Aufgemotzt

C64 mit 4-MHz-Takt und 16-Bit-CPU

Michael Flamm
René Schneider

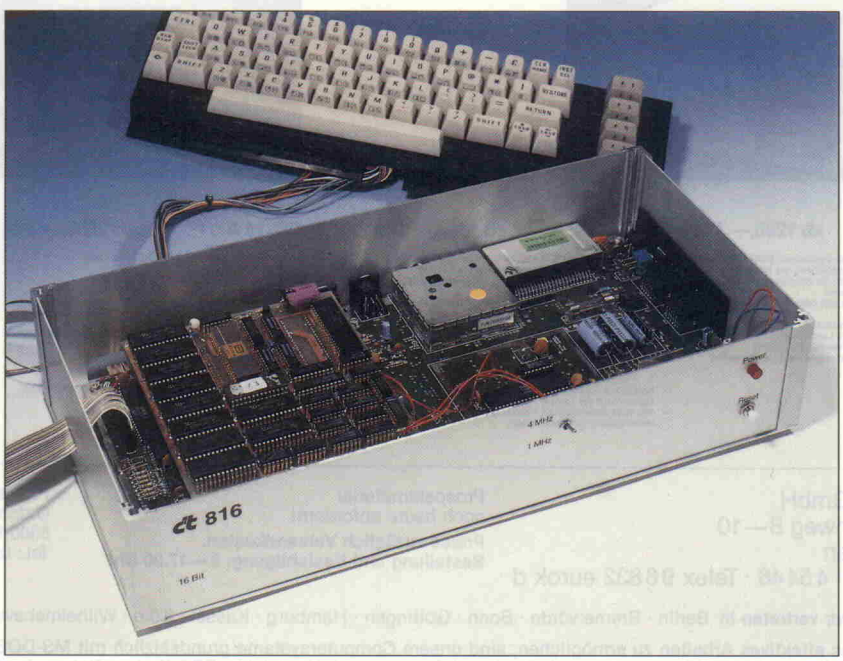
Den C64 schon in den Schuppen geräumt? Das war vielleicht ein bißchen zu voreilig, denn diese '4x4-Erweiterung' macht aus dem betagten C64 einen 16-Bit-Rechner mit ganz neuen Leistungsdimensionen: viermal schneller, viermal mehr Speicher. Möglich macht dies der moderne CMOS-Prozessor 65SC816, eine Weiterentwicklung der 6502-Familie, zu der auch die CPU des C64 gehört. Sie kann im Emulationsmodus die C64-Software ausführen, ist aber eigentlich ein schneller 16-Bit-Prozessor mit erweitertem Befehlssatz, 16 Bit breiten Registern und der Fähigkeit, bis zu 16 MByte Speicher zu adressieren.

Aufmerksame Beobachter der Computer-Szene kennen die 65SC816-CPU bereits. Sie wird auch im Apple IIgs, dem 16-Bit-Nachfolger der Apple-II-Serie, eingesetzt. Allerdings läuft er in der hier beschriebenen Karte mit 4 MHz (im IIgs mit 2,8 MHz) und macht den C64 bis zu viermal schneller. In bestimmten Fällen liegt die Beschleunigung sogar noch darüber, doch dazu später mehr. Über die Eigenschaften des 65SC816 informiert ausführlich die Applikation in diesem Heft.

Rechner im Rechner

Um die Leistung des C64 zu steigern, kann man nicht einfach die CPU auswechseln und die Taktfrequenz erhöhen, da der Video-Chip und der dynamische Speicher keinen höheren Takt vertragen. Die Grundidee dieser Erweiterung beruht auf einem 65SC816-Einplatinencomputer mit eigenem schnellen Takt und eigenem schnellen Speicher, der den Adreßraum des C64 in dem von der 65SC816-CPU adressierbaren Raum einschließt. Der C64 wird so zur Erweiterung für den Einplatinencomputer, der die Versorgungsspannung, die Verbindungen zu den externen Geräten (Bildschirm, Laufwerk und so weiter . . .) und nicht zuletzt eine Vielfalt an Software mitliefert. Das Ergebnis ist der c't816, ein leistungsfähiger 16-Bit-Rechner, der nebenbei auch noch C64-kompatibel ist. Genau dieses Konzept steckt übrigens auch im IIgs von Apple.

Da 256 KByte CMOS-RAM und die Logik für die Einbindung der C64-Hardware einigen Platz erfordern, ist der Einbau der 65SC816-Karte in das C64-Gehäuse nicht möglich. Auch wurde vom Anschluß über den Expansion-Port abgesehen, der diesen blockieren würde, und EPROM-Bänke und Floppy-Beschleuniger ließen sich nicht weiterverwenden. Der Anschluß der Karte erfolgt mit einem Steckadapter über den CPU-Sockel des C64. Das Motherboard baut man dazu



Das Gehäuse sollte ausreichende Luftzufuhr ermöglichen.

Bereich	Größe	Takt	
$00000 - $0FFFF	64 KByte	1 MHz	C64 Speicher / I/O-Bereich
$10000 - $1FFFF	64 KByte	1 MHz	EPROM (einmal gespiegelt)
$20000 - $3FFFF	128 KByte	4 MHz	ungenutzter Bereich
$40000 - $7FFFF	256 KByte	4 MHz	Gepufferter Speicher

Der Speicher für den 65SC816 kann über eine Pfostenleiste noch erweitert werden.

mitsamt der 65SC816-Karte in ein neues Gehäuse ein, das etwa 8 cm lichte Höhe aufweisen sollte.

Wie es einem professionellen 16-Bit-Rechner angemessen ist, sollte das Gehäuse so stabil sein, daß es als Untersatz für den Monitor dienen kann. Um Störstrahlungen zu verhindern, sollte es außerdem aus Metall sein. Die Tastatur schließt man über ein verlängertes Kabel an das C64-Board an, was problemlos möglich ist. Auf diese Weise erhält man ein professionelles System mit abgesetzter, ergonomisch flacher Tastatur, an dem sich ermüdungsfrei arbeiten läßt.

Die 40 Anschlüsse der C64-CPU 6510 liefern alle notwendigen Signale. Auch die Stromversorgung der 65SC816-Karte erfolgt über den CPU-Sockel. Einzig der von der Karte benötigte 8-MHz-Takt muß mit einer zusätzlichen Leitung vom Video-Chip geholt werden. Doch dazu später mehr.

