+'
        JR Z,L2A4A
        CP '-'
        JR Z,L2A4A
        INC B
L2A4A:
        LD A,B
        PUSH AF
        CALL DECFET
        CALL VALSCN
        PUSH DE
        CALL FETCHZ
        CP ','
        JP NZ,FCERR
        CALL VALSCN
        POP BC
        POP AF
        OR A
        JR NZ,L2ABE
L2A63:
        CALL L2B02
        PUSH DE
        LD D,B
        LD E,C
        CALL L2B02
        EX DE,HL
        POP DE
        XOR A
        LD (CSCLX),A
        LD  A,(DRWANG)
        RRA
        JR NC,L2AB2     ; JP fuer alle geraden Winkel
        PUSH AF         ; Winkel 1 oder 3
        PUSH DE         ;= falscher Code
        PUSH HL         ;= kann ersetzt werden
        CALL GTASPC     ;= durch
        LD A,H          ;=      CALL NEGHL
        OR A            ;=      EX DE,HL
        JR Z,L2A87      ;=
        LD  A,01H       ;=
        LD (CSCLX),A    ;=
L2A87:                  ;=
        EX DE,HL        ;=
        LD C,L          ;=
        POP HL          ;=
        LD  A,(CSCLX)   ;=
        OR A            ;=
        JR Z,L2A91      ;=
        EX (SP),HL      ;=
```

```
L2A91:                  ;=
        EX DE,HL        ;=
        PUSH HL         ;=
        CALL L2B23      ;=
        POP BC          ;=
        POP HL          ;=
        PUSH DE         ;=
        EX DE,HL        ;=
        LD  HL,00H      ;=
L2A9D:                  ;=
        ADD HL,DE       ;=
        DJNZ L2A9D      ;=
        PUSH HL         ;=
        CALL L2B23      ;=
        POP HL          ;=
        ADD HL,DE       ;=
        POP DE          ;=
        LD  A,(CSCLX)   ;=
        OR A            ;=
        JR Z,L2AAE      ;=
        EX DE,HL        ;=
L2AAE:                  ;=
        CALL NEGDE      ;=
        POP AF          ; von hier die ungeraden Winkel
L2AB2:                  ; **** Winkel 0 oder 2
        RRA
        JR NC,L2ABB
        CALL NEGHL      ; Winkel 0 oder 1
        CALL NEGDE
L2ABB:                  ; Winkel 2 oder 3
        CALL L2814
L2ABE:
        LD  A,(DRWFLG)
        ADD A,A
        JR C,L2ACD
        PUSH AF
        PUSH BC
        PUSH DE
        CALL L2805
        POP DE
        POP BC
        POP AF
L2ACD:
        ADD A,A
        JR C,L2AD9
        LD (GRPACY),DE
        LD H,B
        LD L,C
        LD (GRPACX),HL
L2AD9:
        XOR A
        LD (DRWFLG),A
        RET
L2ADE:                  ; ***** DRAW N
        LD  A,040H
        JR L2AE4
L2AE2:                  ; ***** DRAW B
        LD  A,080H
L2AE4:
        LD  HL,DRWFLG
        OR (HL)
```



```
        LD (HL),A
        RET
L2AEA:                  ; ***** DRAW A
        JR NC,L2AF5
        LD A,E
        CP 04H
        JR NC,L2AF5
        LD (DRWANG),A   ; Winkel im Bereich 0-3
        RET
L2AF5:                  ; ***** DRAW S
        JP NC,FCERR
        LD A,D
        OR A
        JP NZ,FCERR
        LD A,E
        LD (DRWSCL),A
        RET
L2B02:
        LD  A,(DRWSCL)
        OR A
        RET Z
        LD  HL,00H
L2B0A:
        ADD HL,DE
        DEC A
        JR NZ,L2B0A
        EX DE,HL
        LD A,D
        ADD A,A
        PUSH AF
        JR NC,L2B15
        DEC DE
L2B15:
        CALL HLFDE
        CALL HLFDE
        POP AF
        RET NC
        LD A,D
        OR 0C0H
        LD D,A
        INC DE
        RET
L2B23:
        LD A,D
        ADD A,A
        JR NC,L2B2E
        LD  HL,NEGD/SCAN1
        PUSH HL
        CALL NEGDE
L2B2E:
        LD A,C
        JP L2827
L2B32:                  ; ***** DRAW C
        JR NC,L2AF5
        LD A,E
        CALL SETATR
        JP C,FCERR
        RET
L2B3C:
        PUSH HL
        CALL SCALXY
```

```
        JP NC,FCERR
        POP HL
        RET
PUTQ:/  2B45            ; legt Byte in die Queue ab
        CALL L2BCF
        LD A,B
        INC A
        INC HL
        AND (HL)
        CP C
        RET Z
        PUSH HL
        DEC HL
        DEC HL
        DEC HL
        EX (SP),HL
        INC HL
        LD C,A
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        LD B,00H
        ADD HL,BC
        LD (HL),E
        POP HL
        LD (HL),C
        RET
GETQ:/  2B60            ; holt Byte aus der Queue
        CALL L2BCF
        LD (HL),00H
        JR NZ,L2B84
        LD A,C
        CP B
        RET Z
        INC HL
        INC A
        AND (HL)
        DEC HL
        DEC HL
        PUSH HL
        INC HL
        INC HL
        INC HL
        LD C,A
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        LD B,00H
        ADD HL,BC
        LD A,(HL)
        POP HL
        LD (HL),C
        OR A
        RET NZ
        INC A
        LD  A,00H
        RET
L2B84:
        LD C,A
```



```
        LD B,00H
        LD  HL,0FB09H
        ADD HL,BC
        LD A,(HL)
        RET
INITQ:/ 2B8D            ; initialisiert die Queue
        PUSH BC
        CALL L2BD9
        LD (HL),B
        INC HL
        LD (HL),B
        INC HL
        LD (HL),B
        INC HL
        POP AF
        LD (HL),A
        INC HL
        LD (HL),E
        INC HL
        LD (HL),D
        RET
BCXQ:/  2B9E
        PUSH AF
        CALL L2BCF
        POP AF
        INC A
        LD (HL),A
        LD C,A
        LD B,00H
        LD  HL,0FB09H
        ADD HL,BC
        LD (HL),E
        RET
NUMQ:/  2BAE
        CALL L2BCF
        SUB 01H
        SBC A,A
        INC A
        LD E,A
        LD A,B
        SUB C
        INC HL
        AND (HL)
        LD L,A
        LD H,00H
        LD D,H
        ADD HL,DE
        RET
LFTQ:/  2BC0            ; Anzahl der Zeichen in der Queue
        CALL L2BCF
        LD A,B
        INC A
        INC HL
        AND (HL)
        LD B,A
        LD A,C
        SUB B
        AND (HL)
        LD L,A
        LD H,00H
        RET
```

```
L2BCF:
        CALL L2BD9
        LD B,(HL)
        INC HL
        LD C,(HL)
        INC HL
        LD A,(HL)
        OR A
        RET
L2BD9:
        RLCA
        LD B,A
        RLCA
        ADD A,B
        LD C,A
        LD B,00H
        LD HL,(QUEUES)
        ADD HL,BC
        RET
MOTOR:/2BE5             ; ************************************
        CP 095H         ; ON
        JR Z,L2BF1
        CP 0EBH         ; OFF
        JP NZ,SNERR
        LD  A,09H
        DEFB 1          ;*      LD  BC,083EH
L2BF1:
        LD A,8          ;*
        DEFB 1          ;*      LD  BC,0D3EH
L2BF4:
        LD A,13         ;*
        DEFB 1          ;*      LD  BC,0C3EH
L2BF7:
        LD A,12         ;*
        OUT (097H),A
        RST 10H
        RET
SOUND:/2BFD             ; ************************************
        CP 095H         ; ON
        JR Z,L2BF4
        CP 0EBH         ; OFF
        JR Z,L2BF7
        CALL GETBYT
        CP 0EH          ; max-Registeradresse
L2C0A:
        JP NC,FCERR
        PUSH AF
        RST 8H
        DEFB ','
        CALL GETBYT
        POP AF
        CP 07H
        JR NZ,L2C20     ; JP falls nicht Register 7
        PUSH AF
        LD A,E
        AND 03FH
        OR 080H
        LD E,A
        POP AF
L2C20:
        JP L40B6
```



```
L2C23:
        DEFB 020H               ;*JR NZ,L2C0A
PLAY:/2C24              ; ************************************
        PUSH HL         ;*
        LD  HL,PLYTAB
        LD (MCLTAB),HL
        LD  A,00H
        LD (PRSCNT),A
        LD  HL,0FFF6H
        ADD HL,SP
        LD (SAVSP),HL
        POP HL
        PUSH AF
L2C39:
        CALL FRMEVL
        EX (SP),HL
        PUSH HL
        CALL FRESTR
        CALL MOVRM
        LD A,E
        OR A
        JR NZ,L2C4F
        LD  E,01H
        LD  BC,02C23H
        LD D,C
        LD C,B
L2C4F:
        POP AF
        PUSH AF
        CALL GETVCP
        LD (HL),E
        INC HL
        LD (HL),D
        INC HL
        LD (HL),C
        INC HL
        LD D,H
        LD E,L
        LD  BC,01CH
        ADD HL,BC
        EX DE,HL
        LD (HL),E
        INC HL
        LD (HL),D
        POP BC
        POP HL
        INC B
        LD A,B
        CP 03H
        JR NC,L2C82
        DEC HL
        RST 10H
        JR Z,L2C75
        PUSH BC
        RST 8H
        DEFB ','
        JR L2C39
L2C75:
        LD A,B
        LD (VOICEN),A
```

```
        CALL L2D60
        INC B
        LD A,B
        CP 03H
        JR C,L2C75
L2C82:
        DEC HL
        RST 10H
        JP NZ,SNERR
        PUSH HL
L2C88:
        XOR A
L2C89:
        PUSH AF
        LD (VOICEN),A
        LD B,A
        CALL L2D7A
        JP C,L2D11
        LD A,B
        CALL GETVCP
        LD A,(HL)
        OR A
        JP Z,L2D11
        LD (MCLLEN),A
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        INC HL
        LD (MCLPTR),DE
        LD E,(HL)
        INC HL
        LD D,(HL)
        INC HL
        PUSH HL
        LD  L,024H
        CALL L2D4A
        PUSH HL
        LD HL,(SAVSP)
        DEC HL
        POP BC
        DI
        CALL BLTUC
        POP DE
        LD H,B
        LD L,C
        LD SP,HL
        EI
        LD  A,0FFH
        LD (MCLFLG),A
        JP MCLSCN
L2CCA:
        LD  A,(MCLLEN)
        OR A
        JR NZ,L2CD3
MCLEND:
        CALL L2D60
L2CD3:
        LD  A,(VOICEN)
        CALL GETVCP
```



```
        LD  A,(MCLLEN)
        LD (HL),A
        INC HL
        LD DE,(MCLPTR)
        LD (HL),E
        INC HL
        LD (HL),D
        LD  HL,00H
        ADD HL,SP
        EX DE,HL
        LD HL,(SAVSP)
        DI
        LD SP,HL
        POP BC
        POP BC
        POP BC
        PUSH HL
        OR A
        SBC HL,DE
        JR Z,L2D0F
        LD  A,0F0H
        AND L
        OR H
        JR NZ,L2D42
        LD  L,024H
        CALL L2D4A
        POP BC
        DEC BC
        CALL BLTUC
        POP HL
        DEC HL
        LD (HL),B
        DEC HL
        LD (HL),C
        JR L2D11
L2D0F:
        POP BC
        POP BC
L2D11:
        EI
        POP AF
        INC A
        CP 03H
        JP C,L2C89
        DI
        LD  A,(INTFLG)
        CP 03H
        JR Z,L2D3D
        LD  A,(PRSCNT)
        RLCA
        JR C,L2D2E
        LD  HL,PLAYCNT
        INC (HL)
        CALL STRTMS
L2D2E:
        EI
        LD  HL,PRSCNT
        LD A,(HL)
        OR 080H
        LD (HL),A
        CP 083H
```

```
        JP NZ,L2C88
L2D3B:
        POP HL
        RET
L2D3D:
        CALL GICINI
        JR L2D3B
L2D42:
        EI
        CALL FCERR
GETVCP: 2D46
        LD  L,02H
        JR GETVC1
L2D4A:
        LD  A,(VOICEN)
GETVC1: 2D4D
        PUSH DE
        LD  DE,VCBA/METREX
        LD H,00H
        ADD HL,DE
        OR A
        JR Z,L2D5E
        LD  DE,025H
L2D5A:
        ADD HL,DE
        DEC A
        JR NZ,L2D5A
L2D5E:
        POP DE
        RET
L2D60:
        LD  A,(PRSCNT)
        INC A
        LD (PRSCNT),A
        LD  E,0FFH
L2D69:
        PUSH HL
        PUSH BC
L2D6B:
        PUSH DE
        LD  A,(VOICEN)
        DI
        CALL PUTQ
        EI
        POP DE
        JR Z,L2D6B
        POP BC
        POP HL
        RET
L2D7A:
        LD  A,(VOICEN)
        PUSH BC
        DI
        CALL LFTQ
        EI
        POP BC
        CP 08H
        RET
PLYTAB: 2D87
        DEFB 'A'
        DEFW L2E97
```



```
        DEFB 'B'
        DEFW L2E97
        DEFB 'C'
        DEFW L2E97
        DEFB 'D'
        DEFW L2E97
        DEFB 'E'
        DEFW L2E97
        DEFB 'F'
        DEFW L2E97
        DEFB 'G'
        DEFW L2E97
        DC 'M'
        DEFW L2DF7
        DC 'V'
        DEFW L2DDF
        DC 'S'
        DEFW L2E17
        DC 'N'
        DEFW L2E7A
        DC 'O'
        DEFW L2E48
        DC 'R'
        DEFW L2E55
        DC 'T'
        DEFW L2E3B
        DC 'L'
        DEFW L2E21
        DEFB 'X'
        DEFW L22CC
        DEFB 0
L2DB8:
        DEFB 16,18,20,22,0,0,2,4,6,8,10,10,12,14,16
        DEFB 0E8H,14,18,14,048H,13,089H,12,0D5H,11
        DEFB 02BH,11,08AH,10,0F3H,9,064H,9,0DDH,8,05EH
        DEFB 8,0E6H,7

L2DDF:                  ; PLAY S
        JR C,L2DE3
        LD  E,08H
L2DE3:
        LD  A,0FH
        CP E
        JR C,L2E38
L2DE8:
        XOR A
        OR D
        JR NZ,L2E38
        LD  L,012H
        CALL L2D4A
        LD  A,040H
        AND (HL)
        OR E
        LD (HL),A
        RET
L2DF7:                  ; PLAY M
        LD A,E
        JR C,L2DFD
        CPL
        INC A
        LD E,A
```

```
L2DFD:
        OR D
        JR Z,L2E38
        LD  L,013H
        CALL L2D4A
        PUSH HL
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        RST 20H
        POP HL
        RET Z
        LD (HL),E
        INC HL
        LD (HL),D
        DEC HL
        DEC HL
        LD  A,040H
        OR (HL)
        LD (HL),A
        RET
L2E17:                  ; PLAY S
        LD A,E
        CP 010H
        JR NC,L2E38
        OR 010H
        LD E,A
        JR L2DE8
L2E21:                  ; PLAY L
        JR C,L2E25
        LD  E,04H
L2E25:
        LD A,E
        CP 041H
        JR NC,L2E38
        LD  L,010H
L2E2C:
        CALL L2D4A
        XOR A
        OR D
        JR NZ,L2E38
        OR E
        JR Z,L2E38
        LD (HL),A
        RET
L2E38:
        CALL FCERR
L2E3B:                  ; PLAY T
        JR C,L2E3F
        LD  E,078H
L2E3F:
        LD A,E
        CP 020H
        JR C,L2E38
        LD  L,011H
        JR L2E2C
L2E48:                  ; PLAY O
        JR C,L2E4C
        LD  E,04H
L2E4C:
```



```
        LD A,E
        CP 09H
        JR NC,L2E38
        LD  L,0FH
        JR L2E2C
L2E55:                  ; PLAY R
        JR C,L2E59
        LD  E,04H
L2E59:
        XOR A
        OR D
        JR NZ,L2E38
        OR E
        JR Z,L2E38
        CP 'A'
        JR NC,L2E38
L2E64:
        LD  HL,00H
        PUSH HL
        LD  L,010H
        CALL L2D4A
        PUSH HL
        INC HL
        INC HL
        LD A,(HL)
        LD (SAVVOL),A
        LD (HL),080H
        DEC HL
        DEC HL
        JR L2EF5
L2E7A:                  ; PLAY N
        JR NC,L2E38
        XOR A
        OR D
        JR NZ,L2E38
        OR E
        JR Z,L2E64
        CP 061H
        JR NC,L2E38
        LD A,E
        LD B,00H
        LD E,B
L2E8B:
        SUB 0CH
        INC E
        JR NC,L2E8B
        ADD A,0CH
        ADD A,A
        LD C,A
        JP L2ECC
L2E97:                  PLAY A-G
        LD B,C
        LD A,C
        SUB 040H
        ADD A,A
        LD C,A
        CALL FETCHR
        JR Z,L2EBE
        CP '#'
        JR Z,L2EBF
        CP '+'
```

```
        JR Z,L2EBF
        CP '-'
        JR Z,L2EB3
        CALL DECFET
        JR L2EBE
L2EB3:
        DEC C
        LD A,B
        CP 'C'
        JR Z,L2EBD
        CP 'F'
        JR NZ,L2EBE
L2EBD:
        DEC C
L2EBE:
        DEC C
L2EBF:
        LD  L,0FH
        CALL L2D4A
        LD E,(HL)
        LD B,00H
        LD  HL,02DB8H
        ADD HL,BC
        LD C,(HL)
L2ECC:
        LD  HL,02DC7H
        ADD HL,BC
        LD A,E
        LD E,(HL)
        INC HL
        LD D,(HL)
L2ED4:
        DEC A
        JR Z,L2EE0
        SRL D
        RR E
        JR L2ED4
L2EDD:
        CALL FCERR
L2EE0:
        ADC A,E
        LD E,A
        ADC A,D
        SUB E
        LD D,A
        PUSH DE
        LD  L,010H
        CALL L2D4A
        LD C,(HL)
        PUSH HL
        CALL FETCHR
        JR Z,L2F02
        CALL VALSC2
L2EF5:
        LD  A,040H
        CP E
        JR C,L2EDD
        XOR A
        OR D
        JR NZ,L2EDD
        OR E
```



```
        JR Z,L2F02
        LD C,E
L2F02:
        POP HL
        LD D,00H
        LD B,D
        INC HL
        LD E,(HL)
        PUSH HL
        CALL UMULT
        EX DE,HL
        CALL CONSIH
        CALL VMOVAF
        LD  HL,02FADH
        CALL MOVFM
        CALL DDIV/DECDIV
        CALL FRCINT
        LD D,H
        LD E,L
L2F21:
        CALL FETCHR
        JR Z,L2F3C
        CP 02EH
        JR NZ,L2F39
        SRL D
        RR E
        ADC HL,DE
        LD  A,0E0H
        AND H
        JR Z,L2F21
        XOR H
        LD H,A
        JR L2F3C
L2F39:
        CALL DECFET
L2F3C:
        LD  DE,05H
        RST 20H
        JR C,L2F43
        EX DE,HL
L2F43:
        LD  BC,0FFF7H
        POP HL
        PUSH HL
        ADD HL,BC
        LD (HL),D
        INC HL
        LD (HL),E
        INC HL
        LD  C,02H
        EX (SP),HL
        INC HL
        LD E,(HL)
        LD A,E
        AND 0BFH
        LD (HL),A
        EX (SP),HL
        LD  A,080H
        OR E
        LD (HL),A
        INC HL
```

```
        INC C
        EX (SP),HL
        LD A,E
        AND 040H
        JR Z,L2F6F
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        POP HL
        LD (HL),D
        INC HL
        LD (HL),E
        INC HL
        INC C
        INC C
        DEFB 0FEH       ;*      CP 0E1H
L2F6F:
        POP HL          ;*
        POP DE
        LD A,D
        OR E
        JR Z,L2F7A
        LD (HL),D
        INC HL
        LD (HL),E
        INC C
        INC C
L2F7A:
        LD  L,07H
        CALL L2D4A
        LD (HL),C
        LD A,C
        SUB 02H
        RRCA
        RRCA
        RRCA
        INC HL
        OR (HL)
        LD (HL),A
        DEC HL
        LD A,D
        OR E
        JR NZ,L2F9A
        PUSH HL
        LD  A,(SAVVOL)
        OR 080H
        LD  BC,0BH
        ADD HL,BC
        LD (HL),A
        POP HL
L2F9A:
        POP DE
        LD B,(HL)
        INC HL
L2F9D:
        LD E,(HL)
        INC HL
        CALL L2D69
        DJNZ L2F9D
        CALL L2D7A
```



```
        JP C,L2CCA
        JP MCLSCN
L2FAD:
        DEFB 040H,0,045H,20
PUT:/2FB1               ; ************************************
        LD B,080H
        DEFB 17                 ;*      LD  DE,06H
GET:/2FB4               ; ************************************
        LD B,0                  ;*
        CP 0EEH         ; SPRITE-Token
        JP Z,PUTSPR
        CP 040H
        CALL Z,CHRGTR
        CP 0DCH         ; STEP-Token
        JR Z,L2FC6
        CP '('
L2FC6:
        LD A,B
        JP Z,GPUTG
        AND A
        JP Z,DGET
        JP DPUT
LOCATE:/2FD1            ; ************************************
        LD DE,(CSRY)    ; Voreinstellung
        PUSH DE
        LD  A,(SCREEN)
        AND A
        JR NZ,L3018     ; JP falls Grafik
        DEC HL
        RST 10H
        CP ','
        JR Z,L2FED
        CALL GETBYT
        INC A           ; +1
        POP DE
        LD D,A          ; Spalte setzen
        PUSH DE
        DEC HL
        RST 10H
        JR Z,L3012
L2FED:
        RST 8H
        DEFB ','
        CP ','
        JR Z,L2FFE
        CALL GETBYT
        INC A           ; +1
        POP DE
        LD E,A          ; Zeile setzen
        PUSH DE
        DEC HL
        RST 10H
        JR Z,L3012
L2FFE:
        RST 8H
        DEFB ','
        CALL GETBYT
        AND A
        LD  A,'y'
```

```
        JR NZ,L3009
        DEC A               ; ='x'
L3009:
        PUSH AF             ; setzen des Cursor-Attributs
        LD  A,01BH
        RST 18H
        POP AF
        RST 18H
        LD  A,'5'
        RST 18H
L3012:
        EX (SP),HL
        CALL POSIT          ; und der Position
        POP HL
        RET
L3018:                      ; Grafik LOCATE
        DEC HL
        RST 10H
        CP ','
        JR Z,L3028
        CALL GETBYT
        POP DE
        LD D,A              ; x Koordinate
        PUSH DE
        DEC HL
        RST 10H
        JR Z,L3030
L3028:
        RST 8H
        DEFB ','
        CALL GETBYT
        POP DE
        LD E,A              ; Y Koordinate
        PUSH DE
L3030:
        EX (SP),HL
        LD (CSRY),HL
        POP HL
        RET
MDM:    3036                ; ***********************************
        PUSH HL             ; MDM on/OFF/STOP
        LD  HL,MDM_OOS
        JR L306B
STOPTP: 303C                ; ***********************************
        PUSH HL             ; STOP ON/OFF/STOP
        LD  HL,STOP_OOS
        JR L306B
SPRTTP: 3042                ; ***********************************
        PUSH HL             ; SPRITE ON/OFF/STOP
        LD  HL,SPRITE_OOS
        JR L306B
INTTRP: 3048                ; ***********************************
        RST 8H              ; INTERVAL ON/OFF/STOP
        DEFB 'E'
        RST 8H
        DEFB 'R'
        RST 8H
        DEFB OFFH           ;* VAL token
        RST 8H
        DEFB 094H           ;*
        PUSH HL
```



```
        LD  HL,INTERVAL_OOS
        JR L306B
STRIG:  3056                ; ***********************************
        RST 8H              ; STRIG () ON/OFF/STOP
        DEFB '('
        CALL GETBYT
        CP 03H              ; Bereich 0-2
        JP NC,FCERR
        RST 8H
        DEFB ')'
        PUSH HL
        LD D,00H
        LD  HL,STRIG_OOS
        ADD HL,DE
        ADD HL,DE
        ADD HL,DE
L306B:
        CALL L30A9
        JR L307E
L3070:
        CALL GETBYT
        DEC A
        CP 0AH              ; Bereich 0-9
        JP NC,FCERR
        LD A,(HL)
        PUSH HL
        CALL L3093
L307E:
        POP HL
        POP AF
        RST 10H
        JP NWSTRT
L3084:
        PUSH HL
        LD  E,0AH
L3087:
        PUSH DE
        PUSH AF
        CALL L3093
        POP AF
        POP DE
        DEC E
        JR NZ,L3087
        JR L307E
L3093:
        LD D,00H
        LD  HL,FNKSWI
        ADD HL,DE
        PUSH HL
        LD  HL,TXPSAV
        ADD HL,DE
        ADD HL,DE
        ADD HL,DE
        CALL L30A9
        LD A,(HL)
        AND 01H
        POP HL
        LD (HL),A
        RET
L30A9:                      ; ***** INTERRUPT  ON/OFF/STOP
        CP 095H             ; ON-token
```

```
        JP Z,ONTRP
        CP 0EBH         ; OFF
        JP Z,OFFTRP
        CP 090H         ; STOP
        JP Z,STPTRP
        JP SNERR
ONGOTP: / 3000          ; ******************************** ON
        CALL 0FF75H
        CP 0CFH         ; MDM
        LD BC,01001H
        RET Z
        CP 0C7H         ; KEY
        LD BC,0000AH
        RET Z
        CP 090H         ; STOP
        LD BC,00A01H
        RET Z
        CP 0EEH         ; SPRITE
        LD BC,00B01H
        RET Z
        CP 0FFH
        SCF
        RET NZ
        PUSH HL
        RST 10H
        CP 0A3H         ; STRIG
        JR Z,L30E7
        CP 085H         ; INT
        JR Z,L30EC
L30E4:
        POP HL
        SCF
        RET

L30E7:                  ; (STRIG)
        POP BC
        LD BC,00C03H
        RET
L30EC:                  ; (INT)
        RST 10H
        CP 'E'
        JR NZ,L30E4
L30F1:
        POP BC
        RST 10H
        RST 8H
        DEFB 'R'
        RST 8H
        DEFB 0FFH
        RST 8H
        DEFB 094H       ; VAL Token
        RST 8H
        DEFB 0F1H       ; =
        CALL FRMQNT
        LD A,D
        OR E
        JP Z,FCERR      ; <>0
        EX DE,HL
        LD (INTVAL),HL
        LD (INTCNT),HL
        EX DE,HL
```



```
        LD BC,0F01H
        DEC HL
        RET

SETGSB: / 3110          ; Setzen der GOSUB Adresse
        PUSH HL
        LD B,A
        ADD A,A
        ADD A,B
        LD L,A
        LD H,00H
        LD BC,0FDECH
        ADD HL,BC
        LD (HL),E
        INC HL
        LD (HL),D
        POP HL
        RET
KEY: / 3120             ; ************************************
        CP 093H         ; LIST
        JR NZ,L3156
        RST 10H
        PUSH HL
        LD HL,FNKSTR    ; Keytabelle
        LD C,05H        ; 2 * 5 Listen
L312B:
        CALL L3139
        CALL L3139
        CALL CRDO       ; CR LF
        DEC C
        JR NZ,L312B
        POP HL
        RET
L3139:                  ; eine Funktionstaste listen
        LD B,010H       ; 16 Zeichen
        CALL KEYCHR
        LD B,03H        ; und drei BLANKS
L3140:
        RST 18H
        DJNZ L3140
        RET
KEYCHR: / 3144          ; Zeichen aus Keydefinition listen
        LD A,(HL)
        CP 07FH
        JR Z,L314D
        CP ' '
        JR NC,L314F
L314D:
        LD A,' '
L314F:
        RST 18H
        INC HL
        DJNZ KEYCHR
        LD A,' '
        RET
L3156:
        CP '('
        JP Z,L3070
        CP 095H         ; ON
        JP Z,L3084
        CP 0EBH         ; OFF
```

```
        JP Z,L3084
        CP 090H         ; STOP
        JP Z,L3084
        CALL GETBYT     ; ******************** KEY x,""
        DEC A
        CP 0AH
        JP NC,FCERR     ; Bereich 1-10
        EX DE,HL
        LD L,A
        LD H,00H
        ADD HL,HL
        ADD HL,HL
        ADD HL,HL
        ADD HL,HL
        LD  BC,FNKSTR
        ADD HL,BC
        PUSH HL         ; Anfangsadresse des Keys in Tabelle
        EX DE,HL
        RST 8H
        DEFB ','
        CALL FRMEVL
        PUSH HL
        CALL FRESTR
        LD B,(HL)
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        POP HL
        EX (SP),HL
        LD  C,0FH       ; 15 Zeichen
        LD A,B
        AND A
        JR Z,L31A4
L3197:                  ; Zeichen in Tabelle eintragen
        LD  A,(DE)
        AND A
        JP Z,FCERR
        LD (HL),A
        INC DE
        INC HL
        DEC C
        JR Z,L31A9
        DJNZ L3197
L31A4:
        LD (HL),B
        INC HL
        DEC C
        JR NZ,L31A4
L31A9:
        LD (HL),C
        CALL FNKSB
        POP HL
        RET
CLICK:  31AF            ; **********************************
        CP 095H         ; ON
        JR Z,L31B8
        SUB 0EBH        ; OFF
        JP NZ,SNERR
L31B8:
        LD (CLICKVAR),A ; 0 falls OFF
```



```
        RST 10H
        RET
TIME:   3180            ; **********************************
        RST 10H
        PUSH HL
        LD HL,(TIMEVAR)
        CALL INEG2
        POP HL
        RET
GETLIN: 3187            ; ***************************** CSRLIN
        RST 10H
        PUSH HL
        LD  A,(SCREEN)
        SUB 01H
        LD  A,(CSRY)
        JR L31F6
STTIME: 3193            ; ***************************** TIME=
        RST 10H
        RST 8H
        DEFB 0F1H       ; =
        CALL FRMQNT
        LD (TIMEVAR),DE
        RET
PLAYF:  31DE
        RST 10H
        CALL GETBYT
        CP 04H          ; Bereich 0-3
        JP NC,FCERR
        PUSH HL
        LD  A,(MUSIKF)
        DEC E
        JP M,L31FE
L31EF:
        RRCA
        DEC E
        JP P,L31EF
        LD  A,00H
L31F6:
        JR NC,L31F9
        DEC A
L31F9:
        CALL CONIA
        POP HL
        RET
L31FE:
        AND 07H
        JR Z,L31F9
        LD  A,0FFH
        JR L31F9
STICK:  3206            ; ***************************** STICK
        CALL CONINT
        CP 03H
        JP NC,FCERR     ; BEREICH 0-2
        DEC A
        LD B,A
        JP M,L3238      ; JP falls STICK(0)
        LD  A,07H
        DI
        OUT (088H),A    ; Register 7
        IN A,(090H)
        AND 0BFH        ; AND 10111111B
```

```
        OUT (08CH),A    ; enable input reg 14
        LD  A,0EH       ; Register 14
        OUT (088H),A
        IN A,(090H)     ; Lesen
        EI
        DJNZ L322B      ; shiften falls STICK(1)
        RRCA
        RRCA
        RRCA
        RRCA
L322B:
        AND 0FH
        LD E,A
        LD D,00H
        LD  HL,L3253    ; Tabelle STICK-Wert --> Richtungswert
        ADD HL,DE
        LD A,(HL)
        JP SNGFLT
L3238:                  ; ***** STICK(0)
        LD  HL,0FD88H
        LD A,(HL)       ; Dekodierung der Tastatur
        ADD A,A
        RL E
        DEC HL
        DEC HL
        LD A,(HL)
        ADD A,A
        RL E
        INC HL
        LD A,(HL)
        ADD A,A
        RL E
        DEC HL
        DEC HL
        LD A,(HL)
        ADD A,A
        RL E
        LD A,E
        JR L322B

L3253:
        DEFB 0,5,1,0,3,4,2,3,7,6,8,7,0,5,1,0

TRIGF:/ 3263            ; ***************************** STRIG
        CALL GETBYT
        XOR C
        LD  A,(DE)
        DEC A
        JP M,L3279      ; JP falls STRIG 0
        IN A,(098H)
        JR Z,L326F      ; JP falls STRIG 1
        RRCA
L326F:
        AND 010H
L3271:
        LD  A,0FFH
        JR Z,L3276
        INC A
L3276:                  ; 0 oder -1 als wert
        JP CONIA
L3279:                  ; **** STRIG 0
```



```
        LD  A,(0FD88H)
        AND 01H
        JR L3271        ; Space-Taste
PDL:/ 3280              ;************************************
        CALL CONINT
        LD B,A
        DEC A
        CP 04H
        JP NC,FCERR     ; Bereich 1-4
        XOR A
        SCF
L328C:
        RLA
        DJNZ L328C      ; 1, 2, 4 oder 8
        PUSH AF
        LD  A,07H
        DI
        OUT (088H),A
        IN A,(090H)
        LD E,A
        OR 040H         ; enable output reg 14
        OUT (08CH),A
        LD  A,0EH       ; reg 14
        OUT (088H),A
        LD  A,0EEH      ; forward low alle anderen high
        OUT (08CH),A
        LD  A,0FFH      ; alle high
        OUT (08CH),A
        POP BC
        LD  C,00H
L32AB:
        IN A,(098H)
        AND B           ; TA bis TD je nach PDL
        JR Z,L32B3      ; JP falls fertig
        INC C
        JR NZ,L32AB
L32B3:
        LD  A,07H
        CALL L40B6      ; Ruecksetzen des Registers 7
        EI
        LD A,C
        JP SNGFLT
PAD:/ 32BD              ; ****************************************
        CALL CONINT
        CP 04H
        JP NC,FCERR     ; Bereich 0-3
        DEC A
        JP M,L32D6      ; PAD(0)
        DEC A
        LD  A,(PADX)
        JP M,SNGFLT     ; PAD(1)
        LD  A,(PADY)
        JP Z,SNGFLT     ; PAD(2)
L32D6:                  ; OFFH bei 0     und 1 bei 3
        LD  A,07H
        DI
        OUT (088H),A
        IN A,(090H)
        PUSH AF
        OR 040H
        OUT (08CH),A    ; enable output reg 14
```

```
        LD  A,0EH
        OUT (088H),A    ; reg 14 setzen
        POP AF
        PUSH AF
        JP M,L32F4      ; JP falls PAD(0)
        LD  A,05H
        OUT (08CH),A    ; left * forward high alle anderen low
        IN A,(098H)     ; read TA-TD*Trigger
        LD H,A
        JR L331F
L32F4:
        LD  C,08H
        CALL L333F
        JR C,L331B
        CALL L332E
        PUSH DE
        CALL L332E
        POP BC
        LD A,B
        SUB D
        JR NC,L3309
        CPL
        INC A
L3309:
        CP 05H
        JR NC,L32F4
        LD A,C
        SUB E
        JR NC,L3313
        CPL
        INC A
L3313:
        CP 05H
        JR NC,L32F4
        LD (PADY),DE
L331B:
        LD  A,05H
        OUT (08CH),A
L331F:
        LD  A,07H
        OUT (088H),A
        POP AF
        OUT (08CH),A    ; Reg 7 ruecksetzen
        EI
        LD A,H
        RRCA
        SBC A,A
        CPL
        JP CONIA
L332E:
        LD  C,0AH
        CALL L333F
        LD D,L
        PUSH DE
        LD  C,08H
        CALL L333F
        POP DE
        LD E,L
        XOR A
        LD H,A
        RET
```



```
L333F:
        CALL L3361
        LD B,08H
        LD A,C
L3345:
        AND 0EH
        PUSH AF
        OUT (08CH),A
        IN A,(098H)
        LD H,A
        ADD A,A
        ADD A,A
        ADD A,A
        ADD A,A
        RL L
        POP AF
        OR 01H
        OUT (08CH),A
        DJNZ L3345
        OR 04H
        OUT (08CH),A
        LD A,H
        RRA
        RET
L3361:
        LD  A,05H
        OR C
        PUSH AF
        OUT (08CH),A
L3367:
        IN A,(098H)
        AND 02H
        JR Z,L3367
        POP AF
        AND 0BH
        OUT (08CH),A
        EX (SP),HL
        EX (SP),HL
        EX (SP),HL
        EX (SP),HL
        RET
RETSWI: 3377            ; ************************** SWITCH=
        RST 10H
        PUSH HL
        LD  A,(SWIFLG)
        JP L31F9
SWITCH: 337F            ; ************************** SWITCH
        LD  A,00H
        JR Z,L3387      ; falls Stmtende
        RST 8H
        DEFB 090H       ; STOP
        RET NZ
        INC A
L3387:
        LD (STPOPT),A
        CALL CLSALL     ; Files schliessen
        DI
        PUSH HL
        LD (SPSAVE),SP  ; Stack pointer sichern
        LD  A,0FH
        OUT (088H),A
```

```
        IN A,(090H)
        LD C,A
        XOR 04H          ; Bankwechsel
        LD B,A
        AND 04H
        JR NZ,L33A7      ; JP falls neue BANK 02 ist
        CALL CHKBNK
        JP C,FCERR
L33A7:
        LD A,B
        OUT (08CH),A     ; Wechsel ausfuehren
        LD B,C
        LD C,A
        LD HL,(FRSTID)
        LD  DE,0534AH
        OR A
        SBC HL,DE
        JP NZ,INIENT     ; Initialisieren falls nicht geschehen
        LD SP,(SPSAVE)
        LD  HL,L33EB
L33BF:                   ; Parameter ueber Tabelle in zweite
        LD E,(HL)        ; Bank uebertragen
        INC HL
        LD D,(HL)        ; Adresse
        LD A,D
        OR E
        JR Z,L33D7
        INC HL
        LD A,(HL)        ; Laenge
        INC HL
        PUSH HL
        EX DE,HL
        LD E,A
L33CC:
        CALL GETBNK
        LD (HL),A
        INC HL
        DEC E
        JR NZ,L33CC
        POP HL
        JR L33BF
L33D7:
        LD  HL,STPOPT
        CALL GETBNK
        AND A
        JR Z,L33E5
        LD  A,03H
        LD (INTFLG),A    ; falls SWITCH STOP
L33E5:
        CALL FNKSB
        POP HL
        EI
        RET

L33EB:
        DEFW CSRY                ;*33EB
        DEFB 2
        DEFW TTYPOS
        DEFB 1
        DEFW LPOS
        DEFB 1
```



```
        DEFW CSRSW
        DEFB 1
        DEFW LINTTB
        DEFB 24
        DEFW LINLEN
        DEFB 1
        DEFW CNSDFG
        DEFB 1
        DEFW FNKSWI
        DEFB 1
        DEFW CAPST
        DEFB 1
        DEFW CLICK
        DEFB 1
        DEFW REPCNT
        DEFB 1
        DEFW OLDKEYS
        DEFB 20
        DEFW SCREEN
        DEFB 1
        DEFW SPRSIZ
        DEFB 1
        DEFW VDPreg0
        DEFB 1
        DEFW RG1SAV
        DEFB 1
        DEFW STATFL
        DEFB 1
        DEFW 0

CHKBNK: 3420             ; ***** 2. Bank Testen
        LD A,B
        OUT (08C),A      ; Einschalten
        LD  HL,0C000H
L3426:
        LD A,(HL)
        CPL
        LD (HL),A
        CP (HL)
        CPL
        LD (HL),A
        JR NZ,L345E
        INC L
        JR NZ,L3426
        LD HL,(0C000H)
        EX DE,HL
        LD A,C
        OUT (08CH),A
        LD HL,(0C000H)
        RST 20H
        JR NZ,L345D
        LD A,L
        CPL
        LD L,A
        LD A,H
        CPL
        LD H,A
        LD (0C000H),HL
        LD A,B
        OUT (08CH),A
        LD DE,(0C000H)
```

```
        LD A,C
        OUT (08CH),A
        RST 20H
        LD A,L
        CPL
        LD L,A
        LD A,H
        CPL
        LD H,A
        LD (0C000H),HL
        JR Z,L345E
L345D:
        DEFB 0F6H       ;*      OR 037H
L345E:
        SCF             ;*
        LD A,C
        OUT (08CH),A
        RET
GETBNK: 3463            ; ***** Byte aus der anderen Bank holen
        PUSH DE         ; B neue Bank   C jetzige Bank  A Byte
        LD A,B
        OUT (08CH),A
        LD D,(HL)
        JR L3470
PUTBNK: 346A            ; ***** Byte in die andere Bank speichern
        PUSH DE
        LD D,A
        LD A,B
        OUT (08CH),A
        LD (HL),D
L3470:
        LD A,C
        OUT (08CH),A
        LD A,D
        POP DE
        RET
JMPBNK: 3476            ; ***** auf HL in der neuen Bank springen
        LD  A,0FH
        DI
        OUT (088H),A
        LD A,B
        OUT (08CH),A
        EI
L347F:
        JP (HL)
CALBNK: 3480            ; ***** andere Bank ueber CALL aufrufen
        LD  A,0FH       ; B neue Bank C jetzige Bank HL Call
        DI              ;       Adresse
        OUT (088H),A
        LD A,B
        OUT (08CH),A
        EI
        PUSH BC
        CALL L347F
        POP BC
        LD  A,0FH
        DI
        OUT (088H),A
        LD A,C
        OUT (08CH),A
        EI
```



```
        RET
RSTFNK: 3498    ; **** Funktionstastenbelegung initialisieren
        LD  BC,0A0H
        LD  DE,FNKSTR
        LD  HL,FNKROM
        LDIR
        JP FNKSB
DSKO$:  34A6            ; ************************************
        CALL LFF8A
        JR L34D6
SETS:   34AB            ; ************************************
        CALL LFF8D
        JR L34D6
NAME:   34B0            ; ************************************
        CALL LFF90
        JR L34D6
KILL:   34B5            ; ************************************
        CALL LFF93
        JR L34D6
IPL:    34BA            ; ************************************
        CALL LFF96
        JR L34D6
DKCOPY: 34BF            ; ************************************
        CALL LFF99
        JR L34D6
CMD:    34C4            ; ************************************
        CALL LFF9C
        JR L34D6
DSKF:   34C9            ; ************************************
        CALL LFF9F
        JR L34D6
DSKI$:  34CE            ; ************************************
        CALL LFFA2
        JR L34D6
ATTR$:  34D3            ; ************************************
        CALL LFFA5
L34D6:
        JP FCERR
L34D9:                  ; ***** Initialisierung der IO-Ports
        LD  A,0EH
        LD  E,0FFH
        CALL L40B6      ; reg 14 auf OFFH
        LD  A,07H
        LD  E,080H
        CALL L40B6      ; reg 7     reg 15 output reg 14 input
        LD  A,092H      ; 8255   A,B Input  C Output
        OUT (097H),A
        LD  A,010H
        OUT (096H),A    ; Casette einschalten
        LD  A,0FFH
        OUT (011H),A    ; Printer STROBE disable
        LD  BC,01051H   ; 80-Zeichen-Karte Initialisieren
        LD  HL,L3511
L34F9:
        LD A,B
        DEC A
        OUT (050H),A
        OUTD
        JR NZ,L34F9
        RET
```

```
        .RADIX 16
        DEFB 6D,50,5C,4,1C,4,18,1A,0,8,60,8,0,0,0
L3511:
        0
        .RADIX 10

BREAKX: 3512            ; ***** CTRL STOP testen
        IN A,(09AH)
        AND 0F0H
        OR 06H
        OUT (096H),A
        IN A,(099H)
        AND 022H
        RET NZ          ; NZ falls nicht gedrueckt
        PUSH HL
        LD HL,(PUTPNT)
        LD (GETPNT),HL
        POP HL
        LD  A,(0FD7BH)  ; Stoptaste loeschen
        AND 0DFH
        LD (0FD7BH),A
        LD  A,0DH
        LD (REPCNT),A
        SCF
        RET
VDPWRT: 3536            ; ***** VDP-Register schreiben
        LD A,B          ; B Byte   C register
        DI
        OUT (081H),A
        LD A,C
        OR 080H
        OUT (081H),A
        EI
        RET
INITXT: 3541            ; ***** Textschirm initialisieren
        LD  A,(VDPReg0)
        AND 01H         ; nur Bit 0 beibehalten enable video
        LD (VDPReg0),A  ; im Register 0
        LD B,A
        LD  C,00H
        CALL VDPWRT     ; Register 0 setzen
        LD  A,(RG1SAV)
        AND 11100111B   ; 16K,active, VDPinterrupt
        OR 010H         ; Textmode,  16*16 Sprites, Faktor 2
        LD (RG1SAV),A
        LD B,A
        LD  C,01H
        CALL VDPWRT     ; setzen
        LD  BC,02H      ; reg 2 auf 0
        CALL VDPWRT     ; 0 - 3FF Basisbildschirm
        LD  BC,0104H    ; reg 4 auf 1
        CALL VDPWRT     ; 800 - FFF  Maskenspeicher
        CALL L3759      ; setzen der Vorder-/Hintergrundfarbe
        LD  HL,0101H
        LD (CSRY),HL    ; Cursor-Position
        CALL L370A      ; Bildschirm loeschen
        CALL L3584      ; setzen der Chargeneratormasken
        LD  A,01BH
        RST 18H
        LD  A,'y'
        RST 18H
```



