SMON - komplett

Die Stärken dieses Super-Maschinensprache-Monitors sind hauptsächlich die mächtigen Such- und Trace-Befehle zum Austesten von Programmen in Maschinensprache. Neben dem kompletten Listing und der kompletten Anleitung mit einer Tabelle sämtlicher Funktionen und einer Tabelle wichtiger Einsprungadressen finden Sie zwei Erweiterungen, einen vollständigen Diskmonitor und einen Disassembler, der auch illegale Opcodes disassembliert. Ein Programm, mit dem auch Profis gerne arbeiten.

Ich kann mich noch gut an unsere ersten Schritte in Maschinensprache erinnern. Ausgerüstet mit einer Befehlsliste für den 6502 und einem in Basic geschriebenen »Mini-Monitor« entstanden Programme, die 3 und 5 addieren und das Ergebnis im Speicher ablegen konnten. Dazu mußten wir die Befehlcodes aus der Liste heraussuchen und dann in den Speicher »POKEn«. Jeder Sprung mußte von Hand ausgerechnet werden, jeder falsch herausgesuchte Befehl führte zum Programmabsturz. Der erste Disassembler - ein Programm zur Anzeige der Maschinenbefehle in Assemblersprache - war für uns die Offenbarung. Von nun an konnten wir Maschinenprogramme analysieren und daraus lernen. Zum Verständnis der Maschinensprache ist es nämlich noch weit mehr als bei anderen Sprachen wichtig, vorhandene Programme zu verstehen und sich dabei die wichtigsten Techniken anzueignen.

Mit der Zeit wuchsen unsere Ansprüche, ein Assembler mußte her, um die neugewonnenen Erkenntnisse auch auszuprobieren. Das war zuerst wieder ein Basic-Programm, langsam und wenig komfortabel, aber immerhin. Wir schrieben unsere ersten kleinen Routinen, vor allem, um vorhandene Maschinenprogramme unseren eigenen Wünschen anzupassen. Mit dem AMON für den VC 20 bekamen wir dann einen Monitor, der (fast) alle unsere Wünsche erfüllte. Als wir jedoch auf den C 64 umstiegen, mußten wir feststellen, daß es für diesen Computer nichts gab, das uns zufriedenstellen konnte. Der einzige Ausweg: selbst programmieren. So entstand im Laufe eines Jahres SMON. Ursprünglich hatten wir nur vor, die Funktionen von AMON für den C64 zu programmieren, aber dabei blieb es nicht. Immer neue Befehle und Routinen kamen hinzu, bis wir endlich zufrieden waren.

Was bietet SMON?

Zunächst ist alles enthalten, was zum »Standard« gehört: Memory-Dump, also die Anzeige des Speicherinhalts in Hex-Bytes, mit Änderungsmöglichkeiten, ein Disassembler mit Änderungsmöglichkeit sowie Routinen zum Laden, Abspeichern und Starten von Maschinenprogrammen. Darüber hinaus gibt es einen kleinen Direktassembler, der sogar Labels verarbeitet, Befehle zum Verschieben im Speicher mit und ohne Umrechnen der Adressen und Routinen zum Umrechnen von Hex-, Dezimal- und Binärzahlen. Der besondere Clou von SMON liegt aber zweifellos in seinen leistungsfähigen Suchroutinen und vor allem im Trace-Modus. Damit lassen sich Maschinenprogramme Schritt für Schritt abarbeiten und kontrollieren.

Der Monitor benötigt für alle Eingaben die hexadezimale Schreibweise, das heißt zu den Zahlen 1 bis 9 kommen noch die Buchstaben A (für dez. 10) bis F (für dez. 15) hinzu.

Bei der Eingabe von Adressen ist folgendes zu beachten: [ANFADR] bedeutet exakt die Startadresse, [ENDADR] bedeutet hierbei die erste Adresse hinter dem gewählten Bereich. Im Normalfall ist die Eingabe mit und ohne Leerzeichen zulässig. Beim Abweichen von dieser Regel wird darauf besonders verwiesen. Tippen Sie zuerst das Hauptprogramm (Listing 1) mit dem MSE ab. Befindet sich SMON auf Ihrer Diskette, kann er mit LOAD "SMON $C000",8,1 geladen und mit dem Befehl SYS 49152 gestartet werden. Geben Sie vor dem SYS-Befehl aber NEW ein, um einen späteren »OUT OF MEMORY« zu verhindern.

Assemblieren

A [ANFADR]

Assemblierung beginnt bei der angegebenen Adresse

Beispiel:

A 4000 Beginn bei Startadresse $4000

Nach Eingabe von »RETURN« erscheint auf dem Bildschirm die gewählte Adresse mit einem blinkenden Cursor. Die Befehle werden so eingegeben, wie sie der Disassembler zeigt: LDY #00 oder LDA 400E,Y und so weiter. »RETURN« schließt die Eingabe der Zeile ab. Bei fehlerhafter Eingabe springt der Cursor wieder in die Anfangsposition zurück. Ansonsten wird der Befehl disassembliert und nach Ausgabe der Hex-Bytes gelistet. Zur Korrektur vorhergehender Zeilen gehen Sie mit dem Cursor zur Anfangsposition (hinter die Adresse) zurück, schreiben den Befehl neu und gehen nach »RETURN« mit dem Cursor wieder in die letzte Zeile. Falls Ihnen bei Sprüngen (Branch-Befehl, JSR und JMP) die Zieladressen noch nicht bekannt sind, geben Sie einfach sogenannte »Label« ein.

Ein Label besteht aus dem Buchstaben »M« (für Marke) und einer zweistelligen Hex-Zahl von 01 bis 30.

Beispiel: BCC M01

Wenn Sie die Zieladresse für diesen Sprung erreicht haben, dann kennzeichnen Sie diese mit eben dieser »Marke«.

Beispiel: M01 LDY #00

Einzelne Bytes nimmt der Assembler an, indem Sie diese mit einem Punkt kennzeichnen: .00 oder .AB. In diesem Modus werden die Eingaben natürlich nicht disassembliert.

Nach Beendigung des Assemblierens geben Sie »F« ein. Danach sehen Sie alle Ihre Eingaben noch einmal aufgelistet und korrigieren dann bei Bedarf wie beim Disassembler (!) angegeben.

Probieren Sie einmal das folgende Beispiel:

A 4000

Der Assembler meldet sich mit: »4000« und einem blinkenden Cursor. Geben Sie nun ein (die Adressen erscheinen automatisch):

```
4000 LDY #00        4009 CPY #12
4002 LDA 400E,Y     400B BCC 4002
4005 JSR FFD2       400D BRK
4008 INY
```

Die folgenden Bytes werden wie beschrieben mit einem Punkt eingegeben. Sie werden nicht disassembliert.

```
400E .0D        4017 .54
400F .0D        4018 .20
4010 .53        4019 .53
4011 .4D        401A .55
4012 .4F        401B .50
4013 .4E        401C .45
4014 .20        401D .52
4015 .49        401E .0D
4016 .53        401F .0D
```

Drücken Sie anschließend »F«. Ihr Programm wird nochmal aufgelistet. Starten Sie es nun mit »G 4000«. Es erscheint ein Text auf dem Bildschirm - lassen Sie sich überraschen.

Disassemblieren

D [ANFADR,ENDADR]

disassembliert den Bereich von ANFADR bis ENDADR, wobei ENDADR nicht eingegeben werden muß. Wird keine Endadresse eingegeben, erscheint zunächst nur eine Zeile:

```
ADR         HEXBYTES    BEFEHL
4000        A0 00       LDY #00
```

Mit der SPACE-Taste wird der jeweils nächste Befehl in der gleichen Art und Weise gezeigt. Wünschen Sie eine fortlaufende Ausgabe, drücken Sie »RETURN«. Die Ausgabe wird dann so lange fortgesetzt, bis eine weitere Taste gedrückt wird oder bis ENDADR erreicht ist. Mit »RUN/STOP« springen Sie jederzeit in den Eingabemodus zurück.

Das Komma, das vor der Adresse auf dem Bildschirm erscheint, ist ein »hidden command« (verstecktes Kommando). Es braucht nicht eingegeben zu werden, da es automatisch beim Disassemblieren angezeigt wird. So ermöglicht es ein einfaches Ändern des Programms. Fahren Sie mit dem Cursor auf den zu ändernden Befehl und überschreiben Sie ihn mit dem neuen. Wenn Sie jetzt »RETURN« drücken, erkennt SMON das Komma als Befehl und führt ihn im Speicher aus. Achten Sie aber darauf, daß der neue Befehl die gleiche Länge (in Byte) hat und füllen Sie gegebenenfalls mit »NOPs« auf. Zur Kontrolle können Sie den geänderten Bereich noch einmal disassemblieren.

Lassen Sie als Beispiel einmal das Programm (siehe Befehl »A«) ab 4000 disassemblieren (»D 4000 4011«). Ändern Sie nun den ersten Befehl auf LDY #01. Die Änderung zeigt sich daran, daß die HEX-Bytes automatisch den neuen Wert annehmen. Starten Sie nun das Programm nochmals mit »G 4000«. Jetzt erscheint der Text mit nur einer Zeile Abstand auf dem Bildschirm.

Starten eines Maschinenprogramms (Go)

G [ADRESSE]

startet ein Maschinenprogramm, das bei ADRESSE beginnt. Das Programm muß mit einem BRK-Befehl abgeschlossen werden, damit ein Rücksprung in SMON erfolgen kann. Wird nach »G« keine Adresse eingegeben, benutzt SMON die, die mit dem letzten BRK erreicht worden ist und bei der Register-Ausgabe als PC auftaucht. Mit dem »R«-Befehl (siehe unten) werden die Register vorher auf gewünschte Werte gesetzt.

Memory-Dump

M [ANFADR ENDADR]

gibt die HEX-Werte des Speichers sowie die zugehörigen ASCII-Zeichen aus. Auch hier kann auf die Eingabe einer Endadresse verzichtet werden. Die Steuerung der Ausgabe entspricht der beim Disassemblieren.

Beispiel:

M 4000 gibt die Inhalte der Speicherstellen $4000 bis $4007 aus. Weiter geht es wie beim Disassemblieren mit SPACE oder RETURN. Die Bytes können ebenfalls durch Überschreiben geändert werden, allerdings nicht die ASCII-Zeichen. Verantwortlich dafür ist der Doppelpunkt, der am Anfang jeder Zeile ausgegeben wird, ein weiterer »hidden command«. Wenn Ihre Änderung nicht durchgeführt werden kann, weil Sie zum Beispiel versuchen, ins ROM zu schreiben, wird ein »?« als Fehlermeldung ausgegeben.

Registeranzeige

R zeigt den gegenwärtigen Stand der wichtigsten 6510-Register an: Programmzähler (PC), Status-Register (SR), Akkumulator (AC), X-Register (XR), Y-Register (YR), Stackpointer (SP). Außerdem werden die einzelnen Flags des Status-Registers mit 1 für »gesetzt« und 0 für »nicht gesetzt« angezeigt. Durch Überschreiben werden die Inhalte auf einen gewünschten Wert gesetzt. Die Flags können allerdings nicht einzeln verändert werden, sondern nur durch Überschreiben des Wertes von SR.

Exit

X springt ins Basic zurück. Alle Basic-Pointer bleiben erhalten. Sie können also zum Beispiel direkt im Programm fortfahren, wenn Sie zwischendurch mit SMON einige Speicherstellen kontrolliert haben.