Zwei CPUs

Auf der 65SC816-Karte befindet sich neben der 16-Bit-CPU auch noch der 6510, um den C64 jederzeit wieder in seinen ursprünglichen Zustand versetzen zu können. Dies kann dann erforderlich sein, wenn Programme illegale Opcodes des 6510 verwenden oder Karten im Expansion-Port die Speicher-Konfigurationen verändern, was die 65SC816-Karte nicht merken kann. Ferner gibt es ein 32-KByte-EPROM und bis zu 256 KByte Arbeitsspeicher, der über eine Pfostenleiste weiter ausgebaut werden kann. Zwei weitere Stecksockel für 64 KByte schnelles CMOS-RAM dienen zur Simulation des C64-Arbeitsspeichers.

Von C64-Programmen kann der Arbeitsspeicher des 65SC816 nicht unmittelbar genutzt werden. Es liegt daher nahe, diesen Speicher im 6502-Emulationsmodus als RAM-Floppy zu nutzen. Zu diesem Zweck ist eine Batterie-Pufferung vorgesehen, die den Speicherinhalt auch bei ausgeschaltetem Rechner erhält. Das mit der Platine erhältliche EPROM enthält bereits eine RAM-Floppy-Routine, die den Inhalt einer ganzen Diskette dauerhaft und schnell verfügbar macht. Dazu am Schluß des Artikels mehr.

Die kleine Transistorschaltung sperrt bei absinkender Betriebsspannung den zur Dekodierung dienenden 74HC138 frühzeitig und verhindert, daß der Prozessor 'mit letzter Kraft' den Speicherinhalt zerstört.

Synchron, aber schnell

Die ersten Unterschiede zu den üblichen Einplatinencomputern bestehen in der Takterzeugung: Weil der 65SC816 auch auf den dynamischen Speicher und die verschiedenen Ein-/Ausgabebausteine des C64 zugreifen wird, muß der Takt des 65SC816 mit dem internen Takt des C64 synchronisiert sein. Der Video-Chip liefert mit dem 8-MHz-Dot-Clock ein geeignetes Signal, das, durch zwei geteilt, als Taktsignal für den 16-Bit-Prozessor dient.

Dieser Takt wird erst mit dem Ausgang Q-hold von FF2 verknüpft, bevor er zum Clock-Eingang des 16-Bit Prozessors gelangt. Im Falle eines Zugriffes auf einen langsamen Speicherbereich (zum Beispiel EPROMs können kaum innerhalb eines 250-ns-Prozessorzyklus ausgelesen werden) kann Q-hold die High-Phase des CPU-Taktes verlängern, so daß dieser einem 1-MHz-Takt entspricht.

Q-hold geht in drei Fällen auf high. Die erste Möglichkeit ist, daß der Schalter, der zur Wahl der Taktfrequenz dient, sich in 1-MHz-Stellung befindet. In diesem Fall bleibt der Eingang von FF2 ständig auf Nullpotential: Der Prozessor erhält nur noch verlängerte Taktzyklen. Ähnlich, aber dynamisch wirkt die Dekodierungslogik auf den Takt des 65SC816, falls dieser das EPROM oder den Speicher des C64 anspricht. FF3 und FF4 bilden eine Zähleinheit und setzen Q-hold zurück, wenn sie erkannt haben, daß der langsame Zugriff vollständig abgelaufen ist.

Drittens hängt Q-hold vom RDY-Signal aus dem C64 ab. Der Video-Chip benutzt gewöhnlich diese Leitung, um dem Prozessor mitzuteilen, daß er für eine gewisse Zeit den internen Bus des C64 zu eigenen Zwecken braucht: RDY geht low. Für den 6510 bedeutet dies immer 'Programmablauf stoppen', da es sonst zu Buskonflikten käme. Wie sich aber später zeigen wird, beansprucht der 65SC816 im Gegensatz zum 6510 nur selten den C64-Bus. Dies wird von der Taktlogik berücksichtigt: der 16-Bit-Prozessor wird erst dann gestoppt (indem Q-hold einen hohen Pegel annimmt), wenn er und der Video-Chip gleichzeitig auf den Speicher des C64 zugreifen wollen.

```
PAL14L8

Steuer-PAL fuer 65SC816-Karte


EPB HR  A13   SIM   A16   ADR1 A17   A14   A18  A12  LR     GND
RW  A15 EXTR2 PORT4 EXTR1 IO   PORTO EPROM INT2 INT1 OFF816 VCC


/INT1  = /HR * EPB * OFF816 * /A16 * /A17 * /A18 * A15 * RW
     + HR * EPB * /A13 * OFF816 * /A16 * /A17 * /A18 * A15 * RW
     + HR * EPB * A13 * OFF816 * /A16 * /A17 * /A14 * /A18 * A15
        * /LR * RW
     + EPB * OFF816 * SIM * /A16 * /A17 * /A18 * A15 * RW

/INT2  = OFF816 * /A16 * /A17 * /A18 * /RW

/EPROM  =  OFF816 *  A16 * /A17 * /A18 *  RW
      + /EPB *  OFF816 * /SIM * /A16 * /A17 * /A18 *  A15 *  RW

/PORTO  =  OFF816 * /A16 *  ADR1 * /A17 * /A18 * /RW

/IO  = /A13 *  OFF816 *  A14 *  A12 *  A15

/EXTR1  =  OFF816 * /A16 * /A17 * /A18 * /A15

/PORT4  =  OFF816 * /A16 *  ADR1 * /A17 *  A18 * /RW

/EXTR2  =  HR * EPB * A13 * OFF816 * /SIM * /A16 * /A17 * A14
        * /A18 * A15 * /LR * RW
      + HR * EPB * A13 * OFF816 * /SIM * /A16 * /A17 * /A18
        * A15 * LR *  RW
      +  EPB *  OFF816 *  SIM * /A16 * /A17 * /A18 *  A15 * /RW
      + /EPB *  OFF816 *  SIM * /A16 * /A17 * /A18 *  A15 *  RW


PAL 16H2

Adress-Dekoder fuer 65SC816-Karte


A7  A6  A5  A4  A3 A2   A1 A10 A0  GND
A11 A15 A12 A14 ZP ADR1 A9 A8  A13 VCC



ADR1  = /A6 * /A7 * /A5 * /A13 * /A4 * /A8 * /A3 * /A9 * /A2
        * /A14 * /A1 * /A12 * /A10 * /A15 *  A0 * /A11

ZP    = /A13 * /A9 * /A14 * /A12 * /A10 * /A15 * /A11
```

Die PALs erzeugen unter anderem die Select-Signale für das EPROM und das Simulations-RAM.

