Eprombrenner für den Plus 4

Unterlagen und Infos zu dem Braunroth EPROM-Brenner für den PLUS 4

Ich hatte mir in den 90er Jahren einen EPROM Brenner für den Plus 4 zugelegt, allerdings die meisten Unterlagen und Disketten wohl entsorgt. Da ich mich wieder mit der Materie beschäftige, bin ich auf der Suche nach den nötigen Infos, um den Brenner wieder in Betrieb zu nehmen und zu verbessern. Ich habe daher die gefundenen Unterlagen und Infos aus einigen Zeitschriften versucht hier zusammenzufassen.

Es gab einige Infos, scheinbar noch vom Kauf von Braunroth und vom Ing. Uwe Peters (leider verstorben) in den Dokumenten von Uwe Peters heißt der Brenner Eprog 86032, keine Ahnung mehr warum ich die habe, es scheint aber ein anderer Brenner zu sein. Weiter Infos aus den Zeitschriften Compute mit 8/88 und den Compute mit Sonderzeitschriften 5/88, 6/88 und 1/89. Alle Infos habe ich in dieses PDF eingefügt. Da der Brenner einige Schwächen hat, will ich diese, wenn er läuft, abstellen. Dazu gehören Gehäuse, Textoolsockel, Beschriftung und auch eine Bedienungsanleitung und das Brenner-Programm sollen am Ende dabei herauskommen.

Sollte jemand noch Original Unterlagen oder Programme dazu haben, gerne eine Info an mich, ebenso bei Fehlern, Verbesserungen oder sollte ich Copyrights verletzt haben auch dann gerne Info an mich.

Vielen Dank

Gruß pitdahl

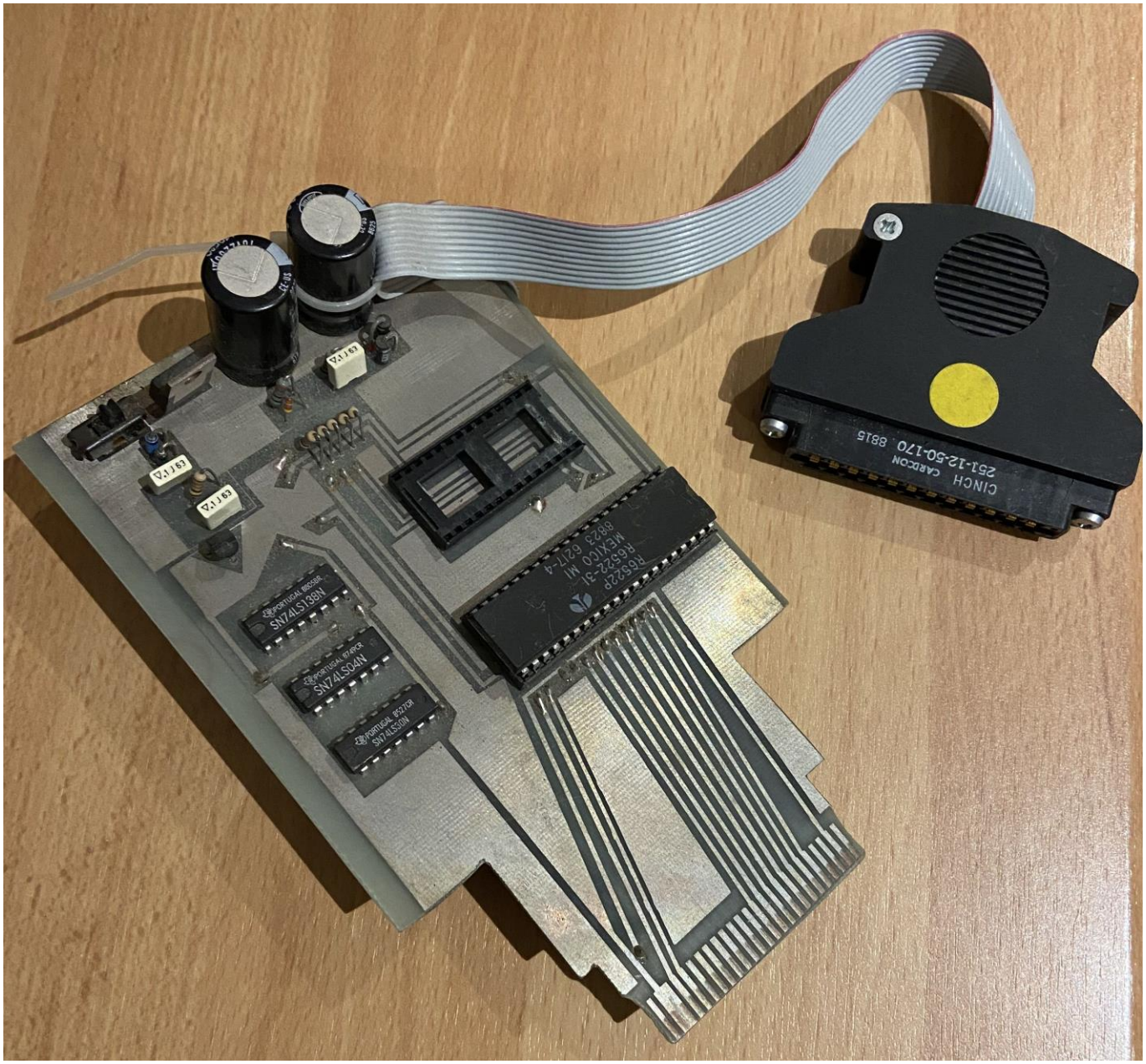
Die Daten zu dem Brenner von damals:

```
Erweiterung : Eprombrenner
System      : Plus4
Preis       : ca. 78-95 DM
Hersteller  : Drohne-Soft
Bezugsquelle: Jürgen Braunroth, Moordorfer Str.30, 3057
Neustadt 1
```

Es soll auch noch das Schaltbild in dieses Dokument und die Kurse aus den Compute mit Zeitschriften, auch hier sind einige Hardware und Programmierung Tipps enthalten. Aber nun erst mal ein Bild, vor irgendwelchen Verbesserungen ☺

Hier ein Bild meines Brenners:

Eprombrenner für den Plus 4



Brennen mit dem Plus 4.

Quelle: Compute mit 8/88

Egal, ob Änderungen am Betriebssystem gewünscht oder Programme als Module benötigt werden, am Eprom-Brenner kommt kein ernsthafter Computerfan vorbei. Mit dem Brenner der Firma Jürgen Braunroth gibt es jetzt endlich auch einen für den „Kleinen" von Commodore.

Mit dem Brenner ersteht der Computerbesitzer ein brauchbares Gerät zum Programmieren aller Eproms der Typen 2764 bis 27256. Zum Lieferumfang gehören der Prommer, die Steuersoftware und eine ausführliche sechsseitige Anleitung.
Die Firma Braunroth bringt demnächst auch eine Eprombank auf den Markt.

Wir haben eine solche bereits bestellt und werden Ihnen in der nächsten Ausgabe auch über diese sinnvolle Ergänzung einen Testbericht liefern. Auch sollten sich Interessierte nicht unseren im nächsten C-16 Sonderheft beginnenden Kurs „Eprombrennen - leicht gemacht" entgehen lassen .

Endlich ist er da, der Eprombrenner für den Plus 4.

Der Epromer wird am User-Port betrieben. Die Platine des Prommers

Eprombrenner für den Plus 4

wird in den Expansionsport des Commodore Plus 4 gesteckt. Die Software befindet sich auf den mitgelieferten Disketten. Nach dem Einschalten und Starten kann der Benutzer mit Hilfe eines kleinen Menüs zwischen drei Eprom-Typen wahlen:
A - 2764,
B - 27128,
C – 27256.
Geben Sie nun den gewünschten Typ ein, so gelangen Sie in das Hauptmenü.
Im zweiten Menü stehen folgende Punkte zur Auswahl:

1. Programm beenden,
2. Sprung in den Monitor,
3. Eprom -Typ ändern,
4. Eprom - ganz lesen,
5. Eprom - teilweise lesen,
6. Eprom - Inhalt prüfen,
7. Eprom - programmieren.

Der Punkt 2 erlaubt das Laden und Speichern der Programme die gebrannt werden sollen. Es steht der Bereich von $ 4000 bis $ BFFF zur Verfügung.
Das Arbeiten im Monitor des Brenners entspricht dem des integrierten Monitors des Plus 4.

Auch ausgelesene Programme aus schon gebrannten Eproms stehen in diesem Bereich. Sie können verändert, abgespeichert oder direkt gebrannt werden. Der Punkt 3 des Menüs erlaubt eine Änderung des Eproms. Dieses kann erforderlich werden, wenn der ausgelesene Typ ein anderer ist als der, der gebrannt werden soll. Die Punkte 4 und 5 (lesen) unterscheiden sich kaum. In Punkt 4 werden Eproms komplett gelesen und können dann editiert, abgespeichert oder gebrannt werden. Bei Auswahl des Punktes 5 erscheint ein Untermenü mit der Abfrage nach der Startadresse und der Endadresse (Adresse $0000 / $4000).
Der Punkt 6 des Hauptmenüs erlaubt es, den Speicherinhalt mit dem des Eproms zu vergleichen. Stimmt ein Byte nicht überein, zeigt der Rechner dieses mit Hinweis auf die entsprechende Adresse an.
Der Punkt 7 ist der wohl wichtigste Punkt des Menüs. Wenn Sie diesen Punkt anwählen, sollte das zu brennende Programm an der richtigen Stelle im Rechner stehen. Es erscheint die Abfrage nach dem Algorithmus (normal oder schnell).

Es hat sich erwiesen, dass sich der schnelle Algorithmus bei den meisten Eproms als der Bessere erweist.
Beachtet werden muss unbedingt die Programmierspannung, welche durch den auf dem Brenner befindlichen Schalter auf 12,5 V (rot) oder 21 V geschaltet werden kann.
Diese Spannungen und Algorithmen dürften für die meisten heute verwendeten Eproms genügen.
Der Brenner ist hinreichend menügesteuert, so dass hier keine Probleme mit dem Brennen auch eigener Programme auftreten dürften. Die Beschreibung ist knapp und sehr verständlich. Leider wurde an einem Gehäuse, welches diesem Gerät gut stehen würde, gespart, so dass der Gesamteindruck etwas darunter leidet.
Auch ist die Verarbeitung des Brenners leider alles andere als sauber, was dieses ansonsten gute Gerät doch erheblich für den gehobenen Einsatz abwertet. Wer allerdings über genügend Platz hinter seinem Plus 4 verfügt und schon immer auf einen Brenner gewartet hat, ist mit diesem Gerät nicht schlecht bedient.

Volker Harms

Eproms-Chips mit Pfiff!

Quelle: Compute mit Sonderheft 5/88

Nachdem wir von vielen Lesern angesprochen und gebeten wurden, einen Kurs über das Eprombrennen zu veröffentlichen, haben wir diese Idee nun aufgegriffen. In drei Folgen können Sie sich nun ausführlich zu diesem Thema informieren. Dabei werden auch Hardware-Informationen geliefert, die das Verstehen der Zusammenhänge erleichtern.

1.0 Allgemeines

Die Bezeichnung EPROM steht als Abkürzung für "Erasable Programmable Read Only Memory”. Es handelt sich also um einen NUR-LESE-SPEICHER (ROM). Im ROM befindet sich das Betriebssystem, im PLUS 4 die eingebaute Software und allgemein die sogenannte Modul-Software.