```
        LD  A,'5'
        RST 18H         ; Cursor-Modus
        RET
L3584:                  ;************* Erzeugen der Zeichen-
        CALL LFFAB      ; generator-masken und Uebertragen
        LD  HL,MLTNAM/TXTCGP    ; in den Videospeicher,
        CALL SETWRT     ; VDP fuer write ab 800H vorbereiten
        CALL L359D      ; erzeugen des ASCII Zeichensatzes
        XOR A
        LD (LINWRK),A
        LD  A,0A0H
L3596:
        CALL L35B0      ; erzeugen des Grafikzeichensatzes
        INC A
        JR NZ,L3596
        RET
L359D:                  ; ********
        XOR A           ; normale Zeichen
        CALL L35A3      ; erzeugen
        LD  A,0FFH      ; inverse Zeichen erzeugen
L35A3:
        LD (LINWRK),A
        LD  A,020H      ; 1. gespeicherter ASCII code
L35A8:
        CALL L35B0      ; Maske erzeugen
        INC A
        JP P,L35A8
        RET
L35B0:                  ; ***** Zeichengenerator-Maske fuer ein
        PUSH AF         ;       Zeichen erzeugen
        CALL GETPAT     ; Maske aus Tabelle holen
        LD  DE,PATWRK
        LD B,08H        ; 8 Zeichen pro Maske
L35B9:
        LD  A,(DE)
        LD  HL,LINWRK
        XOR (HL)        ; Invers-oder Normalmodus
        OUT (080H),A    ; ausgeben zum VDP
        INC DE
        DJNZ L35B9
        POP AF
        RET
GETPAT: 35C5            ; ***** eine Zeichengeneratormaske
        LD H,00H        ;       aus der Tabelle expandieren
        LD L,A
        CP 020H
        JR C,L35DD
        CP 07FH
        JR C,L35DF
        CP 0A0H
        JR C,L35DD
        CP 0E0H
        JR NC,L35DD
        LD  DE,04012H   ; Basis adresse fuer Grafikzeichen
        JR L35E2
L35DD:
        LD  L,020H      ; neuer code fuer nicht vorhandene
L35DF:                  ;       Zeichencodes
        LD  DE,040D8H   ; Basisadresse fuer codes 20-7F
L35E2:
        LD B,H
```

```
        LD C,L
        ADD HL,HL       ; *2
        ADD HL,BC       ; *3
        ADD HL,HL       ; *6
        ADD HL,DE       ; code*6+Basis
        LD  DE,PATWRK   ; Zieladresse
        CALL L35EE      ; erste 4 Bytes fuer Maske
L35EE:
        PUSH DE
        LD C,(HL)
        INC HL
        LD D,(HL)
        INC HL
        LD E,(HL)       ; 3 Bytes aus Tabelle
        INC HL
        EX (SP),HL      ; HL Zieladresse
        LD B,04H        ; aus 3 Tabellen- 4 Maskenbytes erzeugen
L35F8:
        PUSH HL
        LD  L,06H
        XOR A
L35FC:
        RL E
        RL D
        RL C
        RLA
        DEC L
        JR NZ,L35FC     ; 6 mal links shiften
        ADD A,A         ; shift A links um 2 Bit
        ADD A,A
        POP HL          ; zieladresse+..
        LD (HL),A       ; Maskenbyte xxxxxx00 speichern
        INC HL
        DJNZ L35F8
        EX DE,HL        ; neue Zieladresse
        POP HL          ; neue Tabellenadresse fuer naechste
        RET             ;       4 Bytes
INIGRP: LD  HL,00H      ; ******* Schirm fuer Grafik 1
        LD (CSRY),HL    ;       Initialisieren
        LD  HL,GRPNAM   ; Cursor position 0,0
        CALL SETWRT     ; ab Adresse 1800H
        XOR A           ; Grafik im VDP
        LD B,03H
L361F:
        OUT (080H),A
        INC A
        JR NZ,L361F     ; 256*0 schreiben
        DJNZ L361F      ; 3*256*0
        CALL CLSHRS
        LD  A,(VDPReg0)
        OR 02H          ; M3=1
        LD (VDPReg0),A
        LD B,A
        LD  C,00H
        CALL VDPWRT     ; reg 0 auf grafik 1
        LD  A,(RG1SAV)
        AND 0E7H        ; M1=0  M2=0  M3=1
        LD (RG1SAV),A
        LD B,A
        LD  C,01H
```



```
        CALL VDPWRT     ; reg 1
        LD  BC,0602H
        CALL VDPWRT     ; reg 2 Namensbasis=0
        LD  BC,0FF03H
        CALL VDPWRT     ; reg 3 Farbbasis=2000H
        LD  BC,0304H
        CALL VDPWRT     ; reg 4 Maskenbasis=1800H
        LD  BC,03605H
        CALL VDPWRT     ; reg 5 Spriteattribut=1B00H
        LD  BC,0706H
        CALL VDPWRT     ; reg 6 Spritemasken=3800H
        JR CLRSPR
INIMLT: LD  HL,00H      ; ****** Screen 2 initialisieren
        LD (CSRY),HL
        LD  A,(VDPReg0)
        AND 01H
        LD (VDPReg0),A
        LD B,A
        LD  C,00H
        CALL VDPWRT
        LD  A,(RG1SAV)
        AND 0E7H
        OR 08H          ; M1=0  M2=1  M3=0
        LD (RG1SAV),A
        LD B,A
        LD  C,01H
        CALL VDPWRT
        LD  BC,0202H    ; name=800H
        CALL VDPWRT
        LD  BC,04H      ; Masken=0
        CALL VDPWRT
        LD  BC,03605H   ; Spriteattribute=1B00H
        CALL VDPWRT
        LD  BC,0706H    ; Spritemaske=3800H
        CALL VDPWRT
        LD  HL,MLTNAM/TXTCGP
        CALL SETWRT     ; schreiben ab 800H
        LD  DE,06H      ; 6*4*256
L36AA:
        LD  C,04H       ; 4*256
L36AC:
        LD A,D
        LD B,020H
L36AF:
        OUT (080H),A    ; Codes 0-255
        INC A
        DJNZ L36AF      ; 256*
        DEC C
        JR NZ,L36AC
        LD D,A
        DEC E
        JR NZ,L36AA
        CALL L37B7
CLRSPR: LD  A,(RG1SAV)  ; ***** Sprites loeschen
        AND 0FCH
        LD  HL,SPRSIZ
        OR (HL)
        LD (RG1SAV),A
        LD B,A
```

```
        LD  C,01H
        CALL VDPWRT     ; Spritegroesse erneuern
        LD  HL,SPRPAT   ; immer 3800H
        CALL SETWRT     ; zum Schreiben vorbereiten
        LD  BC,800H     ; Bytes fuer Sprites
L36D9:
        XOR A
        OUT (080H),A    ; Spritemuster auf 0 setzen
        DEC BC
        LD A,B
        OR C
        JR NZ,L36D9
L36E1:                  ; Farbe aller Sprites auf Hintergrund
        LD  A,(front/COLOR)     ; und Sprites unsichtbar machen
        LD E,A
        LD  HL,SPRATR   ; 1B00H
        LD  BC,02000H
L36EB:                  ; Farbe fuer alle Spritebytes auf
        LD  A,0D1H      ; Vordergrundfarbe setzen
        CALL WRTVDP     ; Vertikalposition y=209 d.h. unsichtbar
        INC HL
        INC HL
        LD A,C
        CALL WRTVDP     ; Basisadresse fuer Spritemuster
        INC HL
        INC C
        LD  A,(SPRSIZ)
        RRCA
        RRCA
        JR NC,L3702     ; jp falls Spritegroesse 8 Bytes
        INC C           ; sonst 3*8 weitere Bytes reservieren
        INC C
        INC C
L3702:
        LD A,E          ; Farbe
        CALL WRTVDP     ; des Sprites
        INC HL
        DJNZ L36EB      ; fuer 32 Sprites
        RET
L370A:                  ; ********* Loeschen des Bildschirms
        LD  HL,00H      ;       evt. Keys einblenden
        CALL SETWRT     ; Adresse 0 setzen
        LD  BC,03C0H    ; 960 Zeichen
L3713:
        XOR A
        OUT (080H),A    ; auf null setzen
        DEC BC
        LD A,B
        OR C
        JR NZ,L3713
        CALL CSHOME     ; cursor home
        LD  HL,LINTTB   ; newlinetable initialisieren
        LD B,018H
L3723:
        LD (HL),B       ; 24-1
        INC HL
        DJNZ L3723
        JP FNKSB
WRTVDP: 372A            ; ********* HL als Adresse zum VDP
        PUSH AF         ;   und A in Videospeicher schreiben
        CALL SETWRT
```



```
        EX (SP),HL
        EX (SP),HL
        POP AF
        OUT (080H),A
        RET
RDVDP:  3734            ; ********* Byte vom Videospeicher
        CALL SETRD      ;    Adresse HL nach A lesen
        EX (SP),HL
        EX (SP),HL
        IN A,(084H)
        RET
SETWRT: 373C            ; ********* HL als Videoadresse setzen
        LD A,L          ;       und zum schreiben vorbereiten
        DI
        OUT (081H),A    A => VDP
        LD A,H
        OR 040H = 64
        OUT (081H),A
        EI
        RET
SETRD:  3747            ; ********* HL als Videoadresse setzen
        LD A,L          ;       und zum lesen vorbereiten
        DI
        OUT (081H),A
        LD A,H
        OUT (081H),A
        EI
        RET
CHGCLR: 3750            ; ******* Farbe wechseln im Textmodus
        LD  A,(SCREEN)  ;         bzw. Randfarbe im Grafikmodus
        AND A
        LD  A,(BORCLR)
        JR NZ,L3762
L3759:                  ; farbe fuer Textmodus aendern
        LD HL,(front/COLOR)
        LD A,L
        ADD A,A         ; Vordergrundfarbe um 4 bit shiften
        ADD A,A
        ADD A,A         ; p=Bildpunkt  h=Hintergrundfarbe
        ADD A,A         ; ppppphhh
        OR H
L3762:
        LD B,A
        LD  C,07H
        JP VDPWRT       ; setzen register 7 Text/(Grafikrand)
TOTEXT: 3768            ; ******* einschalten Textmodus
        LD  A,(SCREEN)
        AND A
        RET Z           ; ist schon eingeschaltet!
        XOR A
        LD (SCREEN),A
        CALL LFFB1
        JP INITXT       ; falls nicht, initialisieren
CLS:    3777            ; ************************************
        RET NZ
        PUSH HL
        CALL L377E
        POP HL
        RET
L377E:
```

```
        LD  A,(SCREEN)
        DEC A
        JR Z,CLSHRS     ; bei screen 1
        JP P,L37B7      ; bei screen 2
        LD  A,0CH       ; bei screen 0 nur Form-Feed ausgeben
        RST 18H
        RET
CLSHRS: / 3788          ; ***** Loeschen des hochaufloesenden
        LD  A,(BORCLR)  ;       Bildschirmes
        CALL L3762      ; Randfarbe setzen
        LD  BC,GRPNAM   ; Grafik-Namen
        PUSH BC
        LD  HL,GRPCCL   ; 2000H-37FF auf Hintergrundfarbe
        CALL SETWRT
L379B:
        LD  A,(back/COLOR)
        OUT (080H),A
        DEC BC
        LD A,B
        OR C
        JR NZ,L379B
        LD  HL,00H
        POP BC
        CALL SETWRT     ; 0-17FFH auf 0
L37AC:
        XOR A
        OUT (080H),A
        DEC BC
        LD A,B
        OR C
        JR NZ,L37AC
        JP L36E1        ; Sprites unsichtbar und Farbe loeschen
L37B7:                  ; ******** loeschen des SCREEN 2
        LD  A,(BORCLR)  ;       Bildschirms
        CALL L3762      ; Randfarbe
        LD  HL,00H
        LD  BC,0600H
        CALL SETWRT
        LD  HL,back/COLOR  ; 0-5FF auf Hintergrundfarbe setzen
L37C9:
        LD A,(HL)
        ADD A,A
        ADD A,A
        ADD A,A
        ADD A,A
        OR (HL)
        OUT (080H),A            ; pppphhhh
        DEC BC
        LD A,B
        OR C
        JR NZ,L37C9
        JP L36E1                ; Spritefarbe-und Position

CHGMOD: / 37D9          ; ***** initialisieren des Bildschirms
        LD  A,(SCREEN)  ;       je nach Screeninhalt
        DEC A
        JP M,INITXT
        JP Z,INIGRP
        JP INIMLT

SAVSCN: / 37E6          ; ***** csave ...,S   save "CAS:...",S
```



```
        CALL CWRTON     ; CAS zum Schreiben des Bildschirms
        LD  A,(SCREEN)  ;       vorbereiten
        DEC A
        JP M,L3823      ; scr 0
        JR NZ,L3806     ; scr 2
        LD  HL,00H      ; screen 1 speichern
        LD  BC,1800H
        CALL L38DD      ; 0-17FFH speichern     Muster
        LD  HL,GRPCCL
        LD  BC,1800H
        CALL L38DD      ; und 2000H-37FFH       Farbe
        JR L380F
L3806:
        LD  HL,00H
        LD  BC,0600H
        CALL L38DD      ; 0-5FFH                Farben
L380F:
        LD  HL,SPRATR
        LD  BC,080H
        CALL L38DD      ; 1B00H-1B7FH   Spriteattribute
        LD  HL,SPRPAT
        LD  BC,800H     ; 3800H-3FFFH   Spritemuster
        CALL L38DD
        JR L382C
L3823:
        LD  HL,00H
        LD  BC,03C0H
        CALL L38DD      ; 0-3BFH        Textspeicher
L382C:
        CALL CTWOFF
        POP HL
        RET
LODSCN: / 3831          ; ****** Laden von Kasettenscreens
        LD  A,(CASATR)
        LD (SCREEN),A
        CALL CHGMOD     ; setzen des gespeicherten Bildschirms
        CALL CSROON
        LD  A,(SCREEN)
        DEC A
        JP M,L3877
        JR NZ,L385A
        LD  HL,00H      ; Lesen von SCREEN 1
        LD  BC,1800H
        CALL L38CF      ; Muster
        LD HL,GRPCCL
        LD  BC,1800H
        CALL L38CF      ; Farbe
        JR L3863
L385A:
        LD  HL,00H
        LD  BC,0600H
        CALL L38CF      ; Farben
L3863:
        LD  HL,SPRATR
        LD  BC,080H
        CALL L38CF      ; Spriteattribute
        LD  HL,SPRPAT
        LD  BC,800H
        CALL L38CF      ; Spritemuster
        JR L3880
L3877:
```

```
        LD  HL,00H
        LD  BC,03C0H
        CALL L38CF      ; textbildschirm
L3880:
        CALL CTOFF
        LD HL,(TXPSAV)
        RET
VRFSCN:/ 3887           ; ****** Verify des Bildschirms
        CALL CSROON
        LD  A,(SCREEN)
        DEC A
        JP M,L38C4
        JR NZ,L38A7
        LD  HL,00H
        LD  BC,1800H
        CALL L38EB
        LD  HL,GRPCCL
        LD  BC,1800H
        CALL L38EB
        JR L38B0
L38A7:
        LD  HL,00H
        LD  BC,0600H
        CALL L38EB
L38B0:
        LD  HL,SPRATR
        LD  BC,080H
        CALL L38EB
        LD  HL,SPRPAT
        LD  BC,800H
        CALL L38EB
        JR L3880
L38C4:
        LD  HL,00H
        LD  BC,03C0H
        CALL L38EB
        JR L3880
L38CF:                  ; ****** Videobereich lesen von CAS
        CALL L3905      ; Adresse setzen zum Lesen ohne DI/EI
L38D2:
        CALL CASIN
        OUT (080H),A
        DEC BC
        LD A,B
        OR C
        JR NZ,L38D2
        RET
L38DD:                  ; ****** Videobereich schreiben auf CAS
        CALL L390E
L38E0:
        IN A,(084H)
        CALL CASOUT
        DEC BC
        LD A,B
        OR C
        JR NZ,L38E0
        RET
L38EB:                  ; ***** Videospeicher gegen CAS
        CALL L390E      ;       verifizieren
        IN A,(084H)     ; HL Startadresse BC Anzahl
        LD E,A
```



```
        CALL CASIN
        CP E
        JR NZ,L38FD
        DEC BC
        LD A,B
        OR C
        JR NZ,L38E0
        RET
L38FD:                  ; verify error
        CALL CTOFF
        LD  E,014H
        JP ERROR
L3905:                  ; ******* Setzen einer Videoadresse fuer
        LD A,L          ;         nachfolgendes Schreiben ohne
        OUT (081H),A    ;         DI/EI
        LD A,H
        OR 040H
        OUT (081H),A
        RET
L390E:                  ; ******* setzen der Videoadresse fuer
        LD A,L          ;         folgendes Lesen
        OUT (081H),A
        LD A,H
        OUT (081H),A
        RET
CHPLPT:/ 3915
        PUSH AF
L3916:
        CALL BREAKX     ; test ctrl-stop
        JR C,L392C      ; C= Unterbrechung
        CALL CHPSTT     ; Status
        JR Z,L3916      ; NZ =bereit
        POP AF
L3921:                  ; ********** Zeichen zur Centronics-
        PUSH AF         ;       schnittstelle senden
        OUT (010H),A
        XOR A
        OUT (011H),A
        DEC A
        OUT (011H),A
        POP AF
        RET
L392C:                  ; Drucken eines Zeichens unterbrochen!
        XOR A
        LD (LPOS),A     ; Ruecksetzen
        LD  A,0DH
        CALL L3921      ; CR ausgeben
        JP DIOERR       ; dev I/O error
CHPSTT:/ 3938           ; ********** Status der Centronics
        IN A,(012H)     ; NZ Geraet bereit
        RRCA
        CCF
        SBC A,A
        RET
POSIT:/ 393E            ; ******* Cursor positionieren
        LD  A,01BH      ; L     Zeile 1-24
        RST 18H         ; H     Spalte 1-80
        LD  A,'Y'
        RST 18H
        LD A,L
        ADD A,01FH
```

```
        RST 18H
        LD A,H
        ADD A,01FH
        RST 18H
        RET
CHPUT:/ 3940            ; ****** Ausgabe eines Zeichens
        PUSH HL         ;   auf den Bildschirm in jedem Modus
        PUSH DE
        PUSH BC
        PUSH AF
        LD  A,(SCREEN)
        AND A
        JP NZ,GRPPRT
        CALL LFFAE
        CALL CKERCS
        POP AF
        PUSH AF
        CALL L396B
        CALL CXDPCS
POPALL:/ 3966
        POP AF
PBDHRT:/ 3967
        POP BC
        POP DE
        POP HL
        RET
L396B:                  ; ******** Bildschirmtreiber 40 Zeichen
        LD C,A
        LD  HL,ESCCNT   ; >0 ESC gewesen
        LD A,(HL)
        AND A
        JP NZ,L3A1B
        LD A,C
        CP ' '
        JR C,L399A      ; Steuerzeichen
        LD HL,(CSRY)
        CP 07FH
        JP Z,L3B55      ; del
        CALL PUTCOD     ; ASCII-Zeichen ausgeben
        CALL L3AB9      ; Versetzen des Cursors
        RET NZ
        XOR A
        CALL SETTRM
        LD H,01H
L398E:
        CALL L3AD6      ; Cursor rechts
        RET NZ          ; ret falls kein Zeilenende
        CALL L3ADE      ; Cursor setzen
        LD  L,01H
        JP DELLNO       ; Zeile 0 loeschen
L399A:                  ; alle Steuerzeichen ausser del
        LD  HL,L39B5-2
        LD  C,12        ; anzahl Steuerzeichen
INDJMP:/ 399F
        INC HL
        INC HL
        AND A
        DEC C
        RET M           ; ret mit M Tabellenende
        CP (HL)         ; vergleich mit Empfangenem Zeichen
        INC HL
```



```
        JR NZ,INDJMP    ; NZ weitersuchen
        LD C,(HL)
        INC HL
        LD B,(HL)       ; Ausfuehrungsadresse in BC
        LD HL,(CSRY)    ; Cursor
        CALL L39B3
        XOR A           ; Steuerzeichen ausgegeben
        RET
L39B3:
        PUSH BC
        RET

L39B5:
        DEFB 7
        DEFW L40BE
        DEFB 8
        DEFW L3AC1
        DEFB 9
        DEFW L3AE6
        DEFB 10
        DEFW L398E
        DEFB 11
        DEFW CSHOME
        DEFB 12
        DEFW L370A
        DEFB 13
        DEFW L3AF6
        DEFB 27
        DEFW L3A15
        DEFB 28
        DEFW ADVCUR
        DEFB 29
        DEFW L3AC1
        DEFB 30
        DEFW L3ACC
        DEFB 31
        DEFW L3AD6

L39D9:
        DEFB 'j'
        DEFW L370A
        DEFB 'E'
        DEFW L370A
        DEFB 'K'
        DEFW ECL
        DEFB 'J'
        DEFW L3B76
        DEFB 'l'
        DEFW L3B5E
        DEFB 'L'
        DEFW L3B29
        DEFB 'M'
        DEFW L3AFA
        DEFB 'Y'
        DEFW L3A12
        DEFB 'A'
        DEFW L3ACC
        DEFB 'B'
        DEFW L3AD6
        DEFB 'C'
```

```
        DEFW L3AB9
        DEFB 'D'
        DEFW L3ACA
        DEFB 'H'
        DEFW CSHOME
        DEFB 'p'
        DEFW L3A66
        DEFB 'q'
        DEFW L3A67
        DEFB 'x'
        DEFW L3AOC
        DEFB 'y'
        DEFW L3AOF

L3AOC:          ; ESC x
        LD A,1
        DEFB 1                  ;*LD  BC,023EH
L3AOF:          ; ESC y
        LD A,2                  ;*
        DEFB 1                  ;*      LD  BC,043EH
L3A12:          ; ESC Y
        LD A,4                  ;*
        DEFB 1                  ;*      LD  BC,OFF3EH
L3A15:          ; ESC ausfuehren

        LD A,OFFH               ;*
        LD (ESCCNT),A
        RET
L3A1B:                  ; ESC gewesen
        JP P,L3A29
        LD (HL),00H     ; 1. Zeichen nach ESC
        LD A,C
        LD  HL,L39D9-2  ; Tabelle fuer ESC Sequenzen
        LD  C,17
        JP INDJMP
L3A29:
        DEC A
        JR Z,L3A4A
        DEC A
        JR Z,L3A54
        DEC A
        LD (HL),A       ; 0 falls vorher 3 sonst 1
        LD  A,(LINLEN)
        LD  DE,CSRX     ; Cursor nach ESC Y Zeile Spalte
        JR Z,L3A3F      ; setzen        0-79/0-23
        LD (HL),03H     ; auf 3 fuer Spalte
        CALL GETLEN     ; Zeilenzahl holen
        DEC DE
L3A3F:
        LD B,A
        LD A,C
        SUB ' '         ; offset abziehen
        CP B            ; Vergleich gegen groesste Zeile/Spalte
        INC A           ; +1 fuer 1-24/1-80
        LD (DE),A       ; setzen falls kleiner
        RET C
        LD A,B
        LD (DE),A       ; max wert falls falsche Angabe
        RET
L3A4A:                  ; ESC x ausfuehren
        LD (HL),A       ; ESCcount=0
```



```
        LD A,C
        SUB '4'
        JR Z,L3A5B      ; Cursormodus 0
        DEC A
        JR Z,L3A62      ; oder Cursorswitch 0
        RET
L3A54:                  ; ESC y ausfuehren
        LD (HL),A
        LD A,C
        SUB '4'
        JR NZ,L3A5F     ; Cursorswitch aendern
        INC A           ; Cursormodus 1
L3A5B:
        LD (CSTYLE),A   ; 0 oder 1
        RET
L3A5F:
        DEC A
        RET NZ
        INC A           ; ESC y 5   Cursorswitch 1
L3A62:
        LD (CSRSW),A    ; Cursorswitch 0 oder 1
        RET
L3A66:                  ; ESC p invers einschalten
        DEFB 03EH       ;*      LD  A,0AFH
L3A67:                  ; ESC q invers abschalten
        XOR A           ;*
        LD (REVFLG),A
        RET
CXDPCS: /3A6C           ; ******* Cursor an x,y setzen
        LD  A,(CSRSW)
        AND A
        RET Z           ; falls switch>0
DSPCSR: / 3A71
        PUSH AF
        LD HL,(CSRY)    ; Cursoradresse x in H y in L
        PUSH HL
        CALL GETVRM     ; Videocode holen
        LD (CODSAV),A   ; und sichern sichern
        LD L,A
        LD H,00H
        ADD HL,HL       ; *2
        ADD HL,HL       ; *4
        ADD HL,HL       ; *8
        LD  DE,800H
        ADD HL,DE       ; + BASIS
        CALL L3BFF      ; Videorammuster lesen
        LD  HL,OFDBBH
        LD B,08H        ; Cursor voll,
        LD  A,(CSTYLE)
        AND A
        JR Z,L3A96
        LD B,03H        ; oder halb hoch
L3A96:
        LD A,(HL)
        CPL             ; Muster invertieren
        LD (HL),A
        DEC HL
        DJNZ L3A96
        LD  HL,0DF8H
        CALL L3C1C      ; Muster  als Cursorcodemuster ausgeben
        POP HL
```

```
        LD  C,0BFH      ; Cursorcode
        JR L3AB4        ; setzen
CKERCS: 3AA7            ; ********** Cursor loeschen
        LD  A,(CSRSW)
        AND A
        RET Z           ; falls Cursorswitch > 0
ERACSR: 3AAC
        PUSH AF
        LD HL,(CSRY)
        LD  A,(CODSAV)  ; alter Code vor Cursor
        LD C,A
L3AB4:
        CALL PUTVRM     ; auf Cursorposition setzen
        POP AF
        RET
L3AB9:                  ; **** ESC C    Cursor rechts
        LD  A,(LINLEN)
        CP H
        RET Z           ; Z falls Zeilenende
        INC H
        JR L3ADE
BS:     3AC1            ; Backspace
        CALL L3ACA      ; nach links
        RET NZ
        LD  A,(LINLEN)  ; falls Zeilenanfang auf Vorzeile
        LD H,A          ; solange nicht Zeile 1
        DEFB 011H               ;* LD DE,03E25H
L3ACA:                  ; ESC D         Cursor links
        DEC H                   ;*
        DEFB 03EH               ;* LD A,02DH
L3ACC:                  ; ESC A oder RS Cursor unten
        DEC L                   ;*
        RET Z
        JR L3ADE
ADVCUR: 3AD0            ; FS Cursor rechts
        CALL L3AB9
        RET NZ
        LD H,01H        ; falls Zeilenende
L3AD6:
        CALL GETLEN     ; Zeilenzahl holen
        CP L
        RET Z           ; Z falls letzte Zeile
        JR C,L3AE2
        INC L           ; naechste Zeile
L3ADE:
        LD (CSRY),HL
        RET
L3AE2:                  ; falls Zeile zu gross
        DEC L
        XOR A
        JR L3ADE
L3AE6:                  ; TAB
        LD  A,' '
        CALL L396B      ; Space ausgeben
        LD  A,(CSRX)
        DEC A
        AND 07H         ; 8 ter-Grenze
        JR NZ,L3AE6     ; falls nicht, Space
        RET
CSHOME: 3AF4            ; ESC H oder VT
        LD  L,01H
```



```
L3AF6:                  ; CR cursor to start of line
        LD H,01H
        JR L3ADE
L3AFA:                  ; ESC M
        CALL L3AF6
DELLNO: 3AFD            ; Loeschen einer Zeile
        CALL GETLEN
        SUB L
        RET C           ; Zeilenzahl-act y
        JP Z,L3B5E
        PUSH HL
        PUSH AF
        LD C,A
        LD B,00H
        CALL GETTRM
        LD L,E
        LD H,D
        INC HL
        LDIR            ; Zeilenlaengen verschieben
        LD  HL,FSTPOS
        DEC (HL)
        POP AF
        POP HL
L3B18:
        PUSH AF
        INC L
        CALL L3C04
        DEC L
        CALL L3C21
        INC L
        POP AF
        DEC A
        JR NZ,L3B18
        JP L3B5E
L3B29:                  ; ESC L
        CALL L3AF6
INSLNO: 3B2C            ; Zeile einfuegen
        CALL GETLEN
        LD H,A
        SUB L
        RET C
        JP Z,L3B5E
        LD L,H
        PUSH HL
        PUSH AF
        LD C,A
        LD B,00H
        CALL GETTRM
        LD L,E
        LD H,D
        DEC HL
        LDDR
        POP AF
        POP HL
L3B45:
        PUSH AF
        DEC L
        CALL L3C04
        INC L
        CALL L3C21
        DEC L
```

```
        POP AF
        DEC A
        JR NZ,L3B45
        JR L3B5E
L3B55:                  ; DEL
        CALL BS
        RET Z
        LD  C,00H
        JP PUTVRM
L3B5E:                  ; ESC 1 Zeile loeschen
        LD H,01H
ECL:    3B60            ; ESC K bis Zeilenende loeschen
        CALL TERMIN
        PUSH HL
        CALL L3C8A
        CALL SETWRT
        POP HL
L3B6B:
        XOR A
        OUT (080H),A
        INC H
        LD  A,(LINLEN)
        CP H
        JR NC,L3B6B
        RET
L3B76:                  ; ESC J bis ende der Seite loeschen
        PUSH HL
        CALL ECL
        POP HL
        CALL GETLEN
        CP L
        RET C
        RET Z           ; ret falls seitenende
        LD H,01H
        INC L
        JR L3B76
ERAFNK: 3B86            ; Zeile 24 loeschen
        CALL LFF4E
        XOR A
        LD (CNSDFG),A
        PUSH HL
        LD  L,24
        CALL L3B5E
        POP HL
        RET
FNKSB:  3B95            ; ****** Keys anzeigen
        LD  A,(CNSDFG)
        AND A
        RET Z
        LD  A,(SCREEN)  ; falls Keys eingeschaltet
        AND A
        RET NZ
DSPFNK: 3B9F            ; und SCREEN 0
        CALL LFF51
        PUSH HL
        LD  A,0FFH
        LD (CNSDFG),A
        LD  A,(CSRY)
        CP 24
        LD  A,0AH
        CALL Z,OUTCHAR  ; LF ausgeben falls Cursor in Zeile 24
```



```
        LD  HL,REVFLG
        LD A,(HL)
        PUSH AF
        PUSH HL
        LD (HL),01H     ; Inversflag setzen
        LD  A,(SHCTRL)
        RRCA
        LD  HL,FNKSTR
        LD  A,01H
        JR C,L3BC9      ; C falls keys 1-5
        LD  HL,FNKSTR+5*16      ; 0FA6EH
        XOR A
L3BC9:
        LD (FNKSWI),A   ; 0 keys 6-10   1 keys 1-5
        LD  DE,LINWRK
        LD  A,(LINLEN)
        CP 40
        JR Z,L3BD7      ; Z falls Zeilenlaenge 40
        DEC DE
L3BD7:
        LD  C,05H       ; 5 keys
L3BD9:
        LD B,07H        ; je in 7 Zeichen
        XOR A
        LD (DE),A       ; erst ein Space vorweg
        INC DE
L3BDE:
        PUSH BC
        LD C,(HL)       ; key Zeichen holen
        INC HL
        PUSH DE
        CALL CNVCOD     ; in Videocode konvertieren
        POP DE
        LD A,C
        LD (DE),A       ; und ausgeben
        INC DE
        POP BC
        DJNZ L3BDE      ; Key noch nicht fertig
        LD A,C
        LD  BC,09H
        ADD HL,BC       ; naechste Keypuffer-Adresse
        LD C,A
        DEC C
        JR NZ,L3BD9     ; Keys noch nicht fertig
        LD  L,24
        CALL L3C21      ; Zeile 24 schreiben
        POP HL
        POP AF
        LD (HL),A
        POP HL
        RET
L3BFF:                  ; ****** Zeichenmuster aus VRAM
        PUSH HL         ;        nach LINWRK lesen
        LD B,08H
        JR L3C0E
L3C04:                  ; ***** Zeile aus VRAM lesen (Videocode)
        PUSH HL
        LD H,01H
        CALL L3C8A
        LD  A,(LINLEN)
        LD B,A
```

```
L3C0E:                  ; ****** B VRAM bytes ab HL lesen
        CALL SETRD
        LD  HL,LINWRK
L3C14:
        IN A,(084H)
        LD (HL),A
        INC HL
        DJNZ L3C14
        POP HL
        RET
L3C1C:                  ; ****** Zeichenmuster aus LINWRK
        PUSH HL         ;        ins VRAM schreiben
        LD B,08H
        JR L3C2B
L3C21:                  ; ****** Zeile LINWRK ab Position (HL)
        PUSH HL         ;        ins VRAM schreiben
        LD H,01H
        CALL L3C8A
        LD  A,(LINLEN)
        LD B,A
L3C2B:
        CALL SETWRT
        LD  HL,LINWRK
L3C31:
        LD A,(HL)
        OUT (080H),A
        INC HL
        DJNZ L3C31
        POP HL
        RET
GETCOD: 3C39            ; ****** ASCII-Zeichen vom
        CALL GETVRM     ;        Bildschirm lesen
        CP 060H
        JR C,L3C49
        SUB 060H
        CP 060H
        JR C,L3C49
        ADD A,040H
        RET
L3C49:
        ADD A,020H
        RET
GETVRM: 3C9C            ; ****** Zeilen/Spalten adresse in HL
        PUSH HL         ; in Videoadresse wandeln und Videocode
        CALL L3C8A      ; lesen
        CALL SETRD
        EX (SP),HL
        EX (SP),HL
        IN A,(084H)
        LD C,A
        POP HL
        RET
CNVCOD: 3CSA            ; ******* Wandlung ASCII --> Videocode
        LD A,C
        SUB 020H
        JR C,L3C6F
        CP 05FH
        JR C,L3C70
        CP 080H
        JR C,L3C6F
        CP 0C0H
```



```
        JR NC,L3C6F
        ADD A,040H
        LD C,A
        RET
L3C6F:
        XOR A
L3C70:
        LD C,A
        LD  A,(REVFLG)
        AND A
        RET Z
        LD A,C
        ADD A,060H
        LD C,A
        RET
PUTCOD: 3C7B            ;***** ASCIIzeichen auf Cursorposition
        CALL CNVCOD     ;      HL (Spalte/Zeile) position
PUTVRM: 3C7E            ;      schreiben
        PUSH HL
        CALL L3C8A
        CALL SETWRT
        LD A,C
        OUT (080H),A
        POP HL
        RET
L3C8A:
        LD A,L
        LD E,H
        LD H,00H
        LD D,H
        ADD A,A ; *2
        ADD A,A
        ADD A,L ; *5
        ADD A,A ; *10
        ADD A,A
        RL H    ; *20
        ADD A,A
        RL H    ; *40
        LD L,A  ; HL=ZEILE*40
        ADD HL,DE       ; +Spalte
        LD  DE,0FFD7H   ; -41 wegen 1-24/1-80
        ADD HL,DE
        LD  A,(LINLEN)
        CP 40
        RET Z
        INC HL
        RET
GETTRM: 3CA7            ; Zeilenbeschreibung aus Tabelle fuer
        PUSH HL         ; Zeile L holen
        LD  DE,LINTTB-1
        LD H,00H
        ADD HL,DE
        LD A,(HL)
        EX DE,HL        ; Tabellenadresse nach DE
        POP HL
        AND A
        RET
TERMIN: 3CB3            ; Zeilenbeschreibung auf AF setzen
        DEFB 03EH       ;*      LD  A,0AFH
UNTERM: 3CB4            ; Zeilenbeschreibung auf 0 setzen
        XOR A           ;*
```

```
SETTRM: 3CB5            ; Zeilenbeschreibung auf A setzen
        PUSH AF
        CALL GETTRM
        POP AF
        LD (DE),A
        RET
GETLEN: 3CBC            ; Bildschirmzeilenzahl holen
        LD  A,(CNSDFG)  ; 0 odef OFFH
        ADD A,18H
        RET
KEYINT: 3CC2            ; ****** INTERRUPT-Bearbeitung
        PUSH HL
        PUSH DE
        PUSH BC
        PUSH AF
        CALL LFE79
        LD  A,(MDMFLG)
        AND A
        JR Z,L3CD5      ; Z falls kein Modem vorhanden
        IN A,(022H)
        RRCA
        JP NC,RS2INT
L3CD5:
        IN A,(085H)
        AND A
        JP P,INTRET     ; P falls kein VDP (Timer)-Interrupt
        EI
        LD (STATFL),A   ; Status sichern
        AND 020H
        LD  HL,SPRITE_OOS
        CALL NZ,REQTRP  ; NZ falls Spriteueberlappung
        LD  HL,TRGFLG
        IN A,(098H)
        AND 30H         ; Trigger maskieren
        LD C,A
        XOR (HL)        ; alter Inhalt
        AND (HL)
        LD (HL),C       ; neuen Inhalt speichern
        ADD A,A
        ADD A,A
        PUSH AF
        LD  HL,OFE15H
        CALL M,REQTRP   ; M falls Joystick 2 trigger
        POP AF
        ADD A,A
        LD  HL,OFE12H
        CALL M,REQTRP   ; M falls joystick 1
        IN A,(09AH)
        AND OFOH
        ADD A,08H
        OUT (096H),A
        IN A,(099H)
        AND 01H
        LD C,A
        LD  HL,SPCFLG
        XOR (HL)
        AND (HL)
        LD (HL),C
        LD  HL,STRIG_OOS
        CALL NZ,REQTRP  ; NZ falls Spacetaste
        LD HL,(INTCNT)
```



```
        DEC HL
        LD A,H
        OR L
        JR NZ,L3D2D     ; Z falls INTERVAL abgelaufen
        LD  HL,INTERVAL_OOS
        CALL REQTRP
        LD HL,(INTVAL)
L3D2D:
        LD (INTCNT),HL
        LD HL,(TIMEVAR)
        INC HL          ; TIME weiterzaehlen
        LD (TIMEVAR),HL
        CALL LFF5A
        IN A,(098H)
        ADD A,A
        JP P,L3D44      ; P falls CAS READY
        LD  A,08H
        OUT (097H),A    ; CAS ON wenn keine Taste gedrueckt
L3D44:
        LD  A,(MUSIKF)
        LD C,A
        XOR A
L3D49:
        RR C            ; Queue nicht leer
        PUSH AF         ; A=Generatornummer
        PUSH BC
        CALL C,L40E5    ; C falls ein Ton zu spielen ist
        POP BC
        POP AF
        INC A
        CP 03H
        JR C,L3D49
        LD  HL,SCNCNT
        DEC (HL)
        JR NZ,INTRET    ; falls Z, Keyboard abfragen
        LD (HL),03H
        XOR A
        LD (CLIKFL),A   ; CLICK loeschen
        CALL L3D86      ; SCAN Keyboard
        JR NZ,INTRET    ; NZ falls keine Taste hinzugekommen
        LD  HL,REPCNT
        DEC (HL)
        JR NZ,INTRET    ; Autorepeatzeit abgelaufen, falls Z
        LD (HL),01H
        LD  HL,OLDKEYS
        LD  DE,OLDKEYS+1
        LD  BC,0AH
        LD (HL),OFFH    ; alten Keyspeicher loeschen
        LDIR            ; um autorepeat zuzulassen
        CALL L3DAE
INTRET: 3D80
        POP AF
        POP BC
        POP DE
        POP HL
        EI
        RET
L3D86:
        IN A,(09AH)
        AND OFOH        ; CAS signale nicht veraendern
        LD C,A
```

```
        LD B,0BH        ; 11 Zeilen
        LD  HL,ACTKEYS
L3D90:
        LD A,C
        OUT (096H),A    ; Spalte
        IN A,(099H)     ; Zeile
        LD (HL),A       ; Speichern
        INC C
        INC HL
        DJNZ L3D90
        LD  DE,ACTKEYS  ; HL auf alten Matrix codes
        LD B,0BH
L3D9F:
        DEC DE
        DEC HL
        LD  A,(DE)
        CP (HL)         ; Vergleich neu alt
        JR NZ,L3DA9     ; Veraenderung
        DJNZ L3D9F
        JR L3DAE        ; keine Veraenderung
L3DA9:
        LD  A,0DH
        LD (REPCNT),A   ; Autorepeatzeit neu setzen
L3DAE:
        LD B,0BH
        LD  HL,OLDKEYS
        LD  DE,ACTKEYS
L3DB6:
        LD  A,(DE)
        LD C,A
        XOR (HL)
        AND (HL)
        LD (HL),C
        CALL NZ,L3DE9   ; nur die Bits der Zeilen gesetzt
        INC DE          ; die inzwischen hinzugekommen sind
        INC HL
        DJNZ L3DB6
L3DC2:                  ; ****** NZ falls Zeichen in
        LD HL,(GETPNT)  ;        Tastaturpuffer
        LD  A,(PUTPNT)
        SUB L
        RET
CHSNS:  3DCA
        PUSH HL
        PUSH DE
        PUSH BC
        LD  A,(SCREEN)
        AND A
        JR NZ,L3DE2
        LD  A,(FNKSWI)  ; screen 0
        LD  HL,SHCTRL
        XOR (HL)
        LD  HL,CNSDFG
        AND (HL)
        RRCA
        CALL C,DSPFNK   ; Keys abhaengig von shift anzeigen
L3DE2:                  ;       falls SCREEN 0,1
        CALL L3DC2
        POP BC
        POP DE
        POP HL
```



```
        RET
L3DE9:                  ; Tastennummer in der Matrix berechnen
        PUSH HL
        PUSH DE
        PUSH BC
        PUSH AF
        LD  A,0BH
        SUB B
        ADD A,A
        ADD A,A
        ADD A,A
        LD C,A
        LD B,08H
        POP AF
L3DF7:
        RRA
        PUSH BC
        PUSH AF
        CALL C,L3EAE    ; Carry falls Taste gedrueckt
        POP AF          ; in C die Tastennummer
        POP BC
        INC C
        DJNZ L3DF7
        JP PBDHRT

L3E05:
        DEFB 0AH
        DEFW L3EC3
        DEFB 011H
        DEFW L3F10
        DEFB 02BH
        DEFW L3EE3
        DEFB 02CH
        DEFW L3F29
        DEFB 02EH
        DEFW L3F3D
        DEFB 035H
        DEFW L3FD8
        DEFB 036H
        DEFW L3FFB
        DEFB 038H
        DEFW L3FD8
        DEFB 03DH
        DEFW L3F92
        DEFB 03EH
        DEFW L3FCB
        DEFB 043H
        DEFW L3FD8
        DEFB 044H
        DEFW L3FE7
        DEFB 0FFH
        DEFW L3FD8

L3E2C:          ; fuer shift 0-9
        DEFM ')!@#$%^&*('

L3E36:                  ; fuer A-Z
        DEFW L400A
        DEFW L400A
        DEFW L3FFA      ; left*right graf
        DEFW L3FFA
```

```
        DEFW L400A
        DEFW L400A
        DEFW L3F0A      ; shift right graf
        DEFW L3F0A
        DEFW L400A
        DEFW L400A
        DEFW L3F0C      ; shift left graf
        DEFW L3F0C
        DEFW L400A
        DEFW L400A
        DEFW L3F06      ; shift
        DEFW L3EF8      ; nothing

L3E56:                  ; fuer Tasten 0AH-10H
        DEFW L3E6C
        DEFW L3E5E
        DEFW L3E65
        DEFW L3E5E

L3E5E:
        DEFM ':'',=./-'         ;** NORMAL

L3E65:
        DEFM ';"<+>?_'          ;** SHIFT

L3E6C:
        DEFB ';"<+>?',01FH      ;** CTRL

L3E73:                  ; fuer Tasten 2B-2D
        DEFW L3E7E
        DEFW L3E81
        DEFW L3E7E
        DEFW L3E7B

L3E7B:
        DEFM '[\]'              ;* normal
L3E7E:
        DEFM '{~}'              ;* shift
L3E81:
        DEFB 27,28,29           ;* ctrl


L3E84:          ; Tasten 2E-57
        DEFB 8,30,0,0,0,0,27,0,13,29,0,0,0,0,0,0,18,31
        DEFB 32,9,127,0,0,0,0,28
        DEFM '0123456789+-*/.,'

L3EAE:                  ; **** grobes Vorsortieren der Zeichen
        LD A,C
        CP 0FFH
        RET Z
        LD  HL,L3E05
        CALL LFF81
L3EB8:
        CP (HL)
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        INC HL
        PUSH DE         ; Tastencode in A
```



```
        RET C           ; ret zur Tabellenadresse falls Zeichen
        POP DE          ; kleiner als Tabellencode
        JR L3EB8
L3EC3:                  ; Tasten 0-9
        ADD A,030H
        LD B,A
        LD  A,(SHCTRL)
        RRCA
        LD E,A
        LD A,B
        JR C,L3EE0      ; C falls nicht shift gedrueckt
        LD A,E
        AND 01H
        LD B,00H
        LD  HL,L3E2C
        ADD HL,BC
        LD A,(HL)
        JR NZ,L3EE0     ; NZ falls nicht CTRL
        CP '@'
        JR C,L3EE0
        SUB '@'         ; shift ctrl @ ist NUL
L3EE0:
        JP L400A
L3EE3:                  ; Tasten 11-2A  ( A-Z )
        LD  A,(SHCTRL)
        AND 0FH
        ADD A,A
        LD E,A
        LD D,00H
        LD  HL,L3E36
        ADD HL,DE
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        LD A,C
        SUB 010H        ; a=1 b=2 ...
        JP (HL)
L3EF8:                  ; A-Z ohne special
        ADD A,040H
        LD B,A
        LD  A,(CAPST)
        AND A
        LD A,B
        JR NZ,L3EE0
        OR 020H         ; falls CAPS LOCK aus
        JR L3EE0
L3F06:                  ; A-Z   shift
        ADD A,040H
        JR L3EE0
L3F0A:                  ; A-Z   right Grafik
        ADD A,01AH
L3F0C:
        ADD A,09FH      ; A-Z   left Grafik
        JR L3EE0
L3F10:                  ; Tasten 0AH-10H
        LD  HL,L3E56
        SUB 0AH         ; 0..6
L3F15:
        LD C,A
        LD  A,(SHCTRL)
```

