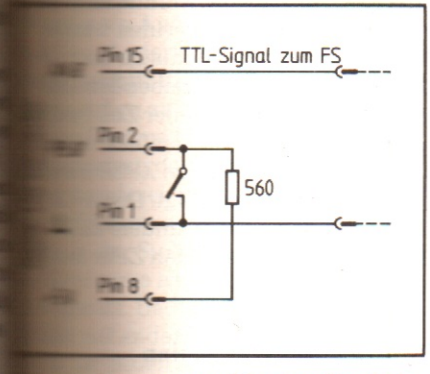
Edmund Möller

Apple-II steuert Fernschreiber

Die ASCII-Drucker sind oft ebenso teuer wie ein kleiner Mikrocomputer; Baudot-Fernschreiber sind hingegen gebraucht recht preiswert erhältlich und stellen trotz ihres eingeschränkten Zeichenvorrates daher eine interessante Alternative zu Matrixdruckern dar. Der folgende Beitrag beschreibt die Ansteuerung eines solchen Fernschreibers mit dem Apple-II.

Nach einigen Ergänzungen und Versuchen aus [1] das hier beschriebene Programm entstanden. Um z. B. Programm-Listings besser aufbewahren zu können, wurde ein Zeilenzähler eingebaut, der nach jeweils 64 Zeilen 10 Leerzeilen einfügt, so daß das Endlospapier in DIN A4 große Abschnitte zerfällt und bequem abgeheftet werden kann.



Hardware zum wahlweisen Zuschalten der Bildschirm-Ausgabe

Um den Drucker unabhängig vom Bildschirmausgabeprogramm betreiben zu können, wurde auch ein Zeichenzähler eingebaut, mit dessen Hilfe die volle Zeilenbreite von hier 64 Zeichen ausgenutzt werden kann. Ein Umschalter erlaubt, das Bildschirmausgabeprogramm umzuschalten. Dann werden maximal 40 Zeichen pro Zeile geschrieben. Auch bei schon existierenden Programmen manchmal nötig, weil der Zähler im Druckprogramm die ggf. vorhandenen TAB-Befehle nicht verarbeitet.

Einige Zeichen des ASCII-Zeichensatzes, die der Fernschreiber nicht besitzt, wurden durch vorhandene Zeichen ersetzt. Das Zeichen * für die Multiplikation wurde durch ein X ersetzt. Die Zeichen < und > werden durch runde Klammern dargestellt. Das ist sicherlich nicht ideal, aber man kann die richtige Bedeutung fast immer aus dem Zusammenhang erkennen. Statt des Semikolons (;) erscheint das Klingelzeichen, und die Anführungszeichen (") werden durch das Auslassungszeichen (') ersetzt.

In der hier beschriebenen Version belegt das Programm den Speicherbereich von $7F00 bis $7FFF und kann auf übliche Weise von der Kassette eingelesen werden. Dies ist für einen 32-KByte-Apple das obere Ende des RAM-Bereichs. Die Sicherung des Druckprogramms erfolgt in Basic durch Eingabe von HIMEM: 32510 im Direktmodus.

Das Programm wird aktiviert durch Belegen der Speicherplätze 36 und 37 in der Zero-Page mit der Anfangsadresse 7F00 oder in Basic durch POKE 54,0:POKE 55,127, was dasselbe bewirkt. Wenn das Programm später im EPROM liegt, kann der Aufruf über PR#... erfolgen. Die Rückkehr zur reinen Bildschirmausgabe erfolgt über PR#0.

Für die Ausgabe des seriellen Baudot-Signals und für das Festlegen des Ausgabemodus werden vorhandene Ports benutzt, die am Game I/O-Connector zur Verfügung stehen.

Die Ausgabe erfolgt über den Annunciator-Output AN0 (Game I/O-Connector Pin 15). Hier steht bereits TTL-Pegel zur Verfügung, mit dem direkt ein Transistor für die Steuerung des Fernschreiber-Linienstroms angesteuert werden kann. Entsprechende Schaltungen wurden in ausreichender Zahl bereits veröffentlicht [2, 3].

Das Kriterium für den Ausgabemodus (Drucker allein oder mit dem Bildschirm) wird über einen Schalter dem Single-Bit-Input PB0 (Game I/O-Connector Pin 2) zugeführt (*Bild 1*).

Ein Listing des Programms ist in *Bild 2* dargestellt. Hier hat sich das Programm selbst aufgelistet!

An zwei Stellen im Programm sind freie Speicherplätze vorhanden, die noch für kleine Ergänzungen verwendet werden können.

```
7F00- 20 4A FF A9 40 85 FE 85
7F08- FF A9 0A 85 FB A9 00 85
7F10- FA A9 1F 85 36 EA EA EA
7F18- EA EA EA EA EA D0 03 20
7F20- 4A FF A5 45 C9 8D D0 09
7F28- 20 6F 7F C6 FE F0 4F D0
7F30- 29 29 3F AA BD C0 7F 85
7F38- FD 29 20 C5 FC F0 0E 85
7F40- FC A8 F0 04 A9 1B D0 02
7F48- A9 1F 20 8F 7F A5 FD 20
7F50- 8F 7F E6 FA A5 FA C5 FF
7F58- B0 CE AD 61 C0 C9 80 B0
7F60- 0A EA EA EA EA 20 3F FF
7F68- 4C F0 FD 20 3F FF 60 A9
7F70- 08 20 8F 7F A9 02 20 8F
7F78- 7F A9 00 85 FA 60 20 6F
7F80- 7F C6 FB D0 F9 A9 40 85
7F88- FE A9 0A 85 FB D0 CB 48
7F90- AD 58 C0 20 BA 7F A0 04
7F98- 68 4A 90 07 48 AD 59 C0
7FA0- 4C A7 7F 48 AD 58 C0 20
7FA8- BA 7F 88 10 EB AD 59 C0
7FB0- 20 B5 7F 68 60 A9 35 20
7FB8- A8 FC A9 48 20 A8 FC 60
7FC0- 00 03 19 0E 09 01 0D 1A
7FC8- 14 06 08 0F 12 1C 0C 18
7FD0- 16 17 0A 05 10 07 1E 13
7FD8- 1D 15 11 2D 00 00 00 00
7FE0- 24 34 25 29 3A 24 31 25
7FE8- 2F 32 1D 31 2C 23 3C 3D
7FF0- 36 37 33 21 2A 30 35 27
7FF8- 26 38 2E 28 2F 3E 32 39
X
```

Bild 2. Hex-Dump des Fernschreiber-Ausgabenprogramms für einen 32-KByte-Apple. Es findet am oberen Speicherende Platz

Literatur

[1] Baudot-Ausgabeprogramm für den 6502. FUNKSCHAU 1979, Heft 1.

[2] *Pietsch, H. J.*: Amateurfunk-Fernschreibtechnik RTTY. RPB 25, Franzis-Verlag, München.

[3] Fernschreiber-Ansteuerung. FUNKSCHAU 1979, Heft 26.

```
COMMAND? C
  *** CHANGE TEXT ROUTINE ***
GLOBAL ODER EINZELN? G/E E
TEXT ALT ?
65

TEXT NEU ?
6500
ALT:
..65 MAINZ 42
NEU
..6500 MAINZ 42
AENDERN J/N/E (E=EXIT)
```

Bild 2. Beispiel für den Umgang mit der Änderungsroutine, die durch Druck auf Taste C angesprungen wird

Um Zeilen mit führenden Leerstellen speichern und lesen zu können, schreibt der Verfasser an den Anfang eines jeden Satzes, der eingerückt wird, ein Steuerzeichen für den Drucker, das nicht gedruckt wird, aber von Basic wie ein normaler Buchstabe behandelt wird. Im Falle des Druckers „Paper Tiger" ist es ein CNTRL-B, das normale Schriftbreite anwählt.

Literatur

[1] Basic-Texteditor. FUNKSCHAU 1980, Heft 1, und 1980, Heft 22.

[2] Schnelle Stichwortsuche beim CBM. FUNKSCHAU 1980, Heft 23.

SSTV-Ausgabe mit dem CBM

Slow-Scan-Television (SSTV) ist eine Methode zur Übertragung von Standbildern mit der Bandbreite eines Sprachsignals. Das abgedruckte Programm liefert an PA 0 des CBM-User-Ports ein vollständiges SSTV-Signal. Das rechteckförmige Signal kann bei geeigneter Abschwächung unmittelbar in den Mikrofon-Eingang eines Senders eingespeist werden, da dort die durch die Rechteckform entstandenen Oberwellen gefiltert werden.

Das Maschinenprogramm zur Erzeugung des SSTV-Signals befindet sich in Zeile 6600 bis 6770. Es wird im zweiten Kassettenpuffer-Bereich abgelegt. Man beschreibt den Bildschirm mit REV-SPACE. Das SSTV-Signal ist sodann ein getreues Abbild des Bildschirminhaltes. Anstelle von REV-SPACE kann jedes andere auf der Tastatur verfügbare ASCII-Zeichen verwendet werden, indem die eingeklammerte Ziffer 160 in Zeile 6630 gegen die entsprechende Zeichencodierung ausgetauscht wird. Der Bildschirmspeicher wird punktweise abgefragt und mit dez. 160, der internen Codierung für REV-SPACE, verglichen. Ist ein Zeichen vorhanden, so wird der Akku mit dez. 40 geladen und 6mal heruntergezählt; das entspricht drei Perioden einer Frequenz von etwa 2250 Hz. Bei unbeschriebenem Bildschirm wird dez. 61 geladen und 4mal heruntergezählt, was zwei Perioden von 1500 Hz ergibt. Nach Ablauf einer Zeile werden sieben Perioden einer Frequenz von 1200 Hz als Zeilensynchronimpuls ausgegeben.

```
100 GOSUB6600:GOSUB200
110 S=0
120 GOSUB180:GOSUB190:SYS866
130 S=S+1:IFS<3THEN120
140 GETQ$:IFQ$="■"THEN110
150 IFQ$="■"THEN170
160 GOTO140
170 GOSUB190:END
180 POKE59467,16:POKE59466,15:POKE59464,51
185 FORI=1TO150:NEXT:RETURN
190 POKE59467,21:POKE59466,16:RETURN
200 FORI=1TO100:PRINT"■ ■ ■ ■ ■ ■ ■ ■ ";
205 NEXT:RETURN
210 REM
220 REM
6600 FORX=866TO999:READS:POKEX,S:NEXT:RETURN
6610 DATA120,160,128,132,2,160,0,132
6620 DATA1,140,240,3,140,241,3,177
6630 DATA1,201,(160),208,18,162,0,142
6640 DATA67,232,160,40,136,208,253,232
6650 DATA224,6,208,243,76,152,3,162
6660 DATA0,142,67,232,160,61,136,208
6670 DATA253,232,224,4,208,243,238,240
6680 DATA3,230,1,165,1,208,10,230
6690 DATA2,165,2,201,132,208,2,88
6700 DATA96,173,240,3,201,40,208,191
6710 DATA162,0,142,67,232,160,80,136
6720 DATA208,253,232,224,14,208,243,169
6730 DATA0,141,240,3,238,241,3,173
6740 DATA241,3,201,5,208,8,169,0
6750 DATA141,241,3,76,113,3,56,165
6760 DATA1,233,40,133,1,165,2,233
6770 DATA0,133,2,76,113,3
```

Mit Hilfe des VIA-Bausteins 6522 im CBM läßt sich der Bildschirminhalt am User Port als fertiges Nf-Signal ausgeben, um einen Sender in der Betriebsart „Slow-Scan-Television" zu betreiben

Jede Zeile wird 5mal wiederholt, so daß insgesamt 125 SSTV-Zeilen entstehen. Nach Durchlauf des Bildes kehrt das Programm zum Basic zurück.

Der Bildimpuls wird hier mit entsprechenden POKE-Befehlen an CB 2 des User-Ports erzeugt. Dieses Verfahren hat sich im praktischen Betrieb bewährt, da die Ausgabe eines Tones an CB 2 mit Hilfe des VIA erfolgt und der Rechner unterdessen weiterarbeiten kann. So kann man den 1200-Hz-Ton stehen lassen, während der Rechner ein neues Bild zeichnet. Die Ausgänge PA 0 und CB 2 müssen bei diesem Verfahren jedoch über ein Oder-Gatter verknüpft werden.

Die Zeilen 100 bis 205 enthalten ein Demonstrationsprogramm, mit dessen Hilfe senkrechte Streifen auf den Bildschirm geschrieben werden. Das Bild wird danach 3mal ausgegeben. Anschließend kann die Ausgabe mit CRSR „↑" wiederholt oder mit CRSR „←" beendet werden.