Probieren Sie alle bisher beschriebenen Befehle in Ruhe aus und machen Sie sich mit SMON vertraut. Arbeiten Sie auch parallel den Kurs über Assemblerprogrammierung in dieser Ausgabe durch. Alle Beispiele dort sind auf SMON abgestimmt.

I/O-SET

I0 1 legt die Device-Nummer für LOAD und SAVE auf 1 (Kassette). Jedes Laden und Abspeichern erfolgt jetzt auf das angegebene Gerät. Die voreingestellte Device-Nummer ist 8 (für die Floppy also: I0 8). Wenn Sie nur mit der Floppy arbeiten, brauchen Sie diesen Befehl also nicht.

LOAD

L"name" lädt ein Programm vom angegebenen Gerät (wie oben beschrieben) an die Originaladresse in den Speicher. Die Basic-Zeiger bleiben bei diesem Ladevorgang unbeeinflußt, das heißt, sie werden nicht verändert.

Beispiel: Unser Monitor soll an seiner Originaladresse ($C000) im Speicher stehen. Also brauchen Sie ihn nur mit »L"SMON"« zu laden, damit er dort erscheint. Wenn Sie einmal ein Programm an eine andere als die Originaladresse laden wollen, dann bietet Ihnen SMON dazu folgende Möglichkeit: »L"name" ADRESSE« lädt ein Programm an die angegebene Adresse. Nehmen Sie doch bitte noch einmal unser letztes Test-Programm und geben es mit dem Assembler ab Adresse $4000 ein. Speichern Sie es mit »S"SUPERTEST" 4000 4023« ab und laden es dann

1. an die Originaladresse (L"SUPERTEST") und
2. an eine andere Adresse (mit L"SUPERTEST"5000 zum Beispiel nach $5000).

Schauen Sie sich danach mit dem Disassembler-Befehl beide Routinen einmal an. Sie werden feststellen, daß beide Programme zwar bis auf die BRANCH-Befehle gleich aussehen, daß das Programm in $5000 aber nicht funktionieren kann, da es eine falsche Adresse verwendet (5002 LDA 400E,Y). Ein anderes Beispiel dazu: Ein Autostart-Programm beginnt bei $0120, läßt sich aber in diesem Bereich nicht untersuchen, da dort der Prozessor-STACK (im Bereich von $0100 bis $01FF) liegt, der vom Prozessor selbständig verändert wird. Wenn Sie nun L"name" 4120 eingeben, befindet sich das Programm anschließend bei $4120 (nicht an der Originaladresse $0120) und Sie können es ohne Einschränkungen - von den falschen Absolut-Adressen abgesehen - disassemblieren.

SAVE

S"name", ANFADR ENDADR speichert ein Programm von ANFADR bis ENDADR-1 unter »name« auf die Floppy ab, da diese - wie wir ja inzwischen wissen - das voreingestellte Gerät ist. Wenn Sie auf Kassette abspeichern wollen, setzen Sie vorher mit »I0 1« die Device-Nummer auf 1.

Beispiel: S"SUPERTEST"4000 4020 speichert das Programm mit dem Namen »SUPERTEST« (es steht im Speicher

von $4000 bis $401F) auf Diskette ab. Bitte beachten Sie auch bei diesem Befehl, daß die Endadresse auf das nächste Byte hinter dem Programm gesetzt wird.

Printer-Set

P0 2 setzt die Primäradresse für den Drucker auf 2. Voreingestellt ist hier die 4 als Gerätenummer (zum Beispiel für Commodore-Drucker). Vielleicht haben Sie es ja schon bemerkt: Bei allen Ausgabe-Befehlen (wie D, M etc.) können Sie auch den Drucker ansprechen, wenn Sie das Kommando geshiftet eingeben. Die Ausgabe erfolgt dann gleichzeitig auf Bildschirm und Drucker. (Beachten Sie bitte die Änderung für die Druckerausgabe am Schluß des Artikels.)

Ein bißchen Rechnerei

Die folgende Befehlsgruppe enthält Befehle zur Zahlenumrechnung. Sie wissen ja: Der Mensch mit seinen zehn Fingern neigt eher zur dezimalen Rechenweise, aber der Computer bevorzugt das Binärsystem, weil er nur zwei Finger hat (siehe Netzstecker). Ein Kompromiß ist das Hexadezimalsystem, denn das versteht keiner von beiden. Um Verständnisschwierigkeiten mit Ihrem Liebling aus dem Weg zu gehen, haben Sie aber SMON.

Umrechnung Dez→Hex

(Dezimalzahl) rechnet die Dezimalzahl in die entsprechende Hexadezimalzahl um. Hierbei können Sie die Eingabe in beliebiger Weise vornehmen, da SMON Zahlen bis 65 535 umrechnet. Beispiel: #12, #144, #3456, #65533 und so weiter.

Umrechnung Hex→Dez

$ (Hexadezimalzahl) rechnet die Hexadezimalzahl in die entsprechende Dezimalzahl um. Die Eingabe muß hierbei zweistellig beziehungsweise vierstellig erfolgen. Ist diese Zahl kleiner als $100 (=255), wird zusätzlich auch der Binärwert ausgegeben.

Beispiel: $12, $0012, $0D, $FFD2 etc. In den ersten drei Beispielen erfolgt die Anzeige auch in binärer Form.

Umrechnung Binär→Hex,Dez

% (Binärzahl (achtstellig)) rechnet die Binärzahl in die entsprechenden Hexa- und Dezimalzahlen um. Bei diesem Befehl müssen Sie genau acht Binärzahlen eingeben. Falls Sie einmal versehentlich mehr eingeben sollten, werden nur die ersten acht zur Umrechnung herangezogen. Beispiel: %00011111, %10101011

Add-Sub

? 2340+156D berechnet die Summe der beiden vier (!)-stelligen Hex-Zahlen. Neben der Addition ist auch Subtraktion möglich.

Programme auf dem Rangierbahnhof

Occupy (Besetzen)

O (ANFADR ENDADR HEX-Wert) belegt den angegebenen Bereich mit dem vorgegebenen HEX-Wert. Beispiel: O 5000 8000 00 füllt den Bereich von $5000 bis $7FFF mit Nullen.

Man kann mit »OCCUPY« aber nicht nur Speicherbereiche löschen, sondern auch mit beliebigen Werten belegen. Häufig hat man das Problem, festzustellen, welcher Speicherplatz von einem Programm wirklich benutzt wird. Wir füllen den in Frage kommenden Bereich dann zuerst zum Beispiel mit »AA« und laden dann unser Programm. Probieren Sie bitte das folgende Beispiel: Füllen Sie den Speicherbereich von $3000 bis $6000 mit $AA und laden Sie dann unser SUPERTEST-Programm. Beim Disassemblieren können Sie erkennen, daß unser kleines Programm exakt zwischen vielen »AA« eingebettet ist.

Write

W (ANFADRalt ENDADRalt ANFADRneu) verschiebt den Speicherbereich von ANFADRalt bis ENDADRalt nach ANFADRneu ohne Umrechnung der Adressen! Unser kleines Testprogramm möge noch einmal als Beispiel dienen: W 4000 4020 6000 verschiebt das oben angesprochene Programm von $4000 nach $6000.

Hierbei werden weder die absoluten Adressen umgerechnet noch die Tabellen geändert. Letzteres ist sicherlich erwünscht, aber denken Sie daran, daß das verschobene Programm nun nicht mehr lauffähig ist, da die absoluten Adressen nicht mehr stimmen (zum Beispiel bei dem Befehl LDA 400E,Y). Falls Sie jetzt »G6000« eingeben, um das Programm zu starten, werden Sie sich sicherlich wundern, daß es dennoch läuft. Doch löschen Sie einmal das Programm in $4000 (mit »O4000 4100 AA«) und starten das Programm in $6000 noch einmal! Seltsam, nicht? Abhilfe schafft der nächste Befehl.

Variation

V (ANFADRalt ENDADRalt ANFADRneu ANFADR ENDADR) rechnet alle absoluten Adressen im Bereich von ANFADR bis ENDADR, die sich auf ANFADRalt bis ENDADRalt beziehen, auf ANFADRneu um. Kompliziert? Nicht, wenn Sie sich klarmachen, daß die ersten drei Adressen exakt den Eingaben beim »W«-Befehl entsprechen. Neu hinzu kommen nur die beiden Adressen für den Bereich, in dem die Änderung tatsächlich erfolgt.

Um unser mit »W« schon verschobenes Programm auch wieder lauffähig zu machen, geben Sie folgendes ein: V 4000 4020 6000 6000 600E. Damit werden alle Absolutadressen, die im Bereich von $6000 bis $600E - dahinter steht die Tabelle - liegen und sich bisher auf $4000 bis $4020 bezogen haben, auf den neuen Bereich umgerechnet. Probieren geht wie immer über kapieren.

Eine Zusammenfassung dieser beiden Befehle ermöglicht:

Convertieren

(Verschieben eines Programmes mit Adreßumrechnung.)

C (ANFADRalt ENDADRalt ANFADRneu ANFADRges ENDADRges) verschiebt das Programm von ANFADRalt bis ENDADRalt zur ANFADRneu und zwar mit Umrechnung der Adressen zwischen ANFADRges und ENDADRges

An unserem kleinen Testprogramm läßt sich wieder einmal demonstrieren, wie der Befehl eingesetzt wird. Laden Sie es also mit »L"SUPERTEST"« und schauen es mit »D 4000« an. Jetzt wollen wir an der Adresse $4008 einen 3-Byte-Befehl einfügen: C 4008 4020 400B 4000 4011 verschiebt das Programm von $4008 bis $4020 zur neuen Anfangsadresse $400B. Dabei werden im Bereich von $4000 bis $4011 (neue Endadresse des »aktiven« Programmes!) die Sprungadressen umgerechnet. Nun können Sie ab Adresse $4008 einen 3-Byte-Befehl einfügen, zum Beispiel STY 0286. Dazu geben Sie bitte ein:

```
A 4008
4008 STY 0286
F
```

Überzeugen Sie sich davon, daß SMON die Befehle korrekt umgerechnet hat, indem Sie unser Beispiel disassemblieren (D 4000) und anschließend mit G 4000 starten. Besitzer eines Farbmonitors werden in helle Begeisterung ausbrechen. Vorsicht ist geboten, wenn Tabellen oder Text vorhanden sind. SMON wird versuchen, diese als Befehle zu disassemblieren und gegebenenfalls umzurechnen. Dabei können unvorhersehbare Verfälschungen auftreten. Aus diesem Grunde ist im Beispiel die Endadresse des zu ändernden Bereiches auf $4011 und nicht etwa auf $4023 gelegt worden. Wenn Sie größere Programme zu verschieben haben, sollten Sie die Kommandos W und V anwenden beziehungsweise einen Assembler einsetzen (zum Beispiel Hypra-Ass), der es Ihnen gestattet, beliebige Einfügungen,

Verschiebungen und sonstige Änderungen vorzunehmen. Das C-Kommando eignet sich in erster Linie für kleinere Änderungen innerhalb eines Programms.

BASIC-DATA

B (Anfadr Endadr)

wandelt das Maschinenprogramm von ANFADR bis ENDADR-1 in Basic-DATA-Zeilen um.

