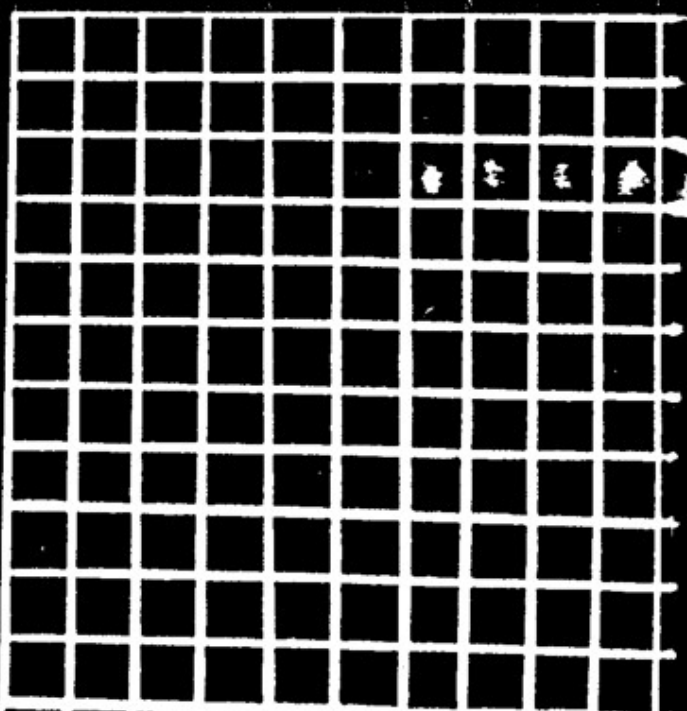


***** L. WALTER

REARENZ/MCGORRHH

DAS SINCLAIR SPECTRUM ROM



HueberSoftware

Das Betriebssystem des ZX-Spectrum

1. Einleitung: Die »Vorgeschichte«.

Der Sinclair ZX80 und der ZX81 sind mit hinlänglicher Sicherheit jedem Leser bekannt. Das seinerzeit Neue und auch weiterhin Verblüffende an diesen »Systemen« war und bleibt eine schon fast genial zu nennende Reduzierung der benötigten Bauteile auf das absolute Minimum. Hierzu wurden alle Kniffe eingesetzt, die der Spezialist erst mit Mühe durchschaut. Diese Kniffe gehen weit über das »normale« Ausnutzen der käuflichen Komponenten, insbesondere des Prozessors Z80, hinaus. Jede Eigenschaft, jedes vorhandene Signal, auch in ungewöhnlicher und unvorhergesehener Verwendung, wurde eingesetzt. Es sei in diesem Zusammenhang nur an die Bilderzeugung erinnert, die aus einer erstaunlichen Zweckentfremdung des Interrupts gewonnen wird.

Die Prinzipien beim Entwurf des ZX80 waren damit:

- konsequentes Minimieren des Bauteileaufwandes
- Ausgleich durch erhöhten Programmieraufwand
- dadurch höhere Entwicklungskosten
- aber niedrige Stückkosten in der Produktion,

also ein klarer Weg zum Massenprodukt. Als das Konzept sich bewährte, folgte auf den diskret (mit käuflichen Bauteilen) aufgebauten ZX80 bald der ZX81 mit einem kundenspezifischen Gate-Array, in dem die früheren TTL-Schaltkreise zusammengefaßt wurden. Für das gesamte Gerät waren damit alle Funktionen in einem Baustein erfaßt, welche über Prozessor, RAM und ROM hinausgingen. Das Betriebssystem der Rechner ZX80 und ZX81 zeigte starke Spuren dieser Arbeit. Hier war offensichtlich und bewundernswert gekämpft worden, um in der Hardware auch das Nötigste womöglich zu sparen. Im Falle des Arbeitsspeichers und seiner Verwaltung schoß man allerdings über das Ziel hinaus; allerdings waren zur Planungsphase RAM-Bausteine nicht annähernd so billig wie derzeit. In dem überknappen 1 Kilobyte drängeln sich die Systemvariablen, das Bildschirm-RAM, ein BASIC-Programm und dessen Variable. Der Anteil des Bildschirmspeichers ist nur so groß, wie an Zeichen innerhalb der Zeilen wirklich vorhanden ist. Bei leerem Bildschirm schrumpft er auf ein Byte pro Zeile und wird durch eine Platzmach-Routine

gedehnt, wann immer ein Zeichen in eine Position auf dem Schirm zu bringen ist, von der aus nach rechts gesehen noch kein Zeichen steht. Bei all diesen Operationen verschieben sich das BASIC-Programm und Variable im Speicher; eine mühselige Tat. Da der Prozessor weiterhin die meiste Zeit mit dem Aufbereiten des Bildes beschäftigt ist, wird die Rechengeschwindigkeit drastisch verringert. Das Anfügen eines 16 Kilobyte-Moduls verändert dieses Vorgehen natürlich nicht, schneller wird der ZX80/81 damit also in keinem Fall, auch wenn jetzt für einen stabilen Speicher Platz wäre.

Der Prozessor Z80 kann, wie fast alle 8-Bit-Prozessoren, einen Gesamtspeicher von 64 KByte mit seinen 16 Adreßleitungen ansprechen. Die höchstwertigste dieser Leitungen, A15, dient im ZX81 jedoch zum Umschalten zwischen den Betriebsarten »Denken« und »Bild darstellen«. Sie ist somit zur Speicheradressierung nicht verwendbar; der adressierbare Platz reduziert sich auf 32 KByte. Hier von verbleiben für das RAM 16 KByte. Mehr als diese 16 KByte an Arbeitsspeicher ist also zunächst, ohne besondere Maßnahmen, gar nicht einbaubar. Der ZX-Spectrum besitzt diese Einschränkung nicht mehr.

Der Autor spricht sich nicht frei davon, bei der Markteinführung von ZX80 und ZX81 mit ein wenig Überheblichkeit gelächelt zu haben. Intensive Beschäftigung, insbesondere mit der Hardware, haben diese Einstellung aufgehoben. Selbst bei der Einschätzung, daß es sich hier jedenfalls mehr um Spielzeuge denn um industriell verwertbare Geräte handelt, gebührt der Entwicklungsarbeit Anerkennung.

1.1 Der ZX-Spectrum

Der Sinclair ZX-Spectrum soll hier in seiner Nachfolge der bisherigen ZX80 und ZX81 betrachtet werden, aber auch in Abgrenzung zu anderen Geräten der gleichen Preisklasse. Es ist das Ziel dieser Einführung, dem Leser einen Überblick über die spezifischen Merkmale zu verschaffen und deren Wertung zu ermöglichen. Diese erscheint neben dem Auflisten und Erläutern des Betriebssystems vorrangig.

Der ZX-Spectrum basiert weitgehend auf seinem Vorgänger, dem ZX81. Weiße Teile der Programmierung lassen dies erkennen. Seine Hardware ist aufwendiger, schließlich vermittelt er farbige Darstellung, und die Reduzierung der Hardware erscheint nicht mehr so kraß wie bei jenem; vielleicht hat man sich auch daran gewöhnt. Ermöglicht

durch gefallene Bauteilpreise sind bereits im Grundmodell 16 Kilobyte an Arbeitsspeicher vorhanden, die auf 48 KByte erweiterbar sind. Zusammen mit dem Betriebssystem von 16 KByte nutzt der Spectrum also den möglichen Speicherbereich vollständig aus. Da der Prozessor Z80 über spezielle Befehle und eine Steuerleitung für I/O-Operationen verfügt, belasten die weiterhin angeschlossenen Systemeinheiten wie Tastatur usw. diesen Speicherplatz nicht, denn mit den IN- und OUT-Befehlen sind die gesamten 64 KByte als I/O-Adressen oder Ports nutzbar. Auch hiervon wird noch ausführlich die Rede sein.

Der grundsätzlich vorhandene Speicherbereich von 16 KByte erlaubt eine feste Anlage des Bildschirmspeichers, und wieder einmal waren die Konstrukteure gründlich: Im Spectrum gibt es keinen zeichenorientierten Bildschirmspeicher, dessen Inhalt erst durch das Zeichensatz-ROM eine grafische Form findet; hierin unterscheidet er sich von den meisten anderen Geräten. Der Bildschirm ist in direkten Punkten organisiert; jedem Punkt entspricht ein Bit im Speicher. Das bedeutet, daß bereits beim Schreiben eines Zeichens auf den Schirm die punktmäßige Form des gewünschten Zeichens dem Zeichensatz entnommen wird und als »Bild« in den Speicher gelangt. Damit sind Zeichendarstellung und Einzelpunktgrafik identisch. Bei einer Darstellung von 22 Bildschirmzeilen mit je 32 Zeichen ergibt sich die Punktauflösung, da jedes Zeichen aus 8x8 Punkten besteht, mit 176 Punkten in vertikaler und 256 Punkten in horizontaler Richtung. Der Aufbau des Bildschirms erlaubt auch die Funktion OVER, eine Überlagerung von Zeichen, ohne das untere, früher geschriebene, zu löschen; weiterhin die benutzerdefinierten Grafiksymbole, ebenfalls Unterscheidungsmerkmale zu anderen Geräten. Auf die Spitze getrieben scheint die Frage der Farb-»Attribute« zu einem Zeichenfeld, welche als permanente, transparente, zeitweise und transparente zeitweise gewählt werden können!

2. Das Betriebssystem

Unter dem Betriebssystem eines Rechners versteht man nicht etwa die Sprache, in der man sich mit ihm verständigt, sondern lediglich den Teil der Software, welcher die inneren Einheiten des Systems anbindet und verwaltet. Hierunter sind hauptsächlich die Eingabeeinheiten, also Tastatur, externes Speichermedium und andere, sowie die Ausgabeeinheiten, also Bildschirm, Drucker und wiederum externe Speichermedien, zu fassen. Bei den meisten Rechnern ist dieses Betriebssystem im Speicher von

anderen Teilen weitgehend getrennt, auch bei BASIC-Rechnern. Es sei daran erinnert, daß die meisten Geräte mit Microsoft-BASIC arbeiten, welches dann als kompakter Block neben dem Betriebssystem im Speicher steht. Beim ZX-Spectrum stellt man fest, daß die Entwickler das Microsoft-BASIC zumindest recht gut kannten; allerdings stehen der Sprach-Interpreter und das Betriebssystem nicht voneinander getrennt. Der Aufbau ist eher chronologisch der Entwicklung aus dem ZX81.

2.1 Die Tastatur (ab \$028E)

Die Einbindung der Tastatur in die Gesamtschaltung ist ein Meisterkniff an minimalem Aufwand, welcher direkt aus dem ZX81 entnommen wurde und der sich um den Befehl

0296 ED 78 NXTREI IN A,(C) ;

konzentriert. Dieser Maschinenbefehl gibt die Register B und C des Z80 als Adresse aus, welche gleichbedeutend mit Zeile und Spalte der Tastaturmatrix sind, welche jeweils angesprochen ist. Ein Tastendruck wird dann entsprechend ausgewertet.

Wenn diese Routine selbst auch fast völlig dem ZX81 entstammt, so ist bei der internen Codierung doch ein wesentlicher Unterschied sichtbar: Der Spectrum arbeitet mit der genormten ASCII-Zeichentabelle, auch bezüglich der Groß- und Kleinbuchstaben. Dies ist leider nicht selbstverständlich, denn der ZX81 kannte keinerlei Übereinstimmung mit bekannten Codierungen. Das absolut Originäre an Sinclair-Rechnern ist die Reaktion auf Tasteneingaben. Ein kompliziertes System an logischen Abfragen gibt keinesfalls immer einen Buchstaben pro Tastendruck weiter; da alle BASIC-Funktionen im Ein-Tasten-Verfahren eingegeben werden. Abhängig von der Syntaxprüfung und der beiden verschiedenen (!) Shift-Tasten erscheint bei einem Tastendruck eine von bis zu sechs möglichen Funktionen. Diese Sechsfachbelegung der Tasten ist zumindest in der Anfangsphase der Beschäftigung mit dem Spectrum außerordentlich lästig und nur bedingt faszinierend.

Im Bereich von \$0095 bis \$0204 stehen die Namen der BASIC-Befehle. Wie in den bekannten Microsoft-BASICs ist das letzte Zeichen jedes Namens mit \$80 versehen, um mit dieser Hilfe das Wortende zu erkennen. Im Anschluß an diese Tabelle findet sich die Zuordnung der Tastaturmatrix zu den ASCII-Zahlenwerten. Durch deren Auswertung bei der Syntaxprüfung wird dann ein Zeichen oder auch ein BASIC-Befehl erkannt. Die Tokens (1-Byte-Kürzel) der einzelnen Funktionen wurden in Gruppen im Bereich von \$022C bis \$028D angelegt.

2.2 Der Recorderanschluß

Die umfangreichen Routinen der Recorderbedienung stehen im Bereich von \$04C2 bis \$09F3. Vom ZX81 ist noch bekannt, daß der Anschluß des Videosignals und des Recorder-Ausganges physikalisch identisch waren. In ähnlicher Weise sind beim ZX-Spectrum der Recorderanschluß und die Farbausgabe des Bildschirmrandes verknüpft worden. So sind die Zustände beim Arbeiten mit dem Recorder leichter ersichtlich; auch können technische Fehler wie falscher Pegel eventuell erkannt werden.

Die Recorder-Routinen haben gegenüber dem ZX81 weitgehende Verbesserungen erhalten, was den Vorspann (Header) betrifft und die Sicherheit des Arbeitens mit dem Recorder, beim ZX81 häufig ein Zufallsprodukt, enorm erhöht. Der stabile Bildschirm des Spectrum gestattet das Anzeigen der auf dem Band gefundenen Header (Programmnamen). Neu ist die Routine MERGE, das Anfügen von Programmen und Variablen.

2.3 Der Tongenerator

Im Speicherbereich von \$03B5 bis \$04A9 stehen einige Routinen, welche den eingebauten Lautsprecher betreiben. Der Anschluß ist sehr einfach gehalten, der Lautsprecher ist an dem Ausgangs-Port SFE, Bit 4, angeschlossen. Wird per Programm dort eine »Null« ausgegeben, so ist der Stromfluß eingeschaltet, eine »Eins« schaltet den Strom wieder aus. Der ZX-Spectrum beinhaltet also keinen »Sound-Controller« wie andere Geräte; ein Ton muß vom Prozessor selbst durch Ein- und Ausschaltknackse des Lautsprechers im entsprechenden Rhythmus erzeugt werden. Um eine annähernde Reinheit eines solchermaßen gewonnenen Tones zu erzielen, sind die Zeiten des Ein- und Ausschaltens genau stabil einzuhalten. Es ist daher nicht zulässig, die Tonerzeugung durch einen Programminterrupt zu unterbrechen. Während eines Tones sind daher Interrupts gesperrt; Tastatur und Uhrzeit arbeiten kurzzeitig nicht. Die Werte der Periodendauer gewinnt der Spectrum mittels seiner Rechenroutinen, indem die gewünschte Zeitdauer und die Frequenz in Periodendauer bzw. Zählimpulse und Anzahl von Perioden umgerechnet werden.

2.4 Der Bildschirm

Wie bereits erwähnt, besitzt der ZX-Spectrum einen festen Bildschirmspeicher, welcher pro Bildpunkt ein Bit reserviert. Der Bildschirmspeicher beginnt bei \$4000 (16384)

und geht bis \$5800 (22528), hat also eine Länge von 6144 Bytes. Dies entspricht 192 Zeilen zu je 256 Punkten, da in ein Byte 8 Punkte passen. Die 192 Zeilen entsprechen 24 Textzeilen. Der Spectrum unterscheidet allerdings in interessanter Weise zwischen »oberem« und »unterem« Bildschirmteil. Dem Benutzer sind innerhalb eines BASIC-Programmes die 22 oberen Zeilen zugänglich, also auch 176 vertikale Punktpositionen. Der untere Teil mit 2 Textzeilen dient den Eingaben. Dieser untere Teil kann allerdings dynamisch vergrößert werden, wenn entsprechend mehr eingetippt wird. Damit sind hier BASIC-Zellen nicht auf die Breite einer Zeile begrenzt.

Der Leser mag sich an dieser Stelle die unterschiedlichen Verfahren vor Augen halten, welche zum Editieren von Programmzeilen angeboten werden. Ungeschlagen dürften die Commodore-Rechner aller Typen mit ihrem völlig wahlfreien Bildschirm-Editieren sein, ebenfalls übernommen beispielsweise in die Rechner von EPSON. An jeder beliebigen Stelle des Bildschirms kann eine Zeile modifiziert und durch ein Return neu eingegeben werden. Besonders beim Vervielfältigen von Zeilen ist dies hilfreich. Eine Zumutung dagegen APPLE-Computer, wo eine Zeile bei Änderungen neu getippt werden muß. Der Spectrum ist hier mittelmäßig komfortabel, da er wie mancher andere eine EDIT-Funktion beinhaltet: Eine ausgesuchte Zeile wird zum Editieren bereitgestellt, neue Eingaben überschreiben jedoch nicht etwa ältere, welche erst mit DELETE zu entfernen sind. Diese Eigenschaft wurde dem ZX81 entnommen.

Der Bildschirmspeicher enthält, es sei nochmals gesagt, nur Punkte. Es lassen sich hierdurch Zeichen grafisch addieren, ebenso ist das Zeichnen von Linien durch vorhandene Zeichen kein Problem. Grafische Figuren und Text sind ganz einfach identisch; eine Tatsache, die bei der Frage des Ausdrucks noch zu diskutieren sein wird.

Auf den Speicherbereich der Punkte folgen die »Attribute« mit 768 Bytes. Es gibt also pro Zeichenfeld (8x8 Punkte) eine Farbkennzeichnung, insgesamt 24 mal 32 Felder. Die Aussage, der Spectrum besitze hochauflösende Farbgrafik, wäre wie bei fast allen anderen Geräten auch irreführend. Tatsächlich lassen sich Farbinformationen nur für ein Zeichenfeld, nicht jedoch für jeden einzelnen Punkt, festlegen. Eine wirkliche Darstellung von beispielsweise 8 Farben, die an jedem Punkt des Schirms willkürlich wählbar wären, benötigte 3 Bit pro Punkt und im Bildschirmfeld des Spectrum somit 18432 Bytes (18 Kilobytes). Man überlege allerdings, daß auch der jetzige Weg schon 6 3/4 KBytes

für Punkt- und Farbinformationen verschlingt! Es sei nochmals gesagt, daß auch Geräte wie Commodore VC20 und C64 keine wirkliche Einzelpunkt-Farbgrafik im strengen Sinn besitzen.

Der ZX81 müht sich mit sehr ausgeklügelter Software um den Aufbau des Bildes, indem der Interrupt stets eine Bildzeile erzeugt. Der NMI schaltet dabei das stabile Bild ein und aus. Im Spectrum ist die Bilderzeugung eine Frage der Hardware; eine enorm höhere Rechengeschwindigkeit ist die Folge. Von der Bilderzeugung selbst ist der Z80 nicht belastet, lediglich von der Verwaltung des Bildschirminhaltes, was selbstverständlich ist.

2.5 Der Druckerpuffer

Eine weitere Speicherseite (256 Bytes oder 1/4 KByte), feststehend bei \$5b00 (23296), dient der Aufsummierung von Informationen für den Drucker. Hiermit sind nun schon die ersten 7 KByte des Speichers belegt. Der ZX81 mit seinem bescheidenen einen Kilobyte reservierte dem Drucker 32 kostbare Speicherplätze, welche den selben Zweck erfüllen. Im Spectrum ist jedoch auch der Druckerspeicher in Punkten orientiert; auch hier sind Text und Grafik mischbar. Die Farbinformationen sind für den Drucker natürlich unnötig. Weitere Angaben zur Frage des Druckers finden sich im Abschnitt Erweiterungen und I/O-Probleme.

2.6 Das Floppy-Betriebssystem

Hierzu ist eine klare Aussage angebracht — es gibt keines. Die auf der Tastatur zwar vorhandenen Befehle zur Ansprache eines solchen Speichermediums laufen im Spectrum-ROM schlicht auf Fehlermeldungen. Erweiterungen dieser Art können daher niemals einfach angesteckt werden, sondern verlangen ein externes ROM oder den Wechsel des eingebauten. Mehr hierzu im Abschnitt Erweiterungen.

Damit sind die Einzelpunkte des Betriebssystems übergreifend erläutert; der Leser kann den Spectrum eher mit anderen Systemen vergleichen.

3. Der BASIC-Interpreter

Der BASIC-Interpreter leistet die Aufgaben der Zeileninterpretation, der Befehlsausführung, der Berechnung mathematischer Ausdrücke, wozu er die arithmetischen Routi-

nen selbst und den Gleitkommarechner enthält, welcher unter anderem die Polynomberechnung der trigonometrischen Funktionen usw. ausführt. Der Spectrum enthält besonders bei letzteren mehr Funktionen, wie alle inversen, welche sich in anderen Rechnern nur durch Äquivalenzgleichungen ermitteln lassen. Der Befehlsvorrat wurde gegenüber dem ZX81 enorm erweitert; man erinnere sich daran, daß im Handbuch des ZX81 steht, »Befehle wie DATA und READ gäbe es zwar, der ZX81 käme aber ohne sie aus«; mußte er wohl oder übel. Der Spectrum enthält diese und weitere Befehle. Enorme Schwierigkeiten bereiten weiterhin die logischen Befehle wie AND und OR, welche grundsätzlich anders als von allen Interpretern bekannt reagieren und keineswegs für logische Prüfungen innerhalb eines Bytes oder innerhalb dezimaler Zahlen bis 255 tauglich sind. Das Behandeln von Bits innerhalb eines Bytes scheidet hier aus, solches müßte durch mühseliges Teilen und Prüfen geschehen.

Dankbar registriert die Befehle PLOT, DRAW und CIRCLE, wer schon einmal auf einem VC20 oder C64 mit der »Einzelpunktgrafik« gearbeitet hat, für die dort kein ähnlicher Komfort geboten wird. Beim Befehl CIRCLE sei als Kuriosität angemerkt, daß dieser aus Geschwindigkeitsgründen mäßig feine Geradenstücke zur Bildung von Kreissegmenten heranzieht.

Ein kritischer Gedanke gebührt der bis zu sechsfachen Belegungen von Tasten. Man überlege sich, daß der Grund darin liegt, unbedingt eine »Ein-Tasten-Erzeugung« aller BASIC-Befehle zu erreichen. Durch die verschiedenen Shift-Modi, welche vor dieser »einen« Taste zu bewältigen sind, erhöht sich die Anzahl der zu drückenden Tasten häufig über die Anzahl der Buchstaben hinaus, welche eigentlich den BASIC-Befehl ausgemacht hätten. Kurioses Beispiel sind die roten Tokens unter den Buchstabentasten, im Listing ab \$0246: Für die eckigen und geschweiften Klammern, mithin für ein einziges Zeichen, muß erst Symbol Shift mit Caps Shift gedrückt werden, um in den »E«-Cursor-Mode zu gelangen, dann schließlich Symbol Shift zusammen mit dem unglücklichen Zeichen, welches man eigentlich haben möchte ...

(Offensichtlich sind manche Konstrukteure bereits derart geschifft, durchcontroliert und reverst, daß das Erstaunen über den TI59 mit seiner »2ND«-Taste in Vergessenheit geraten ist. Auch eine Art Lernprozeß.)

Im restlichen Bereich des Speicherplatzes von \$386E bis \$3CFF ist das ROM leer und enthält als Inhalt \$FF.

4. Der Zeichensatz

In den letzten drei Speicherseiten von \$3D00 bis \$3FFF stehen die darstellbaren Text- und Grafikzeichen. Da die Zeichen in der geläufigen 8x8-Matrix wiedergegeben werden, sind pro Zeichen acht Bytes notwendig. Insgesamt kennt der Spectrum also 96 Zeichen. Die groben Grafiksymbbole in Viertelzeichentechnik berechnet er, diese finden sich daher nicht im Zeichensatz. Um den Zeichensatz auch in seinen Details klarzustellen, erfolgt hier ein vergrößerter Abdruck. Jedem Punkt eines Zeichens entsprechen dabei vier Drucknadeln; die Reihenfolge der Zeichen entspricht dem ROM-Inhalt. Zu der Form der Zeichen ist bemerkenswert, daß diese an allen vier Kanten eine Punktposition freilassen; dies ist ungewöhnlich. Da die Positionen der Zeichen auf dem Bildschirm keinen Abstand besitzen, ist es notwendig, die Zeichen links oder rechts sowie oben und unten nicht ganz an den Rand der Matrix gehen zu lassen, da diese sonst auf dem Bildschirm aneinander geraten würden. Die Maßnahmen, an allen vier Seiten Luft zu lassen, reduziert allerdings die Matrix der Zeichen von 8x8 auf 6x6. Hiermit ist kein besonders schöner Zeichensatz mehr machbar, wie der nachfolgende Originalabdruck zeigt.

4.1 Erweiterung, I/O-Probleme

Technische Fragen externer Anschlüsse sind Stoff eines Hardware-Handbuches; an dieser Stelle sollen daher die softwareseitigen Fragen beantwortet werden.

Die Floppy-Disk-Befehle CAT, ERASE, FORMAT und MOVE laufen an der Speicherstelle \$1793 schlicht auf die lapidare Meldung: »Invalid stream«. Hierfür ist im Spectrum keine Steuersoftware vorhanden. Der Anschluß einer Floppy-Disk oder des geplanten Microkassettenlaufwerkes (ZX Microdrive) bedingt einen ROM-Wechsel.

Für die Ansprache von Input- und Outputerweiterungen kennt der Spectrum die Befehle IN und OUT, deren Bedeutung schlicht dem Z80-Code entstammt. Eine Ausgabe erfolgt bei \$1E7D mit »OUT (C),A«, eine Eingabe bei \$34A8 mit »IN A,(C)«. Letzter Befehl ist Teil der Rechenroutinen. Zum Verständnis dieser Befehle sei das Handbuch des Rechners und des Z80 empfohlen. Unter Berücksichtigung derjenigen Ports, welche der ZX-Spectrum für eigene Zwecke belegt und an denen die Systemeinheiten wie Tastatur, Lautsprecher, Recorder und Drucker angeschlossen sind, kann der Anwender eigene externe Schaltungen

gen direkt aus BASIC-Programmen mit diesen Befehlen betreiben. Vor der unvollständigen Dekodierung der internen Systemeinheiten sei allerdings gewarnt, da in der Regel nur jeweils ein Bit des Adreßbusses während einer I/O-Operation Null sein muß, um das gewünschte Gerät anzusprechen. So steht beispielsweise geschrieben, der Drucker werde mit Port 251 angesprochen, das entspräche einer Null an A2. Diese Null alleine spricht tatsächlich den Drucker an; alles andere ist derzeit aber egal, so daß ein (unzulässiges) OUT nach Port Null so etwa alle Systemeinheiten aktivierte...

5. Entstehung dieses Buches

Am Beginn stand die Aufgabe, für den Spectrum einen optimalen Drucker bereitzustellen. Wie dem Leser geläufig ist, drucken die Sinclair-Drucker auf aluminisiertem Papier und sind beispielsweise für Korrespondenz keinesfalls geeignet. Dieser Drucker sollte nicht nur möglichst viele Schriftarten beherrschen, sondern selbstverständlich auch die Grafik des Spectrums direkt und sauber wiedergeben. Da der Druckerpuffer im Spectrum nur Punkte beinhaltet, nicht jedoch Zeicheninformationen, wurden Eingriffe in das Betriebssystem nötig, um allen Komfort bieten zu können.

Die Vorgehensweise war abenteuerlich: Ein Disassembler für Z80-Maschinenprogramme stand lediglich lauffähig auf dem ZX81 zur Verfügung. Dieser erhielt daraufhin eine ROM-Erweiterung, welche das Spectrum-ROM aufnahm. Der nächste Schritt war eine Rechnerkopplung zwischen dem ZX81 und dem CBM 3032, wobei der ZX81 veranlaßt wurde, das Spectrum-ROM zu disassemblieren und dem CBM 3032 jede Zeile zu übertragen, welche dieser auf Diskette ablegte. Anschließend wurde das gesamte Listing mittels eines Z80-Assemblers (eigens geschrieben) auf einem CBM 8032 bearbeitet, kommentiert, jeweils wieder rückassembliert und mit dem Original-ROM verglichen, da sich immerhin Fehler einschleichen konnten. Insbesondere durch letzteren Vorgang ist sichergestellt, daß der Inhalt des bearbeiteten Listings frei von Fehlern ist, denn die Übereinstimmung mit dem tatsächlichen Spectrum-ROM ist jederzeit prüfbar.

Jedenfalls ergab sich im Verlaufe dieser Arbeit die Entwicklung einer Anschlußbaugruppe für Drucker, welche ohne zusätzliche Software mit den direkten Befehlen LPRINT, LLIST und COPY arbeitet und sogar ein hervorragendes Kopieren des Bildschirms in Einzelnadelgrafik erlaubt.

DAS KOMPLETTE ASSEMBLER-LISTING DES SINCLAIR SPECTRUM BETRIEBSSYSTEMS.

ADRESS CODE | QUELLZEILE

0000	:	ZX - SPECTRUM - HAUPTFILE	
0000		.LIB SPEC0000-S	
0000	:	SINCLAIR ZX SPECTRUM TEIL 0000	
0000	:		
0000		*= \$4000	
4000	RAMBEG **		: ANFANG DES RAMBEREICHS
4000	:		
4000	BILD ***+6144		: BILDSCHIRMSPEICHER
5800	:		
5800	ATTRSP ***+768		: ATTRIBUTSPEICHER
5800	:		
5800	PTRBUF ***+256		: PUFFERBEREICH FÜR DRUCKER-
5C00			: AUSGABE
5C00	:		
5C00	:	START DER SYSTEMVARIABLEN UND POINTER	
5C00	:		
5C00	KSTATE ***+8		: 2*4 SPEICHERPLATZE FÜR TASTATUR
5C08	LASTK ***+1		: ZULETZT GEDRUCKTE TASTE
5C09	REPDEL ***+1		: TV FRAMES DELETE BEFORE REPEAT
5C0A	REPPER ***+1		: TV FRAMES PER REPEAT
5C0B	DEFADD ***+2		: ARGUMENT FOR FUNCTION
5C0D	KDATA ***+1		: COLOR INPUT FROM KEYBOARD
5C0E	TVDATA ***+2		: COLOR, AT AND TAB POS.
5C10	-STRMS ***+38		: OFFENE KANÄLE
5C36	CHARS ***+2		: ZEICHENSATZADRESSE -256
5C38	RASP ***+1		: LÄNGE FÜR EINEN WARMPIEPSE
5C39	PIP ***+1		: LÄNGE DES TASTENKLICKS
5C3A	ERRNR ***+1		: FÜR MELDUNGEN: NUMMER -1
5C3B			: KEINE MELDUNG AUSZUGEBEN: \$FF
5C3B	FLAGS ***+1		: VARIOUS FLAGS
5C3C			: BIT 1: PRINTER IST ANGESPROCHEN
5C3C	TVFLAG ***+1		: FLAGS FÜR BILDSCHIRM
5C3D	ERRSP ***+2		: ERROR STACKPOINTER
5C3F	LISTSP ***+2		: LIST RETURN ADRESSE
5C41	MODE ***+1		: MODES K / L / C / E / G
5C42	NEWPPC ***+2		: NUMMER DER ZEILE, WOHIN
5C44			: GESPRUNGEN WIRD (GOTO USW.)
5C44	NSPPC ***+1		: BEFEHL DER ZEILE BEI SPRUNG
5C45	PPC ***+2		: AKTUELLE BASICZEILENNUMMER
5C47	SUBPPC ***+1		: ZEIGER AUF BEFEHL DER ZEILE
5C48	BORDCR ***+1		: BORDER COLOUR * 8 / ATTR. LO
5C49	EPPC ***+2		: AKTUELLE EDITORZEILE
5C4B	VARS ***+2		: BEGINN DER VARIABLEN
5C4D	DEST ***+2		: VARIABLENADRESSE BEI EINER
5C4F			: ZUWEISUNG
5C4F	CHANS ***+2		: POINTER FÜR KANALDATEN
5C51	CURCHL ***+2		: AKTUELLE I/O-INFO-ADRESSE
5C53	PROG ***+2		: START DES BASICPROGRAMMS
5C55	NXTLIN ***+2		: ADRESSE DER NÄCHSTEN BASICZEILE
5C57	DATADD ***+2		: ZEIGT AUF ENDEBYTE DER
5C59			: LETZTEN DATEN
5C59	ELINE ***+2		: ADRESSE EINES EINGEGEBENEN
5C5B			: BEFEHLS
5C5B	KCUR ***+2		: CURSORADRESSE
5C5D	CHADD ***+2		: NÄCHSTES ZU INTERPRETIERENDES

5C5F			; BASICZEICHEN
5C5F		XPTR ***+2	; ZEIGT BEI EINGABEFEHLER AUF
5C61			; DAS ZEICHEN NACH DEM ?
5C61		WORKSP ***+2	; DERZEITIGER WORKSPACE
5C63		STKBOT ***+2	; ANFANG DES CALCULATORSTACK
5C65		STKEND ***+2	; ANFANG DES FREIEN SPEICHERS
5C67		BREG ***+1	; CALCULATOR'S B REGISTER
5C68		MEM ***+2	; CALC MEM AREA (NORM.=MEMBOT)
5C6A		FLAGS2 ***+1	
5C6B		DFSZ ***+1	; ZEILENZAHL +1 IM UNTEREN
5C6C			; BILDSCHIRMTEIL
5C6C		STOP ***+2	; NUMMER DER OBERSTEN ZEILE
5C6E			; EINES LISTINGS
5C6E		OLDPPC ***+2	; ZEILENNUMMER FÜR CONTINUE
5C70		OSPCC ***+1	; NACHSTER BEFEHL FÜR CONTINUE
5C71		FLAGX ***+1	
5C72		STRLEN ***+2	; LENGTH OF STRING TYPE DEST.
			IN ASSGNMT
5C74		TADDR ***+2	; ADDRESS OF NEXT ITEM IN SYNTAX
			TABLE
5C76		SEED ***+2	; SEED FÜR RND / WIRD DURCH
5C78			; RANDOMIZE GESETZT
5C78		FRAMES ***+3	; TV FRAME COUNTER
5C7B		UDG ***+2	; USER DEF. GRAPHICS ADRESSE
5C7D		COORDS ***+2	; KOORDINATEN DES LETZTEN PLOT
5C7F		PPOSN ***+1	; 33-COL PRINTER POS.-ZEIGER
5C80		PRCC ***+2	; FÜR PRINTER-BUFFER
5C82		ECHOE ***+2	; 33-COL/24-ZEILENNR. FÜR INPUT
5C84		DFCC ***+2	; PRINTADRESSE IM DISPLAYFILE
5C86		DFCCL ***+2	; WIE DFCC FÜR UNTEREN TEIL
5C88		SPOSN ***+2	; 33-COL/24-ZEILENNR. FÜR PRINT
5C8A		SPOSNL ***+2	; 33-COL/24-ZEILENNR., UNT. TEIL
5C8C		SCRCT ***+1	; SCROLL-ZÄHLER
5C8D		ATTRP ***+1	; AKTUELLE FARBEN, PERMANENT
5C8E		MASKP ***+1	; " " , TRANSPARENT
5C8F		ATTRT ***+1	; " " , TEMPORARY
5C90		MASKT ***+1	; " " , TRANSP., TEMP.
5C91		PFLAG ***+1	
5C92		MEMBOT ***+30	; CALCULATOR SPEICHERBEREICH
5CB0		NMIREG ***+2	; FÜR NMI-ROUTINE BENUTZT
5CB2		RAMTOP ***+2	; LETZTER SPEICHERPLATZ FÜR BASIC
5CB4		PRAMT ***+2	; LETZTER RAMSPEICHERPLATZ
5CB6		KANMEM	
5CB6		; *****	
5CB6			
5CB6			
5CB6		***\$0	
0000	F3	RESET D1	; START DES BETRIEBSSYSTEMS
0001	AF	XOR A	; REG A LÖSCHEN
0002	11 FF FF	LD DE,\$FFFF	; OBERSTE MÖGLICHE RAMZELLE
0005	C3 CB 11	JP RESET1	
0008			
0008		; RESTART - VEKTOREN	
0008			
0008	2A 5D 5C	ERRAUS LD HL,(CHADD)	; DER ERRORZEIGER WIRD AUF DIE
000B	22 5F 5C	LD (XPTR),HL	; EINGABE GESETZT, WELCHE NICHT
000E	18 43	JR \$53	; MEHR INTERPRETIERT WERDEN KANN
0010			


```

0010 C3 F2 15 PRTOUT JP $15F2 ; BUCHSTABEN IN REG A AUSDRUCKEN
0013 ;
0013 FF .BYT $FF,$FF,$FF,$FF,$FF
0014 FF
0015 FF
0016 FF
0017 FF
0018 ;
0018 2A 5D 5C GETAKT LD HL,(CHADD) ; HOLT AKTUELLES ZEICHEN AUS
0018 7E LD A,(HL) ; PROGRAMM ODER BEI EINGABE
001C CD 7D 00 AUSWER CALL $7D ; PRÜFEN AUF STEUERZEICHEN
001F D0 RET NC ; DRUCKBARES ZEICHEN/ BASICTOKEN
0020 ;
0020 CD 74 00 GETNXT CALL $74 ; HOLT NACHSTES ZEICHEN
0023 1B F7 JR AUSWER
0025 FF .BYT $FF,$FF,$FF
0026 FF
0027 FF
0028 ;
0028 C3 5B 33 CALRUF JP $335B ; SPRUNG IN RECHENROUTINE
0028 FF .BYT $FF,$FF,$FF,$FF,$FF
002C FF
002D FF
002E FF
002F FF
0030 ;
0030 C5 REST30 PUSH BC ; RESERVIERE SPEICHERPLATZ IN
0031 2A 61 5C LD HL,(WORKSP) ; WORKSPACE. ANZAHL = BC
0034 E5 PUSH HL
0035 C3 9E 16 JP $169E
0038 ;
0038 ; INTERRUPT - ROUTINE (UHR UND TASTATUR)
0038 ;
0038 F5 INTERR PUSH AF ; REGISTER RETTEN
0039 E5 PUSH HL
003A 2A 7B 5C LD HL,(FRAMES) ; FRAMES INKREMENTIEREN
003D 23 INC HL ; ZEIT IN 1/50 SEKUNDEN
003E 22 7B 5C LD (FRAMES),HL
0041 7C LD A,H ; DIE 2 BYTES VON FRAMES
0042 B5 OR L ; AUF NULL PRÜFEN
0043 20 03 JR NZ,$48
0045 FD 34 40 INC (IY+$40) ; WENN NULL, DRITTES BYTE
0048 ; INKREMENTIEREN
0048 C5 PUSH BC
0049 D5 PUSH DE
004A CD BF 02 CALL KEYBOA ; TASTATURABFRAGE
004D D1 POP DE ; REGISTER ZURÜCKLADEN
004E C1 POP BC
004F E1 POP HL
0050 F1 POP AF
0051 FB EI
0052 C9 RET
0053 ;
0053 E1 ERROR POP HL ; ADRESSE DER FEHLERSTELLE LADEN
0054 6E LD L,(HL) ; DIE DORT STEHENDE FEHLERNUMMER
0055 FD 75 00 LD (IY+0),L ; IN 'ERRNR' SCHREIBEN
0058 ED 7B 3D 5C LD SP,(ERRSP) ; SP RESTAURIEREN
005C C3 C5 16 JP $16C5

```

```

005F
005F FF          .BYT $FF,$FF,$FF,$FF,$FF,$FF,$FF
0060 FF
0061 FF
0062 FF
0063 FF
0064 FF
0065 FF
0066
0066          ; NICHT MASKIERBAREN INTERRUPT BEARBEITEN
0066
0066 F5          NMI      PUSH AF
0067 E5          PUSH HL
0068 2A B0 5C     LD HL,(NMIREG) ; FALLS NMIREG = NULL, WIRD EIN
0068 7C          LD A,H        ; KALTSTART BEI $0 DURCHFÜHRT,
006C B5          OR L
006D 20 01       JR NZ,$70
006F E9          JP (HL)
0070 E1          POP HL        ; IN ALLEN ANDEREN FÄLLEN PASSIERT
0071 F1          POP AF        ; N I C H T S
0072 ED 45       RETN
0074
0074 2A 5D 5C     LD HL,(CHADD) ; PROGRAMMZEIGER ERHÖHEN
0077 23          INC HL
0078 22 5D 5C     LD (CHADD),HL
0078 7E          LD A,(HL)     ; UND DAS NEUE ZEICHEN LADEN
007C C9          RET
007D
007D          ; VERGLEICH DER BASIC - ZEICHEN
007D
007D FE 21       CP $21        ; RÜCKKEHR, WENN >= $21
007F D0          RET NC        ; ALSO BEI ALLEN DRUCKBAREN
0080          ; ZEICHEN ODER TOKENS
0080 FE 0D       CP $D         ; ODER CARRIAGE RETURN
0082 C8          RET Z
0083
0083 FE 10       CP $10        ; RETURN MIT CARRY BEI $00...$0F
0085 D8          RET C
0086
0086 FE 18       CP $18
0088 3F          CCF
0089          ; BEEINFLUSSUNG VON CHADD, DEM AKTUELLEN
0089          ; BASIC-ZEICHENZEIGER
0089 D8          RET C        ; RETURN MIT CARRY BEI $18...$20
008A
008A 23          INC HL        ; HIER BEI $10-$17 INKREMENTIEREN
008B FE 16       CP $16        ; $16, $17 AUSBLENDEN
008D 3B 01       JR C,$90      ; BEI $10-15 NICHT INKREMENTIEREN
008F 23          INC HL        ; NUR BEI 16, 17 NOCHMAL ERHÖHEN
0090 37          SCF
0091 22 5D 5C     LD (CHADD),HL ; AKTUELLES BASIC-ZEICHEN SETZEN
0094 C9          RET          ; MIT CARRY SET
0095
0095          ; -----
0095          ;
0095          ; HIER STEHEN UMFANGREICHE TABELLEN, BEGINNEND MIT:
0095          ; ALLE BASIC - STATEMENTS IN ASCII-TEXTFORM
0095          ; DAS LETZTE ZEICHEN IST MIT $80 GEODERT
0095

```

0095 BF
 0096 52 4E
 0098 C4
 0099 49 4E
 009E A4
 009F 50
 00A0 C9
 00A1 46
 00A2 CE
 00A3 50 4F 49 4E
 00A7 D4
 00AB 53 43
 00AE A4
 00AF 41 54 54
 00B2 D2
 00B3 41
 00B4 D4
 00B5 54 41
 00B7 C2
 00B8 56 41 4C
 00BB A4
 00BC 43 4F 44
 00BF C5
 00C0 56 41
 00C2 CC
 00C3 4C 45
 00C5 CE
 00C6 53 49
 00C8 CE
 00C9 43 4F
 00CB D3
 00CC 54 41
 00CE CE
 00CF 41 53
 00D1 CE
 00D2 41 43
 00D4 D3
 00D5 41 54
 00D7 CE
 00D8 4C
 00D9 CE
 00DA 45 58
 00DC D0
 00DD 49 4E
 00DF D4
 00E0 53 51
 00E2 D2
 00E3 53 47
 00E5 CE
 00E6 41 42
 00E8 D3
 00E9 50 45 45
 00EC CB
 00ED 49
 00EE CE
 00EF 55 53
 00F1 D2
 00F2 53 54 52

.BYT \$BF ; FRAGEZEICHEN
 .BYT 'RN', \$C4 ; RND
 .BYT 'INKEY', \$A4 ; INKEY\$
 .BYT 'P', \$C9 ; PI
 .BYT 'F', \$CE ; FN
 .BYT 'POIN', \$D4 ; POINT
 .BYT 'SCREEN', \$A4 ; SCREEN\$
 .BYT 'ATT', \$D2 ; ATTR
 .BYT 'A', \$D4 ; AT
 .BYT 'TA', \$C2 ; TAB
 .BYT 'VAL', \$A4 ; VAL\$
 .BYT 'COD', \$C5 ; CODE
 .BYT 'VA', \$CC ; VAL
 .BYT 'LE', \$CE ; LEN
 .BYT 'SI', \$CE ; SIN
 .BYT 'CO', \$D3 ; COS
 .BYT 'TA', \$CE ; TAN
 .BYT 'AS', \$CE ; ASN
 .BYT 'AC', \$D3 ; ACS
 .BYT 'AT', \$CE ; ATN
 .BYT 'L', \$CE ; LN
 .BYT 'EX', \$D0 ; EXP
 .BYT 'IN', \$D4 ; INT
 .BYT 'SQ', \$D2 ; SQR
 .BYT 'SG', \$CE ; SGN
 .BYT 'AB', \$D3 ; ABS
 .BYT 'PEE', \$CB ; PEEK
 .BYT 'I', \$CE ; IN
 .BYT 'US', \$D2 ; USR
 .BYT 'STR', \$A4 ; STR\$

00F3	A4	.BYT 'CHR', \$A4 ; CHR\$
00F6	43 48 52	
00F9	A4	.BYT 'NO', \$D4 ; NOT
00FA	4E 4F	
00FC	D4	
00FD	42 49	.BYT 'BI', \$CE ; BIN
00FF	CE	
0100	4F	.BYT 'O', \$D2 ; OR
0101	D2	
0102	41 4E	.BYT 'AN', \$C4 ; AND
0104	C4	
0105	3C	.BYT '<', \$BD ; <=
0106	BD	
0107	3E	.BYT '>', \$BD ; >=
0108	BD	
0109	3C	.BYT '<', \$BE ; <>
010A	BE	
010B	4C 49 4E	.BYT 'LIN', \$C5 ; LINE
010E	C5	
010F	54 48 45	.BYT 'THE', \$CE ; THEN
0112	CE	
0113	54	.BYT 'T', \$CF ; TO
0114	CF	
0115	53 54 45	.BYT 'STE', \$D0 ; STEP
0118	D0	
0119	44 45	.BYT 'DEF F', \$CE ; DEF FN
011E	CE	
011F	43 41	.BYT 'CA', \$D4 ; CAT
0121	D4	
0122	46 4F	.BYT 'FORMA', \$D4 ; FORMAT
0127	D4	
0128	4D 4F 56	.BYT 'MOV', \$C5 ; MOVE
012B	C5	
012C	45 52 41 53	.BYT 'ERAS', \$C5 ; ERASE
0130	C5	
0131	4F 50	.BYT 'OPEN ', \$A3 ; OPEN #
0136	A3	
0137	43 4C	.BYT 'CLOSE ', \$A3 ; CLOSE #
013D	A3	
013E	4D 45 52 47	.BYT 'MERG', \$C5 ; MERGE
0142	C5	
0143	56 45	.BYT 'VERIF', \$D9 ; VERIFY
0148	D9	
0149	42 45 45	.BYT 'BEE', \$D0 ; BEEP
014C	D0	
014D	43 49	.BYT 'CIRCL', \$C5 ; CIRCLE
0152	C5	
0153	49 4E	.BYT 'IN', \$CB ; INK
0155	CB	
0156	50 41 50 45	.BYT 'PAPE', \$D2 ; PAPER
015A	D2	
015B	46 4C 41 53	.BYT 'FLAS', \$CB ; FLASH
015F	CB	
0160	42 52	.BYT 'BRIGH', \$D4 ; BRIGHT
0165	D4	
0166	49 4E	.BYT 'INVERS', \$C5 ; INVERSE
016C	C5	
016D	4F 56 45	.BYT 'OVE', \$D2 ; OVER

0170	D2	.BYT 'OU',\$D4 ; OUT
0171	4F 55	
0173	D4	.BYT 'LPRIN',\$D4 ; LPRINT
0174	4C 50	
0179	D4	.BYT 'LLIS',\$D4 ; LLIST
017A	4C 4C 49 53	
017E	D4	.BYT 'STO',\$D0 ; STDP
017F	53 54 4F	
0182	D0	.BYT 'REA',\$C4 ; READ
0183	52 45 41	
0186	C4	.BYT 'DAT',\$C1 ; DATA
0187	44 41 54	
018A	C1	.BYT 'RESTOR',\$C5 ; RESTORE
018B	52 45	
0191	C5	.BYT 'NE',\$D7 ; NEW
0192	4E 45	
0194	D7	.BYT 'BORDE',\$D2 ; BORDER
0195	42 4F	
019A	D2	.BYT 'CONTINU',\$C5 ; CONTINUE
019B	43 4F	
01A2	C5	.BYT 'DI',\$CD ; DIM
01A3	44 49	
01A5	CD	.BYT 'RE',\$CD ; REM
01A6	52 45	
01AB	CD	.BYT 'FO',\$D2 ; FOR
01A9	46 4F	
01AB	D2	.BYT 'GO T',\$CF ; GO TO
01AC	47 4F 20 54	
01B0	CF	.BYT 'GO SU',\$C2 ; GO SUB
01B1	47 4F	
01B6	C2	.BYT 'INPU',\$D4 ; INPUT
01B7	49 4E 50 55	
01BB	D4	.BYT 'LOA',\$C4 ; LOAD
01BC	4C 4F 41	
01BF	C4	.BYT 'LIS',\$D4 ; LIST
01C0	4C 49 53	
01C3	D4	.BYT 'LE',\$D4 ; LET
01C4	4C 45	
01C6	D4	.BYT 'PAUS',\$C5 ; PAUSE
01C7	50 41 55 53	
01CB	C5	.BYT 'NEX',\$D4 ; NEXT
01CC	4E 45 58	
01CF	D4	.BYT 'POK',\$C5 ; POKE
01D0	50 4F 4B	
01D3	C5	.BYT 'PRIN',\$D4 ; PRINT
01D4	50 52 49 4E	
01D8	D4	.BYT 'PLO',\$D4 ; PLOT
01D9	50 4C 4F	
01DC	D4	.BYT 'RU',\$CE ; RUN
01DD	52 55	
01DF	CE	.BYT 'SAV',\$C5 ; SAVE
01E0	53 41 56	
01E3	C5	.BYT 'RANDOMIZ',\$C5 ; RANDOMIZE
01E4	52 41	
01EC	C5	.BYT 'I',\$C6 ; IF
01ED	49	
01EE	C6	.BYT 'CL',\$D3 ; CLS
01EF	43 4C	

```

01F1 D3
01F2 44 52 41 .BYT 'DRA', $D7 ; DRAW
01F5 D7
01F6 43 4C 45 41 .BYT 'CLEA', $D2 ; CLEAR
01FA D2
01FB 52 45 .BYT 'RETUR', $CE ; RETURN
0200 CE
0201 43 4F 50 .BYT 'COP', $D9 ; COPY
0204 D9
0205 ;
0205 ; =====
0205 ;
0205 ; TABELLE DER ZUORDNUNG KEYBOARD MATRIX - ASCII
0205 ; DIESE TABELLE FÜHRT VON DER VERDRAHTUNG DER TASTEN
0205 ; AUF IHREN ASCII-WERT.
0205 ;
0205 42 KEYTAB .BYT $42 ; B REIHE 1
0206 48 .BYT $48 ; H
0207 59 .BYT $59 ; Y
0208 36 .BYT $36 ; 6
0209 35 .BYT $35 ; 5
020A 54 .BYT $54 ; T
020B 47 .BYT $47 ; G
020C 56 .BYT $56 ; V
020D ;
020D 4E .BYT $4E ; N REIHE 2
020E 4A .BYT $4A ; J
020F 55 .BYT $55 ; U
0210 37 .BYT $37 ; 7
0211 34 .BYT $34 ; 4
0212 52 .BYT $52 ; R
0213 46 .BYT $46 ; F
0214 43 .BYT $43 ; C
0215 ;
0215 4D .BYT $4D ; M REIHE 3
0216 4B .BYT $4B ; K
0217 49 .BYT $49 ; I
0218 3B .BYT $3B ; 8
0219 33 .BYT $33 ; 3
021A 45 .BYT $45 ; E
021B 44 .BYT $44 ; D
021C 5B .BYT $5B ; X
021D ;
021D 0E .BYT $0E ; SYMBOL SHIFT REIHE 4
021E 4C .BYT $4C ; L
021F 4F .BYT $4F ; O
0220 39 .BYT $39 ; 9
0221 32 .BYT $32 ; 2
0222 57 .BYT $57 ; W
0223 53 .BYT $53 ; S
0224 5A .BYT $5A ; Z
0225 ;
0225 20 .BYT $20 ; LEERTASTE REIHE 5
0226 0D .BYT $0D ; ENTER
0227 50 .BYT $50 ; P
0228 30 .BYT $30 ; 0
0229 31 .BYT $31 ; 1
022A 51 .BYT $51 ; Q

```

```

022B 41      .BYT $41      ; A
022C          ;
022C          ; -----
022C          ;
022C          ; TABELLEN DER BASIC-TOKENS
022C          ;
022C          ; (DURCHSUCHUNG NACH FESTSTELLUNG VON ASCII-WERT
022C          ; UND ZUSTAND DER FUNKTIONSTASTEN)
022C          ; HIER ZUERST:
022C          ; DIE GRÜNEN BEFEHLE AUF DEN TASTEN A...Z
022C          ;
022C E3      .BYT $E3      ; READ
022D C4      .BYT $C4      ; BIN
022E E0      .BYT $E0      ; LPRINT
022F E4      .BYT $E4      ; DATA
0230 B4      .BYT $B4      ; TAN
0231 BC      .BYT $BC      ; SGN
0232 BD      .BYT $BD      ; ABS
0233 BB      .BYT $BB      ; SQR
0234 AF      .BYT $AF      ; CODE
0235 B0      .BYT $B0      ; VAL
0236 B1      .BYT $B1      ; LEN
0237 C0      .BYT $C0      ; USR
0238 A7      .BYT $A7      ; PI
0239 A6      .BYT $A6      ; INKEY$
023A BE      .BYT $BE      ; PEEK
023B AD      .BYT $AD      ; TAB
023C B2      .BYT $B2      ; SIN
023D BA      .BYT $BA      ; INT
023E E5      .BYT $E5      ; RESTORE
023F A5      .BYT $A5      ; RND
0240 C2      .BYT $C2      ; CHR$
0241 E1      .BYT $E1      ; LLIST
0242 B3      .BYT $B3      ; COS
0243 B9      .BYT $B9      ; EXP
0244 C1      .BYT $C1      ; STR$
0245 BB      .BYT $BB      ; LN
0246          ;
0246          ; DIE ROTEN BEFEHLE UNTER DEN TASTEN A...Z
0246          ;
0246 7E      .BYT $7E      ; SCHLANGENLINIE
0247 DC      .BYT $DC      ; BRIGHT
0248 DA      .BYT $DA      ; PAPER
0249 5C      .BYT $5C      ; SCHRAGSTRICH RÜCKWARTS
024A B7      .BYT $B7      ; ATN
024B 7B      .BYT $7B      ; GESCHWEIFTE KLAMMER AUF
024C 7D      .BYT $7D      ; GESCHWEIFTE KLAMMER ZU
024D DB      .BYT $DB      ; CIRCLE
024E BF      .BYT $BF      ; IN
024F AE      .BYT $AE      ; VAL$
0250 AA      .BYT $AA      ; SCREEN$
0251 AB      .BYT $AB      ; ATTR
0252 DD      .BYT $DD      ; INVERSE
0253 DE      .BYT $DE      ; OVER
0254 DF      .BYT $DF      ; OUT
0255 7F      .BYT $7F      ; COPYRIGHT
0256 B5      .BYT $B5      ; ASN
0257 D6      .BYT $D6      ; VERIFY

```

0258	7C	.BYT \$7C	; STRICH
0259	D5	.BYT \$D5	; MERGE
025A	5D	.BYT \$5D	; ECKIGE KLAMMER ZU
025B	DB	.BYT \$DB	; FLASH
025C	B6	.BYT \$B6	; ACS
025D	D9	.BYT \$D9	; INK
025E	5B	.BYT \$5B	; ECKIGE KLAMMER AUF
025F	D7	.BYT \$D7	; BEEP
0260			
0260		; BEFEHLE ÜBER DEN TASTEN 0...9	
0260			
0260	0C	.BYT \$0C	; DELETE
0261	07	.BYT \$07	; EDIT
0262	06	.BYT \$06	; SHIFT LOCK
0263	04	.BYT \$04	; VIDEO NORMAL
0264	05	.BYT \$05	; INVERSE VIDEO
0265	08	.BYT \$08	; CURSOR NACH LINKS
0266	0A	.BYT \$0A	; CURSOR NACH UNTEN
0267	0B	.BYT \$0B	; CURSOR NACH OBEN
0268	09	.BYT \$09	; CURSOR NACH RECHTS
0269	0F	.BYT \$0F	; GRAFIK
026A			
026A		; ROTE BEFEHLE AUF DEN TASTEN A...Z	
026A			
026A	E2	.BYT \$E2	; STOP
026B	2A	.BYT \$2A	; *
026C	3F	.BYT \$3F	; ?
026D	CD	.BYT \$CD	; STEP
026E	C8	.BYT \$C8	; >=
026F	CC	.BYT \$CC	; TO
0270	CB	.BYT \$CB	; THEN
0271	5E	.BYT \$5E	; PFEIL NACH OBEN
0272	AC	.BYT \$AC	; AT
0273	2D	.BYT \$2D	; -
0274	2B	.BYT \$2B	; +
0275	3D	.BYT \$3D	; =
0276	2E	.BYT \$2E	; .
0277	2C	.BYT \$2C	; ,
0278	3B	.BYT \$3B	; SEMIKOLON
0279	22	.BYT \$22	; "
027A	C7	.BYT \$C7	; <=
027B	3C	.BYT \$3C	; <
027C	C3	.BYT \$C3	; NOT
027D	3E	.BYT \$3E	; >
027E	C5	.BYT \$C5	; OR
027F	2F	.BYT \$2F	; /
0280	C9	.BYT \$C9	; <>
0281	60	.BYT \$60	; POUND SIGN
0282	C6	.BYT \$C6	; AND
0283	3A	.BYT \$3A	; !
0284			
0284		; ROTE BEFEHLE UNTER DEN TASTEN 0...9	
0284			
0284	D0	.BYT \$D0	; FORMAT
0285	CE	.BYT \$CE	; DEF FN
0286	AB	.BYT \$AB	; FN
0287	CA	.BYT \$CA	; LINE
0288	D3	.BYT \$D3	; OPEN


```

0289 D4          .BYT $D4          ; CLOSE
028A D1          .BYT $D1          ; MOVE
028B D2          .BYT $D2          ; ERASE
028C A9          .BYT $A9          ; POINT
028D CF          .BYT $CF          ; CAT
028E            ;
028E            ; =====
028E            ;
028E            ; ENDE DER BEFEHLSTABELLEN
028E            ;
028E            ; =====
028E            ;
028E            ; KEYBOARD-ABFRAGEROUTINE
028E            ; SIE LIEFERT IN (E) EINE GEDRUCKTE TASTE ALS WERT
028E            ; $00-$27 SOWIE IN (D) DIE AUSSAGE ÜBER SHIFT-FUNKTION
028E            ;
028E 2E 2F          LD L,$2F          ; ZEIGER AUF ENDE DER
0290              ; KEYCODE-TABELLE +8
0290 11 FF FF      LD DE,$FFF          ; KEINE TASTE
0293 01 FE FE      LD BC,$FE          ; C = I/O-PORT DER TASTATUR
0296              ; B = MASKE MIT EINER NULL
0296              ; ZUM FESTSTELLEN EINER TASTE
0296 ED 78          NXTREI IN A,(C)    ; ABFRAGEN 1 SPALTE DER TASTATUR
0298 2F            CPL                ; EINE '1' SOLL EIN
0299              ; TASTENDRUCK SEIN
0299 E6 1F          AND $1F          ; NUR DIE UNTERSTEN 5 BIT SIND
029B              ; MIT TASTEN VERBUNDEN
029B 28 0E          JR Z,NOKEY        ; KEIN TASTENDRUCK
029D 67            LD H,A             ; MATRIXWERT NACH H (1-5 EINSSEN)
029E 7D            LD A,L             ; STARTWERT AUS L HOLEN
029F 14            TEST3 INC D         ; D HÄLT $FF BEI 1-2 GE-
02A0              ; DRÜCKTEN TASTEN
02A0 C0            RET NZ             ; RETURN BEI MEHR ALS ZWEI TASTEN
02A1 D6 0B          KEINTA SUB B      ; A ENTHÄLT DEN TABELLEN-
02A3              ; ZEIGER DES KEYCODE
02A3 CB 3C          SRL H             ; TASTENDRUCK SUCHE
02A5 30 FA          JR NC,KEINTA      ; BIS EINE EINS IM CARRY
02A7 53            LD D,E             ; FÜR ZWEI ZULASSIGE TASTEN
02A8              ; NOCHMAL $FF LADEN
02A8 5F            LD E,A             ; TASTENCODE ($00-$27) NACH E
02A9 20 F4          JR NZ,TEST3       ; NOCHMAL TASTE IN LAUFENDER
02AB              ; REIHE SUCHE
02AB              ;
02AB 2D            NOKEY DEC L         ; NACHSTE REIHE ZU JE FÜNF TASTEN
02AC CB 00          RLC B             ; MASKE IN B UM EINE STELLE NACH
02AE              ; LINKS SCHIEBEN
02AE 38 E6          JR C,NXTREI       ; NACHSTE TASTENREIHE
02B0              ;
02B0              ; WENN DIE NULL VON B IN CARRY GELANGT, ROUTINE FERTIG
02B0              ;
02B0 7A            LD A,D             ; TEST AUF FUNKTIONSTASTEN
02B1 3C            INC A              ; BEI EINER TASTE STEHT HIER $FF
02B2 CB            RET Z              ; RETURN, WENN NUR EINE TASTE
02B3              ; GEDRÜCKT WAR
02B3 FE 28          CP $2B            ; TASTE 'CAPS SHIFT' ?
02B5 CB            RET Z              ; JA
02B6 FE 19          CP $19            ; TASTE 'SYMBOL SHIFT' ?
02B8 CB            RET Z              ; JA

```

```

02B9 7B          LD A,E          ; REGISTER E UND D VERTAUSCHEN
02BA 5A          LD E,D          ;
02BB 57          LD D,A          ; ZUR PRÜFUNG DER
02BC FE 1A       CP $18         ; TASTE 'SYMBOL SHIFT'
02BE C9          RET            ; ZERO-FLAG GESETZT BEI 'SYMBOL
02BF             ; SHIFT + ANDERE TASTE
02BF             .END
02BF             .LIB SPEC0400-S
02BF             ; SINCLAIR ZX SPECTRUM TEIL 0400
02BF             ;
02BF             ; AUFRUF DER TASTATURABFRAGE UND AUSWERTUNG DER EINGABEN
02BF             ; ERFOLGT IM INTERRUPT, ALLE 1/50 SEKUNDEN
02BF             ;
02BF CD 8E 02    KEYBOA CALL $2BE ; TASTATURABFRAGE
02C2 C0          RET NZ
02C3             ; HIER ERFOLGT WEITERE AUSWERTUNG IN DEN FÄLLEN:
02C3             ; KEIN ODER EINFACHER TASTENDRUCK
02C3             ; 'CAPS SHIFT' PLUS ANDERE TASTE
02C3             ; 'SYMBOL SHIFT' PLUS ANDERE TASTE
02C3             ;
02C3             ; DIE TASTENAUSWERTUNG BENUTZT DIE 8 BYTES 'KSTATE'
02C3             ; IN ZWEI GRUPPEN ZU VIER BYTES, UM ZWEI FOLGENDE
02C3             ; GEDRÜCKTE TASTEN VERARBEITEN ZU KÖNNEN
02C3             ;
02C3             ; BYTE 0 : BELEGTKENNUNG (FF=FREI, 0=BELEGT)
02C3             ; BYTE 1 : REPEAT-ZÄHLER (3...0)
02C3             ; BYTE 2 : REPDEL (WARTEZEIT VOR BEGINN DES REPEAT)
02C3             ; BYTE 3 : CODE DER GEDRÜCKTEN TASTE, ASCII ODER TOKEN
02C3             ;
02C3 21 00 5C    LD HL,KSTATE
02C6 CB 7E       BIT 7,(HL)      ; IST TASTATURSPEICHER FREI ?
02C8 20 07       JR NZ,BLOCK2   ; SPRUNG BEI 'UNBENUTZT'
02CA 23          INC HL
02CB 35          DEC (HL)        ; REPEAT-ZÄHLER HERUNTERZÄHLEN
02CC 2B          DEC HL
02CD 20 02       JR NZ,BLOCK2
02CF 36 FF       LD (HL),$FF     ; BLOCK FREIGEBEN
02D1 7D          BLOCK2 LD A,L
02D2 21 04 5C    LD HL,KSTATE+4 ; BLOCK2 ANWÄHLEN
02D5 BD          CP L            ; WAREN WIR BEI BLOCK2 ?
02D6 20 EE       JR NZ,$2C6     ; WENN NICHT: FREIPRÜFUNG 2
02D8             ;
02D8 CD 1E 03    CALL $31E       ;
02DB D0          RET NC         ; ENDE, WENN KEINE TASTE
02DC 21 00 5C    LD HL,KSTATE
02DF BE          CP (HL)        ; NOCH DIE SELBE TASTE GEDRÜCKT?
02E0 2B 2E       JR Z,KEYREP    ; WENN JA, REPEAT
02E2 EB          EX DE,HL
02E3 21 04 5C    LD HL,KSTATE+4 ; BLOCK2 PRÜFEN
02E6 BE          CP (HL)        ; NOCH DIE SELBE TASTE GEDRÜCKT?
02E7 2B 27       JR Z,KEYREP    ; WENN JA, REPEAT
02E9 CB 7E       BIT 7,(HL)    ; BLOCK2 FREI ?
02EB 20 04       JR NZ,$2F1
02ED EB          EX DE,HL      ; KSTATE IN HL ZURÜCKLADEN
02EE CB 7E       BIT 7,(HL)    ; FREIPRÜFUNG
02F0 CB          RET Z         ; ENDE DER TASTATURABFRAGE
02F1             ; BEI BELEGT
02F1

```

```

02F1      : NEUE TASTE GEDRÜCKT
02F1      :
02F1 5F      LD E,A
02F2 77      LD (HL),A      : BYTE 0
02F3 23      INC HL        : ZEIGT AUF BYTE 1
02F4 36 05    LD (HL),5     : REPEAT-ZÄHLER AUF 5 SETZEN
02F6 23      INC HL        : ZEIGT AUF BYTE 2
02F7 3A 09 5C LD A,(REPDEN) : ZEITVORGABE FÜR VERZÖGERUNG
02FA 77      LD (HL),A     : IN BYTE 2 LÄDEN
02FB 23      INC HL        : ZEIGT AUF BYTE 3
02FC FD 4E 07 LD C,(IY+7)   : MODE
02FF FD 56 01 LD D,(IY+1)   : FLAGS
0302 E5      PUSH HL
0303 CD 33 03 CALL $333    : TASTENCODE (TOKEN) ERZEUGEN
0306 E1      POP HL
0307 77      LD (HL),A     : UND IN BYTE 3 SPEICHERN
0308 32 08 5C LD (LASTK),A : NEU SPEICHERN
0308        : NEU SPEICHERN
0308 FD CB 01 EE SET 5,(IY+1) : BIT 5, (FLAGS) ZEIGT
030F C9      RET          : NEUEN TASTENDRUCK AN
0310      :
0310      : REPEAT - FUNKTION
0310      :
0310 23      KEYREP INC HL   : REPEAT ZÄHLER ANWAHLEN
0311 36 05    LD (HL),5     : UND WIEDER AUF 5 VORSETZEN
0313 23      INC HL        : WARTENZEIT ANWAHLEN
0314 35      DEC (HL)       : UND HERUNTERZÄHLEN
0315 C0      RET NZ
0316      :
0316 3A 0A 5C LD A,(REPPER) : WIEDERHOLRATE SETZEN
0319 77      LD (HL),A     : IN BYTE 2
031A 23      INC HL        : ZEIGT AUF BYTE 3 (TASTE SELBST)
031B 7E      LD A,(HL)     : TASTENWERT HOLEN
031C 1B EA    JR $30B      : SETZE LETZEN TASTENDRUCK
031E      :
031E 42      LD B,0        : FUNKTIONSTASTEN MERKEN
031F 16 00    LD D,0
0321 7B      LD A,E
0322 FE 27    CP $27       : 'CAPS SHIFT'
0324 D0      RET NC        : ODER KEINE TASTE : RETURN
0325      :
0325 FE 18    CP $18       : 'SYMBOL SHIFT'
0327 20 03    JR NZ,$32C   : SPRUNG ZUR WEITEREN AUSWERTUNG
0329 CB 7B    BIT 7,B      : RETURN BEI 'SYMBOL SHIFT'
032B C0      RET NZ       : ALLEIN
032C 21 05 02 LD HL,KEYTAB : ADRESSE DER KEYCODE-TABELLE
032F 19      ADD HL,DE     : AKTUELLES ZEICHEN ALS ASCII-
0330 7E      LD A,(HL)    : WERT AUS DER TABELLE HOLEN
0331 37      SCF          : MIT CARRY WIRD 'ZEICHEN
0332 C9      RET          : GEFUNDEN' SIGNALISIERT
0333      : -----
0333      :
0333      : AUSWERTUNG DES ERMITTELTEN TASTENCODES
0333      : UND BILDUNG DER BASIC-TOKEN JE NACH EINGABEMODUS
0333      : (CURSOR MODE K, L, C UND E)
0333      :
0333 7B      LD A,E        : ASCII-CODE DER GEDRUCKTEN TASTE
0334 FE 3A    CP $3A      : BEI ZIFFERN, SPACE, ENTER

```

```

0336 38 2F      JR C,$367      ; ODER BEIDEN SHIFTTASTEN
0338 0D         DEC C          ; NUR BUCHSTABEN A...Z
0339 FA 4F 03   JP M,$34F
033C 28 03      JR Z,$341
033E C6 4F      ADD $4F        ; REVERS-BUCHSTABEN LIEGEN AB $90
0340 C9         RET
0341            ;
0341            ; 'GRÜNE TOKENS ÜBER TASTEN A...Z'
0341            ;
0341 21 EB 01    LD HL,$1EB
0344 04         INC B
0345 28 03      JR Z,TABDIR
0347            ;
0347            ; 'ROTE TOKENS UNTER DEN TASTEN A...Z'
0347            ;
0347 21 05 02    LD HL,KEYTAB
034A            ;
034A            ; ENDGÜLTIGES TOKEN WIRD AUS DER
034A            ; ENTSPRECHENDEN TABELLE GELADEN
034A            ;
034A 16 00      TABDIR LD D,0      ; E ENTHÄLT DEN ASCII-WERT
034C            ; DES ZEICHENS
034C 19         ADD HL,DE        ; IN HL STEHT BASISADRESSE
034D            ; MINUS $41 BZW. $30
034D 7E         LD A,(HL)
034E C9         RET
034F            ;
034F            ; 'ROTE TOKENS AUF DEN TASTEN A...Z'
034F            ;
034F 21 29 02    LD HL,$229
0352 CB 40      BIT 0,B
0354 28 F4      JR Z,TABDIR      ; SYMBOL SHIFT + BUCHSTABE
0356 CB 5A      BIT 3,D          ; K-MODUS
0358 28 0A      JR Z,$364
035A FD CB 30 5E BIT 3,(1Y+4B)    ; FLAG52 PRÜFEN AUF 'CAPS LOCK'
035E C0         RET NZ
035F 04         INC B            ; SHIFT ?
0360 C0         RET NZ
0361 C6 20      ADD $20          ; ASCII-WERT FÜR KLEINSCHREIBUNG
0363 C9         RET
0364            ;
0364 C6 A5      ADD $A5          ; DIREKT ERZEUGBARE TOKENS
0366 C9         RET            ; DURCH ADDITION ($E6..)
0367            ;
0367 FE 30      CP $'0'          ; ZEICHEN < $30 ?
0369 DB         RET C
036A 0D         DEC C            ; MODUS
036B FA 9D 03   JP M,$39D        ; BEI K, L, C
036E 20 19      JR NZ,$389        ; BEI G
0370            ;
0370            ; 'ROTE TOKENS UNTER 0...9'
0370            ;
0370 21 54 02    LD HL,$254
0373 CB 68      BIT 5,B
0375 28 D3      JR Z,TABDIR      ; OHNE 'CAPS SHIFT'
0377 FE 38      CP $'8'
0379 30 07      JR NC,$382
037B D6 20      SUB $20          ; AUS $30-$37 MACHE $10-$17

```



```

037D 04          INC B
037E CB          RET Z          ; WENN OHNE SHIFT
037F C6 0B       ADD B          ; ADDIERE B FÜR FARBCODE 18..1F
0381 C9          RET
0382             ; TASTEN '8' UND '9' (CODES FÜR BRIGHT UND FLASH)
0382 D6 36       SUB $36        ; ERZEUGE 2 UND 3 OHNE SHIFT
0384 04          INC B
0385 CB          RET Z          ; OHNE SHIFT
0386 C6 FE       ADD $FE        ; ERZEUGE 0 UND 1 MIT SHIFT
0388 C9          RET
0389             ;
0389             ; 'GRAFIKZEICHEN AUF DEN TASTEN 0...9'
0389             ;
0389 21 30 02     LD HL,$230     ; (MIT 'CAPS SHIFT')
038C FE 39       CP #'9'        ; TEST AUF 'GRAPHICS'
038E 2B BA       JR Z,TABDIR
0390 FE 30       CP #'0'        ; TEST AUF 'DELETE'
0392 2B B6       JR Z,TABDIR
0394 E6 07       AND 7          ; NUR TASTEN '1..7', AUS 8 WIRD 0
0396 C6 80       ADD $80        ; $80-$87 SIND DIE GRAFIKZEICHEN
0398 04          INC B
0399 CB          RET Z          ; OHNE SHIFT GEDRÜCKT: RETURN
039A EE 0F       XOR $F        ; ERZEUGE STEUERZEICHEN ÜBER
039C             ; DEN TASTEN 1...8
039C C9          RET          ; (CODES: $88-$8F)
039D 04          INC B
039E CB          RET Z          ; WENN OHNE SHIFT
039F CB 6B       BIT 5,B        ; PRÜFE AUF 'CAPS SHIFT'
03A1             ;
03A1             ; 'BEFEHLE ÜBER DEN TASTEN 0...9'
03A1             ;
03A1 21 30 02     LD HL,$230
03A4 20 A4       JR NZ,TABDIR
03A6 D6 10       SUB $10        ; AUS $30-$39 MACHE $20-$29
03A8 FE 22       CP $22        ; VERGLEICHE AUF 'AT-SIGN'
03AA 2B 06       JR Z,$3B2
03AC FE 20       CP $20        ; VERGLEICHE AUF UNTERSTREICHEN
03AE C0          RET NZ
03AF 3E 5F       LD A,$5F      ; UNDERLINE
03B1 C9          RET
03B2 3E 40       LD A,'$'
03B4 C9          RET
03B5             ; -----
03B5             ;
03B5             ; LAUTSPRECHER - ROUTINEN
03B5             ;
03B5             ; DER LAUTSPRECHER WIRD MIT BIT 4 VON PORT $FE AKTIVIERT
03B5             ; UND EINGESCHALTET, SOBALD EIN 'OUT' EINER NULL ERFOLGT.
03B5             ; DURCH OUT $FE MIT BIT 4 = EINS WIRD ER ABGESCHALTET.
03B5             ; RYTHMUS DES EIN- UND AUSSCHALTENS IST DIE TONFREQENZ.
03B5             ;
03B5 F3          PIEPEN DI      ; TASTATUR UND UHR LAUFEN
03B6 7D          LD A,L        ; WÄHREND PIEPTON NICHT
03B7 CB 3D       SRL L
03B9 CB 3D       SRL L
03BB 2F          CPL
03BC E6 03       AND 3
03BE 4F          LD C,A

```

03BF	06 00	LD B,0	
03C1	DD 21 D1 03	LD IX,\$3D1	; BASISADRESSE DER ZEITSCHLEIFE
03C5	DD 09	ADD IX,BC	
03C7	3A 4B 5C	LD A,(BORDCR)	; AKTUELLE BORDERCOLOUR BEI
03CA	E6 3B	AND \$3B	; AKTIVIERUNG DES LAUTSPRECHERS
03CC	0F	RRCA	; IN BITS 0..2 VON
03CD	0F	RRCA	; REG A BRINGEN
03CE	0F	RRCA	
03CF	F6 0B	OR B	; KASSETTENAUSGANG ABSCHALTEN
03D1			
03D1	00	NOP	
03D2	00	NOP	
03D3	00	NOP	
03D4	04	INC B	
03D5	0C	INC C	
03D6	0D	HALBZE DEC C	; ZEITSCHLEIFE FÜR HALBE
03D7	20 FD	JR NZ,HALBZE	; ZYKLUSLÄNGE
03D9	0E 3F	LD C,\$3F	
03DB	05	DEC B	
03DC	C2 D6 03	JP NZ,HALBZE	
03DF			
03DF	EE 10	XOR \$10	; BIT 4 INVERTIEREN, UM LAUT-
03E1	D3 FE	OUT (\$FE),A	; SPRECHER ABWECHSELND EIN-
03E3			; UND AUSZUSCHALTEN
03E3	44	LD B,H	; REG B WIEDER LADEN
03E4	4F	LD C,A	; REG A RETTEN
03E5	CB 67	BIT 4,A	; MITTE DES ZYKLUS ?
03E7	20 09	JR NZ,TONMIT	; JA
03E9			
03E9	7A	LD A,D	; NEIN: DE AUF 0 PRÜFEN
03EA	B3	OR E	
03EB	2B 09	JR Z,TONEND	; 0 = PIEPER ENDE
03ED	79	LD A,C	; REG A ZURÜCK
03EE	4D	LD C,L	; REG C WIEDER LADEN
03EF	1B	DEC DE	; SCHLEIFENZÄHLER -1
03F0	DD E9	JP (IX)	; UND NOCH MAL ZUR SCHLEIFE
03F2			
03F2	4D	TONMIT LD C,L	; REG C WIEDER LADEN
03F3	0C	INC C	
03F4	DD E9	JP (IX)	; UND WEITER
03F6			
03F6	FB	TONEND EI	; INTERRUPT WIEDER FREIGEBEN
03F7	C9	RET	
03F8			
03F8			; BEFEHLSAUSFÜHRUNG DES 'BEEP'
03F8			
03F8	EF	RST CALRUF	; AUFRUF CALCULATOR
03F9			; UM DIE WERTE FÜR TONHÖHE
03F9			; (P BZW I=INT(P)) UND DIE
03F9			; ZEIT T ZU BERECHNEN
03F9	31	.BYT \$31	
03FA	27	.BYT \$27	
03FB	C0	.BYT \$C0	
03FC	03	.BYT \$03	
03FD	34	.BYT \$34	
03FE	EC	.BYT \$EC	
03FF	6C	.BYT \$6C	
0400	9B	.BYT \$9B	

```

0401 1F          .BYT $1F
0402 F5          .BYT $F5
0403 04          .BYT $04
0404 A1          .BYT $A1
0405 0F          .BYT $0F
0406 3B          .BYT $3B
0407
0407 21 92 5C    LD HL, MEMBOT
040A 7E          LD A, (HL)      ; EXPONENT VON I HOLEN
040B A7          AND A
040C 20 5E       JR NZ, ERRTON   ; NICHT NULL: ERROR
040E 23          INC HL
040F 4E          LD C, (HL)      ; VORZEICHENBYTE IN C
0410 23          INC HL
0411 46          LD B, (HL)      ; LOW-BYTE IN B UND A
0412 7B          LD A, B
0413 17          RLA
0414 9F          SBC A           ; TEST OB -128 <= I <= +127
0415 B9          CP C
0416 20 54       JR NZ, ERRTON   ; FALLS NICHT: ERROR
0418 23          INC HL
0419 BE          CP (HL)
041A 20 50       JR NZ, ERRTON
041C
041C 7B          LD A, B         ; LOWBYTE IN A
041D C6 3C       ADD $3C         ; -60 BIS +67
041F F2 25 04    JP P, $425      ; AKTZEPTIEREN
0422 E2 6C 04    JP PO, ERRTON   ; OVERFLOWERROR: -128 BIS -61
0425
0425            ; OKTAVE DES TONS IN B SUCHEN
0425
0425 06 FA       LD B, $FA       ; 6 OKTAVEN UNTER MITTLEREM C
0427 04          OKTAV INC B     ; BEGINNEN
0428 D6 0C       SUB 12          ; PRO OKTAVE 12 TÖNE ABZIEHEN
042A 30 FB       JR NC, OKTAV
042C C6 0C       ADD 12          ; A: ZEIGER AUF HALBTON
042E            ; DIESER OKTAVE
042E C5          PUSH BC         ; NUMMER DER OKTAVE RETTEN
042F 21 6E 04    LD HL, TONC     ; BASISADRESSE DER HALBTÖNE
0432 CD 06 34    CALL $3406      ; HALBTON IN A IN DEN
0435 CD B4 33    CALL $33B4      ; CALCULATOR STACK BRINGEN
0438
0438 EF          RST CALRUF      ; UND DIE ENDGÜLTIGE
0439 04          .BYT $04        ; TONHÖHE BERECHNEN
043A 3B          .BYT $3B
043B
043B F1          POP AF          ; OKTAVNUMMER HOLEN UND
043C 86          ADD (HL)        ; AUF EXPONENT ADDIEREN * 2
043D 77          LD (HL), A
043E
043E EF          RST CALRUF
043F C0          .BYT $C0        ; FREQUENZ IN MEMO SPEICHERN
0440 02          .BYT $02
0441 31          .BYT $31
0442 3B          .BYT $3B
0443
0443 CD 94 1E    CALL $1E94      ; ZEIT MUSS <= 10 SEIN
0446 FE 0B      CP $0B

```

```

0448 30 22      JR NC,ERRTON      ; SONST ERROR
044A           ;
044A EF        RST CALRUF        ; BERECHNUNG FREQUENZ * ZEIT
044B E0        .BYT $E0          ; ERMITTELTE FREQUENZ HOLEN
044C 04        .BYT $04          ; T * F BILDEN
044D E0        .BYT $E0
044E 34        .BYT $34
044F 80        .BYT $80
0450 43        .BYT $43
0451 55        .BYT $55
0452 9F        .BYT $9F
0453 80        .BYT $80
0454 01        .BYT $01
0455 05        .BYT $05
0456 34        .BYT $34
0457 35        .BYT $35
0458 71        .BYT $71
0459 03        .BYT $03
045A 38        .BYT $38
045B           ;
045B CD 99 1E   CALL $1E99        ; ZEITSCHLEIFENWERT HOLEN
045E C5        PUSH BC           ; ZWISCHENSPEICHERN
045F CD 99 1E   CALL $1E99        ; WERT FREQUENZ * ZEIT NACH BC
0462 E1        POP HL
0463 50        LD D,B           ; NACH DE ANZAHL
0464 59        LD E,C           ; DER DURCHLAUFE
0465           ;
0465 7A        LD A,D           ; FALLS DE = 0,
0466 B3        OR E
0467 CB        RET Z            ; DIREKT RETURN,
0468           ;
0468 1B        DEC DE           ; SONST DE -1 UND
0469 C3 B5 03   JP PIEPEN        ; ZUR TONAUSGABE
046C           ;
046C CF        ERRTON RST ERR AUS
046D 0A        .BYT $0A
046E           ;
046E           ; HALBTONTABELLE
046E           ;
046E 89        TONC .BYT $89,$02,$D0,$12,$86
046F 02
0470 D0
0471 12
0472 86
0473 89        TONCIS .BYT $89,$0A,$97,$60,$75
0474 0A
0475 97
0476 60
0477 75
0478 89        TOND .BYT $89,$12,$D5,$17,$1F
0479 12
047A D5
047B 17
047C 1F
047D 89        TONDIS .BYT $89,$1B,$90,$41,$02
047E 1B
047F 90
0480 41

```


0481	02	
0482	89	TONE .BYT \$89,\$24,\$D0,\$53,\$CA
0483	24	
0484	D0	
0485	53	
0486	CA	
0487	89	TONF .BYT \$89,\$2E,\$9D,\$36,\$B1
0488	2E	
0489	9D	
048A	36	
048B	B1	
048C	89	TONFIS .BYT \$89,\$3B,\$FF,\$49,\$3E
048D	3B	
048E	FF	
048F	49	
0490	3E	
0491	89	TONG .BYT \$89,\$43,\$FF,\$6A,\$73
0492	43	
0493	FF	
0494	6A	
0495	73	
0496	89	TONGIS .BYT \$89,\$4F,\$A7,\$00,\$54
0497	4F	
0498	A7	
0499	00	
049A	54	
049B	89	TONA .BYT \$89,\$5C,\$00,\$00,\$00
049C	5C	
049D	00	
049E	00	
049F	00	
04A0	89	TOMB .BYT \$89,\$69,\$14,\$F6,\$24
04A1	69	
04A2	14	
04A3	F6	
04A4	24	
04A5	89	TONH .BYT \$89,\$76,\$F1,\$10,\$05
04A6	76	
04A7	F1	
04A8	10	
04A9	05	
04AA		;
04AA		; DIESE ROUTINE WIRD NICHT BENUTZT
04AA		;
04AA	CD FB 24	CALL \$24FB
04AD	3A 3B 5C	LD A,(FLAGS)
04B0	B7	ADD A
04B1	FA BA 1C	JP M,\$1CBA
04B4	E1	POP HL
04B5	D0	RET NC
04B6	E5	PUSH HL
04B7	CD F1 2B	CALL \$2BF1
04BA	62	LD H,D
04BB	6B	LD L,E
04BC	0D	DEC C
04BD	FB	RET M
04BE	09	ADD HL,BC
04BF	CB FE	SET 7,(HL)

```

04C1 C9 RET
04C2 ; -----
04C2 ;
04C2 ; KASSETTENREKORDER UNTERPROGRAMME
04C2 ;
04C2 ; BYTEABSPEICHERROUTINE
04C2 ;
04C2 21 3F 05 LD HL,SAVLOA ; RETURNADRESSE FÜR
04C5 E5 PUSH HL
04C6 21 80 1F LD HL,$1F80 ; KONSTANTE FÜR 5 SEKUNDEN
04C9 ; HEADERTON
04C9 CB 7F BIT 7,A
04CB 28 03 JR Z,$4D0 ; HEADER SPEICHERN
04CD 21 98 0C LD HL,$C98 ; KONSTANTE FÜR 2 SEKUNDEN
04D0 0B EX AF,AF ; PROGRAMM/HEADER-FLAG RETTEN
04D1 13 INC DE
04D2 DD 2B DEC IX
04D4 F3 DI ; INTERRUPT SPERREN
04D5 3E 02 LD A,2 ; BORDERCOLOUR ROT UND 'MIC' EIN
04D7 47 LD B,A
04D8 ;
04D8 10 FE HEADER DJNZ HEADER ; BITPERIODE HEADERTON
04DA D3 FE OUT ($FE),A ; 'MIC' EIN/AUSSCHALTEN
04DC EE 0F XOR $F
04DE 06 A4 LD B,$A4 ; KONSTANTE BITPERIODE
04E0 2D DEC L
04E1 20 F5 JR NZ,HEADER
04E3 05 DEC B
04E4 25 DEC H ; WIEDERHOLUNG, BIS HL
04E5 F2 DB 04 JP P,HEADER ; KLEINER NULL WIRD
04E8 ;
04E8 ; SYNCHRONPULS SENDEN
04E8 ;
04E8 06 2F LD B,$2F
04EA 10 FE SYNC1 DJNZ SYNC1 ; 'MIC' AUS
04EC D3 FE OUT ($FE),A ; 'MIC' EIN UND ROT
04EE 3E 0D LD A,$D ; FÜR 'MIC' AUS UND 'CYAN'
04F0 06 37 LD B,$37
04F2 10 FE SYNC2 DJNZ SYNC2
04F4 D3 FE OUT ($FE),A ; 'MIC' AUS UND 'CYAN'
04F6 ;
04F6 01 0E 3B LD BC,$3B0E
04F9 0B EX AF,AF ; FLAG ZURÜCKHOLEN
04FA 6F LD L,A ; UND NACH L ZUM ABSPEICHERN
04FB C3 07 05 JP $507
04FE ;
04FE 7A BYTEAU LD A,D ; DE AUF NULL TESTEN
04FF B3 OR E
0500 2B 0C JR Z,$50E ; BYTES ENDE UND "PARITY" SENDEN
0502 DD 6E 00 LD L,(IX+0) ; SONST NÄCHSTES BYTE
0505 7C LD A,H ; "PARITY"-BILDUNG IN A
0506 AD XOR L ; MIT DEM AKTUELLEN BYTE
0507 67 LD H,A ; UND IN H RETTEN
0508 3E 01 LD A,1 ; 'MIC' EIN UND 'BLAU'
050A 37 SCF ; CARRY ALS ENDEMARKIERUNG
050B C3 25 05 JP SAVEB ; FÜR 8 BITS SETZEN UND SENDEN
050E ;
050E 6C LD L,H ; ENDE-"PARITY" NACH L

```

```

050F 18 F4      JR $505
0511           ;
0511 79      BITEND LD A,C      ; TEIL2 DES BITS SENDEN UND
0512 CB 78      BIT 7,B      ; HIERFÜR ZERO-FLAG SETZEN
0514           ;
0514 10 FE      BITANF DJNZ BITANF      ; ZEITSCHLEIFE
0516 30 04      JR NC,BITOUT      ; WENN NULLBIT
0518 06 42      LD B,$42
051A 10 FE      BITSET DJNZ BITSET      ; BEI 1 WEITER VERZÖGERN
051C D3 FE      BITOUT OUT ($FE),A      ; TEIL1: 'MIC' EIN UND 'BLAU'
051E           ; TEIL2: 'MIC' AUS UND 'GELB'
051E 06 3E      LD B,$3E
0520 20 EF      JR NZ,BITEND      ; TEIL2 AUSGEBEN
0522 05      DEC B
0523 AF      XOR A      ; CARRY LÖSCHEN UND
0524 3C      INC A      ; 'MIC' EIN MIT 'BLAU'
0525           ; BEIM ERSTEN BIT EINES BYTE IST CARRY GESETZT.
0525           ; SONST IMMER GELÖSCHT. DIES DIENT BEIM ROTIEREN
0525           ; DES L-REG ALS BEGRENZUNG AUF 8 BITS.
0525 CB 15      SAVEB RL L      ; BIT 7 DER AUSGABE INS CARRY
0527 C2 14 05      JP NZ,BITANF      ; UND WEITER BIS 8 BITS FERTIG
052A           ;
052A 1B      DEC DE      ; BYTEZÄHLER -1
052B DD 23      INC IX      ; ADRESSE NÄCHSTES BYTE
052D 06 31      LD B,$31
052F 3E 7F      LD A,$7F
0531 DB FE      IN A,($FE)      ; TEST DER 'BREAK'-TASTE
0533 1F      RRA
0534 D0      RET NC      ; 'BREAK' GEDRÜCKT
0535 7A      LD A,D
0536 3C      INC A
0537 C2 FE 04      JP NZ,BYTEAU      ; AUSGABE BIS DE = $FFFF IST
053A 06 3B      LD B,$3B
053C 10 FE      SAVEDE DJNZ SAVEDE      ; ETWAS VERZÖGERN
053E C9      RET
053F           ;
053F           ; NACH SAVE ODER LOAD HIER HIN ZURÜCKKEHREN
053F           ;
053F F5      SAVLOA PUSH AF      ; CARRY RETTEN
0540 3A 4B 5C      LD A,(BORDCR)      ; ORIGINAL BORDER COLOUR
0543 E6 3B      AND $3B      ; NEHMEN UND IN
0545 0F      RRCA      ; BITS 0..2
0546 0F      RRCA
0547 0F      RRCA
0548 D3 FE      OUT ($FE),A      ; BORDER ORIGINAL
054A 3E 7F      LD A,$7F      ; BREAKTASTE PRÜFEN
054C DB FE      IN A,($FE)
054E 1F      RRA
054F FB      EI      ; INTERRUPT FREIGEBEN
0550 3B 02      JR C,$554      ; BREAK NICHT GEDRÜCKT
0552           ;
0552 CF      RST ERR AUS      ; SONST MELDUNG
0553 0C      .BYT $0C
0554           ;
0554 F1      POP AF      ; FLAGS ZURÜCKHOLEN
0555 C9      RET
0556           ;
0556           ; DIESE ROUTINE WIRD BEI 'LOAD' UND 'VERIFY' AUFGERUFEN

```

```

0556      ;
0556 14      INC D      ; ZERO FLAG LÖSCHEN
0557 08      EX AF,AF   ; UND RETTEN
0558 15      DEC D      ; D WIEDER ORIGINAL
0559 F3      DI         ; INTERRUPT ABSCHALTEN
055A 3E 0F   LD A,$0F   ; BORDER COLOUR WEISS
055C 03 FE   OUT ($FE),A
055E 21 3F 05 LD HL,SAVLOA ; RETURNADRESSE
0561 E5      PUSH HL    ; AUF STACK LEGEN
0562 DB FE   IN A,($FE) ; EINMAL ZUM INITIALISIEREN LESEN
0564 1F      RRA
0565 E6 20   AND $20    ; UND DAS 'EAR'-BIT MERKEN
0567 F6 02   OR 2       ; BORDER COLOUR ROT
0569 4F      LD C,A     ; RETTEN ($22 = AUS, $02 = EIN)
056A BF      CP A       ; UM RET NZ ZU ÜBERSPRINGEN
056B      ;
056B C0      LOABRK RET NZ ; EXIT BEI BREAK
056C CD E7 05 LOABEG CALL FLANK1 ; CARRY GELÖSCHT: KEINE FLANKE
056F 30 FA   JR NC,LOABRK ; GEFUNDEN ODER BREAK GEDRÜCKT
0571      ;
0571 21 15 04 LD HL,$415
0574 10 FE   LDWART DJNZ LDWART ; WARTESCHLEIFE ETWA 1 SEKUNDE
0576 2B      DEC HL
0577 7C      LD A,H
057B B5      OR L
0579 20 F9   JR NZ,LDWART
057B CD E3 05 CALL FLANK2 ; NOCH MAL AUF 2 FLANKEN WARTEN
057E 30 EB   JR NC,LOABRK ; FALLS NICHT: ERROR
0580      ;
0580      ; NUR EIN HEADERSIGNAL AKTZEPTIEREN
0580      ;
0580 06 9C   HEADIN LD B,$9C ; ZEITKONSTANTE
0582 CD E3 05 CALL FLANK2 ; ZWEI FLANKEN ERWARTEN
0585 30 E4   JR NC,LOABRK ; FALLS NICHT: ERROR
0587 3E C6   LD A,$C6 ; MAXIMALER ZEITABSTAND
0589 B8      CP B       ; CIRCA 3000 T ZYKLEN
058A 30 E0   JR NC,LOABEG ; SONST NOCH MAL SUCHEN
058C 24      INC H      ; 256 DOPPELFLANKEN ABWARTEN
058D 20 F1   JR NZ,HEADIN
058F      ;
058F      ; JETZT WIRD EIN SYNCHRONIMPULS ERWARTET
058F      ;
058F 06 C9   SYNWAR LD B,$C9 ; ZEITKONSTANTE
0591 CD E7 05 CALL FLANK1
0594 30 D5   JR NC,LOABRK
0596 78      LD A,B      ; ZWEI KURZ AUF EINANDERFOLGENDE
0597 FE D4   CP $D4      ; FLANKEN BILDEN DEN SYNC-IMPULS
0599 30 F4   JR NC,SYNWAR
059B CD E7 05 CALL FLANK1 ; ABFALLENDE SYNC-FLANKE
059E D0      RET NC      ; MUSS EXISTIEREN
059F      ;
059F      ; DIE BYTES KÖNNEN GELADEN ODER VERIFIZIERT WERDEN
059F      ;
059F 79      LD A,C      ; BORDER COLOURS AUF BLAU
05A0 EE 03   XOR 3       ; BZW. GELB SCHALTEN
05A2 4F      LD C,A
05A3 26 00   LD H,0      ; FÜR "PARITY"-PRÜFUNG
05A5 06 B0   LD B,$B0    ; ZEITKONSTANTE FÜR 'FLAG'-BYTE