```
        AND 03H         ; nur shift und ctrl
        ADD A,A
        LD E,A
        LD D,00H
        ADD HL,DE       ; specialtabelle
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        LD E,C
        ADD HL,DE       ; Codeoffset
        LD A,(HL)       ; Code holen
        JR L3F3A

L3F29:                  ; Taste 2BH
        LD  A,(SHCTRL)
        AND 0CH         ; Grafik
        JP PE,L3F3D
        RRCA
        RRCA
        RRCA
        LD  A,'|'
        JR NC,L3F3A
        LD  A,''''
L3F3A:
        JP L400A

L3F3D:                  ; Taste 2C, 2D
        LD A,C
        LD  HL,L3E73
        SUB 02BH        ; 0..2
        JR L3F15

L3F45:                  ; TASTEN 29-2E
        .RADIX 16
        DEFB 0A1,  0, 3B, 3A,0A1,0BB    ; 29    ; ;     y
        DEFB 0AC,  0, 22, 27,0AC,0C6    ;       " '     z
        DEFB 0AD, 1B, 7B, 5B,0AD,0C7    ;       { [     {
        DEFB 0AF, 1C, 7E, 5C,0AF,0C9    ;       ~ \     ~
        DEFB 0B5, 1D, 7D, 5D,0B5,0CF    ;       } ]     }
        DEFB   8   8  7C, 60,  8   8    ; 2E    |  `      BS
        .RADIX 10

L3F69:                  ; TASTEN 29-2E  ( Y-BS ) alternativ
        SUB 01FH        ; - 1FH
        LD L,A
        LD H,00H
        ADD HL,HL       ; *2
        LD C,L
        LD B,H
        ADD HL,HL       ; *4
        ADD HL,BC       ; *6
        LD  BC,L3F45-03CH
        ADD HL,BC
        LD  A,(CAPST)
        AND A
        LD B,00H
        JR NZ,L3F81
        LD B,010H       ; falls CAPS
L3F81:
        LD  A,(SHCTRL)
```



```
        AND 0FH
        OR B
        LD B,05H
L3F89:
        RRCA
        JR NC,L3F8F
        INC HL
        DJNZ L3F89
L3F8F:
        JP L3FE2

L3F92:                          ; Taste 38-3C Funktionstasten
        LD  A,(SHCTRL)
        RRCA
        JR C,L3F9C      ; C kein shift
        LD A,C
        ADD A,05H
        LD C,A
L3F9C:
        LD E,C
        LD D,00H
        LD  HL,FNKFLG-38H
        ADD HL,DE
        LD A,(HL)
        AND A
        JR NZ,L3FBA     ; NZ = ON KEY definiert
L3FA7:
        EX DE,HL
        ADD HL,HL
        ADD HL,HL
        ADD HL,HL
        ADD HL,HL
        LD  DE,FNKSTR-380H
        ADD HL,DE
        EX DE,HL
L3FB1:
        LD  A,(DE)
        AND A
        RET Z           ; Z falls Key nicht definiert oder ende
        CALL L400A      ; Zeichen in Queue
        INC DE
        JR L3FB1
L3FBA:
        LD HL,(CURLIN)
        INC HL
        LD A,H
        OR L
        JR Z,L3FA7
        LD  HL,0FD43H
        ADD HL,DE
        ADD HL,DE
        ADD HL,DE
        JP REQTRP       ; ON KEY GOSUB einleiten
L3FCB:                          ; Taste 3D  CLS
        CALL LFF78
        LD  A,(SHCTRL)
        RRCA
        LD  A,0BH       ; mit shift VT
        ADC A,00H       ; ohne shift FF
        JR L400A
L3FD8:                          ; Taste 2E-34, 3E-42  INS-DEL,PRINT,->
```

```
        CALL LFF7B              ; numerisches Feld, 36, 37
        LD E,A
        LD D,00H
        LD  HL,L3E56
        ADD HL,DE
L3FE2:
        LD A,(HL)
        AND A
        RET Z
        JR L400A
L3FE7:                          ; Taste 43      CAPS
        LD  A,0FH
        OUT (088H),A    ; register 15
        LD  HL,CAPST
        LD A,(HL)
        CPL             ; CAPS invertieren
        AND 020H        ; Maske
        LD (HL),A       ; Speichern
        IN A,(090H)
        AND 0DFH        ; LED ausblenden
        OR (HL)         ; evtl. setzen
        OUT (08CH),A    ; zurueck zum Port
L3FFA:
        RET
L3FFB:                          ; Taste 35      STOP
        LD  A,(SHCTRL)
        RRCA
        RRCA
        LD  A,03H       ; ETX
        JR NC,L4005     ; jp falls CTRL
```



```
        INC A
L4005:
        LD (INTFLG),A           ; 4 bei STOP   3 bei CTRL STOP
        JR C,L4019      ; C falls nicht CTRL
L400A:
        LD HL,(PUTPNT)  ; auf Queue speichern
        LD (HL),A
        CALL L4034      ; HL-pointer incrementieren
        LD  A,(GETPNT)
        CP L
        RET Z           ; falls beide gleich nicht abspeichern
        LD (PUTPNT),HL  ; da sonst ueberlauf entsteht
L4019:
        LD  A,(CLICK)   ; ? click on
        AND A
        RET Z
        LD  A,(CLIKFL)  ; intern click ?
        AND A
        RET NZ
        LD  A,0FH       ; falls beide ja dann click
        LD (CLIKFL),A
        OUT (097H),A    ; sound Bit setzen
        LD  A,0AH
L402C:
        DEC A
        JR NZ,L402C     ; warten
        LD  A,0EH
        OUT (097H),A    ; sound bit ruecksetzen
        RET
L4034:          ; ***** Keybord buffer queue pointer
        INC HL  ;       inkrementieren
        LD A,L
        CP 0B3H
        RET NZ
        LD  HL,KEYBUF
        RET

CHGET/TRYIN: 403D       ; ******** Zeichen aus dem Keybordpuffer
        PUSH HL         ;       holen falls vorhanden, sonst auf
        PUSH DE         ;       Zeichen warten
        PUSH BC
L4040:
        CALL CHSNS
        JR Z,L4040      ; Zeichen da ?
        LD  HL,INTFLG
        LD A,(HL)
        CP 04H
        JR NZ,L404F     ; NZ kein Stop
        LD (HL),00H     ; stop ruecksetzen
L404F:
        LD HL,(GETPNT)
        LD C,(HL)       ; Zeichen
        CALL L4034
        LD (GETPNT),HL  ; pointer update
        LD A,C          ; Zeichen nach A
        JP PBDHRT
CKCNTC: 4050
        PUSH HL
        LD  HL,00H
        CALL ISCNTC
        POP HL
```

```
        RET
GICINI: 4066            ; Generatoren Initialisieren
        PUSH HL
        PUSH DE
        PUSH BC
        PUSH AF
        LD  HL,MUSIKF
        LD B,071H
        XOR A           ; 71H auf 0
L4070:
        LD (HL),A
        INC HL
        DJNZ L4070
        LD  DE,VOICAQ
        LD B,07FH
        LD  HL,080H
L407C:
        PUSH HL
        PUSH DE
        PUSH BC
        PUSH AF         ; Generatornummer 0-2
        CALL INITQ
        POP AF
        ADD A,08H       ; 8,9,10
WRTGIC: 4086
        LD  E,00H
        CALL L40B6      ; Amplitude auf 0
        SUB 08H
        PUSH AF
        LD  L,0FH
        CALL GETVC1     ; Parameter 15
        EX DE,HL
        LD  HL,L40B1
        LD  BC,05H
        LDIR            ; 15-19 setzen
        POP AF
        POP BC
        POP HL
        POP DE
        ADD HL,DE       ; naechster Generator
        EX DE,HL
        INC A
        CP 03H
        JR C,L407C
        LD  A,07H
        LD  E,0B8H
        CALL L40B6      ; port a input, alle Geraeusche aus
        JP POPALL       ;               alle Toene an
L40B1:
        DEFB 4,4,078H,0C8H,0FFH

L40B6:                  ; ***** 3-8910 Register setzen
        OUT (088H),A    ; A registernummer      E inhalt
        PUSH AF
        LD A,E
        OUT (08CH),A
        POP AF
        RET
BEEP:   46BE            ; ***********************************
        XOR A
        LD  E,055H
```



```
        CALL L40B6      ; ton A Periode 55H
        LD E,A
        INC A
        CALL L40B6      ; ton A1 0      gesamt 55H
        LD  E,0BEH
        LD  A,07H
        CALL L40B6      ; enable generator A
        LD E,A
        INC A
        CALL L40B6      ; Lautstaerke 7
        LD  BC,07D0H
        CALL L40DD      ; warten
        JR GICINI       ; init Generatoren
L40DD:
        DEC BC
        EX (SP),HL
        EX (SP),HL
        LD A,B
        OR C
        JR NZ,L40DD
        RET
L40E5:
        LD B,A
        CALL GETVCP
        DEC HL
        LD D,(HL)
        DEC HL
        LD E,(HL)
        DEC DE
        LD (HL),E
        INC HL
        LD (HL),D
        LD A,D
        OR E
        RET NZ
        LD A,B
        LD (QUEUEN),A
        CALL L418C
        CP OFFH
        JR Z,L415A
        LD D,A
        AND 0E0H
        RLCA
        RLCA
        RLCA
        LD C,A
        LD A,D
        AND 01FH
        LD (HL),A
        CALL L418C
        DEC HL
        LD (HL),A
        INC C
L4110:
        DEC C
        RET Z
        CALL L418C
        LD D,A
        AND 0C0H
        JR NZ,L412B
        CALL L418C
```

```
        LD E,A
        LD A,B
        RLCA
        CALL L40B6
        INC A
        LD E,D
        CALL L40B6
        DEC C
        JR L4110
L412B:
        LD H,A
        AND 080H
        JR Z,L413F
        LD E,D
        LD A,B
        ADD A,08H
        CALL L40B6
        LD A,E
        AND 010H
        LD  A,0DH
        CALL NZ,L40B6
L413F:
        LD A,H
        AND 040H
        JR Z,L4110
        CALL L418C
        LD D,A
        CALL L418C
        LD E,A
        LD  A,0BH
        CALL L40B6
        INC A
        LD E,D
        CALL L40B6
        DEC C
        DEC C
        JR L4110
L415A:
        LD A,B
        ADD A,08H
        LD  E,00H
        CALL L40B6
        INC B
        LD  HL,MUSIKF
        XOR A
        SCF
L4168:
        RLA
        DJNZ L4168
        AND (HL)
        XOR (HL)
        LD (HL),A
STRTMS: 416E                    ; ****** Musik starten
        LD  A,(MUSIKF)
        OR A
        RET NZ
        LD  HL,PLAYCNT
        LD A,(HL)
        OR A
        RET Z
        DEC (HL)
```



```
        LD  HL,01H
        LD (VCBA/METREX),HL
        LD (VCBB),HL
        LD (VCCC),HL
        LD  A,07H               ; alle drei Kanaele
        LD (MUSIKF),A
        RET
L418C:
        LD  A,(QUEUEN)
        PUSH HL
        PUSH DE
        PUSH BC
        CALL GETQ
        JP PBDHRT


        .RADIX 16
L4198:                  ; Zeichengeneratormasken
        DEFB 000,000,000,000,000,000    ; SPACE
        DEFB 020,082,008,000,002,000    ; !
        DEFB 051,045,000,000,000,000    ; "
        DEFB 051,04F,094,0F9,045,000    ; =
        DEFB 021,0EA,01C,02B,0C2,000    ; $
        DEFB 0C3,021,008,042,061,080    ; %
        DEFB 042,084,02A,092,066,000    ; &
        DEFB 010,084,000,000,000,000    ; '

        DEFB 010,084,010,040,081,000    ; (
        DEFB 040,081,004,010,084,000    ; )
        DEFB 022,0A7,008,072,0A2,000    ; *
        DEFB 000,082,03E,020,080,000    ; +
        DEFB 000,000,000,000,082,010    ; ,
        DEFB 000,000,03E,000,000,000    ; -
        DEFB 000,000,000,001,086,000    ; .
        DEFB 000,000,084,021,008,000    ; /

        DEFB 072,029,0AA,0CA,027,000    ; 0
        DEFB 021,08A,008,020,08F,080    ; 1
        DEFB 072,020,084,062,00F,080    ; 2
        DEFB 072,020,08C,00A,027,000    ; 3
        DEFB 010,0C5,024,0F8,041,000    ; 4
        DEFB 0FA,00E,004,008,04E,000    ; 5
        DEFB 031,008,03C,08A,027,000    ; 6
        DEFB 0FA,021,008,020,082,000    ; 7

        DEFB 072,028,09C,08A,027,000    ; 8
        DEFB 072,028,09E,008,046,000    ; 9
        DEFB 000,002,000,000,080,000    ; :
        DEFB 000,002,000,000,082,010    ; ;
        DEFB 018,0C6,030,060,0C1,080    ; <
        DEFB 000,00F,080,0F8,000,000    ; =
        DEFB 0C1,083,006,031,08C,000    ; >
        DEFB 072,020,084,020,002,000    ; ?

        DEFB 072,020,09A,0AA,0A7,000    ; @
        DEFB 021,048,0A2,0FA,028,080    ; A
        DEFB 0F1,024,09C,049,02F,000    ; B
        DEFB 031,028,020,081,023,000    ; C
        DEFB 0E1,044,092,049,04E,000    ; D
```

```
DEFB 0FA,008,03C,082,00F,080    ; E
DEFB 0FA,008,03C,082,008,000    ; F
DEFB 072,028,02E,08A,027,000    ; G

DEFB 08A,028,0BE,08A,028,080    ; H
DEFB 070,082,008,020,087,000    ; I
DEFB 038,041,004,092,046,000    ; J
DEFB 08A,04A,030,0A2,048,080    ; K
DEFB 082,008,020,082,00F,080    ; L
DEFB 08B,06A,0AA,08A,028,080    ; M
DEFB 08B,02C,0AA,09A,068,080    ; N
DEFB 072,028,0A2,08A,027,000    ; O

DEFB 0F2,028,0BC,082,008,000    ; P
DEFB 072,028,0A2,0AA,046,080    ; Q
DEFB 0F2,028,0BC,0A2,048,080    ; R
DEFB 072,028,01C,00A,027,000    ; S
DEFB 0F8,082,008,020,082,000    ; T
DEFB 08A,028,0A2,08A,027,000    ; U
DEFB 08A,028,0A2,051,042,000    ; V
DEFB 08A,028,0AA,0AB,068,080    ; W

DEFB 08A,025,008,052,028,080    ; X
DEFB 08A,028,09C,020,082,000    ; Y
DEFB 0F8,021,008,042,00F,080    ; Z
DEFB 071,004,010,041,007,000    ; [
DEFB 000,008,010,020,040,080    ; \
DEFB 070,041,004,010,047,000    ; ]
DEFB 021,048,080,000,000,000    ; ^
DEFB 000,000,000,000,00F,080    ; _

DEFB 040,081,000,000,000,000    ; `
DEFB 000,007,002,07A,027,080    ; a
DEFB 082,00B,032,08B,02B,000    ; b
DEFB 000,007,022,082,027,000    ; c
DEFB 008,026,0A6,08A,066,080    ; d
DEFB 000,007,022,0FA,007,000    ; e
DEFB 010,0A2,03E,020,082,000    ; f
DEFB 000,006,0A6,099,0A0,09C    ; g

DEFB 082,00F,022,08A,028,080    ; h
DEFB 020,006,008,020,087,000    ; i
DEFB 010,003,004,010,049,018    ; j
DEFB 041,004,094,061,044,080    ; k
DEFB 060,082,008,020,087,000    ; l
DEFB 000,00D,02A,0AA,0AA,080    ; m
DEFB 000,00B,032,08A,028,080    ; n
DEFB 000,007,022,08A,027,000    ; o

DEFB 000,00B,032,0CA,0C8,020    ; p
DEFB 000,006,0A6,099,0A0,082    ; q
DEFB 000,00B,032,082,008,000    ; r
DEFB 000,007,0A0,0F0,02F,000    ; s
DEFB 041,00F,010,041,023,000    ; t
DEFB 000,009,024,092,046,080    ; u
DEFB 000,008,0A2,089,042,000    ; v
DEFB 000,008,0AA,0AA,0A5,000    ; w

DEFB 000,008,094,021,048,080    ; x
DEFB 000,008,0A2,099,0A0,09C    ; y
DEFB 000,00F,084,021,00F,080    ; z
```



```
DEFB 018,082,010,020,081,080    ; {
DEFB 020,082,000,020,082,000    ; |
DEFB 0C0,082,004,020,08C,000    ; }
DEFB 042,0A1,000,000,000,000    ; ~

DEFB 010,041,004,01C,041,004    ; graphic code 0A0H
DEFB 0FF,0FF,0FF,000,000,000
DEFB 010,041,004,0F0,000,000
DEFB 010,041,004,0F0,041,004
DEFB 000,000,000,0F0,041,004
DEFB 01C,071,0C7,01C,071,0C7
DEFB 0E3,08E,038,01C,071,0C7
DEFB 0E3,08E,038,0E3,08E,038

DEFB 0FC,010,041,004,010,041    ; 0A8H
DEFB 082,008,020,082,008,03F
DEFB 004,010,041,004,010,07F
DEFB 000,000,000,000,000,03F
DEFB 000,00F,0C0,0FC,000,000
DEFB 0E3,08E,038,000,000,000
DEFB 0FC,000,000,000,000,000
DEFB 0FE,018,061,086,018,07F

DEFB 000,000,000,01C,041,004    ; 0B0H
DEFB 000,000,000,01C,071,0C7
DEFB 010,041,004,0FC,041,004
DEFB 000,000,000,0FF,0FF,0FF
DEFB 0FE,008,020,082,008,020
DEFB 01C,071,0C7,000,000,000
DEFB 000,000,000,0FC,041,004
DEFB 010,041,004,0FC,000,000

DEFB 000,000,000,0E3,08E,038    ; 0B8H
DEFB 010,041,004,01C,000,000
DEFB 082,00C,038,0E3,0CF,0BF
DEFB 04B,0FF,0FF,0FD,0E3,00C
DEFB 004,021,084,021,084,020
DEFB 085,027,08C,031,0E4,0A1
DEFB 081,006,008,010,060,081
DEFB 031,0EC,0E1,087,037,08C

DEFB 01C,071,0C7,0E3,08E,038    ; 0C0H
DEFB 031,0EF,0FF,0FF,0F7,08C
DEFB 000,000,000,0C0,081,004
DEFB 010,041,002,004,000,000
DEFB 010,041,008,0C0,000,000
DEFB 082,008,020,082,008,020
DEFB 030,0CF,0FF,0CC,0C3,01E
DEFB 030,0C7,0BF,0FD,0E3,00C

DEFB 004,010,041,004,010,041    ; 0C8H
DEFB 0FF,0FF,0FF,0FF,0FF,0FF
DEFB 0FF,0EF,038,0E3,008,020
DEFB 000,000,000,0A9,05A,095
DEFB 004,010,0C7,01C,0F7,0FF
DEFB 0A1,00A,010,0A1,00A,010
DEFB 000,000,000,004,021,004
DEFB 031,0EF,0FF,0FC,0C3,01E

DEFB 0FD,0F3,0C7,01C,030,041    ; 0D0H
DEFB 000,000,000,0FC,000,000
```

```
        DEFB 0A9,05A,095,0A9,05A,095
        DEFB 010,041,004,010,041,004
        DEFB 000,081,03E,010,080,000    ; ARROW RIGHT
        DEFB 000,084,03E,040,080,000    ; ARROW LEFT
        DEFB 021,0CA,088,020,080,000    ; ARROW UP
        DEFB 020,082,02A,070,080,000    ; ARROW DOWN

        DEFB 038,041,004,092,046,000    ; OD8H          J
        DEFB 000,000,000,001,086,000                    .
        DEFB 072,028,01C,00A,027,000                    S
        DEFB 000,009,024,092,046,080                    u
        DEFB 000,00F,084,021,00F,080                    z
        DEFB 000,009,024,092,046,080                    u
        DEFB 041,004,094,061,044,080                    k
        DEFB 020,006,008,020,087,000    ; ODFH          i

        .RADIX 10

COLOR:/ 4552            ; ************************************
        LD  BC,FCERR
        PUSH BC
        LD DE,(front/COLOR)
        PUSH DE         ; Voreinstellung Farbe
        CP ','
        JR Z,L456C
        CALL GETBYT     ; Vordergrundfarbe holen
        POP DE
        CP 010H
        RET NC          ; range error falls >=16
        LD E,A          ; neue Farbe
        PUSH DE
        DEC HL
        RST 10H
        JR Z,L458E      ; Ende der eingabe
L456C:
        RST 8H
        DEFB ','
        JR Z,L458E
        CP ','
        JR Z,L4581
        CALL GETBYT     ; Hintergrundfarbe
        POP DE
        CP 010H
        RET NC          ; Test >=16
        LD D,A          ; sonst nach D
        PUSH DE
        DEC HL
        RST 10H
        JR Z,L458E
L4581:
        RST 8H
        DEFB ','
        CALL GETBYT     ; Randfarbe holen
        POP DE
        CP 010H
        RET NC          ; max Test
        LD (BORCLR),A
        PUSH DE
L458E:
        POP DE
        POP AF
```



```
        PUSH HL
        EX DE,HL
        LD (front/COLOR),HL     ; Setzen der Farbe
        CALL CHGCLR             ; und Wechseln
        POP HL
        RET
SCREEN:/ 459A           ; ************************************
        CP ','
        JR Z,L45B1
        CALL GETBYT     ; screen 0..2 holen
        CP 03H
        JP NC,FCERR     ; max Test
        LD (SCREEN),A
        PUSH HL
        CALL CHGMOD     ; screen setzen
        POP HL
        DEC HL
        RST 10H
        RET Z           ; Z keine Spritegroesse
L45B1:
        RST 8H
        DEFB ','
        CALL GETBYT     ; 2. Parameter
        LD  A,(SCREEN)
        AND A
        LD A,E
        JR Z,L45CB
        CP 04H          ; max Test fuer Grafik
        JP NC,FCERR
        LD (SPRSIZ),A   ; Spritegroesse setzen
        PUSH HL
        CALL CLRSPR     ; Sprites neu initialisieren
        POP HL
        RET
L45CB:                  ; screen 0 2.parameter
        AND A
        JP Z,ERAFNK     ; Keyanzeige loeschen
        JP DSPFNK       ; Keys anzeigen
SPRITE:/ 45D2           ; ************************************
        RST 10H
        CP '$'
        JP NZ,SPRTTP
        CALL L4623      ; Sprite-videoadresse holen
        PUSH HL         ; fuer SPRITE$()=
        EX DE,HL
        CALL SETWRT     ; Setzen zum Schreiben
        POP HL
        CALL FRMEQL     ; String berechnen
        PUSH HL
        CALL FRESTR     ; freigeben
        LD C,(HL)       ; Laenge
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)       ; DE Adresse
        CALL L4641      ; maximale Laenge fuer sprite holen
        LD B,A
        INC C
L45F2:
        DEC C
        JR Z,L45FD      ; falls gegebener String zu Ende
```

```
        LD  A,(DE)      ; byte vom String
        OUT (080H),A    ; ausgeben
        INC DE
        DJNZ L45F2      ; max bytes -1
        POP HL
        RET
L45FD:
        XOR A           ; String zu Ende
L45FE:
        OUT (080H),A
        EX (SP),HL
        EX (SP),HL
        DJNZ L45FE      ; mit 0 bis Maxlaenge
        POP HL
        RET
RETSPR: 4606            ; ***************************************
        CALL L4622      ; =SPRITE$()  Adresse aufsetzen
        PUSH HL
        EX DE,HL
        CALL SETRD      ; zum Lesen
        CALL L4641      ; Maxlaenge
        PUSH AF
        CALL STRINI     ; Stringraum schaffen
        LD HL,(DSCPTR)  ; Adresse
        POP BC
L4619:
        IN A,(084H)
        LD (HL),A
        INC HL
        DJNZ L4619      ; transfer
        JP PUTNEW
L4622:                  ; ** $() fuer sprite auswerten und Adresse
        RST 10H         ;  des Spritemusters im Videoram aufsetzen
L4623:
        RST 8H
        DEFB '$'
        RST 8H
        DEFB '('
        CALL GETBYT     ; Spritenummer holen
        RST 8H
        DEFB ')'
        PUSH HL
        EX DE,HL
        LD H,00H
        ADD HL,HL
        ADD HL,HL
        ADD HL,HL       ; *8 bytes
        CALL L4641
        JR NC,L463A
        ADD HL,HL
        ADD HL,HL       ; oder 32 bytes je nach Spritegroesse
L463A:
        LD  DE,SPRPAT   ; Basisadresse
        ADD HL,DE
        EX DE,HL        ; nach DE
        POP HL
        RET
L4641:                  ; ***** gibt Bytes fuer Spritemuster
        LD  A,(SPRSIZ)  ; je nach Spritegroesse
        RRCA
        RRCA
```



```
        LD  A,08H       ; kleine Sprites
        RET NC
        LD  A,020H      ; grosse Sprites
        RET
PUTSPR: 464C            ; ***************************************
        DEC B
        JP M,FCERR
        CALL CHKMOD     ; wirklich Grafikmodus?
        RST 10H
        CALL GETBYT     ; Spriteebene
        CP 32
        JP NC,FCERR     ; max Spriteebene
        PUSH HL
        EX DE,HL
        LD H,00H
        ADD HL,HL
        ADD HL,HL
        LD  DE,SPRATR   ; Spriteattribute entsprechend Ebene
        ADD HL,DE       ; aufsetzen
        EX (SP),HL
        RST 8H
        DEFB ','
        CP ','
        JR Z,L4698
        CALL NEGD/SCAN1 ; Position x,y
        EX (SP),HL
        LD A,E
        CALL WRTVDP     ; y Position
        LD A,B
        ADD A,A
        LD A,C
        LD B,00H        ; no early clock
        JR NC,L4680     ; falls Position > 0
        ADD A,32        ; +32
        LD B,080H       ; set early clock
L4680:
        INC HL
        CALL WRTVDP     ; x Position
        INC HL
        INC HL
        CALL RDVDP      ; Farbe holen
        AND 0FH
        OR B            ; evtl. clockbit setzen
        CALL WRTVDP     ; zurueckschreiben
        DEC HL
        DEC HL
        DEC HL
        EX (SP),HL
        DEC HL
        RST 10H
        POP BC
        RET Z
        PUSH BC
L4698:
        RST 8H
        DEFB ','
        CP ','
        JR Z,L46BC
        CALL GETBYT     ; Farbe holen
        CP 010H
        JP NC,FCERR     ; max Farbe
```

```
        EX (SP),HL
        INC HL
        INC HL
        INC HL
        CALL RDVDP      ; early clock und farbe lesen
        AND 080H        ; clock maskieren
        OR E            ; + Farbe
        CALL WRTVDP     ; schreiben
        DEC HL
        DEC HL
        DEC HL
        EX (SP),HL
        DEC HL
        RST 10H
        POP BC
        RET Z
        PUSH BC
L46BC:
        RST 8H
        DEFB ','
        CALL GETBYT     ; Spritenummer
        LD  A,(SPRSIZ)
        RRCA
        RRCA
        LD A,E
        JR NC,L46D0     ; jp fals kleine Sprites
        CP 64
        JP NC,FCERR     ; nicht groesser als 63 sonst 255
        ADD A,A
        ADD A,A
L46D0:
        EX (SP),HL
        INC HL
        INC HL
        CALL WRTVDP     ; Sprite-Namen setzen
        POP HL
        RET
VPOKE:  ; 46D8          ; ************************************
        CALL FRMEVL     ; Adresse
        PUSH HL
        CALL FRCINT
        LD  DE,04000H
        RST 20H
        JP NC,FCERR     ; nicht groesser als 4000H
        EX (SP),HL
        RST 8H
        DEFB ','
        CALL GETBYT     ; Inhalt
        EX (SP),HL
        CALL WRTVDP     ; und Schreiben
        POP HL
        RET
VPEEK:  ; 46F2          ; ************************************
        CALL FRCINT     ; Adresse
        LD  DE,04000H
        RST 20H
        JP NC,FCERR     ; nicht groesser als 4000H
        CALL RDVDP      ; und lesen
        JP SNGFLT       ; wandeln
GRPPRT: ; 4702          ; ************************************
        POP AF          ; print im Grafik-Modus
```



```
        PUSH AF
        CP ODH
        JR Z,L4767      ; Z bei CR
        CP 020H
        JR C,L4764      ; nix bei Steuerzeichen
        CALL GETPAT     ; muster fuer ASCII-code aus Generator-
        LD  A,(front/COLOR)     ; rom holen
        LD (ATRBYT),A   ; auf Vordergrundfarbe setzen
        LD HL,(CSRY)
        LD C,H          ; x
        LD B,00H
        LD E,L          ; y
        LD D,B
        CALL SCALXY     ; skalieren
        JR NC,L4764     ; NC ausserhalb
        CALL MAPXYC     ; koord. wandeln
        LD  DE,PATWRK   ; Muster-Start
        LD  C,08H
L472A:
        LD B,06H
        CALL FETCHC     ; Position holen
        PUSH HL
        PUSH AF         ; sichern
        LD  A,(DE)      ; Muster-Byte
L4732:
        ADD A,A
        PUSH AF
        CALL C,SETC     ; Punkt evtl. setzen
        CALL L49B6      ; spezial rechts
        POP HL
        JR C,L4741      ; C falls Zeilenende
        PUSH HL
        POP AF
        DJNZ L4732      ; naechstes Tabellenzeichen
L4741:
        POP AF
        POP HL          ; Anfangsposition holen
        CALL STOREC     ; zurueckspeichern
        CALL TDOWNC     ; abwaerts
        JR C,L474F      ; C falls Bildschirmende
        INC DE
        DEC C
        JR NZ,L472A     ; NZ noch Zeichen in Muster-Speicher?
L474F:
        CALL CHKMOD
        LD  A,(CSRX)
        JR Z,L475D      ; Z falls screen 1
        ADD A,24        ; screen 2    6*4 pixel
        JR C,L4767      ; C Schirmende
        JR L4761
L475D:
        ADD A,06H       ; 6 Pixel weiter
        JR C,L4767      ; C Schirmende
L4761:
        LD (CSRX),A
L4764:
        JP POPALL
L4767:                  ; Uebertrag nach y
        XOR A
        LD (CSRX),A     ; neue x-Position 0
```

```
        CALL CHKMOD
        LD  A,(CSRY)
        JR Z,L4776          ; Z screen 1
        ADD A,32
        DEFB 1              ;*       LD  BC,08C6H
L4776:
*       ADD A,8             ;*       y:=y+8
        CP 192
        JR C,L477D          ; max y
        XOR A               ; falls groesser Null
L477D:
        LD (CSRY),A
        JR L4764
PRLOGO:/ 4782               ; ********** SPECTRAVIDEO EINLEITUNG
        CALL INIGRP         ; 3610 / Grafik initialisieren / Cursor auf (010)
        CALL INIGRP         ; grafik /3610
        LD  A,01H
        LD (SCREEN),A       ; screen 1 /FE3A
        INC A
        LD (ATRBYT),A       ; Mittelgruen /FA13 / Farbe auf 2 setzen
        CALL L47DF          ; spectravideologo
Unterstreichung {LD  A,0DH  ; magenta
       1        {LD  BC,56  ; x
                {LD  DE,100 ; y
                {LD H,72    ; Laenge
                {CALL L4832 ; Linie 1
Unterstreichung {LD  A,08H  ; mittelrot
       2        {LD  BC,58
                {LD  DE,104
                {LD H,70
                {CALL L4832 ; Linie 2
Unterstreichung {LD  A,0AH  ; dunkelgelb
       3        {LD  BC,60
                {LD  DE,108
                {LD H,68
                {CALL L4832 ; Linie 3
        LD  HL,04846H
L47BE:
        LD A,(HL)           ; Farbe holen
        CP 0FFH             ;   vergl. Tabelle 1
        JR Z,L47CE          ; bei Z -> Ende, sonst nächste Farbe
        LD (ATRBYT),A       ; speichern /FA13
        PUSH HL
        CALL L47DF          ; spectravideo Logo
        POP HL
        INC HL              ; Tabelle erhöhen
        JR L47BE            ; nochmal
L47CE:
        LD B,02H                    ; 2* 0ffffH
        LD  HL,00H
L47D3:
        DEC HL
        LD A,H
        OR L
        JR NZ,L47D3
        DJNZ L47D3                  ; warten
        XOR A
        LD (SCREEN),A               ; Textmodus /FE3A mit 0 laden
        RET                         ;
L47DF:                              ; ***** Ausgabe Spectravideo
        LD  BC,40           ; x
```

(Margin notes: "Ende: Warte u. Textmodus einschalten"; "Initialisierung & Vorbereitung")


```
        LD  DE,82           ; y
        CALL MAPXYC         ; in chr Koordinaten
        LD  DE,04849H       ; Spectra- Tabelle
        LD B,08H            ; 8 Zeilen
L47ED:
        PUSH BC             ; 8 Zeilen-Wert speichern         CMASK    KOORDINATEN
        CALL FETCHC         ; Position holen /4943 : FAC2 -> A / FAC0 -> HL
        PUSH HL             ; sichern
        PUSH AF             ; MASKE & Koordinate abspeichern
        LD B,0BH            ; 11 Zeichen
L47F5:
        PUSH BC
        LD  A,(DE)          ; Zeichen aus Tabelle
        LD B,08H            ; 8 Punkte pro Zeichen
L47F9:
        ADD A,A
        PUSH AF             ; Punktwert speichern
        CALL C,L481A        ; Punkt setzen falls Bit /4 Punkte setzen
        CALL RIGHTC         ; Cursor 2 Positionen nach rechts
        CALL RIGHTC         ; naechste Position Faktor 2
        POP AF              ; Punktwert laden
        DJNZ L47F9          ;
        INC DE              ; naechstes Zeichen
        POP BC
        DJNZ L47F5
        POP AF
        POP HL              ; alte Startposition
        CALL STOREC         ; speichern
        CALL DOWNC
        CALL DOWNC          ; naechste Zeile
        POP BC              ; Anzahl Zeilen
        DJNZ L47ED          ; Dec & JNZ
        RET                 ; ENDE für akt. Farbe
L481A:                      ; ******* vier Punkte setzen
        CALL SETC           ; Punkt setzen / 4988 /Farbe in ATRBYT, Koord. in CLOC/CMASK
        CALL RIGHTC         ; nach rechts / 49CF
        CALL SETC           ; Punkt setzen / 4988
        CALL DOWNC          ; nach unten / 4A2D
        CALL SETC           ; Punkt setzen / 4988
        CALL LEFTC          ; nach links / 49F8
        CALL SETC           ; Punkt setzen / 4988
        JP UPC              ; Ausgangspunkt / 4A57
L4832:                      ; ********** line in 4 Punkt Einheiten
        LD (ATRBYT),A       ; Farbe /FA13                      siehe 47ED
        PUSH HL
        CALL MAPXYC         ; Wandeln Koordinaten / 48E9  x->BC / y->DE / CMASK & KOORD.
        POP BC              ; Laenge in B
L483A:
        CALL L481A          ; 4 Pkt. setzen
        CALL RIGHTC         ; 2 Positionen nach rechts
        CALL RIGHTC         ; naechste Position
        DJNZ L483A          ;
        RET
L4846: / Tabelle 1
        DEFB 0AH,0EH,0FFH           ; Farben dunkelgelb, grau   Tabelle

        .RADIX 16
L4849: / Tabelle 2          ; Matrix fuer Spectravideo
        DEFB 0FE,07E,07E,07C,0FD,0F3,0F3,01B,03C,03F,03F
        DEFB 07F,03F,07F,07E,0FD,0FB,0F3,03B,03E,07E,07F
        DEFB 030,031,030,060,031,09B,033,033,066,060,0C6
```

```
        DEFB 01F,09F,0BC,030,031,0FB,0F3,066,066,0FC,08C
        DEFB 00F,0CF,09E,030,019,0FB,0F6,066,06C,0F9,098
        DEFB 000,06C,018,010,019,0B3,036,0CC,0CD,083,030
        DEFB 007,0E6,00F,09F,019,0BB,037,0CC,0F9,0F3,0E0
        DEFB 003,0F3,00F,0DF,099,09B,037,08C,0F3,0E7,0C0

        .RADIX 10
SCAXY:  48A1            ; ********** Koordinatenpruefung
        PUSH HL         ; HL
        PUSH BC         ; BC x Koordinate   DE y Koordinate
        LD B,01H
        EX DE,HL
        LD A,H
        ADD A,A         ; y high * 2
        JR NC,L48AF     ; NC nicht negativ
        LD  HL,00H      ; 0 falls negativ
        JR L48B7
L48AF:
        LD  DE,192
        RST 20H
        JR C,L48B9      ; C falls < 192
        EX DE,HL
        DEC HL          ; sonst 191
L48B7:
        LD B,00H        ; y nicht im Bereich
L48B9:
        EX (SP),HL      ; x holen
        LD A,H
        ADD A,A
        JR NC,L48C3     ; NC nicht negativ
        LD  HL,00H      ; falls <0 =0 setzen
        JR L48CB
L48C3:
        LD  DE,256
        RST 20H
        JR C,L48CD      ; < 256 ?
        EX DE,HL
        DEC HL          ; sonst 255
L48CB:
        LD B,00H        ; wenigstens eine Koordinate ausserhalb
L48CD:                  ; X in HL
        POP DE          ; y in DE
        CALL CHKMOD     ; Grafikmodus ?
        JR Z,L48DB      ; Z screen 1
        SRL L
        SRL L           ; /4
        SRL E
        SRL E           ; /4 falls screen 2
L48DB:
        LD A,B
        RRCA
        LD B,H
        LD C,L
        POP HL          ; x in BC,   y in DE.  C falls ausserhalb
        RET
CHKMOD: 48E1            ; ********** Test ob Grafikmodus sonst
        LD  A,(SCREEN)  ;       Fehler Z screen 1 / FE3A
        DEC A
        JP M,FCERR      ; function call error / A5
        RET
MAPXYC: 48E9            ; ****** mapping x,y in CHR Koordinaten
                        ;        x in BC / y in DE
```



```
        PUSH BC         ; x sichern
        CALL CHKMOD
        JR NZ,L491D     ; NZ screen 2 / MAPXYC für SC2
        LD D,C          ; mapping screen 1
        LD A,C
        AND 07H         ; untere 3 Bits entscheiden
        LD C,A
        LD  HL,L4915
        ADD HL,BC       ; die Bitposition
        LD A,(HL)       ; chrmaske aus Tabelle
        LD (CMASK),A
        LD A,E          ; nun chr Position berechnen
        RRCA
        RRCA
        RRCA
        AND 01FH        ; y div 8
        LD B,A          ; -> B          (BC:=y div (8*256))
        LD A,D
        AND 0F8H        ; (x DIV 8)*8
        LD C,A          ; -> C
        LD A,E
        AND 07H         ; y mod 8
        OR C            ; die x-Koordinate ist hier hoeherwertiger
        LD C,A          ; (darstellung durch VDP)
        LD  HL,00H
        ADD HL,BC
        LD (CHKROM/CLOC),HL     ; chrposition im VRAM / FAC#
        POP BC                  ; Koordinaten => Characterposition (FAC#)
        RET
L4915: Tabelle  ; masken fuer bits
        DEFB 128,64,32,16,8,4,2,1

L491D: MAPXYC für SC2 ; mapping screen 2
        LD A,C
        RRCA
        LD  A,0F0H
        JR NC,L4925     ; gerade oder ungerade?
        LD  A,0FH
L4925:
        LD (CMASK),A    ; entscheidet die Bitmaske
        LD A,C
L4929:
        ADD A,A
        ADD A,A
        AND 0F8H
        LD C,A          ; C:=(x*4 div 8)*8
        LD A,E
        AND 07H
        OR C
        LD C,A          ; Lowbyte fuer chrposition
        LD A,E
        RRCA
        RRCA
        RRCA
        AND 07H
        LD B,A          ; high byte
        LD  HL,00H
        ADD HL,BC
        LD (CHKROM/CLOC),HL     ; und speichern
        POP BC
        RET             ; siehe oben!
```

```
FETCHC:  4943          ; ******* chr Koordinaten holen
         LD  A,(CMASK)    ; bitpixel Koordinate
         LD HL,(CHKROM/CLOC)      ; CHR Koordinate
         RET
STOREC:  494A          ; ******* wie oben speichern
         LD (CMASK),A
         LD (CHKROM/CLOC),HL
         RET
READC:   4951          ; ******* Farbe eines Bildpunktes lesen
         PUSH BC
         PUSH HL
         CALL FETCHC      ; Koordinaten
         LD B,A
         CALL CHKMOD
         JR NZ,L4976
         CALL RDVDP       ; screen 1 chr lesen
         AND B            ; Pixel herausfischen
         PUSH AF
         LD  BC,GRPCCL
         ADD HL,BC
         CALL RDVDP       ; Farbbyte lesen
         LD B,A
         POP AF
         LD A,B
         JR Z,L4971       ; Z Hintergrundfarbe nehmen
L496D:
         RRCA
         RRCA
         RRCA
         RRCA             ; Vordergrundfarbe
L4971:
         AND OFH          ; rest Ausblenden
         POP HL
         POP BC
         RET              ; Farbe in A
L4976:                    ; screen 2
         CALL RDVDP       ; ist schon Farbe
         INC B
         DEC B
         JP P,L4971
         JR L496D
SETATR:  4980          ; ******* setzen einer Farbe
         CP 16
         CCF
         RET C            ; C Farbnummer zu gross
         LD (ATRBYT),A    ; FA13
         RET
SETC:    4988          ; ******* setzen einer Farbe fuer Bild-
         PUSH HL          ;       punkt in beiden Grafikmodi
         PUSH BC          ;       mit Holen der Koordinaten
         CALL CHKMOD      ; Grafik?
         CALL FETCHC      ; Koordinaten holen
         JR NZ,L499A      ; NZ screen 2
         PUSH DE
         CALL L4B5C       ; MASKE und MUSTER holen
         POP DE
         POP BC
         POP HL
         RET
L499A:
         LD B,A           ; Maske
```



```
         CALL RDVDP       ; Farbbyte lesen
         LD C,A           ; sichern
         LD A,B
         CPL
         AND C
         LD C,A           ; nur noch Farbe von Restpixel
         LD  A,(ATRBYT)   ; FA13
         INC B
         DEC B
         JP P,L49AF       ; linker oder rechter Teil
         ADD A,A
         ADD A,A
         ADD A,A
         ADD A,A
L49AF:
         OR C             ; einblenden
         CALL WRTVDP      ; zurueckschreiben
         POP BC
         POP HL
         RET
L49B6:                    ; ******* spezial rechts
         PUSH HL
         CALL CHKMOD
         JP NZ,L4A75
         CALL FETCHC
         RRCA
         JR NC,L4AOE      ; NC es war nicht die rechte Pixel-
         LD A,L           ;       position im CHR
         AND OF8H
         CP 248
         LD  A,080H       ; neue Bitmaske
         JR NZ,L49DC      ; NZ nicht der CHR ganz rechts
         JP L4A56         ; Ueberlauf nach Y
RIGHTC:  49CF             ; ******* nach rechts
         PUSH HL
         CALL CHKMOD
         JP NZ,L4A87
         CALL FETCHC      ; Koordinaten
         RRCA             ; neue Bitmaske
         JR NC,L4AOE      ; C wenn Ueberlauf in naechsten chr Block
L49DC:
         PUSH DE
         LD  DE,08H       ; chrposition+8
         JR L4A09         ; und speichern
L49E2:                    ; ******* special left
         PUSH HL
         CALL CHKMOD
         JP NZ,L4A98
         CALL FETCHC
         RLCA             ; Bitmaske
         JR NC,L4AOE      ; NC kein Ueberlauf in neuen CHR
         LD A,L
         AND OF8H
         LD  A,01H        ; sonst neue Bitmaske
         JR NZ,L4A05      ; NZ kein y Ueberlauf
         JR L4A56
LEFTC:   49F8             ; ********* Grafik-Cursor links
         PUSH HL
         CALL CHKMOD
         JP NZ,L4AA8
         CALL FETCHC
```

```
        RLCA
        JR NC,L4AOE
L4AO5:
        PUSH DE
        LD  DE,OFFF8H
L4AO9:
        ADD HL,DE
        LD (CHKROM/CLOC),HL
        POP DE
L4AOE:
        LD (CMASK),A
        AND A           ; raus mit NC!
        POP HL
        RET
TDOWNC: 4A14            ; ****** special down
        PUSH HL
        PUSH DE
        LD HL,(CHKROM/CLOC)
        CALL CHKMOD
        JP NZ,L4AC2
        LD  DE,1700H
        RST 20H
        JR C,L4A38      ; C innerhalb des unkritischen Bereichs
        LD A,L
        INC A
        AND 07H
        JR NZ,L4A38
        JR L4A55        ; erzeugt C fuer y Ueberlauf
DOWNC:  4A20            ; ****** Grafik-Cursor abwaerts
        PUSH HL
        PUSH DE
        LD HL,(CHKROM/CLOC)
        CALL CHKMOD
        JP NZ,L4AD2
L4A38:
        INC HL
        LD A,L
        LD  DE,0F8H     ; negativer offset wegen Darstellung VDP
        JR L4A69
TUPC:   4A3F            ; ****** special oben
        PUSH HL
        PUSH DE
        LD HL,(CHKROM/CLOC)
        CALL CHKMOD
        JP NZ,L4AD9
        LD  DE,256
        RST 20H
        JR NC,L4A64     ; NC kann keinen Ueberlauf geben
        LD A,L
        AND 07H
        JR NZ,L4A64     ; kann auch keinen geben
L4A55:
        POP DE
L4A56:
        SCF             ; Signal Ueberlauf fuer Cursorbewegung
        POP HL          ; in Zusammenhang mit T..C Routinen
        RET             ; keine Aenderung der CHR-Koordinaten
UPC:    4A59            ; ****** Grafik-Cursor nach oben
        PUSH HL
        PUSH DE
        LD HL,(CHKROM/CLOC)
```



