Daten

Wenn sie ein Spielmodul in Ihren Computer eingeschoben haben war das Ihr erster Kontakt zur Welt der EPROMs: Erasable Programmable Read-Only Memory. Das bedeutet, daß Daten, die in einem EPROM gespeichert sind, nur noch gelesen und erst durch ein bestimmtes Verfahren wieder gelöscht werden können.

Die Vorteile dieser Art der Datenspeicherung sind enorm:
— Das Programm ist sofort nach dem Einschalten des Computers im Speicher vorhanden.
— Besondere Funktionen, wie automatisches Laden des Directory nach dem Einschalten sind möglich.
— Problemlose Handhabung, da zum Laden eines Programms keine Kenntnisse von Programmiersprachen notwendig sind.

Die Nachteile liegen zum einen in den reltaiv hohen Kosten für die EPROMs (16 bis 160 Mark) und die notwendige Steckkarte (zirka 50 Mark). Zum anderen darin, daß es zu Überschneidungen im Speicherbereich kommen kann, wenn Programme von Diskette nachgeladen werden.

Wie funktioniert aber die Datenspeicherung auf EPROMs? Damit Sie die Daten des internen Speichers Ihres Computers in den Speicher eines EPROMs übertragen können, brauchen Sie einen »EPROM-Brenner«. Damit wird die Verbindung zwischen Computer und EPROM hergestellt und für die notwendige Programmierspannung gesorgt.

Das einzige, was jetzt noch fehlt, ist ein Programm, das die Datenübertragung steuert. Das Prinzip der EPROM-Programmierung beruht darauf, daß Ladungen in die Speicherzellen des EPROMs transportiert werden. Dabei wird nur zwischen zwei Ladungszuständen unterschieden: geladen und ungeladen. Die einzelnen Ladungszustände werden vom Computer entweder als logische 1 (high) oder logische 0 (low) interpretiert. Jede dieser Speicherzellen enthält somit ein Bit. Auch in EPROMs werden Daten in binärer Weise gespeichert. Die Speicherzellen eines neuen EPROMs sind normalerweise ungeladen (logische 1). Wird nun eine Programmierspannung (zwischen 12,5 und 25 V) auf eine dieser Zellen gelegt, so ändert sich ihr Potential, sie wird »geladen« (logisch 0). Dabei werden die am Datenbus des EPROMs anliegenden Daten in die durch den Adreßbus angegebene Adresse des EPROMs übernommen. Es können keine Bits, die durch eine bereits vorgenommene Programmierung auf logisch 0 (low) gesetzt sind, beim Programmieren wieder in logisch 1 (high) umgewandelt werden. Damit die Ladung der Speicherzellen auch nach dem Wegnehmen der Programmierspannung erhalten bleibt, ist jede Zelle von einer semipermeablen Isolierschicht umgeben. Entsprechend dem Programm wird so, immer acht Speicherzellen (ein Byte) auf einmal, das gesamte Programm aus dem Computer in den EPROM übertragen.

Natürlich hängt die Länge des übertragbaren Programms von der Speicherkapazität der verwendeten EPROMs ab. Die meistverwendeten Typen können dabei zwischen 2 und 16 KByte speichern, es gibt aber auch schon EPROMs mit 32 KByte. Ein Programm mit 8 KByte Länge kann so entweder auf einem 8-KByte-EPROM oder auf zwei 4-KByte-EPROMs gespeichert werden.

Wie das »Erasable« im Namen der EPROMs schon andeutet, ist der Speicherinhalt nicht auf alle Zeiten festgeschrieben. Der Inhalt eines EPROMs wird durch Bestrahlung mit ultraviolettem Licht wieder gelöscht. Dazu ist im Gehäuse des EPROMs ein rundes Fenster ausgespart, durch das die UV-Strahlen auf den Chip einwirken können. Beim Löschen wird die Isolationsschicht der Speicherstellen in beide Richtungen durchlässig, was eine Entladung zur Folge hat.

Der EPROM ist danach wieder programmierbar. Ein EPROM kann so zwischen 25 und 30 mal gelöscht und neu programmiert werden. Am billigsten ist es, die EPROMs zum Löschen einfach in die Sonne zu legen, leider auch am langsamsten. Schnell geht es mit einem speziellen EPROM-Löschgerät, das ist der teuerste Weg. Im Normalfall reicht eine einfache Höhensonne. Die Löschzeit beträgt dann, je nach Entfernung der EPROMs zur UV-Quelle, zwischen 15 und 25 Minuten.