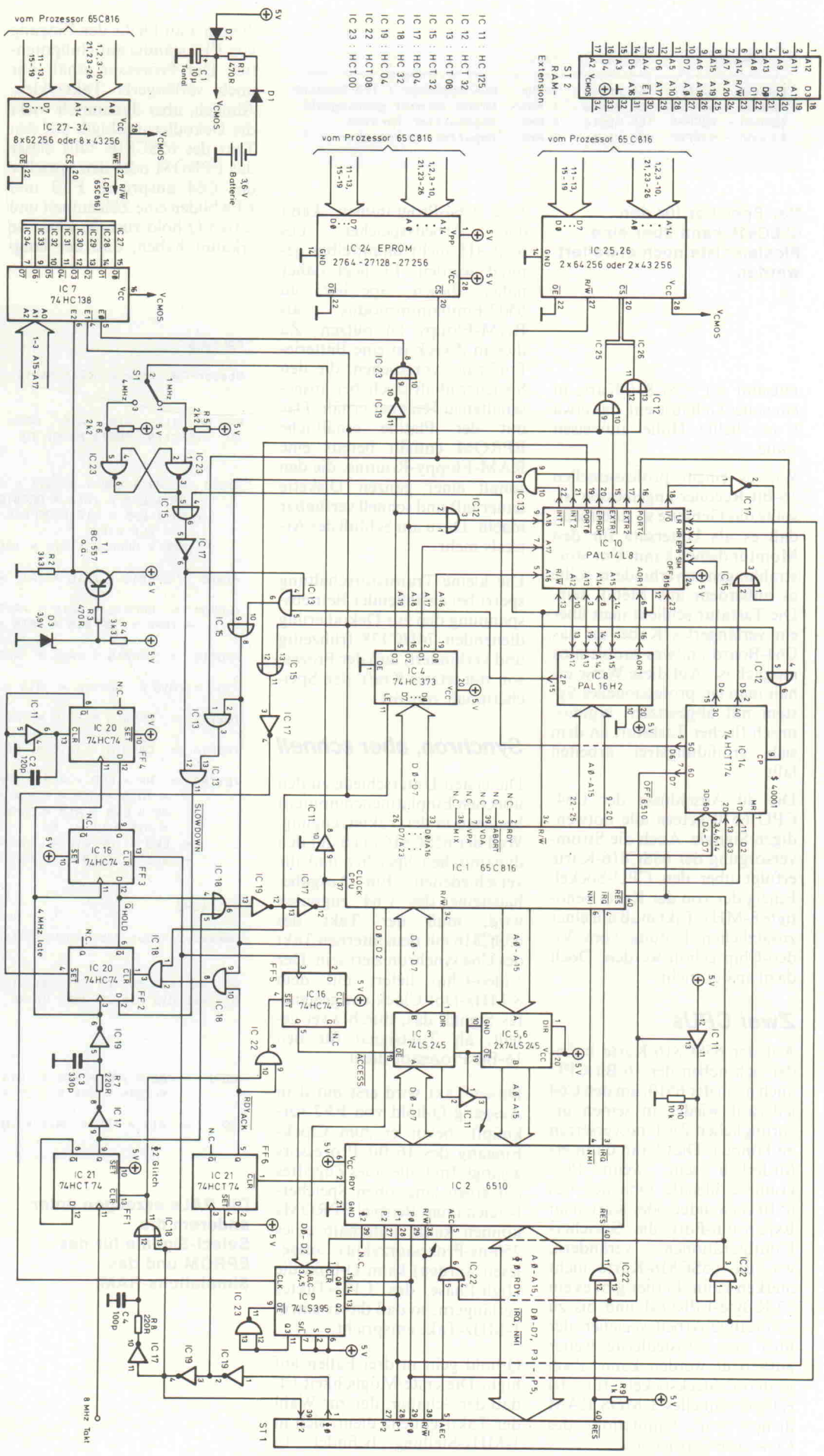
Greift der 16-Bitter nur auf seinen schnellen CMOS-Speicher zu, kann der 65SC816 im Gegensatz zur C64-CPU parallel zum Video-Chip arbeiten. Die 65SC816-Karte kann dann sogar mehr als die vierfache Geschwindigkeit des C64 erreichen. Im Normalfall liegt die Steigerung jedoch leicht unter dem Faktor 4, wie später noch beschrieben wird.

Lockere Bindung

Die einzige Verbindung, über die der 65SC816 mit dem C64 und der Außenwelt kommuniziert, besteht aus drei 74LS245-Buffern, die seine Adreß- und Datenleitungen und die R/W-Leitung mit dem Bus des C64 zusammenschließen können. Ein direkte Verbindung beider Busse würde immer auch die Dekodierungslogik des C64 aktivieren, auch wenn der Zugriff nur auf einen externen Speicherbereich erfolgen soll. Die Verbindung ist also getrennt, außer wenn der 65SC816 den 64er anspricht.

Diese trennbare Verbindung hat auch bei der Beschleunigung von C64-Programmen einen Sinn. Beim Schreiben in den Speicherbereich des C64 erhält nämlich das Simulations-RAM (IC25, IC26) dieselben Daten wie der C64, womit ständig eine exakte Kopie des C64-Speichers vorhanden ist. Will der 16-Bit-Prozessor nun aus diesem Bereich lesen, braucht die Taktlogik den 65SC816 nicht zu bremsen, da die Daten aus dem externen RAM bezogen werden können. Für die unteren 32 KByte ist dies immer gültig. Im oberen Speicherbereich dagegen muß man mehrere Konfigurationen vorsehen, die den verschiedenen Möglichkeiten des C64 entsprechen.

Man muß sich entscheiden, was man im externen Speicher haben will: entweder eine Kopie des internen RAMs oder der System-ROMs. Da eine Beschleunigung von BASIC-Programmen besonders sinn-



Die Schaltung ist fast ausschließlich in stromsparender CMOS-Technik ausgelegt.