Uwe Mach

Literatur

SSTV mit dem AIM-65. FUNKSCHAU 19[illegible] Heft 14.

Herwig Feichtinger

AIM liest CBM-Kassetten

Vielleicht haben Sie sich als AIM-65- oder PC-100-Besitzer manchmal darüber geärgert, daß ein PET-, CBM- oder VC-20-Inhaber ein intelligentes Basic-Programm auf Kassette besaß und kein anderer Datentransfer zu Ihrem Computer möglich war, als mühsam das Listing von Hand abzutippen. Damit ist jetzt Schluß.

Das im *Bild* aufgelistete 6502-Programm ermöglicht es, Kassetten, die von irgendeinem Commodore-Computer mit einem Basic-Programm beschrieben wurden, in den AIM einzulesen. Dabei findet gleichzeitig die nötige Anpassung der Basic-Zeiger und der byteweisen internen Befehlsdarstellung (Tokens) statt.

Basic-Tokens von AIM und CBM

Hex	AIM	CBM	Hex	AIM	CBM	Hex	AIM	CBM
80	END	END	99	LIST	PRINT	B2	FRE	=
81	FOR	FOR	9A	CLEAR	CONT	B3	POS	<
82	NEXT	NEXT	9B	GET	LIST	B4	SQR	SGN
83	DATA	DATA	9C	NEW	CLR	B5	RND	INT
84	INPUT	INPUT#	9D	TAB(	CMD	B6	LOG	ABS
85	DIM	INPUT	9E	TO	SYS	B7	EXP	USR
86	READ	DIM	9F	FN	OPEN	B8	COS	FRE
87	LET	READ	A0	SPC(	CLOSE	B9	SIN	POS
88	GOTO	LET	A1	THEN	GET	BA	TAN	SQR
89	RUN	GOTO	A2	NOT	NEW	BB	ATN	RND
8A	IF	RUN	A3	STEP	TAB(	BC	PEEK	LOG
8B	RESTORE	IF	A4	+	TO	BD	LEN	EXP
8C	GOSUB	RESTORE	A5	–	FN	BE	STR$	COS
8D	RETURN	GOSUB	A6	*	SPC(	BF	VAL	SIN
8E	REM	RETURN	A7	/	THEN	C0	ASC	TAN
8F	STOP	REM	A8	↑	NOT	C1	CHR$	ATN
90	ON	STOP	A9	AND	STEP	C2	LEFT$	PEEK
91	NULL	ON	AA	OR	+	C3	RIGHT$	LEN
92	WAIT	WAIT	AB	>	–	C4	MID$	STR$
93	LOAD	LOAD	AC	=	*	C5		VAL
94	SAVE	SAVE	AD	<	/	C6		ASC
95	DEF	VERIFY	AE	SGN	↑	C7		CHR$
96	POKE	DEF	AF	INT	AND	C8		LEFT$
97	PRINT	POKE	B0	ABS	OR	C9		RIGHT$
98	CONT	PRINT#	B1	USR	>	CA		MID$

Die Bedienung erfolgt so: Man startet das Maschinenprogramm mit Taste F1 vom Monitor aus, sobald man den Kassettenrecorder gestartet hat. Wenn ein Programm gefunden wurde, hält der Recorder automatisch an (die Fernbedienungsleitung muß deshalb angeschlossen sein), und auf dem AIM-Display erscheint der Programmname sowie die Endadresse, die das Programm im AIM-RAM einnehmen wird.

Drückt man jetzt N oder eine andere Taste außer Y, so läuft der Recorder bis zum nächsten Programm, ohne etwas zu laden. Drückt man aber Y, so lädt der AIM das Programm zunächst ins RAM, führt dann alle Anpassungen aus und druckt – wenn der Drucker eingeschaltet ist – eine Liste der gelesenen Zeilennummern. Findet er ein Token, das zu einem CBM-Befehl gehört, den es im AIM-Basic nicht gibt, so druckt er dieses Token hexadezimal neben die Zeilennummer. Im Programm selbst ersetzt er den CBM-Befehl dann durch eine REM-Anweisung.

Sobald das Programm geladen ist, befindet man sich automatisch im Basic-Interpreter – er wurde vom Ladeprogramm bereits mit initialisiert, wobei das Basic-Programm ab hex 0461 abgespeichert wurde.

Nach dem Programmladen wird auch eine Error-Meldung ausgedruckt, die die Zahl der aufgetretenen Parityfehler angibt. Es ist möglich, daß ein oder zwei Errors gemeldet werden, ohne daß dies einen Einfluß auf die Fehlerfreiheit des Programms hat. Außerdem fällt auf, daß der AIM mit dem Laden schon bei der Hälfte der Gesamtprogrammlänge fertig ist: Commodore-Computer zeichnen nämlich jedes Programm zweimal nacheinander auf Kassette auf.

Da beim CBM Steuerzeichen-Codes in Anführungszeichen (z. B. hinter PRINT) verwendet werden, die AIM-Tokens sind, erscheinen manchmal merkwürdige Dinge in Gänsefüßchen. Daran braucht man sich nicht zu stören, denn solche CBM-Steuerzeichen muß man ja ohnehin für den AIM ändern – wie etwa Bildschirm löschen, Cursorbewegungen usw.

In den meisten Fällen wird das geladene CBM-Programm nicht ohne Änderungen auf dem AIM laufen, weil das CBM-Basic einen umfangreicheren Befehlssatz als der AIM bietet und, wie erwähnt, auch Anpassungen bei manchen Steuerzeichen hinter PRINT nötig sein können. Trotzdem spart man sich mit der Kassetten-Leseroutine eine Menge Tipparbeit beim Programmtransfer vom CBM auf den AIM.

Das Programm stützt sich auf leider fehlerhaft in [1] und [2] abgedruckte bzw. nicht funktionsfähige Vorläuferversionen. Das Laden von VC-20-Programmen erfordert eine Korrektur des Basic-Anfangszeigers und wurde noch nicht erprobt.

Literatur

[1] *Kemp, David:* Reading PET tapes. Micro 1979, June.

[2] *Rehnke, Eric; Reardon, Mark:* Read PET tapes with your AIM. Compute 1980, issue 3.

```
)=0200 A9 37 8D 02 A8 A9 EE 8D 0C A8 20 94 03 AD 6A 04
) 0210 C9 01 D0 F6 20 13 EA 20 13 EA 20 89 03 A2 00 BD
) 0220 6F 04 C9 20 F0 06 20 7A E9 E8 D0 F3 20 3E E8 18
) 0230 AD 6D 04 69 62 8D 6D 04 AD 6E 04 69 00 20 46 EA
) 0240 AD 6D 04 20 46 EA 20 3E E8 20 73 E9 20 89 03 C9
) 0250 59 F0 05 8D 6A 04 D0 B2 A9 08 85 F5 A2 FF A0 FF
) 0260 88 D0 FD CA D0 F8 C6 F5 D0 F2 20 94 03 C8 8C 69
) 0270 04 20 13 EA C6 F4 A5 F4 20 46 EA 20 3E E8 B9 08
) 0280 04 48 20 7A E9 C8 68 10 F5 A2 6A 86 F6 86 73 A9
) 0290 04 85 F7 85 74 20 13 EA D0 0F 20 22 03 A2 00 C8
) 02A0 A5 F6 81 FA AA A5 F7 91 FA 85 FB 86 FA 20 22 03
) 02B0 D0 03 4C 2D 03 20 22 03 85 F8 20 22 03 85 F9 A2
) 02C0 00 A5 F8 38 F9 0E 04 85 F8 A5 F9 C8 F9 0E 04 90
) 02D0 07 85 F9 E8 88 4C C1 02 88 A5 F8 79 0E 04 85 F8
) 02E0 8A 09 30 20 7A E9 C8 C8 C0 08 D0 D3 A5 F8 09 30
) 02F0 20 7A E9 20 3E E8 20 22 03 F0 9A 48 C9 22 D0 0C
) 0300 68 20 22 03 F0 8F C9 22 D0 F7 F0 EA 68 10 E7 AA
) 0310 BD 96 03 30 09 8A 20 46 EA 20 3E E8 A9 8E 91 F6
) 0320 D0 D4 E6 F6 D0 02 E6 F7 A0 00 B1 F6 60 20 22 03
) 0330 A6 F6 A5 F7 86 75 85 76 A2 00 A9 10 86 7F 85 80
) 0340 A2 FE 9A A9 00 48 85 01 85 10 85 60 85 B0 A2 1C
) 0350 BD 85 CE 95 BE CA D0 F8 A9 03 85 9B A9 61 85 5E
) 0360 A9 B9 85 02 A9 14 85 12 A9 0A 85 13 A9 FF 85 82
) 0370 A9 4C 85 00 85 03 85 9C 85 BB A2 87 A9 BF 86 BC
) 0380 85 BD 86 04 85 05 4C 7F B2 48 AD 00 A8 49 30 8D
) 0390 00 A8 68 60 A9 04 85 F1 A9 61 85 F0 A9 10 85 F4
) 03A0 20 C5 03 30 03 4C 94 03 C6 F4 D0 F4 20 C5 03 30
) 03B0 FB B0 04 A9 FE E6 F4 91 F0 E6 F0 D0 02 E6 F1 20
) 03C0 C5 03 10 ED 60 A0 11 20 F9 03 E0 3C B0 08 E0 28
) 03D0 B0 F3 88 10 F2 60 A0 09 84 F3 20 F9 03 20 EE 03
) 03E0 90 02 E6 F3 6A 88 D0 F5 2A 49 FF 46 F3 60 20 F9
) 03F0 03 86 F2 20 F9 03 E4 F2 60 A2 00 E8 2C 00 A8 30
) 0400 FA E8 2C 00 A8 10 FA 60 45 52 52 4F 52 D3 10 27
) 0410 E8 03 64 00 0A 00 80 81 82 83 84 84 85 86 87 88
) 0420 89 8A 8B 8C 8D 8E 8F 90 92 93 94 00 95 96 97 97
) 0430 98 99 9A 00 00 00 00 9B 9C 9D 9E 9F A0 A1 A2 A3
) 0440 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3
) 0450 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF C0 C1 C2 C3
) 0460 C4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
) 010C 4C 00 02
```

Mit diesem Hilfsprogramm, hier als Hex-Dump abgedruckt, kann der AIM-65 (alias PC-100) Basic-Kassetten der Computer CBM, PET und VC-20 lesen. Die Programmprüfsumme (vgl. mc 1982, Heft 2, S. 36) ist 312F für FROM = 200 TO = 461

Adressen im Programm

00F0, 00F1	Adressenzähler
00F2-00FB	vom Programm benützt
0200	Startadresse, Header lesen
021F	Filename ausgeben
022F	Endadresse ausgeben
0258	Verzögerung bis zum 2. Blockanfang
026A	Programm einlesen (ab 0461)
0274	Error-Anzahl ausgeben
0295	Zeilenzeiger umrechnen und Zeilennummern anzeigen
030C	PET-Tokens in AIM-Tokens umrechnen
032D	Basic-Interpreter initialisieren

Unterprogramme:

0389	Recorder ein/ausschalten
0394	Kompletten Block von der Kassette lesen
03C5	Ein Byte einlesen; N = 1, wenn nur Shorts, C = 0 bei Parity-Error
03EE	Ein Bit einlesen
03F9	Schwingungs-Periodendauer in X liefern
0408	Tabelle mit Zehnerpotenzen; „ERRORS" in ASCII
0416	Token-Umrechnungstabelle

AIM-65-Systemadressen:

B27F	Basic-Warmstartadresse
A800-A80C	Kassetten-Eingang und Recordersteuerung
EA46	Byte als zwei ASCII-Hex-Zeichen ausgeben
E97A	ASCII-Zeichen ausgeben
E83E	Leerraum ausgeben
EA13	Display löschen (CRLF)
E973	Ein ASCII-Zeichen von der Tastatur holen
0000-00DE	Von Basic verwendete Zero-Page-Zellen
010C	Sprungvektor für F1-Taste

Datenformat der CBM-Kassettenaufzeichnung

Header-Block

10 Sekunden Shorts als Vorspann
9 Synchronisations-Bytes (hex 89...81)
1 Byte als Flag (Programmfile = 01, Datenfile = 04)
2 Byte Programm-Anfangsadresse (Low/High Byte)
2 Byte Programm-Endadresse (Low/High Byte)
Max. 128 ASCII-Zeichen als Programmname
Rest des Blocks (Blocklänge 192 Zeichen): hex 20
1 Byte Prüfsumme
Ca. 50 ms Shorts

} wird einmal komplett wiederholt

Programm-Block

2 Sekunden Shorts als Vorspann
9 Synchronisations-Bytes (hex 89...81)
n Datenbytes von Anfangs- bis Endadresse
1 Byte Prüfsumme
Ca. 50 ms Shorts

} wird einmal komplett wiederholt

Jedes Byte wird beginnend mit dem niederwertigsten Bit übertragen und mit einem Parity-Bit auf 9 Bit ergänzt. Die Frequenzfolge „Byte"-„Long" kennzeichnet den Beginn eines Bytes. Ein Einsbit wird als Long-Short, ein Nullbit als Short-Long übertragen. Short ist dabei eine (nur eine) 2,78-kHz-Schwingung, Long eine mit 1,96 kHz und „Byte" eine mit 2,47 kHz. Ein Byte dauert dadurch knapp über 8 ms. Basic-Programme werden nicht in ASCII-Form, sondern mit Tokens als reiner Speicherauszug aufgezeichnet.