```


05A7	1B 1F	JR \$5CB	
05A9			
05A9	0B	EX AF,AF	; FLAGS ZURÜCKHOLEN
05AA	20 07	JR NZ,\$5B3	; NUR BEI FLAGPRÜFUNG (1. BYTE)
05AC	30 0F	JR NC,VERIFY	
05AE	DD 75 00	LD (IX+0),L	; GELADENES BYTE SPEICHERN
05B1	1B 0F	JR LADWEI	; UND DAS NACHSTE LADEN
05B3	CB 11	LOAFLG RL C	; RETTE CARRY
05B5	AD	XOR L	; ERSTES BYTE = TYP-FLAG ?
05B6	CO	RET NZ	; NEIN ERROR
05B7			
05B7	79	LD A,C	; SONST CARRY WIEDERHOLEN
05B8	1F	RR A	
05B9	4F	LD C,A	
05BA	13	INC DE	; DIESES INKREMENT WIRD UNTEN
05BB	1B 07	JR \$5C4	; WIEDER RÜCKGANGIG GEMACHT
05BD			
05BD	DD 7E 00	VERIFY LD A,(IX+0)	; BEI VERIFY ORIGINALBYTE
05C0	AD	XOR L	; HOLEN UND VERGLEICHEN
05C1	CO	RET NZ	; VERIFY-ERROR (CARRY GELÖSCHT)
05C2			
05C2	DD 23	LADWEI INC IX	; ADRESSE FÜR LOAD/VERIFY ERHÖHEN
05C4	1B	DEC DE	; ZÄHLER -1
05C5	0B	EX AF,AF	; FLAGS RETTEN
05C6	06 B2	LD B,\$B2	; ZEITKONSTANTE LADEN
05C8	2E 01	LD L,1	; ENDEKENNUNG BEIM 9. SCHIEBEN
05CA			
05CA	CD E3 05	BITHOL CALL FLANK2	; LÄNGE DER PULSE HOLEN
05CD	D0	RET NC	; ZEITÜBERSCHREITUNG
05CE	3E CB	LD A,\$CB	; ZEITVERGLEICH, UM DARAUSS
05D0	B8	CP B	; EINE '0' ODER '1' IM CARRY ZU
05D1	CB 15	RL L	; GEWINNEN UND IN L ZU SCHIEBEN
05D3	06 B0	LD B,\$B0	; ZEITKONSTANTE NACHSTES BIT
05D5	D2 CA 05	JP NC,BITHOL	; BYTE NOCH NICHT KOMPLETT
05D8			
05D8	7C	LD A,H	; "PARITY" HOLEN
05D9	AD	XOR L	; UND NACHSTES BILDEN
05DA	67	LD H,A	; NEUES "PARITY" NACH H
05DB	7A	LD A,D	
05DC	B3	OR E	
05DD	20 CA	JR NZ,\$5A9	; ENDE NOCH NICHT ERREICHT
05DF	7C	LD A,H	; LETZTES "PARITY"-BYTE MUSS
05E0	FE 01	CP 1	; NULL SEIN: CARRY GESETZT
05E2	C9	RET	; SONST CARRY GELÖSCHT; ERROR
05E3			
05E3			; DIESES UNTERPROGRAMM ERFASST PEGELWECHSEL (FLANKE)
05E3			; AM KASSETTENREKORDER-EINGANG. REG B DIENT ALS
05E3			; ZEITZÄHLER. INNERHALB DIESER FESTGELEGTE ZEIT
05E3			; MÜSSEN DIE GEFORDERTEN FLANKEN AUFTRETEN, SONST
05E3			; ERROR MIT CARRY GELÖSCHT
05E3			
05E3	CD E7 05	FLANK2 CALL FLANK1	; AUF ZWEI FLANKEN PRÜFEN
05E6	D0	RET NC	; RETURN BEI ERROR
05E7			
05E7	3E 16	FLANK1 LD A,\$16	; EINE FLANKE PRÜFEN
05E9	3D	WARTLD DEC A	; ETWAS WARTEN
05EA	20 FD	JR NZ,WARTLD	
05EC	A7	AND A	; CARRY LÖSCHEN

05ED	04	FLANKE	INC B	; ZEITZÄHLER ERHÖHEN
05EE	CB		RET Z	; ZEITENDE: CARRY + ZERO GESETZT
05EF				
05EF	3E 7F		LD A,\$7F	
05F1	DB FE		IN A,(\$FE)	; BREAKTASTE UND 'EAR'
05F3	1F		RRA	; BREAK GEDRÜCKT:
05F4	D0		RET NC	; CARRY GELÖSCHT UND NICHT ZERO
05F5				
05F5	A9		XOR C	; IST EINE FLANKE AUFGETRETEN ?
05F6	E6 20		AND \$20	
05F8	28 F3		JR Z,FLANKE	; NEIN: WEITER WARTEN
05FA				
05FA				; INNERHALB DER ERLAUBTEN ZEIT WURDE EINE FLANKE ERKANNT
05FA				
05FA	79		LD A,C	
05FB	2F		CPL	; IN C FLANKENART MERKEN
05FC	4F		LD C,A	
05FD	E6 07		AND 7	; BORDER COLOUR AUSBLENDEN
05FF	F6 08		OR 8	; 'MIC' AUS
0601	D3 FE		OUT (\$FE),A	
0603	37		SCF	; CARRY SETZEN: OK
0604	C9		RET	
0605				
0605				; EINSPRUNG BEI 'LOAD', 'SAVE', 'VERIFY' UND 'MERGE'
0605				; UNTERSCHIEDUNG DER BEFEHLE GESCHIEHT MITTELS 'TADDR'
0605				
0605	F1	KASHAU	POP AF	; RETURNADRESSE VERNICHTEN
0606	3A 74 5C		LD A,(TADDR)	; \$E0 ABZIEHEN, UM 0 FÜR SAVE,
0609	D6 E0		SUB \$E0	; 1 FÜR LOAD, 2 FÜR VERIFY
060B	32 74 5C		LD (TADDR),A	; UND 3 FÜR MERGE ZU ERHALTEN
060E	CD BC 1C		CALL \$1C8C	; NAMENSPARAMETER IN CALC.-STACK
0611	CD 30 25		CALL \$2530	; SYNTAXPRÜFUNG ?
0614	28 3C		JR Z,\$652	; SPRUNG, FALLS NUR SYNTAXPRÜFUNG
0616	01 11 00		LD BC,17	; LÄNGE DES NAMENS: 17
0619	3A 74 5C		LD A,(TADDR)	
061C	A7		AND A	
061D	28 02		JR Z,\$621	; BEI SAVE
061F	0E 22		LD C,34	; SONST LÄNGE 34
0621	F7	KASPAC	RST \$30	; SPEICHER RESERVIEREN
0622	D5		PUSH DE	; STARTADRESSE IN IX
0623	DD E1		POP IX	; BRINGEN
0625	06 0B		LD B,11	
0627	3E 20		LD A,0	
0629	12	CLRNAM	LD (DE),A	; FILENAME LÖSCHEN
062A	13		INC DE	
062B	10 FC		DJNZ \$0629	
062D				
062D	DD 36 01 FF		LD (IX+1),255	; ZEIGT 'KEIN NAME' AN
0631	CD F1 2B		CALL \$2BF1	; NAMENSPARAMETER HOLEN
0634	21 F6 FF		LD HL,\$FFF6	; -10 LADEN
0637	0B		DEC BC	
0638	09		ADD HL,BC	; LÄNGE AUF 10 PRÜFEN
0639	03		INC BC	
063A	30 0F		JR NC,\$64B	; <= 10
063C	3A 74 5C		LD A,(TADDR)	; FALLS NICHT SAVE:
063F	A7		AND A	
0640	20 02		JR NZ,NAMOK	; MEHR ZULASSEN
0642				

0642	CF	NAMERR	RST ERR AUS	; FEHLERMELDUNG:
0643	OE		.BYT \$OE	; 'INVALID FILE NAME'
0644				
0644	7B	NAMOK	LD A,B	
0645	B1		OR C	
0646	2B 0A		JR Z,\$652	; OHNE NAMENSANGABE
0648	01 0A 00		LD BC,10	; NAME AUF 10 ZEICHEN BEGRENZEN
0648	DD E5		PUSH IX	; ADRESSE DES NAMENS
064D	E1		PDP HL	; INS HL REG
064E	23		INC HL	; UND EINS ERHÖHEN
064F	EB		EX DE,HL	; POINTERTAUSCH
0650	ED B0		LDIR	; DEN NAMEN UMSPEICHERN
0652				
0652	DF		RST GETAKT	; NÄCHSTES ZEICHEN IM INPUT LESEN
0653	FE E4		CP \$E4	; TOKEN 'DATA' ?
0655	20 49		JR NZ,\$6A0	; NEIN
0657	3A 74 5C		LD A,(TADDR)	; AUF MERGE PRÜFEN
065A	FE 03		CP 3	
065C	CA 8A 1C		JP Z,\$1C8A	; 'MERGE' MIT 'DATA': ERROR
065F	E7		RST GETNXT	; NOCH EIN ZEICHEN HOLEN
0660	CD B2 28		CALL \$28B2	; IN VARIABLENTABELLE SUCHEN
0663	CB F9		SET 7,C	; BIT 7 DES ARRAYNAMENS SETZEN
0665	30 0B		JR NC,\$672	; BEI EINEM VORHANDENEN ARRAY
0667	21 00 00		LD HL,0	; SONST NEUES ARRAY
066A	3A 74 5C		LD A,(TADDR)	
066D	3D		DEC A	
066E	2B 15		JR Z,\$685	; NUR 'LOAD' ZUGELASSEN
0670				
0670	CF		RST ERR AUS	; SONST ERROR:
0671	01		.BYT \$01	; 'VARIABLE NOT FOUND'
0672				
0672	C2 8A 1C		JP NZ,\$1C8A	
0675	CD 30 25		CALL \$2530	; BEI SYNTAXPRÜFUNG
0678	2B 18		JR Z,\$692	; SPRUNG
067A	23		INC HL	; SONST LANGENBYTE LOW
067B	7E		LD A,(HL)	
067C	DD 77 0B		LD (IX+\$B),A	
067F	23		INC HL	; UND HIGH IN DEN
0680	7E		LD A,(HL)	; WORKSPACE
0681	DD 77 0C		LD (IX+\$C),A	; UMSPEICHERN
0684	23		INC HL	
0685	DD 71 0E		LD (IX+\$E),C	; ARRAYNAME EBENFALLS SPEICHERN
0688	3E 01		LD A,1	; DEFAULT: ZAHLENARRAY
068A	CB 71		BIT 6,C	
068C	2B 01		JR Z,\$68F	; SPRUNG, WENN RICHTIG
068E	3C		INC A	; SONST BUCHSTABENARRAY
068F	DD 77 00		LD (IX+0),A	; IM ERSTEN HEADERBYTE MERKEN
0692	EB		EX DE,HL	; HL RETTEN
0693	E7		RST GETNXT	; EIN ZEICHEN HOLEN UND
0694	FE 29		CP ')'	; AUF KLAMMER PRÜFEN
0696	20 DA		JR NZ,\$672	; WENN NICHT: ERROR C
0698	E7		RST GETNXT	; CHADD+1, NÄCHSTES ZEICHEN
0699	CD EE 1B		CALL \$1BEE	; AUSSPRUNG BEI SYNTAXPRÜFUNG
069C	EB		EX DE,HL	; HL ZURÜCKHOLEN
069D	C3 5A 07		JP \$75A	; UND WEITER
06A0				
06A0	FE AA		CP \$AA	; IST TOKEN = 'SCREENS' ?
06A2	20 1F		JR NZ,\$6C3	; SPRUNG, WENN NICHT

06A4	3A 74 5C	LD A,(TADDR)	
06A7	FE 03	CP 3	; BEI MERGE IST 'SCREEN\$' ALS
06A9	CA 8A 1C	JP Z,\$1C8A	; NAME NICHT ZUGELASSEN
06AC	E7	RST GETNXT	; EIN ZEICHEN HOLEN
06AD	CD EE 1B	CALL \$1BEE	; AUSSPRUNG BEI SYNTAXPRÜFUNG
06B0	DD 36 0B 00	LD (IX+\$B),0	; LÄNGE DES DISPLAY- UND
06B4	DD 36 0C 1B	LD (IX+\$C),\$1B	; ATTRIBUTBEREICHS IST \$1B00
06B8	21 00 40	LD HL,\$4000	; UND DIE STARTADRESSE
06BB	DD 75 0D	LD (IX+\$D),L	; IST \$4000
06BE	DD 74 0E	LD (IX+\$E),H	
06C1	1B 4D	JR \$710	
06C3			
06C3	FE AF	CP \$AF	; IST TOKEN = 'CODE' ?
06C5	20 4F	JR NZ,\$716	; WENN NICHT
06C7	3A 74 5C	LD A,(TADDR)	; BEI MERGE IST 'CODE'
06CA	FE 03	CP 3	
06CC	CA 8A 1C	JP Z,\$1C8A	; NICHT ZUGELASSEN
06CF	E7	RST GETNXT	; EIN ZEICHEN HOLEN
06D0	CD 4B 20	CALL \$204B	; PRÜFEN, OB EINGABEBEENDE
06D3	20 0C	JR NZ,\$6E1	; NEIN
06D5	3A 74 5C	LD A,(TADDR)	; BEI SAVE IST 'CODE'
06D8	A7	AND A	
06D9	CA 8A 1C	JP Z,\$1C8A	; NICHT ZUGELASSEN
06DC	CD E6 1C	CALL \$1CE6	; EINE NULL AUF DEN CALCULATOR-
06DF	1B 0F	JR \$6F0	; STACK SCHREIBEN FÜR 'START'
06E1			
06E1			; STARTADRESSE SUCHEN
06E1			
06E1	CD 82 1C	CALL \$1C82	; DIE ERSTE ZAHL HOLEN
06E4	DF	RST GETAKT	; EIN ZEICHEN HOLEN UND AUF
06E5	FE 2C	CP ','	; KOMMA PRÜFEN
06E7	2B 0C	JR Z,\$6F5	; JA, ZAHL WAR STARTADRESSE
06E9	3A 74 5C	LD A,(TADDR)	; ERROR, FALLS BEI
06EC	A7	AND A	; 'SAVE' NUR DIE STARTADRESSE
06ED	CA 8A 1C	JP Z,\$1C8A	; UND KEINE LÄNGE ANGEBEN
06F0	CD E6 1C	CALL \$1CE6	; NULL ALS LÄNGE AUF CALCULATOR-
06F3	1B 04	JR \$6F9	; STACK SCHREIBEN
06F5			
06F5	E7	RST GETNXT	; EIN ZEICHEN UND
06F6	CD 82 1C	CALL \$1C82	; DIE LÄNGE HOLEN
06F9			
06F9	CD EE 1B	CALL \$1BEE	; WEITER ZUM NÄCHSTEN BEFEHL,
06FC			; FALLS SYNTAXPRÜFUNG
06FC	CD 99 1E	CALL \$1E99	; LÄNGE INS BC REGISTER BRINGEN
06FF	DD 71 0B	LD (IX+\$B),C	; UND IM HEADER
0702	DD 70 0C	LD (IX+\$C),B	; SPEICHERN
0705	CD 99 1E	CALL \$1E99	; EBENSO MIT DER
0708	DD 71 0D	LD (IX+\$D),C	; STARTADRESSE
070B	DD 70 0E	LD (IX+\$E),B	; VERFAHREN
070E	60	LD H,B	; STARTADRESSE ALS POINTER
070F	69	LD L,C	; INS HL REGISTER
0710	DD 36 00 03	LD (IX+0),3	; TYP '3' FÜR SCREEN\$ UND CODE
0714	1B 44	JR \$75A	
0716			
0716	FE CA	CP \$CA	; IST TOKEN = 'LINE' ?
0718	2B 09	JR Z,\$723	; JA
071A	CD EE 1B	CALL \$1BEE	; NÄCHSTER BEFEHL BEI SYNTAXPR.
071D	DD 36 0E 80	LD (IX+\$E),\$80	; FALLS KEINE WEITEREN PARAMETER

```

0721 18 17          JR $73A
0723
0723 3A 74 5C        LD A,(TADDR)      ; PRÜFEN, OB 'SAVE', DENN EINE
0726 A7              AND A              ; ZEILENNUMMER MUSS FOLGEN
0727 C2 8A 1C        JP NZ,$1C8A       ; NICHT 'SAVE'
072A E7              RST GETNXT         ; EIN ZEICHEN HOLEN
072B CD 82 1C        CALL $1C82         ; ZEILENNUMMER AUF CALC-STACK
072E CD EE 1B        CALL $1BEE        ; BEI SYNTAXPRÜFUNG EXIT
0731 CD 99 1E        CALL $1E99        ; SONST ZEILENNUMMER IN BC
0734 DD 71 0D        LD (IX+$D),C      ; BRINGEN UND
0737 DD 70 0E        LD (IX+$E),B      ; ABSPEICHERN
073A DD 36 00 00     LD (IX+0),0       ; 'LINE' UND 'OHNE WEITERE AN-
073E                                     ; GABEN' SIND VOM TYP '0'
073E
073E 2A 59 5C        LD HL,(ELINE)     ; ZEIGER AUF VARIABLENENDE
0741 ED 5B 53 5C     LD DE,(PROG)      ; ZEIGER AUF BASICPROGRAMMANFANG
0745 37              SCF
0746 ED 52           SBC HL,DE          ; LÄNGE VON PROGRAMM UND
0748 DD 75 0B        LD (IX+$B),L      ; VARIABLEN AUSRECHNEN UND
074B DD 74 0C        LD (IX+$C),H      ; ABSPEICHERN
074E 2A 4B 5C        LD HL,(VARS)      ; NUR DIE PROGRAMMLÄNGE
0751 ED 52           SBC HL,DE          ; BERECHNEN UND
0753 DD 75 0F        LD (IX+$F),L      ; ABSPEICHERN
0756 DD 74 10        LD (IX+$10),H
0759 EB              EX DE,HL           ; PROGRAMMANFANG NACH HL
075A
075A                ; DER HEADER IST NUN FERTIG:
075A                ; IX+ 0: TYP
075A                ; IX+ 1 BIS +10: NAME ODER IX+1=$FF, WENN KEIN NAME
075A                ; IX+11/12: LÄNGE
075A                ; IX+13 BIS IX+16: VERSCHIEDENE ANGABEN
075A
075A 3A 74 5C        LD A,(TADDR)      ; PRÜFEN, OB 'SAVE'
075D A7              AND A
075E CA 70 09        JP Z,$970         ; JA
0761
0761 E5              PUSH HL            ; POINTER FÜR 'LOAD' ETC. RETTEN
0762 01 11 00        LD BC,17          ; IN IX DIE ADRESSE DES
0765 DD 09           ADD IX,BC          ; ZWEITER HEADERS BILDEN
0767 DD E5           PUSH IX            ; UND ZWISCHENSPEICHERN
0769 11 11 00        LD DE,17          ; 17 BYTES LADEN
076C AF              XOR A              ; 'HEADER' ANMERKEN
076D 37              SCF                ; 'LOAD' ANMERKEN
076E CD 56 05        CALL $556         ; HEADER LADEN
0771 DD E1           POP IX             ; ZWEITE HEADERADRESSE ZURÜCK
0773 30 F2           JR NC,$767        ; UND IN DER SCHLEIFE BLEIBEN,
0775                                     ; BIS HEADER GEFUNDEN
0775 3E FE           LD A,$FE          ; KANAL S ÖFFNEN
0777 CD 01 16        CALL $1601
077A FD 36 52 03     LD (IY+$52),3     ; SCROLLING-ZÄHLER SETZEN
077E 0E 80           LD C,$80          ; DEFAULT FÜR HEADER FALSCH
0780 DD 7E 00        LD A,(IX+0)       ; DIE BEIDEN HEADER VERGLEICHEN
0783 DD BE EF        CP (IX+$EF)
0786 20 02           JR NZ,$78A        ; WAR NOCH NICHT RICHTIG
0788 0E F6           LD C,$F6          ; ES MÜSSEN 10 ZEICHEN STIMMEN
078A FE 04           CP 4              ; TYP > 4 IST UNSINN
078C 30 D9           JR NC,$767        ; NOCH MAL HEADER LADEN
078E

```


078E 11 C0 09	LD DE,\$9C0	; BASISADRESSE DER MELDUNGEN
0791 C5	PUSH BC	; BC ZWISCHENSPEICHERN
0792 CD 0A 0C	CALL \$C0A	; DEN TYP AUF SCHIRM AUSGEBEN
0795 C1	POP BC	; BC WIEDER HOLEN
0796 DD E5	PUSH IX	; DE AUF DEN GELADENEN HEADER
0798 D1	POP DE	; ZEIGEN LASSEN
0799 21 F0 FF	LD HL,\$FFFF	; HL AUF DEN ERSTEN HEADER
079C 19	ADD HL,DE	; ZEIGEN LASSEN
079D 06 0A	LD B,10	; 10 ZEICHEN PRÜFEN
079F 7E	LD A,(HL)	
07A0 3C	INC A	; WAR EIN NAME ANGEZEIGT ?
07A1 20 03	JR NZ,\$7A6	; JA
07A3 79	LD A,C	; FALLS NICHT, STIMMT
07A4 80	ADD B	; DER NAME IMMER!
07A5 4F	LD C,A	; (\$F6 + \$0A = 0)
07A6		
07A6 13	INC DE	
07A7 1A	LD A,(DE)	; ERSTES ZEICHEN VERGLEICHEN
07A8 BE	CP (HL)	; UND AUF SCHIRM AUSGEBEN
07A9 23	INC HL	
07AA 20 01	JR NZ,\$7AD	; HEADER STIMMEN NICHT
07AC 0C	INC C	; SONST ZÄHLER +1
07AD D7	RST PRTOUT	; AUSGABE, BIS
07AE 10 F6	DJNZ \$7A6	; B AUF NULL GEZÄHLT
07B0 CB 79	BIT 7,C	; WENN HEADER STIMMT, IST C
07B2 20 B3	JR NZ,\$767	; POSITIV, SONST NOCH MAL HEADER
07B4		; HOLEN
07B4		
07B4		; WEITER NUR, WENN DER RICHTIGE HEADER GEFUNDEN WURDE
07B4 3E 0D	LD A,\$D	; EIN CARRIAGE RETURN
07B6 D7	RST PRTOUT	; AUSGEBEN
07B7 E1	POP HL	; POINTER ZURÜCK
07B8 DD 7E 00	LD A,(IX+0)	; 'SCREEN\$' ODER
07BB FE 03	CP 3	; 'CODE'
07BD 28 0C	JR Z,\$7CB	; JA
07BF 3A 74 5C	LD A,(TADDR)	; 'LOAD'-BEFEHL ?
07C2 3D	DEC A	
07C3 CA 08 08	JP Z,\$808	; JA
07C6 FE 02	CP 2	; 'MERGE'-BEFEHL
07C8 CA 86 08	JP Z,\$8B6	; JA
07C8		
07C8		; VERIFY-ROUTINE
07C8		
07C8 E5	PUSH HL	; POINTER RETTEN
07CC DD 6E FA	LD L,(IX+\$FA)	; LÄNGE IN HL
07CF DD 66 FB	LD H,(IX+\$FB)	; LADEN
07D2 DD 5E 0B	LD E,(IX+\$B)	; LÄNGE DES GELADENEN
07D5 DD 56 0C	LD D,(IX+\$C)	; HEADERS HOLEN
07D8 7C	LD A,H	; WAR LÄNGENANGABE
07D9 B5	OR L	; DER EINGABE NULL ?
07DA 28 0D	JR Z,\$7E9	; JA
07DC ED 52	SBC HL,DE	; FALLS NEUE LÄNGE GRÖßER ALS
07DE 38 26	JR C,\$806	; ALTE, DANN ERROR
07E0		
07E0 28 07	JR Z,\$7E9	; GLEICHE LÄNGE IST IN ORDNUNG
07E2 DD 7E 00	LD A,(IX+0)	; BEI VERIFY MÜSSEN DIE LÄNGEN
07E5 FE 03	CP 3	; ÜBEREINSTIMMEN

```

07E7 20 1D      JR NZ,$806      ; SONST ERROR
07E9           ;
07E9 E1         POP HL          ; STARTPOINTER HOLEN UND
07EA 7C         LD A,H          ; AUF NULL PRÜFEN
07EB B5         OR L
07EC 20 06      JR NZ,$7F4
07EE DD 6E 0D   LD L,(IX+$D)    ; WENN START = 0, DANN WIRD
07F1 DD 66 0E   LD H,(IX+$E)    ; DER STARTPOINTER DES GELADENEN
07F4 E5         PUSH HL         ; HEADERS BENUTZT
07F5 DD E1      POP IX          ; STARTPOINTER NACH IX
07F7 3A 74 5C   LD A,(TADDR)    ; 'LOAD' ODER 'VERIFY' UNTER-
07FA FE 02      CP 2            ; SCHEIDEN
07FC 37         SCF             ; CARRY GESETZT = 'LOAD'
07FD 20 01      JR NZ,$800      ; 'LOAD'
07FF A7         AND A           ; CARRY GELÖSCHT: 'VERIFY'
0800 3E FF      LD A,$FF        ; DATENBYTES ANMERKEN
0802           .END
0802           .LIB SPEC0800-S
0802           ; SINCLAIR ZX SPECTRUM TEIL 0800
0802           ;
0802           ; SUBROUTINAUFRUF FÜR ALLE LADEVORGÄNGE, EGAL OB LOAD
0802           ; VERIFY ODER MERGE
0802           ;
0802 CD 56 05    CALL $556
0805 DB         RET C           ; RETURN, FALLS KEIN FEHLER
0806           ;
0806 CF         RST ERR AUS      ; MELDUNG:
0807 1A         .BYT $1A        ; 'TAPE LOADING ERROR'
0808           ;
0808           ; 'LOAD'-BEFEHLROUTINE
0808           ;
0808 DD 5E 0B    LD E,(IX+$B)    ; LÄNGE AUS DEM GELADENEN
080B DD 56 0C    LD D,(IX+$C)    ; HEADER HOLEN
080E E5         PUSH HL         ; ZIELPOINTER RETTEN
080F 7C         LD A,H          ; HL IST FÜR EIN NOCH NICHT
0810 B5         OR L            ; DEKLARIERTES ARRAY = 0
0811 20 06      JR NZ,$819      ; TRIFFT NICHT ZU
0813 13         INC DE          ; DE +3 FÜR NAME UND LÄNGE
0814 13         INC DE
0815 13         INC DE
0816 EB         EX DE,HL
0817 1B 0C      JR $825
0819           ;
0819 DD 6E FA    LD L,(IX+$FA)    ; LÄNGE DES VORHANDENEN
081C DD 66 FB    LD H,(IX+$FB)    ; PROGRAMMS MIT VARIABLEN
081F EB         EX DE,HL        ; PRÜFEN, OB
0820 37         SCF             ; ZUSÄTZLICHER SPEICHERPLATZ
0821 ED 52      SBC HL,DE        ; BENÖTIGT WIRD
0823 3B 09      JR C,$82E        ; NEIN
0825           ;
0825 11 05 00    LD DE,5         ; 5 BYTES MEHR
082B 19         ADD HL,DE
0829 44         LD B,H          ; IN BC BRINGEN UND
082A 4D         LD C,L
082B CD 05 1F    CALL $1F05      ; SPEICHERPLATZTEST DURCHFÜHREN
082E           ;
082E E1         POP HL          ; STARTADRESSE HOLEN
082F DD 7E 00    LD A,(IX+0)    ; PRÜFEN, OB EIN

```

0832	A7	AND A	; BASICPROGRAMM GELADEN WIRD
0833	28 3E	JR Z,\$873	; JA
0835	7C	LD A,H	
0836	B5	OR L	; NEUES ARRAY ?
0837	28 13	JR Z,\$84C	; JA
0839	2B	DEC HL	; SONST LANGE DES BEREITS
083A	46	LD B,(HL)	; EXISTENTEN ARRAYS HOLEN
083B	2B	DEC HL	
083C	4E	LD C,(HL)	
083D	2B	DEC HL	; HL ZEIGT AUF ALTEN NAMEN
083E	03	INC BC	; LANGE +3 FÜR NAMEN
083F	03	INC BC	; UND LANGE
0840	03	INC BC	
0841	DD 22 5F 5C	LD (XPTR),IX	; IX ZWISCHENSPEICHERN
0845	CD EB 19	CALL RAUS2	; ALTES ARRAY WEGWERFEN
0848	DD 2A 5F 5C	LD IX,(XPTR)	; IX WIEDER HOLEN
084C			
084C	2A 59 5C	LD HL,(ELINE)	; HL AUF ENDEMARKIERUNG (\$80)
084F	2B	DEC HL	; DER VARIABLEN SETZEN
0850	DD 4E 0B	LD C,(IX+\$B)	; LANGE DES NEUEN
0853	DD 46 0C	LD B,(IX+\$C)	; ARRAYS IN BC UND
0856	C5	PUSH BC	; ZWISCHENSPEICHERN
0857	03	INC BC	; BC + 3 FÜR NAME UND LANGE
0858	03	INC BC	
0859	03	INC BC	
085A	DD 7E FD	LD A,(IX+\$FD)	; NAME AUS ALTEM HEADER FÜR DAS
085D	F5	PUSH AF	; NEUE ARRAY HOLEN UND RETTEN
085E	CD 55 16	CALL \$1655	; 'BC' SPEICHERPLATZE BESCHAFFEN
0861	23	INC HL	; DEN NAMEN
0862	F1	POP AF	; DES ARRAYS
0863	77	LD (HL),A	; EINSCHREIBEN
0864	D1	POP DE	; LANGE EBENFALLS
0865	23	INC HL	; HOLEN UND
0866	73	LD (HL),E	
0867	23	INC HL	
0868	72	LD (HL),D	; EINSCHREIBEN
0869	23	INC HL	; HL ZEIGT AUF DEN ERSTEN PLATZ,
086A			; DER VOM BAND GELADEN WIRD
086A	E5	PUSH HL	; STARTADRESSE NACH
086B	DD E1	POP IX	; IX BRINGEN
086D	37	SCF	; 'LOAD' UND
086E	3E FF	LD A,\$FF	; DATEN ANMERKEN
0870	C3 02 0B	JP \$802	; ZUR LADEROUTINE
0873			
0873			; LADEN EINES BASICPROGRAMMS MIT VARIABLEN
0873			
0873	EB	EX DE,HL	; ZIELADRESSE ZWISCHENSPEICHERN
0874	2A 59 5C	LD HL,(ELINE)	; VARIABLENENDE SUCHEN
0877	2B	DEC HL	
0878	DD 22 5F 5C	LD (XPTR),IX	; ZWISCHENSPEICHERN
087C	DD 4E 0B	LD C,(IX+\$B)	; LANGE DES NEUEN
087F	DD 46 0C	LD B,(IX+\$C)	; HEADERS HOLEN,
0882	C5	PUSH BC	; ZWISCHENSPEICHERN UND DAS
0883	CD E5 19	CALL RAUS1	; DERZEITIGE PROGRAMM WEGWERFEN
0886	C1	POP BC	; LANGE WIEDER HOLEN
0887	E5	PUSH HL	; ZEIGER AUF START UND
0888	C5	PUSH BC	; DIE LANGE ZWISCHENSPEICHERN
0889	CD 55 16	CALL \$1655	; BC SPEICHERPLATZE FREIMACHEN

088C	DD 2A 5F 5C	LD IX, (XPTR)	; IX WIEDER HOLEN
0890	23	INC HL	; VARIABLE VARS
0891	DD 4E 0F	LD C, (IX+\$F)	; MUSS NEU
0894	DD 46 10	LD B, (IX+\$10)	;
0897	09	ADD HL, BC	;
0898	22 4B 5C	LD (VARS), HL	; GESETZT WERDEN
089B	DD 66 0E	LD H, (IX+\$E)	; ZEILENNUMMER ANGEZEIGT
089E	7C	LD A, H	;
089F	E6 C0	AND \$C0	;
08A1	20 0A	JR NZ, \$BAD	; NEIN
08A3	DD 6E 0D	LD L, (IX+\$D)	; SONST NEWPPC UND
08A6	22 42 5C	LD (NEWPPC), HL	;
08A9	FD 36 0A 00	LD (IY+\$A), 0	; NSPPC NEU SETZEN
08AD			;
08AD	D1	POP DE	; LANGE UND
08AE	DD E1	POP IX	; STARTADRESSE HOLEN
08B0	37	SCF	; 'LOAD' UND
08B1	3E FF	LD A, \$FF	; DATEN ANMERKEN
08B3	C3 02 0B	JP \$B02	; ZUR LADEROUTINE
08B6			;
08B6		; MERGE-ROUTINE	
08B6			;
08B6	DD 4E 0B	LD C, (IX+\$B)	; LANGE DES
08B9	DD 46 0C	LD B, (IX+\$C)	; DATENBLOCKS HOLEN
08BC	C5	PUSH BC	; UND ZWISCHENSPEICHERN
08BD	03	INC BC	; LANGE +1 UND BC SPEICHERPLATZE
08BE	F7	RST \$30	; IN WORKSPACE BESCHAFFEN
08BF	36 80	LD (HL), \$80	; EINE ENDMARKIERUNG SETZEN
08C1	EB	EX DE, HL	; STARTADRESSE NACH HL
08C2	D1	POP DE	; LANGE NACH DE
08C3	E5	PUSH HL	; START ZWISCHENSPEICHERN
08C4	E5	PUSH HL	; UND ZUSÄTZLICH NACH
08C5	DD E1	POP IX	; IX BRINGEN
08C7	37	SCF	; LADEN UND
08C8	3E FF	LD A, \$FF	; DATEN ANMERKEN
08CA	CD 02 0B	CALL \$B02	; LADEROUTINE AUFRUFEN
08CD	E1	POP HL	; START NEU ZURÜCK IN HL
08CE	ED 5B 53 5C	LD DE, (PROG)	; DE ZEIGT AUF START ALT
08D2			;
08D2		; DIE NEUEN ZEILEN WERDEN IN DAS ALTE PROGRAMM EINGEFÜGT	
08D2			;
08D2	7E	LD A, (HL)	; PRÜFEN, OB
08D3	E6 C0	AND \$C0	; FERTIG
08D5	20 19	JR NZ, \$BF0	; JA
08D7			;
08D7	1A	LD A, (DE)	; ZEILENNUMMER HIGH VERGLEICHEN
08D8	13	INC DE	; UND BEIDE POINTER +1
08D9	BE	CP (HL)	;
08DA	23	INC HL	;
08DB	20 02	JR NZ, \$BDF	; NICHT GLEICH
08DD	1A	LD A, (DE)	; ZEILENNUMMER LOW VERGLEICHEN
08DE	BE	CP (HL)	;
08DF	1B	DEC DE	; BEIDE POINTER WIEDER
08E0	2B	DEC HL	; ORIGINAL
08E1	30 0B	JR NC, \$BEB	; PLATZ FÜR NEUE ZEILE GEFUNDEN
08E3	E5	PUSH HL	; SONST START DER NÄCHSTEN
08E4	EB	EX DE, HL	; ZEILE SUCHEN
08E5	CD 8B 19	CALL \$198B	;

08E8	E1	POP HL	
08E9	18 EC	JR \$8D7	; IM ALTEN PROGRAMM WEITERSUCHEN
08EB			
08EB	CD 2C 09	CALL \$92C	; NEUE ZEILE EINFÜGEN
08EE	18 E2	JR \$8D2	; UND WEITER SUCHEN
08F0			
08F0			; AB HIER NEUE VARIABLEN EINFÜGEN
08F0			
08F0	7E	LD A, (HL)	; VARIABLENNAME HOLEN
08F1	4F	LD C, A	
08F2	FE 80	CP \$80	; FERTIG ?
08F4	C8	RET Z	; JA
08F5			
08F5	E5	PUSH HL	; AKTUELLEN NEUEN POINTER RETTEN
08F6	2A 4B 5C	LD HL, (VARS)	; ALTEN POINTER HOLEN
08F9	7E	LD A, (HL)	; VARIABLENNAME UND
08FA	FE 80	CP \$80	; -ENDE PRÜFEN
08FC	28 25	JR Z, \$923	; ENDE EREICHT
08FE	B9	CP C	; STIMMT NAME ?
08FF	28 08	JR Z, \$909	; JA
0901	C5	PUSH BC	; VARIABLENNAME RETTEN
0902	CD 88 19	CALL \$1988	; NÄCHSTE ALTE VARIABLE SUCHEN
0905	C1	POP BC	; NAME ZURÜCKHOLEN
0906	EB	EX DE, HL	; POINTER WIEDER RICHTIG
0907	18 F0	JR \$8F9	; UND WEITERSUCHEN
0909			
0909	E6 E0	AND \$E0	; LÄNGER VARIABLENNAME ?
090B	FE A0	CP \$A0	
090D	20 12	JR NZ, \$921	; NEIN
090F	D1	POP DE	; DE ZEIGT AUF DEN ERSTEN BUCH-
0910	D5	PUSH DE	; STABEN DES NEUEN NAMENS
0911	E5	PUSH HL	; RETTE POINTER AUF ALTEN NAMEN
0912			
0912	23	INC HL	; POINTER +1, DA DAS ERSTE
0913	13	INC DE	; BYTE SCHON GEPRÜFT WURDE
0914	1A	LD A, (DE)	; DEN REST DES NAMENS
0915	BE	CP (HL)	; VERGLEICHEN
0916	20 06	JR NZ, \$91E	; NICHT GEFUNDEN
0918	17	RLA	; LETZTES ZEICHEN ?
0919	30 F7	JR NC, \$912	; NEIN
091B	E1	POP HL	; ADRESSE DES ALTEN NAMENS
091C	18 03	JR \$921	; UND ERSETZEN
091E			
091E	E1	POP HL	
091F	18 E0	JR \$901	; WEITERSUCHEN
0921			
0921	3E FF	LD A, \$FF	; ANMERKEN: VARIABLE ERSETZEN
0923			; A = \$80: VARIABLE HINZUFÜGEN
0923	D1	POP DE	; ADRESSE 'NEU' HOLEN
0924	EB	EX DE, HL	; POINTER RICHTIG SETZEN
0925	3C	INC A	; ZEROFLAG FÜR ERSETZEN = 1
0926	37	SCF	; ANMERKEN: VARIABLENBEHANDLUNG
0927	CD 2C 09	CALL \$92C	; UND DIE VARIABLE EINTRAGEN
092A	18 C4	JR \$8F0	; NÄCHSTE VARIABLE UNTERSUCHEN
092C			
092C			; SUBROUTINE ZUM EINFÜGEN EINER ZEILE ODER VARIABLEN
092C			; BEI 'MERGE'
092C			

092C	20 10	JR NZ,\$93E	; HINZUFÜGEN
092E	08	EX AF,AF	; FLAGS RETTEN
092F	22 5F 5C	LD (XPTR),HL	; 'NEU'-POINTER RETTEN
0932	EB	EX DE,HL	
0933	CD 88 19	CALL \$19B8	; NACHSTE ZEILE/VARIABLE SUCHEN
0936	CD EB 19	CALL RAUS2	; ALTE ZEILE/VARIABLE ENTFERNEN
0939	EB	EX DE,HL	; POINTER UND
093A	2A 5F 5C	LD HL,(XPTR)	
093D	08	EX AF,AF	; FLAGS WIEDER HOLEN
093E	08	EX AF,AF	; FLAGS WIEDER RETTEN
093F	D5	PUSH DE	; ZIELADRESSE SPEICHERN
0940	CD 88 19	CALL \$19B8	; NACHSTE ZEILE/VARIABLE UND
0943			; DEREN LANGE SUCHEN
0943	22 5F 5C	LD (XPTR),HL	; ZEILE/VARIABLE 'NEU' RETTEN
0946	2A 53 5C	LD HL,(PROG)	; PROG ZWISCHENSPEICHERN
0949	E3	EX (SP),HL	; UND POINTER 'NEU' HOLEN
094A	C5	PUSH BC	; LANGE RETTEN
094B	08	EX AF,AF	; FLAG ZURÜCK
094C	38 07	JR C,\$955	; SPRUNG, WENN VARIABLE NEU
094E	28	DEC HL	; NEUE ZEILE VOR DER ZIELADRESSE
094F	CD 55 16	CALL \$1655	; DEN PLATZ FREIMACHEN
0952	23	INC HL	; KORREKTUR
0953	18 03	JR \$958	
0955			
0955	CD 55 16	CALL \$1655	; PLATZ FÜR NEUE VARIABLE MACHEN
0958	23	INC HL	; ERSTE FREIE STELLE
0959	C1	POP BC	; LANGE HOLEN
095A	D1	POP DE	; PROG HOLEN UND
095B	ED 53 53 5C	LD (PROG),DE	; ZURÜCKSCHREIBEN
095F	ED 58 5F 5C	LD DE,(XPTR)	; NEUEN POINTER HOLEN
0963	C5	PUSH BC	; LANGE UND
0964	D5	PUSH DE	; POINTER 'NEU' RETTEN
0965	EB	EX DE,HL	; POINTER RICHTIG SETZEN ZUM
0966	ED 80	LDIR	; KOPIEREN DER ZEILE/VARIABLE
0968	E1	POP HL	; POINTER 'NEU'
0969	C1	POP BC	; LANGE DER ZEILE/VARIABLEN
096A	D5	PUSH DE	; RETTE POINTER 'ALT'
096B	CD EB 19	CALL RAUS2	; ZEILE/VARIABLE AUS WORKSPACE
096E	D1	POP DE	; ENTFERNEN, POINTER 'ALT'
096F	C9	RET	; ZURÜCKHOLEN UND FERTIG
0970			
0970		; SUBROUTINE FÜR SAVE	
0970			
0970	E5	PUSH HL	; POINTER RETTEN
0971	3E FD	LD A,\$FD	; KANAL K ÖFFNEN
0973	CD 01 16	CALL \$1601	
0976	AF	XOR A	; A = 0 FÜR ERSTE MELDUNG
0977	11 A1 09	LD DE,\$9A1	; ADRESSE DER KASSETTENMELDUNGEN
097A	CD 0A 0C	CALL \$C0A	; AUSGABE 'START TAPE'
097D	FD CB 02 EE	SET 5,(1Y+2)	; MERKEN: BILDSCHIRM LÖSCHEN
0981	CD 04 15	CALL \$15D4	; AUF EINEN TASTENDRUCK WARTEN
0984	DD E5	PUSH IX	; ADRESSE DES HEADERS RETTEN
0986	11 11 00	LD DE,17	; 17 BYTES HEADER MÜSSEN
0989	AF	XOR A	; (A = HEADER)
098A	CD C2 04	CALL \$4C2	; ABGESPEICHERT WERDEN
098D	DD E1	POP IX	; HEADERADRESSE ZURÜCK
098F	06 32	LD B,30	; B MIT 30 FÜR
0991	76	HALT	; ZEITVERZÖGERUNG VON EINER

```

0992 10 FD          DJNZ $991          ; SEKUNDE LADEN UND WARTEN
0994 DD 5E 0B       LD E, (IX+$B)      ; IN DE DIE LÄNGE DES ABZU-
0997 DD 56 0C       LD D, (IX+$C)      ; SPEICHERNDEN DATENBLOCKS LADEN
099A 3E FF          LD A, $FF          ; $FF = DATENBLOCK
099C DD E1          POP IX              ; ANFANG DES BLOCKES IN IX
099E C3 C2 04       JP $4C2            ; UND ZUM ABSPEICHERN
09A1                ;
09A1                ; MELDUNGEN FÜR KASSETTENREKORDER-BETRIEB
09A1                ;
09A1 80             KASMEI .BYT $80
09A2 53 74          .BYT 'Start tape, then press any key', $AE
09C0 AE
09C1 0D             .BYT $0D, 'Program:', $A0
09C2 50 72
09CA A0
09CB 0D             .BYT $0D, 'Number array:', $A0
09CC 4E 75
09D9 A0
09DA 0D             .BYT $0D, 'Character array:', $A0
09DB 43 68
09EB A0
09EC 0D             .BYT $0D, 'Bytes:', $A0
09ED 42 79
09F3 A0
09F4
09F4                ;
09F4                ; -----
09F4                ; AUSGABEROUTINEN FÜR BILDSCHIRM UND DRUCKER
09F4                ;
09F4                ; REG A ENTHÄLT JEWELNS DAS AUSZUGEBENDE ZEICHEN, EIN
09F4                ; TOKEN ODER EIN STEUERZEICHEN
09F4                ;
09F4 CD 03 0B       AUSGAB CALL POSHOL ; PRINTPOSITION HOLEN
09F7 FE 20          CP ' '             ; ALLE DRUCKBAREN ZEICHEN UND
09F9 D2 D9 0A       JP NC, PRTCHA      ; TOKENS
09FC FE 06          CP 6                ; FÜR $00 - $05 ? DRUCKEN
09FE 3B 69          JR C, PRTFRA
0A00 FE 18          CP $18              ; DESGLEICHEN FÜR $18 - $1F
0A02 30 65          JR NC, PRTFRA
0A04 21 0B 0A       LD HL, CONTRO-6    ; ADRESSE DER CONTROL-ZEICHEN-
0A07 5F             LD E, A             ; TABELLE -6, DA NUR $06 BIS $17
0A08 16 00          LD D, 0            ; ZUGELASSEN
0A0A 19             ADD HL, DE          ; OFFSET IN TABELLE BERECHNEN
0A0B 5E             LD E, (HL)          ; OFFSET HOLEN,
0A0C 19             ADD HL, DE          ; FÜR RETURN-ADRESSE ADDIEREN
0A0D E5             PUSH HL             ; UND AUF DEN STACK SCHIEBEN
0A0E C3 03 0B       JP POSHOL          ; PRINTPOSITION HOLEN
0A11                ;
0A11                ; TABELLE DER OFFSETS FÜR CONTROL-ZEICHEN
0A11                ;
0A11 4E             CONTRO .BYT $4E     ; PRINT KOMMA
0A12 57             .BYT $57            ; EDIT
0A13 10             .BYT $10            ; CURSOR NACH LINKS
0A14 29             .BYT $29            ; CURSOR NACH RECHTS
0A15 54             .BYT $54            ; CURSOR NACH UNTEN
0A16 53             .BYT $53            ; CURSOR NACH OBEN
0A17 52             .BYT $52            ; DELETE
0A18 37             .BYT $37            ; ENTER

```

```

0A19 50          .BYT $50          ; NICHT BENUTZT
0A1A 4F          .BYT $4F          ; NICHT BENUTZT
0A1B 5F          .BYT $5F          ; INK
0A1C 5E          .BYT $5E          ; PAPER
0A1D 5D          .BYT $5D          ; FLASH
0A1E 5C          .BYT $5C          ; BRIGHT
0A1F 5B          .BYT $5B          ; INVERSE
0A20 5A          .BYT $5A          ; OVER
0A21 54          .BYT $54          ; AT
0A22 53          .BYT $53          ; TAB
0A23             ;
0A23             ; CURSOR NACH LINKS
0A23             ; B ENTHÄLT DIE ZEILENNUMMER UND C DIE SPALTENPOSITION
0A23             ; DIE ZAHLEN FÜR ZEILENNUMMER WERDEN VON UNTEN UND
0A23             ; DIE SPALTENPOSITION VON RECHTS GEZÄHLT
0A23             ;
0A23 0C          INC C              ; 1 NACH LINKS ZÄHLEN
0A24 3E 22       LD A,34            ; LINKER RAND ERREICHT ?
0A26 B9          CP C
0A27 20 11       JR NZ,$A3A        ; NEIN
0A29 FD CB 01 4E BIT 1,(IY+1)      ; PRINTERAUSGABE ?
0A2D 20 09       JR NZ,$A3B        ; JA
0A2F 04          INC B              ; SONST ZEILE 1 HÖHER
0A30 0E 02       LD C,2            ; SPALTE AUF 2 SETZEN
0A32 3E 18       LD A,24           ; BILDSCHIRMOBERKANTE ERREICHT ?
0A34 BB          CP B
0A35 20 03       JR NZ,$A3A        ; NEIN
0A37 05          DEC B              ; JA, ZEILE WIE VORMER UND
0A38 0E 21       LD C,33           ; AUF ERSTE SPALTE SETZEN
0A3A C3 D9 0D    JP $DD9
0A3D             ;
0A3D             ; CURSOR EINS NACH RECHTS
0A3D             ; BC WIE OBEN BEI EINSTIEG
0A3D             ; DIESE ROUTINE ENTSPRICHT IN BASIC: PRINT OVER1;CHR$32;
0A3D             ;
0A3D 3A 91 5C    LD A,(PFLAG)      ; PFLAG RETTEN
0A40 F5          PUSH AF
0A41 FD 36 57 01 LD (IY+$57),1     ; PFLAG AUF OVER 1 SETZEN
0A45 3E 20       LD A,
0A47 CD 65 0B    CALL $B65          ; LEERZEICHEN AUSGEBEN
0A4A F1          POP AF             ; UND PFLAG ZURÜCKHOLEN
0A4B 32 91 5C    LD (PFLAG),A
0A4E C9          RET
0A4F             ;
0A4F             ; BEHANDLUNG VON EINEM CARRIAGE RETURN (CR):
0A4F             ; BEI PRINTERAUSGABE WIRD DER PRINTERBUFFER AUSGEGEBEN.
0A4F             ; BEI BILDSCHIRMAUSGABE WIRD ERST GETESTET, OB DER
0A4F             ; BILDSCHIRM NOCH OBEN GEROLLT (SCROLL) WERDEN MUSS
0A4F             ;
0A4F FD CB 01 4E BIT 1,(IY+1)      ; PRINTER ?
0A53 C2 CD 0E    JP NZ,$ECD        ; JA, ZUR BUFFERAUSGABE
0A56             ;
0A56 0E 21       LD C,33           ; LINKEN RAND SETZEN
0A5B CD 53 0C    CALL $C55         ; EVTL. SCROLLING DURCHFÜHREN
0A5B 05          DEC B              ; ZEILE 1 NACH UNTEN
0A5C C3 D9 0D    JP $DD9
0A5F             ;
0A5F             ; 'PRINT KOMMA' SUBROUTINE:

```

```

0A5F      ; TABAUSGABE: SPALTE 0 ODER 16
0A5F      ;
0A5F  CD 03 0B      CALL POSHOL      ; ZEILE UND SPALTE IN BC HOLEN
0A62  79            LD A,C
0A63  3D            DEC A            ; 2 SPALTEN NACH RECHTS
0A64  3D            DEC A
0A65  E6 10         AND $10         ; 0 ODER 16 DARAUS MACHEN
0A67  18 5A         JR TABFIL       ; UND NÖTIGE SPACES AUSGEBEN
0A69      ;
0A69  3E 3F         PRTFRA LD A,'?'  ; NICHT DRUCKBARE ZEICHEN
0A6B  18 6C         JR PRTCHA       ; DURCH FRAGEZEICHEN ERSETZEN
0A6D      ;
0A6D      ; STEUERZEICHEN MIT OPERANDEN BEHANDELN
0A6D      ; BEI 2 OPERANDEN (AT UND TAB): EINSTIEG $0A75
0A6D      ; EIN OPERAND (INK BIS OVER): EINSTIEG $0A7A
0A6D      ; STEUERZEICHEN IMMER IN TVDATA MERKEN, BEI 2 OPERANDEN
0A6D      ; DEN ERSTEN IN TVDATA+1 MERKEN (AUSGABE AUF $0A6D)
0A6D      ;
0A6D  11 87 0A     ZWEIOP LD DE,EINOP
0A70  32 0F 5C     LD (TVDATA+1),A ; ERSTEN OPERANDEN SPEICHERN
0A73  18 0B         JR AUSSET
0A75      ;
0A75      ; EINSTIEG BEI AT UND TAB
0A75      ;
0A75  11 6D 0A     LD DE,ZWEIOP    ; AUSGABE ÄNDERN
0A7B  18 03         JR $A7D
0A7A      ;
0A7A  11 87 0A     LD DE,EINOP     ; AUSGABE ÄNDERN UND STEUER-
0A7D  32 0E 5C     LD (TVDATA),A   ; ZEICHEN IN TVDATA SPEICHERN
0A80      ;
0A80  2A 51 5C     AUSSET LD HL,(CURCHL) ; AKTIVEN AUSGABEKANAL
0A83  73           LD (HL),E       ; NEU SETZEN
0A84  23           INC HL
0A85  72           LD (HL),D
0A86  C9           RET
0A87      ;
0A87  11 F4 09     EINOP LD DE,AUSGAB ; OPERANDEN GEHOLT UND AUSGABE
0A8A  CD 80 0A     CALL AUSSET     ; WIEDER NORMAL
0A8D  2A 0E 5C     LD HL,(TVDATA)  ; STEUERZEICHEN UND OP1
0A90  57           LD D,A          ; LETZTEN OPERAND RETTEN
0A91  7D           LD A,L          ; STEUERZEICHEN:
0A92  FE 16        CP $16          ; INK BIS OVER ?
0A94  DA 11 22     JP C,$2211      ; JA
0A97  20 29        JR NZ,PRTTAB    ; SPRUNG BEI TAB
0A99      ;
0A99      ; BEHANDLUNG VON AT
0A99      ;
0A99  44           LD B,H          ; ZEILENNUMMER (ERSTER OP)
0A9A  4A           LD C,D          ; SPALTENNUMMER
0A9B  3E 1F        LD A,31         ; DA RÜCKWÄRTS GEZÄHLT WIRD,
0A9D  91           SUB C           ; ENTSPRECHEND UMRECHNEN
0A9E  3B 0C        JR C,$AAC       ; FALLS BEREICHSÜBERSCHREITUNG
0AA0  C6 02        ADD 2           ; 2 (OFFSET) ADDIEREN, DAMIT
0AA2  4F           LD C,A          ; C 2-33 ENTHÄLT
0AA3  FD CB 01 4E  BIT 1,(IY+1)    ; PRINTERAUSGABE ?
0AA7  20 16        JR NZ,$ABF     ; JA: LPRINT AT
0AA9      ;
0AA9  3E 16        LD A,22         ; ZEILENNUMMER UMRECHNEN

```

0AAB	90	SUB B	
0AAC	DA 9F 1E	JP C,\$1E9F	; ERROR: INTEGER OUT OF RANGE
0AAF			
0AAF	3C	INC A	; 2 ALS OFFSET ADDIEREN
0AB0	47	LD B,A	
0AB1	04	INC B	; UND IN B BRINGEN
0AB2	FD CB 02 46	BIT 0,(IY+2)	; UNTERER TEIL
0AB6	C2 55 0C	JP NZ,\$C55	; JA, EVTL. SCROLLING DURCHFÜHREN
0AB9	FD BE 31	CP (IY+\$31)	; INNERHALB DES BILDSCHIRMS
0ABC	DA 86 0C	JP C,\$C86	; NEIN: OUT OF SCREEN
0ABF	C3 D9 0D	JP \$DD9	; JA: RESTLICHE PARAMETER SETZEN
0AC2			
0AC2		; TAB-AUSFÜHRUNG	
0AC2			
0AC2	7C	PRTTAB LD A,H	; ERSTER OPERAND
0AC3	CD 03 0B	TABFIL CALL POSHOL	; POSITION HOLEN
0AC6	81	ADD C	; SPALTE ADDIEREN
0AC7	3D	DEC A	; SPACESANZAHL BERECHNEN (0-31),
0AC8	E6 1F	AND \$1F	; DIE AUSZUGEBEN SIND
0ACA	C8	RET Z	; FALLS KEINE SPACE
0ACB			
0ACB	57	LD D,A	; D ZUM ZÄHLEN BENUTZEN
0ACC	FD CB 01 C6	SET 0,(IY+1)	; FÜHRENDEN SPACE UNTERDRÜCKEN
0AD0	3E 20	PRTSPA LD A,	; DIE LEERZEICHEN
0AD2	CD 3B 0C	CALL \$C3B	; AUSGEBEN
0AD5	15	DEC D	; ZÄHLER -1
0AD6	20 FB	JR NZ,\$AD0	; NOCH NICHT FERTIG
0ADB	C9	RET	
0AD9			
0AD9		; DRUCKBARE ZEICHEN BZW. TOKENS AUSGEBEN	
0AD9			
0AD9	CD 24 0B	PRTCHA CALL \$B24	
0ADC			
0ADC		; NEUE ZEILEN- UND SPALTENNUMMER SOWIE PIXEL-ADRESSE	
0ADC		; IN DEN SYSTEMVARIABLEN SETZEN	
0ADC			
0ADC	FD CB 01 4E	NEUSTO BIT 1,(IY+1)	; PRINTER ?
0AE0	20 1A	JR NZ,\$AFC	; JA
0AE2			
0AE2	FD CB 02 46	BIT 0,(IY+2)	; BEI UNTEREM BILDSCHIRMTEIL
0AE6	20 08	JR NZ,\$AF0	; SPRINGEN
0AEB	ED 43 88 5C	LD (SPOSN),BC	; WERTE DES HAUPTTEILS SPEICHERN
0AEC	22 84 5C	LD (DFCC),HL	
0AEF	C9	RET	
0AF0	ED 43 8A 5C	LD (SPOSNL),BC	; WERTE UNTERER TEIL SPEICHERN
0AF4	ED 43 82 5C	LD (ECHOE),BC	
0AFB	22 86 5C	LD (DFCCL),HL	
0AFB	C9	RET	
0AFC			
0AFC	FD 71 45	LD (IY+\$45),C	; PRINTER BUFFER POINTER
0AFF	22 80 5C	LD (PRCC),HL	; NEU SETZEN
0B02	C9	RET	
0B03			
0B03		; BILDSCHIRM- ODER PRINTERPOSITIONEN IN BC UND HL HOLEN	
0B03	FD CB 01 4E	POSHOL BIT 1,(IY+1)	; PRINTER ?
0B07	20 14	JR NZ,\$B1D	; JA
0B09	ED 4B 88 5C	LD BC,(SPOSN)	; HAUPTSCHIRMPARAMETER
0B0D	2A 84 5C	LD HL,(DFCC)	; LADEN


```

OB10 FD CB 02 46      BIT 0, (IY+2)
OB14 C8               RET Z           ; RETURN BEI HAUPTBILDSCHIRM
OB15 ED 4B BA 5C      LD BC, (SPOSNL) ; SONST UNTERE SCHIRMPARAMETER
OB19 2A 86 5C         LD HL, (DFCCL) LADEN
OB1C C9              RET
OB1D FD 4E 45         LD C, (IY+$45) ; DIE PRINTERBUFFERWERTE
OB20 2A 80 5C         LD HL, (PRCC)   ; HOLEN
OB23 C9              RET
OB24                  ;
OB24                  ; ZEICHEN ODER TOKENS AUSGEBEN
OB24                  ;
OB24 FE 80            CP $80           ; NORMALES ZEICHEN ($20 - $7F) ?
OB26 38 3D            JR C, $B65       ; JA
OB28 FE 90            CP $90
OB2A 30 26            JR NC, $B52      ; ALLE TOKENS UND ZEICHEN > $8F
OB2C 47              LD B, A           ; HIER NUR $80 BIS $8F
OB2D CD 38 0B         CALL $B38       ; GRAFIKZEICHEN GENERIEREN
OB30 CD 03 0B         CALL POSHOL     ; HL IST VERMURKST: WIEDER HOLEN
OB33 11 92 5C         LD DE, MEMBOT
OB36 18 47            JR $B7F
OB38                  ;
OB38                  ; AUS DEN BITS 0-3 WIRD DAS GRAFIKZEICHEN GENERIERT
OB38                  ;
OB38 21 92 5C         LD HL, MEMBOT   ; RAMADRESSE ZUM ZWISCHEN-
OB38                  ; SPEICHERN DES GRAFIKZEICHENS
OB38 CD 3E 0B         CALL $B3E       ; 2 * AUFRUFEN
OB3E CB 18            RR B             ; BIT 0 ODER 2 AUSWERTEN
OB40 9F              SBC A            ; A ENTHALT $0F, FALLS BIT
OB41 E6 0F            AND $0F         ; GESETZT WAR, SONST $00
OB43 4F              LD C, A          ; IN C ZWISCHENSPEICHERN
OB44 CB 18            RR B             ; BIT 1 ODER 3 AUSWERTEN
OB46 9F              SBC A            ; WIE VORHER, NUR JETZT
OB47 E6 F0            AND $F0         ; $F0 ODER $00
OB49 B1              OR C             ; MIT NIEDEREM ERGEBNIS ODERN
OB4A 0E 04            LD C, 4         ; BITMUSTER 4 * BENUTZEN
OB4C 77              LD (HL), A       ; SPEICHERN
OB4D 23              INC HL           ; POINTER +1
OB4E 0D              DEC C            ; ZÄHLER -1
OB4F 20 FB            JR NZ, $B4C     ; NOCH NICHT 4 *
OB51 C9              RET
OB52                  ;
OB52                  ; TOKEN CODES UND UDG
OB52                  ;
OB52 06 A5            PRTOU SUB $A5
OB54 30 09            JR NC, $B5F     ; WENN TOKEN CODES
OB56 C6 15            ADD $15         ; UDG'S JETZT $00 BIS $0F
OB58 C5              PUSH BC          ; POSITION RETTEN
OB59 ED 4B 7B 5C      LD BC, (UDG)   ; ADRESSE DES UDG-BEREICHS
OB5D 18 0B            JR $B6A
OB5F                  ;
OB5F CD 10 0C         PRTTO CALL PRTTOK ; TOKEN IN BEFEHL UMWANDELN,
OB62 C3 03 0B         JP POSHOL      ; AUSGEBEN UND MIT POS. HOLEN
OB65                  ;
OB65                  ; NORMALES ZEICHEN AUSGEBEN
OB65                  ;
OB65 C5              PUSH BC          ; POSITION RETTEN
OB66 ED 4B 36 5C      LD BC, (CHARS) ; ADRESSE DES ZEICHENBEREICHS
OB6A EB              EX DE, HL

```

0B6B 21 3B 5C	LD HL, FLAGS	
0B6E CB 86	RES 0, (HL)	; FÜHRENDEN SPACE ZULASSEN
0B70 FE 20	CP	; LEERZEICHEN
0B72 20 02	JR NZ, \$B76	; NEIN
0B74 CB C6	SET 0, (HL)	; FALLS DOCH: UNTERDRÜCKEN
0B76 26 00	PRTREA LD H, 0	; NUN DAS ZEICHEN TATSÄCHLICH
0B78		; BERECHNEN UND AUSGEBEN
0B78 6F	LD L, A	; DAZU IN HL DIE ADRESSE BILDEN
0B79 29	ADD HL, HL	; * 8, DA 8 * 8 MATRIX
0B7A 29	ADD HL, HL	
0B7B 29	ADD HL, HL	
0B7C 09	ADD HL, BC	; BASISADRESSE ADDIEREN
0B7D C1	POP BC	; POSITION WIEDER HOLEN
0B7E EB	EX DE, HL	; STARTADRESSE DES ZEICHENS IN DE
0B7F		
0B7F		; ALLE 8*8 ZEICHEN AUSGEBEN
0B7F		
0B7F 79	LD A, C	; SPALTENNUMMER
0B80 3D	DEC A	; -1, ALSO EINS WEITER
0B81 3E 21	LD A, 33	; AUF ERSTE SPALTE SETZEN, FALLS
0B83 20 0E	JR NZ, \$B93	; ERFORDERLICH. SPRUNG = NEIN
0B85 05	DEC B	; JA: ZEILENZÄHLER KORRIGIEREN
0B86 4F	LD C, A	; SPALTE AUF 33 (=GANZ LINKS)
0B87 FD CB 01 4E	BIT 1, (IY+1)	; PRINTER ?
0B8B 28 06	JR Z, \$B93	; NEIN
0B8D D5	PUSH DE	; STARTADRESSE RETTEN
0B8E CD CD 0E	CALL \$ECD	; PRINTERBUFFER AUSGEBEN
0B91 D1	POP DE	; STARTADRESSE ZURÜCKHOLEN
0B92 79	LD A, C	; NEUE SPALTENNUMMER
0B93 B9	CP C	; NEUE ZEILE
0B94 D5	PUSH DE	; ZEICHENADRESSE RETTEN
0B95 CC 55 0C	CALL Z, \$C55	; NEUE ZEILE: EVTL. SCROLLING
0B98 D1	POP DE	; ZEICHENADRESSE ZURÜCK
0B99 C5	PUSH BC	; ZWISCHENSPEICHERN
0B9A E5	PUSH HL	
0B9B 3A 91 5C	LD A, (PFLAG)	
0B9E 06 FF	LD B, \$FF	
0BA0 1F	RRA	; PFLAG BIT 0 UNTERSUCHEN
0BA1 3B 01	JR C, \$BA4	
0BA3 04	INC B	; OVER1: B=\$FF, SONST 0
0BA4 1F	RRA	; PFLAG BIT 2 UNTERSUCHEN
0BA5 1F	RRA	
0BA6 9F	SBC A	
0BA7 4F	LD C, A	; INVERSE1: C=\$FF, SONST 0
0BA8 3E 0B	LD A, B	; A ALS PIXELZÄHLER MIT 8 LADEN
0BAA A7	AND A	; CARRY LÖSCHEN
0BAB FD CB 01 4E	BIT 1, (IY+1)	; PRINTER ?
0BAF 2B 05	JR Z, \$BB6	; NEIN
0BB1 FD CB 30 CE	SET 1, (IY+\$30)	; PRINTERBUFFER NICHT LEER
0BB5 37	SCF	; PRINTER WIRD BENUTZT
0BB6 EB	EX DE, HL	; ZIEL- UND BASISADRESSE TAUSCHEN
0BB7		
0BB7		; DAS ZEICHEN WIRD IN EINER 8-FACHEN SCHLEIFE AUSGEGEBEN
0BB7		
0BB7 0B	EX AF, AF	; CARRYFLAG RETTEN (SET=PRINTER)
0BB8 1A	LD A, (DE)	
0BB9 A0	AND B	; OVER: 0 ZEICHEN WIE SPACE
0BBA AE	XOR (HL)	

0BB8	A9	XOR C	; INVERSE BERÜCKSICHTIGEN
0BBC	12	LD (DE),A	; ERGEBNIS SPEICHERN
0BB0	08	EX AF,AF	; CARRY UND ZÄHLER ZURÜCK
0BBE	38 13	JR C,\$BD3	; BEI PRINTER
0BC0	14	INC D	; ZIELADRESSE +1 (PAGE)
0BC1	23	INC HL	; BASISADRESSE +1
0BC2	3D	DEC A	; ZÄHLER -1
0BC3	20 F2	JR NZ,\$BB7	; NOCH NICHT 8 *
0BC5			
0BC5	EB	EX DE,HL	
0BC6	25	DEC H	
0BC7	FD CB 01 4E	BIT 1,(IY+1)	; ATTRIBUT-BYTE NUR BEI
0BC8	CC DB 08	CALL Z,\$BDB	; BILDSCHIRMHANDLING BEARBEITEN
0BCE	E1	POP HL	; ORIGINAL ZIEL- UND
0BCF	C1	POP BC	; POSITIONSADRESSE
0BD0	0D	DEC C	; SPALTE +1
0BD1	23	INC HL	; ZIELADRESSE +1
0BD2	C9	RET	
0BD3			
0BD3	08	EX AF,AF	; PRINTERFLAG (CARRY) RETTEN
0BD4	3E 20	LD A,32	; OFFSET ADDIEREN
0BD6	83	ADD E	
0BD7	5F	LD E,A	
0BD8	08	EX AF,AF	; FLAGS ZURÜCK FÜR SCHLEIFE
0BD9	18 E6	JR \$BC1	
0BDB			
0BDB			
0BDB			
0BDB	7C	LD A,H	; HÖHERES ADRESSBYTE DES ZIELS
0BDC	0F	RRCA	; DURCH 8 DIVIDIEREN,
0BDD	0F	RRCA	; UM DEN ENTSPRECHENDEN
0BDE	0F	RRCA	; BILDSCHIRMTEIL FESTZUSTELLEN
0BDF	E6 03	AND 3	
0BE1	F6 58	OR \$58	; HÖHERES BYTE DES ATTRIBUT-
0BE3	67	LD H,A	; SPEICHERS BILDEN
0BE4	ED 5B 8F 5C	LD DE,(ATTRT)	; D = ATTRT UND E = MASKT
0BE8	7E	LD A,(HL)	; ALTES ATTRIBUT HOLEN UND
0BE9	AB	XOR E	; MIT MASKT UND
0BEA	A2	AND D	; ATTRT VERKNÜPFEN
0BEB	AB	XOR E	
0BEC	FD CB 57 76	BIT 6,(IY+87)	
0BF0	2B 08	JR Z,\$BFA	; NICHT BEI PAPER9 SPRINGEN
0BF2	E6 C7	AND \$C7	; ALTE FARBE AUSBLENDEN
0BF4	CB 57	BIT 2,A	; TEST AUF INK = DUNKEL
0BF6	20 02	JR NZ,\$BFA	; NEIN
0BF8	EE 38	XOR \$38	; JA: PAPERCOLOUR = WEISS
0BFA	FD CB 57 66	BIT 4,(IY+87)	; INK = 9
0BFE	2B 08	JR Z,\$C08	; NEIN
0C00	E6 F8	AND \$F8	; ALTE INKCOLOUR WEGWERFEN
0C02	CB 6F	BIT 5,A	; UND BEI PAPERCOLOUR = WEISS
0C04	20 02	JR NZ,\$C08	; INKCOLOUR = DUNKEL
0C06	EE 07	XOR 7	; ANDERNFALLS INKCOLOUR = WEISS
0C08	77	LD (HL),A	; NEUEN ATTRIBUTWERT SPEICHERN
0C09	C9	RET	
0C0A		.END	
0C0A		.LIB SPECOC00-5	
0C0A		; SINCLAIR ZX SPECTRUM TEIL 0C00	
0C0A			

0C0A		;	SUBROUTINE ZUM AUSGEBEN VON MELDUNGEN UND TOKENS.
0C0A		;	DE ENTHALT DIE BASISADRESSE DER JEWELIGEN TABELLE
0C0A		;	UND A DIE NUMMER DER MELDUNG ODER DES TOKEN
0C0A		;	
0C0A	E5	PRTMEL	PUSH HL ; HÖHERES BYTE DES
0C0B	26 00		LD H,0 ; LETZTEN STACK-EINTRAGS AUF
0C0D	E3		EX (SP),HL ; NULL SETZEN, UM NACHFOLGENDE
0C0E	18 04		JR \$C14 ; SPACES ZU UNTERDRÜCKEN
0C10		;	
0C10	11 95 00	PRTTOK	LD DE,\$95 ; ADRESSE DER TOKENTABELLE
0C13	F5		PUSH AF ; NUMMER RETTEN
0C14	CD 41 0C		CALL \$C41 ; START IN DER TABELLE SUCHEN
0C17	38 09		JR C,\$C22 ; DIREKT AUSGEBEN
0C19	3E 20		LD A, ; BEI CARRY GELÖSCHT EVTL. EIN
0C1B	FD CB 01 46		BIT 0,(IY+1) ; LEERZEICHEN AUSGEBEN
0C1F	CC 3B 0C		CALL Z,\$C3B
0C22		;	
0C22		;	AUSGABE DER MELDUNG ODER TOKENS, BIS EIN ZEICHEN MIT
0C22		;	GESETZTEM BIT 7 GEFUNDEN WIRD
0C22		;	
0C22	1A		LD A,(DE) ; ZEICHEN AUS TABELLE LADEN
0C23	E6 7F		AND \$7F ; BIT 7 AUSBLENDEN
0C25	CD 3B 0C		CALL \$C3B ; UND AUSGABE
0C28	1A		LD A,(DE) ; NOCH MAL DAS ZEICHEN LADEN
0C29	13		INC DE ; POINTER IN TABELLE +1
0C2A	87		ADD A ; FALLS BIT 7 = 0, IST DAS CARRY
0C2B	30 F5		JR NC,\$C22 ; GELÖSCHT: WEITER AUSGEBEN
0C2D		;	
0C2D	D1		POP DE ; D = 0 FÜR MELDUNGEN, SONST
0C2E			CP \$48 ; D = 0 BIS \$5A FÜR TOKENS
0C2E	FE 48		JR Z,\$C35 ; SPRUNG, FALLS DAS LETZTE
0C30	28 03		CP \$82 ; ZEICHEN EIN \$ WAR
0C32	FE 82		RET C ; FALLS ZEICHEN KLEINER \$41 (=A)
0C34	D8		RET C ; WAR, RETURN
0C35		;	
0C35	7A		LD A,D ; BEI MELDUNGEN, 'RND', 'INKEYS'
0C36	FE 03		CP 3 ; UND 'PI' RETURN, SONST
0C38	D8		RET C
0C39	3E 20		LD A, ; IMMER EINEN SPACE AUSGEBEN
0C3B		;	
0C3B		;	AUSGABE EINES ZEICHENS IN REGISTER A UND
0C3B		;	RETEN DER REGISTER BC, DE UND HL
0C3B		;	
0C3B	D5		PUSH DE
0C3C	D9		EXX
0C3D	D7		RST PRTOU ; AUSGABE DES ZEICHENS
0C3E	D9		EXX
0C3F	D1		POP DE
0C40	C9		RET
0C41		;	
0C41		;	IN TABELLE (=DE) DEN START DER MELDUNG ODER
0C41		;	DES TOKENS SUCHEN
0C41		;	
0C41	F5		PUSH AF ; NUMMER RETTEN
0C42	EB		EX DE,HL ; HL = ANFANG DER TABELLE
0C43	3C		INC A ; NUMMER +1 FÜR SUCHSCHLEIFE
0C44	CB 7E		BIT 7,(HL) ; BEI JEDEM GESETZTEM BIT 7
0C46	23		INC HL ; REG A HERUNTERTÄZLEN BIS NULL

0C47 2B FB	JR Z,\$C44	; ERREICHT (= MELDUNG GEFUNDEN),
0C49 3D	DEC A	; SONST NUR HL INKREMENTIEREN
0C4A 20 FB	JR NZ,\$C44	
0C4C EB	EX DE,HL	; DE ZEIGT AUF DIE MELDUNG
0C4D F1	POP AF	; NUMMER ZURÜCK IN REG A
0C4E FE 20	CP 32	; NUMMERN 0 - 31 RETURN
0C50 DB	RET C	
0C51 1A	LD A,(DE)	; SONST ERSTES ZEICHEN LADEN
0C52 D6 41	SUB \$41	; UND AUF > \$41 PRÜFEN, UM EVTL.
0C54 C9	RET	; EINEN SPACE AUSZUGEBEN
0C55		
0C55	; SUBROUTINE ZUM TESTEN, OB EIN SCROLLING NOTWENDIG IST.	
0C55	; REGISTER B ENTHÄLT DIE ZU TESTENDE ZEILENNUMMER	
0C55		
0C55 FD CB 01 4E	BIT 1,(1Y+1)	; FALLS PRINTERAUSGABE,
0C59 C0	RET NZ	; DIREKT RETURN
0C5A		
0C5A 11 D9 0D	LD DE,\$DD9	; RETURNADRESSE \$DD9
0C5D D5	PUSH DE	; AUF STACK
0C5E 7B	LD A,B	
0C5F FD CB 02 46	BIT 0,(1Y+2)	; TEST TV-FLAG: SPRUNG,
0C63 C2 02 0D	JP NZ,\$D02	; FALLS 'INPUT AT'
0C66		; (ANMERKUNG: ZEILEN WERDEN
0C66		; VON UNTEN GEZÄHLT!)
0C66 FD BE 31	CP (1Y+\$31)	; ZEILENNUMMER > ALS DFSZ?
0C69 3B 1B	JR C,\$C86	; JA, ERROR
0C6B C0	RET NZ	; ZEILENNUMMER < DFSZ
0C6C		; WEITER, WENN BEIDE GLEICH
0C6C FD CB 02 66	BIT 4,(1Y+2)	; AUTOMATISCHES LISTING?
0C70 2B 16	JR Z,\$C88	; NEIN
0C72 FD 5E 2D	LD E,(1Y+\$2D)	; ZEILENZÄHLER HOLEN
0C75 1D	DEC E	; UND 1 SUBTRAHIEREN
0C76 2B 5A	JR Z,\$CD2	; FALLS 0: SCROLLING AUSFÜHREN
0C78 3E 00	LD A,0	; KANAL K ERÖFFNEN
0C7A CD 01 16	CALL \$1601	
0C7D ED 7B 3F 5C	LD SP,(LISTSP)	; STACKPOINTER SETZEN
0C81 FD CB 02 A6	REG 4,(1Y+2)	; UND MERKEN, DASS AUTOMATISCHES
0C85 C9	RET	; LISTEN BEENDET
0C86		
0C86 CF	RST ERRAUS	; MELDUNG 'OUT OF SCREEN'
0C87 04	.BYT \$04	
0C88		
0C88 FD 35 52	DEC (1Y+\$52)	; SCROLL-ZÄHLER -1
0C8B 20 45	JR NZ,\$CD2	; ZUM DIREKTEN SCROLLING
0C8D		; AUSGABE DER MELDUNG 'SCROLL?'
0C8D 3E 1B	LD A,24	
0C8F 90	SUB B	; SCROLL-ZÄHLER
0C90 32 8C 5C	LD (SCRCT),A	; ZURÜCKSETZEN
0C93 2A 8F 5C	LD HL,(ATTRT)	; ATTRT UND MASKT RETTEN
0C96 E5	PUSH HL	
0C97 3A 91 5C	LD A,(PFLAG)	; PFLAG EBENFALLS RETTEN
0C9A F5	PUSH AF	
0C9B 3E FD	LD A,\$FD	; KANAL K ÖFFNEN
0C9D CD 01 16	CALL \$1601	
0CA0 AF	XOR A	; REG A=0 (ERSTE MELDUNG),
0CA1 11 FB 0C	LD DE,SCROLL	; DE MIT ADRESSE DER TABELLE
0CA4 CD 0A 0C	CALL \$C0A	; LADEN UND AUSGABE 'SCROLL?'
0CA7 FD CB 02 EE	SET 5,(1Y+2)	; MERKE: UNT. TEIL NACH TASTEN-

OCAB
 OCAB 21 38 5C
 OCAE CB DE
 OCB0 CB AE
 OCB2 D9
 OCB3 CD D4 15
 OCB6 D9
 OCB7 FE 20
 OCB9 28 45
 OCB8 FE E2
 OCB0 28 41
 OCBF F6 20
 OCC1 FE 6E
 OCC3 28 38
 OCC5 3E FE
 OCC7 CD 01 16
 OCCA F1
 OCCB 32 91 5C
 OCCE E1
 OCCF 22 8F 5C
 OCD2
 OCD2
 OCD2
 OCD2 CD FE 0D
 OCD5 FD 46 31
 OCD8 04
 OCD9 0E 21
 OCDB C5
 OCDC CD 9B 0E
 OCDF 7C
 OCE0 0F
 OCE1 0F
 OCE2 0F
 OCE3 E6 03
 OCE5 F6 58
 OCE7 67
 OCE8 11 E0 5A
 OCEB
 OCEB 1A
 OCEC 4E
 OCED 06 20
 OCEF EB
 OCF0 12
 OCF1 71
 OCF2 13
 OCF3 23
 OCF4 10 FA
 OCF6 C1
 OCF7 C9
 OCF8
 OCF8
 OCF8
 OCF8 80
 OCF9 73 63
 OCFF BF
 OD00
 OD00 CF
 OD01 OC

LD HL, FLAGS
 SET 3, (HL)
 RES 5, (HL)
 EXX
 CALL \$15D4
 EXX
 CP \$20
 JR Z, \$D00
 CP \$E2
 JR Z, \$D00
 OR \$20
 CP \$6E
 JR Z, \$D00
 LD A, \$FE
 CALL \$1601
 POP AF
 LD (PFLAG), A
 POP HL
 LD (ATTR), HL
 ; DRUCK LÖSCHEN
 ; L-MODUS SETZEN UND
 ; 'KEINE TASTE BISHER' ANMERKEN
 ; 1 ZEICHEN HOLEN
 ; FALLS TASTENDRUCK 'BREAK',
 ; 'STOP', 'N' ODER 'n' WAR,
 ; MELDUNG 'BREAK - CONT repeats'
 ; AUSGEBEN
 ; SONST KANAL 'S' ERÖFFNEN
 ; PFLAG ZURÜCKHOLEN
 ; ATTR UND MASKT EBENFALLS
 ; BILDSCHIRM ROLLEN UND PARAMETER NEU SETZEN
 CALL \$DFF
 LD B, (IY+\$31)
 INC B
 LD C, \$21
 PUSH BC
 CALL \$E9B
 LD A, H
 RRCA
 RRCA
 RRCA
 AND 3
 OR \$58
 LD H, A
 LD DE, \$5AE0
 ; GANZEN BILDSCHIRM ROLLEN
 ; ZEILENNUMMER OBERER TEIL NEU
 ; ERSTE SPALTE SETZEN
 ; UND ZWISCHENSPEICHERN
 ; DAS ENTSPRECHENDE ATTRIBUTE
 ; FÜR DIESEN TEIL SUCHEN
 ; ERGEBNIS IN HL
 ; ZEIGER AUF ERSTES ATTRIBUTE
 ; DER UNTERSTEN ZEILE
 LD A, (DE)
 LD C, (HL)
 LD B, 32
 EX DE, HL
 LD (DE), A
 LD (HL), C
 INC DE
 INC HL
 DJNZ \$CF0
 POP BC
 RET
 ; ZÄHLER FÜR 32 * AUSTAUSCHEN
 ; DIE ERSTEN ATTRIBUTE TAUSCHEN
 ; UND DEN NÄCHSTEN 32 BYTES
 ; DIE GLEICHEN WERTE ZUWEISEN
 ; ZEILEN-/SPALTENNUMMER DER
 ; UNTERSTEN ZEILE ZURÜCKHOLEN
 ; MELDUNG 'SCROLL?'
 SCROLL .BYT \$80, 'scroll', \$BF
 RST ERRASUS
 .BYT \$0C
 ; MELDUNG:
 ; 'BREAK - CONT repeats'

```

OD02      ;
OD02      ; UNTEREN BILDSCHIRMTEIL BEHANDELN
OD02      ;
OD02 FE 02      CP 2      ; ERRORAUSGABE, FALLS UNTERER
OD04 38 80      JR C,$C86 ; TEIL ZU GROSS
OD06 FD 86 31   ADD (IY+$31)
OD09 D6 19      SUB $19   ; RETURN, WENN
OD0B D0         RET NC    ; KEIN SCROLLING NOTWENDIG
OD0C ED 44      NEG      ; ANZAHL DER SCROLLS IN A
OD0E C5         PUSH BC   ; ZEILEN-/SPALTENNUMMER RETTEN
OD0F 47         LD B,A    ; ANZAHL ZWISCHENSPEICHERN
OD10 2A 8F 5C   LD HL,(ATTRT) ; ATTRT, MASKT UND
OD13 E5         PUSH HL
OD14 2A 91 5C   LD HL,(PFLAG) ; PFLAG ZWISCHENSPEICHERN
OD17 E5         PUSH HL
OD18 CD 4D 0D   CALL AKTCOL ;
OD1B 78         LD A,B    ; SCROLLINGZAHL ZURÜCK NACH A
OD1C F5         PUSH AF   ; ZWISCHENSPEICHERN
OD1D 21 6B 5C   LD HL,DFSZ ;
OD20 46         LD B,(HL) ; WERT VON DFSZ IN REG B UND
OD21 78         LD A,B    ; DIESER +1 NACH
OD22 3C         INC A     ;
OD23 77         LD (HL),A ; DFSZ ZURÜCKSCHREIBEN
OD24 21 89 5C   LD HL,SPOSN+1
OD27 BE         CP (HL)   ; SPRUNG, FALLS NUR DER UNTERE
OD28 38 03      JR C,$D2D ; TEIL GESCROLLT WERDEN SOLL
OD2A 34         INC (HL)  ; SONST SPOSN-HIGH +1 UND DEN
OD2B 06 18      LD B,$18  ; GANZEN BILDSCHIRM ROLLEN
OD2D CD 00 0E   CALL $E00 ; (REG B) ZEILEN ROLLEN
OD30 F1         POP AF    ; SCROLL-ZÄHLER HOLEN UND
OD31 3D         DEC A     ; 1 SUBTRAHIEREN
OD32 20 E8      JR NZ,$D1C ; NOCH MAL ROLLEN
OD34 E1         POP HL
OD35 FD 75 57   LD (IY+$57),L ; SONST PFLAG UND
OD38 E1         POP HL
OD39 22 8F 5C   LD (ATTRT),HL ; ATTRT UND MASKT WIEDER HOLEN
OD3C ED 4B 88 5C LD BC,(SPOSN) ; FALLS SPOSN GEÄNDERT WURDE,
OD40 FD CB 02 86 RES 0,(IY+2) ; DEN WERT VON DFCC
OD44 CD D9 0D   CALL $D09 ; NEU BERECHNEN
OD47 FD CB 02 C6 SET 0,(IY+2) ; BEHANDLUNG UNT. TEIL ANMERKEN
OD4B C1         POP BC   ; ZEILEN/SPALTENNUMMER ZURÜCK
OD4C C9         RET
OD4D      ;
OD4D      ; DIESE SUBROUTINE HOLT DIE AKTUELLEN FARBEN IN DIE
OD4D      ; 'TRANSPARENTEN' VARIABLEN ATTRT UND MASKT
OD4D      ;
OD4D AF        AKTCOL XOR A ; A = 0
OD4E 2A 8D 5C   LD HL,(ATTRP) ; AKTUELLE ATTRP UND MASKP
OD51 FD CB 02 46 BIT 0,(IY+2)
OD53 28 04      JR Z,$D5B ; OBERER BILDSCHIRMTEIL
OD57 67         LD H,A    ; UNTERER TEIL: A UND
OD58 FD 6E 0E   LD L,(IY+$E) ; BORDER BENUTZEN FÜR
OD5B 22 8F 5C   LD (ATTRT),HL ; ATTRT UND MASKT
OD5E      ; PFLAG NEU SETZEN (FÜR UNTEREN TEIL IST A = 0)
OD5E 21 91 5C   LD HL,PFLAG
OD61 20 02      JR NZ,$D65 ; IMMER SPRUNG BEIM UNTEREN TEIL
OD63      ;
OD63 7E         LD A,(HL) ; OBERER TEIL: ALTEN WERT HOLEN

```

0D64	0F	RRCA	; UND DIE UNGERADEN BITS (7,5..)
0D65	AE	XOR (HL)	; IN DIE GERADEN KOPIEREN
0D66	E6 55	AND \$55	; BEIM UNTEREN SCHIRMTEIL WERDEN
0D68	AE	XOR (HL)	; NUR DIE GERADEN BITS
0D69	77	LD (HL),A	; GELÖSCHT (BITS 6,4,2,0)
0D6A	C9	RET	
0D6B			
0D6B		; 'CLS'-BEFEHL	
0D6B			
0D6B	CD AF 0D	CALL \$DAF	; GANZEN BILDSCHIRM LÖSCHEN
0D6E	21 3C 5C	LD HL,TVFLAG	; UNT. TEIL NACH TASTENDRUCK
0D71	CB AE	RES 5,(HL)	; NICHT LÖSCHEN
0D73	CB C6	SET 0,(HL)	; UNTEREN TEIL SETZEN
0D75	CD 4D 0D	CALL AKTCOL	; BORDER NACH ATTR KOPIEREN
0D78	FD 46 31	LD B,(IY+\$31)	; DFSZ LADEN UND UNTEREN TEIL
0D7B	CD 44 0E	CALL \$E44	; DES BILDSCHIRMS LÖSCHEN
0D7E	21 C0 5A	LD HL,\$5AC0	; ADRESSE ATTRIBUT VON ZEILE 22
0D81	3A 8D 5C	LD A,(ATTRP)	; ATTRP WIRD ALS ATTRIBUT FÜR
0D84	05	DEC B	; DEN UNT. TEIL BENUTZT
0D85	18 07	JR \$DBE	; IN DIE SCHLEIFE SPRINGEN
0D87	0E 20	LD C,32	; 32 ZEICHEN PRO ZEILE
0D89	2B	DEC HL	
0D8A	77	LD (HL),A	; WERTE SETZEN
0D8B	0D	DEC C	
0D8C	20 FB	JR NZ,\$DB9	; ZEILE NOCH NICHT FERTIG
0D8E	10 F7	DJNZ \$DB7	; UNTEREN TEIL NOCH NICHT FERTIG
0D90			
0D90	FD 36 31 02	LD (IY+\$31),2	; 2 ZEILEN ALS UNT. TEIL SETZEN
0D94	3E FD	LD A,\$FD	; KANAL K ÖFFNEN
0D96	CD 01 16	CALL \$1601	
0D99	2A 51 5C	LD HL,(CURCHL)	; ADRESSE AKT. KANAL UND
0D9C	11 F4 09	LD DE,AUSGAB	; AUSGABEADRESSE AUF \$09F4
0D9F	A7	AND A	; (CARRY LÖSCHEN)
0DA0	73	LD (HL),E	; SETZEN BZW. IN DER
0DA1	23	INC HL	; ZWEITEN SCHLEIFE AUF
0DA2	72	LD (HL),D	
0DA3	23	INC HL	
0DA4	11 AB 10	LD DE,\$10AB	; DIE EINGABEADRESSE
0DA7	3F	CCF	
0DAB	38 F6	JR C,\$DA0	
0DAA	01 21 17	LD BC,\$1721	; ZEILE-/SPALTENNUMMER DER
0DAD	18 2A	JR \$DD9	; ERSTEN UNT. ZEILE
0DAF			
0DAF		; BILDSCHIRM LÖSCHEN	
0DAF			
0DAF	21 00 00	LD HL,0	; VARIABLE 'COORDS' LÖSCHEN
0DB2	22 7D 5C	LD (COORDS),HL	
0DB5	FD CB 30 86	RES 0,(IY+\$30)	; FLAGS2: BILDSCHIRM GELÖSCHT
0DB9	CD 94 0D	CALL \$D94	; EIN-/AUSGABEADR. UND KANAL K
0DBC			; ORIGINAL SETZEN
0DBC	3E FE	LD A,\$FE	; KANAL S ÖFFNEN
0DBE	CD 01 16	CALL \$1601	
0DC1	CD 4D 0D	CALL AKTCOL	; PERMANENTEN WERTE BENUTZEN
0DC4	06 18	LD B,\$18	; 24 ZEILEN DES BILDSCHIRMS
0DC6	CD 44 0E	CALL \$E44	; LÖSCHEN
0DC9	2A 51 5C	LD HL,(CURCHL)	; AUSGABEADRESSE
0DCC	11 F4 09	LD DE,AUSGAB	; AUF \$09F4
0DCF	73	LD (HL),E	; SETZEN

0DD0	23	INC HL	
0DD1	72	LD (HL),D	
0DD2	FD 36 52 01	LD (IY+\$52),1	; SCROLL-ZÄHLER RÜCKSETZEN
0DD6	01 21 18	LD BC,\$1821	; ERSTE ZEILE UND ERSTE SPALTE
0DD9			
0DD9			; PRINTPOSITION AUF BILDSCHIRM SETZEN
0DD9			; EINSPRUNG MIT ZEILEN-/SPALTENNUMMER IN BC ODER, FALLS
0DD9			; PRINTER ANGESPROCHEN, SPALTENPOSITION IN C FÜR PRINTER
0DD9			
0DD9	21 00 5B	LD HL,PTBUF	; ADRESSE PRINTERPUFFER
0DDC	FD CB 01 4E	BIT 1,(IY+1)	; PRINTER ANGESPROCHEN?
0DE0	20 12	JR NZ,\$DF4	; JA
0DE2	78	LD A,B	; ZEILENNUMMER NACH A
0DE3	FD CB 02 46	BIT 0,(IY+2)	
0DE7	28 05	JR Z,\$DEE	; BEI HAUPTBILDSCHIRMTTEIL
0DE9	FD 86 31	ADD (IY+\$31)	; DFSZ ADDIEREN UND 24 SUBTRA-
0DEC	D6 18	SUB \$18	; HIEREN, WEIL VON UNTEN
0DEE			; GEZÄHLT WIRD
0DEE	C5	PUSH BC	; ZWISCHENSPEICHERN
0DEF	47	LD B,A	; KOPIE NACH B, UM DIE ADRESSE
0DF0	CD 9B 0E	CALL \$E9B	; DER ZEILE IN HL ZU BERECHNEN
0DF3	C1	POP BC	; ZEILEN-/SPALTENZAHL ZURÜCK
0DF4	3E 21	LD A,33	; SPALTENZAHL UMRECHNEN, DA
0DF6	91	SUB C	; DIESE VON HINTEN GEZÄHLT WIRD
0DF7	3F	LD E,A	; NEUE ADRESSE BILDEN
0DF8	16 00	LD D,0	; UND ZUSAMMEN MIT
0DFA	19	ADD HL,DE	; ZEILEN-/SPALTENNUMMERN
0DFB	C3 DC 0A	JP NEUSTO	; SPEICHERN
0DFE			
0DFE			; SUBROUTINE ZUM ROLLEN (SCROLLING) DES BILDSCHIRMS
0DFE			; ANZAHL DER GEROLLTEN ZEILEN MUSS IN B STEHEN
0DFE			
0DFE	06 17	LD B,23	; EINSTIEG NACH FRAGE 'SCROLL?'
0E00	CD 9B 0E	CALL \$E9B	; STARTADRESSE DER ZEILE SUCHEN
0E03	0E 08	LD C,8	; 8 PIXEL-ZEILEN
0E05	C5	PUSH BC	; ZEILEN-/PIXELZÄHLER RETTEN
0E06	E5	PUSH HL	; STARTADRESSE EBENFALLS
0E07	78	LD A,B	; PRÜFEN, OB OBERSTE ZEILE
0E08	E6 07	AND 7	; EINES BILDSCHIRMDRITTELS
0E0A	78	LD A,B	; GESCROLLT WERDEN SOLL
0E0B	20 0C	JR NZ,\$E19	; NEIN
0E0D			; JA: OBERSTE ZEILE MUSS IN
0E0D			; UNTERSTE DES 1 HÖHEREN TEILS
0E0D			; KOPIERT WERDEN
0E0D	EB	EX DE,HL	; RETTE HL
0E0E	21 E0 F8	LD HL,\$F8E0	; ES WIRD DIE NOTWENDIGE ADRESSE
0E11	19	ADD HL,DE	; FÜR DE BERECHNET
0E12	EB	EX DE,HL	; HL WIEDER HOLEN
0E13	01 20 00	LD BC,32	; 32 ZEICHEN
0E16	3D	DEC A	; ZÄHLER -1
0E17	ED B0	LDIR	; UND 32 ZEICHEN KOPIEREN
0E19			
0E19			; PIXEL-ZEILEN INNERHALB DER BILDSCHIRMDRITTEL SCROLLEN
0E19			
0E19	EB	EX DE,HL	; FÜR DE DIE ZIELADRESSE
0E1A	21 E0 FF	LD HL,\$FFE0	; BERECHNEN, 32 BYTES VORHER
0E1D	19	ADD HL,DE	
0E1E	EB	EX DE,HL	; HL WIEDER HOLEN

0E1F	47	LD B,A	; ZEILENNUMMER ZWISCHENSPEICHERN
0E20	E6 07	AND 7	; ANZAHL DER IN DIESEM BILD-
0E22	0F	RRCA	; DRITTEL VORHANDENEN ZEICHEN
0E23	0F	RRCA	; BERECHNEN
0E24	0F	RRCA	
0E25	4F	LD C,A	; UND IN C BRINGEN
0E26	7B	LD A,B	; ZEILENNUMMER IN A ZURÜCK
0E27	06 00	LD B,0	; BC = ANZAHL 'ALLE ZEICHEN'
0E29	ED B0	LDIR	; UND TRANSFERIEREN
0E2B	06 07	LD B,7	
0E2D	09	ADD HL,BC	; HL +\$0700 FÜR NÄCHSTES DRITTEL
0E2E	E6 F8	AND \$F8	; SPRUNG, FALLS WEITERE DRITTEL
0E30	20 DB	JR NZ,\$E0D	; BEARBEITET WERDEN MÜSSEN
0E32			
0E32		; OBIGE ROUTINE MUSS 8 MAL, FÜR JEDE PIXELZEILE EINMAL,	
0E32		; DURCHLAUFEN WERDEN	
0E32			
0E32	E1	POP HL	; STARTADRESSE NOCHMAL HOLEN
0E33	24	INC H	; +1 FÜR NÄCHSTE PIXELZEILE
0E34	C1	POP BC	; PIXELZÄHLER HOLEN
0E35	0D	DEC C	; PIXEL -1
0E36	20 CD	JR NZ,\$E05	; NOCH NICHT 8 *
0E38			
0E38		; DIE ATTRIBUTE MÜSSEN AUCH NOCH GESROLLT WERDEN	
0E38			
0E38	CD 88 0E	CALL \$E8B	; ATTRIBUTEADRESSE BERECHNEN
0E3B	21 E0 FF	LD HL,\$FFE0	; DIFFERENZ VON 32 FÜR ATTRIBUTE
0E3E	19	ADD HL,DE	; VON DE ABZIEHEN
0E3F	EB	EX DE,HL	
0E40	ED B0	LDIR	; ATTRIBUTE VERSCHIEBEN
0E42	06 01	LD B,1	; UNTERSTE ZEILE LÖSCHEN
0E44			
0E44		; DIESE ROUTINE LÖSCHT DIE ANZAHL ZEILEN VON UNTEN, DIE	
0E44		; DURCH B BESTIMMT WERDEN	
0E44			
0E44	C5	PUSH BC	; ZEILENNUMMER RETTEN
0E45	CD 9B 0E	CALL \$E9B	; ADRESSE IN HL BERECHNEN
0E48	0E 0B	LD C,B	; 8 PIXELZEILEN
0E4A	C5	PUSH BC	; ZEILENNUMMER, PIXELZÄHLER UND
0E4B	E5	PUSH HL	; STARTADRESSE ZWISCHENSPEICHERN
0E4C	7B	LD A,B	; ZEILENZÄHL NACH A
0E4D	E6 07	AND 7	
0E4F	0F	RRCA	; HIERMIT DIE ZEICHENZÄHL
0E50	0F	RRCA	; BERECHNEN UND IN
0E51	0F	RRCA	
0E52	4F	LD C,A	; C BRINGEN
0E53	7B	LD A,B	
0E54	06 00	LD B,0	; BC AUF 1 WENIGER ALS
0E56	0D	DEC C	; ZEICHENZÄHL SETZEN
0E57	54	LD D,H	; STARTADRESSE NACH DE
0E58	5D	LD E,L	
0E59	36 00	LD (HL),0	; EIN PIXEL LÖSCHEN
0E5B	13	INC DE	; FÜR SCHLEIFE DE +1
0E5C	ED B0	LDIR	; ALLE WEITEREN PIXELS LÖSCHEN
0E5E	11 01 07	LD DE,\$701	; FÜR PIXELREIHE IM NÄCHSTEN
0E61	19	ADD HL,DE	; DRITTEL \$0701 ADDIEREN
0E62	3D	DEC A	; ZEILENZÄHL -1
0E63	E6 F8	AND \$F8	; NUR DEN 'DRITTELZÄHLER' NACH