Stückliste

Halbleiter

IC1	65SC816-4
IC2	6510
IC3,5,6	74LS245
IC4	74HC373
IC7	74HC138
IC9	74LS395
IC11	74HC125
IC12,15	74HCT32
IC13	74HCT08
IC14	74HCT174
IC16,20	74HC74
IC17	74HC04
IC18	74HC32
IC19	74HC04
IC21	74HCT74
IC22	74HCT08
IC23	74HCT00
IC24	27256
IC25-34	62256/43256 120 ns
IC8	PAL 16H2
IC10	PAL 14L8
T1	BC557B o.ä.
D1,2	1N4148
D3	Zener 3V9

Widerstände

R1	470
R2	3k3
R3	470
R4	3k3
R5	2k2
R6	2k2
R7	220
R8	220
R9	1k
R10	10k

Kondensatoren

C1	10 µF, 16 V, Tantal
C2	120 pF, Styro
C3	330 pF, Styro
C4	100 pF, Styro
CB	13 × 100 nF

Fassungen

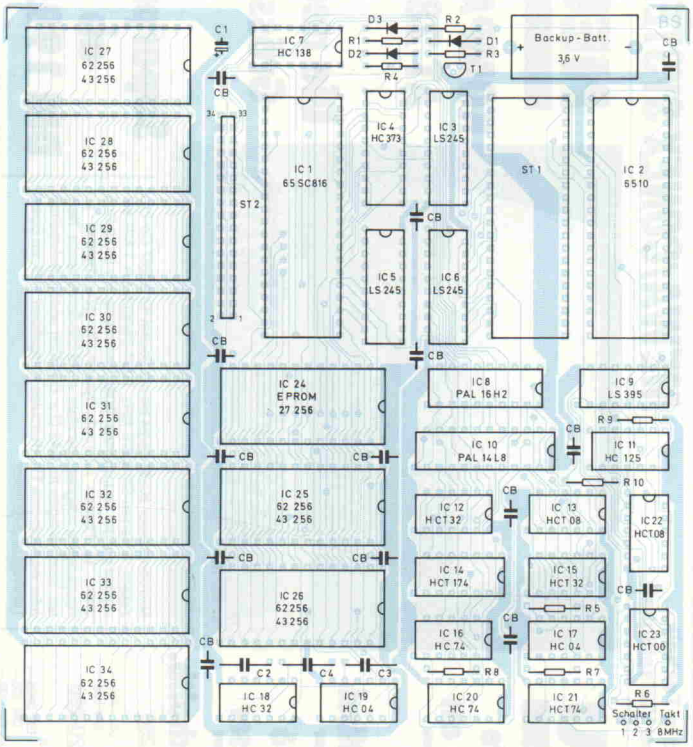
2 × 40polig
1 × 40polig, Federkontakte
1 × 40polige Wrap-Fassung
1 × 24polig RM 7,5
11 × 28polig
5 × 20polig
12 × 14polig
3 × 16polig

Sonstiges

S1 Min. Schalter 1 × um
Pfostenleiste 2reihig, 34polig
1 × Lötstift 1 mm ∅
40poliger Steckadapter, etwa von
Gunther Gudehus
Liether Ring 33
2200 Kl. Nordende

oder entsprechende Steckstifte, etwa von
Fischer metroplast
Nottebohmstraße 55
5880 Lüdenscheid
0 23 51 / 41 7 40 + 44 69

Lithium-Batterie 3,6 V

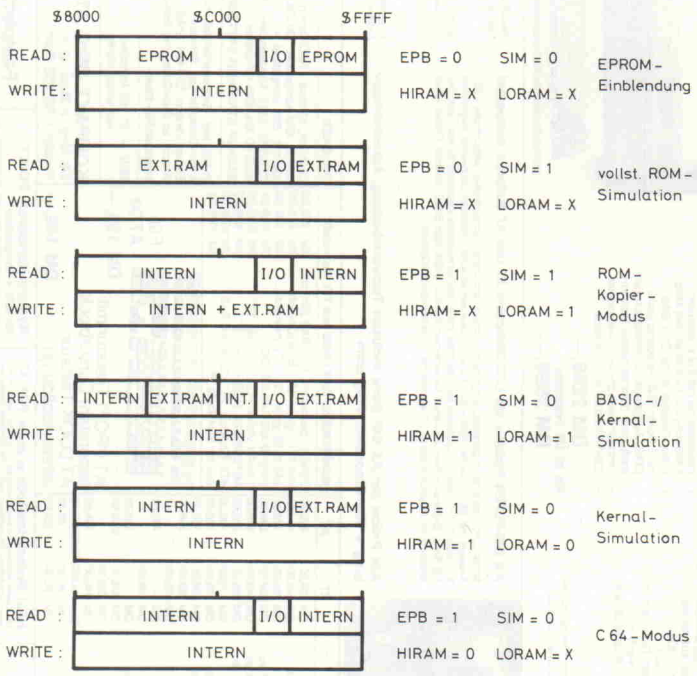


Bei den Logik-Chips sind unbedingt die angegebenen Typen zu verwenden.

voll ist, wurde ein ROM-Simulationsmodus vorgesehen. Das Schreiben in das externe RAM ist nur in einem speziellen Modus erlaubt; sonst lehnt die Dekodierungslogik alle Schreibzugriffe ab. Beim Start kopiert die Software der Karte den ROM-residenten Interpreter und das Betriebssystem in das schnelle, externe RAM. Von diesem Zeitpunkt an werden alle Lesezugriffe, die eigentlich dem ROM gelten, auf das externe RAM umgeleitet.

Die Vorteile sind erheblich: Erstens kann der Anwender, dem das EPROM-Programmieren schon immer unheimlich war, Betriebssystemänderungen ohne Probleme durchführen – zum Beispiel indem er ein neues Betriebssystem wählt, das sich auf der RAM-Disk befindet. Zweitens laufen alle Programme, die öfter in die ROM-Routinen einspringen, bis zu viermal schneller.

Das 'bis zu viermal schneller' verdient eine Erklärung: Weil der dynamische Speicher im

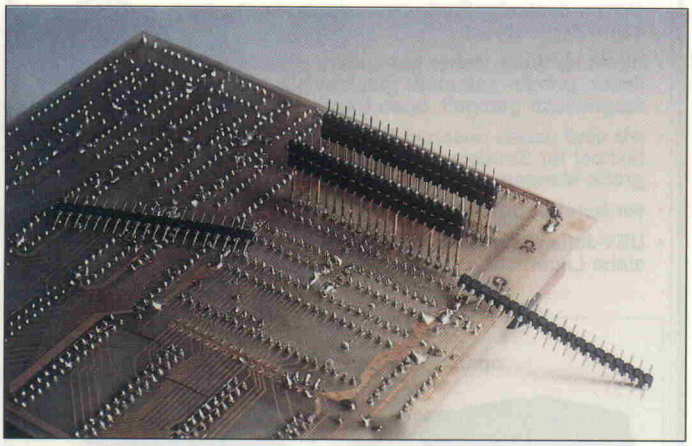


Die Zugriffe auf den I/O-Bereich werden bei jeder Konfiguration in den C64 gelenkt.