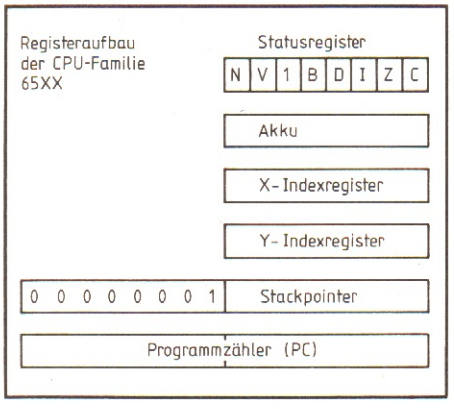
Herwig Feichtinger

65XX-Befehlssatz

Die Prozessoren der Familie 65XX haben alle den gleichen Befehlssatz, der hier übersichtlich zusammengestellt ist. Zu dieser CPU-Familie gehören u. a. die Typen 6502 (AIM-65, PC-100, KIM-1, CBM, PET, Apple-II/III, Pearcom u. a. Computer) und 6504 (im Einplatinencomputer EMUF).

Bei den Befehlen mit mehreren Adressierungsarten ist die Beeinflussung des Statusregisters mit angegeben; bei den übrigen Befehlen wurde sie der Übersichtlichkeit halber weggelassen (PLA wie LDA, PHA wie STA, DEX wie DEC usw.). Die Interrupt-Vektoren haben folgende festverdrahtete Adressen:

FFFA	NMI Low (nicht bei 6504)
FFFB	NMI High (nicht bei 6504)
FFFC	RES Low
FFFD	RES High
FFFE	IRQ Low
FFFF	IRQ High



Register des VIA-Bausteins 6522

Adresse	*Register*
XXX0	Datenregister Port B
XXX1	Datenregister Port A, mit Handshake
XXX2	Port-B-Datenrichtungsregister
XXX3	Port-A-Datenrichtungsregister
XXX4	Timer 1 Low lesen/schreiben
XXX5	Timer 1 High lesen/schreiben/starten
XXX6	Timer-1-Latch Low
XXX7	Timer-1-Latch High
XXX8	Timer 2 Low lesen/schreiben, IRQ rücksetzen
XXX9	Timer 2 High lesen/schreiben, IRQ rücksetzen
XXXA	Schieberegister
XXXB	ACR (Steuer-Hilfsregister)
XXXC	PCR (Peripherie-Steuerregister)
XXXD	IFR (Interrupt-Flagregister)
XXXE	IER (Interrupt-Enable-Register)
XXXF	Port-A-Datenregister (ohne Handshake)

Register des RIOT-Bausteins 6532

Adresse	*Register*
XXX0	Datenregister Port A
XXX1	Datenrichtungsregister Port A
XXX2	Datenregister Port B
XXX3	Datenrichtungsregister Port B
XXX6	Timer lesen
XXX7	Interrupt-Flags lesen/löschen
XX14-17	Timer schreiben, 1/8/64/1024 µs
XX1C-1F	Wie 14-17, jedoch mit Interrupt
XX04	PA-7-Interrupt bei H-L-Flanke programmieren
XX05	PA-7-Interrupt bei L-H-Flanke programmieren
XX06	PA-7-Interrupt sperren

Befehle m. mehreren Adressierungsarten

Mn.	Im	Ab	ZP	Ac	X)	)Y	ZX	ZY	AX	AY	NZCV
ADC	69	6D	65	--	61	71	75	--	7D	79	xxxx
AND	29	2D	25	--	21	31	35	--	3D	39	xx--
ASL	--	0E	06	0A	--	--	16	--	1E	--	xxx-
BIT	--	2C	24	--	--	--	--	--	--	--	xx-x
CMP	C9	CD	C5	--	C1	D1	D5	--	DD	D9	xxx-
CPX	E0	EC	E4	--	--	--	--	--	--	--	xxx-
CPY	C0	CC	C4	--	--	--	--	--	--	--	xxx-
DEC	--	CE	C6	--	--	--	D6	--	DE	--	xx--
EOR	49	4D	45	--	41	51	55	--	5D	59	xx--
INC	--	EE	E6	--	--	--	F6	--	FE	--	xx--
LDA	A9	AD	A5	--	A1	B1	B5	--	BD	B9	xx--
LDX	A2	AE	A6	--	--	--	--	B6	--	BE	xx--
LDY	A0	AC	A4	--	--	--	B4	--	BC	--	xx--
LSR	--	4E	46	4A	--	--	56	--	5E	--	xxx-
ORA	09	0D	05	--	01	11	15	--	1D	19	xx--
ROL	--	2E	26	2A	--	--	36	--	3E	--	xxx-
ROR	--	6E	66	6A	--	--	76	--	7E	--	xxx-
SBC	E9	ED	E5	--	E1	F1	F5	--	FD	F9	xxxx
STA	--	8D	85	--	81	91	95	--	9D	99	----
STX	--	8E	86	--	--	--	--	96	--	--	----
STY	--	8C	84	--	--	--	94	--	--	--	----

Befehle mit nur einer Adressierungsart

BCC 90	BCS B0	BEQ F0	BMI 30
BNE D0	BPL 10	BRK 00	BVC 50
BVS 70	CLC 18	CLD D8	CLI 58
CLV B8	DEX CA	DEY 88	INX E8
INY C8	JMI 6C	JMP 4C	JSR 20
NOP EA	PHA 48	PHP 08	PLA 68
PLP 28	RTI 40	RTS 60	SEC 38
SED F8	SEI 78	TAX AA	TAY A8
TSX BA	TXA 8A	TXS 9A	TYA 98

Adressierungsarten

Im=Immediate; Ab=Absolute; ZP=Zeropage; Ac=Accu; X)=x-indiziert indirekt;)Y=indirekt y-indiziert; ZX=Zeropage x-indiziert; ZY=Zeropage y-indiziert; AX=absolut x-indiziert; AY=absolut y-indiziert.

Flags im Statusregister

N=Negativ (Bit 7); Z=Zero; C=Carry; I=Interrupt-Disable; D=Decimal Mode; V=Overflow (Bit 6).

```
C32A03    01290 OUTCH   JP 032AH        ;TRS-80 AUSGABEROUTINE
3E20      01300 SPACE   LD A,20H        ;DRUCKT C MAL SPACE
CDDC4A    01310         CALL OUTCH
0D        01320         DEC C           ;C MAL WIEDERHOLEN
20F8      01330         JR NZ,SPACE
C9        01340         RET
          01350         ; UP DRUCKT PRUEFSUMME
3A9A43    01360 PRINT   LD A,(ARG3H)    ;SUMMENAUSDRUCK ERWUENSCHT?
A7        01370         AND A
C8        01380         RET Z           ;NEIN
E5        01390 PRINT2  PUSH HL
21304B    01400         LD HL,TEXT      ;'PRUEFSUMME=' DRUCKEN
7E        01410 PRINT1  LD A,(HL)
CDDC4A    01420         CALL OUTCH
FE3D      01430         CP '='          ;ENDE DES TEXTES?
23        01440         INC HL
20F7      01450         JR NZ,PRINT1    ;NEIN
DDE5      01460         PUSH IX         ;IX NACH HL
E1        01470         POP HL
CDC44A    01480         CALL OUTHL      ;IX HEXAD. AUSDRUCKEN
3E0D      01490         LD A,0DH        ;CR
CDDC4A    01500         CALL OUTCH
DD210000  01510         LD IX,0         ;SUMME=0
FD210100  01520         LD IY,1         ;BLOCKLAENGE=256
E1        01530         POP HL
C9        01540         RET
0009      01550         DEFW 900H       ;S ANF END
53        01560         DEFB 'S'        ;BESTIMMT SUMME
0D        01570         DEFB 0DH        ;VON ANF BIS END
2A9543    01580         LD HL,(ARG1)
ED5B9743  01590         LD DE,(ARG2)
DD210000  01600         LD IX,0         ;SUMME=0
0600      01610         LD B,0
4E        01620 LOOP    LD C,(HL)
DD09      01630         ADD IX,BC       ;AUFADDIEREN
A7        01640         AND A
ED52      01650         SBC HL,DE       ;ENDE ERREICHT?
19        01660         ADD HL,DE
23        01670         INC HL
20F6      01680         JR NZ,LOOP      ;NEIN
CDED4A    01690         CALL PRINT2
C9        01700         RET
00        01710         NOP             ;PLATZ FUER SPRUNGBEFEHL
00        01720         NOP
50        01730 TEXT    DEFM 'PRUEFSUMME='
52        4B32 55       4B33 45       4B34 46       4B35 53       4B3
     4B37 4D       4B38 4D       4B39 45       4B3A 3D       0000
01740          END
TOTAL ERRORS
```

Der Befehl POP für CBM 3001

Das vorliegende Programm erweitert den Basic-Befehlsvorrat der CBM-Rechner der Serie 3001 um den Befehl POP. Wie bereits im Funkschau-Sonderheft 33, „Mikrocomputer-Anwendungen", von H. G. Joepgen im Artikel „Gewaltsamer Rücksprung aus Unterprogrammen" ausführlich behandelt, kann mit dem Befehl POP aus Unterprogrammen auch über mehrere Unterprogrammebenen hinweg herausgesprungen werden.

Das in *Bild 1* als Hex-Dump aufgeführte Programm wird mit SYS 826 aktiviert. Danach kann der Befehl in der Form @ POP benutzt werden, allerdings nicht im Direktmodus. Wird das Zeichen @ weggelassen, so erfolgt die Meldung SYNTAX ERROR. Falls einmal der Befehl POP ohne vorheriges GOSUB ausgeführt wird, gibt der Rechner POP WITHOUT GOSUB ERROR aus.

Ein kleines Demonstrationsprogramm zeigt *Bild 2*, wenngleich der Vorteil des Befehls POP eigentlich erst bei stärker geschachtelten Programmen zur Geltung kommt.

Eine Einschränkung gilt allerdings für die Anwendung der Anweisung POP. Die Benutzung nach einer IF...THEN Anweisung ist nicht möglich.

Jürgen Bonfert

...chiedene Parameter von Ein-/Ausga-...routinen stehen) angebundenen ...eibmaschinenroutine und können ...alerweise entfallen.

...in Bild 2 abgedruckte Quellpro-...mm wurde mit dem Programmpaket ...TASM (Editor-Assembler) von Radio ...bearbeitet und ausgedruckt.

...die reichliche Kommentierung ...das Verständnis dieses relativ ein-...Programmes auch dem Anfänger ...Maschinenprogrammierung keine ...wierigkeiten bereiten. *Bild 1* zeigt ...Beispieldurchläufe der beiden Be-... Unter anderem hat sich das Druck-...mm selbst ausgedruckt.

...ur

...ichtinger, H.: Prüfsumme erkennt fal-...Eingabe. FUNKSCHAU 1981, H. 2, ...