```
        CALL CHKMOD
        JP NZ,L4AE8
L4A64:
        LD A,L
        DEC HL
        LD  DE,0FF08H   ; Offset fuer aufwaerts
L4A69:
        AND 07H
        JR NZ,L4A6E     ; NZ schon alles durch DEC HL klar
        ADD HL,DE       ; neuer CHR Block
L4A6E:
        LD (CHKROM/CLOC),HL     ; speichern
        AND A           ; NC fuer ok
        POP DE
        POP HL
        RET
L4A75:                  ; ***** TRIGHTC Screen 2
        CALL FETCHC
        AND A
        LD  A,0FH
        JP M,L4ABC
        LD A,L
        AND 0F8H
        CP 0F8H
        JR NZ,L4A90
        JR L4A56
L4A87:                  ; ***** RIGHTC Screen 2
        CALL FETCHC
        AND A
        LD  A,0FH
        JP M,L4ABC
L4A90:
        PUSH DE
        LD  DE,08H
        LD  A,0F0H
        JR L4AB7
L4A98:                  ; ***** TLEFTC Screen 2
        CALL FETCHC
        AND A
        LD  A,0F0H
        JP P,L4ABC
        LD A,L
        AND 0F8H
        JR NZ,L4AB1
        JR L4A56
L4AA8:                  ; ***** LEFTC Screen 2
        CALL FETCHC
        AND A
        LD  A,0F0H
        JP P,L4ABC
L4AB1:
        PUSH DE
        LD  DE,0FFF8H
        LD  A,0FH
L4AB7:
        ADD HL,DE
        LD (CHKROM/CLOC),HL
        POP DE
L4ABC:
        LD (CMASK),A
        AND A
```

```
        POP HL
        RET
L4AC2:                  ; ***** TDOWNC Screen 2
        LD  DE,0500H
        RST 20H
        JR C,L4AD2
        LD A,L
        INC A
        AND 07H
        JR NZ,L4AD2
        SCF
        POP DE
        POP HL
        RET
L4AD2:                  ; ***** DOWNC Screen 2
        INC HL
        LD A,L
        LD  DE,0F8H
        JR L4AED
L4AD9:                  ; ***** TUPC Screen 2
        LD  DE,0100H
        RST 20H
        JR NC,L4AE8
        LD A,L
        AND 07H
        JR NZ,L4AE8
        SCF
        POP DE
        POP HL
        RET
L4AE8:                  ; ***** UPC Screen 2
        LD A,L
        DEC HL
        LD  DE,0FF08H
L4AED:
        AND 07H
        JR NZ,L4AF2
        ADD HL,DE
L4AF2:
        LD (CHKROM/CLOC),HL
        AND A
        POP DE
        POP HL
        RET
NSETCX: 4A F9           ; *********
        CALL CHKMOD
        JP NZ,L4BB7
        PUSH HL         ; Screen 1
        CALL FETCHC
        EX (SP),HL
        ADD A,A
        JR C,L4B1F      ; C  ist Bit ganz links
        PUSH AF
        LD  BC,0FFFFH   ; -1
        RRCA
L4B0C:
        ADD HL,BC
        JR NC,L4B54
        RRCA
        JR NC,L4B0C
        POP AF
```



```
        DEC A
        EX (SP),HL
        PUSH HL
        CALL L4B5C
        POP HL
        LD  DE,08H
        ADD HL,DE
        EX (SP),HL
L4B1F:
        LD A,L
        AND 07H
        LD C,A
        LD A,H
        RRCA
        LD A,L
        RRA
        RRCA
        RRCA
        AND 03FH
        POP HL
        LD B,A
        JR Z,L4B43
L4B2F:
        XOR A
        CALL WRTVDP     ; aktuell auf 0
        LD  DE,GRPCCL
        ADD HL,DE
        LD  A,(ATRBYT)
        CALL WRTVDP     ; zugehoerige Farbe setzen
        LD  DE,0E008H
        ADD HL,DE       ; aktuell+8
        DJNZ L4B2F      ; naechstes Byte
L4B43:
        DEC C
        RET M
        PUSH HL
        LD  HL,L4B4D
        ADD HL,BC
        LD A,(HL)
        JR L4B5B
L4B4D:
        DEFB 080H,0C0H,0E0H,0F0H,0F8H,0FCH,0FEH
L4B54:
        ADD A,A
        DEC A
        CPL
        LD B,A
        POP AF
        DEC A
        AND B
L4B5B:
        POP HL
L4B5C:                  ; ******
        LD B,A          ; Maske
        CALL RDVDP      ; Muster lesen
        LD C,A
        LD  DE,GRPCCL
        ADD HL,DE
        CALL RDVDP      ; Farbe lesen
        PUSH AF         ; sichern
```

```
        AND OFH
        LD E,A
        POP AF
        SUB E
        LD D,A
        LD A,B
        OR C
        CP OFFH
        LD  A,(ATRBYT)
        JR Z,L4B9C
        CP E
        JR Z,L4B8F
        ADD A,A
        ADD A,A
        ADD A,A
        ADD A,A
        CP D
        JR Z,L4B93
        PUSH HL
        PUSH DE
        PUSH AF
        CALL L4B93
        POP AF
        POP DE
        POP HL
        OR E
        JP WRTVDP
L4B8F:
        LD A,B
        CPL
        AND C
        DEFB 011H           ;*      LD  DE,0B178H
L4B93:
        LD A,B              ;*
        OR C                ;*
L4B95:
        LD  DE,0E000H
        ADD HL,DE
        JP WRTVDP
L4B9C:
        CP E
        JR Z,L4BA9
        ADD A,A
        ADD A,A
        ADD A,A
        ADD A,A
        CP D
        JR Z,L4BA9
        LD A,B
        CPL
        DEFB 6              ;*      LD B,0AFH
L4BA9:
        XOR A               ;*
        PUSH HL
        PUSH DE
        CALL L4B95
        POP DE
        POP HL
        LD  A,(ATRBYT)
        JP WRTVDP
L4BB7:                      ; ****** NSETCX Screen 2
```



```
        PUSH HL
        CALL SETC
        CALL RIGHTC
        POP HL
        DEC L
        JR NZ,L4BB7
        RET
GTASPC: 4BC3                ; ****** Voreinstellung Seiten-
        LD  DE,0100H        ;        verhaeltnis
        LD L,E
        LD H,D
        RET
PNTINI: 4BC9                ; ********** Initialisierung fuer PAINT
        CP 010H             ; Farbe speichern C zu gross
        CCF
        LD (BRDATR),A       ; F641
        RET
SCANR:  4BD0                ; ******* Scan rechts brdatr Farbe suchen
        LD  HL,00H          ; B -> FILNAM
        LD C,L              ; DE maximale Pixelzahl zum scannen
        CALL CHKMOD         ; Grafik
        JR NZ,L4C3D
        LD A,B              ; SCANR Screen 1
        LD (FILNAM),A
        XOR A
        LD (0F9A1H),A
        LD  A,(BRDATR)      ; farbe
        LD B,A
L4BE5:
        CALL READC          ; Farbe lesen
        CP B
        JR NZ,L4BF8         ; NZ naechste Position
        DEC DE              ; Bordercolor gefunden
        LD A,D
        OR E
        RET Z
        CALL L49B6          ; rechts
        JR NC,L4BE5
        LD  DE,00H
        RET
L4BF8:
        CALL L4C9A
        PUSH DE
        CALL FETCHC
        LD (CSAVE),HL
        LD (CSAVEM),A
        LD  DE,00H
L4C08:
        INC DE
        CALL L49B6          ; rechts
        JR C,L4C19
        CALL READC          ; Farbe lesen
        CP B
        JR Z,L4C19          ; identisch?
        CALL L4C9A
        JR L4C08
L4C19:
        PUSH DE
        CALL FETCHC         ; Koordinaten
        PUSH HL
        PUSH AF             ; sichern
```

```
        LD HL,(CSAVE)
        LD  A,(CSAVEM)
        CALL STOREC     ; alte Koordinaten
        EX DE,HL
        LD (OF99FH),HL
        LD  A,(FILNAM)
        AND A
        CALL NZ,NSETCX
        POP AF
        POP HL
        CALL STOREC     ; aktuelle setzen
        POP HL
        POP DE
        JP L4C95
L4C3D:                  ; ****** SCANR Screen 2
        CALL L4CB3
        JR NC,L4C4F
        DEC DE
        LD A,D
        OR E
        RET Z
        CALL L49B6
        JR NC,L4C3D
        LD  DE,00H
        RET
L4C4F:
        CALL FETCHC     ; Koordinaten
        LD (CSAVE),HL
        LD (CSAVEM),A   ; speichern
        LD  HL,00H
L4C5B:
        INC HL
        CALL L49B6      ; rechts
        RET C
        CALL L4CB3
        JR NC,L4C5B
        RET
SCANL:  4C66            ; ****** scan left
        LD  HL,00H
        LD C,L
        CALL CHKMOD
        JR NZ,L4CA6
        XOR A
        LD (OF9A1H),A
        LD  A,(BRDATR)
        LD B,A
L4C77:
        CALL L49E2      ; links
        JR C,L4C8B
        CALL READC
        CP B
        JR Z,L4C88
        CALL L4C9A
        INC HL
        JR L4C77
L4C88:
        CALL RIGHTC
L4C8B:
        PUSH HL
        LD DE,(OF99FH)
        ADD HL,DE
```



```
        CALL NSETCX
        POP HL
L4C95:                  ; ****
        LD  A,(OF9A1H)
        LD C,A
        RET
L4C9A:                  ; ****
        PUSH HL
        LD  HL,ATRBYT
        CP (HL)
        POP HL
        RET Z
        INC A
        LD (OF9A1H),A
        RET
L4CA6:                  ; ***** scan1 screen 2
        CALL L49E2
        RET C
        CALL L4CB3
        JP C,RIGHTC
        INC HL
        JR L4CA6
L4CB3:                  ; ***** Farbe setzen falls ungleich
        CALL READC      ;  Begrenzer und ungleich der gesetzten-Farbe
        LD B,A
        LD  A,(BRDATR)
        SUB B
        SCF
        RET Z           ; ret mit Z und C falls gleich
        LD  A,(ATRBYT)  ;       Begrenzerfarbe
        CP B
        RET Z           ; ret Z NC falls gleich der Farbe
        CALL SETC       ; ungleich Farbe setzen
        LD  C,01H
        AND A           ; NC NZ Farbe gesetzt
        RET
PIXSIZ: 4CC9            ; ****** Pixelgroesse
        CALL CHKMOD
        LD  A,04H       ; 4
        RET
PGINIT: 4CCF            ; ****** Initialisierung PUT grafik
        LD (ARYPTR),HL
        PUSH BC
        ADD A,A
        LD C,A
        LD B,00H
        LD  HL,L4CEF
        ADD HL,BC
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        LD (OFA15H),HL  ; Sprungadresse
        POP HL
        SRL H
        RR L
        SRL H
        RR L
        LD (CSAVE),HL   ; BC DIV 4
        RET
```

```
L4CEF:
        DEFW L4CF9
        DEFW L4CFB
        DEFW L4CFF
        DEFW L4D00
        DEFW L4CFD

L4CF9:                  ; Grafik PUT OR
        OR E
        RET
L4CFB:                  ; Grafik PUT AND
        AND E
        RET
L4CFD:                  ; Grafik PUT XOR
        XOR E
        RET
L4CFF:                  ; Grafik PUT PRESET
        CPL
L4D00:                  ; Grafik PUT PSET
        RET

NREAD:  4D01            ; **********
        CALL L4D78
L4D04:
        PUSH BC
        PUSH DE
        PUSH HL
        CALL READC
        PUSH AF         ; Farbe speichern
        CALL RIGHTC
        POP AF
        POP HL
        POP DE
        POP BC
        CALL L4D50
        DEC BC
        LD A,B
        OR C
        JR NZ,L4D04     ; mehr ?
        LD A,D
        DEC A
        CALL Z,L4D50
        JR L4D4C
NWRITE: 4D21            ; ***********
        CALL L4D78
        LD D,01H
        PUSH HL
        PUSH DE
L4D28:
        PUSH BC
        CALL READC
        POP BC
        POP DE
        POP HL
        LD E,A
        CALL L4D63
        PUSH HL
        PUSH DE
        PUSH BC
        CALL PUTFN
        AND 0FH
```



```
        LD (ATRBYT),A
        CALL SETC
        CALL RIGHTC
        POP BC
        DEC BC
        LD A,B
        OR C
        JR NZ,L4D28
        POP DE
        POP HL
L4D4C:
        LD (ARYPTR),HL
        RET
L4D50:
        LD E,A
        CALL L4D6D
        OR E
        LD (CSAVEM),A
        DEC D
        RET NZ
        LD (HL),A
        INC HL
        LD D,02H
        XOR A
        LD (CSAVEM),A
        RET
L4D63:
        DEC D
        JR NZ,L4D6D
        LD A,(HL)
        LD (CSAVEM),A
        INC HL
        LD D,02H
L4D6D:
        LD  A,(CSAVEM)
        RLCA
        RLCA
        RLCA
        RLCA
        LD (CSAVEM),A
        RET
L4D78:
        LD BC,(CSAVE)
        LD HL,(ARYPTR)
        XOR A
        LD (CSAVEM),A
        LD D,02H
        RET


DECSUB/DSUB: 4D86       ;***********
        LD  HL,ARG
        LD A,(HL)
        OR A
        RET Z           ; ret falls 0
        XOR 080H        ; -ARG
        LD (HL),A
        JR L4D9A
DADOS:  4D91            ; **********
        CALL VMOVAM     ; ARG setzen aus HL
```

```
        RET NC  ; NC falls kein Dezimalstellenshift noetig
        LD A,C
L4EDE:                  ; ******** A Bytes nach rechts shiften
        PUSH BC
        PUSH DE
        LD D,A
        LD  C,04H
L4EE3:
        LD B,D
        PUSH HL
        OR A
L4EE6:
        LD A,(HL)
        RRA
        LD (HL),A
        INC HL
        DJNZ L4EE6
        POP HL
        DEC C
        JR NZ,L4EE3
        POP DE
        POP BC
        RET

DECSRD: /4EF3           ; ******** FACCU um eine dezimale nach
        LD  HL,FACCU+1  ;           rechts shiften
L4EF6:                  ; ******** eine Zahl ...
        LD  A,08H       ;                     HL^ auf mantisse
        JR L4EDE
L4EFA:                  ; wird Dezimalstellenshift benoetigt ?
        POP AF
        RET NC          ; NC nein
        JR L4EF6

DECMUL/DMULT: /4EFE     ; ***** Multiplikation FACCU:=FACCU*ARG
        RST SIGN
        RET Z           ; Z falls FACCU-Vorzeichen negativ
        LD  A,(ARG)
        OR A
        JP Z,ZERO       ; falls Exponent -64
        LD B,A
        LD  HL,FACCU
        XOR (HL)
        AND 080H
        LD C,A          ; 80 falls Vorzeichen verschieden sonst 0
        LD A,B
        AND 07FH
        LD B,A          ; ARG Vorzeichen positiv
        LD A,(HL)
        AND 07FH
        ADD A,B
        LD B,A          ; Summe der Exponenten
        LD (HL),00H     ; loeschen exp FACCU
        AND 0C0H
        RET Z
        CP 0C0H         ; es darf nur eins von beiden gesetzt sein
        JR NZ,L4F24
        JP DVERR
L4F24:
        LD A,B
        ADD A,040H      ; Exponent+64
```



```
        AND 07FH
        RET Z           ; Z falls Exponent 0 war   ( FACCU*1 )
        OR C            ; Vorzeichen einblenden
        DEC HL
        LD (HL),A       ; Ergebnis-Exp und Vorzeichen speichern
        LD  DE,ARG-2
        LD  BC,08H
        LD  HL,FACCU+7
        PUSH DE
        LDDR            ; move FACCU vor ARG
        INC HL
        XOR A
        LD B,08H
L4F3D:
        LD (HL),A
        INC HL
        DJNZ L4F3D      ; Loeschen FACCU
        POP DE
        LD  BC,L4F9B
        PUSH BC
DECMRN: /4F46
        CALL L4FA2
        PUSH HL
        LD  BC,08H
        EX DE,HL
        LDDR
        EX DE,HL
        LD  HL,ARG-10
        LD B,08H
        CALL L4E5D      ; Addieren de:=de+hl
        POP DE
        CALL L4FA2
        LD  C,07H
        LD  DE,ARG+7
L4F62:
        LD  A,(DE)
        OR A
        JR NZ,L4F6A
        DEC DE
        DEC C
        JR L4F62
L4F6A:
        LD  A,(DE)
        DEC DE
        PUSH DE
        LD  HL,0F93AH   ; FACCU+23
L4F70:
        ADD A,A
        JR C,L4F7B
        JR Z,L4F89
L4F75:
        LD  DE,08H
        ADD HL,DE
        JR L4F70

L4F7B:
        PUSH AF
        LD B,08H
        LD  DE,FACCU+7
        PUSH HL
        CALL L4E5D
```

```
COS:/ 5008          ; ************************************
      LD  HL,0548EH   ; 1/(2*PI)
      CALL DMULTO     ; faccu/(2*pi)
      LD  A,(FACCU)
      AND 07FH
      LD (FACCU),A
      LD  HL,0544EH   ; 1/4
      CALL L535B      ; cos auf sin abbilden
      CALL NEG
      JR L50D7
SIN:/ 5001
      LD  HL,0548EH
      CALL DMULTO     ; faccu/(2*PI)
L50D7:
      LD  A,(FACCU)
      OR A
      CALL M,L53A8
      CALL PHF        ; push faccu
      CALL VINT       ; integer
      CALL MAF        ; move faccu -> arg
      CALL PPF        ; pop faccu
      CALL DECSUB/DSUB; faccu-arg
      LD  A,(FACCU)
      CP 040H
      JP C,L511A
      LD  A,(FACCU+1)
      CP 025H
      JP C,L511A
      CP 075H
      JP NC,L5111
      CALL MAF        ;       -> arg
      LD  HL,0543CH   ; 1/2
      CALL MFM        ;       -> faccu
      CALL DECSUB/DSUB
      JP L511A
L5111:
      LD  HL,ONE
      CALL MAM
      CALL DECSUB/DSUB
L511A:
      LD  HL,0551AH   ; adresse der Koeffizienten
      JP L53BB        ; Reihensumme berechnen
TAN:/ 5120    ;****************************************
      CALL PHF
      CALL COS
      CALL XTF
      CALL SIN        ; sin -> faccu
      CALL PPA        ; cos -> arg
      LD  A,(ARG)
      OR A
      JP NZ,DDIV/DECDIV       ; sin/cos
      JP DVERR
ATN:/ 5139          ; ********************************
      LD  A,(FACCU)
      OR A
      RET Z
      CALL M,L53A8
      CP 041H
      JP C,L5161
      CALL MAF        ; faccu -> arg
      LD  HL,ONE
```



```
      CALL MFM        ; 1     -> faccu
      CALL DDIV/DECDIV        ; 1/x
      CALL L5161
      CALL MAF        ; faccu -> arg
      LD  HL,0546EH   ; PI/2
      CALL MFM
      JP DECSUB/DSUB  ; PI/2-
L5161:
      LD  HL,05476H   ; 0.2679..
      CALL L5370
      JP M,L5191
      CALL PHF
      LD  HL,0547EH   ; SQR(3)
      CALL L5355
      CALL XTF
      LD  HL,0547EH
      CALL DMULTO
      LD  HL,ONE
      CALL L535B
      CALL PPA
      CALL DDIV/DECDIV
      CALL L5191
      LD  HL,05486H   ; 0.523598...
      JP L5355
L5191:
      LD  HL,0555BH   ; Koeffizienten
      JP L53BB
LOG:/ 5197          ; ************************************
      RST 28H
      JP M,FCERR
      JP Z,FCERR
      LD  HL,FACCU
      LD A,(HL)
      PUSH AF
      LD (HL),041H
      LD  HL,05456H   ; SQR(10)
      CALL L5370
      JP M,L51B5
      POP AF
      INC A
      PUSH AF
      LD  HL,FACCU
      DEC (HL)
L51B5:
      POP AF
      LD (TEMP3),A
      CALL PHF
      LD  HL,ONE
      CALL L5355
      CALL XTF
      LD  HL,ONE
      CALL L535B
      CALL PPA
      CALL DDIV/DECDIV
      CALL PHF
      CALL L5361
      CALL PHF
      CALL PHF
      LD  HL,054F1H   ; Koeffizienten
      CALL L53CD
      CALL XTF
```

```
        LD  HL,054D0H    ; Koeffizienten
        CALL L53CD
        CALL PPA
        CALL DDIV/DECDIV
        CALL PPA
        CALL DECMUL/DMULT
        LD  HL,0545EH    ; LG(e^2)
        CALL L5355
        CALL PPA
        CALL DECMUL/DMULT
        CALL PHF
        LD  A,(TEMP3)
        SUB 041H
        LD L,A
        ADD A,A
        SBC A,A
        LD H,A
        CALL CONSIH
        CALL CONDS
        CALL PPA
        CALL DADO/DECADD
        LD  HL,05466H    ; 1/LG(e)
        JP DMULTO
SQR:/   5222             ; *********************************
        RST 28H
        RET Z
        JP M,FCERR
        CALL MAF
        LD  A,(FACCU)
        OR A
        RRA
        ADC A,02OH
        LD (ARG),A
        LD  A,(FACCU+1)
        OR A
        RRCA
        OR A
        RRCA
        AND 033H
        ADD A,01OH
        LD (ARG+1),A
        LD  A,07H
L5244:
        LD (TEMP3),A
        CALL PHF
        CALL PHA
        CALL DDIV/DECDIV
        CALL PPA
        CALL DADO/DECADD
        LD  HL,0543CH    ; 1/2
        CALL DMULTO
        CALL MAF
        CALL PPF
        LD  A,(TEMP3)
        DEC A
        JR NZ,L5244
        JP MFA
EXP:/   5268             ; ************************************
        LD  HL,05434H    ; LG(e)
        CALL DMULTO
        CALL PHF
```



```
        CALL FRCINT
        LD A,L
        RLA
        SBC A,A
        CP H
        JR Z,L5291
        LD A,H
        OR A
        JP P,L528E
        CALL VALDBL
        CALL PPF
        LD  HL,DBLZER
        JP MFM
L528E:
        JP DVERR
L5291:
        LD (TEMP3),HL
        CALL FRCDBL
        CALL MAF
        CALL PPF
        CALL DECSUB/DSUB
        LD  HL,0543CH    ; 1/2
        CALL L5370
        PUSH AF
        JR Z,L52B1
        JR C,L52B1
        LD  HL,0543CH    ; 1/2
        CALL L535B
L52B1:
        CALL PHF
        LD  HL,054B7H    ; Koeffizienten
        CALL L53BB
        CALL XTF
        LD  HL,05496H    ; Koeffizienten
        CALL L53BO
        CALL PPA
        CALL PHA
        CALL PHF
        CALL DECSUB/DSUB
        LD  HL,ARG-9
        CALL MMF
        CALL PPA
        CALL PPF
        CALL DADO/DECADD
        LD  HL,ARG-9
        CALL MAM
        CALL DDIV/DECDIV
        POP AF
        JR C,L52F2
        JR Z,L52F2
        LD  HL,05456H    ; SQR(10)
        CALL DMULTO
L52F2:
        LD  A,(TEMP3)
        LD  HL,FACCU
        LD C,(HL)
        ADD A,(HL)
        LD (HL),A
        XOR C
        RET P
        JP DVERR
```

```
RND:/ 5300          ; ************************************
      RST 28H
      LD  HL,RNDX
      JR Z,L5334    ; Z FACCU=0 alte Zahl nehmen und ausgeben
      CALL M,MMF    ; M FACCU<0 zur berechnung neuer
      LD  HL,ARG-9  ;             zufallszahl benutzen
      LD  DE,RNDX
      CALL L5392
      LD  HL,05424H ; 0.211324*10^-64
      CALL MAM
      LD  HL,0541CH ; 0.143898*10^-64
      CALL MFM
      LD  DE,ARG-2
      CALL DECMRN
      LD  DE,FACCU+8
      LD  HL,0F985H
      LD B,07H
      CALL MOVE1
      LD  HL,RNDX
      LD (HL),00H
L5334:
      CALL MFM
      LD  HL,FACCU
      LD (HL),040H
      XOR A
      LD (FACCU+8),A
      JP DECNRM
RNDINI:/ 5343       ; ********************************
      LD  DE,0542CH ; ... *10-64
      LD  HL,RNDX
      JP L5392
RNDMN2:/ 534C
      CALL CONSIH
      LD  HL,RNDX
      JP MMF
L5355:              ; ********** FACCU:=FACCU+HL
      CALL MAM
      JP DADO/DECADD
L535B:              ; ********** FACCU:=FACCU-HL
      CALL MAM
      JP DECSUB/DSUB
L5361:              ; ********** FACCU^2
      LD  HL,FACCU
DMULTO:/ 5364       ; ********** FACCU:=FACCU*HL
      CALL MAM
      JP DECMUL/DMULT
L536A:              ; ********** FACCU:=FACCU/HL
      CALL MAM
      JP DDIV/DECDIV
L5370:              ; ********** FACCU mit HL vergleichen
      CALL MAM
      JP XDCOMP
MAF: / 5376         ; ********** MOVE FACCU -> ARG
      LD  HL,FACCU
MAM: / 5379         ; ********** MOVE HL    -> ARG
      LD  DE,ARG
L537C:              ; ********** MOVE HL    -> DE
      EX DE,HL
      CALL L5392    ; DE -> HL
      EX DE,HL
      RET
```



```
MFA:/ 5382          ; ********** MOVE ARG   -> FACCU
      LD  HL,ARG
MFM:/ 5385          ; ********** MOVE HL    -> FACCU
      LD  DE,FACCU
      JR L537C
MMA:/ 538A          ; ********** MOVE ARG   -> HL
      LD  DE,ARG
      JR L5392
MMF:/ 538F          ; ********** MOVE FACCU -> HL
      LD  DE,FACCU
L5392:              ; ********** MOVE DE    -> HL
      LD B,08H
      JP MOVE1 5626
XTF:/ 5397          ; **** Austausch Top of Stack gegen FACCU
      POP HL        ; ARG wird benutzt
      LD (FBUFFR),HL
      CALL PPA      ; pop arg
      CALL PHF      ; push faccu
      CALL MFA      ; arg -> faccu
      LD HL,(FBUFFR)
      JP (HL)
L53A8:              ; ************ FACCU wird negiert
      CALL NEG      ; danach RET zum Aufrufer, gibt dieser RET
      LD  HL,NEG    ; wird FACCU noch einmal negiert
      EX (SP),HL
      JP (HL)
L53B0:              ; *********** Reihensumme mi X^2 beginnen
      LD (FBUFFR),HL
      CALL L5361    ; FACCU^2
      LD HL,(FBUFFR)
      JR L53CD      ; reihensumme
L53BB:              ; *********** = (K1*FACCU^2+K2...)*FACCU
      LD (FBUFFR),HL
      CALL PHF
      LD HL,(FBUFFR)
      CALL L53B0
      CALL PPA
      JP DECMUL/DMULT; FACCU^3
L53CD:              ; ************* HORNER mit Koeffizienten
      LD A,(HL)     ;                Adresse in HL
      PUSH AF       ; anzahl Koeffizienten
      INC HL
      PUSH HL       ; Koeffizientenadresse
      LD  HL,FBUFFR
      CALL MMF      ; x-potenz -> FBUFFR
      POP HL
      CALL MFM      ; Koeffizient -> FACCU
L53DB:
      POP AF
      DEC A
      RET Z    ; Z kein weiterer Koeffizient
      PUSH AF
      PUSH HL
      LD  HL,FBUFFR
      CALL DMULTO   ; Koeffizient*x-Potenz
      POP HL
      CALL MAM      ; neuer Koeffizient -> ARG
      PUSH HL
      CALL DADO/DECADD; Produkt+ARG
      POP HL
      JR L53DB
```

```
        ; ******** Transferfunktionen fuer doppelt genaue Zahlen

PHA:/ 53F1              ; ************** PUSH ARG
        LD  HL,ARG+7
        JR L53F9
PHF:/ 53F6              ; ************** PUSH FACCU
        LD  HL,FACCU+7
L53F9:                  ; ************** PUSH HL
        LD  A,04H
        POP DE
L53FC:
        LD B,(HL)
        DEC HL
        LD C,(HL)
        DEC HL
        PUSH BC
        DEC A
        JR NZ,L53FC
        EX DE,HL
        JP (HL)
PPA:/ 5406              ; ************** POP ARG
        LD  HL,ARG
        JP L540F
PPF:/ 540C              ; ************** POP FACCU
        LD  HL,FACCU
L540F:                  ; ************** POP HL
        LD  A,04H
        POP DE
L5412:
        POP BC
        LD (HL),C
        INC HL
        LD (HL),B
        INC HL
        DEC A
        JR NZ,L5412
        EX DE,HL
        JP (HL)
L541C:

        .RADIX 16
L541C:
        DEFB 000,014,038,098,020,042,008,021

L5424:
        DEFB 000,021,013,024,086,054,005,019

L542C:
        DEFB 000,040,064,096,051,037,023,058

L5434:                  ; LG(e)
        DEFB 040,043,042,094,048,019,003,024

L543C:                  ; 0.5
        DEFB 040,050

DLBZER:/ 5446           ; 0
        DEFB 000,000,000,000,000,000,000,000
```



```
ONE:/ 5446              ; 1
        DEFB 041,010,000,000,000,000,000,000

L544E:                  ; 0.25
        DEFB 040,025,000,000,000,000,000,000

L5456:                  ; wurzel 10
        DEFB 041,031,062,027,076,060,016,084

L545E:                  ; LG(e^2)
        DEFB 040,086,085,088,096,038,006,050

L5466:                  ; M     1/LG(e)
        DEFB 041,023,002,058,050,092,099,040

L546E:                  ; PI/2
        DEFB 041,015,070,079,063,026,079,049

L5476:
        DEFB 040,026,079,049,019,024,031,012

L547E:                  ; wurzel 3
        DEFB 041,017,032,005,008,007,056,089

L5486:
        DEFB 040,052,035,098,077,055,098,030

L548E:                  ; 1/(2*PI)
        DEFB 040,015,091,054,094,030,091,090

L5496:
        DEFB 4
        DEFB 041,010,000,000,000,000,000,000
        DEFB 043,015,093,074,015,023,060,031
        DEFB 044,027,009,031,069,040,085,016
        DEFB 044,044,097,063,035,057,040,058

L54B7:
        DEFB 3
        DEFB 042,018,031,023,060,015,092,075
        DEFB 043,083,014,006,072,012,093,071
        DEFB 044,051,078,009,019,091,051,062

L54D0:
        DEFB 4
        DEFB 0C0,071,043,033,082,015,032,026
        DEFB 041,062,050,036,051,012,079,008
        DEFB 0C2,013,068,023,070,024,015,003
        DEFB 041,085,016,073,019,087,023,089

L54F1:
        DEFB 5
        DEFB 041,010,000,000,000,000,000,000
        DEFB 0C2,013,021,004,078,035,001,056
        DEFB 042,047,092,052,056,004,038,073
        DEFB 0C2,064,090,066,082,074,009,043
        DEFB 042,029,041,057,050,017,023,023

L551A:
        DEFB 8
        DEFB 0C0,069,021,056,092,029,018,009
```

```
        DEFB 041,038,017,028,086,038,057,071
        DEFB 0C2,015,009,044,099,047,048,001
        DEFB 042,042,005,086,089,066,073,055
        DEFB 0C2,076,070,058,059,068,032,091
        DEFB 042,081,060,052,049,027,055,013
        DEFB 0C2,041,034,017,002,024,003,098
        DEFB 041,062,083,018,053,007,017,096

L555B:
        DEFB 8
        DEFB 0BF,052,008,069,039,004,000,000
        DEFB 03F,075,030,071,049,013,048,000
        DEFB 0BF,090,081,034,032,024,070,050
        DEFB 040,011,011,007,094,018,040,029
        DEFB 0C0,014,028,056,008,055,048,084
        DEFB 040,019,099,099,099,094,089,067
        DEFB 0C0,033,033,033,033,033,031,060
        DEFB 041,010,000,000,000,000,000,000

        .RADIX 10

L559C:          ; ********* SGN fuer FLOATINGpoint =^ RST 28H
        LD  A,(FACCU)
        OR A
        RET Z
SIGNC:  55A1            ; ******** Vorzeichen-Test
        LD  A,(FACCU)   ;        falls negativ  A=-1
        JR L55A7        ;        falls positiv  A=1
                        ; *********
L55A6:                  ;        falls positiv  A=-1
        CPL             ;        falls negativ  A=1
L55A7:                  ;
        RLA
SIGNS:  55A8            ; ********* falls Carry   -1
        SBC A,A         ;           sonst         1 in A
        RET NZ
INRART: 55AA
        INC A
        RET
ZERO:   55AC            ; Nullsetzen des FACCU Exponenten
        XOR A
        LD (FACCU),A
        RET
ABSFN:  55B1            ; ***************************************
        CALL VSIGN      ; ABS(FACCU)
        RET P
VNEG:   55B5            ; *************** FACCU:=-FACCU
        RST 30H         ; je nach Datentyp wird negiert
        JP M,INEG
        JP Z,TMERR
NEG:    55BC            ; *************** FACCU:=-FACCU
        LD  HL,FACCU    ; fuer Floatingpoint
        LD A,(HL)
        OR A
        RET Z   ; Z falls 0
        XOR 080H
        LD (HL),A
        RET
SGN:    55C6            ; ***************************************
        CALL VSIGN      ; SGN(FACCU)
CONIA:  55C9            ; ******* convert A to INTEGER in FACCU
        LD L,A          ; 0 1 -1
```



```
        RLA
        SBC A,A
        LD H,A          ; 0 oder OFFH
        JP MAKINT
VSIGN:  55D0            ; *********** SGN unabhengig vom Datentyp
        RST 30H
        JP Z,TMERR
        JP P,L559C      ; kein integer
        LD HL,(FACLOW)
ISIGN:  55DA            ; ********* SGN(HL) als INTEGER
        LD A,H
        OR L
        RET Z
        LD A,H
        JR L55A7

        ; Transferfunktionen fuer einfachgenaue Zahlen
        ; R Registeradresse fuer einfach genaue Zahlen wie folgt
        ;       C               B       E       D
        ;       Vorzeichen      MSD     ...     LSD
        ;        Exponent

PUSHF:  55E0            ; ****************** PUSH FACCU
        EX DE,HL
        LD HL,(FACLOW)
        EX (SP),HL
        PUSH HL
        LD HL,(FACCU)
        EX (SP),HL
        PUSH HL
        EX DE,HL
        RET
MOVFM:  55ED            ; ******************* move HL -> FACCU
        CALL MOVRM
MOVFR:  55F0            ; ******************* move R  -> FACCU
        EX DE,HL
        LD (FACLOW),HL
        LD H,B
        LD L,C
        LD (FACCU),HL
        EX DE,HL
        RET
MOVRF:  55FB            ; ******************* move FACCU -> R
        LD HL,(FACLOW)
        EX DE,HL
        LD HL,(FACCU)
        LD C,L
        LD B,H
        RET
MOVRMI: 5605            ; ******************* move HL -> R
        LD C,(HL)       ; mit tausch DE <-> BC
        INC HL
        LD B,(HL)
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        INC HL
        RET
MOVRM:  560E            ; ******************* move HL -> R
        LD E,(HL)
```

```
        INC HL
GETBCD: 5610
        LD D,(HL)
        INC HL
        LD C,(HL)
        INC HL
        LD B,(HL)
INXHRT: 5615
        INC HL
        RET
MOVMF:  5617            ; ******************** move FACCU -> HL
        LD  DE,FACCU
MOVE:   561A
        LD B,04H
        JR MOVE1

        ;*************** transfers fuer jeden Datentyp
VMOVAM: 561E            ; *************** MOVE  HL -> ARG
        LD  DE,ARG
L5621:                  ; *************** MOVE  HL -> DE
        EX DE,HL
VMOVE:  5622            ; *************** MOVE  DE -> HL
        LD  A,(VALTYP)
        LD B,A
MOVE1:  5626            ; move von vorn
        LD  A,(DE)
        LD (HL),A
        INC DE
        INC HL
        DJNZ MOVE1
        RET
MOVE1R: 562D            ; move von hinten
        LD  A,(DE)
        LD (HL),A
        DEC DE
        DEC HL
        DJNZ MOVE1R
        RET


        ; ******* INTEGER Transfers
VMOVFA: 5634            ; ********* move ARG -> FACLOW
        LD  HL,ARG
VMOVFM: 5637            ; ********* move HL  -> FACLOW
        LD  DE,L5621    ; HL->DE
        JR L5642
VMOVAF: 563C            ; ********* move FACLOW -> ARG
        LD  HL,ARG
VMOVMF: 563F            ; ********* move FACLOW -> HL
        LD  DE,VMOVE
L5642:
        PUSH DE
VDFACS: 5643
        LD  DE,FACCU
        LD  A,(VALTYP)
        CP 02H
        RET NZ
        LD  DE,FACLOW
        RET
FCOMP:  5C59            ; ******* Vergleich einfachgenauer Zahlen
        LD A,C          ;         1 wenn R < FACCU
```



```
        OR A            ;         0 wenn R = FACCU
        JP Z,L559C      ;        -1 wenn R > FACCU
        LD  HL,L55A6
        PUSH HL
        RST 28H
        LD A,C
        RET Z           ; Z falls FACCU=0
        LD  HL,FACCU
        XOR (HL)
        LD A,C
        RET M           ; verschieden im Vorzeichen
        CALL L5668
        RRA
        XOR C
        RET
L5668:                  ; ************
        LD A,C          ; Exponenten und Mantissenvergleich
        CP (HL)
        RET NZ          ; Exp verschieden
        INC HL
        LD A,B
        CP (HL)
        RET NZ
        INC HL
        LD A,E
        CP (HL)
        RET NZ
        INC HL
        LD A,D
        SUB (HL)
        RET NZ
        POP HL
        POP HL
        RET
ICOMP:  567A            ; ************ Integervergleich DH   EL
        LD A,D          ;         1 wenn DE < HL
        XOR H           ;         0 wenn DE = HL
        LD A,H          ;        -1 wenn DE > HL
        JP M,L55A7
        CP D
        JR NZ,L5686
        LD A,L
        SUB E
        RET Z
L5686:
        JP SIGNS
XDCOMP: 5689            ; ****** Vergleich doppelt genauer Zahlen
        LD  DE,ARG
        LD  A,(DE)
        OR A
        JP Z,L559C
        LD  HL,055A6H
        PUSH HL
        RST 28H
        LD  A,(DE)
        LD C,A
        RET Z
        LD  HL,FACCU
        XOR (HL)
        LD A,C
        RET M
```

```
        LD B,08H
L56A1:
        LD  A,(DE)
        SUB (HL)
        JR NZ,L56AB
        INC DE
        INC HL
        DJNZ L56A1
        POP BC
        RET
L56AB:
        RRA
        XOR C
        RET
DCOMP:  56AE            ; ****** Vergleich doppelt genauer Zahlen
        CALL XDCOMP     ;        1 wenn ARG < FACCU
        JP NZ,L55A6     ;        0 wenn ARG = FACCU
        RET             ;       -1 wenn ARG > FACCU


FRCINT: 56B5            ; ************************************
        RST 30H         ; CINT(FACCU)
        LD HL,(FACLOW)  ; alles nach INTEGER wandeln
        RET M           ; ist schon Integer
        JP Z,TMERR      ; falls string
CONIS:  56BD
        CALL QINTA
        JP C,DVERR
        EX DE,HL
MAKINT: 56C4            ; ****** HL nach FACCU als INTEGER
        LD (FACLOW),HL
VALINT:                 ; ****** FACCU als INTEGER kennzeichnen
        LD A,02H
L56C9:  56C7            ; ****** FACCU typ setzen
        LD (VALTYP),A
        RET

CONIS2: 56CD            ; ******
        LD  BC,032C5H
        LD  DE,08076H   ; -32768
        CALL FCOMP      ; Vergleich
        RET NZ
        LD  HL,08000H   ; -32768 als Integer
L56DA:  56DD
        POP DE
        JR MAKINT


FRCSNG: 56E5            ; ************************************
        RST 30H         ; CSNG
        RET PO
        JP M,CONSI
        JP Z,TMERR
CONSD:  56F3            ; ********** convert double -> single
        CALL VALSNG
        CALL NUMLEN
        INC HL
        LD A,B
        OR A
        RRA
        LD B,A
```



```
        JP DECROB
CONSI:  56F6            ; ********** INTEGER --> single
        LD HL,(FACLOW)
CONSIH: 56F9            ; ********** INTEGER HL --> single
        LD A,H
L56F7:
        OR A
        PUSH AF
        CALL M,INEGHL
        CALL VALSNG
        EX DE,HL
        LD  HL,00H
        LD (FACCU),HL
        LD (FACLOW),HL
        LD A,D
        OR E
        JP Z,PPSWRT
        LD  BC,0500H
        LD  HL,FACCU+1
        PUSH HL
        LD  HL,L575B
L5718:
        LD  A,0FFH
        PUSH DE
        LD E,(HL)
        INC HL
        LD D,(HL)
        INC HL
        EX (SP),HL
        PUSH BC
L5721:
        LD B,H
        LD C,L
        ADD HL,DE
        INC A
        JR C,L5721
        LD H,B
        LD L,C
        POP BC
        POP DE
        EX DE,HL
        INC C
        DEC C
        JR NZ,L573B
        OR A
        JR Z,L574F
        PUSH AF
        LD  A,040H
        ADD A,B
        LD (FACCU),A
        POP AF
L573B:
        INC C
        EX (SP),HL
        PUSH AF
        LD A,C
        RRA
        JR NC,L574A
        POP AF
        ADD A,A
        ADD A,A
```

```
        ADD A,A
        ADD A,A
        LD (HL),A
        JR L574E
L574A:
        POP AF
        OR (HL)
        LD (HL),A
        INC HL
L574E:
        EX (SP),HL
L574F:
        LD A,D
        OR E
        JR Z,L5755
        DJNZ L5718
L5755:
        POP HL
        POP AF
        RET P
        JP NEG

L575B:
        DEFB OFOH,0D8H,018H,OFCH,09CH,OFFH,OF6H,OFFH,OFFH,OFFH

FRCDBL:/5765    ; ************************************
        RST 30H ; CDBL
        RET NC
        JP Z,TMERR
        CALL M,CONSI    ; falls Integer erst Integer -> Single
CONDS:/5769     ; ************ Single -> Double
        LD  HL,00H
        LD (FACCU+4),HL
        LD (FACCU+6),HL
        LD A,H
        LD (FACCU+8),A
VALDBL:/576A    ; ********** typsetzen FACCU=Double
        LD  A,08H
        JR L5780
VALSNG:/5776    ; ********** typsetzen FACCU=Single
        LD  A,04H
L5780:
        JP L56C9

CHKSTR/FRCSTR:/5783     ; ********* FACCU String ?
        RST 30H
        RET Z
        JP TMERR        ; sonst Type mismatch
QINTA:/5788     ; ********** FACCU floating --> INTEGER
        LD  HL,L57E5    ; Ergebnis in HL
        PUSH HL
        LD  HL,FACCU
        LD A,(HL)
        AND 07FH
        CP 046H
        RET NC          ; NC Exponent zu gross
        SUB 041H
        JR NC,L579F     ; C Exponent zu klein
        OR A
        POP DE
```



```
        LD  DE,00H
        RET
L579F:
        INC A           ; richtiger Exponent
        LD B,A          ; nach B
        LD  DE,00H      ; Null setzen alter Inhalt
        LD C,D          ; C zaehlt Dezimalstellen
        INC HL          ; erstes Zahlenbyte
L57A6:
        LD A,C
        INC C
        RRA             ; welche Dezimalstelle
        LD A,(HL)
        JR C,L57B2      ; C falls ungerade Stelle
        RRA
        RRA
        RRA
        RRA             ; andere Dezimalstelle
        JR L57B3
L57B2:
        INC HL          ; naechstes mal neues Byte
L57B3:                  ; ************* HL:=HL*10+A
        AND OFH
        LD (DECTMP),HL
        LD H,D
        LD L,E
        ADD HL,HL
        RET C
        ADD HL,HL
        RET C
        ADD HL,DE
        RET C
        ADD HL,HL
        RET C
        LD E,A
        LD D,00H
        ADD HL,DE
        RET C
        EX DE,HL
        LD HL,(DECTMP)
        DJNZ L57A6
        LD  HL,08000H
        RST 20H
        LD  A,(FACCU)
        RET C           ; C falls >8000H geworden
        JR Z,L57E1      ; falls gleich
        POP HL
        OR A
        RET P           ; P falls Zahl positiv war
        EX DE,HL
        CALL INEGHL     ; sonst noch negieren
        EX DE,HL
        OR A
        RET
L57E1:
        OR A
        RET P
        POP HL
        RET
L57E5:
        SCF
```

```
        RET
DCXBRT: 57E7
        DEC BC
        RET
*FIXER: 57E9            ; ***************************************
        RST 30H         ; FIX
        RET M   ; falls INTEGER
        RST 28H
        JP P,VINT       ; bei >= 0
        CALL NEG
        CALL VINT
        JP VNEG
VINT:   57F8            ; ***************************************
        RST 30H         ; INT
        RET M
        LD  HL,FACCU+8
        LD  C,14
        JR NC,L5809     ; falls Double
        JP Z,TMERR      ; falls String
INT:    5804
        LD  HL,FACCU+3
        LD  C,6
L5809:
        LD  A,(FACCU)
        OR A
        JP M,L5829
        AND 07FH
        SUB 041H
        JP C,ZERO
        INC A
        SUB C
        RET NC
        CPL
        INC A
        LD B,A
L581D:
        DEC HL
        LD A,(HL)
        AND 0F0H
        LD (HL),A
        DEC B
        RET Z
        XOR A
        LD (HL),A
        DJNZ L581D
        RET
L5829:                  ; falls negativ
        AND 07FH
        SUB 041H
        JR NC,L5835
        LD  HL,0FFFFH
        JP MAKINT
L5835:
        INC A
        SUB C
        RET NC
        CPL
        INC A
        LD B,A
        LD  E,00H
```