C64 auch als Video-Speicher dient, müssen alle Schreiboperationen auch im 64er landen – der 16-Bit-Prozessor kann ja nicht unterscheiden, ob er allgemeine Daten oder Videoinformationen bearbeitet. Da der 65SC816 bei einem Zugriff in den C64 gebremst wird, laufen die Programme, die nicht auf eine Bildschirmausgabe verzichten können, dann nur annähernd viermal schneller. Eine Ausnahme berücksichtigt die Dekodierlogik aber: Da die Zero-Page und der Stapelspeicherbereich ganz sicher keine Daten für den Video-Chip enthalten, wird in diesem oft benutzten Bereich nur in das externe RAM geschrieben, also mit 4-MHz-Takt.

Zwei Ports

Der 6510 besitzt bekannterweise einen internen parallelen Port, der die Adresse $0001 belegt. Drei Leitungen werden für die Dekodierung herangezogen, die drei anderen dienen zur Kommunikation mit dem Kassetten-



Der Steckadapter läßt sich aus einem Wrap-Sockel und geeigneten Steckstiften herstellen.

gerät. Da der 65SC816 diesen Port nicht besitzt, wird er hardwaremäßig nachgebildet. Verzichtet man auf den Kassettenport, genügt dazu ein 74LS395-Schieberegister, das als paralleler Ausgangsport betrieben wird. Falls der 16-Bit-Prozessor den Zustand der Ausgangsleitungen erfahren möchte, braucht er nur die Adresse $00001 im RAM auszulesen.

Bei einem Reset verlangt der 6510, daß die drei Leitungen, die zur Konfigurationswahl dienen, sich auf hohem Pegel befinden. (Wenn die ROM-Konfiguration nicht gewählt ist, findet der Prozessor keine Start-Routine!) Mit Hilfe einer Rückkopplung kann diese Voraussetzung auch mit dem 74LS395 erfüllt werden: Die Ausgangsleitungen sind erst dann aktiv, wenn einmal in die Adresse $0001 geschrieben wurde.

Genauso wichtig ist, daß der 65SC816 beim Starten den Reset-Vektor aus dem externen EPROM in Bank 1 bezieht. Die C64-Startroutine sollte zwar auch aufgerufen werden, aber erst, nachdem eine Kopie der internen ROMs im externen schnellen RAM erstellt ist. Damit dies möglich ist, muß eine zusätzliche Konfiguration eingeführt werden: eine 32-KByte-EPROM-Einblendung in den oberen Speicherbereich der Bank 0 (da der Reset-Vektor sich an den Adressen $0FFFC-$0FFFD befindet). Außerdem muß es eine Möglichkeit geben, die internen ROMs auszulesen.

Ein zweiter Port ist also notwendig: Ein 74HCT174, dem die Adresse $40001 zugeteilt ist (er ist auch an den Adressen $80001 und $C0001 zu finden!), erfüllt diese Aufgabe. OFF6510 (Bit 7) spielt die Rolle eines Software-Umschalters. Wenn dieser Ausgang den Zustand logisch eins annimmt, wird der 65SC816 ausgeschaltet, der 6510 startet wie gewöhnlich und der C64 befindet sich in seinem Originalzustand.

Weil der 6510 bei einem Zugriff auf die Adresse $0001 den Datenbus intern absperrt, ist es leider unmöglich, den 65SC816 per Software wieder zu aktivieren. SPEED (Bit 6) ist ebenfalls ein Software-Schalter. Solange dieses Bit gesetzt ist, erhält der 65SC816 nur 1-MHz-Taktzyklen. Ist das SPEED-Bit gelöscht, hängt die Taktfrequenz vom Schalter S1 ab. Sie beträgt entweder 1 oder 4 MHz. Mit diesem Schalter kann jederzeit, auch bei laufendem Programm, der 1-MHz-Takt erzwungen werden.

Die zwei übrigen Leitungen, EPB (Bit 5) und SIM (Bit 4), dienen zur Konfigurationswahl in den oberen 32K der Bank 0. Für diesen Speicherbereich kommen nämlich vier Möglichkeiten in Frage:

- die erwähnte 32-KByte-EPROM-Einblendung,
- eine ähnliche RAM-Einblendung, die dem Anwender einen 28 KByte großen ROM-Simulationsbereich zur Verfügung stellt (dort könnte zum Beispiel ein 16-Bit-Betriebssystem Platz finden),
- ein ROM-Auslesemodus, der gleichzeitig das Schreiben in das externe RAM erlaubt,
- eine Konfiguration, die der üblichen BASIC-, Kernal- oder RAM-Wahl entspricht.

Richtige Familie einkaufen

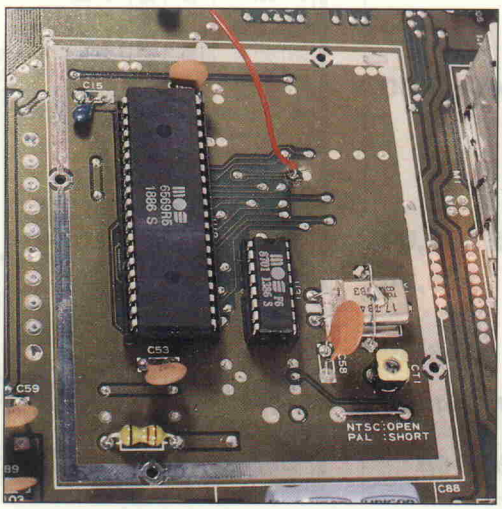
Beim Einkauf der ICs und beim Bestücken sollte man unbedingt die angegebenen Logikfamilien verwenden. Es kann weder HC durch HCT ersetzt werden, noch andersherum.

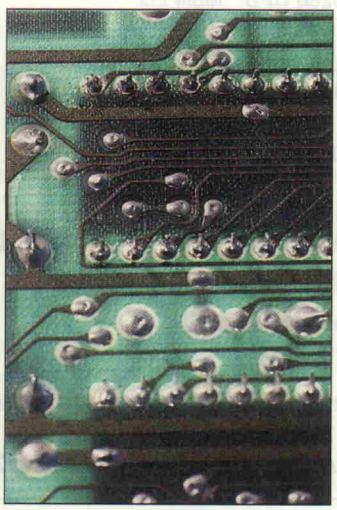
Zum Ausbau der C64-Platine trennt man die beiden Gehäuseteile voneinander und entfernt die Tastatur. Ist der 6510-Prozessor eingelötet – was bei manchen Rechnern der Fall ist –, muß man diesen auslöten und durch einen Sockel ersetzen. Diese Operation mag manchem unbehaglich sein, sie ist aber relativ unkritisch. Voraussetzung sind lediglich ein guter (spitzer) Lötkolben (mindestens 16 W), eine Lötsaugpumpe und eventuell ein Stück Lötsauglitze.