...ie, H.: ZETBUG – ein komfortabler ...-Monitor. FUNKSCHAU 1980, H. 11, ...101...104.


```
033A A9 4C 85 79 A9 47 85 7A
0342 A9 03 85 7B 60 C9 40 F0
034A 07 C9 3A B0 F7 4C 7D 00
0352 A4 78 C0 02 F0 F3 A2 03
035A 20 70 00 DD A0 03 D0 25
0362 CA D0 F5 A9 FF 85 47 20
036A AA C2 9A C9 0D F0 19 20
0372 D0 FD 20 42 CA A2 04 BD
037A A0 03 20 D2 FF CA D0 F7
0382 A2 1C 4C 6A C3 4C 03 CE
038A 68 68 68 68 68 20 70 00
0392 20 0E C8 98 18 65 77 85
039A 77 90 02 E6 78 60 00 50
03A2 4F 50 3F 00 00 00 00 00
```

Bild 1. Das Programm als Hex-Dump

```
10 REM DEMONSTRATIONSPROGRAMM
20 REM   ZUM BEFEHL '@POP'
30 REM
40 REM (C) 10.11.1980 BY JUERGEN BONFERT
50 REM
60 REM VERSION 1: OHNE POP
70 REM
80 PRINT"BITTE NUR ZIFFERN EINGEBEN"
90 INPUTA$
100 GOSUB200
110 IFF=1THENPRINT"FEHLER":GOTO80
120 STOP
200 GOSUB300
210 IFF=1THENRETURN
220 PRINT"OK.":RETURN
300 F=0:FORN=1TOLEN(A$)
310 A=ASC(MID$(A$,N,1)):
    IFA<48ORA>57THENF=1:RETURN
320 NEXT:RETURN
400 REM
410 REM VERSION 2: MIT POP
420 REM
430 PRINT"BITTE NUR ZIFFERN EINGEBEN"
440 INPUTA$
450 GOSUB500
460 PRINT"OK.":STOP
500 GOSUB600
520 RETURN
600 FORN=1TOLEN(A$)
610 A=ASC(MID$(A$,N,1)):IFA<48ORA>57THEN630
620 NEXT:RETURN
630 @POP:@POP:PRINT"FEHLER":GOTO430
READY.
```

Bild 2. Dieses Demonstrationsprogramm zeigt die Vorteile von POP

Alfred Lenk

Display für den User-Port des CBM

Der Selbstbau eines Interface, das am User-Port des CBM arbeiten kann, ist schon mit einfachen Mitteln möglich, so daß auch „Softwareleute" keine Hemmungen haben sollten, sich des User-Port zu bedienen. Dabei erweist sich eine optische Beobachtungsmöglichkeit des Datenflusses als wertvolle Hilfe.

Der ersten Eingebung folgend wird man daran denken, die Datenleitungen über Treiber und LEDs anzuzapfen. Das funktioniert zwar sehr gut, aber es ist nicht so einfach möglich, die Richtung des Datenflusses anzuzeigen. Wenn man bereit ist, einen kleinen Teil der Bildschirmfläche des CBM zu opfern, kann man ein Schnittstellendisplay auch per Programm erzeugen.

Bild 1 zeigt das komplette Listing. Es wurde mit Hilfe eines käuflichen Übersetzers erstellt und gilt für das Betriebssystem des CBM. Die Assembler-Kürzel sind für Anwender interessant, die das Programm nicht im Bereich des zweiten Kassettenbuffers speichern wollen. Der erste Programmteil verändert in bekannter Weise den IRQ-Vektor, so daß er auf den eigentlichen Anfang des Displayprogramms zeigt ($0349). Der absolute Sprungbefehl in $03B6 sorgt für ein ordnungsgemäßes Weiterarbeiten des Betriebssystems. Auf diese Weise wird das Programm 60mal pro Sekunde abgearbeitet und die Anzeige auf dem Bildschirm folgt, für das Auge trägheitslos, den Ereignissen auf dem User-Port. Wer das Display in ein Basic-Programm einarbeiten will, kann das Programm in *Bild 2* verwenden, das auch zeigt, wie eine Darstellung auf dem Bildschirm aussehen kann: Die Bits 0...3 sind als Ausgang programmiert und führen alle High-Pegel. Die Bits 4...7 sind als Eingang programmiert und zeigen eine duale Neun in positiver Logik. Das CA1-Flag-Bit reagiert auf negative Flanken

```
033A            1 *=826
033A 78         2 SEI
033B A590       3 LDA$90
033D 4967       4 EOR#$67
033F 8590       5 STA$90
0341 A591       6 LDA$91
0343 49E5       7 EOR#$E5
0345 8591       8 STA$91
0347 58         9 CLI
0348 60        10 RTS
0349 18        11 CLC
034A A000      12 LDY#0
034C A207      13 LDX#7
034E A901      14 LDA#1
0350 8DE983    15 STA 33769
0353 ADE983    16 M3 LDA 33769
0356 2D43E8    17 AND 59459
0359 F006      18 BEQ M1
035B 98        19 TYA
035C 69B0      20 ADC#176
035E 4C6403    21 JMP M2
0361 98        22 M1 TYA
0362 6930      23 ADC#48
0364 9DB583    24 M2 STA 33717,X
0367 ADE983    25 LDA 33769
036A 2D4FE8    26 AND 59471
036D F005      27 BEQ M4
036F A951      28 LDA#81
0371 4C7603    29 JMP M5
0374 A920      30 M4 LDA#32
0376 9DDD83    31 M5 STA 33757,X
0379 C8        32 INY
037A CA        33 DEX
037B 0EE983    34 ASL 33769
037E 90D3      35 BCC M3
0380 A902      36 LDA#2
0382 8DBF83    37 STA 33727
0385 A901      38 LDA#1
0387 8DBE83    39 STA 33726
038A 2D4CE8    40 AND 59468
038D F005      41 BEQ M6
038F A010      42 LDY#16
0391 4C9603    43 JMP M7
0394 A00E      44 M6 LDY#14
0396 A902      45 M7 LDA#2
0398 2D4DE8    46 AND 59469
039B F004      47 BEQ M8
039D 98        48 TYA
039E 697F      49 ADC#127
03A0 A8        50 TAY
03A1 98        51 M8 TYA
03A2 8DE683    52 STA 33766
03A5 A920      53 LDA#32
03A7 2D4CE8    54 AND 59468
03AA F005      55 BEQ M9
03AC A951      56 LDA#81
03AE 4CB303    57 JMP M10
03B1 A920      58 M9 LDA#32
03B3 8DE783    59 M10 STA 33767
03B6 4C2EE6    60 JMP$E62E
03B9           61 .ENDE
```

Bild 1. Disassembler-Listing des Programms zur Beobachtung des User-Port

Tabelle: Programmierung des User-Port in Basic

1. POKE 59459,0	PA0...7 sind als Eingänge geschaltet (Zustand bei Kaltstart)
2. POKE 59459,255	Wie 1., aber als Ausgang.
3. X=PEEK(59471)	Lädt den momentanen Zustand des User-Port in die Variable X.
4. POKE 59471,X	Schreibt den Wert X in dualer Form auf die Bits, die als Ausgang programmiert sind.
5. POKE 59468, PEEK(59468) AND254	Die Eingangsleitung CA1 setzt das CA1-Flag-Bit mit einer negativen Schaltflanke (Zustand bei Kaltstart)
6. POKE 59468, PEEK(59468)OR1	Wie 5., aber positive Schaltflanke
7. WAIT 59469,2	Stoppt die Programmausführung, bis das CA1-Flag-Bit gesetzt ist.
8. X=PEEK(59457)	Wie 3., es wird zusätzlich das CA1-Flag-Bit gelöscht
9. POKE 59457,X	Wie 4., es wird zusätzlich das CA1-Flag-Bit gelöscht
10. POKE 59468, PEEK(59468)OR224	Setzt die CB2-Leitung auf High-Pegel.
11. POKE 59468,(PEEK (59468)AND31 OR192	Setzt die CB2-Leitung auf Low-Pegel.

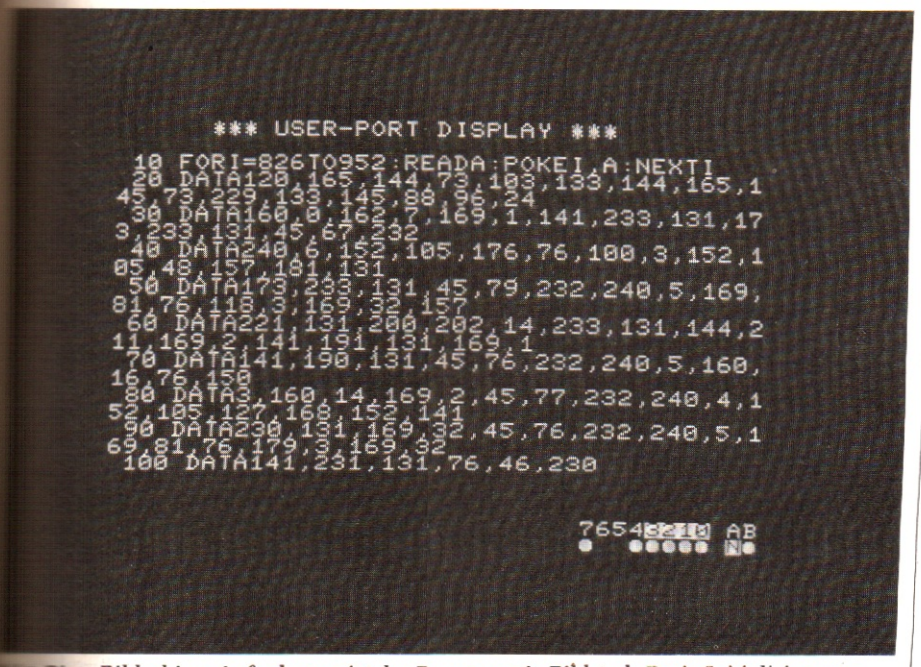


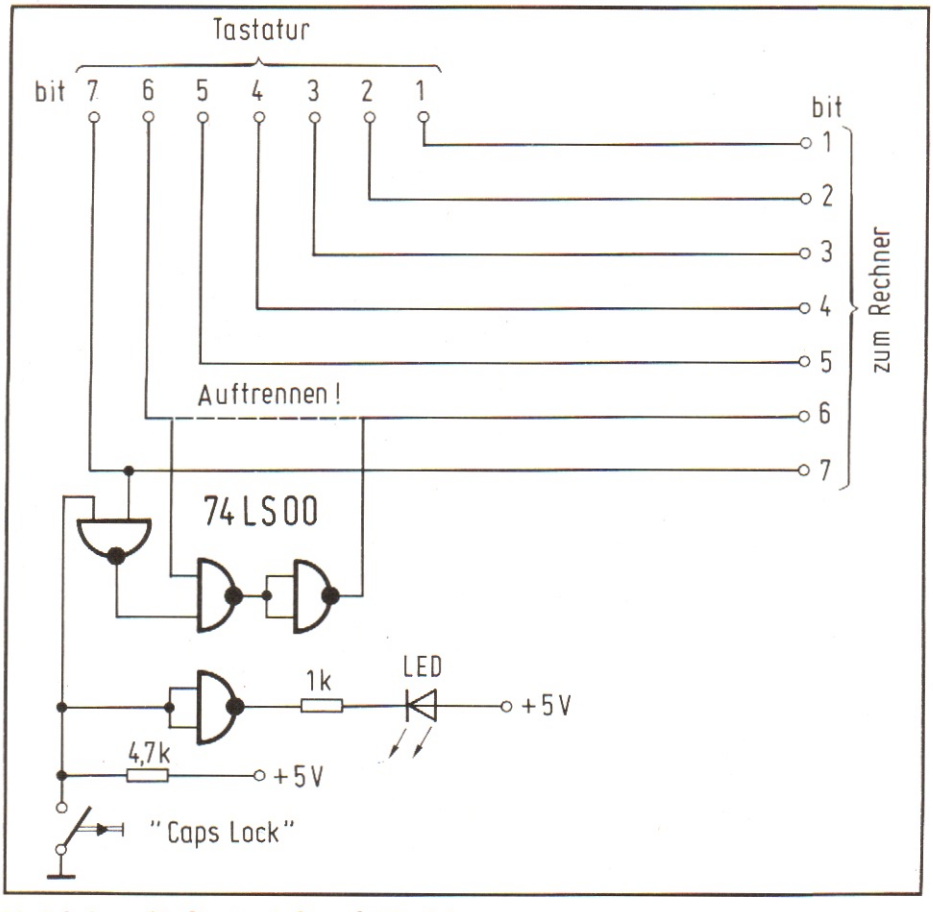
Bild 2. Diese Bildschirm-Aufnahme zeigt das Programm in Bild 1 als Basic-Initialisierungsroutine und rechts unten den Zustand des User-Port beim CBM-3000

und ist gesetzt. Die CB2-Leitung führt ebenfalls High-Pegel. Diese oder eine ähnliche Anzeige des User-Port kann durch SYS(826) auf den Bildschirm gebracht werden, ist dann permanent vorhanden und durch erneutes SYS(826) wieder außer Betrieb zu setzen (empfehlenswert bei Kassettenoperationen). Das Programm springt nur eigene und Bildschirmadressen an und ist leicht zu verschieben. Benutzer des PET-Betriebssystems müssen die Adresse des IRQ-Vektors und den letzten Jump ändern, wobei das Programm vier Bytes länger wird. Die Änderungen kann man dem Franzis-Sonderheft Nr. 33, „Mikrocomputer-Anwendungen", entnehmen. Der Ausgangsbaustein für den User-Port stellte sich bei Schaltungsfehlern als sehr robust heraus, und so steht eigenen Interface-Experimenten nur noch die Beschaffung einer passenden direkten Steckverbindung im Wege. Für die Programmierung des User-Port in Basic gibt die *Tabelle* noch eine Starthilfe.