Der entscheidende Vorteil der ROMs besteht darin, dass sie beim Abschalten der Betriebsspannung des Rechners nicht ihren Inhalt verlieren. Diesem Umstand ist es zu verdanken, dass der Computer bei Inbetriebnahme stets mit der gleichen Einschaltmeldung und blinkendem Cursor seine Bereitschaft signalisiert.

Der eigentliche Speicher der ROMs ist ein kleiner Mikrochip, der bei EPROMs durch das Fenster gut sichtbar ist. Wegen der Sicherheit und der einfacheren Handhabung ist dieser Chip auf einen keramischen Träger montiert. Das Fenster dient zum Löschen der EPROMS, aber darüber später mehr.

Eprombrenner für den Plus 4

1.1 EPROM-Typen und Bezeichnungen

Leider ist die Beschriftung der EPROMs nicht einheitlich und manche wichtige Information ist verschlüsselt dargestellt. Die gebräuchlichsten EPROMs. sind die Typen der 27er Reihe:
2764 64 KBit: 8 KByte Speicher
27128 128 KBit : 16 KByte Speicher
27256 256 KBit : 32 KByte Speicher
Für die Zugriffszeit gibt es verschiedene Codes. Die gängigste Bezeichnung ist
“M” für 250ns (Nanosekunden). Aber auch
"B” für 450 ns,
"C” für 350 ns und
"D” für 250 ns
sind oft verwendete Bezeichnungen.

Der wohl wichtigste Punkt, die Programmierspannung, ist bei manchen Herstellern nur dem (selten greifbaren) Datenblatt zu entnehmen. Allerdings sind heute überwiegend EPROMs mit 12,5 V Programmierspannung im Handel. Auf manchen EPROMs ist die erforderliche Programmierspannung auch direkt angegeben. Eine gängige Bezeichnung für 12,5 V ist auch "F1".

1.2 Die Pinbelegung

EPROMs der 27er Reihe haben 28 Füße (Pins), die wie folgt nummeriert sind.: die Kerbe ist oben, links oben von der Kerbe ist Pin 1. Von Pin 1 wird nach unten weitergezählt bis Pin 14, Unten rechts ist Pin 15 und oben rechts von der Kerbe ist Pin 28. Die Kerbe ist wichtig für die Einbaurichtung. Falsches Einsetzen kann zur Zerstörung des EPROMs führen!
Die Pinbelegung im direkten Vergleich der Typen 2764 bis 27256 ist in Skizze 1 dargestellt. Die Abkürzungen haben dabei folgende Bedeutung:

VCC = +5 V Betriebsspannung
GND = Masseanschluss bzw. gemeinsamer Minuspol
VPP = +12,5V oder 21V Programmierspannung
A0-A14 = Adressleitungen
01 -07 = Datenleitungen
PGM = Programmierimpuls Eingang
OE = Eingang zum Durchschalten der Daten
CE = Eingang zum Aktivieren des EPROMs
NC = nicht belegt

Am Expansionsport des C-16 bzw. PLUS 4 (siehe Skizze 2) entsprechen die Bezeichnungen wie folgt der EPROM Pinbelegung:

Exp.-Port :	EPROM
25	GND
J	A13
L	A11
N	A9
R	A7
T	A5
V	A3
X	A1
14	07
16	05
18	03
20	01
10	OE
2	VCC
K	A12
M	A10
P	A8
S	A6
U	A4
W	A2
Y	A0
15	06
17	04
19	02
21	00
B	CE

Schaltungsvorschläge hierzu werden im nächsten Kapitel angeboten.

2.0 Eproms und Computer – Zusammenspiel

Sowohl der Plus/4 als auch der C-16/176 ist für den Betrieb externer ROMs vorgesehen. Im PLUS 4 können diese unter anderem gegen die eingebaute Software ausgetauscht werden. Beide Rechnertypen eignen sich zum Betrieb von exROMs auf Modulplatinen am Expansionsport.

Wer aber Mut genug hat, kleine Eingriffe in seinem Rechner vorzunehmen, der hat sehr interessante Möglichkeiten, Module (externe ROMs) im Rechner zu betreiben.

2.1 Rechnerinterne Schaltlogik

Wenn man den Schaltplan des C-16 mit dem des PLUS/4 vergleicht, wird man feststellen, dass die ROMs durch ein IC mit der Bezeichnung 74LS139 geschaltet werden. Im PLUS/4 hat dieses Bauteil die Kennung U20 (im C-16 U14). Dieses IC hat 16 Pins, von denen im C16 sieben und im PLUS 4 neun Pins belegt sind. In der Folge bezeichne ich diesen Baustein nur noch mit "U20”. Die Nummerierung der Pins ist wieder wie beim EPROM links von. der Kerbe mit 1 beginnend.

U20 schaltet die ROMs wie folgt:

Pin 4 : BASIC-ROM (U23 im PLUS 4, U3 im C16) an Pin 22
Pin 12 : KERNAL (U24 im PLUS 4, U4 im C16) an Pin 22
Pin 6 : C1 Low Expansionsport Pin B
Pin 10: C1 High Expansionsport Pin 6
Pin 7 : C2 Low Expan-

Eprombrenner für den Plus 4

sionsport Pin 7
Pin 9 : C2 High Expan-
sionsport Pin 8
Pin 5 : Eingeb.Software
U25 (Low) an Pin 22 (nur
PLUS/4)
Pin 11: Eingeb.Software
U26 (High) an Pin 22 (nur
PLUS/4

Wenn nun z.B. am Expansionsport das SCRIPT PLUS-Modul betrieben wird, schaltet U20 nach Aufruf des Moduls von Pin 6 (U20) das Low-Eprom an Pin 22 und dann das High Eprom von Pin 10 (20) an Pin 22 ein.

2.2 EPROMs im C-16 bzw. PLUS 4

Da die Pins der eingebauten ROMs (Basic-Rom und Kernal) kompatibel zu den Pins der EPROMs sind, lassen sich EPROMs relativ einfach in den Rechner einbauen. Mit einem für Elektronikarbeiten geeigneten Lötkolben kann der interessierte Anwender auf die ROMs (U23/U24 bzw. U3/U4) 28-polige Sockel rückwärtig (huckepack) auflöten, wobei lediglich die Pins 22 der Sockel nicht mit den Pins 22 der ROMs verbunden werden. Die Pins 22 werden herausgebogen, um daran die Schaltleitungen von U20 anzuschließen. Im C-16 bieten sich für die Einschaltung die Pins fünf und elf für das LOW- und das High EPROM an. Es versteht sich von selbst, dass die ROMs beim Löten der Fassungen vorsichtig entnommen werden, jetzt könnte im C16 z.B. die im PLUS 4 eingebaute Software betrieben werden. Die Einschaltmeldung ist identisch mit der des PLUS 4: 3-PLUS-1 on
KEY F1.

Sofern beim Huckepackverfahren die Höhe ausreicht, könnten nun auf den ersten EPROM-Satz weiter Sockel aufgelötet werden.

Im PLUS 4 sind von U20 die Pins 5 und 11 -bereits durch die eingebaute Software belegt. Deshalb werden dort die Schaltbeine (Pin 22) der ersten beiden Sockel mit Pin 6 (für Low) und Pin 10 (für High) von U20 verbunden. Dies gilt auch für den zweiten Huckepack-Satz im
C-16.

2.3 Abschaltmöglichkeit

Modulsoftware, die über C1, also Pin 6 und 10 von U20, betrieben wird, belegt zum Einschalten die Funktionstaste F2.

Bei Diskettenbetrieb ist aber die Belegung mit "DLOAD" sehr nützlich und sollte auch erhalten bleiben. Dies lässt sich sehr einfach dadurch erreichen, dass zwischen Pin 28 des Sockels und Pin 28 des EPROMs ein Miniaturschalter (1xUM) gesetzt wird. Pin 28 des Sockels und Pin 28 des EPROMs werden nicht in den Sockel gesteckt, sondern herausgebogen. Der mittlere Pol des Umschalters wird per Kabel mit Pin 28 des EPROMs, und einer der 4 äußeren Pole des Schalters mit Pin 28 des Sockels verbunden. Beim High EPROM wird dann nur Pin 28 des EPROMs herausgebogen und mit Pin 28 des Low EPROMs verbunden. Schaltskizze ist Bild 3. Das bisher gesagte gilt für EPROMs 27128.

2.4 Größere EPROMs -mehr Kapazität

Der C16 bzw. PLUS 4 verwaltet nur ROMs bis maximal 32 KB, aufgeteilt auf ein LOW EPROM und ein HIGH-EPROM (= 2 x 27128). Wenn nun alle Steck- und Schaltmöglichkeiten intern genutzt werden, bieten die Rechner Platz für 3 Module mit 32 KB. Mit einem kleinen Trick lässt sich diese Kapazität verdoppeln: durch Verwendung von EPROMs 27256 (32 KB). Wenn wir die Pinbelegung der Typen 27128 und 27256 in Skizze 1 vergleichen, stellen wir fest, dass Pin 27 unterschiedlich belegt ist. Einmal ist Pin 27 als Programmier-impuls Eingang und einmal als Adressleitung A14 gekennzeichnet. Der Rest ist identisch, Durch eine kleine Manipulation an Pin 27 kann das EPROM 27256 in. 2 EPROMs 27128 aufgeteilt werden: das Potential wird verschoben.
In der Praxis sieht das so aus: Liegt an Pin 27 +5 V an, erkennt der Rechner die "obere" Hälfte des EPROMs als "sein 128er", liegt an Pin 27 Masse an, erkennt der Rechner die "untere" Hälfte. Nach diesem Prinzip arbeitet auch die EPROM-Bank der Fa. BRAUNROTH, die auf 12 Steckplätzen Platz für insges. 392 KB Programme bietet. Das entsprechende Schaltbild ist in Skizze 4 dargestellt. Zum Schalten wird ein Miniatur Kippschalter 2x UM mit Mitte-Null-Stellung benötigt. In Mittelstellung ist das Modul abgeschaltet und die F-Tasten sind wieder frei.

3.0 Modulsoftware

Das Angebot an Modulsoftware für den C16 bzw. PLUS 4 ist nicht berauschend. SKRIPT PLUS, CALC PLUS, TURBO PLUS, 3-PLUS-1 in Englisch und in Deutsch, und damit ist das Angebot schon erschöpft. Was bleibt, ist die Entwicklung eigener Module, wobei es jedem Anwender selbst überlassen ist, welche Programme er als Modul betreiben möchte. Jeder User kennt das Problem: bestimmte Programme werden häufig benötigt und die Disketten mit ihnen sind nicht zu finden. Dann stört, selbst wenn das entsprechende Programm kurz ist, schon das Laden und Starten und man wünscht, gewisse Programme sollten auf Tastendruck bereitstehen. Das ist der Ansatzpunkt für eigene Module.

3.1 Autostartmodule

Eprombrenner für den Plus 4

Die ideale Ergänzung zur Diskette sind sogenannte AUTOSTART-MODULE. Hierbei muss das Modul nicht erst durch Drücken einer F-Taste initialisiert werden, sondern es wird beim Einschalten des Rechners oder bei RESET aktiviert. Vorausgesetzt natürlich, das Modul ist nicht ausgeschaltet.
Mit der Inbetriebnahme des Computers (oder bei RESET) erscheint auf dem Bildschirm nicht die gewohnte Einschaltmeldung, sondern ein Menü, in dem die auf dem Modul befindlichen Programme aufgelistet sind. Außerdem wird gezeigt, mit welchen Tasten die Programme aufgerufen werden. Natürlich gehört dazu auch ein Menüpunkt "AUTOSTARMODUL VERLASSEN", um wieder in den normalen Modus zu gelangen.