```
L583D:
        DEC HL
        LD A,(HL)
        LD D,A
        AND 0F0H
        LD (HL),A
        CP D
        JR Z,L5847
        INC E
L5847:
        DEC B
        JR Z,L5852
        XOR A
        LD (HL),A
        CP D
        JR Z,L5850
        INC E
L5850:
        DJNZ L583D
L5852:
        INC E
        DEC E
        RET Z
        LD A,C
        CP 06H
        LD  BC,010C1H
        LD  DE,00H
        JP Z,FADD
        EX DE,HL
        LD (ARG+6),HL
        LD (ARG+4),HL
        LD (ARG+2),HL
        LD H,B
        LD L,C
        LD (ARG),HL
        JP DADO/DECADD
UMULT:  5873            ; ************** INTEGER Multiplikation
        PUSH HL         ;                 ohne Vorzeichen
        LD  HL,00H      ; DE:=DE*BC
        LD A,B
        OR C
        JR Z,L588D      ; falls BC=0
        LD  A,16        ; Stellenzahl
L587D:
        ADD HL,HL       ; HL*2
        JP C,BSERR
        EX DE,HL
        ADD HL,HL
        EX DE,HL        ; DE*2
        JR NC,L588A
        ADD HL,BC
        JP C,BSERR
L588A:
        DEC A
        JR NZ,L587D
L588D:
        EX DE,HL
        POP HL
        RET
ISUB:   5890            ; ************* INTEGER Subtraktion
        LD A,H          ; HL:=HL-DE
```

```
        RLA
        SBC A,A
        LD B,A
        CALL INEGHL
        LD A,C
        SBC A,B
        JR L589E
IADD:   /5898           ; ************* INTEGER Addition
        LD A,H          ; HL:=HL+DE
        RLA             ; Ergebnis kann Floatingpoint-Zahl sein
        SBC A,A
L589E:
        LD B,A          ; 0 Vorzeichen HL war positiv sonst -1
        PUSH HL
        LD A,D
        RLA
        SBC A,A
        ADD HL,DE
        ADC A,B
        RRCA
        XOR H
        JP P,L56DA
        PUSH BC
        EX DE,HL
        CALL CONSIH
        POP AF
        POP HL
        CALL PUSHF
        CALL CONSIH
        POP BC
        POP DE
        JP FADD
IMULT:  /5898           ; ************** INTEGER Multiplikation
        LD A,H          ; HL:=HL*DE
        OR L            ; Ergebnis kann einfache Genauigkeit haben
        JP Z,MAKINT
        PUSH HL
        PUSH DE
        CALL L5947
        PUSH BC
        LD B,H
        LD C,L
        LD  HL,00H
        LD  A,010H
L58CE:
        ADD HL,HL
        JR C,L58F0
        EX DE,HL
        ADD HL,HL
        EX DE,HL
        JR NC,L58D9
        ADD HL,BC
        JR C,L58F0
L58D9:
        DEC A
        JR NZ,L58CE
        POP BC
        POP DE
L58DE:
        LD A,H
        OR A
```



```
        JP M,L58E8
        POP DE
        LD A,B
        JP L594F
L58E8:
        XOR 080H
        OR L
        JR Z,L5901
        EX DE,HL
        JR L58F2
L58F0:
        POP BC
        POP HL
L58F2:
        CALL CONSIH
        POP HL
        CALL PUSHF
        CALL CONSIH
        POP BC
        POP DE
        JP FMULT
L5901:
        LD A,B
        OR A
        POP BC
        JP M,MAKINT
        PUSH DE
        CALL CONSIH
        POP DE
        JP NEG
IDIV:   /590F           ; ***************** INTEGER Division
        LD A,H          ; FACCU:=DE/HL
        OR L
        JP Z,DVOERR
        CALL L5947
        PUSH BC
        EX DE,HL
        CALL INEGHL
        LD B,H
        LD C,L
        LD  HL,00H
        LD  A,011H
        PUSH AF
        OR A
        JR L5931
L5927:
        PUSH AF
        PUSH HL
        ADD HL,BC
        JR NC,L5930
        POP AF
        SCF
        JR L5931
L5930:
        POP HL
L5931:
        LD A,E
        RLA
        LD E,A
        LD A,D
        RLA
```

```
        LD D,A
        LD A,L
        RLA
        LD L,A
        LD A,H
        RLA
        LD H,A
        POP AF
        DEC A
        JR NZ,L5927
        EX DE,HL
        POP BC
        PUSH DE
        JP L58DE
L5947:
        LD A,H
        XOR D
        LD B,A
        CALL L594E
        EX DE,HL
L594E:
        LD A,H
L594F:
        OR A
        JP P,MAKINT
INEGHL: 5953            ; ****************** INTEGER negation
        XOR A           ; HL:=-HL
        LD C,A
        SUB L
        LD L,A
        LD A,C
        SBC A,H
        LD H,A
        JP MAKINT
INEG:   5950            ; *********** Integer negation des FACCU
        LD HL,(FACLOW)
        CALL INEGHL
        LD A,H
        XOR 080H
        OR L
        RET NZ
INEG2:  5968
        XOR A
        JP L56F7
IMOD:   596C            ; ***************** INTEGER MOD
        PUSH DE         ; ergebnis in FACCU
        CALL IDIV
        XOR A
        ADD A,D
        RRA
        LD H,A
        LD A,E
        RRA
        LD L,A
        CALL VALINT
        POP AF
        JR L594F

        ; einfach genaue Operationen
FADDS:  5970            ; ************** ADDITION
        CALL MOVRM      ; FACCU:=FACCU+HL
```



```
FADD:   5980            ; ************** Addition
        CALL CONASD     ; FACCU:=FACCU+R
        CALL CONDS 5926
        JP DADO/DECADD
FSUB:   5989            ; ************** Subtraktion
        CALL NEG        ; FACCU:=R-FACCU
        JR FADD
FMULT:  598E            ; ************** Multiplikation
        CALL CONASD     ; FACCU:=FACCU*R
        CALL CONDS
        JP DECMUL/DMULT
FDIVT:  5997            ; ************** Division
        POP BC          ; FACCU:=FACCU/TOS
        POP DE
FDIV:   5999            ; ************** Division
        LD HL,(FACLOW)  ; FACCU:=FACCU/R
        EX DE,HL
        LD (FACLOW),HL
        PUSH BC
        LD HL,(FACCU)
        EX (SP),HL
        LD (FACCU),HL
        POP BC
        CALL CONASD
        CALL CONDS
        JP DDIV/DECDIV
CONASD: 59B2            ; ***** convert single R --> double ARG
        EX DE,HL
        LD (ARG+2),HL
        LD H,B
        LD L,C
        LD (ARG),HL
        LD  HL,00H
        LD (ARG+4),HL
        LD (ARG+6),HL
        RET
DCRART: 59C5
        DEC A
        RET
DCXHRT: 59C7
        DEC HL
        RET
POPHRT: 59C9
        POP HL
        RET
FINDBL: 59CB            ; ******************
        EX DE,HL
        LD  BC,OFFH     ; C=0 single   C=OFFH Integer
        LD H,B
        LD L,B
        CALL MAKINT     ; FACCU:=0 INTEGER
        EX DE,HL
        LD A,(HL)
        CP '&'
        JP Z,OCTCNS
        CP '-'
        PUSH AF         ; Vorzeichen speichern
        JR Z,L59E5
        CP '+'
        JR Z,L59E5
        DEC HL
```

```
L59E5:                  ; hier gehts nach Ziffer weiter
        RST 10H
        JP C,L5AB8      ; wenn Ziffer
        CP '.'
        JP Z,L5A81
        CP 'e'
        JR Z,L59F4
        CP 'E'
L59F4:
        JR NZ,L5A10
        PUSH HL         ; Exponent
        RST 10H
        CP 'l'          ; el
        JR Z,L5A06
        CP 'L'
        JR Z,L5A06
        CP 'q'          ; eq
        JR Z,L5A06
        CP 'Q'
L5A06:
        POP HL
        JR Z,L5A0F      ; Z bei l oder q
        RST 30H
        JR NC,L5A27     ; NC falls double
        XOR A
        JR L5A28
L5A0F:
        LD A,(HL)
L5A10:                  ; kein Exponent, Ziffer oder
        CP '%'
        JP Z,L5A94      ; force integer Ende
        CP '#'
        JP Z,L5AA2      ; force double Ende
        CP '!'
        JP Z,L5AA3      ; force single Ende
        CP 'd'
        JR Z,L5A27
        CP 'D'
        JR NZ,L5A50
L5A27:
        OR A
L5A28:                  ; force Z single   NZ double
        CALL L5AA9
        RST 10H
        PUSH DE
        LD D,00H
        CALL MINPLS
        LD C,D
        POP DE
L5A34:
        RST 10H
        JR NC,L5A4A
        LD A,E
        CP 0CH
        JR NC,L5A46     ; NC falls > 11
        RLCA
        RLCA
        ADD A,E
        RLCA            ; A*10
        ADD A,(HL)
        SUB '0'
```



```
        LD E,A          ; +(HL)-'0'
        JR L5A34
L5A46:
        LD  E,080H
        JR L5A34
L5A4A:
        INC C
        JR NZ,L5A50
        XOR A
        SUB E
        LD E,A
L5A50:
        RST 30H
        JP M,L5A66
        LD  A,(FACCU)
        OR A
        JR Z,L5A66
        LD A,D
        SUB B
        ADD A,E
        ADD A,040H
        LD (FACCU),A
        OR A
        CALL M,L5A7E
L5A66:
        POP AF
        PUSH HL
        CALL Z,VNEG
        RST 30H
        JR NC,L5A79
        POP HL
        RET PE
        PUSH HL
        LD  HL,POPHRT
        PUSH HL
        CALL CONIS2
        RET
L5A79:
        CALL DECROU
        POP HL
        RET
L5A7E:
        JP DVERR
L5A81:                  ; Dezimalpunkt gefunden!
        RST 30H
        INC C
        JR NZ,L5A50
        JR NC,L5A91
        CALL L5AA9
        LD  A,(FACCU)
        OR A
        JR NZ,L5A91
        LD D,A
L5A91:
        JP L59E5
L5A94:                  ; bei % force integer
        RST 10H
        POP AF
        PUSH HL
        LD  HL,POPHRT
        PUSH HL
```

```
        LD  HL,FRCINT
        PUSH HL
        PUSH AF
        JR L5A50
L5AA2:                  ; force double
        OR A
L5AA3:                  ; force single
        CALL L5AA9
        RST 10H
        JR L5A50
L5AA9:                  ; force floatingpoint
        PUSH HL
        PUSH DE
        PUSH BC
        PUSH AF
        CALL Z,FRCSNG
        POP AF
        CALL NZ,FRCDBL
        POP BC
        POP DE
        POP HL
        RET
L5AB8:                  ; Ziffer
        SUB '0'
        JP NZ,L5AC5     ; Stelle >0
        OR C
        JP Z,L5AC5
        AND D
        JP Z,L59E5
L5AC5:
        INC D
        LD A,D
        CP 07H
        JR NZ,L5ACF     ; NZ noch nicht 7. Stelle
        OR A
        CALL L5AA9      ; force single bei 7. Stelle
L5ACF:
        PUSH DE
        LD A,B
        ADD A,C
        INC A
        LD B,A
        PUSH BC
        PUSH HL
        LD A,(HL)
        SUB '0'
        PUSH AF
        RST 30H
        JP P,L5B03      ; P wenn kein Integer
        LD HL,(FACLOW)
        LD  DE,0CCDH    ; 3277  kleinste Zahl
        RST 20H         ; die bei Multiplikation mit
        JR NC,L5B00     ; 10 Ueberlauf ergibt
        LD D,H
        LD E,L
        ADD HL,HL
        ADD HL,HL
        ADD HL,DE
        ADD HL,HL       ; hl:=HL*10
        POP AF
        LD C,A
```



```
        ADD HL,BC       ; +A
        LD A,H
        OR A
        JP M,L5AFE      ; Ueberlauf durch addition
        LD (FACLOW),HL
L5AF8:
        POP HL
        POP BC
        POP DE
        JP L59E5        ; naechste Stelle
L5AFE:
        LD A,C
        PUSH AF
L5B00:
        CALL CONSI
L5B03:
        POP AF
        POP HL
        POP BC
        POP DE
        JR NZ,L5B15
        LD  A,(FACCU)
        OR A
        LD  A,00H
        JR NZ,L5B15
        LD D,A
        JP L59E5
L5B15:
        PUSH DE
        PUSH BC
        PUSH HL
        PUSH AF
        LD  HL,FACCU
        LD (HL),01H
        LD A,D
        CP 010H
        JR C,L5B26
        POP AF
        JR L5AF8
L5B26:
        INC A
        OR A
        RRA
        LD B,00H
        LD C,A
        ADD HL,BC
        POP AF
        LD C,A
        LD A,D
        RRA
        LD A,C
        JR NC,L5B38
        ADD A,A
        ADD A,A
        ADD A,A
        ADD A,A
L5B38:
        OR (HL)
        LD (HL),A
        JR L5AF8
```

```
INPRT:/ 5B3C        ; ******************** Textausgabe
      PUSH HL
      LD  HL,INTXT  ; ' in '
      CALL STROUT
      POP HL
LINPRT:/ 5B44       ; **********
      LD  BC,STROUI
      PUSH BC
LINOUT:/ 5B48       ; ******** Ausgabe von HL als Dezimalzahl
      CALL MAKINT
      XOR A
      LD (TEMP3),A
      LD  HL,0F8F3H
      LD (HL),' '
      OR (HL)
      JR L5B73      ; als vorzeichenlose Zahl
FOUT: / 5B5A        ; ******************
      XOR A         ; clear edit flags
PUFOUT:/ 5B5B       ; ******************** BINARY --> ASCII
      CALL L5E8D    ; HL Ausgabe-Puffer
      AND 08H
      JR Z,L5B61    ; NZ wenn Vorzeichen verlangt
      LD (HL),'+'

L5B61:
      EX DE,HL
      CALL VSIGN    ; Vorzeichen feststellen
      EX DE,HL
      JP P,L5B73    ; P nicht negativ
      LD (HL),'-'
      PUSH BC
      PUSH HL
      CALL VNEG     ; invertieren
      POP HL
      POP BC
      OR H

L5B73:
      INC HL
      LD (HL),'0'
      LD  A,(TEMP3)
      LD D,A
      RLA
      LD  A,(VALTYP)
      JP C,L5C29    ; C print using call
      JP Z,L5C21    ; Ende falls edit flag=0
      CP 04H
      JP NC,L5BD3   ; NC= Single oder Double
      LD  BC,00H    ; keine , oder .
      CALL L5E09    ; integer --> ASCII

L5B8F:
      LD  HL,0F8F3H ; Start ASCII-Puffer
      LD B,(HL)
      LD  C,' '
      LD  A,(TEMP3)
      LD E,A
      AND 020H
      JR Z,L5BA9    ; Z falls kein Bereichstest gewuenscht
      LD A,B        ; 1. Zeichen = ' '
      CP C
      LD  C,'*'     ; wenn nein ersetzen
      JR NZ,L5BA9
      LD A,E
```



```
      AND 04H
      JR NZ,L5BA9
      LD B,C
L5BA9:
      LD (HL),C
      RST 10H
      JR Z,L5BC1    ; Z falls Pufferende
      CP 'E'
      JR Z,L5BC1
      CP 'D'
      JR Z,L5BC1
      CP '0'
      JR Z,L5BA9
      CP ','
      JR Z,L5BA9
      CP '.'
      JR NZ,L5BC4
L5BC1:
      DEC HL
      LD (HL),'0'
L5BC4:
      LD A,E
      AND 010H
      JR Z,L5BCC
      DEC HL
      LD (HL),'$'
L5BCC:
      LD A,E
      AND 04H
      RET NZ
      DEC HL
      LD (HL),B
      RET
L5BD3:
      PUSH HL
      CALL NUMLEN
      LD D,B
      INC D
      LD  BC,0300H
      LD  A,(FACCU)
      SUB 03FH
      JR C,L5BEB
      INC D
      CP D
      JR NC,L5BEB
      INC A
      LD B,A
      LD  A,02H
L5BEB:
      SUB 02H
      POP HL
      PUSH AF
      CALL L5DBC
      LD (HL),'0'
      CALL Z,INXHRT
      CALL L5DE1
L5BFA:
      DEC HL
      LD A,(HL)
      CP '0'
      JR Z,L5BFA
```

```
        CP '.'
        CALL NZ,INXHRT
        POP AF
        JR Z,L5C22
L5C08:
        LD (HL),'E'
        INC HL
        LD (HL),'+'
        JP P,L5C14
        LD (HL),'-'
        CPL
        INC A
L5C14:
        LD B,02FH
L5C16:
        INC B
        SUB 0AH
        JR NC,L5C16
        ADD A,03AH
        INC HL
        LD (HL),B
        INC HL
        LD (HL),A
L5C21:
        INC HL
L5C22:
        LD (HL),00H
        EX DE,HL
        LD  HL,0F8F3H
        RET
L5C29:
        INC HL
        PUSH BC
        CP 04H
        LD A,D
        JP NC,L5C98
        RRA
        JP C,L5D1F
        LD  BC,0603H
        CALL L5DB4
        POP DE
        LD A,D
        SUB 05H
        CALL P,L5D94
        CALL L5E09
L5C45:
        LD A,E
        OR A
        CALL Z,DCXHRT
        DEC A
        CALL P,L5D94
L5C4E:
        PUSH HL
        CALL L5B8F
        POP HL
        JR Z,L5C57
        LD (HL),B
        INC HL
L5C57:
        LD (HL),00H
        LD  HL,FBUFFR
```



```
L5C5C:
        INC HL
L5C5D:
        LD  A,(TEMP2)
        SUB L
        SUB D
        RET Z
        LD A,(HL)
        CP ' '
        JR Z,L5C5C
        CP '*'
        JR Z,L5C5C
        DEC HL
        PUSH HL
L5C6E:
        PUSH AF
        LD  BC,L5C6E
        PUSH BC
        RST 10H
        CP '-'
        RET Z
        CP '+'
        RET Z
        CP '$'
        RET Z
        POP BC
        CP '0'
        JR NZ,L5C91
        INC HL
        RST 10H
        JR NC,L5C91
        DEC HL
        JR L5C8B
L5C89:
        DEC HL
        LD (HL),A
L5C8B:
        POP AF
        JR Z,L5C89
        POP BC
        JR L5C5D
L5C91:
        POP AF
        JR Z,L5C91
        POP HL
        LD (HL),'%'
        RET
L5C98:
        PUSH HL
        RRA
        JP C,L5D25
        CALL NUMLEN
        LD D,B
        LD  A,(FACCU)
        SUB 04FH
        JR C,L5CB3
        POP HL
        POP BC
        CALL FOUT
        LD  HL,FBUFFR
        LD (HL),'%'
```

```
        RET
L5CB3:
        RST 28H
        CALL NZ,L5EDO
        POP HL
        POP BC
        JP M,L5CD6
        PUSH BC
        LD E,A
        LD A,B
        SUB D
        SUB E
        CALL P,L5D94
        CALL L5DA8
        CALL L5DE1
        OR E
        CALL NZ,L5DA2
        OR E
        CALL NZ,L5DCE
        POP DE
        JP L5C45
L5CD6:
        LD E,A
        LD A,C
        OR A
        CALL NZ,DCRART
        ADD A,E
        JP M,L5CE1
        XOR A
L5CE1:
        PUSH BC
        PUSH AF
        CALL M,L5EA9
        POP BC
        LD A,E
        SUB B
        POP BC
        LD E,A
        ADD A,D
        LD A,B
        JP M,L5CFB
        SUB D
        SUB E
        CALL P,L5D94
        PUSH BC
        CALL L5DA8
        JR L5DOC
L5CFB:
        CALL L5D94
        LD A,C
        CALL L5DD1
        LD C,A
        XOR A
        SUB D
        SUB E
        CALL L5D94
        PUSH BC
        LD B,A
        LD C,A
L5DOC:
        CALL L5DE1
```



```
        POP BC
        OR C
        JR NZ,L5D16
        LD HL,(TEMP2)
L5D16:
        ADD A,E
        DEC A
        CALL P,L5D94
        LD D,B
        JP L5C4E
L5D1F:
        PUSH HL
        PUSH DE
        CALL CONSI
        POP DE
L5D25:
        CALL NUMLEN
        LD E,B
        RST 28H
        PUSH AF
        CALL NZ,L5EDO
        POP AF
        POP HL
        POP BC
        PUSH AF
        LD A,C
        OR A
        PUSH AF
        CALL NZ,DCRART
        ADD A,B
        LD C,A
        LD A,D
        AND 04H
        CP 01H
        SBC A,A
        LD D,A
        ADD A,C
        LD C,A
        SUB E
        PUSH AF
        JP P,L5D56
        CALL L5EA9
        JR NZ,L5D56
        PUSH HL
        CALL DECSRD
        LD  HL,FACCU
        INC (HL)
        POP HL
L5D56:
        POP AF
        PUSH BC
        PUSH AF
        JP M,L5D5D
        XOR A
L5D5D:
        CPL
        INC A
        ADD A,B
        INC A
        ADD A,D
        LD B,A
```

```
        LD  C,00H
        CALL Z,L5DBC
        CALL L5DE1
        POP AF
        CALL P,L5D9C
        CALL L5DCE
        POP BC
        POP AF
        JR NZ,L5D82
        CALL DCXHRT
        LD A,(HL)
        CP 02EH
        CALL NZ,INXHRT
        LD (TEMP2),HL
L5D82:
        POP AF
        LD  A,(FACCU)
        JR Z,L5D8B
        ADD A,E
        SUB B
        SUB D
L5D8B:
        PUSH BC
        CALL L5C08
        EX DE,HL
        POP DE
        JP L5C4E
L5D94:
        OR A
L5D95:
        RET Z
        DEC A
        LD (HL),'0'
        INC HL
        JR L5D95
L5D9C:
        JR NZ,L5DA2
L5D9E:
        RET Z
        CALL L5DCE
L5DA2:
        LD (HL),'0'
        INC HL
        DEC A
        JR L5D9E
L5DA8:
        LD A,E
        ADD A,D
        INC A
        LD B,A
        INC A
L5DAD:
        SUB 03H
        JR NC,L5DAD
        ADD A,05H
        LD C,A
L5DB4:
        LD  A,(TEMP3)
        AND 040H
        RET NZ
        LD C,A
```



```
        RET
L5DBC:
        DEC B
        JP P,L5DCF
        LD (TEMP2),HL
        LD (HL),02EH
L5DC5:
        INC HL
        LD (HL),'0'
        INC B
        LD C,B
        JR NZ,L5DC5
        INC HL
        RET
L5DCE:
        DEC B
L5DCF:
        JR NZ,L5DD9
L5DD1:
        LD (HL),'.'
        LD (TEMP2),HL
        INC HL
        LD C,B
        RET
L5DD9:
        DEC C
        RET NZ
        LD (HL),','
        INC HL
        LD  C,03H
        RET
L5DE1:
        PUSH DE
        PUSH HL
        PUSH BC
        CALL NUMLEN
        LD A,B
        POP BC
        POP HL
        LD  DE,0F924H
        SCF
L5DEE:
        PUSH AF
        CALL L5DCE
        LD  A,(DE)
        JR NC,L5DFB
        RRA
        RRA
        RRA
        RRA
        JR L5DFC
L5DFB:
        INC DE
L5DFC:
        AND 0FH
        ADD A,'0'
        LD (HL),A
        INC HL
        POP AF
        DEC A
        CCF
```

```
        JR NZ,L5DEE
        JR L5E38
L5E09:
        PUSH DE
        LD  DE,L5E3E
        LD  A,05H
L5E0F:
        CALL L5DCE
        PUSH BC
        PUSH AF
        PUSH HL
        EX DE,HL
        LD C,(HL)
        INC HL
        LD B,(HL)
        PUSH BC
        INC HL
        EX (SP),HL
        EX DE,HL
        LD HL,(FACLOW)
        LD B,02FH
L5E22:
        INC B
        LD A,L
        SUB E
        LD L,A
        LD A,H
        SBC A,D
        LD H,A
        JR NC,L5E22
        ADD HL,DE
        LD (FACLOW),HL
        POP DE
        POP HL
        LD (HL),B
        INC HL
        POP AF
        POP BC
        DEC A
        JR NZ,L5E0F
L5E38:
        CALL L5DCE
        LD (HL),A
        POP DE
        RET

L5E3E:
        DEFB 10,27,0E8H,3,64,0,0A,0,1,0

FOUTB:  LD B,1          ; *************** Ausgabe als Binaerzahl
        JR L5E52
FOUTD:  LD B,03H        ; *************** Ausgabe als Oktalzahl
        JR L5E52
FOUTH:  LD B,04H        ; ********** Ausgabe als Hexadezimalzahl
L5E52:
        PUSH BC
        CALL FRQINT
        LD  DE,0F903H
```



```
        XOR A
        LD (DE),A
        POP BC
        LD C,A
L5E5D:
        PUSH BC
        DEC DE
L5E5F:
        AND A
        LD A,H
        RRA
        LD H,A
        LD A,L
        RRA
        LD L,A
        LD A,C
        RRA
        LD C,A
        DJNZ L5E5F
        POP BC
        PUSH BC
L5E6D:
        RLCA
        DJNZ L5E6D
        ADD A,'0'
        CP 03AH
        JR C,L5E78
        ADD A,07H
L5E78:
        LD (DE),A
        POP BC
        LD A,L
        OR H
        JR NZ,L5E5D
        EX DE,HL
        RET
NUMLEN: RST 30H         ; ******.gibt fuer Floatingpoint-Zahlen
        LD  HL,FACCU+7  ; die Stellenzahl in B und die Adresse
        LD B,14         ; des letzten benutzten Bytes des FACCU
        RET NC          ; in HL
        LD  HL,FACCU+3
        LD B,6
        RET
L5E8D:                  ; ********************************
        LD (TEMP3),A    ; speichern Editflags
        PUSH AF
        PUSH BC
        PUSH DE
        CALL FRCDBL     ; doppelt genaue Zahl erzeugen
        LD  HL,DBLZER
        LD  A,(FACCU)
        AND A
        CALL Z,MFM      ; falls Exponent-Stelle 0 Zahl 0 setzen
        POP DE
        POP BC
        POP AF
        LD  HL,0F8F3H
        LD (HL),' '
        RET
L5EA9:                  ; ************************
```

```
        PUSH HL
        PUSH DE
        PUSH BC
        PUSH AF
        CPL
        INC A
        LD E,A
        LD  A,01H
        JP Z,L5ECA
        CALL NUMLEN
        PUSH HL
L5EB9:
        CALL DECSRD
        DEC E
        JR NZ,L5EB9
        POP HL
        INC HL
        LD A,B
        RRCA
        LD B,A
        CALL DECROB
        CALL L5EE2
L5ECA:
        POP BC
        ADD A,B
        POP BC
        POP DE
        POP HL
        RET
L5EDO:                  ; *****************
        PUSH BC
        PUSH HL
        CALL NUMLEN
        LD  A,(FACCU)
        SUB 04OH
        SUB B
        LD (FACCU),A
        POP HL
        POP BC
        OR A
        RET
L5EE2:                  ; *****************
        PUSH BC
        CALL NUMLEN
L5EE6:
        LD A,(HL)
        AND OFH
        JR NZ,L5EF3
        DEC B
        LD A,(HL)
        OR A
        JR NZ,L5EF3
        DEC HL
        DJNZ L5EE6
L5EF3:
        LD A,B
        POP BC
        RET
SNGEXP: 5EF6            ; ****** einfach genaue Exponierung
        CALL CONASD     ; FACCU:=FACCU^R
        CALL CONDS
```



```
        CALL PHA
        CALL XTF
        CALL PPA
DBLEXP: 5F0S            ; ******* doppelt genaue Exponierung
        LD  A,(ARG)     ; FACCU:=FACCU^ARG
        OR A
        JP Z,L5F71
        LD H,A
        LD  A,(FACCU)
        OR A
        JP Z,L5F7B
        CALL PHF
        CALL L6048
        JR C,L5F58
        EX DE,HL
        LD (TEMP8),HL
        CALL VALDBL
        CALL PPA
        CALL L6048
        CALL VALDBL
        LD HL,(TEMP8)
        JP NC,L5F88
        LD  A,(ARG)
        PUSH AF
        PUSH HL
        CALL MFA
        LD  HL,FBUFFR
        CALL MMF
        LD  HL,ONE
        CALL MFM
        POP HL
        LD A,H
        OR A
        PUSH AF
        JP P,L5F54
        XOR A
        LD C,A
        SUB L
        LD L,A
        LD A,C
        SBC A,H
        LD H,A
L5F54:
        PUSH HL
        JP L5FC2
L5F58:
        CALL VALDBL
        CALL MFA
        CALL XTF
        CALL LOG
        CALL PPA
        CALL DECMUL/DMULT
        JP EXP
INTEXP: 5F60            ; ************ Integer exponentiation
        LD A,H          ; HL:=DE^HL
        OR L
        JR NZ,L5F77
L5F71:
        LD  HL,01H
        JP L5F85
L5F77:
```

```
        LD A,D
        OR E
        JR NZ,L5F88
L5F7B:
        LD A,H
        RLA
        JR NC,L5F82
        JP DVOERR
L5F82:
        LD  HL,00H
L5F85:
        JP MAKINT
L5F88:
        LD (TEMP8),HL
        PUSH DE
        LD A,H
        OR A
        PUSH AF
        CALL M,INEGHL
        LD B,H
        LD C,L
        LD  HL,01H
L5F97:
        OR A
        LD A,B
        RRA
        LD B,A
        LD A,C
        RRA
        LD C,A
        JR NC,L5FA5
        CALL L603B
        JR NZ,L5FF1
L5FA5:
        LD A,B
        OR C
        JR Z,L600C
        PUSH HL
        LD H,D
        LD L,E
        CALL L603B
        EX DE,HL
        POP HL
        JR Z,L5F97
        PUSH BC
        PUSH HL
        LD  HL,FBUFFR
        CALL MMF
        POP HL
        CALL CONSIH
        CALL CONDS
L5FC2:
        POP BC
        LD A,B
        OR A
        RRA
        LD B,A
        LD A,C
        RRA
        LD C,A
        JR NC,L5FD4
```



```
        PUSH BC
        LD  HL,FBUFFR
        CALL DMULTO
        POP BC
L5FD4:
        LD A,B
        OR C
        JR Z,L600C
        PUSH BC
        CALL PHF
        LD  HL,FBUFFR
        PUSH HL
        CALL MFM
        POP HL
        PUSH HL
        CALL DMULTO
        POP HL
        CALL MMF
        CALL PPF
        JR L5FC2
L5FF1:
        PUSH BC
        PUSH DE
        CALL FRCDBL
        CALL MAF
        POP HL
        CALL CONSIH
        CALL CONDS
        LD  HL,FBUFFR
        CALL MMF
```

```
        LD  HL,FBUFFR
        CALL MMF
        CALL MFA
        POP BC
        JR L5FD4
L600C:
        POP AF
        POP BC
        RET P
        LD  A,(VALTYP)
        CP 02H
        JR NZ,L601E
        PUSH BC
        CALL CONSIH
        CALL CONDS
        POP BC
L601E:
        LD  A,(FACCU)
        OR A
        JR NZ,L602F
        LD HL,(TEMP8)
        OR H
        RET P
        LD A,L
        RRCA
        AND B
        JP DVERR
L602F:
        CALL MAF
        LD  HL,ONE
        CALL MFM
        JP DDIV/DECDIV
L603B:
        PUSH BC
        PUSH DE
        CALL IMULT
        LD  A,(VALTYP)
        CP 02H
        POP DE
        POP BC
        RET
L6048:
        CALL MFA
        CALL PHA
        CALL VINT
        CALL PPA
        CALL XDCOMP
        SCF
        RET NZ
        JP QINTA
L605C:
        DEC HL
        RST 10H
        RET Z
        RST 8H
        DEFB ','
DIM:    6061            ; ********************* DIM
        LD  BC,L605C
        PUSH BC
        DEFB OF6H       ;*      OR 0AFH    Erzeugen
PTRGET: 6066
```



```
        XOR A           ;*                 Finden
        LD (DIMFLG),A   ; der Variablen
        LD C,(HL)
PTRGT2: 6068
        CALL LFECD
        CALL ISLET
        JP C,SNERR      ; jp falls kein Buchstabe
        XOR A
        LD B,A
        RST 10H
        JR C,L607E
        CALL ISLET2
        JR C,L6087
L607E:
        LD B,A
L607F:
        RST 10H
        JR C,L607F
        CALL ISLET2
        JR NC,L607F
L6087:
        CP '&'
        JR NC,L60A2
        LD  DE,L60B0    ; return Adresse
        PUSH DE
        LD D,02H
        CP '%'          ; Test auf folgendes %
        RET Z
        INC D
        CP '$'          ;                   $
        RET Z
        INC D
        CP '!'          ;                   !
        RET Z
        LD D,08H
        CP '#'          ;                   #
        RET Z
        POP AF
L60A2:
        LD A,C          ; erster Buchstabe des Namens
        AND 07FH
        LD E,A
        LD D,00H
        PUSH HL
        LD  HL,DEFTBL-41H       ; DEFTYP Tabelle
        ADD HL,DE
        LD D,(HL)       ; definierten Typ holen
        POP HL
        DEC HL
L60B0:                  ; D Laenge des Datentyps
        LD A,D
        LD (VALTYP),A
        RST 10H
        LD  A,(SUBFLG)
        DEC A
        JP Z,ERSFIN
        JP P,NOARYS
        LD A,(HL)
        SUB 30H
        JP Z,L617C
        SUB 033H
```

```
          JP Z,L617C
NOARYS:   /60CA
          XOR A
          LD (SUBFLG),A
          PUSH HL
          LD  A,(NOFUNS)
          OR A
          LD (PARMFLG),A
          JR Z,L6114
          LD HL,(PRMLEN)
          LD  DE,PARM1
          ADD HL,DE
          LD (ARYTA2),HL
          EX DE,HL
          JR L60FC
L60E5:
          LD  A,(DE)
          LD L,A
          INC DE
          LD  A,(DE)
          INC DE
          CP C
          JR NZ,L60F8
          LD  A,(VALTYP)
          CP L
          JR NZ,L60F8
          LD  A,(DE)
          CP B
          JP Z,L6166
L60F8:
          INC DE
          LD H,00H
          ADD HL,DE
L60FC:
          EX DE,HL
          LD  A,(ARYTA2)
          CP E
          JP NZ,L60E5
          LD  A,(0F8E2H)
          CP D
          JR NZ,L60E5
          LD  A,(PARMFLG)
          OR A
          JR Z,L6128
          XOR A
          LD (PARMFLG),A
L6114:
          LD HL,(ARYTAB)
          LD (ARYTA2),HL
          LD HL,(VARTAB)
          JR L60FC
PTRGTN:   /6117
          CALL PTRGET
PTRGTR:   /6122
          RET
L6123:
          LD D,A
          LD E,A
          POP BC
          EX (SP),HL
          RET
```



```
L6128:
          POP HL
          EX (SP),HL
          PUSH DE
          LD  DE,PTRGTR
          RST 20H
          JR Z,L6123
          LD  DE,RETVAR
          RST 20H
          POP DE
          JR Z,L6169
          EX (SP),HL
          PUSH HL
          PUSH BC
          LD  A,(VALTYP)
          LD C,A
          PUSH BC
          LD B,00H
          INC BC
          INC BC
          INC BC
          LD HL,(STREND)
          PUSH HL
          ADD HL,BC
          POP BC
          PUSH HL
          CALL BLTU
          POP HL
          LD (STREND),HL
          LD H,B
          LD L,C
          LD (ARYTAB),HL
L6158:
          DEC HL
          LD (HL),00H
          RST 20H
          JR NZ,L6158
          POP DE
          LD (HL),E
          INC HL
          POP DE
          LD (HL),E
          INC HL
          LD (HL),D
          EX DE,HL
L6166:
          INC DE
          POP HL
          RET
L6169:
          LD (FACCU),A
          LD H,A
          LD L,A
          LD (FACLOW),HL
          RST 30H
          JR NZ,L617A
          LD  HL,089EH
          LD (FACLOW),HL
L617A:
          POP HL
          RET
```

```
L617C:
        PUSH HL
        LD HL,(DIMFLG)
        EX (SP),HL
        LD D,A
L6182:
        PUSH DE
        PUSH BC
        CALL INTIDX
        POP BC
        POP AF
        EX DE,HL
        EX (SP),HL
        PUSH HL
        EX DE,HL
        INC A
        LD D,A
        LD A,(HL)
        CP ','
        JP Z,L6182
        CP '
         P ']'
        JP NZ,SNERR
L619E:
        RST 10H
        LD (TEMP2),HL
        POP HL
        LD (DIMFLG),HL
        LD  E,00H
        PUSH DE
        DEFB 011H       ;*      LD DE,0F5E5H
ERSFIN: 61AA
        PUSH HL         ;*
        PUSH AF         ;*
        LD HL,(ARYTAB)
        DEFB 03EH       ;*      LD A,019H
L61B0:
        ADD HL,DE       ;*
        LD DE,(STREND)
        RST 20H
        JR Z,L61E5
        LD E,(HL)
        INC HL
        LD A,(HL)
        INC HL
        CP C
        JR NZ,L61C7
        LD  A,(VALTYP)
        CP E
        JR NZ,L61C7
        LD A,(HL)
        CP B
L61C7:
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        INC HL
        JR NZ,L61B0
        LD  A,(DIMFLG)
```



```
        OR A
        JP NZ,DDERR
        POP AF
        LD B,H
        LD C,L
        JP Z,POPHRT
        SUB (HL)
        JP Z,L623F
BSERR:  61DF
        LD  DE,09H
        JP ERROR
L61E5:
        LD  A,(VALTYP)
        LD (HL),A
        INC HL
        LD E,A
        LD D,00H
        POP AF
        JP Z,FCERR
        LD (HL),C
        INC HL
        LD (HL),B
        INC HL
        LD C,A
        CALL GETSTK
        INC HL
        INC HL
        LD (TEMP3),HL
        LD (HL),C
        INC HL
        LD  A,(DIMFLG)
        RLA
        LD A,C
L6205:
        LD  BC,0BH
        JR NC,L620C
        POP BC
        INC BC
L620C:
        LD (HL),C
        PUSH AF
        INC HL
        LD (HL),B
        INC HL
        CALL UMULT
        POP AF
        DEC A
        JR NZ,L6205
        PUSH AF
        LD B,D
        LD C,E
        EX DE,HL
        ADD HL,DE
        JP C,OMERR
        CALL REASON
        LD (STREND),HL
L6226:
        DEC HL
        LD (HL),00H
        RST 20H
        JR NZ,L6226
```

```
        INC BC
        LD D,A
        LD HL,(TEMP3)
        LD E,(HL)
        EX DE,HL
        ADD HL,HL
        ADD HL,BC
        EX DE,HL
        DEC HL
        DEC HL
        LD (HL),E
        INC HL
        LD (HL),D
        INC HL
        POP AF
        JR C,L626F
L623F:
        LD B,A
        LD C,A
        LD A,(HL)
        INC HL
        DEFB 016H       ;*      LD D,0E1H
L6244:
        POP HL          ;*
        LD E,(HL)
        INC HL
        LD D,(HL)
        INC HL
        EX (SP),HL
        PUSH AF
        RST 20H
        JP NC,BSERR
        CALL UMULT
        ADD HL,DE
        POP AF
        DEC A
        LD B,H
        LD C,L
        JR NZ,L6244
        LD  A,(VALTYP)
        LD B,H
        LD C,L
        ADD HL,HL
        SUB 04H
        JR C,L6267
        ADD HL,HL
        JR Z,L626C
        ADD HL,HL
L6267:
        OR A
        JP PO,L626C
        ADD HL,BC
L626C:
        POP BC
        ADD HL,BC
        EX DE,HL
L626F:
        LD HL,(TEMP2)
        RET
PRINUS: /6273           ; ************** PRINT USING
        CALL FRMCHK
```



```
        CALL CHKSTR/FRCSTR
        RST 8H
        DEFB ';'
        EX DE,HL
        LD HL,(FACLOW)
        JR L6289
L6281:
        LD  A,(USFLG)
        OR A
        JR Z,L6294
        POP DE
        EX DE,HL
L6289:
        PUSH HL
        XOR A
        LD (USFLG),A
        INC A
        PUSH AF
        PUSH DE
        LD B,(HL)
        INC B
        DEC B
L6294:
        JP Z,FCERR      ; falls Leerstring
        INC HL
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        JR L62B8        ; B Laenge HL Adresse des USING Strings
L629E:                  ; Stringausdruck in USING
        LD E,B          ; sichern der Laenge
        PUSH HL
        LD  C,02H       ; mindest Laenge
L62A2:
        LD A,(HL)
        INC HL
        CP '\'
        JP Z,L63C9      ; string fertig
        CP ' '
        JR NZ,L62B0     ; nur ' ' innerhalb \ \
        INC C
        DJNZ L62A2
L62B0:
        POP HL
        LD B,E          ; USING Laenge
        LD  A,'\'
L62B4:
        CALL L63FA
        RST 18H
L62B8:                  ; Start eines neuen Elements fuer USING
        XOR A
        LD E,A
        LD D,A
L62BB:
        CALL L63FA
        LD D,A
        LD A,(HL)
        INC HL
        CP '!'          ; nur erstes Zeichen des Strings ausgeben
        JP Z,L63C6
```

```
        CP '#'          ; Zahl anfang
        JR Z,L6301
        DEC B
        JP Z,L63B2      ; Stringende
        CP '+'
        LD  A,08H
        JR Z,L62BB
        DEC HL
        LD A,(HL)
        INC HL
        CP '.'
        JR Z,L631B
        CP '\'
        JR Z,L629E      ; String
        CP (HL)
        JR NZ,L62B4
        CP '$'
        JR Z,L62FA      ; Anfang $$
        CP '*'
        JR NZ,L62B4
        INC HL
        LD A,B
        CP 02H
        JR C,L62F3
        LD A,(HL)
        CP '$'
L62F3:
        LD  A,020H
        JR NZ,L62FE
        DEC B
        INC E
        DEFB OFEH       ;*      CP OAFH
L62FA:
        XOR A           ;*
        ADD A,010H
        INC HL
L62FE:
        INC E
        ADD A,D
        LD D,A
L6301:
        INC E
        LD  C,00H
        DEC B
        JR Z,L634F
        LD A,(HL)
        INC HL
        CP '.'
        JR Z,L6326
        CP '#'
        JR Z,L6301
        CP ','
        JR NZ,L6330
        LD A,D
        OR 040H
        LD D,A
        JR L6301
L631B:
        LD A,(HL)
        CP '#'
        LD  A,'.'
```



```
        JP NZ,L62B4
        LD  C,01H
        INC HL
L6326:
        INC C
        DEC B
        JR Z,L634F
        LD A,(HL)
        INC HL
        CP '#'
        JR Z,L6326
L6330:
        PUSH DE
        LD  DE,L634D
        PUSH DE
        LD D,H
        LD E,L
        CP '^'
        RET NZ
        CP (HL)
        RET NZ
        INC HL
        CP (HL)
        RET NZ
        INC HL
        CP (HL)
        RET NZ
        INC HL
        LD A,B
        SUB 04H
        RET C
        POP DE
        POP DE
        LD B,A
        INC D
        INC HL
        DEFB 0CAH       ;*      JP Z,LD1EB
L634D:
        POP DE          ;*
        EX DE,HL        ;*
L634F:
        LD A,D
        DEC HL
        INC E
        AND 08H
        JR NZ,L636B
        DEC E
        LD A,B
        OR A
        JR Z,L636B
        LD A,(HL)
        SUB '-'
        JR Z,L6366
        CP 0FEH
        JR NZ,L636B
        LD  A,08H
L6366:
        ADD A,04H
        ADD A,D
        LD D,A
        DEC B
```

```
L636B:
        POP HL
        POP AF
        JR Z,L63BB
        PUSH BC
        PUSH DE
        CALL FRMEVL
        POP DE
        POP BC
        PUSH BC
        PUSH HL
        LD B,E
        LD A,B
        ADD A,C
        CP FORSZC
        JP NC,FCERR
        LD A,D
        OR 080H
        CALL PUFOUT
        CALL STROUT
L6389:
        POP HL
        DEC HL
        RST 10H
        SCF
        JR Z,L639A
        LD (USFLG),A
        CP 03BH
        JR Z,L6399
        RST 8H
        DEFB ','
        DEFB 6          ;*      LD B,0D7H
L6399:
        RST 10H         ;*
L639A:
        POP BC
        EX DE,HL
        POP HL
        PUSH HL
        PUSH AF
        PUSH DE
        LD A,(HL)
        SUB B
        INC HL
        LD D,00H
        LD E,A
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        ADD HL,DE
        LD A,B
        OR A
        JP NZ,L62B8
        JR L63B6
L63B2:
        CALL L63FA
        RST 18H
L63B6:
        POP HL
        POP AF
```