„Capitals-Lock"-Taste

„Capitals-Lock" wird bei vorneh- Tastaturen ein Schalter bezeichnet, iem die Kleinbuchstaben der Tasta- Großbuchstaben (Capitals) umge- elt werden. So ein Schalter leistet Dienste, wenn ein Programm nur uchstaben akzeptiert und man immer einen Finger auf der Shift- haben möchte. So verlangen sogar Basic-Interpreter die Eingabe der mme in Großbuchstaben. liegende Schaltung benötigt nur nziges TTL-IC, um die gewünschte ion zu realisieren. Aufgabe der ung ist es, die ASCII-Zeichen von dezimal 60 bis 7F in die Zeichen bis 5F zu wandeln. Das kann werden, indem Bit 6 der Tasta- auf Null gesetzt wird, wenn gleich 1 ist. Schaltung (*Bild*) wird zwischen den ung der Tastaturplatine und die ung zum Terminal/Mikrocomputer Die Bauteile lassen sich sicher im Tastaturgehäuse unterbringen. die Belastung der Tastaturelektronik ung wie möglich zu halten, wird S-Baustein anstelle von Standard- verwendet. Das übriggebliebene Gatter wird zur Anzeige des Zu- „Caps Lock" verwendet.

Jürgen Plate

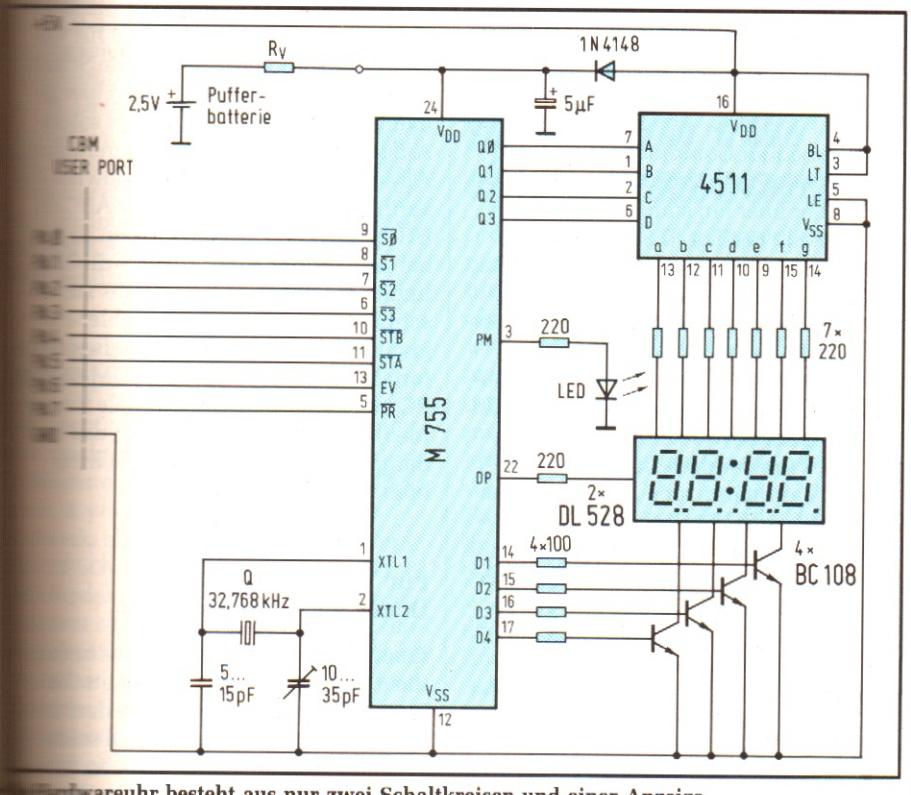


Die Schaltung für die „Capitals-Lock"-Funktion

Gietmann, Hermann Funke

Hardwareuhr für CBM

Die CBM-Computer sind mit einer Softwareuhr ausgestattet. Wem deren Genauigkeit nicht ausreicht, findet hier eine Hardwarelösung.



Die Hardwareuhr besteht aus nur zwei Schaltkreisen und einer Anzeige

Tabelle: Tastaturbedienungen für die verschiedenen Funktionen der Uhr

Bedienung	Auswirkung
TI$ = „120000“	Die interne Uhr des Rechners wird gesetzt.
DA$ = „140781“	Das Datum wird mit 14. 7. 81 festgelegt.
#	Die Hardwareuhr wird mit dem Inhalt von TI$ und DA$ geladen, der Dezimalpunkt beginnt zu blinken.
\	Die Uhrzeit und das Datum wird kontinuierlich oben auf dem Bildschirm angezeigt. Nochmaliges Drücken der Taste unterdrückt die Anzeige.
/	Anzeigenwechsel zwischen Zeit und Datum auf dem Uhrendisplay.
!	Der Drucker fertigt eine Hardcopy mit Uhrzeit und Datum an.
*	TI$ und DA$ werden von der Hardwareuhr in die interne Uhr übernommen.

Auf einer relativ kleinen Platine sind der Uhrenschaltkreis, der Decoder, die 4stellige, 13 mm hohe Anzeige sowie die Treibertransistoren untergebracht. Die Schnittstelle zum Mikrocomputer bildet ein bidirektionaler Bus von 4 Bit Breite, der mit Hilfe von zwei Steuerleitungen wahlweise als Daten- oder Adreßbus benutzt wird. Zusätzlich sind noch zwei Anschlüsse für Sonderfunktionen verfügbar (*Bild*).

Die Vorteile der Hardwarelösung

Hinsichtlich der Genauigkeit erfüllt die Hardwareuhr alle Anforderungen für den kommerziellen Einsatz sowie für naturwissenschaftliche Anwendungen. Wird der Rechner ausgeschaltet oder fällt das Netz aus, übernimmt ein kleiner Akku auf der Platine die Spannungsversorgung der Uhr, die Anzeige ist dabei ausgeschaltet. Die Platine wird mit einer Flachbandleitung mit dem User-Port des Rechners verbunden, lediglich der 5-V-Anschluß ist aus dem Rechner herauszuführen.
Die Uhr kann mit wenigen Steuerbefehlen gesetzt oder gelesen werden, außerdem kann die Uhrenanzeige auch als Display für andere Daten benutzt werden.

Die Software dazu

Die Uhrzeit und das Datum können per Programm gestellt und gelesen werden. Das Programm umfaßt etwa 500 Byte, kann am oberen Ende des RAM-Bereiches, im ROM-Bereich oder integriert im Bereich des Betriebssystems liegen. Bestimmte Tasten des Rechners werden mit Sonderfunktionen belegt und ermöglichen so die einfache Steuerung der Hardwareuhr (*Tabelle*).
Der Datums- und Monatszähler arbeitet bis zum Schaltjahr richtig und muß am 1. März korrigiert werden. Die aktuelle Zeit kann vom Programm aus mit PRINT TI$ abgerufen werden, ebenso wie das Datum mit PRINT DA$ beispielsweise in Protokolle eingefügt werden kann.
Gegen Erstattung der Unkosten ist ein vollständig dokumentiertes Assembler-Listing bei den Verfassern (Ahornstr. 9, 4174 Issum 1) erhältlich.
Grundsätzlich ist der Anschluß dieser Hardwareuhr an alle Rechner mit einer programmierbaren Parallelschnittstelle von mindestens 8 Bit möglich. Das Handling des Uhrenbausteins ist ausführlich in [1] beschrieben.

Literatur

[1] Datenblatt und Beschreibung M 755/M 756. SGS/ATES.

Michael Konz

CBM-EPROM-Programmer

Obwohl die Programmierung von EPROMs eine recht einfache Sache ist, verfügt doch nur eine Minderheit über entsprechende Vorrichtungen. Die große Mehrheit verzichtet entweder ganz auf den Einsatz dieser Festwertspeicher oder läßt sich die Programmierung teures Geld kosten. Um dem abzuhelfen, ist hier die Schaltung eines äußerst flexiblen Programmiergerätes beschrieben, das mit einem vergleichsweise geringen Aufwand an Hardware auskommt und direkt an den User-Port eines CBM 3000 oder 4000 angeschlossen werden kann. Mit der Software ist die Programmierung selbst von 4-KByte-PROMs möglich.

Die Schaltung in *Bild 1* ist universell verwendbar für die Typen 2508, 2516 und 2532, also 1-K-, 2-K- und 4-K-EPROMS. Die Auswahl erfolgt per Software.
Wer sich die Pinbelegung dieser EPROMs (*Bild 2*) anschaut, der wird feststellen, daß beim 2532 außer den acht Dateneingängen zwölf Adreßeingänge und ein Steuereingang vorhanden sind. (Beim 2516 sind es elf Adreßleitungen und zwei Steueranschlüsse.) Zusammen sind das 21 Eingänge.
Am User-Port des CBM stehen aber auch bei bester Ausnutzung aller auf diese Anschlußleiste gelegten I/O-Leitungen nur zwölf Ports zur Verfügung. Würden zur Abdeckung der übrigen neun Leitungen Ports des IEC-Interfaces herangezogen werden, so verlöre der CBM eben dieses Interface. Ein Betrieb von Peripheriegeräten wäre nicht mehr möglich. Eine andere Möglichkeit wäre, die zwölf zur Verfügung stehenden Leitungen zu multiplexen. Das aber würde einen erhöhten Hardware-Aufwand erfordern. Daher wurde eine dritte Alternative gewählt: Die Adreßleitungen werden nicht mehr direkt vom Rechner aus angesteuert, sondern über einen vom Computer getakteten Binärzähler. Damit werden für die Adressierung des EPROMs nur noch zwei Leitungen benötigt, nämlich Zähler-Reset und Zähler-Clock. Zwei weitere Leitungen steuern die Anschlüsse PD/PGM und $\overline{CS}$ bzw. A_{11} und PD/$\overline{PGM}$ beim 2532. Bleiben noch genau acht Ports für die Dateneingänge des EPROMs.
Die *Tabelle* zeigt, wie diese Leitungen am User-Port anzuschließen und wie sie intern geschaltet sind.

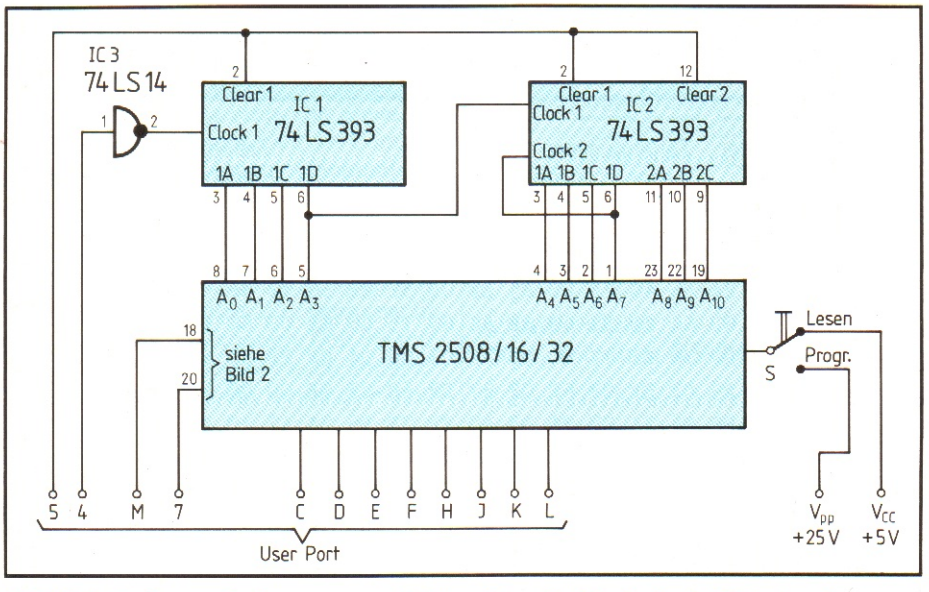


Bild 1. Die Hardware der Programmierschaltung ist recht einfach, der Schalter S muß vor dem Programmieren von Hand betätigt werden

Die Hardware

Es finden Binärzähler vom Typ 74 LS 393 Verwendung. Jeder dieser Bausteine enthält zwei 4-Bit-Zähler. Der Schmitt-Trigger 74 LS 14 ist eigentlich eine Verlegenheitslösung. Beim Aufbau der Schaltung wurde nämlich festgestellt, daß sich der Clock-Eingang des ersten Zählers beim besten Willen nicht exakt vom Rechner takten ließ. Nach Einschalten des Schmitt-Triggers ging es dagegen problemlos. Pin 18 des EPROMs, der beim 2508 und 2516 als Eingang für den Programmierimpuls dient, wird beim 2532 als zwölfter Adreßeingang verwendet. Hardwaremäßig ist dafür jedoch keine Umschaltung erforderlich, das erledigt die Software. Lediglich der Umschalter S muß vor Beginn des Programmiervorgangs betätigt werden, um die Programmierspannung von 25 V auf den Pin 21 zu schalten. Ohne sie findet keine Programmierung statt.
Leider stand für diese Umschaltung kein Port mehr zur Verfügung, sonst hätte sie natürlich auch vom Rechner aus gesteuert werden können.
Die Spannungsversorgung des Programmers erfolgt über ein eigenes Netzteil. Die Schaltung dafür ist in *Bild 3* wiedergegeben. Zu ihr ist nicht viel zu sagen. Man kann sie in dieser oder ähnlicher Form in jedem Datenbuch finden, das die verwendeten Stabilisatoren 7805 und LM 317 aufführt.
Das Netzteil liefert außer der über den Trimmer P1 einzustellenden Programmierspannung von 25 V auch eine 5-V-Spannung, da am User-Port keine 5 V zur Verfügung stehen. Anstelle des Trimmers kann natürlich auch ein entsprechend dimensionierter Festwiderstand gesetzt werden.