0E65	47	LD B,A	; B BRINGEN UND EVTL.
0E66	20 E5	JR NZ,\$E4D	; WEITERE DRITTEL BEARBEITEN
0E68			
0E68		; PRÜFEN, OB 8* DIE ROUTINE DURCHLAUFEN WURDE	
0E68			
0E68	E1	POP HL	; STARTADRESSE ZURÜCK
0E69	24	INC H	; UND 1 ADDIEREN
0E6A	C1	POP BC	; ZEILEN-/PIXELZÄHLER HOLEN
0E6B	0D	DEC C	; PIXEL -1
0E6C	20 DC	JR NZ,\$E4A	; NOCH NICHT 8 *
0E6E	CD 88 0E	CALL \$E8B	; ADRESSE UND ZAHL DER ATTRIBUT-
0E71			; BYTES SUCHEN
0E71	62	LD H,D	
0E72	6B	LD L,E	
0E73	13	INC DE	
0E74	3A 8D 5C	LD A,(ATTRP)	; ATTRP FÜR HAUPTTEIL BENUTZEN
0E77	FD CB 02 46	BIT 0,(IY+2)	; TVFLAG: HAUPTTEIL BILDSCHIRM?
0E7B	28 03	JR Z,\$E80	; JA
0E7D	3A 4B 5C	LD A,(BORDCR)	; SONST BORDERCOLOUR BENUTZEN
0E80	77	LD (HL),A	; ERSTES SETZEN
0E81	0B	DEC BC	; UND ZÄHLER -1
0E82	ED 80	LDIR	; DEN REST EINSCHREIBEN
0E84	C1	POP BC	; ZEILENNUMMER ZURÜCK UND
0E85	0E 21	LD C,33	; SPALTENZAHL AUF ERSTE SETZEN
0E87	C9	RET	
0E88			
0E88		; BERECHNE ZU EINER BILDSCHIRMSTELLE DIE ADRESSE	
0E88		; DER ATTRIBUT-INFORMATIONEN	
0E88			
0E88	7C	LD A,H	; HÖHERES BYTE LADEN
0E89	0F	RRCA	; MIT 32 MALNEHMEN
0E8A	0F	RRCA	
0E8B	0F	RRCA	
0E8C	3D	DEC A	; EINS ABZIEHEN
0E8D	F6 50	OR \$50	; BASISADRESSE DER ATTRIBUT-
0E8F			; INFORMATIONEN
0E8F	67	LD H,A	; HÖHERES BYTE GEFUNDEN
0E90	EB	EX DE,HL	; LOW BYTE BLEIBT GLEICH
0E91	61	LD H,C	; C IST IMMER NULL
0E92	6B	LD L,B	; ZEILENNUMMER
0E93	29	ADD HL,HL	; EBENFALLS MIT 32 MALNEHMEN
0E94	29	ADD HL,HL	
0E95	29	ADD HL,HL	
0E96	29	ADD HL,HL	
0E97	29	ADD HL,HL	
0E98	44	LD B,H	; NACH BC ZURÜCKLADEN
0E99	4D	LD C,L	
0E9A	C9	RET	
0E9B			
0E9B		; BILDE SPEICHERADRESSE EINER PUNKTREIHE	
0E9B		; DES SCHIRMS IN HL	
0E9B			
0E9B	3E 18	LD A,\$18	
0E9D	90	SUB B	; ZEILE WIRD VON UNTEN BEZAHLT
0E9E	57	LD D,A	; MIT 32 MALNEHMEN
0E9F	0F	RRCA	; DURCH DREIMALIGES RECHTSROTIEREN
0EA0	0F	RRCA	; (EBENSO GUT WIE 5* LINKS)
0EA1	0F	RRCA	

0EA2 E6 E0	AND \$E0	; DIE DREI GÜLTIGEN BITS NEHMEN
0EA4 6F	LD L,A	; NIEDERES BYTE SETZEN
0EA5 7A	LD A,D	; HÖHERES BYTE
0EA6 E6 18	AND \$18	; KANN NUR \$40/\$48/\$50 SEIN
0EA8 F6 40	OR \$>BILD	; STARTADRESSE DES BILDSCHIRMS
0EAA 67	LD H,A	; HÖHERES BYTE SETZEN
0EAB C9	RET	
0EAC		
0EAC	; COPY SCREEN	
0EAC		
0EAC F3	DI	
0EAD 06 80	LD B,\$80	; 176 (22*8) DRUCKZEILEN
0EAF		; ZU JE EINER NADEL
0EAF 21 00 40	LD HL,BILD	; STARTADRESSE BILDSCHIRM
0EB2 E5	NXLINE PUSH HL	
0EB3 C5	PUSH BC	
0EB4 CD F4 0E	CALL PRLINE	; EINE NADELZEILE AUSGEBEN
0EB7 C1	POP BC	
0EB8 E1	POP HL	
0EB9 24	INC H	; 256 BYTES WEITER: NACHSTE ZEILE
0EBA 7C	LD A,H	
0EBB E6 07	AND 7	
0EBD 20 0A	JR NZ,NXLINE1	; NOCH INNERHALB EINER ZEILE
0EBF 7D	LD A,L	
0EC0 C6 20	ADD 32	; 32 BYTES WEITER: NACHSTE ZEILE
0EC2 6F	LD L,A	
0EC3 3F	CCF	; IST L ÜBERGELAUFEN?
0EC4 9F	SBC A	; BEI ÜBERL. FF, SONST 00.
0EC5 E6 FB	AND \$FB	; BIT 0..3 ZÄHLEN NADELZEILEN
0EC7 84	ADD H	; \$FB ADDIEREN, WENN L=00
0ECB 67	LD H,A	
0EC9 10 E7	NXLINE1 DJNZ NXLINE	; WEITER MIT DEN 176 ZEILEN
0ECB 18 0D	JR \$EDA	; ENDE COPY
0ECD		
0ECD	; PRINTER BUFFER \$5B00..\$5BFF AUSDRUCKEN	
0ECD		
0ECD F3	DI	
0ECE 21 00 5B	LD HL,PTRBUF	; PRINTERBUFFERADRESSE LADEN
0ED1 06 08	LD B,B	; 8 NADELREIHEN AUSGEBEN
0ED3 C5	PUSH BC	
0ED4 CD F4 0E	CALL PRLINE	
0ED7 C1	POP BC	
0ED8 10 F9	DJNZ \$ED3	
0EDA 3E 04	LD A,4	; ABSCHALTBIT FÜR DRUCKER
0EDC D3 FB	OUT (\$FB),A	; AUSGEBEN
0EDE FB	EI	
0EDF		
0EDF 21 00 5B	LD HL,PTRBUF	; PRCC=0
0EE2 FD 75 46	LD (IY+\$46),L	
0EE5	; GANZEN PRINTER BUFFER LÖSCHEN	
0EE5 AF	XOR A	
0EE6 47	LD B,A	
0EE7 77	CLRPRB LD (HL),A	; NULL EINSCHREIBEN
0EE8 23	INC HL	
0EE9 10 FC	DJNZ CLRPRB	; 255 MAL!
0EEB		
0EEB FD CB 30 8E	RES 1,(IY+\$30)	; FLAG52: ANMERKEN, DASS
0EEF		; PRINTERBUFFER LEER

```

0EEF 0E 21          LD C,33
0EF1 C3 D9 0D      JP $DD9
0EF4              ;
0EF4              ; AUSGABE EINER DRUCKERZEILE (32 BYTES NADELGRAFIK)
0EF4              ;
0EF4 78            PRLINE LD A,B          ; MENGE DER NOCH ZU DRUCKENDEN
0EF5              ; NADELREIHEN
0EF5 FE 03        CP 3
0EF7 9F           SBC A
0EF8 E6 02        AND 2                ; WENN NUR NOCH 2 ZEILEN ZU
0EFA D3 FB        OUT ($FB),A          ; DRUCKEN: PRINTERMOTOR BREMSSEN
0EFC 57           LD D,A
0EFD              ;
0EFD CD 54 1F     CALL $1F54           ; STOPTASTE PRÜFEN
0F00 38 0A        JR C,$F0C
0F02 3E 04        LD A,4              ; ABSCHALTBIT FÜR PRINTER
0F04 D3 FB        OUT ($FB),A
0F06 FB           EI
0F07 CD DF 0E     CALL $EDF           ; CLEAR PRINTER BUFFER
0F0A CF           RST ERRASUS         ; MELDUNG:
0F0B 0C           .BYT $0C           ; 'BREAK - CONT REPEATS'
0F0C              ;
0F0C DB FB        IN A,($FB)          ; STATUS DES DRUCKERS LADEN
0F0E 87           ADD A               ; BIT6 MUSS NULL SEIN, SONST
0F0F FB           RET M              ; IST KEIN DRUCKER DA
0F10 30 EB        JR NC,$EFD         ; WARTEN, WENN BIT7 NULL WAR.
0F12              ;
0F12              ; ALLES ZUR AUSGABE BEREIT
0F12              ;
0F12 0E 20        LD C,32             ; 32 BYTES AUSGEBEN
0F14 5E           PRTBYT LD E,(HL)     ; AUSZUGEBENDES BYTE LADEN
0F15 23           INC HL
0F16 06 08        LD B,B
0F18 CB 12        PRTBIT RL D         ; BIT 7 VON PORT $FB IST DIE
0F1A              ; DRUCKERNADEL
0F1A CB 13        RL E               ; E WIRD BITWEISE AUSGEGEBEN
0F1C CB 1A        RR D               ; D TRÄGT DAS DRUCKBYTE
0F1E              ;
0F1E DB FB        IN A,($FB)          ;
0F20 1F           RRA                ; BIT 0 MELDET DRUCKER READY
0F21 30 FB        JR NC,$F1E
0F23              ;
0F23 7A           LD A,D
0F24 D3 FB        OUT ($FB),A        ; HERAUS MIT DER EINEN NADEL
0F26 10 F0        DJNZ PRTBIT        ; NÄCHSTES DER 32 BYTES
0F28 0D           DEC C
0F29 20 E9        JR NZ,PRTBYT
0F2B C9           RET
0F2C              ;
0F2C              ; =====
0F2C              ; EDITIEREN EINER BILDSCHIRMZEILE
0F2C              ; AUFRUF DURCH HAUPTROUTINE ZUM EINGEBEN EINER BASIC-
0F2C              ; ZEILE ODER BEI EINEM INPUTBEFEHL IN EINEM PROGRAMM
0F2C              ;
0F2C              ;
0F2C 2A 3D 5C     LD HL,(ERRSP)       ; RETTE ERRORSTACKPOINTER
0F2F E5           PUSH HL
0F30 21 7F 10     LD HL,$107F        ; RETURNADRESSE FÜR EDITORERROR
0F33 E5           PUSH HL            ; AUF STACK UND HIERFÜR

```

OF34 ED 73 3D 5C	LD (ERRSP),SP	; DEN ERRORSTACKPOINTER SETZEN
OF38 CD D4 15	CALL \$15D4	; EIN ZEICHEN VON DER TASTATUR
OF3B F5	PUSH AF	; HOLEN UND FETTEN
OF3C 16 00	LD D,0	; DAUER DES TASTATUR-FLICKS
OF3E FD 5E FF	LD E,(IY+\$FF)	; HOLEN
OF41 21 C8 00	LD HL,\$C8	; EBENFALLS DIE TONHOHE
OF44 CD B5 03	CALL \$3B5	; UND ENTSPRECHEND PFELEN
OF47 F1	POP AF	; ZEICHEN HOLEN
OF48 21 38 0F	LD HL,\$F38	; STACK MIT \$F38
OF4B E5	PUSH HL	; VORBESETZEN
OF4C FE 18	CP \$18	; ALLE ZEICHEN, TOKENS UND
OF4E 30 31	JR NC,\$F81	; GRAFISCH ZEICHEN ÜBERNEHMEN
OF50 FE 07	CP 7	; KOMMA EBENFALLS
OF52 38 2D	JR C,\$F81	; EDITORZEICHEN (DEL., CURSOR...)
OF54 FE 10	CP \$10	; DANN SPRUNG
OF56 38 3A	JR C,\$F92	
OF58		
OF58		
OF58		
OF5B 01 02 00	LD BC,2	; 2 PLATZE BEI 'INK' UND 'PAPER'
OF5B 57	LD D,A	; KOPIE DES TASTENDRUCKS
OF5C FE 16	CP \$16	; 'INK' UND 'PAPER'
OF5E 38 0C	JR C,\$F6C	; JA
OF60		
OF60 03	INC BC	; FÜR 'AT' UND 'TAB' 3 PLATZE
OF61 FD CB 37 7E	BIT 7,(IY+55)	; SPRUNG, WENN NICHT
OF63 CA 1E 10	JP Z,\$101E	; 'INPUT LINE'
OF68		
OF68 CD D4 15	CALL \$15D4	; NÄCHSTES ZEICHEN HOLEN UND
OF6B 5F	LD E,A	; NACH E BRINGEN
OF6C		
OF6C CD D4 15	CALL \$15D4	; EIN WEITERES ZEICHEN FÜR
OF6F D5	PUSH DE	; STEUERCODES HOLEN, DE RETTEN
OF70 2A 5B 5C	LD HL,(KCUR)	; 'K'-MODUS SETZEN
OF73 FD CB 07 86	RES 0,(IY+7)	; 2 ODER 3 PLATZE BESORGEN
OF77 CD 55 16	CALL \$1655	; STEUERCODES WIEDER HOLEN
OF7A C1	POP BC	; UND SPEICHERN
OF7B 23	INC HL	
OF7C 70	LD (HL),B	; BEI 'INK' UND 'PAPER' WIRD DER
OF7D 23	INC HL	; ZWEITE WIEDER ÜBERSCHRIEBEN
OF7E 71	LD (HL),C	
OF7F 1B 0A	JR \$F8B	
OF81		
OF81		
OF81		
OF81		
OF81 FD CB 07 86	RES 0,(IY+7)	; K-MODUS SETZEN
OF85 2A 5B 5C	LD HL,(KCUR)	; CURSORPOSITION
OF88 CD 52 16	CALL \$1652	; EIN SPEICHERPLATZ BESORGEN
OF8B 12	LD (DE),A	; ZEICHEN/CODE SPEICHERN
OF8C 13	INC DE	; CURSORPOSITION +1 UND
OF8D ED 53 5B 5C	LD (KCUR),DE	; SPEICHERN
OF91 C9	RET	; NORMAL IN DIE EDITORSCHLEIFE
OF92		
OF92		
OF92		
OF92 5F	LD E,A	; CODE NACH DE
OF93 16 00	LD D,0	; BRINGEN

0F93	21 99 0F	LD HL,\$F99	; ADRESSE DER EDITORZEICHEN
0F98	19	ADD HL,DE	; ADRESSE DES ZEICHENS BESTIMMEN
0F99	5E	LD E,(HL)	; UND DEN OFFSET NACH E LADEN
0F9A	19	ADD HL,DE	; DIE ROUTINENADRESSE BERECHNEN
0F9B	E5	PUSH HL	; DIESE AUF DEN STACK SCHREIBEN,
0F9C	2A 5B 5C	LD HL,(KCUR)	; CURSORADRESSE HOLEN
0F9F	C9	RET	; IN DIE ROUTINE SPRINGEN
0FA0			
0FA0			; OFFSET DER EDITORSTEUERZEICHEN
0FA0			
0FA0	09	.BYT \$09	; EDIT
0FA1	66	.BYT \$66	; CURSOR LINKS
0FA2	6A	.BYT \$6A	; CURSOR RECHTS
0FA3	50	.BYT \$50	; CURSOR NACH UNTEN
0FA4	B5	.BYT \$B5	; CURSOR NACH OBEN
0FA5	70	.BYT \$70	; DELETE
0FA6	7E	.BYT \$7E	; ENTER
0FA7	CF	.BYT \$CF	; SYMBOL SHIFT
0FA8	D4	.BYT \$D4	; GRAPHICS
0FA9			
0FA9			; 'EDIT-KEY'-ROUTINE
0FA9			
0FA9	2A 49 5C	LD HL,(EPPC)	; ZEILENNUMMER HOLEN
0FAC	FD CB 37 6E	BIT 5,(IY+55)	; SPRUNG, WENN IM
0FB0	C2 97 10	JP NZ,\$1097	; 'INPUT'-MODUS
0FB3	CD 6E 19	CALL \$196E	; STARTADRESSE DER ZEILE SUCHEN
0FB6	CD 95 16	CALL \$1695	; UND DIE ZEILENNUMMER DAZU
0FB9	7A	LD A,D	; ZEILENNUMMER = 0 ?
0FBA	B3	OR E	
0FBB	CA 97 10	JP Z,\$1097	; NUR EDITORBEREICH LÖSCHEN
0FBE	E5	PUSH HL	; ADRESSE DER ZEILE RETTEN
0FBF	23	INC HL	; DIE LÄNGE DER
0FC0	4E	LD C,(HL)	; ZEILE HOLEN
0FC1	23	INC HL	
0FC2	46	LD B,(HL)	
0FC3	21 0A 00	LD HL,10	; 10 ZUR ZEILENLÄNGE ADDIEREN,
0FC6	09	ADD HL,BC	; UM FESTZUSTELLEN, OB NOCH
0FC7	44	LD B,H	; GENÜGEN PLATZ VORHANDEN IST
0FCB	4D	LD C,L	
0FC9	CD 05 1F	CALL \$1F05	; SPEICHERPLATZTEST
0FCC	CD 97 10	CALL \$1097	; EDITORBEREICH LÖSCHEN
0FCF	2A 51 5C	LD HL,(CURCHL)	; AKTUELLE KANALADRESSE LADEN
0FD2	E3	EX (SP),HL	; UND RETTEN, ADRESSE DER ZEILE
0FD3	E5	PUSH HL	; HOLEN UND ZWISCHENSPEICHERN
0FD4	3E FF	LD A,\$FF	; KANAL R ÖFFNEN, UM DIE ZEILE IN
0FD6	CD 01 16	CALL \$1601	; DEN EDITORBEREICH ZU KOPIEREN
0FD9	E1	POP HL	; STARTADRESSE DER ZEILE
0FDA	2B	DEC HL	; UND UM 1 VERMINDERN
0FDB	FD 35 0F	DEC (IY+\$F)	; ZEILENNUMMER -1, DAMIT KEIN
0FDE			; CURSOR GEDRUCKT WIRD
0FE	CD 55 18	CALL \$1855	; BASICZEILE LISTEN
0FE1	FD 34 0F	INC (IY+\$F)	; ZEILENNUMMER WIEDER NORMAL
0FE4	2A 59 5C	LD HL,(ELINE)	; START DER ZEILE IM EDITOR-
0FE7	23	INC HL	; BEREICH HOLEN, ZEILENNUMMER
0FE8	23	INC HL	; UND LÄNGE ÜBERGEHEN, UM
0FE9	23	INC HL	
0FEA	23	INC HL	; DIE ADRESSE FÜR
0FEB	22 5B 5C	LD (KCUR),HL	; KCUR ZU FINDEN

OFFE E1	POP HL	; VORHERIGE KANALADRESSE
OFFF CD 15 16	CALL \$1615	; HOLEN UND DIE FLAGS SETZEN
OFF2 C9	RET	; RETURN NACH EDITORSCHLEIFE
OFF3	.END	
OFF3	.LIB SPEC1000-S	
OFF3	; SINCLAIR ZX SPECTRUM TEIL 1000	
OFF3	;	
OFF3	; CURSOR-DOWN-ROUTINE	
OFF3	;	
OFF3 FD CB 37 6E	BIT 5, (IY+\$37)	; INPUT MODUS ?
OFF7 20 08	JR NZ, \$1001	; JA
OFF9 21 49 5C	LD HL, EPPC	
OFFC CD 0F 19	CALL \$190F	; NACHSTE ZEILENNUMMER SUCHEN
OFFF 18 6D	JR \$106E	; UND EIN LISTING AUSGEBEN
1001 FD 36 00 10	LD (IY+0), 16	; ERRNR: STOP IN INPUT
1005 18 1D	JR \$1024	
1007	;	
1007	; CURSOR EINS NACH LINKS	
1007	;	
1007 CD 31 10	CALL \$1031	; CURSOR BEWEGEN
100A 18 05	JR SETKCU	; UND KCUR SETZEN
100C	;	
100C	; CURSOR EINS NACH RECHTS	
100C	;	
100C 7E	LD A, (HL)	; AKTUELLES ZEICHEN AUF
100D FE 0D	CP \$D	; CARRIAGE RETURN PRÜFEN
100F CB	RET Z	; JA: RETURN
1010 23	INC HL	; SONST KCUR AUF NACHSTES
1011 22 5B 5C	SETKCU LD (KCUR), HL	; ZEICHEN SETZEN
1014 C9	RET	
1015	;	
1015	; LÖSCHEN EINES ZEICHENS BEIM EDITIEREN	
1015	;	
1015 CD 31 10	CALL \$1031	; CURSOR NACH LINKS
1018 01 01 00	LD BC, 1	; UND DAS ZEICHEN
101B C3 EB 19	JP RAUS2	; ENTFERNEN
101E	;	
101E CD D4 15	CALL WARTA	; ZWEI ZEICHEN VON DER
1021 CD D4 15	CALL WARTA	; TASTATUR WEGWERFEN
1024	;	
1024 E1	POP HL	; AUFRUF VON EDITOR UND
1025 E1	POP HL	; EDITOR-ERROR WEGWERFEN
1026 E1	POP HL	; ALTEN WERT VON ERRSP
1027 22 3D 5C	LD (ERRSP), HL	; WIEDER LADEN
102A FD CB 00 7E	BIT 7, (IY+0)	; RETURN, FALLS KEIN
102E C0	RET NZ	; FEHLER AUFGETRETEN IST
102F F9	LD SP, HL	; SONST SPRUNG IN
1030 C9	RET	; ERRORROUTINE
1031	;	
1031	; CURSOR NACH LINKS BIS MAXIMAL AN DEN ANFANG DER ZEILE	
1031	; BEWEGEN. HL ZEIGT AUF CURSORPOSITION	
1031	;	
1031 37	SCF	; DE AUF ELINE (EDITIEREN) ODER
1032 CD 95 11	CALL EDDE	; AUF WORKSP (INPUT) SETZEN
1035 ED 52	SBC HL, DE	; CARRY WIRD GESETZT, FALLS DER
1037 19	ADD HL, DE	; CURSOR AM ANFANG DER ZEILE IST
1038 23	INC HL	
1039 C1	POP BC	; EINMAL RETURNADRESSE WEGWERFEN

103A DB	RET C	; RETURN IN EDITORSCHLEIFE, WENN
103B		; CURSOR AM ANFANG DER ZEILE IST
103B C5	PUSH BC	; RETURNADRESSE WIEDER SETZEN
103C 44	LD B,H	; CURSORADRESSE NACH BC
103D 4D	LD C,L	
103E		
103E 62	LD H,D	; ZEICHENADRESSE NACH HL
103F 6B	LD L,E	
1040 23	INC HL	; +1
1041 1A	LD A,(DE)	; ZEICHEN HOLEN
1042 E6 F0	AND \$F0	; ZEICHEN GRÖßER \$0F
1044 FE 10	CP \$10	; UND KLEINER \$20 ?
1046 20 09	JR NZ,\$1051	; NEIN
1048 23	INC HL	; FÜR PARAMETER +1
1049 1A	LD A,(DE)	; NOCH MAL DAS ZEICHEN HOLEN
104A D6 17	SUB \$17	
104C CE 00	ADC 0	; \$16 UND \$17 ERGIBT HIER \$00
104E 20 01	JR NZ,\$1051	
1050 23	INC HL	
1051 A7	AND A	; CARRY LÖSCHEN
1052 ED 42	SBC HL,BC	
1054 09	ADD HL,BC	; HL WIE VORHER
1055 EB	EX DE,HL	
1056 3B E6	JR C,\$103E	
1058 C9	RET	
1059		
1059		; CURSOR EINS NACH OBEN BEWEGEN
1059		
1059 FD CB 37 6E	BIT 5,(IY+\$37)	; FLAGX: INPUT MODUS ?
105D C0	RET NZ	; JA
105E		
105E 2A 49 5C	LD HL,(EPPC)	; AKTUELLE ZEILENNUMMER UND
1061 CD 6E 19	CALL \$196E	; DIE STARTADRESSE HOLEN
1064 EB	EX DE,HL	; HL AUF EINE ZEILE DAVOR SETZEN
1065 CD 95 16	CALL ZSUCHE	; UND DEREN ZEILENNUMMER HOLEN
1068 21 4A 5C	LD HL,EPPC+1	
106B CD 1C 19	CALL \$191C	; ZEILENNUMMER SPEICHERN UND
106E CD 95 17	CALL LISTAU	; EIN LISTING AUSGEBEN
1071 3E 00	LD A,0	; KANAL K WIEDER ERÖFFNEN
1073 C3 01 16	JP OPKAN	
1076		
1076 FD CB 37 7E	BIT 7,(IY+\$37)	; FLAGX
107A 2B AB	JR Z,\$1024	; NUR BEI 'INPUT LINE'
107C		; NICHT SPRINGEN
107C C3 81 0F	JP \$F81	
107F		
107F		; EINSTIEG, WENN EIN FEHLER BEIM EDITIEREN AUFTRAT
107F		
107F FD CB 30 66	BIT 4,(IY+4B)	; KANAL K ?
1083 2B A1	JR Z,\$1026	; NEIN
1085 FD 36 00 FF	LD (IY+0),\$FF	; FEHLERNUMMER LÖSCHEN
1089 16 00	LD D,0	
108B FD 5E FE	LD E,(IY+\$FE)	; EIN PIEPSEK ALS
108E 21 90 1A	LD HL,\$1A90	
1091 CD B5 03	CALL \$3B5	; WARNUNG AUSGEBEN
1094 C3 30 0F	JP \$F30	; UND ZUM EDITOR ZURÜCK
1097		
1097		; EDITORBEREICH ODER WORKSPACE LÖSCHEN

1097		PUSH HL	; POINTER AUF FREIEN PLATZ RETTEN
1097	E5	CALL EDHLD	; DE AUF ERSTES UND HL AUF
1098	CD 90 11	DEC HL	; LETZTES ZEICHEN SETZEN
1098	2B	CALL RAUS1	; SPEICHER FREIGEBEN
109C	CD E5 19	LD (KCUR),HL	; CURSORADRESSE UND
109F	22 5B 5C	LD (IY+7),0	; MODUS 'K' SETZEN
10A2	FD 36 07 00	POP HL	
10A6	E1	RET	
10A7	C9		
10A8			
10A8			
10A8			
10A8			
10A8			
10A8	FD CB 02 5E	TASTIN BIT 3, (IY+2)	; BEI MODUSWECHSEL DIE EDITOR-
10AC	C4 1D 11	CALL NZ, \$111D	; ZEILE ANZEIGEN
10AF	A7	AND A	; CARRY LÖSCHEN
10B0	FD CB 01 6E	BIT 5, (IY+1)	; CARRY UND ZERO FLAGS SIND
10B4	CB	RET Z	; GELÖSCHT, WENN KEINE TASTE
10B5			; GEDRÜCKT WURDE
10B5	3A 0B 5C	LD A, (LASTK)	; SONST TASTENCODE HOLEN UND
10B8	FD CB 01 AE	RES 5, (IY+1)	; DIESES VERMERKEN
10BC	F5	PUSH AF	; RETTE TASTE
10BD	FD CB 02 6E	BIT 5, (IY+2)	; FALLS NOTIG, DEN UNTEREN
10C1	C4 6E 0D	CALL NZ, \$D6E	; TEIL DES BILDSCHIRMS LÖSCHEN
10C4	F1	POP AF	; TASTE IN A ZURÜCK
10C5	FE 20	CP #	; ALLE ASCII-ZEICHEN UND TOKENS
10C7	30 52	JR NC, \$111B	; ÜBERNEHMEN
10C9			
10C9	FE 10	CP \$10	; STEUERZEICHEN VON \$10 BIS \$1F
10CB	30 2D	JR NC, \$10FA	
10CD			
10CD	FE 06	CP 6	; 'MODE'-ZEICHEN UND 'CAPS'-LOCK
10CF	30 0A	JR NC, MODCAP	
10D1			
10D1	47	LD B, A	; NUR FLASH, BRIGHT UND INVERSE
10D2			; CODES BEHANDELN
10D2	E6 01	AND 1	; BIT 0 FÜR AUS (0) ODER EIN (1)
10D4	4F	LD C, A	; AUSBLENDEN UND IN C BRINGEN
10D5	7B	LD A, B	; ZEICHEN WIEDER ZURÜCK
10D6	1F	RRA	; TASTENCODES FÜR FLASH IN \$12,
10D7	C6 12	ADD \$12	; BRIGHT \$13 UND INVERSE \$14
10D9	1B 2A	JR \$1105	; WANDELN
10DB			
10DB	20 09	MODCAP JR NZ, MODES	; BEI 'MODE'-CODES
10DD	21 6A 5C	LD HL, FLAGS2	; NUR 'CAPS'-LOCK BEHANDELN
10E0	3E 0B	LD A, B	; DAZU BIT 3 VON FLAGS2
10E2	AE	XOR (HL)	
10E3	77	LD (HL), A	; INVERTIEREN
10E4	1B 0E	JR \$10F4	
10E6			
10E6	FE 0E	MODES CP \$E	; NUR \$0E (SYMBOL SHIFT) UND \$0F
10E8	DB	RET C	; (GRAPHIKS) WEITER, REST RETURN
10E9	D6 0D	SUB \$D	
10EB	21 41 5C	LD HL, MODE	
10EE	BE	CP (HL)	; HAT DER MODUS SICH GEÄNDERT ?
10EF	77	LD (HL), A	; NEUEN MODUS MERKEN
10F0	20 02	JR NZ, \$10F4	; JA, VERÄNDERUNG

```

10F2      ;
10F2 36 00      LD (HL),0      ; SONST L-MODUS
10F4 FD CB 02 DE SET 3,(1Y+2)    ; ANMERKEN: DER MODUS KANN SICH
10F8 BF      CP A      ; GEANDERT HABEN UND CARRY
10F9 C9      RET      ; FÜR RETURN LÖSCHEN
10FA      ;
10FA 47      LD B,A
10FB E6 07      AND 7      ; NUR BIT 0 - 2 ZULASSEN
10FD 4F      LD C,A      ; UND IN C BRINGEN
10FE 3E 10      LD A,$10      ; CODE FÜR 'INK'
1100 CB 58      BIT 3,B      ; FALLS UNGESHIFTETER CODE,
1102 20 01      JR NZ,$1105
1104 3C      INC A      ; DANN NACH A DEN 'PAPER'-CODE
1105 FD 71 D3      LD (1Y+$D3),C ; PARAMETER SPEICHERN
1108 11 0D 11      LD DE,TAST12 ; INPUT ÄNDERN
110B 18 06      JR $1113
110D      ;
110D 3A 0D 5C      TAST12 LD A,(KDATA) ; NOCH EINE TASTE ALS PARAMETER
1110 11 AB 10      LD DE,TASTIN ; HOLEN UND INPUT WIEDER NORMAL
1113      ;
1113 2A 4F 5C      LD HL,(CHANS) ; AKTUELLE KANALADRESSE
1116 23      INC HL ; UND AUF INPUT ZEIGEN LASSEN
1117 23      INC HL
1118 73      LD (HL),E ; DEN NEUEN INPUT-EINSPRUNG
1119 23      INC HL ; SPEICHERN
111A 72      LD (HL),D
111B 37      SCF
111C C9      RET
111D      ;
111D      ; DIESE ROUTINE WIRD IMMER AUFGERUFEN, WENN DER
111D      ; EDITOR- ODER INPUTBEREICH IN DEN UNTEREN
111D      ; BILDSCHIRMTAIL GESCHRIEBEN WERDEN SOLL
111D      ;
111D CD 4D 0D      CALL AKTCOL ; AKTUELLE FARBEN BLEIBEN
1120 FD CB 02 9E      RES 3,(1Y+2) ; KEINE MODE-ÄNDERUNG
1124 FD CB 02 AE      RES 5,(1Y+2) ; UNTEREN TEIL NICHT LÖSCHEN
1128 2A 8A 5C      LD HL,(SPOSNL) ; AKT. WERTE DES UNTEREN BILD-
112B E5      PUSH HL ; SCHIRMTAILS RETTEN
112C 2A 3D 5C      LD HL,(ERRSP) ; DEN ERRORSTACKPOINTER
112F E5      PUSH HL ; ZWISCHENSPEICHERN,
1130 21 67 11      LD HL,EDERR ; ALS FEHLERRETURNADRESSE
1133 E5      PUSH HL ; AUF DEN STACK LEGEN UND
1134 ED 73 3D 5C      LD (ERRSP),SP ; STACKPOINTER HIERFÜR MERKEN
1138 2A 82 5C      LD HL,(ECHOE) ; ECHOE ZWISCHENSPEICHERN (END
113B E5      PUSH HL ; OF INPUT)
113C 37      SCF
113D CD 95 11      CALL EDDE ; DE AUF WORKSPACE SETZEN
1140 EB      EX DE,HL ; HL = ANFANG, DE = ENDE
1141 CD 7D 18      CALL $187D ; UND ZEILE AUSGEBEN
1144 EB      EX DE,HL
1145 CD E1 18      CALL $18E1 ; CURSOR ANZEIGEN
1148 2A 8A 5C      LD HL,(SPOSNL) ; AKTUELLEN WERT SPOSNL MIT
114B E3      EX (SP),HL ; ECHOE AUSTAUSCHEN
114C EB      EX DE,HL ; UND ECHOE NACH DE
114D CD 4D 0D      CALL AKTCOL ; AKTUELLE FARBEN NOCH MAL SETZEN
1150      ;
1150      ; DEN REST EINER ZEILE MIT LEERZEICHEN FÜLLEN
1150      ;

```

```

1150 3A 8B 5C RESTLE LD A, (SPOSNL+1) ; AKTUELLE BILDSCHIRMZEILE MIT
1153 92 SUB D ; DER ALTEN VERGLEICHEN
1154 38 26 JR C, $117C ; KEINE LEERZEICHENAUSGABE NOTIG
1156 20 06 JR NZ, REST1 ; NICHT DIE GLEICHE ZEILE
1158 7B LD A, E ; VON DER ALTEN DIE NEUE SPALTEN-
1159 FD 96 50 SUB (IY+$50) ; POSITION ABZIEHEN
115C 30 1E JR NC, $117C ; KEINE LEERZEICHEN ERFORDERLICH
115E 3E 20 REST1 LD A, ; LEERZEICHEN LADEN
1160 D5 PUSH DE ; POINTER ZWISCHENSPEICHERN
1161 CD F4 09 CALL $9F4 ; AUSGABE
1164 D1 POP DE
1165 1B E9 JR RESTLE ; UND WEITER
1167 ;
1167 ; FEHLERBEHANDLUNG IM EDITOR
1167 ;
EDERR LD D, 0
LD E, (IY+$FE) ; =RASP
LD HL, $1A90
CALL $3B5 ; WARNPIEPSEER AUSGEBEN
LD (IY+0), $FF ; ERRORNUMMER LÖSCHEN
LD DE, (SPOSNL) ; AKTUELLEN WERT SPOSNL FÜR
JR $117E ; ECHOE HOLEN

;
; NORMALER AUSSTIEG BEI AUSGABE DER EDITOR- ODER INPUT-
; INPUTZEILE
;
POP DE ; NEUE POSITION FÜR ECHOE
POP HL ; FEHLERADRESSE WEGWERFEN
POP HL ; ALTEN ERRORSTACKPOINTER
LD (ERRSP), HL ; ZURÜCKHOLEN
POP BC ; SPOSNL ALT HOLEN
PUSH DE ; SPOSNL NEU MERKEN
CALL $DD9 ; SYSTEMVARIABLE SETZEN
POP HL ; SPOSNL NEU IN ECHOE SPEICHERN
LD (ECHOE), HL
LD (IY+$26), 0 ; ERRORZEIGER LÖSCHEN
RET

;
; HL AUF DIE LETZTE POSTION, DE AUF DIE ERSTE ENTWEDER
; DES EDITORBEREICHS ODER DES WORKSPACEBEREICHS SETZEN
;
EDHLE LD HL, (WORKSP) ; ENDE EDITORBEREICH
DEC HL ; CARRY LÖSCHEN
AND A ; ANFANG EDITORBEREICH
LD DE, (ELINE) ; EDITORMODE RETURN
BIT 5, (IY+$37)
RET Z
LD DE, (WORKSP) ; SONST DE AUF WORKSPACE
RET C
LD HL, (STKBOT) ; HL EVENTUELL AUF STKBOT SETZEN
RET

;
; ROUTINE HOLT DIE VERSTECKEN FLOATINGPOINT-ZAHLEN
; EINER BASIC-ZEILE ZURÜCK
;
HOLFLO LD A, (HL) ; IST DAS ZEICHEN EIN MERKER FÜR
CP $E ; EINE FLOATINGPOINT-ZAHL ?
LD BC, 6 ; FALLS JA, DANN 6 PLATZE

```



```

11AD CC E8 19      CALL Z,RAUS2      ; BESCHAFFEN
11B0 7E            LD A,(HL)        ; NOCH MAL HOLEN ZUR ÜBERPRÜFUNG
11B1 23            INC HL           ; BASICZEILENPOINTER +1
11B2 FE 0D        CP $0D           ; ENDE BEI CARRIAGE RETURN
11B4 20 F1        JR NZ,HOLFLO
11B6 C9           RET
11B7
11B7 ;
11B7 ; =====
11B7 ;
11B7 ; ES FOLGT DIE SYSTEMINITIALISIERUNG BEIM BEFEHL
11B7 ; 'NEW' ODER DURCH RESET BEIM EINSCHALTEN
11B7 ; UNTERSCHIEDUNG DURCH REG A: $FF = 'NEW'
11B7 ;
11B7 NEW DI        ; INTERRUPT SPERREN
11B8 3E FF        LD A,$FF        ; NEW MERKEN
11BA ED 5B 82 5C  LD DE,(RAMTOP)  ; DIE ALTE RAMOBERGRENZE MERKEN
11BE D9           EXX             ; ZWEITEN REGISTERSATZ LADEN
11BF ED 4B 84 5C  LD BC,(PRAMT)   ; LETZTES SPEICHERBYTE
11C3 ED 5B 38 5C  LD DE,(RASP)    ; PIEPSLANGE EINER WARNUNG
11C7 2A 7B 5C     LD HL,(UDG)
11CA D9           EXX
11CB
11CB ; =====
11CB ;
11CB ; RESETROUTINE MIT VORBEREITUNG ALLER POINTER NACH START
11CB ;
11CB ; =====
11CB RESET1 LD B,A      ; ZWISCHENSPEICHERN
11CC 3E 07        LD A,7          ; BORDER COLOUR WEISS
11CE D3 FE        OUT ($FE),A     ; SETZEN
11D0 3E 3F        LD A,$3F
11D2 ED 47        LD I,A
11D4 00           NOP             ; KURZE VERZÖGERUNG
11D5 00           NOP
11D6 00           NOP
11D7 00           NOP
11D8 00           NOP
11D9 00           NOP
11DA 62           LD H,D          ; RAM-TEST: BEI NEW = RAMTOP
11DB 6B           LD L,E          ; BEI RESET = $FFFF
11DC 36 02        LD (HL),2       ; DAS GESAMTE RAM
11DE 2B           DEC HL          ; OBERHALB VON $3FFF
11DF BC           CP H            ; MIT $02 BESCHREIBEN
11E0 20 FA        JR NZ,$11DC
11E2 A7           RANTES AND A    ; CARRY LÖSCHEN
11E3 ED 52        SBC HL,DE       ; BEIM LETZTEN SPEICHERPLATZ IST
11E5 19           ADD HL,DE       ; CARRY GESETZT
11E6 23           INC HL          ; POINTER +1
11E7 30 06        JR NC,RAMFER    ; RAMENDE ERREICHT
11E9 35           DEC (HL)        ; SPEICHERPLATZ -1 ($01)
11EA 2B 03        JR Z,RAMFER     ; RAMERROR
11EC 35           DEC (HL)        ; SPEICHER -1 ($00)
11ED 2B F3        JR Z,RANTES     ; ZELLE OK
11EF
11EF RAMFER DEC HL  ; ZEIGT AUF LETZTE RAMSTELLE
11F0 D9           EXX            ; BEI 'NEW':
11F1 ED 43 B4 5C  LD (PRAMT),BC   ; SYSTEM VARIABLE SETZEN,
11F5 ED 53 38 5C  LD (RASP),DE   ; BEI RESET OHNE BEDEUTUNG
11F9 22 7B 5C     LD (UDB),HL

```

11FC D9	EXX	; NORMALER REGISTERSATZ
11FD 04	INC B	
11FE 28 19	JR Z,SETTOP	; BEI 'NEW'
1200 22 B4 5C	LD (PRAMT),HL	; BEI RESET LETZTES BYTE SETZEN
1203 11 AF 3E	LD DE,\$3EAF	; LETZTES BYTE DES ZEICHENS 'U'
1206 01 AB 00	LD BC,\$AB	; = 21 (BUCHSTABEN) * 8 BYTES
1209 EB	EX DE,HL	; 21 BUCHSTABEN
120A ED 88	LDDR	; AN DIE RAMOBERGRENZE KOPIEREN
120C EB	EX DE,HL	
120D 23	INC HL	; ZEIGT AUF DEN ERSTEN KOPIERTEN
120E 22 7B 5C	LD (UDG),HL	; BUCHSTABEN UND SPEICHERN
1211 2B	DEC HL	; -1 FÜR RAMTOP
1212 01 40 00	LD BC,\$40	; RASP (WARNTONLÄNGE) UND PIP
1215 ED 43 38 5C	LD (RASP),BC	; (TASTATURKlick) SETZEN
1219 22 B2 5C	LD (RAMTOP),HL	; FREIEN RAMBEREICH SETZEN
121C 21 00 3C	LD HL,CHAR0-256	; VARIABLE CHARS SETZEN
121F 22 36 5C	LD (CHARS),HL	
1222		
1222 2A B2 5C	LD HL,(RAMTOP)	
1225 36 3E	LD (HL),\$3E	; LETZTE RAMZELLE AUF \$3E
1227 2B	DEC HL	; UND DIE VORLETZTE AUF 0 LASSEN
1228 F9	LD SP,HL	; STACKPOINTER SETZEN
1229 2B	DEC HL	
122A 2B	DEC HL	; UND -2 ERGIBT
122B 22 3D 5C	LD (ERRSP),HL	; ERROR STACKPOINTER
122E ED 56	IM 1	; INTERRUPTMODUS 1
1230 FD 21 3A 5C	LD IY,ERRNR	; IY BEHALT IMMER DIESEN WERT!
1234 FB	EI	; INTERRUPT FÜR TASTATURABFRAGE
1235		; UND INTERNE UHR FREIGEBEN
1235 21 B6 5C	LD HL,KANMEM	; BASISADRESSE FÜR
1238 22 4F 5C	LD (CHANS),HL	; KANALINFORMATIONEN
123B 11 AF 15	LD DE,\$15AF	; GRUNDDATEN DER
123E 01 15 00	LD BC,21	; KANALINFORMATION AUS TABELLE
1241 EB	EX DE,HL	; LADEN
1242 ED B0	LDIR	
1244		
1244 EB	EX DE,HL	
1245 2B	DEC HL	; DATADD AUF LETZTE ADRESSE
1246 22 57 5C	LD (DATADD),HL	; DER KANALINFO SETZEN
1249 23	INC HL	; +1 ERGIBT ANFANGSWERT
124A 22 53 5C	LD (PROG),HL	; FÜR PROG UND VARS
124D 22 4B 5C	LD (VARS),HL	
1250 36 80	LD (HL),\$80	; ENDEMARKIERUNG DER VARIABLEN
1252 23	INC HL	; +1 ERGIBT ANFANGSADRESSE
1253 22 59 5C	LD (ELINE),HL	; FÜR ELINE UND DIESE STELLE
1256 36 0D	LD (HL),\$D	; MIT CR BESCHREIBEN (NULLZEILE)
1258 23	INC HL	
1259 36 80	LD (HL),\$80	; NOCH EINE ENDEMARKIERUNG
125B 23	INC HL	; +1 ERGIBT BASISADRESSE
125C 22 61 5C	LD (WORKSP),HL	; FÜR WORKSPACE, UNGTERGRENZE
125F 22 63 5C	LD (STKBOT),HL	; DES CALCULATOR-STACK UND
1262 22 65 5C	LD (STKEND),HL	; DES SPARE-SPACE
10 1265 3E 38	LD A,\$38	; COLOUR-VARIABLEN AUF:
1267 32 BD 5C	LD (ATTRP),A	; FLASH 0, BRIGHT 0
126A 32 BF 5C	LD (ATTRT),A	; PAPER 7 UND INK 0 SETZEN
126D 32 4B 5C	LD (BORDER),A	
1270 21 23 05	LD HL,\$523	; REPDEL INITIALISIEREN
1273 22 09 5C	LD (REPDEL),HL	

1276	FD 35 C6	DEC (IY+\$C6)	; KSTATE 0 UND 4 AUF
1279	FD 35 CA	DEC (IY+\$CA)	; KEINE NEUE TASTE SETZEN
127C	21 C6 15	LD HL,\$15C6	; ANFANGSWERTE DER
127F	11 10 5C	LD DE,STRMS	; OFFENEN KANALE
1282	01 0E 00	LD BC,\$E	; AUS TABELLE KOPIEREN
1285	ED B0	LDIR	
1287	FD CB 01 CE	SET 1,(IY+1)	; PRINTER FLAG
128B	CD DF 0E	CALL \$EDF	; PRINTER-BUFFER LÖSCHEN
128E	FD 36 31 02	LD (IY+\$31),2	; UNTEREN TEIL DES BILDSCHIRMS
1292	CD 6B 0D	CALL \$D6B	; SETZEN UND KOMPLETT LÖSCHEN
1295	AF	XOR A	
1296	11 3B 15	LD DE,\$153B	; AUSGABE DES COPY-RIGHT
1299	CD 0A 0C	CALL \$C0A	; STATEMENTS
129C	FD CB 02 EE	SET 5,(IY+2)	; MERKER: UNTEREN BILDSCHIRMTEIL
12A0	1B 07	JR \$12A9	; LÖSCHEN UND IN DEN INTERPRETER
12A2			
12A2			=====
12A2			
12A2			; HAUPTSCHLEIFE DES AUSFÜHRUNGSPROGRAMMS
12A2			
12A2	FD 36 31 02	HAUPT LD (IY+\$31),2	; UNTERER BILDSCHIRMTEIL AUF
12A6			; ZWEI ZEILEN SETZEN
12A6	CD 95 17	CALL LISTAU	; LISTING AUSGEBEN
12A9	CD B0 16	CALL \$16B0	; EDITOR, WORKSPACE UND
12AC			; CALC.-STACK LÖSCHEN
12AC	3E 00	HAUEDI LD A,0	; KANAL 0 ERÖFFNEN VOR EINSPRUNG
12AE	CD 01 16	CALL OPKAN	; IN DEN EDITOR
12B1	CD 2C 0F	CALL \$F2C	; EDITORAUFRUF ZUM EINGEBEN
12B4			; EINER ZEILE
12B4	CD 17 1B	CALL \$1B17	; UND AUF SYNTAXFEHLER PRÜFEN
12B7	FD CB 00 7E	BIT 7,(IY+0)	
12BB	20 12	JR NZ,KORRIN	; KEIN SYNTAXFEHLER
12BD	FD CB 30 66	BIT 4,(IY+4B)	; FALLS EIN ANDERER KANAL
12C1	2B 40	JR Z,\$1303	; ALS 'K' BENUTZT WURDE, SPRUNG
12C3	2A 59 5C	LD HL,(ELINE)	; HL ZEIGT AUF ANFANG DER ZEILE
12C6	CD A7 11	CALL HOLFLO	
12C9	FD 36 00 FF	LD (IY+0),\$FF	; ERRORNUMMER LÖSCHEN
12CD	1B DD	JR HAUEDI	; UND OHNE DAS LISTING ZU ÄNDERN
12CF			; WIEDER IN DEN EDITOR
12CF			
12CF	2A 59 5C	KORRIN LD HL,(ELINE)	; ANFANG DER KORREKTEN ZEILE
12D2	22 5D 5C	LD (CHADD),HL	; NACH CHADD
12D5	CD FB 19	CALL \$19FB	; ZEILENUMMER IN 'BC' HOLEN
12D8	7B	LD A,B	
12D9	B1	OR C	; ZEILENUMMER IN ORDNUNG ?
12DA	C2 5D 15	JP NZ,\$155D	; JA: IN DAS PROGRAMM EINFÜGEN
12DD			
12DD	DF	RST GETAKT	; IST DAS ERSTE ZEICHEN
12DE	FE 0D	CP \$D	; DER ZEILE EIN CR ?
12E0	2B C0	JR Z,HAUPT	; JA
12E2	FD CB 30 46	BIT 0,(IY+4B)	; FALLS NOTIG, DEN GANZEN
12E6	C4 AF 0D	CALL NZ,\$DAF	; BILDSCHIRM LÖSCHEN
12E9	CD 6E 0D	CALL \$D6E	; UNTEREN TEIL IMMER LÖSCHEN
12EC	3E 19	LD A,\$19	; SCROLLING-ZÄHLER SETZEN
12EE	FD 96 4F	SUB (IY+\$4F)	
12F1	32 BC 5C	LD (SCRCT),A	
12F4	FD CB 01 FE	SET 7,(IY+1)	; ANMERKEN: ZEILE AUSFÜHREN
12F8	FD 36 00 FF	LD (IY+0),\$FF	; ERRORNUMMER LÖSCHEN

12FC	FD 36 0A 01	LD (IY+#A),1	
1300	CD 8A 1B	CALL \$1BBA	; UND DIE ZEILE INTERPRETIEREN
1303			
1303			; RÜCKKEHR NACH AUSFÜHRUNG DER ZEILE BZW. PROGRAMM
1303			; AN DIESE STELLE ZUR AUSGABE EINER MELDUNG
1303			; DER INTERRUPT MUSS FREIGEgeben SEIN !
1303			
1303	76	HALT	
1304	FD CB 01 AE	RES 5,(IY+1)	; BEREIT FÜR EINE NEUE TASTE
1308	FD CB 30 4E	BIT 1,(IY+4B)	; DEN EVTL. BENUTZTEN
130C	C4 CD 0E	CALL NZ,\$ECD	; PRINTERBUFFER LÖSCHEN
130F	3A 3A 5C	LD A,(ERRNR)	; ERRORNUMMER HOLEN
1312	3C	INC A	
1313	F5	PUSH AF	; UND RETTEN
1314	21 00 00	LD HL,0	; DIE VARIABLEN FLAGX, XFTR
1317	FD 74 37	LD (IY+\$37),H	; UND DEFADD AUF NULL SETZEN
131A	FD 74 26	LD (IY+\$26),H	
131D	22 0B 5C	LD (DEFADD),HL	
1320	21 01 00	LD HL,1	
1323	22 16 5C	LD (STRMS+6),HL	
1326	CD 80 16	CALL \$16B0	; ARBEITSSPEICHER UND CALCULATOR-
1329			; STACK RÜCKSETZEN
1329	FD CB 37 AE	RES 5,(IY+\$37)	; EDITOR-MODUS EIN
132D	CD 6E 0D	CALL \$D6E	; UNTEREN BILDSCHIRM LÖSCHEN
1330	FD CB 02 EE	SET 5,(IY+2)	; UND FLAG ZUM LÖSCHEN SETZEN
1334	F1	POP AF	; ERRORNUMMER WIEDER HOLEN
1335	47	LD B,A	
1336	FE 0A	CP \$A	; ERRORNUMMERN 0-9 ?
1338	38 02	JR C,ERRAU	; JA
133A	C6 07	ADD 7	; OFFSET ADDIEREN
133C	CD EF 15	CALL \$15EF	; AUSGABE DER ERRORNUMMER
133F	3E 20	LD A,0	; UND EINEN SPACE ZUSÄTZLICH
1341	D7	RST PRTOU	
1342	78	LD A,B	; NUMMER WIEDER HOLEN,
1343	11 91 13	LD DE,MELDU	; LADEN DER TABELLENADRESSE DER
1346	CD 0A 0C	CALL \$COA	; (ERROR-)MELDUNGEN UND AUSGABE
1349	AF	XOR A	
134A	11 36 15	LD DE,\$1536	; ZUSÄTZLICH KOMMA
134D	CD 0A 0C	CALL \$COA	; UND SPACE AUSGEBEN
1350	ED 4B 45 5C	LD BC,(PPC)	; AKTUELLE ZEILENNUMMER HOLEN
1354	CD 1B 1A	CALL \$1A1B	; UND AUSGEBEN
1357	3E 3A	LD A,0	; DOPPELPUNKT DANACH
1359	D7	RST PRTOU	
135A	FD 4E 0D	LD C,(IY+\$D)	; AKTUELLE STATEMENTNUMMER
135D	06 00	LD B,0	; IN BC HOLEN
135F	CD 1B 1A	CALL \$1A1B	; UND AUSGABE DES STATEMENTS
1362	CD 97 10	CALL \$1097	; EDITOR-SPEICHER LÖSCHEN
1365	3A 3A 5C	LD A,(ERRNR)	; ERRORNUMMER HOLEN
1368	3C	INC A	
1369	2B 1B	JR Z,\$1386	; KEIN ERROR: NORMALES ENDE
136B	FE 09	CP 9	; STOP-BEFEHL ?
136D	2B 04	JR Z,\$1373	; JA
136F	FE 15	CP \$15	; BREAKSTATEMENT ?
1371	20 03	JR NZ,\$1376	; NEIN
1373	FD 34 0D	INC (IY+\$D)	; SONST NACH 'CONT'-EINGABE AB
1376			; NÄCHSTEM BEFEHL FORTFAHREN
1376	01 03 00	LD BC,3	; OLDPPC UND OSPCC MÜSSEN FÜR
1379	11 70 5C	LD DE,OSPCC	; 'CONT' GESETZT WERDEN

```

137C 21 44 5C      LD HL,NSPPC
137F CB 7E         BIT 7,(HL)      ; ERFOLGTE 'BREAK' DIREKT NACH
1381                                     ; EINEM SPRUNGBEFEHL?
1381 28 01         JR Z,$1384      ; JA, DANN NEWPPC UND NSPPC
1383                                     ; NEHMEN
1383 09           ADD HL,BC        ; SONST PPC UND SUBPPC
1384 ED B8         LDDR
1386 FD 36 0A FF   LD (IY+$A),$FF ; NSPPC LÖSCHEN: KEIN SPRUNG
138A FD CB 01 9E   RES 3,(IY+1)   ; FLAGS: K-MODUS SETZEN
138E C3 AC 12      JP HAUEDI      ; ZURÜCK IN DIE HAUPTSCHLEIFE
1391                                     .END
1391                                     .LIB SPEC1400-S
1391               ; SINCLAIR ZX SPECTRUM TEIL 1400
1391               ;
1391               ; MELDUNGEN DES BETRIEBSSYSTEMS
1391               ; DAS LETZTE BYTE EINER MELDUNG IST MIT $80 GEODERT
1391               ;
1391 80           MELDU .BYT $80      ; WIRD ÜBERSPRUNGEN
1392 4F         MELD0 .BYT '0',$CB  ; OK
1393 CB
1394 4E 45      MELD1 .BYT 'NEXT without FO',$D2
13A3 D2
13A4 56 61      MELD2 .BYT 'Variable not foun',$E4
13B5 E4
13B6 53 75      MELD3 .BYT 'Subscript wron',$E7
13C4 E7
13C5 4F 75      MELD4 .BYT 'Out of memor',$F9
13D1 F9
13D2 4F 75      MELD5 .BYT 'Out of scree',$EE
13DE EE
13DF 4E 75      MELD6 .BYT 'Number too bi',$E7
13EC E7
13ED 52 45      MELD7 .BYT 'RETURN without GOSU',$C2
1400 C2
1401 45 6E      MELD8 .BYT 'End of fil',$E5
140B E5
140C 53 54      MELD9 .BYT 'STOP statemen',$F4
1419 F4
141A 49 6E      MELDA .BYT 'Invalid argumen',$F4
1429 F4
142A 49 6E      MELDB .BYT 'Integer out of rang',$E5
143D E5
143E 4E 6F      MELDC .BYT 'Nonsense in BASI',$C3
144E C3
144F 42 52      MELDD .BYT 'BREAK - CONT repeat',$F3
1462 F3
1463 4F 75      MELDE .BYT 'Out of DAT',$C1
146D C1
146E 49 6E      MELDF .BYT 'Invalid file nam',$E5
147E E5
147F 4E 6F      MELDG .BYT 'No room for lin',$E5
148E E5
148F 53 54      MELDH .BYT 'STOP in INPU',$D4
149B D4
149C 46 4F      MELDI .BYT 'FOR without NEX',$D4
14AB D4
14AC 49 6E      MELDJ .BYT 'Invalid I/O devic',$E5
14BD E5

```


14BE 49 6E	MELDK .BYT 'Invalid colour', \$F2
14CB F2	
14CC 42 52	MELDL .BYT 'BREAK into progr', \$ED
14DD ED	
14DE 52 41	MELDM .BYT 'RAMTOP no goo', \$E4
14EB E4	
14EC 53 74	MELDN .BYT 'Statement los', \$F4
14F9 F4	
14FA 49 6E	MELDO .BYT 'Invalid strea', \$ED
1507 ED	
1508 46 4E	MELDP .BYT 'FN without DE', \$D6
1515 C6	
1516 50 61	MELDQ .BYT 'Parameter erro', \$FC
1524 F2	
1525 54 61	MELDR .BYT 'Tape loading erro', \$FC
1536 F2	
1537 2C	MELDS .BYT \$2C, \$A0 ; ", "
1538 A0	
1539 7F	COPRIG .BYT \$7F, ' 1982 Sinclair Research Ltd', \$E4
153A 20 31	
1554 E4	
1555	;
1555 3E 10	NOROOM LD A, \$10 ; MELDUNG 'G'
1557 01 00 00	LD BC, 0 ; BC LOSCHEN
155A C3 13 13	JP \$1313
155D	;
155D	; EINE NEUE BASIC-ZEILE INS PROGRAMM EINFÜGEN
155D	; WENN DIE ZEILE SCHON EXISTIERT, DANN ERSETZEN ODER,
155D	; FALLS NUR EINE ZEILENNUMMER EINGEGEBEN WURDE,
155D	; DIESE ZEILE LÖSCHEN
155D	;
155D ED 43 49 SC	EINFUE LD (EPPC), BC ; NEUE ZEILE ZUR AKTUELLEN MACHEN
1561 2A 5D 5C	LD HL, (CHADD) ; CHADD IN BC BRINGEN
1564 EB	EX DE, HL
1565 21 55 15	LD HL, NOROOM ; ADRESSE DER MELDUNG AUF STACK
1568 E5	PUSH HL
1569 2A 61 5C	LD HL, (WORKSP) ; LÄNGE DER NEUEN ZEILE AB ZEILENNUMMER BIS EINSCHLIESSLICH
156C 37	SCF ; CARRIAGE RETURN BERECHNEN
156D ED 52	SBC HL, DE ; UND ZWISCHENSPEICHERN
156F E5	PUSH HL ; ZEILENNUMMER INS HL REG
1570 60	LD H, B ; UM ÜBERPRÜFEN ZU KÖNNEN, OB
1571 69	LD L, C ; DIESE ZEILE SCHON EXISTIERT
1572 CD 6E 19	CALL \$196E ; NICHT VORHANDEN: NUR EINFÜGEN
1575 20 06	JR NZ, EINFU1 ; LÄNGE DER ALTEN ZEILE BERECHNEN
1577 CD B8 19	CALL \$19B8 ; UND ENTFERNEN
157A CD EB 19	CALL RAUS2
157D	;
157D C1	EINFU1 POP BC ; LÄNGE DER NEUEN ZEILE HOLEN
157E 79	LD A, C ; UND AUF 'NULL'-LÄNGE (OHNE CP)
157F 3D	DEC A ; PRÜFEN:
1580 B0	OR B
1581 2B 2B	JR Z, \$15AB ; JA: ZEILE NUR ENTFERNEN,
1583	; NICHTS EINFÜGEN
1583	;
1583 C5	PUSH BC ; LÄNGE NEU RETTEN
1584 03	INC BC ; 4 ADDIEREN, UM DIE
1585 03	INC BC ; ZEILENNUMMER UND DIE
1586 03	INC BC ; LÄNGE MIT DER ZEILE

1587	03	INC BC	; ABSPEICHERN ZU KÖNNEN
1588	2B	DEC HL	; ZEIGT AUF EIN BYTE VOR DER
1589			; EINZUFÜGENDEN STELLE
1589	ED 5B 53 5C	LD DE, (PROG)	; ZWISCHENSPEICHERN
158D	D5	PUSH DE	
158E	CD 55 16	CALL MACHPL	; PLATZ FÜR NEUE ZEILE SCHAFFEN
1591	E1	POP HL	; UND PROG WIEDER
1592	22 53 5C	LD (PROG), HL	; AUF ALTEN WERT SETZEN
1595	C1	POP BC	; ZEILENLANGE IN BC HOLEN
1596	C5	PUSH BC	
1597	13	INC DE	; DE ZEIGT AUF DAS ENDE DES
1598			; FREIGEMACHTEN SPEICHERS
1598	2A 61 5C	LD HL, (WORKSP)	; HL AUF ENDE
1598	2B	DEC HL	
159C	2B	DEC HL	; DER EINZUFÜGENDE ZEILE SETZEN
159D	ED 88	LDDR	; UND DIE NEUE ZEILE EINSPEICHERN
159F	2A 49 5C	LD HL, (EPPC)	; ZEILENNUMMER HOLEN
15A2	EB	EX DE, HL	; UND NACH DE BRINGEN
15A3	C1	POP BC	; LANGE WIEDER HOLEN
15A4	70	LD (HL), B	; DIESE PARAMETER
15A5	2B	DEC HL	; VOR DIE EINGEBAUTE
15A6	71	LD (HL), C	; ZEILE EINFÜGEN
15A7	2B	DEC HL	
15A8	73	LD (HL), E	; ZEILENNUMMER LOW
15A9	2B	DEC HL	
15AA	72	LD (HL), D	; ZEILENNUMMER HIGH
15AB	F1	POP AF	; ADRESSE DER NOROOM-MELDUNG VER-
15AC	C3 A2 12	JP HAUPT	; NICHTEN UND LISTING AUSGEBEN
15AF			
15AF			; KANALINFORMATIONEN ZUM INITIALISIEREN
15AF			; K = KEYBOARD (TASTATUR)
15AF			; S = SCREEN (BILDSCHIRM)
15AF			; R = WORKSPACE (ARBEITSSPEICHER)
15AF			; P = PRINTER (DRUCKER)
15AF			
15AF	F4	INIKAN .BYT \$F4, \$09	; NORMALE AUSGABE
15B0	09		
15B1	AB	.BYT \$AB, \$10	; TASTATUR-INPUT
15B2	10		
15B3	4B	.BYT 'K'	
15B4	F4	.BYT \$F4, \$09	; NORMALE AUSGABE
15B5	09		
15B6	C4 15	.WOR INVIO	
15B8	53	.BYT 'S'	
15B9	81	.BYT \$81, \$0F	; ADDIERE ZEICHEN
15BA	0F		
15BB	C4 15	.WOR INVIO	
15BD	52	.BYT 'R'	
15BE	F4	.BYT \$F4, \$09	; NORMALE AUSGABE
15BF	09		
15C0	C4 15	.WOR INVIO	
15C2	50	.BYT 'P'	
15C3	80	.BYT \$80	
15C4			
15C4	CF	INVIO RST ERR AUS	; FEHLERMELDUNG:
15C5	12	.BYT \$12	; 'INVALID I/O'
15C6			
15C6			; STREAM DATEN

```

15C6      ;
15C6 01   STDATA .BYT $01,$00      ; $FD ERGIBT KANAL 1
15C7 00
15C8 06   .BYT $06,$00      ; $FE " " S
15C9 00
15CA 08   .BYT $08,$00      ; $FF " " R
15CB 00
15CC 01   .BYT $01,$00      ; $00 " " V
15CD 00
15CE 01   .BYT $01,$00      ; $01 " " V
15CF 00
15D0 06   .BYT $06,$00      ; $02 " " S
15D1 00
15D2 10   .BYT $10,$00      ; $03 " " P
15D3 00
15D4      ;
15D4      ; DIESE SUBROUTINE ÜBERWACHT DEN AUFRUF DER AKTUELLEN
15D4      ; INPUT-SUBROUTINE
15D4      ;
15D4 FD CB 02 6E WARTA BIT 5,(IY+2) ; UNTEREN BILDSCHIRMTEIL
15D8 20 04      JR NZ,$15DE      ; NICHT LÖSCHEN
15DA FD CB 02 DE      SET 3,(IY+2) ; SONST MODUSWECHSEL ANMERKEN
15DE CD E6 15      CALL $15E6      ; INPUTROUTINE AUFRUFEN
15E1 D8      RET C      ; ALLES IN ORDNUNG
15E2 28 FA      JR Z,$15DE      ; KEINE TASTE: ZERO UND CARRY
15E4      ; GELÖSCHT
15E4 CF      RST ERRAUS      ; SONST ERORMELDUNG:
15E5 07      .BYT $07      ; 'END OF FILE'
15E6      ;
15E6      ; INPUTROUTINE FÜR DEN GERADE AKTUELLEN FILE
15E6      ;
15E6 D9      EXX      ; REGISTER RETTEN
15E7 E5      PUSH HL
15E8 2A 51 5C      LD HL,(CURCHL) ; ANFANGSADRESSE DER AKTUELLEN
15EB 23      INC HL      ; KANALINFORMATION HOLEN UND 2
15EC 23      INC HL      ; ADDIEREN, UM DIE INPUTADRESSE
15ED 18 08      JR INDCAL      ; HOLEN ZU KÖNNEN
15EF      ;
15EF      ; ALLGEMEINE AUSGABEROUTINE
15EF      ; AUSGA2 MIT AUSZUGEBENDEM ZEICHEN IN REG A
15EF      ;
15EF 1E 30      AUSGA1 LD E,$30      ; $30 ZU A ADDIEREN
15F1 83      ADD E
15F2 D9      AUSGA2 EXX      ; REGISTER RETTEN
15F3 E5      PUSH HL
15F4 2A 51 5C      LD HL,(CURCHL) ; ANFANGSADRESSE DER KANALINFO
15F7      ;
15F7      ; DIE AKTUELLE AUSGABE- ODER EINGABEROUTINE AUFRUFEN
15F7      ; HL ZEIGT AUF DIE ADRESSE, AN DER DIE SPRUNGADRESSE
15F7      ; ZU FINDEN IST
15F7      ;
15F7 5E      INDCAL LD E,(HL)      ; DIE RICHTIGE
15F8 23      INC HL      ; ADRESSE IN DE LADEN
15F9 56      LD D,(HL)      ; UND FÜR INDIREKTEN
15FA EB      EX DE,HL      ; SPRUNG IN HL BRINGEN
15FB CD 2C 16      CALL INDJMP
15FE E1      POP HL      ; REGISTER ZURÜCKHOLEN
15FF D9      EXX

```

```

1600 C9          RET
1601          ;
1601          ; ROUTINE, UM EINEN KANAL ZU ERÖFFNEN
1601          ; A ENTHÄLT GÜLTIGE 'STREAM'-NUMMER UND DER
1601          ; ENTSPRECHENDE KANAL WIRD ERÖFFNET
1601          ;
1601 87          OPKAN ADD A          ; MIT 2 MULTIPLIZIEREN
1602 C6 16      ADD $16          ; $16 ALS BASIS ADDIEREN
1604 6F        LD L,A          ; UND ALS LOW-ADRESSE IN L
1605 26 5C      LD H,$5C        ; H MIT $5C LADEN (=$5C1044)
1607 5E        LD E,(HL)        ; DIE ZWEI BYTES FÜR DEN
1608 23        INC HL          ; AKTUELLEN 'STREAM' HOLEN
1609 56        LD D,(HL)
160A 7A        LD A,D
160B B3        OR E            ; UND AUF NULL PRÜFEN
160C 20 02     JR NZ,$1610
160E          ;
160E CF        ISTREA RST ERRAUS  ; FALLS NULL:
160F 17        .BYT $17        ; 'INVALID STREAM'
1610          ;
1610 1B        DEC DE          ; STREAMDATA -1
1611 2A 4F 5C   LD HL,(CHANS)    ; POINTER FÜR KANALDATEN
1614 19        ADD HL,DE        ; DIE BENÖTIGTE BASISADRESSE
1615 22 51 5C   LD (CURCHL),HL  ; IN HL BILDEN UND SPEICHERN
161B FD CB 30 A6 RES 4,(IY+4B)  ; NICHT KANAL 'K'
161C 23        INC HL          ; HL +4, UM DEN KANALCODE
161D 23        INC HL
161E 23        INC HL
161F 23        INC HL
1620 4E        LD C,(HL)        ; HOLEN ZU KENNEN
1621 21 2D 16   LD HL,KLOOK     ; TABELLENADRESSE LADEN
1624 CD DC 16   CALL SUCHTA     ; UND IN DER TABELLE SUCHEN
1627 D0        RET NC          ; NICHT GEFUNDEN: RETURN
1628 16 00      LD D,0          ; SONST OFFSET IN DE BRINGEN
162A 5E        LD E,(HL)        ; UND ZU HL ALS
162B 19        ADD HL,DE        ; INDIREKTE SPRUNGADRESSE ADDIEREN
162C E9        INDJMP JP (HL)
162D          ;
162D 4B        KLOOK .BYT 'K',$06 ; K UND OFFSET 6 = $1634
162E 06
162F 53        .BYT 'S',$12     ; S UND OFFSET 18 = $1642
1630 12
1631 50        .BYT 'P',$1B     ; P UND OFFSET 27 = $164D
1632 1B
1633 00        .BYT 0
1634          ;
1634 FD CB 02 C6 KANALK SET 0,(IY+2) ; UNTERER BILDSCHIRMTEIL
163B FD CB 01 AE RES 5,(IY+1)      ; FERTIG FÜR EINE TASTE
163C FD CB 30 E6 SET 4,(IY+4B)    ; KANAL K WIRD BENUTZT
1640 1B 04      JR KANS1
1642          ;
1642 FD CB 02 86 KANALS RES 0,(IY+2) ; GANZER BILDSCHIRM
1646 FD CB 01 8E KANS1 RES 1,(IY+1) ; PRINTER NICHT IN BENUTZUNG
164A C3 4D 0D    JP $D4D
164D          ;
164D FD CB 01 CE KANALP SET 1,(IY+1) ; PRINTER WIRD BENUTZT
1651 C9          RET
1652          ;

```

1652		; DIESE ROUTINE SCHAFFT EINEN BENÖTIGTEN SPEICHERRAUM.
1652		; DIE BYTEZAHL MUSS IN BC ÜBERGEBEN WERDEN UND
1652		; HL ZEIGT DIREKT HINTER DIE SPEICHERPOSITION,
1652		; AN DER PLATZ BENÖTIGT WIRD
1652		;
1652	01 01 00	NUREIN LD BC,1 ; EINSPRUNG FÜR EIN BYTE
1655	E5	MACHPL PUSH HL ; RETTE DEN POINTER
1656	CD 05 1F	CALL \$1F05 ; TEST, OB GENÜGENDE SPEICHER DA
1659	E1	POP HL ; POINTER ZURÜCK
165A	CD 64 16	CALL \$1664 ; BETROFFENE POINTER KORRIGIEREN
165D	2A 65 5C	LD HL,(STKEND) ; NEUES STKEND IN HL
1660	EB	EX DE,HL ; AUSTAUSCH POINTER ALT UND NEU
1661	ED 8B	LDDR ; UND PLATZ SCHAFFEN
1663	C9	RET ; HL ZEIGT AUF 'ANFANG-1' UND
1664		; DE AUF DEN LETZTEN FREIEN PLATZ
1664		;
1664		; DIESE SUBROUTINE VERANDERT ALLE SYSTEMVARIABLEN
1664		; (POINTER), DIE AUF POSITIONEN HINTER DEM SPEICHERPLATZ
1664		; ZEIGEN (=HL), AN DEM PLATZ ETC. GESCHAFFEN WERDEN
1664		; SOLL. BC MUSS DIE ANZAHL DER BYTES ENTHALTEN
1664		;
1664	F5	POINTE PUSH AF ; REGISTER UND
1665	E5	PUSH HL ; STARTPOSITION RETTEN
1666	21 4B 5C	LD HL,VAR5 ; ADRESSE DER ERSTEN
1669	3E 0E	LD A,14 ; DER 14 SYSTEMVARIABLE N LADEN
166B		;
166B	5E	POINTL LD E,(HL) ; ADRESSIERTEN
166C	23	INC HL ; POINTER IN REG DE LADEN
166D	56	LD D,(HL)
166E	E3	EX (SP),HL ; ZEIGER AB WO UND ZEIGER AUF DIE
166F		; ZU PRÜFENDE VARIABLE TAUSCHEN
166F	A7	AND A ; CARRY LÖSCHEN
1670	ED 52	SBC HL,DE ; CARRY WIRD GESETZT, FALLS DIE
1672	19	ADD HL,DE ; SYSTEMVARIABLE GEÄNDERT
1673		; WERDEN MUSS
1673	E3	EX (SP),HL ; ZEIGERTAUSCH
1674	30 09	JR NC,NOCHA ; NICHT ÄNDERN
1676	D5	PUSH DE ; ALTEN WERT RETTEN
1677	EB	EX DE,HL
1678	09	ADD HL,BC ; BC ZUM ALTEN WERT ADDIEREN
1679	EB	EX DE,HL
167A	72	LD (HL),D ; UND DEN NEUEN WERT
167B	2B	DEC HL ; ABSPEICHERN
167C	73	LD (HL),E
167D	23	INC HL
167E	D1	POP DE ; ALTER WERT ZURÜCK
167F	23	NOCHA INC HL
1680	3D	DEC A ; ALLE VARIABLEN BEARBEITET
1681	20 EB	JR NZ,POINTL ; NEIN WEITER
1683		;
1683	EB	EX DE,HL ; ALTER WERT VON STKEND
1684	D1	POP DE ; REGISTER ZURÜCKHOLEN
1685	F1	POP AF
1686	A7	AND A ; DIFFERENCE ZWISCHEN STKEND
1687	ED 52	SBC HL,DE ; 'ALT' UND DEM 'PLATZ' BERECHNEN
1689	44	LD B,H ; NACH BC BRINGEN UND
168A	4D	LD C,L ; 1 ADDIEREN
168B	03	INC BC


```

168C 19          ADD HL,DE          ; STKEND ALT WIEDER HERSTELLEN
168D EB          EX DE,HL          ; UND NACH DE BRINGEN
168E C9          RET
168F             ;
168F             ; HOLEN DER NUMMER DER ZEILE, DIE DURCH HL ADRESSIERT
168F             ; WIRD. FALLS UNGÜLTIG, TEST OB DE AUF EINE GÜLTIGE
168F             ; NUMMER ZEIGT. TRIFFT DIES AUCH NICHT ZU, SO WIRD
168F             ; 0000 ALS ZEILENNUMMER IN DE GELADEN (NUMMER <10000).
168F             ; NORMALERWEISE ENTHALT DE DIE ZEILENNUMMER UND HL
168F             ; DIE STARTADRESSE DIESER ZEILE BEI RETURN
168F             ;
168F 00          ZEINUL .BYT $00,$00 ; ZEILENNUMMER NULL
1690 00
1691 EB          ZDAVOR EX DE,HL     ; POINTERTAUSCH: EINE ZEILE DAVOR
1692 11 8F 16    LD DE,ZEINUL      ; VERSUCHEN
1695             ; HIER EINSTIEG
1695 7E          ZSUCHE LD A,(HL)    ; HIGHBYTE DER NUMMER
1696 E6 C0       AND $C0            ; TESTEN
1698 20 F7       JR NZ,ZDAVOR       ; BEI >10000
169A 56          LD D,(HL)          ; SONST DE LADEN MIT
169B 23          INC HL              ; DER NUMMER
169C 5E          LD E,(HL)
169D C9          RET
169E             ;
169E             ; ROUTINE WIRD NORMALERWEISE ÜBER RESTART 30,
169E             ; (BC) PLATZE BESORGEN, AUFGERUFEN
169E             ; STACK ENTHALT DAHER ALS LETZTEN WERT 'WORKSP'
169E             ; UND DAVOR DIE ANZAHL 'BC'
169E             ;
169E 2A 63 5C    RESERV LD HL,(STKBOT) ; AKTUELLER WERT VON STKBOT
16A1 2B          DEC HL              ; -1 UM LETZTEN PLATZ VON
16A2             ; WORKSPACE ZU ERHALTEN
16A2 CD 55 16    CALL MACHPL        ; PLATZ BESCHAFFEN
16A5 23          INC HL              ; AUF DEN ZWEITEN NEUEN
16A6 23          INC HL              ; PLATZ ZEIGEN
16A7 C1          POP BC              ; WORKSP ALT WIEDER HERSTELLEN
16A8 ED 43 61 5C LD (WORKSP),BC
16AC C1          POP BC              ; ANZAHL ZURÜCKHOLEN
16AD EB          EX DE,HL            ; DE AUF ANFANG+2 UND HL AUF
16AE 23          INC HL              ; PLATZ NACH DEM FREIRAUM
16AF C9          RET                ; ZEIGEN LASSEN
16B0             ;
16B0             ; ROUTINE LÖSCHT DEN EDITORBEREICH, DEN WORKSPACE
16B0             ; UND DEN CALCULATOR STACK
16B0             ;
16B0 2A 59 5C    CLREDI LD HL,(ELINE) ; EDITORBEREICH NUR MIT
16B3 36 0D       LD (HL),$D          ; CARRIAGE RETURN ; UND
16B5 22 5B 5C    LD (KCUR),HL
16B8 23          INC HL
16B9 36 80       LD (HL),$80         ; ENDESYMBOL BESCHREIBEN
16BB 23          INC HL
16BC 22 61 5C    LD (WORKSP),HL
16BF 2A 61 5C    CLRWOR LD HL,(WORKSP) ; WORKSPACE LÖSCHEN
16C2 22 63 5C    LD (STKBOT),HL
16C5 2A 63 5C    CLRCAL LD HL,(STKBOT) ; CALCULATOR-STACK
16C8 22 65 5C    LD (STKEND),HL     ; LÖSCHEN
16CB E5          PUSH HL
16CC 21 92 5C    LD HL,MEMBOT       ; MEM AUF CALCULATOR-BEREICH

```

```

16CF 22 68 5C      LD (MEM),HL      ; SETZEN
16D2 E1            POP HL          ; STKEND WIEDER IN HL
16D3 C9            RET
16D4              ;
16D4              ; DIE EDITORZEILE WIEDER ENTFERNEN
16D4              ;
16D4 ED 58 59 5C   LD DE,(ELINE)   ; ADRESSE DES EINGEGEBEN
16D8 C3 E5 19      JP RAUS1        ; KOMMANDOS UND SPRUNG
16DB              ; ZUM SPEICHERENTFERNEN
16DB              ;
16DB              ; SUBROUTINE ZUM DURCHSUCHEN VON TABELLEN (ENDE = $00)
16DB              ; HL ZEIGT AUF DEREN ANFANG UND C ENTHALT DAS ZU
16DB              ; SUCHENDE ZEICHEN. CARRY GESETZT = GEFUNDEN
16DB              ;
16DB 23            SUCHT1 INC HL      ; +1 FÜR NÄCHSTES TABELLENPAAR
16DC 7E            SUCHTA LD A,(HL)   ; EINSTIEG IN DIE ROUTINE
16DD A7            AND A            ; WERT 0 = ENDE DER TABELLE
16DE C8            RET Z
16DF B9            CP C            ; VERGLEICH
16E0 23            INC HL          ; AUF NÄCHSTEN EINTRAG ZEIGEN
16E1 20 F8         JR NZ,SUCHT1     ; WAR NOCH NICHT RICHTIG
16E3 37            SCF             ; SONST CARRY SETZEN FÜR GEFUNDEN
16E4 C9            RET
16E5              ;
16E5              ; CLOSE#-SUBROUTINE ZUM SCHLIESSEN VON STREAMS
16E5              ; FÜR STREAMS $00 - $03 WERDEN DIE GRUNDDATEN IMMER
16E5              ; GESETZT, SO DASS DIESE NICHT GESCHLOSSEN WERDEN KÖNNEN
16E5              ;
16E5 CD 1E 17      CALL STRDAT      ; DATEN DES STREAM HOLEN
16E8 CD 01 17      CALL CLOKSP     ; CODE DES KANALS PRÜFEN
16E8 01 00 00      LD BC,0         ; DEFAULT FÜR STREAMDATEN = NULL
16EE 11 E2 A3      LD DE,$A3E2    ; AUF > STREAM 3 PRÜFEN
16F1 EB            EX DE,HL        ; ($A3E2+$5C16+2*STREAMNR.)
16F2 19            ADD HL,DE
16F3 38 07         JR C,$16FC      ; NICHT 0 BIS 3
16F5 01 D4 15      LD BC,$15D4    ; SONST GRUNDDATENADRESSE
16F8 09            ADD HL,BC       ; DES STREAMS BERECHNEN UND
16F9 4E            LD C,(HL)       ; DIESE HOLEN
16FA 23            INC HL
16FB 46            LD B,(HL)
16FC EB            EX DE,HL
16FD 71            LD (HL),C       ; NULLEN ODER DIE GRUNDDATEN
16FE 23            INC HL          ; DES STREAMS SPEICHERN
16FF 70            LD (HL),B
1700 C9            RET
1701              ;
1701              ; STREAMCODE K, S ODER P PRÜFEN UND STREAM SCHLIESSEN
1701              ;
1701              ;
1701 E5            CLOKSP PUSH HL    ; ADRESSE DER STREAMDATEN RETTEN
1702 2A 4F 5C      LD HL,(CHANS)   ; ADRESSE DER KANALINFORMATIONEN-
1705 09            ADD HL,BC       ; PLATZE, KANALDATEN DES ZU
1706 23            INC HL          ; SCHLIESSENDEN FILES SUCHEN
1707 23            INC HL          ; DIE SUBROUTINEN-ADRESSE
1708 23            INC HL          ; ÜBERGEHEN UND
1709 4E            LD C,(HL)       ; DEN KANAL-CODE HOLEN
170A EB            EX DE,HL        ; RETTE HL
170B 21 16 17      LD HL,CSTRTA    ; BASISADRESSE DER
170E CD DC 16      CALL SUCHTA     ; CLOSESTREAM-TABELLE UND SUCHEN

```

```

1711 4E          LD C,(HL)      ; OFFSET IN BC BRINGEN
1712 06 00       LD B,0
1714 09          ADD HL,BC      ; OFFSET ZU HL FÜR INDIREKTEN
1715 E9          JP (HL)       ; SPRUNG ADDIEREN
1716            ;
1716            ; TABELLE FÜR CLOSE STREAM
1716            ;
1716 4B          CSTRTA .BYT 'K', $05 ; K OFFSET $05 = $171C
1717 05          .BYT 'S', $03 ; S OFFSET $03 = $171C
1718 53          .BYT 'P', $01 ; P OFFSET $01 = $171C
1719 03
171A 50
171B 01
171C            ;
171C            ; CLOSE STREAM
171C            ;
171C E1          CLOSTR POP HL    ; ADRESSE DER STREAMDATA HOLEN
171D C9          RET
171E            ;
171E            ; NACH BC DIE DATEN EINES STREAM HOLEN
171E            ;
171E CD 94 1E    STRDAT CALL $1E94 ; STREAMNR VOM CALC-STACK HOLEN
1721 FE 10       CP $10          ; > 16 ERGIBT ERROR
1723 38 02       JR C,$1727
1725            ;
1725 CF          RST ERRASUS      ; FEHLERMELDUNG:
1726 17          .BYT $17        ; 'INVALID STREAM'
1727            ;
1727 C6 03       ADD 3            ; 3 ADDIEREN UND
1729 07          RLCA            ; MAL 2
172A 21 10 5C    LD HL,STRMS    ; BASISADRESSE DER STREAMDATEN
172D 4F          LD C,A          ; BENÖTIGTE STREAMADRESSE
172E 06 00       LD B,0
1730 09          ADD HL,BC      ; BERECHNEN UND
1731 4E          LD C,(HL)      ; DIE DATENBYTES HOLEN
1732 23          INC HL
1733 46          LD B,(HL)
1734 2B          DEC HL          ; HL WIEDER AUF DATENBYTES SETZEN
1735 C9          RET
1736            ;
1736            ; OPENW-SUBROUTINE
1736            ; KANALCODE MUSS K, S ODER P SEIN
1736            ;
1736 EF          STROPE RST CALRUF ; CALCULATOR BENUTZEN
1737 01          .BYT $01
1738 38          .BYT $38
1739 CD 1E 17     CALL STRDAT    ; DATEN DES STREAM HOLEN
173C 7B          LD A,B          ; WENN BEIDE BYTES NULL SIND,
173D B1          OR C            ; WAR ES EIN GESCHLOSSENER FILE
173E 2B 16       JR Z,STROP1
1740 EB          EX DE,HL        ; HL RETTEN
1741 2A 4F 5C     LD HL,(CHANS)  ; BASISADRESSE DER KANALINFO
1744 09          ADD HL,BC      ; UND ADRESSE
1745 23          INC HL          ; DES ZU ÖFFNENDEN KANALS
1746 23          INC HL          ; BERECHNEN, UM DESSEN
1747 23          INC HL
1748 7E          LD A,(HL)      ; CODE ZU HOLEN
1749 EB          EX DE,HL        ; HL ZURÜCK

```

174A FE 4B	CP 'K'	; KANALCODE AUF 1, 5 ODER 6
174C 2B 0B	JR Z, STROP1	; PRÜFEN
174E FE 53	CP 'S'	
1750 2B 04	JR Z, STROP1	
1752 FE 50	CP 'P'	
1754 20 CF	JR NZ, \$1725	; FALLS NICHTS DAVON: ERROR
1756 CD 5D 17	STROP1 CALL \$175D	; STREAMDATEN NACH DE BRINGEN
1759 73	LD (HL), E	; UND IN STREAMSPEICHER ABLEGEN
175A 23	INC HL	
175B 72	LD (HL), D	
175C C9	RET	
175D		
175D E5	PUSH HL	; HL RETTEN
175E CD F1 2B	CALL \$2BF1	; PARAMETER DES KANALCODES HOLEN
1761 7B	LD A, B	
1762 B1	OR C	; STREAMDATEN OK?
1763 20 02	JR NZ, \$1767	; JA
1765		
1765 CF	RST ERR AUS	; INVALID FILENAME
1766 0E	.BYT \$0E	
1767		
1767 C5	PUSH BC	; LANGE DES AUSDRUCKS RETTEN
1768 1A	LD A, (DE)	; ERSTES ZEICHEN HOLEN
1769 E6 DF	AND \$DF	; UND IN GROSSBUCHSTABE WANDELN
176B 4F	LD C, A	; NACH C FÜR TABSUCHE BRINGEN
176C 21 7A 17	LD HL, OPTAB	
176F CD DC 16	CALL SUCHTA	; IN TABELLE SUCHEN
1772 30 F1	JR NC, \$1765	; NICHT GEFUNDEN: ERROR
1774 4E	LD C, (HL)	; OFFSET ZUM HL-REG ADDIEREN
1775 06 00	LD B, 0	; UM INDIREKTEN SPRUNG
1777 09	ADD HL, BC	; AUSFÜHREN ZU KÖNNEN
1778 C1	POP BC	; LANGE ZURÜCK IN BC
1779 E9	JP (HL)	; UND ZUR ROUTINE
177A		
177A	; TABELLE FÜR STREAMERÖFFNUNG	
177A		
177A 4B	OPTAB .BYT 'K', \$06	; K OFFSET 06 = \$17B1
177B 06		
177C 53	.BYT 'S', \$0B	; K OFFSET 0B = \$17B5
177D 0B		
177E 50	.BYT 'P', \$0A	; K OFFSET 0A = \$17B9
177F 0A		
1780 00	.BYT \$00	
1781		
1781 1E 01	OPENK LD E, 1	; DATEN = \$01 UND \$00
1783 1B 06	JR \$178B	
1785		
1785 1E 06	OPENS LD E, 6	; DATEN = \$06 UND \$00
1787 1B 02	JR \$178B	
1789		
1789 1E 10	OPENP LD E, \$10	; DATEN = \$10 UND \$00
178B 0B	DEC BC	; LANGE -1
178C 7B	LD A, B	; ERROR WENN LANGE
178D B1	OR C	; NICHT 1 WAR
178E 20 D5	JR NZ, \$1765	
1790 57	LD D, A	; D LÖSCHEN UND
1791 E1	POP HL	; HL ZURÜCKHOLEN
1792 C9	RET	

```

1793 ;
1793 ; CAT-, ERASE-, FORMAT- UND MOVEBEFEHLE ERGEBEN
1793 ; DIE FEHLERMELDUNG 'INVALID STREAM'
1793 ;
1793 18 90 JR #1725
1795 ;
1795 ; LIST- UND LLISTROUTINE
1795 ; AUSGABE DER ZEILENNUMMER, TOKENUMWANDLUNG,
1795 ; CURSORDARSTELLUNG USW.
1795 ;
1795 ED 73 3F 5C LISTAU LD (LISTSP),SP ; STACKPOINTER SPEICHERN
1799 FD 36 02 10 LD (IY+2),#10 ; AUTOMATISCHES LISTING IM
179D ; HAUPTBILDSCHIRM ANMERKEN
179D CD AF 0D CALL $DAF ; DIESEN TEIL LÖSCHEN
17A0 FD CB 02 C6 SET 0,(IY+2) ; FLAGS: EDITORFLACHE EINSCHALTEN
17A4 FD 46 31 LD B,(IY+#31) ; DFSZ LADEN UND UNTEREN
17A7 CD 44 0E CALL $E44 ; BILDSCHIRMTEIL LÖSCHEN
17AA FD CB 02 86 RES 0,(IY+2) ; EDITOR WIEDER AUS
17AE FD CB 30 C6 SET 0,(IY+#30) ; FLAGS2: BILDSCHIRM GELÖSCHT
17B2 2A 49 5C LD HL,(EPPC) ; AKTUELLE ZEILENNUMMER UND
17B5 ED 5B 6C 5C LD DE,(STOP) ; OBERSTE NUMMER HOLEN
17B9 A7 AND A ; FALLS AKTUELLE NUMMER KLEINER
17BA ED 52 SBC HL,DE ; ALS OBERSTE IST, DIESE
17BC 19 ADD HL,DE ; NEU SETZEN
17BD 38 22 JR C,#17E1
17BF ;
17BF D5 PUSH DE ; OBERSTE NUMMER RETTEN
17C0 CD 6E 19 CALL $196E ; ADRESSE DER AKTUELLEN ZEILE
17C3 11 C0 02 LD DE,#2C0 ; BERECHNEN
17C6 EB EX DE,HL
17C7 ED 52 SBC HL,DE
17C9 E3 EX (SP),HL ; ERGEBNIS AUF STACK
17CA CD 6E 19 CALL $196E
17CD C1 POP BC ; ERGEBNIS IN BC
17CE C5 PUSH BC
17CF CD 8B 19 CALL $198B ; ADRESSE NÄCHSTE ZEILE IN DE
17D2 C1 POP BC ; ERGEBNIS WIEDER IN BC
17D3 09 ADD HL,BC ; STARTADRESSE DER NÄCHSTEN ZEILE
17D4 38 0E JR C,#17E4 ; FERTIG UND ZUM AUSLISTEN
17D6 EB EX DE,HL
17D7 56 LD D,(HL) ; ZEILENNUMMER DER NÄCHSTEN
17D8 23 INCHL ; ZEILE NACH DE BRINGEN
17D9 5E LD E,(HL)
17DA 2B DEC HL
17DB ED 53 6C 5C LD (STOP),DE ; OBERSTE (S-TOP) ZEILE NEU
17DF 18 ED JR #17CE ; UND WEITERSUCHEN
17E1 ;
17E1 22 6C 5C LD (STOP),HL
17E4 2A 6C 5C LD HL,(STOP) ; OBERSTE ZEILENNUMMER IN HL
17E7 CD 6E 19 CALL $196E ; UND DIE ZEILE SUCHEN
17EA 2B 01 JR Z,#17ED ; SPRUNG, WENN DIESE GEFUNDEN
17EC EB EX DE,HL ; SONST ADRESSE AUS DE BENUTZEN
17ED CD 33 18 CALL $1833 ; FÜR AUSGABE DES LISTINGS
17F0 FD CB 02 A6 RES 4,(IY+2) ; TVFLAG: AUTOMATISCHES LISTING
17F4 C9 RET ; WIEDER ABSCHALTEN
17F5 .END
17F5 .LIB SPEC1800-S
17F5 ; SINCLAIR ZX SPECTRUM TEIL 1800

```



```

17F5 ;
17F5 ; EINSTIEG BEI 'LLIST'
17F5 ;
17F5 3E 03 LD A,3 ; KANAL 3 FÜR DRUCKER OFFNEN
17F7 1B 02 JR $17FB
17F9 ;
17F9 ; EINSTIEG BEI 'LIST'
17F9 ;
17F9 3E 02 LD A,2 ; KANAL 2 FÜR HAUPTBILDSCHIRM
17FB FD 36 02 00 LD (IY+2),0 ; TVFLAG: NORMALES LISTING
17FF CD 30 25 CALL $2530 ; KANAL OFFNEN, FALLS KEINE
1802 C4 01 16 CALL NZ,$1601 ; SYNTAXPRÜFUNG
1805 DF RST GETAKT ; AKTUELLES ZEICHEN LADEN UND
1806 CD 70 20 CALL $2070 ; NACHSEHEN, OB EIN ANDERER
1809 ; KANAL ZU OFFNEN IST
1809 3B 14 JR C,$181F ; NEIN
180B DF RST GETAKT ; AKTUELLES ZEICHEN HOLEN
180C FE 3B CP ;
180E 2B 04 JR Z,$1814 ; WENN SEMIKOLON
1810 FE 2C CP ;
1812 20 06 JR NZ,$181A ; WENN NICHT KOMMA
1814 E7 RST GETNXT ; NÄCHSTES ZEICHEN HOLEN, PRÜFEN
1815 CD 82 1C CALL $1C82 ; OB NUMERISCHER AUSDRUCK
181B 1B 0B JR $1822
181A CD E6 1C CALL $1CE6 ; OHNE SEMIKOLON ODER KOMMA,
181D 1B 03 JR $1822 ; NULL BENUTZEN
181F ;
181F CD DE 1C CALL $1CDE ; ZEILENNUMMER BZW. NULL HOLEN
1822 CD EE 1B CALL $1BEE ; FALLS SYNTAXPRÜFUNG, NÄCHSTER
1825 ; BEFEHL
1825 CD 99 1E CALL $1E99 ; ZEILENNUMMER NACH BC
182B 7B LD A,B ; HIGH-BYTE IN DEN ERLAUBTEN
1829 E6 3F AND $3F ; BEREICH BRINGEN UND DIE
182B 67 LD H,A ; GANZE ZEILENNUMMER NACH
182C 69 LD L,C ; HL KOPIEREN
182D 22 49 5C LD (EPPC),HL ; DIESE ABSPEICHERN
1830 CD 6E 19 CALL $196E ; STARTADRESSE DER ZEILE SUCHEN
1833 ; FALLS DIESE NICHT EXISTIERT,
1833 ; DIE ADRESSE DER NÄCHSTEN ZEILE
1833 1E 01 LD E,1 ; FLAG 'VOR DER AKTUELLEN ZEILE'
1835 ;
1835 ; SCHLEIFE ZUM LISTEN MEHRERER ZEILEN
1835 ;
1835 CD 55 1B CALL $1855 ; EINE BASICZEILE LISTEN
183B D7 RST PRTOUT ; EIN CARRIAGE RETURN AUSGEBEN
1839 FD CB 02 66 BIT 4,(IY+2) ; AUTOMATISCHES LISTING ?
183D 2B F6 JR Z,$1835 ; NEIN
183F 3A 6B 5C LD A,(DFSZ) ; IM HAUPTBILDSCHIRM NOCH WAS
1842 FD 96 4F SUB (IY+$4F) ; FREI ?
1845 20 EE JR NZ,$1835 ; JA
1847 AB XOR E ; RETURN, FALLS SCHIRM VOLL UND
184B C8 RET Z ; DIE AKT. ZEILE GELISTET WURDE
1849 ;
1849 E5 PUSH HL ; FALLS NICHT,
184A D5 PUSH DE
184B 21 6C 5C LD HL,STOP ; STOP AUF NEUEM WERT SETZEN,
184E CD 0F 19 CALL $190F ; DIE NÄCHSTE ZEILE MIT
1851 D1 POP DE ; SCROLLING AUSGEBEN UND