B 4000 4020

Unser Testprogramm wird in DATA-Werte umgerechnet und dann mit Zeilennummer 32000 beginnend im Basic-Speicher abgelegt. Ein im Speicher befindliches Basic-Programm (zum Beispiel ein Basic-Lader) mit kleineren Zeilennummern kann dann diese DATA-Zeilen benutzen.

Wenn Sie das Testprogramm wie oben beschrieben umgewandelt haben, überzeugen Sie sich mit »LIST« von der Ausführung. Dann können Sie folgendes eingeben:

10 FOR I=16384 TO 16415 : READ D :POKE I,D : NEXT

In Verbindung mit den oben erzeugten DATA-Zeilen (und RUN!) hätten Sie wieder das ursprüngliche Maschinenprogramm im Speicher. Falls Sie dieses Beispiel durchführen wollen, denken Sie bitte daran, daß Sie nach Erstellung der DATAs das Originalprogramm zum Beispiel mit OCCUPY (O 4000 4020 AA) überschreiben, damit Sie die richtige Ausführung überprüfen können. Der BRK-Befehl am Ende des Testprogramms bewirkt einen Sprung zum SMON zurück. Wollen Sie ein Maschinenprogramm von Basic aus starten und auch wieder dorthin zurückgelangen, muß der letzte Befehl ein RTS sein. Probieren Sie es aus, indem Sie das Basic-Programm um 20 SYS 16384 erweitern.

KONTROLLE

K (Anfadr Endadr)

listet die ASCII-Zeichen im gewünschten Bereich. Es werden jeweils 32 Zeichen pro Zeile ausgegeben, so daß man sich einen schnellen Überblick über Texte oder Tabellen verschaffen kann.

Beispiel: K 4000 listet die ersten 32 Zeichen unseres Programms. Die weitere Ausgabe ist genau wie beim Disassemblieren durch Druck auf SPACE oder RETURN möglich. Auch hier können Sie wie bei den anderen Bildschirm-Ausgabebefehlen Änderungen durch einfaches Überschreiben vornehmen (natürlich nicht im ROM und nur mit ASCII-Zeichen!).

Als Beispiel wollen wir einmal im Basic »herumpfuschen«. Das geht natürlich nicht so ohne weiteres, weil das Basic im ROM steht und damit nicht verändert werden kann. Tippen Sie bitte folgendes ein:

W A000 C000 A000

Auf den ersten Blick eine unsinnige Anweisung; der Speicher soll von A000 bis C000 nach A000 verschoben werden. Dieser Befehl entspricht exakt der Basic-Schleife

FOR I = 40960 TO 49152 : POKE I, PEEK (I) : NEXT

Nun ist es aber so, daß beim PEEK das ROM gelesen, beim POKE aber ins darunterliegende RAM geschrieben wird. Wir erreichen also, daß das Basic ins RAM kopiert wird. Jetzt müssen wir dafür sorgen, daß das Betriebssystem sein Basic aus dem RAM und nicht aus dem ROM holt. Zuständig dafür ist die Speicherstelle 0001. Geben Sie bitte »M 0001« ein und überschreiben Sie die »37« mit »36«.

Es passiert gar nichts. Jetzt tritt unser K-Kommando in Aktion. Geben Sie ein: K A100 A360

Was Sie sehen, sind die Basic-Befehlswörter und -Meldungen. Schalten Sie mit SHIFT/CBM auf Kleinschrift, dann erkennen Sie, daß der jeweils letzte Buchstabe eines Befehlswortes groß geschrieben ist (Endekennung). Jetzt ändern Sie durch Überschreiben das »LIST« (A100) in »LUST« und »ERROR« (A360) in »FAELER«. (Bei »FAELER« müssen Sie ein Zeichen vor »ERROR« beginnen, sonst paßt es nicht.) Verlassen Sie jetzt SMON mit »X« und geben Sie danach ein:

POKE 1,54

SMON schaltet nämlich beim »X«-Befehl immer auf das Basic-ROM zurück, daher müssen wir wieder auf unser geändertes Basic umschalten. Schreiben Sie nun einen Basic-Dreizeiler und versuchen Sie, diesen zu LISTen. Ergebnis? Versuchen Sie es jetzt einmal mit »LUST«. Ihrer weiteren Phantasie sind keine Grenzen mehr gesetzt...

Wie oben angesprochen stellt SMON eine Reihe verschiedener Suchroutinen zur Verfügung, die im folgenden an vielen Beispielen beschrieben werden. Alle diese Befehle bestehen aus zwei Zeichen und beginnen mit dem Buchstaben »F«.

FIND

F (HEX-WERT(e), Anfadr Endadr)

sucht nach einzelnen HEX-Werten innerhalb eines bestimmten Bereichs. Das zweite Zeichen (hinter F) ist hier ein Leerzeichen und darf nicht weggelassen werden! Die Bereichsangabe kann wie bei allen folgenden Befehlen entfallen, dann wird der gesamte Speicher durchsucht.

Beispiel: Wir suchen alle Befehle LDY #01, also die Werte A0 01 im Bereich von $2000 bis $6000.

F A0 01, 2000 6000 (die Leerzeichen zwischen den Hex-Bytes dürfen nicht weggelassen werden!). Es erscheinen alle Speicherstellen, die die gesuchten Bytes enthalten, also zum Beispiel 4000.

FA (Adresse, Anfadr Endadr)

sucht alle Befehle, die eine bestimmte Adresse als Operanden haben (absolut). Die Adresse braucht nicht vollständig angegeben zu werden, es kann das Jokerzeichen »*« benutzt werden.

1. Beispiel: Wir suchen alle JSR FFD2-Befehle im Bereich $2000 bis $6000.

FAFFD2,2000 6000

Es erscheinen alle Befehle disassembliert, die FFD2 im Operanden enthalten (also auch LDA FFD2 oder STA FFD2,Y...).

2. Beispiel: Wir suchen alle Befehle, die auf den Grafikbereich ($D000 bis $DFFF) zugreifen.

FAD* * *,2000 6000

Der Joker kann aber auch zum Beispiel zur Suche im Bereich $D000 bis $D0FF dienen: FAD0* *,2000 6000

FR (Adresse, Anfadr Endadr)

sucht nach relativen Sprungzielen. Anders als bei absoluten Sprüngen (JMP, JSR) benutzen die Branch-Befehle eine relative Adressierung, also zum Beispiel »Verzweige 10 vor« oder »37 zurück«. Solche Sprünge lassen sich mit dem FA-Kommando nicht finden. Hier wird »FR« eingesetzt.

Beispiel: Gesucht werden alle Branch-Befehle, die die Adresse $4002 anspringen.

FR4002,2000 6000

Natürlich können solche Befehle nur höchstens 128 Byte vom Sprungziel entfernt sein. Die Bereichsangabe ist hier also viel zu groß gewählt (SMON stört dies allerdings nicht). Der Einsatz des Jokers ist hier ebenfalls wie oben beschrieben möglich.

FT (Anfadr Endadr)

sucht Tabellen im angegebenen Bereich. SMON behandelt dabei alles, was sich nicht disassemblieren läßt, als Tabelle. Beispiel: Wir suchen Tabellen oder Text im Bereich $2000 bis $6000.

FT 2000 6000

FZ (Adr, Anfadr Endadr)
sucht alle Befehle, die Zeropage-Adressen haben.
1. Beispiel: FZC5,2000 6000 findet alle Befehle, die C5 adressieren, also zum Beispiel BIT $C5, LDA (C5), Y etc.
2. Beispiel: FZF*,2000 6000 findet alle Befehle, die den Bereich zwischen $F0 und $FF adressieren.
3. Beispiel: FZ**,2000 6000 findet sämtliche Befehle mit Zeropage-Adressierung.

FI (Operand, Anfadr Endadr)
sucht alle Befehle mit unmittelbarer Adressierung (immediate).
Beispiel: Gesucht werden Befehle, die zum Beispiel das Y-Register mit 01 laden. FI01,2000 6000 findet LDY #01 in Adresse $4000.
Sie sehen, SMON bietet eine Fülle von verschiedensten FIND-Routinen, mit denen alles gesucht und auch gefunden (!) werden kann.

= 4000 6000
vergleicht den Speicherinhalt ab $4000 mit dem ab $6000. Das erste nicht übereinstimmende Byte wird angezeigt und der Vergleich wird abgebrochen.
Wenn Sie also ein Maschinenprogramm geschrieben und überarbeitet haben und Sie wissen nicht mehr genau, worin eigentlich der Unterschied zwischen der 76. und der 77. Version besteht, gehen Sie so vor: Laden Sie zuerst Version 76 und verschieben Sie diese mit dem »W«-Befehl in einen freien Speicherbereich. Laden Sie dann Version 77 und führen Sie den »=«-Befehl durch. Sofort finden Sie den Unterschied und können mit der Arbeit an Version 78 beginnen ...
Wir wollen uns bei der Beschreibung der Trace-Befehle auf Anwendungsbeispiele konzentrieren. Zum Aufbau der Routine sei nur so viel gesagt: Gesteuert wird sie mit Hilfe des Prozessor-Interrupts, weil nur damit ein Eingriff ins laufende Maschinenprogramm möglich ist. Während des Trace-Ablaufs wird deswegen der Bildschirm kurzfristig aus- und eingeschaltet, weil alle anderen Interruptanforderungen wie zum Beispiel durch den Video-Chip, verhindert werden müssen. Da die Befehle eines Programms nicht nur angezeigt, sondern auch wirklich ausgeführt werden, ist der »SEI«-Befehl mit großer Vorsicht zu verwenden. Doch dazu später mehr. Wir wollen ein neues, besser geeignetes Beispiel verwenden als bisher. Tippen Sie also das folgende Miniprogramm mit dem Assembler ein (A 4000):

```
4000  LDA  #30    lade den Akku mit (ASCII-) 0
4002  JSR  FFD2   gib Akku auf dem Bildschirm aus
4005  CLC
4006  ADC  #01    erhöhe Akku um 1
4008  CMP  #39    vergleiche Akku mit (ASCII-) 9
400A  BCC  4002   springe, wenn Akku kleiner, zurück
400C  BRK         springe in SMON zurück
```

Starten Sie das Programm mit »G 4000«. Es muß die Zahlen von 0 bis 8 auf den Bildschirm schreiben.

◆ Trace-Stop
TS (Startadresse Stoppadresse)
Starten Sie nun unser Programm mit TS 4000 4009. Die ersten Befehle werden ausgeführt (die Null ausgegeben, der Akku erhöht etc.), dann stoppt das Programm bei Adresse $4009 und springt in die Registeranzeige.
Genau genommen ist »TS« gar kein Trace-Befehl, das Programm läuft nämlich bis zur gewählten Stoppadresse in Echtzeit durch. Dort angekommen, können Sie die Register prüfen und gegebenenfalls durch Überschreiben ändern. Mit »G«, »TW« oder »TB« (wird später erklärt) ohne weitere Adresseneingaben können Sie dann im Programmlauf fortfahren. SMON merkt sich nämlich, wo er stehengeblieben ist und arbeitet ab dieser Adresse weiter, wenn Sie nicht eine neue angeben.
Sinnvoll ist dieser Befehl immer dann, wenn in einem längeren Programm nur bestimmte Teile »getraced« werden sollen, der Anfang aber durchlaufen werden muß, um Variable zu setzen oder Benutzereingaben zu erfragen. Auch wenn man nicht ganz sicher ist, ob eine bestimmte Passage überhaupt jemals durchlaufen wird, kann man das mit »TS« überprüfen.
Zwei Einschränkungen gibt es allerdings wegen der Arbeitsweise dieses Befehls: SMON setzt im Programm an die Stoppadresse einen BRK-Befehl und merkt sich, welcher Befehl dort stand, um ihn wieder zurückzuschreiben. Deshalb funktioniert »TS« nur im RAM, nicht aber zum Beispiel im Basic oder im Betriebssystem. Auch darf die Speicherstelle, in der sich SMON den ausgetauschten Befehl merkt ($02BC) vom Programm nicht verändert werden, sonst ist eine korrekte Reparatur nicht mehr möglich.
Der wohl am häufigsten und vielseitigsten eingesetzte Trace-Befehl ist sicherlich »TW«.

