Luidger Röckrath

Der geknackte TRS-80

Über den internen Aufbau von Basic-Interpretern legen die Hersteller verbreiteter Tischcomputer meist den Mantel des Schweigens. So erfordert es schon einige Arbeit, diese Nuß zu knacken. Die Ergebnisse dieser Arbeit (über 500 Adressen und einiges mehr) sind im folgenden dargestellt.

Im Franzis-Sonderheft „Mikrocomputer-Anwendungen" erschienen für mehrere 6502-Computer RAM/ROM-Listings [1], d. h. Aufstellungen aller wichtigen Adressen sowohl für Programmspeicher als auch für Datenspeicher. Dies ist erforderlich, weil die meisten Hersteller sich über diese Dinge ausschweigen, obwohl sie zur optimalen Ausnutzung des Computers für den Anwender aus verschiedenen Gründen notwendig sind [2].
Erstens für den Basic-Programmierer, der Programme für andere Computer adaptieren muß und dazu z. B. die Tastaturadressen benötigt, aber auch für Programmierer, die es entgegen allen Hindernissen nicht sein lassen können, ihren Computer in Maschinensprache zu programmieren, sind besonders die Adressen der Ein-/Ausgaberoutinen von Interesse. Und schließlich für jene Unverbesserlichen, die meinen, ihr Basic sei noch nicht gut genug und könnte durch einige zusätzliche Befehle sinnreich erweitert werden.
Das RAM/ROM-Listing für den TRS-80 enthält für alle drei Gruppen etwas: für den Basic-Programmierer alle wichtigen I/O-Adressen (einschließlich aller Tastaturadressen), für die passionierten Maschinenprogrammierer alle wichtigen Unterprogramme, die sie zum Anpassen fremder oder zum Entwickeln eigener Programme benötigen. Um es dieser Gruppe besonders leicht zu machen, enthält die Tabelle eine genaue Beschreibung, was das Unterprogramm eigentlich tut und welche Register verändert werden. Die letzte Gruppe sei auf den vorletzten Abschnitt dieses Artikels verwiesen, in dem ausführlich die zahlreichen Möglichkeiten des Eingriffs in den Basic-Interpreter erläutert sind.

Einige grundsätzliche Bemerkungen

Wer schon mal in der Tabelle gestöbert hat, der wird sich gewundert haben, daß nur die Beeinflussung der 8080-Register angegeben ist, obwohl der TRS-80 einen Z80 als Zentraleinheit beinhaltet, der doch erheblich mehr Register hat. Des Rätsels Lösung ist ebenso einfach wie enttäuschend: Der TRS-80 arbeitet mit einem 8080-Basic. Man hat sich nur die Mühe gemacht, die Sprünge, wenn möglich, in relative Sprünge umzuwandeln. Lediglich die systemspezifische Ein-/Ausgabe ist im Z80-Code programmiert; dort wird auch einmal das X-Indexregister benutzt, aber zuvor gerettet. So stehen der ganze Zweitregistersatz und die Indexregister für Erweiterungen zur Verfügung.
Das Level-2-Basic des TRS-80 ist ein Microsoft-Basic, d. h. der Programmtext wird nicht so abgespeichert wie er eingegeben wurde, sondern zuvor in einen Zwischencode (1 Byte pro Befehl, zwischen hex 80 und FA) umgewandelt. Diese Zwischencodes sind in [3] auf Seite E/1 aufgelistet. Bei der Programmausführung muß jetzt also nicht mehr das ganze Befehlswort erkannt und dann aus einer Tabelle eine Adresse herausgesucht werden, sondern der Zwischencode dient als Zeiger auf eine Tabelle, die die Ansprungadressen der Befehle und Funktionen enthält. So nimmt die Interpretation des Programmes nicht mehr so viel Zeit in Anspruch wie z. B. beim Level-1-Basic. Nach der gleichen Methode arbeiten die meisten handelsüblichen Interpreter (PET, Apple usw.).

Unterschiedliche Versionen

Das RAM/ROM-Listing wurde von einer Version (V.2) des Level-2-Basic erstellt, die Mitte 1980 verkauft wurde. Bei dieser Version sind der READ/DATA-Fehler [4] und das Prellen der Eingabetastatur ausgemerzt. Zum Vergleich wurde eine ältere Version (V.1) herangezogen, welche die beiden oben genannten Fehler noch hat. Die Unterschiede der V.2 gegenüber der V.1 seien im folgenden dargestellt:

1. Der Tastendruck SHIFT ← wird nicht mehr erkannt aufgrund einer Änderung in der Tastaturdecodierungstabelle. (SHIFT ← hat nun die Funktion einer CONTROL-Taste.)
2. Zugunsten der Entprellung wurden die Texte „RADIO SHACK LEVEL 2 BASIC" und „MEMORY SIZE" zu „R/S L2 BASIC" und „MEM SIZE" gekürzt. Dies hat noch eine Änderung der Zeiger zur Folge.
3. Um beim Laden von der Kassette die Fehlerträchtigkeit zu erniedrigen, wurden zwei Schleifen geringfügig verlängert.
4. Einige kleine Änderungen am PRINT-Befehl, die bewirken, daß

a) mehrere @ in einer PRINT-Anweisung möglich sind,
b) nicht mehr zwischen @ und SHIFT @ unterschieden wird,
c) das Argument von TAB nicht mehr modulo 64, sondern modulo 128 vorbehandelt wird.

5. Der READ/DATA-Fehler wurde behoben.
6. Durch eine Änderung bei CLOAD ist es nicht mehr möglich, eine Kassettenrecorder-Nummer anzugeben (CLOAD#-1, „A" führt zu Fehlermeldung).

Wie von Radio Shack bestätigt, existieren auch andere Versionen, bei denen z. B. nur der READ-/Data-Fehler ausgemerzt ist. Bei der Programmentwicklung sollten diese Unterschiede beach-

tet werden, um die Kompatibilität zu wahren.

Eingriff in den Interpreter

Der Level-2-Interpreter ist zwar in zwei ROMs abgespeichert, und somit nicht zu ändern, aber an sehr vielen Stellen verzweigt die Programmausführung ins RAM. Dort kann leicht in den Programmablauf eingegriffen werden.

1. Alle wichtigen Ein-/Ausgaberoutinen werden über die DCBs angesprungen, die die Treiberadressen enthalten. Ändert man diese Adressen, so können leicht eigene Ein-/Ausgaberoutinen mit dem Basic-Interpreter in Verbindung treten [5], [6], [7].

2. Wichtige Unterprogramme des Basic werden über die RSTs 8 bis 20 (hex) angesprungen, deren Vektoren im RAM liegen. In praktisch allen Programmsegmenten des Basic-Interpreters werden diese Unterprogramme benötigt. Eine Analyse des entsprechenden Programmteiles zeigt schnell, wie man zweckmäßig eingreifen kann. Dann wird nur noch der Vektor geändert, nach Ansprung des entsprechenden RST geprüft, ob das richtige Programmsegment aufruft und schon kann die eigene Erweiterung folgen [8].

3. Eine einfache und fast unerschöpfliche Möglichkeit des Eingriffes sind die zahlreichen Disk-Basic-Befehle und Funktionen. Diese werden erkannt; sodann wird ins RAM verzweigt, wo Sprünge zu den eigentlichen Programmsegmenten zur Ausführung dieser Befehle und Funktionen stehen. Diese Sprünge zu ändern und eigene neue Befehle zu implentieren, ist kein Problem. Wie [9] zeigt, muß dies noch nicht einmal dazu führen, daß die Zusammenarbeit mit dem Disk-Basic gestört ist.

4. Zur Erweiterung vorhandener Befehle des Level-2-Basic durch das Disk-Basic (z. B. PRINT#, was im Disk-Basic zum Schreiben auf Diskette dient), rufen diese Befehle Unterprogramme im RAM auf. Normalerweise sind diese Adressen mit Return-Befehlen (C9) belegt, aber sie können leicht durch entsprechende Sprünge ersetzt werden. Auch hier analysiert man zweckmäßigerweise das zu erweiternde Programmsegment und wird bald herausfinden, wie man günstig eingreifen kann. Man muß beachten, daß eine Adresse unter Umständen aus verschiedenen Programmteilen aufgerufen wird.