Nun aber zur Praxis. Mit dem abgedruckten Assemblerlisting ist es möglich, jedes beliebige Basicprogramm in EPROMs zu brennen. Die Programme werden damit automatisch gestartet und sind gegen Programmabruch geschützt. Das Programm »EPROM-Maker« steht im Speicherbereich zwischen $8000 und $8100 und muß vor jedem anderen Programm in die Speicherzellen 0 bis 100 des EPROMs gebrannt werden. In Adresse $8004 beginnt die Meldung »CMB80«, die das Betriebssystem während seiner Initialisierungsroutine abfragt. Dadurch wird bewirkt, daß in die, in den Speicherstellen $8000 und $8001 abgelegten Adresse, ($800A) verzweigt wird. Dort wird dann der eigentliche Basic-Lader nach $C000 geladen. Vier Speicherstellen sind vor dem jeweiligen Brennen des EPROMs zu verändern. Dazu gehen Sie wie folgt vor:

1. Laden sie das zu brennende Basicprogramm ganz normal in den Speicher.
2. Ermitteln Sie den Inhalt der Speicherstellen 2049, 2050, 45 und 46 mittels PEEK.
3. Rechnen Sie die Werte in Hexadezimalzahlen um und schreiben Sie diese auf.
4. Laden Sie nun einen Monitor
5. Laden Sie den EPROM-Maker und starten Sie den Monitor
6. Schreiben Sie den Wert der Speicherzelle 2049 in den LDA-Befehl in 80AC, den von 2050 in 80B1, den von 45 in 80B6 und den von 46 in 80BA.

Sie haben dann ein individuelles Startprogramm für Ihr Basicprogramm.

7. Laden Sie die Treibersoftware für Ihren EPROM-Brenner.
8. Übertragen Sie Ihren EPROM-Maker in die ersten 100 Speicherzellen des EPROMs.
9. Nun können Sie Ihr Basicprogramm (ab $0800) mit EPROM-Start 100 brennen. Reine Maschinenprogramme können Sie so natürlich auch brennen, wenn Sie diese zuvor in DATA-Zeilen mit Ladeschleife umgewandelt haben.

Das Programm ist auf die Platine von M. Frank zugeschnitten, so daß bis zu 16 KByte lange Programme problemlos gebrannt werden können. Wer über die Platine nicht verfügt, ist leider auf 8 KByte beschränkt. Wenn Sie die Autostartfunktion nicht benötigen, brauchen Sie den EPROM-Maker für Maschinenprogramme natürlich nicht. Hier genügt es, den EPROM einfach ab Speicherzelle 0 mit dem gewünschten Speicherbereich zu brennen und durch SYS (Startadresse) zu starten. (A. Wängler/M. Frank/gk)

brennerei

Nicht aus Schottland und auch nicht trinkbar, aber dennoch gehaltvoll: EPROMs, fest programmierbare Speicher für Ihren C 64. Wie man sich eigene Programm-Module herstellt und was dahinter steckt, zeigt dieser Bericht.