◆ Trace Walk
TW (Startadresse)
Starten Sie unser Beispiel jetzt mit TW 4000
Der erste Befehl (LDA #30 in Adresse $4000) wird ausgeführt, SMON stoppt und zeigt dann die Inhalte aller Register in der gleichen Reihenfolge wie beim »R«-Kommando sowie den nächsten Befehl an. Im Akku steht jetzt 30, der Programmzähler zeigt auf $4002. Jetzt drücken Sie eine Taste. Der nächste Befehl (JSR FFD2) wird ausgeführt, der Programmzähler zeigt auf $FFD2. Achten Sie auf den Stackpointer: Sein Inhalt hat sich um 2 vermindert, weil der Prozessor auf dem Stack die Adresse abgelegt hat, an die er nach Beendigung der Subroutine zurückspringen soll. Der nächste angezeigte Befehl ist ein indirekter Sprung über $0326. Mit dem nächsten Tastendruck wird er durchgeführt.
Und so geht es munter weiter. Verzweifeln Sie nicht, wenn Sie auch nach den nächsten zehn Tastendrücken immer noch irgendwo im Betriebssystem »herumtracen« und von unserem Beispielprogramm weit und breit nichts mehr zu sehen ist. Ausnahmsweise ist unser Liebling einmal nicht im »Land der Träume« verschwunden, sondern tut, was er soll: Er arbeitet brav einen Befehl nach dem anderen ab, was zur Routine $FFD2 gehört, und das ist reichlich viel. Also bewegen Sie Ihre Finger, Sie haben's ja nicht anders gewollt. Irgendwann einmal, nach mehreren hundert gedrückten Tasten, befinden Sie sich plötzlich wieder in der Registeranzeige von SMON. Das Programm ist beendet. Nun werden Sie enttäuscht fragen, was man wohl mit einem Trace-Modus anfangen soll, der schon bei kleinsten Beispielprogrammen ein völlig undurchschaubares Chaos erzeugt? Nur Geduld, die Rettung naht in Gestalt der Taste »J«.
Falls ihre Hand noch nicht in Gips liegt, starten Sie das Ganze nochmal von vorn mit »TW 4000«. Diesmal drücken Sie aber jedesmal, wenn als nächster Befehl »JSR FFD2« angezeigt wird, auf »J«. Der Effekt ist, daß die gesamte Subroutine auf einen Schlag abgearbeitet wird und Sie sofort wieder auf dem nächsten Befehl unseres Beispiels landen. Daß wir nicht gemogelt und die Befehle von »JSR FFD2« einfach unterschlagen haben, sehen Sie daran, daß der Akku tatsächlich auf dem Bildschirm ausgegeben worden ist (rechts neben FFD2). Jetzt können Sie unser Beispiel in aller Ruhe bis zum Ende durchgehen und verfolgen, wie der Akku erhöht wird, wie der Vergleich das Statusregister beeinflußt und wie entsprechend der Rücksprung in die Schleife erfolgt.
Sie dürfen die »J«-Taste auch dann benutzen, wenn Sie schon mitten in der Subroutine sind. Aber hierbei ist äußerste Vorsicht geboten: Die Rücksprungadresse muß unbedingt oben auf dem Stack liegen, wenn Sie »J« drücken. Hat nämlich der Prozessor Werte auf dem Stack abgelegt (mit PHA oder PHP), dann erfolgt der Sprung irgendwo hin, nur nicht zurück ins Programm. Achten Sie deshalb genau auf die Anzeige des Stackpointers. Wenn dessen Wert genau so

groß ist wie bei Beginn der Subroutine, kann nichts passieren. Sonst hilft nur noch der Reset-Taster, den Sie ja inzwischen hoffentlich eingebaut haben, oder eine ruhige Hand, die die Büroklammer an Pin 1 und 3 des User-Ports hält (Kostenpunkt der Reparatur bei Abrutschen liegt bei zirka 100 Mark ...).

»TW« bricht automatisch mit der Registeranzeige ab, wenn im Programm ein »BRK«-Befehl auftaucht. Wenn Ihnen das zu lange dauert oder Sie zwischendurch ein Register ändern möchten, können Sie den Trace-Modus jederzeit mit der Stopp-Taste verlassen. Anschließend können Sie wie bei »TS« beschrieben fortfahren.

Im Gegensatz zu »TS« können Sie mit »TW« auch im ROM herumstöbern; Sie haben es ja bei der Subroutine $FFD2 bereits getan. Einzige Einschränkung beim »TW«-Befehl: Ihr Programm darf keinen »SEI« enthalten, da dieser den Interrupt und damit auch den Trace-Modus lahmlegt. Verlassen Sie in diesem Falle »TW« mit STOP und starten erneut hinter dem »SEI«-Befehl. Allerdings müssen Sie in Kauf nehmen, daß das Programm normalerweise nicht mehr korrekt arbeitet.

Das nächste Programm soll als weiteres Beispiel für den TW-Modus dienen. Geben Sie es folgendermaßen ein:

```
5000  LDA  #00   lädt den Akku mit »0«
5002  TAX        überträgt den Akku ins X-Register
5003  .0C        ein mysteriöses Byte
5004  LDA  #04   lädt den Akku mit »4«
5006  TAY        überträgt den Akku ins Y-Register
5007  BRK        springt in SMON
```

Wenn wir dieses kleine Programm abarbeiten, müßte das X-Register auf »0« stehen, während Akku und Y-Register mit »4« geladen sind. Starten wir also das Programm mit »G 5000« und schauen uns die Register an.

Seltsamerweise enthalten alle Register eine »0«. Vorsichtig, wie wir sind, überschreiben wir die drei Register mit »FF«, um die Veränderung deutlich kontrollieren zu können.

Dann starten wir mit »G 5000« ein zweites Mal. Gegen alle Gesetze der Vernunft erscheint wieder das »falsche« Ergebnis – alle drei Register sind »0«. Hier soll uns jetzt der TW-Modus weiterhelfen, indem er uns zeigt, was in Wirklichkeit passiert.

Geben wir »TW 5000« ein. Der erste Befehl (LDA #00) ist durchgeführt, im Akku erscheint die Null. Jetzt steht der Programmzähler auf dem folgenden Befehl »5002 TAX«. Nach Drücken einer Taste wird dieser Befehl ausgeführt und es erscheint die Null im X-Register. Beim folgenden Befehl müssen wir feststellen, daß der Disassembler nicht in der Lage ist, ihn zu interpretieren – er gibt drei Sternchen aus. Hierbei handelt es sich um unser Byte »0C«.

Wieder ein Tastendruck; und dann erkennen wir, daß etwas Merkwürdiges passiert ist. Der Prozessor hat augenscheinlich den nächsten Befehl (LDA #04) übersprungen und steht schon auf dem folgenden »TAY«. So also wird unser Programm abgearbeitet. Damit ist auch das »falsche« Ergebnis erklärt. Bleibt nur noch die Frage nach dem Grund für dieses seltsame Verhalten. Und der ist sicherlich in dem mysteriösen Byte »OC« zu suchen. Hierbei handelt es sich um einen der »inoffiziellen« Opcodes, die aufgrund der Prozessorarchitektur vorhanden sind und in manchen Programmen ihr Unwesen treiben – wie wir zu unserem Leidwesen erfahren mußten. Das Byte »OC« wirkt wie ein »NOP«, der eine Länge von 3 Byte hat. Deshalb wird der folgende 2-Byte-Befehl (LDA #04) verschluckt.

Es gibt noch einiges zu entdecken am 6502 und 6510 – TW macht's möglich.

Häufig ist es nicht sinnvoll, ein Programm von Anfang an im TW-Modus laufenzulassen. Zum anderen sind gerade Schleifen, die per Hand mit »TW« durchlaufen werden müssen, eine ermüdende Angelegenheit. Hier bietet SMON neben dem bereits beschriebenen »TS« eine weitere Trace-Möglichkeit an:

◆ Trace Break

TB (Adresse Anzahl der Durchläufe)

◆ Trace Quick

TQ (Adresse)

Geben Sie als Beispiel folgendes Programm ein:

```
6000  LDY  #00    Y als Zähler auf »0«
6002  LDA  600E,Y  Werte von $600E ff. sollen geladen
                   werden
6005  JSR  FFD2   Ausgabe der Zeichen auf dem
                   Bildschirm
6008  INY         der Zähler wird erhöht
6009  CPY  #0E    Zähler schon »14«?
600B  BCC  6000   wenn nein, dann nächsten Wert holen
601D  BRK
```

Bei $600E soll nun ein Text stehen, den das Programm ausgibt. Die einfachste Art, mit SMON Texte in den Speicher zu schreiben, besteht im »K«-Befehl. Geben Sie

K 600E

ein (danach natürlich Return) und drücken Sie die STOP-Taste. Fahren Sie mit dem Cursor an das erste ausgegebene Zeichen (vermutlich ein Punkt) und schreiben Sie – ohne Anführungszeichen:

»FEHLER BEHOBEN«

Drücken Sie dann Return, um die Zeile an den Rechner zu übergeben. Wenn Sie das Programm starten, werden Sie wieder einmal Gelegenheit haben, sich in Ruhe etwas zu trinken zu holen (Prost!), denn das Programm enthält einen dummen Fehler und beschäftigt den Computer für eine lange, lange Zeit. Genauer gesagt, bis Sie ihn mit Reset (zum Beispiel durch RUN/STOP-RESTORE) erlösen.

Nun soll SMON helfen, diesen Fehler zu lokalisieren. Setzen Sie zuerst einmal einen Breakpoint bei $6002 und begrenzen die Durchläufe auf die maximale Anzahl:

TB 6002 0E

und starten mit

TQ 6000

den Quicktrace bei $6000. Das Programm läuft so lange, bis zum 14. Mal die Adresse $6002 erreicht wird und springt dann in den TW-Modus. Wenn Sie sich jetzt die Registerinhalte genau anschauen, müßte ihnen der Fehler geradezu ins Auge springen. Wie groß sollte denn das Y-Register sein? Welchen Wert sollte der Akku haben? NA?!

Das »Gedächtnis« von SMON

Wenn Sie Programme mit SMON untersuchen oder verändern wollen, müssen Sie noch wissen, welche Speicherstellen SMON verwendet. Es soll ja Monitorprogramme geben, die die Basic-Zeiger als Arbeitsspeicher benutzen, so daß ein Basic-Programm nach dem Rücksprung aus dem Monitor gelöscht ist. SMON tut so etwas nicht. Aber natürlich braucht er auch Speicherstellen, um sich Werte merken zu können. Damit Sie Konflikten von Anfang an aus dem Wege gehen können, sind die wichtigsten hier dargestellt.