Nun schnell den TRS-80 angeschaltet und an die Arbeit! Wem das hier alles noch zu abstrakt ist, der möge sich einmal [7], [8] und [9] anschauen, drei Beispiele für die ersten drei Methoden. Sollte es jemandem an Ideen ermangeln, auch dem kann geholfen werden. [10] und [11] dürften für die nächsten Feierabende ausreichen.

Das RAM/ROM-Listing

Das RAM/ROM-Listing enthält über 500 ROM-, RAM- und I/O-Adressen. Bei wichtigen Unterprogrammen ist angegeben, welche Register verändert werden (xx). Ist in dieser Spalte ein konkreter Zahlenwert angegeben, so bedeutet dies, daß nach der Rückkehr von diesem Unterprogramm das entsprechende Register immer diesen Wert beinhaltet. Das Register HL dient bei der Ausführung von Basic-Programmen als Zeiger auf den Programmtext. Hat es diese Funktionen, so ist ein P in der entsprechenden Spalte zu sehen.

Literatur

[1] *Martin, R.; Smode, D.:* ROM und RAM in PET und CBM. Franzis-Sonderheft „Mikrocomputer-Anwendungen".

[2] *Feichtinger, H.:* Meine Meinung... FUNKSCHAU 1979, Heft 24, Seite 1418.

[3] Radio Shack: Level II Basic Reference Manual.

[4] *Neske, W.:* TRS-80 – ein Computer mit kleinem Fehler. Franzis-Sonderheft „Programme für Kleincomputer und Taschenrechner".

[5] *Duddek, G.:* Ein preiswerter Drucker für den TRS-80. Franzis-Sonderheft „Hobbycomputer 2".

[6] *Hofer, R.:* TRS-80-Treiberprogramm für Schreibmaschinen-Drucker. FUNKSCHAU 1980, Heft 7, Seite 91.

[7] *Röckrath, L.:* Repeatfunktion für den TRS-80 (in Vorbereitung).

[8] *Röckrath, L.:* RESTORE N für den TRS-80 (in Vorbereitung).

[9] *Röckrath, L.:* Double-Precision-Funktionen (in Vorbereitung).

[10] *Feichtinger, H.:* Super-Basic für AIM-65 und PC-100. FUNKSCHAU 1980, Heft 25, Seite 103.

[11] *Klein, R. D.:* S-100-System mit Basic und schnellem Kassetten-Interface. Franzis-Sonderheft „Hobbycomputer 1".

ROM- und RAM-Adressen

ROM-Adressen Level-II-Basic-Interpeter
Initialisierung und Ein-/Ausgabe

					AF	BC	DE	HL
0000	0000	00000	00000	RST 0, Basic-Kaltstart				
0008	0008	00008	00008	RST 18, Basic-UP s. 1C96				
0010	0010	00016	00016	RST 10, Basic-UP s. 1D78				
0018	0018	00024	00024	RST 18, Basic-UP s. 1C90				
0020	0020	00032	00032	RST 20, Basic-UP s. 25D9				
0028	0028	00040	00040	RST 28, wird bei Drücken der Break-Taste angespr.				
002B	002B	00043	00043	INCH1-Ansprung (über DCB), Tastaturabfrage: ASCII-Code gedrückter Taste in A, wenn keine Taste gedrückt ist, A=0	xxxx		4015	
0030	0030	00048	00048	RST 30, unbenutzt				
0033	0033	00051	00051	OUTCH-Ansprung (über DCB), Ausgabe des Akkuinhaltes auf den Bildschirm	xx		401D	
0038	0038	00056	00056	RST 38H, unbenutzt				
003B	003B	00059	00059	PRINT-Ansprung (über DCB), Ausgabe des Akkuinhaltes auf den Drucker	xxxx		4025	
0049	004F	00073	00079	INCH2, wie INCH1, es wird aber gewartet, bis eine Taste gedrückt wird	xxxx		4015	

					AF	BC	DE	HL
0050	005F	00080	00095	Tabelle für die Tastaturdecodierung				
0060	0065	00096	00101	DELAY, Zeitschleife 14,6 µs × BC	0044	0000		
0066	0074	00102	00116	Reset-Ansprung (NMI-Vektor). Wenn Floppy angeschlossen, Basic-Kaltstart, sonst Warmstart				
0075	0104	00117	00260	Basic-Initialisierung:				
0075	008D	00117	00141	DCBs, RST-Vektoren und I/O-Buffer einrichten				
008E	00AB	00142	00171	Disk-Basic-Zeiger initialisieren (werden beim Laden des Disk-Basic geändert)				
00AC	00F8	00172	00248	MEM SIZE anfordern oder selbst Speicherende suchen				
00F9	0104	00249	00260	NEW, „R/S L2 BASIC" ausdrucken und Sprung zur Hauptschleife				
0105	010D	00261	00269	Text „MEM SIZE"				
010E	011B	00270	00283	Text „R/S L2 BASIC(CR)"				
011C	012C	00284	00300	Tastaturentprellung				
012D	0131	00301	00305	L3-Error				
0132	0132	00306	00306	POINT A=00 } Ansprung der drei Grafikbefehle				
0135	0135	00309	00309	SET A=80 } und Setzen des Flags (A).				
0138	0138	00312	00312	RESET A=01 }				
013A	017D	00314	00381	Aus den Koordinaten werden die Adresse im Bildschirmram und die Maske errechnet				
017E	019C	00382	00412	In Abhängigkeit von dem Flag wird ein POINT, SET oder RESET ausgeführt				
019D	01C8	00413	00456	INKE$-Funktion				
01BC	01C8	00444	00456	Leerstring nach X				
01C9	01D2	00457	00466	CLS-Befehl, Bildschirm wird gelöscht	1Fxx			
01D3	01D8	00467	00472	RANDOM-Befehl (als zufällige Größe wird das R-Register benutzt)				
01D9	01F7	00473	00503	Impuls auf Kassette ausgeben	xxxx	00		FC00
01F8	01FD	00504	00509	Kassettenrecorder abschalten	xxxx			
01FE	021D	00510	00541	Kassettenrecordernummer decodieren und Kassettenrecorder einschalten	xxxx	xxxx	xxxx	P
0215	021D	00533	00541	Kassettenrecorder einschalten	xxxx			
021E	0220	00542	00544	Bit 7 von Port (FF) zurücksetzen	xxxx			FC00
0221	022B	00545	00555	(403D) ∧ H ∨ L nach (403D) und zum Port FF	xxxx			
022C	0234	00556	00564	Stern in rechter, oberer Ecke des Bildschirms umschalten	xxxx			
0235	0260	00565	00608	Byte von Kassette lesen (wird im Akku übergeben)	xxxx			
0241	0260	00577	00608	Bit (b) von Kassette lesen (2 . A + b ⧫ A)	xxxx			FC00
0261	0283	00609	00643	Byte (im Akku) zweimal auf Kassette aufzeichnen				
0264	0283	00612	00643	Byte (im Akku) auf Kassette aufzeichnen				
0284	0292	00644	00658	Kassette einschalten, 255 mal 0 und A5 aufzeichnen	xxxx	xxxx	xxxx	P
0293	02A8	00659	00680	Kassette einschalten, auf A5 warten und Sternchen auf Bildschirm setzen	2Axx	xxxx	xxxx	P
02A9	0329	00681	00809	SYSTEM-Befehl				
02B2	02B2	00690	00690	Ansprung				
02CE	0313	00718	00787	Objektfile von Kassette laden				
0314	031C	00788	00796	Wort (in HL) von Kassette laden (L, H)	xxxx			xxxx
031D	0329	00797	00809	Ansprung des Objektfile oder irgendeiner anderen Adresse				
032A	0347	00810	00839	Ausgabe des Akkuinhaltes auf Bildschirm (00), Drucker (01) oder Kassette (80) in Abhängigkeit von (409C)	xx			
033A	0347	00826	00839	OUTCH2, wie OUTCH, zusätzlich Cursorposition nach (40A6)	xx			
0348	0357	00840	00855	Position des Cursors in der Bildschirmzeile nach A	xxxx			
0358	0360	00856	00864	INCH3, wie INCH1	xxxx			
0361	0383	00865	00899	INLINE, Eingabe von max. 240 Zeichen in den I/O-Buffer mit allen Cursorfunktionen. Wenn Breack gedrückt wird, Rückkehr mit gesetzten Carry-Flag, bei Drücken von Clear wird der Bildschirm gelöscht und die Eingabe beginnt von vorne. HL enthält die Bufferanfangsadresse –1	xxxx		401D	xxxx
0384	038A	00900	00906	INCH4, wie INCH2	xxxx			
038B	039B	00907	00923	CR auf Drucker ausgeben, wenn Druckkopf nicht in Position 0, (409C) auf 0 setzen	xxxx			
039C	03C1	00924	00961	Zeichen in A auf Drucker ausgeben, Zeichenzähler (409B) incrementieren und bei 0A, 0C und oD auf 0 setzen				
03C2	03E2	00962	00994	Routine zum Aufruf der Ein-/Ausgabe über die DCBs. (DCB-Typ in B, DCB-Adresse in DE)				
03E3	0457	00995	01111	Tastaturabfrage und Decodierung (Ansprung nur über INCH1-4, da nur dann die Register gerettet werden, siehe dort)				
0458	058C	01112	01420	Bildschirmausgabe (Ansprung nur über DCB, da nur dann Register gerettet werden und die Cursoradresse übergeben wird, siehe OUTCH)				
04B8	058C	01208	01420	Bildschirm-Steuerbefehle				