```

1852	E1	POP HL	; WEITER
1853	18 E0	JR \$1835	; IN DER LISTSCHLEIFE
1855			
1855			; AUSGABE EINER KOMPLETTEN BASICZEILE
1855	ED 48 49 5C	LD BC, (EPPC)	; AKTUELLE ZEILENNUMMER HOLEN
1859	CD 80 19	CALL \$1980	; UND VERGLEICHEN, OB AKTUELLE,
185C			; EINE DAVOR ODER DANACH
185C	16 3E	LD D, \$3E	; DEFAULT CURSOR FÜR AKT. ZEILE
185E	28 05	JR Z, \$1865	; SPRUNG BEI AKTUELLE ZEILE
1860	11 00 00	LD DE, 0	; DE AUF 01 FÜR ZEILE 'DAVOR' UND
1863	CB 13	RL E	; = 0 FÜR ZEILE 'DANACH'
1865	FD 73 2D	LD (IY+\$2D), E	; ZEILENMERKER SPEICHERN
1868	7E	LD A, (HL)	; HIGHBYTE DER ZEILENNUMMER
1869	FE 40	CP \$40	; LADEN UND RETURN, WENN LISTING
186B	C1	POP BC	
186C	D0	RET NC	; FERTIG
186D			
186D	C5	PUSH BC	
186E	CD 28 1A	CALL \$1A2B	; ZEILENNUMMER MIT FÜHRENDEN
1871			; SPACES AUSGEBEN
1871	23	INC HL	
1872	23	INC HL	; HL AUF ERSTES ZEICHEN IN
1873	23	INC HL	; DER ZEILE SETZEN
1874	FD CB 01 86	RES 0, (IY+1)	; FLAGS: FÜHRENDE SPACES ERLAUBT
1878	7A	LD A, D	; CURSORCODE HOLEN
1879	A7	AND A	; UND SPRUNG, WENN
187A	28 05	JR Z, \$1881	; CURSOR NICHT GEDRUCKT WIRD
187C			
187C	D7	RST PRTOU	; SONST CURSOR AUSGEBEN
187D	FD CB 01 C6	SET 0, (IY+1)	; FÜHRENDE SPACES UNTERDRÜCKEN
1881	D5	PUSH DE	; RETTEN
1882	EB	EX DE, HL	; ZEILENZEIGER NACH DE
1883	FD CB 30 96	RES 2, (IY+48)	; KEIN GANSEFUSSMODUS ""
1887	21 3B 5C	LD HL, FLAGS	
188A	CB 96	RES 2, (HL)	; AUSGABE IN K-MODUS
188C	FD CB 37 6E	BIT 5, (IY+55)	; INPUTMODUS ?
1890	28 02	JR Z, \$1894	; NEIN
1892	CB D6	SET 2, (HL)	; JA: L-MODUS SETZEN
1894			
1894	2A 5F 5C	LD HL, (XPTR)	; SYNTAXERRORZEIGER LADEN
1897	A7	AND A	; CARRY LÖSCHEN
1898	ED 52	SBC HL, DE	; UND SOLANGE SPRINGEN, BIS
189A	20 05	JR NZ, \$18A1	; ERROR ANGEZEIGT WERDEN SOLL
189C	3E 3F	LD A, '2'	; EIN BLINKENDES '2'
189E	CD C1 18	CALL \$18C1	; AN DER ERRORSTELLE DRUCKEN
18A1	CD E1 18	CALL \$18E1	; UNTERSUCHEN, OB CURSOR DRUCKEN
18A4	EB	EX DE, HL	; HL WIEDER RESTAURIEREN
18A5	7E	LD A, (HL)	; EIN ZEICHEN DER ZEILE LADEN
18A6	CD B6 18	CALL \$18B6	; AUF ZAHLENMERKER UNTERSUCHEN
18A9	23	INC HL	; ZEIGER +1
18AA	FE 0D	CP \$D	; WENN ZEICHEN EIN CARRIAGE
18AC	28 06	JR Z, \$18B4	; RETURN WAR, DANN ENDE
18AE	EB	EX DE, HL	; HL WIEDER NACH DE
18AF	CD 37 19	CALL \$1937	; ZEICHEN AUSGEBEN
18B2	18 E0	JR \$1894	; UND WEITER IN DER ZEILE
18B4	D1	POP DE	; ENDE DER ZEILE BEI CR
18B5	C9	RET	

```

8B6      ;
8B6      ; WENN ZEICHEN EIN ZAHLENMERKER IST, DANN DIE FLOATING-
8B6      ; POINTZAHL UBERGEHEN
8B6      ;
8B6      CP $E      ; RETURN, WENN KEIN ZAHLENMERKER
8B8      C0      RET NZ
8B9      ;
8B9      23      INC HL      ; SONST HL +6 ZUM
8BA      23      INC HL      ; UBERSPRINGEN DES MERKERS
8BB      23      INC HL      ; UND DER 5 BYTE LANGEN
8BC      23      INC HL      ; FLOATINGPOINTZAHL
8BD      23      INC HL
8BE      23      INC HL
8BF      7E      LD A,(HL)    ; NACHSTES ZEICHEN LADEN
8C0      C9      RET
8C1      ;
8C1      ; AUSGABE EINES BLINKENDEN ZEICHENS
8C1      ;
8C1      D9      EXX      ; REGISTER UND
8C2      2A 8F 5C LD HL,(ATTRT) ; ATTRT + MASKT FETTER
8C5      E5      PUSH HL
8C6      CB 8C      RES 7,H      ; BLINKEN
8C8      CB FD      SET 7,L      ; EINSCHALTEN
8CA      22 8F 5C LD (ATTRT),HL ; IN ATTRT + MASKT EINSCHREIBEN
8CD      21 91 5C LD HL,PFLAG   ; PFLAG AUCH
8DD      56      LD D,(HL)      ;
8DE      D5      PUSH DE      ; ZWISCHENSPEICHERN
8DF      36 00      LD (HL),0    ; INVERSE + OVER = 0,
8E0      ;        ; PAPER ODER INK 109
8E1      ;        ; ZEICHEN AUSGEBEN
8E1      CD F4 09 CALL $9F4      ; PFLAG UND
8E2      E1      POP HL      ;
8E3      FD 74 57 LD (1Y+$57),H ; ATTRT + MASKT
8E4      E1      POP HL      ; WIEDER HERSTELLEN
8E5      22 8F 5C LD (ATTRT),HL ; REGISTER ZURUCK
8E6      D9      EXX
8E7      C9      RET
8E8      ;
8E8      ; AUSGABE DES CURSORS
8E8      ; FALLS AKTUELLE AUSGABEPOSITION NICHT CURSORPOSITION
8E8      ; IST, SOFORT RETURN, SONST DEN ENTSPRECHENDEN CURSOR
8E8      ; (C/E/G/K/L) DRUCKEN
8E8      ;
8E8      2A 5B 5C LD HL,(KCUR)   ; CURSORADRESSE LADEN
8E9      A7      AND A      ; CARRY LÖSCHEN
8EA      ED 52      SBC HL,DE    ; POSITION RICHTIG
8EB      C0      RET NZ      ; NEIN
8EC      ;
8EC      LD A,(MODE) ; JA: MODUS UNTERSUCHEN
8ED      RLC A      ; HAL 2
8EE      JR Z,$18F3 ; WENN NICHT 'G' ODER 'E'
8EF      ADD $43    ; SONST $43 ALS OFFSET ADDIEREN
8F0      JR $1909
8F1      ;
8F1      LD HL,FLAGS
8F2      RES 3,(HL) ; K-MODUS ANMERKEN
8F3      LD A,'K'   ; CURSOR K LADEN
8F4      BIT 2,(HL) ; UND AUSGEBEN,
8F5      JR Z,$1909 ; WENN K-MODUS
8F6      ;
8F6      LD HL,FLAGS
8F7      RES 3,(HL) ; K-MODUS ANMERKEN
8F8      LD A,'K'   ; CURSOR K LADEN
8F9      BIT 2,(HL) ; UND AUSGEBEN,
8FA      JR Z,$1909 ; WENN K-MODUS
8FB      ;
8FB      LD HL,FLAGS
8FC      RES 3,(HL) ; K-MODUS ANMERKEN
8FD      LD A,'K'   ; CURSOR K LADEN
8FE      BIT 2,(HL) ; UND AUSGEBEN,
8FF      JR Z,$1909 ; WENN K-MODUS

```

```

18FE CB DE          SET 3,(HL)      ; SONST L-MODUS
1900 3C             INC A           ; REG A ENTHALT 'L'
1901 FD CB 30 5E    BIT 3,(IY+$30) ; SPRUNG, WENN
1905 28 02          JR Z,$1909     ; IN L-MODUS
1907 3E 43          LD A,'C'       ; SONST BLEIBT NUR NOCH C-MODUS
1909 05             PUSH DE        ; ZWISCHENSPEICHERN
190A CD C1 18       CALL $18C1     ; BLINKENDEN CURSOR AUSGEBEN
190D 01             POP DE
190E C9             RET

190F               ;
190F               ; BEI EINSTIEG ZEIGT HL AUF STOP ODER EPPC. AM ENDE
190F               ; ENTHALT DIE JEWELIGE VARIABLE DIE ZEILENNUMMER
190F               ;
190F 5E             LD E,(HL)       ; DIE DERZEITIGE ZEILEN-
1910 23             INC HL          ; NUMMER NACH DE LADEN
1911 56             LD D,(HL)
1912 E5             PUSH HL         ; POINTER RETTEN
1913 EB             EX DE,HL        ; NACH HL BRINGEN UND
1914 23             INC HL          ; 1 ADDIEREN
1915 CD 6E 19       CALL $196E     ; STARTADRESSE DER AKTUELLEN
1918               ; ODER NACHSTEN ZEILE SUCHEN
1918 CD 95 16       CALL $1695     ; DIE ZEILENNUMMER DIESER LADEN
191B E1             POP HL          ; POINTER WIEDER HOLEN
191C FD CB 37 6E    BIT 5,(IY+$37) ; INPUT-MODUS ?
1920 C0             RET NZ         ; JA, RETURN
1921               ;
1921 72             LD (HL),D       ; SYSTEMVARIABLE MIT
1922 2B             DEC HL          ; DER GEFUNDENEN
1923 73             LD (HL),E       ; ZEILENNUMMER BESCHREIBEN
1924 C9             RET
1925               ;
1925               ; ROUTINE GIBT ZEICHEN EINER BASICZEILE AUS. BEI ZAHLEN
1925               ; WERDEN FÜHRENDE SPACES UNTERDRÜCKT (A=$FF) ODER NICHT
1925               ;
1925 7B             LD A,E          ; NACH E KOPIEREN
1926 A7             AND A           ; TEST
1927 FB             RET M           ; RETURN BEI $FF
1928 18 0D          JR $1937       ; SONST AUSGABE
192A               ;
192A               ; WANDLUNG EINER ZEILENNUMMER IN HL IN EINE DEZIMALZAHL
192A               ; DAZU ENTHALT BC, JE NACH AUFRUF, DIE WERTE -1000, -100
192A               ; ODER -10 (DESHALB ADDITION!)
192A               ;
192A AF             XOR A           ; REG A = 0 ZUM ZAHLEN
192B 09             ADD HL,BC       ; ES WIRD PROBEWEISE BC ADDIERT
192C 3C             INC A           ; UND DER ZÄHLER INKREMENTIERT
192D 38 FC          JR C,$192B     ; NOCH KEIN UNDERFLOW
192F ED 42          SBC HL,BC       ; EINMAL ZUVIEL WIEDER ABZIEHEN
1931 3D             DEC A           ; UND ZÄHLER -1
1932 28 F1          JR Z,$1925     ; FALLS NICHTS, PRÜFEN AUF SPACE
1934 C3 EF 15       JP $15EF       ; SONST EINE DEZIMALZAHL DRUCKEN
1937               ;
1937               ; HIER ALLE ZEICHEN, CONTROLCODES UND TOKENS AUSGEBEN
1937               ;
1937 CD 1B 2D       CALL $2D1B     ; AUF DEZIMALZAHL PRÜFEN
193A 30 30          JR NC,$196C    ; FALLS JA, DIREKT AUSGEBEN
193C FE 21          CP '!'         ; ALLE STEUERZEICHEN UND SPACE
193E 38 2C          JR C,$196C    ; AUSGEBEN

```

1940	FD CB 01 96	RES 2, (IY+1)	; ANMERKEN: AUSGABE Y-MODUS
1944	FE CB	CP #CB	
1946	2B 24	JR Z, \$196C	; SPRUNG BEI 'THEN'
1948	FE 3A	CP ''	
194A	20 0E	JR NZ, \$195A	; ALLES AUSSER '' SPRUNG
194C	FD CB 37 6E	BIT 5, (IY+\$37)	; DOPPELPUNKT IM INPUT-MODUS
1950	20 16	JR NZ, \$196B	; AUSGEBEN
1952	FD CB 30 56	BIT 2, (IY+\$30)	; SPRUNG, WENN DOPPELPUNKT
1956	2B 14	JR Z, \$196C	; NICHT IM TEXTMODUS ''
1958	1B 0E	JR \$196B	; SONST AUSGEBEN
195A	FE 22	CP ''	; ZEICHEN FÜR TEXTMODUS -
195C	20 0A	JR NZ, \$196B	; NEIN
195E	F5	PUSH AF	; ZEICHEN RETTEN
195F	3A 6A 5C	LD A, (FLAGS2)	; TEXTMODUSBIT INVERTIEREN
1962	EE 04	XOR 4	
1964	32 6A 5C	LD (FLAGS2), A	
1967	F1	POP AF	; '' ZURÜCKHOLEN
1968	FD CB 01 D6	SET 2, (IY+1)	; ANMERKEN: NACHSTES ZEICHEN
196C			; IM L-MODUS
196C	D7	RST PRTOU	; DAS ZEICHEN AUSGEBEN
196D	C9	RET	
196E			
196E		; SUBROUTINE ZUM SUCHEN DES ANFANGS EINER ZEILE NACH	
196E		; DEREN ZEILENNUMMER (IN HL). WIRD DIESE ZEILE NICHT	
196E		; GEFUNDEN, WIRD DIE STARTADRESSE DER NACHSTEN ZEILE	
196E		; ÜBERGEBEN, JEWEILS IN HL, UND IN DE DIE ADRESSE DER	
196E		; ZEILE DAVOR. FALLS ZEILE GEFUNDEN WURDE, IST DAS	
196E		; ZEROFLAG GESETZT	
196E			
196E	E5	PUSH HL	; ZWISCHENSPEICHERN
196F	2A 53 5C	LD HL, (PROG)	; SYSTEMVARIABLE
1972	54	LD D, H	; PROG NACH DE BRINGEN
1973	5D	LD E, L	
1974	C1	POP BC	; ZEILENNUMMER IN BC
1975	CD 80 19	CALL \$1980	; VERGLEICH DER VORGEgebenEN
1978			; ZEILENNUMMER MIT DER DERZEIT
1978			; ADRESSIERTEN
1978	D0	RET NC	; RETURN: ZEILE GEFUNDEN
1979	C5	PUSH BC	; SONST DIE NACHSTE
197A	CD 8B 19	CALL \$198B	; ZEILE SUCHEN
197D	EB	EX DE, HL	; POINTER TAUSCHEN UND
197E	1B F4	JR \$1974	; NOCHMAL SUCHEN
1980			
1980		; VERGLEICH EINER ZEILENNUMMER IN BC MIT EINER DURCH HL	
1980		; ADRESSIERTEN ZEILE	
1980			
1980	7E	LD A, (HL)	; HIGHBYTE DER ZEILENNUMMER
1981	8B	CP B	; MIT GESUCHTER VERGLEICHEN
1982	C0	RET NZ	; NICHT GLEICH
1983	23	INC HL	; SONST LOWBYTE
1984	7E	LD A, (HL)	; LADEN UND
1985	2B	DEC HL	
1986	89	CP C	; VERGLEICHEN: GLEICH UND
1987	C9	RET	; GRÖßER: CARRY GELÖSCHT
1988			
1988		; DIESE ROUTINE SUCHT DAS DURCH DAS REGISTER D	
1988		; BESTIMMTE ZEICHEN EINER ZEILE ODER DAS ZEICHEN,	
1988		; TOKEN ETC. WELCHES IN REG E ENTHALTEN IST	


```

1988      ;
1988 23      INC HL      ; POINTER +3
1989 23      INC HL
198A 23      INC HL
198B      ; NORMALER EINSTIEG
198B 22 5D 5C      LD (CHADD),HL      ; AUF DERZEITIGES BYTE SETZEN
198E 0E 00      LD C,0      ; TEXTMODUS AUS
1990      ; SUCHSCHLEIFE
1990 15      DEC D      ; D ALS ZÄHLER -1
1991 08      RET Z      ; ZEICHEN GEFUNDEN
1992      ;
1992 E7      RST GETNXT      ; NÄCHSTES ZEICHEN HOLEN
1993 8B      CP E      ; MIT GESUCHTEM VERGLEICHEN
1994 20 04      JR NZ,$199A      ; UNGLEICH
1996 A7      AND A      ; GEFUNDEN: CARRY- UND ZEROFLAG
1997 C9      RET      ; GELÖSCHT BEI RETURN
1998 23      INC HL
1999 7E      LD A,(HL)      ; NÄCHSTES ZEICHEN
199A CD B6 18      CALL $18B6      ; ÜBER ZAHLEN HINWEGLESEN
199D 22 5D 5C      LD (CHADD),HL      ; NEU SETZEN
19A0 FE 22      CP ""      ; GANSEFUSS FÜR TEXTMODUS ?
19A2 20 01      JR NZ,$19A5      ; NEIN
19A4 0D      DEC C      ; SONST ANMERKEN
19A5 FE 3A      CP ':'      ;
19A7 28 04      JR Z,$19AD      ; BEI DOPPELPUNKT WEITERSUCHEN
19A9 FE CB      CP $CB      ; IST TOKEN = 'THEN' ?
19AB 20 04      JR NZ,$19B1      ; NEIN
19AD CB 41      BIT 0,C      ; TEXTMODUS ?
19AF 28 DF      JR Z,$1990      ; NEIN
19B1 FE 0D      CP $D      ; CARRIAGE RETURN ?
19B3 20 E3      JR NZ,$1998      ; NEIN
19B5 15      DEC D      ; ZÄHLER -1
19B6 37      SCF      ; CARRYFLAG FÜR RETURN SETZEN
19B7 C9      RET
19B8      ;
19B8      ; SUBROUTINE ZUM SUCHEN DER NÄCHSTEN BASICZEILE ODER DER
19B8      ; NÄCHSTEN VARIABLEN
19B8      ;
19B8 E5      PUSH HL      ; RETTE POINTER
19B9 7E      LD A,(HL)      ; ERSTES ZEICHEN HOLEN
19BA FE 40      CP $40
19BC 3B 17      JR C,$19D5      ; UND BEI ZEILENSUCHE: SPRUNG
19BE CB 6F      BIT 5,A      ; BEI STRING- ODER ARRAY-
19C0 28 14      JR Z,$19D6      ; VARIABLEN: SPRUNG
19C2 87      ADD A      ; BIT 6 TESTEN: EINFACHE UND
19C3 FA C7 19      JP M,$19C7      ; FOR-NEXT VARIABLE: SPRUNG
19C6 3F      CCF      ; NUR VARIABLE MIT LANGEN NAMEN
19C7 01 05 00      LD BC,5      ; NUMERISCHE VARIABLE BENÖTIGEN
19CA 30 02      JR NC,$19CE      ; 5 UND FOR-NEXT
19CC 0E 12      LD C,$12      ; VARIABLE 18 SPEICHERPLATZE
19CE 17      RLA      ; CARRY IST NUR BEI LANGEN NAMEN
19CF 23      INC HL      ; GELÖSCHT
19D0 7E      LD A,(HL)
19D1 30 FB      JR NC,$19CE      ; SUCHE BIS DIESE ZU ENDE SIND
19D3 18 06      JR $19DB      ; IMMER SPRUNG BEI VARIABLEN
19D5      ;
19D5 23      INC HL      ; ZEILENSUCHE: LOWZEILENNUMMER
19D6 23      INC HL      ; UND AUF LOWLANGE ZEIGEN

```

```

1907 4E          LD C,(HL)          ; DIE LANGE IN BC
1908 23          INC HL
1909 46          LD B,(HL)          ; LADEN
190A 23          INC HL              ; HL ZEIGT
190B 09          ADD HL,BC          ; AUF NACHSTE ZEILE/VARIABLE
190C D1          POP DE
190D
190D             ;
190D             ; SUBROUTINE ZUM BERECHNEN EINER DIFFERENZ IN BC.
190D             ; HL UND DE SIND BEI RETURN VERTAUSCHT
190D             ;
190D A7          DIFFER AND A        ; CARRY LÖSCHEN
190E ED 52       SBC HL,DE          ; DIFFERENZ BILDEN UND
190E 44          LD B,H             ; NACH BC BRINGEN
190E 4D          LD C,L
190E 19          ADD HL,DE          ; SUBTRAKTION RÜCKGANGIG MACHEN
190E EB          EX DE,HL          ; POINTER TAUSCHEN
190E C9          RET
190E
190E             ;
190E             ; DIESE ROUTINE ENTFERNT SPEICHERBEREICHE UND KORRIGIERT
190E             ; ALLE ENTSPRECHENDEN POINTER.
190E             ; AM ERSTEN EINSTIEG ENTHÄLT DE DIE ERSTE ZU ENTFERNENDE
190E             ; UND HL DIE ERSTE, NICHT MEHR ZU ENTFERNENDE, SPEICHER-
190E             ; STELLE. BEIM ZWEITEN EINSTIEGSPUNKT ENTHÄLT HL DIE
190E             ; ERSTE ZU ENTFERNENDE SPEICHERSTELLE UND BC DIE ANZAHL
190E             ;
190E CD DD 19     RAUS1 CALL DIFFER    ; ANZAHL AUSRECHNEN
190E C5          RAUS2 PUSH BC        ; ANZAHL ZWISCHENSPEICHERN
190E 78          LD A,B             ; KOMPLEMENT VON BC BILDEN,
190E 2F          CPL                ; WEIL ALLE POINTER 'OBERHALB'
190E 47          LD B,A             ; DES ZU LÖSCHENDEN SPEICHER-
190E 79          LD A,C             ; BEREICHS KORRIGIERT WERDEN
190E 2F          CPL                ; MÜSSEN
190E 4F          LD C,A
190E 03          INC BC
190E CD 64 16     CALL $1664         ; POINTER NEU SETZEN
190E EB          EX DE,HL          ; 'ERSTE' SPEICHERSTELLE WIEDER
190E E1          POP HL            ; IN DE UND DIFFERENZ IN HL
190E 19          ADD HL,DE          ; ADDIERT ERGIBT ERSTE, NICHT ZU
190E             ; ZU ENTFERNENDE SPEICHERSTELLE
190E D5          PUSH DE           ; ADRESSE ZWISCHENSPEICHERN
190E ED B0       LDIR              ; UNSPEICHERN
190E E1          POP HL            ; ERSTE STELLE IN HL
190E C9          RET
190E
190E             ;
190E             ; EINLESEN EINER ZEILENNUMMER BEIM EDITIEREN.
190E             ; BEI EINEM DIREKTKOMMANDO WIRD DIESE ZU NULL GESETZT.
190E             ; DAS ERGEBNIS STEHT IMMER IN BC
190E             ;
190E 2A 59 5C     LD HL,(ELINE)      ; POINTER AUF EDITORZEILE
190E 2B          DEC HL             ; CHADD AUF POSITION VOR
190E 22 5D 5C     LD (CHADD),HL      ; DER ZEILENNUMMER SETZEN
190E E7          RST GETNXT         ; NÄCHSTES ZEICHEN HOLEN
190E 21 92 5C     LD HL,MEMB0T      ; CALCULATOR-SPEICHER ALS
190E 22 65 5C     LD (STKEND),HL    ; CALCULATORSTACK SETZEN
190E CD 3B 2D     CALL $2D3B        ; ZEILENNUMMER ODER 0 EINLESEN
190E CD A2 2D     CALL $2DA2        ; NUMMER IN BC BRINGEN (HEY)
190E 3B 04       JR C,$1A15        ; FALLS NUMMER >65535: ERROR
190E 21 F0 D8     LD HL,$DBF0      ; SONST TEST AUF 10000

```

```

1A14 09          ADD HL,BC
1A15 DA BA 1C    JP C,$1C8A      ; ZEILENNUMMER >10000: ERROR
1A18 C3 C5 16    JP $16C5        ; CALCULATORSTACK NORMAL
1A1B             ;
1A1B             ; ZEILENNUMMERAUSGABE
1A1B             ; INHALT BC WIRD IN DEZIMAL GEWANDELT UND AUSGEGEBEN
1A1B             ;
1A1B D5          PUSH DE         ; REGISTER
1A1C E5          PUSH HL         ; RETTEN
1A1D AF          XOR A           ; REG A = 0
1A1E CB 78       BIT 7,B
1A20 20 20       JR NZ,$1A42
1A22 60          LD H,B          ; DEZIMALZAHL FÜR DIE
1A23 69          LD L,C          ; AUSGABE NACH HL
1A24 1E FF       LD E,$FF       ; KEINE FÜHRENDEN SPACES
1A26 18 0B       JR $1A30       ; UND AUSGEBEN
1A28             ;
1A28             ; DIE DURCH HL ADRESSIERTE (HEXADEZIMALE) ZEILENNUMMER
1A28             ; IN DEZIMAL WANDELN UND AUSGEBEN (MIT FÜHRENDEN SPACES)
1A28             ;
1A28 D5          PUSH DE         ; RETTEN
1A29 56          LD D,(HL)       ; ZEILENNUMMER NACH DE
1A2A 23          INC HL
1A2B 5E          LD E,(HL)
1A2C E5          PUSH HL         ; POINTER HIERAUF RETTEN
1A2D EB          EX DE,HL        ; NUMMER IN HL
1A2E 1E 20       LD E,' '       ; FÜHRENDE SPACES ZULASSEN
1A30 01 18 FC    LD BC,$FC18    ; -1000 LADEN
1A33 CD 2A 19    CALL $192A      ; ERSTES DIGIT AUSGEBEN
1A36 01 9C FF    LD BC,$FF9C    ; -100 LADEN
1A39 CD 2A 19    CALL $192A      ; NÄCHSTES DIGIT
1A3C 0E F6       LD C,$F6       ; -10 LADEN
1A3E CD 2A 19    CALL $192A      ; ZEHNERDIGIT
1A41 7D          LD A,L          ; EINERSTELLE ALS
1A42 CD EF 15    CALL $15EF      ; DEZIMALZAHL AUSGEBEN
1A45 E1          POP HL         ; REGISTER ZURÜCKHOLEN
1A46 D1          POP DE
1A47 C9          RET
1A48             ;
1A48             ; =====
1A48             ;
1A48             ; BASICBEFEHLSINTERPRETATION
1A48             ;
1A48             ; OFFSETTABELLE DER BASICBEFEHLE FÜR DIE ZWEITE TABELLE
1A48             ;
1A48 B1          BEOFF .BYT $B1   ; DEF FN
1A49 CB          .BYT $CB        ; CAT
1A4A BC          .BYT $BC        ; FORMAT
1A4B BF          .BYT $BF        ; MOVE
1A4C C4          .BYT $C4        ; ERASE
1A4D AF          .BYT $AF        ; OPEN#
1A4E B4          .BYT $B4        ; CLOSE#
1A4F 93          .BYT $93        ; MERGE
1A50 91          .BYT $91        ; VERIFY
1A51 92          .BYT $92        ; BEEP
1A52 95          .BYT $95        ; CIRCLE
1A53 98          .BYT $98        ; INK
1A54 98          .BYT $98        ; PAPER

```

1A55	98	.BYT \$98	; FLASH
1A56	98	.BYT \$98	; BRIGHT
1A57	98	.BYT \$98	; INVERSE
1A58	98	.BYT \$98	; OVER
1A59	98	.BYT \$98	; OUT
1A5A	7F	.BYT \$7F	; LPRINT
1A5B	81	.BYT \$81	; LLIST
1A5C	2E	.BYT \$2E	; STOP
1A5D	6C	.BYT \$6C	; READ
1A5E	6E	.BYT \$6E	; DATA
1A5F	70	.BYT \$70	; RESTORE
1A60	48	.BYT \$48	; NEW
1A61	94	.BYT \$94	; BORDER
1A62	56	.BYT \$56	; CONTINUE
1A63	3F	.BYT \$3F	; DIM
1A64	41	.BYT \$41	; REM
1A65	2B	.BYT \$2B	; FOR
1A66	17	.BYT \$17	; GO TO
1A67	1F	.BYT \$1F	; GO SUB
1A68	37	.BYT \$37	; INPUT
1A69	77	.BYT \$77	; LOAD
1A6A	44	.BYT \$44	; LIST
1A6B	0F	.BYT \$0F	; LET
1A6C	59	.BYT \$59	; PAUSE
1A6D	2B	.BYT \$2B	; NEXT
1A6E	43	.BYT \$43	; POKE
1A6F	2D	.BYT \$2D	; PRINT
1A70	51	.BYT \$51	; PLOT
1A71	3A	.BYT \$3A	; RUN
1A72	6D	.BYT \$6D	; SAVE
1A73	42	.BYT \$42	; RANDOMIZE
1A74	0D	.BYT \$0D	; IF
1A75	49	.BYT \$49	; CLS
1A76	5C	.BYT \$5C	; DRAW
1A77	44	.BYT \$44	; CLEAR
1A78	15	.BYT \$15	; RETURN
1A79	5D	.BYT \$5D	; COPY
1A7A			
1A7A			; PARAMETERTABELLE FÜR DIE BEFEHLE
1A7A			; BYTES IM BEREICH VON \$00 BIS \$0B GEBEN DIE WEITEREN
1A7A			; NOTWENDIGEN PARAMETER FÜR DIE EINZELNEN BEFEHLE AN
1A7A			
1A7A			; PAR00: KEINE WEITEREN PARAMETER
1A7A			; PAR01: BEI LET, EINE VARIABLE WIRD GEBRAUCHT
1A7A			; PAR02: EIN NUMERISCHER/STRING-AUSDRUCK MUSS FOLGEN
1A7A			; PAR03: NUMERISCHER AUSDRUCK KANN FOLGEN, SONST 0
1A7A			; PAR04: EINE EINFACHE VARIABLE MUSS FOLGEN
1A7A			; PAR05: EIN PARAMETERSATZ KANN FOLGEN
1A7A			; PAR06: EIN NUMERISCHER AUSDRUCK MUSS FOLGEN
1A7A			; PAR07: FARBENBEHANDLUNG ETC.
1A7A			; PAR08: ZWEI DURCH ',' GETRENNTE, NUMERISCHE AUSDRÜCKE
1A7A			; PAR09: WIE 08, ABER FARBENAUSDRÜCKE DÜRFEN VORANGEHEN
1A7A			; PAR0A: EIN STRING-AUSDRUCK MUSS FOLGEN
1A7A			; PAR0B: BEARBEITET KASSETTENROUTINEN
1A7A			
1A7A	01	PALET .BYT \$01	
1A7B	3D	.BYT \$3D	
1A7C	02	.BYT \$02	

1A7D 06	PAGOTO .BYT \$06	
1A7E 00	.BYT \$00	
1A7F 67 1E	.WOR \$1E67	
1A81 06	PAIF .BYT \$06	
1A82 CB	.BYT \$CB	; THEN
1A83 05	.BYT \$05	
1A84 F0 1C	.WOR \$1CF0	
1A86 06	PAGOSU .BYT \$06	
1A87 00	.BYT \$00	
1A88 ED 1E	.WOR \$1EED	
1A8A 00	PASTOP .BYT \$00	
1A8B EE 1C	.WOR \$1CEE	
1A8D 00	PARETU .BYT \$00	
1A8E 23 1F	.WOR \$1F23	
1A90 04	PAFOR .BYT \$04	
1A91 3D	.BYT '='	
1A92 06	.BYT \$06	
1A93 CC	.BYT \$CC	; TO
1A94 06	.BYT \$06	
1A95 05	.BYT \$05	
1A96 03 1D	.WOR \$1D03	
1A98 04	PANEXT .BYT \$04	
1A99 00	.BYT \$00	
1A9A AB 1D	.WOR \$1DAB	
1A9C 05	PAPRIN .BYT \$05	
1A9D CD 1F	.WOR \$1FCD	
1A9F 05	PAINPU .BYT \$05	
1AA0 B9 20	.WOR \$20B9	
1AA2 05	PADIM .BYT \$05	
1AA3 02 2C	.WOR \$2C02	
1AA5 05	PAREM .BYT \$05	
1AA6 B2 1B	.WOR \$1BB2	
1AAB 00	PANEW .BYT \$00	
1AA9 B7 11	.WOR \$11B7	
1AAB 03	PARUN .BYT \$03	
1AAC A1 1E	.WOR \$1EA1	
1AAE 05	PALIST .BYT 05	
1AAF F9 17	.WOR \$17F9	
1AB1 08	PAPOKE .BYT \$08	
1AB2 00	.BYT \$00	
1AB3 B0 1E	.WOR \$1EB0	
1AB5 03	PARAND .BYT \$03	
1AB6 4F 1E	.WOR \$1E4F	
1AB8 00	PACONT .BYT \$00	
1AB9 5F 1E	.WOR \$1E5F	
1ABB 03	PACLEA .BYT \$03	
1ABC AC 1E	.WOR \$1EAC	
1ABE 00	PACLS .BYT \$00	
1ABF 6B 0D	.WOR \$0D6B	
1AC1 09	PAPLOT .BYT \$09	
1AC2 00	.BYT \$00	
1AC3 DC 22	.WOR \$22DC	
1AC5 06	PAPPAUS .BYT \$06	
1AC6 00	.BYT \$00	
1AC7 3A 1F	.WOR \$1F3A	
1AC9 05	PAREAD .BYT \$05	
1ACA ED 1D	.WOR \$1DED	
1ACC 05	PADATA .BYT \$05	

1ACD	27	1E		.WOR	\$1E27
1ACF	03		PAREST	.BYT	\$03
1AD0	42	1E		.WOR	\$1E42
1AD2	09		PADRAW	.BYT	\$09
1AD3	05			.BYT	05
1AD4	82	23		.WOR	\$2382
1AD6	00		PACOPY	.BYT	\$00
1AD7	AC	0E		.WOR	\$0EAC
1AD9	05		PALPRI	.BYT	\$05
1ADA	C9	1F		.WOR	\$1FC9
1ADC	05		PALLIS	.BYT	\$05
1ADD	F5	17		.WOR	\$17F5
1ADF	0B		PASAVE	.BYT	\$0B
1AE0	0B		PALOAD	.BYT	\$0B
1AE1	0B		PAVERI	.BYT	\$0B
1AE2	0B		PAMERG	.BYT	\$0B
1AE3	0B		PABEEP	.BYT	\$0B
1AE4	00			.BYT	\$00
1AE5	F8	03		.WOR	\$03F8
1AE7	09		PACIRC	.BYT	\$09
1AE8	05			.BYT	\$05
1AE9	20	23		.WOR	\$2320
1AEB	07		PAINK	.BYT	\$07
1AEC	07		PAPAPE	.BYT	\$07
1AED	07		PAFLAS	.BYT	\$07
1AEE	07		PABRIG	.BYT	\$07
1AEF	07		PAINVE	.BYT	\$07
1AF0	07		PAOVER	.BYT	\$07
1AF1	0B		PAOUT	.BYT	\$0B
1AF2	00			.BYT	\$00
1AF3	7A	1E		.WOR	\$1E7A
1AF5	06		PABORD	.BYT	\$06
1AF6	00			.BYT	\$00
1AF7	94	22		.WOR	\$2294
1AF9	05		PADEFN	.BYT	\$05
1AFA	60	1F		.WOR	\$1F60
1AFC	06		PAOPEN	.BYT	\$06
1AFD	2C			.BYT	,
1AFE	0A			.BYT	\$0A
1AFF	00			.BYT	\$00
1B00	36	17		.WOR	\$1736
1B02	06		PACLOS	.BYT	\$06
1B03	00			.BYT	\$00
1B04	E5	16		.WOR	\$16E5
1B06	0A		PAFORM	.BYT	\$0A
1B07	00			.BYT	\$00
1B08	93	17		.WOR	\$1793
1B0A	0A		PANOVE	.BYT	\$0A
1B0B	2C			.BYT	,
1B0C	0A			.BYT	\$0A
1B0D	00			.BYT	\$00
1B0E	93	17		.WOR	\$1793
1B10	0A		PAERAS	.BYT	\$0A
1B11	00			.BYT	\$00
1B12	93	17		.WOR	\$1793
1B14	00		PACAT	.BYT	\$00
1B15	93	17		.WOR	\$1793
1B17					

```

1B17 ; =====
1B17 ;
1B17 ; HAUPTROUTINE DES BASICINTERPRETERS MIT SYNTAXPRÜFUNG
1B17 ;
1B17 FD CB 01 BE RES 7, (IY+1) ; SYNTAXPRÜFUNG ANMERKEN
1B1B CD FB 19 CALL #19FB ; CHADD AUF ERSTES ZEICHEN NACH
1B1E ; DER ZEILENNUMMER SETZEN
1B1E AF XOR A ; POINTER IN DIE BASICZEILE
1B1F 32 47 5C LD (SUBPPC), A ; AUF NULL SETZEN
1B22 3D DEC A
1B23 32 3A 5C LD (ERRNR), A ; KEIN ERROR
1B26 1B 01 JR #1B29
1B28 E7 RST GETNXT ; DAS NÄCHSTE ZEICHEN EINLESEN
1B29 CD BF 16 CALL #16BF ; WORKSPACE LÖSCHEN
1B2C FD 34 0D INC (IY+$D) ; SUBPPC +1
1B2F FA BA 1C JP M, $1C8A ; NUR 127 ZEICHEN PRO ZEILE ; nowsens
1B32 DF RST GETAKT ; EIN ZEICHEN HOLEN
1B33 06 00 LD B, 0 ; B FÜR TABELLE MIT 0 LADEN
1B35 FE 0D CP $D ; ZEILENENDE ?
1B37 2B 7A JR Z, $1BB3 ; JA
1B39 FE 3A CP ' ' ; TRENNUNGSZEICHEN ?
1B3B 2B EB JR Z, $1B28 ; JA, OBEN WEITER
1B3D ;
1B3D 21 76 1B LD HL, BRKTST ; RETURNADRESSE FÜR
1B40 E5 PUSH HL ; BREAKTEST AUF STACK LEGEN
1B41 4F LD C, A ; ZEICHEN NACH C KOPIEREN
1B42 E7 RST GETNXT ; POINTER IN DER ZEILE +1
1B43 79 LD A, C ; VORHERIGES ZEICHEN LADEN
1B44 D6 CE SUB $CE ; $CE VON DEN TOKENS ABZIEHEN
1B46 ; BEREICH JETZT $00 - $31
1B46 DA BA 1C JP C, $1C8A ; WAR KEIN TOKEN: ERROR nowsens
1B49 4F LD C, A ; FÜR TABELLE NACH C BRINGEN
1B4A 21 4B 1A LD HL, BEFOFF ; ADRESSE DER BEFEHLSOFFSETTAB.
1B4D 09 ADD HL, BC ; TOKENWERT ADDIEREN, UM DEN
1B4E 4E LD C, (HL) ; OFFSET NACH C LADEN, DIES
1B4F 09 ADD HL, BC ; ERGIBT ZEIGER AUF DIE BEFEHLS-
1B50 1B 03 JR #1B55 ; PARAMETER AB $1A7A UND SPRUNG
1B52 ;
1B52 ; DIE PARAMETER DER BEFEHLE WERDEN AUSGEWERTET
1B52 ;
1B52 2A 74 5C PARHOL LD HL, (TADDR) ; ZEIGER AUF NÄCHSTEN PARAMETER
1B55 7E LD A, (HL) ; IN REG A LADEN
1B56 23 INC HL ; POINTER +1
1B57 22 74 5C LD (TADDR), HL ; FÜR NÄCHSTEN PARAMETER
1B5A 01 52 1B LD BC, $1B52 ; RETURNADRESSE FÜR WEITERE
1B5D C5 PUSH BC ; PARAMETER HOLEN UND AUF STACK
1B5E 4F LD C, A ; ZWISCHENSPEICHERN
1B5F FE 20 CP ' ' ; IST ES EIN SEPARATOR (, ETC.)
1B61 30 0C JR NC, $1B6F ; JA
1B63 21 01 1C LD HL, $1C01 ; ADRESSE DER PARAMETEROFFSET-
1B66 ; TABELLE
1B66 06 00 LD B, 0
1B68 09 ADD HL, BC ; DEN PARAMETER DAZUADDIEREN
1B69 4E LD C, (HL) ; OFFSET AUS TABELLE LADEN
1B6A 09 ADD HL, BC ; UND ADDIEREN FÜR
1B6B E5 PUSH HL ; RETURNAUFRUF
1B6C DF RST GETAKT ; LETZTEN BEFEHL NOCH MAL
1B6D 05 DEC B ; LADEN UND B AUF $FF SETZEN

```

```

186E C9          RET          ; HIER SPRUNG AN DIE BERECHNETE
186F              ; ADRESSE
186F              ;
186F              ; DER SEPARATOR GEMASS PARAMETERTABELLE, STEHT IN REG C,
186F              ; MUSS AN DIESER STELLE IN DER ZEILE ZU FINDEN SEIN
186F              ;
186F DF          RST GETAKT    ; ZEICHEN NOCH MAL LADEN
1870 B9          CP C          ; UND VERGLEICHEN
1871 C2 BA 1C    JP NZ,$1C8A   ; WENN UNGLEICH: ERROR
1874 E7          RST GETNXT    ; SONST POINTER AUF NACHSTES
1875 C9          RET          ; ZEICHEN SETZEN
1876              ;
1876              ; NACH JEDEM RICHTIGEN BEFEHL WIRD AN DIESE STELLE, ZUM
1876              ; PRÜFEN DER BREAKTASTE, ZURÜCKGEFÜHRT
1876              ;
1876 CD 54 1F    BRKST CALL $1F54 ; TEST, OB BREAK GEDRÜCKT
1879 38 02      JR C,$1B7D      ; NEIN
187B              ;
187B CF          RST ERR AUS    ; SONST MELDUNG 'BREAK' INTO
187C 14          .BYT $14       ; PROGRAMM 'AUSGEBEN'
187D              ;
187D FD CB 0A 7E ; BIT 7, (IY+10) ; IST EIN SPRUNG AUSZUFÜHREN ?
1881 20 71      JR NZ,$1BF4     ; NEIN
1883              ;
1883 2A 42 5C    LD HL,(NEWPPC) ; SONST NEUE ZEILENNUMMER LADEN
1886 CB 7C      BIT 7,H         ; SPRUNG IM PROGRAMM ?
1888 28 14      JR Z,$1B9E      ; JA, DIE ZEILE SUCHEN
188A              ; SONST IST ES DER BEFEHL RUN
188A              ;
188A              ; ROUTINE FÜR 'RUN'
188A              ; DAS SYNTAX/RUN-FLAG (7 VON FLAGS) IST DANN GESETZT
188A              ;
188A 21 FE FF    LD HL,$FFFE     ; STARTZEILE MIT -2 VORBESETZEN
188D 22 45 5C    LD (PPC),HL
1890 2A 61 5C    LD HL,(WORKSP) ; HL ZEIGT AUF ENDEMARKIERUNG
1893 2B          DEC HL          ; DES EDITORBEREICHS UND DE WIRD
1894 ED 58 59 5C ; LD DE,(ELINE) ; MIT DER ADRESSE VOR DIESEM
1898 1B          DEC DE          ; BEREICH GELADEN
1899 3A 44 5C    LD A,(NSPPC)   ; REG A ENTHÄLT DEN NÄCHSTEN
189C 1B 33      JR $1BD1        ; BEFEHL ZUR WEITERVERARBEITUNG
189E              ;
189E              ; NEUE ZEILE NACH EINEM SPRUNGBEFEHL SUCHEN
189E              ;
189E CD 6E 19    CALL $196E     ; ADRESSE DER ZEILE BESTIMMEN
18A1 3A 44 5C    LD A,(NSPPC)   ; BEFEHLSNUMMER DER ZEILE LADEN
18A4 28 19      JR Z,$1BBF      ; DIE ZEILE WURDE GEFUNDEN
18A6 A7          AND A          ; SONST MUSS BEFEHLSNUMMER 0 SEIN
18A7 20 43      JR NZ,$1BEC     ; WENN NICHT, ERROR
18A9              ;
18A9 47          LD B,A          ; REG A RETTEN
18AA 7E          LD A,(HL)       ; HIGH BYTE DER ZEILENNUMMER
18AB E6 C0      AND $C0         ; LADEN UND AUF GÜLTIGKEIT PRÜFEN
18AD 7B          LD A,B         ; REG A ZURÜCK
18AE 2B 0F      JR Z,$1BBF      ; WEDER NACH PROGRAMMIERUNG NOCH
18B0              ; FALSCHER ZEILENNUMMER
18B0              ;
18B0 CF          RST ERR AUS    ; MELDUNG:
18B1 FF          .BYT $FF       ; 'OK'

```

```

1BB2      ;
1BB2      ; EINSTIEG BEI DEM BEFEHL 'REM'
1BB2      ; DURCH ENTFERNEN DER RETURNADRESSE BRKTST WIRD DER REST
1BB2      ; DER ZEILE IGNORIERT
1BB2      ;
1BB2 C1      POP BC      ; BRKTST WEGWERFEN
1BB3      ;
1BB3      ; BEHANDLUNG DER ZEILE, WENN DAS ENDE GEFUNDEN WURDE
1BB3      ;
1BB3 CD 30 25      CALL $2530      ; FALLS SYNTAXPRÜFUNG, DIREKT
1BB6 CB      RET Z      ; RETURN
1BB7      ;
1BB7 2A 55 5C      LD HL,(NXTLIN) ; POINTER AUF NÄCHSTE ZEILE
1BB8 3E C0      LD A,$C0      ; LADEN UND UNTERSUCHEN, OB DAS
1BB8 A6      AND (HL)      ; PROGRAMM ZU ENDE IST
1BB8 C0      RET NZ      ; JA
1BBE      ;
1BBE AF      XOR A      ; NEIN: BEFEHL ZU 0 SETZEN
1BBF      ;
1BBF      ; DIESE ROUTINE HOLT DIE NEUE ZEILENNUMMER NACH 'PPC'
1BBF      ; UND SUCHT DEN BEGINN DER DARAUFFOLGENDEN ZEILE
1BBF      ;
1BBF FE 01      CP 1      ; HIERMIT WIRD BEFEHL 0 ZU
1BC1 CE 00      ADC 0      ; BEFEHL 1 GESETZT
1BC3 56      LD D,(HL)      ; ZEILENNUMMER IN
1BC4 23      INC HL      ; DE LADEN UND
1BC5 5E      LD E,(HL)
1BC6 ED 53 45 5C      LD (PPC),DE ; IN PPC SPEICHERN
1BCA 23      INC HL      ; LÄNGE DIESER ZEILE
1BCB 5E      LD E,(HL)      ; NACH DE LADEN
1BCC 23      INC HL
1BCD 56      LD D,(HL)
1BCE EB      EX DE,HL      ; LÄNGE UND POINTER AUSTAUSCHEN
1BCF 19      ADD HL,DE      ; HL ZEIGT AUF HIGH BYTE DER
1BD0 23      INC HL      ; ZEILENNUMMER (=ANFANG) DER
1BD1      ; NÄCHSTEN ZEILE, DE AUF 1.
1BD1      ; BEFEHL-1 DER VORHERIGEN ZEILE
1BD1      ;
1BD1      ; VARIABLE FÜR NÄCHSTE ZEILE SETZEN:
1BD1      ;
1BD1 22 55 5C      LD (NXTLIN),HL ; BEGINN DER NÄCHSTEN ZEILE
1BD4 EB      EX DE,HL
1BD5 22 5D 5C      LD (CHADD),HL ; CHADD ZEIGT AUF NÄCHSTEN
1BD8      ; BEFEHL-1
1BD8 57      LD D,A      ; BEFEHL KOPIEREN
1BD9 1E 00      LD E,0      ; E MIT 0 VORBESETZEN
1BDB FD 36 0A FF      LD (IY+$A),$FF ; NSPPC: KEIN JUMP ANMERKEN
1BDF 15      DEC D      ; BEFEHLSNUMMER-1 NACH
1BE0 FD 72 0D      LD (IY+$D),D ; SUBPPC BRINGEN
1BE3 CA 2B 1B      JP Z,$1B2B ; NEUE ZEILE: ERSTEN BEFEHL
1BE6      ; UNTERSUCHEN
1BE6 14      INC D      ; BEFEHLSZAHL DER ZEILE KOR-
1BE7      ; RIGIEREN UND DIE ADRESSE-1
1BE7 CD 8B 19      CALL $198B ; DAZU BESTIMMEN
1BEA 2B 0B      JR Z,$1BF4 ; BEFEHL GEFUNDEN
1BEC      ;
1BEC CF      RST ERR AUS ; MELDUNG:
1BED 16      .BYT $16 ; 'STATEMENT LOST'

```

```

1BEE ;
1BEE CD 30 25 CALL $2530 ; SYNTAXPRÜFUNG ?
1BF1 C0 RET NZ ; NEIN: RETURN (PROGRAMMLAUF)
1BF2 C1 POP BC ; JA: 2 * RETURNADRESSE
1BF3 C1 POP BC ; WEGWERFEN
1BF4 ;
1BF4 ; NACHSTEN BEFEHL FINDEN:
1BF4 ; BEI CARRIAGE RETURN IN DER NACHSTEN ZEILE, BEI
1BF4 ; ':' IN DER GLEICHEN ZEILE.
1BF4 ; ALLE ANDEREN ZEICHEN BEDEUTEN SYNTAX-ERROR
1BF4 ;
1BF4 DF RST GETAKT ; AKTUELLES ZEICHEN LADEN
1BF5 FE 0D CP $D ; ZEILENENDE ?
1BF7 2B BA JR Z,$1BB3 ; JA, NACHSTE ZEILE SUCHEN
1BF9 FE 3A CP ':' ; BEFEHLSTRENNUNG ?
1BFB CA 2B 1B JP Z,$1B2B ; JA: WEITER IN DER INTERPRETER-
1BFE ; SCHLEIFE ZUM NACHSTEN BEFEHL
1BFE C3 8A 1C JP $1C8A ; SONST ERROR 'NONSENS..'
1C01 .END
1C01 .LIB SPEC1C00-S
1C01 ; SINCLAIR ZX SPECTRUM TEIL 1C00
1C01 ;
1C01 ; DIE OFFSETTABELLE FÜR DIE EINZELNEN BEFEHLSPARAMETER
1C01 ;
1C01 0F .BYT $0F ; PAR00 $1C10
1C02 1D .BYT $1D ; PAR01 $1C1F
1C03 4B .BYT $4B ; PAR02 $1C4E
1C04 09 .BYT $09 ; PAR03 $1C0D
1C05 67 .BYT $67 ; PAR04 $1C6C
1C06 0B .BYT $0B ; PAR05 $1C11
1C07 7B .BYT $7B ; PAR06 $1C82
1C08 8E .BYT $8E ; PAR07 $1C96
1C09 71 .BYT $71 ; PAR08 $1C7A
1C0A B4 .BYT $B4 ; PAR09 $1CBE
1C0B 81 .BYT $81 ; PAR0A $1C8C
1C0C CF .BYT $CF ; PAR0B $1CDB
1C0D ;
1C0D ;
1C0D ; PARAMETER 03: ES KANN EINE ZAHL FOLGEN
1C0D ;
1C0D CD DE 1C CALL $1CDE ; EINE ZAHL EINLESEN, 0 FALLS
1C10 ; KEINE GEFUNDEN WIRD
1C10 ;
1C10 ; PARAMETER 00: ES DÜRFEN KEINE ANGABEN FOLGEN Z.B. COPY
1C10 ;
1C10 BF CP A ; ZEROFLAG SETZEN
1C11 ;
1C11 ; PARAMETER 05: ES KÖNNEN AUSDRÜCKE FOLGEN:
1C11 ; Z. B. PRINT"SPECTRUM"
1C11 ;
1C11 C1 POP BC ; RETURNADRESSE WEGWERFEN
1C12 CC EE 1B CALL Z,$1BEE ; NUR BEI PAR00 UND 01 SYNTAX-
1C13 ; PRÜFUNG + ZUM NACHSTEN BEFEHL
1C15 EB EX DE,HL ; ZEILENZEIGER NACH DE
1C16 ; JETZT KANN, NACH ERFOLGTER PRÜFUNG, AUS DER PARAMETER-
1C16 ; TABELLE DER EINZELNEN BEFEHLE DIE SPRUNGADRESSE
1C16 ; GELADEN UND AUF DEN STACK GESCHOBEN WERDEN
1C16 ;

```

1C16	2A 74 5C	LD HL,(TADDR)	; ZEIGER IN DIE TABELLE
1C19	4E	LD C,(HL)	; REGISTER BC MIT
1C1A	23	INC HL	
1C1B	46	LD B,(HL)	; DER SPRUNGADRESSE LADEN
1C1C	EB	EX DE,HL	; ZEILENZEIGER NACH HL ZURÜCK
1C1D	C5	PUSH BC	; SPRUNG IN DIE ROUTINE
1C1E	C9	RET	; DURCH RETURNBEFEHL
1C1F			
1C1F		; PARAMETER 01: VARIABLENZUWEISUNG BEI LET	
1C1F			
1C1F	CD B2 28	CALL \$28B2	; NACHSEHEN, OB DIE VARIABLE
1C22			; SCHON BENUTZT WORDEN IST
1C22	FD 36 37 00	LD (IY+\$37),0	; FLAGX INITIALISIEREN
1C26	30 0B	JR NC,\$1C30	; VARIABLE WURDE SCHON BENUTZT
1C28	FD CB 37 CE	SET 1,(IY+55)	; NEUE VARIABLE ANMERKEN
1C2C	20 18	JR NZ,\$1C46	; ALLES OK, AUSSER BEI EINEM
1C2E			; NICHT DIMENSIONIERTEN ARRAY
1C2E			
1C2E	CF	RST ERR AUS	; ERROR 'VARIABLE NOT FOUND'
1C2F	01	.BYT \$01	; AUSGEBEN
1C30			
1C30	CC 96 29	CALL Z,\$2996	; EINFACHE STRING- ODER ARRAY-
1C33			; PARAMETER AUF DEN CALCULATOR-
1C33			; STACK BRINGEN
1C33	FD CB 01 76	BIT 6,(IY+1)	; SPRUNG BEI NUMERISCHER
1C37	20 0D	JR NZ,\$1C46	; VARIABLEN
1C39	AF	XOR A	; REG A = 0
1C3A	CD 30 25	CALL \$2530	; SYNTAXPRÜFUNG ?
1C3D	C4 F1 2B	CALL NZ,\$2BF1	; NEIN: STRING- ODER
1C40			; ARRAYPARAMETER HOLEN
1C40	21 71 5C	LD HL,FLAGX	; BIT 0 BEI EINFACHEN STRINGS
1C43	B6	OR (HL)	; SETZEN UND DAMIT ANMERKEN, DIE
1C44	77	LD (HL),A	; ALTEN WERTE ZU ENTFERNEN
1C45	EB	EX DE,HL	; HL ZEIGT AUF STRING ODER EIN
1C46			; ELEMENT EINES ARRAYS
1C46			
1C46		; FÜR ALLE NUMERISCHEN UND NEUEN STRING- ODER	
1C46		; STRINGARRAYVARIABLEN ENTHÄLT C DEN VARIABLENNAMEN.	
1C46		; FÜR ALTE STRING- ODER STRINGARRAYVARIABLE ENTHÄLT BC	
1C46		; DIE LÄNGE FÜR DIE ZUWEISUNG	
1C46			
1C46	ED 43 72 5C	LD (STRLEN),BC	
1C4A	22 4D 5C	LD (DEST),HL	; FÜR ZUWEISUNG SETZEN
1C4D	C9	RET	
1C4E			
1C4E		; PARAMETER 02: AKTUELLE BERECHNUNG FÜR ZUWEISUNG EINER	
1C4E		; VARIABLEN IN EINEM LET-BEFEHL DURCHFÜHREN	
1C4E			
1C4E	C1	POP BC	; EINE RETURNADRESSE WEGWERFEN
1C4F	CD 56 1C	CALL \$1C56	; ZUWEISUNG DURCHFÜHREN
1C52	CD EE 1B	CALL \$1BEE	; ENDEPRÜFUNG, ZUR LAUFZEIT
1C55	C9	RET	; JEDOCH ZUM NÄCHSTEN BEFEHL
1C56			
1C56		; DIESE ROUTINE WIRD VON LET, READ UND INPUT BENUTZT UM	
1C56		; ERST EINE VARIABLE ZU BERECHNEN UND DIESE DANN	
1C56		; ZUZUWEISEN. INPUT BENUTZT FLAGX UND STEIGT BEIM	
1C56		; ZWEITEN BEFEHL EIN	
1C56			


```

1C56 3A 3B 5C      LD A,(FLAGS)
1C59 F5            PUSH AF          ; FLAGS ODER FLAGX RETTEN
1C5A CD FB 24      CALL $24FB      ; NACHSTEN AUSDRUCK BERECHNEN
1C5D F1            POP AF          ; FLAGS/FLAGX ZURÜCK
1C5E FD 56 01      LD D,(IY+1)    ; FLAGS NEU LADEN
1C61 AA            XOR D           ; DIE ART UND DER AUSDRUCK MUSSEN
1C62 E6 40          AND $40        ; ÜBEREINSTIMMEN
1C64 20 24          JR NZ,$1C8A    ; ERROR 'NONSENS...'
1C66 CB 7A          BIT 7,D        ; ZUWEISUNG DURCHFÜHREN, FALLE
1C68 C2 FF 2A      JP NZ,$2AFF    ; ZUR LAUFZEIT
1C6B C9            RET             ; BEI SYNTAXPRÜFUNG RETURN
1C6C
1C6C ;
1C6C ; PARAMETER 04: EINSTIEG FÜR FOR..NEXT-BEFEHLE
1C6C ;
1C6C CD B2 2B      CALL $2B82      ; DIE BENUTZTE VARIABLE SUCHE
1C6F F5            PUSH AF          ; STATUSREGISTER RETTEN
1C70 79            LD A,C          ; PRÜFEN, OB DIE VARIABLE
1C71 F6 9F          OR $9F         ; EINE FOR..NEXT-VARIABLE
1C73 3C            INC A           ; IST
1C74 20 14          JR NZ,$1C8A    ; NEIN: ERROR
1C76 F1            POP AF          ; SONST STATUS ZURÜCKLADEN
1C77 1B A9          JR $1C22       ; UND DIE ZUWEISUNG VORBEREITEN
1C79
1C79 ;
1C79 ; DIE FOLGENDEN ROUTINEN DIENEN DEM BERECHNEN VON
1C79 ; NUMERISCHEN AUSDRÜCKEN. DAS JEWEILIGE ERGEBNIS KOMMT
1C79 ; ALS LETZTES AUF DEN CALCULATORSTACK
1C79 ; DER ERSTE EINSTIEGSPUNKT DIENT DEM BERECHNEN VON ZWEI
1C79 ; DURCH KOMMA GETRENNTEN AUSDRÜCKEN (PAR 09)
1C79 ;
1C79 E7            RST GETNXT       ; CHADD+1, NACHSTES ZEICHEN HOLEN
1C7A CD 82 1C      CALL $1C82      ; ERSTEN AUSDRUCK AUSRECHNEN
1C7D FE 2C          CP ','         ; WENN TRENNUNG NICHT DURCH
1C7F 20 09          JR NZ,$1C8A    ; KOMMA, DANN ERROR
1C81 E7            RST GETNXT       ; NACHSTES ZEICHEN LADEN
1C82
1C82 ;
1C82 ; AB HIER NUR EINEN AUSDRUCK BERECHNEN
1C82 ;
1C82 CD FB 24      CALL $24FB      ; NACHSTEN AUSDRUCK BERECHNEN
1C85 FD CB 01 76   BIT 6,(IY+1)    ; RETURN, WENN DAS ERGEBNIS
1C89 C0            RET NZ         ; NUMERISCH, SONST:
1C8A
1C8A ;
1C8A CF            RST ERR AUS      ; MELDUNG:
1C8B 0B            .BYT $0B        ; 'NONSENS IN BASIC'
1C8C
1C8C ;
1C8C ; BERECHNUNG EINES EINFACHEN STRING-AUSDRUCKS
1C8C ;
1C8C CD FB 24      CALL $24FB      ; NACHSTEN AUSDRUCK BERECHNEN
1C8F FD CB 01 76   BIT 6,(IY+1)    ; BIT6 FLAGS =0 = STRING,
1C93 CB            RET Z          ;
1C94 1B F4          JR $1C8A       ; SONST ERROR
1C96
1C96 ;
1C96 ; PARAMETER 07: SETZEN DER DAUERHAFTEN FARBEN
1C96 ;
1C96 FD CB 01 7E   BIT 7,(IY+1)    ; TEST SYNTAX/RUN-FLAG
1C9A FD CB 02 86   RES 0,(IY+2)    ; HAUPTBILDSCHIRM ANMERKEN
1C9E C4 4D 0D      CALL NZ,$D4D    ; AUFRUF WÄHREND PROGRAMMLAUF
1CA1 F1            POP AF          ; RETURNADRESSE WEGWERFEN
1CA2 3A 74 5C      LD A,(TADDR)   ; TADDR-LOW LADEN UND IN DEN

```

1CA5	D6 13	SUB #13	; BEREICH \$D9 BIS \$DE FÜR DIE
1CA7			; TOKENS INK BIS OVER BRINGEN
1CA7	CD FC 21	CALL \$21FC	; TEMPORÄRE FARBEN ÄNDERN
1CAA	CD EE 1B	CALL \$1BEE	; ZUM NÄCHSTEN BEFEHL BEI
1CAD			; SYNTAXPRÜFUNG
1CAD	2A 8F 5C	LD HL, (ATTRT)	; SONST DIE TEMPORÄREN FARB-
1CB0	22 8D 5C	LD (ATTRP), HL	; WERTE ZU DEN DAUERHAFTEN
1CB3	21 91 5C	LD HL, PFLAG	; MACHEN
1CB6	7E	LD A, (HL)	; DIE GERADEN BITS IN DIE
1CB7	07	RLCA	; UNGERADEN KOPIEREN, SODASS DIE
1CB8	AE	XOR (HL)	; TEMPORÄREN EBENFALLS ZU DEN
1CB9	E6 AA	AND \$AA	; DAUERHAFTEN WERDEN
1CBB	AE	XOR (HL)	
1CBC	77	LD (HL), A	
1CBD	C9	RET	
1CBE			
1CBE			; PARAMETER 09: DIESE ROUTINE WIRD VON PLOT, DRAW UND
1CBE			; CIRCLE BENUTZT, UM ERST EINMAL DIE DEFAULTWERTE VON
1CBE			; FLASH, BRIGHT UND PAPER AUF 8 ZU SETZEN
1CBE			
1CBE	CD 30 25	CALL \$2530	; SPRUNG, WENN NUR
1CC1	2B 13	JR Z, \$1CD6	; SYNTAXPRÜFUNG
1CC3	FD CB 02 86	RES 0, (IY+2)	; TVFLAG: HAUPTBILDSCHIRM
1CC7	CD 4D 0D	CALL \$D4D	; TEMPORÄRE FARBEN DES HAUPT-
1CCA			; BILDSCHIRMS SETZEN
1CCA	21 90 5C	LD HL, MASKT	
1CCD	7E	LD A, (HL)	; VON MASKT NUR DIE 'INK'-WERTE
1CCE	F6 F8	OR \$F8	; ÜBERNEHMEN, FLASH, BRIGHT
1CD0	77	LD (HL), A	; UND PAPER SIND DAMIT 8
1CD1	FD CB 57 86	RES 6, (IY+\$57)	; PFLAG: NICHT PAPER 9 SETZEN
1CD5	DF	RST GETAKT	; NOCH MAL DAS LETZTE ZEICHEN
1CD6	CD E2 21	CALL \$21E2	; DIE LOKALEN, DOMINANTEN FARB-
1CD9			; WERTE BEARBEITEN
1CD9	1B 9F	JR \$1C7A	; DIE ERSTEN ZWEI OPERANDEN
1CDB			; VERARBEITEN
1CDB			
1CDB			; PARAMETER 0B: ALLE KASSETTENROUTINEN
1CDB			
1CDB	C3 05 06	JP \$605	
1CDE			
1CDE			; ROUTINE UM EINEN NUMERISCHEN AUSDRUCK ZU BERECHNEN.
1CDE			; ES WIRD DER WERT NULL ÜBERGEBEN, FALLS KEIN AUSDRUCK
1CDE			; VORHANDEN IST
1CDE			
1CDE	FE 0D	CP \$D	; ZEILENENDE ?
1CE0	2B 04	JR Z, \$1CE6	; JA
1CE2	FE 3A	CP ':'	; TRENNUNGSZEICHEN ?
1CE4	20 9C	JR NZ, \$1C82	; NEIN
1CE6			
1CE6			; SONST DEN CALCULATOR BENUTZEN UND EINE NULL IM
1CE6			; CALCULATORSTACK ADDIEREN
1CE6			
1CE6	CD 30 25	CALL \$2530	; FALLS SYNTAXPRÜFUNG, NICHTS
1CE9	C8	RET Z	; BERECHNEN UND RETURN
1CEA	EF	RST CALRUF	; SONST CALCULATORAUFRUF
1CEB	A0	.BYT \$A0	; LETZTER WERT = 0
1CEC	38	.BYT \$38	; ENDE, DAMIT WURDE DIE NULL
1CED	C9	RET	; ADDIERT

```

1CEE      ;
1CEE      ; DIE FOLGENDEN ROUTINEN DIENEN DER BEFEHLSAUSFÜHRUNG
1CEE      ;
1CEE      ; BEFEHL STOP
1CEE      ;
1CEE CF      RST ERR AUS      ; NUR AUSGABE VON
1CEF 08      .BYT $08        ; 'STOP STATEMENT'
1CF0      ;
1CF0      ; BEFEHL IF
1CF0      ;
1CF0 C1      POP BC          ; RETURNADRESSE $1876 WEGWERFEN
1CF1 CD 30 25 CALL $2530      ; SPRUNG BEI
1CF4 28 0A    JR Z,$1D00      ; SYNTAXPRÜFUNG
1CF6 EF      RST CALRUF      ; CALCULATORAUFRUF
1CF7 02      .BYT $02        ; LÖSCHE LETZTEN CALCULATOR-
1CF8 38      .BYT $38        ; STACKWERT; ENDE
1CF9 EB      EX DE,HL        ; HL ZEIGT AUF DEN LETZTEN WERT
1CFA CD E9 34 CALL $34E9      ; TEST AUF NULL
1CFD DA B3 1B JP C,$1BB3     ; WERT NACH IF NICHT WAHR: DIE
1D00      ; ZEILE IST FERTIG, ZUR NÄCHSTEN
1D00 C3 29 1B JP $1B29       ; SONST ZUM NÄCHSTEN BEFEHL
1D03      ; NACH THEN
1D03      ;
1D03      ; BEFEHL FOR
1D03      ;
1D03 FE CD    CP $CD          ; MIT STEP (SCHRITTWEITE) ?
1D05 20 09    JR NZ,$1D10     ; NEIN
1D07 E7      RST GETNXT      ; CHADD+1, NÄCHSTES ZEICHEN
1D08 CD 82 1C CALL $1C82      ; WERT FÜR STEP HOLEN
1D08 CD EE 1B CALL $1BEE      ; ZUM NÄCHSTEN BEFEHL GEE
1D0E      ; BEI SYNTAXPRÜFUNG
1D0E 18 06    JR $1D16        ; ANDERNFALLS WEITER
1D10      ;
1D10 CD EE 1B CALL $1BEE      ; SYNTAXPR.: NÄCHSTER BEFEHL
1D13 EF      RST CALRUF      ; SONST CALCULATORAUFRUF
1D14 A1      .BYT $A1        ; UND EINE 1 ALS DEFAULTWERT
1D15 38      .BYT $38        ; FÜR STEP NEHMEN
1D16      ;
1D16      ; DIE LETZTEN DREI WERTE DES CALCULATORSTACKS SIND:
1D16      ; DER WERT DER VARIABLE (W), DIE OBERGRENZE (O) UND
1D16      ; DIE SCHRITTWEITE (S)
1D16      ;
1D16 EF      RST CALRUF      ; CALCULATORAUFRUF
1D17 C0      .BYT $C0        ; W,O,S, S NACH MEMO
1D18 02      .BYT $02        ; W,O, S LÖSCHEN
1D19 01      .BYT $01        ; O,W, VERTAUSCHEN
1D1A E0      .BYT $E0        ; O,W,S, MEMO HOLEN
1D1B 01      .BYT $01        ; O,S,W, VERTAUSCHEN
1D1C 38      .BYT $38        ; ENDE
1D1D      ;
1D1D      ;
1D1D CD FF 2A CALL $2AFF      ; VARIABLE SUCHEN (W) ODER
1D20      ; ANLEGEN
1D20 22 68 5C LD (MEM),HL    ; ZU EINEM SPEICHERBEREICH
1D23      ; MACHEN
1D23 2B      DEC HL          ; ZEIGT AUF VARIABLENNAME
1D24 7E      LD A,(HL)       ; LADE DEN NAMEN
1D25 CB FE    SET 7,(HL)     ; BIT 7 SETZEN: FOR-VARIABLE

```

1D27	01 06 00	LD BC,6	; EINFACHE VARIABLE: 6 PLATZE
1D2A	09	ADD HL,BC	; ÜBERSPRINGEN
1D2B	07	RLCA	; WAR ES SCHON EINE FOR-VARIABLE?
1D2C	38 06	JR C,\$1D34	; JA
1D2E	0E 0D	LD C,\$D	; SONST 13 ZUSÄTZLICHE
1D30	CD 55 16	CALL \$1655	; SPEICHERPLATZE BESCHAFFEN
1D33	23	INC HL	; HL ZEIGT AUF 'OBERGRENZE'
1D34	E5	PUSH HL	; ZWISCHENSPEICHERN
1D35	EF	RST CALRUF	; O,S, CALCULATORAUFRUF
1D36	02	.BYT \$02	; O, LÖSCHE SCHRITTWEITE
1D37	02	.BYT \$02	; -,LÖSCHE OBERGRENZE
1D38	38	.BYT \$38	; ENDE (DE ZEIGT AUF '0')
1D39	E1	POP HL	; POINTER ZURÜCKHOLEN
1D3A	EB	EX DE,HL	; POINTER FÜR TRANSFER TAUSCHEN
1D3B	0E 0A	LD C,10	; ZEHN BYTES FÜR '0' UND
1D3D	ED B0	LDIR	; 'S' VERSCHIEBEN
1D3F	2A 45 5C	LD HL,(PPC)	; DERZEITIGE ZEILENNUMMER
1D42	EB	EX DE,HL	; TAUSCH ZEILENNUMMER - POINTER
1D43	73	LD (HL),E	; ZEILENNUMMER AN FOR-VARIABLE
1D44	23	INC HL	; DRANHÄNGEN
1D45	72	LD (HL),D	
1D46	FD 56 0D	LD D,(IY+\$D)	; ÜBER SUBPPC DEN NÄCHSTEN
1D49	14	INC D	; BEFEHL LADEN UND EBENFALLS
1D4A	23	INC HL	
1D4B	72	LD (HL),D	; AN FOR-VARIABLE HÄNGEN
1D4C			
1D4C			; ES FOLGT DER TEST, OB EINE FOR-NEXT-SCHLEIFE AUSGE-
1D4C			; FÜHRT WERDEN KANN. WENN JA: RETURN, SONST MUSS
1D4C			; NÄCHSTE BEFEHL HINTER NEXT GEFUNDEN WERDEN
1D4C			
1D4C	CD DA 1D	CALL \$1DDA	; OBERGRENZE FOR-NEXT ERREICHT?
1D4F	D0	RET NC	; NEIN
1D50			
1D50	FD 46 38	LD B,(IY+\$38)	; VARIABLENNAME LADEN
1D53	2A 45 5C	LD HL,(PPC)	; DERZEITIGE ZEILENNUMMER
1D56	22 42 5C	LD (NEWPPC),HL	; ZUR NEUEN ZEILENNUMMER MACHEN
1D59	3A 47 5C	LD A,(SUBPPC)	; BEFEHLSZAHL IN DER ZEILE
1D5C	ED 44	NEG	; LADEN, ZWEIFERKOMPLEMENT BILDEN
1D5E	57	LD D,A	; NACH REG D BRINGEN
1D5F	2A 5D 5C	LD HL,(CHADD)	; AKTUELLER WERT CHADD
1D62	1E F3	LD E,\$F3	; DAS TOKEN 'NEXT' WIRD GESUCHT
1D64	C5	PUSH BC	; VARIABLENNAME SPEICHERN
1D65	ED 4B 55 5C	LD BC,(NXTLIN)	; AKT. WERT VON NÄCHSTER ZEILE
1D69	CD 86 1D	CALL \$1D86	; IM PROGRAMM JETZT 'NEXT' SUCHEN
1D6C	ED 43 55 5C	LD (NXTLIN),BC	; NEUER WERT 'NÄCHSTE ZEILE'
1D70	C1	POP BC	; VARIABLENNAME ZURÜCK
1D71	38 11	JR C,\$1D84	; KEIN NEXT GEFUNDEN: ERROR
1D73	E7	RST GETNXT	; CHADD+1, NÄCHSTES ZEICHEN NACH
1D74			; NEXT LADEN: VARIABLENNAME
1D74	F6 20	OR \$20	; KLEIN- UND GROSSBUCHSTABEN ZU-
1D76	B8	CP B	; LASSEN UND VERGLEICHEN
1D77	28 03	JR Z,\$1D7C	; NAME STIMMT
1D79	E7	RST GETNXT	; CHADD +1 UND
1D7A	18 E8	JR \$1D64	; WEITERSUCHEN
1D7C			
1D7C			; NEWPPC ENTHÄLT NUN DIE ZEILENNUMMER, IN DER DAS
1D7C			; RICHTIGE NEXT-STATEMENT GEFUNDEN WURDE. DIE BEFEHLS-
1D7C			; ZAHL DES 'NEXT' WIRD GESUCHT UND IN NSPPC GESPEICHERT

```

107C      ;
107C E7      RST GETNXT      ; CHADD+1
107D 3E 01    LD A,1          ; DA 0 RUCKWARTS BEZAHLT WURDE.
107F 92      SUB D            ; MUSS DIESER WERT VON 1 SEIN
1080 32 44 5C LD (NSPPC),A    ; NSPPC ABGEZOGEN WERDEN
1083 C9      RET
1084      ;
1084 CF      RST ERR AUS      ; FEHLERMELDUNG:
1085 11      .BYT $11         ; 'FOR without NEXT'
1086      ;
1086      ; DIESE ROUTINE DURCHSUCHT DAS PROGRAMM NACH DATA,
1086      ; 'DEF FN' UND 'NEXT'. TOKEN IM REGISTER E UND START-
1086      ; ADRESSE DES SUCHENS IN HL
1086      ;
1086 7E      LD A,(HL)         ; AKTUELLES ZEICHEN
1087 FE 3A    CP ':'           ; WEITERE BEFEHLE IN DER ZEILE?
1089 28 18    JR Z,$1DA3      ; JA
108B 23      INC HL           ; HL ZEIGT AUF ZEILENNUMMERHIGH
108C 7E      LD A,(HL)         ; DER FOLGENDEN ZEILE
108D E6 C0    AND $C0         ; PROGRAMMENDE?
108F 37      SCF              ; CARRY SETZEN FÜR PROGRAMMENDE
1090 C0      RET NZ           ; RETURN BEI ENDE
1091      ;
1091 46      LD B,(HL)         ; NACHSTE
1092 23      INC HL
1093 4E      LD C,(HL)         ; ZEILENNUMMER LADEN UND
1094 ED 43 42 5C LD (NEWPPC),BC ; IN NEWPPC SPEICHERN
1098 23      INC HL           ; ZEILENLANGE
1099 4E      LD C,(HL)
109A 23      INC HL
109B 46      LD B,(HL)         ; LADEN
109C E5      PUSH HL           ; ZEIGER RETTEN
109D 09      ADD HL,BC         ; ENDE DER DERZEITIGEN ZEILE
109E 44      LD B,H            ; AUSRECHNEN UND NACH BC
109F 4D      LD C,L            ; BRINGEN (=NACHSTE ZEILE -1)
10A0 E1      POP HL           ; ZEIGER WIEDER IN HL LADEN
10A1 16 00    LD D,0           ; BEFEHLSZÄHLER AUF 0 SETZEN
10A3 C5      PUSH BC          ; ZEILENENDE RETTEN
10A4 CD 8B 19 CALL $198B      ; BEFEHLE DER ZEILE UNTERSUCHEN
10A7 C1      POP BC           ; ZEILENENDE IN BC ZURÜCKHOLEN
10A9 D0      RET NC           ; RETURN, WENN ETWAS GEFUNDEN
10AB 18 E0    JR $10BB        ; SONST IN DER NÄCHSTEN
10AB      ; ZEILE WEITERSUCHEN
10AB      ;
10AB      ; BEFEHL NEXT
10AB      ; DIE LAUFVARIABLE WIRD UM DEN STEP-WERT ERHÖHT
10AB      ;
10AB FD CB 37 4E BIT 1,(IY+$37) ; VARIABLE GEFUNDEN?
10AF C2 2E 1C JP NZ,$1C2E      ; NEIN: ERROR
10B2 2A 4D 5C LD HL,(DEST)    ; ADRESSE DER VARIABLEN
10B5 CB 7E    BIT 7,(HL)      ; TEST DES NAMENS
10B7 28 1F    JR Z,$10DB      ; ERROR: 'NEXT WITHOUT FOR'
10B9 23      INC HL           ; NAME ÜBERSIEHEN UND DIE
10BA 22 6B 5C LD (MEM),HL     ; VARIABLE ZUM TEMPORÄREN
10BD      ; SPEICHER DEKLARIEREN
10BD EF      RST CALRUF       ; CALCULATORAUFRUF
10BE E0      .BYT $E0         ; M, HOLE MEM0
10BF E2      .BYT $E2         ; M,S HOLE MEM2

```


1DC0	0F	.BYT \$0F	; W+S ADDIERE 'STEP' ZUR LAUF-
1DC1			; VARIABLEN
1DC1	C0	.BYT \$C0	; W+S SPEICHERE IN MEMO
1DC2	02	.BYT \$02	; LÖSCHEN
1DC3	38	.BYT \$38	; ENDE
1DC4			
1DC4	CD DA 1D	CALL \$1DDA	; LAUFWERT GEGEN OBERGRENZE
1DC7			; PRÜFEN
1DC7	D8	RET C	; RETURN, WENN ENDE ERREICHT
1DC8	2A 68 5C	LD HL, (MEM)	; LOWBYTEADRESSE DER ZEILENNUM-
1DCB	11 0F 00	LD DE, \$F	; MER DES SCHLEIFENBEGINNS
1DCE	19	ADD HL, DE	; BERECHNEN
1DCF	5E	LD E, (HL)	; DIESE ZEILENNUMMER
1DD0	23	INC HL	
1DD1	56	LD D, (HL)	; NACH DE LADEN
1DD2	23	INC HL	
1DD3	66	LD H, (HL)	; BEFEHLSZAHL IN DER ZEILE LADEN
1DD4	EB	EX DE, HL	; TAUSCH, DAMIT WIE BEIM GO TO
1DD5	C3 73 1E	JP \$1E73	; VERFAHREN WERDEN KANN
1DD8			
1DD8	CF	RST ERR AUS	; FEHLERMELDUNG
1DD9	00	.BYT \$00	; 'NEXT WITHOUT FOR'
1DDA			
1DDA			; ÜBERPRÜFUNG, OB DIE OBER-/UNTERGRENZE EINER FOR-
1DDA			; NEXT-SCHLEIFE ERREICHT (JE NACH VORZEICHEN VON STEP)
1DDA			
1DDA	EF	RST CALRUF	; CALCULATORAUFRUF
1DD8	E1	.BYT \$E1	; O, HOLE MEMO
1DDC	E0	.BYT \$E0	; O, W, HOLE MEM1
1DD0	E2	.BYT \$E2	; O, W, S HOLE MEM2
1DDE	36	.BYT \$36	; O, W, S < NULL?
1DDF	00	.BYT \$00	; CALACULATORSPRUNG UM 2, FALLS
1DE0	02	.BYT \$02	; DIES ZUTRIFFT (NXTNEG)
1DE1	01	.BYT \$01	; W, O, AUSTAUSCH
1DE2	03	NXTNEG .BYT \$03	; W-O ODER O-W, SUBTRAKTION
1DE3	37	.BYT \$37	; ERGEBNIS > NULL?
1DE4	00	.BYT \$00	; CALCULATORSPRUNG (NXTEND),
1DE5	04	.BYT \$04	; WENN DIES ZUTRIFFT
1DE6	38	.BYT \$38	; ENDE
1DE7	A7	AND A	; CARRY LÖSCHEN: FOR-NEXT NOCHT
1DE8	C9	RET	; NICHT ZU ENDE
1DE9			
1DE9	38	NXTEND .BYT \$38	; ENDE
1DEA	37	SCF	; FOR-NEXT-SCHLEIFE FERTIG
1DEB	C9	RET	
1DEC			
1DEC			; BEFEHL READ
1DEC			; CHADD WIRD ALS ZEIGER ENTLANG DER EINZELNEN DATA-
1DEC			; STATEMENTS BENUTZT. DATADD ZEIGT AUF DAS AKTUELLE
1DEC			; ELEMENT DER DATENLISTE (NÖTIG FÜR MEHRERE READBEFEHLE)
1DEC			
1DEC	E7	RST GETNXT	; CHADD+1, NÄCHSTES ZEICHEN
1DED	CD 1F 1C	CALL \$1C1F	; NACHSEHEN, OB DIE VARIABLE
1DF0			; SCHON BENUTZT WORDEN IST
1DF0	CD 30 25	CALL \$2530	; BEI SYNTAXPRÜFUNG REST
1DF3	28 29	JR Z, \$1E1E	; ÜBERSPRINGEN
1DF5	DF	RST GETAKT	; CHADD NOCHMAL HOLEN, UM
1DF6	22 5F 5C	LD (XPTR), HL	; DIESES ZWISCHENZUSPEICHERN

1DF9	2A 57 5C	LD HL,(DATADD)	; ZEIGER DER DATENLISTE LADEN
1DFC	7E	LD A,(HL)	; FALLS KOMMA ALS TRENNZEICHEN,
1DFD	FE 2C	CP ','	
1DFF	28 09	JR Z,\$1E0A	; GEHT DIE AUFGABUNG WEITER
1E01	1E E4	LD E,\$E4	; SONST NACHSTES DATA-STATEMENT
1E03	CD 86 1D	CALL \$1D86	; IM PROGRAMM SUCHEN
1E06	30 02	JR NC,\$1E0A	; 'DATA' WURDE GEFUNDEN
1E08			
1E08	CF	RST ERR AUS	; SONST FEHLERMELDUNG:
1E09	0D	.BYT \$0D	; 'OUT OF DATA'
1E0A			
1E0A	CD 77 00	CALL \$77	; CHADD +1
1E0D	CD 56 1C	CALL \$1C56	; DEN WERT HOLEN UND DER
1E10			; VARIABLEN ZUWEISEN
1E10	DF	RST GETAKT	; AKTUELLEN WERT CHADD IN HL.
1E11	22 57 5C	LD (DATADD),HL	; ZUM ZWISCHENSPEICHERN, LADEN
1E14	2A 5F 5C	LD HL,(XPTR)	; ZEIGER AUF READ-BEFEHL WIEDER
1E17			; LADEN UND
1E17	FD 36 26 00	LD (IY+\$26),0	; XPTRHI LOSCHEN
1E1B	CD 78 00	CALL \$78	; HL NUR IN CHADD SPEICHERN
1E1E	DF	RST GETAKT	; AKTUELLES ZEICHEN JETZT
1E1F	FE 2C	CP ','	; EIN KOMMA?
1E21	28 09	JR Z,\$1DEC	; JA: WEITERE DATEN EINLESEN
1E23	CD EE 1B	CALL \$1BEE	; SONST ENDEPRÜFUNG (=ENDE BEI
1E26			; SYNTAXPRÜFUNG)
1E26	C9	RET	; RETURN IM PROGRAMMLAUF
1E27			
1E27			; BEFEHL DATA
1E27			; BEIM PROGRAMMLAUF WERDEN DIE DATA-STATEMENTS
1E27			; ÜBERSPRUNGEN
1E27			
1E27	CD 30 25	CALL \$2530	; PROGRAMMLAUF?
1E2A	20 0B	JR NZ,\$1E37	; JA
1E2C	CD FB 24	CALL \$24FB	; NACHSTEN AUSDRUCK AUSWERTEN
1E2F	FE 2C	CP ','	; TRENNZEICHEN VORHANDEN?
1E31	C4 EE 1B	CALL NZ,\$1BEE	; NEIN: AUF ENDE PRÜFEN
1E34	E7	RST GETNXT	; CHADD+1 UND WEITER IN DER
1E35	1B F5	JR \$1E2C	; DATASTATEMENTSCHLEIFE
1E37			
1E37	3E E4	LD A,\$E4	; DATABEFEHL ÜBERGEHEN
1E39			
1E39			; ROUTINE ZUM ÜBERSPRINGEN VON PROGRAMMTEILEN
1E39			
1E39	47	LD B,A	; REG BC EINE HOHE ZAHL ZUWEISEN
1E3A	ED B9	CPDR	; UND DAS TOKEN SUCHEN
1E3C	11 00 02	LD DE,\$200	; AB DEM 2. ZEICHEN DANACH
1E3F	C3 BB 19	JP \$198B	; DEN NACHSTEN BEFEHL SUCHEN
1E42			
1E42			; BEFEHL RESTORE
1E42			; EIN OPERAND WIRD ALS ZEILENNUMMER INTERPRETIERT,
1E42			; WENN KEINER VORHANDEN, DEFAULTWERT NULL
1E42			
1E42	CD 99 1E	CALL \$1E99	; ZEILENNUMMER NACH BC BRINGEN
1E45	60	LD H,B	; UND INS HL REGISTER
1E46	69	LD L,C	; KOPIEREN
1E47	CD 6E 19	CALL \$196E	; DIE ADRESSE DER ZEILE SUCHEN
1E4A	28	DEC HL	; DAVON 1 SUBTRAHIEREN UND
1E4B	22 57 5C	LD (DATADD),HL	; IN DATADD SPEICHERN

```

1E4E C9          RET
1E4F          ;
1E4F          ; BEFEHL RANDOMIZE
1E4F          ; FALLS DER OPERAND NULL IST, WIRD STATTDESSEN FRAMES
1E4F          ; ALS ERSATZ GENOMMEN
1E4F          ;
1E4F CD 99 1E    CALL $1E99      ; WERT NACH BC HOLEN
1E52 78          LD A,B          ; UND AUF
1E53 B1          OR C            ; NULL TESTEN
1E54 20 04       JR NZ,$1E5A     ; WAR NICHT NULL
1E56 ED 48 78 5C LD BC,(FRAMES) ; SONST FRAMES LADEN
1E5A ED 43 76 5C LD (SEED),BC   ; UND IN SEED SPEICHERN
1E5E C9          RET
1E5F          ;
1E5F          ; BEFEHL CONTINUE
1E5F          ; DIE ENTSPRECHENDEN ZEILEN- UND BEFEHLSNUMMERN WERDEN
1E5F          ; GELADEN, UM DANN EIN 'GO TO' AUSZUFÜHREN
1E5F          ;
1E5F 2A 6E 5C    LD HL,(OLDPPC)  ; ZEILENNUMMER
1E62 FD 56 36    LD D,(IY+$36)   ; BEFEHLSZAHL IN DIESER ZEILE
1E65 18 0C       JR $1E73
1E67          ;
1E67          ; BEFEHL GO TO
1E67          ; DIE ZEILENNUMMER SOLLTE IM BEREICH VON 0 - 9999 SEIN,
1E67          ; ES WIRD JEDOCH NUR AUF >61439 GEPRÜFT
1E67          ;
1E67 CD 99 1E    CALL $1E99      ; ZEILENNUMMER NACH BC
1E6A 60          LD H,B          ; NACH HL KOPIEREN
1E6B 69          LD L,C
1E6C 16 00       LD D,0
1E6E 7C          LD A,H
1E6F FE F0       CP $F0          ; TEST > 61439 ?
1E71 30 2C       JR NC,$1E9F     ; JA: ERROR
1E73          ;
1E73 22 42 5C    LD (NEWPPC),HL  ; SONST ZEILENNUMMER SPEICHERN
1E76 FD 72 0A    LD (IY+$A),D    ; BEFEHLSZAHL SPEICHERN
1E79 C9          RET
1E7A          ;
1E7A          ; BEFEHL OUT
1E7A          ; DIE ZWEI PARAMETER WERDEN VOM CALCULATORSTACK
1E7A          ; GEHOLT UND AUSGEGEBEN
1E7A          ;
1E7A CD 85 1E    CALL $1E85      ; BEIDE PARAMETER HOLEN
1E7D ED 79       OUT (C),A        ; REG A AUF PORT (C) AUSGEBEN
1E7F C9          RET
1E80          ;
1E80          ; BEFEHL POKE
1E80          ; DIE BEIDEN PARAMETER WERDEN, WIE BEI OUT, VOM
1E80          ; CALCULATORSTACK GENOMMEN
1E80          ;
1E80 CD 85 1E    CALL $1E85      ; PARAMETER HOLEN UND
1E83 02          LD (BC),A        ; DEN POKE-BEFEHL AUSFÜHREN
1E84 C9          RET
1E85          ;
1E85          ; DIE ZWEI OBERSTEN PARAMETER VOM CALCULATORSTACK
1E85          ; ENTNEHMEN: DER ERSTE MUSS IM BEREICH VON 0 - 255 SEIN,
1E85          ; (ES WIRD DAS ZWEIERKOMPLEMENT GEBILDET, FALLS NEGATIV)
1E85          ; DER ZWEITE IM BEREICH VON 0 - 65535 (INTEGER)

```

```

1E85 ;
1E85 CD D5 2D CALL $2DD5 ; ERSTEN PARAMETER HOLEN
1E88 38 15 JR C,$1E9F ; ERROR, WENN ZU GROSS
1E8A 28 02 JR Z,$1E8E ; SPRUNG, WENN POSITIV
1E8C ED 44 NEG ; ZWEIERKOMPLEMENTBILDUNG
1E8E F5 PUSH AF ; ZWISCHENSPEICHERN
1E8F CD 99 1E CALL $1E99 ; DEN ZWEITEN PARAMETER NACH BC
1E92 F1 POP AF ; BRINGEN UND DEN ERSTEN WIEDER
1E93 C9 RET ; LADEN
1E94 ;
1E94 ; SUBROUTINE, UM INTEGERZAHLEN VOM CALCULATORSTACK ZU
1E94 ; HOLEN. INTEG1 FÜR ZAHLEN VON 0 - 255 (1 BYTE),
1E94 ; INTEG2 FÜR ZAHLEN VON 0 - 65535 (2 BYTE)
1E94 ;
1E94 CD D5 2D INTEG1 CALL $2DD5 ; LETZTEN WERT VOM CALC.-STACK
1E97 18 03 JR $1E9C
1E99 CD A2 2D CALL $2DA2 ; LETZTEN WERT VOM CALC.-STACK
1E9C 38 01 JR C,$1E9F ; ZAHL ZU GROSS
1E9E C8 RET Z ; RETURN NUR POSITIVE ZAHLEN IM
1E9F ; ENTSPRECHENDEN BEREICH
1E9F ;
1E9F CF RST ERR AUS ; FEHLERMELDUNG:
1EA0 0A .BYT $0A ; 'INTEGER OUT OF RANGE'
1EA1 ;
1EA1 ; BEFEHL RUN
1EA1 ; PARAMETER VON RUN WIRD ÜBER GO TO ZUGEWIESEN, DANACH
1EA1 ; WIRD EIN 'RESTORE 0' UND 'CLEAR' AUSGEFÜHRT
1EA1 ;
1EA1 CD 67 1E CALL $1E67 ; ÜBER GO TO DEN NEWPPC SETZEN
1EA4 01 00 00 LD BC,0 ; RESTORE 0
1EA7 CD 45 1E CALL $1E45 ; AUSFÜHREN
1EAA 18 03 JR $1EAF ; UND ZUM BEFEHL CLEAR
1EAC ;
1EAC ; BEFEHL CLEAR
1EAC ; HIERMIT WERDEN DIE VARIABLEN UND DER BILDSCHIRM
1EAC ; GELOESCHT. RAMTOP UND DER STACK WERDEN NEU ANGELEGT
1EAC ;
1EAC CD 99 1E CALL $1E99 ; WERT NACH BC HOLEN
1EAF 78 LD A,B ; DIESEN AUF
1EB0 B1 OR C ; NULL TESTEN
1EB1 20 04 JR NZ,$1EB7
1EB3 ED 4B B2 5C LD BC,(RAMTOP) ; WENN 0, DANN RAMTOP NEHMEN
1EB7 C5 PUSH BC ; ZWISCHENSPEICHERN
1EB8 ED 5B 4B 5C LD DE,(VARS) ; DEN SPEICHERBEREICH
1EBC 2A 59 5C LD HL,(ELINE)
1EBF 2B DEC HL ; ALLER VERWENDETE VARIABLEN
1EC0 CD E5 19 CALL RAUS1 ; FREIGEBEN
1EC3 CD 6B 0D CALL $D6B ; BILDSCHIRM LÖSCHEN
1EC6 2A 65 5C LD HL,(STKEND) ; AKTUELLER WERT VON STACKEND
1EC9 11 32 00 LD DE,$32 ; 50 FREIE SPEICHERPLATZE
1ECC 19 ADD HL,DE ; ZUM TEST ADDIEREN
1ECD D1 POP DE ; STKEND ODER $0000 HOLEN
1ECE ED 52 SBC HL,DE ; TEST, OB GRENZE ZU TIEF
1ED0 30 08 JR NC,$1EDA ; JA, ERROR
1ED2 2A B4 5C LD HL,(PRAMT) ; TEST OBERGRENZE MIT PRAMT
1ED5 A7 AND A ; CARRY LÖSCHEN
1ED6 ED 52 SBC HL,DE ; UND SUBTRAHIEREN
1ED8 30 02 JR NC,$1EDC ; ALLES IN ORDNUNG