In der Zeropage belegt SMON den Bereich von $00A4 bis $00B6. Dort stehen Systemvariable für die Kassettenspeicherung und die RS232-Schnittstelle. Diese werden nur während des Betriebs der Kassette oder von RS232 gebraucht, sind ansonsten aber frei. Außerdem werden die Speicherstellen $00FB bis $00FF benutzt, die sowieso zur freien Verfügung des Anwenders vorgesehen sind. Alle anderen Zeiger in der Zeropage, also insbesondere die Speicherverwaltung für Basic, bleiben unbeeinflußt.

Als weiteren Arbeitsspeicher benutzt SMON den Bereich von $02A8 bis $02C0. Auch dieser Bereich wird vom Betriebssystem nicht benutzt, so daß keine Konflikte entstehen dürften. Beim Assemblieren wird zusätzlich noch der Kassettenpuffer als Speicher für die Label benötigt. Dieser bleibt ansonsten aber auch unverändert; das ist wichtig, wenn Maschinenroutinen dort abgelegt werden sollen.

Alles in allem ist SMON also recht verträglich.

SMON verschieben? - Mit SMON!

Eine Reihe von Anfragen hat uns erreicht, ob man SMON nicht mit Hilfe des »W«-, »V«- oder »C«-Kommandos verschieben könne. Alle Versuche in dieser Richtung seien fehlgeschlagen. Einige Leser meinten auch, in der V-Routine müsse ein Fehler stecken. Diesmal sind wir jedoch völlig schuldlos; es gibt nämlich einige Befehle in SMON, die keine Sprungadressen sind und sich trotzdem auf den Bereich ($C000-) beziehen, in dem SMON steht.

Dazu gehören in erster Linie die oben erwähnten Einsprungadressen, deren High-Byte natürlich geändert werden muß, wenn SMON in einem anderen Speicherbereich laufen soll. Es gibt aber auch Befehle, die eine Adresse im Programm in einem Vektor ablegen müssen. Disassemblieren Sie einmal den Anfang von SMON mit »D C000 C00B«. Sie erhalten

```
LDA   #14    Low-Byte der BREAK-Routine von SMON
STA   0316   im Break-Vektor speichern
LDA   #C2    High-Byte (!) siehe oben
STA   0317   siehe oben
BRK
```

Damit wird der Break-Vektor des Betriebssystems auf den SMON gesetzt und mit dem anschließenden — und jedem weiteren BRK-Befehl — springt das Programm in SMONs BREAK-Routine. Wenn SMON in einem anderen Bereich als $C000 laufen soll, dann müssen diese Befehle geändert werden.

Heraussuchen kann man sie mit »FIC*,C000 D000«. Sie wissen doch noch, was diese Anweisung bedeutet: Suche mir alle Befehle, die ein Register unmittelbar mit einem Wert laden, der mit $C beginnt. Aber Vorsicht! Nicht alles, was da angezeigt wird, muß auch geändert werden! Um Ihnen weitere Stunden sinnlosen Herumbrütens zu ersparen, wollen wir als Beispiel zeigen, wie man SMON in den Bereich $9000 bis $A000 verlegen kann. Natürlich geht das im Prinzip für jeden anderen Bereich genauso; wir selbst haben insgesamt fünf SMON-Versionen für fünf verschiedene Speicherbereiche, von denen eine immer paßt.

1. Wir verschieben zuerst das ganze Programm ohne Umrechnen in den neuen Bereich:
W C000 CFFA 9000
2. Nun lassen wir alle absoluten (3-Byte-)Befehle umrechnen. Die Tabellen am Anfang von SMON bleiben verschont:
V C000 CFFA 9000 920B 9FD2
3. Als nächstes ändern wir die High-Bytes der Befehlsadresse. Geben Sie
»M 902B 906B«
ein und ändern Sie in jedem zweiten Byte das »C« durch Überschreiben in »9«. Vergessen Sie nicht, am Ende jeder Zeile »RETURN« zu drücken, damit Ihre Änderung auch übernommen wird.
4. Nun sind die Befehle mit Immediate-Adressierung an der Reihe. Sie müssen so geändert werden, daß sie sich auf den neuen Bereich $9... beziehen. Suchen Sie sie mit
FIC*,9000 9FFA
heraus. Sie erhalten

```
9005  LDA  #C2  ändern
9124  CPX  #C0  nicht ändern
9386  LDY  #C0  ändern
9441  CMP  #C0  nicht ändern
987F  LDX  #C3  nicht ändern
988D  LDX  #C1  nicht ändern
9992  LDA  #C1  nicht ändern
9C2C  LDA  #CC  ändern
9C5B  LDA  #C2  ändern
9CF4  LDA  #CC  ändern
9DA1  LDX  #CC  ändern
9E03  LDA  #CC  ändern
9E6C  CMP  #C0  nicht ändern
9F71  LDY  #CF  ändern
```

Sie sehen, es gibt keine Regel, welche Befehle zu ändern sind und welche nicht. Aus diesem Grunde müssen Sie diese Änderungen »von Hand« vornehmen.

5. Die Adressen im Diskmonitor müssen ebenfalls umgestellt werden. Dazu geben Sie bitte ein:
M 9FD8 9FE4
und ändern Sie jedes zweite Byte wie unter Punkt 3 beschrieben.

Vergessen Sie bitte auf keinen Fall, Ihre neue(n) Version(en) unter neuem Namen zu speichern. Sie lassen sich dann mit LOAD "Name",8,1 von Diskette laden und mit dem entsprechenden SYS (zum Beispiel 36864 bei SMON $9000) starten. Denken Sie auch daran, nach dem Laden und vor dem SYS ein NEW einzugeben, sonst beschwert sich der B-Befehl mit einem OUT OF MEMORY ERROR.

Probieren Sie nun alle Befehle durch. Sie müssen genauso arbeiten wie bisher. Vor allem können Sie jetzt auch Programme wie »DOS 5.1« oder »Turbo Tape« untersuchen, die im $C000-Bereich stehen. Achten Sie aber, wenn Sie »SMON $9000« von Basic aus benutzen, darauf, daß das Basic ihn nicht überschreibt. String-Variable werden nämlich von $A000 nach unten hin aufgebaut und bis $9E09 ist nicht viel Platz. Schützen Sie im Zweifelsfalle den Bereich, indem Sie nach dem Laden des SMON $9000 eingeben:

NEW : POKE 56,144 : POKE 55,0

Damit ist SMON vor Überschreiben geschützt. Das ist natürlich bei dem SMON $C000 nicht nötig, weil Basic in diesen Bereich nicht hineinkommt.

Die Befehle des Disk-Monitors

Da das Arbeiten mit dem Disk-Monitor besondere Aufmerksamkeit verlangt (nach Murphys Gesetzen führen Fehleingaben in der Regel zu unlesbaren Disketten), wird er mit einem eigenen Kommando eingeschaltet. Leider waren alle halbwegs sinnvollen Buchstaben (»D« wie Diskette oder »F« wie Floppy) schon vergeben, deshalb haben wir uns für ein schlichtes »Z« wie Zuversicht entschieden.

-Z schaltet den Disk-Monitor ein

Die Rahmenfarbe ändert sich auf Gelb, der gewohnte ».« am Anfang einer Zeile ändert sich in »*«. Dies alles hat den Zweck, Ihnen deutlich zu machen, daß es jetzt ernst wird. Intern wird jetzt das Basic abgeschaltet, weil der Disk-Monitor einen 256 Byte großen Puffer benötigt. Dieser liegt von $BF00 bis $C000 im RAM unter dem Basic, weil er dort am wenigsten stören kann.

READ: R (Track Sektor)

Liest einen Block von der Diskette in den Computer. Track und Sektor müssen als Hexzahlen eingegeben werden. Die erste Zeile des Blocks wird ausgegeben. Da wir dazu normale SMON-Routinen verwenden, steht als Speicheradresse $BF00. Das »BF« können Sie vorerst ignorieren. Die weitere Ausgabe des Hexdump erfolgt anders als gewohnt mit der Taste »SHIFT«. STOP bricht die Ausgabe ab. Sie können die Hex-Bytes überschreiben und damit ändern. Eine dauerhafte

Änderung erfolgt aber erst beim Zurückschreiben auf die Diskette (siehe Befehl »W«). Geben Sie nur »R« ohne Track und Sektor ein, wird der logisch (!) nächste Block eingelesen.

MEMORY-DUMP: M

Zeigt den gerade im Puffer befindlichen Block nochmals auf dem Bildschirm an.

Genau wie beim R-Befehl können Sie die Ausgabe mit »SHIFT« und »STOP« steuern und Änderungen vornehmen.

WRITE: W (Track Sektor)

Schreibt einen Block aus dem Puffer auf die Diskette zurück. Ähnlich wie bei »R« kann die Angabe von Track und Sektor entfallen. Es wird dann der Track und Sektor des letzten R-Befehls benutzt. Das ist in fast allen Fällen auch der richtige.

ERROR: @

Liest den Fehlerkanal aus, gibt ihn aber nur aus, wenn wirklich ein Fehler vorhanden war. (»00, OK, 00, 00« wird unterdrückt.)

EXIT: X

Verläßt den Disk-Monitor und springt in den SMON zurück. Dabei wird die Rahmenfarbe auf Blau zurückgeschaltet und es erscheint wieder der ».« am Anfang der Zeile. Das Basic wird wieder eingeschaltet. Wollen Sie nun mit SMON-Kommandos auf den Puffer zugreifen, müssen Sie Basic wieder abschalten ($36 in Speicherstelle $0001).

Die folgenden Beispiele sollen Ihnen die Arbeit mit dem Disk-Monitor verdeutlichen.

Achtung! Benutzen Sie unbedingt zum Üben eine Diskette, die Sie nicht mehr brauchen!

Weder wir noch der Verlag haften dafür, wenn Ihr Lieblingsprogramm oder die mühsam erstellte Adreßdatei unwiederbringlich dahin sind. Daß das sehr sehr schnell gehen kann, wissen wir aus eigener Erfahrung ...

Am besten machen Sie von einer Ihrer Diskette eine Kopie, die Sie zum Üben benutzen können.

Reparatur eines gelöschten Files

Sicher ist Ihnen das auch schon passiert: Sie wollen Ihr Programm mit Namen »Schrott« löschen, geben als Abkürzung »S:S*« ein und merken in dem Moment, in dem Sie »RETURN« drücken, daß auf der Diskette auch alle Versionen von »SMON« waren, außerdem auch noch »Springvogel«, »Soccer« etc. Verzweifeln müssen Sie nur, wenn auch diese letzte SMON-Version mit dem Disk-Monitor dabei war. Ansonsten behalten Sie die Ruhe und verfahren Sie wie im folgenden beschrieben.

Laden Sie also jetzt SMON, legen Sie Ihre »Übungsdiskette« (!) ins Laufwerk und löschen Sie eins der ersten Programme mit dem üblichen Scratch-Kommando. Nun starten Sie SMON und drücken »Z«. Der Bildschirm ändert seine Farbe wie beschrieben und am Anfang der Zeile erscheint der »*«. Jetzt geben Sie ein:

R 12 00

Auf dem Bildschirm erscheint die erste Zeile der BAM, die bei jeder Diskette auf Track 18, Sektor 0 abgelegt ist. Die ersten beiden Bytes enthalten »12 01« und geben damit den logisch nächsten Block an. In diesem Falle wäre das der erste Block des Directory. Wenn Sie mit »SHIFT« die Bildschirmausgabe fortsetzen, erkennen Sie etwa in der Mitte den Diskettennamen. Lassen Sie die Ausgabe durchlaufen, bis wieder der »*« erscheint. Nun geben Sie »R« ohne weitere Angaben ein. Damit erhalten Sie den Koppel-Block, also Track 18, Sektor 1, den ersten Directory-Block. (Natürlich hätten Sie auch gleich »R 12 01« eintippen können, aber wir wollen ja zeigen, wie die Befehle funktionieren.)