					AF	BC	DE	HL
04B8	04BC	01208	01212	Cursor ON	(0E)			
04BD	04BF	01213	01215	Cursor OFF	(0F)			
04C0	04CD	01216	01229	Cursor HOME (64 cpl)	(1C)			
04CE	04D9	01230	01241	Cursor BACKSPACE	(08)			
04DA	04EB	01242	01259	Cursor ↓	(18)			
04E7	04EB	01255	01259	Cursor ←	(1A)			
04EC	04F5	01260	01269	Cursor ↑	(19)			
04F1	04F5	01265	01269	Cursor →	(1B)			
04F6	0505	01270	01285	32cpl	(17)			
0506	0540	01286	01344	Auswertung der Steuerbefehle				
0541	0563	01345	01379	Zeichen auf Bildschirm setzen und Scroll, wenn nötig				
0564	0572	01380	01394	New-Line	(0A-0D)			
0573	058C	01395	01420	Clear to end of line	(1E)			
057C	058C	01404	01420	Clear to end of frame	(1F)			
058D	05D0	01421	01488	Drucker-Treiber				
05D1	05D8	01489	01496	Drucker bereit? j,Z=1.	xxxx			
05D9	0673	01497	01651	Unterprogramm für INLINE wie INLINE, maximale Anzahl der Zeichen in B, Bufferanfangsadresse in HL Bei Rückkehr Anzahl der eingegebenen Zeichen in B	xxxx	xxxx	401D	
0674	06CB	01652	01739	Wenn Breack gedrückt ist, Sprung zur Basic-Initialisierung, sonst Testen ob Floppy vorhanden Wenn ja, DOS-Laden und starten				
06CC	06CC	01740	01740	Adresse für Rücksprung von SYSTEM-Programmen zum Basic (Basic-Warmstart)				
06D2	0707	01746	01799	RST-Vektoren, DCBs usw. (werden bei Initialisierung ins RAM übertragen (4000–4035)				
Arithmetik								
0708	07F7	01800	02039	X + 1 ↓ X	xxxx	xxxx	xxxx	xxxx
0710	07F7	01808	02039	(HL...) + X ↓ X	xxxx	xxxx	xxxx	xxxx
0713	07F7	01811	02039	BCDE − X ↓ X	xxxx	xxxx	xxxx	xxxx
0716	07F7	01814	02039	BCDE + X ↓ X	xxxx	xxxx	xxxx	xxxx
0778	077C	01912	01916	0 ↓ X	0044			
07A8	07B1	01960	01969	Rundung				
07B2	07B6	01970	01974	OV-Error				
07B7	07C2	01975	01986	Festkommaaddition: (HL...) + CDE ↓ CDE	xxxx	xx	xxxx	+2
07D7	07F7	02007	02039	CDE um A-Bitpositionen nach rechts schieben	00xx	xxxx	xxxx	xx
07F8	07FB	02040	02043	Konstante 1!				
07FC	0808	02044	02056	Konstanten für LOG-Reihe				
0809	0896	02057	02189	LOG(X) ↓ X	xxxx	xxxx	xxxx	xxxx
0841	0896	02057	02189	X . ln2 ↓ X	xxxx	xxxx	xxxx	xxxx
0847	0896	02119	02189	X . BCDE ↓ X	xxxx	xxxx	xxxx	xxxx
0897	0906	02199	02311	X / 10 ↓ X	xxxx	xxxx	xxxx	xxxx
08A2	0906	02211	02310	BCDE / X ↓ X	xxxx	xxxx	xxxx	xxxx
0907	093D	02311	02365	Vorbehandlung der Exponenten und Vorzeichen bei Multiplikation und Division				
093E	0954	02366	02388	X . 10 ↓ X	xxxx	xxxx	xxxx	xxxx
0955	0963	02389	02403	Text X (für Single und Double Presision) Wenn X = 0, Z,P Wenn X < 0, C,S Wenn X > 0, keine	xxxx			
0964	0976	00404	02422	Akku in Fließkommazahl (in X) umformen	xxxx	xxxx	xxxx	xxxx
0977	0989	02423	02441	ABS(X) ↓ X	xxxx	xxxx	xxxx	xxxx
0982	0989	02434	02441	− X ↓ X	xxxx			4123
098A	0993	02442	02451	SGN(X) ↓ X	xxxx			xxxx
0994	09A3	02452	02467	Test X, auch für Integer (siehe oben 0955) bei String in X TM-Error	xxxx			xxxx
09A4	09B0	02468	02480	X ↓ (SP)			xxxx	
09B1	09BE	02481	02494	(HL...) ↓ BCDE ↓ X		xxxx	xxxx	+4
09B4	09BE	02484	02494	BCDE ↓ X			xxxx	
09BF	09CA	02495	02506	X ↓ BCDE		xxxx	xxxx	4125
09C2	09CA	02498	02506	(HL...) ↓ BCDE		xxxx	xxxx	+4
09CB	09DE	02507	02526	X ↓ (HL..)	xxxx	00	4125	+4
09D2	09DE	02514	02526	(HL...HL+(40AF)) ↓ (DE..)	xxxx	00	+4	+4

Fortsetzung folgt

Luidger Röckrath

Der geknackte TRS-80

2. Teil

Das Innenleben von Tischcomputern ist in den meisten Handbüchern verhältnismäßig schlecht dokumentiert. Firmen, die ihren Computern komplette Assembler-Listings der Betriebssystem-ROMs beipacken, wie etwa Rockwell (AIM-65), Siemens (PC-100), Eltec (Eurocom-1/2) und wenige andere, bilden lobenswerte, aber seltene Ausnahmen. mc versucht, den Mangel an Dokumentation auszugleichen – hier also Teil 2 der ROM- und RAM-Adressen des TRS-80, Level II.