Assemblerlisting zum Brennen eines EPROMs

```
.. 8000  0A        ASL             Autostartadresse des Programmes
.. 8001  80        ???             $ 800A
.. 8002  94 E3     STY $E3,X       E394 Adresse (Restore Taste)
.. 8004  C3        ???             Autostartkennung (CBM80)
.. 8005  C2        ???             C3 C2 CD 38 30
.. 8006  CD 38 30  CMP $3038
.. 8009  00        BRK
.. 800A  A0 C0     LDY #$C0        Eigentlicher Basicloader (steht ab 8029)
.. 800C  84 FF     STY $FF         wird nach $C000 geladen
.. 800E  A0 80     LDY #$80        ↑
.. 8010  84 FD     STY $FD         ↑
.. 8012  A0 29     LDY #$29        ↑
.. 8014  84 FC     STY $FC         ↑
.. 8016  A0 00     LDY #$00        ↑
.. 8018  84 FE     STY $FE         ↑
.. 801A  B1 FC     LDA ($FC),Y     ↑
.. 801C  91 FE     STA ($FE),Y     ↑
.. 801E  C0 D6     CPY #$D6        ↑
.. 8020  F0 04     BEQ $8026       ↑
.. 8022  C8        INY             ↑
.. 8023  4C 1A 80  JMP $801A       ↑
.. 8026  4C 00 C0  JMP $C000       Verzweigt zum Basicloader
.. 8029  A2 05     LDX #$05        BASICLOADER TEIL 1
.. 802B  8E 16 D0  STX $D016       Hier wird der 1.Teil des Basic
.. 802E  A2 00     LDX #$00        Programmes in den Basicspeicher
.. 8030  86 FC     STX $FC         (ab $0800) geladen
.. 8032  86 FE     STX $FE         ↑
.. 8034  A2 08     LDX #$08        ↑
.. 8036  86 FD     STX $FD         ↑
.. 8038  A2 81     LDX #$81        ↑
.. 803A  86 FF     STX $FF         ↑
.. 803C  A0 00     LDY #$00        ↑
.. 803E  B1 FE     LDA ($FE),Y     ↑
.. 8040  91 FC     STA ($FC),Y     ↑
.. 8042  E6 FC     INC $FC         ↑
.. 8044  A6 FC     LDX $FC         ↑
.. 8046  E0 00     CPX #$00        ↑
.. 8048  D0 02     BNE $804C       ↑
.. 804A  E6 FD     INC $FD         ↑
.. 804C  E6 FE     INC $FE         ↑
.. 804E  A6 FE     LDX $FE         ↑
.. 8050  E0 00     CPX #$00        ↑
.. 8052  D0 EA     BNE $803E       ↑
.. 8054  E6 FF     INC $FF         ↑
.. 8056  A6 FF     LDX $FF         ↑
.. 8058  E0 A0     CPX #$A0        ↑
.. 805A  D0 E2     BNE $803E       ↑
.. 805C  A9 01     LDA #$01        Platine wird auf zweiten 8kb Block
.. 805E  8D 00 DE  STA $DE00       umgeschalten
.. 8061  A2 80     LDX #$80        BASICLOADER TEIL 2
.. 8063  86 FF     STX $FF         ↑
.. 8065  84 FE     STY $FE         ↑
.. 8067  B1 FE     LDA ($FE),Y     ↑
.. 8069  91 FC     STA ($FC),Y     ↑
.. 806B  E6 FC     INC $FC         ↑
.. 806D  A6 FC     LDX $FC         ↑
.. 806F  E0 00     CPX #$00        ↑
.. 8071  D0 02     BNE $8075       ↑
.. 8073  E6 FD     INC $FD         ↑
.. 8075  E6 FE     INC $FE         ↑
.. 8077  A6 FE     LDX $FE         ↑
.. 8079  E0 00     CPX #$00        ↑
.. 807B  D0 EA     BNE $8067       ↑
.. 807D  E6 FF     INC $FF         ↑
.. 807F  A6 FF     LDX $FF         ↑
.. 8081  E0 A0     CPX #$A0        ↑
.. 8083  D0 E2     BNE $8067       ↑
.. 8085  A2 02     LDX #$02        1. 8KB Block der Platine wird wieder
.. 8087  8E 00 DE  STX $DE00       eingeschalten
.. 808A  78        SEI
.. 808B  20 A3 FD  JSR $FDA3       Initialisierung des Betriebssystems
.. 808E  20 50 FD  JSR $FD50       ↑
.. 8091  20 15 FD  JSR $FD15       ↑
.. 8094  20 5B FF  JSR $FF5B       ↑
.. 8097  20 18 E5  JSR $E518       ↑
.. 809A  58        CLI             ↑
.. 809B  20 53 E4  JSR $E453       Initialisierung von Basic
.. 809E  20 BF E3  JSR $E3BF       ↑
.. 80A1  20 22 E4  JSR $E422       ↑
.. 80A4  A2 FB     LDX #$FB        ↑
.. 80A6  9A        TXS             ↑
.. 80A7  A9 EF     LDA #$EF        ↑
.. 80A9  8D 28 03  STA $0328       ↑
.. 80AC  A9 0E     LDA #$0E        Hier werden die aktuellen Programmzeiger
.. 80AE  8D 01 08  STA $0801       geschrieben ,die fuer jedes Programm neu
.. 80B1  A9 08     LDA #$08        ermittelt werden muessen. (Siehe
.. 80B3  8D 02 08  STA $0802       Text)PEEK 2049,PEEK 2050 => 2.
.. 80B6  A9 D3     LDA #$D3        Programmzeile
.. 80B8  85 2D     STA $2D
.. 80BA  A9 46     LDA #$46        PEEK 45 ,PEEK 46 => Beginn der Variablen
.. 80BC  85 2E     STA $2E
.. 80BE  A9 8B     LDA #$8B
.. 80C0  8D 00 03  STA $0300
.. 80C3  A9 E3     LDA #$E3
.. 80C5  8D 01 03  STA $0301
.. 80C8  A9 E3     LDA #$E3
.. 80CA  8D 07 03  STA $0307
.. 80CD  8D 33 03  STA $0333
.. 80D0  A9 34     LDA #$34
.. 80D2  8D 06 03  STA $0306
.. 80D5  8D 32 03  STA $0332
.. 80D8  A9 02     LDA #$02
.. 80DA  8D 00 DE  STA $DE00
.. 80DD  A9 52     LDA #$52        Ablegen des Befehls RUN + chr$(13) im
.. 80DF  8D 77 02  STA $0277       Tastaturpuffer und Einsprung in die Basic
.. 80E2  A9 55     LDA #$55        Eingaberoutine (Autostart)
.. 80E4  8D 78 02  STA $0278
.. 80E7  A9 4E     LDA #$4E
.. 80E9  8D 79 02  STA $0279
.. 80EC  A9 0D     LDA #$0D
.. 80EE  8D 7A 02  STA $027A
.. 80F1  A9 04     LDA #$04
.. 80F3  85 C6     STA $C6
.. 80F5  4C 74 A4  JMP $A474
.. 80F8  00        BRK
.. 80F9  00        BRK
.. 80FA  00        BRK
.. 80FB  00        BRK
.. 80FC  00        BRK
.. 80FD  00        BRK
.. 80FE  00        BRK
.. 80FF  00        BRK
.. 8100  01 00     ORA ($00,X)
.
```