```

```

1EDA      ;
1EDA CF      RST ERR AUS      ; FEHLERMELDUNG:
1EDB 15      .BYT $15      ; 'RAMTOP NO GOOD'
1EDC      ;
1EDC EB      EX DE,HL      ; TAUSCH, UM RAMTOP
1EDD 22 B2 5C LD (RAMTOP),HL ; NEU ZU SETZEN
1EE0 D1      POP DE      ; RETURNADRESSE BRKTST
1EE1 C1      POP BC      ; ERRORRETURNADRESSE
1EE2 36 3E    LD (HL), $3E ; 'GO SUB' ENDEMARKIERUNG
1EE4 2B      DEC HL      ; AUF DEN STACK UND
1EE5 F9      LD SP,HL     ; STACKPOINTER NEU SETZEN
1EE6 C5      PUSH BC     ; ERRORRETURNADRESSE UND DEN
1EE7 ED 73 3D 5C LD (ERRSP),SP ; STACKPOINTER SPEICHERN
1EE8 EB      EX DE,HL     ; RETURNADRESSE BRKTST IN HL
1EEC E9      JP (HL)      ; BRINGEN UND HIERÜBER RETURN
1EED      ;
1EED      ; BEFEHL GO SUB
1EED      ; DER AKTUELLE WERT VON PPC UND SUBPPC +1 WERDEN IM
1EED      ; 'GO SUB'-STACK GESPEICHERT
1EED      ;
1EED D1      POP DE      ; RETURNADRESSE BRKTST
1EEE FD 66 0D LD H, (IY+$D) ; SUBPPC LADEN UND
1EF1 24      INC H      ; UM 1 ERHÖHEN
1EF2 E3      EX (SP),HL  ; ERRORADRESSE MIT BEFEHLSZAHL
1EF3 33      INC SP      ; TAUSCHEN UND DEN 'L'-PLATZ
1EF4      ; WIEDER FEIGEBEN
1EF4 ED 4B 45 5C LD BC, (PPC) ; PPC IM
1EF8 C5      PUSH BC     ; STACK SPEICHERN
1EF9 E5      PUSH HL     ; ERRORADRESSE WIEDER AUF STACK
1EFA ED 73 3D 5C LD (ERRSP),SP ; ERRORSTACKPOINTER NEU SETZEN
1EFE D5      PUSH DE     ; RETURNADRESSE BRKTST AUF STACK
1EFF CD 67 1E CALL $1E67 ; NEWPPC, NSPPC SETZEN
1F02 01 14 00 LD BC, $14 ; VOR AUSFÜHRUNG VON GO SUB NOCH
1F05      ; EIN SPEICHERTEST DURCHFÜHREN
1F05      ;
1F05      ; TESTROUTINE FÜR BENÖTIGTEN SPEICHERPLATZ (IN BC)
1F05      ;
1F05 2A 65 5C LD HL, (STKEND) ; ZU STACKEND DEN
1F08 09      ADD HL,BC     ; BENÖTIGTEN PLATZ ADDIEREN
1F09 38 0A    JR C, $1F15 ; ERGEBNIS > 65535 ($FFFF)
1F0B EB      EX DE,HL     ; MIT ZUSÄTZLICHEN 80
1F0C 21 50 00 LD HL, $50 ; PLATZEN TESTEN
1F0F 19      ADD HL,DE     ;
1F10 38 03    JR C, $1F15 ; WIEDER >65535
1F12 ED 72    SBC HL,SP    ; MIT STACKPOINTER VERGLEICHEN
1F14 D8      RET C      ; GENÜGEN PLATZ VORHANDEN
1F15      ;
1F15      ; FEHLERMELDUNG 'OUT OF MEMORY'
1F15      ;
1F15 2E 03    LD L, 3      ; DA LAUFZEITFEHLER, ERFOLGT DIE
1F17 C3 55 00 JP $55      ; MELDUNG NICHT ÜBER RSTB
1F1A      ;
1F1A      ; ROUTINE ZUM BERECHNEN DES FREIEN SPEICHERPLATZES (FRE)
1F1A      ; DER FREIE PLATZ ERGIBT SICH DURCH: PRINT65535-USR7962
1F1A      ;
1F1A 01 00 00 LD BC, 0      ; SPEICHERTEST OHNE ZUSÄTZLICHE
1F1D CD 05 1F CALL $1F05 ; PLATZE DURCHFÜHREN
1F20 44      LD B, H      ; ERGENIS NACH

```

```

1F21 4D          LD C,L          ; BC BRINGEN
1F22 C9          RET
1F23            ;
1F23            ; BEFEHL RETURN
1F23            ; ZEILENUMMER UND BEFEHLSZAHL IN DER ZEILE WERDEN VOM
1F23            ; 'GO SUB'-STACK GENOMMEN
1F23            ;
1F23 C1          POP BC          ; RETURNADRESSE BRÄTST
1F24 E1          POP HL          ; ERRORRETURNADRESSE
1F25 D1          POP DE          ; LETZTER EINTRAG IM 'GO SUB'-
1F26 7A          LD A,D          ; STACK MIT
1F27 FE 3E       CP $3E          ; ENDEMERKER VERGLEICHEN
1F29 28 0B       JR Z,$1F36      ; WENN JA: ERROR
1F2B 3B          DEC SP          ; STACKPOINTER NEU SETZEN
1F2C E3          EX (SP),HL      ; BEFEHLSZAHL MIT ERRORRETURN-
1F2D            ; ADRESSE AUSTAUSCHEN
1F2D EB          EX DE,HL        ; BEFEHLSZAHL NACH D
1F2E ED 73 3D 5C LD (ERRSP),SP  ; ERRORSTACKPOINTER NORMAL
1F32 C5          PUSH BC         ; RETURNADRESSE BRÄTST ZURÜCK
1F33 C3 73 1E    JP $1E73        ; UND NEWPPC + NSPPC NEU SETZEN
1F36            ;
1F36 D5          PUSH DE         ; ENDEMERKER UND
1F37 E5          PUSH HL        ; ERRORADRESSE AUF STACK ZURÜCK
1F38 CF          RST ERRAUS      ; FEHLERMELDUNG:
1F39 06          .BYT $06        ; 'RETURN WITHOUT GOSUB'
1F3A            ;
1F3A            ; BEFEHL PAUSE
1F3A            ; ES WERDEN HIERBEI DIE INTERRUPTS GEZÄHLT, DIE 50 MAL
1F3A            ; PRO SEKUNDE (FÜR DIE TASTATURABFRAGE) AUFTRETEN. WENN
1F3A            ; EINE TASTE GEDRÜCKT WIRD, WIRD DIE PAUSE EBENFALLS
1F3A            ; BEENDET
1F3A            ;
1F3A CD 99 1E    CALL $1E99      ; ZÄHLWERT NACH BC LADEN
1F3D 76          HALT            ; AUF NÄCHSTEN INTERRUPT WARTEN
1F3E 0B          DEC BC          ; ZÄHLER -1
1F3F 78          LD A,B          ; FERTIG GEZÄHLT
1F40 B1          OR C            ;
1F41 28 0C       JR Z,$1F4F      ; JA
1F43 78          LD A,B          ; FALLS BC VOR DEC BC SCHON 0
1F44 A1          AND C          ; WAR, IST BC JETZT $FFFF
1F45 3C          INC A          ; DIES WIRD GETESTET
1F46 20 01       JR NZ,$1F49     ; BC WAR NICHT NULL
1F48 03          INC BC         ; SONST BC WIEDER AUF NULL
1F49 FD CB 01 6E BIT 5,(IY+1)    ; SETZEN UND WARTEN, BIS EINE
1F4D            ; TASTE GEDRÜCKT WIRD
1F4D 28 EE       JR Z,$1F3D      ; NOCH NICHT DER FALL
1F4F            ;
1F4F FD CB 01 AE RES 5,(IY+1)    ; PAUSE FERTIG, KEINE TASTE
1F53 C9          RET            ; GEDRÜCKT ANMERKEN
1F54            ;
1F54            ; SUBROUTINE, UM AUF GEDRÜCKTE BREAK-TASTE ZU PRÜFEN
1F54            ;
1F54 3E 7F       LD A,$7F        ; PORTADRESSE BILDEN
1F56 DB FE       IN A,($FE)      ; TASTENMATRIX LADEN
1F58 1F          RRA             ; BIT 0 INS CARRY
1F59 DB          RET C           ; RETURN, WENN NICHT GEDRÜCKT
1F5A            ;
1F5A 3E FE       LD A,$FE        ; SONST PORTADRESSE FÜR SHIFT

```

3 in A, 7FF

1F5C	DB FE	IN A, (#FE)	; BILDEN UND TASTENMATRIX LADEN
1F5E	1F	RRA	; CARRY IST GELÖSCHT, WENN BEIDE
1F5F	C9	RET	; TASTEN GEDRÜCKT, SONST GESETZT
1F60			
1F60		; BEFEHL DEF FN	
1F60		; ZUR LAUFZEIT WIRD 'DEF FN' ÜBERSPRUNGEN (WIE DATA).	
1F60		; BEI DER SYNTAXPRÜFUNG WIRD DER AUSDRUCK GEPRÜFT	
1F60			
1F60	CD 30 25	CALL #2530	; SYNTAXPRÜFUNG ?
1F63	28 05	JR Z, #1F6A	; JA
1F65	3E CE	LD A, #CE	; TOKEN FÜR DEF FN LADEN
1F67	C3 39 1E	JP #1E39	; UND ÜBERGEHEN DES BEFEHLS
1F6A			
1F6A	FD CB 01 F6	SET 6, (IY+1)	; NUMERISCHE VARIABLE ANMERKEN
1F6E	CD 8D 2C	CALL #2C8D	; IST AKTUELLES ZEICHEN EIN
1F71			; BUCHSTABE ?
1F71	30 16	JR NC, #1F89	; NEIN
1F73	E7	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
1F74	FE 24	CP '\$'	; DOLLARZEICHEN ?
1F76	20 05	JR NZ, #1F7D	; NEIN
1F78	FD CB 01 B6	RES 6, (IY+1)	; JA: STRINGVARIABLE
1F7C	E7	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
1F7D	FE 28	CP '('	; KLAMMER AUF ?
1F7F	20 3C	JR NZ, #1FBD	; NEIN: ERROR
1F81	E7	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
1F82	FE 29	CP ')'	; KLAMMER ZU (=KEINE PARAMETER)?
1F84	28 20	JR Z, #1FA6	; JA
1F86			
1F86		; SCHLEIFE, UM ALLE PARAMETER NACHEINANDER ABZUARBEITEN	
1F86			
1F86	CD 8D 2C	CALL #2C8D	; AKT. ZEICHEN = BUCHSTABE ?
1F89	D2 8A 1C	JP NC, #1C8A	; NEIN: ERROR
1F8C	EB	EX DE, HL	; HL IN DE ZWISCHENSPEICHERN
1F8D	E7	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
1F8E	FE 24	CP '\$'	; DOLLARZEICHEN ?
1F90	20 02	JR NZ, #1F94	; NEIN
1F92	EB	EX DE, HL	; JA, HL-NEU IN DE RETTEN
1F93	E7	RST GETNXT	; CHADD+1, ...
1F94	EB	EX DE, HL	; ZEIGER AUF LETZTEN NAMEN IN HL
1F95	01 06 00	LD BC, 6	; 6 SPEICHERPLATZE NACH DEM
1F98	CD 55 16	CALL #1655	; NAMEN FREIMACHEN
1F98	23	INC HL	; EINEN 'ZÄHLENMERKER'
1F9C	23	INC HL	; IN DIE ERSTE FREIE POSITION
1F9D	36 0E	LD (HL), #E	; NACH DEM NAMEN SCHREIBEN
1F9F	FE 2C	CP ','	; WENN AKT. ZEICHEN KEIN KOMMA
1FA1	20 03	JR NZ, #1FA6	; IST, AUS SCHLEIFE SPRINGEN
1FA3	E7	RST GETNXT	; SONST SOLLTEN WEITERE PARA-
1FA4	18 E0	JR #1F86	; METER VORHANDEN SEIN
1FA6			
1FA6	FE 29	CP ')'	; KLAMMER ZU MUSS EXISTIEREN
1FA8	20 13	JR NZ, #1FBD	; NICHT: ERROR
1FAA	E7	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
1FAB	FE 3D	CP '='	; MUSS EIN '=' SEIN
1FAD	20 0E	JR NZ, #1FBD	; NEIN: ERROR
1FAF	E7	RST GETNXT	; CHADD+1
1FB0	3A 3B 5C	LD A, (FLAGS)	; ART DER FUNKTION
1FB3	F5	PUSH AF	; ZWISCHENSPEICHERN
1FB4	CD FB 24	CALL #24FB	; DEFINITION WIE EINEN AUSDRUCK


```

1FB7
1FB7 F1
1FB8 FD AE 01
1FB8 E6 40
1FBD C2 8A 1C
1FC0 CD EE 1B
1FC3
1FC3
1FC3
1FC3
1FC3 CD 30 25
1FC6 E1
1FC7 C8
1FC8 E9
1FC9
1FC9
1FC9
1FC9
1FC9 3E 03
1FCB 1B 02
1FCD 3E 02
1FCF CD 30 25
1FD2 C4 01 16
1FDS CD 4D 0D
1FDB CD DF 1F
1FDB CD EE 1B
1FDE C9
1FDF
1FDF DF
1FE0 CD 45 20
1FE3 2B 0D
1FE5 CD 4E 20
1FE8 2B FB
1FEA CD FC 1F
1FED
1FED CD 4E 20
1FF0 2B F3
1FF2 FE 29
1FF4 C8
1FF5
1FF5
1FF5
1FF5
1FF5 CD C3 1F
1FF8
1FF8 3E 0D
1FFA D7
1FFB C9
1FFC
1FFC
1FFC
1FFC
1FFC DF
1FFD FE AC
1FFF 20 0D

POP AF
XOR (IY+1)
AND $40
JP NZ,$1C8A
CALL $1BEE

; BEARBEITEN
; ART WIEDER HOLEN UND MIT
; ERGEBNIS DES AUSDRUCKS
; VERGLEICHEN
; UNGLEICH: ERROR
; RETURN ÜBER ENDEPRÜFUNG

; STACKKORREKTURROUTINE FÜR VERSCHIEDENE GELEGENHEITEN
; BEI DER SYNTAXPRÜFUNG

CALL $2530
POP HL
RET Z
JP (HL)

; SYNTAXPRÜFUNG
; EINE RETURNADRESSE VON STACK
; RETURN BEI SYNTAXPRÜFUNG
; LAUFZEIT KEINE KORREKTUR
; DURCH INDIREKTEN SPRUNG

; BEFEHLE LPRINT UND PRINT
; DER ERFORDERLICHE KANAL WIRD GEÖFFNET

LD A,3
JR $1FCF
LD A,2
CALL $2530
CALL NZ,$1601
CALL $D4D
CALL $1FDF
CALL $1BEE
RET

; KANAL P ÖFFNEN
; KANAL S ÖFFNEN
; SYNTAXPRÜFUNG ?
; NEIN: ERÖFFNEN
; TEMPORÄRE FARBVARIABLE SETZEN
; PRINT-STEUERROUTINE
; ENDEPRÜFUNG

RST GETAKT
CALL $2045
JR Z,$1FF2
CALL $204E
JR Z,$1FE5
CALL $1FFC

; ERSTES ZEICHEN HOLEN
; SPRUNG, WENN PRINTAUSDRUCK ZU
; ENDE IST
; IRGEND EIN POSITIONIERZEICHEN?
; JA
; EINZELNEN PRINTAUSDRUCK
; BEARBEITEN
; POSITIONIER- ODER ENDEZEICHEN?
; JA: WEITER AUSDRUCKEN
; WENN KLAMMER ZU, RETURN

CALL $204E
JR Z,$1FE5
CP ')'
RET Z

; SUBROUTINE, UM EIN CARRIAGE RETURN AUSZUDRUCKEN, ABER
; NUR ZUR LAUFZEIT

CALL $1FC3

LD A,$D
RST PRTOU
RET
.END
.LIB SPEC2000-S

; SINCLAIR ZX SPECTRUM TEIL 2000

; SUBROUTINE ZUM AUSGEBEN VON AUSDRUCKEN BEI PRINT ETC.

RST GETAKT
CP $AC
JR NZ,$200E

; AKTUELLES ZEICHEN LADEN
; TOKEN FÜR 'AT'
; NEIN

```

2001	CD 79 1C	CALL \$1C79	; ZWEI PARAMETER BERECHNEN UND
2004			; AUF DEM CALC.-STACK ABLEGEN
2004	CD C3 1F	CALL \$1FC3	; RETURN, WENN SYNTAXPRÜFUNG
2007	CD 07 23	CALL \$2307	; PARAMETER NACH BC BRINGEN
200A	3E 16	LD A,\$16	; STEUERZEICHEN FÜR 'AT' LADEN
200C	18 10	JR \$201E	
200E			
200E	FE AD	CP \$AD	; TOKEN FÜR 'TAB'?
2010	20 12	JR NZ,\$2024	; NEIN
2012	E7	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
2013	CD 82 1C	CALL \$1C82	; EINEN PARAMETER BERECHNEN UND
2016			; AUF DEM CALC.-STACK ABLEGEN
2016	CD C3 1F	CALL \$1FC3	; RETURN BEI SYNTAXPRÜFUNG
2019	CD 99 1E	CALL \$1E99	; PARAMETER NACH BC BRINGEN
201C	3E 17	LD A,\$17	; STEUERZEICHEN FÜR 'TAB' LADEN
201E	D7	RST PRTOUT	; AT ODER TAB AUSGEBEN
201F	79	LD A,C	; ERSTEN UND
2020	D7	RST PRTOUT	
2021	78	LD A,B	; ZWEITEN PARAMETER AUSGEBEN
2022	D7	RST PRTOUT	
2023	C9	RET	
2024			
2024	CD F2 21	CALL FARBUN	; FARBWERTAUSDRÜCKE?
2027	D0	RET NC	; JA
2028	CD 70 20	CALL \$2070	; UNTERSUCHEN, OB STREAM
2028			; GEÄNDERT WERDEN MUSS
2028	D0	RET NC	; JA
202C			
202C			; DIE ZU DRUCKENDEN ZEICHEN MÜSSEN ENTWEDER EIN STRING
202C			; ODER EIN NUMERISCHER AUSDRUCK SEIN
202C			
202C	CD FB 24	CALL \$24FB	; DEN AUSDRUCK UNTERSUCHEN
202F	CD C3 1F	CALL \$1FC3	; RETURN BEI SYNTAXPRÜFUNG
2032	FD CB 01 76	BIT 6,(IY+1)	; FLAGS: AUSDRUCKART PRÜFEN
2036	CC F1 2B	CALL Z,\$2BF1	; WENN STRING, DIE NOTWENDIGEN
2039			; PARAMETER HOLEN
2039	C2 E3 2D	JP NZ,\$2DE3	; FALLS NUMERISCHER AUSDRUCK
203C			
203C			; AUSGABESCHLEIFE FÜR EINEN STRING
203C			
203C	78	LD A,B	; STRINGENDE
203D	B1	OR C	; ERREICHT?
203E	0B	DEC BC	; (ZÄHLER DER STRINGLÄNGE -1)
203F	CB	RET Z	; JA
2040	1A	LD A,(DE)	; ZEICHEN LADEN
2041	13	INC DE	; POINTER +1
2042	D7	RST PRTOUT	; UND ZEICHEN AUSDRUCKEN
2043	18 F7	JR \$203C	; ZUM SCHLEIFENANFANG
2045			
2045			; SUBROUTINE ZUM UNTERSUCHEN, OB EINE AUSGABE ZU ENDE
2045			; IST. TRIFFT DIES ZU, SO IST DAS ZEROFLAG GESETZT
2045			
2045	FE 29	CP ')'	; KLAMMER ZU?
2047	CB	RET Z	
2048	FE 0D	CP \$D	; CARRIAGE RETURN?
204A	CB	RET Z	
204B	FE 3A	CP ' '	; DOPPELPUNKT?
204D	C9	RET	

```

204E      ;
204E      ; SUBROUTINE ZUM POSITIONIEREN BEIM AUSDRUCKEN
204E      ;
204E      RST GETAKT      ; AKTUELLES ZEICHEN HOLEN
204F      CP ' '          ; IST ES EIN SEMIKOLON?
2051      JR Z,$2067      ; JA
2053      CP ','          ; EIN KOMMA?
2055      JR NZ,$2061     ; NEIN
2057      CALL $2530      ; FALLS SYNTAXPRÜFUNG,
205A      JR Z,$2067      ; NICHTS AUSDRUCKEN
205C      LD A,6          ; SONST KOMMA-STEUERZEICHEN
205E      RST PRTOUT      ; LADEN UND AUSGEBEN
205F      JR $2067
2061      ;
2061      CP ' '          ; IST ES EIN HOCHKOMMA?
2063      RET NZ          ; NEIN: RETURN
2064      ;
2064      CALL $1FF5      ; CARRIAGE RETURN AUSGEBEN
2067      RST GETNXT      ; CHADD +1, NACHSTES ZEICHEN
2068      CALL $2045      ; PRINTENDE ERREICHT?
206B      JR NZ,$206E     ; NEIN
206D      POP BC          ; JA: EINE RETURNADRESSE WEG
206E      CP A            ; ZEROFLAG BEI POSITIVIERZEICHEN
206F      RET            ; IMMER GESETZT
2070      ;
2070      ; SUBROUTINE ZUM STREAM-ANDERN, FALLS DER BENUTZER
2070      ; ES WÜNSCHT
2070      ;
2070      CP '0'          ; FALLS NICHT '0',
2072      SCF              ; RETURN IMMER MIT CARRY GESETZT
2073      RET NZ
2074      RST GETNXT      ; CHADD +1, NACHSTES ZEICHEN
2075      CALL $1C82      ; PARAMETER AUF CALC.-STACK
2078      AND A            ; CARRY LÖSCHEN
2079      CALL $1FC3      ; RETURN BEI SYNTAXPRÜFUNG
207C      CALL $1E94      ; PARAMETER IN A HOLEN
207F      CP $10          ; STREAM > 16 ?
2081      JP NC,$160E     ; JA: ERROR
2084      CALL $1601      ; KANAL ÖFFNEN
2087      AND A            ; CARRY LÖSCHEN
2088      RET
2089      ;
2089      ; BEFEHL INPUT
2089      ; EVENTUELLE AUSGABEN BEIM INPUT WERDEN IM UNTEREN
2089      ; BILDSCHIRMTTEIL GEDRUCKT
2089      ;
2089      CALL $2530      ; SYNTAXPRÜFUNG ?
208C      JR Z,$2096      ; JA
208E      LD A,1          ; KANAL K
2090      CALL $1601      ; ÖFFNEN
2093      CALL $D6E        ; UNTEREN SCHIRMTTEIL LÖSCHEN
2096      LD (Y+2),1      ; UND AUF EINE ZEILE BEGRENZEN
209A      CALL $20C1      ; SUBROUTINE ZUR INPUTAUSFÜHRUNG
209D      CALL $1BEE      ; BEI SYNTAXPRÜFUNG ZUM NACHSTEN
20A0      ; BEFEHL
20A0      LD BC,(SPOSN)   ; AKTUELLE PRINTPOSITION LADEN
20A4      LD A,(DFSZ)     ; IST DIESE ÜBER DEM
20A7      CP B            ; UNTEREN BILDSCHIRMTTEIL ?

```

```

20A8 38 03 JR C,$20AD ; JA
20AA 0E 21 LD C,33 ; SONST PRINTPOSITION AUF DEN
20AC 47 LD B,A ; ANFANG DES UNT. TEILS SETZEN
20AD ED 43 88 5C LD (SPOSN),BC ; PRINTPOSITION NEU SETZEN
20B1 3E 19 LD A,25 ; SCROLLING-ZÄHLER
20B3 90 SUB B
20B4 32 8C 5C LD (SCRCT),A ; NEU SETZEN
20B7 FD CB 02 86 RES 0,(IY+2) ; HAUPTBILDSCHIRM EINSCHALTEN
20BB CD D9 0D CALL $DD9 ; SYSTEMVARIABLEN SETZEN
20BE C3 6E 0D JP $D6E ; UND RETURN ÜBER ROUTINE ZUM
20C1 ; LÖSCHEN DES UNTEREN SCHIRMTTEILS
20C1 ;
20C1 ; EIGENTLICHE INPUTROUTINE
20C1 ;
20C1 CD 4E 20 CALL $204E ; POSITIONIERZEICHEN VORHANDEN ?
20C4 28 FB JR Z,$20C1 ; JA
20C6 FE 28 CP '(' ; SPRUNG, WENN ERSTES ZEICHEN
20C8 20 0E JR NZ,$20D8 ; NICHT KLAMMER AUF IST
20CA E7 RST GETNXT ; CHADD +1, NÄCHSTES ZEICHEN
20CB CD DF 1F CALL $1FDF ; NORMALE PRINTROUTINE AUFRUFEN
20CE DF RST GETAKT ; LETZTES ZEICHEN MUSS
20CF FE 29 CP ')' ; KLAMMER ZU SEIN,
20D1 C2 8A 1C JP NZ,$1C8A ; SONST ERROR
20D4 E7 RST GETNXT ; CHADD +1, NÄCHSTES ZEICHEN
20D5 C3 B2 21 JP $21B2 ; PRÜFEN, OB WEITERE INPUTTERME
20D8 ;
20D8 FE CA CP $CA ; TOKEN FÜR 'LINE' ?
20DA 20 11 JR NZ,$20ED ; NEIN
20DC E7 RST GETNXT ; CHADD +1, NÄCHSTES ZEICHEN
20DD CD 1F 1C CALL $1C1F ; ZIELADRESSE DER VARIABLEN
20E0 ; BESTIMMEN
20E0 FD CB 37 FE SET 7,(IY+$37) ; 'INPUT LINE' ANMERKEN
20E4 FD CB 01 76 BIT 6,(IY+1) ; STRINGVARIABLE ?
20E8 C2 8A 1C JP NZ,$1C8A ; NEIN: ERROR
20EB 18 0D JR $20FA ; ZUR PROMPT-AUSGABE
20ED ;
20ED ; NORMALE INPUTVARIABLEN
20ED ;
20ED CD 8D 2C CALL $2C8D ; SPRUNG, WENN AKTUELLES ZEICHEN
20F0 D2 AF 21 JP NC,$21AF ; KEIN BUCHSTABE IST
20F3 CD 1F 1C CALL $1C1F ; ZIELADRESSE DER VARIABLEN
20F6 ; BESTIMMEN
20F6 FD CB 37 BE RES 7,(IY+$37) ; 'NOT INPUT LINE' ANMERKEN
20FA ;
20FA ; DIE PROMPTAUSGABE WIRD IM WORKSPACE AUFGEBAUT
20FA ;
20FA CD 30 25 CALL $2530 ; SPRUNG, WENN
20FD CA B2 21 JP Z,$21B2 ; SYNTAXPRÜFUNG
2100 CD BF 16 CALL $16BF ; WORKSPACE AUF NULL SSETZEN
2103 21 71 5C LD HL,FLAGX
2106 CB B6 RES 6,(HL) ; STRINGERGEBNIS ANMERKEN
2108 CB EE SET 5,(HL) ; INPUTMODUS ANMERKEN
210A 01 01 00 LD BC,1 ; FÜR PROMPT NUR EINE POSITION
210D CB 7E BIT 7,(HL) ; 'LINE' ?
210F 20 08 JR NZ,$211C ; JA
2111 3A 3B 5C LD A,(FLAGS) ; NUMERISCHE EINGABE ?
2114 E6 40 AND $40
2116 20 02 JR NZ,$211A ; JA

```

2118	0E 03	LD C,3	; STRINGEINTRAG BRAUCHT 3 PLATZE
211A	B6	OR (HL)	; BITS VON FLAG FÜR NUMERISCH
211B	77	LD (HL),A	; SETZEN
211C	F7	RST #30	; BENÖTIGTE SPEICHERPLATZE
211D			; FREIMACHEN
211D	36 0D	LD (HL),#D	; CARRIAGE RETURN ALS ABSCHLUSS
211F	79	LD A,C	; BIT 6 DES C REGISTERS TESTEN
2120	0F	RRCA	
2121	0F	RRCA	; SPRUNG, WENN
2122	30 05	JR NC,\$2129	; EIN PLATZ BENÖTIGT WIRD
2124	3E 22	LD A,""	; ANFUHRUNGSGEICHEN IN
2126	12	LD (DE),A	; DIE ERSTEN ZWEI
2127	2B	DEC HL	
2128	77	LD (HL),A	; PLATZE SCHREIBEN
2129	22 5B 5C	LD (KCUR),HL	; CURSORPOSITION SPEICHERN
212C	FD CB 37 7E	BIT 7,(IY+#37)	; "INPUT LINE"
2130	20 2C	JR NZ,\$215E	; JA
2132	2A 5D 5C	LD HL,(CHADD)	; CHADD UND
2135	E5	PUSH HL	
2136	2A 3D 5C	LD HL,(ERRSP)	; ERRORSTACKPOINTER AUF DEN
2139	E5	PUSH HL	; STACK ZWISCHENSPEICHERN
213A	21 3A 21	LD HL,\$213A	; RETURNADRESSE FÜR ERRORS
213D	E5	PUSH HL	
213E	FD CB 30 66	BIT 4,(IY+#30)	; FLAGS2: WIRD KANAL K BENUTZT ?
2142	2B 04	JR Z,\$2148	; NEIN
2144	ED 73 3D 5C	LD (ERRSP),SP	; JA: ERRORSTACKPOINTER NEU
2148	2A 61 5C	LD HL,(WORKSP)	; HL AUF BEGINN VON "INPUT LINE"
2148			; SETZEN UND EVENTUELLE
2148	CD A7 11	CALL \$11A7	; FLOATINGPOINTZAHLEN ENTFERNEN
214E	FD 36 00 FF	LD (IY+0),\$FF	; "KEIN ERROR" ANMERKEN
2152	CD 2C 0F	CALL \$F2C	; INPUT ÜBER EDITOR HOLEN UND
2155	FD CB 01 BE	RES 7,(IY+1)	; SYNTAXFLAG ZUR INPUTPRÜFUNG
2159	CD B9 21	CALL \$21B9	; SETZEN UND DEN INPUT TESTEN
215C	18 03	JR \$2161	
215E			
215E	CD 2C 0F	CALL \$F2C	; DIE INPUTZEILE HOLEN
2161			
2161	FD 36 22 00	LD (IY+\$22),0	; CURSORADRESSE ZURÜCKSETZEN
2165	CD D6 21	CALL \$21D6	; INPUT ÜBER KANAL K
2168	20 0A	JR NZ,\$2174	; NEIN
216A	CD 1D 11	CALL \$111D	; EINGABEZEILE AUF DEN SCHIRM
216D	ED 4B B2 5C	LD BC,(ECHOE)	; KOPIEREN UND POSITION ECHOE
2171	CD D9 0D	CALL \$DD9	; ZUR AKTUELLEN IM UNTEREN
2174			; BILDSCHIRMTEIL MACHEN
2174	21 71 5C	LD HL,FLAGX	
2177	CB AE	RES 5,(HL)	; "EDITOR-MODUS" ANMERKEN
2179	CB 7E	BIT 7,(HL)	; "INPUT LINE"
217B	CB BE	RES 7,(HL)	; (DIESES IMMER RÜCKSETZEN)
217D	20 1C	JR NZ,\$219B	; JA
217F	E1	POP HL	; RETURNADRESSE \$213A WEGWERFEN
2180	E1	POP HL	; ERRORSTACKPOINTER AUF
2181	22 3D 5C	LD (ERRSP),HL	; DEN ALTEN WERT SETZEN
2184	E1	POP HL	; CHADD ORIGINAL IN
2185	22 5F 5C	LD (XPTR),HL	; XPTR SPEICHERN
2188	FD CB 01 FE	SET 7,(IY+1)	; PROGRAMMLAUF ANMERKEN
218C	CD B9 21	CALL \$21B9	; UND DIE ZUWEISUNG AUSFÜHREN
218F	2A 5F 5C	LD HL,(XPTR)	; CHADD WIEDER AUF DEN
2192	FD 36 26 00	LD (IY+\$26),0	; ORIGINALWERT SETZEN UND


```

2196 22 5D 5C      LD (CHADD),HL      ; XPTR LÖSCHEN
2199 18 17          JR $21B2
2198
2198 2A 63 5C      LD HL,(STKBOT)      ; LÄNGE DER 'LINE'
219E ED 5B 61 5C   LD DE,(WORKSP)      ; IM WORKSPACE
21A2 37            SCF
21A3 ED 52          SBC HL,DE          ; BERECHNEN UND
21A5 44            LD B,H
21A6 4D            LD C,L              ; NACH BC KOPIEREN
21A7 CD B2 2A       CALL $2AB2          ; PARAMETER AUF DEN STACK
21AA CD FF 2A       CALL $2AFF          ; ZUWEISUNG AUSFÜHREN
21AD 18 03          JR $21B2
21AF
21AF CD FC 1F       CALL $1FFC          ; PRINTANWEISUNGEN AUSFÜHREN
21B2 CD 4E 20       CALL $204E          ; SIND POSITIONIERZEICHEN VOR-
21B5                                     ; HANDEN (MEHREFE INPUTS/ZEILE)?
21B5 CA C1 20       JP Z,$20C1          ; JA, WEITER IN INPUTSCHLEIFE
21B8 C9            RET                  ; SONST INPUT FERTIG
21B9
21B9 ; SUBROUTINE FÜR INPUTZUWEISUNG
21B9 ; ERSTER AUFRUF MIT SYNTAXFLAG GESETZT UND ZWEITER
21B9 ; MIT SYNTAXFLAG ZURÜCKGESETZT (=PROGRAMMLAUF)
21B9
21B9 2A 61 5C      LD HL,(WORKSP)      ; CHADD AUF ERSTE POSITION
21BC 22 5D 5C      LD (CHADD),HL      ; IM WORKSPACE SETZEN
21BF DF            RST GETAKT          ; AKTUELLES ZEICHEN LADEN
21C0 FE E2          CP $E2             ; TOKEN FÜR 'STOP' ?
21C2 28 0C          JR Z,$21D0          ; JA
21C4 3A 71 5C      LD A,(FLAGX)        ; SONST WERTZUWEISUNG DER
21C7 CD 59 1C       CALL $1C59          ; VARIABLEN DURCHFÜHREN
21CA DF            RST GETAKT          ; IST AKTUELLES ZEICHEN
21CB FE 0D          CP $D              ; EIN CARRIAGE RETURN ?
21CD CB            RET Z                ; JA
21CE
21CE CF            RST ERRAUS           ; FEHLERMELDUNG:
21CF 0B            .BYT $0B             ; 'NONSENSE IN BASIC'
21D0
21D0 CD 30 25       CALL $2530          ; KEIN ERROR BEI
21D3 CB            RET Z                ; SYNTAXPRÜFUNG
21D4
21D4 ; ERTES ZEICHEN DES INPUT WAR STOP
21D4
21D4 CF            RST ERRAUS           ; MELDUNG:
21D5 10            .BYT $10             ; 'STOP IN INPUT'
21D6
21D6 ; SUBROUTINE ZUM PRÜFEN, OB KANAL K IM INPUT BENUTZT
21D6 ; WIRD. FALLS JA, IST DAS ZEROFLAG GESETZT
21D6
21D6 2A 51 5C      LD HL,(CURCHL)      ; STARTADRESSE DER
21D9 23            INC HL                ; KANALINFORMATION LADEN
21DA 23            INC HL                ; 4 ADDIEREN, UM
21DB 23            INC HL
21DC 23            INC HL
21DD 7E            LD A,(HL)            ; DEN KANAL ZU PRÜFEN
21DE FE 4B          CP 'K'              ; KANAL K ?
21E0 C9            RET
21E1
21E1 ; UNTERPROGRAMME ZUR BEHANDLUNG VON FARBANWEISUNGEN

```


21E1			
21E1	E7	RST GETNXT	; CHADD +1, NACHSTES ZEICHEN
21E2		; NORMALER EINSTIEG	
21E2	CD F2 21	FAREIN CALL FARBUN	; TEST AUF FARBANWEISUNGEN
21E5	D8	RET C	; KEINE FARBANWEISUNG
21E6	DF	RST GETAKT	; AKTUELLES ZEICHEN LADEN
21E7	FE 2C	CP ' ,	; WENN ES EIN TRENNUNGSZEICHEN,
21E9	28 F6	JR Z, \$21E1	; KOMMA ODER SEMIKOLON, IST,
21EB	FE 3B	CP ' ;	; WEITERE FARBANWEISUNGEN
21ED	28 F2	JR Z, \$21E1	; BEARBEITEN
21EF	C3 8A 1C	JP \$1C8A	; SONST ERROR
21F2			
21F2	FE D9	FARBUN CP \$D9	; WENN TOKEN NICHT IM BEREICH
21F4	D8	RET C	; VON \$D9 BIS \$DE (INK BIS OVER)
21F5	FE DF	CP \$DF	; IST, RETURN MIT CARRY GESETZT;
21F7	3F	CCF	; KEINE FARBANWEISUNG
21F8	D8	RET C	
21F9			
21F9	F5	PUSH AF	; SONST TOKEN RETTEN
21FA	E7	RST GETNXT	; CHADD +1
21FB	F1	POP AF	; TOKENS IN STEUERZEICHEN VON
21FC	D6 C9	SUB \$C9	; \$10 BIS \$15 UMWANDELN
21FE	F5	PUSH AF	; STEUERZEICHEN ZWISCHENSPEICHERN
21FF	CD 82 1C	CALL \$1C82	; PARAMETER AUF DEM CALC.-STACK
2202	F1	POP AF	; ABLEGEN UND STEUERZEICHEN
2203			; ZURÜCKHOLEN
2203	A7	AND A	
2204	CD C3 1F	CALL \$1FC3	; AUSSPRUNG, WENN NUR SYNTAX-
2207	F5	PUSH AF	; PRÜFUNG, SONST ZEICHEN RETTEN
2208	CD 94 1E	CALL \$1E94	; PARAMETER VOM CALC.-STACK
220B	57	LD D, A	; NACH REGISTER D BRINGEN
220C	F1	POP AF	; STEUERZEICHEN WIEDER HOLEN
220D	D7	RST PRTOU	; UND AUSGEBEN
220E	7A	LD A, D	; EBENFALLS DEN PARAMETER
220F	D7	RST PRTOU	; AUSGEBEN
2210	C9	RET	
2211			
2211		; DIE FOLGENDEN UNTERPROGRAMME WERDEN VON DER PRINT-	
2211		; ROUTINE AUFGERUFEN. DABEI WERDEN DIE SYSTEMVARIABLEN	
2211		; ATTR, MASKT UND PFLAG ENTSPRECHEND DEN ANWEISUNGEN	
2211		; GEANDERT (NUR DIE TEMPORAREN). REG A ENTHALT DAS	
2211		; STEUERZEICHEN UND REG D DEN PARAMETER	
2211			
2211	D6 11	SUB \$11	; SUBTRAKTION UND ADDITION
2213	CE 00	ADC 0	; DIENEN ZUR BESTIMMUNG VON INK
2215	28 1D	JR Z, \$2234	; UND PAPER: SPRUNG
2217			
2217	D6 02	SUB 2	; EBENFALLS AUF DIESE ART
2219	CE 00	ADC 0	; BRIGHT UND FLASH BESTIMMEN
221B	28 56	JR Z, \$2273	; SPRUNG BEI DIESEN
221D			
221D	FE 01	CP 1	; STEUERZEICHEN = OVER ?
221F	7A	LD A, D	; PARAMETER NACH A
2220	06 01	LD B, 1	; MASKE FÜR OVER SETZEN
2222	20 04	JR NZ, \$2228	; SPRUNG BEI OVER
2224			
2224	07	RLCA	; HIER INVERSE-BEARBEITUNG;
2225	07	RLCA	; BIT2=0 = INVERSE0, SONST

2226			; INVERSE1
2226	06 04	LD B,4	; MASKE ZUM AUSBLENDEN
2228	4F	LD C,A	; WERT RETTEN
2229	7A	LD A,D	; DEN PARAMETER TESTEN:
222A	FE 02	CP 2	; NUR 0 UND 1 ZUGELASSEN
222C	30 16	JR NC,\$2244	; FALSCHER WERT: ERROR
222E	79	LD A,C	; WERT ZURÜCK IN A
222F	21 91 5C	LD HL,PFLAG	; ADRESSE VON PFLAG
2232	18 38	JR \$226C	; ZUM ANDERN LADEN
2234			
2234		; BEHANDLUNG VON INK UND PAPER	
2234			
2234	7A	LD A,D	; PARAMETER NACH A
2235	06 07	LD B,7	; B MIT MASKE FÜR INK LADEN
2237	38 05	JR C,\$223E	; SPRUNG BEI INK
2239	07	RLCA	; SONST PARAMETER 3 STELLEN
223A	07	RLCA	; NACH LINKS SCHIEBEN FÜR PAPER
223B	07	RLCA	; (MULTIPLIKATION MIT 8)
223C	06 38	LD B,\$38	; MASKE FÜR PAPER IN B LADEN
223E	4F	LD C,A	; WERT ZWISCHENSPEICHERN
223F	7A	LD A,D	; PARAMETER AUF
2240	FE 0A	CP 10	; AUF 0 -9 TESTEN
2242	38 02	JR C,\$2246	; PARAMETER IN ORDNUNG
2244			
2244	CF	RST ERR AUS	; FEHLERMELDUNG:
2245	13	.BYT \$13	; 'INVALID COLOUR'
2246			
2246	21 8F 5C	LD HL,ATTRT	; ADRESSE VON ATTRT ZUM ANDERN
2249			; VON ATTRT, MASKT, PFLAG LADEN
2249	FE 08	CP 8	; SPRUNG BEI PAPER UND INK
224B	38 08	JR C,\$2258	; MIT PARAMETER VON 0 - 7
224D	7E	LD A,(HL)	; ATTRT LADEN
224E	28 07	JR Z,\$2257	; BEI PAPER UND INK = 8 WIRD
2250			; NICHTS GEÄNDERT
2250	B0	OR B	; PAPER/INK=9: PAPER/INK-FARBEN
2251	2F	CPL	; MÜSSEN SCHWARZ UND WEISS
2252	E6 24	AND \$24	
2254	28 01	JR Z,\$2257	; SPRUNG BEI PAPER/INK = SCHWARZ
2256	78	LD A,B	; PAPER/INK = WEISS
2257			
2257	4F	LD C,A	; WERT ZWISCHENSPEICHERN
2258			
2258	79	LD A,C	; WERT ZURÜCKLADEN
2259	CD 6C 22	CALL \$226C	; ATTRT WIRD GEÄNDERT
225C			
225C		; JETZT WIRD MASKT BEARBEITET	
225C			
225C	3E 07	LD A,7	; MASKT WIRD NUR BEI PAPER/INK
225E	BA	CP D	; = 8 ODER 9 GEÄNDERT:
225F	9F	SBC A	; A ENTHÄLT DANN \$FF, SONST 0
2260	CD 6C 22	CALL \$226C	; MIT B ALS MASKE MASKT ANDERN
2263			
2263		; ZULETZT NOCH PFLAG BEARBEITEN	
2263			
2263	07	RLCA	; DIE MASKE FÜR
2264	07	RLCA	; PFLAG WIRD GEBILDET
2265	E6 50	AND \$50	; NUR BITS 6 UND 4
2267	47	LD B,A	; NACH B BRINGEN

```

2268 3E 08          LD A,8          ; PFLAG WIRD NUR VERANDERT,
226A 8A             CP D            ; WENN FAER/INY = 9 IST;
226B 9F             SBC A           ; REG A DANN $FF, SONST 0
226C               ;
226C               ; SUBROUTINE ZUM SETZEN DER FARBDDETAILS. REG HL ENTHALT
226C               ; DIE ADRESSE, REG B DIE MASKE UND REG A DEN NEUEN WERT
226C               ;
226C AE             XOR (HL)         ; DURCH DIE DOPPELTE EXOR-VER-
226D 40             AND B           ; KNÜPFUNG IN VERBINDUNG MIT DEM
226E AE             XOR (HL)         ; 'LOG. UND B' WERDEN NUR DIE
226F 77             LD (HL),A        ; GEWÜNSCHTEN BITS GEANDERT
2270 23             INC HL           ; POINTER AUF NÄCHSTE FARB-
2271 78             LD A,B           ; VARIABLE SETZEN UND MASKE IN
2272 C9             RET              ; A BRINGEN
2273               ;
2273               ; BEHANDLUNG VON FLASH UND BRIGHT
2273               ;
2273 9F             SBC A             ; ZEROFLAG FÜR BRIGHT SETZEN
2274 7A             LD A,D           ; PARAMETER NACH A UND NACH
2275 0F             RRCA             ; RECHTS ROTIEREN (BIT7=BIT0ALT)
2276 06 80          LD B,$80         ; MASKE FÜR FLASH NACH B LADEN
2278 20 03          JR NZ,$227D      ; SPRUNG BEI FLASH
227A 0F             RRCA             ; FÜR BRIGHT NOCH MAL ROTIEREN
227B 06 40          LD B,$40         ; MASKE FÜR BRIGHT
227D 4F             LD C,A           ; WERT ZWISCHENSPEICHERN
227E 7A             LD A,D           ; PARAMETER NOCH MAL HOLEN
227F FE 08          CP B             ; NUR DIE WERTE 0,1 UND 8
2281 28 04          JR Z,$2287       ; SIND ZUGELASSEN
2283 FE 02          CP 2             ; SONST ERROR
2285 30 8D          JR NC,$2244
2287               ;
2287 79             LD A,C           ; WERT WIEDER IN A
2288 21 8F 5C       LD HL,ATTRT      ; UND ADRESSE ATTRT ZUM
228B CD 6C 22       CALL $226C       ; ÄNDERN LADEN
228E 79             LD A,C           ; WERT WIEDER NACH A
228F 0F             RRCA             ; DAS 'SETZBIT' (3) WIRD NACH
2290 0F             RRCA             ; BIT 7 (FLASH) BZW. 6 (BRIGHT)
2291 0F             RRCA             ; GEBRACHT UND MASKT
2292 1B DB          JR $226C         ; ENTSPRECHEND VERÄNDERT
2294               ;
2294               ; BEFEHL BORDER
2294               ; DER PARAMETER VON BORDER WIRD ÜBER EINEN 'OUT'-BEFEHL
2294               ; AUSGEGEBEN. ANSCHLIESSEND WIRD DER PARAMETER IN BORDCR
2294               ; GESPEICHERT
2294               ;
2294 CD 94 1E       CALL $1E94        ; PARAMETER HOLEN
2297 FE 08          CP B             ; NUR 0 - 7 ZUGELASSEN
2299 30 A9          JR NC,$2244      ; SONST ERROR
229B D3 FE          OUT ($FE),A      ; PARAMETER AUSGEBEN
229D 07             RLCA             ; DIESEN UM 3 BIT NACH
229E 07             RLCA             ; LINKS SCHIEBEN
229F 07             RLCA
22A0 CB 6F         BIT 3,A           ; BORDERFARBE = HELL ?
22A2 20 02          JR NZ,$22A6      ; JA: EDITORBEREICH = SCHWARZ
22A4 EE 07          XOR 7            ; SONST WEISS
22A6 32 4B 5C       LD (BORDCR),A   ; NEUE BORDERFARBE
22A9 C9             RET
22AA               ;

```

22AA		; SUBROUTINE ZUM BERECHNEN EINER 'PIXEL' (PUNKT)-ADRESSE	2
22AA		; AUF DEM BILDSCHIRM	2
22AA		; AUFRUF VON POINT UND PLOT MIT ADRESSE DES PUNKTES IN	2
22AA		; BC. BEI RETURN ENTHÄLT HL DIE ADRESSE DES BYTES IM	2
22AA		; ENTSPRECHENDEN BILDSCHIRMBEREICH UND A DIE BITPOSITION	2
22AA		; (DES PUNKTES) IN DIESEM BYTE	2
22AA		;	2
22AA	3E AF	PKTADR LD A,\$AF ; DIE Y-KOORDINATE (IN B) DARF	2
22AC	90	SUB B ; NICHT GRÖßER ALS 175 SEIN	2
22AD	DA F9 24	JP C,\$24F9 ; SONST ERROR	2
22B0	47	LD B,A ; B = 175 - Y-KOORDINATE	2
22B1		; (SUBTRAKTION: ES WIRD VON	2
22B1		; UNTEN GEZÄHLT, 175 = 22 BILD-	2
22B1		; SCHIRMZEILEN * 8 LINIEN -1)	2
22B1	A7	AND A ; D7 BIS B0 SIND DIE BITS VON B,	2
22B2	1F	RRA ; C7 BIS C0 VON C	2
22B3	37	SCF	2
22B4	1F	RRA	2
22B5	A7	AND A	2
22B6	1F	RRA	2
22B7	AB	XOR B ; DIE MANIPULATIONEN ERGEBEN	2
22B8	E6 FB	AND \$FB ; FOLGENDES BIT-ERGEBNIS	2
22BA	AB	XOR B ; (=HIGHADRESSE) IN H:	2
22BB	67	LD H,A ; 0,1,0,B7,B6,B2,B1,B0	2
22BC	79	LD A,C ; MIT DER X-KOORDINATE IN C	2
22BD	07	RLCA ; WIRD ÄHNLICH VERFAHREN,	2
22BE	07	RLCA ; SODASS SICH IM L-REGISTER	2
22BF	07	RLCA ; (=LOWADRESSE) FOLGENDES	2
22C0	AB	XOR B ; BITMUSTER ERGIBT:	2
22C1	E6 C7	AND \$C7 ; B5,B4,B3,C7,C6,C5,C4,C3	2
22C3	AB	XOR B	2
22C4	07	RLCA	2
22C5	07	RLCA	2
22C6	6F	LD L,A	2
22C7	79	LD A,C ; IN REG A WIRD DIE BITADRESSE	2
22C8	E6 07	AND 7 ; INNERHALB DIESES BYTES	2
22CA	C9	RET ; GEBILDET	2
22CB		;	2
22CB		; SUBROUTINE FÜR DIE POINT-FUNKTION	2
22CB		;	2
22CB	CD 07 23	CALL \$2307 ; Y-KOORDINATE NACH B, X NACH C	2
22CE	CD AA 22	CALL PKTADR ; DAZUGEHÖRIGE PUNKTADRESSE	2
22D1		; BESTIMMEN	2
22D1	47	LD B,A ; B DIENT ALS BITZÄHLER	2
22D2	04	INC B	2
22D3	7E	LD A,(HL) ; DAS ENTSPRECHENDE BYTE LADEN	2
22D4	07	RLCA ; UND SOLANGE ROTIEREN, BIS DAS	2
22D5	10 FD	DJNZ \$22D4 ; GESUCHTE BIT GLEICH BIT0	2
22D7	E6 01	AND 1 ; IN REG A IST UND DIESES	2
22D9	C3 28 2D	JP \$2D28 ; BIT IM CALC.-STACK SPEICHERN	2
22DC		; DAS BIT IST 1 FÜR INK UND	2
22DC		; 0 FÜR PAPER	2
22DC		;	2
22DC		; BEFEHL PLOT	2
22DC		; BEIM EINSTIEG LIEGEN DIE KOORDINATEN AUF DEM CALC.-	2
22DC		; STACK. UNTER BERÜCKSICHTIGUNG VON 'INVERSE' UND 'OVER'	2
22DC		; (IN PFLAG) WIRD DER PUNKT ENTSPRECHEND GESETZT	2
22DC		;	2

```

2DC CD 07 23 PLOTHA CALL $2307 ; Y-KOORDINATE NACH B, X NACH C
2DF CD E5 22 CALL PLOTTE ; PLOTSUBROUTINE
2E2 C3 4D 0D JP ARCOL ; TEMPORARE FARBEN SETZEN
2E5 ;
2E5 ED 43 7D 5C PLOTTE LD (COORDS),BC ; SYSTEMVARIABLE COORDS SETZEN
2E9 CD AA 22 CALL PKTADR ; DIE PUNKTADRESSE BESTIMMEN
2EC 47 LD B,A ; BITNUMMER NACH B
2ED 04 INC B ; +1 ZUM RICHTIGEN ZAHLEN
2EE 3E FE LD A,$FE ; NULLMASKE IN A WIRD SOLANGE
2F0 0F RRCA ; ROTIERT, BIS DIE RICHTIGE
2F1 10 FD DJNZ $22F0 ; BITPOSITION GEFUNDEN IST
2F3 47 LD B,A ; NULLBITPOSITION NACH B
2F4 7E LD A,(HL) ; DAS BYTE, IN DEM EIN PUNKT
2F5 ; GESETZT WERDEN SOLL, LADEN
2F5 FD 4E 57 LD C,(IY+$57) ; PFLAGS: TEST AUF
2F8 CB 41 BIT 0,C ; 'OVER1'
2FA 20 01 JR NZ,$22FD ; JA
2FC A0 AND B ; NEIN BIT NULL SETZEN
2FD CB 51 BIT 2,C ; 'INVERSE1'
2FF 20 02 JR NZ,$2303 ; JA
2301 AB XOR B ; DAS ERGENIS IST:
2302 2F CPL ; INV.1 UND OVER1: BIT BLEIBT
2303 ; INV.1 UND OVER0: BIT = 0
2303 ; INV.0 UND OVER1: BIT NEGIEREN
2303 ; INV.0 UND OVER0: BIT = 1
2303 77 LD (HL),A ; ERGEBNIS SPEICHERN
2304 C3 DB 0B JP $BDB ; DAS ATTRIBUTBYTE NOCH SETZEN
2307 ;
2307 ; SUBROUTINE, UM BC MIT DEN LETZTEN ZWEI WERTEN, IM
2307 ; BEREICH JEWEILS VON $00 BIS $FF, DES CALC.-STACK ZU
2307 ; LADEN. D UND E ENTHALTEN JEWEILS +/-1 FÜR DIE
2307 ; ZEICHENRICHTUNG
2307 ;
2307 CD 14 23 CALL STAINA ; LETZTEN WERT DES CALC.-STACK
230A 47 LD B,A ; HOLEN UND NACH BC BRINGEN
230B C5 PUSH BC ; ZWISCHENSPEICHERN
230C CD 14 23 CALL STAINA ; ZWEITEN WERT VOM CALC.-STACK
230F 59 LD E,C ; +/- 1 ZWEITER WERT NACH E
2310 C1 POP BC ; ERSTEN WERT ZURÜCKHOLEN
2311 51 LD D,C ; +/- 1 ERSTER WERT NACH D
2312 4F LD C,A ; ZWEITER WERT NACH C
2313 C9 RET
2314 ;
2314 ; LETZTE FLOATINGPOINT-ZAHL VOM CALC.-STACK NACH A
2314 ; LADEN, BEREICH $00 BIS $FF. C ENTHÄLT +1 FÜR POSITIVE
2314 ; UND -1 FÜR NEGATIVE WERTE
2314 ;
2314 CD D5 2D STAINA CALL $2DD5 ; FLOATINGPOINT-ZAHL HOLEN
2317 DA F9 24 JP C,$24F9 ; ERROR, WENN ZU GROSS
231A 0E 01 LD C,1 ; C=1 FÜR POSITIV
231C C8 RET Z ; RETURN POSITIV
231D 0E FF LD C,$FF ; SONST C=-1 FÜR NEGATIV
231F C9 RET
2320 ;
2320 ; BEFEHL CIRCLE
2320 ; ES WIRD EIN ANGENÄHERTER KREIS MIT RADIUS 'R' UM DIE
2320 ; KOORDINATEN X,Y (=MITTE) GEZEICHNET. ALLE DREI WERTE
2320 ; WERDEN ZUERST GERUNDET

```


2320		;		
2320	DF	KREIS	RST GETAKT	; AKTUELLES ZEICHEN LADEN
2321	FE 2C		CP ,	; WENN ES KEIN KOMMA IST,
2323	C2 BA 1C		JP NZ,\$1C8A	; ERROR
2326	E7		RST GETNXT	; CHADD +1, NACHSTES ZEICHEN
2327	CD 82 1C		CALL \$1C82	; RADIUS IN DEN CALC.-STACK HOLEN
232A	CD EE 1B		CALL \$1BEE	; WENN SYNTAXPRÜFUNG, AUSSPRUNG
232D	EF		RST CALRUF	; CALCULATOR AUFRUFEN
232E				; CALC.-STACK ENTHALT: X,Y,R
232E	2A		.BYT \$2A	; ABS BILDEN
232F	3D		.BYT \$3D	; R WIEDER AUF STACK
2330	3B		.BYT \$3B	; ENDE
2331	7E		LD A,(HL)	; EXPONENT DES RADIUS LADEN
2332	FE 81		CP \$81	; RADIUS KLEINER 1 ?
2334	30 05		JR NC,\$233B	; NEIN
2336	EF		RST CALRUF	; FALLS KLEINER, IM
2337	02		.BYT \$02	; CALC.-STACK LÖSCHEN: X,Y
2338	3B		.BYT \$3B	; ENDE
2339	1B A1		JR PLOTHA	; NUR DIE MITTE PLOTTEN
233B		;		
233B	EF		RST CALRUF	
233C	A3		.BYT \$A3	; PI/2 AUF STACK: X,Y,R,PI/2
233D	3B		.BYT \$3B	; CALC-ENDE UND EXPONENT
233E	36 83		LD (HL),\$83	; VON PI/2 AUF 2*PI ÄNDERN
2340	EF		RST CALRUF	; X,Y,R,2*PI
2341	C5		.BYT \$C5	; 2*PI NACH MEM5
2342	02		.BYT \$02	; 2*PI LÖSCHEN: X,Y,R
2343	3B		.BYT \$3B	; CALC-ENDE
2344	CD 7D 24		CALL \$247D	; STARTPARAMETER INITIALISIEREN
2347		;		
2347		;	DER KREIS WIRD AUF GERADENSTÜCKE ZURÜCK-	
2347		;	GEFÜHRT, DIE MIT DER 'DRAW'-SUBROUTINE	
2347		;	GEZEICHNET WERDEN. DIE ANZAHL DER GERADEN-	
2347		;	STÜCKE IST A UND IN BC ENTHALTEN	
2347		;		
2347	C5		PUSH BC	; GERADENZÄHLER RETTEN
2348	EF		RST CALRUF	; X,Y,R
2349	31		.BYT \$31	; X,Y,R,R
234A	E1		.BYT \$E1	; X,Y,R,R,SIN(PI/A)
234B	04		.BYT \$04	; X,Y,R,R*SIN(PI/A)
234C	3B		.BYT \$3B	; ENDE
234D	7E		LD A,(HL)	; TEST, OB 1. TEILSTÜCK
234E	FE 80		CP \$80	; 1.5 IST
2350	30 0B		JR NC,\$235A	; NEIN
2352	EF		RST CALRUF	; SONST BIS AUF X,Y
2353	02		.BYT 02	; ALLES LÖSCHEN UND NUR
2354	02		.BYT 02	; DEN MITTELPUNKT PLOTTEN
2355	3B		.BYT \$3B	
2356	C1		POP BC	; STACKKORREKTUR
2357	C3 DC 22		JP PLOTHA	; PUNKT PLOTTEN
235A		;		
235A	EF		RST CALRUF	; X,Y,R,R*SIN(PI/A)
235B	C2		.BYT \$C2	; R*SIN(PI/A) NACH MEM2
235C	01		.BYT \$01	; X,Y,R*SIN(PI/A),R
235D	C0		.BYT \$C0	; R NACH MEM0 SPEICHERN
235E	02		.BYT \$02	; X,Y,R*SIN(PI/A)
235F	03		.BYT \$03	; X,Y-R*SIN(PI/A)
2360	01		.BYT \$01	; Y-R*SIN(PI/A),X

2361	E0	.BYT \$E0	; Y-R*SIN(PI/A),Y,R
2362	0F	.BYT \$0F	; Y-R*SIN(PI/A),Y+R
2363	C0	.BYT \$C0	; Y+R NACH MEMO SPEICHERN
2364	01	.BYT \$01	; I=Y+R,J=Y-R*SIN(PI/A)
2365	31	.BYT \$31	; I,J,J
2366	E0	.BYT \$E0	; I,J,J,I
2367	01	.BYT \$01	; I,J,I,J
2368	31	.BYT \$31	; I,J,I,J,J
2369	E0	.BYT \$E0	; I,J,I,J,J,I
236A	A0	.BYT \$A0	; I,J,I,J,J,I,0
236B	C1	.BYT \$C1	; MEM1 LÖSCHEN
236C	02	.BYT \$02	; I,J,I,J,J,I
236D	3B	.BYT \$3B	; ENDE
236E			
236E	FD 34 62	INC (IY+\$62)	; EXP. MEM2 +1
2371	CD 94 1E	CALL \$1E94	; Y+R NACH A LADEN
2374	6F	LD L,A	; IN HL KOPIEREN
2375	E5	PUSH HL	; ZWISCHENSPEICHERN
2376	CD 94 1E	CALL \$1E94	; Y-R*SIN(PI/A) HOLEN
2379	E1	POP HL	; UND NACH
237A	67	LD H,A	; H KOPIEREN
237B	22 7D 5C	LD (COORDS),HL	; COORDS DAMIT SETZEN
237E	C1	POP BC	; GERADENZÄHLER HOLEN
237F	C3 20 24	JP \$2420	; UND DIE GERADENSTÜCKE
2382			; ZEICHNEN
2382			
2382			; BEFEHL DRAW
2382			; DIE STARTKOORDINATEN (X0,Y0) EINER GERADEN
2382			; SIND IN COORDS ENTHALTEN. WENN AUSSER DEN
2382			; ENDPUNKTEN X UND Y KEINE WEITEREN PARAMETER
2382			; ANGEZEIGT SIND, WIRD EINE GERADE VON Y0,Y0
2382			; NACH X+X0,Y+Y0 GEZEICHNET.
2382			
2382	DF	RST GETAKT	; AKTUELLES ZEICHEN LADEN
2383	FE 2C	CP ,	; UND AUF KOMMA VERGLEICHEN
2385	2B 06	JR Z,\$238D	; JA: 3. PARAMETER FOLGT
2387	CD EE 1B	CALL \$1BEE	; ZUM NÄCHSTEN BEFEHL BEI
238A			; DER SYNTAXPRÜFUNG
238A	C3 77 24	JP \$2477	; SONST DIE GERADE ZEICHNEN
238D			
238D	E7	RST GETNXT	; NÄCHSTES ZEICHEN HOLEN
238E			; (DIES IST DER WINKEL)
238E	CD 82 1C	CALL \$1C82	; WINKEL AUF DEN CALC.-STACK
2391	CD EE 1B	CALL \$1BEE	; ZUM NÄCHSTEN BEFEHL WÄHREND
2394			; DER SYNTAXPRÜFUNG
2394	EF	RST CALRUF	; X,Y,G, G=WINKEL
2395	C5	.BYT \$C5	; G NACH MEM5 SPEICHERN
2396	A2	.BYT \$A2	; X,Y,G,.5
2397	04	.BYT \$04	; X,Y,G/2
2398	1F	.BYT \$1F	; X,Y,SIN(G/2)
2399	31	.BYT \$31	; X,Y,SIN(G/2),SIN(G/2)
239A	30	.BYT \$30	; X,Y,SIN(G/2),(1 OD.0), G=0 ?
239B	30	.BYT \$30	; X,Y,SIN(G/2),(0 OD.1), INV.
239C	00	.BYT \$00	; SPRUNG, WENN WINKEL
239D	06	.BYT \$06	; <>0 NACH ZEWIN
239E	02	.BYT \$02	; X,Y, WINKEL WAR 0, Z. B.
239F	3B	.BYT \$3B	; 2*N*PI
23A0	C3 77 24	JP \$2477	; GERADE ZEICHNEN

3A3				231
3A3	C0	ZEIWIN .BYT \$C0	; SIN(G/2) NACH MEMO	231
3A4	02	.BYT \$02	; X,Y	231
3A5	C1	.BYT \$C1	; Y NACH MEMO SPEICHERN	231
3A6	02	.BYT \$02	; X	231
3A7	31	.BYT \$31	; X,X	231
3A8	2A	.BYT \$2A	; X,X'=ABS(X)	231
3A9	E1	.BYT \$E1	; X,X',Y	231
3AA	01	.BYT \$01	; X,Y,X'	231
3AB	E1	.BYT \$E1	; X,Y,X',Y	231
3AC	2A	.BYT \$2A	; X,Y,X',Y'=ABS(Y)	231
3AD	0F	.BYT \$0F	; X,Y,X'+Y'	231
3AE	E0	.BYT \$E0	; X,Y,X'+Y',SIN(G/2)	231
3AF	05	.BYT \$05	; X,Y,Z'=(X'+Y')/SIN(G/2)	231
3B0	2A	.BYT \$2A	; X,Y,Z=ABS(Z')	231
3B1	E0	.BYT \$E0	; X,Y,Z,SIN(G/2)	231
3B2	01	.BYT \$01	; X,Y,SIN(G/2),Z	231
3B3	3D	.BYT \$3D	; Z IN FLOATINGP.-FORM	231
3B4	38	.BYT \$38	; X,Y,SIN(G/2),Z	231
3B5	7E	LD A,(HL)	; EXPONENT VON Z LADEN	231
3B6	FE 81	CP \$81	; Z >= 1 ?	231
3B8	30 07	JR NC,\$23C1	; JA	231
3BA	EF	RST CALRUF	; X,Y,SIN(G/2),Z	231
3BB	02	.BYT \$02	; SIN(G/2) UND Z	231
3BC	02	.BYT \$02	; LÖSCHEN	231
3BD	38	.BYT \$38	; X,Y	231
3BE	C3 77 24	JP \$2477	; GERADE ZEICHNEN	231
3C1				231
3C1	CD 7D 24	CALL \$247D	; PARAMETER INITIALISIEREN	231
3C4	C5	PUSH BC	; GERADENZÄHLER RETTEN	231
3C5	EF	RST CALRUF	; X,Y,SIN(G/2),Z	231
3C6	02	.BYT \$02	; X,Y,SIN(G/2)	231
3C7	E1	.BYT \$E1	; X,Y,SIN(G/2),SIN(G/2*A)	231
3C8	01	.BYT \$01	; X,Y,SIN(G/2*A),SIN(G/2)	231
3C9	05	.BYT \$05	; X,Y,SIN(G/2*A)/SIN(G/2)=W	231
3CA	C1	.BYT \$C1	; W IN MEM1 SPEICHERN	231
3CB	02	.BYT \$02	; X,Y	231
3CC	01	.BYT \$01	; Y,X	231
3CD	31	.BYT \$31	; Y,X,X	231
3CE	E1	.BYT \$E1	; Y,X,Y,W	231
3CF	04	.BYT \$04	; Y,X,X*W	231
3D0	C2	.BYT \$C2	; X*W IN MEM2 SPEICHERN	231
3D1	02	.BYT \$02	; Y,X	231
3D2	01	.BYT \$01	; X,Y	231
3D3	31	.BYT \$31	; X,Y,Y	231
3D4	E1	.BYT \$E1	; X,Y,Y,W	231
3D5	04	.BYT \$04	; X,Y,Y*W	231
3D6	E2	.BYT \$E2	; X,Y,Y*W,X*W	231
3D7	E5	.BYT \$E5	; X,Y,Y*W,X*W,B	231
3D8	E0	.BYT \$E0	; X,Y,Y*W,X*W,B,G/A	231
3D9	03	.BYT \$03	; X,Y,Y*W,X*W,B-G/A	231
3DA	A2	.BYT \$A2	; X,Y,Y*W,X*W,B-G/A,.5	231
3DB	04	.BYT \$04	; X,Y,Y*W,X*W,(B-G/A)/2=F	231
3DC	31	.BYT \$31	; X,Y,Y*W,X*W,F,F	231
3DD	1F	.BYT \$1F	; X,Y,Y*W,X*W,F,SIN(F)	231
3DE	C5	.BYT \$C5	; SIN(F) IN MEM5 SPEICHERN	231
3DF	02	.BYT \$02	; X,Y,Y*W,X*W,F	231
3E0	20	.BYT \$20	; X,Y,Y*W,X*W,COS(F)	231

E1	C0	.BYT \$C0	; COS(F) IN MEMO SPEICHERN
E2	02	.BYT \$02	; X,Y,Y*W,Y*W
E3	C2	.BYT \$C2	; X*W IN MEM2 SPEICHERN
E4	02	.BYT \$02	; X,Y,Y*W
E5	C1	.BYT \$C1	; Y*W IN MEM1 SPEICHERN
E6	E5	.BYT \$E5	; X,Y,Y*W,SIN(F)
E7	04	.BYT \$04	; X,Y,Y*W*SIN(F)
E8	E0	.BYT \$E0	; X,Y,Y*W*SIN(F),X*W
E9	E2	.BYT \$E2	; X,Y,Y*W*SIN(F),X*W,COS(F)
EA	04	.BYT \$04	; X,Y,Y*W*SIN(F),X*W,COS(F)
EB	0F	.BYT \$0F	; X,Y,Y*W*SIN(F)+X*W*COS(F)=U
EC	E1	.BYT \$E1	; X,Y,U,Y*W
ED	01	.BYT \$01	; X,Y,Y*W,U
EE	C1	.BYT \$C1	; U IN MEM1 SPEICHERN
EF	02	.BYT \$02	; X,Y,Y*W
F0	E0	.BYT \$E0	; X,Y,Y*W,COS(F)
F1	04	.BYT \$04	; X,Y,Y*W*COS(F)
F2	E2	.BYT \$E2	; X,Y,Y*W*COS(F),X*W
F3	E5	.BYT \$E5	; X,Y,Y*W*COS(F),X*W,SIN(F)
F4	04	.BYT \$04	; X,Y,Y*W*COS(F),X*W*SIN(F)
F5	03	.BYT \$03	; X,Y,Y*W*COS(F)-X*W*SIN(F)=V
F6	C2	.BYT \$C2	; V NACH MEM2 SPEICHERN
F7	2A	.BYT \$2A	; X,Y,V'=ABS(V)
F8	E1	.BYT \$E1	; X,Y,V',U
F9	2A	.BYT \$2A	; X,Y,V',U'=ABS(U)
FA	0F	.BYT \$0F	; X,Y,V'+U'
FB	02	.BYT \$02	; X,Y, DE ZEIGT AUF
FC	3B	.BYT \$3B	; EXPONENT VON V'+U'
FD	1A	LD A,(DE)	; EXPONENT LADEN
FE	FE B1	CP \$B1	; V'+U' < 1 ?
00	C1	POP BC	; (GERADENZÄHLER HOLEN)
01	DA 77 24	JP C,\$2477	; JA: NUR GERADE ZEICHNEN
04	C5	PUSH BC	; GERADENZÄHLER RETTEN
05	EF	RST CALRUF	; X,Y
06	01	.BYT \$01	; Y,X
07	3B	.BYT \$3B	; ENDE
08	3A 7D 5C	LD A,(COORDS)	; X0 LADEN UND IM
09	CD 2B 2D	CALL \$2D2B	; CALC.-STACK SPEICHERN
0E	EF	RST CALRUF	; Y,X,X0
0F	C0	.BYT \$C0	; X0 IN MEMO SPEICHERN
10	0F	.BYT \$0F	; Y,X+X0
11	01	.BYT \$01	; X+X0,Y
12	3B	.BYT \$3B	; ENDE
13	3A 7E 5C	LD A,(COORDS+1)	; Y0 LADEN UND IM
16	CD 2B 2D	CALL \$2D2B	; CALC.-STACK SPEICHERN
19	EF	RST CALRUF	; X+X0,Y,Y0
1A	C5	.BYT \$C5	; Y0 IN MEM5 SPEICHERN
1B	0F	.BYT \$0F	; X+X0,Y+Y0
1C	E0	.BYT \$E0	; X+X0,Y+Y0,X0
1D	E5	.BYT \$E5	; X+X0,Y+Y0,X0,Y0
1E	3B	.BYT \$3B	; ENDE
1F	C1	POP BC	; GERADENZÄHLER HOLEN
20		.END	
20		.LIB SPEC2400-S	
20		; SINCLAIR ZX SPECTRUM TEIL 2400	
20		;	
20		; IM FOLGENDEN WERDEN DIE GERADENTEILE GEZEICHNET	
20		; AUF DEM CALC.-STACK LIEGEN DIE WERTE:	

```

2420      ; XO+X,YO+Y,Xn,Yn
2420      ; ALS ZWISCHENWERTE WERDEN BENUTZT:
2420      ; Un=Xn+1-Xn, Vn=Yn+1-Yn
2420      ;
2420 05      DEC B      ; TEILSTÜCKE FERTIG ?
2421 28 3C      JR Z,$245F      ; JA: LETZTES ZEICHNEN
2423 18 14      JR $2439
2425 EF      GERADS RST CALRUF
2426 E1      .BYT $E1      ; Un-1
2427 31      .BYT $31      ; Un-1,Un-1
2428 E3      .BYT $E3      ; Un-1,Un-1,COS(G/A)
2429 04      .BYT $04      ; Un-1,Un-1*COS(G/A)
242A E2      .BYT $E2      ; Un-1,Un-1*COS(G/A),Vn-1
242B E4      .BYT $E4      ; Un-1,Un-1*COS(G/A),Vn-1,SIN(G/A)
242C 04      .BYT $04      ; Un-1,Un-1*COS(G/A),Vn-1*SIN(G/A)
242D 03      .BYT $03      ; Un-1,Un-1*COS(G/A)-Vn-1*SIN(G/A)
242E C1      .BYT $C1      ; Un=Un-1*COS... NACH MEM1
242F 02      .BYT $02      ; Un-1
2430 E4      .BYT $E4      ; Un-1,SIN(G/A)
2431 04      .BYT $04      ; Un-1*SIN(G/A)
2432 E2      .BYT $E2      ; Un-1*SIN(G/A),Vn-1
2433 E3      .BYT $E3      ; Un-1*SIN(G/A),Vn-1,COS(G/A)
2434 04      .BYT $04      ; Un-1*SIN(G/A),Vn-1*COS(G/A)
2435 0F      .BYT $0F      ; Vn=Un-1*SIN(G/A)+Cn-1*COS(G/A)
2436 C2      .BYT $C2      ; Vn NACH MEM2 SPEICHERN
2437 02      .BYT $02      ; -, (XO+X... SIND NOCH AUF
2438 38      .BYT $38      ; DEM CALC.-STACK)
2439 C5      PUSH BC      ; GERADENZÄHLER RETTEN
243A EF      RST CALRUF      ; XO+X,YO+Y,Xn,Yn
243B C0      .BYT $C0      ; Yn NACH MEMO SPEICHERN
243C 02      .BYT $02      ; XO+X,YO+Y,Xn
243D E1      .BYT $E1      ; XO+X,YO+Y,Xn,Un
243E 0F      .BYT $0F      ; XO+X,YO+Y,Xn+Un=Xn+1
243F 31      .BYT $31      ; XO+X,YO+Y,Xn+1,Xn+1
2440 38      .BYT $38      ; ENDE
2441 3A 7D 5C      LD A,(COORDS)      ; DAS BEREITS ERREICHTE Xn'
2444      ; NACH REG A LADEN
2444 CD 28 2D      CALL $2D28      ; DIESES AUF DEN STACK LEGEN
2447 EF      RST CALRUF      ; XO+X,YO+Y,Xn+1,Xn+1,Xn'
2448 03      .BYT $03      ; XO+X,YO+Y,Xn+1,Xn+1-Xn'=Un'
2449 E0      .BYT $E0      ; XO+X,YO+Y,Xn+1,Un',Yn
244A E2      .BYT $E2      ; XO+X,YO+Y,Xn+1,Un',Yn,Vn
244B 0F      .BYT $0F      ; ...,Un',Yn+Vn=Yn+1
244C C0      .BYT $C0      ; Yn+1 NACH MEMO SPEICHERN
244D 01      .BYT $01      ; ...,Un'
244E E0      .BYT $E0      ; ...,Un',Yn+1
244F 38      .BYT $38      ; ENDE
2450 3A 7E 5C      LD A,(COORDS+1)      ; Yn' (WIE Xn') LADEN UND
2453 CD 28 2D      CALL $2D28      ; AUF DEN CALC.-STACK BRINGEN
2456 EF      RST CALRUF      ; ...,Un',Yn+1,Yn'
2457 03      .BYT $03      ; ...,Un',Yn+1-Yn'=Vn'
2458 38      .BYT $38      ; XO+X,YO+Y,Xn+1,Yn+1,Un',Vn'
2459 CD B7 24      CALL $24B7      ; TEILSTÜCK ZEICHNEN
245C C1      POP BC      ; ZÄHLER ZURÜCKHOLEN
245D 10 C6      DJNZ GERADS      ; WEITER IN DER SCHLEIFE,
245F      ; BIS ALLE TEILE GEZEICHNET
245F      ;
245F EF      RST CALRUF      ; DIE ENDKOORDINATEN DES

```

2460	02	.BYT \$02	; LETZTEN TEILSTÜCKE WERDEN
2461	02	.BYT \$02	; GELÖSCHT
2462	01	.BYT \$01	; $Y0+Y, X0+Y$
2463	38	.BYT \$38	; ENDE
2464	3A 7D 5C	LD A, (COORDS)	; Ym = ENDPUNKT LADEN
2467	CD 28 2D	CALL \$2D28	; AUF DEN STACK BRINGEN
246A	EF	RST CALRUF	; $Y0+Y, Y0+X, Ym$
246B	03	.BYT \$03	; $Y0+Y, X0+X-Ym$
246C	01	.BYT \$01	; $X0+X-Ym, Y0+Y$
246D	38	.BYT \$38	; ENDE
246E	3A 7E 5C	LD A, (COORDS+1)	; Ym (WIE Ym) LADEN UND
2471	CD 28 2D	CALL \$2D28	; AUF DEM STACK SPEICHERN
2474	EF	RST CALRUF	; $X0+X-Ym, Y0+Y, Ym$
2475	03	.BYT \$03	; $X0+X-Ym, Y0+Y-Ym$
2476	38	.BYT \$38	; ENDE UND DEN LETZTEN GERADEN-
2477	CD B7 24	CALL \$24B7	; TEIL BIS $Y0+Y, Y0+Y$ ZEICHNEN
247A	C3 4D 0D	JP \$D4D	; TEMPORÄRE FARBEN SETZEN
247D			
247D			; SUBROUTINE ZUM BERECHNEN DER ANFANGSPARAMETER.
247D			; BEIM AUFRUF DURCH DIE SUBROUTINE ZUM ZEICHNEN
247D			; DES KREISES LIEGEN X,Y,R (RADIUS, HIER=2) UND
247D			; BEI AUFRUF DURCH 'DRAW' X,Y,SIN(G/2),Z AUF DEM
247D			; CALCULATORSTACK. DIE RECHNUNGEN WERDEN NUR AB
247D			; Z AUFGEFÜHRT
247D			
247D	EF	RST CALRUF	; Z,
247E	31	.BYT \$31	; Z,Z
247F	28	.BYT \$28	; Z,SQR(Z)
2480	34	.BYT \$34	; Z,SQR(Z),2, KONSTANTE 2
2481	32	.BYT \$32,\$00	; SPEICHERN
2482	00		
2483	01	.BYT \$01	; Z,2,SQR(Z)
2484	05	.BYT \$05	; Z,2/SQR(Z)
2485	E5	.BYT \$E5	; Z,2/SQR(Z),G
2486	01	.BYT \$01	; Z,G,2/SQR(Z)
2487	05	.BYT \$05	; Z,G*SQR(Z)/2
2488	2A	.BYT \$2A	; Z,ABS(G*SQR(Z)/2)
2489	38	.BYT \$38	; Z,A1=ABS(G*SQR(Z)/2)
248A	CD 05 2D	CALL \$2DD5	; A1 IN REG A LADEN
248D	38 06	JR C,\$2495	; A>256: 252 LADEN
248F	E6 FC	AND \$FC	; 4*INT(A1/4) BILDEN
2491	C6 04	ADD 4	; 4 ADDIEREN
2493	30 02	JR NC,\$2497	; SPRUNG BEI < 256
2495	3E FC	LD A,\$FC	; SONST 252 LADEN
2497	F5	PUSH AF	; TEILSTÜCKE SPEICHERN
2498	CD 28 2D	CALL \$2D28	; TEILSTÜCKE AUCH AUF
249B			; CALC.-STACK LEGEN
249B	EF	RST CALRUF	; Z,A(=TEILSTÜCKE)
249C	E5	.BYT \$E5	; Z,A,G
249D	01	.BYT \$01	; Z,G,A
249E	05	.BYT \$05	; Z,G/A
249F	31	.BYT \$31	; Z,G/A,G/A
24A0	1F	.BYT \$1F	; Z,G/A,SIN(G/A)
24A1	C4	.BYT \$C4	; SIN(G/A) NACH MEM4
24A2	02	.BYT \$02	; Z,G/A
24A3	31	.BYT \$31	; Z,G/A,G/A
24A4	A2	.BYT \$A2	; Z,G/A,G/A,.5
24A5	04	.BYT \$04	; Z,G/A,.5*G/A

24A6	1F	.BYT \$1F	; Z,G/A,SIN(.5*G/A)
24A7	C1	.BYT \$C1	; SIN(.5*G/A) NACH MEM1
24A8	01	.BYT \$01	; Z,SIN(.5*G/A),G/A
24A9	C0	.BYT \$C0	; G/A NACH MEM0 KOPIEREN
24AA	02	.BYT \$02	; Z,SIN(.5*G/A)
24AB	31	.BYT \$31	; Z,SIN(.5*G/A),SIN(.5*G/A)
24AC	04	.BYT \$04	; Z,SIN(.5*G/A)*SIN(.5*G/A)=S*S
24AD	31	.BYT \$31	; Z,S*S,S*S
24AE	0F	.BYT \$0F	; Z,2*S*S
24AF	A1	.BYT \$A1	; Z,2*S*S,1
24B0	03	.BYT \$03	; Z,2*S*S-1
24B1	1B	.BYT \$1B	; Z,1-2*S*S
24B2	C3	.BYT \$C3	; 1-2*S*S NACH MEM3
24B3	02	.BYT \$02	; Z
24B4	3B	.BYT \$3B	; ENDE
24B5	C1	POP BC	; ZÄHLER FÜR GERADENTEIL-
24B6	C9	RET	; STÜCKE NACH BC LADEN
24B7			
24B7		; SUBROUTINE ZUM ZEICHNEN VON GERADEN	
24B7			
24B7	CD 07 23	CALL \$2307	; ABS(Y) NACH B, ABS(X)
24BA			; NACH C, VORZEICHEN Y NACH D,
24BA			; VORZEICHEN X NACH E LADEN
24BA	79	LD A,C	; X UND Y VERGLEICHEN,
24BB	8B	CP B	; SODASS DER GRÖßERE WERT
24BC	30 06	JR NC,\$24C4	; WERT NACH H UND
24BE	69	LD L,C	; DER KLEINERE NACH L
24BF			; GELADEN WERDEN KANN
24BF	05	PUSH DE	; RICHTUNGEN RETTEN
24C0	AF	XOR A	; A LÖSCHEN
24C1	5F	LD E,A	; E LÖSCHEN: 1. SCHRITT;
24C2			; +/-1,0 (=HORIZONTAL)
24C2	1B 07	JR \$24CB	
24C4			
24C4	B1	OR C	; BEI X>=Y: PRÜFEN, OB
24C5			; BEIDE PARAMETER 0
24C5	C8	RET Z	; JA, RETURN
24C6	6B	LD L,B	; ABS(Y) NACH L
24C7	41	LD B,C	; ABS(X) ANCH B
24C8	D5	PUSH DE	; RICHTUNGEN RETTEN
24C9	16 00	LD D,0	; D LÖSCHEN: 1. SCHRITT
24CB			; 0,+/-1 (=VERTIKAL)
24CB	60	LD H,B	; DER GRÖßERE WERT VON
24CC			; ABS(X) BZW. ABS(Y) NACH H
24CC	7B	LD A,B	; NACH A KOPIEREN UND
24CD	1F	RRA	; DURCH 2 DIVIDIEREN
24CE	85	ADD L	; TEST AUF L+H/2>256
24CF	3B 03	JR C,\$24D4	; JA: DIAGONALSCHRITT
24D1	BC	CP H	; L+H/2<H, DANN HORIZONTAL
24D2	3B 07	JR C,\$24DB	; ODER VERTIKALEN SCHRITT
24D4	94	SUB H	; A - H BILDEN UND NACH
24D5	4F	LD C,A	; C KOPIEREN
24D6	D9	EXX	; 2. REGISTERSATZ EINSCHALTEN
24D7	C1	POP BC	; RICHTUNG NACH BC
24D8	C5	PUSH BC	; WIEDER ZWISCHENSPEICHERN
24D9	1B 04	JR \$24DF	; UND ZEICHNEN
24DB	4F	LD C,A	; A IN C RETTEN
24DC	D5	PUSH DE	; RICHTUNG RETTEN

4DD D9	EXX	; 2. REGISTERSATZ
4DE C1	POP BC	; RICHTUNG NACH BC
4DF 2A 7D 5C	LD HL, (COORDS)	; COORDS NACH HL (START)
4E2 78	LD A, B	; Y-SCHRITT NACH A
4E3 84	ADD H	; Y-KOORDINATE ADDIEREN
4E4 47	LD B, A	; NACH B KOPIEREN
4E5 79	LD A, C	; ZUM Y-SCHRITT 1 UND
4E6 3C	INC A	; X-KOORDINATE ADDIEREN
4E7 85	ADD L	
4EB 38 0D	JR C, \$24F7	; WEITER AUF BEREICHSÜBER-
4EA		; SCHREITUNG TESTEN
4EA 28 0D	JR Z, \$24F9	; KEIN CARRY UND ZERO: ERROR
4EC 3D	DEC A	; A WIEDER KORRIGIEREN
4ED 4F	LD C, A	; NACH C LADEN
4EE CD E5 22	CALL PLOTTE	; DEN PUNKT ZEICHNEN
4F1 D9	EXX	; NORMALER REGISTERSATZ
4F2 79	LD A, C	; X-WERT NACH A KOPIEREN
4F3 10 D9	DJNZ \$24CE	; IN DER SCHLEIFE WEITER
4F5 D1	POP DE	; STACK KORRIGIEREN
4F6 C9	RET	
4F7 28 F3	JR Z, \$24EC	; ZERO UND CARRY SIND
24F9		; IN ORDNUNG
24F9		
24F9 CF	RST ERR AUS	; FEHLERMELDUNG:
24FA 0A	.BYT \$0A	; 'INTEGER OUT OF RANGE'
24FB		
24FB	; -----	
24FB	;	
24FB	; UNTERPROGRAMME ZUR AUSWERTUNG VON AUSDRÜCKEN	
24FB	; DAS ERGEBNIS WIRD ALS LETZTER WERT AUF DEM CALC.-STACK	
24FB	; ABGELEGT (NUMERISCH), BEI STRINGS 5 BYTES MIT	
24FB	; FOLGENDER BEDEUTUNG: DAS ERSTE IST NICHT DEFINIERT,	
24FB	; DAS ZWEITE + DRITTE SIND DIE STARTADRESSE DES STRING	
24FB	; UND DIE LETZTEN ZWEI GEBEN DIE LÄNGE AN.	
24FB	; BIT 6 VON FLAGS WIRD 1 BEI NUMERISCHEN UND 0 BEI	
24FB	; STRING-AUSDRÜCKEN	
24FB	;	
24FB DF	AUSDRU RST GETAKT	; ERSTES ZEICHEN LADEN
24FC 06 00	LD B, 0	; B MIT 0 ALS MERKER BESETZEN
24FE C5	PUSH BC	; UND ZWISCHENSPEICHERN
24FF 4F	LD C, A	; ZEICHEN ZUM SUCHEN NACH C
2500 21 96 25	LD HL, \$2596	; BASISADRESSE DER FUNKTIONS-
2503		; TABELLE
2503 CD DC 16	CALL \$16DC	; IN DER TABELLE SUCHEN
2506 79	LD A, C	; ZEICHEN NACH A ZURÜCK
2507 D2 84 26	JP NC, \$2684	; ZEICHEN NICHT GEFUNDEN
250A 06 00	LD B, 0	; B MIT 0 BESETZEN UND IN
250C 4E	LD C, (HL)	; C DEN OFFSET LADEN
250D 09	ADD HL, BC	; ZU HL ADDIERT GIBT DIE
250E E9	JP (HL)	; EINSTIEGSADRESSE
250F		
250F	; SUBROUTINE ZUM UNTERSUCHEN AUF ANFÜHRUNGSSTRICH ...	
250F		
250F CD 74 00	CALL \$74	; CHADD +1, NÄCHSTES ZEICHEN
2512 03	INC BC	; ZÄHLER +1
2513 FE 0D	CP \$D	; CARRIAGE RETURN ?
2515 CA 8A 1C	JP Z, \$1C8A	; JA: ERROR
2518 FE 22	CP ...	; GANSEFUSS ?

```

251A 20 F3      JR NZ,$250F      ; NEIN
251C CD 74 00    CALL $74        ; CHADD +1, ...
251F FE 22      CP ''          ; WENN NACHSTES ZEICHEN EBEN-
2521 C9         RET             ; FALLS '' IST, ZEROFLAG SETZEN
2522           ;
2522           ; SUBROUTINE UNTERSUCHT, OB DIE BENÖTIGTEN ZWEI
2522           ; KOORDINATEN FOLGEN
2522           ;
2522 E7          RST GETNXT      ; CHADD +1, NACHSTES ZEICHEN
2523 FE 28      CP '('         ; KLAMMER AUF ?
2525 20 06      JR NZ,$252D     ; NEIN: ERROR
2527 CD 79 1C    CALL $1C79     ; DIE BEIDEN WERTE IN DEN CALC.-
252A           ; STACK EINLESEN
252A DF        RST GETAKT      ; AKTUELLES ZEICHEN HOLEN
252B FE 29      CP ')'         ; KLAMMER ZU ?
252D C2 BA 1C    JP NZ,$1C8A    ; NEIN: ERROR
2530           ;
2530           ; DIESE UNTERROUTINE DIENST ZUM ÜBERPRÜFEN, OB GERADE
2530           ; NUR EINE SYNTAXPRÜFUNG STATTFINDET ODER OB EIN
2530           ; PROGRAMM AM LAUFEN IST (ZEROFLAG ZURÜCKGESETZT)
2530           ;
2530 FD CB 01 7E  BIT 7,(IY+1)   ; BIT 7 = 0 BEDEUTET SYNTAX-
2534 C9         RET             ; PRÜFUNG: ZEROFLAG GESETZT
2535           ;
2535           ; SUBROUTINE ZUM SUCHEN DES ZEICHENS AN DEN KOORDINATEN
2535           ; X,Y DURCH 'SCREEN$'. NORMAL WERDEN NUR DIE IM
2535           ; ZEICHENSATZROM (AB $3D00) ABGELEGTE ZEICHEN GEFUNDEN:
2535           ; $20 BIS $7F
2535           ;
2535 CD 07 23      CALL $2307     ; Y-KOORDINATE NACH B, X NACH C
2538 2A 36 5C    LD HL,(CHARS)  ; BASISADRESSE DES ZEICHEN-
253B 11 00 01    LD DE,$100     ; SATZES -256, DESHALB WIRD
253E 19         ADD HL,DE       ; DIE ADDITION DURCHFÜHRT
253F 79         LD A,C          ; X NACH A BRINGEN
2540 0F         RRCA           ; X MIT 32 MULTIPLIZIEREN,
2541 0F         RRCA           ; DAMIT DIE LOWBYTE-ADRESSE
2542 0F         RRCA           ; BESTIMMT WERDEN KANN
2543 E6 E0      AND $E0
2545 AB        XOR B
2546 5F        LD E,A           ; ERGEBNIS NACH E
2547 79        LD A,C          ; DIE HIGHBYTE-ADRESSE
2548 E6 18      AND $18         ; WIRD BESTIMMT UND
254A EE 40      XOR $BILD      ; (BIT 6 WIRD IMMER GESETZT)
254C 57        LD D,A          ; NACH D GEBRACHT
254D 06 60      LD B,96        ; ES GIBT 96 ZEICHEN
254F C5        PUSH BC         ; ZÄHLER RETTEN (B)
2550 D5        PUSH DE         ; BILDSCHIRMZEIGER UND ADRESSE
2551 E5        PUSH HL         ; DES ZEICHENSATZES EBENFALLS
2552 1A        LD A,(DE)       ; ERSTE PUNKTREIHE DES ZEICHENS
2553 AE        XOR (HL)        ; MIT ERSTER PUNKTREIHE DES
2554           ; ZEICHENSATZES VERGLEICHEN
2554 28 04      JR Z,$255A      ; ERSTE REIHE STIMMT
2556 3C        INC A           ; ZEICHEN REVERSE ?
2557 20 1A      JR NZ,$2573    ; NEIN
2559 3D        DEC A           ; A WIEDER $FF FÜR REVERSE
255A 4F        LD C,A          ; C=0: NORMAL, $FF = REVERSE
255B 06 07      LD B,7         ; 7 ZEILEN DES ZEICHENS NOCH
255D 14        INC D           ; VERGLEICHEN

```

255E 23	INC HL	
255F 1A	LD A,(DE)	: NACHSTE ZEICHEN
2560 4E	XOR (HL)	: VERGLEICHEN
2561 A9	XOR C	: MIT VERGLEICHER ER VERGLEICHEN
2562 C0 0F	JR NZ,\$2573	: FALSCHES ZEICHEN
2564 10 F7	DJNZ \$255D	: WEITER VERGLEICHEN
2566 C1	POP BC	: ZEICHEN GEFUNDEN: ZEICHENSATZ-
2567 C1	POP BC	: UND BILDSCHIRMZEIGER WEGWERFEN
2568 C1	POP BC	: ZEICHENNUMMER IN B
2569 3E 80	LD A,\$80	: DA VON HINTEN BEHALT WURDE,
256B 90	SUB B	: MUSS DIE NUMMER VON \$80 SUB-
256C		: TRAHIERT WERDEN
256C 01 01 00	LD BC,1	: EIN SPEICHERPLATZ FÜR DAS
256F F7	RST \$30	: ZEICHEN FREIMACHEN
2570 12	LD (DE),A	: DAS ZEICHEN AUSPEICHERN
2571 18 0A	JR \$257D	: ZEICHEN NOCH IM STACK ABLEGEN
2573		
2573 E1	POP HL	: ZEICHENSATZZEIGER HOLEN
2574 11 08 00	LD DE,B	: B ADDIEREN, UM AUF DAS
2577 19	ADD HL,DE	: NACHSTE ZEICHEN ZU ZEIGEN
2578 D1	POP DE	: BILDSCHIRMZEIGER UND
2579 C1	POP BC	: ZEICHENZÄHLER WIEDER HOLEN
257A 10 D3	DJNZ \$254F	: IN DIE SUCHSCHLEIFE ZURÜCK
257C 48	LD C,B	: KEIN ZEICHEN GEFUNDEN: LEER-
257D C3 B2 2A	JP \$2AB2	: STRING IM STACK SPEICHERN
2580		
2580		: UNTERPROGRAMM, UM DEN WERT VON ATTR (X,Y) ZU BESTIMMEN
2580		
2580 CD 07 23	CALL \$2307	: Y-KOORDINATE NACH B, X NACH C
2583 79	LD A,C	: ES WIRD DIE ATTRIBUTEADRESSE
2584 0F	RRCA	: BESTIMMT (IN HL)
2585 0F	RRCA	: ZEILE*32
2586 0F	RRCA	
2587 4F	LD C,A	: IN C MERKEN
2588 E6 E0	AND \$E0	
258A A8	XOR B	: DIE SPALTE DAZUADDIEREN
258B 6F	LD L,A	: IN L SPEICHERN
258C 79	LD A,C	: HIGHBYTEADRESSE BILDEN
258D E6 03	AND 3	
258F EE 58	XOR \$58	: ATTRIBUTEBEREICH \$5800-\$58FF
2591 67	LD H,A	: NACH H BRINGEN
2592 7E	LD A,(HL)	: DAS ATTRIBUT LADEN
2593 C3 28 2D	JP \$2D28	: UND IM STACK SPEICHERN
2596		
2596		: OFFSETTABELLE FÜR DIE ENTWICKLUNG VON AUSDRÜCKEN
2596		
2596 22	.BYT ' ', \$1C	: \$25B3 TEXTMODUS
2597 1C		
2598 28	.BYT '(', \$4F	: \$25E8 KLAMMER AUF
2599 4F		
259A 2E	.BYT '.', \$F2	: \$268D PUNKT FÜR DEZIMAL
259B F2		
259C 2B	.BYT '+', \$12	: \$25AF ADDITIONEN
259D 12		
259E A8	.BYT \$A8, \$56	: \$25F5 FN (FUNKTION)
259F 56		
25A0 A5	.BYT \$A5, \$57	: \$25F8 RND
25A1 57		

```

25A2 A7          .BYT $A7,$84      ; $2627 PI
25A3 B4
25A4 A6          .BYT $A6,$8F      ; $2634 INKEY$
25A5 BF
25A6 C4          .BYT $C4,$E6      ; $268D BIN
25A7 E6
25A8 AA          .BYT $AA,$BF      ; $266B SCREEN$
25A9 BF
25AA AB          .BYT $AB,$C7      ; $2672 ATTR
25AB C7
25AC A9          .BYT $A9,$CE      ; $267B POINT
25AD CE
25AE 00          .BYT $00
25AF
25AF             ; ES FOLGEN DIE EINZELNEN UNTERPROGRAMME, DIE MIT HILFE
25AF             ; DER OFFSETTABELLE AUFGERUFEN WERDEN
25AF
25AF             ;
25AF             ; PLUSZEICHEN
25AF
25AF E7          RST GETNXT          ; CHADD +1, NACHSTES ZEICHEN
25B0 C3 FF 24    JP $24FF           ; SPRUNG IN DIE HAUPTROUTINE
25B3
25B3             ; TEXTMODUS: STRINGS, EINFACHE UND MEHRFACHE, DIE MIT
25B3             ; ' ' EINGESCHLOSSEN SIND, BEARBEITEN
25B3
25B3 DF          RST GETAKT          ; AKTUELLES ZEICHEN LADEN
25B4 23          INC HL              ; ZEIGT JETZT AUF STRINGANFANG
25B5 E5          PUSH HL             ; ZWISCHENSPEICHERN
25B6 01 00 00    LD BC,0            ; STRINGLÄNGE = 0 SETZEN
25B7 CD 0F 25    CALL $250F          ; ZWEITEN GANSEFUSS SUCHEN
25B8 20 1B       JR NZ,$25D9         ; KEINEN GEFUNDEN
25B9 CD 0F 25    CALL $250F          ; SONST 3., 5. ETC. SUCHEN
25BA 28 FB       JR Z,$25BE          ; WEITERSUCHEN
25BB CD 30 25    CALL $2530          ; BEI SYNTAXPRÜFUNG MUSS
25BC 28 11       JR Z,$25D9         ; BIT 6 VON FLAGS GESETZT WERDEN
25BD F7          RST $30             ; SONST SPEICHERPLATZ IM WORK-
25BE              ; SPACE FÜR DEN STRING MIT AB-
25BF              ; SCHLIESSENDEN ANFÜHRUNGSSTRICH
25C0              ; BESORGEN
25C1 E1          POP HL              ; ZEIGER AUF STRINGANFANG HOLEN
25C2 D5          PUSH DE             ; ZEIGER AUF FREIEN PLATZ RETTEN
25C3 7E          LD A,(HL)           ; ERSTES STRINGZEICHEN LADEN
25C4 23          INC HL              ; ZEIGER +1
25C5 12          LD (DE),A           ; ZEICHEN ABSPEICHERN
25C6 13          INC DE              ; WORKSPACEZEIGER +1
25C7 FE 22       CP ' '             ; WAR ES GANSEFUSS ?
25C8 20 FB       JR NZ,$25CB         ; NEIN: WEITER UMSPEICHERN
25C9 7E          LD A,(HL)           ; FOLGT NOCH EIN GANSEFUSS ?
25CA 23          INC HL              ; (ZEIGER +1, EIN GANSEFUSS
25CB FE 22       CP ' '             ; WIRD UNTERDRÜCKT)
25CC 28 F2       JR Z,$25CB         ; JA: WEITERKOPIEREN
25CD 0B          DEC BC              ; TATSÄCHLICHE STRINGLÄNGE
25CE D1          POP DE              ; STRINGANFANG ZURÜCK IN DE
25CF 21 3B 5C    LD HL,FLAGS         ; HIER IMMER EINSTIEG, WENN BIT6
25D0 CB B6       RES 6,(HL)          ; VON FLAGS FÜR STRINGS GESETZT
25D1              ; WERDEN MUSS
25D2 CB 7E       BIT 7,(HL)          ; WÄHREND EINES PROGRAMMLAUF
25D3 C4 B2 2A    CALL NZ,$2AB2      ; WIRD DER STRING GESTACKT