					AF	BC	DE	HL
09D3	09DE	02515	02526	(DE..DE+(40AF)) (HL..)	xxxx	00	+4	+4
				(+4, wenn Single Precision Zahl in X)				
09DF	09F3	02527	02547	Vorbereitung der Argumente für Subtraktion und Addition	xxxx	xx		4125
09F4	0A0B	02548	02571	Y ⧫ X	xxxx	00	xxxx	xxxx
09FC	0A0B	02556	02571	X ⧫ Y	xxxx	00	xxxx	xxxx
0A03	0A0B	02563	02571	4121 ⧫ DE (!), 411D ⧫ DE(#)	xxxx		xxxx	
0A0C	0A25	02572	02597	Vergleich: X–BCDE (Single)	xxxx			xxxx
0A26	0A38	02598	02616	UP für Vergleich				
0A39	0A48	02617	02632	Vergleich: HL–DE (Integer)	xxxx			
0A49	0A77	02633	0269	Vergleich: X–Y (Double)	xxxx	xxxx	xxxx	xxxx
0AAF	0A77	02639	02679	Vergleich: X– (DE) (Double)	xxxx	xxxx	xxxx	xxxx
				Nach allen Vergleichen ist bei Gleichheit				
				der Operanden das Zero-Flag gesetzt				
0A7F	0AB0	02687	02736	CINT(X) ⧫ X ⧫ HL	xxxx	xxxx	xxxx	xxxx
0AB1	0ADA	02737	02778	CSNG(X) ⧫ X	xxxx	xxxx	xxxx	xxxx
0ADB	0AF3	02779	02803	CDBL(X) ⧫ X	xxxx	xxxx	xxxx	xxxx
0A9A	0AA2	02714	02722	2 ⧫ (40AF), HL ⧫ X	02			
0A9D	0AA2	02717	02722	2 ⧫ (40AF)	02			
0AEC	0AF3	02796	02803	8 ⧫ (40AF)	08	043E		
0AEF	0AF3	02799	02803	4 ⧫ (40AF)	04			
0AF4	0AFA	02804	02810	Wenn in X kein String, TM-Error	xxxx			
0AF6	0AFA	02806	02810	TM-Error				
0AFB	0B25	02811	02853	Unterprogramm für CINT, FIX und INT				
0B26	0B36	02854	02870	FIX(X) ⧫ X	xxxx	xxxx	xxxx	xxxx
0B37	0B9F	02871	02975	INT(X) ⧫ X	xxxx	xxxx	xxxx	xxxx
0BA0	0BA9	02976	02985	Unterprogramm für INT				
0BAA	0BC6	02986	03014	Unterprogramm für DIM				
0BC7	0C6F	03015	03183	Integer-Arithmetik				
0BC7	0BF1	03015	03057	HL + DE ⧫ HL ⧫ X	xxxx	xxxx	xxxx	xxxx
0BD2	0BF1	03026	03057	HL– DE ⧫ HL ⧫ X	xxxx	xxxx	xxxx	xxxx
0BF2	005A	03058	03162	HL . DE ⧫ HL ⧫ X	xxxx	xxxx	xxxx	xxxx
				Bei Overflow wird in Fließkommaformat umgewandelt				
0C5B	0C6F	03163	03183	UP für Integeraddition				
0C70	0E64	03184	03684	Double-Precision-Arithmetik				
0C70	0D32	03184	03378	X – Y ⧫ X	xxxx	xxxx	xxxx	xxxx
0C77	0D32	03191	03378	X + Y ⧫ X	xxxx	xxxx	xxxx	xxxx
0D33	0D44	03379	03396	Festkommaaddition	xxxx	00	4124	412E
0D45	0D56	03397	03414	Festkommasubtraktion	xxxx	00	4124	412E
0D57	0D68	03415	03432	Unterprogramm für Addition				
0D69	0D8F	03433	03471	(HL...HL–8) um A-Bits nach links schieben	xxxx	xx	0000	
0D90	0D96	03476	03478	(4123..411C) um 1 Bit nach links schieben	xxxx		0000	4123
0D97	0DA0	03479	03488	(HL..HL+8) um 1 Bit nach rechts schieben	xxxx	00		+8
0DA1	0DD3	03489	03439	X .Y ⧫ X	xxxx	xxxx	xxxx	xxxx
0DD4	0DDB	03440	03447	Konstante 10#				
0DD8	0DDB	03444	03447	Konstante 10!				
0DDC	0E38	03448	03640	X / 10 ⧫ X	xxxx	xxxx	xxxx	xxxx
0DE5	0E38	03557	03640	X / Y ⧫ X	xxxx	xxxx	xxxx	xxxx
0E39	0E4C	03641	03660	Unterprogramm für Multiplikation und Division				
0E4D	0E64	03661	03684	X · 10 ⧫ X	xxxx	xxxx	xxxx	xxxx

					AF	BC	DE	HL
Konvertierung für Zahlen-Ein- und Ausgabe								
0E65	0FA6	03685	04006	Umwandlung eines Strings (Zeiger: HL) in eine Zahl (in X)	xxxx	xxxx	xxxx	P
				Unterprogramme:				
0EFB	0F09	03835	03849	Wenn Z=1 ist, wird X in eine Fließkommazahl einfacher Genauigkeit umgewandelt, sonst doppelte Genauigkeit	xxxx			
0F0A	0F17	03864	03863	Typrichtige Multiplikation mit 10	xxxx	xxxx	xxxx	xxxx
0F18	0F28	03864	03880	Typrichtige Division durch 10	xxxx	xxxx		
0F89	0F93	03977	03987	X + A ⧫ X (single)	xxxx	xxxx	00xx	xxxx
0FA7	0FBC	04007	04028	Text „–in–"und HL dezimal ausdrucken	xxxx	xxxx	00xx	xxxx
0FAF	0FBC	04015	04028	HL dezimal ausdrucken	xxxx	xxxx	00xx	xxxx
0FBD	1363	04029	04963	Zahl in X in String umwandeln	xxxx	xxxx	xxxx	xxxx
				Dieser wird in (4130) ff. abgelegt und mit 0 beendet				
1364	13E1	04964	05089	Daten für Umwandlung von Zahlen in Strings				
1364	136B	04964	04971	Konstante $1{\cdot}10^{10}$ #				
136C	1373	04972	04979	Konstante $1{\cdot}10^{15}$ #				
1374	137B	04980	04987	Konstante $1{\cdot}10^{16}$ #				
137C	1383	04988	04995	Konstante ·5 #				
1380	1383	04992	04995	Konstante ·5 !				
1384	138B	04996	05003	Konstante $1{\cdot}10^{16}$ #				
Single Precision Funktionen								
13E2	13E6	05090	05094	UP für SQR und TAN, bewirkt Multiplikation des Ergebnisses mit −1.				xxxx
13E7	1478	05095	05240	SQR(X) ⧫ X (nach SQR(X)=EXP(0.5.LOG(X))	xxxx	xxxx	xxxx	xxxx
13F2	1478	05106	05240	(SP) ⬌ X ⧫ X	xxxx	xxxx	xxxx	xxxx
1439	1478	05177	05240	EXP(X) ⧫ X	xxxx	xxxx	xxxx	xxxx
1479	1499	95241	05273	Konstanten für EXP-Reihe				
149A	14C8	05274	05320	Reihe: $y = c_2x+c_3x^3+c_4x^5$... berechnen HL zeigt auf die Koeffizienten	xxxx	xxxx	xxxx	xxxx
14A9	14C8	05289	05320	Reihe: $y = c_1+c_2x+c_3x^2$... berechnen	xxxx	xxxx	xxxx	xxxx
14C9	1540	05321	05440	RND(X) ⧫ X	xxxx	xxxx	xxxx	xxxx
14F0	1540	05360	05440	Erzeugung einer reellen Zufallszahl (in X)	xxxx	xxxx	xxxx	xxxx
1541	158A	05441	05514	COS(X) ⧫ X	xxxx	xxxx	xxxx	xxxx
1547	158A	05447	05514	SIN(X) ⧫ X	xxxx	xxxx	xxxx	xxxx
158B	158E	05515	05518	Konstante π/2!				
158F	1592	05519	95522	Konstante 0.25!				
1593	15A7	05523	06643	Konstanten für SIN, COS-Reihe				
15A8	15BC	05544	05564	TAN(X) ⧫ X	xxxx	xxxx	xxxx	xxxx
15BD	15E2	05565	05602	ATN(X) ⧫ X	xxxx	xxxx	xxxx	xxxx
15E3	1607	05603	05639	Konstanten für ATN-Reihe				
Tabellen								
1608	164F	05640	05711	Sprungtabelle für Funktionen (Zwischencodes D7-FA)				
1650	1821	05712	06177	Tabelle der Basic-Keywords. Der erste Buchstabe ist durch das gesetzte Bit 7 gekennzeichnet. Die Keywords sind nach aufsteigenden Zwischencodes (80-FA) sortiert				
1822	1899	06178	06297	Sprungtabelle für Befehle (Zwischencodes 80-BB)				
189A	18A0	06298	06304	Prioritätscodes für Operatoren				
18A1	18AA	06305	06314	Sprungtabelle für Typanpassung				
18AB	18C8	06315	06344	Sprungtabelle für Grundrechenarten und Vergleich. Für jeden der drei numerischen Datentypen sind diese 5 Adressen enthalten				
18C9	18F6	06345	06390	Tabelle der Fehlerabkürzungen nach aufsteigenden Fehlercodes sortiert				
18F7	191C	06391	06428	Daten, die bei Initialisierung ins RAM übertragen werden (4080–40A6)				
Programmeingabe und -ausführung								
191D	1923	06429	06435	Text „Error"				
1924	1928	06436	06440	Text „–in–"				
1929	192F	06441	06447	Text „READY(CR)"				
1930	1935	06448	06453	Text „Break"				
1936	1954	0654	06484	Unterprogramm für FOR und GOSUB usw. (holt Daten vom Stack zurück)				