Für jedes Beinchen führt man dieselben Schritte durch. Der Pin wird kurz, aber kräftig erhitzt und das Lot mit der Pumpe abgesaugt. Das Lötauge und der IC-Pin müssen von Lötzinn völlig frei sein. Sollte dies bei einigen Pins nicht auf Anhieb gelingen, gehe man weiter zum jeweils nächsten, um den Prozessor nicht an einer Stelle zu stark zu erhitzen. Vor dem zweiten Durchgang empfiehlt es sich, die nicht vollständig freigelegten Pins erneut zu verzinnen, weil die Saugwirkung der Pumpe sich sonst nicht voll auf das Lötzinn konzentriert.

Anschließend wird der Prozessor aus der Platine herausgezogen, falls er nicht von selbst herausfällt. Wenn einige Beinchen noch klemmen sollten, hilft ein weiterer Durchgang: Mit der Pinzette können die klemmenden Pins durch Wackeln lokalisiert werden. Diese werden nochmals verzinnt, dann kommt wieder die Lötsaugpumpe zum Einsatz. Auf keinen Fall darf die CPU mit Gewalt aus der Platine herausgehebelt werden. Nachdem man die Fassung eingelötet hat, sollte man noch nachprüfen, daß zwischen den Leiterbahnen keine Lötzinnbrücken erzeugt wurden. Fassungen mit Federkontakten sind übrigens zu bevorzugen, da sich der Adapter dann leichter einstecken läßt.

Den Adapter selbst stellt man am einfachsten aus einem 40poligen Wrap-Sockel her, der von der Rückseite in die 65SC816-Karte eingelötet wird. In ihn steckt man entweder einen 'zweiseitig männlichen' 40poligen Steckadapter oder entsprechende Einzelkontakte, mit denen die Karte dann in den CPU-Sockel auf dem Motherboard gesteckt wird. Durch den Wrap-Sockel ist zwischen C64-Board und 65SC816-Karte





Den 8-MHz-Takt findet man beim Video-Chip.

Beim Auslöten der CPU muß jeder Pin möglichst vollständig von Lötzinn befreit werden.

```
10 REM Kopierprogramm von Floppy-Disk
20 REM auf RAM-Disk
50 OPEN 1,8,0, "Datei"
60 OPEN 2,1,1, "Datei"
70 GET#1,A$
80 IF ST<>0 THEN 110
90 PRINT#2,A$;
100 GOTO 70
110 PRINT#2,A$;
120 CLOSE 1 : CLOSE 2
```

Auf diese Weise kann die RAM-Disk angesprochen werden.

noch genügend Platz für eventuelle Piggy-Backs in den ROM-Sockeln.

Den 8-MHz-Takt holt man sich vom Video-Chip, der sich unter dem Abschirmblech in der Mitte der Platine befindet. Dieses wird abgenommen. Man kann die Leitung direkt an Pin 22 des Video-Chips anlöten, aber es ist empfehlenswert, zuerst nachzusehen, ob das Dot-Clock-Signal nicht anderswo in der Nähe des Video-Chips vorhanden ist, wie es die Abbildung zeigt.

Einschalten . . .

. . . sollte man den c't816 erst nach sorgfältiger Kontrolle auf einwandfreie Lötstellen und korrekte Bestückung. Die Reset-Routine im EPROM springt zuerst in Bank 1, damit der ROM-Auslesemodus eingeschaltet werden kann. Aber bevor diese Konfiguration gewählt wird, fragt die Routine die RUN/STOP-Taste ab. Falls diese gedrückt ist, wird der Prozessor 6510 aktiviert. Ist sie nicht gedrückt, kopiert der 65SC816 den BASIC-Interpreter und das Kernal-Betriebssystem in sein schnelles RAM.

Einige ROM-Änderungen werden bereits zu diesem Zeitpunkt ausgeführt: Die RAM-Disk erhält die Peripheriegeräte-Nummer 1 (das Kassettengerät ist vom 65SC816 aus sowieso nicht ansprechbar), indem alle Ein-/Ausgaberoutinen für den RAM-Disk-Betrieb geändert werden. Weiter wird ein 'Bremsbefehl' (SPEED = 1) beim Öffnen eines Kanals auf den seriellen IEC-Bus eingeführt. Sonst kann das Diskettenlaufwerk oder der Printer nicht schnell genug antworten: Der Computer würde einen 'Device not present Error' ausgeben. Ein ähnlicher 'Du kannst wieder mit 4 MHz arbeiten'-Befehl wird dann ausgeführt, wenn alle Kanäle auf den IEC-Bus wieder geschlossen sind.

Eine interessante Erweiterungsmöglichkeit für die Startroutine wäre eine Routine, die auf der RAM-Disk nach einem neuen Betriebssystem sucht und dieses lädt, falls sie eins findet. Andernfalls könnte wieder in die hier abgebildete Startprozedur verzweigt werden.

RAM-Disk

Das gepufferte RAM der Erweiterung ermöglicht eine dauerhafte RAM-Disk, von der pro Sekunde 30 KByte Informationen geladen werden können. Diese Geschwindigkeit ist ganz schön verwirrend, da Diskettenoperationen sonst immer eine Verschnaufpause bieten. Jetzt droht der Streß . . .

Die RAM-Disk ist so ausgelegt, daß sie aus Rechnersicht wie ein Peripheriegerät erscheint. Alle Ein-/Ausgaberoutinen, die über die 'offizielle' Sprungleiste abgerufen werden (das heißt OPEN, CLOSE, LOAD, SAVE, CHKIN, CHKOUT, CHRIN, CHROUT), erkennen die RAM-Disk an der Geräteadresse 1. Von BASIC aus kann man also mit den gewöhnlichen I/O-Befehlen arbeiten. Die gängigen Fehlermeldungen werden auch unterstützt, wobei einige eine neue zusätzliche Bedeutung bekommen:

– Falls der Rechner die RAM-Disk anspricht und diese noch nicht installiert wurde, erscheint ein 'Device not present Error'.
– Bei einem SAVE

```
Benchmark                    4 MHz      1 MHz
-----------------------------------------------
1) Nur Rechnen:

10 FOR A=1 TO 1000          9,1 s      39 s
20 X=SIN(A*PI/180)
30 NEXT A                   Faktor: 4,3

2) Rechnen und Anzeigen:

10 FOR A=1 TO 1000          23,5 s     84,5 s
20 PRINT SIN(A*PI/180)
30 NEXT A                   Faktor: 3,5

3) RAM-Disk-Access:

05 OPEN 1,1,1, "TEST.DATEI"
10 FOR A=1 TO 1000
20 PRINT#1,A                3,75 s     14 s
30 NEXT A
40 CLOSE 1                  Faktor: 3,75

4):

wie 3) auf Floppy           36 s       41 s

                            Faktor: 1,14
```