3.2 MODUL-RESET

Beim Einschalten des Rechners und beim Drücken der RESET-Taste wird ein MODUL-RESET durchgeführt. Dabei werden nacheinander die Module 15, 10, 5 und 0 eingeschaltet. Dann wird in dem jeweils eingeschalteten Modul in den Adressen $8007 bis $8009 nach der Kennung "CBM" gesucht. Ist die Kennung vorhanden, wird aus der Adresse $8006 die Modul Nummer gelesen. Das Basic ROM z.B. hat die Modulnummer 00 und die 3-PLUS-1- Software die Nr. OC. Hat die gelesene Modul-Nummer den Wert 1, wird dieses Modul gestartet. Ein einfacher AUTOSTART Kopf könnte etwa so aussehen:

8000 4C 0A 80 00 00 00 01 42
8008 43 4d

in Adresse $8000 erfolgt der absolute Sprung ($40 = JMP) an die eigentliche Startadresse des Programms, in diesem Fall $800A. In $8007 bis $8009 sind die C (=$42), B (=$43), M (= $4D) -Kennung enthalten, die ein Modul erst als solches erkennen lässt. in $8006 steht die Modulnummer 01. Dieser Kopf würde sich nur für ein größeres Maschinenprogramm eignen. Für ein AUTOSTART-Menü fehlen noch ein paar Informationen.
Dazu später mehr.

3.3 SPEICHERKONFIGURATION und EINSCHALTADRESSEN

Die Tabelle zeigt die möglichen Speicherkonfigurationen mit Einschaltadressen. Die Umschaltlogik im C16 bzw. PLUS 4 erlaubt es, aus diesen 16 Kombinationsmöglichkeiten die gewünschte ROM-Konfiguration aufzurufen. Beispiel: der Anwender hat ein eigenes KERNAL entwickelt, möchte dieses aber nicht gegen das Original austauschen, sondern am Expansionport über C1 betreiben. Es soll nun also das BASIC-ROM zusammen mit dem neuen KERNAL eingeschaltet werden. Diese Konfguration entspricht der Nr. 8 der Tabelle ($FDD8). Das folgende kleine Programm würde die Einschaltung realisieren:
Siehe Beispiele

```
Beispiel 1:
>1000 A2 05 LDX #$08       : Nummer der Konfig. in X-Register
>1002 9D DO FD STA $FDD0,X : Konfig.einschalten (+ X-Reg.)
>1005 86 FB STX $FB        : gefundene Nummer in $FB ablegen
>1007 4C 00 80 JMP $8000   : Modul initialisieren

Beispiel 2:
>1000 A9 03 LDA #$03       : LOW-Byte der Startadresse
>1002 A0 80 LDY #$80       : HIGH-Byte der Startadresse
>1004 8D FE 02 STA $02FE   : Startadresse LOW
>1007 8C FF 02 STY $02FF   : Startadresse HIGH
>100A A2 05 LDX #$05       : Modulnummer (siehe Tabelle)
>100C 86 FB STX SFB        : Modulnummer in $FB eintragen
>100E 4C C9 FC JMP $FCC9   : Modul einschalten und starten
```

BEFEHL	LOW ($8000 - $BFFF):	HIGH ($C000-$FFFF):
.STA $FDD0	BASIC-ROM	KERNAL-ROM
.STA $FDD1	EING.SOFTW.LOW PLUS/4	KERNAL-ROM
.STA $FDD2	EXT. SPEICHER LOW (C1)	KERNAL-ROM
.STA $FDD3	EXT. SPEICHER LOW (C2)	KERNAL-ROM
.STA $FDD4	BASIC-ROM	EING.SOFTW.HIGH PLUS/4
.STA $FDD5	EING.SOFTW.HIGH PLUS/4	EING.SOFTW.HIGH PLUS/4

Eprombrenner für den Plus 4

.STA $FDD6	EXT. SPEICHER LOW (C1)	EING.SOFTW.HIGH PLUS/4
.STA $FDD7	EXT. SPEICHER LOW (C2)	EING.SOFTW.HIGH PLUS/4
.STA $FDD8	BASIC-ROM	EXT. SPEICHER HIGH (C1)
.STA $FDD9	EING.SOFTW.LOW PLUS 4	EXT. SPEICHER HIGH (C1)
.STA $FDDA	EXT. SPEICHER LOW (C1)	EXT. SPEICHER HIGH (C1)
.STA $FDDB	EXT. SPEICHER LOW (C2)	EXT. SPEICHER HIGH (C1
.STA $FDDC	BASIC-ROM	EXT. SPEICHER HIGH (C2)
.STA $FDDD	EING.SOFTW.LOW PLUS 4	EXT. SPEICHER HIGH (C2)
.STA $FDDE	EXT. SPEICHER LOW (C1)	EXT. SPEICHER HIGH (C2)
.STA $FDDF	EXT. SPEICHER LOW (C2)	EXT. SPEICHER HIGH (C2)

Mit diesem Beispiel wird die achte Möglichkeit aus der Tabelle eingeschaltet. Von $8000 bis $BFFF liegt das BASIC ROM und von $C000 bis $FFFF das eigene Kernal. Übrigens, das Betriebssystem des PLUS 4 (und C-16) gibt zum Einschalten und Starten der ROMs einige Hilfen. Bei $FCC9 (DEC 64713) beginnt eine Routine, die das Modul einschaltet, dessen Nummer im X-Register steht. Dann wird ein Sprung zu der Adresse ausgeführt, dessen LOW-Byte in $02FE und dessen HIGH Byte in $02FF steht. Dies sollte die Startadresse sein. Mit dem zweiten Beispiel wird die eingebaute Software des PLUS 4 gestartet. Diese kleinen Beispiele können natürlich für eigene Zwecke abgeändert werden. Sie sollen dem interessierten Anwender lediglich einen Hinweis auf die Vielzahl der Möglichkeiten bieten, über die der PLUS 4 verfügt. Mehrfach ist jetzt die Adresse $FB aufgetaucht. Diese Adresse enthält immer die aktuelle Speicherbankkonfiguration. Das Interrupt-Programm wird jede 6Ostel Sekunde abgearbeitet. Nach dem Interrupt-Programm wird die Speicherkonfiguration eingeschaltet, deren Nummer in $FB steht. Normalerweise ist dies der Wert $00 (siehe Tabelle: BASIC-ROM und KERNAL ROM). Würde in den Beispielen der Wert in $FB nicht geändert, würde der nächste Interrupt wieder das normale Betriebssystem einschalten.

Literaturhinweis: ALLES ÜBER DEN PLUS 4 UND ROM-LISTING PLUS 4. Beide von Markt und Technik.

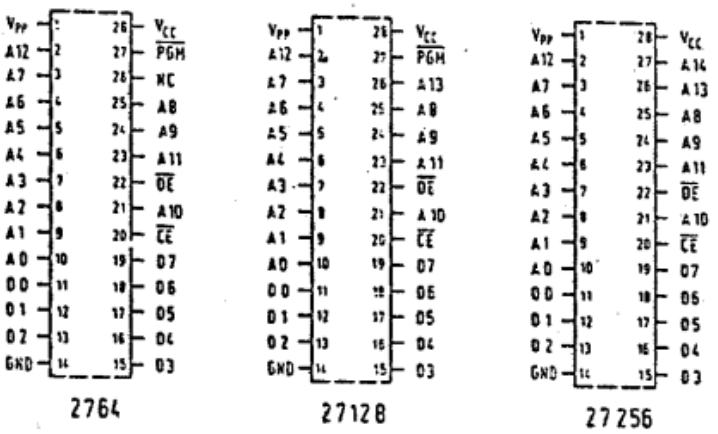
Heinz-Werner Schrimpf

Eprombrenner für den Plus 4

SKIZZE 1

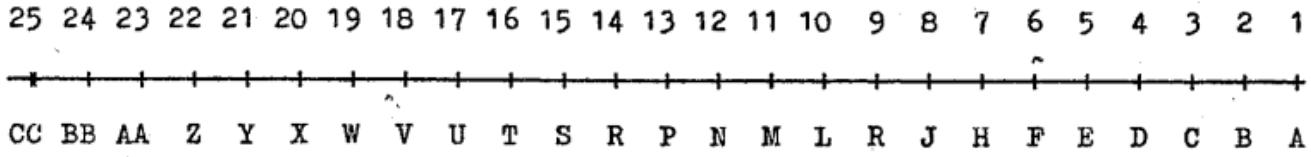
Schaltungsvorschläge

Pinbelegung
im Vergleich



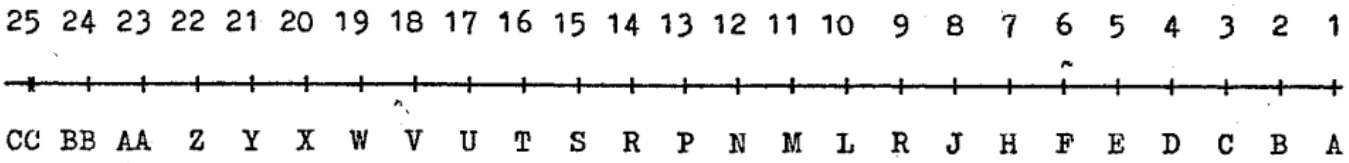
SKIZZE 2

Belegung Expansionsport

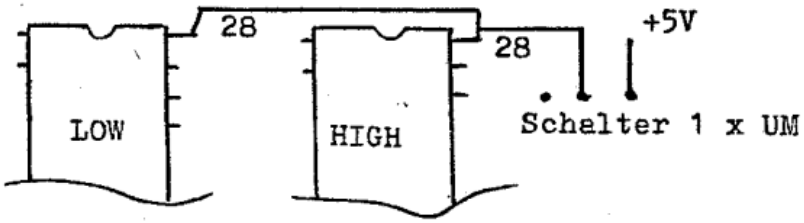


Eprombrenner für den Plus 4

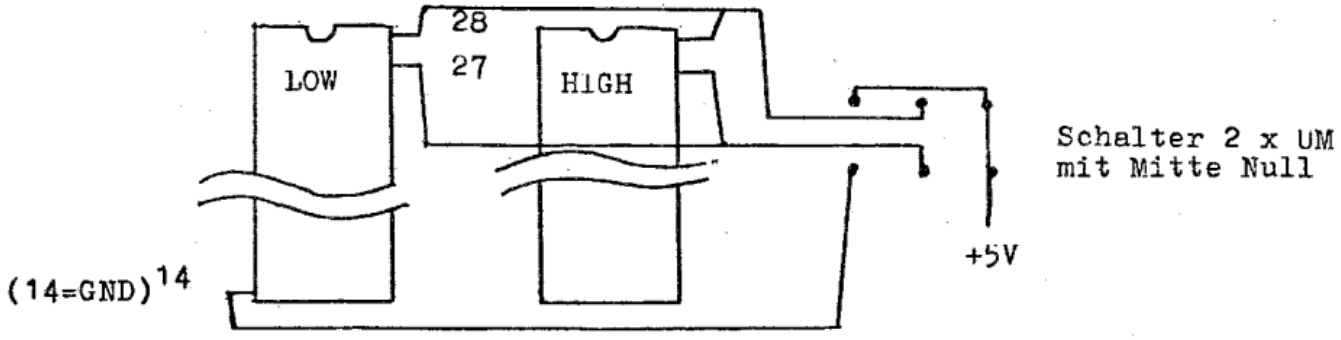
SKIZZE 2 Belegung Expansionsport



SKIZZE 3 Modul abschalten



SKIZZE 4 Um- und Abschalten EPROMs 27256



Quelle: Artikel aus Compute mit Sonderheft 6/88

Nachdem wir Sie im ersten Teil dieses Kurses in die Grundlagen eingeführt und die Möglichkeiten aufgezeigt haben, erhalten Sie in der Fortsetzung ausführliche Hinweise zur Modultechnik. Diese Hinweise können als Grundlage verwendet werden, um eigene Module zu programmieren.