					AF	BC	DE	HL
1955	1962	06485	06498	Unterprogramm, macht Platz für einzufügende Programmzeile oder Variable. Vorher wird getestet, ob noch genügend Platz frei ist.				
1963	197D	06499	06525	Unterprogramm, testet ob noch 2.C Bytes Speicher frei. Wenn nicht, OM-Error		xxxx 00		
197A	197D	06522	06525	OM-Error				
197E	1990	06526	06544	Implizites END (Erreichen des Programmendes ohne END-Anweisung). Nur bei dieser Programmbeendung wird der NR-Error erkannt!				
1991	1A18	06545	06680	Fehlerbehandlung				
1991	1991	06545	06545	SN-Error in DATA-Zeile				
1997	1997	06551	06551	SN-Error				
199A	199A	06554	06554	/O-Error				
199D	199D	06557	06557	NF-Error				
19A0	19A0	06560	06560	RW-Error				
19A2	19A2	06562	06562	Fehlercode in E (zwischen 0 und 2C)				
1A19	1AF7	06681	06903	Hauptschleife:				
1A3F	1A75	06719	06773	Programmeingabe unter AUTO				
1A76	1AA6	06774	06822	Programmeingabe und Zwischencodeerzeugung				
1AA7	1AF7	06823	06903	Neue Zeile in Programmtext einfügen				
1AF8	1B0F	06904	06927	Zeiger im ganzen Programmtext erneuern		xxxx	xxxx	xxxx
1AFC	1B0F	06908	06927	Zeiger im Programmtext ab DE erneuern		xxxx	xxxx	xxxx
1B10	1B48	06928	06984	Argumente von LIST, DELETE usw. analysieren Die Adresse des Anfangs der ersten Zeile ist nachher in BC, die zweite Zeilennummer in (SP)		xxxx xxxx	xxxx	P
1B2C	1B48	06956	06984	Sucht Zeile mit Nummer DE im Programmtest. Ist diese Zeile vorhanden, sind beim Rücksprung C und Z gesetzt und die Adresse der Zeile ist in BC. Ist die Zeile nicht vorhanden, enthält BC die Adresse der nächsten Zeile bzw. des Programmendes		xxxx xxxx		xxxx
1B49	1BB2	06985	07090	NEW-Befehl:				
1B49	1B49	06985	06985	alles löschen				
1B61	1B61	07009	07009	nur Variablen löschen				
1B9A	1B9A	07066	07066	Stack neu initialisieren				
1BB3	1BBF	07091	07103	„?" ausdrucken und INLINE (siehe 0361)				
1BC0	1C8F	07104	07311	Text in I/O-Buffer in Zwischencode umwandeln (von HL an). Bei der Rückkehr ist in BC die Länge des Zwischencodes +5, in HL der Anfang des Buffers-3. (Der Zwischencode beginnt 2 Bytes vor Anfang des I/O-Buffers.)		xxxx xxxx	xxxx	xxxx
1C90	1C95	07312	07317	RST 18H UP: Vergleich HL-DE, die Flagbeeinflussung entspricht den normalen Vergleichsbefehlen		xxxx		
1C96	1CA0	07318	07328	RST 8H UP: (HL) wird mit dem dem RST 8 folgenden Byte verglichen. Bei Gleichheit wird der RST 10 angesprungen, sonst wird ein SN-Error erkannt		xxxx		P
1CA1	1D1D	07329	07453	FOR-TO-STEP-Schleife auswerten				
1D1E	1D90	07454	07568	Steuerung der Programmausführung: Syntaxprüfung, Ansprung der Befehle über Sprungtabelle				
1D78	1D90	07544	07568	RST 10H UP: analysiert Programmtext. HL dient als Zeiger und wird nachgestellt. Ist (HL+1) eine Zahl, so ist bei Rückkehr das Carry-Flag gesetzt, bei 3A und 00 das Zero-Flag. Space und LF werden übergangen		xxxx		P
1D91	1D9A	07569	07578	RESTORE-Befehl				
1D9B	1DAD	07579	07597	Behandlung von Tastendrücken während Programmausführung oder Auflistung				
1DAE	1DE3	07598	07652	Explizites END (siehe auch oben 197E)				
1DB4	1DE3	07604	07651	Programmunterbrechung (Break)				
1DE4	1DF6	07652	07670	CONT-Befehl				
1DF7	1DFF	07671	07679	TRON-Befehl (TRACE-Flag wird auf AF gesetzt)				
1DF8	1DFF	07672	07679	TROFF-Befehl (TRACE-Flag wird auf 0 gesetzt)				
1E00	1E3C	07680	07640	DEFSTR-Befehl				
1E03	1E3C	07683	07640	DETINT-Befehl				
1E06	1E3C	07686	07640	DEFSNG-Befehl				
1E09	1E3C	07689	07640	DEFDBL-Befehl Alle 4 Befehle manipulieren die Typencodetabelle (siehe 4101–411A)				
1E3D	1E44	07741	048	UP, wenn in (HL) ein Buchstabe ist, ist das nicht Carry gesetzt		xxxx		
1E45	1E4E	07749	07758	Ganzzahligen Wert (<= 32767) des Ausdruckes ermitteln (in DE), (bei Überschreiten dieser Grenze FC-Error)		xxxx xxxx	xxxx	P
1E4A	1E4E	07754	07758	FC-Error				
1E4F	1E79	07759	07801	String in Zahl umwandeln. Nur für positive ganze Zahlen bis 65530 (Zeilennummern). Ergebnis in DE		xxxx	xxxx	P
1E7A	1EA2	07802	07842	CLEAR-Befehl				
1EA3	1EDD	07843	07901	RUN-Befehl				