```

```

25E5 C3 12 27 JP $2712 ; ZEILE WEITER UNTERSUCHEN
25E8 ;
25E8 ; EINSTIEG BEI KLAMMER AUF
25E8 ;
25E8 E7 RST GETNXT ; CHADD +1, NACHSTES ZEICHEN
25E9 CD FB 24 CALL $24FB ; UNTERSUCHUNGSRoutine AUFRUFEN
25EC FE 29 CP ')' ; EINE KLAMMER ALS ABSCHLUSS ?
25EE C2 8A 1C JP NZ,$1C8A ; NEIN: ERROR
25F1 E7 RST GETNXT ; SONST NACHSTES ZEICHEN UND
25F2 C3 12 27 JP $2712 ; WEITER UNTERSUCHEN
25F5 ;
25F5 C3 BD 27 JP $27BD ; SPRUNG FÜR 'FN'
25F8 ;
25F8 ; EINSTIEG BEI RND
25F8 ;
25F8 CD 30 25 CALL $2530 ; BEI SYNTAXPRÜFUNG WIRD
25FB 2B 2B JR Z,$2625 ; KEINE ZUFALLSZAHl BERECHNET
25FD ED 4B 76 5C LD BC,(SEED) ; AKTUELLER WERT VON SEED
2601 CD 2B 2D CALL $2D2B ; AUF DEM CALC.-STACK SPEICHERN
2604 EF RST CALRUF ; CALCULATORAUFRUF
2605 A1 .BYT $A1 ; EINE 1 SPEICHERN
2606 0F .BYT $0F ; DIESE ZU SEED ADDIEREN
2607 34 .BYT $34 ; ZAHl SPEICHERN: HIER 75
2608 37 .BYT $37,$16
2609 16
260A 04 .BYT $04 ; (SEED+1) * 75
260B 34 .BYT $34 ; ZAHl SPEICHERN:
260C 80 .BYT $80,$41,$00,$00,$80
260D 41
260E 00
260F 00
2610 80
2611 32 .BYT $32 ; (SEED+1)*75/65537 DURCHFÜHREN
2612 02 .BYT $02 ; LÖSCHEN: NUR DER REST BLEIBT
2613 A1 .BYT $A1 ; EINE 1 STACKEN
2614 03 .BYT $03 ; REST-1 BILDEN
2615 31 .BYT $31 ; ERGEBNIS DUPLIZIEREN
2616 3B .BYT $3B ; ENDE
2617 CD A2 2D CALL $2DA2 ; DEN LETZTEN WERT ALS NEUEN
261A ED 43 76 5C LD (SEED),BC ; SEED-WERT BENUTZEN
261E 7E LD A,(HL) ; EXPONENT 'LETZTER WERT' LADEN
261F A7 AND A ; SPRUNG, WENN EXPONENT
2620 2B 03 JR Z,$2625 ; NULL IST
2622 D6 10 SUB $10 ; SONST EXPONENT REDUZIEREN:
2624 77 LD (HL),A ; ENTSPRICHT DIVISION MIT 65536
2625 1B 09 JR $2630
2627 ;
2627 ; DIE ZAHl PI ALS LETZTEN WERT AUF DEN CALC.-STACK
2627 ; BRINGEN
2627 ;
2627 CD 30 25 CALL $2530 ; FALLS SYNTAXPRÜFUNG,
262A 2B 04 JR Z,$2630 ; WEITERPRÜFEN
262C EF RST CALRUF ; SONST CALCULATOR AUFRUFEN
262D A3 .BYT $A3 ; PI/2 AUF DEN STACK SPEICHERN
262E 3B .BYT $3B ; ENDE
262F 34 INC (HL) ; EXPONENT +1 = 2*(PI/2)
2630 ;
2630 E7 RST GETNXT ; NACHSTES ZEICHEN EINLESEN

```



```

2631 C3 C3 26          JP #26C3          ; UND WEITER UNTERSUCHEN
2634                  ;
2634                  ; EINSPRUNG BEI INKEY$
2634                  ;
2634 01 5A 10          LD BC,#105A        ; PRIORITÄT $10, LESECODE $5A
2637 E7               RST GETNXT          ; NÄCHSTES ZEICHEN LADEN UND
2638 FE 23            CP 'M'              ; AUF 'M' PRÜFEN
263A CA 0D 27         JP Z,#270D          ; JA: NUMERISCHES ARGUMENT
263D 21 3B 5C         LD HL,FLAGS         ; SONST STRINGFLAG
2640 CB B6            RES 6,(HL)          ; SETZEN
2642 CB 7E            BIT 7,(HL)          ; BEI SYNTAXPRÜFUNG
2644 2B 1F            JR Z,#2665          ; SPRUNG
2646 CD 9E 02         CALL #28E           ; SONST ZEICHENCODE IN DE HOLEN
2649 0E 00            LD C,0              ; LEERSTRING ANMERKEN
264B 20 13            JR NZ,#2660         ; WENN MEHRERE TASTEN GEDRÜCKT
264D CD 1E 03         CALL #31E           ; TASTENCODE TESTEN
2650 30 0E            JR NC,#2660         ; LEERSTRING STACKEN, WENN KEIN
2652                  ; BRAUCHBARES ZEICHEN
2652 15               DEC D                ; D AUF $FF SETZEN
2653 5F               LD E,A              ; TASTENCODE NACH E BRINGEN,
2654 CD 33 03         CALL #333           ; UM DIESEN ZU DEKODIEREN
2657 F5               PUSH AF             ; ASCII-ZEICHEN RETTEN
265B 01 01 00        LD BC,1             ; EINEN SPEICHERPLATZ
265B F7               RST #30             ; BESORGEN
265C F1               POP AF              ; ASCII-ZEICHEN ZURÜCKHOLEN
265D 12               LD (DE),A           ; UND ABSPEICHERN
265E 0E 01            LD C,1              ; STRINGLÄNGE
2660 06 00            LD B,0              ; AUF 1 SETZEN
2662 CD B2 2A         CALL #2AB2          ; STRING AUF DEM STACK ABLEGEN
2665 C3 12 27         JP #2712           ; ZUR WEITEREN UNTERSUCHUNG
2668                  ;
2668                  ; EINSTIEG BEI SCREEN$
2668                  ;
2668 CD 22 25         CALL #2522          ; AUF 2 KOORDINATEN PRÜFEN
266B C4 35 25         CALL NZ,#2535      ; AUFRUF NUR IM PROGRAMM
266E E7               RST GETNXT          ; CHADD +1, NÄCHSTES ZEICHEN UND
266F C3 DB 25         JP #25DB           ; WEITER UNTERSUCHEN
2672                  ;
2672                  ; EINSTIEG BEI ATTR
2672                  ;
2672 CD 22 25         CALL #2522          ; AUF 2 KOORDINATEN PRÜFEN
2675 C4 80 25         CALL NZ,#2580      ; AUFRUF NUR IM PROGRAMM
2678 E7               RST GETNXT          ; CHADD +1, NÄCHSTES ZEICHEN
2679 1B 49            JR #26C3           ; NUMERISCH SETZEN
267B                  ;
267B                  ; EINSTIEG BEI POINT
267B                  ;
267B CD 22 25         CALL #2522          ; AUF 2 KOORDINATEN PRÜFEN
267E C4 CB 22         CALL NZ,#22CB      ; AUFRUF NUR IM PROGRAMM
2681 E7               RST GETNXT          ; CHADD +1, NÄCHSTES ZEICHEN
2682 1B 3F            JR #26C3           ; NUMERISCH SETZEN
2684                  ;
2684                  ; EIN ZEICHEN AUF ALPHANUMERISCH PRÜFEN
2684                  ;
2684 CD 88 2C         CALL #2C88          ; ALPHANUMERISCH ?
2687 30 56            JR NC,#26DF         ; NEIN
2689 FE 41            CP 'A'              ; IST ES EIN BUCHSTABE ?
268B 30 3C            JR NC,#26C9        ; JA

```



```

2680 ;
2680 ; ROUTINE ZUM BEARBEITEN VON DEZIMALZAHLEN (AUCH "BIN")
2680 ;
2680 CD 30 25          CALL $0530      ; SYNTAXPRÜFUNG
2690 20 23            JR NZ,$26B5    ; NEIN
2692 ;
2692 ; BEI DER EINGABE EINER ZEILE (SYNTAXPRÜFUNG) WIRD EINE
2692 ; ZAHL IN EINE FLOATINGPOINTZAHL UMGEWandelt UND IN DIE
2692 ; BASICZEILE KOPIERT
2692 ;
2692 CD 9B 2C          CALL $2C98      ; IN FLOATINGPOINTZAHL WANDeln
2695 DF              RST GETAKT      ; HL ZEIGT AUF LETZTES DIGIT +1
2696 01 06 00         LD BC,6        ; DIE ZAHL BENÖTIGT 6 SPEICHER-
2699 CD 55 16         CALL $1655      ; PLATZE IN DER BASICZEILE
269C 23              INC HL          ; ZEIGER AUF ERSTEN FREIEN PLATZ
269D 36 0E           LD (HL),#E      ; CODE FÜR ZAHLENMERKER
269F 23              INC HL          ; NÄCHSTER PLATZ UND SO
26A0 EB              EX DE,HL        ; KOPIEREN NACH DE BRINGEN
26A1 2A 65 5C        LD HL,(STKEND) ; STACKENDE ALT LADEN
26A4 0E 05           LD C,5         ; 5 BYTES MUSSEN KOPIERT WERDEN
26A6 A7              AND A          ; CARRY LÖSCHEN
26A7 ED 42           SBC HL,BC      ; STACKENDE NEU BERECHNEN
26A9 22 65 5C        LD (STKEND),HL ; UND SPEICHERN
26AC ED 80           LDIR           ; DIE FLOATINGPOINTZAHL KOPIEREN
26AE EB              EX DE,HL        ; ZEILENPOINTER NACH HL, AUF DAS
26AF 2B              DEC HL         ; LETZTE EINGEFÜGTE BYTE SETZEN
26B0 CD 77 00        CALL $77       ; UND CHADD NEU SETZEN
26B3 1B 0E           JR $26C3       ; NUMERISCH ANMERKEN
26B5 ;
26B5 ; DER FOLGENDE TEIL WIRD IM PROGRAMMLAUF BEARBEITET
26B5 ;
26B5 DF              RST GETAKT      ; HL ZEIGT AUF AKTUELLES ZEICHEN
26B6 23              INC HL          ; HL +1, UM DAS NÄCHSTE
26B7 7E              LD A,(HL)       ; ZEICHEN ZU LADEN
26B8 FE 0E           CP #E          ; SUCHEN, BIS DER ZAHLEN-
26BA 20 FA           JR NZ,$26B6     ; MERKER GEFUNDEN WIRD
26BC 23              INC HL          ; ZEIGT AUF ERSTES BYTE DER ZAHL
26BD CD B4 33        CALL $33B4     ; FP-ZAHL IN DEN CALC.-STACK
26C0 22 5D 5C        LD (CHADD),HL  ; KOPIEREN UND CHADD NEU SETZEN
26C3 ;
26C3 FD CB 01 F6     SET 6,(IY+1)    ; FÜR NUMERISCHES ERGEBNIS MUSS
26C7 ;              ; BIT 6 VON FLAGS GESETZT WERDEN
26C7 1B 14           JR $26DD       ; UND WEITER UNTERSUCHEN
26C9 ;
26C9 ; SUBROUTINE SUCHT VARIABLE (NUMERISCH, STRING)
26C9 ; IM VARIABLEN- ODER PROGRAMMBEREICH. DIE VARIABLE
26C9 ; ODER DIE PARAMETER BEI STRINGVAR. WERDEN IN DEN
26C9 ; CALCULATORSTACK GEBRACHT
26C9 ;
26C9 CD B2 2B        CALL $2B82      ; VARIABLENNAMEN SUCHEN
26CC DA 2E 1C        JP C,$1C2E     ; NICHT GEFUNDEN: ERROR
26CF CC 96 29        CALL Z,$2996   ; STRINGPARAMETER IN DEN STACK
26D2 ;              ; BRINGEN ODER DIE ANFANGSADRESSE
26D2 ;              ; EINER NUMER. VARIABLEN HOLEN
26D2 3A 3B 5C        LD A,(FLAGS)
26D5 FE C0           CP $C0         ; AUF NUMERISCH TESTEN
26D7 3B 04           JR C,$26DD     ; NICHT DER FALL
26D9 23              INC HL         ; SONST MUSS DIE VARIABLE IN

```

```

26DA CD B4 33          CALL $33B4      ; DEN CALC.-STACK KOPIERT WERDEN
26DD 18 33             JR $2712        ; WEITER UNTERSUCHEN
26DF                   ;
26DF                   ; IM FOLGENDEN WERDEN VERSCHIEDENE OPERATOREN GEPRÜFT
26DF                   ;
26DF 01 DB 09          LD BC,$9DB      ; PRIORITÄT 9, KODE $DB
26E2 FE 2D             CP '-'          ; MINUSZEICHEN ?
26E4 28 27             JR Z,$270D      ; JA
26E6                   ;
26E6 01 18 10          LD BC,$1018     ; PRIORITÄT $10, KODE $18
26E9 FE AE             CP $AE          ; IST ES VAL$ ?
26EB 28 20             JR Z,$270D      ; JA
26ED D6 AF             SUB $AF         ; KODES VON $AF (CODE) BIS $C3
26EF                   ; (NOT) UNTERSUCHEN
26EF DA BA 1C          JP C,$1C8A      ; NEIN: ERROR
26F2 01 F0 04          LD BC,$4F0     ; PRIORITÄT 4, KODE $F0
26F5 FE 14             CP $14         ; IST ES NOT ?
26F7 28 14             JR Z,$270D      ; JA
26F9 D2 BA 1C          JP NC,$1C8A     ; NICHT IM BEREICH: ERROR
26FC                   ;
26FC                   ; TOKENS JETZT IN OPERATIONSKODES WANDELN
26FC                   ;
26FC 06 10             LD B,$10        ; PRIORITÄT $10
26FE C6 DC             ADD $DC         ; BEREICH $DC BIS $EF
2700 4F               LD C,A          ; KOPIE NACH C
2701 FE DF             CP $DF         ; CODE, VAL UND LEN AUSBLENDEN
2703 30 02             JR NC,$2707
2705 CB B1             RES 6,C         ; FÜR DIESE BIT 6 ZU 0 SETZEN
2707 FE EE             CP $EE         ; STR$ UND CHR$ ?
2709 3B 02             JR C,$270D      ; NEIN
270B CB B9             RES 7,C         ; JA: BIT 7 AUF 0 SETZEN
270D                   ;
270D                   ; DER PRIORITÄTSKODE IN B UND DER OPERATIONSKODE IN C
270D                   ; WERDEN AUF DEM STACK ABGELEGT, BEVOR DER NÄCHSTE TEIL
270D                   ; DES AUSDRUCKS UNTERSUCHT WIRD
270D                   ;
270D C5               PUSH BC          ; DIE BEIDEN KODES SPEICHERN
270E E7               RST GETNXT       ; CHADD +1, NÄCHSTES ZEICHEN
270F C3 FF 24          JP $24FF        ; HOLEN UND WEITERSUCHEN
2712                   ;
2712                   ; ES WIRD JETZT DER WEITERE AUSDRUCK UNTERSUCHT AUF
2712                   ; KLAMMER, ENDE ETC.
2712                   ;
2712 DF               RST GETAKT      ; AKTUELLES ZEICHEN LADEN UND
2713 FE 28             CP '('          ; AUF KLAMMER VERGLEICHEN
2715 20 0C             JR NZ,$2723     ; KEINE KLAMMER
2717 FD CB 01 76       BIT 6,(IY+1)    ; FLAGS TESTEN AUF NUMERISCH
271B 20 17             JR NZ,$2734     ; JA: AUSDRUCK IN KLAMMERN
271D CD 52 2A          CALL $2A52      ; PARAMETER DES 'LETZTEN WERTS'
2720                   ; VERÄNDERN
2720 E7               RST GETNXT       ; DAS NÄCHSTE ZEICHEN
2721 18 F0             JR $2713        ; LADEN UND UNTERSUCHEN
2723                   ;
2723                   ; ROUTINE, UM FÜR DIE DIVERSEN OPERATOREN (+,*,NOT ETC.)
2723                   ; DIE PRIORITÄTEN UND DEN OPERATIONSKODE ZU SUCHEN
2723                   ;
2723 06 00             LD B,0          ; REGISTER BC MIT DEM ORIGINAL-
2725 4F               LD C,A          ; KODE ZUM SUCHEN LADEN

```

2726	21 95 27	LD HL,\$2795	; ADRESSE DER OPERATORENTABELLE
2729	CD DC 16	CALL \$16DC	; UND IN DER TABELLE NACHSEHEN
272C	30 06	JR NC,\$2734	; NICHTS GEFUNDEN
272E	4E	LD C,(HL)	; OPERATIONS-KODE LADEN
272F	21 ED 26	LD HL,\$26ED	; PRIORITÄTENTABELLE-\$C3
2732	09	ADD HL,BC	; OPERATIONS-KODE ADDIEREN,
2733	46	LD B,(HL)	; DAMIT DIE PRIORITÄT IN B LADEN
2734			
2734	D1	POP DE	; VORHERGEHENDE OPERATION UND
2735			; PRIORITÄT VOM STACK HOLEN
2735	7A	LD A,D	; DIESE PRIORITÄT MIT DER
2736	B8	CP B	; JETZIGEN VERGLEICHEN
2737	38 3A	JR C,\$2773	; JETZIGE IST HÖHER
2739	A7	AND A	; SIND BEIDE PRIORITÄTEN 0 ?
273A	CA 18 00	JP Z,GETAKT	; JA
273D	C5	PUSH BC	; AKTUELLEN WERTE AUF DEN STACK
273E			
273E	21 3B 5C	LD HL,FLAGS	
2741	7B	LD A,E	; VORHERIGER OPERATIONS-KODE
2742	FE ED	CP \$ED	; = 'USR' ?
2744	20 06	JR NZ,\$274C	; NEIN
2746	CB 76	BIT 6,(HL)	; 'USR' MIT ZAHLEN ?
2748	20 02	JR NZ,\$274C	; JA
274A	1E 99	LD E,\$99	; LETZTEN OPERATIONS-KODE NEU
274C	D5	PUSH DE	; VORHERIGE WERTE AUF STACK
274D	CD 30 25	CALL \$2530	; WÄHREND SYNTAX-PRÜFUNG
2750	28 09	JR Z,\$275B	; NICHTS WEITER MACHEN
2752	7B	LD A,E	; OPERATIONS-KODE NACH A UND
2753	E6 3F	AND \$3F	; BITS 6+7 AUSBLENDEN:
2755	47	LD B,A	; = CALCULATOR-OFFSET
2756	EF	RST CALRUF	; CALCULATOR AUFRUFEN
2757	3B	.BYT \$3B	; DIESE AKTUELLE OPERATION
2758	3B	.BYT \$3B	; DURCHFÜHREN UND ENDE
2759	18 09	JR \$2764	
275B			
275B			; DIE ART DER LETZTEN OPERATION MIT DER ZU
275B			; UNTERSUCHENDEN VERGLEICHEN
275B			
275B	7B	LD A,E	; VORHERIGER OPERATIONS-KODE
275C	FD AE 01	XOR (IY+1)	; MIT FLAGS EXOR
275F	E6 40	AND \$40	; NUR BIT 6 BETRACHTEN
2761	C2 8A 1C	JP NZ,\$1C8A	; MUSS 0 SEIN, SONST ERROR
2764			
2764			; DIE ART DER VORHERIGEN OPERATION MUSS NOCH
2764			; IN FLAGS ANGEMERKT WERDEN
2764			
2764	D1	POP DE	; VORHERIGER OPERATIONS-KODE
2765	21 3B 5C	LD HL,FLAGS	; FLAGS MIT DEFAULT
2768	CB F6	SET 6,(HL)	; NUMERISCH VORBESETZEN
276A	CB 7B	BIT 7,E	; VORHERIGER WERT NUMERISCH ?
276C	20 02	JR NZ,\$2770	; JA
276E	CB B6	RES 6,(HL)	; FÜR STRINGS FLAG ZURÜCKSETZEN
2770	C1	POP BC	; JETZIGE OPERATIONS-KODES IN BC
2771	18 C1	JR \$2734	; BRINGEN UND WEITER UNTERSUCHEN
2773			
2773			; FALLS DIE JETZIGE PRIORITÄT HÖHER ALS DIE VORHERIGE
2773			; IST, WERDEN BEIDE OPERATIONEN AUF DEM STACK GESPEI-
2773			; CHERT. WENN DIE JETZIGE OPERATION EINE STRINGBE-

```

2773 ; BEARBEITUNG BEDEUTET, MUSS DIES NOCH IM
2773 ; OPERATIONSCODE ANGEMERKT WERDEN
2773 ;
2773 05 PUSH DE ; VORHERIGE OPERATION AUF STACK
2774 79 LD A,C ; JETZIGER OPERATIONS-KODE
2775 FD CB 01 76 BIT 6,(IY+1) ; FLAGS: NUMERISCH ?
2779 20 15 JR NZ,$2790 ; JA
277B E6 3F AND $3F ; STRINGS: BITS 6+7 WEG
277D C6 0B ADD B ; UND B ADDIEREN
277F 4F LD C,A ; ERGEBNIS NACH C ZURÜCK
2780 FE 10 CP $10 ; OPERATION = 'AND' ?
2782 20 04 JR NZ,$278B ; NEIN
2784 CB F1 SET 6,C ; JA: NUMERISCH ANMERKEN
2786 1B 0B JR $2790
278B ;
278B 3B D7 JR C,$2761 ; DIE OPERATIONEN '-',*,/,^,OR
278A ; SIND BEI STRINGS UNZULÄSSIG
278A FE 17 CP $17 ; OPERATION = '+' ?
278C 2B 02 JR Z,$2790 ; JA
278E CB F9 SET 7,C ; DIE ANDEREN OPERATIONEN ERGEBEN
2790 ; EIN NUMERISCHES RESULTAT
2790 C5 PUSH BC ; JETZIGEN WERTE AUF DEN STACK
2791 E7 RST GETNXT ; NÄCHSTES ZEICHEN LADEN UND
2792 C3 FF 24 JP $24FF ; ZUR WEITEREN UNTERSUCHUNG
2795 ;
2795 ; TABELLE ZUM UMSETZEN DER OPERATOREN IN OPERATIONS-KODES
2795 ;
2795 2B .BYT '+',$CF ; ADDITION
2796 CF
2797 2D .BYT '-',$C3 ; SUBTRAKTION
2798 C3
2799 2A .BYT '*',$C4 ; MULTIPLIKATION
279A C4
279B 2F .BYT '/',$C5 ; DIVISION
279C C5
279D 5E .BYT '^',$C6 ; EXPONENT (HOCH)
279E C6
279F 3D .BYT '=', $CE ; ZUWEISUNG (GLEICH)
27A0 CE
27A1 3E .BYT '>',$CC ; GRÖßER
27A2 CC
27A3 3C .BYT '<',$CD ; KLEINER
27A4 CD
27A5 C7 .BYT $C7,$C9 ; KLEINER GLEICH (<=)
27A6 C9
27A7 CB .BYT $CB,$CA ; GRÖßER GLEICH (>=)
27A8 CA
27A9 C9 .BYT $C9,$CB ; UNGLEICH (<>)
27AA CB
27AB C5 .BYT $C5,$C7 ; ODER (OR)
27AC C7
27AD C6 .BYT $C6,$CB ; UND (AND)
27AE CB
27AF 00 .BYT $00
27B0 ;
27B0 ; TABELLE DER ZUGEHÖRIGEN PRIORITÄTEN
27B0 ;
27B0 06 .BYT $06 ; -

```

```

27B1 08 .BYT $08 ; *
27B2 08 .BYT $08 ; /
27B3 0A .BYT $0A ;
27B4 02 .BYT $02 ; OF
27B5 03 .BYT $03 ; AND
27B6 05 .BYT $05 ; =
27B7 05 .BYT $05 ; =
27B8 05 .BYT $05 ; =
27B9 05 .BYT $05 ; =
27BA 05 .BYT $05 ; =
27BB 05 .BYT $05 ; =
27BC 06 .BYT $06 ; +
27BD ;
27BD ; SUBROUTINE ZUR ENTWICKLUNG VON BENUTZERDEFINIERTEN
27BD ; FUNKTIONEN (DEF FN)
27BD ;
27BD CD 30 25 CALL $2530 ; IM PROGRAMMLAUF
27C0 20 35 JR NZ,$27F7 ; DEN SPRUNG AUSFÜHREN
27C2 E7 RST GETNXT ; ERSTES ZEICHEN DES NAMEN HOLEN
27C3 CD 80 20 CALL $2C8D ; IST ES EIN BUCHSTABE ?
27C6 D2 8A 1C JP NC,$1C8A ; NEIN: ERROR
27C9 E7 RST GETNXT ; NACHSTES ZEICHEN
27CA FE 24 CP '$' ; AUF '$' PRÜFEN
27CC F5 PUSH AF ; ZEROFLAG ZWISCHENSPEICHERN
27CD 20 01 JR NZ,$27D0 ; ES IST KEIN '$'-ZEICHEN
27CF E7 RST GETNXT ; DAS FOLGENDE ZEICHEN
27D0 FE 28 CP '(' ; MUSS EINE KLAMMER SEIN
27D2 20 12 JR NZ,$27E6 ; SONST ERROR
27D4 E7 RST GETNXT ; NACHSTES ZEICHEN KLAMMER ZU ?
27D5 FE 29 CP ')' ;
27D7 28 10 JR Z,$27E9 ; JA: KEIN ARGUMENT
27D9 CD FB 24 CALL $24FB ; SONST DIE SUBROUTINE ZUR AUS-
27DC ; WERTUNG VON AUSDRÜCKEN ZUR
27DC ; SYNTAXPRÜFUNG AUFRUFEN
27DC DF RST GETAKT ; IST DAS ZEICHEN NACH DEM
27DD FE 2C CP ',' ; AUSDRUCK EIN KOMMA ?
27DF 20 03 JR NZ,$27E4 ; NEIN: KEINE WEITEREN ARGUMENTE
27E1 E7 RST GETNXT ; SONST NACHSTES ZEICHEN LADEN
27E2 18 F5 JR $27D9 ; UND DIE FUNKTION WEITER UNTER-
27E4 ; SUCHEN
27E4 ;
27E4 FE 29 CP ')' ; IST AKTUELLES ZEICHEN KLAMMER ?
27E6 C2 8A 1C JP NZ,$1C8A ; NEIN: ERROR
27E9 E7 RST GETNXT ; CHADD +1
27EA 21 3B 5C LD HL,FLAGS ; IN FLAGS EINE STRINGFUNKTION
27ED C8 B6 RES 6,(HL) ; SETZEN
27EF F1 POP AF ; ZEROFLAG VON OBEN ZURÜCKHOLEN
27F0 28 02 JR Z,$27F4 ; SPRUNG BEI STRINGFUNKTION
27F2 C8 F6 SET 6,(HL) ; SONST NUMERISCH ANMERKEN
27F4 C3 12 27 JP $2712 ; DIE ZEILE WEITER UNTERSUCHEN
27F7 ;
27F7 ; EINSTIEG FÜR FUNKTIONEN IM PROGRAMMLAUF
27F7 ;
27F7 E7 RST GETNXT ; ERSTEN BUCHSTABEN DES NAMEN
27F8 E6 DF AND $DF ; LADEN, ZUM GROSSBUCHSTABEN
27FA 47 LD B,A ; MACHEN UND NACH B KOPIEREN
27FB E7 RST GETNXT ; NACHSTES ZEICHEN HOLEN UND
27FC D6 24 SUB '$' ; AUF STRING UNTERSUCHEN

```


27FE	4F	LD C,A	; C = 0: STRING, SONST NUMERISCH
27FF	20 01	JR NZ,\$2802	; BEI NUMERISCHER FUNKTION
2801	E7	RST GETNXT	; KLAMMER AUF ÜBERGEHEN
2802		.END	
2802		.LIB SPEC2800-S	
2802		; SINCLAIR ZX SPECTRUM TEIL 2800	
2802			
2802	E7	RST GETNXT	; 1. ZEICHEN 1. ARGUMENT LADEN
2803	E5	PUSH HL	; POINTER ZWISCHENSPEICHERN
2804	2A 53 5C	LD HL,(PROG)	; PROGRAMMANFANG LADEN
2807	2B	DEC HL	; -1 FÜR DIE SUCHROUTINE
2808	11 CE 00	LD DE,\$CE	; 'DEF FN' WIRD GESUCHT
280B	C5	PUSH BC	; NAME UND FUNKTIONSART RETTEN
280C	CD 86 1D	CALL \$1D86	; PROGRAMM ABSUCHEN
280F	C1	POP BC	; NAME UND FUNKTIONSART ZURÜCK
2810	30 02	JR NC,\$2814	; SPRUNG BEI GEFUNDENEM 'DEF FN'
2812			
2812	CF	RST ERR AUS	; SONST FEHLERMELDUNG:
2813	1B	.BYT \$1B	; 'FN WITHOUT DEF'
2814			
2814	E5	PUSH HL	; ZEIGER AUF 'DEF FN' FÜR EVTL.
2815			; WEITERE SUCHE RETTEN
2815	CD AB 2B	CALL \$28AB	; NAME DER FUNKTION HOLEN
281B	E6 DF	AND \$DF	; NUR GROSSBUCHSTABEN
281A	B8	CP B	; STIMMT DER FUNKTIONSNAME ?
281B	20 0B	JR NZ,\$2825	; NEIN, WEITERSUCHEN
281D	CD AB 2B	CALL \$28AB	; DAS NÄCHSTE ZEICHEN LADEN
2820	D6 24	SUB '\$'	; UND AUF STRING PRÜFEN
2822	B9	CP C	; (C ENTHÄLT 0 FÜR STRING)
2823	2B 0C	JR Z,\$2831	; DIE FUNKTIONSART STIMMT
2825	E1	POP HL	; ZEIGER AUF 'DEF FN' ZURÜCK
2826	2B	DEC HL	; -1 FÜR DIE SUCHE
2827	11 00 02	LD DE,\$200	; DE FÜR DIE SUCHROUTINE MIT 2
282A	C5	PUSH BC	; LADEN, BC RETTEN, UND DAS ENDE
282B	CD 8B 19	CALL \$198B	; DES 'DEF FN'-BEFEHLS SUCHE
282E	C1	POP BC	; NAME UND FUNKTIONSART ZURÜCK
282F	1B D7	JR \$280B	; IM PROGRAMM WEITERSUCHEN
2831			
2831		; DER RICHTIGE DEF FN-BEFEHL WURDE GEFUNDEN.	
2831			
2831	A7	AND A	; BEI STRINGS (HL ZEIGT AUF \$)
2832	CC AB 2B	CALL Z,\$28AB	; DIE KLAMMER AUF SUCHE
2835	D1	POP DE	; ZEIGER AUF 'DEF FN' WEGWERFEN
2836	D1	POP DE	; ZEIGER AUF DAS ERSTE ARGUMENT
2837	ED 53 5D 5C	LD (CHADD),DE	; DER FUNKTION NACH CHADD LADEN
283B	CD AB 2B	CALL \$28AB	; KLAMMER AUF ÜBERSPRINGEN
283E	E5	PUSH HL	; ZEIGER DARAUF RETTEN
283F	FE 29	CP ')'	; 'FN' HAT KEIN ARGUMENT, WENN
2841	2B 42	JR Z,\$28B5	; NÄCHSTES ZEICHEN '=': SPRUNG
2843	23	INC HL	; ZEIGT AUF DAS NÄCHSTE ZEICHEN
2844	7E	LD A,(HL)	; ZEICHEN LADEN UND AUF
2845	FE 0E	CP \$E	; ZAHLENMERKER PRÜFEN
2847	16 40	LD D,\$40	; BIT6 FÜR NUMERISCH VORBESETZEN
2849	2B 07	JR Z,\$2852	; NUMERISCH: SPRUNG
284B	2B	DEC HL	; HL ZEIGT WIEDER AUF KLAMMER
284C	CD AB 2B	CALL \$28AB	; DIESE ÜBERSPRINGEN
284F	23	INC HL	; AUF NÄCHSTES ZEICHEN SETZEN
2850	16 00	LD D,0	; IN D STRING MERKEN

2852		INC HL	; ZEIGT AUF DAS 1. VON 5 BYTES
2852	23	PUSH HL	; ZEIGER ZWISCHENSPEICHERN
2853	E5	PUSH DE	; ART DER FUNKTION RETTEN
2854	D5	CALL \$24FB	; ARGUMENT AUSWERTEN
2855	CD FB 24	POP AF	; FUNKTIONSART WIEDER LADEN
2858	F1	XOR (IY+1)	; UND MIT FLAGS BIT 6
2859	FD AE 01	AND \$40	; VERGLEICHEN
285C	E6 40	JR NZ,\$288B	; STIMMT NICHT: ERROR
285E	20 2B	POP HL	; ZEIGER AUF 1. ZEICHEN
2860	E1	EX DE,HL	; NACH DE BRINGEN
2861	EB	LD HL,(STKEND)	; HL MIT STACKEND LADEN
2862	2A 65 5C	LD BC,5	; STACKEND UM 5 VERMINDERN
2865	01 05 00	SBC HL,BC	; (= 5 BYTES ENTFERNEN)
2868	ED 42	LD (STKEND),HL	; UND DEN NEUEN WERT SPEICHERN
286A	22 65 5C	LDIR	; ERGEBNIS HINTER 'DEF FN'
286D	ED B0		; KOPIEREN
286F		EX DE,HL	; HL AUF DAS LETZTE ZEICHEN
286F	EB	DEC HL	; DES ERGEBNISSES SETZEN
2870	2B	CALL \$28AB	; HL MUSS AUF DAS ERSTE ZEICHEN
2871	CD AB 2B		; DANACH ZEIGEN
2874		CP ')'	; IST DIESES EINE KLAMMER ZU
2874	FE 29	JR Z,\$2885	; JA, KEINE ARGUMENTE MEHR
2876	2B 0D	PUSH HL	; SONST ZEIGER DARAUFG RETTEN
2878	E5	RST GETAKT	; AKTUELLES ZEICHEN LESEN
2879	DF	CP ','	; IST ES EIN KOMMA FÜR WEITERE
287A	FE 2C		; ARGUMENTE ?
287C		JR NZ,\$288B	; NEIN, PARAMETER STIMMEN NICHT:
287C	20 0D		; ERROR
287E		RST GETNXT	; CHADD +1
287E	E7	POP HL	; ZEIGT WIEDER AUF DAS KOMMA
287F	E1	CALL \$28AB	; DAS NÄCHSTE ARGUMENT SUCHEN
2880	CD AB 2B	JR \$2843	; (HL) UND BEARBEITEN
2883	1B BE		
2885		PUSH HL	; ZEIGER AUF KLAMMER RETTEN
2885	E5	RST GETAKT	; NOCHMAL DAS AKTUELLE ZEICHEN
2886	DF	CP ')'	; UND AUF KLAMMER ZU VERGLEICHEN
2887	FE 29	JR Z,\$288D	; ALLES IN ORDNUNG
2889	2B 02		
288B		RST ERR AUS	; WENN NICHT, FEHLERMELDUNG:
288B	CF	.BYT \$19	; 'PARAMETER ERROR'
288C	19		
288D		POP DE	; ZEIGER AUF ')'
288D	D1	EX DE,HL	; IN HL BRINGEN
288E	EB	LD (CHADD),HL	; UND CHADD SETZEN
288F	22 5D 5C	LD HL,(DEFADD)	; ALTER WERT VON DEFADD LADEN
2892	2A 0B 5C	EX (SP),HL	; AUF DEM STACK ABLEGEN UND DIE
2895	E3	LD (DEFADD),HL	; ANFANGSADRESSE DER ARGUMENTE
2896	22 0B 5C		; VOM STACK NACH DEFADD BRINGEN
2899		PUSH DE	; ZEIGER AUF ') RETTEN
2899	D5	RST GETNXT	; CHADD +2, UM ')'
289A	E7	RST GETNXT	; UND '=' ZU ÜBERSPRINGEN
289B	E7	CALL \$24FB	; DIE FUNKTION ENTWICKELN
289C	CD FB 24	POP HL	; ZEIGER AUF ') ZURÜCK NACH
289F	E1	LD (CHADD),HL	; CHADD BRINGEN
28A0	22 5D 5C	POP HL	; DEN ALTEN WERT VON
28A3	E1	LD (DEFADD),HL	; DEFADD WIEDER HOLEN
28A4	22 0B 5C	RST GETNXT	; CHADD +1 UND WEITER
28A7	E7		

```

29AB C3 12 27 JP $2712 ; IN DER BASICZEILE
28AB ;
28AB ; ROUTINE ZUM ÜBERSPRINGEN VON FUNKTIONSTEILEN (ZEICHEN
28AB ; ($21) NUR MIT HL, DENN CHADD MUSS ERHALTEN BLEIBEN
28AB ;
28AB 23 INC HL ; ZEIGER +1
28AC 7E LD A,(HL) ; KODE LADEN
28AD FE 21 CP ' ' ; SPACE UND STEUERZEICHEN
28AF 38 FA JR C,$28AB ; WERDEN ÜBERSPRUNGEN
28B1 C9 RET
28B2 ;
28B2 ; SUBROUTINE ZUM SUCHEN VON VARIABLEN
28B2 ; SUCHE ENTWEDER IM VARIABLENBereich ODER, BEI 'DEF FN',
28B2 ; IM ARGUMENTBEREICH DER FUNKTION
28B2 ;
28B2 FD CB 01 F6 SET 6,(IY+1) ; FLAGS: DEFAULT = NUMERISCH
28B6 DF RST GETAKT ; DAS AKTUELLE ZEICHEN LADEN
28B7 CD 8D 2C CALL $2C8D ; UND AUF BUCHSTABE PRÜFEN
28BA D2 8A 1C JP NC,$1CBA ; ERROR, WENN KEIN BUCHSTABE
28BD E5 PUSH HL ; ZEIGER AUF BUCHSTABEN RETTEN
28BE E6 1F AND $1F ; NUR BITS 4-0 NACH
28C0 4F LD C,A ; REG C BRINGEN
28C1 E7 RST GETNXT ; DAS NÄCHSTE ZEICHEN LADEN
28C2 E5 PUSH HL ; UND DEN POINTER HIERAUF RETTEN
28C3 FE 28 CP '(' ; KLAMMER AUF ?
28C5 28 28 JR Z,$28EF ; JA: ARRAY
28C7 CB F1 SET 6,C ; DEFAULT KURZER NAME
28C9 FE 24 CP '$' ; IST ES EIN STRING (-ARRAY) ?
28CB 28 11 JR Z,$28DE ; JA
28CD CB E9 SET 5,C ; NORMALE ZAHL MERKEN
28CF CD 88 2C CALL $2C88 ; IST DAS ZEICHEN EIN BUCHSTABE?
28D2 30 0F JR NC,$28E3 ; NEIN: KURZER NAME
28D4 ;
28D4 ; DAS LETZTE ZEICHEN EINES VARIABLENNAMENS SUCHEN
28D4 ;
28D4 CD 88 2C CALL $2C88 ; BUCHSTABE ?
28D7 30 16 JR NC,$28EF ; NEIN, ENDE GEFUNDEN
28D9 CB B1 RES 6,C ; UNTERSCHIEDUNGSBIT LANGER NAME
28DB E7 RST GETNXT ; NÄCHSTES ZEICHEN LADEN UND
28DC 18 F6 JR $28D4 ; WEITERPRÜFEN
28DE ;
28DE E7 RST GETNXT ; CHADD +1, HINTER '$' SETZEN
28DF FD CB 01 B6 RES 6,(IY+1) ; FÜR STRINGS UND STRINGARRAYS
28E3 ; BIT6 VON FLAGS RÜCKSETZEN
28E3 ;
28E3 ; WENN DAS HIGHBYTE VON DEFADD NICHT NULL IST (=DEF FN)
28E3 ; UND WENN ZUR LAUFZEIT, DANN WIRD DIE SUCHE NACH
28E3 ; DEN ARGUMENTEN VON 'DEF FN' DURCHFÜHRT
28E3 ;
28E3 3A 0C 5C LD A,(DEFADD+1) ; DEFADD-HIGH
28E6 A7 AND A ; AUF 0 TESTEN
28E7 28 06 JR Z,$28EF ; NICHT 'DEF FN'
28E9 CD 30 25 CALL $2530 ; SYNTAXPRÜFUNG ?
28EC C2 51 29 JP NZ,$2951 ; NEIN, PROGRAMMLAUF
28EF ;
28EF 41 LD B,C ; 'ARTENBYTE' NACH B KOPIEREN
28F0 CD 30 25 CALL $2530 ; SYNTAXPRÜFUNG ?
28F3 20 08 JR NZ,$28FD ; NEIN

```

28F5	79	LD A,C	; VOM 'ARTENBYTE' DIE
28F6	E6 E0	AND \$E0	; NAMENBITS WEGWEPFEN
28F8	CB FF	SET 7,A	; SYNTAXPRÜFUNG ANMERKEN
28FA	4F	LD C,A	; UND NACH C KOPIEREN
28FB	18 37	JR \$2934	
28FD			
28FD		; WAHREND DES PROGRAMMLAUFES WIRD DIE SUCHE NACH DER	
28FD		; VARIABLEN IM VARIABLENBEREICH DURCHFÜHRT	
28FD			
28FD	2A 4B 5C	LD HL,(VARS)	; VARIABLENPOINTER LADEN
2900	7E	LD A,(HL)	; DIE VARIABLENNAMEN VERGLEICHEN
2901	E6 7F	AND \$7F	; NUR BITS 0-6
2903	28 2D	JR Z,\$2932	; HIER AUSSPRUNG, FALLS ENDE-
2905			; MARKIERUNG (\$90) DES
2905			; VARIABLENBEREICHES GEFUNDEN
2905	B9	CP C	; STIMMT DER NAME "
2906	20 22	JR NZ,\$292A	; NEIN
2908	17	RLA	; REG MIT 4 MULTIPLIZIEREN, UM
2909	B7	ADD A	; DIE BITS 5 UND 6 ZU TESTEN
290A	F2 3F 29	JP P,\$293F	; (5) STRINGS UND ARRAYS
290D	38 30	JR C,\$293F	; EINFACHE UND FOR-NEXTVARIABLEN
290F			
290F		; LANGE NAMEN GANZ UNTERSUCHEN	
290F			
290F	D1	POP DE	; ZEIGER AUF ZWEITES ZEICHEN
2910	D5	PUSH DE	; NACH DE
2911	E5	PUSH HL	; ZEIGER AUF 1. ZEICHEN RETTEN
2912	23	INC HL	; JEDES ZEICHEN NACHEINANDER
2913	1A	LD A,(DE)	; LADEN UND VERGLEICHEN
2914	13	INC DE	
2915	FE 20	CP	; LEERZEICHEN IGNORIEREN
2917	28 FA	JR Z,\$2913	
2919	F6 20	OR \$20	; IN KLEINBUCHSTABEN WANDELN
291B	BE	CP (HL)	; VERGLEICHEN
291C	28 F4	JR Z,\$2912	; STIMMT
291E	F6 80	OR \$80	; LETZTER BUCHSTABE DES
2920	BE	CP (HL)	; NAMENS "
2921	20 06	JR NZ,\$2929	; NEIN: NÄCHSTE VARIABLE PRÜFEN
2923	1A	LD A,(DE)	; DAS LETZTE ZEICHEN NOCH MAL
2924	CD 88 2C	CALL \$2C88	; UND AUF BUCHSTABEN PRÜFEN
2927	30 15	JR NC,\$293E	; RICHTIGE VARIABLE GEFUNDEN
2929			
2929		; WENN DIE NAMEN NICHT STIMMEN, MUSS DIE NÄCHSTE	
2929		; VARIABLE IM VARIABLENBEREICH GESUCHT WERDEN.	
2929		; HL IST DER ZEIGER DARAUF	
2929			
2929	E1	POP HL	; ZEIGER WIEDER HOLEN
292A	C5	PUSH BC	; BC ZWISCHENSPEICHERN
292B	CD 88 19	CALL \$1988	; NÄCHSTE VARIABLE SUCHE
292E	EB	EX DE,HL	; ZEIGER MÜSSEN GETAUSCHT WERDEN
292F	C1	POP BC	; BC WIEDER ZURÜCKHOLEN
2930	18 CE	JR \$2900	; ZUM VERGLEICH ZURÜCK
2932			
2932	CB FB	SET 7,B	; ANMERKEN: VARIABLE NICHT
2934			; GEFUNDEN
2934			
2934		; BEI SYNTAXPRÜFUNG HIER HIN	
2934			

2934	D1	POP DE	; ZEIGER AUF 2. ZEICHEN
2935	DF	RST GETAKT	; AKTUELLES ZEICHEN LADEN
2936	FE 28	CP '('	; KLAMMER AUF (FÜR ARRAY) ?
2938	28 09	JR Z,\$2943	; JA
293A	CB E8	SET 5,B	; SONST KEIN ARRAY ANMERKEN
293C	18 0D	JR \$294B	
293E			
293E		; DER RICHTIGE NAME IST GEFUNDEN	
293E			
293E	D1	POP DE	; BEI LANGEN NAMEN 1 ZEIGER MEHR
293F			; WEGWERFEN ALS BEI NORMALEN
293F	D1	POP DE	; ZEIGER AUF 2. ZEICHEN UND
2940	D1	POP DE	; AUF 1. ZEICHEN WEGWERFEN
2941	E5	PUSH HL	; ZEIGER AUF LETZTES ZEICHEN
2942			; RETTEN
2942	DF	RST GETAKT	; AKTUELLES ZEICHEN LADEN
2943	CD 88 2C	CALL \$2C8B	; UND NOCHMAL DEN VARIABLEN-
2946	30 03	JR NC,\$294B	; NAMEN ÜBERSPRINGEN: DIES
2948	E7	RST GETNXT	; WURDE WEITER OBEN ABER BEREITS
2949	18 FB	JR \$2943	; DURCHFÜHRT!
294B			
294B	E1	POP HL	; ZEIGER AUF LETZTES ZEICHEN
294C			; EINES VARIABLENNAMEN HOLEN
294C	CB 10	RL B	; ZUM FLAGSETZEN NACH LINKS RO-
294E	CB 70	BIT 6,B	; TIEREN UND BIT 6 (= 5 ALT)
2950	C9	RET	; TESTEN
2951			
2951		; UNTERPROGRAMM, UM DIE ARGUMENTE VON 'DEF FN' AUF	
2951		; DEN CALC.-STACK ABZULEGEN. DER EINSPRUNG ERFOLGT VON	
2951		; DER VARIABLENSUCHROUTINE AUS	
2951			
2951	2A 0B 5C	LD HL,(DEFADD)	; ZEIGT AUF 1. ZEICHEN IM
2954			; ARGUMENTBEREICH
2954	7E	LD A,(HL)	; ERSTES ZEICHEN LADEN
2955	FE 29	CP ')'	; WENN NUR KLAMMER ZU, DANN
2957	CA EF 28	JP Z,\$28EF	; IM VARIABLENBEREICH SUCHE
295A			
295A	7E	LD A,(HL)	; ERSTES ARGUMENT LADEN
295B	F6 60	OR \$60	; = EINFACHE NUMERISCHE VARIABLE
295D	47	LD B,A	; NACH B BRINGEN
295E	23	INC HL	; DAS NÄCHSTE ZEICHEN
295F	7E	LD A,(HL)	; LADEN
2960	FE 0E	CP \$E	; MERKER FÜR ZAHLEN ?
2962	28 07	JR Z,\$296B	; JA
2964	2B	DEC HL	; FÜR STRINGS SICHERSTELLEN,
2965	CD AB 28	CALL \$28AB	; DASS HL AUF EIN ZEICHEN
2968	23	INC HL	; NACH DEM '\$' ZEIGT
2969	CB AB	RES 5,B	; STRING-VARIABLE ANMERKEN
296B			
296B	7B	LD A,B	; VARIABLENART UND TYP NACH A
296C	B9	CP C	; MIT DEM GESUCHTEN VERGLEICHEN
296D	2B 12	JR Z,\$29B1	; RICHTIG GEFUNDEN
296F	23	INC HL	; SONST MIT HL DIE
2970	23	INC HL	; 5 BYTES EINER VARIABLEN
2971	23	INC HL	; ODER DIE STRINGPARAMETER
2972	23	INC HL	; ÜBERSPRINGEN
2973	23	INC HL	
2974	CD AB 28	CALL \$28AB	; DAS NÄCHSTE ZEICHEN HOLEN

2977	FE 29	CP ') '	; WENN DIESES EINE KLAMMER IST,
2979	CA EF 28	JP Z,\$28EF	; DANN IM VARIABLENB. SUCHEN
297C	CD AB 28	CALL \$28AB	; HL AUF DAS NÄCHSTE ARGUMENT
297F	18 D9	JR \$295A	; SETZEN UND UNTERSUCHEN
2981			
2981	CB 69	BIT 5,C	; ARGUMENT NUMERISCH ?
2983	20 0C	JR NZ,\$2991	; JA
2985	23	INC HL	; ZEIGT AUF 1. DER 5 AUF DEN
2986			; STACK ABZULEGENDEN BYTES
2986	ED 58 65 5C	LD DE,(STKEND)	; STACKEND LADEN
298A	CD C0 33	CALL \$33C0	; DIE 5 BYTES AUF DEM CALC.-
298D	EB	EX DE,HL	; STACK ABLEGEN UND DAS
298E	22 65 5C	LD (STKEND),HL	; NEUE STACKENDE SPEICHERN
2991	D1	POP DE	; DIE ZEIGER DER VARIABLENSUCH-
2992	D1	POP DE	; ROUTINE WEGWERFEN
2993	AF	XOR A	; CARRY UND ZEROFLAG LÖSCHEN, UM
2994	3C	INC A	; DAMIT ANZUZEIGEN, DASS NICHTS
2995	C9	RET	; MEHR GESTACKT WERDEN MUSS
2996			
2996			; SUBROUTINE ZUM SUCHEN VON STRINGPARAMETERN IM
2996			; VARIABLENBEREICH ODER ZUM FINDEN DER BASISADRESSE
2996			; EINES ARRAYS
2996			
2996	AF	XOR A	; ARRAYFLAG LÖSCHEN
2997	47	LD B,A	; REG B LÖSCHEN
2998	CB 79	BIT 7,C	; SYNTAXPRÜFUNG ?
299A	20 4B	JR NZ,\$29E7	; JA
299C	CB 7E	BIT 7,(HL)	; ARRAY ?
299E	20 0E	JR NZ,\$29AE	; JA
29A0	3C	INC A	; NORMALER STRING ANMERKEN
29A1	23	INC HL	; ERSTES BYTE ÜBERGEHEN
29A2	4E	LD C,(HL)	; DIE LÄNGENBYTES LOW
29A3	23	INC HL	
29A4	46	LD B,(HL)	; UND HIGH LADEN
29A5	23	INC HL	
29A6	EB	EX DE,HL	; ZEIGER NACH DE
29A7	CD B2 2A	CALL \$2AB2	; REGS A - E AUF STACK ABLEGEN
29AA	DF	RST GETAKT	; AKTUELLES ZEICHEN LADEN
29AB	C3 49 2A	JP \$2A49	
29AE			
29AE	23	INC HL	; BASISADRESSE EINES ELEMENTS
29AF	23	INC HL	; VON EINEM ARRAY BESTIMMEN, DIE
29B0	23	INC HL	; LÄNGENBYTES WERDEN ÜBERGANGEN
29B1	46	LD B,(HL)	; DIMENSION LADEN
29B2	CB 71	BIT 6,C	; ZAHLENARRAY ?
29B4	28 0A	JR Z,\$29C0	; JA
29B6	05	DEC B	; EIN STRINGARRAY MIT DIM =1 KANN
29B7			; WIE EIN NORMALER STRING
29B7	28 EB	JR Z,\$29A1	; BEHANDELT WERDEN: SPRUNG
29B9	EB	EX DE,HL	; ZEIGER IN DE ZWISCHENSPEICHERN
29BA	DF	RST GETAKT	; AKTUELLES ZEICHEN LADEN
29BB	FE 28	CP ' ('	; WENN ES NICHT EINE KLAMMER
29BD	20 61	JR NZ,\$2A20	; IST, DANN ERROR
29BF	EB	EX DE,HL	
29C0	EB	EX DE,HL	; ZEIGER NACH DE BRINGEN
29C1	18 24	JR \$29E7	
29C3			
29C3			; NACHFOLGEND DIE SCHLEIFE ZUM SUCHEN DER PARAMETER


```

29C3      ; EINES ARRAYELEMENTS. B DIENT ALS DIMENSIONSZÄHLER.
29C3      ; STRINGARRAYS HABEN EINE DIMENSION WENIGER ALS ANGEGE-
29C3      ; BEN, DA DER LETZTE TEIL ALS BESCHREIBUNG EINES
29C3      ; TEILSTRINGS DIENT
29C3      ;
29C3 E5      PUSH HL      ; ZÄHLER ZWISCHENSPEICHERN
29C4 DF      RST GETAKT   ; AKTUELLES ZEICHEN LADEN UND
29C5 E1      POP HL       ; ZÄHLER WIEDER HOLEN
29C6 FE 2C    CP ','      ; WENN ES EIN KOMMA IST, SPRUNG
29C8 2B 20    JR Z,$29EA   ; UM WEITERE BESCHREIBUNG
29CA         ; ZU BEARBEITEN
29CA CB 79    BIT 7,C      ; SYNTAXPRÜFUNG ?
29CC 2B 52    JR Z,$2A20   ; NEIN: ERROR
29CE CB 71    BIT 6,C      ; STRINGARRAY ?
29D0 20 06    JR NZ,$29DB  ; JA
29D2 FE 29    CP ')'      ; WENN DAS ZEICHEN NUNMEHR KEINE
29D4 20 3C    JR NZ,$2A12  ; KLAMMER IST: ERROR
29D6 E7      RST GETNXT   ; SONST CHADD +1
29D7 C9      RET
29D8         ;
29D8 FE 29    CP ')'      ; IST ZEICHEN EINE KLAMMER ?
29DA 2B 5C    JR Z,$2A4B   ; JA, WEITERE BESCHREIBUNG DES
29DC         ; (TEIL-) STRINGS UNTERSUCHEN
29DC FE CC    CP $CC       ; IST ES DAS TOKEN 'TO' ?
29DE 20 32    JR NZ,$2A12  ; NEIN
29E0 DF      RST GETAKT   ; HL MIT CHADD LADEN
29E1 2B      DEC HL       ; 1 ZEICHEN ZURÜCK
29E2 22 5D 5C LD (CHADD),HL ; CHADD NEU SETZEN
29E5 1B 5E    JR $2A45     ; TEILSTRING ENTWICKELN
29E7         ;
29E7         ; NORMALER EINSTIEGSPUNKT IN DIESE ROUTINE
29E7         ;
29E7 21 00 00 LD HL,0      ; ZÄHLER ZU NULL SETZEN
29EA E5      PUSH HL      ; ZÄHLER ZWISCHENSPEICHERN
29EB E7      RST GETNXT   ; CHADD +1
29EC E1      POP HL       ; ZÄHLER ZURÜCKHOLEN
29ED 79      LD A,C       ; BESCHREIBUNGSBYTE NACH A
29EE FE C0    CP $C0      ; SYNTAXPRÜFUNG FÜR ARRAYS ODER
29F0         ; STRINGS ?
29F0 20 09    JR NZ,$29FB  ; NEIN
29F2 DF      RST GETAKT   ; AKTUELLES ZEICHEN LADEN UND
29F3 FE 29    CP ')'      ; VERGLEICHEN
29F5 2B 51    JR Z,$2A4B   ; JA: ZÄHLEN DER ELEMENTE FERTIG
29F7 FE CC    CP $CC       ; IST ES DAS TOKEN 'TO' ?
29F9 2B E5    JR Z,$29E0   ; JA: TEILSTRING BEARBEITEN
29FB         ;
29FB C5      PUSH BC      ; DIM-ZÄHLER UND BESCHREIBUNGS-
29FC E5      PUSH HL      ; BYTE SOWIE DEN ELEMENTEZÄHLER
29FD         ; ZWISCHENSPEICHERN
29FD CD EE 2A CALL $2AEE   ; GRÖSSE DER DIMENSION NACH DE
2A00 E3      EX (SP),HL   ; TAUSCH ZÄHLER/VARIABLENZEIGER
2A01 EB      EX DE,HL     ; ZÄHLER NACH DE, GRÖSSE NACH HL
2A02 CD CC 2A CALL $2ACC   ; 'BESCHREIBUNG' DES NÄCHSTEN
2A05         ; ELEMENTS UNTERSUCHEN
2A05 3B 19    JR C,$2A20   ; NICHT MEHR IM BEREICH: ERROR
2A07 0B      DEC BC       ; ZÄHLER -1, WEIL DIE ELEMENTE
2A08         ; VOR DEM GESUCHTEN ABGEZÄHLT
2A08         ; WERDEN MÜSSEN

```


2A08	CD F4 2A	CALL \$2AF4	; ZÄHLER*DIM (HL*DE) BILDEN
2A0B	09	ADD HL,BC	; ELEMENTNUMMER ADDIEREN
2A0C	01	POP DE	; VARIABLENZEIGER WIEDER HOLEN
2A0D	C1	POP BC	; DIMENSION UND ARTENBYTE ZURÜCK
2A0E	10 B3	DJNZ \$29C3	; WEITER BIS B = 0
2A10			
2A10	CB 79	BIT 7,C	; FALLS SYNTAXPRÜFUNG,
2A12	20 66	JR NZ,\$2A7A	; DANN ERROR
2A14	E5	PUSH HL	; ZÄHLER RETTEN
2A15	CB 71	BIT 6,C	; STRINGARRAY ?
2A17	20 13	JR NZ,\$2A2C	; JA
2A19	42	LD B,D	; VARIABLENZEIGER NACH BC
2A1A	4B	LD C,E	
2A1B	DF	RST GETAKT	; AKTUELLES ZEICHEN
2A1C	FE 29	CP ')'	; MUSS KLAMMER ZU SEIN
2A1E	2B 02	JR Z,\$2A22	
2A20			
2A20	CF	RST ERR AUS	; SONST FEHLERMITTEILUNG:
2A21	02	.BYT \$02	; 'SUBSCRIPT OUT OF RANGE'
2A22			
2A22	E7	RST GETNXT	; CHADD +1
2A23	E1	POP HL	; ZÄHLER ZURÜCKHOLEN
2A24	11 05 00	LD DE,5	; 5 BYTES FÜR JEDES ARRAYELEMENT
2A27	CD F4 2A	CALL \$2AF4	; BENÖTIGTE BYTEZAHL VOR DEM
2A2A			; GESUCHTEN ELEMENT BERECHNEN
2A2A	09	ADD HL,BC	; HL ZEIGT AUF DEN PLATZ VOR
2A2B			; DEM GESUCHTEN ELEMENT
2A2B	C9	RET	
2A2C			
2A2C			; STRINGARRAYS WEITER BEARBEITEN
2A2C			
2A2C	CD EE 2A	CALL \$2AEE	; DIE LETZTE DIM-GRÖSSE HOLEN
2A2F	E3	EX (SP),HL	; VARIABLENZEIGER AUF STACK UND
2A30			; ZÄHLER NACH HL
2A30	CD F4 2A	CALL \$2AF4	; ZÄHLER*DIM-GRÖSSE AUSRECHEN
2A33	C1	POP BC	; VARIABLENZEIGER NACH BC
2A34	09	ADD HL,BC	; HL ZEIGT AUF DEN
2A35	23	INC HL	; ANFANG DES STRINGS
2A36	42	LD B,D	; LETZTE DIMENSIONSGRÖSSE ALS
2A37	4B	LD C,E	; LANGE NACH BC
2A38	EB	EX DE,HL	; STRINGANFANG NACH DE
2A39	CD B1 2A	CALL \$2AB1	; DIESE PARAMETER AUF DEM
2A3C			; CALC.-STACK ABLEGEN
2A3C	DF	RST GETAKT	; AKTUELLES ZEICHEN LADEN
2A3D	FE 29	CP ')'	; IST DIESER TEIL FERTIG ?
2A3F	2B 07	JR Z,\$2A4B	; JA
2A41	FE 2C	CP ','	; WENN ES KEIN KOMMA IST,
2A43	20 0B	JR NZ,\$2A20	; DANN ERROR
2A45	CD 52 2A	CALL \$2A52	; PARAMETER MODIFIZIEREN
2A48	E7	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
2A49	FE 2B	CP '('	; WENN DIESER KLAMMER AUF IST,
2A4B	2B FB	JR Z,\$2A45	; MUSS EIN WEITERER TEIL UNTER-
2A4D			; SUCHT WERDEN
2A4D	FD CB 01 B6	RES 6,(IY+1)	; STRINGERGEBNIS ANMERKEN
2A51	C9	RET	
2A52			
2A52			; SUBROUTINE ZUM BEARBEITEN VON TEILSTRINGS
2A52			

2A52	CD 30 25	CALL \$2530	; SYNTAXPRÜFUNG ?
2A53	C4 F1 2B	CALL NZ,\$2BF1	; NEIN: PARAMETER VOM
2A58			; CALC.-STACK HOLEN
2A58	E7	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
2A59	FE 29	CP ')'	; WENN ')', DANN IST DIE ANGABE
2A5B	2B 50	JR Z,\$2AAD	; NUR '()', SPRUNG
2A5D	D5	PUSH DE	; STARTADRESSE AUF STACK
2A5E	AF	XOR A	; REG A = 0 UND
2A5F	F5	PUSH AF	; ZWISCHENSPEICHERN
2A60	C5	PUSH BC	; LÄNGE ZWISCHENSPEICHERN
2A61	11 01 00	LD DE,1	; DEFAULTANFANG: 1. BUCHSTABEN
2A64	DF	RST GETAKT	; ERSTES ZEICHEN LADEN
2A65	E1	POP HL	; LÄNGE NACH HL
2A66	FE CC	CP \$CC	; IST ES DAS TOKEN 'TO' ?
2A68	2B 17	JR Z,\$2AB1	; JA (1. PARAMETER = 1)
2A6A	F1	POP AF	; REG A (=0) ZURÜCKHOLEN
2A6B	CD CD 2A	CALL \$2ACD	; NACH BC DEN 1. PARAMETER LADEN
2A6E			; REG A ENTHÄLT \$FF, FALLS
2A6E			; BEREICHSÜBERSCHREITUNG
2A6E	F5	PUSH AF	; REG A ZWISCHENSPEICHERN
2A6F	50	LD D,B	; DEN ERSTEN PARAMETER
2A70	59	LD E,C	; NACH DE KOPIEREN
2A71	E5	PUSH HL	; DIE LÄNGE ZWISCHENSPEICHERN
2A72	DF	RST GETAKT	; AKTUELLES ZEICHEN LADEN
2A73	E1	POP HL	; LÄNGE NACH HL LADEN
2A74	FE CC	CP \$CC	; IST ES DAS TOKEN 'TO' ?
2A76	2B 09	JR Z,\$2AB1	; JA
2A78	FE 29	CP ')'	; SONST MUSS ES EINE ') SEIN
2A7A	C2 BA 1C	JP NZ,\$1C8A	; WENN NICHT: ERROR
2A7D			
2A7D			; HIER WIRD EIN EINZELNES ZEICHEN EINES STRINGS,
2A7D			; Z. B. C\$(7), BEARBEITET
2A7D			
2A7D	62	LD H,D	; PARAMETER NACH DE: DAS LETZTE
2A7E	6B	LD L,E	; IST GLEICH DEM ERSTEN ZEICHEN
2A7F	1B 13	JR \$2A94	
2A81			
2A81	E5	PUSH HL	; LÄNGE ZWISCHENSPEICHERN
2A82	E7	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
2A83	E1	POP HL	; LÄNGE WIEDER NACH HL
2A84	FE 29	CP ')'	; WENN KLAMMER, DANN GIBT ES NUR
2A86	2B 0C	JR Z,\$2A94	; 1 PARAMETER
2A88	F1	POP AF	; REG A ZURÜCKHOLEN
2A89	CD CD 2A	CALL \$2ACD	; DEN 2. PARAMETER NACH BC
2A8C	F5	PUSH AF	; REG A RETTEN (\$FF = ERROR)
2A8D	DF	RST GETAKT	; AKTUELLES ZEICHEN LADEN
2A8E	60	LD H,B	; ZWEITEN PARAMETER NACH
2A8F	69	LD L,C	; HL KOPIEREN
2A90	FE 29	CP ')'	; ES MUSS EINE KLAMMER SEIN
2A92	20 E6	JR NZ,\$2A7A	; SONST ERROR
2A94	F1	POP AF	; ERRORANZEIGE ZURÜCKHOLEN
2A95	E3	EX (SP),HL	; 2. PARAMETER AUF STACK, ANFANG
2A96			; NACH HL
2A96	19	ADD HL,DE	; 1. PARAMETER ZUM ANFANG
2A97	2B	DEC HL	; ADDIEREN UND -1: = 1. ZEICHEN
2A98			; DES TEILSTRINGS
2A98	E3	EX (SP),HL	; RETTEN UND 2. PARAMETER HOLEN
2A99	A7	AND A	; CARRY LÖSCHEN

2A9A	ED 52	SBC HL,DE	; 2. - 1. PARAMETER ERGIBT LÄNGE
2A9C	01 00 00	LD BC,0	; NEUE LÄNGE = 0
2A9F	38 07	JR C,\$2AA8	; NEGATIVE LÄNGE WIRD NULLLÄNGE
2AA1	23	INC HL	; LÄNGE KORRIGIEREN
2AA2	A7	AND A	; BEREICHSÜBERSCHREITUNG ?
2AA3	FA 20 2A	JP M,\$2A20	; JA: ERROR
2AA6	44	LD B,H	; NEUE LÄNGE NACH BC
2AA7	4D	LD C,L	; KOPIEREN
2AA8	D1	POP DE	; START 1. ZEICHEN TEILSTRING
2AA9	FD CB 01 B6	RES 6,(IY+1)	; FLAGS: STRINGERGEBNIS MERKEN
2AAD	CD 30 25	CALL \$2530	; SYNTAXPRÜFUNG ?
2AB0	CB	RET Z	; JA: RETURN
2AB1			
2AB1		; SUBROUTINE, UM PARAMETER, DIE IN DEN REGISTERN A - E	
2AB1		; ENTHALTEN SIND, AUF DEM CALC.-STACK ABZULEGEN.	
2AB1		; REG A = 0 BEDEUTET STRING VON EINEM ARRAY ODER EIN	
2AB1		; TEILSTRING, EINE 1 SIGNALISIERT EINEN EINFACHEN	
2AB1		; STRING, DESSEN ALTER WERT ENTFERNT WERDEN KANN.	
2AB1			
2AB1	AF	XOR A	; TEILSTRING ODER STRING VON
2AB2			; EINEM ARRAY ANMERKEN
2AB2	FD CB 01 B6	RES 6,(IY+1)	; FLAGS: STRINGERGEBNIS MERKEN
2AB6	C5	PUSH BC	; ZWISCHENSPEICHERN
2AB7	CD A9 33	CALL \$33A9	; TEST, OB NOCH 5 BYTES FREI
2ABA	C1	POP BC	; BC WIEDER LADEN
2ABB	2A 65 5C	LD HL,(STKEND)	; STACKENDE LADEN: 1. FREIER
2ABE	77	LD (HL),A	; PLATZ, DIE PARAMETER EINZELN
2ABF	23	INC HL	; EINSCHREIBEN
2AC0	73	LD (HL),E	; ANFANG EINES STRINGS
2AC1	23	INC HL	
2AC2	72	LD (HL),D	
2AC3	23	INC HL	
2AC4	71	LD (HL),C	; LÄNGE DES STRINGS
2AC5	23	INC HL	
2AC6	70	LD (HL),B	
2AC7	23	INC HL	
2AC8	22 65 5C	LD (STKEND),HL	; NEUES STACKENDE
2ACB	C9	RET	
2ACC			
2ACC		; SUBROUTINE ZUM EINLESEN EINER INTEGERZAHL INS BC-REG.	
2ACC		; DAS ERGEBNIS DARF NICHT GRÖßER ALS DER INHALT VON HL	
2ACC		; SEIN, SONST ERROR. ERRORANZEIGE IN REG A: \$FF = ERROR	
2ACC			
2ACC	AF	XOR A	; KEIN ERROR ANMERKEN
2ACD	D5	PUSH DE	; DE UND HL
2ACE	E5	PUSH HL	; ZWISCHENSPEICHERN
2ACF	F5	PUSH AF	; DESGLEICHEN ERRORANZEIGE
2AD0	CD 82 1C	CALL \$1C82	; NÄCHSTEN AUSDRUCK BERECHNEN
2AD3			; UND AUF DEN CALC.-STACK LEGEN
2AD3	F1	POP AF	; ERRORREGISTER ZURÜCK
2AD4	CD 30 25	CALL \$2530	; WENN SYNTAXPRÜFUNG,
2AD7	28 12	JR Z,\$2AEB	; DANN REST ÜBERSPRINGEN
2AD9	F5	PUSH AF	; ERRORREGISTER RETTEN
2ADA	CD 99 1E	CALL \$1E99	; LETZTER WERT VOM CALC.-STACK
2ADD			; NACH BC BRINGEN
2ADD	D1	POP DE	; ERRORREGISTER NACH D
2ADE	78	LD A,B	; FALLS DER AUSDRUCK 0 ERGAB,
2ADF	B1	OR C	

```

2AE0 37          SCF          ; DANN BEDEUTET DIES EINEN
2AE1 28 05      JR Z,$2AE8    ; FEHLER
2AE3           ;
2AE3 E1         POP HL       ; LIMIT ZUM TESTEN NACH HL
2AE4 E5         PUSH HL      ; BRINGEN
2AE5 A7         AND A        ; CARRY LÖSCHEN
2AE6 ED 42      SBC HL,BC     ; VERGLEICHEN: CARRY GELÖSCH:
2AE8           ; KEIN ERROR, SONST:
2AE8 7A         LD A,D        ; ERRORREGISTER NACH A
2AE9 DE 00      SBC 0         ; WENN ALLES IN ORDNUNG, BLEIBT
2AEB           ; DER WERT IN REG A, SONST WIRD
2AEB           ; EINE 1 SUBTRAHIERT: <0=ERROR
2AEB E1         POP HL       ; HL UND DE WIEDER
2AEC D1         POP DE       ; VOM STACK HOLEN
2AED C9         RET
2AEE           ;
2AEE           ; DIESE SUBROUTINE LADT REGISTER DE AUS DEN SPEICHER-
2AEE           ; PLATZEN (DE+1), (DE+2)
2AEE           ;
2AEE EB         EX DE,HL      ; ADRESSE NACH HL
2AEF C3         INC HL        ; +1 = (DE+1)
2AF0 5E         LD E,(HL)     ; E VON (DE+1) LADEN
2AF1 C3         INC HL        ; +1 = (DE+2)
2AF2 56         LD D,(HL)     ; D VON (DE+2) LADEN
2AF3 C9         RET
2AF4           ;
2AF4           ; SUBROUTINE ZUM BERECHNEN VON DE * HL UND TEST, DASS
2AF4           ; DAS ERGEBNIS IN HL KLEINER ALS 65536 IST
2AF4           ;
2AF4 CD 30 25    CALL $2530    ; WAHREND DER SYNTAXPRÜFUNG
2AF7 CB         RET Z         ; WIRD DER WERT NICHT BERECHNET
2AF8 CD A9 30    CALL $30A9    ; MULTIPLIKATION AUSFÜHREN
2AFB DA 15 1F    JP C,$1F15    ; ERROR: 'OUT OF MEMORY' (ES
2AFE C9         RET           ; WIRD EINE SPEICHERADRESSE
2AFF           ; UNTERSTELLT)
2AFF           ;
2AFF           ; BEFEHL LET
2AFF           ; ES WIRD DIE TATSÄCHLICHE ZUWEISUNG BEI LET, READ UND
2AFF           ; INPUT DURCHGEFÜHRT
2AFF           ;
2AFF 2A 4D 5C    LD HL,(DEST)  ; DERZEITIGE ZIELADRESSE LADEN
2B02 FD CB 37, 4E BIT 1,(IY+$37) ; FLAGX: EXISTIERT DIE VARIABLE
2B06           ; SCHON ?
2B06 28 5E      JR Z,$2B66    ; JA
2B08           ;
2B08 01 05 00    LD BC,5       ; DEFAULTLÄNGE NUMERISCH
2B0B 03         INC BC        ; +1 FÜR JEDES ZEICHEN VOM NAMEN
2B0C 23         INC HL
2B0D 7E         LD A,(HL)     ; ZEICHEN LADEN
2B0E FE 20      CP ' '        ; LEERZEICHEN IGNORIEREN
2B10 28 FA      JR Z,$2B0C
2B12 30 0B      JR NC,$2B1F   ; SPRUNG MIT $21 - $FF
2B14 FE 10      CP $10        ; ALS ENCODE $00 - $0F
2B16 38 11      JR C,$2B29    ; ZULASSEN
2B18 FE 16      CP $16        ; DESGLEICHEN DEN BEREICH
2B1A 30 0D      JR NC,$2B29   ; VON $16 - $1F ZULASSEN
2B1C 23         INC HL        ; $10 - $15 ÜBERSPRINGEN
2B1D 1B ED      JR $2B0C

```

2B1F			
2B1F	CD 88 2C	CALL \$2C88	; TEST: ZEICHEN ALPHANUMERISCH
2B22	3B E7	JR C,\$2B08	; JA: ZEICHEN FÜR LANGE NAMEN
2B24	FE 24	CP '\$'	; ZEICHEN '\$' FÜR STRING
2B26	CA C0 2B	JP Z,\$2B00	; JA
2B29			
2B29			; FÜR EINE NEUE NUMERISCHE VARIABLE WERDEN '9C' PLATZE
2B29			; GEBRAUCHT (NAME + WERT). DANACH WIRD DIE VARIABLE
2B29			; KOMPLETT KOPIERT
2B29			
2B29	79	LD A,C	; LANGE NACH A
2B2A	2A 59 5C	LD HL,(ELINE)	; HL AUF DAS ENDE DES VARIABLEN-
2B2D	2B	DEC HL	; BEREICHS (BYTE = \$20) SETZEN
2B2E	CD 55 16	CALL \$1655	; PLATZ FÜR DIE VARIABLE MACHEN
2B31	23	INC HL	; AUF DAS ERSTE NEUE BYTE ZEIGEN
2B32	23	INC HL	; UND AUF DAS ZWEITE
2B33	EB	EX DE,HL	; DIESER ZEIGER NACH DE
2B34	D5	PUSH DE	; UND ZWISCHENSPEICHERN
2B35	2A 4D 5C	LD HL,(DEST)	; ZEIGER AUF VARIABLENNAMEN-
2B38			; ANFANG LADEN
2B38	1B	DEC DE	; AUF ERSTES NEUES BYTE ZEIGEN
2B39	D6 06	SUB 6	; DIE ANZAHL DER ZUSÄTZLICHEN
2B3B	47	LD B,A	; ZEICHEN EINES LANGEN NAMENS
2B3C			; NACH B BRINGEN
2B3C	2B 11	JR Z,\$2B4F	; WENN 0: NORMALER NAME
2B3E			
2B3E	23	INC HL	; DIE ZUSÄTZLICHEN ZEICHEN
2B3F	7E	LD A,(HL)	; LADEN
2B40	FE 21	CP '!'	; ZEICHEN IM BEREICH \$00 - \$20
2B42	3B FA	JR C,\$2B3E	; IGNORIEREN
2B44	F6 20	OR \$20	; ALLE ZU KLEINBUCHSTABEN MACHEN
2B46	13	INC DE	; ZEIGER ERHÖHEN UND
2B47	12	LD (DE),A	; EINSCHREIBEN
2B48	10 F4	DJNZ \$2B3E	; WIEDERHOLEN, BIS B = 0 IST
2B4A			
2B4A	F6 80	OR \$80	; DAS LETZTE ZEICHEN MIT
2B4C	12	LD (DE),A	; \$80 ODERN ZUM MARKIEREN
2B4D	3E C0	LD A,\$C0	; MERKBYTE LANGER NAME
2B4F	2A 4D 5C	LD HL,(DEST)	; ADRESSE ERSTER BUCHSTABE
2B52	AE	XOR (HL)	; ERSTER BUCHSTABE EXOR MIT
2B53			; 0=KURZ ODER \$C0= LANG
2B53	F6 20	OR \$20	; KLEINBUCHSTABE SETZEN
2B55	E1	POP HL	; ZEIGER AUF ZWEITES ZEICHEN
2B56			; WEGWERFEN
2B56	CD EA 2B	CALL \$2BEA	; DEN BUCHSTABEN EINSCHREIBEN
2B59			; UND HL AUF DAS NEUE ENDEBYTE
2B59			; (= \$90) SETZEN
2B59			
2B59	E5	PUSH HL	; ZIELPOINTER RETTEN
2B5A	EF	RST CALRUF	; CALCULATOR AUFRUFEN UND
2B5B	02	.BYT \$02	; 5 BYTES LÖSCHEN
2B5C	3B	.BYT \$3B	; ENDE
2B5D	E1	POP HL	; ZIELPOINTER ZURÜCK
2B5E	01 05 00	LD BC,5	; ZAHLENLÄNGE = 5
2B61	A7	AND A	; CARRY LÖSCHEN
2B62	ED 42	SBC HL,BC	; HL ZEIGT AUF DEN ERSTEN DER 5
2B64	1B 40	JR \$2BA6	; PLATZE UND SPRUNG ZUM KOPIEREN
2B66			


```

2B66      ; BEARBEITUNG EINER BEREITS VORHANDENEN VARIABLEN
2B66      ;
2B66 FD CB 01 76      BIT 6, (IY+1)      ; FLAGS: STRINGVARIABLE ?
2B6A 2B 06      JR Z, $2B72      ; JA
2B6C      ;
2B6C      ; DER ALTE ZAHLENWERT EINER NUMERISCHEN VARIABLEN WIRD
2B6C      ; DURCH DEN NEUEN ÜBERSCHRIEBEN. HL MUSS DESHALB
2B6C      ; KORRIGIERT WERDEN
2B6C      ;
2B6C 11 06 00      LD DE, 6      ; 6 ADDIEREN, DAMIT HL HINTER
2B6F 19      ADD HL, DE      ; DIE VARIABLE ZEIGT
2B70 1B E7      JR $2B59      ; WIE NORMALE VARIABLE WEITER-
2B72      ; BEHADELN
2B72      ;
2B72      ; STRINGVARIABLEN BEARBEITEN
2B72      ;
2B72 2A 4D 5C      LD HL, (DEST)      ; ANFANGSADRESSE LADEN
2B75 ED 4B 72 5C      LD BC, (STLEN)      ; STRINGLÄNGE NACH BC LADEN
2B79 FD CB 37 46      BIT 0, (IY+$37)      ; FLAGX: KOMPLETTER EINFACHER
2B7D      ; STRING ?
2B7D 20 30      JR NZ, $2BAF      ; JA
2B7F      ;
2B7F      ; BEARBEITEN VON TEILSTRINGS UND STRINGS VON ARRAYS
2B7F      ;
2B7F 7B      LD A, B      ; LÄNGE DES STRINGS AUF
2B80 B1      OR C      ; NULL PRÜFEN
2B81 CB      RET Z      ; BEI NULL: RETURN
2B82 E5      PUSH HL      ; ANFANGSPONTER RETTEN
2B83 F7      RST $30      ; PLATZ IM WORKSPACE BESORGEN
2B84 D5      PUSH DE      ; ZEIGER AUF ERSTEN PLATZ RETTEN
2B85 C5      PUSH BC      ; LÄNGE EBENFALLS
2B86 54      LD D, H      ; DE AUF LETZTEN PLATZ SETZEN
2B87 5D      LD E, L
2B88 23      INC HL      ; HL DAHINTER ZEIGEN LASSEN
2B89 36 20      LD (HL), ' '      ; EIN SPACE EINSCHREIBEN
2B8B ED 8B      LDDR      ; ALLE WEITEREN PLATZE MIT
2B8D      ; LEERZEICHEN BESCHREIBEN
2B8D E5      PUSH HL      ; ZEIGER ZWISCHENSPEICHERN
2B8E CD F1 2B      CALL $2BF1      ; PARAMETER VOM CALC.-STACK
2B91 E1      POP HL      ; HOLEN UND ZEIGER ZURÜCKLADEN
2B92 E3      EX (SP), HL      ; ZEIGER MIT LÄNGE TAUSCHEN
2B93 A7      AND A      ; CARRY LÖSCHEN
2B94 ED 42      SBC HL, BC      ; DIE BEIDEN LÄNGEN VERGLEICHEN
2B96 09      ADD HL, BC
2B97 30 02      JR NC, $2B9B      ; STRING NICHT ZU LANG
2B99 44      LD B, H      ; NEUE LÄNGE ZU LANG: STRING
2B9A 4D      LD C, L      ; GEKÜRZT, BC = LÄNGE
2B9B E3      EX (SP), HL      ; LÄNGE MIT ZEIGER TAUSCHEN
2B9C EB      EX DE, HL      ; ZEIGER NACH DE, STRINGANFANG
2B9D      ; NACH HL
2B9D 7B      LD A, B      ; STRINGLÄNGE NOCH MAL
2B9E B1      OR C      ; AUF NULL TESTEN
2B9F 2B 02      JR Z, $2BA3      ; BEI NULL NICHTS EINSCHREIBEN
2BA1 ED 80      LDIR      ; SONST NEUEN STRING IN DEN
2BA3      ; WORKSPACE EINSCHREIBEN
2BA3 C1      POP BC      ; NEUE LÄNGE,
2BA4 D1      POP DE      ; ZEIGER AUF NEUEN BEREICH UND
2BA5 E1      POP HL      ; ANFANGSADRESSE HOLEN

```



```

2BA6      ;
2BA6      ; SUBROUTINE ZUM EINSCHREIBEN EINER NUMERISCHEN
2BA6      ; VARIABLEN VOM CALC.-STACK ODER EINES STRINGS
2BA6      ; VOM WORKSPACE IN DEN VARIABLENBEREICH
2BA6      ;
2BA6 EB      EX DE,HL      ; ZEIGER TAUSCHEN
2BA7 7B      LD A,B        ; LANGE AUF NULL
2BA8 B1      OR C          ; PRÜFEN
2BA9 CB      RET Z        ; NULL: NICHTS EINSCHREIBEN
2BAA 05      PUSH DE       ; ZIELADRESSE RETTEN
2BAB ED B0   LDIR          ; TRANSFER DURCHFÜHREN
2BAD E1      POP HL        ; ZIELADRESSE NACH HL
2BAE C9      RET
2BAF      ;
2BAF      ; BEARBEITUNG EINES KOMPLETTEN, NEUEN UND EINFACHEN
2BAF      ; STRINGS (VON LET HERKOMMEND)
2BAF      ;
2BAF 2B      DEC HL        ; HL AUF DEN BUCHSTABEN DES
2BB0 2B      DEC HL        ; VARIABLENNAMENS SETZEN
2BB1 2B      DEC HL        ; (=DEST -3)
2BB2 7E      LD A,(HL)     ; BUCHSTABEN LADEN
2BB3 E5      PUSH HL       ; ZEIGER AUF EXISTIERENDEN
2BB4      ; STRING RETTEN
2BB4 C5      PUSH BC        ; LANGE ZWISCHENSPEICHERN
2BB5 CD C6 2B CALL $2BC6    ; DEN NEUEN STRING ZUM
2BB8      ; VARIABLENBEREICH HINZUFÜGEN
2BB8 C1      POP BC        ; LANGE UND ZEIGER AUF
2BB9 E1      POP HL        ; DEN ALTEN STRING ZURÜCKHOLEN
2BBA 03      INC BC        ; LANGE +3 BILDEN FÜR
2BBB 03      INC BC        ; NAMEN UND STRINGLANGE
2BBC 03      INC BC
2BBD C3 E8 19 JP RAUS2      ; UND DEN ALTEN STRING ENTFERNEN
2BC0      ;
2BC0      ; NEUE EINFACHE STRINGS BEARBEITEN
2BC0      ;
2BC0 3E DF    LD A,$DF      ; MASKE FÜR BUCHSTABEN LADEN
2BC2 2A 4D 5C LD HL,(DEST)  ; ADRESSE DES BUCHSTABENS
2BC5 A6      AND (HL)       ; DIESEN MASKIEREN UND
2BC6 F5      PUSH AF        ; ZWISCHENSPEICHERN
2BC7 CD F1 2B CALL $2BF1    ; STRINGPARAMETER (LANGE,
2BCA EB      EX DE,HL       ; ANFANG) NACH HL
2BCB 09      ADD HL,BC       ; LANGE ADDIEREN (ERGIBT ENDE+1)
2BCC C5      PUSH BC        ; LANGE ZWISCHENSPEICHERN
2BCD 2B      DEC HL         ; AUF LETZTEN PLATZ ZEIGEN
2BCE 22 4D 5C LD (DEST),HL  ; ZEIGER ZWISCHENSPEICHERN
2BD1 03      INC BC         ; LANGE +3 FÜR:
2BD2 03      INC BC         ; NAMENSBYTE UND STRINLANGE
2BD3 03      INC BC
2BD4 2A 59 5C LD HL,(ELINE) ; HL AUF ENDEBYTE
2BD7 2B      DEC HL         ; DES VARIABLENBEREICHS SETZEN
2BD8 CD 55 16 CALL $1655    ; DEN BENÖTIGTEN PLATZ BESORGEN
2BD8 2A 4D 5C LD HL,(DEST)  ; ZEIGER ZURÜCKHOLEN
2BDE C1      POP BC         ; LANGE NACH BC HOLEN
2BDF C5      PUSH BC
2BE0 03      INC BC         ; +1 (FÜR NULLSTRING)
2BE1 ED B8   LDDR          ; STRING +1 BYTE KOPIEREN
2BE3 EB      EX DE,HL       ; HL AUF DIE BYTES DER
2BE4 23      INC HL         ; LÄNGENPOSITION SETZEN

```

```

2BE5 C1          POP BC          ; LANGE HOLEN
2BE6 70          LD (HL),B       ; LANGE HIGH UND
2BE7 2B          DEC HL
2BE8 71          LD (HL),C       ; LANGE LOW EINSCHREIBEN
2BE9 F1          POP AF         ; VARIABLENNAMEN ZURÜCKHOLEN
2BEA            ;
2BEA            ; SUBROUTINE ZUM EINSCHREIBEN DES ERSTEN ZEICHENS EINES
2BEA            ; VARIABLENNAMENS (ALTES ENDEBYTE MIT $80). HL ZEIGT
2BEA            ; AM ENDE AUF DIE NEUE ENDE($80)-POSITION
2BEA            ;
2BEA 2B          DEC HL          ; ZEIGER KORRIGIEREN
2BEB 77          LD (HL),A       ; NAMENSBYTE EINSCHREIBEN
2BEC 2A 59 5C    LD HL,(ELINE)   ; NEUE POSITION
2BEF 2B          DEC HL          ; DES ENDEBYTES NACH HL
2BF0 C9          RET
2BF1            ;
2BF1            ; SUBROUTINE ZUM LADEN DES LETZTEN EINTRAGS VOM
2BF1            ; CALCULATORSTACK. DIE WERTE KÖNNEN EINE VARIABLE
2BF1            ; ODER STRINPARAMETER SEIN
2BF1            ;
2BF1 2A 65 5C    LD HL,(STKEND)  ; ZEIGER AUF STACKENDE
2BF4 2B          DEC HL
2BF5 46          LD B,(HL)       ; DIE EINZELNEN BYTES
2BF6 2B          DEC HL
2BF7 4E          LD C,(HL)       ; NACHEINANDER IN DIE
2BF8 2B          DEC HL
2BF9 56          LD D,(HL)       ; REGISTER A - E LADEN
2BFA 2B          DEC HL
2BFB 5E          LD E,(HL)
2BFC 2B          DEC HL
2BFD 7E          LD A,(HL)
2BFE 22 65 5C    LD (STKEND),HL ; NEUES STACKENDE
2C01 C9          RET
2C02            .END
2C02            .LIB SPEC2C00-S
2C02            ; SINCLAIR ZX SPECTRUM TEIL 2C00
2C02            ;
2C02            ; BEFEHL DIM
2C02            ; DIESE ROUTINE DIENT ZUM ANLEGEN DER ARRAYS. WENN
2C02            ; BEREITS EIN ARRAY UNTER DEM GLEICHEN NAMEN EXISTIERT,
2C02            ; SO WIRD DAS ALTE ÜBERSCHRIEBEN. DAS GANZE ARRAY WIRD
2C02            ; BEIM ANLEGEN MIT 0 (NUMERISCH) ODER $20 (SPACE, BEI
2C02            ; STRINGS) BESCHRIEBEN
2C02            ;
2C02 CD B2 2B    CALL $28B2      ; IM VARIABLENBREICH SUCHEN
2C05 C2 8A 1C    JP NZ,$1C8A    ; SPRUNG BEI EINEM FEHLER
2C08 CD 30 25    CALL $2530      ; PROGRAMMLAUF ?
2C0B 20 0B       JR NZ,$2C15    ; JA
2C0D CB B1       RES 6,C        ; FÜR SYNTAXPRÜFUNG STRINGARRAYS
2C0F            ; ZU NORMALEN MACHEN
2C0F CD 96 29    CALL $2996      ; KLAMMERAUSDRUCK PRÜFEN
2C12 CD EE 1B    CALL $1BEE      ; ZUM NÄCHSTEN BEFEHL VORGEHEN
2C15 3B 0B       JR C,$2C1F     ; SPRUNG BEI NEUEM ARRAY
2C17 C5          PUSH BC        ; BESCHREIBUNGSBYTE RETTEN
2C18 CD B8 19    CALL $19B8      ; ANFANG DER NÄCHSTEN VARIABLEN
2C1B            ; SUCHEN
2C1B CD EB 19    CALL RAUS2      ; DAS ALTE ARRAY ENTFERNEN
2C1E C1          POP BC        ; BESCHREIBUNGSBYTE HOLEN

```

2C1F		SET 7,C	; BIT 7 BESCHREIBUNGSBYTE SETZEN
2C1F	CB F9	LD B,0	; DIMENSIONSZÄHLER = 0
2C21	06 00	PUSH BC	; ZWISCHENSPEICHERN
2C23	C5	LD HL,1	; ELEMENTGRÖSSE 1 FÜR STRINGS
2C24	21 01 00	BIT 6,C	; SPRUNG, WENN ES EIN
2C27	CB 71	JR NZ,\$2C2D	; STRINGARRAY IST
2C29	20 02	LD L,5	; SONST NUMERISCH: GRÖSSE = 5
2C2B	2E 05	EX DE,HL	; DIE ELEMENTGRÖSSE NACH DE
2C2D	EB	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
2C2E	E7	LD H,\$FF	; LIMIT FÜR ARRAYGRÖSSE SETZEN
2C2F	26 FF	CALL \$2ACC	; EIN ARRAYPARAMETER HOLEN
2C31	CD CC 2A	JP C,\$2A20	; ERROR, WENN ZU GROSS
2C34	DA 20 2A	POP HL	; DIMENSION HOLEN
2C37	E1	PUSH BC	; BEI JEDER SCHLEIFE DEN
2C38	C5		; PARAMETER ZWISCHENSPEICHERN
2C39		INC H	; DIMENSION +1
2C39	24	PUSH HL	; UND WIEDER AUF DEN STACK
2C3A	E5	LD H,B	; PARAMETER NACH HL KOPIEREN
2C3B	60	LD L,C	
2C3C	69	CALL \$2AF4	; GESAMTZAHL DER BYTES BERECHNEN
2C3D	CD F4 2A	EX DE,HL	; (DE * HL) UND WIEDER NACH DE
2C40	EB	RST GETAKT	; AKTUELLES ZEICHEN HOLEN
2C41	DF	CP ','	; KOMMA FÜR WEITERE PARAMETER ?
2C42	FE 2C	JR Z,\$2C2E	; JA
2C44	28 EB	CP ')'	; ES MUSS EINE KLAMMER SEIN
2C46	FE 29	JR NZ,\$2C05	; WENN NICHT: ERROR
2C48	20 BB	RST GETNXT	; CHADD +1
2C4A	E7	POP BC	; DIMENSIONSZÄHLER HOLEN
2C4B	C1	LD A,C	; BESCHREIBUNGSBYTE NACH A
2C4C	79	LD L,B	; DIMENSION NACH HL
2C4D	68	LD H,0	; BRINGEN
2C4E	26 00	INC HL	; DIMENSION +2
2C50	23	INC HL	
2C51	23	ADD HL,HL	; DANN MIT 2 MULTIPLIZIEREN
2C52	29	ADD HL,DE	; UND DIE BERECHNETE ARRAYGRÖSSE
2C53	19		; ADDIEREN, UM DEN INSGESAMT NOT-
2C54			; WENDIGEN SPEICHERBEDARF ZU ER-
2C54			; HALTEN (=ARRAYLÄNGE GESAMT)
2C54	DA 15 1F	JP C,\$1F15	; ARRAY ZU GROSS: ERROR
2C57	D5	PUSH DE	; ANZAHL DER ELEMENTE,
2C58	C5	PUSH BC	; BESCHREIBUNGSBYTE UND
2C59	E5	PUSH HL	; GESAMTGRÖSSE RETTEN
2C5A	44	LD B,H	; GESAMTGRÖSSE NACH BC
2C5B	4D	LD C,L	; KOPIEREN
2C5C	2A 59 5C	LD HL,(ELINE)	; ENDE DES VARIABLENBEREICHS
2C5F	2B	DEC HL	; BESTIMMEN (\$80 PLATZ)
2C60	CD 55 16	CALL \$1655	; DEN SPEICHERBEREICH FREIMACHEN
2C63	23	INC HL	; AUF ERSTE NEUE POSITION SETZEN
2C64	77	LD (HL),A	; DEN MARKIERTEN BUCHSTABEN DES
2C65			; NAMENS EINSCHREIBEN
2C65	C1	POP BC	; GESAMTLÄNGE HOLEN UND
2C66	0B	DEC BC	
2C67	0B	DEC BC	; 3 SUBTRAHIEREN
2C68	0B	DEC BC	
2C69	23	INC HL	
2C6A	71	LD (HL),C	; LÄNGE LOW UND
2C6B	23	INC HL	

2C6C	70	LD (HL),B	; LANGE HIGH EINTRAGEN
2C6D	C1	POP BC	; DIMENSIONSZÄHLER HOLEN
2C6E	78	LD A,B	; UND NACH A KOPIEREN
2C6F	23	INC HL	
2C70	77	LD (HL),A	; DIMENSION EINTRAGEN
2C71	62	LD H,D	; HL AUF LETZTEN PLATZ
2C72	68	LD L,E	; DES ARRAYS SETZEN
2C73	1B	DEC DE	; DE AUF VORLETZTEN
2C74	36 00	LD (HL),0	; LETZTER PLATZ MIT 0
2C76	CB 71	BIT 6,C	; ODER, WENN STRINGARRAY,
2C78	28 02	JR Z,\$2C7C	
2C7A	36 20	LD (HL),	; MIT LEERZEICHEN BESCHREIBEN
2C7C	C1	POP BC	; ELEMENTZÄHL HOLEN UND
2C7D	ED 88	LDDR	; ARRAY +1 PLATZ LÖSCHEN
2C7F	C1	POP BC	; EIN DIMENSIONSBYTE HOLEN
2C80	70	LD (HL),B	; UND DIESES, ERST HIGH,
2C81	2B	DEC HL	
2C82	71	LD (HL),C	; DANN LOW, EINSCHREIBEN
2C83	2B	DEC HL	
2C84	3D	DEC A	; DIMENSIONSZÄHLER -1
2C85	20 FB	JR NZ,\$2C7F	; NOCH WAS EINGETRAGEN
2C87	C9	RET	
2C88			
2C88		; SUBROUTINE ZUM PRÜFEN AUF ALPHANUMERISCH. CARRY IST	
2C88		; GESETZT BEI BUCHSTABEN UND ZIFFERN	
2C88			
2C88	CD 1B 2D	CALL ZIFFER	; AUF ZIFFERN PRÜFEN
2C88	3F	CCF	
2C8C	DB	RET C	; BEI ZIFFER RETURN
2C8D			
2C8D		; SUBROUTINE ÜBERPRÜFT AUF BUCHSTABEN. CARRY IST BEI	
2C8D		; BUCHSTABEN GESETZT	
2C8D			
2C8D	FE 41	CP 'A'	; GROSSBUCHSTABEN GEHEN VON
2C8F	3F	CCF	; \$41 BIS
2C90	D0	RET NC	
2C91	FE 5B	CP \$5B	; \$5B (=Z +1)
2C93	DB	RET C	
2C94	FE 61	CP 'a'	; KLEINBUCHSTABEN ENTSPRECHEND
2C96	3F	CCF	; VON \$61 BIS
2C97	D0	RET NC	
2C98	FE 7B	CP \$7B	; \$7B (=z+1)
2C9A	C9	RET	
2C9B			
2C9B		; SUBROUTINE ZUM WANDELN VON DEZIMALZAHLEN ODER, MIT	
2C9B		; DEM ZUSATZ 'BIN', BINÄRZAHLEN IN FLOATINGPOINTZAHLEN,	
2C9B		; DIE DANN ALS LETZTES ERGEBNIS AUF DEM CALC.-STACK	
2C9B		; ABGELEGT WERDEN	
2C9B			
2C9B	FE C4	DEZFLO CP \$C4	; IST DAS ZEICHEN 'BIN' ?
2C9D	20 19	JR NZ,\$2C9B	; NEIN
2C9F	11 00 00	LD DE,0	; ERGEBNIS MIT 0 VORBESETZEN
2CA2	E7	BINFLO RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
2CA3	D6 31	SUB '1'	; EINE 1 SUBTRAHIEREN: 1 ERGIBT
2CA5	CE 00	ADC 0	; NACH DER ADDITION EINE 0 MIT
2CA7			; MIT CARRY GELÖSCHT, EINE 0
2CA7			; ERGIBT 0, ABER CARRY GESETZT
2CA7	20 0A	JR NZ,\$2C93	; BEI ALLEN ANDEREN ZEICHEN

2CA9	EB	EX DE,HL	; BISHERIGES ERGEBNIS NACH HL
2CAA	3F	CCF	; CARRY INVERTIEREN: FÜR 1 IST
2CAB			; CARRY JETZT GESETZT, 0 NICHT
2CAB	ED 6A	ADC HL,HL	; VORHERIGES ERGEBNIS 1-STELLE
2CAD			; NACH LINKS SCHIEBEN UND CARRY
2CAD			; IN BIT 0 ADDIEREN
2CAD	DA AD 31	JP C,\$31AD	; 65536: ERROR
2CB0	EB	EX DE,HL	; ERGEBNIS ZWISCHENSPEICHERN
2CB1	18 EF	JR BINFLO	; UND WEITERE ZEICHEN WANDELEN
2CB3			
2CB3	42	LD B,D	; ERGEBNIS NACH BC KOPIEREN
2CB4	4B	LD C,E	
2CB5	C3 2B 2D	JP \$2D2B	; UND AUF DEM STACK ABLEGEN
2CB8			
2CB8			; BEI DEZIMALZAHLEN ERST DEN GANZZAHLIGEN TEIL
2CB8			; BERÜCKSICHTIGEN
2CB8			
2CB8	FE 2E	CP .	; ERSTES ZEICHEN = DEZIMALPUNKT
2CBA	2B 0F	JR Z,\$2CCB	; KEIN GANZANTEIL
2CBC	CD 3B 2D	CALL \$2D3B	; TEIL VOR DEM PUNKT ALS LETZTEN
2CBF			; WERT AUF DEN CALC.-STACK
2CBF	FE 2E	CP .	; FOLGT NACHKOMMATEIL ?
2CC1	2D 2B	JR NZ,\$2CEB	; NEIN: EXPONENT UNTERSUCHEN
2CC3	E7	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
2CC4	CD 1B 2D	CALL ZIFFER	; IST ES EINE ZIFFER ?
2CC7	3B 22	JR C,\$2CEB	; NEIN: EXPONENT UNTERSUCHEN
2CC9	1B 0A	JR \$2CD5	; NACHKOMMATEIL BEARBEITEN
2CCB			
2CCB			; ZAHLEN, DIE MIT PUNKT ANFANGEN, BEARBEITEN
2CCB			
2CCB	E7	RST GETNXT	; CHADD +1, NÄCHSTES ZEICHEN
2CCC	CD 1B 2D	CALL ZIFFER	; IST ES EINE ZAHL ?
2CCF	DA 8A 1C	JP C,\$1CBA	; NEIN: ERROR
2CD2	EF	RST CALRUF	; CALCULATORAUFRUF
2CD3	A0	.BYT \$A0	; NULL FÜR DIE ZAHLEN OHNE
2CD4	3B	.BYT \$3B	; GANZZAHLIGEN ANTEIL SPEICHERN
2CD5			
2CD5	EF	RST CALRUF	
2CD6	A1	.BYT \$A1	; 1 IM CALC.-STACK SPEICHERN
2CD7	C0	.BYT \$C0	; NACH MEMO KOPIEREN
2CD8	02	.BYT \$02	; DIE 1 IM CALC.-STACK LÖSCHEN
2CD9	3B	.BYT \$3B	; ENDE (DER LETZTE WERT IM STACK
2CDA			; IST JETZT EINE 0 (OHNE TEIL
2CDA			; VOR DEM KOMMA) ODER DER GANZ-
2CDA			; ZAHLIGE ANTEIL)
2CDA	DF	RST GETAKT	; AKTUELLES ZEICHEN LADEN UND
2CDB	CD 22 2D	CALL \$2D22	; AUF ZIFFER TESTEN UND DANN
2CDE			; AUF CALC.-STACK SPEICHERN
2CDE	3B 0B	JR C,\$2CEB	; NICHT: EXPONENT UNTERSUCHEN
2CE0	EF	RST CALRUF	; CALCULATOR AUFRUFEN
2CE1	E0	.BYT \$E0	; MEMO HOLEN
2CE2	A4	.BYT \$A4	; ZEHN AUF STACK SPEICHERN
2CE3	05	.BYT \$05	; MEMO/10 BILDEN: 1 WIRD .1,
2CE4			; DIES ZU .01 USW
2CE4	C0	.BYT \$C0	; ERGEBNIS NACH MEMO ZURÜCK
2CE5	04	.BYT \$04	; EINGELESENE ZAHL MIT (MEMO)
2CE6			; MULTIPLIZIEREN, ALSO STELLEN-
2CE6			; WERT NACH DEM KOMMA


```

2CE6 0F          .BYT $0F          ; ZUM BISHERIGEN ERGEBNIS
2CE7 38          .BYT $38          ; ADDIEREN (LETZTER WERT), ENDE
2CE8 E7          RST GETNXT        ; CHADD +1
2CE9 18 EF      JR $2CDA          ; WEITER IN DER SCHLEIFE
2CEB            ;
2CEB            ; EXPONENT UNTERSUCHEN
2CEB            ;
2CEB FE 45      CP 'E'            ; FOLGT EIN 'E'
2CED 28 03      JR Z,$2CF2
2CEF FE 65      CP 'e'            ; ODER EIN 'e' FÜR EXPONENT ?
2CF1 C0         RET NZ            ; NEIN
2CF2            ;
2CF2 06 FF      LD B,$FF          ; B IST VORZEICHENFLAG: $FF= '+'
2CF4 E7         RST GETNXT        ; CHADD +1, NÄCHSTES ZEICHEN
2CF5 FE 2B      CP '+'            ; POSITIVER EXPONENT ?
2CF7 28 05      JR Z,$2CFE        ; JA
2CF9 FE 2D      CP '-'            ; NEGATIVER EXPONENT ?
2CFB 20 02      JR NZ,$2CFF       ; NEIN: KEINE ANGABE: '+'
2CFD 04         INC B             ; B = 0 FÜR NEGATIV
2CFE E7         RST GETNXT        ; CHADD +1, NÄCHSTES ZEICHEN
2CFF CD 1B 2D   CALL ZIFFER       ; FOLGT DEZIMALZAHL ?
2D02 38 CB      JR C,$2CCF       ; NEIN: ERROR
2D04 C5         PUSH BC           ; VORZEICHENFLAG RETTEN
2D05 CD 3B 2D   CALL $2D3B        ; EXPONENT AUF STACK SPEICHERN
2D0B CD D5 2D   CALL $2DD5        ; EXPONENT NACH REG A HOLEN
2D0B C1         POP BC           ; VORZEICHENFLAG ZURÜCKHOLEN
2D0C DA AD 31   JP C,$31AD        ; EXPONENT > 255: ERROR
2D0F A7         AND A             ; EXPONENT > 127 ?
2D10 FA AD 31   JP M,$31AD        ; JA: ERROR
2D13 04         INC B             ; NEGATIVER EXPONENT ?
2D14 28 02      JR Z,$2D1B       ; NEIN
2D16 ED 44      NEG              ; JA: 2-KOMPLEMENT BILDEN
2D18 C3 4F 2D   JP $2D4F         ; EXPONENT ZUR ZAHL HINZUFÜGEN
2D1B            ;
2D1B            ; SUBROUTINE ZUM UNTERSUCHEN AUF ZIFFERN. BEI ZIFFERN
2D1B            ; WIRD DAS CARRYFLAG GESETZT
2D1B            ;
2D1B FE 30      ZIFFER CP '0'
2D1D D8         RET C
2D1E FE 3A      CP ':'            ; ASCIIWERT VON '9' +1
2D20 3F         CCF
2D21 C9         RET
2D22            ;
2D22            ; SUBROUTINE ZUM ABLEGEN EINES DIBITS (ZIFFER) AUF
2D22            ; DEM CALC.-STACK ALS LETZTER WERT
2D22            ;
2D22 CD 1B 2D   CALL ZIFFER       ; WENN KEINE ZIFFER,
2D25 D8         RET C             ; DANN DIREKT RETURN
2D26 D6 30      SUB '0'          ; $30 FÜR WANDLUNG SUBTRAHIEREN
2D28            ;
2D28            ; SUBROUTINE LEGT BINÄRZAHL IN A ALS FLOATINGPOINT-
2D28            ; ZAHL AUF DEM CALC.-STACK ALS LETZTEN WERT AB
2D28            ;
2D28 4F         LD C,A            ; ZAHL NACH C KOPIEREN
2D29 06 00      LD B,0           ; UND B ZU NULL SETZEN
2D2B            ;
2D2B            ; SUBROUTINE ZUM ABLEGEN DER INTEGERZAHL IN BC AUF DEM
2D2B            ; CALC.-STACK ALS LETZTER WERT IN FLOATINGPOINTFORMAT.