Die Programmierung

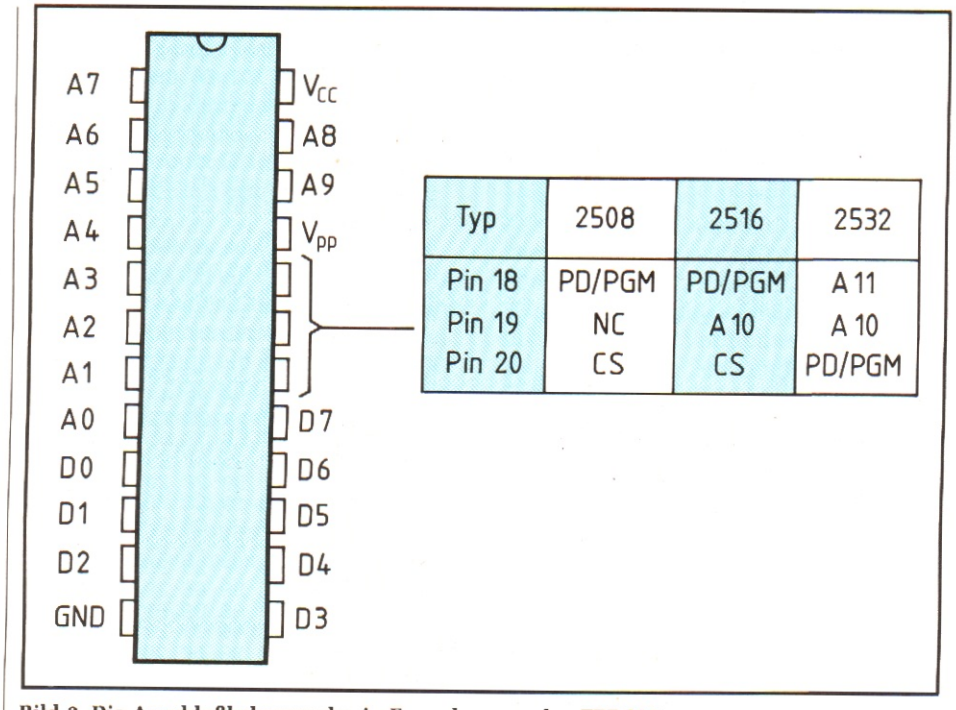
Um eine Programmierung überhaupt ermöglichen, benötigen die Typen 2508 und 2516 ein High-Potential (5 V) an C und ein Low-Potential (0 V) an PD/PGM. Nach Anwahl der zu programmierenden Adresse und Ausgabe der Daten muß PD/PGM-Eingang für genau 50 Millisekunden (mit einer Toleranz von 5 ms) auf High-Potential gelegt werden. Nach dem die Leitung wieder ihr altes Po

genommen hat, kann der Erfolg der Programmierung sofort verifiziert werden. Anschließend ist die nächste Adresse an der Reihe. Anders beim 2532. Das 4 K-EPROM benötigt neben der obligatorischen 25 V an Vpp High-Potential an PD/$\overline{PGM}$. (Achtung: Der PD/$\overline{PGM}$-Anschluß des 2532 ist der $\overline{CS}$-Eingang des 2516/08.)
Die Programmierung wird auf diese Leitung ein 50 ms dauernder Low-Impuls gegeben.
In den Datenblättern für die EPROMs sind gewisse Mindestzeiten angegeben, die zwischen Datenänderungen auf den verschiedenen Leitungen verstreichen müssen. Doch diese Zeiten sind durch relativ langsamen Schaltvorgänge im Rechner, die sich ja über mehrere Mikrosekunden erstrecken, in jedem Fall garantiert.

Die Software

Der kurzer Basic-Teil übernimmt die Ausgabe einer Bedienungsanleitung auf dem Bildschirm. Der Hauptteil des Programms ist dagegen in Maschinensprache geschrieben (*Bild 4*). Dieser Hauptteil wird nach Ausführung der Basic-Routine automatisch gestartet.
Zunächst fragt der Rechner nach einem Speicherbereich. Dieser Speicherbereich ist je nach gewähltem EPROM-Typ bis 4 KByte. Es kann jedoch auch ein kleinerer Bereich eingegeben werden. In diesen Bereich werden Daten vom EPROM eingelesen und nur Daten, die in diesem Bereich stehen, können in das EPROM gebrannt werden. Ein zu kleiner Bereich oder ein Bereich, der unter 2000 beginnt, wird zurückgewiesen.
Mit dieser Prozedur stehen folgende Kommandos zur Verfügung:

- Der Speicherbereich kann neu definiert werden.
- Die im EPROM stehenden Daten werden ab der auf die Frage ‚EPROM=' einzugebenden EPROM-Adresse in den definierten Speicherbereich eingelesen.
- Die im definierten Speicherbereich stehenden Daten werden ab der eingegebenen EPROM-Adresse programmiert. Es erscheint zunächst die Aufforderung, die Programmierspannung einzuschalten (S). Der Benutzer muß die Ausführung mit einem Tastendruck quittieren. Zur Anzeige, daß der Programmiervorgang läuft, wird das ‚PROGRAMMING' in der Kopfzeile invers dargestellt. Unterhalb der Kommandozeile wird jede programmierte Page angezeigt (01...02...03). Die Programmierung kann jederzeit mit der STOP-Taste unterbrochen werden. Bei den Typen 2508 und 2516 erfolgt auf jedes programmierte Byte sofort ein Verify. Bei einem Programmierfehler wird der Vorgang unterbrochen und es bleibt dem Benutzer überlassen, ob er den Programmiervorgang mit ‚Y' fortzusetzen wünscht, oder ob er ihn mit ‚N' ganz abbrechen will. Beim 2532 erfolgt dieses automatische Verify allerdings nicht.
  Nach Abschluß der Programmierung wird der Benutzer wieder aufgefordert, die Programmierspannung abzuschalten. Auch diese Handlung ist mit einem Tastendruck zu quittieren.
- V – Mit dem Verify-Befehl können die im Speicher stehenden Daten mit denen im EPROM verglichen werden. Bei einer Nichtübereinstimmung wird genauso verfahren wie beim P-Befehl.
- C – Mit diesem Befehl kann überprüft werden, ob ein EPROM vollständig gelöscht ist. Dabei wird jedes Byte mit dem Wert $FF verglichen. Eine Fehlermeldung erfolgt, wenn ein Byte ungleich $FF ist.



Typ	2508	2516	2532
Pin 18	PD/PGM	PD/PGM	A 11
Pin 19	NC	A 10	A 10
Pin 20	CS	CS	PD/PGM