```
        JP NZ,L6281
L63BB:
        CALL C,CRDO
        EX (SP),HL
        CALL FRETM2
        POP HL
        JP FINPRT
L63C6:
        LD  C,01H
        DEFB 03EH       ;*      LD  A,0F1H
L63C9:
        POP AF          ;*
        DEC B
        CALL L63FA
        POP HL
        POP AF
        JR Z,L63BB
        PUSH BC
        CALL FRMEVL
        CALL CHKSTR/FRCSTR
        POP BC
        PUSH BC
        PUSH HL
        LD HL,(FACLOW)
        LD B,C
        LD  C,00H
        LD A,B
        PUSH AF
        CALL LEFTUS
        CALL STRPRT
        LD HL,(FACLOW)
        POP AF
        SUB (HL)
        LD B,A
        LD  A,020H
        INC B
L63F3:
        DEC B
        JP Z,L6389
        RST 18H
        JR L63F3
L63FA:
        PUSH AF
        LD A,D
        OR A
        LD  A,02BH
        CALL NZ,OUTCHAR
        POP AF
        RET
L6404:
        CALL LFF09
OUTCON: 6407            ; ***************** Ausgabe eines
        CALL ISFLIO     ;  Zeichens auf den Bildschirm
        JP NZ,FILOUT
        LD  A,(ODEVLINK)
        OR A
        JP Z,TTYCHR     ; Z nicht auf Drucker  6435
        POP AF
OUTDLP: 6415            ; ***************** Ausgabe eines
        PUSH AF         ;  Zeichens auf den Drucker
        CP 09H          ; HT test
```

```
        JR NZ,L6428
L641A:
        LD  A,' '       ; SPACE ausgeben
        CALL OUTDLP
        LD  A,(LPOS)    ; bis LPOS die naechste
        AND 07H         ; 8 ter Stelle
        JR NZ,L641A     ; erreicht hat
        POP AF
        RET
L6428:
        SUB ODH         ; CR Test
        JR Z,L6436
        JR C,L6439      ; 0-0CH direkt ausgeben
        CP 013H         ; Test gegen US
        JR C,L6439      ; alle Steuerzeichen ausser CR
        LD  A,(LPOS)    ; aendern LPOS nicht
        INC A
L6436:
        LD (LPOS),A
L6439:
        POP AF
LPTCHR: 643A            ; nun wird das Zeichen ausgegeben
        JP CHPLPT
FINLPT: 6430            ; ***********
        XOR A           ; Ausgabe zum Drucker beendet
        LD (ODEVLINK),A ; naechste Ausgabe zum Schirm
        LD  A,(LPOS)
        OR A
        RET Z           ; falls ein Zeichen ausgegeben wurde
PRINTW: 6446
        LD  A,ODH       ; kommt noch ein CR
        CALL LPTCHR
        LD  A,0AH       ; LF hinterher
        CALL LPTCHR
        XOR A
        LD (LPOS),A     ; und LPOS wird wieder geloescht
        RET
TTYCHR: 6455            ; ************************
        POP AF          ; Zeichen holen
LINPT1: 6456
        PUSH AF         ; und wieder sichern
        CALL CHPUT 3F4D
        LD  A,(CSRX)    ; laeuft von 1..80
        DEC A
        LD (TTYPOS),A   ; TTYPOS von 0..79
        POP AF
        RET
CRDONZ: 6463            ; **********************
        LD  A,(TTYPOS)  ; falls ein Zeichen ausgegeben wurde
        OR A
        RET Z
        JR CRDO         ; kommt hier ein CR/LF hinterher
FININL: 646A            ; ********************
        LD (HL),00H     ; beenden einer Eingabe Zeile mit 0
        CALL ISFLIO
        LD  HL,BUFMIN
        JR NZ,CRFIN     ; falls Datei IO nichts weiter
CRDO:   6474
        CALL LFED9      ; CR / LF auf output device
        LD  A,ODH
        RST 18H
```



```
        LD  A,0AH
CRFIND: 647C
        RST 18H
CRFIN:  6480
        CALL ISFLIO
        JR Z,L6484      ; jp falls keine Datei IO
        XOR A
        RET
L6484:
        LD  A,(ODEVLINK)
        OR A
        JR Z,L648F
        XOR A           ; Ausgabegeraet Drucker
        LD (LPOS),A     ; sonst LOPS zuruecksetzen
        RET
L648F:
        XOR A           ; Ausgabegeraet Bildschirm
        LD (TTYPOS),A
        XOR A
        RET
ISCNTC: 6495            ; ************************
        PUSH HL         ; PRUEFT STOP TASTE
        LD  HL,INTFLG
        DI
        LD A,(HL)       ; alten Inhalt holen
        LD (HL),00H     ; Zuruecksetzen
        POP HL
        EI
        AND A
        RET Z           ; Z falls STOP taste nicht gedrueckt
        CP 03H
        JR Z,L64B5      ; Z falls CTRL STOP
        PUSH HL
        LD  HL,INTFLG
L64A9:                  ; STOP Warteschleife
        DI
        LD A,(HL)
        LD (HL),00H
        EI
        AND A
        JR Z,L64A9
        POP HL
        CP 03H
        RET NZ          ; NZ falls zweites mal STOP
L64B5:                  ; Abbruch wegen CTRL STOP
        PUSH HL
        CALL CKSTTP
        JR NC,L64CB     ; NC falls Stoptrap nicht definiert
        CALL CHSNS
        CALL NZ,CHGET/TRYIN
        LD  HL,STOP_00S
        DI
        CALL REQTRP
L64C8:
        EI
        POP HL
        RET
L64CB:
        LD HL,(PUTPNT)
        LD (GETPNT),HL  ; Loeschen des Eingabepuffers
        POP HL
```

```
        LD  A,03H       ; Ausgabe von ^C CR LF
        CALL CTRLPT
        LD SP,(SAVSTK)
        PUSH BC
        JP STOP
CKSTTP: 6405            ; Pruefen ob STOP definiert ist
        LD  A,(STOP_OOS)
        RRCA
        RET NC          ; NC falls kein STOP ON
        LD HL,(STOP_OOS+1)
        LD A,H
        OR L
        RET Z           ; Z falls kein ON STOP GOSUB
        LD HL,(CURLIN)
        INC HL
        LD A,H
        OR L
        RET Z           ; Z falls im direkten Modus
        SCF             ; C fuer aktiven STOP Trap
        RET
INKEY:  6413            ; ********************** INKEY$
        RST 10H
        PUSH HL
        CALL CHSNS      ; Scan Keyboard
        JR Z,L6506      ; Z kein Zeichen
        CALL CHGET      ; Zeichen holen
        PUSH AF
        CALL STRIN1     ; String aus einem Zeichen erzeugen
        POP AF
        LD E,A
        CALL L6B26
L6506:                  ; falls kein Zeichen vorhanden
        LD  HL,089EH    ; wird ein String mit der Laenge 0
        LD (FACLOW),HL  ; als Ergebnis zurueckgegeben
        LD  A,03H
        LD (VALTYP),A
        POP HL
        RET
OUTCH1: 6513            ; Ausgabe eines Zeichens
        RST 18H
        CP OAH
        RET NZ          ; falls LF gewesen
        LD  A,ODH
        RST 18H         ; CR anfuegen
        CALL CRFIN      ; und ausgabe beenden
        LD  A,OAH       ; A wieder mit richtigem Zeichen setzen
        RET
BLTU:   6520
        CALL REASON
BLTUC:  6523            ; *** Speicherbereich kopieren
        PUSH BC         ; BC Quelle     HL Ziel     DE Bremse
        EX (SP),HL      ; Es wird abwaerts kopiert bis DE genau
        POP BC          ; erreicht ist
L6526:                  ; **** HL Quelle    BC Ziel     DE Bremse
        RST 20H
        LD A,(HL)
        LD (BC),A
        RET Z
        DEC BC
        DEC HL
        JR L6526
```



```
GETSTK: 652E            ; Pruefen ob C Worte auf den Stack passen
        PUSH HL         ; HL ist neuer Stack
        LD HL,(STREND) F7F2
        LD B,00H
        ADD HL,BC
        ADD HL,BC
        DEFB 03EH       ;*      LD  A,0E5H
REASON: 6537            ; Pruefung gegen overflow in Stackbereich
        PUSH HL         ;*
        LD  A,088H
        SUB L
        LD L,A
        LD  A,0FFH
        SBC A,H
        LD H,A          ; HL:=FF88-HL
        JR C,OMERR
        ADD HL,SP
        POP HL
        RET C
OMERR:  6545
        CALL LINKER   AE5
        LD HL,(STKTOP)  F546
        DEC HL
        DEC HL          F7FD
        LD (SAVSTK),HL
OMERRR: 6550            ; out of memory
        LD  DE,07H
        JP ERROR  9074
SCRATH: 6556            ; ************************* NEW
        RET NZ          ; falls NEW ... syntax error
SCRTCH: 6557
        LD HL,(TXTTAB)  ; Programm anfang
        CALL TOFF       ; trace aus
        LD (AUTFLG),A   ; clear AUTO
        LD (PTRFLG),A
        LD (HL),A       ; Programm loeschen
        INC HL
        LD (HL),A       ; durch 2 null Bytes
        INC HL
        LD (VARTAB),HL  ; Variablenbereich loeschen
RUNC:   656A            ; entry fuer RUN
        CALL LFEF7
        LD HL,(TXTTAB)
        DEC HL
CLEARC: 6571            ; entry fuer CLEAR
        CALL LFEAF
        LD (TEMP),HL
CLEARO: 6577
        CALL INITRP     ; Initialisieren der Traps
        LD B,01AH       ; 26 Buchstaben
        LD  HL,DEFTBL
        CALL LFEE2
L6582:
        LD (HL),08H     ; alle Variablen doppelte Genauigkeit
        INC HL
        DJNZ L6582
        CALL RNDINI     ; Zufallszahlen initialisieren
        XOR A
        LD (ONEFLG),A   ; ON ERROR GOTO 0
        LD L,A
        LD H,A
```

```
        LD (ONELIN),HL
        LD (OLDTXT),HL
        LD HL,(MEMSIZ)
        LD (FRETOP),HL  ; Setzen des maximale Speichers
        CALL RESTOR     ; RESTORE
        LD HL,(VARTAB)
        LD (ARYTAB),HL  ; Loeschen der ARRAYs
        LD (STREND),HL  ; und der STRINGS
        CALL CLSALL     ; alle Dateien schliessen
        LD  A,(NLONLY)
        AND 01H
        JR NZ,STKINI
        LD (NLONLY),A
STKINI: 65B5            ; BASIC Stack initialisieren
        POP BC
        LD HL,(STKTOP)
        DEC HL
        DEC HL
        LD (SAVSTK),HL  ; Stackptr:= String data-2
        INC HL
        INC HL
STKERR: 65C0
        CALL LFEA0
        LD SP,HL
        LD  HL,TEMPST
        LD (TEMPPT),HL  ; leerer String Literal Pool
        CALL FINLPT     ; Ausgabe von CR zum Drucker
        CALL FINPRT     ; Ausgabe wieder auf Bildschirm
        XOR A
        LD H,A
        LD L,A
        LD (PRMLEN),HL
        LD (NOFUNS),A
        LD (PARMLN2),HL
        LD (FUNACT),HL
        LD (PRMSTK),HL
        LD (SUBFLG),A
        PUSH HL
        PUSH BC         ; ret Adresse
GTMPRT: 65E7
        LD HL,(TEMP)    ; Programmtext-pointer
        RET
ONTRP:  65EB            ; TRAP ON
        DI
        LD A,(HL)
        AND 04H         ; TRAP Request maskieren
        OR 01H          ; ON Bit setzen
        CP (HL)
        LD (HL),A
        JR Z,L65F9
        AND 04H
        JR NZ,L6634     ; wenn in der OFF Zeit TRAPsignal war
L65F9:
        EI
        RET
OFFTRP: 65FB            ; TRAP OFF
        DI
        LD A,(HL)
        LD (HL),00H     ; Trap abschalten
        JR L6608
STPTRP: 6601            ; TRAP STOP
```



```
        DI
        LD A,(HL)
        PUSH AF
        OR 02H          ; STOP bit setzen
        LD (HL),A       ; und speichern
        POP AF
L6608:
        XOR 05H
        JR Z,L6647      ; JP falls Request und ON
        EI
        RET
RSTTRP: 660E            ; wieder einschalten des TRAPs
        DI
        LD A,(HL)
        AND 05H         ; STOP loeschen
        CP (HL)
        LD (HL),A
        JR NZ,L662D     ; JP falls STOP gesetzt war
        EI
        RET
REQTRP: 6618            ; TRAP signalisieren falls eingeschaltet
        LD A,(HL)
        AND 01H
        RET Z           ; ret falls nicht ON
        LD A,(HL)
        OR 04H
        CP (HL)
        RET Z           ; ret falls er noch laeuft
        LD (HL),A       ; TRAP Request signalisieren
        XOR 05H
        RET NZ          ; ret falls TRAP STOP
        LD  A,(ONGSBF)
        INC A
        LD (ONGSBF),A   ; ON GOSUB FLAG erhoehen
        RET
L662D:                  ; pruefen ob eingeschaltet werden muss
        XOR 05H
        JR Z,L6634      ; es muss wohl
        EI
        RET
SETTRP: 6633            ; ON GOSUB erhoehen
        DI
L6634:
        LD  A,(ONGSBF)
        INC A
        LD (ONGSBF),A
        EI
        RET
FRETRP: 663D            ; TRAP Request freigeben
        DI
        LD A,(HL)
        AND 03H         ; Request Bit loeschen
        CP (HL)
        LD (HL),A       ; TRAP Request loeschen
        JR NZ,L6647     ; JP falls TRAP Request aktiv war
L6645:
        EI
        RET
L6647:                  ; Loeschen der TRAP- Stufe
        LD  A,(ONGSBF)
        SUB 01H
```

```
        JR C,L6645      ; JP falls ON GOSUB flag = 0
        LD (ONGSBF),A
        EI
        RET
INITRP: 6653            ; TRAPs initialisieren
        LD  HL,TRPTBL
        LD B,015H       ; 21 TRAPs
        XOR A
L6659:
        LD (HL),A
        INC HL
        LD (HL),A
        INC HL
        LD (HL),A
        INC HL
        DJNZ L6659
        LD  HL,FNKFLG   ; ON KEY GOSUB Flags auf null setzen
        LD B,0AH        ; 10 Funktionstasten
L6666:
        LD (HL),A
        INC HL
        DJNZ L6666
        LD (ONGSBF),A   ; und ON GOSUB Flag loeschen
        RET
GOTRP:  666E            ; *************** TRAP suchen, der
        LD  A,(ONEFLG)  ; ausgefuehrt werden muss
        OR A
        RET NZ          ; ret falls ON ERROR definiert
        PUSH HL
        LD HL,(CURLIN)
        LD A,H
        AND L
        INC A
        JR Z,L668B      ; JP falls direkter Modus
        LD  HL,TRPTBL
        LD B,015H
L6681:
        LD A,(HL)
        CP 05H
        JR Z,L668D      ; JP falls TRAP Request und eingeschaltet
L6686:
        INC HL
        INC HL
        INC HL
        DJNZ L6681
L668B:                  ; direkter Modus oder kein TRAP
        POP HL
        RET
L668D:                  ; TRAP Request und TRAP ON
        PUSH BC         ; Trap counter
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)       ; DE Zeilennummer fuer TRAP
        DEC HL
        DEC HL
        LD A,D
        OR E
        POP BC          ; Trap counter
        JR Z,L6686      ; JP falls Zeilennummer 0 ist
        PUSH DE
```



```
        PUSH HL
        CALL FRETRP     ; Trap freigeben
        CALL STPTRP     ; und TRAP STOP ausfuehren
        LD  C,03H
        CALL GETSTK     ; 3 Worte holen
        POP BC          ; Trap pointer
        POP DE          ; Trap zeilennummer
        POP HL          ; HL bei entry
        EX (SP),HL
        POP HL
        JP GOSUB2
RESTOR: 66AE            ; ***************************** RESTORE
        EX DE,HL
        LD HL,(TXTTAB)
        JR Z,L66C2
        EX DE,HL
        CALL LINGET     ; Zeilennummer holen
        PUSH HL
        CALL FNDLIN     ; im Programm suchen
        LD H,B
        LD L,C
        POP DE
        JP NC,USERR     ; Error falls nicht vorhanden
L66C2:
        DEC HL
RESFIN: 66C3
        LD (DATPTR),HL  ; Pointer in Programmtext
        EX DE,HL
        RET
STOPP:  66C8            ; ***************************** STOP
        JP NZ,STOPTP    ; noch mehr also nach STOP ..
STOP:   66CB
        RET NZ          ; Error falls hier noch was kommt
STOPRG: 66CC
        INC A
        JR CONSTP
ENDST:  66CF            ; ***************************** END
        RET NZ          ; evt. Syntax error
        XOR A
        LD (ONEFLG),A   ; ON ERROR loeschen
        PUSH AF
        CALL Z,CLSALL   ; alle Dateien schliessen
        POP AF
CONSTP: 66D9
        LD (SAVTXT),HL  ; Programm pointer speichern
        LD  HL,TEMPST
        LD (TEMPPT),HL  ; String literale loeschen
        DEFB 021H       ;*      LD  HL,0FFF6H
STPEND: 66E3
        OR 0FFH         ;*
        POP BC          ; return adresse
ENDCON: 66E6
        LD HL,(CURLIN)
        PUSH HL         ; aktuelle Zeilennummer
        PUSH AF         ; Status
        LD A,L
        AND H
        INC A
        JR Z,L66F9      ; jp falls direkter Modus
        LD (OLDLIN),HL  ; sonst speichern
        LD HL,(SAVTXT)
```

```
        LD (OLDTXT),HL  ; auch den Pointer ins Programm
L66F9:
        CALL FINLPT     ; Druckerpuffer leeren
        CALL CRDONZ     ; CR LF evtl. ausgeben
        POP AF          ; status holen
        LD  HL,BRKTXT
        JP NZ,ERRFIN    ; es war STOP also BREAK ..
        JP STPRDY
CTROPT: 6790            ; **** ^O ausgeben
        LD  A,0FH
CTRLPT: 6798            ; **** Control Zeichen ausgeben
        PUSH AF
        PUSH HL
        CALL TOTEXT     ; Textmodus einschalten
        POP HL
        LD  A,'^'
        RST 18H
        POP AF
        ADD A,040H
        RST 18H
        JP CRDO         ; mit folgendem CR
CONT:   67AB            ******************************** CONT
        LD HL,(OLDTXT)
        LD A,H
        OR L
        LD  DE,011H
        JP Z,ERROR      ; JP falls alter Programm Pointer 0 ist
        LD DE,(OLDLIN)
        LD (CURLIN),DE  ; Zeilennummer setzen
        RET
TON:    67C5            ; ***************************** TRON
        DEFB 03EH       ;*      LD  A,0AFH
TOFF:   67C8            ; ***************************** TROFF
        XOR A           ;*
        LD (TRCFLG),A
        RET
SWAP:   67CB            ; ***************************** SWAP
        CALL PTRGET     ;                          A,B
        PUSH DE         ; Variablenpointer sichern
        PUSH HL         ; Programm Pointer
        LD  HL,SWPTMP
        CALL VMOVE      ; A nach TMP schieben
        LD HL,(ARYTAB)
        EX (SP),HL
        RST 30H
        PUSH AF
        RST 8H
        DEFB ','
        CALL PTRGET     ; naechster Variablen pointer
        POP AF
        LD B,A          ; alter Variablentyp
        RST 30H
        CP B
        JP NZ,TMERR     ; JP falls Typen verschieden
        EX (SP),HL
        EX DE,HL
        PUSH HL
        LD HL,(ARYTAB)
        RST 20H
        JR NZ,L676B     ; JP falls komplette ARRAYs
        POP DE
```



```
        POP HL
        EX (SP),HL
        PUSH DE
        CALL VMOVE      ; B nach A schieben
        POP HL
        LD  DE,SWPTMP
        CALL VMOVE      ; TMP nach b schieben
        POP HL
        RET
L676B:
        JP FCERR
ERASE:  6706            ; ***************************** ERASE
        LD  A,01H
        LD (SUBFLG),A
        CALL PTRGET     ; Variablenpointer holen
        PUSH HL
        LD (SUBFLG),A
        LD H,B
        LD L,C
        DEC BC
        DEC BC
        DEC BC
        DEC BC
        DEC BC
        ADD HL,DE
        EX DE,HL
        LD HL,(STREND)
L6786:                  ; Verschieben der verbleibenden Variablen
        RST 20H
        LD  A,(DE)
        LD (BC),A
        INC DE
        INC BC
        JR NZ,L6786
        DEC BC
        LD H,B
        LD L,C
        LD (STREND),HL
        POP HL
        LD A,(HL)
        CP ','
        RET NZ          ; erase fertig
        RST 10H         ; sonst nachstes Zeichen
        JR ERASE        ; und weiter
POPAHT: 67B3
        POP AF
        POP HL
        RET
ISLET:  67B6            ; *** pruefen ob (HL) Grossbuchstabe ist
        LD A,(HL)
ISLET2: 67BF            ; ***     "   "   A     "           "
        CP 'A'
        RET C
        CP 'Z'+1
        CCF
        RET
CLEAR:  67A6            ; ***************************** CLEAR
        JP Z,CLEARC
        CALL LFEAF
        CALL INTID2     ; String space
        DEC HL
```

```
        RST 10H
        PUSH HL
        LD HL,(HIMEM)
        LD B,H
        LD C,L
        LD HL,(MEMSIZ)  ; BC=HIMEM      HL=MEMSIZ
        JR Z,L67E6      ; JP falls nur String space
        POP HL
        RST 8H
        DEFB ','
        PUSH DE
        CALL FRMQNT
        DEC HL
        RST 10H
        JP NZ,SNERR     ; falls weitere Parameter
        EX (SP),HL
        EX DE,HL        ; HL Parameter
        LD A,H
        AND A
        JP P,FCERR      ; muss > 7FFF sein
        PUSH DE
        LD  DE,0F501H   ; feste Grenze DISK BASIC wird
        RST 20H         ;       nicht geschuetzt
        JP NC,FCERR     ; ist nicht kleiner
        POP DE
        PUSH HL
        LD  BC,0FEF5H   ; negative Dateipuffer- Laenge
        LD  A,(MAXFILES)
L67DF:
        ADD HL,BC
        DEC A
        JP P,L67DF      ; noch mehr Dateipuffer
        POP BC
        DEC HL
L67E6:
        LD A,L
        SUB E
        LD E,A
        LD A,H
        SBC A,D
        LD D,A          ; DE:=HL-DE
        JP C,OMERR
        PUSH HL
        LD HL,(VARTAB)
        PUSH BC
        LD  BC,0A0H     ; Minimum fuer Variablen
        ADD HL,BC
        POP BC
        RST 20H
        JP NC,OMERR
        EX DE,HL
        LD (STKTOP),HL  ; STACK setzen
        LD H,B
        LD L,C
        LD (HIMEM),HL
        POP HL
        LD (MEMSIZ),HL
        POP HL
        CALL CLEARC
        LD  A,(MAXFILES)
        CALL DEFILE
```



```
        LD HL,(TEMP)
        JP NEWSTT
SUBDE:  6814            ; *** DE:=DE-HL
        LD A,L
        SUB E
        LD E,A
        LD A,H
        SBC A,D
        LD D,A
        RET
NEXT:   6821            ; ****************************** NEXT
        LD  DE,00H
L6824:
        CALL NZ,PTRGET  ; Variable holen
        LD (TEMP),HL
        CALL FNDFOR     ; FOR suchen
        JP NZ,NFERR     ; JP bei NEXT without FOR
        LD SP,HL
        PUSH DE         ; Adresse des Index
        LD A,(HL)       ; Vorzeichen des Increments
        PUSH AF
        INC HL
        PUSH DE
        LD A,(HL)
        INC HL
        OR A
        JP M,L6865      ; JP falls INTEGER Typ
        DEC A
        JR NZ,L6843
        LD  BC,08H
        ADD HL,BC
L6843:
        ADD A,04H
        LD (VALTYP),A
        CALL VMOVFM     ; Increment vom Stack holen
        EX DE,HL
        EX (SP),HL
        PUSH HL
        RST 30H
        JR NC,L689F
        CALL MOVRMI     ; Index holen
        CALL FADD       ; Addieren
        POP HL
        CALL MOVMF      ; nach Index speichern
        POP HL
        CALL MOVRM      ; Endwert holen
        PUSH HL
        CALL FCOMP      ; vergleichen mit Index
        JR L688E
L6865:
        LD  BC,0CH
        ADD HL,BC
        LD C,(HL)
        INC HL
        LD B,(HL)       ; BC Increment
        INC HL
        EX (SP),HL
        LD E,(HL)
        INC HL
        LD D,(HL)       ; DE Index
        PUSH HL
```

```
        LD L,C
        LD H,B
        CALL IADD       ; Addieren
        LD  A,(VALTYP)
        CP 02H
        JP NZ,DVERR     ; JP falls Ueberlauf
        EX DE,HL
        POP HL
        LD (HL),D
        DEC HL
        LD (HL),E       ; Wegspeichern des Index
        POP HL
        PUSH DE
        LD E,(HL)
        INC HL
        LD D,(HL)
        INC HL
        EX (SP),HL
        CALL ICOMP      ; Vergleich gegen Grenzwert
L688E:
        POP HL          ; Zeilennummer von FOR
        POP BC
        SUB B
        CALL MOVRM
        JR Z,L68B0      ; Jp falls Grenzwert ueberschritten
        EX DE,HL
        LD (CURLIN),HL
        LD L,C
        LD H,B
        JP NXTCON
L689F:
        CALL DADOS
        POP HL
        CALL VMOVMF
        POP HL
        CALL VMOVAM
        PUSH DE
        CALL XDCOMP
        JR L688E
L68B0:
        LD SP,HL
        LD (SAVSTK),HL
        EX DE,HL
        LD HL,(TEMP)
        LD A,(HL)
        CP ':'
        JP NZ,NEWSTT    ; JP falls kein : zum naechsten Statement
        RST 10H
        CALL L6824      ; naechstes NEXT
ISFLIO: 6862            ; *****  Testen ob die Ein-/Ausgabe
        CALL LFF15      ; an eine Datei gebunden ist
        PUSH HL         ; ret mit NZ falls ja
        LD HL,(PTRFIL)
        LD A,H
        OR L
        POP HL
        RET
STRCMP: 65CD            ; *** Vergleich von Strings
        CALL FRESTR
        LD A,(HL)
        INC HL
```



```
        LD C,(HL)
        INC HL
        LD B,(HL)
        POP DE
        PUSH BC         ; Str Pointer
        PUSH AF         ; Str Laenge
        CALL FRETMP
        POP AF
        LD D,A
        LD E,(HL)
        INC HL
        LD C,(HL)
        INC HL
        LD B,(HL)
        POP HL          ; Str1  D,HL    Str2  E,BC
L68E3:
        LD A,E
        OR D
        RET Z           ; ret mit Z falls beide Laenge 0
        LD A,D
        SUB 01H
        RET C           ; ret mit NZ,C falls Str1 Laenge 0
        XOR A
        CP E
        INC A
        RET NC          ; ret mit NZ,NC falls Str2 Laenge 0
L68EE:
        DEC D
        DEC E
        LD  A,(BC)
        INC BC
        CP (HL)
        INC HL
        JR Z,L68E3      ; jp falls Bytes identisch
        CCF
        JP SIGNS
STRO$:  68FA            ; ***************************** OCT$
        CALL FOUTD
        JR L690C
STRH$:  68FF            ; ***************************** HEX$
        CALL FOUTH
        JR L690C
STRB$:  6904            ; ***************************** BIN$
        CALL FOUTB
        JR L690C
STR$:   6909            ; ***************************** STR$
        CALL FOUT
L690C:
        CALL STRLIT     ; string literal pool entry erzeugen
        CALL FREFAC     ; Adr des Descriptors nach HL
        LD  BC,FINBCK   ; CHR$ adr als Return- Adresse
        PUSH BC
STRCPY: 6916            ; String kopieren
        LD A,(HL)       ; Laenge des Strings
        INC HL
        PUSH HL
        CALL GETSPA     ; pruefen, ob noch Platz fuer String ist
        POP HL
        LD C,(HL)
        INC HL
        LD B,(HL)       ; BC= Adresse des Strings
```

```
        CALL L692F      ; Speichern von Laenge und Adresse
        PUSH HL
        LD L,A          ; L= Laenge des Strings
        CALL L6ACC      ; String von BC (temp. Bereich) nach
        POP DE          ;       DE (string data)
        RET             ; Ret zu CHR$
STRIN1: LD A,01H        ; **** String von 1 Byte Laenge erzeugen
STRINI: CALL GETSPA     ; **** String von A Byte Laenge erzeugen
L692F:                  ; *** String-Descriptor zwischenspeichern
        LD HL,DSCTMP
STRAD1: PUSH HL         ; *** String-Descriptor speichern
        LD (HL),A
PUTDEI: INC HL
        LD (HL),E
        INC HL
        LD (HL),D
        POP HL
        RET
STRLIT: DEC HL          ; **** Erzeugung eines String-Descriptors
STRLTI: LD B,'"'        ; aus einer Konstanten  nach DSCTMP
STRLT3: LD D,B          ; auf Quote " positionieren
STRLT2: PUSH HL         ; Ende Zeichen nach D
        LD C,0FFH       ; Adresse der Quote
L6941:                  ; -1 fuer quote- position
        INC HL
        LD A,(HL)       ; Zeichen holen
        INC C
        OR A
        JR Z,L694D      ; JP falls Eingabe zu Ende
        CP D            ; Ende Zeichen erreicht ?
        JR Z,L694D      ; JP falls ja
        CP B
        JR NZ,L6941     ; evtl. 2. Ende Zeichen
L694D:
        CP '"'          ; falls letztes Zeichen Quote war
        CALL Z,CHRGTR   ; Folgezeichen holen
        EX (SP),HL      ; HL Adresse der ersten Quote
        INC HL
        EX DE,HL        ; Stringanfang nach DE
        LD A,C          ; Stringlaenge
        CALL L692F      ; Speichern in DSCTMP
PUTNEW: LD DE,DSCTMP    ; Speichern des DCSTMP Strings in den
        DEFB 03EH       ;       Literal Pool
PUTTMP: PUSH DE         ;*      LD  A,0D5H
        LD HL,(TEMPPT)  ;*** Speichern eines String-Descriptors
        LD (FACLOW),HL  ;*    (in DE) in den Literal Pool
        LD A,03H        ; Adresse des naechsten Stringentrys
        LD (VALTYP),A   ; Typ String
        CALL VMOVE      ; Kopieren des descriptors
        LD DE,FRETOP
        RST 20H
```



```
        LD (TEMPPT),HL  ; neuer naechster Eintrag
        POP HL
        LD A,(HL)
        RET NZ          ; wenn Stringbereich nicht Uebergelaufen
        LD DE,010H      ; sonst Fehler
        JP ERROR
STROUI: INC HL          ; ****************
STROUT: CALL STRLIT     ; ****************
        CALL FREFAC     ; Stringliteral erzeugen
STRPRT: CALL GETBCD     ; ****************
        INC D           ; Descriptor der Variablen holen
L6987:                  ; Laenge in D   Adresse BC
        DEC D
        RET Z           ; ret bei Stringende oder Laenge 0
        LD A,(BC)
        RST 18H         ; Ausgabe des Zeichens
        CP 0DH
        CALL Z,CRFIN    ; dies falls CR gewesen
        INC BC          ; naechste Adresse
        JR L6987
GETSPA: OR A            ; Berechnen des zur Verfuegung stehenden
        DEFB 0EH        ;    Platzes im Stringbereich
L6995:                  ;*      LD  C,0F1H
        POP AF          ;*
        PUSH AF         ; Stringlaenge sichern
        LD HL,(STKTOP)  ; Anfangsadresse Stringbereich
        EX DE,HL
        LD HL,(FRETOP)  ; Naechster verfuegbarer Stringeintrag
        CPL
        LD C,A
        LD B,0FFH       ; negative Stringlaenge -1
        ADD HL,BC
        INC HL          ; echtes Zweierkomplement
        RST 20H         ; Vergleich gegen Grenze
        JR C,L69AE      ; falls C out of stringspace
        LD (FRETOP),HL  ; neuer Naechster verfuegbarer Stringeintrag
        INC HL
        EX DE,HL        ; neue Adresse fuer diesen String
PPSWRT: POP AF          ; Stringlaenge
        RET
L69AE:
        POP AF
        LD DE,0EH
        JP Z,ERROR      ; JP falls kein Platz mehr
        CP A
        PUSH AF
        LD BC,L6995     ; noch einmal versuchen
        PUSH BC
GARBA2:                 ; Strings aus dem Literal Pool in den
        LD HL,(MEMSIZ)  ;       Stringbereich kopieren
L69BE:                  ; Hoechste verfuegbare Adresse
        LD (FRETOP),HL  ; in aktuellen Stringpointer
        LD HL,00H
        PUSH HL
        LD HL,(STREND)  ; Ende Stringbereich
```

```
        PUSH HL
        LD  HL,TEMPST   ; Adr erster Eintrag der Stringpointer
L69CC:
        LD DE,(TEMPPT)  ; Adr aktueller Eintrag
        RST 20H
        LD  BC,L69CC
        JP NZ,L6A47     ; wenn der erste Eintrag erreicht ist
        LD  HL,PARMPRV  ; Pointer auf die einfachen Variablen
        LD (TEMP9),HL   ; als aktuellen Zeiger speichern
        LD HL,(ARYTAB)  ; Pointer zu ARRAY Variablen
        LD (ARYTA2),HL  ; als Bremse speichern
        LD HL,(VARTAB)  ; Anfang der zu durchsuchenden Variablen
L69E6:
        LD DE,(ARYTA2)  ; falls sie gleich sind
        RST 20H
        JR Z,L69FF      ; wenn alle Variablen aus Bereich geprueft
        LD A,(HL)       ; Typ der Variablen
        INC HL
        INC HL          ; Variablennamen ueberspringen
        INC HL
        CP 03H
        JR NZ,L69F9     ; JP falls nicht String
        CALL L6A48      ; Stringadresse holen
        XOR A           ; HL zeigt auf naechste Variable
L69F9:
        LD E,A          ; A=Typ entspricht Variablenlaenge
        LD D,00H
        ADD HL,DE       ; HL+Variablenlaenge zeigt auf
        JR L69E6        ;       naechste Variable
L69FF:
        LD HL,(TEMP9)   ; alle Variablen geprueft
        LD E,(HL)
        INC HL
        LD D,(HL)
        LD A,D
        OR E
        LD HL,(ARYTAB)
        JR Z,L6A1F      ; JP falls alle Eintraege geprueft
        EX DE,HL
        LD (TEMP9),HL
        INC HL
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        INC HL
        EX DE,HL
        ADD HL,DE
        LD (ARYTA2),HL
        EX DE,HL
        JR L69E6
L6A1E:
        POP BC
L6A1F:
        LD DE,(STREND)
        RST 20H
        JP Z,L6A68      ; JP falls Stringliteral gefunden
        LD A,(HL)       ; Typ des ARRAYs
        INC HL
        CALL MOVRM      ; offset zur naechsten Variablen
        PUSH HL
```



```
        ADD HL,BC
        CP 03H
        JR NZ,L6A1E     ; JP falls kein String-Array
        LD (TEMP8),HL   ; Adresse des naechsten ARRAYs
        POP HL
        LD C,(HL)       ; Anzahl der Indizes
        LD B,00H
        ADD HL,BC
        ADD HL,BC       ; in Worten
        INC HL
L6A3C:
        EX DE,HL
        LD HL,(TEMP8)
        EX DE,HL        ; HL Endadresse der Indizes
        RST 20H         ; DE Adresse der naechsten Variablen
        JR Z,L6A1F      ; JP falls Liste zu Ende
        LD  BC,L6A3C
L6A47:
        PUSH BC
L6A48:                  ; hier wird der String gegen die
        XOR A           ; Testadressen geprueft
        OR (HL)         ; Laenge des Strings
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)       ; DE Stringadresse
        INC HL
        RET Z           ; ret bei Laenge 0
        LD B,H
        LD C,L          ; BC Adresse des naechsten Descriptors
        LD HL,(FRETOP)
        RST 20H
        LD H,B
        LD L,C          ; Zurueckspeichern
        RET C           ; wenn der String im Stringbereich ist
        POP HL          ; ret Adresse
        EX (SP),HL
        RST 20H         ; Test gegen Testadresse auf dem Stack
        EX (SP),HL
        PUSH HL
        LD H,B
        LD L,C
        RET NC          ; ret falls unter der Testadresse
        POP BC
        POP AF
        POP AF
        PUSH HL
        PUSH DE
        PUSH BC
        RET

L6A68:          ; *** Literal in den Stringbereich kopieren
        POP DE  ; Adresse des zuletzt eingetragenen Strings
        POP HL  ; Adresse des naechsten Stringpointers
        LD A,H
        OR L
        RET Z           ; Z falls kein String
        DEC HL
        LD B,(HL)
        DEC HL
        LD C,(HL)       ; BC Adresse des Strings
```

```
        PUSH HL
        DEC HL
        LD L,(HL)
        LD H,00H        ; HL Laenge
        ADD HL,BC       ; HL Endadresse des Strings
        LD D,B
        LD E,C          ; DE Anfangsadresse
        DEC HL
        LD B,H
        LD C,L          ; BC HL Endadresse-1
        LD HL,(FRETOP)
        CALL BLTUC      ; String in Stringbereich kopieren
        POP HL
        LD (HL),C
        INC HL
        LD (HL),B       ; Adresse speichern
        LD H,B
        LD L,C
        DEC HL
        JP L69BE        ; JP um weitere Literaleintraege zu finden

CAT:/ 6A8C              ; ********************** zwei Strings
                        ;       konkatinieren
        PUSH BC
        PUSH HL
        LD HL,(FACLOW)
        EX (SP),HL
        CALL EVAL       ; Zweiten String berechnen
        EX (SP),HL
        CALL CHKSTR/FRCSTR ; es muss ein String sein
        LD A,(HL)       ; Laenge String1
        PUSH HL
        LD HL,(FACLOW)
        PUSH HL
        ADD A,(HL)      ; Laenge beider Strings
        LD  DE,0FH
        JP C,ERROR      ; falls >255 Zeichen
        CALL STRINI     ; Stringspeicher holen
        POP DE
        CALL FRETMP
        EX (SP),HL
        CALL FRETM2
        PUSH HL
        LD HL,(DSCPTR)  ; Adresse zweiter String
        EX DE,HL
        CALL L6AC4      ; String 1 in Arbeitsbereich
        CALL L6AC4      ; String 2
        LD  HL,TSTOP    ; Fortsetzungsadresse fuer expr eval
        EX (SP),HL
        PUSH HL
        JP PUTNEW       ; strl * Str2 in Literal Pool eintragen

L6AC4:                  ; **** String kopieren  DE Zieladresse
        POP HL          ; STACK Quell-Descriptor
        EX (SP),HL      ; HL Stringdescriptor-Adresse
        LD A,(HL)
        INC HL
        LD C,(HL)
        INC HL
```



```
        LD B,(HL)
        LD L,A          ; L Laenge   BC Adresse
L6ACC:
        INC L
L6ACD:
        DEC L
        RET Z           ; Z wenn Stringkopiert
        LD  A,(BC)
        LD (DE),A
        INC BC
        INC DE
        JR L6ACD


FRESTR:/ 6AD5           ; ******
        CALL CHKSTR/FRCSTR      ; es muss ein String sein
FREFAC:/ 6AD8
        LD HL,(FACLOW)  ; Descriptoradresse nach HL
FRETM2:/ 6ADB
        EX DE,HL        ; nach DE
FRETMP:/ 6ADC
        CALL FRETMS     ; ist es auch der letzte Literalpool
        EX DE,HL        ; Eintrag
        RET NZ          ; NZ in DE ist Variablenadresse
        PUSH DE         ;   BC Stringadr des letzten strings
        LD D,B
        LD E,C          ; Stringadresse
        DEC DE
        LD C,(HL)       ; Stringlaenge
        LD HL,(FRETOP)
        RST 20H
        JR NZ,L6AF1     ; ret falls nicht der letzte im Stringbereich
        LD B,A
        ADD HL,BC       ; neue Stringpointer Adresse
        LD (FRETOP),HL
L6AF1:
        POP HL
        RET
FRETMS:/ 6AF3           ; DE descadr
        CALL LFEA6      ; HL^ auf desc, BC adr des letzten strings
        LD HL,(TEMPPT)  ; naechster verfuegbare Stringadresse
        DEC HL
        LD B,(HL)
        DEC HL
        LD C,(HL)       ; Adresse des vorigen Strings
        DEC HL
        RST 20H
        RET NZ          ; NZ falls Loeschen nicht moeglich
        LD (TEMPPT),HL  ; aktuellen Eintrag erneuern
        RET             ; Z String geloescht
LEN:/ 6B04              ; ***************************** LEN
        LD  BC,SNGFLT
        PUSH BC
L6B08:
        CALL FRESTR     ; Adresse des aktuellen Stringdescriptors
        XOR A           ; nach HL
        LD D,A
        LD A,(HL)       ; Laenge
        OR A
        RET             ; RET nach Wandlung in einfach genaue Zahl
ASC:/ 6B10              ; ***************************** ASC
```

```
        LD  BC,SNGFLT
        PUSH BC
ASC2:   6B14
        CALL L6B08      ; Adresse und Laenge des Strings holen
        JP Z,FCERR      ; Z wenn Laenge 0
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)       ; Stringadresse
        LD  A,(DE)      ; erstes Zeichen
        RET             ; in einfach genaue Zahl wandeln

SETSTR: 6B20            ; ****************************** CHR$
        CALL STRIN1     ; String aus einem Zeichen erzeugen
        CALL CONINT     ; Zahl nach Integer wandeln
L6B26:
        LD HL,(DSCPTR)
        LD (HL),E       ; Zeichen in String eintragen
FINBCK: 6B2A
        POP BC
        JP PUTNEW       ; und in Literalpool bringen
STRNG$: 6B2E            ; ****************************** STRING$
        RST 10H
        RST 8H
        DEFB '('
        CALL GETBYT     ; Zeichenzahl holen
        PUSH DE
        RST 8H
        DEFB ','
        CALL FRMEVL     ; Zeichen holen
        RST 8H
        DEFB ')'
        EX (SP),HL
        PUSH HL
        RST 30H
        JR Z,L6B46
        CALL CONINT     ; in INTEGER wandeln
        JR L6B49
L6B46:                  ; Zeichen aus String holen
        CALL ASC2
L6B49:
        POP DE
        CALL L6B52
SPACE$: 6B4D            ; ************************** SPACE$
        CALL CONINT     ; Anzahl in ganze Zahl wandeln
        LD  A,' '       ; String aus SPACE erzeugen
L6B52:                  ; ***** String mit der Laenge in E
        PUSH AF         ; aus dem Zeichen A erzeugen
        LD A,E
        CALL STRINI     ; String erzeugen
        LD B,A
        POP AF
        INC B
        DEC B
        JR Z,FINBCK     ; Z Laenge 0
        LD HL,(DSCPTR)
L6B60:
        LD (HL),A       ; String mit Zeichen setzen
        INC HL
        DJNZ L6B60
        JR FINBCK       ; Abschluss
```



```
LEFT$:  6B66            ; ***************************** LEFT$
        CALL L6BE8
        XOR A           ; offset =0
L6B6A:                  ; right entry
        EX (SP),HL
        LD C,A          ; C Zahl der zu ueberspringenden Zeichen
        DEFB 03EH       ;*      LD  A,0E5H
LEFTUS: 6B6D
        PUSH HL         ;*
L6B6E:
        PUSH HL
        LD A,(HL)       ; Stringlaenge
        CP B
        JR C,L6B75
        LD A,B
        DEFB 011H       ;*      LD  DE,0EH
L6B75:
        LD C,0          ;*
        PUSH BC
        CALL GETSPA     ; Stringbereich gross genug
        POP BC
        POP HL
        PUSH HL
        INC HL
        LD B,(HL)
        INC HL
        LD H,(HL)
        LD L,B
        LD B,00H
        ADD HL,BC       ; offset addieren
        LD B,H
        LD C,L
        CALL L692F      ; pointer speichern
        LD L,A
        CALL L6ACC      ; Kopieren
        POP DE
        CALL FRETMP
        JP PUTNEW       ; Eintragen in Literal Pool
RIGHT$: 6B96            ; ***************************** RIGHT$
        CALL L6BE8
        POP DE
        PUSH DE
        LD  A,(DE)      ; Laenge des Strings
        SUB B           ; - den gewuenschten Zeichen
        JR L6B6A
MID$:   6B9F            ; ***************************** MID$
        EX DE,HL
        LD A,(HL)
        CALL L6BEB      ; BC position      DE Stringadresse
        INC B
        DEC B
        JP Z,FCERR      ; Z falls offset 0
        PUSH BC
        CALL L6CE9
        POP AF
        EX (SP),HL
        LD  BC,L6B6E
        PUSH BC
        DEC A           ; Startposition-1
```

```
        CP (HL)         ; gegen Stringlaenge
        LD B,00H        ; falls >Laenge weiter
        RET NC          ;      bei L6B6E mit Laenge 0
        LD C,A          ; C offset
        LD A,(HL)       ; Stringlaenge
        SUB C           ; A maximale Anzahl
        CP E            ; vergleichen mit MID Parameter
        LD B,A          ; weiter mit Laenge A
        RET C           ;        falls Parameter > restlaenge
        LD B,E          ; sonst Parameter als Laenge
        RET
VAL:    6BC0            ; ************************* VAL
        CALL L6B08      ; intern LEN
        JP Z,SNGFLT     ; falls Laenge 0 fertig
        LD E,A
        INC HL
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        PUSH HL         ; HL Stringadresse
        ADD HL,DE       ; HL Endadresse
        LD B,(HL)       ; Letztes Zeichen
        LD (VLZADR),HL  ; Adresse des letzten Zeichens
        LD A,B
        LD (VLZDAT),A   ; und das Zeichen speichern
        LD (HL),D       ; durch 0 ersetzen
        EX (SP),HL
        PUSH BC
        DEC HL
        RST 10H
        CALL FINDBL     ; Wandeln ASCII nach binaer
        LD  HL,00H
        LD (VLZADR),HL
        POP BC
        POP HL
        LD (HL),B       ; Zeichen zurueckspeichern
        RET
L6BE8:                  ; Syntax- Pruefung fuer LEFT$/RIGHT$
        EX DE,HL
        RST 8H
        DEFB ')'
L6BEB:                  ; Parameter holen fuer LEFT$/RIGHT$/MID$
        POP BC
        POP DE          ; Anzahl der Bytes
        PUSH BC
        LD B,E
        RET
INSTR:  6BF0            ; ************************** INSTR
        RST 10H
        CALL FRMPRN
        RST 30H
        LD  A,01H       ; Voreinstellung der Position
        PUSH AF
        JR Z,L6C0B      ; Z falls Parameter nicht vorhanden
        POP AF
        CALL CONINT
        OR A
        JP Z,FCERR      ; Z falls Position 0
        PUSH AF
        RST 8H
```