Ich hoffe, der erste Teil dieses Kurses hat Ihr Interesse geweckt und Sie warten gespannt auf die Möglichkeit, sich ausführlicher mit der Modultechnik zu befassen und eigene Module zu entwickeln.

4.0 Programmieren eigener Module

Das folgende Kapitel befasst sich mit dem Programmieren eines AUTO-START-MODULS, wie es bereits kurz angeschnitten wurde. Derzeit wird zum "Brennen" von. EPROMs für den PLUS 4 nur ein Eprommer von der Fa. Jürgen Braunroth (Adresse: Moordorfer Str. 30, D-3057 Neustadt 1) angeboten. Daher wird in diesem Artikel bei den Ladeadressen der Programme auf diesen Eprommer Bezug genommen. Natürlich können die eigenen Module auch mit einem Eprommer am C-64 gebrannt werden.

4.1 Grundlagen

Wenn ein 32 KB-Modul geladen ist, liegt der LOW-Teil von $8000 bis $BFFF und der High-Teil von $C000 bis $FFFF. Das heißt aber nicht, dass dieser Bereich komplett für eigene Programme zur Verfügung steht. In allen Konfigurationen ist im Bereich $FC00 bis $FF3F immer das interne ROM und der I/0-Bereich ein-

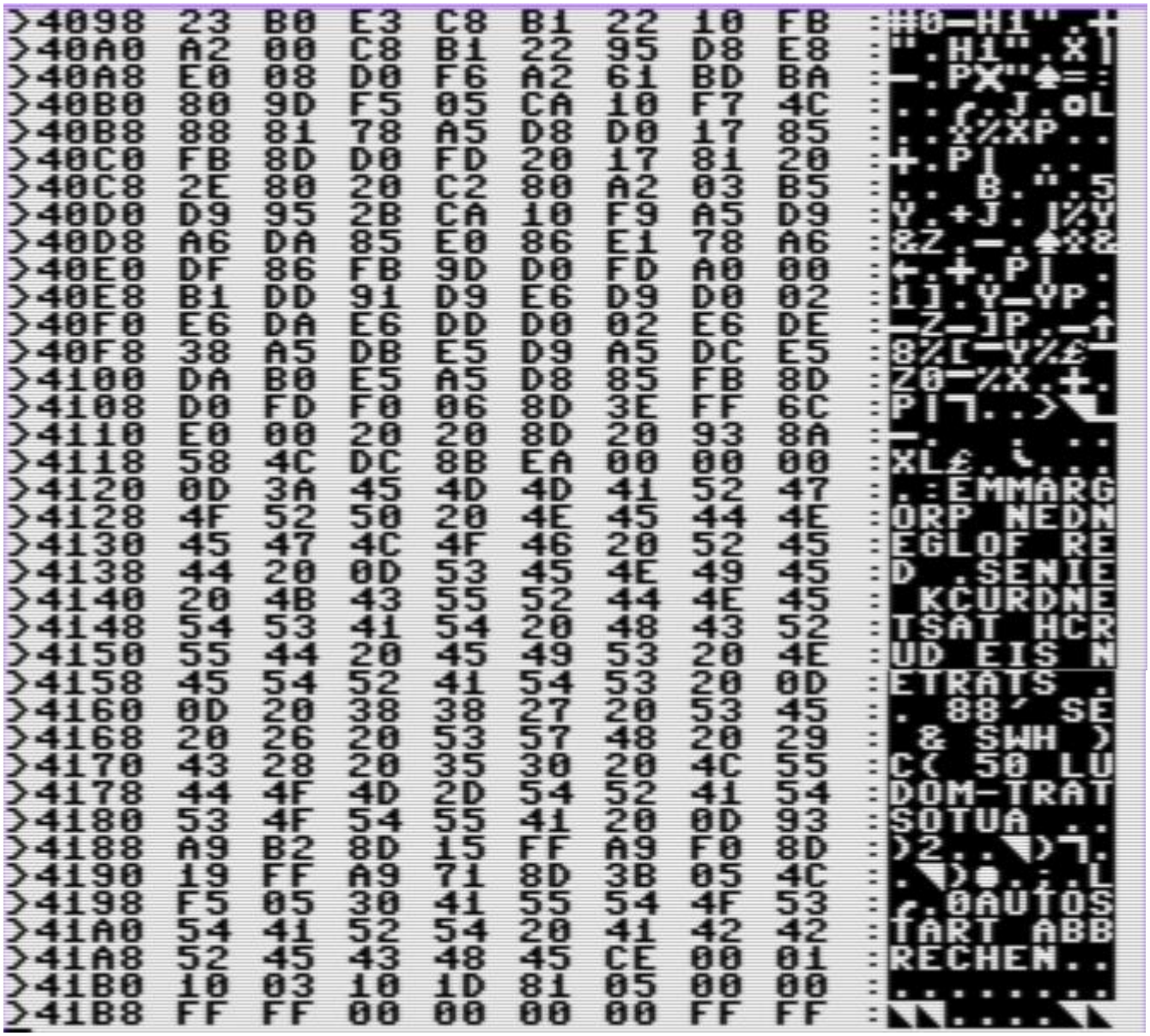
Eprombrenner für den Plus 4

geblendet. Der 0-Bereich erstreckt sich von $FD00 bis $FF3F (DEC 64768 bis 65343) und enthält unter anderem die TED-Register, den R6522-Port und den PLUS 4-User-port. Der feste ROM-Bereich von $FCFF (DEC 64512 bis 64767) enthält die sogenannten Banking-Routinen, mit denen Zugriffe und Sprünge auf andere ROM-Bänke ohne Umweg über das RAM ermöglicht werden. in Kapitel 3.2 wurde bereits darauf hingewiesen, dass bei einem RESET ein Modul mit der Kennziffer "21" vorrangig aufgerufen wird. So einfach wie in dem kurzen Beispielprogramm ist der AUTO-START natürlich nur bei Maschinenprogrammen, wobei noch zu beachten ist, dass hier noch nichts im Rechner initialisiert war. Um also mit dem Rechner arbeiten zu können, müssen zuerst die benötigten Routinen im KERNAL aufgerufen werden (z.B. I/O- INIT, VECTOR INIT etc.). Es würde diesen Rahmen aber sprengen, wenn darauf näher eingegangen würde. Um Ihnen aber die Benutzung vor EPROMs schmackhaft und die Anwendung von AUTO-START-MODULEN möglich zu machen (auch wenn Sie mit der Maschinensprache nicht vertraut sind), haben wir einen Programmvorspann für AUTO-START-MODULE entwickelt, den Sie hier einmal als HEXDUMP und einmal als kommentiertes ASSEMBLERLISTING vorfinden. Dieser Vorspann initialisiert zuerst den Computer wie üblich und stellt dann in einem Menü mehrere Programme ihrer Wahl (eigene BASIC- oder Maschinenprogramme) zur Verfügung, welche auf Tastendruck gestartet werden. die Treibersoftware des PLUS 4 Eprommers stellt für die brennfertigen Programme den Bereich von $4000 bis $BFFF zur Verfügung. Da die zu brennenden Programme im Monitor geladen werden, muss die Ladeadresse $4000 sein. Wenn also Programme zu einem Modul zusammengestellt werden, sollte das Kopfprogramm bei $4000 beginnen. Modulkopfprogramm und mögliche Änderungen.

Hier das kommentierte Assembler Listing. Der Ausdruck ist nicht komplett, sondern umfasst nur die Bereiche, deren Änderungen interessant sein können. Ein komplettes Assembler Listing kann der interessierte Anwender selbst ausdrucken.

Hier der Hexdump der Software ab $4000-$41b8

Eprombrenner für den Plus 4



Disassembled sieht das so aus.

```
>4000   78         SEI           : IRQ unterdrücken
>4001   48         PHA           : Rücksprungadresse vom Stapel löschen
>4002   48         PHA           : „
>4003   4C 0A 80   JMP $800A     : Startadresse Modulprogramm
>4006   01         -             : Modulnummer
>4007   43 42 4D   -             : Kennung „CBM“
>400A   8D D1 FD   STA $FDD1     : ROM-Konfig hier anstatt eingeb. Software
>400D   20 0B F3   JSR $F30B     : I/O-INIT
>4010   20 52 F3   JSR $F352     : RAMTAS
>4013   20 CE F2   JSR $F2CE     : RESTOR
>4016   20 4E D8   JSR $D84E     : EDITOR-RESET
>4019   A9 01      LDA #$01      : ROM-Konfig anstatt eingeb. Software
>401B   85 FB      STA $FB       : Schreibt 01 nach $FB

>401E   A2 6A      LDX #$6A      : Anzahl Zeichen im Kopf
>4020   BD 20 81   LDA $8120,X   : Kopf mit „X“ Zeichen ausgeben
```

Eprombrenner für den Plus 4

```
>4076   A9 9A      LDA #$9A      : LOW-Byte Anfang Menüliste
>4078   A2 B1      LDX #$81      : HIGH-Byte Anfang Menüliste

>40B7   4C 88 81   JMP $8881     : Sprung zum Farbwechsel

>4188   A9 B2      LDA #$B2      : Hintergrundfarbe rot
>418A   8D 15 FF   STA $FF15
>418D   A9 F0      LDA #$F0      : Rahmenfarbe schwarz
>418F   8D 19 FF   STA $FF19     :
>4192   A9 71      LDA #$71      : Zeichenfarbe weiß
>4194   8D 3B 05   STA $053B
```

Wenn dieses Modul nicht an Stelle der im PLUS 4 eingebauten Software betrieben werden soll, sondern über C1 entweder eingebaut (siehe Kapitel 2.2) oder am Expansionsport betrieben werden soll, müssen folgende Adressen geändert werden:
>400A 8D D2 FD :STA $FDD2
:hier D2 anstatt D1
>4019 A9 02 :LDA #$02
:hier 02 anstatt 01

individuell auf das entsprechende Programm bezogen. Diese individuellen Daten setzen sich wie folgt zusammen und werden direkt im Anschluss an das um $80 erhöhte letzte Textzeichen gesetzt:

1.) Programmkennzeichnung

Hier wird angegeben, ob das aufzurufende Programm ein BASIC-Programm oder ein Maschinenprogramm ist. Für das BASIC-Programm steht "00", für das Maschinenprogramm "01". Alle Programme, die mir RUN gestartet werden, sind als BASIC-Pragramme zu kennzeichnen. Dies gilt auch für Maschinenprogramme mit BASIC-Start!

2.) Ladeadresse

Als Ladeadresse wird die Adresse angegeben, an der das Programm nach dem Laden anfängt. Bei Programmen ist das $ 1001. Da es sich um einen 16-Bit-Wert handelt, wird er aufgeteilt in LOW- und HIGH-Byte. Zuerst steht das LOW- und dann das HIGH-Byte, bei BASIC. Programmen also zuerst “01” und dann "10", BASIC-Programme werden nach Aufruf bei $1001 automtisch mit RUN gestartet. Bei Maschinenprogrammen muss die Ladeadresse identisch mit der Startadresse sein. Wenn dies in einem Maschinenprogramm nicht vorgesehen ist, muss das Programm um 3 Bytes ergänzt werden, die vor das eigentliche Programm gesetzt werden. Wenn z.B. die Startadresse bei $3000 liegt, kann mit dem Sprungbefehl "4C" (JMP), "00" (LOW-Byte der Startadresse), "30" (HIGH Byte der Startadresse) vor dem eigentlichen Programm dem Rechner gezeigt werden, wo das Programm beginnt. Hinweis: die Fa. BRAUNRORTH vertreibt eine Diskette mit Hilfsprogrammen zum Programmieren von EPROMs. Darunter befindet sich ein Ebenso muss dann die Modulnummer nach jedem Menüeintrag von 05 in 0A umgeändert werden. Das gilt auch schon für den ersten Eintrag “AUTOSTART ABBRECHEN"!