Programme, die keine Ausgabe vornehmen, laufen sogar mehr als viermal schneller!

```
0001                    ;
0002                    ;   eprom start-routine
0003                    ;
0004   fe80             counter =    2
0005                    ;
0006                    ;
0007                    ;
0008   fe80
0009   fe80                         *=   $0000 - $0180
0010                    ;
0011                    ;
0012   fe80 a2 ff       reset   ldx  #255
0013   fe82 78                  sei
0014   fe83 9a                  txs
0015   fe84 e8                  inx
0016   fe85 8e 16 d0            stx  $d016
0017                    ;
0018   fe88 a0 0a               ldy  #10       ; 12 ms
0019   fe8a ca          res1    dex            ; reset-
0020   fe8b d0 fd               bne  res1      ; taster
0021   fe8d 88                  dey            ; entprell-
0022   fe8e d0 fa               bne  res1      ; schleife.
0023                    ;
0024                    ; ins eprom springen...
0025                    ;
0026   fe90 5c 96 fe 01         jmp  £1.cont
0027                    ;
0028   fe94 a2 00       cont    ldx  #0        ; ports
0029   fe96 8e 03 dc            stx  $dc03     ; initialisieren
0030   fe99 ca                  dex
0031   fe9a 8e 02 dc            stx  $dc02
0032   fe9d ea                  nop
0033   fe9e a9 7f               lda  #$7f
0034   fea0 8d 00 dc            sta  $dc00
0035   fea3 ea                  nop
0036   fea4 ea                  nop
0037   fea5 ea                  nop
0038   fea6 ad 01 dc    entpr   lda  $dc01     ; run/stop taste
0039   fea9 cd 01 dc            cmp  $dc01     ; abfragen
0040   feac d0 f8               bne  entpr     ; falls gedrueckt,
0041                    ;                      ; ...
0042   feae 29 80               and  #$80
0043   feb0 49 80               eor  #$80
0044                    ;                      ; ...
0045   feb2 4f 01 00 04         eor  £4.1      ; prozessoren
0046   feb6 8f 01 00 04         sta  £4.1      ; wechseln.
0047                    ;
0048                    ; dieser teil wird nur ausgefuehrt
0049                    ; falls der 65c816 laeuft...
0050                    ; erstmal das schreiben im ram erlauben.
0051                    ;
0052   feba a9 30               lda  #%00110000
0053   febc 8f 01 00 04         sta  £4.1
0054                    ;
0055   fec0 a2 06               ldx  #endcopy-copy+2
0056   fec2 bf 86 ff 01 cont1   lda  £1.copy,x
0057   fec6 9d 3c 03            sta  828,x
0058   fec9 ca                  dex
0059   feca 10 f6               bpl  cont1
0060                    ;
0061   fecc 18                  clc            ; 16-bit
0062   fecd fb                  xce            ; modus und
0063   fece c2 30               rep  #$30      ; registers.
0064                    ;
0065   fed0 a9 ff 1f            lda  #$1fff    ; basic
0066   fed3 48                  pha            ; rom
0067   fed4 a2 00 a0            ldx  #$a000    ; kopieren.
0068   fed7 9b                  txy
0069   fed8 22 3c 03 00         jsl  £0.828
```

'PROGRAMM',1-Befehl gibt 'File open Error' an, daß die Datei 'PROGRAMM' auf der RAM-Disk schon existiert.
– 'Too many Files open Error' gibt der Rechner aus, falls schon vier Dateien auf der RAM-Disk geöffnet sind und eine fünfte geöffnet werden soll.
Die Sekundäradresse spielt bei einem OPEN dieselbe Rolle wie bei dem IEC-Bus: eine 15 kennzeichnet den Kommandokanal. Um den 'Umstieg' auf die RAM-Disk zu vereinfachen, gelten ähnliche Kommandos wie beim 1541-Laufwerk:

– OPEN 1,1,15,'N'+CHR$(MEM) bewirkt eine Initialisierung des Directory und der Blockbelegungs-Tabelle. Der Wert MEM gibt an, ab welcher physikalischen Speicheradresse die RAM-Disk Dateien speichern kann. Diese Adresse berechnet sich folgendermaßen: ADR = $40000+$800*MEM.

– OPEN 1,1,15,'S:DATEI,PRG' löscht die angegebenen Dateien. Es können beliebig viele Dateinamen angegeben werden.

– OPEN 1,1,15,'V' reorganisiert die Dateien so, daß der größtmögliche kontinuierliche Speicherbereich ab $41000 frei steht. Da die RAM-Disk immer von der obersten physikalischen Speicheradresse aus nach dem nächsten freien Block sucht – die 'oberste' Adresse hängt von der RAM-Bestückung ab –, hat dieser Befehl nur einen Sinn, wenn man Programme gelöscht hat. Beim Aufruf dieses Befehls dürfen keine Dateien auf der RAM-Disk geöffnet sein!!