In diesem Block stehen die ersten acht Programme Ihrer Übungsdiskette, auch der Name des gelöschten ist dabei.

Trotzdem ist dieses Programm tatsächlich gelöscht und erscheint nicht mehr, wenn Sie sich das Directory anzeigen lassen. Vergleichen Sie den Eintrag des gelöschten Programms mit den anderen, fällt auf, daß 3 Byte vor Beginn des Namens bei allen anderen »82« steht (sofern es sich um Programmfiles handelt), bei dem gelöschten aber »00«. Die Reparatur ist nun denkbar einfach: Sie brauchen lediglich die »00« mit »82« zu überschreiben. Einen Haken hat die Sache allerdings noch. Beim SCRATCHEN sind die vom Programm belegten Blöcke in der BAM als frei gekennzeichnet worden und jeder neue Eintrag würde das als gelöscht gekennzeichnete File endgültig überschreiben. Um das zu verhindern, müssen Sie nach erfolgter Reparatur die Diskette validieren (von Basic aus mit Kommando: OPEN 1, 8, 15, "V"). Dabei wird die BAM neu erzeugt und korrigiert.

Schützen eines Files

Da wir gerade dabei sind, wollen wir unser repariertes gelöschtes File gleich ein für allemal gegen Löschen schützen. Diese Möglichkeit des Diskettenoperationssystems (DOS) ist zwar nicht im Handbuch beschrieben, funktioniert aber trotzdem ausgezeichnet. Laden Sie dazu nochmals die erste Seite des Directory mit

R 12 01

und ändern Sie die »82« vor dem Fileeintrag in »C2«. Geben Sie »W« ein, um die Änderung auf Diskette zu schreiben. Verlassen Sie nun SMON mit »X« und lassen Sie sich ein Directory anzeigen. Das geschützte File ist mit einem »>« gekennzeichnet. Versuchen Sie nun, dieses Programm mit dem Scratch-Kommando zu löschen. Es geht nicht! Zum »Entriegeln« brauchen Sie nur das »C2« wieder in »82« zu ändern. Der »>« im Directory verschwindet und das File ist nicht mehr geschützt.

Schützen einer Diskette

Wollen Sie eine ganze Diskette vor versehentlichem Löschen oder Formatieren schützen, gibt es die Möglichkeit, die Löschschutzkerbe abzukleben. Es geht jedoch auch anders.

Achtung! Die im folgenden beschriebene Prozedur läßt sich nicht ohne weiteres rückgängig machen, auch nicht mit dem Disk-Monitor!

Nehmen Sie also eine Diskette, die Sie anschließend »hart formatieren« können (also mit Eingabe einer ID). Starten Sie nun den Disk-Monitor und lesen Sie die BAM mit »R 12 00« ein. Das dritte Byte enthält »41«. Diese »41« ist ein Kennzeichen für das DOS der 1541- oder 4040-Floppy. Ändern Sie diese Byte durch Überschreiben in »45« und speichern Sie die Änderung mit »W« auf die Diskette zurück. Verlassen Sie nun SMON und versuchen Sie, etwas zu löschen. Ergebnis siehe oben. Versuchen Sie auch, die Diskette »weich«, also zum Beispiel mit OPEN 1,8,15,"N:TEST" zu formatieren.

Auch das ist jetzt nicht mehr möglich. Aber es kommt noch besser: Starten Sie noch einmal den Disk-Monitor und versuchen Sie, die Änderung durch Zurückschreiben der »41« an Stelle der »45« rückgängig zu machen. Auch das ist nicht mehr möglich, wir hatten Sie bereits gewarnt! Es bleibt lediglich die Möglichkeit, die Diskette »hart«, zum Beispiel mit OPEN 1,8,15,"N:TEST,TE" zu formatieren. Sollten Sie nun entgegen allen Warnungen doch Ihre Master-Diskette gegen Schreibzugriffe gesichert haben, verraten wir Ihnen ausnahmsweise, wie Sie den Eingriff trotzdem rückgängig machen können. Dazu überlisten wir das DOS des 1541-Laufwerkes, indem wir ihm vorgaukeln, es hätte eine Diskette im Normalformat vor sich. Wir verwenden den Memory-Write-Befehl, mit dem wir in die Speicherstelle 0101 (Zero-Page Adresse) des 1541-RAM einfach ein »A« schreiben. Der CHR$-Code des »A« ist 65, oder in hexadezimaler Schreibweise 41. Erinnern Sie sich? Dieser Wert stand ursprünglich im dritten Byte des Tracks 18, Sektor 0. Mit folgendem kleinen Programm umgehen wir einfach die DOS-Kennzeichnung und wir können die Diskette wieder normal beschreiben. Am sinnvollsten ist es, sofort den SMON zu starten, das

vorher in 45 abgeänderte Byte wieder in 41 zu verwandeln und abzuspeichern. Die Diskette kann dann wieder zum Lesen und Schreiben verwendet werden. Hier nun das kleine Programm:

```
10 OPEN 1,8,15
20 PRINT#1, "M-W"CHR$(1)CHR$(1)CHR$(65)
30 CLOSE1
```

Ändern des Diskettennamens oder der ID

Wir haben bereits oben gesehen, daß in Spur 18, Sektor 0 einer Diskette etwa in der Mitte der Diskettenname gespeichert wird. Dieser Name kann durch einfaches Überschreiben geändert werden; er darf bekanntlich bis zu 16 Zeichen enthalten. Hat Ihr neuer Name weniger Buchstaben als der alte, müssen Sie die Lücken mit »A0« und nicht mit »20« als Leerzeichen ausfüllen. Dies gilt vor allem, wenn Sie mit dieser Methode Filenamen ändern wollen. Das geht natürlich im Prinzip genauso wie eben beschrieben. Hinter dem Diskettennamen ist in Spur 18, Sektor 0 die ID abgelegt. Sie wird beim Formatieren vor jeden Sektor in einen sogenannten Header geschrieben und dient dem DOS zur Identifikation der Diskette. Zusätzlich wird sie noch in der BAM gespeichert, damit sie beim Laden eines Directory mit angezeigt werden kann. Nun ist es grundsätzlich nicht möglich, die ID im Header eines Sektors ohne Formatieren zu ändern, wohl aber die Eintragung in der BAM und damit die ID, die im Directory angezeigt wird. Genau wie beim Namen ist dies durch einfaches Überschreiben in der BAM möglich.

Ändern eines Filetyps

Wenn Sie einmal versucht haben, ein sequentielles File, etwa eine Datei, mit LOAD zu laden, werden Sie gemerkt haben, daß dies nicht möglich ist. Das DOS behauptet einfach, ein solches File existiere nicht und der Computer meldet »FILE NOT FOUND«. Viele Spiele zum Beispiel legen die »Hall of Fame« oder Highscore-Liste als sequentielle Datei ab. Mit dem Disk-Monitor ist es nun aber möglich, den Filetyp im Directory zu verändern. Erinnern Sie sich an die »82«, die im Directory vor jedem Filenamen steht. Bei sequentiellen Files steht dort »81«. Was zu tun ist, werden Sie sich denken können. Na klar, die »81« wird in »82« geändert, und schon ist die Datei ohne weiteres ladbar, natürlich wieder erst nach dem Zurückschreiben mit »W«.

Sinnvoll ist dies natürlich nur von SMON aus (mit Eingabe einer Ladeadresse). Mit »M« oder »K« können Sie dann die Datei ansehen und natürlich auch ändern. Vergessen Sie nicht, die geänderte Datei nach dem Zurückschreiben wieder in ein sequentielles File zu verwandeln. Verblüffen Sie Ihre Freunde doch mal mit einem auf diese Weise »errungenen« High-Score. Die Anerkennung Ihrer Umwelt ist Ihnen sicher!

Ändern der Startadresse eines Programms

Wir haben uns bisher auf Manipulationen in der BAM oder im Directory beschränkt. Wollen wir in einem Programm selbst Änderungen vornehmen, müssen wir etwas tiefer in die »Geheimnisse der Floppy« eindringen. So ist es bisweilen interessant, die Startadresse eines Maschinenprogramms zu kennen oder zu ändern. Dazu gehen wir folgendermaßen vor: Zunächst suchen wir mit »R 12 01« und eventuell weiteren Folgesektoren (12 04, 12 07...) den Fileeintrag im Directory. Die beiden Byte hinter der »82« direkt vor dem Programmnamen geben an, auf welcher Spur und in welchem Sektor das Programm startet. Wenn dort zum Beispiel »0A 04« steht, beginnt das Programm auf Spur 10, Sektor 4. Lesen Sie nun diesen Block mit »R 0A 04« ein. Die ersten beiden Bytes dieses Blocks zeigen auf den nächsten Block des Programms, die beiden nächsten Bytes enthalten die Startadresse in der üblichen Low-High-Byte-Reihenfolge. Zum Ändern der Startadresse überschreiben Sie die Bytes mit der neuen und speichern den Block mit »W« auf die Diskette zurück.

Die Zusammenarbeit mit SMON

Mit all diesen Beispielen sind die Möglichkeiten des Disk-Monitors noch lange nicht erschöpft. Sie sollten Ihnen als Anregung für eigene Experimente dienen. Üben Sie aber unbedingt so lange, bis Sie alle Kommandos aus dem »FF« (oder dezimal 255) beherrschen. Sie ersparen sich damit unnötigen Ärger und durchweinte Nächte. Besonders interessant ist es, von SMON aus auf den Puffer zuzugreifen und die SMON-Befehle auf den Puffer anzuwenden. Erwähnen möchte ich nur die Möglichkeit, Programme für das DOS direkt zu assemblieren und in einem bestimmten Sektor ablegen zu können, die »Find«-Routinen oder das »K«-Kommando für Textänderungen. Da der Puffer im RAM unter dem Basic liegt, muß Basic in solchen Fällen abgeschaltet werden. Ändern Sie dazu mit dem »M«-Befehl in Speicherstelle 0001 die »37« in »36«.

Haben Sie die Arbeit mit SMON beendet, können Sie mit »Z« in den Disk-Monitor schalten und den Pufferbereich mit »W« (Spur, Sektor) abspeichern.

Die Ausgabe von Diskettenfehlern

Beim Arbeiten mit dem Disk-Monitor werden sämtliche Fehler vom Laufwerk direkt, auch ohne Eingabe von »@«, ausgegeben, zum Beispiel »ILLEGAL TRACK OR SECTOR«, wenn Sie mit »R« einen Block lesen wollen, den es gar nicht gibt. Einen Fehler hat das Programm allerdings, den wir nicht verschweigen wollen. Der letzte Block eines Files enthält als Koppeladresse »00 FF«. Da es einen solchen Block nicht geben kann, »weiß« das DOS, daß es am Ende angelangt ist. Versuchen Sie aber, den nächsten Block (Spur 0, Sektor 255!!) mit »R« zu lesen, erscheint als Fehlermeldung nicht, wie es sein müßte, »ILLEGAL BLOCK OR SECTOR«, sondern »SYNTAX ERROR«. Das ist zwar eigentlich unerheblich, sollte aber erwähnt werden. Der Fehler liegt in der Routine, die unsere Zahleneingaben in das richtige Diskettenformat wandelt. Es fehlte einfach der Platz im Programm für eine »korrekte« Umwandlung, wir mußten uns mit einer »Sparroutine« behelfen.