4.2 Modulkopf um Menüeinträge ergänzen

Der Autostartkopf als solcher ist nun funktionsfähig. Was fehlt sind die Menüeinträge. Der erste Eintrag ist bereits vorhanden: AUTOSTART ABBRECHEN”. Wenn Sie sich das Ende des HEXDUMPs ansehen, können Sie im rechten Teil den Klartext des Eintrags erkennen. Der Eintrag beginnt in $419A mit $30 (=ASCII 0). Wird Taste "0" gedrückt, wird der Autostart abgebrochen. Das letzte Zeichen des Texteintrages wurde um $80 erhöht (N = $4E + $80 = $CE). Daran erkennt der Rechner, dass der eigentliche Eintrag hier beendet ist. Die weiteren Daten hinter dem Texteintrag sind COMPACTOR, der Maschinenprogramme komprimiert und die Startadresse so einbaut, da jedes Maschinenprogramm wie ein BASIC-Programm gehandhabt werden kann.

3.) Endadresse

Dazu muss das entsprechende Programm geladen werden. Um aber eindeutig festzustellen, wo das Programm endet, sollte der Speicher vorher mit “00” gefüllt werden
(MONITOR: F 1000 7FFF 00).
Die Endadresse wird dann um 1 erhöht. Geschrieben wird wieder zuerst das LOW-Byte und dann das HIGH-Byte.

4.) Anfangsadresse im EPROM

Wie jeder Speicher hat auch ein EPROM eine fortlaufende

```
EPROM 2764   $0000-$1FFF
EPROM 27128  $0000 $3FFF
EPROM 27256  $0000-$7FFF
```

Eprombrenner für den Plus 4

Adressierung. Die Anfangsadresse im EPROM ist die Adresse, an der das Programm beginnt. Da das EPROM als ROM geladen wird, muss zu der Adresse im EPROM noch $8000 addiert werden. Beispiel: Der AUTO-START-KOPF beginnt im EPROM bei der Adresse $0000. Das entspricht also der Adresse $8000. Unter Berücksichtigung einer gewissen Anzahl an Menüeinträgen sollten für den Kopf 767 Byte . reserviert werden. Das Kopfprogramm belegt dann den Bereich von $0000 bis $02FF im EPROM (entspricht $8000 bis $82FF). Das erste Programm kann bei $0300 im EPROM beginnen.
$8000 zu diesem Wert addiert, ergibt den gefragten Wert von $8300. Dieser wird wieder aufgeteilt in LOW-Byte und HIGH-Byte. Wenn das zu erstellende Modul aus LOW- und HIGH-EPROM besteht, muss zur EPROM-Adresse im HIGH-EPROM $C000 addiert werden.

5.) Modulnummer

Die Modulnummer wird der Liste 3.3 entnommen, Entscheidend dafür ist, wo das Modul betrieben werden soll. Soll es z. B. im PLUS 4 an Stelle der eingebauten Software genutzt werden, ist die Modulnummer "05" , über C1 am Expansionsport oder auch in den Rechner eingebaut ist dies entsprechend "0A" . Die Modulnummer schließt jeden Eintrag ab.

6.) Menüende

Wenn alle Einträge erfolgt sind, genügt es, wenn nach der Modulnummer im letzten Eintrag der Wert “00” hinzugefügt wird. Damit erkennt der Rechner, dass keine weiteren Einträge mehr erfolgen. Natürlich beginnt jeder Menüeintrag mit der Bezeichnung der Taste, mit der das Programm aufgerufen werden soll.

Quelle: Compute mit Sonderheft 1/89

In dieser Ausgabe schließen wir den Kurs »Eprombrennen« ab. Die letzten Geheimnisse werden enthüllt. Gleichzeitig starten wir einen neuen Kurs: Die Programmierung in Assembler!

4.3 Programme ’brennfertig’ zusammenstellen

Zuerst sollten alle zum Brennen vorgesehene Programme auf Anfang (entfällt bei BASIC-Programmen) und Ende (immer +1) überprüft und die Werte notiert werden. Eventuell sind Maschinenprogramme um die Startadresse zu ergänzen. Diese Vorbereitungen sollten sehr gewissenhaft getroffen werden! Es ist nämlich äußerst schwierig, später im gebrannten Modul einen Fehler zu suchen! Für den weiteren Ablauf bietet der Computer einige sehr sinnvolle Hilfen. So lässt sich mit einem Schreibzugriff auf die Adresse $07F8 das ROM ausschalten. Dazu muss in die besagte Adresse nur anstatt "00" der Wert "80" im Direktmodus eingegeben werden. Jetzt kann der Bereich von $1000 bis $FC00 mit "00" gefüllt werden. Dies empfiehlt sich wegen der besseren Übersicht. Der nächste Schritt ist das Laden des Kopfprogrammes (im Direktmodus nach $4000) Die Stammdaten können nun nach Belieben geändert werden. -siehe Kapitel 4.1 Das erste Programm müsste nun an die Adresse $4300 gebracht werden. Bei BASIC- Programmen bietet sich die Möglichkeit an, in Adresse $2B und $2C den Basic-Anfang entsprechend zu verändern. In $2B wird in diesem Fall "00" und in $2C "43" geschrieben Das Programm kann jetzt mit DLOAD geladen werden und steht an der richtigen Adresse bei $4300. Wem die nun folgenden Umrechnungen zu kompliziert erscheinen, der kann sein Modul auch direkt bei $8000 zusammenstellen. Dort sind dann alle entsprechenden Daten direkt abzulesen. In diesem Fall wird der BASIC-Anfangszeiger anstatt auf $4300 auf $8300 gesetzt. Das Kopfprogramm kann entweder mit dem T-Befehl! (Transfer) nach $8000 verschoben werden oder mit dem auf der 1551- DEMO- Diskette befindlichen Programm "LOAD ADRESS CHANGER” auf Ladeadresse $8000 = dec 32768 abgeändert werden. Der Einfachheit halber bleiben wir bei $8000, dürfen aber später nicht vergessen, die Ladeadresse des kompletten Moduls zum Brennen auf $4000 zu verändern.
Das erste Programm soll z.B. „SUPER-COPY” sein. Normal geladen liegt das Programm von $1001 bis $1AF9 + $1AFA. Bei verändertem BASIC-Anfangszeiger und mit DLOAD geladen befindet es sich nun von $8300 bis $8DF8. Der Kopf wird jetzt um diesen Menüeintrag ergänzt:

```
HEXADEZIMAL:  31 53 55 50 45 52 20 43 4F 50 D9
KLARTEXT   :   1  S  U  P  E  R     C  O  P  Y
            00   01  10   FA  1A   00  83       05
   BASIC-Progr.  Ladeadr. Endadr.  Anf.i.EPROM Modul-Nr.
```

Eprombrenner für den Plus 4

Das nächste Programm ist in diesem Beispiel ” LOAD ADRESS” von der DEMO-Disk. im Modul soll es dann "ADRESS-CHANGER” heißen. Es ist ein BASIC Programm und liegt von $1001 bis $17D2 + 1 = $17D3. Dieses Programm könnte jetzt nach $8DF9 geladen werden. Wegen besserer Übersicht ist es jedoch besser, den Rest der angefangenen Zeile freizulassen und eine neue Zeile zu beginnen. Die nächste Zeile beginnt bei $8E00. Also wird der BASIC-Anfangszeiger auf diese Adresse eingestellt, und das Prgramm wird mit DLOAD geladen. Die Überprüfung im Monitor zeigt, dass bei $95CC das letzte Byte des Programmes liegt. Wenn der Rest der Zeile frei bleibt, beginnt das nächste Programm bei $95D0. Doch zuerst wieder der Eintrag:

```
HEXADEZIMAL: 32 41 44 52 45 53 53 2D 43 48 41 4E 47 45 D2
KLARTEXT   :  2  A  D  R  E  S  S  -  C  H  A  N  G  E  R
          00  01  10   D3  17   00  8E       05
BASIC-Progr. Ladeadr. Endadr. Anf.i.EPROM Modul-Nr.
```

Für das nächste Beispiel nehmen wir an, dass es sich um ein Maschinenprogramm handelt, das bei $2000 beginnt und bei $5200 + 1 = $5201 endet. Die Startadresse beträgt $2500. Dieses Programm ist nur fiktiv und soll zeigen, wie ein MC-Programm gehandhabt wird, von dem zusätzlich noch ein Teil auf dem LOW- und ein Teil auf dem HIGH-EPROM enthalten ist. Weil das Programm um 3 Bytes ergänzt werden muss, beginnt es nicht mehr bei $2000, sondern bei $1FFD. Nach $1FFD werden fortlaufend folgende Werte geschrieben: "4C” (JMP), "00" (LOW-Byte der Startadresse und "25" (HIGH-Byte der Startadresse. Nun wird das geänderte Programm im Monitor mit "T 1FFD 5201 95D0" nach $95D0 verschoben. Es ist in Ordnung, wenn dabei der Bildschirm alle verschobenen Adressen scrollend anzeigt. Das Verschobene Programm befindet sich nun im Bereich von $95D0 bis $C7D4. Dieses Programm heißt MC Test. Die Einträge sehen aus wie folgt:

```
HEXADEZIMAL: 33 4D 43 2D 54 45 53 D4
KLARTEXT   :  3 M  C  -  T  E  S  T
             01  FD  1F   01  52    D0   95       05
   MASCH.-Progr. Ladeadr. Endadr.  Anf.i.EPROM Modul-Nr.
```

Und zum Schluss noch ein fiktives BASIC-Programm, das den Namen "BASIC-TEST” tragen soll und im Bereich von $1001 bis $ 1200 + 1 = $1201 liegt. Die neue Ladeadresse ist $0708. Nach Veränderung der BASIC-Anfangszeiger und DLOAD befindet sich das Programm von $C7D8 bis $C9D8 Und wieder die Einträge.

```
HEXADEZIMAL: 34 42 41 53 49 43 2D 54 45 53 D4
 KLARTEXT   : 4  B  A  S  I  C  -  T  E  S  T
             00  01  10   01  12   D8  C7       05       00
    BASIC-Progr. Ladeadr. Endadr. Anf.i.EPROM Modul-Nr. Menüende
```

Unser erstes AUTO-STARTMODUL, bestehend aus LOW- und HIGH-EPROM (2x 27128) ist nun komplett. (Zwar wäre auch im Bereich bis $FC00 noch etwas Platz gewesen, aber zum Demonstrieren genügt das gewählte Beispiel). Das Ganze muss jetzt noch abgespeichert werden und an die richtige Adresse verschoben werden. Das LOW-EPROM wird zunächst wie folgt abgespeichert:

S "LOW",8,8000,BFFF.

Der Rest wird mit

S "HIGH",8,C000,FC00

auf Diskette gebracht. Am einfachsten ist es nun, mit dem ADRESS-CHANGER direkt auf der Diskette die Ladeadressen zu ändern. Beide Adressen müssen nach $4000 (dec 16384) verschoben werden.