Bild 2. Die Anschlußbelegung der in Frage kommenden EPROMs

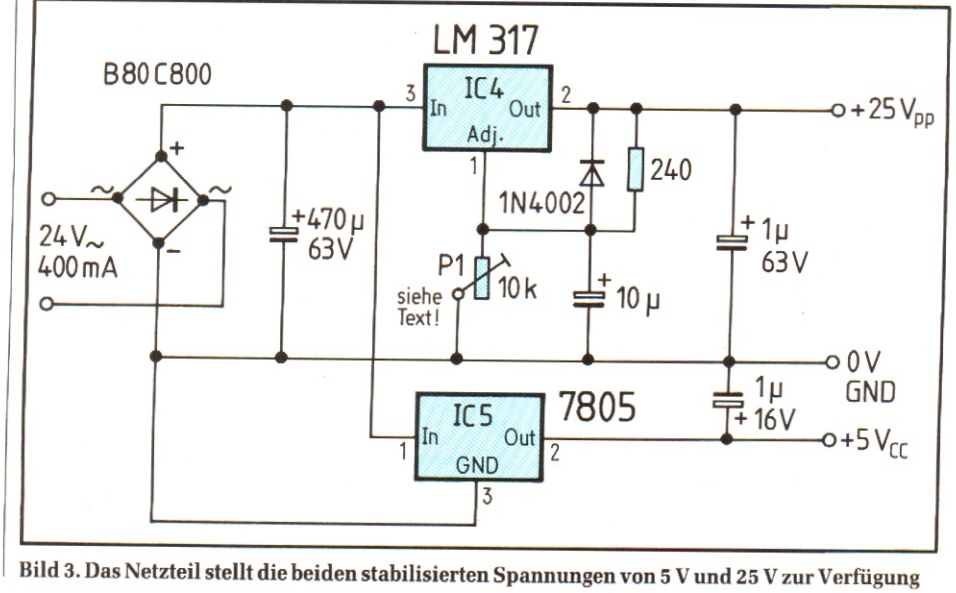


Bild 3. Das Netzteil stellt die beiden stabilisierten Spannungen von 5 V und 25 V zur Verfügung

```
 PC   IRQ   BR AC XR YR SP
B780 E455 2C 34 3A 9D FA
0400 00 0B 04 64 00 99 22 93
0408 22 3B 00 31 04 6E 00 99
0410 22 20 20 20 20 20 20 20
0418 20 20 45 50 52 4F 4D 20
0420 32 35 31 36 20 50 52 4F
0428 47 52 41 4D 4D 49 4E 47
0430 00 62 04 78 00 99 22 A2
0438 A2 A2 A2 A2 A2 A2 A2 A2
0440 A2 A2 A2 A2 A2 A2 A2 A2
0448 A2 A2 A2 A2 A2 A2 A2 A2
0450 A2 A2 A2 A2 A2 A2 A2 A2
0458 A2 A2 A2 A2 A2 A2 A2 22
0460 3B 00 75 04 82 00 99 22
0468 A1 22 A3 37 39 29 22 12
0470 A1 92 22 3B 00 A8 04 8C
0478 00 99 22 A1 20 20 20 4D
0480 20 2D 2D 20 44 45 46 49
0488 4E 45 20 4D 45 4D 4F 52
0490 59 20 41 52 45 41 20 20
0498 20 20 20 20 20 20 20 20
04A0 20 20 12 A1 92 22 3B 00
04A8 DB 04 96 00 99 22 A1 20
04B0 20 20 52 20 2D 2D 20 52
04B8 45 41 44 20 44 41 54 41
04C0 20 49 4E 54 4F 20 4D 45
04C8 4D 4F 52 59 20 20 20 20
04D0 20 20 20 20 20 12 A1 92
04D8 22 3B 00 0E 05 A0 00 99
04E0 22 A1 20 20 20 50 20 2D
04E8 2D 20 42 55 52 4E 20 44
04F0 41 54 41 20 49 4E 54 4F
04F8 20 45 50 52 4F 4D 20 20
0500 20 20 20 20 20 20 20 20
0508 12 A1 92 22 3B 00 41 05
0510 AA 00 99 22 A1 20 20 20
0518 56 20 2D 2D 20 56 45 52
0520 49 46 59 20 44 41 54 41
0528 20 20 20 20 20 20 20 20
0530 20 20 20 20 20 20 20 20
0538 20 20 20 12 A1 92 22 3B
0540 00 74 05 B4 00 99 22 A1
0548 20 20 20 43 20 2D 2D 20
0550 43 48 45 43 4B 20 45 50
0558 52 4F 4D 20 20 20 20 20
0560 20 20 20 20 20 20 20 20
0568 20 20 20 20 20 20 12 A1
0570 92 22 3B 00 A7 05 BE 00
0578 99 22 A1 20 20 20 44 20
0580 2D 2D 20 44 49 53 50 4C
0588 41 59 20 44 41 54 41 20
0590 20 20 20 20 20 20 20 20
0598 20 20 20 20 20 20 20 20
05A0 20 12 A1 92 22 3B 00 DA
05A8 05 C8 00 99 22 A1 20 20
05B0 20 45 20 2D 2D 20 45 58
05B8 49 54 20 20 20 20 20 20
05C0 20 20 20 20 20 20 20 20
05C8 20 20 20 20 20 20 20 20
05D0 20 20 20 20 12 A1 92 22
05D8 3B 00 0D 06 CA 00 99 22
05E0 A1 20 20 20 53 20 2D 2D
05E8 20 53 45 4C 45 43 54 20
05F0 54 59 50 45 20 20 20 20
05F8 20 20 20 20 20 20 20 20
0600 20 20 20 20 20 20 20 12
0608 A1 92 22 3B 00 20 06 CD
0610 00 99 22 A1 22 A0 33 39
0618 29 22 12 A1 92 22 3B 00
0620 53 06 D2 00 99 22 12 A2
0628 A2 A2 A2 A2 A2 A2 A2 A2
0630 A2 A2 A2 A2 A2 A2 A2 A2
0638 A2 A2 A2 A2 A2 A2 A2 A2
0640 A2 A2 A2 A2 A2 A2 A2 A2
0648 A2 A2 A2 A2 A2 A2 A2 92
0650 22 3B 00 7B 06 E6 00 99
0658 22 44 41 54 41 20 46 49
0660 45 4C 44 20 46 52 4F 4D
0668 20 12 30 30 30 30 92 20
0670 54 4F 20 12 30 30 30 30
0678 92 22 00 85 06 F0 00 9E
0680 31 36 37 31 00 00 00 AD
0688 00 C0 C9 40 F0 1C AD C2
0690 0A C9 7F F0 1C A2 17 BD
0698 D9 0A A8 BD C1 0A 9D D9
06A0 0A 98 9D C1 0A CA 10 EF
06A8 30 07 AD C2 0A C9 5D D0
06B0 E4 78 A9 38 8D 11 E8 AD
06B8 10 E8 09 80 8D 10 E8 A9
06C0 3C 8D 11 E8 58 20 84 09
06C8 A9 00 A2 05 95 38 CA 10
06D0 FB A9 08 85 41 20 16 07
06D8 A2 0F A0 00 20 7D 09 A0
06E0 00 20 72 09 85 A7 A5 9E
06E8 F0 FC 20 C4 0A A2 07 DD
06F0 F1 0A F0 05 CA 10 F8 30
06F8 ED E6 A7 20 D2 FF 8A 0A
0700 AA BD FA 0A 48 BD F9 0A
0708 48 A9 20 A2 27 9D C0 83
0710 CA 10 FA 4C C7 0A A0 10
0718 20 72 09 20 35 0A A5 FB
0720 A4 FC C0 10 90 47 85 38
0728 84 39 A0 18 20 72 09 20
0730 35 0A A5 FB 38 85 3A E5
0738 38 A5 FC 85 3B E5 39 90
0740 2C C5 41 B0 28 20 27 0A
0748 A2 0D A0 10 20 7D 09 E6
0750 9F A5 39 20 CA 0A A5 38
0758 20 CA 0A A0 18 84 C6 20
0760 CD 0A C6 9F 20 9D 09 8D
0768 67 82 4C D8 06 20 27 0A
0770 30 A4 20 A8 09 AD 41 E8
0778 91 3E 20 4C 09 20 CA 09
0780 90 F3 58 B0 DF 20 A8 09
0788 AD 41 E8 D1 3E D0 0B 20
0790 4C 09 20 CA 09 90 F1 58
0798 B0 CA 20 E2 09 A0 00 B1
07A0 3E 20 08 0A 90 E9 58 B0
07A8 BB 78 20 19 09 A9 00 85
07B0 40 AD 41 E8 C9 FF D0 1C
07B8 20 4C 09 A5 3D 05 3C D0
07C0 F0 A5 40 D0 9F A2 18 A0
07C8 00 20 7D 09 A0 40 20 72
07D0 09 58 F0 90 20 E2 09 A9
07D8 FF 20 08 0A 90 DA 58 4C
07E0 64 07 20 D3 0A 4C D6 0A
07E8 20 A8 09 A2 11 A0 00 20
07F0 7D 09 A2 04 86 40 A0 07
07F8 A5 3D 20 CA 0A A5 3C 20
0800 CA 0A A9 2D 20 D2 FF 20
0808 D0 0A AD 41 E8 20 CA 0A
0810 20 4C 09 88 10 F1 20 C7
0818 0A C6 40 10 D9 20 0B 0A
0820 90 C9 58 4C 64 07 20 A8
0828 09 20 27 0A A0 6E 20 72
0830 09 A5 41 C9 10 D0 08 AD
0838 40 E8 09 08 8D 40 E8 58
0840 A5 9E F0 FC 20 27 0A 20
0848 9B 0A 78 AD 4B E8 48 29
0850 3F 8D 4B E8 AD 40 E8 09
0858 08 8D 40 E8 A9 FF 8D 43
0860 E8 A0 00 B1 3E 8D 41 E8
0868 20 A9 0A A9 50 A0 C3 8D
0870 44 E8 8C 45 E8 2C 4D E8
0878 50 FB 20 A9 0A E0 10 F0
0880 0F A0 00 8C 43 E8 20 84
0888 09 AD 41 E8 D1 3E D0 3E
0890 20 CA 09 B0 20 20 4C 09
0898 90 09 A0 00 84 C6 A5 3D
08A0 20 CA 0A AD 12 E8 CD 12
08A8 E8 D0 F8 C9 EF F0 06 A5
08B0 3D C5 41 90 9F 68 8D 4B
08B8 E8 20 9B 0A A0 8A 20 72
08C0 09 85 9E 58 A5 9E F0 FC
08C8 20 84 09 4C 64 07 20 E2
08D0 09 A0 00 B1 3E 20 08 0A
08D8 90 B6 B0 D9 A0 9D 20 72
08E0 09 85 A7 A5 9E F0 FC 20
08E8 C4 0A C9 31 F0 14 C9 32
08F0 F0 0A C9 34 D0 ED A9 32
08F8 A2 10 D0 0A A9 16 A2 08
0900 D0 04 A9 08 A2 04 86 41
0908 48 A2 00 A0 11 20 7D 09
0910 68 85 A7 20 CA 0A 4C 64
0918 07 A9 00 85 3C 85 3D AD
0920 10 E8 09 80 8D 10 E8 29
0928 7F 8D 10 E8 A5 41 C9 10
0930 D0 05 A9 CC 8D 4C E8 60
0938 20 19 09 A6 3C A4 3D C4
0940 FC D0 04 E4 FB F0 F0 20
0948 4C 09 D0 EF A9 34 8D 11
0950 E8 A9 3C 8D 11 E8 E6 3C
0958 D0 16 E6 3D A5 3D C5 41
0960 F0 B7 C9 08 D0 08 AD 4C
0968 E8 49 20 8D 4C E8 38 60
0970 18 60 B9 09 0B F0 C0 20
0978 D2 FF C8 D0 F5 86 D8 84
0980 C6 4C C1 0A A5 41 C9 10
0988 F0 05 A9 CC 8D 4C E8 A9
0990 00 8D 43 E8 AD 40 E8 29
0998 F7 8D 40 E8 60 A9 20 A2
09A0 00 9D 80 82 E8 D0 FA 60
09A8 A0 22 20 72 09 20 35 0A
09B0 A5 FC C5 41 B0 0F 78 20
09B8 38 09 A5 38 A4 39 85 3E
09C0 84 3F A0 00 60 20 27 0A
09C8 30 DE E6 3E D0 02 E6 3F
09D0 A6 3F E4 3B 90 0B D0 09
09D8 A6 3E E4 3A 90 03 D0 01
09E0 18 60 A2 13 A0 00 20 7D
09E8 09 A0 29 20 72 09 A5 3D
09F0 20 CA 0A A5 3C 20 CA 0A
09F8 A9 3A 20 D2 FF AD 41 E8
0A00 20 CA 0A A0 33 4C 72 09
0A08 20 CA 0A A0 B1 20 72 09
0A10 58 A5 9E F0 FC 20 C4 0A
0A18 C9 4E F0 05 C9 59 D0 F1
0A20 18 78 20 27 0A C8 60 A0
0A28 00 84 C6 A0 27 A9 20 91
0A30 C4 88 10 FB 60 A9 00 85
0A38 A7 85 40 A5 9E F0 FC 20
0A40 C4 0A C9 14 D0 0F A6 40
0A48 F0 F1 C6 40 E6 A7 20 D2
0A50 FF C6 A7 F0 E6 C9 30 90
0A58 E2 C9 47 B0 DE C9 3A 90
0A60 08 C9 41 90 D6 48 69 08
0A68 24 48 29 0F A6 40 9D 00
0A70 02 68 E6 A7 20 D2 FF E0
0A78 03 F0 06 C6 A7 E6 40 D0
0A80 BA A2 00 BD 00 02 0A 0A
0A88 0A 0A 1D 01 02 48 E8 E8
0A90 E0 04 90 EF 68 85 FB 68
0A98 85 FC 60 A2 0A BD 14 80
0AA0 49 80 9D 14 80 CA 10 F5
0AA8 60 A6 41 E0 10 D0 09 AD
0AB0 40 E8 49 08 8D 40 E8 60
0AB8 AD 4C E8 49 20 8D 4C E8
0AC0 60 4C 7F E0 4C A7 E0 4C
0AC8 34 D5 4C 22 D7 4C 17 D7
0AD0 4C 31 D5 4C 4B E0 4C FF
0AD8 B3 4C 5D E2 4C 85 E2 4C
0AE0 D0 FD 4C 75 E7 4C 6A E7
0AE8 4C CD FD 4C 29 E2 4C 89
0AF0 C3 4D 52 50 56 43 44 45
0AF8 53 15 07 71 07 25 08 84
0B00 07 A8 07 E7 07 E1 07 DB
0B08 08 45 4E 54 45 52 20 43
0B10 4F 4D 4D 41 4E 44 3A 20
0B18 00 53 54 41 52 54 3D 20
0B20 00 20 54 4F 20 45 4E 44
0B28 3D 20 00 46 52 4F 4D 3D
0B30 20 00 45 52 52 4F 52 20
0B38 41 54 20 00 20 49 4E 53
0B40 54 45 41 44 20 4F 46 20
0B48 00 12 45 50 52 4F 4D 20
0B50 43 4F 4D 50 4C 45 54 45
0B58 4C 59 20 45 52 41 53 45
0B60 44 92 00 12 50 52 4F 47
0B68 52 41 4D 4D 49 4E 47 20
0B70 45 50 52 4F 4D 92 00 12
0B78 54 55 52 4E 20 4F 4E 20
0B80 50 52 4F 47 52 41 4D 4D
0B88 49 4E 47 20 50 4F 57 45
0B90 52 92 00 12 53 57 49 54
0B98 43 48 20 50 4F 57 45 52
0BA0 20 4F 46 46 92 00 45 4E
0BA8 54 45 52 20 23 20 4F 46
0BB0 20 4B 42 59 54 45 53 3A
0BB8 20 00 0D 43 4F 4E 54 49
0BC0 4E 55 45 20 28 59 2F 4E
0BC8 29 20 3F 00 00 00 00 AA
```

Bild 4. Der Hex-Dump des vollständigen Programms beeinhaltet auch den Basic-Teil

[illegible]–Der Display-Befehl dient zur Anzeige der im EPROM stehenden Daten, die dabei jedoch nicht in den Speicher übernommen werden. Es werden immer fünf Zeilen zu je acht Byte gleichzeitig angezeigt. Mit ‚Y' können die nächsten vierzig Byte abgerufen werden.
[illegible]–Rückkehr ins Basic.
[illegible]–Auswahl des EPROM-Typs. Der Rechner fragt nach der Anzahl der KBytes.