```
        DEFB ','
        CALL FRMEVL
        CALL CHKSTR/FRCSTR      ; zu durchsuchender String
L6C0B:
        RST 8H
        DEFB ','
        PUSH HL
        LD HL,(FACLOW)
        EX (SP),HL      ; Descriptor auf den Stack
        CALL FRMEVL     ; Suchausdruck berechnen
        RST 8H
        DEFB ')'
        PUSH HL
        CALL FRESTR
        EX DE,HL        ; DE Descriptor Suchstring
        POP BC          ; Programmpointer
        POP HL          ; HL Descriptor String 1
        POP AF          ; A Position
        PUSH BC         ; Programmpointer
        LD  BC,POPHRT
        PUSH BC
        LD  BC,SNGFLT   ; Ergebnis einfachgenau
        PUSH BC
        PUSH AF
        PUSH DE
        CALL FRETM2
        POP DE
        POP AF
        LD B,A
        DEC A
        LD C,A
        CP (HL)
        LD  A,00H
        RET NC          ; NC falls Position > Laenge String 1
        LD  A,(DE)
        OR A
        LD A,B
        RET Z           ; Z falls Suchstringlaenge 0 ist
        LD A,(HL)       ; Laenge String 1
        INC HL
        LD B,(HL)
        INC HL
        LD H,(HL)
        LD L,B          ; HL Adresse String 1
        LD B,00H
        ADD HL,BC       ; Position addieren
        SUB C           ; und von Laenge subtrahieren
        LD B,A          ; soviele Zeichen bleiben
        PUSH BC
        PUSH DE
        EX (SP),HL
        LD C,(HL)       ; Laenge Suchstring
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)       ; DE Adresse Suchstring
        POP HL          ; HL Adresse String 1
L6C4E:                  ; B Restlaenge String 1
        PUSH HL         ;      C Laenge Suchstring
        PUSH DE
        PUSH BC
```

```
L6C51:
        LD  A,(DE)
        CP (HL)
        JR NZ,L6C6B     ; NZ Zeichen verschieden
        INC DE
        DEC C
        JR Z,L6C62      ; Z Suchstring gefunden
        INC HL
        DJNZ L6C51
        POP DE          ; String 1 vorzeitig zu Ende
        POP DE
        POP BC
L6C5F:
        POP DE
        XOR A           ; 0 = nicht gefunden
        RET
L6C62:                  ; gefunden
        POP HL
        POP DE
        POP DE
        POP BC
        LD A,B
        SUB H
        ADD A,C
        INC A           ; A Position
        RET
L6C6B:                  ; Zeichen verschieden
        POP BC
        POP DE
        POP HL
        INC HL          ; naechste Position
        DJNZ L6C4E      ; ein Zeichen weniger im String 1
        JR L6C5F
LHSMID: 6C73            ; ************************** MID$(..)=
        RST 8H
        DEFB '('
        CALL PTRGET
        CALL CHKSTR/FRCSTR ; Stringdescriptor holen
        PUSH HL
        PUSH DE
        EX DE,HL
        INC HL
        LD E,(HL)               ; Stringadresse
        INC HL
        LD D,(HL)               ; in DE
        LD HL,(STREND)
        RST 20H
        JR C,L6C98              ; C keine Variable
        LD HL,(TXTTAB)
        RST 20H
        JR NC,L6C98             ; NC keine Variable
        POP HL
        PUSH HL
        CALL STRCPY
        POP HL
        PUSH HL
        CALL VMOVE
L6C98:
        POP HL
        EX (SP),HL
        RST 8H
```



```
        DEFB ','
        CALL GETBYT     ; Position holen
        OR A
        JP Z,FCERR      ; groesser 0 muss sie sein !
        PUSH AF
        LD A,(HL)
        CALL L6CE9      ; Laenge holen
        PUSH DE
        CALL FRMEQL     ; folgenden Stringausdruck berechnen
        PUSH HL
        CALL FRESTR
        EX DE,HL
        POP HL
        POP BC
        POP AF
        LD B,A          ; Positon
        EX (SP),HL      ; Programmpointer auf den Stack
        PUSH HL         ; HL Descriptoradresse der Variablen
        LD  HL,POPHRT   ; als ret- Adresse
        EX (SP),HL
        LD A,C
        OR A            ; Z falls Anzahl der zu
        RET Z           ;       ersetzenden Zeichen 0 ist
        LD A,(HL)       ; Laenge des Variablenstrings
        SUB B
        JP C,FCERR      ; falls < als Position Fehler
        INC A
        CP C            ; Vergleich gegen zu ersetzende Zeichen
        JR C,L6CC8      ; C falls Restlaenge < ist
        LD A,C
L6CC8:
        LD C,B
        DEC C
        LD B,00H
        PUSH DE
        INC HL
        LD E,(HL)
        INC HL
        LD H,(HL)       ; Adresse des Variablenstrings
        LD L,E          ; nach HL
        ADD HL,BC       ; Position Addieren
        LD B,A
        POP DE
        EX DE,HL
        LD C,(HL)       ; Laenge Stringausdruck
        INC HL
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A
        EX DE,HL        ; DE Adresse des Stringausdrucks
        LD A,C
        OR A
        RET Z           ; Z falls Ausdruck laenge 0
L6CE0:
        LD  A,(DE)      ; Stringausdruck hineinkopieren
        LD (HL),A
        INC DE
        INC HL
        DEC C
        RET Z
```

```
        DJNZ L6CE0
        RET
L6CE9:                  ; maximale Laenge falls nicht angegeben
        LD  E,0FFH
        CP ')'
        JR Z,L6CF4      ; Z es ist kein weiterer Parameter da
        RST 8H
        DEFB ','
        CALL GETBYT     ; sonst nach E holen
L6CF4:
        RST 8H
        DEFB ')'
        RET
FRE:    6CF7            ; ************************** FRE
        LD HL,(STREND)  ; Anfang des freien Bereiches
        EX DE,HL
        LD  HL,00H
        ADD HL,SP
        RST 30H
        JP NZ,GIVDBL    ; NZ falls numerisches Argument
        CALL FREFAC     ; Stringadresse nach HL
        CALL GARBA2     ; Garbage Collection
        LD DE,(STKTOP)
        LD HL,(FRETOP)  ; freien Stringspeicher berechnen
        JP GIVDBL


PINLIN: 6D13            ; ****** Input line routine
        CALL LFF42      ;       Einlesen bis CR oder STOP
        CALL ISFLIO     ;       falls STOP ret mit Carry
        JP NZ,L6F95     ; NZ falls Datei IO
        LD  A,(AUTFLG)
        AND A
        JR NZ,INLIN     ; NZ falls AUTO modus
        LD  L,00H
        JR L6D3A
QINLIN: 6D26            ; fuer Eingaberoutinen ohne Prompt
        CALL LFF45
        LD  A,'?'
        RST 18H
        LD  A,' '
        RST 18H
INLIN:  6D2F
        CALL LFF48
        LD HL,(CSRY)
        DEC L
        CALL NZ,TERMIN
        INC L
L6D3A:
        LD (FSTPOS),HL  ; Cursor position
        XOR A
        LD (INTFLG),A   ; INSERT loeschen
L6D41:
        CALL CHGET      ; Zeichen holen und speichern
        LD  HL,L6D7B-2
        LD  C,0BH
        CALL INDJMP     ; falls in Tabelle springen
        PUSH AF
        CALL NZ,L6D59   ; NZ falls INSERT
        POP AF
        JR NC,L6D41     ; NC falls kein ENTER
```



```
        LD  HL,BUFMIN   ; Pufferanfang
        RET Z           ; Z ENTER
        CCF             ; C CTRL-STOP
L6D58:                  ; ESC   ^[
        RET
L6D59:
        LD  HL,INSFLG
        CP ' '
        JR C,L6D69      ; C falls Steuerzeichen
        PUSH AF
        LD A,(HL)
        AND A
        CALL NZ,L6E1D
        POP AF
        RST 18H
        RET
L6D69:
        LD (HL),00H     ; INSERT loeschen
        RST 18H         ; zum Schirm schicken
L6D6C:
        LD A,'x'
        DEFB 1          ;*      LD  BC,0793EH
L6D6F:
        LD A,'y'        ;*
        PUSH AF
        LD  A,01BH      ; ESC
        RST 18H
        POP AF          ; x oder y
        RST 18H
        LD  A,'4'       ; 4.
        RST 18H
        RET

L6D7B:
        DEFB 8
        DEFW L6E84
        DEFB 12H
        DEFW L6E10
        DEFB 1BH
        DEFW L6D58
        DEFB 2
        DEFW L6F2E
        DEFB 6
        DEFW L6F1B
        DEFB 0EH
        DEFW L6EFA
        DEFB 5
        DEFW L6EDC
        DEFB 3
        DEFW L6DF5
        DEFB 13
        DEFW L6D9C
        DEFB 015H
        DEFW L6ED1
        DEFB 07FH
        DEFW L6E7E

L6D9C:                  ; CR    ^M      ENTER
        CALL L6F81
        LD  A,(AUTFLG)
        AND A
```

```
        JR Z,L6DA7      ; Z wenn nicht im AUTO- Modus
        LD H,01H        ; Spalte 1
L6DA7:
        PUSH HL
        CALL CKERCS     ; Cursor loeschen
        POP HL
        LD  DE,BUF
        LD B,0FEH       ; 254 Zeichen
        DEC L
L6DB2:
        INC L
L6DB3:
        PUSH DE
        PUSH BC
        CALL GETCOD     ; Zeichen vom Bildschirm holen
        POP BC
        POP DE
        LD (DE),A       ; in den Puffer
        INC DE
        DEC B
        JR Z,L6DCF      ; Z wenn 254 Zeichen geholt
        INC H           ; naechste Spalte
        LD  A,(LINLEN)
        CP H
        JR NC,L6DB3     ; NC wenn Zeilenlaenge nicht erreicht
        PUSH DE
        CALL GETTRM     ; ist hier das Zeilenende
        POP DE
        LD H,01H        ; Spalte 1
        JR Z,L6DB2      ; nein naechste Zeile
L6DCF:
        DEC DE
        LD  A,(DE)
        CP ' '
        JR Z,L6DCF      ; nachlaufende Blanks entfernen
        PUSH HL
        PUSH DE
        CALL CXDPCS     ; Cursor anzeigen
        POP DE
        POP HL
        INC DE
        XOR A
        LD (DE),A       ; Eingabezeilenende signalisieren
L6DDF:
        LD  A,0DH       ; CR
        AND A           ; NC NZ fuer ENTER Taste
L6DE2:
        PUSH AF
        CALL TERMIN     ; neues Zeilenende
        CALL POSIT      ; Cursor positionieren
        LD  A,0AH       ; LF
        RST 18H         ; ausgeben
        XOR A
        LD (INSFLG),A   ; INSERT loeschen
        POP AF
        SCF
        POP HL          ; und Pufferadresse
        RET
L6DF4:
        INC L
L6DF5:                  ; ETX   ^C
```



```
        CALL GETTRM     ; ist hier Zeilenende
        JR Z,L6DF4      ; Z also naechste Zeile
        CALL L6D6C      ; Cursor zeigen
        XOR A
        LD (BUF),A      ; Puffer leer
        LD H,01H
        PUSH HL
        CALL GICINI     ; Tongenerator initialisieren
        CALL CKSTTP     ; STOP Trap pruefen
        POP HL
        JR C,L6DDF      ; C ON STOP nicht definiert
        XOR A           ; Z NC als signal
        JR L6DE2
L6E10:                  ; DC2   ^R      INS
        LD  HL,INSFLG
        LD A,(HL)
        XOR 0FFH        ; FLAG invertieren
        LD (HL),A       ; speichern
        JP Z,L6D6C      ; Cursor normal
        JP L6D6F        ; oder halb hoch
L6E1D:                  ; ************ Zeichen einfuegen
        CALL CKERCS     ; Cursor loeschen
        LD HL,(CSRY)    ; Position
        LD  C,00H
L6E25:
        PUSH HL
L6E26:
        PUSH BC
        CALL GETVRM     ; Zeichen holen
        POP DE
        PUSH BC
        LD C,E
        CALL PUTVRM     ; speichern
        POP BC
        LD  A,(LINLEN)
        INC H           ; naechste Spalte
        CP H
        LD A,D
        JR NC,L6E26     ; NC falls Zeilenlaenge nicht erreicht
        POP HL
        CALL GETTRM     ; Zeilenende ?
        JR Z,L6E79
        LD A,C
        AND A
        JR Z,L6E45
        CP 060H
L6E45:
        PUSH AF
        JR NZ,L6E52
        LD  A,(LINLEN)
        CP H
        JR Z,L6E52      ; falls Zeilenlaenge erreicht
        POP AF
        JP CXDPCS
L6E52:
        CALL UNTERM
        INC L
        PUSH BC
        PUSH HL
        CALL GETLEN
        CP L
```

```
        JR C,L6E63      ; NC Zeilenzahl erreicht
        CALL INSLNO
        JR L6E72
L6E63:                  ; scroll
        LD  HL,CSRY
        DEC (HL)
        JR NZ,L6E6A
        INC (HL)
L6E6A:
        LD  L,01H
        CALL DELLNO
        POP HL
        DEC L
        PUSH HL
L6E72:
        POP HL
        POP BC
        POP AF
        JP Z,CXDPCS
        DEC L
L6E79:                  ; naechste Zeile
        INC L
        LD H,01H
        JR L6E25
L6E7E:                  ; DEL          DEL
        LD  A,01CH
        RST 18H
        LD HL,(CSRY)
L6E84:                  ;BS     ^H     <=
        PUSH HL
        CALL CKERCS
        POP HL
        DEC H
        JP NZ,L6E9D
        INC H
        PUSH HL
        DEC L
        JR Z,L6E9C
        LD  A,(LINLEN)
        LD H,A
        CALL GETTRM
        JR NZ,L6E9C
        EX (SP),HL
L6E9C:
        POP HL
L6E9D:
        LD (CSRY),HL
L6EA0:
        LD  A,(LINLEN)
        CP H
        JR Z,L6EB8
        INC H
L6EA7:
        CALL GETVRM     ; Zeichen holen
        DEC H
        CALL PUTVRM     ; und eine Position frueher schreiben
        INC H
        INC H
        LD  A,(LINLEN)
        INC A
        CP H
```



```
        JR NZ,L6EA7
        DEC H
L6EB8:
        LD  C,00H
        CALL PUTVRM
        CALL GETTRM
        JP NZ,CXDPCS
        PUSH HL
        INC L
        LD H,01H
        CALL GETVRM
        EX (SP),HL
        CALL PUTVRM
        POP HL
        JR L6EA0
L6ED1:                  ; NAK   ^U
        CALL CKERCS
        CALL L6F81
        LD (CSRY),HL
        JR L6EE1
L6EDC:                  ; ENQ   ^E
        PUSH HL
        CALL CKERCS
        POP HL
L6EE1:
        CALL GETTRM
        PUSH AF
        CALL ECL
        POP AF
        JR NZ,L6EF0
        LD H,01H
        INC L
        JR L6EE1
L6EF0:
        CALL CXDPCS
        XOR A
        LD (INSFLG),A
        JP L6D6C
L6EFA:                  ; SO    ^N
        CALL CKERCS
        LD HL,(CSRY)
        DEC L
L6F01:
        INC L
        CALL GETTRM
        JR Z,L6F01
        LD  A,(LINLEN)
        LD H,A
        INC H
L6F0C:
        DEC H
        JR Z,L6F16
        CALL GETCOD
        CP ' '
        JR Z,L6F0C
L6F16:
        CALL ADVCUR
        JR L6EF0
L6F1B:                  ; ACK   ^F
        CALL CKERCS
L6F1E:
```

```
        CALL L6F44
        JR Z,L6EF0
        JR C,L6F1E
L6F25:
        CALL L6F44
        JR Z,L6EF0
        JR NC,L6F25
        JR L6EF0
L6F2E:                  ; STX   ^B
        CALL CKERCS
L6F31:
        CALL L6F54
        JR Z,L6EF0
        JR NC,L6F31
L6F38:
        CALL L6F54
        JR Z,L6EF0
        JR C,L6F38
        CALL ADVCUR
        JR L6EF0
L6F44:
        LD HL,(CSRY)
        CALL ADVCUR
        CALL GETLEN
        LD E,A
        LD  A,(LINLEN)
        LD D,A
        JR L6F5D
L6F54:
        LD HL,(CSRY)
        CALL BS
        LD  DE,0101H
L6F5D:
        LD HL,(CSRY)
        RST 20H
        RET Z
        LD  DE,L6F7D
        PUSH DE
        CALL GETCOD
        CP '0'
        CCF
        RET NC
        CP '9'+1
        RET C
        CP 'A'
        CCF
        RET NC
        CP 'Z'+1
        RET C
        CP 'a'
        CCF
        RET NC
        CP 'z'+1
L6F7D:
        LD  A,00H
        INC A
        RET
L6F81:
        DEC L
        JR Z,L6F89
        CALL GETTRM
```



```
        JR Z,L6F81
L6F89:
        INC L
        LD  A,(FSTPOS)
        CP L
        LD H,01H
        RET NZ
        LD HL,(FSTPOS)
        RET
L6F95:                  ; ************** input line from FILE
        CALL LFF4B
        LD B,0FFH       ; 255 Zeichen
        LD  HL,BUF      ; in den Puffer
L6F9D:
        CALL INDSKC     ; Zeichen lesen
        JR C,L6FB8
        LD (HL),A       ; Speichern
        CP 0DH
        JR Z,L6FB2
        CP 09H
        JR Z,L6FAF
        CP ' '          ; alle Steuerzeichen ausser
        JR C,L6F9D      ;  TAB und CR ueberlesen
L6FAF:                  ; TAB aus Datei
        INC HL
        DJNZ L6F9D
L6FB2:                  ; CR aus Datei
        XOR A
        LD (HL),A
        LD  HL,BUFMIN
        RET
L6FB8:
        INC B
        JR NZ,L6FB2     ; NZ wenn Zeichen vorhanden waren
        LD  A,(NLONLY)
        AND 080H
        LD (NLONLY),A
        CALL PRGFIN
        LD  A,(FILMOD/RUNFLG)
        AND A
        JP Z,STPRDY     ; weiter wie vorher
        CALL RUNC       ; falls Programm laden und Starten
        JP NEWSTT

NAMSCN: 6FD3            ; ****** Dateinamen holen
        CALL FRMEVL     ; Stringausdruck berechnen
NAMSC1: 6FD6
        PUSH HL
        CALL FRESTR     ; Descriptor holen
        LD A,(HL)
        OR A
        JR Z,L700C      ; Z Laenge 0
        INC HL
        LD E,(HL)
        INC HL
        LD H,(HL)
        LD L,E
        LD E,A          ; HL Adresse E Laenge
        CALL PARDEV     ; Geraet herausholen
        PUSH AF         ; speichern
        LD  BC,FILNAM   ; Dateinamen Puffer
```

```
        LD D,09H        ; maximal 9 Zeichen
        INC E
L6FEE:
        DEC E
        JR Z,L7026      ; Z wenn String zu Ende
        LD A,(HL)
        CP ' '
        JR C,L700C      ; Steuerzeichen gilt nicht
        CP '.'
        JR Z,L7012      ; Namenserweiterung kommt
        LD (BC),A       ; Zeichen speichern
        INC BC
        INC HL
        DEC D
        JR NZ,L6FEE     ; 9 Zeichen noch nicht erreicht
L7000:
        POP AF
        PUSH AF
        LD D,A          ; D= Geraetecode
        LD  A,(FILNAM)
        INC A
        JR Z,L700C      ; Name der Laenge 0
        POP AF
        POP HL          ; Dateiname in FILNAM
        RET             ;  Geraetecode in A und D
L700C:
        JP DERBFN       ; BAD FILENAME
L700F:
        INC HL
        JR L6FEE
L7012:                  ; Namenserweiterung
        LD A,D
        CP 09H
        JP Z,L700C      ; Z falls erstes Zeichen . war
        CP 03H
        JP C,L700C      ; falls nach 7 oder mehr Zeichen . war
        JR Z,L700F      ; genau nach 6 Zeichen
        LD  A,' '       ; mit Blanks
        LD (BC),A       ; bis auf
        INC BC          ; 6 Zeichen auffuellen
        DEC D
        JR L7012
L7026:                  ; String kuerzer als 9 Zeichen
        LD  A,' '       ; mit Blanks
        LD (BC),A       ; bis
        INC BC          ; auf 9 Zeichen fuellen
        DEC D
        JR NZ,L7026
        JR L7000
SCNBLK: 702F
        LD A,(HL)
        INC HL
        DEC E
        RET
```



```
GETFLP: 7033            ; **** zu der Dateinummer in FACU
        CALL CONINT     ;       die Pufferadresse holen
GETPTR: 7036            ; **** A Dateinummer
        LD L,A
        LD  A,(MAXFILES)
        CP L
        JP C,DERIFN     ; Dateinummer > Maxfiles
        LD H,00H
        ADD HL,HL       ; Dateinummer*2
        EX DE,HL
        LD HL,(FILTAB)
        ADD HL,DE       ; + Tabelle
        LD A,(HL)
        INC HL
        LD H,(HL)
        LD L,A          ; Adresse Dateipuffer
        LD  A,(NLONLY)
        INC A
        RET Z           ; ret falls FF in NLONLY
        LD A,(HL)       ; erstes Byte aus Dateipuffer
        OR A
        RET Z           ; ret Z falls nicht offen
        PUSH HL
        LD  DE,04H
        ADD HL,DE       ; Byte 4
        LD A,(HL)       ; holen
        CP 09H
        JR NC,L7062H    ; JP falls Geraet 9..FF
        CALL LFEDF      ; Geraet 0..8   ( Floppy)
        JP DERIER
L7062:
        POP HL
        LD A,(HL)
        OR A            ; ret mit NZ    Geraet in A
        SCF             ; C             Puffer in HL
        RET
FILSCN: 7067            ;  Dateinummer aus Programm verarbeiten
        DEC HL
        RST 10H
        CP '#'
        CALL Z,CHRGTR   ; falls # naechstes Zeichen
        CALL GETBYT     ; als Zahl holen
        EX (SP),HL      ; Programm auf den Stack
        PUSH HL         ; ret Adresse hinterher
SETFIL: 7073
        CALL GETPTR     ; Dateipuffer holen
        JP Z,DERFND     ; Z nicht geoeffnet
        LD (PTRFIL),HL  ; Pufferadresse speichern
        CALL LFEE8
        RET
OPEN:   7080            ; ********************** OPEN
        LD  BC,FINPRT
        PUSH BC
        CALL NAMSCN     ; Dateinamen pruefen
        LD A,(HL)
        CP 082H         ; FOR token
        LD  E,04H       ; random
        JR NZ,L70AD     ; NZ wenn kein FOR
        RST 10H
        CP 085H         ; INPUT token
        LD  E,01H       ; input
```

```
        JR Z,L70AC
        CP 09CH         ; OUT token
        JR Z,L70A5
        RST 8H          ; sonst darf nur APP END kommen
        DEFB 'A'
        RST 8H
        DEFB 'P'
        RST 8H
        DEFB 'P'
        RST 8H
        DEFB 081H       ; ENDTOKEN
        LD  E,08H       ; append
        JR L70AD
L70A5:
        RST 10H
        RST 8H
        DEFB 0B3H       ; PUT token
        LD  E,02H       ; output
        JR L70AD
L70AC:
        RST 10H
L70AD:                  ; in E ist der Modus  1 2 4 oder 8
        RST 8H
        DEFB 'A'
        RST 8H
        DEFB 'S'
        PUSH DE         ; Modus speichern
        LD A,(HL)
        CP '#'
        CALL Z,CHRGTR   ; # kann auch entfallen
        CALL GETBYT     ; Nummer holen
        OR A
        JP Z,DERIFN
        CALL LFEDO
        DEFB 01EH       ;*      LD  E,0D5H
NULOPN: 70C3
        PUSH DE         ;*
        DEC HL
        LD E,A          ; A Dateinummer
        RST 10H
        JP NZ,SNERR     ; nach AS #. muss Schluss sein
        EX (SP),HL      ; Programm Pointer auf den Stack
        LD A,E          ; Geraet nach L
        PUSH AF         ; Dateinummer
        PUSH HL         ; Geraet
        CALL GETPTR     ; Puffer holen
        JP NZ,DERFAD    ; NZ falls offen
        POP DE          ; Geraet
        LD A,D
        CP 09H
        CALL LFEEB
        JP C,DERIER     ; C falls Geraet 0..8
        PUSH HL
        LD  BC,04H
        ADD HL,BC
        LD (HL),D       ; Geraet speichern
        LD  A,0         ; Funktion 0
        POP HL
        JP GENDSP
CLSFIL: 70EA            ; **** Datei schliessen
        PUSH HL
```



```
        OR A            ; Dateinummer in A
        JR NZ,L70F6     ; NZ falls Dateinummer > 0
        LD  A,(NLONLY)
        AND 01H
        JP NZ,L7472
L70F6:                  ; Schliessen
        CALL GETPTR     ; Puffer holen
        JR Z,L710E      ; Z ist schon zu
        LD (PTRFIL),HL
        PUSH HL
        LD  A,2         ; Funktion 2
        JP C,GENDSP     ; C geraet > 8
        CALL LFEA9
        JP DERIER
NOCLSB: 710A            ; **** CLOSE vom Request handler
        CALL CLRBUF     ; Puffer auf 0 setzen
        POP HL          ; A hat 0
L710E:
        PUSH HL
        LD  DE,07H
        ADD HL,DE
        LD (HL),A       ; Datei Flag setzen #+7
        LD H,A
        LD L,A
        LD (PTRFIL),HL  ; auf 0 setzen
        POP HL
        ADD A,(HL)      ; Modus holen
        LD (HL),00H     ; auf 0 setzen
        POP HL
        RET
LRUN:   711F            ; ******************* RUN "..." C
        SCF
        DEFB 011H       ;*      LD  DE,0AFF6H
LOAD:   7121            ; ********************** LOAD   NC NZ
        DEFB 0F6H       ;*                 OR 0AFH
MERGE:  7122            ; ********************** MERGE  NC Z
        XOR A           ;*              ;*
        PUSH AF         ; Speichern der Instruktion
        CALL NAMSCN     ; Dateiname
        CALL LFEC7
        POP AF
        PUSH AF
        JR Z,L713A      ; Z bei MERGE und RUN
        LD A,(HL)
        SUB ','
        OR A
        JR NZ,L713A     ; NZ kein ,
        RST 10H
        RST 8H
        DEFB 'R'        ; sonst nur R
        POP AF          ; Flags holen
        SCF             ; und RUN setzen
        PUSH AF
L713A:
        PUSH AF
        XOR A           ; Puffer #0
        LD  E,01H       ; Modus 1
        CALL NULOPN     ; Datei oeffnen
        LD HL,(PTRFIL)
        LD  BC,07H
        ADD HL,BC       ; #+7
```

```
        POP AF          ; Flags holen
        SBC A,A         ; FF falls C   0 sonst
        AND 080H
        OR 01H
        LD (NLONLY),A   ; 81  oder  1
        POP AF
        PUSH AF         ; Flags
        SBC A,A
        LD (FILMOD/RUNFLG),A    ; FF falls RUN   0 sonst
        LD A,(HL)       ; #+7
        OR A
        JP M,L71D5
        POP AF
        CALL NZ,SCRTCH  ; Loeschen bei LOAD
        XOR A
        CALL SETFIL     ; Laden
        JP MAIN         ; weiter
SAVE:/  7167            ; ************************** SAVE
        CALL NAMSCN     ; Dateiname
        CALL LFEB5      ;
        DEC HL
        RST 10H
        LD  E,080H      ; Voreinstellung Dateimodus
        SCF             ; C Programmdatei
        JR Z,L717B      ; Z kein ,A / Kinder Bit entfernen
        RST 8H
        DEFB ','
        RST 8H          ; es muss ,A sein
        DEFB 'A'        ; ,S nur bei Floppy oder CSAVE
        OR A            ; NC
        LD  E,02H       ; neuer Modus
L717B:
        PUSH AF
        LD A,D
        CP 09H
        JR C,L718B      ; Geraet 0..8
        LD A,E
        AND 080H
        JR Z,L718B      ; Z wenn keine Floppy und Dateiart ASCII
        LD  E,02H       ; sonst ASCII einschalten
        POP AF
        XOR A           ; C loeschen
        PUSH AF
L718B:
        XOR A           ; dateinummer 0
        CALL NULOPN     ; Datei eröffne
        POP AF
        JR C,L7197      ; C falls Programmdatei
        DEC HL
        RST 10H
        JP LIST         ; falls ASCII dann LISTen
L7197:
        PUSH HL
        CALL SCCPTR
        LD HL,(VARTAB)
        LD (SAVEND),HL
        LD HL,(TXTTAB)
        PUSH HL
        CALL LFF2D
        JP DERBFN       ; error weil nach CAS nur ASCII
SPSVEX:/ 71AB
```



```
        POP HL
        XOR A
        JP CLSFIL
L71B0:                  ; **** speichern von Bytes in Datei
        LD DE,(SAVEND)  ; SAVEND ist Bremse  HL Anfangsadresse
L71B4:
        RST 20H
        RET Z
        LD A,(HL)
        INC HL
        CALL FILOU1
        JR L71B4


L71BD:                  ; ***** Lesen von Bytes von einer Datei
        LD DE,(SAVEND)  ; SAVEND Bremse  HL Anfangsadresse
L71C1:                  ; falls C mit Pruefung auf Speicherplatz
        PUSH AF
        JR C,L71C9
        PUSH DE
        CALL CHKTOP     ; Adresse gegen STKTOP pruefen
        POP DE
L71C9:
        CALL INDSKC     ; Byte holen
        JR C,L71D3      ; C Dateiende
        LD (HL),A
        INC HL
        POP AF
        JR L71C1
L71D3:
        POP AF
        RET

L71D5:                  ; ****
        POP AF
        JP Z,DERSOO     ; 7618
        CALL SCRTCH     ; Programm loeschen
        CALL CLSALL     ; Dateien schliessen
        XOR A
        CALL GETPTR     ; Puffer 0 holen
        LD (HL),080H    ; Modus Binaer setzen
        LD (PTRFIL),HL
        CALL LFF33
        JP DERBFN


L71EE:
        CALL LINKER
        INC HL
        LD (VARTAB),HL
        CALL RUNC
        XOR A
        LD (NLONLY),A
        CALL CLSFIL
        LD  A,(FILMOD/RUNFLG)
        OR A
        JP NZ,NEWSTT
        JP READY


CHKTOP:/ 7209           ; HL gegen STKTOP pruefen
```

```
        EX DE,HL
        LD HL,(STKTOP)
        EX DE,HL
        RST 20H
        RET C
NOROOM: 7210            ; kein Platz mehr vorhanden
        CALL SCRTCH     ; loeschen des Programms
        XOR A
        LD (NLONLY),A
        JP OMERR        ; Fehlermeldung
GETDEV: 721A            ; ***** Geraet aus Dateipuffer in HL
        PUSH HL         ;       nach A holen
        PUSH DE
        LD HL,(PTRFIL)
        LD  DE,04H
        ADD HL,DE
        LD A,(HL)
        POP DE
        POP HL
        RET
RSET:   7227            ; ********************************* RSET
        DEFB 0F6H       ;*      OR 037H
LSET:   7228            ; ********************************* LSET
        SCF             ;*
        PUSH AF
        CALL PTRGET     ; Variable holen
        CALL CHKSTR/FRCSTR      ; muss String sein
        PUSH DE
        RST 8H
        DEFB 0F1H       ; = Token
        CALL FRMEVL     ; rechte Seite berechnen
        POP BC
        EX (SP),HL
        PUSH HL
        PUSH BC
        CALL FRESTR     ; Descriptor holen
        LD B,(HL)
        EX (SP),HL
        LD A,(HL)       ; Laenge
        LD C,A
        PUSH BC
        PUSH HL
        PUSH AF
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)       ; Adresse in DE
        OR A
        JP Z,L72A3      ; Z wenn Laenge 0 String loeschen
        LD HL,(TXTTAB)
        RST 20H
        JR NC,L727E
        LD HL,(VARTAB)
        RST 20H
        JR C,L727E      ; C wenn Programmtext
        LD E,C
        LD D,00H
        LD HL,(STKTOP)
        ADD HL,DE
        EX DE,HL
        LD HL,(FRETOP)
```



```
        RST 20H
        JP C,L72B6
L7267:
        POP AF
        LD A,C
        CALL GETSPA
        POP HL
        POP BC
        EX (SP),HL
        PUSH DE
        PUSH BC
        CALL FRESTR
        POP BC
        POP DE
        EX (SP),HL
        PUSH BC
        PUSH HL
        INC HL
        LD (HL),E
        INC HL
        LD (HL),D
        PUSH AF
L727E:
        POP AF
        POP HL
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)       ; String Adresse
        POP BC          ; Laengen
        POP HL          ; Descriptor
        PUSH DE
        INC HL
        LD E,(HL)
        INC HL
        LD D,(HL)
        EX DE,HL
        POP DE
        LD A,C
        SUB B
        JR NC,L7292
        LD B,C
L7292:
        LD A,C          ; Laenge
        SUB B           ; - vorhandene Zeichen
        LD C,A
        POP AF
        CALL NC,L72AD   ; links mit Blanks fuellen
        INC B
L729A:                  ; dann String kopieren
        DEC B
        JR Z,L72A8
        LD A,(HL)
        LD (DE),A
        INC HL
        INC DE
        JR L729A
L72A3:
        POP BC
        POP BC
        POP BC
```

```
        POP BC
        POP BC
L72A8:
        CALL C,L72AD
        POP HL
        RET
L72AD:                  ; Bereich bis zum Ende mit Blanks fuellen
        LD  A,' '
        INC C
L72B0:
        DEC C
        RET Z
        LD (DE),A
        INC DE
        JR L72B0
L72B6:
        POP AF
        POP HL
        POP BC
        EX (SP),HL
        EX DE,HL
        JR NZ,L72C6
        PUSH BC
        LD A,B
        CALL STRINI
        CALL PUTNEW
        POP BC
L72C6:
        EX (SP),HL
        PUSH BC
        PUSH HL
        PUSH AF
        JP L7267
FIELD:  72CD            ; ************************* FIELD
        DEC HL
        RST 10H
        CP '#'
        CALL Z,CHRGTR
        CALL GETBYT     ; Dateinummer
        PUSH HL
        CALL GETPTR     ; Dateipuffer
        CALL GETBF1
        POP DE
        XOR A
        LD B,A
L72E1:
        LD C,A
        PUSH HL
        ADD HL,BC
        EX DE,HL        ; DE aktueller Puffer offset
        LD A,(HL)
        CP ','
        JP NZ,L7482
        PUSH DE
        PUSH BC
        CALL GTBYTC     ; Anzahl Bytes
        PUSH AF         ; speichern
        RST 8H
        DEFB 'A'
        RST 8H
        DEFB 'S'
```



```
        CALL PTRGET     ; Variable
        CALL CHKSTR/FRCSTR      ; Stringvariable
        POP AF
        POP BC
        EX (SP),HL      ; Programm pointer auf den Stack
        INC B           ; Pufferoffset in HL
        ADD A,C         ; aktuelle+bisherige Bytes
        JR C,L7303      ; C falls >255
        DEC B
L7303:
        JP NZ,DERFOV
        EX DE,HL
        SUB C
        LD (HL),A       ; Laenge des Strings
        ADD A,C
        INC HL
        LD (HL),E
        INC HL
        LD (HL),D       ; Adresse
        POP DE
        POP HL
        JR L72E1

MKI$:   7312            ; ************************* MKI$
        LD  A,02H
        DEFB 1          ;*      LD  BC,043EH
MKS$:   7315            ; ************************* MKS$
        LD A,4          ;*
        DEFB 1          ;*      LD  BC,083EH
MKD$:   7318            ; ************************* MKD$
        LD A,8          ;*
        PUSH AF         ; Laenge der Variablen
        CALL OOCNVF
        POP AF
        CALL STRINI     ; String erzeugen
        LD HL,(DSCPTR)
        CALL VMOVMF     ; Uebertragen
        JP FINBCK

CVI:    732B            ; ************************* CVI
        LD  A,01H
        DEFB 1          ;*      LD  BC,033EH
CVS:    732E            ; ************************* CVS
        LD A,3          ;*
        DEFB 1          ;*      LD  BC,073EH
CVD:    7331            ; ************************* CVD
        LD A,7          ;*
        PUSH AF
        CALL FRESTR     ; Descriptor holen
        POP AF
        CP (HL)
        JP NC,FCERR     ; NC Stringlaenge reicht nicht aus
        INC A
        INC HL
        LD C,(HL)
        INC HL
        LD H,(HL)
        LD L,C          ; HL Stringadresse
        LD (VALTYP),A   ; Anzahl der Bytes -1
        JP VMOVFM
```

```
L7348:                  ; ***** Schliessen aller Dateien
        JR NZ,L7363     ; NZ wenn nicht End of Statement
        PUSH HL
L734B:
        PUSH BC         ; Adresse fuer Schliessen
        PUSH AF
        LD  DE,L7354
        PUSH DE
        PUSH BC
        OR A            ; A Dateinummer
        RET             ; ret zum Schliessen
L7354:          ; Hier geht es nach dem Schliessen weiter
        POP AF  ; Dateinummer
        POP BC  ; Close Adresse
        DEC A
        JP P,L734B      ; P wenn noch weitere Dateien
        POP HL
        RET


L735C:
        POP BC
        POP HL
        LD A,(HL)
        CP ','
        RET NZ
        RST 10H
L7363:
        PUSH BC         ; CLOSE Sprungadresse
        LD A,(HL)
        CP '#'
        CALL Z,CHRGTR
        CALL GETBYT     ; Dateinummer holen
        EX (SP),HL
        PUSH HL
        LD  DE,L735C    ; Return-Adresse fuer weitere Dateinummern
        PUSH DE
        SCF
        JP (HL)

CLOSE:  7375            ; ************************* CLOSE
        LD  BC,CLSFIL   ; Adresse fuer Datei schliessen
        LD  A,(MAXFILES); Anzahl der Dateien
        JR L7348
CLSALL: 737D
        LD  A,(NLONLY)
        OR A
        RET M           ; M falls NLONLY
        LD  BC,CLSFIL   ; Adresse fuer Datei schliessen
        XOR A
        LD  A,(MAXFILES)
        JR L7348

CLSCLR: 7388
        XOR A
        LD B,A
L738D:
        LD A,B
        CALL GETPTR     ; Dateipuffer holen
        LD (HL),00H     ; Modus 0 setzen
        LD  A,(MAXFILES)
```



```
        INC B
        SUB B
        JR NC,L738D
        XOR A
        LD (NLONLY),A
        CALL SCRTCH     ; Programm loeschen
        LD HL,(TXTTAB)
        DEC HL
        LD (HL),00H
        CALL LFEF1
        JP READYR

LFILES: 73AD            ; ************************* LFILES
        LD  A,01H
        LD (ODEVLINK),A
FILES:  73B2            ; ************************* FILES
        CALL LFEB8
        JP FCERR        ; illegal function call
DPUT:   73B8            ; ************************* PUT
        DEFB 0F6H       ;*      OR 0AFH
DGET:   73B9            ; ************************* GET
        XOR A           ;*
        PUSH AF
        CALL FILSCN
        LD  A,04H
        JP C,GENDSP
        CALL LFE7F
        JP DERBFN


FILOUT: 73C9
        POP AF
FILOU1: 73CA            ; **** Ausgabe des Zeichens in A
        PUSH HL         ; in die aktuelle Datei (PTRFIL)
        PUSH DE
        PUSH BC
        PUSH AF
        LD HL,(PTRFIL)
        LD  A,06H
        CALL L73DC
        CALL LFF12
        JP DERBFN
L73DC:
        PUSH AF                 ; Funktion
        PUSH DE
        EX DE,HL
        LD  HL,04H
        ADD HL,DE
        LD A,(HL)               ; Geraet
        EX DE,HL
        POP DE
        CP 09H
        JP C,L7482              ; C wenn Floppy
        POP AF
        EX (SP),HL
        POP HL
        JP GENDSP
INDSKC: 73F1                    ; Einlesen eines Bytes von Datei
        PUSH BC
        PUSH HL
        PUSH DE
```

```
        LD HL,(PTRFIL)          ; 7497
        LD  A,08H               ; Funktion
        CALL L73DC
        CALL LFE82
        JP DERBFN
INDSKE: 7402
        POP DE
        POP HL
        POP BC
        RET
FIXINP: 7406        ; ************************* INPUT$()
        RST 10H
        RST 8H
        DEFB '$'
        RST 8H
        DEFB '('
        PUSH HL
        LD HL,(PTRFIL)
        PUSH HL         ; Dateipointer sichern
        LD  HL,0
        LD (PTRFIL),HL
        POP HL
        EX (SP),HL
        CALL GETBYT     ; (Datei)nummer holen
        PUSH DE
        LD A,(HL)
        CP ','
        JR NZ,L7432
        RST 10H         ; es musste Dateinummer sein
        CALL FILSCN
        CP 01H          ; INPUT Modus
        JP Z,L742F
        CP 04H          ; oder RANDOM
        JP NZ,DERRPE
L742F:
        POP HL
        XOR A
        LD A,(HL)
L7432:
        PUSH AF
        RST 8H
        DEFB ')'
        POP AF
        EX (SP),HL
        PUSH AF
        LD A,L
        OR A
        JP Z,FCERR      ; keine Zeichen koennen wir nicht lesen
        PUSH HL
        CALL STRINI     ; String erzeugen
        EX DE,HL
        POP BC
L7443:
        POP AF
        PUSH AF
        JR Z,L7461
        CALL CHGET/TRYIN        ; Zeichen einlesen
        PUSH AF
        CALL CKCNTC     ; CTRL-C pruefen
        POP AF
L744F:
```



```
        LD (HL),A
        INC HL
        DEC C
        JR NZ,L7443
        POP AF
        POP BC
        POP HL
        CALL LFE97
        LD (PTRFIL),HL
        PUSH BC
        JP PUTNEW
L7461:                  ; Anzahl Zeichen von Datei lesen
        CALL INDSKC
        JP C,DERRPE     ; C wenn Datei zu Ende
        JR L744F
CLRBUF: 7469            ; ********** Dateipuffer loeschen
        CALL GETBUF
        PUSH HL
        LD B,0          ; 256
        CALL DOCLR
L7472:
        POP HL
        RET
DOCLR:  7474            ; ******** B Bytes ab HL auf 0 setzen
        XOR A
L7475:
        LD (HL),A
        INC HL
        DJNZ L7475
        RET
GETBUF: 747A            ; aktuellen Dateipuffer holen
        LD HL,(PTRFIL)
GETBF1: 747D            ; zu HL den Dateipuffer holen
        LD  DE,09H      ;#+9
        ADD HL,DE
        RET
L7482:
        POP AF
        RET
LOC:    7486            ; ****************************** LOC
        CALL LFEB2
        CALL GETFLP
        JP Z,DERFND
        POP BC
        LD  A,0AH
        JP C,GENDSP
        PUSH BC
        CALL LFE9A
        JP DERBFN
LOF:    749A            ; ****************************** LOF
        CALL LFEB2
        CALL GETFLP
        JP Z,DERFND
        LD  A,0CH
        POP BC
        JP C,GENDSP
        PUSH BC
        CALL LFEBB
        JP DERBFN
EOF:    74B0            ; ****************************** EOF
        CALL LFEB2
```

```
        CALL GETFLP
        JP Z,DERFND
        POP BC
        LD  A,0EH
        JP C,GENDSP
        PUSH BC
        CALL LFECA
        JP DERBFN
FPOS:/  73C6            ; ***************************** FPOS
        CALL LFEB2
        CALL GETFLP
        POP BC
        LD  A,010H
        JP C,GENDSP
        PUSH BC
        CALL LFE8B
        JP DERBFN
DIROG:/ 74D9            ; ***
        CALL ISFLIO
        JP Z,GONE
        XOR A
        CALL CLSFIL
        JP DERFDR
FILINP:/74E6            ; ********************** INPUT #
        LD  C,01H
FILGET:/74E8
        CP '#'
        RET NZ
        PUSH BC
        CALL GTBYTC     ; dateinummer
        RST 8H
        DEFB ','
        LD A,E
        PUSH HL
        CALL SETFIL
        LD A,(HL)
        POP HL
        POP BC
        CP C            ; CP 2
        JR Z,L750A      ; Z wenn OUTPUT
        CP 04H
        JR Z,L750A      ; Z wenn RANDOM
        CP 08H
        JR NZ,L7507     ; NZ wenn nicht APPEND
        LD A,C
        CP 02H
L7507:
        JP NZ,DERIFN
L750A:
        LD A,(HL)
        RET
PRGFIN:/75ØC
        LD  BC,GTMPRT
        PUSH BC
        XOR A
        JP CLSFIL
FILIND:/7514
        RST 30H
        LD  BC,DOASIG
        LD  DE,02C20H
        JR NZ,L7534
```