4.4 Brennen-Löschen

Zum Brennen der EPROMs wird zunächst die Treibersoftware geladen und gestartet. Dann wird im Monitor das zu brennende Programm nach $4000 geladen. Es liegt dann im Bereich von $4000 bis $7FFF. Nach dem Brennen des LOW-EPROMs wird mit dem High-Programm genauso verfahren. Wenn der Anwender EPROMs der

Eprombrenner für den Plus 4

Typen 27256 benutzen möchte (siehe 2.4), wird von dem ersten Modul das LOW Programm von $4000 bis $7FFF und das zweite LOW Programm von $8000 bis $BFFF geladen. Es befinden sich dann 2 LOW-Programme auf dem LOW-EPROM. Die HIGH-Programme werden in gleicher Reihenfolge geladen und gebrannt. Eine Verwechslung der Programme führt beim Aufruf zum Absturz. Sollte trotz aller Sorgfalt mal ein EPROM fehlerhaft gebrannt sein, ist das kein größeres Problem. EPROMs lassen sich mit UVC-Licht löschen. Die beste Möglichkeit ist natürlich eine Löschlampe mit Zeitschaltuhr. Die Löschzeit sollte 5 Minuten pro Löschzyklus nicht überschreiten. Es kann sonst passieren, dass ein EPROM zwar leer ist, sich aber nicht mehr programmieren lässt. Falls erforderlich kann der Löschvorgang nach einer kurzen Wartezeit wiederholt werden. Auch bei neuen EPROMs kann beim ersten Brennversuch eine Fehlermeldung kommen. Neue EPROMSs sind nicht immer ganz leer. Dann hilft aber schon eine Löschzeit von ca. 2 Minuten und das EPROM ist leer, d.h. eigentlich voll. Ein leeres EPROM ist komplett mit “FF” gefüllt. 4.5 Letzte Tipps Bei der Beschaffung von EPROMs sollten Sie darauf achten, dass Sie keine C-MOS Typen kaufen, wenn Sie nicht für einen geerdeten Arbeitsplatz sorgen können. CMOS-EPROMs werden schon bei sehr geringer statischer Aufladung zerstört! Weiterhin kann es passieren, dass C-MOS-EPROMs der Typen 27256 mit Programmierspannung 21 V auf dem Eprommer der Fa. BRAUN-ROTH nicht brennbar sind. Wenn Sie bei EPROMs die erforderliche Programmierspannung nicht einwandfrei definieren können, versuchen Sie zuerst die kleinere Spannung (12,5 V). - Achten Sie immer auf die Einbaurichtung! Die Kerbe im EPROM muss mit der entsprechenden Markierung im Sockel übereinstimmen. Zum Schluss bleibt mir nur noch, dem interessierten User viel Spaß zu wünschen.

Heinz-Werner Schrimpf

Elektronik - Technik Ing. Uwe Peters VDI Tannenweg 9, 2351 Trappenkamp

Commodore Service + Vertrieb 10839/10884 Peters/upe 29.10.1986

B e d i e n u n g s a n l e i t u n g
E P R O M – P R O G R A M M E R
E P R O G 8 6 0 3 2 f ü r d e n P l u s 4

Allgemeines:

Der EPROM-Programmer für den Plus 4 wird bei ausgeschalteten Geräten und vom Netz getrennten Stromversorgungseinheiten in den Extension-Port des Plus 4 gesteckt. Das Flachbandkabel wird mit dem Stecker (Markierung nach oben) in den USER-Port gesteckt. Der Spannungswahlschalter des Programmiergerätes sollte sicherheitshalber auf 5 V oder 12,5 V gestellt werden.

Das Treiberprogramm für den EPROM-Programmer sowie das Autostart-Vorprogramm befinden sich auf der Diskette und wird mit SHIFT/RUN gestartet. Das Programm ist über Menüs selbsterklärend. Es können alle gängigen Typen bis zu 32 KByte (27256) gebrannt werden, soweit diese eine 5 Volt Stromversorgung haben und mit Programmierspannungen von 5,0 – 12,5 – 21 bzw. 25 Volt zu programmieren sind.

Einige wichtige Tipps:

- Stecken oder ziehen Sie die EPROMs nicht während der Programmierphase.
- Verkehrtherum eingesteckte EPROMs können zerstört werden
- Vor dem Programmieren ist die richtige Programmierspannung einzustellen. Ist diese unbekannt so beginnt man versuchsweise mit der kleinsten Spannung.
- Ein 12,5 V EPROM kann mit 21 oder 25 V zerstört werden!
- Die Adressen der Programme sind abhängig von der Lage in den EPROMs und der EPROM-Größe unterschiedlich:

EPROM-Type	EPROM	RAM	Im Modul
2764	**$0000 - $1FFF**	**$4000 - $5FFF**	**$8000 - $9FFF**
27128	**$0000 - $3FFF**	**$4000 - $7FFF**	**$8000 - $BFFF**
27256	**$0000 - $7FFF**	**$4000 - $BFFF**	**$8000 - $FFFF**

Ein Programm das im Modul von $9000 - $9FFF laufen soll wird im EPROM von $1000 - $1FFF stehen und zum Programmieren im RAM in den Bereich $5000 - $5FFF verschoben werden müssen.

Der Bereich $FD00 -$FF3F wird vom TED-Chip und vom System benutzt und muss für Anwenderprogramme auf anderen Bänken ausgespart werden. Hier wird immer das KERNAL eingeblendet.

Alle Adressangaben in dieser Anleitung und in den Programmen beziehen sich daher auf den RAM-Bereich und nicht auf den EPROM bzw. Modulbereich.

Das Treiber-Programm läuft von $1000 - $3FFF. Dieser Bereich darf für andere Programme nicht benutzt werden.

Auf Diskette abgespeicherte EPROM-Inhalte können auch von anderen Brennern verarbeitet werden. (z.B. C64 oder C128) Siehe hierzu auch den Absatz „Kompatibilität“.

Die verschiedenen Speicherbänke:

Das Betriebssystem des Plus 4 kann 4 verschiedene ROM-Bereiche verwalten. Dies sind:

1	**BASIC / KERNAL**	**(internes ROM-Betriebssystem)**
2	**3 + 1 Software**	**(Steckplatz U25 + U26 Low und High**
3	**C 1 Low und High**	**(1. externer Steckplatz auf Modul)**
4	**C 2 Low und High**	**(2. externer Steckplatz auf Modul)**

Jeder ROM-Bereich ist 32 KByte groß und in 2 gleichgroße Teile (LOW und HIGH) aufgeteilt. Dies sind dann jeweils ROMs der Größe 27128. Die einzelnen ROM-Bänke können miteinander kombiniert werden sodass maximal 16 Bankkombinationen möglich sind von denen (bei 32 KByte Programmen) jedoch nur die oben angegebenen vier Kombinationen sinnvoll sind. Die ROM-Bereiche liegen stets auf den Adressen:

Low ROM = $8000 - $BFFF 32768 – 49151
High ROM = $C000 - $FFFF 49152 – 65535

Die Bankkonfigurationen werden jeweils durch einen Schreibbefehl auf folgende Speicherstellen angesprochen:

Bank-Nr.	Umschalt-Adresse	Low Bank	High Bank
0	$FDD0 = 64976	BASIC	KERNAL
1	$FDD1 = 64977	3+1 Low	KERNAL
2	$FDD2 = 64978	C 1 Low	KERNAL
3	$FDD3 = 64979	C 2 Low	KERNAL
4	$FDD4 = 64980	BASIC	3+1 HIGH
5	$FDD5 = 64981	3+1 Low	3+1 HIGH
6	$FDD6 = 64982	C 1 Low	3+1 HIGH
7	$FDD7 = 64983	C 2 Low	3+1 HIGH
8	$FDD8 = 64984	BASIC	C 1 HIGH
9	$FDD9 = 64985	3+1 Low	C 1 HIGH
A	$FDDA = 64986	C 1 Low	C 1 HIGH
B	$FDDB = 64987	C 2 Low	C 1 HIGH
C	$FDDC = 64988	BASIC	C 2 HIGH
D	$FDDD = 64989	3+1 Low	C 2 HIGH
E	$FDDE = 64990	C 1 Low	C 2 HIGH
F	$FDDF = 64991	C 2 Low	C 2 HIGH

Im Bereich von $FC00 - $FCFF liegen die sogenannten Banking-Routinen mit denen die Zugriffe und Datenübertragungen zwischen den einzelnen Bänken vorgenommen werden können.

Eine Übersicht über die hauptsächlichsten Banking-Routinen in der Sprungtabelle 1 werden hier angegeben, weitere Informationen sind aus dem ROM-Listing für den PLUS 4 zu ersehen.

64753 = $FCF1 Modulaufruf beim JMP in die aktuelle Bankkonfiguration werden folgende Parameter übergeben:

in $00FB die aktuelle Bankkonfiguration
in $02FE Sprungadresse Low Byte
in $02FF Sprungadresse High Byte

64756 = $FCF4 Modulinitialisierung Prüfen des CBM Zeichens und Eintrag der entsprechenden Modulnummer in die Modultabelle.

64759 = $FCF7 Long-Fetch-Routine Lesen au seiner anderen Bank. Hierbei werden folgende Informationen übergeben:
in $00BE Low Adressbyte
in $00BF High Adressbyte
im Y-Register der Offset der Adresse
im X-Register die Nummer der Zielbank
im Akkumulator die Nummer der aktuellen Bank
danach das gelesene Byte

64762 = $FCFA Long-Jump-Routine Bei diesem Sprung in eine andere Bank werden folgende Informationen übergeben:

in $05F0 Low Byte der Zieladresse
in $05F1 High Byte der Zieladresse
in $05F2 der zu übergebende Akkuinhalt
in $05F3 das zu übergebende X-Register
in $05F4 der zu übergebende Status
im X-Register die Nummer der Zielbank
im Akku die Nummer der aktuellen Bank

64765 = $FCFD Interrupt-Routine Mit generell folgendem Programmablauf:

1. Bankkonfiguration 0 wird eingeschaltet
2. Die KERNAL-IRQ Routine wird durchlaufen
3. Zurück in die ursprüngliche Bankkonfiguration
4. Akku, X-Register und Y-Register vom Stack

Selbststartende Modulprogramme (Autostart-Module)

Bei jeder Initialisierung des Computers sucht der PLUS 4 die 4 ROM-Bänke nach gültigen Programm-Modulen (CBM-Kennzeichen) ab. Findet der Rechner die Zeichen CBM in ASCII in den Speicherstellen $8007 - $8009 so wird das jeweilige Modul als vorhanden in eine Tabelle eingetragen. In dieser Tabelle steht dann die Modulkennnummer aus der Speicherstelle $8006 des entsprechenden Moduls. Bei Modulnummer 1 wird die Initialisierungsroutine des Moduls ab $8000 aufgerufen. Mit der Modulnummer 0 gilt das Modul als nicht vorhanden.

Um ein Autostart-Modul zu erhalten muss man daher nur die Kennnummer 1 verwenden und das Modul in einen Low Platz stecken.

Die ersten Speicherstellen des Moduls müssen dann wie folgt lauten:

Adresse	Hex Wert	Ass.Befehl	Funktion
$8000	$68	PLA	Rücksprungadresse vom
$8001	$68	PLA	Stack in den Akku
$8002	$4C $0A $80	JMP $800A	Sprung zum Programmstart
$8005			Nicht belegt
$8006	$01		Modulkennnummer
$8007	$43 $42 $4d		ASCII „C B M“
$8008			Programmbeginn

In dieser einfachen Weise funktioniert es jedoch nur bei Maschinenprogrammen wenn vorher im PLUS 4 noch nichts initialisiert war. Bei BASIC Programmen ist etwas mehr zu tun und auch einige KERNAL Routinen aufzurufen. Um auch für Anfänger die Benutzung von BASIC Programmen möglich zu machen befindet sich auf der Diskette ein Programmvorspann für Autostartmodule.