```

```

2D2B      ; DAS 1. UND 5. BYTE SIND IMMER 0, DAS 2. GIBT DAS
2D2B      ; VORZEICHEN AN: 0 = POSITIV, $FF = NEGATIV. DAS 3.
2D2B      ; BYTE IST DAS LOWBYTE UND DAS 4. DAS HIGHBYTE. DIE
2D2B      ; ABLAGE ERFOLGT IM 2-KOMPLEMENT
2D2B      ;
2D2B FD 21 3A 5C      LD IY,ERRNR      ; IY NOCH EINMAL INITIALISIEREN
2D2F AF              XOR A              ; REG A = 0
2D30 5F              LD E,A              ; REG E EBENFALLS FÜR POSITIV
2D31 51              LD D,C              ; LOWBYTE NACH D
2D32 48              LD C,B              ; HIGHBYTE NACH C
2D33 47              LD B,A              ; REG B = 0
2D34 CD B6 2A        CALL $2AB6          ; DIE ZAHL AUF DEM STACK ABLEGEN
2D37 EF              RST CALRUF          ; DER CALCULATORAUFRUF DIENT NUR
2D38 3B              .BYT $3B            ; ZUM SETZEN VON HL AUF STKEND-5
2D39 A7              AND A              ; CARRY LÖSCHEN
2D3A C9              RET
2D3B      ;
2D3B      ; SUBROUTINE ZUM EINLESEN GANZZÄHLIGER DEZIMALZAHLEN
2D3B      ; ALS LETZTER WERT AUF DEM CALC.-STACK. AUFRUF ERFOLGT
2D3B      ; Z. B. BEIM EINLESEN EINER DEZIMAL AUS EINER BASICZEILE
2D3B      ; (SIEHE DEZFLO)
2D3B      ;
2D3B F5              PUSH AF              ; ERSTE ZIFFER RETTEN
2D3C EF              RST CALRUF          ; CALCULATOR AUFRUFEN, UM
2D3D A0              .BYT $A0            ; DEN LETZTEN WERT MIT 0
2D3E 3B              .BYT $3B            ; VORZUBESETZEN
2D3F F1              POP AF              ; ERSTE ZIFFER HOLEN
2D40 CD 22 2D        CALL $2D22          ; DIE ZIFFER AUF DEM CALC.-STACK
2D43              ; ALS LETZTER WERT ABLEGEN
2D43 DB              RET C              ; WENN KEINE ZIFFER: RETURN
2D44 EF              RST CALRUF          ; CALCULATOR AUFRUFEN
2D45 01              .BYT $01            ; TAUSCHEN: ZIFFER JETZT VOR-
2D46              ; LETZTER WERT, DAMIT DAS VOR-
2D46              ; HERIGE ERGEBNIS MIT 10
2D46              ; MULTIPLIZIERT WERDEN KANN
2D46 A4              .BYT $A4            ; 10 AUF CALC.-STACK
2D47 04              .BYT $04            ; VORHERIGES ERGEBNIS*10
2D48 0F              .BYT $0F            ; ZUR ZIFFER ADDIEREN
2D49 3B              .BYT $3B            ; ENDE
2D4A CD 74 00        CALL $74            ; DAS NÄCHSTE ZEICHEN EINLESEN
2D4D 1B F1           JR $2D40            ; UND WANDELN
2D4F      ;
2D4F      ; ARITHMETISCHE ROUTINEN
2D4F      ;
2D4F      ; WANDLUNG VON DEZIMALZAHLEN MIT MANTISSE/EXPONENT-
2D4F      ; DARSTELLUNG (xEn) IN EINE FLOATINGPOINTZAHL. X STEHT
2D4F      ; ALS LETZTER WERT BEREITS AUF DEM CALC.-STACK
2D4F      ;
2D4F 07              RLCA                ; FÜR VORZEICHENTEST BIT7 VON A
2D50 0F              RRCA                ; INS CARRY SCHIEBEN, OHNE A ZU
2D51              ; VERÄNDERN
2D51 30 02           JR NC,$2D55          ; BEI POSITIVEM EXPONENT
2D53 2F              CPL                 ; NEG. EXP.: 2-KOMPLEMENT VON A
2D54 3C              INC A               ; BILDEN OHNE CARRY ZU VERÄNDERN
2D55 F5              PUSH AF             ; A ZWISCHENSPEICHERN
2D56 21 92 5C        LD HL,MEMBOT        ; VORZEICHENFLAG IM ERSTEN BYTE
2D59 CD 0B 35        CALL $350B          ; VON MEMO BILDEN: 0=POSITIV,
2D5C              ; 1=NEGATIV

```

2D5C	EF	RST CALRUF	; X STEHT IM CALC.-STACK
2D5D	A4	.BYT \$A4	; X, 10, ZAHL 10 SPEICHERN
2D5E	38	.BYT \$38	; ENDE
2D5F	F1	POP AF	; EXPONENT HOLEN UND DIE EIN-
2D60	08 3F	EXSCHL SRL A	; ZELNEN BITS INS CARRY SCHIEBEN
2D62	30 0D	JR NC, \$2D71	; BIT WAR 0: REG A AUF 0 TESTEN
2D64	F5	PUSH AF	; RESTLICHER EXPONENTWERT RETTEN
2D65	EF	RST CALRUF	; $X', 10^{2^M}$, $M = 0..5$, IM STACK,
2D66			; WOBEI X' EIN ZWISCHENWERT DER
2D66			; MANTISSE BEI DEN MULTIPLIKA-
2D66			; TIONEN MIT 10^N BEDEUTET
2D66	01	.BYT \$01	; 10^{2^M} NACH MEM1 KOPIEREN
2D67	E0	.BYT \$E0	; $X', 10^{2^M}$, V(ORZEICHEN) AUS MEMO
2D68	00	.BYT \$00	; $X', 10^{2^M}$, V=NEGATIV (1):
2D69	04	.BYT \$04	; SPRUNG NACH EXDIV
2D6A	04	.BYT \$04	; $X' * 10^{2^M}$, MULTIPLIKATION
2D6B	33	.BYT \$33	; X' (NEU), IMMER SPRUNG
2D6C	02	.BYT \$02	; NACH EXGET1
2D6D	05	EXDIV .BYT \$05	; $X' / (10^{2^M})$, DIVISION
2D6E	E1	EXGET1 .BYT \$E1	; X' (NEU), 10^{2^M} , MEMO HOLEN
2D6F	38	.BYT \$38	; ENDE
2D70	F1	POP AF	; RESTEXPONENT UND ZEROFLAG
2D71	28 08	JR Z, \$2D7B	; SPRUNG, WENN RESTEXPONENT 0
2D73	F5	PUSH AF	; RESTEXPONENT RETTEN
2D74	EF	RST CALRUF	; $X', 10^{2^M}$
2D75	31	.BYT \$31	; $X', 10^{2^M}, 10^{2^M}$, VERDOPPELN
2D76	04	.BYT \$04	; $X', 10^{2^M(M+1)}$, MULTIPLIZIEREN
2D77	38	.BYT \$38	; ERGIBT $X', 10^{2^M}$ FÜR SCHLEIFE
2D78	F1	POP AF	; RESTEXPONENT ZURÜCKHOLEN UND
2D79	18 E5	JR EXSCHL	; IN DER SCHLEIFE WEITER
2D78	EF	RST CALRUF	; $X', 10^{2^M}$
2D7C	02	.BYT \$02	; $X = X' * 10^N$, LÖSCHEN
2D7D	38	.BYT \$38	; X, ENDE
2D7E	C9	RET	
2D7F			; SUBROUTINE ZUM HOLEN EINER INTEGERZAHL VOM CALC.-STACK
2D7F			;
2D7F	23	INC HL	; ADRESSIERT JETZT DAS VOR-
2D80	4E	LD C, (HL)	; ZEICHEN, DIESES NACH C LADEN
2D81			; \$FF BEDEUTET NEGATIVE ZAHL
2D81	23	INC HL	; LOWBYTE ADRESSIEREN
2D82	7E	LD A, (HL)	; LOWBYTE LADEN UND DAS
2D83	A9	XOR C	; 1-KOMPLEMENT BILDEN, WENN NEG.
2D84	91	SUB C	; NEG.: 1 ADDIEREN: 2-KOMPLEMENT
2D85			; BILDEN
2D85	5F	LD E, A	; LOWBYTE NACH E
2D86	23	INC HL	; AUF HIGHTBYTE ZEIGEN
2D87	7E	LD A, (HL)	; HIGHTBYTE LADEN
2D88	89	ADC C	; UND, WENN NEGATIV, DAS
2D89	A9	XOR C	; 2-KOMPLEMENT BILDEN
2D8A	57	LD D, A	; HIGHBYTE NACH D
2D8B	C9	RET	
2D8C			; SUBROUTINE ZUM ABSPEICHERN EINER INTEGERZAHL AUF
2D8C			; DEM CALC.-STACK (= GEGENSTÜCK ZU OBIGER ROUTINE)
2D8C			;
2D8C	0E 00	LD C, 0	; EINSTIEG FÜR POSITIVE ZAHLEN

2D8E	E5	PUSH HL	; EINSTIEG MIT VORZEICHEN IN C:
2D8F			; \$FF = NEGATIV, 0 = POSITIV
2D8F	36 00	LD (HL),0	; ERSTES BYTE LÖSCHEN
2D91	23	INC HL	; DAS VORZEICHEN IN
2D92	71	LD (HL),C	; DAS ZWEITE BYTE SCHREIBEN
2D93	23	INC HL	; FÜR NEGATIVE ZAHLEN WIRD AS
2D94	7B	LD A,E	; 2-KOMPLEMENT GEBILDET
2D95	A9	XOR C	; ZUERST FÜR DAS LOWBYTE
2D96	91	SUB C	
2D97	77	LD (HL),A	; LOWBYTE SPEICHERN
2D98	23	INC HL	
2D99	7A	LD A,D	; HIGHBYTE BEARBEITEN
2D9A	89	ADC C	
2D9B	A9	XOR C	
2D9C	77	LD (HL),A	; UND SPEICHERN
2D9D	23	INC HL	
2D9E	36 00	LD (HL),0	; 5. BYTE LÖSCHEN
2DA0	E1	POP HL	; ZEIGER AUF ERSTES BYTE ZURÜCK
2DA1	C9	RET	
2DA2			
2DA2			; SUBROUTINE ZUM WANDELN EINER FLOATINGPOINTZahl IN
2DA2			; EINE INTEGERZahl UND ÜBERTRAGEN DES ERGEBNISSES IN BC
2DA2			
2DA2	EF	RST CALRUF	; HL AUF (STKEND)-5 SETZEN
2DA3	38	.BYT \$38	; (=1. BYTE LETZTER WERT)
2DA4	7E	LD A,(HL)	; EXPONENT LADEN
2DA5	A7	AND A	; FALLS DIESER NULL IST, IST ES
2DA6	28 05	JR Z,\$2DAD	; BEREITS EINE INTEGERZahl
2DA8			; SONST FP-Zahl WANDELN:
2DA8	EF	RST CALRUF	; CALCULATORAUFRUF
2DA9	A2	.BYT \$A2	; .5 AUF CALC.-STACK
2DAA	0F	.BYT \$0F	; ADDIEREN
2DAB	27	.BYT \$27	; INTEGER BILDEN:
2DAC	38	.BYT \$38	; -65536<X<65536
2DAD	EF	RST CALRUF	
2DAE	02	.BYT \$02	; ERGEBNIS LÖSCHEN (ES WIRD NUR
2DAF	38	.BYT \$38	; EIN ZEIGER VERÄNDERT)
2DB0	E5	PUSH HL	; DIE BEIDEN ZEIGER IN
2DB1	D5	PUSH DE	; DEN CALC.-STACK RETTEN
2DB2	EB	EX DE,HL	; HL ZEIGT AUF DIE Zahl
2DB3	46	LD B,(HL)	; EXPONENT (MUSS 0 SEIN) LADEN
2DB4	CD 7F 2D	CALL \$2D7F	; VORZEICHEN + 2 BYTES HOLEN
2DB7	AF	XOR A	; REG A UND CARRY LÖSCHEN
2DB8	90	SUB B	; CARRY WIRD GESETZT (=INT. ZU
2DB9			; GROSS), WENN EXPONENT (>0
2DB9	CB 79	BIT 7,C	; ZEROFLAG SETZEN FÜR POSITIV
2DBB	42	LD B,D	; HIGHBYTE NACH B
2DBC	4B	LD C,E	; LOWBYTE NACH C
2DBD	7B	LD A,E	; HIGHBYTE ZUSÄTZLICH IN A
2DBE	D1	POP DE	; DIE 2 ZEIGER IN DEN CALC.-
2DBF	E1	POP HL	; STACK ZURÜCKHOLEN
2DC0	C9	RET	
2DC1			
2DC1			; SUBROUTINE ZUM BERECHNEN VON LOG(2^A). A ENTHÄLT DEN
2DC1			; EXPONENT EINER FLOATINGPOINTZahl. DIE BERECHNUNG DIENT
2DC1			; ZUM BESTIMMEN DER VORKOMMSTELLEN EINER AUSZUGEBENDEN
2DC1			; DEZIMALZahl ODER DER AUF DEN DEZIMALPUNKT FOLGENDEN
2DC1			; NULLEN

```

2DC1
2DC1 57      LOG2A LD D,A      ; A WIRD SO BEARBEITET, DASS AUF
2DC2 17      RLA          ; DEM CALC.-STACK, WENN A
2DC3 9F      SBC A        ; POSITIV, $00,$00,A,$00,$00
2DC4 5F      LD E,A       ; ODER, WENN NEGATIV,
2DC5 4F      LD C,A       ; $00,$FF,A,$FF,$00, ABGELEGT
2DC6 AF      XOR A        ; WIRD. DIE WERTE WERDEN IN DIE
2DC7 47      LD B,A       ; ENTSPRECHENDEN REGISTER FÜR
2DC8 CD B6 2A CALL $2AB6   ; DEN SUBROUTINENAUFRAF GELADEN
2DC8 EF      RST CALRUF   ; A,
2DCC 34      .BYT $34     ; A, DATEN SPEICHERN:
2DCD EF      .BYT $EF     ; LOG2 ZUR BASIS 10
2DCE 1A      .BYT $1A,$20,$9A,$85 ; A,LOG2
2DCF 20
2DD0 9A
2DD1 85
2DD2 04      .BYT $04     ; A*LOG2 (=LOG(2^A))
2DD3 27      .BYT $27     ; INTEGER BILDEN
2DD4 38      .BYT $38     ; INT(LOG(2^A))
2DD5
2DD5 ; FLOATINGPOINTZAHL IN EINE INTEGERZAHL VON EINEM BYTE
2DD5 ; WANDELN, ERGEBNIS IN REG A. WENN DAS ERGEBNIS >255
2DD5 ; IST, ERFOLGT EINE ERRORMELDUNG
2DD5
2DD5 CD A2 2D CALL $2DA2   ; FLOATINGPOINTZAHL ALS INTEGER
2DD8        ; NACH BC BRINGEN
2DD8 D8      RET C        ; ERGEBNIS ZU GROSS (>65536)
2DD9 F5      PUSH AF      ; ERGEBNIS ZWISCHENSPEICHERN
2DDA 05      DEC B        ; HIGHBYTE MUSS NULL SEIN
2DDB 04      INC B
2DDC 28 03   JR Z,$2DE1   ; JA: ERGEBNIS <256
2DDE F1      POP AF       ; SONST ERROR: ERGEBNIS ZURÜCK-
2DDF 37      SCF          ; HOLEN UND DAS CARRYFLAG SETZEN
2DE0 C9      RET
2DE1 F1      POP AF       ; ERGEBNIS ZURÜCK IN A, RETURN
2DE2 C9      RET          ; MIT CARRY GELOESCHT: KEIN ERROR
2DE3
2DE3 ; SUBROUTINE ZUM AUSGEBEN EINER FLOATINGPOINTZAHL DURCH
2DE3 ; 'PRINT'- ODER 'STR$'-BEFEHL
2DE3
2DE3 EF      RST CALRUF   ; X,
2DE4 31      .BYT $31     ; X,X, VERDOPPELN
2DE5 36      .BYT $36     ; X, X<0?
2DE6 00      .BYT $00     ; X, JA: X IST NEGATIV:
2DE7 0B      .BYT $0B     ; SPRUNG NACH FPNEGA
2DE8 31      .BYT $31     ; X,X, VERDOPPELN
2DE9 37      .BYT $37     ; X, X>0?
2DEA 00      .BYT $00     ; X, JA: SPRUNG NACH
2DEB 0D      .BYT $0D     ; FPPOSI
2DEC 02      .BYT $02     ; - LÖSCHEN
2DED 38      .BYT $38     ; ENDE
2DEE 3E 30   LD A,'0'     ; DIE ZAHL WAR NULL
2DF0 D7      RST PRTOU    ; DIESE AUSGEBEN UND FERTIG
2DF1 C9      RET
2DF2
2DF2 ; FÜR NEGATIVE ZAHLEN ERST EIN MINUSZEICHEN AUSGEBEN
2DF2 ; UND DANN ABS(X) BILDEN, SODASS DIE ZAHL IM WEITEREN
2DF2 ; WIE EINE POSITIVE BEHANDELT WERDEN KANN

```



```

2DF2      ;
2DF2 2A    FPNEGA .BYT $2A      ; ABS(X) BILDEN
2DF3 38    .BYT $38            ; ENDE
2DF4 3E 2D LD A, '-'           ; MINUSZEICHEN
2DF6 D7    RST PRTOU          ; AUSGEBEN
2DF7 EF    RST CALRUF         ; CALCULATOR WIEDER AUFRUFEN
2DF8      ;
2DF8      ; X IST IM FOLGENDEN ABS(X)
2DF8      ;
2DF8 A0    FPPOSI .BYT $A0      ; X, 0, NULL AUF DEN STACK
2DF9 C3    .BYT $C3            ; MEM3,
2DFA C4    .BYT $C4            ; MEM4 UND
2DFB C5    .BYT $C5            ; MEM5 MIT 0 BESCHREIBEN
2DFC 02    .BYT $02            ; X, NULL WIEDER ENTFERNEN
2DFD 38    .BYT $38            ; ENDE
2DFE D9    EXX                ; HL', DAS DIE CALCULATOROFFSETS
2DFF E5    PUSH HL            ; ENTHALT, ZWISCHENSPEICHERN
2E00 D9    EXX
2E01 EF    RST CALRUF         ; X, CALCULATORAUFRUF
2E02 31    .BYT $31            ; X,X, VERDOPPELN
2E03 27    .BYT $27            ; X,INT(X)=I
2E04 C2    .BYT $C2            ; X, I NACH MEM2 SPEICHERN
2E05 03    .BYT $03            ; X-I=F, SUBTRAHIEREN
2E06 E2    .BYT $E2            ; F,I, MEM2 HOLEN
2E07 01    .BYT $01            ; I,F, TAUSCHEN
2E08 C2    .BYT $C2            ; I,F, IN MEM2 SPEICHERN
2E09 02    .BYT $02            ; I, F LÖSCHEN
2E0A 38    .BYT $38            ; ENDE
2E0B 7E    LD A,(HL)          ; TEST, OB I < 65536 IST
2E0C A7    AND A
2E0D 20 47 JR NZ,$2E56         ; NEIN
2E0F CD 7F 2D CALL $2D7F       ; I NACH DE LADEN
2E12 06 10 LD B,16             ; ZÄHLER FÜR 16 BITS
2E14 7A    LD A,D              ; HIGHBYTE VON I AUF 0
2E15 A7    AND A               ; TESTEN
2E16 20 06 JR NZ,$2E1E         ; NEIN: >255
2E18 B3    OR E                ; IST I GANZ NULL?
2E19 28 09 JR Z,$2E24          ; JA: NUR NACHKOMMASTELLEN
2E1B 53    LD D,E              ; LOWBYTE NACH D KOPIEREN
2E1C 06 08 LD B,8              ; ZÄHLER FÜR 8 BITS
2E1E D5    PUSH DE             ; DE ÜBER DEN STACK
2E1F D9    EXX                 ; NACH
2E20 D1    POP DE              ; DE KOPIEREN
2E21 D9    EXX                 ; REGS WIEDER NORMAL
2E22 18 57 JR $2E7B
2E24      ;
2E24      ; BEARBEITUNG, WENN NUR EIN NACHKOMMASTEIL VORHANDEN
2E24      ;
2E24 EF    RST CALRUF         ; I(=0),
2E25 E2    .BYT $E2            ; I,F, MEM2 HOLEN
2E26 38    .BYT $38            ; ENDE
2E27 7E    LD A,(HL)          ; EXPONENTENBYTE NACH A LADEN
2E28 D6 7E SUB $7E            ; UM DEN ECHTEN EXPONENTEN ZU
2E2A      ; ERHALTEN, MUSS $7E (=126)
2E2A      ; SUBTRAHIERT WERDEN
2E2A CD C1 2D CALL LOG2A        ; M=A=ABS INT(LOG(2^A))
2E2D 57    LD D,A              ; M NACH D KOPIEREN
2E2E 3A AC 5C LD A,($5CAC)     ; 2. BYTE VON MEM5 LADEN

```

2E31	92	SUB D	; DAVON M SUBTRAHIEREN UND
2E32	32 AC 5C	LD (\$5CAC),A	; ZURÜCKSCHREIBEN
2E35	7A	LD A,D	; M NACH A KOPIEREN
2E36	CD 4F 2D	CALL \$2D4F	; $Y=F*10^M$ AUF DEN STACK BRINGEN
2E39	EF	RST CALRUF	; I,Y
2E3A	31	.BYT \$31	; I,Y,Y, VERDOPPELN
2E3B	27	.BYT \$27	; I,Y,INT(Y)=J,
2E3C	C1	.BYT \$C1	; I,Y,J, J NACH MEM1 SPEICHERN
2E3D	03	.BYT \$03	; I,Y-J
2E3E	E1	.BYT \$E1	; I,Y-J,J, MEM1 LADEN
2E3F	38	.BYT \$38	; I,Y-J,J, ENDE
2E40	CD D5 2D	CALL \$2DD5	; J VOM CALC.-STACK NACH A HOLEN
2E43	E5	PUSH HL	; ZEIGER AUF (Y-J) RETTEN
2E44	32 A1 5C	LD (\$5CA1),A	; ADRESSE 1. BYTE MEM3
2E47	3D	DEC A	; J (IN A) SO BEARBEITEN, DASS
2E48	17	RLA	; EINE NULL ERHALTEN BLEIBT, JEDE
2E49	9F	SBC A	; ANDERE ZAHL JEDOCH 1 ERGIBT
2E4A	3C	INC A	; (DIENT ALS ZÄHLER)
2E4B	21 AB 5C	LD HL,\$5CAB	; ADRESSE 1. BYTE VON MEM5
2E4E	77	LD (HL),A	; ZÄHLER EINSCHREIBEN
2E4F	23	INC HL	; DIESE ZAHL ZU DEN AUSZUGE-
2E50	86	ADD (HL)	; DEN STELLEN VOR DEM KOMMA
2E51	77	LD (HL),A	; ADDIEREN
2E52	E1	POP HL	; ZEIGER AUF (Y-J) ZURÜCKHOLEN
2E53	C3 CF 2E	JP \$2ECF	
2E56			
2E56			; ZAHLEN, DIE GRÖßER ALS 2^{27} SIND, WERDEN SO BEARBEI-
2E56			; TET, DASS 8 STELLEN VOR DEM KOMMA AUSGEGEBEN WERDEN
2E56			
2E56	D6 80	SUB \$80	; DEN ECHTEN EXPONENT BERECHNEN
2E58	FE 1C	CP 28	; KLEINER ALS 28 ?
2E5A	38 13	JR C,\$2E6F	; JA
2E5C	CD C1 2D	CALL LOG2A	; M IN A BERECHNEN
2E5F	D6 07	SUB 7	; M-7 BILDEN
2E61	47	LD B,A	; NACH B KOPIEREN
2E62	21 AC 5C	LD HL,\$5CAC	; M-7 ZUM 2. BYTE VON MEM5
2E65	86	ADD (HL)	; ADDIEREN = ANZAHL DER
2E66	77	LD (HL),A	; AUSZUGEBENDEN VORKOMMASTELLEN
2E67	78	LD A,B	; I MIT
2E68	ED 44	NEG	; $10^{-(M-7)}$ MULTIPLIZIEREN FÜR
2E6A	CD 4F 2D	CALL \$2D4F	; DIE AUSGABE
2E6D	18 92	JR \$2E01	; MIT DER STELLENMÄSSIG KORRI-
2E6F			; GIERTEN ZAHL IN DIE SCHLEIFE
2E6F			; ZURÜCKSPRINGEN
2E6F			
2E6F			; INTEGERTeil VON X IN DEN AUSGABEPUFFER MEM3 UND MEM4
2E6F			; SPEICHERN
2E6F			
2E6F	EB	EX DE,HL	; DE ZEIGT AUF I, HL AUF F
2E70	CD BA 2F	CALL \$2FBA	; MANTISSE VON I NACH DE',DE
2E73			; BRINGEN
2E73	D9	EXX	; DE' HOLEN
2E74	CB FA	SET 7,D	; RICHTIGES NUMERISCHES ERGEBNIS
2E76			; IN D' ANMERKEN
2E76	7D	LD A,L	; EXPONENT VON I NACH A
2E77	D9	EXX	; AUF NORMALE REGS UMSCHALTEN
2E78	D6 80	SUB \$80	; RICHTIGEN EXPONENT BERECHNEN
2E7A	47	LD B,A	; ERGIBT DIE BENÖTIGTE BITZAHL

2E7B		SLA E	; DIE MANTISSE VON I WIRD WIRD
2E7B CB 23		RL D	; BITWEISE NACH LINKS IN MEM4
2E7D CB 12		EXX	; ROTIERT, DABEI WIRD JEDEMAL
2E7F D9		RL E	; EINE DEZIMALKORREKTUR DURCH-
2E80 CB 13		RL D	; GEFÜHRT. DIE 4 BYTES VON I
2E82 CB 12		EXX	; SIND IN DE1 UND DE ENTHALTEN
2E84 D9		LD HL,\$5CAA	; 5. BYTE VON MEM4
2E85 21 AA 5C		LD C,5	; 5 BYTES MUSSEN DEZIMAL
2E88 0E 05			; KORRIGIERT WERDEN
2E8A		LD A,(HL)	; EIN BYTE VON MEM4 HOLEN
2E8A 7E		ADC A	; DURCH ADDITION EIN BIT NACH
2E8B 8F			; LINKS SCHIEBEN UND DAS CARRY
2E8C			; IN BIT 0 ADDIEREN
2E8C		DAA	; DEZIMALKORREKTUR
2E8C 27		LD (HL),A	; BYTE ZURÜCKSCHREIBEN
2E8D 77		DEC HL	; ZEIGER -1
2E8E 2B		DEC C	; ZÄHLER DER BYTES -1
2E8F 0D		JR NZ,\$2E8A	; 5 BYTES NOCH NICHT FESTIG
2E90 20 FB		DJNZ \$2E7B	; NOCH NICHT ALLE BITS VON I
2E92 10 E7			; BEHANDELT
2E94			
2E94			
2E94			
2E94			
2E94			
2E94			
2E94 AF		XOR A	; REG A LÖSCHEN
2E95 21 A6 5C		LD HL,\$5CA6	; 1. BYTE MEM4
2E98 11 A1 5C		LD DE,\$5CA1	; 1. BYTE MEM3
2E98 06 09		LD B,9	; STELLENZÄHLER
2E9D ED 6F		RLD	; LINKES NIBBLE 1. BYTE MEM4
2E9F			; WEGWERFEN
2E9F 0E FF		LD C,\$FF	; FÜHRENDE NULL ANKEREN
2EA1 ED 6F		RLD	; LINKES NIBBLE (HL) IN A,
2EA3			; RECHTES NIBBLE (HL) NACH LINKS
2EA3 20 04		JR NZ,\$2EA9	; DEZIMALSTELLE IN A IST 0
2EA5 0D		DEC C	; TEST AUF FÜHRENDE NULL:
2EA6 0C		INC C	
2EA7 20 0A		JR NZ,\$2EB3	; JA
2EA9 12		LD (DE),A	; DIGIT SPEICHERN
2EAA 13		INC DE	; ZEIGER ERHÖHEN
2EAB FD 34 71		INC (IY+\$71)	; EINE STELLE MEHR AUSZUGEBEN
2EAE FD 34 72		INC (IY+\$72)	; UND EINE MEHR VOR DEM KOMMA
2EB1 0E 00		LD C,0	; KEINE FÜHRENDE NULLEN MEHR
2EB3 CB 40		BIT 0,8	; ZEIGER AUF MEM4 NUR BEI
2EB5 2B 01		JR Z,\$2EB8	; NUR BEI JEDEM
2EB7 23		INC HL	; ZWEITEN DURCHLAUF ERHÖHEN
2EB8 10 E7		DJNZ \$2EA1	; STELLENZÄHLER NOCH NICHT 0
2EBA 3A AB 5C		LD A,(\$5CAB)	; PRÜFEN AUF 9 DIGITS OHNE
2EBD D6 09		SUB 9	; FÜHRENDE NULLEN
2EBF 3B 0A		JR C,\$2ECB	; WENN WENIGER, WEITERE DIGITS
2EC1			; HOLEN (NACHKOMMASTELLEN)
2EC1 FD 35 71		DEC (IY+\$71)	; STELLENZAHL AUF 8 SETZEN
2EC4 3E 04		LD A,4	; ZUM RUNDEN DAS LETZTE DIGIT
2EC6 FD BE 6F		CP (IY+\$6F)	; (4. BYTE MEM4) GEGEN 4 VER-
2EC9			; GLEICHEN, DAMIT DAS CARRY
2EC9			; ENTSPRECHEND GESETZT WIRD
2EC9 1B 41		JR DEZRND	; ZUM RUNDEN

```

2ECB      ;
2ECB      ; DIE NACHKOMMASTELLEN WERDEN JETZT IM AUSGABEPUFFER
2ECB      ; ABGELEGT
2ECB      ;
2ECB EF      RST CALRUF      ; I,
2ECC 02      .BYT $02      ; -, I LÖSCHEN
2ECD E2      .BYT $E2      ; F, MEM2 LADEN
2ECE 38      .BYT $38      ; F, ENDE
2ECF E8      EX DE,HL      ; DE ZEIGT AUF F
2ED0 CD BA 2F CALL $2FBA    ; MANTISSE VON F NACH DE', DE
2ED3 D9      EXX          ; ZWEITER REGISTERSATZ
2ED4 3E 80    LD A,$80      ; EXPONENTENANPASSUNG FÜR DEN
2ED6 95      SUB L        ; NACHKOMMA TEIL DURCHFÜHREN
2ED7 2E 00    LD L,0        ; EXPONENT AUF 0 SETZEN
2ED9 CB FA    SET 7,D      ; NUMERISCHES ERGEBNIS ANMERKEN
2EDB D9      EXX          ; NORMALER REGISTERSATZ
2EDC CD DD 2F CALL SHIFTF   ; DIE SHIFTS ZUM ANGLEICH AN DEN
2EDF          ; VORKOMMA TEIL DURCHFÜHREN
2EDF FD 7E 71 LD A,(1Y+$71) ; MEM5: STELLENZÄHLER HOLEN
2EE2 FE 08    CP 8         ; BEREITS 8 STELLEN ?
2EE4 38 06    JR C,$2EEC   ; NEIN
2EE6 D9      EXX          ; BEI 8 STELLEN NUR DAS HÖCHSTE
2EE7 CB 12    RL D         ; BIT DES HÖCHSTEN BYTE ZUM
2EE9 D9      EXX          ; RUNDEN INS CARRY SCHIEBEN
2EEA 18 20    JR DEZRND    ; ZUM RUNDEN
2EEC          ;
2EEC 01 00 02 LD BC,$200   ; C MIT 0 VORBESETZEN, B AUF 2
2EEF          ; ZUM ZÄHLEN
2EEF 7B      LD A,E        ; ES WIRD JEWEILS EINZELN D',E',
2EF0 CD 8B 2F CALL $2F8B    ; D UND E MIT 10 MULTIPLIZIERT
2EF3 5F      LD E,A        ; C ENTHÄLT JEWEILS DAS CARRY
2EF4 7A      LD A,D        ; ES WIRD ALSO 10*A+C FÜR D'..
2EF5 CD 8B 2F CALL $2F8B    ; BERECHNET
2EF8 57      LD D,A        ;
2EF9 C5      PUSH BC       ; ZÄHLER ZWISCHENSPEICHERN
2EFA D9      EXX          ; DEN ANDEREN REGISTERSATZ HOLEN
2EFB C1      POP BC        ; ZÄHLER ZURÜCK
2EFC 10 F1    DJNZ $2EEF   ; SPRUNG, WENN NOCH NICHT BEIDE
2EFE          ; REGISTERSÄTZE BEARBEITET
2EFE 21 A1 5C LD HL,$5CA1  ; 1. BYTE VON MEM3
2F01 79      LD A,C        ; ERGEBNIS NACH A
2F02 FD 4E 71 LD C,(1Y+$71) ; JETZIGE STELLENZAHL NACH C
2F03 09      ADD HL,BC     ; DAS 1. LEERE BYTE ADRESSIEREN
2F06 77      LD (HL),A     ; A SPEICHERN
2F07 FD 34 71 INC (1Y+$71) ; 1 STELLE MEHR ANMERKEN
2F0A 18 D3    JR $2EDF     ; WEITER, BIS 8 STELLEN
2F0C          ;
2F0C          ; DIE DEZIMALSTELLEN RUNDEN
2F0C          ;
2F0C F5      DEZRND PUSH AF ; CARRY ZWISCHENSPEICHERN
2F0D 21 A1 5C LD HL,$5CA1  ; 1. BYTE MEM3 = 1. STELLE
2F10 FD 4E 71 LD C,(1Y+$71) ; STELLENZAHL = OFFSET
2F13 06 00    LD B,0       ; NACH BC LADEN
2F15 09      ADD HL,BC     ; HINTER DAS LETZTE BYTE ZEIGEN
2F16 41      LD B,C        ; STELLENZAHL NACH B ZUM ZÄHLEN
2F17 F1      POP AF        ; CARRY HOLEN
2F18 2B      DEC HL        ; EINE STELLE IN DER ZAHL ZURÜCK
2F19 7E      LD A,(HL)     ; DAS DIGIT LADEN UND

```

2F1A	CE 00	ADC 0	; DAS CARRY ZUM RUNDEN ADDIEREN
2F1C	77	LD (HL),A	; ERGEBNIS SPEICHERN
2F1D	A7	AND A	; NULLEN AM ENDE WERDEN
2F1E	28 05	JR Z,\$2F25	; NICHT GEZAHLT
2F20	FE 0A	CP 10	; EINE 10 EBENFALLS NICHT, ABER
2F22	3F	CCF	; ES MUSS WEITER GERUNDET WERDEN
2F23	30 08	JR NC,\$2F2D	; SPRUNG BEI NORMALEN DIGIT
2F25	10 F1	DJNZ \$2F18	; WEITER IN DER RUNDUNGSSCHLEIFE
2F27	36 01	LD (HL),1	; HIER WIRD EINE ZUSÄTZLICHE 1
2F29	04	INC B	; NACH DEM RUNDEN BENÖTIGT,
2F2A	FD 34 72	INC (IY+\$72)	; +1 DIGIT UND +1 STELLE VOR
2F2D			; DEM KOMMA
2F2D	FD 70 71	LD (IY+\$71),B	; STELLENZAHL MERKEN
2F30	EF	RST CALRUF	; F, MUSS NOCH
2F31	02	.BYT \$02	; -, GELÖSCHT WERDEN
2F32	38	.BYT \$38	; ENDE
2F33	D9	EXX	; CALCULATOROFFSET
2F34	E1	POP HL	; AUS HL' ZURÜCKHOLEN
2F35	D9	EXX	
2F36			
2F36			; DIE ZAHL KANN AUSGEGEBEN WERDEN
2F36			
2F36	ED 4B AB 5C	LD BC,(\$5CAB)	; STELLENZÄHLER SETZEN
2F3A	21 A1 5C	LD HL,\$5CA1	; 1. BYTE MEM3 =1. BYTE DER ZAHL
2F3D	78	LD A,B	; MEHR ALS 9 STELLEN ?
2F3E	FE 09	CP 9	
2F40	38 04	JR C,\$2F46	; NEIN
2F42	FE FC	CP \$FC	; MEHR ALS 4 NULLEN NACH
2F44			; DEM DEZIMALPUNKT ?
2F44	38 26	JR C,\$2F6C	; JA: EXPONENTIALFORMAT AUSGEBEN
2F46	A7	AND A	; STELLEN VOR DEM PUNKT ?
2F47	CC EF 15	CALL Z,\$15EF	; NEIN: EINE NULL AUSGEBEN
2F4A	AF	XOR A	; A LÖSCHEN
2F4B	90	SUB B	; STELLEN VOR DEM PUNKT ?
2F4C	FA 52 2F	JP M,\$2F52	; JA: AUSGEBEN
2F4F	47	LD B,A	; STELLEN NACH DEM PUNKT IN B
2F50	18 0C	JR \$2F5E	; ZUM ZÄHLEN LADEN UND AUSGEBEN
2F52			
2F52			; DIE STELLEN VOR DEM PUNKT AUSGEBEN
2F52			
2F52	79	LD A,C	; NOCH STELLEN VOM SPEICHER
2F53	A7	AND A	; AUSZUGEBEN (B IST ANZAHL
2F54			; ALLER STELLEN) ?
2F54	28 03	JR Z,\$2F59	; NEIN: ENDNULL AUSGEBEN
2F56	7E	LD A,(HL)	; SONST DIGIT LADEN
2F57	23	INC HL	; ZEIGER ERHÖHEN
2F58	0D	DEC C	; VORKOMMAZÄHLER -1
2F59	CD EF 15	CALL \$15EF	; DIE ZAHL AUSGEBEN
2F5C	10 F4	DJNZ \$2F52	; NOCH NICHT ALLES BEDRUCKT
2F5E	79	LD A,C	; DEN DEZIMALPUNKT AUSGEBEN,
2F5F	A7	AND A	; WENN C NICHT NULL IST,
2F60	C8	RET Z	; SONST AUSGABE FERTIG
2F61	04	INC B	; ZÄHLER FÜR PUNKT +1
2F62	3E 2E	LD A,'.'	; DEN DEZIMALPUNKT
2F64	D7	RST PRTOU	; AUSGEBEN
2F65	3E 30	LD A,'0'	; EVTL. BENÖTIGTE NULLEN
2F67	10 FB	DJNZ \$2F64	; AUSGEBEN
2F69	41	LD B,C	; ZÄHLER FÜR DIE RESTLICHEN

2F6A	18 E6	JR \$2F52	; DIGITS SETZEN UND DRUCKEN
2F6C			
2F6C		; EINSTIEG, WENN DIE ZAHL IM EXPONENTIALFORMAT	
2F6C		; GEDRUCKT WERDEN MUSS	
2F6C			
2F6C	50	LD D,B	; STELLENZAHL NACH D KOPIEREN
2F6D	15	DEC D	; 1 ABZIEHEN ERGIBT EXPONENT
2F6E	06 01	LD B,1	; EINE STELLE VOR DEM DEZIMAL-
2F70			; PUNKT IM EXPONENTIALFORMAT
2F70	CD 4A 2F	CALL \$2F4A	; DIE STELLEN AUSGEBEN
2F73	3E 45	LD A,'E'	; DAS E FÜR DEN EXPONENT
2F75	D7	RST PRTOUT	; AUSGEBEN
2F76	4A	LD C,D	; EXPONENT NACH C
2F77	79	LD A,C	; IN A ZUM TESTEN BRINGEN
2F78	A7	AND A	; NEGATIVER EXPONENT ?
2F79	F2 83 2F	JP P,\$2F83	; NEIN
2F7C	ED 44	NEG	; SONST INVERTIEREN UND
2F7E	4F	LD C,A	; NACH C LADEN
2F7F	3E 2D	LD A,'-'	; MINUSZEICHEN LADEN UND
2F81	18 02	JR \$2F85	; AUSGEBEN
2F83	3E 2B	LD A,'+'	; PLUSZEICHEN FÜR POSITIVEN EXP.
2F85	D7	RST PRTOUT	; ZEICHEN AUSGEBEN
2F86	06 00	LD B,0	; B FÜR EXPONENTENAUSGABE
2F88	C3 1B 1A	JP \$1A1B	; LÖSCHEN UND AUSGEBEN
2F8B			
2F8B		SUBROUTINE ZUM BERECHNEN VON 10*(A)+(C)	
2F8B			
2F8B	D5	PUSH DE	; ZWISCHENSPEICHERN
2F8C	6F	LD L,A	; A NACH
2F8D	26 00	LD H,0	; HL KOPIEREN
2F8F	5D	LD E,L	; ZUSÄTZLICH NOCH NACH
2F90	54	LD D,H	; DE BRINGEN
2F91	29	ADD HL,HL	; 2*A BILDEN
2F92	29	ADD HL,HL	; 4*A BILDEN
2F93	19	ADD HL,DE	; 5*A BILDEN
2F94	29	ADD HL,HL	; 10*A BILDEN
2F95	59	LD E,C	; 10*A+C
2F96	19	ADD HL,DE	; BERECHNEN
2F97	4C	LD C,H	; ÜBERLAUF NACH C
2F98	7D	LD A,L	; ERGEBNIS IN A LADEN
2F99	D1	POP DE	; DE ZURÜCKHOLEN
2F9A	C9	RET	
2F9B			
2F9B		SUBROUTINE ZUM VORBEREITEN DER ADDITION EINER	
2F9B		FLOATINGPOINTZAHL	
2F9B			
2F9B	7E	LD A,(HL)	; EXPONENT LADEN
2F9C	36 00	LD (HL),0	; EXPONENTBYTE LÖSCHEN
2F9E	A7	AND A	; IST ES DIE ZAHL 0 ?
2F9F	C8	RET Z	; JA: RETURN
2FA0	23	INC HL	; VORZEICHENBYTE ADRESSIEREN
2FA1	CB 7E	BIT 7,(HL)	; TEST AUF POSITIV/NEGATIV
2FA3	CB FE	SET 7,(HL)	; DAS BIT IMMER 1 SETZEN
2FA5	2B	DEC HL	; ERSTES BYTE ADRESSIEREN
2FA6	C8	RET Z	; BEI POSITIVER ZAHL FERTIG
2FA7	C5	PUSH BC	; BC ZWISCHENSPEICHERN
2FA8	01 05 00	LD BC,5	; 5 BYTES BEARBEITEN
2FAB	09	ADD HL,BC	; HINTER DAS LETZTE BYTE ZEIGEN

2FAC	41	LD B,C	; B ALS ZÄHLER FÜR SCHLEIFE
2FAD	4F	LD C,A	; EXPONENT ZWISCHENSPEICHERN
2FAE	37	SCF	; CARRY FÜR 2-KOMPLEMENTBILDUNG
2FAF			; SETZEN
2FAF	2B	DEC HL	; ZEIGER EIN BYTE ZURÜCK
2FB0	7E	LD A,(HL)	; BYTE LADEN UND
2FB1	2F	CPL	; DIE 2-KOMPLEMENTBILDUNG
2FB2	CE 00	ADC 0	; DURCHFÜHREN
2FB4	77	LD (HL),A	; ZURÜCKSCHREIBEN
2FB5	10 FB	DJNZ 2FAF	; WIEDERHOLEN BIS 5 BYTES FERTIG
2FB7	79	LD A,C	; EXPONENT UND
2FB8	C1	POP BC	; BC RESTAURIEREN
2FB9	C9	RET	
2FBA			; SUBROUTINE ZUM LADEN VON 2 FLOATINGPOINTZAHLEN IN
2FBA			; DIE PROZESSORREGISTER. HL ZEIGT AUF 1. BYTE DER ERSTEN
2FBA			; UND DE AUF 1. BYTE DER ZWEITEN ZAHL
2FBA			; ZAHL 1: M1 - M5 IN H',B',C',C,B
2FBA			; ZAHL 2: N1 - N5 IN L',D',E',D,E
2FBA			;
2FBA	E5	PUSH HL	; HL UND
2FBB	F5	PUSH AF	; AF ZWISCHENSPEICHERN
2FBC	4E	LD C,(HL)	; M1 LADEN
2FBD	23	INC HL	
2FBE	46	LD B,(HL)	; M2 LADEN
2FBF	77	LD (HL),A	; BEI MULT/DIV VORZEICHEN
2FC0			; SPEICHERN
2FC0	23	INC HL	
2FC1	79	LD A,C	; M1 NACH A
2FC2	4E	LD C,(HL)	; M3 LADEN
2FC3	C5	PUSH BC	; M2/M3 ZWISCHENSPEICHERN
2FC4	23	INC HL	
2FC5	4E	LD C,(HL)	; M4 LADEN
2FC6	23	INC HL	
2FC7	46	LD B,(HL)	; M5 LADEN
2FC8	EB	EX DE,HL	; HL ZEIGT AUF 1. BYTE 2. ZAHL
2FC9	57	LD D,A	; M1 NACH D
2FCA	5E	LD E,(HL)	; N1 LADEN
2FCB	D5	PUSH DE	; M1/N1 ZWISCHENSPEICHERN
2FCC	23	INC HL	
2FCD	56	LD D,(HL)	; N2 LADEN
2FCE	23	INC HL	
2FCF	5E	LD E,(HL)	; N3 LADEN
2FD0	D5	PUSH DE	; N2/N3 ZWISCHENSPEICHERN
2FD1	D9	EXX	; AUF 2. REGISTERSATZ SCHALTEN
2FD2	D1	POP DE	; D'=N2, E'=N3
2FD3	E1	POP HL	; H'=M1, L'=N1
2FD4	C1	POP BC	; B'=M2, C'=M3
2FD5	D9	EXX	; NORMALER REGISTERSATZ
2FD6	23	INC HL	
2FD7	56	LD D,(HL)	; M4 LADEN
2FDB	23	INC HL	
2FD9	5E	LD E,(HL)	; M5 LADEN
2FDA	F1	POP AF	; AF ZURÜCKHOLEN
2FDB	E1	POP HL	; ZEIGER 1. BYTE 1. ZAHL ZURÜCK
2FDC	C9	RET	
2FDD			; SUBROUTINE ZUM SHIFTEN EINER ZAHL UM MAXIMAL 32 BITS
2FDD			

```

2FDD      ; FÜR EINE ADDITION (EXPONENTENANGLEICH)
2FDD      ;
2FDD A7    SHIFTF AND A      ; DIREKT RETURN, FALLS KEINE
2FDE CB    RET Z            ; DIFFERENZ DER EXPONENTEN
2FDF FE 21  CP 33           ; DIFFERENZ >32 ?
2FE1 30 16  JR NC,$2FF9     ; JA: NULL ADDIEREN
2FE3 C5     PUSH BC         ; ZWISCHENSPEICHERN
2FE4 47     LD B,A          ; EXPONENTDIFFERENZ ZUM ZÄHLEN
2FE5       ; DER RECHTSSHIFTS NACH B
2FE5 D9     SHIFTB EXX
2FE6 CB 2D  SRA L           ; L' NACH RECHTS SCHIEBEN
2FE8 CB 1A  RR D            ; D' UND E' MIT
2FEA CB 1B  RR E            ; CARRY NACH RECHTS ROTIEREN
2FEC D9     EXX
2FED CB 1A  RR D            ; EBENFALLS D UND E
2FEF CB 1B  RR E            ; NACH RECHTS ROTIEREN
2FF1 10 F2  DJNZ SHIFTB     ; WEITER, BIS DIFFERENZ 0 IST
2FF3 C1     POP BC          ; BC ZURÜCKLADEN
2FF4 D0     RET NC          ; KEIN CARRY ZU BERÜCKSICHTIGEN
2FF5 CD 04 30 CALL $3004    ; CARRY ADDIEREN
2FF8 C0     RET NZ          ; KEIN ÜBERLAUF NACH LINKS
2FF9 D9     ADDNUL EXX      ; L',D' UND E' HOLEN
2FFA AF     XOR A           ; A LÖSCHEN
2FFB 2E 00  LD L,0          ; L',
2FFD 57     LD D,A          ; D' UND
2FFE 5D     LD E,L          ; E' LÖSCHEN
2FFF D9     EXX             ; NORMALE REGISTER ZURÜCK
3000 11 00 00 LD DE,0       ; DE LÖSCHEN
3003 C9     RET
3004       .END
3004       .LIB SPEC3000-S
3004      ; SINCLAIR ZX SPECTRUM TEIL 3000
3004      ;
3004      ; SUBROUTINE ZUM ADDIEREN EINES CARRYS BEI DER RECHTS-
3004      ; VERSCHIEBUNG EINER ZAHL (SIEHE ROUTINE VORHER)
3004      ;
3004 1C     INC E             ; ES WERDEN JEWELNS DIE
3005 C0     RET NZ           ; EINZELNEN BYTES DER ZAHL
3006 14     INC D            ; NACHEINANDER INKREMENTIERT,
3007 C0     RET NZ           ; SOLANGE SICH NULL ALS ERGEBNIS
3008 D9     EXX             ; EINSTELLT, SONST DIREKT RETURN
3009 1C     INC E
300A 20 01  JR NZ,$300D     ; WENN DAS ZEROFLAG
300C 14     INC D            ; BEIM RETURN GESETZT IST,
300D D9     EXX             ; IST EIN ÜBERLAUF AUFGETRETEN
300E C9     RET
300F      ;
300F      ; DURCHFÜHRUNG DER SUBTRAKTION ZWEIER
300F      ; FLOATINGPOINTZAHLEN
300F      ;
300F EB     SUBTRA EX DE,HL  ; ZEIGER UMSCHALTEN
3010 CD 6E 34 CALL $346E    ; VORZEICHEN DER ZU SUBTRA-
3013       ; HIERENDEN ZAHL INVERTIEREN
3013 EB     EX DE,HL        ; ZEIGER ZURÜCK UND ADDIEREN
3014      ;
3014      ; SUBROUTINE ZUM ADDIEREN ZWEIER FLOATINGPOINTZAH-
3014      ; LEN AUF DEM CALC.-STACK ZU EINEM 'LETZTEN WERT'
3014      ;

```

3014	1A	ADDIER LD A, (DE)	; TEST, OB DIE ERSTEN BYTES
3015	B6	OR (HL)	; BEIDER ZAHLEN 0 SIND
3016	20 26	JR NZ, #303E	; NEIN: VOLLE ADDITION
3018			
3018		; ADDITION VON INTEGERZAHLEN (65535	
3018			
3018	D5	PUSH DE	; ZEIGER AUF 2. ZAHL SETZEN
3019	23	INC HL	; ZEIGT AUF 2. BYTE 1. ZAHL
301A	E5	PUSH HL	; UND RETTEN
301B	23	INC HL	; 3. BYTE 1. ZAHL
301C	5E	LD E, (HL)	; LOWBYTE NACH E LADEN
301D	23	INC HL	; 4. BYTE 1. ZAHL
301E	56	LD D, (HL)	; HIGHBYTE NACH D LADEN
301F	23	INC HL	; ZUM 2. BYTE DER
3020	23	INC HL	; 2. ZAHL GEHEN
3021	23	INC HL	; (=VORZEICHENBYTE)
3022	7E	LD A, (HL)	; NACH A LADEN
3023	23	INC HL	; 3. BYTE 2. ZAHL
3024	4E	LD C, (HL)	; LOWBYTE LADEN
3025	23	INC HL	; 4. BYTE 2. ZAHL
3026	46	LD B, (HL)	; HIGHBYTE LADEN
3027	E1	POP HL	; ZEIGER AUF VORZEICHEN 1. ZAHL
3028	EB	EX DE, HL	; NACH DE, 1. ZAHL NACH HL
3029	09	ADD HL, BC	; DIE 2 ZAHLEN ADDIEREN
302A	EB	EX DE, HL	; ERGEBNIS IN DE, ZEIGER AUF
302B			; VORZEICHEN 1. ZAHL IN HL
302B	8E	ADC (HL)	; DIE ZWEI VORZEICHEN UND DAS
302C			; CARRY ADDIEREN
302C	0F	RRCA	; REG A MUSS NULL SEIN,
302D	CE 00	ADC 0	; SONST IST DAS ERGEBNIS >65536
302F	20 0B	JR NZ, #303C	; ÜBERLAUF: VOLLE ADDITION
3031	9F	SBC A	; VORZEICHEN DES ERGEBNISSES
3032	77	LD (HL), A	; BERECHNEN UND SPEICHERN
3033	23	INC HL	
3034	73	LD (HL), E	; LOWBYTE UND
3035	23	INC HL	
3036	72	LD (HL), D	; HIGHBYTE SPEICHERN
3037	2B	DEC HL	; HL AUF 1. BYTE
3038	2B	DEC HL	; VOM ERGEBNIS SETZEN
3039	2B	DEC HL	
303A	D1	POP DE	; STKEND NACH DE HOLEN
303B	C9	RET	
303C			
303C	2B	DEC HL	; ZEIGER AUF 1. BYTE 1. ZAHL UND
303D	D1	POP DE	; 1. BYTE 2. ZAHL WIEDER SETZEN
303E	CD 93 32	CALL #3293	; BEIDE ZAHLEN ALS VOLLE 3 BYTES
3041			; IN FLOATINGFORM AUF DEN STACK
3041			
3041		; VOLLE ADDITION ZWEIER ZAHLEN (KEINE INTEGERZAHLEN)	
3041			
3041	D9	EXX	; REGISTERTAUSCH, UM DEN ZEIGER
3042	E5	PUSH HL	; AUF NÄCHSTE ZAHL ZU RETTEN
3043	D9	EXX	
3044	D5	PUSH DE	; ZEIGER 1. SUMMAND RETTEN
3045	E5	PUSH HL	; DESGLEICHEN 2. SUMMAND
3046	CD 9B 2F	CALL #2F9B	; 1. ZAHL FÜR DIE ADDITION
3049			; VORBEREITEN
3049	47	LD B, A	; EXPONENTEN IN B RETTEN

304A	EB	EX DE,HL	; ZEIGERTAUSCH FÜR 2. ZAHL
304B	CD 9B 2F	CALL \$2F9B	; DIESE FÜR ADDITION VORBEREITEN
304E	4F	LD C,A	; EXPONENT NACH C
304F	8B	CP B	; VERGLEICH, OB DIE 1. ZAHL
3050			; KLEINER ALS DIE ZWEITE IST
3050	30 03	JR NC,\$3055	; JA
3052	7B	LD A,B	; SONST DIE BEIDEN
3053	41	LD B,C	; EXPONENTEN UND DIE
3054	EB	EX DE,HL	; ZEIGER TAUSCHEN
3055	F5	PUSH AF	; GRÖßEREN EXPONENT RETTEN
3056	90	SUB B	; DIFFERENZ FÜR RECHTSSCHIEBEN
			BERECHNEN
3057	CD BA 2F	CALL \$2FBA	; BEIDE ZAHLEN VOM STACK HOLEN
305A	CD DD 2F	CALL \$2FDD	; ANPASSUNG DER MANTISSEN
			DURCHFÜHREN
305D	F1	POP AF	; GRÖßEREN EXPONENT HOLEN
305E	E1	POP HL	; HL ZEIGT AUF DAS ERGEBNIS
305F	77	LD (HL),A	; EXPONENT ERGEBNIS SPEICHERN
3060	E5	PUSH HL	; ZEIGER ZWISCHENSPEICHERN
3061	6B	LD L,B	; DIE ZWEI RECHTEN BYTES FÜR
3062	61	LD H,C	; DIE ADDITION NACH HL
3063	19	ADD HL,DE	; DIESE ADDIEREN
3064	D9	EXX	; ERGEBNIS IN HL' RETTEN UND
3065	EB	EX DE,HL	; DIE NÄCHSTEN BYTES ZUM
3066	ED 4A	ADC HL,BC	; ADDIEREN HOLEN
3068	EB	EX DE,HL	; ERGEBNIS NACH DE'
3069	7C	LD A,H	; DAS 5. BYTE IN H' UND
306A	8D	ADC L	; L' ADDIEREN
306B	6F	LD L,A	; ERGEBNIS NACH L'
306C	1F	RRA	; WENN ÜBERLAUF BEI DER ADDITION
306D	AD	XOR L	; ZWEIER POSITIVER ZAHLEN ODER
306E			; KEIN ÜBERLAUF BEI 2 NEGATIVEN
306E			; ZAHLEN AUFTRIFF, MUSS DAS
306E			; ERGEBNIS 1 STELLE NACH RECHTS
306E			; VERSCHOBEN WERDEN
306E	D9	EXX	; ERGEBNIS NACH DE UND
306F	EB	EX DE,HL	; DE' BRINGEN
3070	E1	POP HL	; ZEIGER AUF DEN EXPONENT
3071	1F	RRA	; TEST OB SHIFT NOTWENDIG
3072	30 0B	JR NC,\$307C	; NEIN
3074	3E 01	LD A,1	; UM 1 STELLE SCHIEBEN
3076	CD DD 2F	CALL \$2FDD	; RECHTSSHIFT AUSFÜHREN
3079	34	INC (HL)	; EXPONENT UM 1 ERHÖHEN
307A	2B 23	JR Z,\$309F	; SPRUNG, FALLS ÜBERLAUF; ERROR
307C	D9	EXX	; TEST AUF NEGATIVES ERGEBNIS
307D	7D	LD A,L	; VORZEICHENBIT HOLEN UND
307E	E6 80	AND \$80	; ISOLIEREN
3080	D9	EXX	; DIESES BIT IM
3081	23	INC HL	; 2. BYTE DES ERGEBNISSES
3082	77	LD (HL),A	; SPEICHERN
3083	2B	DEC HL	; AUF 1. BYTE ZEIGEN
3084	2B 1F	JR Z,\$30A5	; ERGEBNIS POSITIV; KEIN
3086			; 2-KOMPLEMENT BILDEN
3086	7B	LD A,E	; ERSTES BYTE HOLEN UND
3087	ED 44	NEG	; 2-KOMPLEMENT BILDEN
3089	3F	CCF	; CARRY INVERTIEREN FÜR WEITERE
308A			; 2-KOMPLEMENTBILDUNG
308A	5F	LD E,A	; BYTE ABSPEICHERN

308B	7A	LD A,D	: 2. BYTE LADEN UND
308C	2F	CPL	: INVERTIEREN, MIT DER
308D	CE 00	ADC 0	: ADDITION ERGIBT SICH DAS
308F	57	LD D,A	: 2-KOMPLEMENT, ABSPEICHERN
3090	D9	EXX	
3091	7B	LD A,E	: VORGANG FÜR DIE
3092	2F	CPL	: WEITEREN 2 BYTES
3093	CE 00	ADC 0	: WIEDERHOLEN
3095	5F	LD E,A	
3096	7A	LD A,D	
3097	2F	CPL	
3098	CE 00	ADC 0	
309A	30 07	JR NC,\$30A3	: KEIN CARRY: FERTIG
309C	1F	RRA	: SONST IST DAS ERGEBNIS
309D			: GENAU 2*XX, MANTISSE AUF
309D			: 0.5 SETZEN
309D	D9	EXX	: EXPONENT UM
309E	34	INC (HL)	: 1 ERHÖHEN
309F	CA AD 31	JP Z,\$31AD	: RECHENBEREICH ÜBERSCHRITTEN:
30A2			: FEHLERMELDUNG
30A2	D9	EXX	
30A3	57	LD D,A	: LETZTES BYTE SPEICHERN
30A4	D9	EXX	
30A5	AF	XOR A	: CARRY LÖSCHEN UND ZUM
30A6	C3 55 31	JP \$3155	: NORMALISIEREN
30A9			
30A9		SUBROUTINE ZUM BERECHNEN VON HL=DE*HL	
30A9			
30A9	C5	PUSH BC	: BC ZWISCHENSPEICHERN
30AA	06 10	LD B,16	: B ALS ZÄHLER FÜR 16 BIT
30AC	7C	LD A,H	: HIGHBYTE NACH A UND
30AD	4D	LD C,L	: LOWBYTE NACH C BRINGEN
30AE	21 00 00	LD HL,0	: HL FÜR BERECHNUNG AUF 0
30B1	29	ADD HL,HL	: HL MIT 2 MULTIPLIZIEREN
30B2	38 0A	JR C,\$30BE	: CARRY GESETZT: ÜBERLAUF
30B4	CB 11	RL C	: JEWEILS DAS HÖCHSTE BIT INS
30B6	17	RLA	: CARRY UND DIE RESTLICHEN BITS
30B7			: NACHSCHIEBEN
30B7	30 03	JR NC,\$30BC	: BIT WAR 0: SPRUNG
30B9	19	ADD HL,DE	: SONST DE ADDIEREN
30BA	38 02	JR C,\$30BE	: SPRUNG BEI ÜBERLAUF
30BC	10 F3	DJNZ \$30B1	: 16 MAL ROTIERT UND ADDIERT ?
30BE	C1	POP BC	: JA: BC ZURÜCKHOLEN UND ENDE
30BF	C9	RET	
30C0			
30C0		SUBROUTINE ZUM VORBEREITEN EINER MULTIPLIKATION ODER	
30C0		DIVISION	
30C0			
30C0	CD E9 34	CALL \$34E9	: IST DIE ZAHL 0 ?
30C3	D8	RET C	: JA, RETURN
30C4	23	INC HL	: VORZEICHENBYTE ADRESSIEREN
30C5	AE	XOR (HL)	: IN A VORZEICHEN DES ERGEBNISSES
30C6			: BILDEN: GLEICHE = 0, SONST \$FF
30C6	CB FE	SET 7,(HL)	: BIT FÜR NUMERISCH SETZEN
30C8	2B	DEC HL	: WIEDER AUF EXPONENT ZEIGEN
30C9	C9	RET	
30CA			
30CA		SUBROUTINE ZUM AUSFÜHREN EINER MULTIPLIKATION	

30CA		MULTIP	LD A, (DE)	; TEST, OB DIE ERSTEN BYTES
30CA	1A		OR (HL)	; BEIDER ZAHLEN 0 SIND
30CB	B6		JR NZ, \$30F0	; NEIN: VOLLE MULTIPLIKATION
30CC	20 22		PUSH DE	; JA: INTEGERSMULTIPLIKATION
30CE	D5		PUSH HL	; ZEIGER AUF 2., 1. UND NOCH
30CF	E5		PUSH DE	; EINMAL 2. ZAHL RETTEN
30D0	D5		CALL \$2D7F	; ZAHL IN DE, VORZEICHEN IN C
30D1	CD 7F 2D		EX DE, HL	; DIE ZAHL IN HL BRINGEN UND MIT
30D4	EB		EX (SP), HL	; ZEIGER AUF 2. ZAHL TAUSCHEN
30D5	E3		LD B, C	; VORZEICHEN 1. ZAHL NACH B
30D6	41		CALL \$2D7F	; 2. ZAHL HOLEN
30D7	CD 7F 2D		LD A, B	; VORZEICHEN DES ERGEBNIS
30DA	78		XOR C	; BESTIMMEN UND NACH
30DB	A9		LD C, A	; C KOPIEREN
30DC	4F		POP HL	; 1. ZAHL NACH HL
30DD	E1		CALL \$30A9	; MULTIPLIKATION HL=DE*HL
30DE	CD A9 30		EX DE, HL	; ERGEBNIS NACH DE
30E1	EB		POP HL	; ZEIGER AUF 1. ZAHL HOLEN
30E2	E1		JR C, \$30EF	; ÜBERLAUF: VOLLE MULTIPLIKATION
30E3	38 0A		LD A, D	; TESTEN, OB ERGEBNIS
30E5	7A		OR E	; NULL IST
30E6	B3		JR NZ, \$30EA	; NEIN
30E7	20 01		LD C, A	; SONST EXPONENT = 0
30E9	4F		CALL \$2DBE	; ZAHL IM STACK SPEICHERN
30EA	CD 8E 2D		POP DE	; ZEIGER AUF STKEND HOLEN
30ED	D1		RET	
30EE	C9			
30EF				
30EF	D1		POP DE	; ZEIGER AUF 2. ZAHL ZURÜCK
30F0	CD 93 32		CALL \$3293	; BEIDE ZAHLEN AUF DEN STACK
30F3	AF		XOR A	; A LÖSCHEN, UM VORZEICHEN
30F4	CD C0 30		CALL \$30C0	; 1. ZAHL ZU ÜBERNEHMEN
30F7	DB		RET C	; FERTIG, WENN 1 ZAHL = 0
30F8	D9		EXX	; REGISTERTAUSCH, UM ZEIGER AUF
30F9	E5		PUSH HL	; NÄCHSTE ZAHL ZU RETTEN
30FA	D9		EXX	
30FB	D5		PUSH DE	; ZEIGER AUF MULTIPLIKANT RETTEN
30FC	EB		EX DE, HL	; ZEIGERTAUSCH, UM DIE
30FD	CD C0 30		CALL \$30C0	; 2. ZAHL VORZUBEREITEN
3100	EB		EX DE, HL	; ZEIGER ZURÜCKTAUSCHEN
3101	38 5A		JR C, \$315D	; SPRUNG, WENN 2. ZAHL = 0
3103	E5		PUSH HL	; ZEIGER AUF ERGEBNIS RETTEN
3104	CD BA 2F		CALL \$2FBA	; BEIDE ZAHLEN VOM STACK HOLEN
3107	78		LD A, B	; M5 NACH A HOLEN
3108	A7		AND A	; CARRY FÜR SUBTRAKTION LÖSCHEN
3109	ED 62		SBC HL, HL	; HL FÜR ERGEBNIS AUF 0 SETZEN
310B	D9		EXX	; REGISTERTAUSCH, UM M1 UND
310C	E5		PUSH HL	; M1 (IN HL') ZU RETTEN
310D	ED 62		SBC HL, HL	; HL' MIT 0 VORBESETZEN
310F	D9		EXX	; NORMALE REGISTER
3110	06 21		LD B, 33	; B ALS ZÄHLER AUF 33 SETZEN
3112	18 11		JR \$3125	; SPRUNG IN DIE MULTIPLIKATIONS-
3114				; SCHLEIFE
3114				
3114	30 05		JR NC, \$311B	; BIT IST NICHT GESETZT,
3116				; NICHTS ADDIEREN
3116	19		ADD HL, DE	; SONST MULTIPLIKANT IN
3117	D9		EXX	; IN DE UND DE' ZU

3160	CD FB 2F	CALL \$2FFB	; (DIV REGS AUF 0 SETZEN)
3163	07	RLCA	; AUF NULL SETZEN
3164	77	LD (HL),A	; EXPONENT ABSPEICHERN
3165	38 2E	JR C,\$3195	; SPRUNG BEI 2 ⁻¹²⁸
3167	23	INC HL	; SONST 0 IM 2. BYTE
3168	77	LD (HL),A	; SPEICHERN
3169	2B	DEC HL	; AUF 1. BYTE ZEIGEN
316A	18 29	JR \$3195	; ERGEBNIS EINSCHREIBEN
316C			
316C	06 20	NORMAL LD B,32	; MAXIMAL 32 SHIFTS ZUM
316E	D9	EXX	; NORMALISIEREN DER MANTISSE
316F	CB 7A	BIT 7,D	; IN DEN REGS DE, DE'
3171	D9	EXX	; BIS BIT7 VON D'
3172	20 12	JR NZ,\$3186	; GESETZT IST
3174	07	RLCA	; DIE ENTSPRECHENDEN
3175	CB 13	RL E	; SHIFTS AUSFÜHREN
3177	CB 12	RL D	
3179	D9	EXX	
317A	CB 13	RL E	
317C	CB 12	RL D	
317E	D9	EXX	
317F	35	DEC (HL)	; EXPONENT -1
3180	28 D7	JR Z,\$3159	; WENN EXPONENT < -129, DANN
3182			; EXPONENT AUF -128 SETZEN
3182	10 EA	DJNZ \$316E	; SCHLEIFE NOCH NICHT FERTIG
3184	18 D7	JR \$315D	; BIT7 WURDE NIE 1:
3186			; ERGEBNIS IST NULL
3186	17	RLA	; EIN EVTL. CARRY WIEDER
3187	30 0C	JR NC,\$3195	
3189	CD 04 30	CALL \$3004	; DAZUADDIEREN
318C	20 07	JR NZ,\$3195	; KEIN ÜBERLAUF
318E	D9	EXX	; SONST MANTISSE AUF
318F	16 80	LD D,\$80	; .5 SETZEN
3191	D9	EXX	
3192	34	INC (HL)	; EXPONENT +1
3193	28 18	JR Z,\$31AD	; FEHLER, WENN ÜBERLAUF
3195	E5	PUSH HL	; ERGEBNISZEIGER RETTEN
3196	23	INC HL	; ZEIGT AUF VORZEICHENBYTE
3197	D9	EXX	
3198	D5	PUSH DE	; ERGEBNIS IN DE'
3199	D9	EXX	
319A	C1	POP BC	; NACH BC BRINGEN
319B	78	LD A,B	; DAS VORZEICHENBIT
319C	17	RLA	; WIRD AN DIE RICHTIGE
319D	CB 16	RL (HL)	; STELLE (BIT7 DES HÖCHSTEN
319F	1F	RRA	; ERGEBNISBYTE) KOPIERT
31A0	77	LD (HL),A	; UND ABGESPEICHERT
31A1	23	INC HL	
31A2	71	LD (HL),C	; DIE WEITEREN
31A3	23	INC HL	
31A4	72	LD (HL),D	; ERGEBNISBYTES
31A5	23	INC HL	
31A6	73	LD (HL),E	; ABSPEICHERN
31A7	E1	POP HL	; ZEIGER AUF ERGEBNIS UND
31A8	D1	POP DE	; ZEIGER 2. ZAHL ZURÜCKHOLEN
31A9	D9	EXX	
31AA	E1	POP HL	; ZEIGER AUF NÄCHSTE ZAHL
31AB	D9	EXX	; NACH HL' HOLEN

31AC	C9	RET	
31AD			
31AD	CF	RST ERRAUS	; FEHLERMELDUNG:
31AE	05	.BYT \$05	; ARITHMETIC OVERFLOW
31AF			
31AF		; SUBROUTINE ZUM AUSFÜHREN EINER DIVISION	
31AF			
31AF	CD 93 32	CALL \$3293	; BEIDE ZAHLEN AUF STACK
31B2	EB	EX DE,HL	; ZEIGER TAUSCHEN
31B3	AF	XOR A	; A LÖSCHEN, UM VORZEICHEN
31B4			; 1. ZAHL ZU ÜBERNEHMEN
31B4	CD C0 30	CALL \$30C0	; DIVISION VORBEREITEN
31B7	3B F4	JR C,\$31AD	; ERROR, WENN DIVISOR 0
31B9	EB	EX DE,HL	; ZEIGER WIEDER ZURÜCK
31BA	CD C0 30	CALL \$30C0	; DIVIDEND AUFBEREITEN
31BD	DB	RET C	; WENN DIESER 0 IST, ERGEBNIS 0
31BE	D9	EXX	; ZEIGER AUF NÄCHSTE
31BF	ES	PUSH HL	; ZAHL ZWISCHENSPEICHERN
31C0	D9	EXX	
31C1	D5	PUSH DE	; ZEIGER AUF DIVISOR UND
31C2	E5	PUSH HL	; AUF DIVIDEND RETTEN
31C3	CD BA 2F	CALL \$2FBA	; ZAHLEN VOM STACK HOLEN
31C6	D9	EXX	
31C7	E5	PUSH HL	; BEIDE EXPONENTEN AUF STACK
31C8	60	LD H,B	; DEN GANZEN DIVIDENDEN
31C9	69	LD L,C	; VON BC, BC NACH
31CA	D9	EXX	; HL' UND HL
31CB	61	LD H,C	; KOPIEREN
31CC	68	LD L,B	
31CD	AF	XOR A	; A UND CARRY LÖSCHEN
31CE	06 DF	LD B,\$DF	; B ZUM ZAHLEN SETZEN (-37)
31D0	1B 10	JR \$31E2	
31D2			
31D2	17	RLA	; QUOTIENT NACH LINKS IN
31D3	CB 11	RL C	; DIE REGISTER BC',C,A
31D5	D9	EXX	; ROTIEREN
31D6	CB 11	RL C	
31D8	CB 10	RL B	
31DA	D9	EXX	; REST DES DIVIDENDEN
31DB	29	ADD HL,HL	; IN HL,HL' MIT
31DC	D9	EXX	; 2 MULTIPLIZIEREN
31DD	ED 6A	ADC HL,HL	; DABEI WIRD EIN CARRY
31DF	D9	EXX	; FÜR VOLLES 32-BIT-ERGEBNIS
31E0	3B 10	JR C,\$31F2	; BERÜCKSICHTIGT
31E2			
31E2	ED 52	SBC HL,DE	; VERSUCHSWEISE DEN DIVISOR
31E4	D9	EXX	; VOM REST DES DIVIDENDEN
31E5	ED 52	SBC HL,DE	; SUBTRAHIEREN
31E7	D9	EXX	
31E8	30 0F	JR NC,\$31F9	; SUBTRAKTION GEHT
31EA	19	ADD HL,DE	; SONST WAR DER DIVISOR
31EB	D9	EXX	; GRÖßER ALS DER DIVIDEND;
31EC	ED 5A	ADC HL,DE	; DIVISOR WIEDER ZURÜCK-
31EE	D9	EXX	; ADDIEREN (KEIN DIVISIONSBIT)
31EF	A7	AND A	; CARRY LÖSCHEN
31F0	1B 0B	JR \$31FA	; WEITER IN DER SCHLEIFE
31F2	A7	AND A	; CARRY LÖSCHEN UND DIVISOR
31F3	ED 52	SBC HL,DE	; FÜR VOLLE GENAUIGKEIT

31F5 D9	EXX	; WIEDER AUFADDIEREN (IN DER
31F6 E0 52	SBC HL,DE	; SCHIEBESCHLEIFE)
31F8 D9	EXX	
31F9		
31F9 37	SCF	; 1 BIT IM QUOTIENT SETZEN
31FA 04	INC B	; ZÄHLER -(-1)
31FB FA D2 31	JP M,\$31D2	; NOCH NICHT FERTIG
31FE F5	PUSH AF	; CARRY RETTEN
31FF 2B E1	JR Z,\$31E2	; ES WIRD EIN NUTZLOSER VERSUCH
3201		; GEMACHT, EIN WEITERES BIT IM
3201		; ERGEBNIS ZU GEWINNEN
3201 5F	LD E,A	; DIE 4 MANTISSENBYTES DES
3202 51	LD D,C	; QUOTIENTEN NACH
3203 D9	EXX	; DE UND DE' KOPIEREN
3204 59	LD E,C	
3205 50	LD D,B	
3206 F1	POP AF	; DAS 33. UND 34. BIT
3207 CB 18	RR B	; IN DAS REGISTER B'
3209 F1	POP AF	
320A CB 18	RR B	; BRINGEN
320C D9	EXX	
320D C1	POP BC	; DIE BEIDEN EXPONENTEN HOLEN
320E E1	POP HL	; ZEIGER AUF ERGEBNIS
320F 78	LD A,B	; DEN EXPONENTEN DES ERGEBNISSES
3210 91	SUB C	; IN A BERECHNEN UND ZUM
3211 C3 3D 31	JP \$313D	; NORMALISIEREN
3214		
3214		; SUBROUTINE ZUM ABTRENNEN DES INTEGERTEILS EINER
3214		; VARIABLEN
3214		
3214 7E	LD A,(HL)	; EXPONENT LADEN
3215 A7	AND A	; FALLS DIESER 0 IST, IST ES
3216 C8	RET Z	; BEREITS EINE INTEGERZAHL
3217 FE 81	CP \$81	; EXPONENT MUSS >\$81 (ZAHL >1)
3219 30 06	JR NC,\$3221	; SEIN
321B 36 00	LD (HL),0	; SONST WIRD DER INTEGER-
321D 3E 20	LD A,32	; TEIL AUF 0 GESETZT (32 BIT)
321F 18 51	JR \$3272	
3221		
3221 FE 91	CP \$91	; EXPONENT LADEN UND FÜR
3223		; 16 BIT INTEGER UND DIE ZAHL
3223		; -65536 VERGLEICHEN
3223 20 1A	JR NZ,\$323F	; KANN NICHT -65536 SEIN
3225 23	INC HL	; 4. BYTE FÜR -65536 PRÜFUNG
3226 23	INC HL	; ADRESSIEREN
3227 23	INC HL	
3228 3E 80	LD A,\$80	; DAVON DAS HÖCHSTE BIT
322A A6	AND (HL)	; PRÜFEN
322B 2B	DEC HL	; DIE ZWEI WEITEREN
322C B6	OR (HL)	; BYTES MÜSSEN EBENFALLS
322D 2B	DEC HL	; 0 FÜR -65536 SEIN
322E 20 03	JR NZ,\$3233	; UNGLEICH 0, TEST ENDE
3230 3E 80	LD A,\$80	; DAS VORZEICHEN MUSS
3232 AE	XOR (HL)	; 1 UND DER REST 0 SEIN
3233 2B	DEC HL	; ZEIGT WIEDER AUF EXPONENT
3234 20 36	JR NZ,\$326C	; NICHT -65536
3236 77	LD (HL),A	; SONST WIRD DIE ZAHL
3237 23	INC HL	; -65536 = \$91,\$80,\$00,\$00,\$00

3238 36 FF	LD (HL), \$FF	; DURCH \$00, \$FF, \$00, \$00, \$00
323A 2B	DEC HL	; ERSETZT = -1, DIES IST
323B 3E 18	LD A, 24	; ALLERDINGS EIN FEHLER!
323D 1B 33	JR \$3272	; 24 BIT AUF NULL SETZEN
323F		
323F 30 2C	JR NC, \$326D	; SPRUNG, WENN EXPONENT
3241		; >\$92 IST
3241 05	PUSH DE	; STKEND ZWISCHENSPEICHERN
3242 2F	CPL	; EXPONENT IN DEN BEREICH VON
3243 C6 91	ADD \$91	; 0 (=144 ALT) BIS 15 (=129 ALT)
3245		; UMWANDELN
3245 23	INC HL	; DAS ZWEITE BYTE
3246 56	LD D, (HL)	; NACH DE LADEN
3247 23	INC HL	; UND DAS DRITTE
3248 5E	LD E, (HL)	; NACH E
3249 2B	DEC HL	; HL WIEDER AUF
324A 2B	DEC HL	; DEN EXPONENT SETZEN
324B 0E 00	LD C, 0	; DEFAULT: POSITIVE ZAHL
324D CB 7A	BIT 7, D	; TEST, OB ZAHL POSITIV
324F 2B 01	JR Z, \$3252	; JA
3251 0D	DEC C	; NEGATIV: C ENTHALT \$FF
3252 CB FA	SET 7, D	; BIT FÜR NUMERISCH IN D SETZEN
3254 06 0B	LD B, 8	; TESTEN, OB A >= 8 (NUR 1 BYTE
3256 90	SUB B	; WANDLEN), OHNE A ZU
3257 80	ADD B	; VERÄNDERN
3258 3B 04	JR C, \$325E	; SPRUNG, WENN 2 BYTES
325A		; GEWANDelt WERDEN MÜSSEN
325A 5A	LD E, D	; DAS EINE BYTE NACH E
325B 16 00	LD D, 0	; UND D LÖSCHEN
325D 90	SUB B	; FÜR DIE NÖTIGEN SHIFTS A IN
325E		; DEN BEREICH VON 1-7 BRINGEN
325E 2B 07	JR Z, \$3267	; FALLS 0, KEINE SHIFTS
3260 47	LD B, A	; B ZÄHLT DIE SHIFTS
3261 CB 3A	SRL D	; D UND E SOLANGE NACH RECHTS
3263 CB 1B	RR E	; SHIFTEN, BIS B = 0, D. H.
3265 10 FA	DJNZ \$3261	; DIE KORREKTE ZAHL GEFUNDEN IST
3267 CD BE 2D	CALL \$2D8E	; ERGEBNIS AUF STACK SPEICHERN
326A D1	POP DE	; STKEND IN DE ZURÜCKLADEN
326B C9	RET	
326C		
326C		
326C		
326C 7E	LD A, (HL)	; EXPONENT LADEN
326D D6 A0	SUB \$A0	; \$A0 SUBTRAHIEREN
326F F0	RET P	; RETURN, WENN DEZIMALPUNKT
3270		; ERST AM ENDE ODER NACH DER
3270		; MANTISSE STEHT (NUR INTEGER)
3270 ED 44	NEG	; SONST REST INVERTIEREN, UM
3272		; DIE ANZAHL DER NULLBITS
3272		; ZU BESTIMMEN
3272 05	PUSH DE	; STKEND ZWISCHENSPEICHERN
3273 EB	EX DE, HL	; HL AUF DAS
3274 2B	DEC HL	; 5. BYTE SETZEN
3275 47	LD B, A	; BITZAHL, DIE NULL ZU SETZEN
3276 CB 3B	SRL B	; SIND, NACH B LADEN UND
3278 CB 3B	SRL B	; DURCH 8 DIVIDIEREN, UM DIE
327A CB 3B	SRL B	; VOLLE BYTEZAHL ZU ERHALTEN
327C 2B 05	JR Z, \$3283	; KEIN GANZES BYTE

; GROSSE WERTE FÜR 'X' UNTERSUCHEN

327E 36 00	LD (HL),0	; SONST DIE BYTES
3280 2B	DEC HL	; LÖSCHEN, BIS B
3281 10 FB	DJNZ \$327E	; NULL IST
3283 E6 07	AND 7	; RESTLICHEN BITS IN A
3285 28 09	JR Z,\$3290	; KEIN WEITERES BIT NULL SETZEN
3287 47	LD B,A	; B ALS BITZÄHLER SETZEN
3288 3E FF	LD A,\$FF	; IN A EINE MASKE AUFBAUEN,
328A CB 27	SLA A	; IN DER VON RECHTS DIE BITS,
328C 10 FC	DJNZ \$328A	; (B) ENTSPRECHEND, NULL WERDEN
328E A6	AND (HL)	; LOGISCHES UND MIT DEM BYTE
328F 77	LD (HL),A	; AUSFÜHREN UND ZURÜCKSCHREIBEN
3290 EB	EX DE,HL	; ZEIGER AUF EXPONENT UND
3291 D1	POP DE	; STKEND ZURÜCKHOLEN
3292 C9	RET	
3293		
3293	; SUBROUTINE ZUM ABSPEICHERN VON 2 INTEGERZAHLEN	
3293	; IN FLOATINGPOINTFORM AUF DEM CALC.-STACK	
3293		
3293		
3293 CD 96 32	CALL \$3296	; SUBROUTINE ZUM SPEICHERN
3296 EB	EX DE,HL	; 1 ZAHL AUFRUFEN UND NACH
3297		; ZEIGERTAUSCH DIESE NOCH
3297		; EINMAL AUSFÜHREN (2. ZAHL)
3297		
3297		
3297		
3297		
3297		
3297 7E	LD A,(HL)	; WENN DAS ERSTE BYTE NICHT
3298 A7	AND A	; NULL IST, IST ES KEINE
3299 C0	RET NZ	; INTEGERZAHL: RETURN
329A D5	PUSH DE	; ZEIGER RETTEN
329B CD 7F 2D	CALL \$2D7F	; DAS VORZEICHEN IN C UND
329E		; UND DIE ZAHL IN DE HOLEN
329E AF	XOR A	; REG A LÖSCHEN
329F 23	INC HL	; DAS FÜNFTE
32A0 77	LD (HL),A	
32A1 2B	DEC HL	; UND DAS VIERTE BYTE
32A2 77	LD (HL),A	; AUF NULL SETZEN
32A3 06 91	LD B,\$91	; B AUF \$91 FÜR MAXIMAL 16
32A5		; INTEGERBITS SETZEN
32A5 7A	LD A,D	; WENN D NULL IST, WERDEN NUR
32A6 A7	AND A	; B BIT BENÖTIGT
32A7 20 08	JR NZ,\$32B1	; SPRUNG BEI MEHR ALS 8
32A9 B3	OR E	; E EBENFALLS PRÜFEN
32AA 42	LD B,D	; B AUF 0 SETZEN
32AB 28 10	JR Z,\$32BD	; DIE GANZE ZAHL IST NULL
32AD 53	LD D,E	; E NACH D LADEN UND
32AE 58	LD E,B	; E MIT NULL LADEN
32AF 06 89	LD B,\$89	; B AUF \$89 FÜR MAX. 8 BIT
32B1		; SETZEN (B = EXPONENT)
32B1 EB	EX DE,HL	; ZEIGER NACH DE, ZAHL NACH HL
32B2 05	DEC B	; EXPONENT BEI JEDEM SHIFT
32B3 29	ADD HL,HL	; UM 1 VERMINDERN, BIS
32B4 30 FC	JR NC,\$32B2	; ÜBERLAUF AUFTRETT
32B6 CB 09	RRC C	; VORZEICHEN INS CARRY
32B8 CB 1C	RR H	; DAS VORZEICHEN IN DIE
32BA CB 1D	RR L	; ZAHL EINKOPIEREN
32BC EB	EX DE,HL	; ZAHL UND ZEIGER TAUSCHEN
32BD 2B	DEC HL	

```

32BE 73      LD (HL),E      ; IM DRITTEN BYTE E UND
32BF 2B      DEC HL
32C0 72      LD (HL),D      ; IM ZWEITEN D ABSPEICHERN
32C1 2B      DEC HL
32C2 70      LD (HL),B      ; EXPONENT INS ERSTE BYTE
32C3 D1      POP DE        ; STKEND ZURÜCKHOLEN
32C4 C9      RET

;
;=====
;
; AB HIER UNTERPROGRAMME UND TABELLEN DES
; FLOATING-POINT-CALCULATORS
;
; TABELLE MIT DEN KONSTANTEN NULL, EINS, 0.5, PI/2, ZEHN
;
32C5 00      .BYT $00,$B0,$00 ; NULL
32C6 B0
32C7 00
32C8 40      .BYT $40,$B0,$00,$01 ; EINS
32C9 B0
32CA 00
32CB 01      .BYT $30,$00      ; 0.5
32CC 30
32CD 00      .BYT $F1,$49,$0F,$DA,$A2 ; PI/2
32CE F1
32CF 49
32D0 0F
32D1 DA
32D2 A2      .BYT $40,$B0,$00,$0A ; ZEHN
32D3 40
32D4 B0
32D5 00
32D6 0A
32D7
32D7
32D7
32D7
32D7
;
; TABELLE DER 'OPCODES', DIE DEM 'RST $23'-
; BEFEHL (CALCULATORAUFRUF) FOLGEN MIT DEN
; DAZUGEHÖRIGEN SPRUNGADRESSSEN
;
;
32D7 8F 36   .WOR $368F      ; 00: SPRUNG, WENN WAHR
32D9 3C 34   .WOR $343C      ; 01: TAUSCHEN
32DB A1 33   .WOR $33A1      ; 02: LÖSCHEN
32DD 0F 30   .WOR SUBTRA      ; 03: SUBTRAHIEREN
32DF CA 30   .WOR MULTIP      ; 04: MULTIPLIZIEREN
32E1 AF 31   .WOR $31AF      ; 05: DIVIDIEREN
32E3 51 38   .WOR $3851      ; 06: HOCH 2 ('2)
32E5 1B 35   .WOR $351B      ; 07: ODER
32E7 24 35   .WOR $3524      ; 08: LOGISCHES UND
32E9 3B 35   .WOR $353B      ; 09: UNGLEICH
32EB 3B 35   .WOR $353B      ; 0A: KLEINER ALS
32ED 3B 35   .WOR $353B      ; 0B: STRINGS UNGLEICH
32EF 3B 35   .WOR $353B      ; 0C: NICHT GRÖßER
32F1 3B 35   .WOR $353B      ; 0D: NICHT KLEINER
32F3 3B 35   .WOR $353B      ; 0E: STRINGS GLEICH
32F5 14 30   .WOR ADDIER      ; 0F: ADDIEREN
32F7 2D 35   .WOR $352D      ; 10: STRING AND NUMBER
32F9 3B 35   .WOR $353B      ; 11: STRING '='
32FB 3B 35   .WOR $353B      ; 12: STRING '>'

```

32FD 3B 35
 32FF 3B 35
 3301 3B 35
 3303 3B 35
 3305 9C 35
 3307 DE 35
 3309 BC 34
 330B 45 36
 330D 6E 34
 330F 69 36
 3311 DE 35
 3313 74 36
 3315 B5 37
 3317 AA 37
 3319 DA 37
 331B 33 38
 331D 43 38
 331F E2 37
 3321 13 37
 3323 C4 36
 3325 AF 36
 3327 4A 38
 3329 92 34
 332B 6A 34
 332D AC 34
 332F A5 34
 3331 B3 34
 3333 1F 36
 3335 C9 35
 3337 01 35
 3339 C0 33
 333B A0 36
 333D B6 36
 333F C6 33
 3341 7A 36
 3343 06 35
 3345 F9 34
 3347 9B 36
 3349 B3 37
 334B 14 32
 334D A2 33
 334F 4F 2D
 3351 97 32
 3353 49 34
 3355 1B 34
 3357 2D 34
 3359 0F 34

.WOR \$353B ; 13: STRINGS <>
 .WOR \$353B ; 14: STRING >
 .WOR \$353B ; 15: STRING <
 .WOR \$353B ; 16: STRING =
 .WOR \$359C ; 17: STRINGADDITION
 .WOR \$35DE ; 18: VAL\$
 .WOR \$34BC ; 19: USR\$
 .WOR \$3645 ; 1A: READIN
 .WOR \$346E ; 1B: NEGIEREN
 .WOR \$3669 ; 1C: CODE
 .WOR \$35DE ; 1D: VAL
 .WOR \$3674 ; 1E: LEN
 .WOR \$37B5 ; 1F: SINUS
 .WOR \$37AA ; 20: COSINUS
 .WOR \$37DA ; 21: TANGENS
 .WOR \$3833 ; 22: ARCUSSINUS
 .WOR \$3843 ; 23: ARCUSCOSINUS
 .WOR \$37E2 ; 24: ARCUSTANGENS
 .WOR \$3713 ; 25: LN (LOGARITHMUS)
 .WOR \$36C4 ; 26: EXP (E HOCH X)
 .WOR \$36AF ; 27: INT(EGER)
 .WOR \$384A ; 28: SQR (WURZEL)
 .WOR \$3492 ; 29: SGN (VORZEICHEN)
 .WOR ABSOLU ; 2A: ABS (BETRAG)
 .WOR \$34AC ; 2B: PEEK
 .WOR \$34A5 ; 2C: IN
 .WOR \$34B3 ; 2D: USR MIT ZAHL
 .WOR \$361F ; 2E: STR\$
 .WOR \$35C9 ; 2F: CHR\$
 .WOR \$3501 ; 30: NOT
 .WOR VERDO ; 31: DUPLIZIEREN
 .WOR \$36A0 ; 32: N MOD M-DIVISION
 .WOR \$36B6 ; 33: SPRUNG
 .WOR \$33C6 ; 34: DATEN AUF STACK
 .WOR \$367A ; 35: DJNZ
 .WOR \$3506 ; 36: KLEINER 0
 .WOR \$34F9 ; 37: GRÖßER 0
 .WOR \$369B ; 38: ENDE CALCULATOR
 .WOR \$37B3 ; 39: HOLE ARGUMENT
 .WOR \$3214 ; 3A: ABSCHNEIDEN MIT RUNDEN
 .WOR \$33A2 ; 3B: FP-CALC-2
 .WOR \$2D4F ; 3C: IN FLOATING-ZAHL WANDELN
 .WOR \$3297 ; 3D: WIEDER AUF STACK LEGEN
 .WOR \$3449 ; 3E: POLYNOMENTWICKLUNG
 .WOR \$341B ; 3F: NULL AUF STACK
 .WOR \$342D ; 40: SPEICHERE IN MEMO ETC.
 .WOR \$340F ; 41: HOLE VON MEMO ETC.