					AF	BC	DE	HL
1EB1	1EDD	07857	07901	GOSUB-Befehl				
1EC2	1EDD	07874	07901	GOTO-Befehl				
1ED9	1EDD	07897	07901	UL-Error				
1EDE	1F20	07902	07968	RETURN-Befehl				
1F05	1F20	07941	07968	DATA-Befehl				
1F07	1F20	07943	07968	REM-Befehl				
1F21	1F6B	07969	08043	LET-Befehl				
1F6C	1FAE	08044	08110	ON-Befehle				
1F70	1F94	08048	08084	ON ERROR-Befehl				
1F95	1FAE	08085	08110	ON GOTO (GOSUB)-Befehl				
1FAF	1FF3	08111	08179	RESUME-Befehl				
1FF4	2007	08180	08199	ERROR-Befehl				
2008	2038	08200	08248	AUTO-Befehl (Initialisierung und Sprung zur Hauptschleife)				
2039	2066	08249	08294	IF-THEN-ELSE-Befehl				
2067	2177	08295	08567	PRINT-Befehl				
2067	2067	08295	08295	Ansprung LPRINT				
206F	206F	08303	08303	Ansprung PRINT				
2072	207C	08306	08316	Wenn PRINT# Kassettenrecorder vorbereiten				
2084	20A4	08324	08356	α auswerten				
20F9	2107	08441	08455	wenn Cursor nicht in Position 0, CR ausgeben	0044			
20FE	2107	08446	08455	CR – ausgeben	0044			
2137	2168	08503	08552	TAB auswerten				
2178	217E	08568	08574	Text „?REDO(CR)"				
217F	22B5	08575	08885	INPUT, INPUT# und READ				
219A	219A	08602	08602	Ansprung INPUT und INPUT#				
21EF	21EF	08687	08687	Ansprung READ				
2286	2295	08838	08853	Text „?Extra ignored(CR)"				
22B6	2334	08886	09012	NEXT-Befehl				
2335	25D8	09013	09688	Auswertung von Ausdrücken. Der Ausdruck steht als Text (in Zwischencodes) im RAM, HL dient als Zeiger und wird entsprechend incrementiert. Nach der Auswertung liegt in X das Ergebnis vor				
2335	2335	09013	09013	Ansprung für Ausdruck, der mit einer Klammer beginnt	xxxx	xxxx	xxxx	P
2337	2337	09015	09015	normaler Ansprung	xxxx	xxxx	xxxx	P
2406	2437	09224	09271	Ansprung der Grundrechenarten über Sprungtabelle				
2490	249E	09360	09374	HL / DE ⇒ X	xxxx	xxxx	xxxx	xxxx
249F	2531	09375	09521	Auswertung eines Teilausdruckes bestehend aus Variable, Konstante, NOT Ausdruck, Funktion oder einem in Klammern eingeschlossenen Ausdruck				
24CF	24D8	09423	09432	ERR-Funktion				
24DD	24E6	09437	09446	ERL-Funktion				
24EB	24FE	09451	09470	VARPTR-Funktion				
252C	2531	09516	09521	Auswertung eines Ausdrucks in Klammern (z. B.: Argument für Funktionen)	xxxx	xxxx	xxxx	P
2532	253F	09522	09535	Vorzeichen (–) auswerten				
2540	254D	09536	09549	Wert einer Variablen in Ausdruck einbringen				
254E	258B	09550	09611	Argumente analysieren und Funktionen anspringen (über Tabelle)				
258C	25B7	09612	09695	Stringvergleich				
25C4	25D8	09668	09688	NOT ausführen				
25D9	25E8	09689	09704	RST 20H UP Prüft Typ des Inhaltes von X INTEGER S,C P SINGLE C DOUBLE P STRING Z,C P Je nach Typ werden die Flags beeinflußt	xxxx			
25E9	2602	09705	09730	AND und OR ausführen				
2603	27C8	09731	10184	Verwaltung der Variablentabelle				
2608	2608	09736	09736	Ansprung DIM-Befehl				
260D	260D	09741	09741	Ansprung, Variable suchen und wenn nicht vorhanden, neu einrichten. Beim Rücksprung ist die Adresse in der Variablentabelle in DE	xxxx	xxxx	xxxx	P
260D	2651	09741	09809	Variablennamen prüfen und Typ feststellen				
2652	268D	09810	09869	Variable in Tabelle suchen				
26A0	26CE	09888	09934	Neue Variable einrichten				
26E9	27C8	09961	10184	Verwaltung der Feldvariablen. Suchen, neu Einrichten, Dimensionieren				
273D	2741	10045	10049	BS-Error				
27C9	27D3	10185	10195	MEM-Funktion				
27D4	27F4	10196	10228	FRE-Funktion				

					AF	BC	DG	HL
27F5	27FD	10229	10237	POS-Funktion				
27F8	27FD	10232	10237	Akku als Integerzahl nach HL und X	xxxx			xxxx
27FE	2818	10238	10264	USR-Funktion				
2819	2827	10265	10279	Wert in X wird in den Typ, dessen Code im Akku steht, umgewandelt	xxxx	xxxx	xxxx	xxxx
2828	2835	10280	10293	Wenn Rechner im Direkt-Mode, ID-Error	xxxx			
2831	2835	10289	10293	ID-Error				
2836	2AEE	10294	10990	Stringspaceverwaltung und Stringfunktionen				
2836	2856	10294	10326	STR$-Funktion				
2857	2864	10327	10340	Platz für String mit der Länge A in Stringspace schaffen und Adresse (in DE) in Zwischenspeicher eintragen	xx	xxxx	xxxx	40D3
2865	28A5	10341	10405	Stringkonstante in Zwischenspeicher und X übernehmen. Wenn Zwischenspeicher voll, ST-Error	xxxx	xxxx	40D6	P
28A1	28A5	10401	10405	ST-Error				
28A6	28BE	10406	10430	String ab HL+1 bis Anführungsstrichen oder 0 ausdrucken	xxxx	xxxx	00xx	xxxx
28BF	298E	10431	10638	In der Stringspace Platz für String mit der Länge A machen. Adresse in DE. Wenn nötig, Stringspace umsortieren und überflüssige Strings entfernen. Ist es auch so nicht möglich, genügend Platz zu schaffen, OS-Error	xx	xxxx	xxxx	xxxx
298F	29C5	10639	10693	Stringverknüpfung				
29C6	29D6	10694	10710	String in Stringspace übernehmen				
29CE	29D6	10702	10710	String mit der Länge L in Stringspace übernehmen Zeiger: BC (String) und DE (Adresse in Stringspace)	xxxx	xxxx	xxxx	00
29D7	29F4	10711	10740	Letzten String aus Stringspace und Zwischenspeicher entfernen				
29F5	2A02	10741	10754	Letzten String aus Zwischenspeicher entfernen				
2A03	2A0E	10755	10766	LEN(X) ⇒ X ⇒ HL	xxxx	27F8	xxxx	xxxx
2A0F	2A1E	10767	10782	ASC(X) ⇒ X ⇒ HL	xxxx	27F8	xxxx	xxxx
2A1F	2A2E	10783	10798	CHR$-Funktion				
2A2F	2A60	10799	10848	STRING$-Funktion				
2A61	2A90	10849	10896	LEFT$-Funktion				
2A91	2A99	10897	10905	RIGHT$-Funktion				
2A9A	2AC4	10906	10948	MID$-Funktion				
2AC5	2ADE	10949	10974	VAL-Funktion				
2ADF	2AE6	10975	10982	UP für Stringverarbeitung				
2AE7	2AEE	10983	10990	MID$ auf linker Seite der Zuweisung Nur in Disk-Basic vorhanden, daher Sprung nach 41D9.				
2AEF	2AFA	10991	11002	INP-Funktion				
2AFB	2B00	11003	11008	OUT-Befehl				
2B01	2B0D	11009	11021	Ganzzahliger Wert des Ausdrucks nach DE	xxxx	xxxx	xxxx	P
2B0E	2B1A	11022	11034	2 Argumente (< 256) für OUT analysieren				
2B1B	2B28	11035	11048	Ausdruck auswerten und ganzzahligen Wert (< 256) nach A. Wird dieser Wert überschritten, FC-Error	xxxx	xxxx	xxxx	P
2B29	2B7D	11049	11133	LIST-Befehl				
2B29	2B29	11049	11049	LLIST-Ansprung				
2B2E	2B2E	11054	11054	LIST-Ansprung				
2B75	2B7D	11125	11133	UP für LIST, druckt String (Zeiger HL) bis 0.	xxxx			xxxx
2B7E	2BC5	11134	11205	UP für LIST, Zwischencode im Programmtext (Zeiger HL) in Programmtext mit Keywords zurückverwandeln, welcher nachher im I/O-Buffer vorliegt	xxxx	xxxx	xxxx	P
2BC6	2BF4	11206	11252	DELETE-Befehl				
2BF5	2C1E	11253	11294	CSAVE-Befehl				
2C1F	2CA4	11295	11428	CLOAD und CLOAD?				
2C3D	2CA4	11325	11428	Programm von Kassette laden (D=0) oder mit residentem Programm vergleichen (D=FF). Filename in E				
2CA5	2CA9	11429	11433	Text „BAD(CR)"				
2CAA	2CB0	11434	11440	PEEK-Funktion				
2CB1	2CBC	11441	11452	POKE-Befehl				
2CBD	2E52	11453	11858	USING (formatiertes Ausdrucken von Zahlen und Strings)				
2E53	2FFA	11859	12282	EDIT-Befehl (Zeileneditor)				
2E53	2E53	11859	11859	Ansprung nach SN-Fehlern				
2E60	2E60	11872	11872	normaler Ansprung				
2E60	2E6F	11872	11887	Analyse des Argumentes				
2E70	2E9A	11888	11930	Rückübersetzung des Programmtextes				
2E9B	2E9D	11931	11933	Eingabe				
2E9E	2EAF	11934	11951	Verarbeitung von Zahleneingaben				
2EB0	2F09	11952	12041	Ansprung der Unterbefehle				
2F0A	2F15	12042	12053	Space				
2F16	2F3F	12054	12095	K (Kill)				
2F1C	2F3F	12060	12095	S (Search)				
2F40	2F49	12096	12105	L (List)				
2F4A	2F64	12106	12132	D (Delete)				