```
        LD E,D
        JR L7534
DLINE:/ 752Ø
        LD  BC,FINPRT
        PUSH BC
        CALL FILINP
        CALL PTRGET
        CALL CHKSTR/FRCSTR
        PUSH DE
        LD  BC,LETCON
        XOR A
        LD D,A
        LD E,A
L7534:
        PUSH AF
        PUSH BC
        PUSH HL
L7537:
        CALL INDSKC
        JP C,DERRPE
        CP ' '
        JR NZ,L7545
        INC D
        DEC D
        JR NZ,L7537
L7545:
        CP '"'
        JR NZ,L7557
        LD A,E
        CP ','
        LD  A,'"'
        JR NZ,L7557
        LD D,A
        LD E,A
        CALL INDSKC
        JR C,L759E
L7557:
        LD  HL,BUF
        LD B,0FFH
L755C:
        LD C,A
        LD A,D
        CP '"'
        LD A,C
        JR Z,L758D
        CP 0DH
        PUSH HL
        JR Z,L75B8
        POP HL
        CP 0AH
        JR NZ,L758D
L756D:
        LD C,A
        LD A,E
        CP ','
        LD A,C
        CALL NZ,L75F0
        CALL INDSKC
        JR C,L759E
        CP 0AH
        JR Z,L756D
```

```
        CP ODH
        JR NZ,L758D
        LD A,E
        CP ' '
        JR Z,L7599
        CP ','
        LD  A,ODH
        JR Z,L7599
L758D:
        OR A
        JR Z,L7599
        CP D
        JR Z,L759E
        CP E
        JR Z,L759E
        CALL L75FO
L7599:
        CALL INDSKC
        JR NC,L755C
L759E:
        PUSH HL
        CP '"'
        JR Z,L75A7
        CP ','
        JR NZ,NOSKCR
L75A7:
        CALL INDSKC
        JR C,NOSKCR
        CP ' '
        JR Z,L75A7
        CP ','
        JR Z,NOSKCR
        CP ODH
        JR NZ,L75C1
L75B8:
        CALL INDSKC
        JR C,NOSKCR
        CP OAH
        JR Z,NOSKCR
L75C1:
        LD HL,(PTRFIL)
        LD C,A
        LD  A,012H
        CALL L73DC
        CALL LFE9D
        JP DERBFN
NOSKCR: 7500
        POP HL
L75D1:
        LD (HL),00H
        LD  HL,BUFMIN
        LD A,E
        SUB 020H
        JR Z,L75E2
        LD B,00H
        CALL STRLT3
        POP HL
        RET
L75E2:
        RST 30H
        PUSH AF
```



```
        RST 10H
        POP AF
        PUSH AF
        CALL C,FINDBL
        POP AF
        CALL NC,FINDBL
        POP HL
        RET
L75FO:
        OR A
        RET Z
        LD (HL),A
        INC HL
        DEC B
        RET NZ
        POP AF
        JP L75D1
DERBFN: 75FA
        LD  E,038H
        DEFB 1          ;*      LD  BC,0361EH
DERFAD: 75FD
        LD E,036H       ;*
        DEFB 1          ;*      LD  BC,0391EH
DERFDR: 7600
        LD E,039H       ;*
        DEFB 1          ;*      LD  BC,0351EH
DERFNF: 7603
        LD E,035H       ;*
        DEFB 1          ;*      LD  BC,03C1EH
DERFND: 7606
        LD E,03CH       ;*
        DEFB 1          ;*      LD  BC,0321EH
DERFOV: 7609
        LD E,032H       ;*
        DEFB 1          ;*      LD  BC,0341EH
DERIFN: 760C
        LD E,034H       ;*
        DEFB 1          ;*      LD  BC,0331EH
DERIER: 760F
        LD E,033H       ;*
        DEFB 1          ;*      LD  BC,0371EH
DERRPE: 7612
        LD E,037H       ;*
        DEFB 1          ;*      LD  BC,03A1EH
DERSAP: 7615
        LD E,03AH       ;*
        DEFB 1          ;*      LD  BC,03B1EH
DERSOO: 7618
        LD E,03BH       ;*
        XOR A
        LD (NLONLY),A
        LD (FLBMEM),A
        JP ERROR
BSAVE:  7624            ; ******************************** BSAVE
        CALL NAMSCN     ; Dateiname
        PUSH DE         ; Geraet
        RST 8H
        DEFB ','
        CALL L7701      ; Adresse einlesen
        EX DE,HL
        LD (SAVENT),HL  ; Anfangsadresse
```

```
        EX DE,HL        ; als Voreinstellung Startadresse
        PUSH DE         ; auf dem Stack speichern
        RST 8H
        DEFB ','
        CALL L7701      ; Adresse einlesen
        EX DE,HL
        LD (SAVEND),HL  ; Endadresse
        EX DE,HL
        DEC HL
        RST 10H
        JR Z,L764B      ; Z wenn End of Statement
        RST 8H
        DEFB ','
        CALL L7701      ; Adresse einlesen
        EX DE,HL
        LD (SAVENT),HL  ; Startadresse
        EX DE,HL
L764B:
        POP BC
        POP DE
        PUSH HL
        PUSH BC
        LD A,D
        CP OFEH         ; Geraetebyte fuer Kassettenrecorder
        JP Z,CBSAVE
        LD  A,OFFH
        LD (FLBMEM),A   ; FE99 = 0, wenn BASIC-Prg. geladen
        INC A
        LD  E,02H
        CALL NULOPN     ; oeffnen der Datei
        POP HL
        PUSH HL         ; Anfangsadresse speichern
        CALL L767C
        LD HL,(SAVEND)  ; Endadresse
        CALL L767C
        LD HL,(SAVENT)  ; Startadresse
        CALL L767C
        POP HL
        CALL L71BO      ; Programm abspeichern
        XOR A
        LD (FLBMEM),A
        JP SPSVEX       ; Ende-Behandlung
L767C:                  ; **** HL im Bytereversed Format in Datei
        LD A,L          ; speichern
        CALL FILOU1     ; 73CA
        LD A,H
        JP FILOU1       ; 73CA
BLOAD:  CALL NAMSCN     ; ****************************** BLOAD
        PUSH DE         ; 6FD3 / Dateinamen berechnen
        XOR A           ; Voreinstellungen
        LD (RUNBNF),A   ; Nicht ablaufen lassen! / FE57
        DEC HL
        RST 10H
        LD  BC,00H      ; Offset=0
        JR Z,L76A6
        RST 8H
        DEFB ','
        CP 'R'
        JR NZ,L76A1
        LD (RUNBNF),A   ; <>0 d.h. laufen lassen! / FE57
```



```
        RST 10H
        JR Z,L76A6
        RST 8H
        DEFB ','
L76A1:
        CALL L7701      ; offset holen / String einlesen
        LD B,D
        LD C,E
L76A6:
        POP DE
        PUSH HL
        PUSH BC
        LD A,D
        CP OFEH         ; CASSETTE ?
        JP Z,CBLOAD     ; JA : CAS-Routine
        XOR A
        LD  E,01H       ; Datei zum Lesen oeffnen
        CALL NULOPN     ; Nummer -> A / Code -> E / Ordnernummer -> FILNAM
        LD HL,(PTRFIL)  ; Filename / F997
        LD  DE,07H
        ADD HL,DE
        LD A,(HL)
        AND 01H
        JP Z,FCERR      ; Falls keine Binaerdatei /A5
        POP BC          ; offset
        CALL INDSKC     ; 102 / Byte einlesen
        LD L,A
        CALL INDSKC     ; 102
        LD H,A          ; Anfangsadresse
        ADD HL,BC       ; +offset
        PUSH HL         ; speichern
        CALL INDSKC     ; 102
        LD L,A
        CALL INDSKC     ; 102
        LD H,A          ; Endadresse
        ADD HL,BC       ; +offset
        LD (SAVEND),HL  ; speichern / F9B4
        CALL INDSKC     ; 102
        LD L,A
        CALL INDSKC     ; 102
        LD H,A          ; Startadresse
        ADD HL,BC       ; +offset
        LD (SAVENT),HL  ; Speichern / FE58
        POP HL
        SCF
        CALL L71BD      ; Bereich einlesen
CHKBRN: LD  A,(RUNBNF)  ; 76EA
        OR A
        JR Z,L76FC      ; Z nicht laufen lassen!
        XOR A
        CALL CLSFIL     ; Datei schliessen
        LD  HL,L7472    ; Return- Adresse fuer Binaerprogramm
        PUSH HL         ;   POP HL Programmpointer und RET
        LD HL,(SAVENT)  ; FE68
BLOAD ENDE: JP (HL)     ; zur Startadresse springen
L76FC:
        POP HL
        XOR A
        JP CLSFIL
L7701:                  ; **** Wort einlesen aus Programmstring
```

```
        CALL FRMEVL     ; nach DE
        PUSH HL
        CALL FRQINT
        POP DE
        EX DE,HL
        RET
PARDEV: 7708            ; ******** Geraet aus DAteinamen holen
        CALL LFEA3
        PUSH HL
        LD D,E
        CALL SCNBLK     ; Zeichen holen
        JR Z,L7723      ; Z String zu ende
        CP ':'
        JR C,L772C      ; C falsches Zeichen im Dateinamen
L7719:
        CP ':'
        JR Z,L772F      ;  : gefunden
        CALL SCNBLK
        JP P,L7719      ; P wenn weitere Zeichen
                        ; kein Geraetebezeichner gefunden
L7723:
        LD E,D
        POP HL
        XOR A
        LD  A,0FEH      ; Voreinstellung Kassettenrecorder
        CALL LFEC1
        RET
L772C:
        JP DERBFN
L772F:
        LD A,D
        SUB E
        DEC A
        CP 02H
        JR NC,L773C     ; NC wenn mehr als 2 Zeichen fuer
        CALL LFEE5      ; Geraetenamen
        JP DERBFN
L773C:
        CP 05H
        JP NC,DERBFN    ; NC wenn mehr als 4
        POP BC
        PUSH DE
        PUSH BC
        LD C,A
        LD B,A
        LD  DE,DEVTBL   ; Geraetetabelle
        EX (SP),HL
        PUSH HL
L774B:
        LD A,(HL)       ; Zeichen aus Geraetenamen
        CP 'a'
        JR C,L7756
        CP 'z'+1
        JR NC,L7756
        SUB 020H        ; als Grossbuchstabe
L7756:
        PUSH BC
        LD B,A
        LD  A,(DE)      ; Zeichen aus Tabelle
        INC HL
        INC DE
        CP B            ; Vergleichen
```



```
        POP BC
        JR NZ,L7774     ; wenn < > 0 Naechstes Geraet
        DEC C
        JR NZ,L774B     ; naechstes Zeichen
                        ; Geraet gefunden
L7762:
        LD  A,(DE)      ; Geraete- Bezeichner holen
        OR A
        JP M,L776F      ; sind fertig fuer alles ausser Floppy
        CP '1'
        JR NZ,L7774     ; es ist nicht 1
        INC DE
        LD  A,(DE)
        JR L7774
L776F:
        POP HL
        POP HL
        POP DE
        OR A
        RET
L7774:                  ; naechstes Geraet
        OR A
        JP M,L7762
L7778:
        LD  A,(DE)
        OR A
        INC DE
        JP P,L7778      ; P Name noch nicht zu Ende
        LD C,B
        POP HL
        PUSH HL
        LD  A,(DE)
        OR A
        JR NZ,L774B     ; NZ es sind noch Geraete in der Tabelle
        JP DERBFN
DEVTBL: 7788            ; Tabelle aus Geraetenamen und Byte
        DEFB 'KYBD',0FFH
        DEFB 'CAS',0FEH
        DEFB 'MDM',0FDH
        DEFB 'LPT',0FCH
        DEFB 'CRT',0FBH
        DEFB 0

L779E:                  ; REQUEST HANDLER TABLE
        DEFW KBDDSP     ; KYBD BLOCK
        DEFW CASDSP     ; CAS BLOCK
        DEFW MDMDSP     ; MDM BLOCK
        DEFW LPTDSP     ; LPT BLOCK
        DEFW CRTDSP     ; CRT BLOCK

GENDSP: 77A8            ; request- handler
        CALL LFFOF      ; INPUT
        PUSH HL         ;       HL Dateipufferadresse
        PUSH DE         ;       A  Funktion
        PUSH AF         ;       E  Parameter
        LD  DE,04H
        ADD HL,DE       ; auf Geraet einstellen
        LD  A,0FFH
        SUB (HL)        ; KYBD=0 CAS=1 MDM=2 ...
        ADD A,A         ; *2
        LD E,A
```

```
        LD D,00H
        LD  HL,L779E    ; request- handler- table
        ADD HL,DE
        LD E,(HL)
        INC HL
        LD D,(HL)       ; Geraeteblockadresse
        POP AF
        LD L,A          ; Funktion
        LD H,00H
        ADD HL,DE
        LD E,(HL)
        INC HL
        LD D,(HL)       ; Adresse der Funktion auf dem Geraet
        EX DE,HL
        POP DE
        EX (SP),HL
        RET
KBDDSP: 77CC
        DEFW L77E0      ; Funktion 0 OPEN
        DEFW NOCLSB     ;          2 CLOSE
        DEFW DERS00     ;          4
        DEFW FCERR      ;          6 Put CHR
        DEFW L77EE      ;          8 Get CHR
        DEFW FCERR      ;          A LOC
        DEFW FCERR      ;          C LOF
        DEFW L7807      ;          E EOF
        DEFW FCERR      ;         10 FPOS
        DEFW L7811      ;         12 letztes Zeichen setzen


L77E0:                  ; ***** Keyboard- Datei oeffnen
        XOR A
        LD (KBDPRV),A   ; Zeichenpuffer leeren
        CALL L7987      ; kein ret bei APPEND
        CP 02H
        JP Z,DERBFN     ; kein OUTPUT
        JR L7833        ; Dateipuffer und Filemodus speichern
L77EE:                  ; ***** Zeichen von der Tastatur lesen
        CALL L77F4
        JP INDSKE       ; nur POPs
L77F4:
        LD  HL,KBDPRV
        CALL L796F      ; falls Zeichen in Puffer holen kein ret
        CALL CHGET/TRYIN ; neues Zeichen holen da puffer leer
        CP 01AH         ; ^Z
        SCF
        CCF             ; NC
        RET NZ          ; ret wenn ungleich Dateiende
        LD (KBDPRV),A   ; in dateipuffer schreiben
        SCF             ; C setzen
        RET
L7807:                  ; Keyboard EOF funktion
        PUSH BC
        CALL L77F4      ; Zeichen lesen
        LD  HL,KBDPRV   ; Zeichenpuffer aufsetzen
        JP L797E        ; pruefen auf EOF

L7811:
        LD  HL,KBDPRV   ; Zeichenpuffer
        JP L78E3
CRTDSP: 7817
```



```
        DEFW L782B
        DEFW NOCLSB
        DEFW DERS00
        DEFW L783A
        DEFW FCERR
        DEFW FCERR
        DEFW FCERR
        DEFW FCERR
        DEFW FCERR
        DEFW FCERR

L782B:                  ; CRT   OPEN
        CALL L7987      ; kein APPEND
        CP 01H
        JP Z,DERBFN     ; kein INPUT
L7833:
        LD (PTRFIL),HL
        LD (HL),E       ; MODUS
        POP AF
        POP HL
        RET
L783A:                  ; CRT Put CHR
        POP AF
        CALL LINPT1
        JP PBDHRT
CASDSP: 7841
        DEFW L7855
        DEFW L787F
        DEFW DERS00
        DEFW L7897
        DEFW L78A2
        DEFW FCERR
        DEFW FCERR
        DEFW L78D6
        DEFW FCERR
        DEFW L78E0

L7855:                  ; CAS OPEN
        PUSH HL
        PUSH DE
        LD  BC,06H
        ADD HL,BC
        XOR A
        LD (HL),A       ; aktuelle Pufferposition ist 0
        LD (CASPRV),A   ; Zeichenpuffer leeren
        CALL L7987      ; kein APPEND
        CP 04H
        JP Z,DERBFN     ; kein RANDOM
        CP 01H
        JR Z,L7875      ; Z OPEN FOR INPUT
        LD  A,0EAH      ; ASCII datei oeffnen
        CALL CASOPW     ;    Header mit 0EAH schreiben
L7871:
        POP DE
        POP HL
        JR L7833
L7875:                  ; CAS OPEN FOR INPUT
        LD  C,0EAH
        CALL SRCCAS     ; Datei mit Header 0EAH suchen
        CALL CTOFF      ; abschalten
        JR L7871
```

```
L787F:                  ; CAS CLOSE
        CALL L78F6      ; ret mit Z wenn INPUT modus
        JR Z,L7890      ;       Pufferposition 0
        PUSH HL
        ADD HL,BC
L7886:                  ; Rest des Puffers auf ^Z setzen
        LD (HL),01AH
        INC HL
        INC C
        JR NZ,L7886
        POP HL
        CALL L78E7      ; Schreiben des Puffers
L7890:
        XOR A
        LD (CASPRV),A   ; Zeichenpuffer loeschen
        JP NOCLSB
L7897:                  ; CAS Put CHR
        POP AF
        PUSH AF
        CALL L7904      ; Zeichen in Puffer schreiben
        CALL Z,L78E7    ; Z vollen Puffer wegschreiben
        JP POPALL

L78A2:                  ; Zeichen von CAS Lesen
        CALL L78A8
        JP INDSKE
L78A8:                  ; Zeichen lesen
        EX DE,HL
        LD  HL,CASPRV
        CALL L796F      ; Zeichen aus Puffer holen,
        EX DE,HL        ; sonst selbst holen
        CALL L7914      ; Puffer zeiger aufsetzen und erhoehen
        JR NZ,L78C9     ; NZ noch Zeichen im Puffer
        PUSH HL
        CALL CSROON     ; sonst CAS einschalten
        POP HL
        LD B,00H
L78BC:
        CALL CASIN      ; Zeichen in Puffer lesen
        LD (HL),A
        INC HL
        DJNZ L78BC      ; 256 mal
        CALL CTOFF      ; abschalten
        DEC H
        XOR A
        LD B,A
L78C9:
        LD C,A
        ADD HL,BC
        LD A,(HL)
        CP 01AH
        SCF
        CCF
        RET NZ          ; NZ kein Dateiende
        LD (CASPRV),A
        SCF
        RET
L78D6:                  ; EOF auf CAS
        PUSH BC
        CALL L78A8      ; Zeichen lesen
        LD  HL,CASPRV
```



```
        JP L797E        ; zum EOF Test
L78E0:                  ; Funktion 12
        LD  HL,CASPRV   ; Temppufferadresse
L78E3:
        LD (HL),C       ; mit Zeichen laden
        JP NOSKCR


L78E7:                  ; ***** CAS Puffer wegschreiben
        CALL CWRTON     ; einschalten CAS
        LD B,00H        ; 256 Zeichen
L78EC:
        LD A,(HL)
        CALL CASOUT     ; Zeichen schreiben
        INC HL
        DJNZ L78EC
        JP CTWOFF       ; CAS abschalten

L78F6:                  ; Test ob INPUT modus
        LD A,(HL)
        CP 01H
        RET Z           ; wenn ja ret mit Z
        LD  BC,06H
        ADD HL,BC
        LD A,(HL)       ; sonst Pufferposition
L78FF:
        LD C,A          ; nach C
        LD (HL),00H     ; und Loeschen
        JR L791A
L7904:                  ; ***** Zeichen in die aktuelle
        LD E,A          ; Pufferposition schreiben
        LD  BC,06H
        ADD HL,B        ; aktuelle Pufferposition
        LD A,(HL)       ; holen
        INC (HL)        ; und um 1 erhoehen
        INC HL
        INC HL
        INC HL
        PUSH HL         ; auf Pufferanfang positionieren
        LD C,A
        ADD HL,BC       ; pufferposition addieren
        LD (HL),E       ; Zeichen in Puffer schreiben
        POP HL
        RET
L7914:
        LD  BC,06H
        ADD HL,BC
        LD A,(HL)
        INC (HL)
L791A:                  ; HL auf Pufferanfang positionieren
        INC HL
        INC HL
        INC HL
        AND A           ; Z Pufferposition=0
        RET
MDMDSP: DEFW L791F
        DEFW L7933
        DEFW L7939
        DEFW DERSOO
        DEFW L793E
        DEFW L7943
```

```
        DEFW FCERR
        DEFW FCERR
        DEFW L7948
        DEFW FCERR
        DEFW L794D

L7933:                  ; OPEN Modem
        CALL LFF5D
        JP DERBFN
L7939:                  ; CLOSE Modem
        CALL LFF60
        JR L7950
L793E:
        CALL LFF63
        JR L7950
L7943:
        CALL LFF66
        JR L7950
L7948:                  ; MODEM EOF
        CALL LFF69
        JR L7950
L794D:
        CALL LFF6C
L7950:
        JP FCERR
LPTDSP: 7953
        DEFW L782B
        DEFW L710A
        DEFW DERSOO
        DEFW L7967
        DEFW FCERR
        DEFW FCERR
        DEFW FCERR
        DEFW FCERR
        DEFW FCERR
        DEFW FCERR

L7967:                  ; LPT PutCHAR
        POP AF
        PUSH AF
        CALL OUTDLP
        JP POPALL
L796F:                  ; Byte aus Vorabpuffer
        LD A,(HL)       ; lesen
        LD (HL),OOH     ; Puffer loeschen
        AND A
        RET Z           ; Z wenn puffer leer war
        INC SP
        INC SP          ; Ret Adresse wegnehmen
        CP O1AH         ; auf EOF Testen
        SCF
        CCF
        RET NZ          ; NZ wenn kein EOF
        LD (HL),A       ; in Puffer speichern
        SCF
        RET             ; RET mit C fuer EOF
L797E:
        LD (HL),A       ; speichern des Zeichens
        SUB O1AH        ; Vergleich auf EOF
        SUB O1H         ; erst jetzt C oder NC
        SBC A,A         ; A=0 wenn kein ^Z
```



```
        JP CONIA        ; A=OFFH wenn ^Z
L7987:
        LD A,E          ; Filemodus holen
        CP O8H          ; Vergleich gegen APPEND
        JP Z,DERBFN     ; darf kein APPEND sein
        RET
CHKMDM: 798E            ; Null setzen des MODEM Flags
        DI
        XOR A
        LD (MDMFLG),A
L7993:
        RET
L7994:                  ; Initialisierung des MODEMS
        OUT (023H),A
        EX (SP),HL
        EX (SP),HL
        IN A,(023H)
        CP B
        RET NZ          ; NZ wenn MODEM nicht vorhanden
        INC A
        JR NZ,L7993
        DEC A           ; OFFH
        LD (MDMFLG),A   ; MODEM ist da
        LD  A,080H      ; Divisor Latch einblenden
        OUT (023H),A
        LD  A,014H      ; 9600 BAUD
        OUT (020H),A
        LD  A,00H
        OUT (021H),A
        LD  A,01AH      ; 7 Bits/CHAR   1 Stopbit
        OUT (023H),A    ;       even Parity enable
        LD  A,01H       ; Interrupt on received Data
        OUT (021H),A
        LD  A,0FH       ; OUT1   OUT2   RTS   DTR
        OUT (024H),A
        IN A,(020H)     ; Register freimachen
        IN A,(025H)
        IN A,(026H)
        RET
DIAL:   79C2            ; ****************************** DIAL
        CALL LFF6F
        JP FCERR
SCMTRP: 79C8            ; Modem trap
        PUSH HL
        LD  HL,MDM_OOS
        DI
        CALL REQTRP
        EI
        POP HL
        RET
RCVX:   79D3            ; Test ob Daten eingetroffen
        LD  A,(DATCNT)
        AND A
        RET
RS2INT: 79D8            ; Interrupt von der RS232
        CALL LFF72
        RET
BOOT:   79DC            ;***** pruefen ob Floppy angeschlossen
                        ;       wenn ja Booten
        DI              ; keine Interrupt mehr durchlassen
        XOR A
```

```
        LD B,A
L79DF:                  ; Pruefen ob
        OUT (032H),A    ; Floppy Controller
        LD C,A
L79E2:
        DJNZ L79E2      ; vorhanden ist
        IN A,(032H)
        CP C
        JP NZ,L7A45     ; NZ keine Controller
        INC A
        JR NZ,L79DF     ; NZ Test nicht beendet
        INC A
        OUT (038H),A    ; Single Density
        LD  A,0D1H
        OUT (030H),A    ; Force Interrupt
        CALL L7A49      ; RESTORE TRACK 00 fuer beide Drives
        LD  HL,00H
L79FA:
        DEC HL
        LD A,H
        OR L
        JR Z,L7A45      ; wenn 0 erreicht
        IN A,(030H)     ; Status
        AND 02H         ; Index maskieren
        JR NZ,L79FA     ; NZ kein Indexloch
        LD  HL,00H      ; Indexloch festgestellt
L7A08:                  ; noch einmal pruefen
        DEC HL
        LD A,H
        OR L
        JR Z,L7A45
        EX (SP),HL      ; wegen warten Status
        EX (SP),HL
        EX (SP),HL
        EX (SP),HL
        IN A,(030H)     ; Status Lesen
        AND 02H         ; Index maskieren
        JR Z,L7A08      ; Z kein Index weiter warten
        XOR A           ; 0 in das
        OUT (031H),A    ; Trackregister
        INC A           ; 1 in das
        OUT (032H),A    ; Sektorregister
        LD  HL,0C100H   ; Lade Adresse fuer Sektor 1
        LD  C,033H      ; Datenport
        LD  A,080H      ; READ SECTOR COMMAND
        OUT (030H),A
        EX (SP),HL      ; warten bis Statusregister
        EX (SP),HL      ; gelesen werden darf
L7A28:
        IN A,(034H)     ; Status einlesen
        ADD A,A         ; INTRQ nach Carry     DMARQ nach Sign
        JR C,L7A34      ; C =INTRQ=fertig
        JP P,L7A28      ; P=kein DMARQ weiter warten
        INI             ; Byte lesen und speichern
        JR L7A28        ; weiter warten
L7A34:
        IN A,(030H)     ; Status lesen
        AND 09CH        ; alle Fehlerbedingungen maskieren
        JP Z,LC100      ; Z kein Fehler in den Sektor springen
        LD  HL,RTYCNT   ; falls Fehler
        INC (HL)        ; Zaehler erhoehen
```



```
        LD A,(HL)
        CP 05H          ; falls kleiner 5 Versuche
        JP C,BOOT       ; nochmal das Ganze
L7A45:
        XOR A           ; sonst Floppy
        OUT (034H),A    ; abschalten
        RET
L7A49:                  ; RESTORE fuer beide Laufwerke
        LD  A,0EH       ; MOTOR ON beide SELECT fuer Laufwerk 1
        CALL L7A50
        LD  A,0DH       ;  "              "    "      "     0
L7A50:
        OUT (034H),A
        CALL L7A60
        LD  A,02H       ; RESTORE COMMAND langsamste STEP RATE
        OUT (030H),A
        EX (SP),HL      ; warten bevor Status gelesen werden darf
        EX (SP),HL
L7A5B:
        IN A,(034H)     ; INTRQ DMARQ einlesen
        RLCA            ; INTRQ nach Carry
        JR NC,L7A5B     ; NC kein TRK00 Signal
L7A60:
        IN A,(030H)     ; warten 1793 NOT BUSY
        RRA
        JR C,L7A60
        RET             ; Laufwerk ist noch selektiert

L7A66:
        .RADIX 16
INIDAT: ; 7A66
        DEFB 4A,53,1,1,1,1,0FF,0E0,30,1,0F,4,7,0C3,0,0
        DEFB 0C3,0,0,0F,0C3,0,0,0F2,0FA,032,8B,0FD
        .RADIX 10

FNKROM: ; 7A84
        DEFB ' color ',0,0,0,0,0,0,0,0,0,0
        DEFB ' auto',13,0,0,0,0,0,0,0,0,0,0
        DEFB ' goto ',0,0,0,0,0,0,0,0,0,0
        DEFB ' list ',0,0,0,0,0,0,0,0,0,0
        DEFB ' run',13,0,0,0,0,0,0,0,0,0,0,0

        DEFB ' color 15,4,5',13,0,0
        DEFB ' cload"',0,0,0,0,0,0,0,0,0,0
        DEFB ' cont',13,0,0,0,0,0,0,0,0,0,0
        DEFB ' list .',13,0,0,0,0,0,0,0,0,0
        DEFB 12,'run',13,0,0,0,0,0,0,0,0,0,0,0

        DEFB 0FFH,0FFH
                                ; Routine fuer 0FACOH
        DI
        LD  A,0FH       ; Register 15
        OUT (088H),A    ; selektieren
        IN A,(090H)     ; Bankbyte einlesen
        LD C,A          ; Speichern
        AND 0FEH        ; Cartridge bit loeschen
        OUT (08CH),A    ; Cartridge einblenden
        LD HL,(00H)     ; Byte 0 und 1 lesen
        LD A,L
        CP 0F3H         ; ist Byte 0    DI
        JR NZ,L7B40     ; NZ nein
```

```
        LD A,H
        CP 031H         ; ist BYTE 1    LD SP,
        JP Z,L0         ; Z ja also in Cartridge springen
L7B40:
        LD A,C          ; sonst altes Bankbyte
        OUT (08CH),A    ; wieder setzen
        RET
MON:    ;7B44           ; ****************************** MON
        CALL LFF84
        JP FCERR
MONERR: ;7B4A
        CALL LFFA8
        JP DIOERR
L7B50:  ;INIT           ; ****************** Kaltstart
        DI
        LD  SP,0F4F6H
        LD  HL,00H      ; zuerst warten
L7B57:
        DEC HL
        LD A,H
        OR L
        JR NZ,L7B57
        LD  A,0FH       ; Bankregister Selektieren
        OUT (088H),A
        LD  A,0DFH      ; BANK 01/02     CAPS LED aus
        OUT (08CH),A
        CALL INITIO     ; IO Bausteine initialisieren
        LD HL,(0FFFEH)  ; Wert holen
        LD  DE,0534DH   ; falls Speicher initialisiert
        RST 20H
        JR Z,L7B99      ; Z    nicht loeschen
        LD  HL,0C000H   ; Speichertest ab C000
        LD E,L
        LD D,H
        XOR A
        LD B,0FFH
L7B78:
        LD (HL),A
        INC A
        INC HL
        DJNZ L7B78      ; 255 Bytes schreiben      0..0FEH
        EX DE,HL        ; alte Adresse
        LD  A,03FH      ; noch weitere 63 mal
L7B80:
        LD  C,0FFH      ; 255 Bytes
        LDIR
        DEC A
        JR NZ,L7B80
        LD  HL,0C000H   ; Anfangsadresse
        LD  C,040H      ; 64 mal
L7B8C:
        XOR A
        LD B,0FFH       ; 255 Zeichen
L7B8F:
        CP (HL)         ; mit A vergleichen
L7B90:
        JR NZ,L7B90     ; NZ wenn ein Fehler aufgetreten ist
        INC A           ; naechstes Zeichen
        INC HL          ; mit naechster Adresse
        DJNZ L7B8F      ; vergleichen
        DEC C           ; weitere Versuche ?
```



```
        JR NZ,L7B8C
L7B99:                  ; Versuche durch   ab 0C000 getestet
        LD  HL,0534DH
        LD (0FFFEH),HL  ; Speichertest fertig signalisieren
INIENT: ;7B9F
        LD  SP,0F4F6H
        IM 1
        LD  BC,05B4H    ; Anzahl bytes
        LD  HL,FRSTID   ; ab dieser Adresse
        LD D,H
        LD E,L
        INC DE
        LD (HL),00H
        LDIR            ; loeschen
        LD B,069H       ; Anzahl der Einsprungadressen
        LD  HL,0FE79H   ; Sprung Basis
L7BB6:
        LD (HL),0C9H    ; auf RET setzen
        INC HL
        INC HL
        INC HL          ; naechste Einsprungadresse
        DJNZ L7BB6
        LD  HL,RAMLOW   ; alles darueber ist fuer BASIC
        LD (FIMEM),HL
        CALL L7CA7      ; Speichertest ab 8000H
        LD (BOTTOM),HL  ; untere Speicheradresse setzen
        LD  BC,04EH     ; Anzahl Bytes
        LD  DE,RAMLOW   ; mit denen der BASIC speicher
        LD  HL,CONSTR   ; initialisiert wird
        LDIR
        LD  BC,0DEH     ; Initialisierung der Funktionstasten
        LD  DE,FRSTID
        LD  HL,INIDAT
        LDIR
        CALL GICINI     ; initialisierung des AY-3-8910
        CALL BEEP       ; BEEP
        DI
        XOR A
        LD (ENDBUF),A   ; ans Ende des Puffers
        LD (NLONLY),A   ; fuer Dateien
        LD  A,':'       ; fuer Umschluesselung
        LD (KBUF-1),A
        LD  HL,PRMSTK   ; Adresse des Parameter Stacks
        LD (PARMPRV),HL
        LD (STKTOP),HL
        LD  BC,0C8H
        ADD HL,BC
        LD (MEMSIZ),HL
        LD  A,01H
        LD (0F7EFH),A
        LD (SCNCNT),A
        CALL DEFILE
        CALL STKINI
        LD HL,(BOTTOM)
        XOR A
        LD (HL),A       ; Programmtext initialisieren
        INC HL
        LD (TXTTAB),HL  ; und Programmtextanfang
        CALL SCRTCH     ; alles Loeschen und initialisieren,
        CALL CHKROM/CLOC ; pruefen ob Cartridge vorhanden,
        CALL BOOT       ; pruefen ob floppy vorhanden
```

CONSTR=84C

```
        CALL CHKMDM     ; MODEM test
        CALL L7C2B
        JP READY        ; los gehts mit BASIC
L7C2B:
        LD  A,0FFH      ; SWITCH Flag auf -1 setzen
        LD (SWIFLG),A
        LD  A,0FH
        DI
        OUT (088H),A    ; Bank- Register Adresse
        IN A,(090H)     ; Bankbyte lesen
        LD C,A
        XOR 04H         ; Bank 22 invertieren
        LD B,A          ; nach B
        AND 04H         ; Bank 22 maskieren
        PUSH AF
        CALL NZ,INITXT  ; wenn wir in Bank 02 dann Textbildschirm
        POP AF          ; initialisieren
        LD  HL,L7D9A    ; Text 'initi .. 2. bank'
        JR NZ,L7CA4
        LD (SWIFLG),A   ; auf 0 setzen
        CALL CHKBNK     ; Speicher pruefen in Bank B
        PUSH AF
        CALL PRLOGO     ; Anfangs Bild
        CALL INITXT     ; Textmodus einschalten
        LD  HL,L7D47    ; TEXT SV ext ....
        CALL STROUT     ; ausgeben
        LD HL,(VARTAB)
        EX DE,HL        ; Variablenadresse nach DE
        LD HL,(STKTOP)  ; Stack nach HL
        LD A,L
        SUB E
        LD L,A
        LD A,H
        SBC A,D
        LD H,A          ; HL:=HL-DE
        LD  BC,0FFF2H   ; -14
        ADD HL,BC
        POP AF
        JR C,L7C9E
        PUSH HL
        DI
        IN A,(090H)     ; Bank register lesen
        LD C,A
        XOR 04H         ; BANK 22 invertieren
        OUT (08CH),A    ; Bank selektieren
        LD  HL,00H
        LD (FRSTID),HL
        CALL L7CA7
        LD A,C
        OUT (08CH),A    ; vorige Bank einschalten
        EX DE,HL
        LD HL,(BOTTOM)
        RST 20H
        JR Z,L7C9C
        LD A,H
        ADD A,A
        POP HL
        ADD HL,HL
        LD  DE,04000H
        JP P,L7C97
        ADD HL,DE
```



```
        JR L7C9E
L7C97:
        OR A
        SBC HL,DE
        JR L7C9E
L7C9C:
        POP HL
        ADD HL,HL
L7C9E:
        CALL LINPRT
        LD  HL,L7D8E
L7CA4:
        JP STROUT
L7CA7:                  ; Speicher 8000 bis 80FF pruefen
        LD  HL,08000H
L7CAA:                  ; Speicher ab HL bis H FF pruefen
        LD A,(HL)       ; Ret mit C000 falls Fehler
        CPL             ; sonst mit H 00
        LD (HL),A
        CP (HL)
        CPL
        LD (HL),A
        JR NZ,L7CB6
        INC L
        JR NZ,L7CAA
        RET
L7CB6:
        LD  HL,0C000H
        RET
SETMAX: JP 7CBA         ; ******** Maxfiles setzen
        RST 8H
        DEFB 0B7H       ; FILES Token
        RST 8H
        DEFB 0F1H       ; = Token
        CALL GETBYT     ; Byte holen
        JP NZ,SNERR     ; falls nicht EOS Fehler
        CP 16           ; Maxfiles muss 0..15 sein
        JP NC,FCERR     ; sonst Fehler
        LD (TEMP),HL    ; Programmpointer speichern
        PUSH AF
        CALL CLSALL     ; alle Dateien schliessen
        POP AF
        CALL DEFILE
        CALL CLEARO     ; CLEAR ausfuehren
        JP NEWSTT       ; naechste Anweisung
DEFILE: JP 7CDA         ; ***** Einrichten der Dateipuffer
        PUSH AF         ; maxfiles
        LD HL,(HIMEM)
        LD  DE,0FEF5H   ; Puffergroesse
L7CE1:
        ADD HL,DE       ; wenigstens einmal
        DEC A
        JP P,L7CE1      ; bis A=0FFH
        EX DE,HL
        LD HL,(STKTOP)
        LD B,H
        LD C,L          ; BC STKTOP
        LD HL,(MEMSIZ)
        LD A,L
        SUB C
        LD L,A
```

```
        LD A,H
        SBC A,B
        LD H,A          ; HL:=HL-BC
        POP AF
        PUSH HL
        PUSH AF
        LD  BC,08CH
        ADD HL,BC
        LD B,H
        LD C,L
        LD HL,(VARTAB)
        ADD HL,BC
        RST 20H
        JP NC,OMERR
        POP AF
        LD (MAXFILES),A
        LD L,E
        LD H,D
        LD (FILTAB),HL  ; Dateipufferadresse 0 setzen
        DEC HL
        DEC HL
        LD (MEMSIZ),HL
        POP BC
        LD A,L
        SUB C
        LD L,A
        LD A,H
        SBC A,B
        LD H,A          ; HL:=HL-BC
        LD (STKTOP),HL
        DEC HL
        DEC HL
        POP BC
        LD SP,HL
        PUSH BC
        LD  A,(MAXFILES)
        LD L,A
        INC L
        LD H,00H        ; maxfiles in HL
        ADD HL,HL       ; *2
        ADD HL,DE       ; +Tabellenadresse
        EX DE,HL        ; nach DE
        PUSH DE
        LD  BC,FILLEN
L7D31:
        LD (HL),E
        INC HL
        LD (HL),D
        INC HL
        EX DE,HL
        LD (HL),00H     ; Datei- Modus 0 = geschlossen
        ADD HL,BC
        EX DE,HL
        DEC A
        JP P,L7D31
        POP HL
        LD  BC,09H      ; 9 Bytes mehr fuer Puffer 0
        ADD HL,BC
        LD (NULBUF),HL
        RET
```



```
L7D47:
        DEFB 'SV extended BASIC version 1.1',13,10
        DEFB 'Copyright 1983 (C) by Microsoft corp.',13,10,0
L7D8E:
        DEFB 'Bytes free',0
L7D9A:
        DEFB 'Initializing 2nd bank',0


; ***************************************
L7DB0:
; ***************************************



        ;******************** ALTER UNBENUTZTER CODE
LASTWR: LD A,0B0H
        LD E,L
        OUT (088H),A
        LD  A,0DFH
        OUT (08CH),A
        CALL L34D5
        LD HL,(0FFFEH)
        LD  DE,0534DH
        RST 20H
        JR Z,L7DEC
        LD  HL,0C000H
        LD E,L
        LD D,H
        XOR A
        LD B,0FFH
L7DCB:
        LD (HL),A
        INC A
        INC HL
        DJNZ L7DCB
        EX DE,HL
        LD  A,03FH
L7DD3:
        LD  C,0FFH
        LDIR
        DEC A
        JR NZ,L7DD3
        LD  HL,0C000H
        LD  C,040H
L7DDF:
        XOR A
        LD B,0FFH
L7DE2:
        CP (HL)
L7DE3:
        JR NZ,L7DE3
        INC A
        INC HL
        DJNZ L7DE2
        DEC C
        JR NZ,L7DDF
```

```
L7DEC:
        LD  HL,0534DH
        LD (OFFFEH),HL
        LD  SP,0F4F6H
        IM 1
        LD  BC,05A7H
        LD  HL,FRSTID
        LD D,H
        LD E,L
        INC DE
        LD (HL),00H
        LDIR
        LD B,065H
        LD  HL,SCNCNT
L7E09:
        LD (HL),0C9H
        INC HL
        INC HL
        INC HL
        DJNZ L7E09
        LD  HL,RAMLOW
        LD (HIMEM),HL
        CALL L7EF7
        LD (BOTTOM),HL
        LD  BC,04EH
        LD  DE,RAMLOW
        LD  HL,07F8H
        LDIR
        LD  BC,0DEH
        LD  DE,FRSTID
        LD  HL,079D7H
        LDIR
        CALL L4000
        CALL L4058
        DI
        XOR A
        LD (ENDBUF),A
        LD (NLONLY),A
        LD  A,03AH
        LD (KBUF-1),A
        LD  HL,PRMSTK
        LD (PARMPRV),HL
        LD (STKTOP),HL
        LD  BC,0C8H
        ADD HL,BC
        LD (MEMSIZ),HL
        LD  A,01H
        LD (0F7EFH),A
        CALL L7F2A
        CALL L6526
        LD HL,(BOTTOM)
        XOR A
        LD (HL),A
        INC HL
        LD (TXTTAB),HL
        CALL L64C8
        CALL CHKROM/CLOC
        CALL L794D
        CALL L78FF
        CALL L7E7B
        JP L95B
```



```
L7E7B:
        LD  A,0FFH
        LD (SWIFLG),A
        LD  A,0FH
        DI
        OUT (088H),A
        IN A,(090H)
        LD C,A
        XOR 04H
        LD B,A
        AND 04H
        PUSH AF
        CALL NZ,L353D
        POP AF
        LD  HL,07FEAH
        JR NZ,L7EF4
        LD (SWIFLG),A
        CALL L341C
        PUSH AF
        CALL L471C
        CALL L353D
        LD  HL,07F97H
        CALL L68EE
        LD HL,(VARTAB)
        EX DE,HL
        LD HL,(STKTOP)
        LD A,L
        SUB E
        LD L,A
        LD A,H
        SBC A,D
        LD H,A
        LD  BC,0FFF2H
        ADD HL,BC
        POP AF
        JR C,L7EEE
        PUSH HL
        DI
        IN A,(090H)
        LD C,A
        XOR 04H
        OUT (08CH),A
        LD  HL,00H
        LD (FRSTID),HL
        CALL L7EF7
        LD A,C
        OUT (08CH),A
        EX DE,HL
        LD HL,(BOTTOM)
        RST 20H
        JR Z,L7EEC
        LD A,H
        ADD A,A
        POP HL
        ADD HL,HL
        LD  DE,04000H
        JP P,L7EE7
        ADD HL,DE
        JR L7EEE
L7EE7:
        OR A
```

```
        SBC HL,DE
        JR L7EEE
L7EEC:
        POP HL
        ADD HL,HL
L7EEE:
        CALL L5AB5
        LD  HL,07FDEH
L7EF4:
        JP L68EE
L7EF7:
        LD  HL,08000H
L7EFA:
        LD A,(HL)
        CPL
        LD (HL),A
        CP (HL)
        CPL
        LD (HL),A
        JR NZ,L7F06
        INC L
        JR NZ,L7EFA
        RET
L7F06:
        LD  HL,0C000H
        RET

L7F0A:
        RST 8H
        DEFB 0B7H
        RST 8H
        DEFB 0F1H
        CALL L1A35
        JP NZ,L899
        CP 010H
        JP NC,LF4A
        LD (TEMP),HL
        PUSH AF
        CALL L72EE
        POP AF
        CALL L7F2A
        CALL L64E8
        JP LDEA
L7F2A:
        PUSH AF
        LD HL,(HIMEM)
        LD  DE,0FEF5H
L7F31:
        ADD HL,DE
        DEC A
        JP P,L7F31
        EX DE,HL
        LD HL,(STKTOP)
        LD B,H
        LD C,L
        LD HL,(MEMSIZ)
        LD A,L
        SUB C
        LD L,A
        LD A,H
        SBC A,B
```



```
        LD H,A
        POP AF
        PUSH HL
        PUSH AF
        LD  BC,08CH
        ADD HL,BC
        LD B,H
        LD C,L
        LD HL,(VARTAB)
        ADD HL,BC
        RST 20H
        JP NC,L64B6
        POP AF
        LD (MAXFILES),A
        LD L,E
        LD H,D
        LD (FILTAB),HL
        DEC HL
        DEC HL
        LD (MEMSIZ),HL
        POP BC
        LD A,L
        SUB C
        LD L,A
        LD A,H
        SBC A,B
        LD H,A
        LD (STKTOP),HL
        DEC HL
        DEC HL
        POP BC
        LD SP,HL
        PUSH BC
        LD  A,(MAXFILES)
        LD L,A
        INC L
        LD H,00H
        ADD HL,HL
        ADD HL,DE
        EX DE,HL
        PUSH DE
        LD  BC,FILLEN
L7F81:
        LD (HL),E
        INC HL
        LD (HL),D
        INC HL
        EX DE,HL
        LD (HL),00H
        ADD HL,BC
        EX DE,HL
        DEC A
        JP P,L7F81
        POP HL
        LD  BC,09H
        ADD HL,BC
        LD (NULBUF),HL
        RET

L7F97:
        DEFB 'SV extended BASIC version 1.0',13,10
```

```
        DEFB 'Copyright 1983 (C) by Microsoft corp.',0
L7FDE:
        DEFB 'Bytes free',0
L7FEA:
        DEFB 'Initializing 2nd bank',0
```


Notizen