Abschließend noch ein SMON-Trick, den wir einem aufmerksamen Leser verdanken. Für eine Directory-Ausgabe fehlte der Platz im SMON. Es geht aber hilfsweise so: Laden Sie das Directory zum Beispiel mit

```
L "$" 8000
```

an einen freien Speicherplatz. Mit »M« oder »K« können Sie jetzt das Directory »lesen«. Damit sind alle wichtigen Funktionen für den Umgang mit der Diskette im SMON enthalten.

SMON lüftet Geheimnisse

Zwei Erweiterungen haben wir Ihnen zu Beginn angekündigt, die SMON noch leistungsfähiger machen sollen. Dabei handelt es sich einmal um eine Erweiterung des Disassemblers, mit dem nun auch die »illegalen« Opcodes des 6502 disassembliert werden, zum anderen um neue Funktionen beim Diskmonitor, mit denen Sie in den Innereien Ihrer Floppy herumstöbern können. Nun ist der Speicherplatz bis auf 5 Byte ausgeschöpft, und die 4-KByte-Grenze soll auf keinen Fall überschritten werden. Wir haben daher andere Funktionen herausgenommen, und zwar für die Disassembler-Erweiterung den Diskmonitor und für die Diskmonitor-Erweiterung den Trace-Modus. Beide Erweiterungen sind also nicht gleichzeitig einsetzbar; überhaupt ist es sinnvoll, eigene Versionen für spezielle Anwendungen zusammenzustellen, eine »normale«, eine Spezial-Disk-Version und eine für verschärftes Disassemblieren.

Beginnen wir mit dem letzten: Wie Sie wissen, erscheinen beim Disassemblieren immer drei Sternchen, wenn SMON auf ein Byte trifft, das keinen gültigen 6510-Opcode darstellt. Nun wissen Sie aber vielleicht auch, daß es über den offiziel-

len Befehlssatz hinaus noch einige Befehle gibt, die der Hersteller des Prozessors zwar nicht dokumentiert hat, die aber nichtsdestotrotz funktionieren und in einigen Programmen auch ausgenutzt werden. Es wäre natürlich schön, wenn SMON auch diese »illegalen« Opcodes anzeigen könnte. Unsere Erweiterung macht's möglich.

Wir haben Mnemonics für eine Reihe dieser Befehle eingesetzt und lassen diese von SMON mit einem vorangestellten »*« ausgeben. Übrig bleiben noch zehn Befehle, deren Wirkung aber so komplex ist, daß sie sich beim besten Willen nicht mit einem Mnemonic abkürzen lassen. Sie fallen auch aus der Logik der Prozessorstruktur heraus. Im einzelnen handelt es sich um die Opcodes 0B, 2B, 4B, 6B, 8B, 9C, 9E, AB, CB und EB. Bei diesen Befehlen haben wir keine gemeinsame Struktur entdecken können. Die neuen Mnemonics haben folgende Bedeutung:

LAX	Load Akku and X entspricht LDA und LDX.
DCP	Decrement and ComPare entspricht DEC und CMP.
ISC	Increment and SubtraCt entspricht INC und SBC.
RLA	Rotate Left AND Akku entspricht ROL und AND.
RRA	Rotate Right and Add with carry entspricht ROR und ADC.
SLO	Shift Left OR Akku entspricht ASL und ORA.
SRE	Shift Right and EOR Akku entspricht LSR und EOR.
SAX	Store Akku AND X führt eine UND-Verknüpfung zwischen Akku und X-Register durch und speichert das Ergebnis in der angegebenen Adresse ab.
CRA	CRAsh führt zum »Absturz« des Prozessors.
NOP	NO Operation entspricht dem bekannten NOP, jedoch kann dieser Befehl auch 2 oder 3 Byte lang sein. Dies wird durch die angegebene Adresse deutlich, die in diesem Fall natürlich keinerlei Bedeutung hat.

Über den Sinn dieser Befehle läßt sich sicher streiten; allerdings kommen sie bisweilen in Programmen vor, meist um das Lesen dieser Programme unmöglich zu machen, also als Programmschutz. Von der Verwendung dieser Befehle in eigenen Programmen raten wir auf jeden Fall ab. Erstens wird kein Hersteller garantieren, daß die »illegalen« tatsächlich mit jedem 6510-Prozessor funktionieren, zweitens gibt es keine Funktion, die nicht auch mit den »normalen« Befehlen ebensogut erreicht werden könnte. Und als Programmschutz taugen die »illegalen« spätestens mit der Veröffentlichung dieses Artikels ja auch nichts mehr. Aus diesem Grund haben wir bewußt auf eine Erweiterung des Assemblers in dieser Richtung verzichtet. Sie können also keine normalen Opcodes durch Überschreiben in »illegale« ändern, wohl aber umgekehrt. Es bleibt lediglich die Eingabe als Einzelbyte, was aber hoffentlich zu umständlich ist.

Komfortabler Disketten-Monitor für SMON

Jetzt folgt unser zweiter Leckerbissen in Form eines kleinen aber ungemein wertvollen Zusatzprogrammes für den SMON. Es handelt sich dabei um eine Erweiterung des Disketten-Monitors, mit dem jeder auf einen Schlag die Arbeit von Stunden zunichte machen kann. Geben Sie das Programm wie beschrieben ein, starten Sie SMON wie gewohnt und springen mit »Z« in den Disketten-Monitor. Von hier aus erreichen Sie mit »F« (wie Floppy) die neuen Befehle. Wir haben absichtlich diesen umständlichen Weg gewählt, denn Fehler in diesem Modus wirken noch dramatischer als sonst. Mit diesem Werkzeug haben Sie unmittelbaren Zugriff auf die Eingeweide der Floppy. Jetzt können Sie die folgenden Befehle mit einer Übungsdiskette (!!!) in aller Ruhe durcharbeiten.

M Memory-Dump des Disketten-Monitors

Beispiel: M (ohne weitere Eingabe) listet den Bereich des Floppy-RAM von $0000-$00FF. (Es erscheint zunächst die erste Zeile, weitere Ausgabe mit der SPACE-Taste.)

In diesem Bereich befinden sich unter anderem die Jobspeicher ($00-$04) für die fünf Puffer 0 bis 4 sowie die wichtigsten Variablen des DOS.

M 07 Memory-Dump ab $0700

Die BAM der Diskette wird nach dem Initialisieren in Puffer 4 ($0700 im Floppy-RAM) eingelesen. Schauen Sie sich also mit »M 07« die aktuelle BAM an. Sie könnten jetzt durch einfaches Überschreiben den Inhalt der BAM ändern. (Der Doppelpunkt vor der Zeile wirkt als »hidden command«). Dann schauen Sie sich Ihre Änderung mit »M 07« wieder an. Sie sehen, daß inzwischen der Inhalt des Floppy-RAM geändert wurde. Wenn Sie nun den Jobcode »90« (= Schreibbefehl an den Floppy-Controller) in Speicherstelle $04 bringen, würde die geänderte (falsche!) BAM auf Diskette zurückgeschrieben werden!! Es gibt also genug Möglichkeiten, wie oben angedeutet, die Disketten zu »versauen«.

Für das Ausprobieren noch einige wichtige Speicherstellen und Jobcodes:

$80	Lesen
$90	Schreiben
$C0	»Anschlagen« des Kopfes
$D0	Maschinenprogramme im Puffer ausführen
$E0	Programm im Puffer ausführen mit Hochfahren des Laufwerks

Speicherstellen im Floppy-RAM:

$06/$07	ist Spur- und Sektornummer für den Befehl in Puffer 0
$08/$09	für Puffer 1
$0A/$0B	für Puffer 2
$0C/$0D	für Puffer 3
$0E/$0F	für Puffer 4

Jedem Puffer sind zwei Speicherstellen zugeordnet, eine für den Jobcode ($0000 bis $0004) und eine für Spur und Sektor. Wenn Sie also in Puffer 0 (in $0300 gelegen) einen bestimmten Block einlesen wollen, geben Sie folgende Befehle ein:

»M« liest die Zeropage der Floppy ein — so sehen dann zum Beispiel die ersten Zeilen aus:

```
:0000 01 01 01 FF 03 04 01 34
:0008 23 02 04 50 01 03 0A 11
```

Gehen Sie mit dem Cursor in die erste Zeile und schreiben Sie »80« in die erste Speicherstelle (anstelle der ersten 01). In Speicherstelle $06/$07 (die letzten beiden in der ersten Reihe) die Spur- und die Sektornummer, die gelesen werden soll, zum Beispiel 12 01. Sie sehen dann

```
:0000 80 01 01 FF 03 04 12 01
:0008 unverändert
```

Drücken Sie die RETURN-Taste, Mit »M 03« kann jetzt der eingelesene Block (hier der erste Directory-Block) angesehen werden. Änderungen können durch einfaches Überschreiben vorgenommen werden. Dauerhaft wird Ihre Änderung erst durch Zurückschreiben (nach Spur $12 und Sektor $01) mit dem Jobcode »90« in der ersten Speicherstelle. Nach Änderung der beiden für Puffer 0 zuständigen Adressen ($06/$07) auch an jede beliebige andere Stelle. Das ist wörtlich zu nehmen, denn wir befinden uns hier »unterhalb«

Befehlsübersicht zum SMON

Alle Eingaben erfolgen in der hexadezimalen Schreibweise. In Klammern angegebene Adreßeingaben können entfallen. SMON benutzt dann sinnvolle, vorgegebene Werte.

Bei allen Ausgabe-Befehlen ist gleichzeitig die Ausgabe auf einem Drucker möglich. Dazu werden die Befehle geSHIFTet eingegeben.

A **4000** (Assembler)
symbolischer Assembler (Verarbeitung von Label möglich) Startadresse $4000

B **4000 4200** (Basic-Data)
erzeugt Basic-DATA-Zeilen aus Maschinenprogramm im Bereich von $4000 bis $41FF

C **4010 4200 4013 4000 4200** (Convert)
in ein Programm, das von $4000 bis $4200 im Speicher steht, soll bei 4010 ein 3-Byte-Befehl eingefügt werden. Dazu wird das Programm ab $4010 bis 4200 auf die neue Adresse $4013 verschoben. Alle absoluten Adressen, die innerhalb des Programmbereichs ($4000 bis $4200) stehen, werden umgerechnet, so daß die Sprungziele stimmen.

D **4000 (4100)** (Disassembler)
disassembliert den Bereich von $4000 (bis $4100) mit Ausgabe der Hex-Werte. Änderungen sind durch Überschreiben der Befehle möglich.

F (Find)
findet Zeichenketten (F), absolute Adressen (FA), relative Sprünge (FR), Tabellen (FT), Zeropageadressen (FZ) und Immediate-Befehle (FI)

G **(4000)** (Go)
startet ein Maschinenprogramm, das bei $4000 im Speicher beginnt

I **01** (I/O-Gerät)
stellt die Gerätenummer für Floppy (08 oder 09) oder Datasette (01) ein

K **A000 (A500)** (Kontrolle)
zum schnellen Durchsuchen des Bereichs von $A000 (bis $A500) nach ASCII-Zeichen (32 Byte pro Zeile). Änderungen sind durch Überschreiben der ASCII-Zeichen möglich.