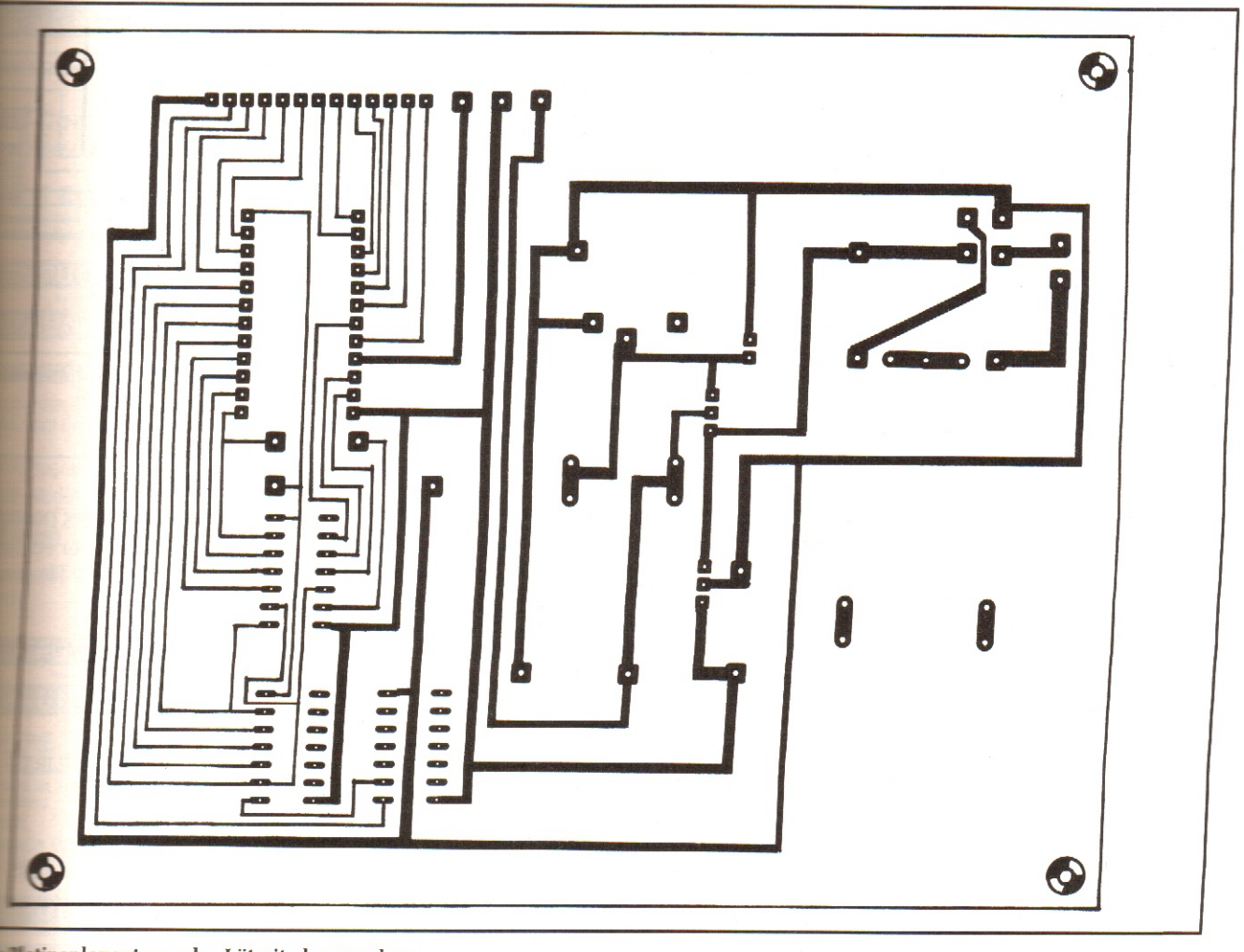
Bei der Eingabe des Basic-Teils des Programms muß darauf geachtet werden, daß kein Zeichen zuviel oder zuwenig eingegeben wird, da sich der Maschinenspracheteil direkt anschließt. Eine Überprüfung ist durch Abfrage der Speicherzellen $2A/2B möglich. Hier muß am Abschluß der Eingabe der Wert [illegible] stehen. Ist das nicht der Fall, sollte das eingegebene Programm noch einmal genau untersuchen.
Anschließend kann der Maschinenspracheteil beginnend bei $0687 eingegeben werden. Er endet bei $0BCB. Auf diese Adresse muß auch der Pointer $2A/2B am Schluß gesetzt werden, d. h. $2A wird mit dem Wert $CB geladen und $2B mit $0B. Nun kann das fertige Programm mit einem normalen SAVE oder DSAVE auf Kassette oder Disk abgespeichert werden. Der Start erfolgt dann einfach mit RUN.
Das Programm ist so ausgelegt, daß es sich automatisch auf die ROM-Version des Rechners einstellt. Daher sind keine Änderungen erforderlich, gleichgültig, ob das Programm auf einem CBM 3000 oder einem CBM 4000 laufen soll. Nicht lauffähig ist es dagegen auf einem CBM 8000 wegen des anderen Bildschirms.
Diese Umstellung geschieht einfach durch entsprechendes Austauschen von Sprungadressen innerhalb einer Tabelle.

Aufbau des Gerätes

Um den Aufbau des Gerätes zu vereinfachen, zeigt *Bild 5* ein Platinen-Layout. Auf ihm sind Netzteil und Programmer vereint (*Bild 6*). Für den Transformator ist ein Typ mit 2 Wicklungen von 12 V/400 mA für Printmontage vorgesehen, doch kann natürlich auch ein Typ mit einer einfachen 24-V-Wicklung Verwendung finden. Die Hintereinanderschaltung der beiden Wicklungen ist dann selbstverständlich überflüssig.
Für das EPROM empfiehlt sich ein Wechselsockel (Textool), um die Speicherbausteine vor Beschädigung zu schützen. Da Teile der Schaltung direkt mit dem Netz in Verbindung stehen, sollte das fertige Gerät unbedingt in ein Gehäuse eingebaut werden. Den Sockel kann man dabei in einem Ausschnitt an der Oberseite des Gehäuses unterbringen und die Verbindung zur Platine gegebenenfalls mit Drahtbrücken herstellen.

Anwendung

Das einzige, was der Benutzer noch zu tun hat, ist, den Rechner mit den zu programmierenden Daten zu füttern. Besonders Besitzer eines auf Basic 4.0 aufgerüsteten CBM 3000 werden sehr



Das Platinenlayout von der Lötseite her gesehen

Tabelle: So sind die Anschlüsse des User-Ports belegt. PIA 1 ist der Interface-Baustein ab Adresse $E810, VIA der ab $E840

Funktion	Anschluß	Rechner	intern
Reset	J2-5	PA 7	PIA 1
Clock	J2-4	CA 2	PIA 1
CS (PD/PGM)	J2-7	PB 3	VIA
PD/PGM (A 11)	J2-M	CB 2	VIA
Data 7	J2-L	PA 7	VIA
Data 6	J2-K	PA 6	VIA
Data 5	J2-J	PA 5	VIA
Data 4	J2-H	PA 4	VIA
Data 3	J2-F	PA 3	VIA
Data 2	J2-E	PA 2	VIA
Data 1	J2-D	PA 1	VIA
Data 0	J2-C	PA 0	VIA

schnell Verwendungsmöglichkeiten für die vielen freien Bytes innerhalb der 4000er ROMs finden. Schaut man sich dieses Betriebssystem nämlich einmal an, dann wird man feststellen, daß über 1,5 K der 18 K mit dem hexadezimalen Wert $AA programmiert und demzufolge unbenutzt sind. Verwendet man statt der Commodore-ROMs nun eigene EPROMs, so lassen sich in dem freien Raum eine Vielzahl nützlicher Routinen oder kleinerer Spracherweiterungen unterbringen, ohne daß man dafür in den wertvollen Bereich von $9000...$AFFF zu gehen braucht.
Leider ist Commodore bei den neueren 4000er-Geräten dazu übergegangen, die ROMs nicht mehr zu sockeln, wohl aus Kostengründen. Und es ist nicht unbedingt zu empfehlen, hier mit dem Lötkolben Abhilfe schaffen zu wollen. Schade, denn durch diese Maßnahme verliert der CBM 4000 eine gehörige Portion an Flexibilität, die doch eigentlich keinem Gerät schaden kann. Wohl dem, der noch einen der alten 3000er besitzt...

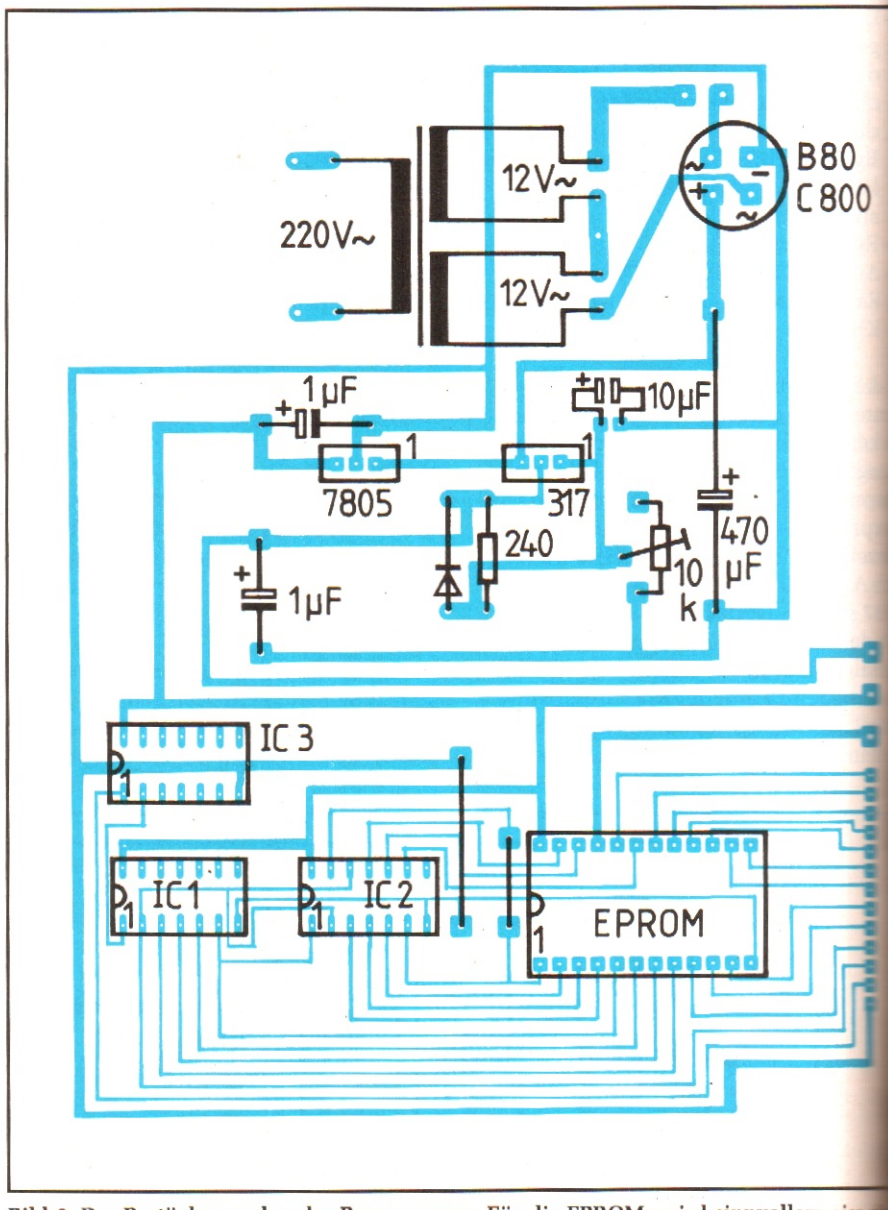


Bild 6. Der Bestückungsplan des Programmers. Für die EPROMs wird sinnvollerweise ein Wechselsockel eingesetzt

AIM-65/PC-100: Zusammenarbeit Forth und Editor

Das folgende Programm in Forth erleichtert die Arbeit mit dem Editor bei der Bearbeitung von Forth-Programmen. Nach dem Initialisieren des Editors und Laden des Programms ins Forth-Dictionary muß nicht mehr zwischen Editor, Monitor und Forth hin- und hergesprungen werden. Durch den Editorbefehl L IN=U ist man sofort in Forth und kann den Text im Editor durch S IN=M la[illegible] Die Rückkehr in den Editor erfolgt [illegible] das Forth-Wort E. *Thomas A[illegible]*

Das ist das Forthprogramm „Zusammenarbeit"

```
= ⟨L⟩
/
OUT=P
FORGET TASK
HEX
BØØ3 Ø1ØA !
: S
 R⟩ R⟩ R⟩
 ØØE3 @ ØØDF !
 SOURCE ;
( EDITOR T )
CODE E
 F6CF JMP,
END-CODE
( NEW VALUE FENCE )
HERE FENCE !
 TASK ;
FINIS
= ⟨L⟩
/
OUT=U
 AIM 65 FORTH V1.[illegible]
S IN=M
OK
E
FORGET TASK
```