Das RAM-Disk-Directory wird bei gleichzeitigem Drücken der Tasten Shift und RUN/STOP abgerufen. Die beiden Tasten bewirken einen SYS-Befehl, der in die Directory-Ausgaberoutine springt. Diese kann also auch aus Programmen angesprungen werden. Informiert wird man über die Anzahl der gespeicherten Dateien, deren Namen und deren Größe in Blöcke. Falls eine Datei zum Zeitpunkt der Directory-Ausgabe geöffnet ist, erscheinen an Stelle der Größe drei Sternchen ('*'). Zuletzt erfährt man noch, wie viele Blöcke auf der RAM-Disk nicht belegt sind.

```
0070                    ;
0071   fedc 68                  pla             ; kernal
0072   fedd a2 00 e0            ldx  #$e000     ; rom
0073   fee0 9b                  txy             ; kopieren.
0074   fee1 22 3c 03 00         jsl  £0.828
0075                    ;
0076   fee5 e2 20               sep  #$20       ; 8-bit akku
0077                    ;
0078   fee7 a9 01               lda  #1         ; change
0079   fee9 8d 3e 03            sta  830        ; source bank.
0080                    ;
0081   feec c2 20               rep  #$20       ; 16-bit akku
0082                    ;
0083   feee a9 00 00            lda  #$0000     ; ram-disk
0084   fef1 85 02               sta  counter    ; installieren
0085                    ;
0086   fef3 a6 02       trans   ldx  counter
0087   fef5 bf a8 ff 01         lda  £1.dest,x
0088   fef9 a8                  tay
0089   fefa bf 8a ff 01         lda  £1.src,x
0090   fefe 48                  pha
0091   feff bf c6 ff 01         lda  £1.len,x
0092   ff03 fa                  plx
0093   ff04 1a                  inc a           ; a = $ffff ?
0094   ff05 f0 0b               beq  trans1     ; dann fertig!
0095   ff07 22 3c 03 00         jsl  £0.828     ; zur kopier-
0096   ff0b e6 02               inc  counter    ; routine
0097   ff0d e6 02               inc  counter    ; naechste
0098   ff0f 4c f5 fe            jmp  trans      ; aenderung...
0099                    ;
0100   ff12 38          trans1  sec             ; emulation
0101   ff13 fb                  xce             ; modus.
0102                    ;
0103                    ; einfache aenderungen
0104                    ; im rom auch noch vollziehen.
0105                    ;
0106   ff14 a9 01               lda  #1         ; weisser
0107   ff16 8d 35 e5            sta  $e535      ; cursor.
0108   ff19 a9 06               lda  #6         ; blauer
0109   ff1b 8d d9 ec            sta  $ecb9+32; rahmen.
0110                    ;
0111   ff1e a2 14               ldx  #20
0112   ff20 bf 68 ff 01 cloop   lda  £1.meldung,x
0113   ff24 9d 7e e4            sta  $e47e,x ; einschalt-
0114   ff27 ca                  dex             ; meldung
0115   ff28 10 f6               bpl  cloop      ; korrigieren.
0116                    ;
0117   ff2a a2 08               ldx  #8
0118   ff2c bf 7d ff 01 cloop1  lda  £1.pdir,x
0119   ff30 9d e7 ec            sta  $ece7,x ; text bei
0120   ff33 ca                  dex             ; shift run/stop
0121   ff34 10 f6               bpl  cloop1     ; aendern.
0122                    ;
0123   ff36 a9 28               lda  #40        ; iec-schleife
0124   ff38 8d 08 ee            sta  $ee08      ; verlaengern.
0125                    ;
0126   ff3b a9 4c               lda  #$4c       ; bei listen
0127   ff3d 8d 11 ed            sta  $ed11      ; oder talk
0128   ff40 a2 f7               ldx  #$f7       ; auf iec-bus
0129   ff42 a9 35               lda  #$35       ; prozessoren
0130   ff44 8d 12 ed            sta  $ed12      ; auf 1 mhz
0131   ff47 8e 13 ed            stx  $ed13      ; schalten.
0132                    ;
0133   ff4a a9 47               lda  #$47       ; 4 mhz bei
0134   ff4c 8d 01 ee            sta  $ee01      ; unlisten oder
0135   ff4f 8e 02 ee            stx  $ee02      ; untalk moeglich?
0136                    ;
0137   ff52 a9 2c               lda  #$2c       ; notwendige
0138   ff54 8d 18 f6            sta  $f618      ; umleitung...
0139   ff57 8e 19 f6            stx  $f619
0140                    ;
0141                    ; rom-simulationsmodus waehlen.
0142                    ;
0143   ff5a a9 20               lda  #%00100000
0144   ff5c 8f 01 00 04         sta  £4.1
0145                    ;
0146   ff60 ff ff ff ff         data $ff,$ff,$ff,$ff
0147                    ;
0148   ff64 5c e2 fc 00         jmp  £0.$fce2 ; zur reset-routine.
0149                    ;
0150                    ;
0151   ff68 43 2d 36 34 meldung asc  "c-64 with 256k,65c816"
0152                    ;
0153   ff7d 53 59 53 36 pdir    data "sys62886",13
0154                    ;
0155                    ;
0156   ff86 54 00 00    copy    mvn  0,0        ; schnelle
0157   ff89 6b                  rtl             ; kopierroutine.
0158                    ;
0159   ff8a             endcopy =    *
0160                    ;
0161                    ;
0162   ff8a 00 d0 23 d0 src     wrd  $d000,$d023,$d5bc
0163   ff90 f0 d5 07 d6         wrd  $d5f0,$d607,$d610
0164   ff96 16 d6 5e d6         wrd  $d616,$d65e,$d684
0165   ff9c f4 d6 ff ff         wrd  $d6f4,$ffff,$ffff
0166   ffa2 ff ff ff ff         wrd  $ffff,$ffff,$ffff
0167                    ;
0168   ffa8 d8 f0 2c f7 dest    wrd  $f0d8,$f72c,$f179
0169   ffae e5 f1 2a f2         wrd  $f1e5,$f22a,$f26f
0170   ffb4 8b f3 c8 f2         wrd  $f38b,$f2c8,$f539
0171   ffba 5f f6 ff ff         wrd  $f65f,$ffff,$ffff
0172   ffc0 ff ff ff ff         wrd  $ffff,$ffff,$ffff
0173                    ;
0174   ffc6 21 00 97 05 len     wrd  $0021,$0597,$0032
0175   ffcc 15 00 07 00         wrd  $0015,$0007,$0004
0176   ffd2 46 00 24 00         wrd  $0046,$0024,$006e
0177   ffd8 2c 00 ff ff         wrd  $002c,$ffff,$ffff
0178   ffde ff ff ff            wrd  $ffff,$ffff,$ffff
0179                    ;
0180                    ;  reset und interrupt
0181                    ;  vektoren fangen bei
0182                    ;  x=$ffe4 an.
0183                    ;
0184   ffe4 ff ff       vectors wrd  $ffff       ; cop
0185   ffe6 ff ff               wrd  $ffff      ; brk
0186   ffe8 ff ff               wrd  $ffff      ; abort
0187   ffea ff ff               wrd  $ffff      ; nmi
0188   ffec ff ff               wrd  $ffff
0189   ffee ff ff               wrd  $ffff      ; irq
0190                    ;
0191   fff0 ff ff               wrd  $ffff
0192   fff2 ff ff               wrd  $ffff
0193   fff4 ff ff               wrd  $ffff      ; cop
0194   fff6 ff ff               wrd  $ffff
0195   fff8 ff ff               wrd  $ffff      ; abort
0196   fffa ff ff               wrd  $ffff      ; nmi
0197   fffc 82 fe               wrd  reset      ; reset
0198   fffe ff ff               wrd  $ffff      ; irq
0199                    ;
0200                    ;
```

Die Startup-Routine kopiert BASIC und Kernal ins externe RAM.