Der Vorspann initialisiert erst den Computer wie üblich und stellt dann in einem Menü mehrere Programme Ihrer Wahl (eigene BASIC oder Maschinenprogramme) zur Verfügung die dann auf Tastendruck gestartet werden können.

Der Autostart – Modul – Vorspann

Der Vorspann befindet sich auf der Diskette und ist bereits in den RAM-Bereich ab $4000 verschoben. Der Vorspann ist 400 Bytes lang und wird an den Anfang eines jeden ROMs in den Bereich $0000 - $0187 geschrieben, wenn dieses ROM automatisch starten soll. Im Anschluss an diesen Vorspann wird dann ab $0188 für jedes, über das Menü zu erreichende Programm, ein Menüeintrag abgelegt. Die Anzahl der Einträge ist nur vom Speicherplatz begrenzt es ist jedoch auf dem Bildschirm eine größere Anzahl als ca. 20 nicht sinnvoll. Der letzte Eintrag wird mit dem Endkennzeichen $00 abgeschlossen. Hiernach kann direkt das erste Programm beginnen.

Programmeinträge können folgendermaßen aussehen:

Adresse	Inhalt	Beschreibung der Funktion
01 = $0001	$41 = „A“	ASCII „A“ als Menükennzeichen für das jeweilige Programm, dies ist auch gleichzeitig die Taste mit der das Programm gestartet wird. Menükennzeichen = $00 ist Menüende
02 = $0002 bis 14 = $000E	„prgm.namE“	Kleingeschriebener Programmname der letzte Buchstabe ist großgeschrieben d.h. das 7. Bit ist = log. 1
15 = $000F	$00 oder $01	Programmkennzeichen: $00 = BASIC Programm $01 = Maschinenprogramm
16 = $0010 17 = $0011	Low Byte High Byte	Programm Anfangsadresse im RAM: Bei BASIC Programmen ist hier der BASIC Pointer TXTTAB von $2B $2C, also üblicherweise $01 $10, einzusetzen.
18 = $0012 19 = $0013	Low Byte High Byte	Programm Endadresse im RAM: Bei BASIC Programmen kann hier der Pointer VARTAB von $2D $2E eingesetzt werden.
20 = $0014 21 = $0015	Low Byte High Byte	Programm Anfangsadresse im EPROM: Bei den Low EPROMs ist hierbei $8000 und bei den High EPROMs $C000 hinzuzuzählen.
22 = $0016	$0A	Bank Konfigurationsnummer in der das entsprechende EPROM gesteckt werden soll.

Möchte man die Möglichkeit haben aus dem Menü ins BASIC zurückkehren zu können so kann der erste, oder ein anderer Menüpunkt mit einem BASIC Dummy Programm belegt werden welches nur aus drei Nullbytes besteht und somit gleich beendet ist. Der Rechner meldet sich sogleich mit READY. Ein Beispiel hierfür zeigt der zweite Eintrag:

23 = $0017	$45 „E“	ASCII „E“ für Ende
24 = $0018	$45 $4e $44 $C5	Programmname endE
28 = $001C	$00	BASIC Programm
29 = $001D	$01 $10	Startadresse $1001
31 = $001F	$03 $10	Endadresse $1003
33 = $0021	$FD $BF	ROM Adresse $BFFD -> 3 Nullbytes
35 = $0023	$05	Bank der 3+1 Software

Die Adresse der Einträge können sich entsprechend der Länge der einzelnen Einträge natürlich ändern. Eine Abbruchmöglichkeit ist im Vorspannprogramm bereits vorgesehen. Soll Diese nicht benutzt werden so kann der entsprechende Teil einfach überschrieben werden.

Der Autostart-Vorspann benutzt folgende Speicherbereiche die daher anderweitig nicht benutzt werden dürfen.

$22 - $24	24 – 26	Hilfs-Pointer
$D8 - $E1	216 – 225	Speicherplatz der Einträge
$05F5 - $0652	1525 – 1618	Startroutine (SYS 1525)

Mit diesem Maschinensprache-Vorspann können ca. 93KByte Programme in z.B. 6 EPROMs vom Typ 27128 auf einmal verwaltet werden. Der Vorspann braucht dafür nur einmal in das niedrigste EPROM gebrannt zu werden. Ein einzelnes Programm kann maximal ca. 30KByte lang sein, da BASIC Programme ins RAM geladen werden und durchaus über Low/High Grenze des EPROMs bei $C000 hinausragen. Maschinenprogramme die aus der Modulbank laufen können maximal 3 x 32 KByte (also 96 KByte) lang sein. (Bankswitching ist hier allerdings im Programm erforderlich).

B E I S P I E L

Zum Brennen eines EPROMs als Autostartmodul

Wir wollen das EPROM-Programmer Programm von der Diskette in ein EPROM vom Typ 27128 brennen und zwar als ein auf Tastendruck aufrufbares Autostartprogramm. Dabei ist es zweckmäßig, das BASIC Programm an das Ende des EPROM Speichers zu legen, damit für weitere Programme und die dazugehörigen Menüeinträge noch Platz bleibt.

1. Laden des Programms (relativ auf den BASIC Anfang mit LOAD)
2. Programmanfangs und Programmendezeiger abfragen ($2b/$2C und $2D/$2E)
3. Programmlänge errechnen (hier sind es $1621 = 5665 Bytes, Programmende – Programmanfang +1)
4. Sprung in den Monitor
5. Anfang des Programms im EPROM ausrechnen (hier ist es $3FFF - $1621 = $29GF) Der zu brennende EPROM Inhalt soll im RAM ab $4000 stehen daher muss das BASIC Programm nach $69DF verschoben werden. Der Monitor Transfer Befehl lautet: T 1001 2622 69DF
   Trick 18b:
   Beim Transportieren kann es zu beliebig vielen Fehlern kommen. Es dürfen sich bei langen Programmen die neue Zieladresse und die Transportadresse beim Aufwärtstransportieren nicht überschneiden. Außerdem muss man natürlich den Bereich von $FD00 - $FD3F vermeiden.
   Man umgeht all diese Schwierigkeiten wenn man die entsprechenden Programmteile mit z.B.
   S „Filename“,8,ANFANG,ENDE auf Diskette abspeichert und z.B. mit einem Diskettenmonitor anschließend einfach die Startadresse in den ersten beiden Bytes des ersten Programmblocks in die neue Zieladresse ändert. Track und Sektor des 1. Programmblocks finden wir in der BAM auf Track 10 Sektor 0.
6. Den Autostart Vorspann einladen und den Menüeintrag erstellen.
7. Das letzte Menü-Endekennzeichen ist die Null bei $41A4. Hierhin muss unser neues Menükennzeichen. Wählen wir die ASC „1“ so gehört an $414A die $31.
8. Es folgt der Programmname z.B. „eprommeR“. Dies ist die Bytefolge $45 $50 $52 $4F $4D $4D $45 $D2
   Für das letzte „R“ wurden $80 dazugezählt um das geshiftete „R“ zu erhalten. Diese Zeichen werden im Speicher von $41A5 bis $41AC abgelegt. (Anstatt des letzten geshifteten Buchstaben kann auch z.B. ein geshiftetes RETURN -> $A0 <- dem Namen folgen. Nur beides gleichzeitig darf man nicht machen dann gibt es den großen Absturz).
9. In Speicherzelle $41AD folgt nun $00 als Kennzeichen für ein BASIC Programm.
10. Das EPROM-Programmer Programm soll von Standard BASIC Anfang $1001 beginnen. Es wird also $41AE mit $01 und $41AF mit $10 belegt.
11. Die Endadresse $2622 ist uns noch von der vorherigen Abfrage bekannt. Es wird daher $41B0 mit $22 und $41B1 mit $26 belegt.
12. Auch die Startadresse im EPROM (wenn das EPROM z.B. im Sockel U25 der 3+1 Software des PLUS 4 sitzt) ist leicht zu errechnen: $8000 + $29DF = $A9DF. Also wird $41B2 mit $DF und $41B3 mit $A9 geladen. Die eingebaute 3+1 Software muss in diesem Fall herausgezogen werden.
13. In $41B4 muss jetzt die Nummer der ROM-Bank
    -Konfiguration angegeben werden, die eingestellt werden muss, damit man auf das EPROM in dem das Programm steht, zugreifen kann. In unserem Fall könnte man $01 $05 $09 oder $0D wählen. Da unser EPROM jedoch im Sockel U25 der 3+1 Software des PLUS 4 stecken soll schreiben wir in $41B4 den Wert $05.
14. Hinweis:
    Wird das KERNAL ROM wie beim Umspeichern ausgeblendet, so wird auch der Character-Generator ausgeblendet. Dadurch erscheint der Bildschirm während der Installation des BASIC Programms dunkel bzw. flimmert, weil der TED-Chip 8360 ins Leere greift solange kein EPROM im angewählten High Sockel sitzt.

15. In $41B5 kann der nächste Menüeintrag beginnen. Gibt es keinen weiteren Menüeintrag so wird in $41B5 der Wert $00 geschrieben.
16. Der Inhalt des EPROMs ist fertig und sollte jetzt mit dem Befehl; S „epromme R“08,4000,7FFF abgespeichert werden.
17. Nun wird das EPROM programmiert und in den 1+3 Low Sockel U25 des PLUS 4 gesteckt. Nach dem Einschalten bzw. nach einem RESET meldet sich das Auswahlmenü und sodann kann auf Tastendruck das Programm gestartet werden.

Ein abschließender Hinweis:

Sehr lange Programme können auf ein Low und ein High EPROM aufgeteilt werden. Dabei tut man so als wäre ein großes 32 KByte EPROM im Low Sockel und berechnet entsprechend die Zeiger des Menüeintrags.

Dann brennt man die untere und die obere Hälfte getrennt in zwei 16K EPROMs wobei das mit der oberen Hälfte in den High Sockel kommt.

Das Byte für die Bank Konfiguration im Menüeintrag muss so gewählt werden, dass das Low und das High EPROM gleichzeitig angesprochen werden und quasi zu einem 32 KByte EPROM verschmelzen.

Der M a s c h i n e n s p r a c h e - M o n i t o r des PLUS 4

Im PLUS 4 ist ein sehr nützlicher Maschinensprache-Monitor fest installiert. Da dieser Monitor sehr effektiv und einfach zu bedienen ist wäre es unsinnig einen weiteren neuen Monitor zu installieren.

Der Machinensprache-Monitor wird als Bestandteil des EPROM-Programmer Programms betrachtet und wird unbedingt zum Editieren der EPROM-Inhalte, zum Laden und Abspeichern sowie zum Verschieben und Anzeigen der Programme benötigt.

Außerdem kann man Assemblerprogramme direkt eingeben (Direktassembler, ohne Makros, Label etc. ….) und sogar disassemblieren. Der Monitor wird mit dem Befehl MONITOR aufgerufen und meldet sich mit einem BREAK und den Registerinhalten.

Die Arbeitsweise mit diesem Monitor ist den meisten Anwendern bereits bekannt und gewohnt, sie steht allerdings auch ausreichend beschrieben in den entsprechenden Handbüchern.