; CALCULATOR:

; AUFRUF NORMAL DURCH RST \$2B (RST CALRUF).
 ; HINTER DEM RST \$2B BEFEHL FOLGEN EIN ODER MEHRERE
 ; BYTES, DIE DIE AUSZUFÜHRENDEN OPERATIONEN GEMÄSS
 ; OBENSTEHENDER TABELLE BESTIMMEN (MAN KANN DIESE
 ; BYTES ALS PSEUDO-OPERATIONSCODES BEZEICHNEN).
 ; DIE RECHENOPERATIONEN BEZIEHEN SICH IM ALLGEMEINEN
 ; AUF DEN ODER DIE 'LETZTEN WERT(E)' IM CALCULATOR-
 ; STACK. EIN LETZTER WERT (GENAU 5 BYTES) KANN EINE

335B			; FLOATINGPOINTZAHL ODER STRINGPARAMETER SEIN.
335B			; ZUSÄTZLICH BENUTZT DER CALCULATOR EINIGE SPEICHER-
335B			; PLATZE ZUR ZWISCHENSPEICHERUNG VON TEILERGEBNISSEN
335B			; ETC., DIE MIT MEMO - MEMS BEZEICHNET SIND UND
335B			; JEWEILS 5 BYTES UMFASSEN.
335B			
335B	CD BF 35	RECHNE CALL \$35BF	; IN HL DIE ADRESSE VOM
335E			; LETZTEN WERT UND IN DE
335E			; STKEND LADEN
335E	78	LD A,B	; EINEN EINFACHEN OPERATIONS-
335F			; OFFSET ODER, BEI REKURSIVEM
335F	32 67 5C	LD (BREG),A	; AUFRUF, DEN ZAHLSTAND
3362			; NACH BREG LADEN
3362	D9	RECH2 EXX	; RETURNADRESSE IN HL SPEICHERN
3363	E3	EX (SP),HL	; RECH2 IST DER EINSTIEG
3364	D9	EXX	; FÜR REKURSIVE AUFRUFE. BEI
3365			; DENEN BREG NICHT VERANDERT
3365			; WERDEN DARF (ZÄHLER)
3365	ED 53 65 5C CLOOP	LD (STKEND),DE	; EINSTIEG, UM DIE EINZELNEN
3369			; OPCODES ABZUARBEITEN, DAZU
3369			; ERST STKEND NEU SETZEN
3369	D9	EXX	; 2. REGISTERSATZ EINSCHALTEN
336A	7E	LD A,(HL)	; AKTUELLEN OPCODE LADEN
336B	23	INC HL	; ZEIGER DARAUF +1
336C	E5	PUSH HL	; UND ZWISCHENSPEICHERN
336D	A7	AND A	; A TESTEN, UM DIE EINFACHEN
336E			; OPCODES VON DEN MEHRFACHEN
336E			; ZU TRENNEN
336E	F2 80 33	JP P,\$3380	; SPRUNG BEI EINFACHEN
3371	57	LD D,A	; OPCODE IN D RETTEN
3372	E6 60	AND \$60	; NUR BIT 5 UND 6
3374	0F	RRCA	; BETRACHTEN UND
3375	0F	RRCA	; NACH BIT 1 UND 2
3376	0F	RRCA	; SCHIEBEN
3377	0F	RRCA	
3378	C6 7C	ADD \$7C	; 2*\$3E ALS BASIS ADDIEREN
337A	6F	LD L,A	; FÜR TABELLE NACH L
337B	7A	LD A,D	; BIT 0 - 4 AUSBLENDEN
337C	E6 1F	AND \$1F	
337E	18 0E	JR \$338E	; UND ZUR ADRESSUCHE
3380			
3380	FE 18	CP \$18	; SPRUNG, WENN NUR EINE
3382	30 08	JR NC,\$338C	; OPERATION, Z. B. SIN(X),
3384			; AUSZUFÜHREN IST
3384	D9	EXX	; BEI DOPPELOPERATIONEN (+ ETC.)
3385	01 FB FF	LD BC,\$FFFB	; MUSS HL AUF DEN ERSTEN
3388	54	LD D,H	; UND DE AUF DEN
3389	5D	LD E,L	; ZWEITEN OPERANDEN
338A	09	ADD HL,BC	; ZEIGEN
338B	D9	EXX	; 2. REGISTERSATZ EINSCHALTEN
338C	07	RLCA	; A*2 FÜR SPRUNGADRESSEN
338D	6F	LD L,A	; NACH HL OFFSET LADEN UND
338E	11 D7 32	LD DE,\$32D7	; NACH DE DIE TABELLEN-
3391	26 00	LD H,0	; ADRESSE
3393	19	ADD HL,DE	; ADRESSE DER BENÖTIGTEN
3394	5E	LD E,(HL)	; SPRUNGADRESSE BERECHNEN
3395	23	INC HL	; UND DIE SPRUNGADRESSE
3396	56	LD D,(HL)	; NACH DE LADEN


```

32FD 3B 35
32FF 3B 35
3301 3B 35
3303 3B 35
3305 9C 35
3307 DE 35
3309 BC 34
330B 45 36
330D 6E 34
330F 69 36
3311 DE 35
3313 74 36
3315 B5 37
3317 AA 37
3319 DA 37
331B 33 38
331D 43 38
331F E2 37
3321 13 37
3323 C4 36
3325 AF 36
3327 4A 38
3329 92 34
332B 6A 34
332D AC 34
332F A5 34
3331 B3 34
3333 1F 36
3335 C9 35
3337 01 35
3339 C0 33
333B A0 36
333D 86 36
333F C6 33
3341 7A 36
3343 06 35
3345 F9 34
3347 9B 36
3349 B3 37
334B 14 32
334D A2 33
334F 4F 2D
3351 97 32
3353 49 34
3355 1B 34
3357 2D 34
3359 0F 34
335B
335B
335B
335B
335B
335B
335B
335B
335B
335B
335B

```

```

.WOR $353B
.WOR $353B
.WOR $353B
.WOR $353B
.WOR $359C
.WOR $35DE
.WOR $34BC
.WOR $3645
.WOR $346E
.WOR $3669
.WOR $35DE
.WOR $3674
.WOR $37B5
.WOR $37AA
.WOR $37DA
.WOR $3833
.WOR $3843
.WOR $37E2
.WOR $3713
.WOR $36C4
.WOR $36AF
.WOR $384A
.WOR $3492
.WOR ABSOLU
.WOR $34AC
.WOR $34A5
.WOR $34B3
.WOR $361F
.WOR $35C9
.WOR $3501
.WOR VERDO
.WOR $36A0
.WOR $3686
.WOR $33C6
.WOR $367A
.WOR $3506
.WOR $34F9
.WOR $369B
.WOR $3783
.WOR $3214
.WOR $33A2
.WOR $2D4F
.WOR $3297
.WOR $3449
.WOR $341B
.WOR $342D
.WOR $340F

```

```

; 13: STRINGS <>
; 14: STRING >
; 15: STRING <
; 16: STRING =
; 17: STRINGADDITION
; 18: VAL$
; 19: USR$
; 1A: READIN
; 1B: NEGIEREN
; 1C: CODE
; 1D: VAL
; 1E: LEN
; 1F: SINUS
; 20: COSINUS
; 21: TANGENS
; 22: ARCUSSINUS
; 23: ARCUSCOSINUS
; 24: ARCUSTANGENS
; 25: LN (LOGARITHMUS)
; 26: EXP (E HOCH X)
; 27: INT(EGER)
; 28: SQR (WURZEL)
; 29: SGN (VORZEICHEN)
; 2A: ABS (BETRAG)
; 2B: PEEK
; 2C: IN
; 2D: USR MIT ZAHL
; 2E: STR$
; 2F: CHR$
; 30: NOT
; 31: DUPLIZIEREN
; 32: N MOD M-DIVISION
; 33: SPRUNG
; 34: DATEN AUF STACK
; 35: DJNZ
; 36: KLEINER 0
; 37: GRÖßER 0
; 38: ENDE CALCULATOR
; 39: HOLE ARGUMENT
; 3A: ABSCHNEIDEN MIT RUNDEN
; 3B: FP-CALC-2
; 3C: IN FLOATING-ZAHL WANDELN
; 3D: WIEDER AUF STACK LEGEN
; 3E: POLYNOMENTWICKLUNG
; 3F: NULL AUF STACK
; 40: SPEICHERE IN MEMO ETC.
; 41: HOLE VON MEMO ETC.

```

; CALCULATOR:

```

; AUFRUF NORMAL DURCH RST $2B (RST CALRUF).
; HINTER DEM RST $2B BEFEHL FOLGEN EIN ODER MEHRERE
; BYTES, DIE DIE AUSZUFÜHRENDEN OPERATIONEN GEMASS
; OBERSTEHENDER TABELLE BESTIMMEN (MAN KANN DIESE
; BYTES ALS PSEUDO-OPERATIONSCODES BEZEICHNEN).
; DIE RECHENOPERATIONEN BEZIEHEN SICH IM ALLGEMEINEN
; AUF DEN ODER DIE 'LETZTEN WERT(E)' IM CALCULATOR-
; STACK. EIN LETZTER WERT (GENAU 5 BYTES) KANN EINE

```

335B		; FLOATINGPOINTZAHL ODER STRINGPARAMETER SEIN.	
335B		; ZUSATZLICH BENUTZT DER CALCULATOR EINIGE SPEICHER-	
335B		; PLATZE ZUR ZWISCHENSPEICHERUNG VON TEILERGEBNISSEN	
335B		; ETC., DIE MIT MEMO - MEMS BEZEICHNET SIND UND	
335B		; JEWEILS 5 BYTES UMFASSEN.	
335B		;	
335B	CD BF 35	RECHNE CALL \$35BF	; IN HL DIE ADRESSE VOM
335E			; LETZTEN WERT UND IN DE
335E			; STKEND LADEN
335E	7B	LD A,B	; EINEN EINFACHEN OPERATIONS-
335F			; OFFSET ODER, BEI REKURSIVEM
335F	32 67 5C	LD (BREG),A	; AUFRUF, DEN ZAHLSRAND
3362			; NACH BREG LADEN
3362	D9	RECH2 EXX	; RETURNADRESSE IN HL SPEICHERN
3363	E3	EX (SP),HL	; RECH2 IST DER EINSTIEG
3364	D9	EXX	; FÜR REKURSIVE AUFRUFE, BEI
3365			; DENEN BREG NICHT VERANDERT
3365			; WERDEN DARF (ZÄHLER)
3365	ED 53 65 5C CLOOP	LD (STKEND),DE	; EINSTIEG, UM DIE EINZELNEN
3369			; OPCODES ABZUARBEITEN, DAZU
3369			; ERST STKEND NEU SETZEN
3369	D9	EXX	; 2. REGISTERSATZ EINSCHALTEN
336A	7E	LD A,(HL)	; AKTUELLEN OPCODE LADEN
336B	23	INC HL	; ZEIGER DARAUF +1
336C	E5	PUSH HL	; UND ZWISCHENSPEICHERN
336D	A7	AND A	; A TESTEN, UM DIE EINFACHEN
336E			; OPCODES VON DEN MEHRFACHEN
336E			; ZU TRENNEN
336E	F2 80 33	JP P,\$3380	; SPRUNG BEI EINFACHEN
3371	57	LD D,A	; OPCODE IN D RETTEN
3372	E6 60	AND \$60	; NUR BIT 5 UND 6
3374	0F	RRCA	; BETRACHTEN UND
3375	0F	RRCA	; NACH BIT 1 UND 2
3376	0F	RRCA	; SCHIEBEN
3377	0F	RRCA	
337B	C6 7C	ADD \$7C	; 2*\$3E ALS BASIS ADDIEREN
337A	6F	LD L,A	; FÜR TABELLE NACH L
337B	7A	LD A,D	; BIT 0 - 4 AUSBLENDEN
337C	E6 1F	AND \$1F	
337E	1B 0E	JR \$338E	; UND ZUR ADRESSUCHE
3380			
3380	FE 1B	CP \$1B	; SPRUNG, WENN NUR EINE
3382	30 0B	JR NC,\$338C	; OPERATION, Z. B. SIN(X),
3384			; AUSZUFÜHREN IST
3384	D9	EXX	; BEI DOPPELOPERATIONEN (+ ETC.)
3385	01 FB FF	LD BC,\$FFFB	; MUSS HL AUF DEN ERSTEN
338B	54	LD D,H	; UND DE AUF DEN
3389	5D	LD E,L	; ZWEITEN OPERANDEN
338A	09	ADD HL,BC	; ZEIGEN
338B	D9	EXX	; 2. REGISTERSATZ EINSCHALTEN
338C	07	RLCA	; A*2 FÜR SPRUNGADRESSEN
338D	6F	LD L,A	; NACH HL OFFSET LADEN UND
338E	11 D7 32	LD DE,\$32D7	; NACH DE DIE TABELLEN-
3391	26 00	LD H,0	; ADRESSE
3393	19	ADD HL,DE	; ADRESSE DER BENÖTIGTEN
3394	5E	LD E,(HL)	; SPRUNGADRESSE BERECHNEN
3395	23	INC HL	; UND DIE SPRUNGADRESSE
3396	56	LD D,(HL)	; NACH DE LADEN

3397	21 65 33	LD HL,CLOOP	; RÜCKEHRADRESSE IN CALC.-
339A			; ROUTINE LADEN
339A	E3	EX (SP),HL	; AUF STACK LEGEN UND HL
339B			; WIEDER ZURÜCKHOLEN
339B	D5	PUSH DE	; SUBROUTINADRESSE FÜR RETURN
339C			; AUF DEN STACK SPEICHERN
339C	D9	EXX	; NORMALER REGISTERSATZ
339D	ED 4B 66 5C	LD BC,(STKEND+1)	; BREG NACH B LADEN
33A1	C9	RET	; INDIREKTEN SPRUNG IN DIE
33A2			; ENTSPRECHENDE ROUTINE.
33A2			; SUBROUTINE ZUM LÖSCHEN DES LETZTEN WERTS IM
33A2			; CALC.-STACK (OPCODE #02). DER AUFRUF FÜHRT
33A2			; NUR AUF DAS OBIGE 'RET', SODASS NUR HL ALS
33A2			; ZEIGER AUF DEN LETZTEN WERT UM 5 BYTES (= 1
33A2			; CALC.-VARIABLE) VERMINDERT WIRD, ALSO AUF
33A2			; DEN VORLETZTEN ZEIGT. DADURCH WIRD DER BIS-
33A2			; HERIGE LETZTE WERT NICHT MEHR ADRESSIERT
33A2			; UND BEI DER NÄCHSTEN OPERATION ÜBERSCHRIEBEN.
33A2			; SUBROUTINE UM EINE OPERATION AUSZUFÜHREN (AUFRUF
33A2			; BEI DER ENTWICKLUNG VON AUSDRÜCKEN, OPCODE #3B)
33A2			; POP AF
33A2	F1	LD A,(BREG)	; RETURNADRESSE CLOOP WEG-
33A3	3A 67 5C	EXX	; WERFEN UND OPCODE LADEN
33A6	D9	JR #336C	; 2. REGISTERSATZ EINSCHALTEN
33A7	1B C3		; BENÖTIGTE ADRESSE BESTIMMEN
33A9			; SUBROUTINE ZUM ÜBERPRÜFEN, OB NOCH 5 SPEICHER-
33A9			; PLATZE (= 1 FLOATINGPOINTZAHL) FREI SIND
33A9			; PLATZ5 PUSH DE
33A9	D5	PUSH HL	; DE UND
33AA	E5	LD BC,5	; HL ZWISCHENSPEICHERN
33AB	01 05 00	CALL #1F05	; DEN TEST FÜR DIE 5 NOT-
33AE	CD 05 1F	POP HL	; WENDIGEN PLATZE MACHEN
33B1	E1	POP DE	; DIE REGISTER WIEDER
33B2	D1	RET	; ZURÜCKLADEN
33B3	C9		
33B4			; SUBROUTINE, UM 1 ZAHL AUF DEN STACK ZU
33B4			; BRINGEN
33B4			; LD DE,(STKEND)
33B4	ED 5B 65 5C	CALL VERDO	; STKEND LADEN
33B8	CD C0 33	LD (STKEND),DE	; DEN TRANSFER AUSFÜHREN
33BB	ED 53 65 5C	RET	; STKEND NEU SETZEN
33BF	C9		
33C0			; SUBROUTINE ZUM UMSPEICHERN EINER FLOATING-
33C0			; POINTZAHL (AUCH OPCODE #31= VERDOPPELN)
33C0			; VERDO CALL PLATZ5
33C0	CD A9 33	LDIR	; SPEICHERPLATZTEST
33C3	ED B0	RET	; TRANSFER DURCHFÜHREN
33C5	C9		
33C6			; SUBROUTINE ZUM ABSPEICHERN EINER FLOATINGPOINT-
33C6			; ZAHL, DIE DEM OPCODE #34 DIREKT FOLGT. DAS 1.
33C6			; BYTE, WELCHES DER EXPONENT IST, GIBT HIERBEI
33C6			; DIE ANZAHL DER NOCH FOLGENDEN MANTISSEN-

33C6		; STELLEN AN (BIT 7 UND 6), DER REST WIRD MIT
33C6		; NULLEN AUFGEFÜLLT
33C6		
33C6	62	LD H,D ; HL FÜR DEN NEUEN LETZTEN
33C7	6B	LD L,E ; WERT SETZEN
33C8	CD A9 33	CALL PLATZ5 ; SPEICHER TESTEN
33C8	D9	EXX ; DEN ZEIGER AUF DEN
33CC	E5	PUSH HL ; NÄCHSTEN OPCODE IN
33CD	D9	EXX ; HL RETTEN
33CE	E3	EX (SP),HL ; OPCODEADRESSE MIT ERGEBNIS-
33CF		; ZEIGER TAUSCHEN
33CF	C5	PUSH BC ; BC ZWISCHENSPEICHERN
33D0	7E	LD A,(HL) ; EXPONENT LADEN UND DIE
33D1	E6 C0	AND \$C0 ; NACHFOLGENDE BYTEZAHL
33D3	07	RLCA ; DURCH DIVISION MIT \$40
33D4	07	RLCA ; ERMITTELN
33D5	4F	LD C,A ; ERGEBNIS NACH C UND
33D6	0C	INC C ; UM 1 ERHÖHEN
33D7	7E	LD A,(HL) ; EXPONENT LADEN
33D8	E6 3F	AND \$3F ; BITS 5 - 0 NEHMEN
33DA	20 02	JR NZ,\$33DE ; ZUR EXPONENTERMITTLUNG,
33DC		; FALLS <0, ANDERNFALLS
33DC	23	INC HL ; BESTIMMT DAS NÄCHSTE BYTE
33DD	7E	LD A,(HL) ; DEN EXPONENTEN
33DE	C6 50	ADD \$50 ; DEN RICHTIGEN EXPONENT
33E0	12	LD (DE),A ; ERMITTELN UND SPEICHERN
33E1	3E 05	LD A,5 ; ANZAHL DER AUFZUFÜLLENDEN
33E3	91	SUB C ; NULLEN IN A BERECHNEN
33E4	23	INC HL ; HL UND DE AUF DAS
33E5	13	INC DE ; 1. BYTE UND BC ALS
33E6	06 00	LD B,0 ; ZÄHLER SETZEN
33E8	ED B0	LDIR ; UMSPEICHERN
33EA	C1	POP BC ; BC VOM STACK ZURÜCK
33EB	E3	EX (SP),HL ; ERGEBNISZEIGER IN HL LADEN
33EC	D9	EXX ; ZEIGER AUF NÄCHSTEN OPCODE
33ED	E1	POP HL ; NACH HL BRINGEN
33EE	D9	EXX
33EF	47	LD B,A ; B ALS ZÄHLER FÜR NULLEN
33F0	AF	XOR A ; A LÖSCHEN
33F1	05	DEC B ; ZÄHLER -1
33F2	C8	RET Z ; KEINE NULL MEHR SPEICHERN
33F3	12	LD (DE),A ; 1 NULL EINSCHREIBEN
33F4	13	INC DE ; ZEIGER (STKEND) ERHÖHEN
33F5	1B FA	JR \$33F1
33F7		
33F7		; SUBROUTINE ZUM SUCHEN DER KONSTANTEN IN
33F7		; DER CALCULATOR TABELLE. DIE NUMMER MUSS IN
33F7		; A STEHEN
33F7		
33F7	A7	AND A ; WENN DIE GESUCHTE KON-
33F8	C8	RET Z ; STANTE GEFUNDEN, RETURN
33F9	F5	PUSH AF ; ZÄHLER ZWISCHENSPEICHERN
33FA	D5	PUSH DE ; DE EBENFALLS
33FB	11 00 00	LD DE,0 ; ES WIRD EIN BLINDES
33FE	CD C8 33	CALL \$33C8 ; LADEN EINER KONSTANTEN
3401		; IN DAS ROM DURCHFÜHRT,
3401		; DAMIT HL BIS ZUR RICHTIGEN
3401		; ADRESSE GEZÄHLT WIRD

```

3401      POP DE      ; DE UND A WIEDER
3401 D1      POP AF      ; LADEN
3402 F1      DEC A      ; ZÄHLER -1
3403 3D      JR $33FB    ; TESTEN, OB GEFUNDEN
3404 1B F2
3406      ;
3406      ; SUBROUTINE ZUM AUSRECHNEN DER ADRESSE EINES
3406      ; 5 BYTE-BEREICHS IN DEM CALCULATORSPEICHER-
3406      ; BEREICH (MEMO BIS MEM5)
3406      ;
3406      LD C,A      ; MEM-NUMMER NACH C
3406 4F      RLCA      ; MIT 4 MULTIPLIZIEREN
3407 07      RLCA
3408 07      ADD C      ; C ADDIEREN, ALSO 5*NUMMER
3409 B1      LD C,A      ; ERGEBNIS NACH
340A 4F      LD B,0      ; BC KOPIEREN UND
340B 06 00    ADD HL,BC  ; ZU HL ADDIEREN
340D 09      RET
340E C9
340F      ;
340F      ; EINE VARIABLE VOM CALCULATORSPEICHER (MEMO
340F      ; BIS MEM5) IN DEN CALC.STACK HOLEN (OPCODES
340F      ; $E0 BIS $E5) ALS LETZTER WERT
340F      ;
340F      PUSH DE      ; ZEIGER AUF ERGEBNIS RETTEN
340F D5      LD HL,(MEM) ; BASISADRESSE DES MEM-BEREICHS
3410 2A 6B 5C CALL $3406 ; ADRESSE VON MEM-X SUCHEN
3413 CD 06 34 CALL $33C0 ; TRANSFER DURCHFÜHREN
3416 CD C0 33 POP HL      ; ZEIGER AUF 1. BYTE DES
3419 E1      RET        ; LETZTEN WERTES IN HL
341A C9
341B      ;
341B      ; SUBROUTINE ZUM ABLÉGEN EINER KONSTANTEN AUF
341B      ; DEM CALC.-STACK (OPCODES $A0 - $A4)
341B      ;
341B      LD H,D      ; HL AUF DIE ADRESSE
341B 62      LD L,E      ; DES ERGEBNIS SETZEN
341C 6B      EXX      ; ZEIGER AUF NÄCHSTEN
341D D9      PUSH HL    ; OPCODE RETTEN
341E E5      LD HL,$32C5 ; BASISADRESSE DER KON-
341F 21 C5 32 ; STANTENTABELLE LADEN
3422      EXX
3422 D9      CALL $33F7   ; BENÖTIGTE KONSTANTE SUCHEN
3423 CD F7 33 CALL $33C8 ; IN DEN CALC.-STACK KOPIEREN
3426 CD C8 33 EXX      ; ZEIGER AUF NÄCHSTEN
3429 D9      POP HL     ; OPCODE WIEDER NACH HL
342A E1      EXX
342B D9      RET
342C C9      .END
342D      .LIB SPEC3400-S
342D      ; SINCLAIR ZX SPECTRUM SPEC3400-S
342D      ;
342D      ; SUBROUTINE ZUM SPEICHERN DES LETZTEN WERTS
342D      ; IN MEMO BIS MEM5 (OPCODES $C0 - $C5)
342D      ;
342D      PUSH HL      ; ZEIGER AUF LETZTEN WERT
342D E5      EX DE,HL   ; RETTEN UND NACH DE KOPIEREN
342E EB      LD HL,(MEM) ; BASISADRESSE DES MEM-BEREICHS
342F 2A 6B 5C CALL $3406 ; ZIELADRESSE AUSRECHNEN
3432 CD 06 34 EX DE,HL ; FÜR DEN TRANSFER MUSS ZIEL-
3433 EB

```



```

3436 CD C0 33      CALL $33C0      ; UND QUELLADRESSE GETAUSCHT
3439              ; WERDEN
3439 EB            EX DE,HL        ; STKEND (=LETZTER WERT +5)
343A              ; WIEDER NACH DE LADEN
343A E1            POP HL         ; ZEIGER AUF LETZTEN WERT
343B C9            RET            ; NACH HL ZURÜCKHOLEN
343C              ;
343C              ; SUBROUTINE ZUM AUSTAUSCHEN DER BEIDEN LETZTEN
343C              ; WERTE IM CALC.-STACK
343C              ;
343C 06 05          Tausch LD B,5    ; B ALS ZÄHLER SETZEN
343E 1A            LD A,(DE)        ; 1 BYTE DER ZWEITEN UND
343F 4E            LD C,(HL)        ; 1 BYTE DER ERSTEN ZAHL
3440 EB            EX DE,HL        ; ZEIGER TAUSCHEN
3441 12            LD (DE),A        ; BEIDE BYTES
3442 71            LD (HL),C        ; ZURÜCKSCHREIBEN
3443 23            INC HL           ; DIE BEIDEN ZEIGER
3444 13            INC DE           ; UM 1 ERHÖHEN
3445 10 F7          DJNZ $343E      ; 5 BYTES NOCH NICHT
3447              ; AUSGETAUSCHT
3447 EB            EX DE,HL        ; ZEIGER KORRIGIEREN
3448 C9            RET
3449              ;
3449              ; SUBROUTINE ZUM GENERIEREN VON POLYNOMEN
3449              ; BEI FUNKTIONEN WIE SINUS, ATN ETC.
3449              ; (OPCODES $86,$88,$8C = $3E IN TABELLE)
3449              ;
3449 47            LD B,A            ; PARAMETER NACH B LADEN
344A CD 5E 33      CALL $335E      ; UND DAMIT BREG ALS
344D              ; ZÄHLER SETZEN
344D              ;
344D              ; DEN LETZTEN WERT 'Z' VORBEREITEN
344D              ;
344D 31            .BYT $31          ; Z,Z, VERDOPPELN
344E 0F            .BYT $0F          ; 2*Z, ADDIEREN
344F C0            .BYT $C0          ; 2*Z, NACH MEM0 SPEICHERN
3450 02            .BYT $02          ; - LÖSCHEN
3451 A0            .BYT $A0          ; 0, NULL IN CALC.-STACK
3452 C2            .BYT $C2          ; 0, NACH MEM2 SPEICHERN
3453              ;
3453              ; IN DER FOLGENDEN SCHLEIFE WERDEN DIE
3453              ; KOEFFIZIENTEN BERECHNET:
3453              ;  $B(M) = 2*Z*B(M-1) - B(M-2) + C(M)$ , WOBEI
3453              ;  $M = 0 \dots N$ . DIE KONSTANTEN  $C(M)$  STEHEN
3453              ; HINTER DEM JEWELIGEN AUFRUF DIESER
3453              ; ROUTINE
3453              ;
3453 31            POLYS .BYT $31      ; B(M),B(M), VERDOPPELN
3454 E0            .BYT $E0          ; B(M),B(M), 2*Z, MEM0 LADEN
3455 04            .BYT $04          ; B(M), 2*Z*B(M), MULTIPLIZIEREN
3456 E2            .BYT $E2          ; B(M), 2*Z*B(M), B(M-1) MEM2 LADEN
3457 C1            .BYT $C1          ; B(M-1) NACH MEM1 SCHREIBEN
3458 03            .BYT $03          ; B(M), 2*Z*B(M)-B(M-1)
3459 3B            .BYT $3B          ; ENDE
345A              ;
345A              ; NÄCHSTE KONSTANTE IN DEN CALC.-STACK LADEN
345A              ;
345A              ;
345A CD C6 33      CALL $33C6      ; B(M), 2*Z*B(M)-B(M-1), C(M+1)

```

3450	CD 62 33	CALL \$3362	; WIEDERAUFRUF DES CALCULATORS,
3460			; OHNE BREG ZU VERÄNDERN
3460	OF	.BYT \$0F	; $B(M), 2 \cdot Z \cdot B(M) - B(M-1) + C(M+1)$
3461	01	.BYT \$01	; $2 \cdot Z \cdot B(M) - B(M-1) + C(M+1), B(M)$
3462	C2	.BYT \$C2	; B(M) NACH MEM2 SPEICHERN
3463	02	.BYT \$02	; $B(M+1) = 2 \cdot Z \cdot B(M) - B(M-1) + C(M+1)$
3464	35	.BYT \$35	; B(M+1), SCHLEIFENZÄHLER -1
3465	EE	.BYT \$EE	; SPRUNG NACH POLYS, WENN
3466			; NOCH NICHT FERTIG
3466	E1	.BYT \$E1	; B(M), B(M-2), MEM2 LADEN
3467	03	.BYT \$03	; KOEFF. = $B(M) - B(M-2)$
3468	38	.BYT \$38	; ENDE
3469	C9	RET	
346A			
346A			; DIE 'ABS(-OLUT)'-FUNKTION (OPCODE \$2A)
346A			
346A	06 FF	ABSOLU LD B,\$FF	; B INITIALISIEREN UND IN
346C	18 06	JR \$3474	; DIE NEGIERROUTINE SPRINGEN
346E			
346E			; SUBROUTINE ZUM WECHSELN DES VORZEICHENS
346E			; DES LETZTEN WERTS IM CALC.-STACK
346E			; (NEGIEREN = OPCODE \$1B)
346E			
346E	CD E9 34	NEGIER CALL \$34E9	; WENN DIE ZAHL DIE NULL
3471	DB	RET C	; IST, NICHTS ÄNDERN
3472	06 00	LD B,0	; B FÜR NEGIEREN SETZEN
3474	7E	LD A,(HL)	; WENN DAS 1. BYTE NULL
3475	A7	AND A	; IST, HANDELT ES SICH
3476	2B 0B	JR Z,\$3483	; UM EINE INTEGERZAHL
3478	23	INC HL	; ZWEITES BYTE ADRESSIEREN
3479	78	LD A,B	; BIT 7 = 1 FÜR 'ABS'
347A	E6 80	AND \$80	; UND = 0 FÜR NEGIEREN
347C	B6	OR (HL)	; BIT7 FÜR ABS SETZEN
347D	17	RLA	; DIESES BIT WIRD ÜBER
347E	3F	CCF	; DAS CARRY INVERTIERT
347F	1F	RRA	
3480	77	LD (HL),A	; BYTE MIT GEÄNDERTEM VOR-
3481			; ZEICHEN ZURÜCKSCHREIBEN
3481	2B	DEC HL	; AUF ERSTES BYTE ZEIGEN
3482	C9	RET	
3483			
3483			; VORZEICHENÄNDERUNGEN BEI INTEGERZAHLEN
3483			
3483	D5	PUSH DE	; STKEND ZWISCHENSPEICHERN
3484	E5	PUSH HL	; ZEIGER AUF DIE ZAHL RETTEN
3485	CD 7F 2D	CALL \$2D7F	; VORZEICHEN NACH C, ZAHL
3488			; NACH DE LADEN
3488	E1	POP HL	; ZEIGER AUF DIE ZAHL ZURÜCK
3489	78	LD A,B	; \$FF FÜR ABS, \$00 FÜR NEG.
348A	B1	OR C	; VORZEICHEN FÜR ABS IMMER \$FF
348B	2F	CPL	; VORZEICHEN INVERTIEREN
348C	4F	LD C,A	; NACH C KOPIEREN UND DIE
348D	CD 8E 2D	CALL \$2D8E	; INTEGERZAHL ZURÜCKSPEICHERN
3490	D1	POP DE	; STKEND WIEDER LADEN
3491	C9	RET	
3492			
3492			; SUBROUTINE ZUM AUSWERTEN DES VORZEICHENS:
3492			; NEGATIV = -1, POSITIV = +1, NULL = 0 (SIGN-

```

3492      ; FUNKTION MIT OPCODE $29) ALS LETZTEN WERT
3492      ; SPEICHERN
3492      ;
3492  CD E9 34      CALL $34E9      ; WENN DIE ZAHL NULL IST,
3495  D8           RET C          ; DIREKT RETURN
3496  D5           PUSH DE       ; STKEND ZWISCHENSPEICHERN
3497  11 01 00     LD DE,1      ; INTEGERZAHL 1 NACH DE
349A  23           INC HL       ; 2. BYTE ADRESSIEREN
349B  CB 16       RL (HL)      ; VORZEICHEN INS CARRY
349D  2B         DEC HL       ; ZEIGER WIEDER AUF 1. BYTE
349E  9F         SBC A        ; FÜR POSITIVES VORZEICHEN 0
349F  4F         LD C,A       ; NEGATIV $FF NACH C
34A0  CD 8E 2D     CALL $2D8E   ; DIE INTEGERZAHL +/-1 SPEICHERN
34A3  D1         POP DE       ; STKEND ZURÜCKLADEN
34A4  C9         RET
34A5
34A5      ; BEFEHL 'IN' (LADEN EINES PORTS, OPCODE $2C)
34A5      ;
34A5  CD 99 1E     CALL $1E99   ; PORTADRESSE NACH BC LADEN
34AB  ED 78       IN A,(C)     ; INPUTBYTE LADEN UND
34AA  1B 04       JR $34B0     ; AUF DEM STACK ABLEGEN
34AC
34AC      ; BEFEHL 'PEEK' (OPCODE $2B)
34AC      ;
34AC  CD 99 1E     CALL $1E99   ; ADRESSE IN BC BRINGEN
34AF  0A         LD A,(BC)    ; BYTE DIESER ADRESSE LADEN
34B0  C3 2B 2D     JP $2D2B    ; UND A AUF DEM STACK
34B3
34B3      ; BEFEHL 'USR' MIT ZAHLEN (OPCODE $2D)
34B3      ; IM MASCHINENPROGRAMM DÜRFEN ALLE PROZESSORREGISTER
34B3      ; AUSSER HL BENUTZT WERDEN. WIRD HL TROTZDEM
34B3      ; GEBRAUCHT, MUSS VOR DEM ENDE DES MASCHINENPROGRAMMS
34B3      ; HL MIT $275B BELADEN WERDEN
34B3      ;
34B3  CD 99 1E     CALL $1E99   ; DIE STARTADRESSE DES MASCHINEN-
34B6  21 2B 2D     LD HL,$2D2B ; PROGRAMMS NACH BC HOLEN
34B9  E5         PUSH HL      ; RETURNADRESSE FÜR BC AUF
34BA  C5         PUSH BC     ; DEN STACK ZU SPEICHERN
34BB  C9         RET         ; INDIREKTER SPRUNG ZUM
34BC                                     ; MASCHINENPROGRAMM-ANFANG
34BC      ; BEFEHL 'USR A$' (OPCODE $19)
34BC      ;
34BC  CD F1 2B     CALL $2BF1   ; PARAMETER DES STRING HOLEN
34BF  0B         DEC BC      ; DIE LÄNGE MUSS
34C0  7B         LD A,B      ; EINS SEIN
34C1  B1         OR C
34C2  20 23       JR NZ,$34E7  ; LÄNGE <>1: ERROR
34C4  1A         LD A,(DE)    ; STRINGNAMEN LADEN UND
34C5  CD 8D 2C     CALL $2C8D   ; AUF BUCHSTABEN TESTEN
34C8  3B 09       JR C,$34D3   ; SPRUNG BEI BUCHSTABEN
34CA  D6 90       SUB $90     ; FÜR DIE UNTERSUCHUNG AUF
34CC                                     ; BENUTZERDEFINIERTER GRAFIK (UDG)
34CC                                     ; $90 ABZIEHEN, ERGIBT 0 - 20
34CC  3B 19       JR C,$34E7   ; NICHT IM BEREICH: ERROR
34CE  FE 15       CP 21       ; BEI > 20 EBENFALLS
34D0  30 15       JR NC,$34E7  ; ERROR
34D2  3C         INC A        ; FÜR DEC A IST HIER EIN

```

34D3			; INC A NOTWENDIG
34D3	3D	DEC A	; DIE OPERATIONEN DIENEN DAZU
34D4	87	ADD A	; DEN OFFSET ZU BERECHNEN UND
34D5	87	ADD A	; NUR 0-20 DER UDG UND A-U DER
34D6	87	ADD A	; BUCHSTABEN ZUZULASSEN, DIE
34D7	FE AB	CP \$AB	; ASCII-WERTE DER BUCHSTABEN
34D9			; WERDEN AUF DEN BEREICH VON
34D9			; 0 - 20 REDUZIERT
34D9	30 0C	JR NC,\$34E7	; BUCHSTABE NACH U: ERROR
34DB	ED 4B 7B 5C	LD BC,(UDG)	; ADRESSE DER ERSTEN DEFINIERTEN
34DF			; GRAFIK NACH BC LADEN
34DF	81	ADD C	; OFFSET ADDIEREN UND NACH
34E0	4F	LD C,A	; BC ZURÜCKLADEN
34E1	30 01	JR NC,\$34E4	; KEIN CARRY ZU ADDIEREN
34E3	04	INC B	; SONST B +1
34E4	C3 2B 2D	JP \$2D2B	; DIE ADRESSE AUF DEN STACK
34E7			; SPEICHERN
34E7			
34E7	CF	RST ERRAUS	; FEHLERMELDUNG:
34E8	09	.BYT \$09	; 'INVALID ARGUMENT'
34E9			
34E9			; SUBROUTINE ZUM TEST, OB EINE ZAHL NULL IST
34E9			
34E9	E5	PUSH HL	; ZEIGER AUF DIE ZAHL UND
34EA	C5	PUSH BC	; BC ZWISCHENSPEICHERN
34EB	47	LD B,A	; REG A IN B RETTEN
34EC	7E	LD A,(HL)	; DAS ERSTE BYTE DER
34ED	23	INC HL	; ZAHL LADEN UND MIT DEN
34EE	B6	OR (HL)	; DREI NACHFOLGENDEN ODERN
34EF	23	INC HL	; (MÜSSEN ALLE 0 SEIN)
34F0	B6	OR (HL)	; DAS CARRY WIRD HIERBEI
34F1	23	INC HL	; GELÖSCHT
34F2	B6	OR (HL)	
34F3	7B	LD A,B	; REG A,
34F4	C1	POP BC	; BC UND ZEIGER AUF
34F5	E1	POP HL	; DIE ZAHL ZURÜCKLADEN
34F6	C0	RET NZ	; ZAHL IST <0
34F7	37	SCF	; ZAHL IST NULL: CARRY
34F8	C9	RET	; SETZEN
34F9			
34F9			; SUBROUTINE ZUM TEST AUF '>0' (OPCODE \$37)
34F9			; TRIFFT DIES ZU, SO WIRD EIN LETZTER WERT
34F9			; VON 1 ÜBERGEBEN, ANDERNFALLS 0
34F9			
34F9	CD E9 34	CALL \$34E9	; TEST AUF NULL
34FC	DB	RET C	; JA: RETURN
34FD	3E FF	LD A,\$FF	; IN DIE ROUTINE '<0' SPRINGEN,
34FF	1B 06	JR \$3507	; ABER DAS GEGENTEIL MIT '\$FF'
3501			; SIGNALISIEREN
3501			
3501			; FUNKTION 'NOT' (OPCODE \$30)
3501			; LETZTER WERT WIRD 1, WENN ALTER LETZTER WERT
3501			; NULL WAR, SONST WIRD IMMER DER LETZTE WERT
3501			; ZU NULL GESETZT
3501			
3501	CD E9 34	CALL \$34E9	; TEST AUF NULL UND SPRUNG,
3504	1B 05	JR \$350B	; UM 0 ODER 1 ZU SETZEN
3506			

```

3506 ; FUNKTION '0' (OPCODE $36)
3506 ;
3506 AF XOR A ; REG A LÖSCHEN
3507 23 INC HL ; AUF VORZEICHEN ZEIGEN
3508 AE XOR (HL) ; EXOR MIT VORZEICHENBYTE
3509 2B DEC HL ; HL KORRIGIEREN
350A 07 RLCA ; VORZEICHENBIT INS CARRY
350B ;
350B ; SUBROUTINE ZUM SPEICHERN EINER 0 (CARRY
350B ; GELÖSCHT) ODER EINER 1 ALS LETZTEN WERT
350B ;
350B E5 PUSH HL ; ERGEBNISZEIGER RETTEN
350C 3E 00 LD A,0 ; DIE ERSTEN
350E 77 LD (HL),A ; BEIDEN BYTES LÖSCHEN
350F 23 INC HL
3510 77 LD (HL),A
3511 23 INC HL
3512 17 RLA ; CARRY IN A SCHIEBEN, SODASS
3513 77 LD (HL),A ; DRITTES BYTE 1 WIRD, WENN
3514 ; CARRY GESETZT WAR
3514 1F RRA ; A WIEDER NULL
3515 23 INC HL ; DAS VIERTE UND
3516 77 LD (HL),A ; FÜNFTE BYTE
3517 23 INC HL ; ZU NULL SETZEN
3518 77 LD (HL),A
3519 E1 POP HL ; ERGEBNISZEIGER ZURÜCKLADEN
351A C9 RET
351B ;
351B ; FUNKTION 'OR' (OPCODE $07)
351B ; DAS ERGEBNIS DIESER OPERATION (X OR Y) IST X,
351B ; WENN Y NULL IST, ANDERNFALLS 1
351B ;
351B EB EX DE,HL ; HL ZEIGT AUF Y (=2. ZAHL)
351C CD E9 34 CALL $34E9 ; TEST, OB Y NULL IST
351F EB EX DE,HL ; ZEIGER WIEDER ORIGINAL
3520 D8 RET C ; Y IST NULL
3521 37 SCF ; CARRY SETZEN UND LETZTEN
3522 1B E7 JR $350B ; WERT ZU 1 SETZEN
3524 ;
3524 ; FUNKTION 'AND' (OPCODE $08)
3524 ; DIE OPERATION 'X AND Y' LIEFERT X ALS ERGEBNIS,
3524 ; WENN Y (<> 0) IST, ANDERNFALLS DEN WERT NULL
3524 ;
3524 EB EX DE,HL ; HL ZEIGT AUF Y
3525 CD E9 34 CALL $34E9 ; Y AUF NULL TESTEN
3528 EB EX DE,HL ; ZEIGER ZURÜCKTAUSCHEN
3529 D0 RET NC ; Y IST NICHT NULL
352A A7 AND A ; CARRY LÖSCHEN UND EINE
352B 1B DE JR $350B ; NULL ALS ERGEBNIS SPEICHERN
352D ;
352D ; FUNKTION 'STRING AND ZAHL' (OPCODE $10)
352D ; DAS ERGEBNIS DER OPERATION 'A$ AND X' IST
352D ; A$, WENN X (<> 0) IST, IM ANDEREN FALLE WIRD
352D ; EIN NULLSTRING ÜBERGEBEN
352D ;
352D EB EX DE,HL ; HL ZEIGT AUF X
352E CD E9 34 CALL $34E9 ; X AUF NULL TESTEN
3531 EB EX DE,HL ; ZEIGER ZURÜCKTAUSCHEN

```


3532	D0	RET NC	; X IST <>0
3533	D5	PUSH DE	; ZEIGER AUF X RETTEN
3534	1B	DEC DE	; HIGHBYTEADRESSE DER LÄNGE
3535	AF	XOR A	; A LÖSCHEN
3536	12	LD (DE),A	; DIE BEIDEN
3537	1B	DEC DE	; LÄNGENBYTES DES STRING
3538	12	LD (DE),A	; LÖSCHEN
3539	D1	POP DE	; ZEIGER AUF DIE ZAHL
353A	C9	RET	; ZURÜCKHOLEN
353B			
353B		; VERGLEICHSDOPERATIONEN (OPCODE \$09-\$0E, \$11-\$16)	
353B		; OPCODE IST BEIM EINSTIEG IN REG B	
353B			
353B	78	LD A,B	; OPCODES IN DEN BEREICH VON
353C	D6 08	SUB B	; \$01-\$06 UND \$09-\$0E BRINGEN
353E	CB 57	BIT 2,A	; DIESEN BEREICH SO ABÄNDERN,
3540	20 01	JR NZ,\$3543	; DASS \$00-\$02, \$04-\$06,
3542	3D	DEC A	; \$08-\$0A UND \$0C-\$0E BLEIBT
3543	0F	RRCA	; AUF \$00-\$07 REDUZIEREN UND
3544			; CARRY SETZEN BZW. LÖSCHEN:
3544			; DIES DIENT ZUR UNTERSCHIED-
3544			; DUNG DER KOMPLEMENTÄREN
3544			; OPERATIONEN
3544	30 08	JR NC,\$354E	
3546	F5	PUSH AF	; FÜR DIE KOMPLEMENTÄREN
3547	E5	PUSH HL	; OPERATIONEN WERDEN
3548	CD 3C 34	CALL TAUSCH	; DIE BEIDEN ARGUMENTE
3548	D1	POP DE	; AUSGETAUSCHT
354C	EB	EX DE,HL	
354D	F1	POP AF	
354E	CB 57	BIT 2,A	; STRINGOPERATIONEN ?
3550	20 07	JR NZ,\$3559	; JA
3552	0F	RRCA	; CARRY FÜR '=' UND '<>'
3553	F5	PUSH AF	; SETZEN UND ZWISCHENSPEICHERN
3554	CD 0F 30	CALL \$300F	; DIE BEIDEN ZIFFERN ZUM
3557	18 33	JR \$358C	; TESTEN SUBTRAHIEREN
3559			
3559		; STRINGOPERATIONEN	
3559			
3559	0F	RRCA	; CARRY FÜR '=' UND '<>'
355A	F5	PUSH AF	; SETZEN UND RETTEN
355B	CD F1 2B	CALL \$2BF1	; 2. STRINGPARAMETER VOM
355E	D5	PUSH DE	; STACK HOLEN UND
355F	C5	PUSH BC	; ZWISCHENSPEICHERN
3560	CD F1 2B	CALL \$2BF1	; 1. STRINGPARAMETER HOLEN
3563	E1	POP HL	; LÄNGE ZWEITER STRING
3564	7C	LD A,H	; STRINGLÄNGE AUF
3565	B5	OR L	; NULL TESTEN
3566	E3	EX (SP),HL	; ADRESSE UND LÄNGE TAUSCHEN
3567	78	LD A,B	; HIGHBYTE DER LÄNGE NACH A
3568	20 08	JR NZ,\$3575	; WENN LÄNGE 2. STRING <>0
356A	B1	OR C	; 1. STRING LÄNGE 0 ?
356B	C1	POP BC	; LÄNGE 2. STRING NACH BC
356C	28 04	JR Z,\$3572	; SPRUNG: BEIDE STRINGS = 0
356E	F1	POP AF	; 2. STRING IST 0 ODER
356F			; KLEINER ALS DER 1.;
356F	3F	CCF	; CARRY INVERTIEREN
3570	18 16	JR \$3588	

3572		POP AF	
3572	F1	JR \$3588	
3573	18 13		
3575		OR C	; 1. STRING=0 (2. NICHT) ?
3575	B1	JR Z,\$3585	; JA
3576	28 0D	LD A,(DE)	; BEIDE STRINGS (DO: DIE
3578	1A	SUB (HL)	; NACHSTEN BYTES VERGLEICHEN
3579	96	JR C,\$3585	; BYTE VOM 1. IST KLEINER
357A	38 09	JR NZ,\$356B	; BYTE VOM 2. IST KLEINER
357C	20 ED	DEC BC	; BEIDE GLEICH, LANGE -1
357E	0B	INC DE	; UND DIE ZEIGER UM
357F	13	INC HL	; 1 ERHÖHEN
3580	23	EX (SP),HL	; LANGE 2. STRING NACH
3581	E3	DEC HL	; HL UND 1 ABZIEHEN
3582	2B	JR \$3564	; WEITERTVERGLEICHEN
3583	18 DF		
3585		POP BC	; DER 1. STRING IST KLEINER
3585	C1	POP AF	; ALS DER 2.:
3586	F1	AND A	; CARRY LÖSCHEN FÜR TEST
3587	A7		
3588		PUSH AF	; ZWISCHENSPEICHERN
3588	F5	RST CALRUF	; FÜR DIE STRINGTESTS WIRD
3589	EF	.BYT \$A0	; EINE 0 ALS LETZTER WERT IM
358A	A0	.BYT \$3B	; CALCULATOR GESPEICHERT
358B	3B		
358C		POP AF	; CARRY IST GESETZT
358C	F1	PUSH AF	; FÜR '*' ODER '<>'
358D	F5	CALL C,\$3501	; NUR DAMN 'NOT' AUFRUFEN
358E	DC 01 35	POP AF	; IN ALLEN ANDEREN
3591	F1	PUSH AF	; FÄLLEN WIRD
3592	F5	CALL NC,\$34F9	; '>0' AUFGERUFEN
3593	D4 F9 34	POP AF	; FÜR '>','<' UND '*'
3596	F1	RRCA	; WIRD NOCHEINMAL
3597	0F	CALL NC,\$3501	; 'NOT' AUFGERUFEN
3598	D4 01 35	RET	
3598	C9		
359C			
359C		; SUBROUTINE ZUM ADDIEREN VON STRINGS (OPCODE \$17)	
359C			
359C	CD F1 2B	CALL \$2BF1	; PARAMETER DES 2. STRING
359F	D5	PUSH DE	; HOLEN UND
35A0	C5	PUSH BC	; ZWISCHENSPEICHERN
35A1	CD F1 2B	CALL \$2BF1	; PARAMETER 1. STRING HOLEN
35A4	E1	POP HL	; LANGE 2. STRING NACH HL
35A5	E5	PUSH HL	
35A6	D5	PUSH DE	; PARAMETER DES 1.
35A7	C5	PUSH BC	; STRING RETTEN
35A8	09	ADD HL,BC	; BEIDE LÄNGEN ADDIEREN
35A9	44	LD B,H	; UND NACH BC KOPIEREN
35AA	4D	LD C,L	
35AB	F7	RST \$30	; DAMIT DEN BENÖTIGTEN SPEI-
35AC			; PLATZ BESORGEN
35AC	CD B2 2A	CALL \$2AB2	; PARAMETER DES NEUEN
35AF			; STRING AUF CALC.-STACK
35AF	C1	POP BC	; PARAMETER 1. STRING
35B0	E1	POP HL	; ZUM KOPIEREN HOLEN
35B1	7B	LD A,B	; TEST, OB DESSEN
35B2	B1	OR C	; LANGE 0 IST

35B3	2B 02	JR Z,\$35B7	; JA, NICHTS KOPIEREN
35B5	ED B0	LDIR	; SONST STRING UMSPEICHERN
35B7	C1	POP BC	; PARAMETER 2. STRING
35B8	E1	POP HL	; ZUM KOPIEREN LADEN
35B9	78	LD A,B	; WIEDER AUF 0
35BA	B1	OR C	; TESTEN
35BB	2B 02	JR Z,\$35BF	; JA, NICHTS ANHANGEN
35BD	ED B0	LDIR	; 2. STRING AN 1. ANFÜGEN
35BF			
35BF			; SUBROUTINE ZUM SETZEN VON DE AUF STKEND UND
35BF			; HL AUF STKEND-5, D. H. 1. BYTE DES LETZTEN
35BF			; WERTES IM CALCULATORSTACK
35BF			
35BF	2A 65 5C	LD HL,(STKEND)	; STKEND LADEN
35C2	11 FB FF	LD DE,\$FFFB	; FÜR ADDITION -5 LADEN
35C5	E5	PUSH HL	; STKEND ZWISCHENSPEICHERN
35C6	19	ADD HL,DE	; STKEND-5 BESTIMMEN
35C7	D1	POP DE	; STKEND NACH DE BRINGEN
35C8	C9	RET	
35C9			
35C9			; FUNKTION 'CHR\$' (OPCODE \$2F)
35C9			
35C9	CD D5 2D	CALL \$2DD5	; LETZTER WERT NACH REG A
35CC	38 0E	JR C,\$35DC	; FEHLERMELDUNG, WENN ZAHL
35CE	20 0C	JR NZ,\$35DC	; >255 ODER NEGATIV
35D0	F5	PUSH AF	; WERT ZWISCHENSPEICHERN
35D1	01 01 00	LD BC,1	; 1 SPEICHERPLATZ IM
35D4	F7	RST \$30	; WORKSPACE BESORGEN
35D5	F1	POP AF	; WERT ZURÜCKHOLEN UND
35D6	12	LD (DE),A	; ABSPEICHERN
35D7	CD B2 2A	CALL \$2AB2	; PARAMETER DES NEUEN
35DA	EB	EX DE,HL	; STRING AUF CALC.-STACK
35DB	C9	RET	; UND DIE ZEIGER ZURÜCKHOLEN
35DC			
35DC	CF	RST ERRAUS	; FEHLERMELDUNG:
35DD	0A	.BYT \$0A	; 'INTEGER OUT OF RANGE'
35DE			
35DE			; FUNKTION 'VAL' (OPCODE \$1D) UND VAL\$ (\$1B)
35DE			
35DE	2A 5D 5C	LD HL,(CHADD)	; CHADD AUF DEM STACK
35E1	E5	PUSH HL	; ZWISCHENSPEICHERN
35E2	78	LD A,B	; OPCODE NACH A
35E3	C6 E3	ADD \$E3	; DIESE OPERATIONEN ERGEBEN
35E5	9F	SBC A	; \$FF FÜR VAL UND 0 FÜR
35E6	F5	PUSH AF	; VAL\$, DIESES FLAG RETTEN
35E7	CD F1 2B	CALL \$2BF1	; STRINGPARAMETER HOLEN
35EA	D5	PUSH DE	; ANFANGSADRESSE RETTEN
35EB	03	INC BC	; LÄNGE +1
35EC	F7	RST \$30	; SPEICHERPLATZ BESORGEN
35ED	E1	POP HL	; ANFANGSADRESSE HOLEN
35EE	ED 53 5D 5C	LD (CHADD),DE	; ZEIGER AUF 1. NEUEN PLATZ
35F2	D5	PUSH DE	; NACH CHADD UND RETTEN
35F3	ED B0	LDIR	; STRING UMKOPIEREN
35F5	EB	EX DE,HL	; IN DIE LETZTE POSITION
35F6	2B	DEC HL	; DES NEUEN STRING EIN
35F7	36 0D	LD (HL),\$D	; CARRIAGE RETURN SCHREIBEN
35F9	FD CB 01 BE	RES 7,(IY+1)	; SYNTAXPRÜFUNG SETZEN
35FD	CD FB 24	CALL \$24FB	; STRING AUF KORREKTE

3600		RST GETAKT	; SYNTAX PRÜFEN
3600	DF	CP \$D	; ZEICHEN NACH STRING
3601	FE 0D	JR NZ,\$360C	; LADEN UND VERGLEICHEN
3603	20 07	POP HL	; KEIN 'CR': ERROR
3605	E1	POP AF	; STARTADRESSE DES STRINGS
3606	F1	XOR (IY+1)	; 'FLAG' NACH A ZURÜCK
3607	FD AE 01		; MIT FLAGS BITS DURCH
360A			; SYNTAXPRÜFUNG GESETZT
360A	E6 40	AND \$40	; VERGLEICHEN
360C	C2 8A 1C	JP NZ,\$1C8A	; FALLS UNGLEICH. ERROR
360F	22 5D 5C	LD (CHADD),HL	; STARTADRESSE NACH CHADD
3612	FD CB 01 FE	SET 7,(IY+1)	; 'LAUFZEIT' SETZEN
3616	CD FB 24	CALL \$24FB	; STRING WIE NÄCHSTEN AUS-
3619			; DRUCK BEHANDeln
3619	E1	POP HL	; CHADD WIEDER
361A	22 5D 5C	LD (CHADD),HL	; RESTAURIEREN
361D	18 A0	JR \$35BF	; UND STKEND ETC. LADEN
361F			
361F		; FUNKTION 'STR \$' (OPCODE \$2E)	
361F			
361F	01 01 00	LD BC,1	; 1 SPEICHERPLATZ IM
3622	F7	RST \$30	; WORKSPACE BESORGEN
3623	22 5B 5C	LD (KCUR),HL	; ADRESSE NACH KCUR
3626	E5	PUSH HL	; AUF DEM STACK RETTEN
3627	2A 51 5C	LD HL,(CURCHL)	; ADRESSE DES AKTUELLEN
362A	E5	PUSH HL	; KANALS RETTEN
362B	3E FF	LD A,\$FF	; KANAL 'R' FÜR DEN
362D	CD 01 16	CALL \$1601	; WORKSPACE ÖFFNEN
3630	CD E3 2D	CALL \$2DE3	; DEN LETZTEN WERT AUS DEM
3633			; CALC.-STACK IN DEN WORK-
3633			; SPACE AUSGEBEN
3633	E1	POP HL	; AKTUELLE KANALADRESSE UND
3634	CD 15 16	CALL \$1615	; DIE FLAGS DAZU WIEDER
3637			; HERSTELLEN
3637	D1	POP DE	; STARTADRESSE DES STRINGS
3638	2A 5B 5C	LD HL,(KCUR)	; CURSORADRESSE - STARTADRESSE
363B	A7	AND A	; ERGIBT
363C	ED 52	SBC HL,DE	; DIE LÄNGE DES STRINGS
363E	44	LD B,H	; DIESE NACH BC KOPIEREN
363F	4D	LD C,L	
3640	CD B2 2A	CALL \$2AB2	; DIE STRINGPARAMETER AUF
3643			; DEM CALC.-STACK SPEICHERN
3643	EB	EX DE,HL	; ZEIGER KORRIGIEREN
3644	C9	RET	
3645			
3645		; SUBROUTINE FÜR 'READ-IN' (OPCODE \$1A)	
3645			
3645	CD 94 1E	CALL \$1E94	; NUMERISCHEN PARAMETER
3648			; NACH REG A HOLEN
3648	FE 10	CP 16	; DIESER MUSS KLEINER 16
364A	D2 9F 1E	JP NC,\$1E9F	; SEIN, SONST ERROR
364D	2A 51 5C	LD HL,(CURCHL)	; AKTUELLE KANALADRESSE
3650	E5	PUSH HL	; AUF DEM STACK RETTEN
3651	CD 01 16	CALL \$1601	; DEN SPEZIFIZIERTEN KANAL
3654			; ÖFFNEN
3654	CD E6 15	CALL \$15E6	; ZEICHEN HOLEN (WIE NORMA-
3657			; LERWEISE EINE TASTE)
3657	01 00 00	LD BC,0	; DEFAULTLÄNGE = 0

```

365A 30 03      JR NC,$365F      ; SPRUNG: KEIN ZEICHEN
365C 0C          INC C           ; LANGE +1
365D F7          RST $30         ; 1 SPEICHERPLATZ BESORGEN
365E 12          LD (DE),A       ; ZEICHEN DORT SPEICHERN
365F CD B2 2A    CALL $2AB2      ; STRINGPARAMETER AUF
3662             ; STACK SPEICHERN
3662 E1          POP HL          ; AKTUELLE KANALADRESSE UND
3663 CD 15 16    CALL $1615      ; DIE PARAMETER ZURÜCKHOLEN
3666 C3 BF 35    JP $35BF        ; ZEIGER HL UND DE SETZEN
3669             ;
3669             ; FUNKTION 'CODE' (OPCODE $1C)
3669             ; AUSGABE DES ASCIIWERTES EINES ZEICHENS
3669             ;
3669 CD F1 2B    CALL $2BF1       ; STRINGPARAMETER HOLEN
366C 7B          LD A,B          ; STRINGLANGE AUF
366D B1          OR C            ; NULL PRÜFEN
366E 2B 01      JR Z,$3671       ; SPRUNG, WENN NULL
3670 1A          LD A,(DE)       ; ZEICHEN LADEN
3671 C3 2B 2D    JP $2D2B        ; A ALS LETZTEN WERT
3674             ; IM STACK SPEICHERN
3674             ;
3674             ; FUNKTION 'LEN' (OPCODE $1E)
3674             ;
3674 CD F1 2B    CALL $2BF1       ; STRINGPARAMETER HOLEN
3677 C3 2B 2D    JP $2D2B        ; BC AUF DEM CALC.-
367A             ; STACK SPEICHERN
367A             ;
367A             ; SUBROUTINE ZUM VERMINDERN VON 'BREG' UM 1,
367A             ; WELCHES ALS SCHLEIFENZÄHLER DIENST (NACH-
367A             ; BILDUNG DES PROZESSORBEFEHLS DJNZ, HIER
367A             ; DER OPCODE $35)
367A             ;
367A D9          EXX              ; 2. REGISTERSATZ EINSCHALTEN
367B E5          PUSH HL         ; ZEIGER AUF NÄCHSTEN OPCODE
367C             ; ZWISCHENSPEICHERN
367C 21 67 5C    LD HL,BREG      ; ADRESSE VON BREG LADEN
367F 35          DEC (HL)        ; 1 SUBTRAHIEREN
3680 E1          POP HL         ; ZEIGER RESTAURIEREN
3681 20 04      JR NZ,$3687      ; NOCH NICHT FERTIG
3683 23          INC HL          ; NÄCHSTEN OPCODE ÜBER-
3684 D9          EXX              ; SPRINGEN, NORMALE REGISTER
3685 C9          RET             ; WIEDER EINSCHALTEN
3686             ;
3686             ; SUBROUTINE 'JUMP' (OPCODE $33) ZUM ÜBER-
3686             ; SPRINGEN VON CALCULATOROPCODES
3686             ;
3686 D9          EXX              ; 2. REGISTERSATZ EINSCHALTEN
3687 5E          LD E,(HL)        ; SPRUNGOFFSET LADEN
3688 7B          LD A,E          ; FÜR POSITIVEN OFFSET 0
3689 17          RLA              ; UND FÜR NEGATIVEN $FF IN
368A 9F          SBC A           ; A BILDEN UND NACH
368B 57          LD D,A          ; D KOPIEREN, ADDITION ZU
368C 19          ADD HL,DE       ; HL ERGIBT DIE ADRESSE
368D D9          EXX              ; NORMALE REGISTER EINSCHALTEN
368E C9          RET
368F             ;
368F             ; SUBROUTINE ZUM ÜBERSPRINGEN VON CALCULATOR-
368F             ; BEFEHLEN, WENN DER LETZTE WERT <>0 IST

```



```

368F      ; (OPCODE $00)
368F 13    INC DE      ; DAS DRITTE BYTE
3690 13    INC DE      ; ADRESSIEREN UND
3691 1A    LD A, (DE)   ; NACH A LADEN
3692 1B    DEC DE      ; ZEIGER AUF ERSTES BYTE
3693 1B    DEC DE      ; ZURÜCKSETZEN
3694 A7    AND A        ; WERT = 0
3695 20 EF  JR NZ, $3686 ; NEIN: ÜBERSPRINGEN
3697 D9    EXX          ; SONST NUR DEN OFFSET
3698 23    INC HL       ; DURCH INKREMENTIEREN
3699 D9    EXX          ; VON HL ÜBERGEHEN
369A C9    RET

;
; SUBROUTINE ZUM BEENDEN DER CALCULATOR-
; OPERATIONEN (OPCODE $38)
;
369B      POP AF        ; RETURNADRESSE $3365
369B F1    ; WEGWERFEN
369C      EXX          ; HL (NORMAL NÄCHSTER
369C D9    EX (SP), HL  ; CALC.-OPCODE) FÜR
369D E3    EXX          ; RETURN AUF DEM STACK
369E D9    RET          ; SPEICHERN
369F C9

;
; SUBROUTINE ZUM BERECHNEN VON N MOD M
; (OPCODE $32). DER GANZZAHLIGE ANTEIL
; INT (N/M) ERGIBT DEN LETZTEN WERT, DER
; REST DER DIVISION DEN VORLETZTEN WERT
; IM CALCULATORSTACK
;
36A0      RST CALRUF    ; N,M, AUFRUF CALCULATOR
36A0      .BYT $C0      ; N,M, M IN MEMO SPEICHERN
36A1 C0    ; N, M LÖSCHEN
36A2 02    .BYT $02    ; N,N, VERDOPPELN
36A3 31    .BYT $31    ; N,N,M, MEMO LADEN
36A4 E0    .BYT $E0    ; N,N/M, DIVIDIEREN
36A5 05    .BYT $05    ; N, INT(N/M), INTEGER BILDEN
36A6 27    .BYT $27    ; N, INT(N/M), M MEMO LADEN
36A7 E0    .BYT $E0    ; N,M, INT(N/M), VERTAUSCHEN
36A8 01    .BYT $01    ; N,M, INT(N/M) INT(N/M) IN MEMO
36A9 C0    .BYT $C0    ; N,M*INT(N/M), MULTIPLIZIEREN
36AA 04    .BYT $04    ; N-M*INT(N/M), = REST
36AB 03    .BYT $03    ; REST, INT(N/M), MEMO LADEN
36AC E0    .BYT $E0    ; ENDE
36AD 3B    .BYT $3B
36AE C9    RET

;
; FUNKTION 'INTEGER' (OPCODE $27)
;
36AF      RST CALRUF    ; X, CALCULATORAUFRUF
36AF      .BYT $31      ; X,X, VERDOPPELN
36B0 31    .BYT $36      ; X, (1 OD. 0), <0?
36B1 36    .BYT $00      ; SPRUNG, WENN
36B2 00    .BYT $04      ; <NULL NACH INTINV
36B3 04    .BYT $3A      ; INT(X), REST ABSCHNEIDEN
36B4 3A    .BYT $3B      ; ENDE
36B5 3B    RET
36B6 C9
36B7 31    INTINV .BYT $31 ; X,X, VERDOPPELN
36B8 3A    .BYT $3A      ; X, INT(X), REST ABSCHNEIDEN
36B9 C0    .BYT $C0      ; X, INT(X), IN MEMO SPEICHERN

```

36BA	03	.BYT \$03	; X-INT(X), SUBTRAHIEREN
36BB	E0	.BYT \$E0	; X-INT(X), INT(X), MEMO LADEN
36BC	01	.BYT \$01	; INT(X), X-INT(X), TAUSCHEN
36BD	30	.BYT \$30	; INT(X), (1 DD.0), NOT
36BE	00	.BYT \$00	; SPRUNG, WENN REST
36BF	03	.BYT \$03	; NULL IST: NEGINT
36C0	A1	.BYT \$A1	; INT(X), 1, 1 SPEICHERN
36C1	03	.BYT \$03	; INT(X)-1
36C2	3B	NEGINT .BYT \$3B	; ENDE
36C3	C9	RET	
36C4			
36C4		; DIE FUNKTION E^X (OPCODE \$26, EXP)	
36C4			
36C4	EF	RST CALRUF	; X, CALCULATORAUFRUF
36C5	3D	.BYT \$3D	; X, IMMER IN FLOATINGP.
36C6	34	.BYT \$34	; X, 1/LN2, KONSTANTE 1/LN2
36C7	F1	.BYT \$F1, \$3B, \$AA, \$3B, \$29	; SPEICHERN
36C8	38		
36C9	AA		
36CA	38		
36CB	29		
36CC	04	.BYT \$04	; X/LN2=Z, DIVIDIEREN
36CD	31	.BYT \$31	; Z, Z, VERDOPPELN
36CE	27	.BYT \$27	; Z, INT(Z)=N, INTEGER BILDEN
36CF	C3	.BYT \$C3	; Z, N, IN MEM3 SPEICHERN
36D0	03	.BYT \$03	; Z-N=Y, SUBTRAHIEREN
36D1	31	.BYT \$31	; Y, Y, VERDOPPELN
36D2	0F	.BYT \$0F	; 2*Y, ADDIEREN
36D3	A1	.BYT \$A1	; 2*Y, 1, 1 SPEICHERN
36D4	03	.BYT \$03	; 2*Y-1
36D5	88	.BYT \$88	; POLYNOMENTWICKLUNG AUFRUFEN
36D6	13	.BYT \$13, \$36	; ES FOLGEN DIE
36D7	36		
36D8	58	.BYT \$58, \$65, \$66	; DIE 8 NOTWENDIGEN
36D9	65		
36DA	66		
36DB	9D	.BYT \$9D, \$78, \$65, \$40	; KONSTANTEN
36DC	78		
36DD	65		
36DE	40		
36DF	A2	.BYT \$A2, \$60, \$32, \$C9	
36E0	60		
36E1	32		
36E2	C9	.BYT \$E7, \$21, \$F7, \$AF, \$24	
36E3	E7		
36E4	21		
36E5	F7		
36E6	AF		
36E7	24	.BYT \$EB, \$2F, \$B0, \$B0, \$14	
36E8	EB		
36E9	2F		
36EA	B0		
36EB	B0		
36EC	14	.BYT \$EE, \$7E, \$8B, \$94, \$58	
36ED	EE		
36EE	7E		
36EF	8B		
36F0	94		

36F1 58
 36F2 F1
 36F3 3A
 36F4 7E
 36F5 F8
 36F6 CF
 36F7 E3
 36F8 38
 36F9 CD D5 2D
 36FC 20 07
 36FE 38 03
 3700 86
 3701 30 09
 3703
 3703 CF
 3704 05
 3705
 3705 38 07
 3707
 3707 96
 3708 30 04
 370A
 370A
 370A ED 44
 370C 77
 370D C9
 370E
 370E EF
 370F 02
 3710 A0
 3711 38
 3712 C9
 3713
 3713
 3713
 3713
 3713
 3713
 3713
 3713
 3713 EF
 3714 3D
 3715 31
 3716 37
 3717 00
 3718 04
 3719 38
 371A
 371A CF
 371B 09
 371C
 371C A0
 371D 02
 371E 38
 371F 7E
 3720 36 80
 3722 CD 28 2D
 3725 EF
 3726 34

.BYT \$F1,\$3A,\$7E,\$F8,\$CF

.BYT \$E3 ; 2*W,N, MEM3 LADEN
 .BYT \$38 ; ENDE
 CALL \$2DD5 ; N IN REG A LADEN
 JR NZ,\$3705 ; WENN N NEGATIV, SPRUNG
 JR C,\$3703 ; ERROR, WENN N>255
 ADD (HL) ; EXPONENT ADDIEREN
 JR NC,\$370C ; KEIN ÜBERLAUF

RST ERRAUS ; FEHLERMELDUNG:
 .BYT \$05 ; 'NUMBER TOO BIG'

JR C,\$370E ; WENN N<-255, DANN DAS
 ; ERGEBNIS ZU NULL SETZEN
 SUB (HL) ; EXPONENT SUBTRAHIEREN
 JR NC,\$370E ; HIER EXPONENT >0 BEDEU-
 ; TET, DASS ZAHL NULL ZU
 ; SETZEN IST (UNTERLAUF)
 NEG ; EXPONENT KORRIGIEREN
 LD (HL),A ; DEN RICHTIGEN EXPONENT
 RET ; EINSCHREIBEN

RST CALRUF ; BEI 'UNTERLAUF', DAS
 .BYT \$02 ; ERGEBNIS ZU
 .BYT \$A0 ; NULL SETZEN = LETZTER
 .BYT \$38 ; WERT
 RET

; DIE FUNKTION 'LN(X)' (OPCODE \$25)
 ; X WIRD BEI LN(X) AUF 2 ARTEN BERECHNET. DIE
 ; UNTERSCHIEDUNG WIRD DARAN GETROFFEN, OB
 ; X<.8 ODER >.8 IST, NACHDEM X IN DEN BEREICH
 ; VON .5<=X<1 GEBRACHT WURDE

RST CALRUF ; X, CALCULATORAUFRUF
 .BYT \$3D ; X, X IN FLOATINGP.-FORM
 .BYT \$31 ; X,X, VERDOPPELN
 .BYT \$37 ; X,(1 OD.0), X NEGATIV?
 .BYT \$00 ; SPRUNG, WENN X
 .BYT \$04 ; POSITIV: LNPOS
 .BYT \$38 ; SONST ENDE UND:

RST ERRAUS ; FEHLERMELDUNG:
 .BYT \$09 ; 'INVALID ARGUMENT'

; LNPOS
 .BYT \$A0 ; X,0, 0 SPEICHERN
 .BYT \$02 ; X, 0 WIEDER LÖSCHEN
 .BYT \$38 ; ENDE
 LD A,(HL) ; EXPONENT LADEN
 LD (HL),\$80 ; X ZU X' MACHEN
 CALL \$2D28 ; X',E, EXPONENT SPEICHERN
 RST CALRUF ; X',E
 .BYT \$34 ; KONSTANTE 128 SPEICHERN:

3727	38	.BYT \$3B,\$00	; X',E,128
3728	00		
3729	03	.BYT \$03	; X',E', E'=E-128
372A	01	.BYT \$01	; E',X'
372B	31	.BYT \$31	; E',X',X'
372C	34	.BYT \$34	; E',X',X',.8, KONSTANTE .8
372D	F0	.BYT \$F0,\$4C,\$CC,\$CC,\$CD	; SPEICHERN
372E	4C		
372F	CC		
3730	CC		
3731	CD		
3732	03	.BYT \$03	; E',X',X'-.8
3733	37	.BYT \$37	; TEST, OB GRÖßER 0
3734	00	.BYT \$00	; E',X', SPRUNG, WENN X'
3735	08	.BYT \$08	; GRÖßER .8 IST: GROSS8
3736	01	.BYT \$01	; X',E', AUSTAUSCHEN
3737	A1	.BYT \$A1	; X',E',1, 1 SPEICHERN
3738	03	.BYT \$03	; X',E'-1
3739	01	.BYT \$01	; E'-1,X'
373A	38	.BYT \$38	; ENDE
373B	34	INC (HL)	; E'-1,2*X'
373C	EF	RST CALRUF	; E'-1,2*X'
373D	01	GROSS8 .BYT \$01	; X'',E''
373E			; IM FOLGENDEN BEDEUTET X'';
373E			; 2*X'', WENN X'<.8 WAR, SONST
373E			; IMMER X' UND E'';
373E			; E'-1, WENN X'<.8 WAR ODER
373E			; E', WENN X'>.8 WAR
373E	34	.BYT \$34	; X',E',LN2, KONSTANTE
373F	F0	.BYT \$F0,\$31,\$72,\$17,\$FB	; LN2 SPEICHERN
3740	31		
3741	72		
3742	17		
3743	F8		
3744	04	.BYT \$04	; X'',E''*LN2
3745	01	.BYT \$01	; E''*LN2=Y,X''
3746	A2	.BYT \$A2	; Y,X'',.5, .5 SPEICHERN
3747	03	.BYT \$03	; Y,X''-.5
3748	A2	.BYT \$A2	; Y,X''-.5,.5
3749	03	.BYT \$03	; Y,X''-1
374A	31	.BYT \$31	; Y,X''-1,X''-1
374B	34	.BYT \$34	; Y,X''-1,X''-1,2.5, KONSTANTE
374C	32	.BYT \$32,\$20	; 2.5 SPEICHERN
374D	20		
374E	04	.BYT \$04	; Y,X''-1,2.5X''-2.5
374F	A2	.BYT \$A2	; Y,X''-1,2.5X''-2.5,.5
3750	03	.BYT \$03	; Y,X''-1,2.5X''-3
3751	BC	.BYT \$BC	; POLYNOMENTWICKLUNG MIT
3752	11	.BYT \$11,\$AC	; INSGESAMT 12 KONSTANTEN
3753	AC		
3754	14	.BYT \$14,\$09	
3755	09		
3756	56	.BYT \$56,\$DA,\$A5	
3757	DA		
3758	A5		
3759	59	.BYT \$59,\$30,\$C5	
375A	30		
375B	C5		

```

375C 5C          .BYT $5C,$90,$AA
375D 90
375E AA
375F 9E          .BYT $9E,$70,$6F,$61
3760 70
3761 6F
3762 61
3763 A1          .BYT $A1,$CB,$DA,$96
3764 CB
3765 DA
3766 96
3767 A4          .BYT $A4,$31,$9F,$B4
3768 31
3769 9F
376A B4          .BYT $E7,$A0,$FE,$5C,$FC
376B E7
376C A0
376D FE
376E 5C
376F FC
3770 EA          .BYT $EA,$1B,$43,$CA,$36
3771 1B
3772 43
3773 CA
3774 36
3775 ED          .BYT $ED,$A7,$9C,$7E,$5E
3776 A7
3777 9C
3778 7E
3779 5E          .BYT $F0,$6E,$23,$80,$93
377A F0
377B 6E
377C 23
377D 80
377E 93
377F 04          .BYT $04          ; Y=LN(2^E'',LN X''
3780 0F          .BYT $0F          ; LN((2^E'')*X'') = LN X
3781 38          .BYT $38          ; ENDE
3782 C9          RET
3783
3783 ;
3783 ; SUBROUTINE ZUM REDUZIEREN DES ARGUMENTS
3783 ; BEI SINUS UND COSINUS IN DEN BEREICH
3783 ; VON -.5<=V<.5
3783 ;
3783 EF          RST CALRUF          ; X
3784 3D          .BYT $3D          ; X, X IN FLOATINGP.-FORM
3785 34          .BYT $34          ; X, 1/(2*PI), KONSTANTE
3786 EE          .BYT $EE,$22,$F9,$83,$6E ; SPEICHERN
3787 22
3788 F9
3789 83
378A 6E
378B 04          .BYT $04          ; X/(2*PI)
378C 31          .BYT $31          ; X/(2*PI),X/(2*PI)
378D A2          .BYT $A2          ; X/(2*PI),X/(2*PI),.5
378E 0F          .BYT $0F          ; X/(2*PI),X/(2*PI)+.5
378F 27          .BYT $27          ; X/(2*PI),INT(X/(2*PI)+.5)
3790 03          .BYT $03          ; Y=X/(2*PI)-INT(X/(2*PI)+.5)

```



```

3791 31      .BYT $31      ; Y,Y
3792 0F      .BYT $0F      ; 2*Y
3793 31      .BYT $31      ; 2*Y,2*Y
3794 0F      .BYT $0F      ; 4*Y
3795 31      .BYT $31      ; 4*Y,4*Y
3796 2A      .BYT $2A      ; 4*Y,ABS(4*Y)
3797 A1      .BYT $A1      ; 4*Y,ABS(4*Y),1
3798 03      .BYT $03      ; 4*Y,ABS(4*Y)-1=Z
3799 31      .BYT $31      ; 4*Y,Z,Z
379A 37      .BYT $37      ; 4*Y,Z,(1 OD.0) >0?
379B C0      .BYT $C0      ; ERGEBNIS DAVON NACH MEMO
379C 00      .BYT $00      ; SPRUNG, WENN >0 NACH
379D 04      .BYT $04      ; ZPOS1
379E 02      .BYT $02      ; 4*Y
379F 38      .BYT $38      ; ENDE
37A0 C9      RET

;
ZPOS1 .BYT $A1      ; 4*Y,Z,1
      .BYT $03      ; 4*Y,Z-1
      .BYT $01      ; Z-1,4*Y
      .BYT $36      ; Z-1,(1 OD.0) <0?
      .BYT $00      ; SPRUNG, WENN <0 NACH
      .BYT $02      ; YNEGAT
      .BYT $1B      ; 1-Z, INVERTIEREN
YNEGAT .BYT $38      ; ENDE
      RET

;
; FUNKTION COSINUS (OPCODE $20)
;
RST CALRUF      ; X
      .BYT $39      ; V, ARGUMENT HOLEN
      .BYT $2A      ; ABS(V)
      .BYT $A1      ; ABS(V),1
      .BYT $03      ; ABS(V)-1
      .BYT $E0      ; MEMO HOLEN
      .BYT $00      ; SPRUNG, WENN MEMO
      .BYT $06      ; 1 IST: SINCOS
      .BYT $1B      ; 1-ABS(V), INVERTIEREN
      .BYT $33      ; 1-ABS(V)=W, SPRUNG NACH
      .BYT $03      ; SINCOS

;
; SUBROUTINE ZUM BERECHNEN DES SINUS
; (OPCODE $1F)
;
RST CALRUF      ; X
      .BYT $39      ; W, ARGUMENT HOLEN
SINCOS .BYT $31      ; W,W
      .BYT $31      ; W,W,W
      .BYT $04      ; W,W*W
      .BYT $31      ; W,W*W,W*W
      .BYT $0F      ; W,2*W*W
      .BYT $A1      ; W,2*W*W,1
      .BYT $03      ; W,2*W*W-1=Z

;
      .BYT $86      ; POLYNOMENTWICKLUNG MIT
      .BYT $14,$E6   ; 6 KONSTANTEN
      .BYT $5C,$1F,$0B

```

```

37C2 1F
37C3 0B
37C4 A3
37C5 8F
37C6 38
37C7 EE
37C8 E9
37C9 15
37CA 63
37CB BB
37CC 23
37CD EE
37CE 92
37CF 0D
37D0 CD
37D1 ED
37D2 F1
37D3 23
37D4 5D
37D5 1B
37D6 EA
37D7
37D7 04
37D8 38
37D9 C9
37DA
37DA
37DA
37DA
37DA EF
37DB 31
37DC 1F
37DD 01
37DE 20
37DF 05
37E0 38
37E1 C9
37E2
37E2
37E2
37E2
37E2
37E2
37E2
37E2
37E2 CD 97 32
37E5 7E
37E6 FE 81
37E8 38 0E
37EA EF
37EB A1
37EC 1B
37ED 01
37EE 05
37EF 31
37F0 36
37F1 A3

.BYT $A3,$8F,$38,$EE

.BYT $E9,$15,$63,$8B,$23

.BYT $EE,$92,$0D,$CD,$ED

.BYT $F1,$23,$5D,$1B,$EA

; W, (SIN(PI*W/2))/W
; SIN(PI*W/2) = SIN(X) ODER
; COS(X)
.BYT $04
.BYT $38
RET

;
; SUBROUTINE ZUM BERECHNEN DES TANGENS
; (OPCODE $21)
;
RST CALRUF
; X
.BYT $31
; X,X
.BYT $1F
; X,SIN(X)
.BYT $01
; SIN(X),X
.BYT $20
; SIN(X),COS(X)
.BYT $05
; TAN(X)=SIN(X)/COS(X)
.BYT $38
; ENDE
RET

;
; SUBROUTINE ZUM BERECHNEN VON ARCUSTANGENS
; (OPCODE $24)
;
; ES ERFOLGT EINE UNTERTEILUNG IN DREI FÄLLE:
; FÜR -1<X<1: W=0, Y=X
; FÜR 1<=X: W=PI/2, Y=-1/X
; FÜR X<=-1: W=-PI/2, Y=-1/X
;
CALL $3297
LD A,(HL)
CP $B1
JR C,TANGE1
RST CALRUF
.BYT $A1
.BYT $1B
.BYT $01
.BYT $05
.BYT $31
.BYT $36
.BYT $A3
; X IN FLOATINGP.-FORM
; EXPONENT LADEN
; 1 < ABS(X) ?
; JA
; X, CALCULATORAUFRUF
; X,1, 1 SPEICHERN
; X,-1, INVERTIEREN
; -1,X, TAUSCHEN
; -1/X, DIVIDIEREN
; -1/X,-1/X
; -1/X,(1 00.0), <0?
; -1/X,(1 00.0),PI/2

```

37F2	01	.BYT \$01	; -1/X,PI/2,(1 DD.0)
37F3	00	.BYT \$00	; SPRUNG FÜR 1<=X NACH
37F4	06	.BYT \$06	; TANGE2
37F5	1B	.BYT \$1B	; -1/X,-PI/2, NEGIEREN
37F6	33	.BYT \$33	; SPRUNG NACH
37F7	03	.BYT \$03	; TANGE2
37F8	EF	TANGE1 RST CALRUF	; Y
37F9	A0	.BYT \$A0	; Y,0
37FA	01	TANGE2 .BYT \$01	; W,Y, TAUSCHEN
37FB	31	.BYT \$31	; W,Y,Y
37FC	31	.BYT \$31	; W,Y,Y,Y
37FD	04	.BYT \$04	; W,Y,Y*Y
37FE	31	.BYT \$31	; W,Y,Y*Y,Y*Y
37FF	0F	.BYT \$0F	; W,Y,2*Y*Y
3800	A1	.BYT \$A1	; W,Y,2*Y*Y,1
3801	03	.BYT \$03	; W,Y,2*Y*Y-1=Z
3802	8C	.BYT \$8C	; POLYNOMENTWICKLUNG MIT
3803	10	.BYT \$10,\$B2	; 12 KONSTANTEN
3804	B2		
3805	13	.BYT \$13,\$0E	
3806	0E		
3807	55	.BYT \$55,\$E4,\$8D	
3808	E4		
3809	8D		
380A	5B	.BYT \$5B,\$39,\$BC	
380B	39		
380C	BC		
380D	5B	.BYT \$5B,\$9B,\$FD	
380E	9B		
380F	FD		
3810	9E	.BYT \$9E,\$00,\$36,\$75	
3811	00		
3812	36		
3813	75	.BYT \$A0,\$DB,\$EB,\$B4	
3814	A0		
3815	DB		
3816	EB		
3817	B4	.BYT \$63,\$42,\$C4	
3818	63		
3819	42		
381A	C4	.BYT \$E6,\$B5,\$09,\$36,\$BE	
381B	E6		
381C	B5		
381D	09		
381E	36		
381F	BE	.BYT \$E9,\$36,\$73,\$1B,\$5D	
3820	E9		
3821	36		
3822	73		
3823	1B		
3824	5D	.BYT \$EC,\$DB,\$DE,\$63,\$BE	
3825	EC		
3826	DB		
3827	DE		
3828	63		
3829	BE	.BYT \$F0,\$61,\$A1,\$B3,\$0C	
382A	F0		
382B	61		

```

382C A1
382D B3
382E 0C
382F 04
3830 0F
3831 38
3832 C9
3833
3833 ;
3833 ; FUNKTION ARCUSSINUS (OPCODE $22)
3833 ;
3833 RST CALRUF ; X, CALCULATORAUFRUF
3834 .BYT $31 ; X,X
3835 .BYT $31 ; X,X,X
3836 .BYT $04 ; X,X*X
3837 .BYT $A1 ; X,X*X,1
3838 .BYT $03 ; X,X*X-1
3839 .BYT $1B ; X,1-X*X
383A .BYT $2B ; X,SQR(1-X*X)
383B .BYT $A1 ; X,SQR(1-X*X),1
383C .BYT $0F ; X,1+SQR(1-X*X)
383D .BYT $05 ; X/1+SQR(1-X*X)=TAN(Y/2)
383E .BYT $24 ; Y/2, ARCUSTANGENS
383F .BYT $31 ; Y/2,Y/2
3840 .BYT $0F ; Y = ARCUSSINUS X
3841 .BYT $3B ; ENDE
3842 RET
3843
3843 ;
3843 ; FUNKTION ARCUSCOSINUS (OPCODE $23)
3843 ;
3843 RST CALRUF ; X
3844 .BYT $22 ; ARCSIN X
3845 .BYT $A3 ; ARCSIN X,PI/2
3846 .BYT $03 ; ARCSIN X -PI/2
3847 .BYT $1B ; PI/2 - ARCSIN X = ARCCOS X
3848 .BYT $3B ; ENDE
3849 RET
384A
384A ;
384A ; SUBROUTINE ZUR BERECHNUNG DER QUADRATWURZEL
384A ; (OPCODE $28)
384A ;
384A RST CALRUF ; X
384B .BYT $31 ; X,X
384C .BYT $30 ; X,(1 OD.0), X = 0 ?
384D .BYT $00 ; SPRUNG, WENN X = 0
384E .BYT $1E ; NACH LETZWE
384F .BYT $A2 ; X,.5, .5 AUF DEN STACK,
3850 .BYT $3B ; UM X^.5 ZU BERECHNEN
3851
3851 ;
3851 ; SUBROUTINE ZUR BERECHNUNG VON X^Y
3851 ; (OPCODE $06)
3851 ;
3851 RST CALRUF ; X,Y
3852 .BYT $01 ; Y,X
3853 .BYT $31 ; Y,X,X
3854 .BYT $30 ; Y,X,(1 OD.0), X=0 ?
3855 .BYT $00 ; SPRUNG, WENN X=0
3856 .BYT $07 ; NACH XNULL
3857 .BYT $25 ; Y, LN(X)

```

```

3858 04          .BYT $04          ; Y*LN(X)
3859 38          .BYT $38          ; ENDE UND SPRUNG, UM
385A C3 C4 36    JP $36C4          ; EXP(Y*LN(X)) ZU BILDEN
385D            ;
385D 02          XNULL .BYT $02      ; Y
385E 31          .BYT $31          ; Y,Y
385F 30          .BYT $30          ; Y,(1 00.0), Y = 0 ?
3860 00          .BYT $00          ; SPRUNG NACH EINSSP,
3861 09          .BYT $09          ; WENN Y=0
3862 A0          .BYT $A0          ; Y,0
3863 01          .BYT $01          ; 0,Y
3864 37          .BYT $37          ; 0,(1 00.0), Y>0 ?
3865 00          .BYT $00          ; SPRUNG NACH LETZWE,
3866 06          .BYT $06          ; WENN Y > 0
3867 A1          .BYT $A1          ; SONST DURCH DIE DIVISION
3868 01          .BYT $01          ; I/O FEHLER ERZEUGEN:
3869 05          .BYT $05          ; 'ARITHMETIC OVERFLOW'
386A 02          EINSSP .BYT $02     ; -, LÖSCHEN
386B A1          .BYT $A1          ; 1
386C 38          LETZWE .BYT $38    ;
386D C9          RET
386E            ;
386E            ; BIS ZUM ZEICHENSATZ ENTHÄLT DAS ROM NUR NOCH $FF
386E            ;
386E            .END
386E            **$3D00
3D00            CHARRO
3D00            .END

```

ERRORS = 0000

ENDE DES ASSEMBLERS