2F65	2F74	12133	12148	C (Change)
2F75	2FD1	12149	12241	H (Hack and insert)
2F78	2FD1	12152	12241	X (Extend)
2F7D	2FD1	12157	12241	I (Insert)
2FD2	2FDF	12242	12255	◆ (Backspace)
2FE0	2FF5	12256	12277	Enter (Exit)
2FE3	2FF5	12259	12277	E (Save Changes and Exit)
2FF6	2FFA	12278	12282	Q (Cancel and Exit)
2FFF	2FFF	12287	12287	Ende des ROMs

I/O-Adressen

I/O-Ports:

FF	FF	00255	00255	Universeller Ein/Ausgabeport für Systemanwendungen: Bit 0,1: Steuern die Spannung am AUX-Stecker: 00 und 11 = 0,45 V, 01 = 0,9 V, 10 = 0 V. Bit 2: Wenn dieses Bit gesetzt ist, ist das Reedrelais angesteuert, also der Kassettenrecorder angeschaltet. Bit 3: Umschaltung 64/32 cpl (0/1) Bit 7: Eingabeport. Wenn am EAR-Stecker eine Spannung auftritt, wird dieses Bit gesetzt. Es behält diesen Zustand, bis es durch Schreiben zurückgesetzt wird

Memory mapped I/O:

37DE	37DE	14302	14302	Communication Status Address
37DF	37DF	14303	14303	Communication Data Address
37E0	37E0	14304	14304	Interrupt Latch Address
37E1	37E1	14305	14305	Disk Drive Latch
37E4	37E4	14308	14308	Cassette Select Latch
37E8	37E8	14312	14312	Line Printer Address
37EC	37EC	14316	14316	Floppy Disc Controller Address
3800	3BFF	14336	15359	Tastatur-Adressen:

nur mit Expansion

				Bit	7	6	5	4	3	2	1	0
3801	3801	14337	14337		G	F	E	D	C	B	A	@
3802	3802	14338	14338		O	N	M	L	K	J	I	H
3804	3804	14340	14340		W	V	U	T	S	R	Q	P
3808	3808	14344	14344							Z	Y	X
3810	3810	14352	14352		7	6	5	4	3	2	1	0
3820	3820	14368	14368		/	●	–	,	;	:	9	8
3840	3840	14400	14400		Space	◆	◆	◆	◆	Br	Cl	CR

(Br = Break, Cl = Clear, CR = Enter) SHIFT

3880	3880	14464	14464	Bei diesen Adressen wird genau eine Zeile mit 8 Tasten abgefragt. Wenn eine Taste gedrückt ist, ist das entsprechende Bit gesetzt
387F	387F	14463	14463	Abfrage aller Tasten außer Shift. Wenn dieser Speicherplatz 0 ist, ist keine Taste gedrückt
38FF	38FF	14591	14591	Dergleichen mit Shift
3C00	3FFF	15360	16383	Bildschirm-RAM (enthält kein Bit 6!)

RAM-Adressen

4000	4014	16384	16404	RST-Vektoren, wird einer der RSTs 8-38H ausgeführt, so wird an diese Adressen gesprungen:
4000	4000	16384	16384	RST 8H-Vektor (C3 96 1C = JP 1C96)
4003	4005	16387	16387	RST 10H-Vektor (C3 78 1D = JP 1D78)
4006	4008	16390	16390	RST 18H-Vektor (C3 90 1C = JP 1C90)
4009	4009	16393	16393	RST 20H-Vektor (C3 D9 25 = JP 25D9)
400C	400C	16396	16396	RST 28H-Vektor (C9 00 00 = RET), wird von INCH bei Drücken der Break-Taste angesprungen
400F	400F	16399	16399	RST 30H-Vektor (C9 00 00 = RET)
4012	4012	16402	16402	RST 38H-Vektor (FB C9 00 = EI,RET) Die RSTs 30H und 38H werden im Level-II-Basic nicht verwendet, wohl aber bei Anschluß der Expansion oder eines Floppies
4015	402C	16405	16428	Device Control Blocks (DCB)
4015	401C	16405	16412	Keyboard DCB
4015	4015	16405	16405	DCB-Typ (01)
4016	4017	16406	16407	Treiberadresse (03E3)
4018	401C	16408	16412	(00 00 00 4B 49)
401D	4024	16413	16420	Display DCB
401D	401D	16413	16413	DCB-Typ (06)
401E	401E	16414	16415	Treiberadresse (0458)
4020	4021	16416	16417	Cursoradresse (3C00)

4022	4022	16418	16418	Cursorzeichen (00), 0 wenn Cursor ausgeschaltet, sonst das Zeichen, welches auf der Cursorposition stand
4023	4024	16419	16420	(44 4F)
4025	402C	16421	16428	Printer DCB
4025	4025	16421	16421	DCB-Typ (07)
4026	4027	16422	16423	Treiberadresse (058D)
4028	4028	16424	16424	Anzahl der Zeilen pro Seite (43)
4029	4029	16425	16425	Nummer der Zeile in laufender Seite (00)
402A	402D	16426	16428	(00 50 52)
402D	4032	16429	16434	(0D 00 50 C7 00 00)
4033	4035	16435	16437	Wird angesprungen, wenn DCB-Typen nicht übereinstimmen (3E 00 C9 = LD A,0,RET)
4036	403C	16438	16444	Letzter Tastaturstatus, dient dazu festzustellen, welche Taste neu gedrückt worden ist
403D	403D	16445	16445	= OUT (FF)
403E	407F	16446	16511	Wird nur vom DOS belegt, kann also im LEVEL-II-Basic z. B. für kleine USR-Routinen verwendet werden
4050	4051	16464	16465	FDC-Interrupt-Vector
4052	4053	16466	16467	Communications Interrupt Vector
405E	405F	16478	16479	25 ms „Heartbeat"-Interrupt
4080	408D	16512	16525	UP für Division
408E	408F	16526	16527	Startadresse von USR-Funktion (1E4A=FC-Error)
4090	4092	16528	16530	Zwischenspeicher für RND (40 E6 4D)
4093	4095	16531	16533	UP für INP (DB 00 C9 = IN A,(00),RET)
4096	4098	16534	16536	UP für OUT (D3 00 C9 = OUT (00),A,RET)
4099	4099	16537	16537	Speicher für INKEY$-Funktion, enthält letzten Tastendruck während des Programmes bzw. 0(00)
409A	409A	16538	16538	Letzter Errorcode für ERR (00)
409B	409B	16539	16539	Position des Cursors in der Zeile (00)
409C	409C	16540	16540	Flag für Ausgabe (00 = Display, 80 = Kassette, 01 = Printer) (00)
409D	409D	16541	16541	Anzahl der Zeichen pro Zeile auf dem Bildschirm (wird für Zahlenausgabe benötigt) (40)
409E	409E	16542	16542	Letzte Tabulatorposition (30)
409F	409F	16543	16543	n. v.
40A0	40A1	16544	16545	Anfang der Stringspace (434C) 32461
40A2	40A3	16546	16547	Aktuelle Zeilennummer (FFFE)
40A4	40A5	16548	16549	Anfang des Programmtextes (42E9) 22529
40A6	40A6	16550	16550	Position des Cursors in der Zeile auf dem Bildschirm (20)
40A7	40A8	16551	16552	Anfang des I/O-Puffers (41E8) 16872
40A9	40A9	16553	16553	Flag für Input (00 = Keyboard, sonst Kassette)
40AA	40AC	16554	16556	Zwischenspeicher für RND
40AD	40AD	16557	16557	n. v.
40AE	40AE	16558	16558	DIM-Flag (00 = kein DIM)
40AF	40AF	16559	16559	Typ-Code des Inhaltes des X-Registers
40B0	40B0	16560	16560	Flag für Zwischencodeerzeugung (nach DATA = 4E, sonst 0)
40B1	40B2	16561	16562	Letzter Speicherplatz, der für Basic zur Verfügung steht 32511
40B3	40D5	16563	16597	Zwischenspeicher für momentan verarbeitete Strings
40B3	40B4	16563	16564	Zeiger auf nächsten freien Zwischenspeicherplatz
40B5	40D5	16565	16597	11 Zwischenspeicher für Strings (Länge, Adresse)
40D3	40D5	16595	16597	vorläufiger Zwischenspeicher, vor Eintragung in den Zwischenspeicher
40D6	40D7	16598	16599	Letztes freie Byte im Stringspace
40D8	40D8	16600	16600	Flag für Zahlenausgabe
40D8	40D9	16600	16601	Flag für Feldverwaltung
40DA	40DB	16602	16603	Zeilennummer, der DATA-Zeile, die gerade gelesen wird
40DC	40DC	16604	16604	Flag zur Sperrung von Feldvariablen (0 = Felder freigegeben)
40DD	40DD	16605	16605	Flag
40DE	40DE	16606	16606	Flag für INPUT (00 = Keyboard, sonst READ)
40DF	40E0	16607	16608	Startadresse von Objektfiles (mit SYSTEM geladen), wird auch anderweitig verwendet
40E1	40E1	16609	16609	AUTO-Flag (00 = kein AUTO)
40E2	40E3	16610	16611	AUTO-Anfangsadresse
40E4	40E5	16612	16613	AUTO-Increment
40E6	40E7	16614	16615	Adresse des Befehls, der gerade verarbeitet wird
40E8	40E9	16616	16617	Wert mit dem Stackpointer initialisiert wird, wird durch FOR, GOSUB usw. erniedrigt.
40EA	40EB	16618	16619	Zeile in der der letzte Fehler auftrat (für ERL)
40EC	40ED	16620	16621	Aktuelle Zeile für „."-Option
40EE	40EF	16622	16623	Adresse des Befehl, bei dessen Ausführung der letzte Fehler auftrat
40F0	40F1	16624	16625	Anfangsadresse der Errortraproutine. Wenn kein Errortrap benutzt wird, beinhalten diese Speicherplätze 0
40F2	40F2	16626	16626	Fehlerflag: Wird beim Auftreten eines Errortraps gesetzt (FF) und durch RESUME wieder zurückgesetzt (00)
40F3	40F4	16627	16628	Zwischenspeicher für die Auswertung von Ausdrücken
40F5	40F6	16629	16630	Zeilennummer der letzten Programmunterbrechung (durch Fehler oder Break)
40F7	40F8	16631	16632	Adresse der letzten Programmunterbrechnung, 0 wenn keine aufgetreten, oder Fortführung der Programmausführung nicht möglich