Kompatibilität mit anderen EPROM-Brennern:

Das vorliegende EPROM-Programmier- und Autostart-Programm kann auch mit anderen EPROM-Programmiergeräten auf anderen Systemen (z.B. C64 oder C128) benutz werden.

In dieser Anwendung kann der entsprechende EPROM-Inhalt komplett mit der EPROM Software und dieser Anleitung erstellt und auf einem Diskettenlaufwerk (z.B. 1541-1551-1571-1581) abgespeichert werden.

Mit einem beliebigen Programmiergerät kann dann auf einem anderen System der EPROM-Inhalt wieder gelesen und dann zum Programmieren benutzt werden.

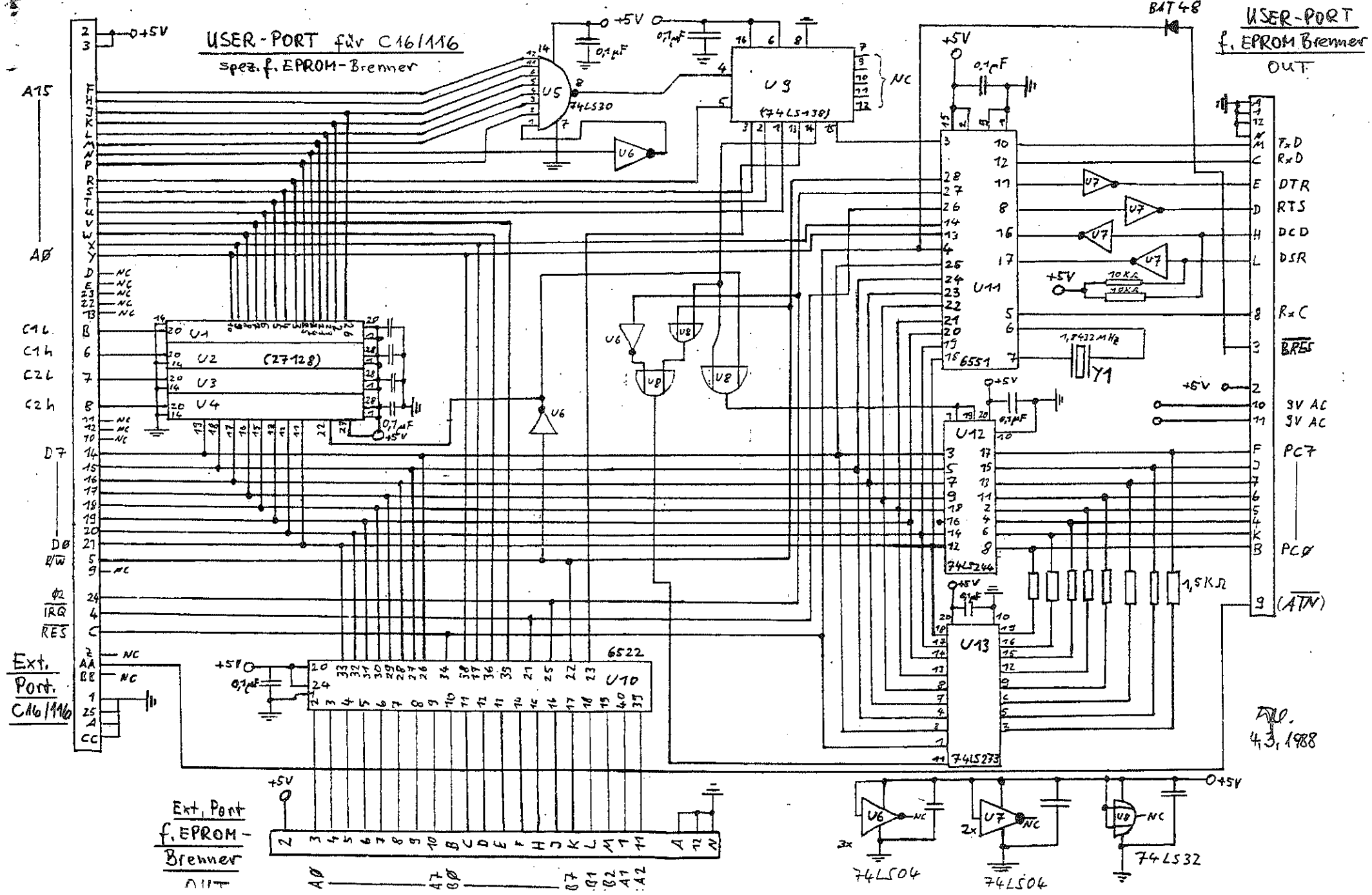
Kann das fremde System den EPROM-Inhalt nicht an die richtige Stelle ins RAM schreiben (weil z.B. im 64er der RAM Anfang bei $1000 anstatt wie beim PLUS 4 bei $4000 liegt) so kann man entweder:

a) Vor dem Abspeichern den Inhalt z.B. mit
   T 4000 7FFF 1000
   transportiere von $4000 bis $7FFF nach $1000
   den EPROM-Inhalt verschieben, oder
b) Einfach die Ladeadresse auf der Diskette am Anfang des ersten Datenblocks (dies steht auf der BAM) mit einem Diskettenmonitor in die gewünschte Adresse ändern. Dies ist besonders wichtig, wenn ein transportieren im RAM durch ein langes Programm, durch Überschneidungen beim Transport nach oben oder beim Transport über ausgesparte RAM-Bereiche (KERNAL, TED etc.) erforderlich wird.

Die fertigen EPROMs kann man dann leicht entweder:

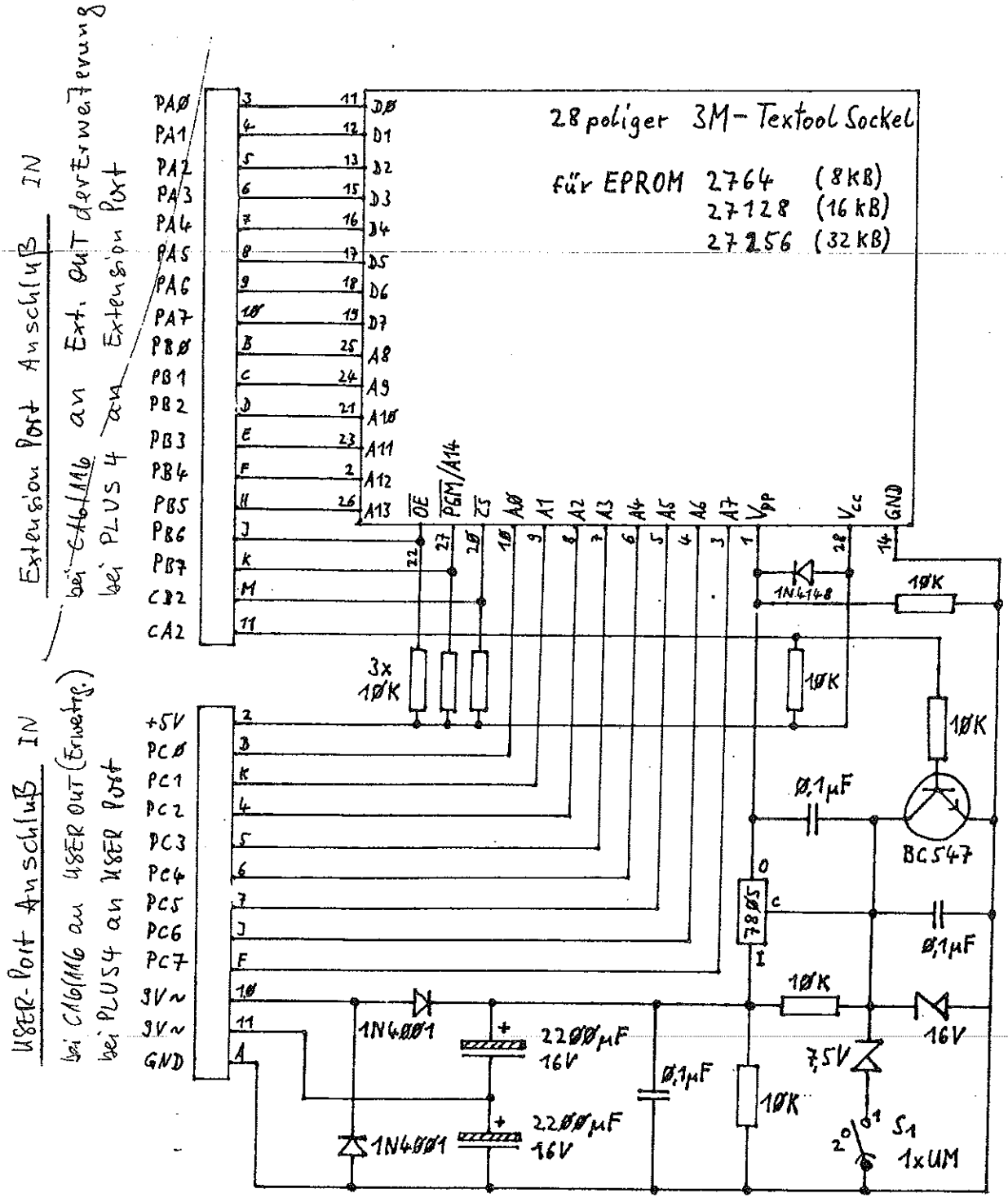
a) Auf den freigemachten Plätzen der 3+1 Software oder
b) Auf einer Modulkarte (möglich mit Gehäuse) problemlos betrieben werden.

Auf diese Weise haben wir bereits eine große Anzahl von PLUS 4 Rechnern mit einer Industrie-Software (z.B. Steuerung von CNC-Maschinen oder Prüfautomaten) versehen, sodass diese Rechner gleich vom Einschalten her nur noch die speziellen, industriellen Aufgaben erfüllen. Die Funktionen der Computer, die Ansteuerung von Druckern, Floppys etc. bleibt natürlich für die neue Software im vollen Umfang erhalten.


USER-PORT für C16/116
spez. f. EPROM-Brenner
USER-PORT
f. EPROM Brenner
OUT
Ext. Port C16/116
Ext. Port f. EPROM-Brenner
A15
A0
D7
D0
R/W
Φ2
IRQ
RES
C1L
C1H
C2L
C2H
U1
U2
U3
U4
(27128)
U5
74LS30
U6
U7
U8
U9
(74LS138)
NC
U10
6522
U11
6551
U12
74LS244
U13
74LS273
Y1
1,7432 MHz
10 KΩ
1,5 KΩ
0,1 µF
+5V
BAT 48
TxD
RxD
DTR
RTS
DCD
DSR
RxC
BRES
9V AC
9V AC
PC7
PC0
(ATN)
3x
74LS04
2x
74LS04
74LS32
A0
A7
D0
D7
R/W
A1
A2
DIO
4.3.1988

E P R O M - B R E N N E R

In dieser Ausführung, in Verbindung mit der USER-PORT-ERWEITERUNG für den C16/C116 oder den PLUS 4, ist das Schaltbild direkt zu verwenden. Bei Verwendung des EPROM - BRENNERS für den PLUS 4 - ohne die USER-PORT-ERWEITERUNG - müssen die Ports PA und PB durch den Baustein U 10 (6522) sowie deren Ansteuerschaltungen U 5 bis U 9 nachgerüstet werden. Das Pinning der Anschlüsse an den beiden Steckern USER-PORT-IN sowie Ext.-PORT-IN muß dann auch entsprechend geändert werden.



Umschalten der Programmierspannung mit Hilfe des Schalters

S_1: in Stellung 1 ist V_{pp} = 12,5 Volt
in Stellung 2 ist V_{pp} = 21,0 Volt