40F9	40FA	16633	16634	Anfang der Variablentabelle
40FB	40FC	16635	16636	Anfang der Arraytabelle
40FD	40FE	16637	16638	Ende der Arraytabelle
40FF	4100	16639	16640	Datazeiger, zeigt auf das Trennzeichen nach den zuletzt gelesenen Daten
4101	411A	16641	16666	Tabelle, enthält in alphabetischer Reihenfolge für jeden Buchstaben einen Typcode.
				Beinhaltet ein Variablenname keine Typbezeichnung, wird der Typ der Tabelle entnommen.
				Durch NEW wird 04 in die ganze Tabelle geschrieben.
				Durch DEFSTR, DEFINT, DEFSNG und DEFDBL wird dieser Wert geändert
411B	411B	16667	16667	TRACE-Flag (00 = TROFF, AF = TRON)
411C	411C	16668	16668	Carry für Schiebeoperationen
411D	4124	16669	16676	X-Register, wichtigstes Register für Werte aller 4 Typen.
				Enthält z. B. das Ergebnis eines Ausdruckes oder Argument und Ergebnis bei Funktionsaufrufen
411D	4124	16669	16676	Zahl doppelter Genauigkeit:
				411D enthält das niederwertigste Byte, 4123 das höchstwertigste und 4124 den Exponenten
4121	4124	16673	16676	Zahl einfacher Genauigkeit: Abspeicherung ähnlich wie bei Zahlen doppelter Genauigkeit
4121	4122	16673	16674	Integerzahl: LSB/MSB (4121/4122)
4121	4122	16673	16674	Strings: Adresse, die auf Zwischenspeicher oder Variablentabelle zeigt, wo String abgespeichert ist
4125	4126	16675	16678	Zwischenspeicher für Arithmetik
4127	412E	16679	16686	Y-Register, wichtiges Register besonders für 16stellige Arithmetik
				(enthält den zweiten Operanden). Abspeicherung wie im X-Register
412F	412F	16687	16687	Carry für Schiebeoperationen
4130	4149	16688	16713	Output-Buffer, enthält nach der Rückkonvertierung von Zahlen den entsprechenden String
414A	4151	16714	16721	Zwischenspeicher für Arithmetik
4152	41E4	16722	16868	Zeiger für Erweiterung des Befehlssatzes durch das Disk-Basic:
4152	41A5	16722	16805	Zeiger für neue Disk-Basic-Befehle und Funktionen.
				Bei der Initialisierung des Basic wird auf alle Zeiger
				C3 2D 01 = JP 012D = L3-Error geschrieben
4152	4154	16722	16724	CVI-Funktion
4155	4157	16725	16727	FN-Funktion
4158	415A	16728	16730	CVS-Funktion
415B	415D	16731	16733	DEF-Befehl
415E	4160	16734	16736	CVD-Funktion
4161	4163	16737	16739	EOF-Funktion
4164	4166	16740	16742	LOC-Funktion
4167	4169	16743	16745	LOF-Funktion
416A	416C	16746	16748	MKI$-Funktion
416D	416F	16749	16751	MKS$-Funktion
4170	4172	16752	16754	MKD$-Funktion
4173	4175	16755	16757	CMD-Befehl
4176	4178	16758	16760	TIME$-Funktion
4179	417B	16761	16763	OPEN-Befehl
417C	417E	16764	16766	FIELD-Befehl
417F	4181	16767	16769	GET-Funktion
4182	4184	16770	16772	PUT-Befehl
4185	4187	16773	16775	CLOSE-Befehl
4188	418A	16776	16778	LOAD-Befehl
418B	418D	16779	16781	MERGE-Befehl
418E	4190	16782	16784	NAME-Befehl
4191	4193	16785	16787	KILL-Befehl
4194	4196	16788	16790	&-Funktion
4197	4199	16791	16793	LSET-Befehl
419A	419C	16794	16796	RSET-Befehl
419D	419F	16797	16799	INSTR-Funktion
41A0	41A2	16900	16802	SAVE-Befehl
41A3	41A5	16803	16805	LINE-Befehl
41A6	41E4	16806	16868	Zeiger zur Erweiterung vorhandener Befehle durch das Disk-Basic (z. B.: PRINT). Diese Adressen
				werden vom Basicinterpreter aus verschiedenen Programmsegmenten als Unterprogramme
				angesprungen. Bei der Initialisierung werden die Zeiger durch einen RET ersetzt
41E5	41E7	16869	16870	(3A 00 2C) Alle folgenden Adressen sind nicht die Adressen selbst, sondern die der Zeiger
40A7		16551		Input/Outputbuffer
40A4		16548		Programmtext: Zeiger auf nächste Zeile (LSB/MSB)/Zeilennummer (LSB/MSB)/Text/00...
				(weitere gleichartige Zeilen) ...0000
40F9		16633		Einfache Variablen: Typ/2. Buchstabe/1. Buchstabe/Wert/...
40FB		16635		Feldvariablen: Typ/2. Buchstabe/1. Buchstabe/Anzahl aller folgenden Bytes
				(LSB/MSB)/Anzahl der Dimensionen/Tiefe jeder Dimension (LSB/MSB)/Werte/...
40FD		16637		Freier Speicher
40E8		16616		Anfangswert des Stacks für Rücksprungadressen
40A0		16544		Anfangswert des Stacks für Programmdaten (z. B. FOR-Schleifen)
40A0+1		16544+1		Anfang des Stringspace
40D6		16598		Letztes freie Byte des Stringspace. Alles, was darüber liegt, ist mit Zeichenketten belegt
40B1		16561		Letztes RAM-Speicherbyte oder der um 2 erniedrigte Eingabewert bei MEM SIZE