L **(4000)** (Load)
lädt ein Maschinenprogramm an die richtige oder eine angegebene Adresse ($4000)

M **4000 (4400)** (Memory-Dump)
gibt den Inhalt des Speichers von $4000 (bis $43FF) in Hex-Byte und ASCII-Code aus. Änderungen sind durch Überschreiben der Hex-Zahlen möglich.

O **4000 4500 AA** (Occupy)
füllt den Speicherbereich von $4000 bis $4500 mit vorgegebenem Byte ($AA) aus

P **05** (Printer)
setzt Geräteadresse 5 für Drucker

R (Register)
zeigt die Registerinhalte und Flags an. Änderungen sind durch Überschreiben möglich.

S **"Test" 4000 5000** (Save)
speichert ein Programm von $4000 bis $4FFF unter dem Namen »Test« ab

TW **(4000)** (Trace Walk)
führt auf Tastendruck den jeweils nächsten Maschinenbefehl aus und zeigt die Registerinhalte an. Subroutinen können in Echtzeit durchlaufen werden (»J«). Wird keine Startadresse eingegeben, beginnt »TW« bei der letzten mit »R« angezeigten Adresse.

TB **4010 05** (Trace Break)
setzt einen Haltepunkt für den Schnellschrittmodus bei $4010. Der Schnellschrittmodus wird unterbrochen, nachdem $4010 zum fünften Mal erreicht worden ist.

TQ **4000** (Trace quick)
Schnellschrittmodus, springt beim Erreichen eines Haltepunktes in den Einzelschrittmodus

TS **4000 4020** (Trace stop)
arbeitet ein Programm ab $4000 in Echtzeit ab und springt beim Erreichen von $4020 in die Registeranzeige. Von dort aus kann (nach eventueller Änderung der Register) mit »G« oder »TW« fortgefahren werden. »TS« arbeitet nur im RAM-Speicher.

V **6000 6200 4000 4100 4200** (Verschieben)
ändert in einem Programm von $4100 bis $41FF alle absoluten Adressen, die sich auf den Bereich von $6000 bis $6200 beziehen, auf einen neuen Bereich, der bei $4000 beginnt.

W **4000 4300 5000** (Write)
verschiebt den Speicherinhalt von $4000 bis $42FF nach $5000 ohne Umrechnung der Adressen (zum Beispiel Tabellen)

X (Exit)
springt aus dem Monitor-Programm ins Basic zurück

**49152**
Dezimalzahl umrechnen

$ **002B**
4stellige Hex-Zahl umrechnen

% **01101010**
8stellige Binärzahl umrechnen

? **0344 + 5234**
Addition oder Subtraktion zweier 4stelliger Hex-Zahlen

= **4000 5000** (Vergleich)
vergleicht den Speicherinhalt ab $4000 mit dem ab $5000

Z (Diskmonitor)
ruft den Diskmonitor auf. Dieser verfügt über folgende Befehle:

R **(12 01)** (Read)
liest Track $12, Sektor $01 von der Diskette in einen Puffer im Speicher. Fehlt die Angabe von Track und Sektor, wird der logisch (!) nächste Sektor gelesen.

W **(12 01)** (Write)
schreibt den Puffer im Speicher nach Track $12, Sektor $01 auf die Diskette. Ohne Angabe von Track und Sektor werden die letzten Eingaben von »R« benutzt.

M (Memory-Dump)
zeigt den Pufferinhalt als Hexdump (wie normales »M«). Weitere Ausgabe mit CBM-Taste, Abbruch mit STOP. Werte können durch Überschreiben geändert werden.

X (Exit)
springt in SMON zurück

F (weitere Disketten-Befehle initialisieren)
sind die Befehle initialisiert, gilt:
M (07)
Memory-Dump (Floppy-RAM/ROM)
V 6000 0400
Verschieben eines 256-Byte-Blocks von $6000 in den Laufwerkspuffer 1 beziehungsweise in das Floppy-RAM
@
normale Disketten-Befehle senden
X
zurück zum normalen Disketten-Monitor

der Controllerebene, die unter anderem für die Prüfung auf Einhaltung der zulässigen Spur und Sektorgrenzen verantwortlich ist. Es erfolgt also keine Fehlermeldung, wenn Sie versuchen sollten, mit Ihrer Floppy bis in die des Nachbarn zu schreiben (zum Beispiel mit der Spur 152).

Entsprechende Lese- und Schreibübungen können mit den anderen Puffern durchgeführt werden. Denken Sie daran, erst ist die Spur- beziehungsweise Sektornummer für den entsprechenden Puffer (in der zweiten Zeile!) einzugeben, bevor Sie in Zeile 1 den Jobcode mit einem »RETURN« übergeben, denn mit Druck auf die RETURN-Taste wird Ihr Befehl ausgeführt. Und noch eins: Quälen Sie bitte dabei Ihren Schreibkopf nicht mehr als unbedingt erforderlich, sonst könnte er sich mechanisch verklemmen und nur noch mit einem Eingriff in die Floppymechanik wieder »befreit« werden.

Falls Sie die Ausgaben 1/85 (Seite 151) und 3/85 (Seite 103 bis 135) der 64'er besitzen, können Sie sich dort über andere Speicherstellen der Floppy und die weitere Anwendung der Jobcodes informieren.

Der Befehl @ ohne weitere Angaben fragt den Fehlerkanal ab, ansonsten dient er zur Befehlsübermittlung an die Floppy.

Beispiel:		
	@	Fehlerkanal
	@I	Initialisierungsbefehl oder
	@S:name	Befehl zum Scratchen und so weiter.

Bedingt durch die verschiedenen Versionen, springt dieser Befehl manchmal in den »normalen« Disketten-Monitor zurück, erkennbar an dem »*« am Zeilenanfang. Sie müssen dann wieder ein »F« eingeben.

Mit X gelangt man wieder in den Disketten-Monitor.

Zum Abschluß ein sehr hilfreicher Befehl namens »V«, der es erlaubt, Speicherbereiche aus dem Computer in den Laufwerkspuffer zu verschieben. Folgende einfache Syntax gilt dabei: **V von nach**

Um zum Beispiel ein Maschinenprogramm von $6000 in den Puffer 1 zu bekommen, geben Sie folgendes ein:

V 6000 0400

Dabei wird immer eine ganze Seite, also 256 Byte, übertragen. Was das Programm dort soll, fragen Sie? Führen Sie es doch einfach aus (Jobcode $D0 in Speicherstelle $01 schreiben); oder schreiben Sie es mit dem Jobcode »90« in einen beliebigen Sektor der Diskette.

Wenn Sie dann Ihre Floppy so richtig durcheinander gebracht haben und nichts läuft mehr, brauchen Sie nicht zu verzweifeln. Außer einem eventuell festhängenden Lesekopf passiert der Floppy nichts, nur Ihren Disketten.

Hinweise zum Abtippen

Tippen Sie die beiden Erweiterungsprogramme (Listing 2 und 3 beziehungsweise bei der M&T-Version Listing 4 und 5) mit dem MSE-Programm ab und speichern Sie die fertigen Programme. Die Programme für die M&T-Version haben sinnigerweise ein M&T im Namen.

Laden und starten Sie dann Ihren SMON $C000. Geben Sie ein: L"NDISASS"

Damit werden die neuen Befehle automatisch über den bisherigen Disketten-Monitor geladen. Sie müssen nun aber noch aktiviert werden. Geben Sie dazu G CF0D ein.

SMON meldet sich sofort mit seiner Registeranzeige wieder. Sie sollten nun diese Version unbedingt abspeichern, zum Beispiel mit S"SMON NDISASS" C000 CF3D

Wenn Sie nun das Programm »ILLEGAL-CODE« (Listing 6) laden und mit D 4000 disassemblieren, sehen Sie die »illegalen« Opcodes schön geordnet nacheinander.

Um die neuen Befehle des Disketten-Monitors in SMON einzubinden, gehen Sie ganz ähnlich vor. Nach dem Abtippen und Speichern des Programms »FLOPPYMON« muß natürlich SMON C000 geladen und gestartet werden. Anschließend geben Sie ein: L"FLOPPYMON" und aktivieren es mit G CDD8 (64'er-Version) beziehungsweise G CDB6 (M&T-Version).

Zum Speichern geben Sie S"SMON-FLOPPY"C000 CFFF ein. Das gilt für beide Versionen.

(Dietrich Weineck/ah)

SMON-Speicherstellen

Folgende Zeropage-Adressen werden benutzt:

FLAG	$AA	Universalflag
ADRCODE	$AB	Adressierungscode für Assembler/Disassembler
COMMAND	$AC	SMON-Befehlscode
BEFCODE	$AD	Befehlscode Ass./Disass.
LOPER	$AE	Low-Operand für Ass./Disass.
HOPER	$AF	High-Operand für Ass./Disass.
BEFLEN	$B6	Befehlslänge Ass./Disass.
PCL	$FB	SMON-Programmcounter Low-Byte
PCH	$FC	SMON-Programmcounter High-Byte

Außerhalb der Zeropage benutzt SMON die Bereiche:

PCHSAVE	$02A8	
PCLSAVE	$02A9	
SRSAVE	$02AA	
AKSAVE	$02AB	dienen der Zwischenspeicherung
XRSAVE	$02AC	der angegebenen Register
YRSAVE	$02AD	
SPSAVE	$02AE	
PRINTER	$02AF	Printernummer
IO.NR	$02B0	Devicenummer
MEM	$02B1	Buffer bis $02B8
TRACEBUF	$02B8 bis $02BF	Buffer für Trace-Modus

Dann folgen die von Diskmonitor benötigten Adressen:

SAVEX	$02C1	Zwischenspeicherung der X- und Y-Register
TMPTRCK	$02C2	
TMPSECTO	$02C3	Zwischenspeicher für Track und Sektor
DCMDST	$02D0	Diskkommandostring
TRACK	$02D8	
SECTO	$02DB	Track und Sektornummer
BUFFER	$033C bis $03FC	Buffer für Label, nur für Assembler

Einsprungadressen von SMON-Routinen

Die Angaben in Klammern beziehen sich auf die M&T-Version

;	(TICK)	$CADB	($CACF)
#	(BEFDEC)	$C92E	($C92F)
$	(BEFHEX)	$C908	($C909)
%	(BEFBIN)	$C91C	($C910)
,	(KOMMA)	$C6FC	($C6B1)
:	(COLON)	$C41D	($C40B)
;	(SEMIS)	$C3B6	($C3A0)
=	(COMP)	$CAF5	($CAE9) - V Kommando
?	(ADDSUB)	$C89A	($C89B)
A	(ASSEMBLER)	$C6D1	($C6BC)
B	(BASICDATA)	$C96C	($C96D)
C	(CONVERT)	$CA3D	($CA32)

D (DISASS.)	$C55D	($C542)	P (SETPRINTER)	$C83D	($C829)
F (FIND)	$CB11	($CB0C)	R (REGISTER)	$C386	($C370)
G (GO)	$C3E3	($C3CD)	S (LOADSAVE)	$C84E	($C83A)
I (IO.SET)	$C844	($C830)	T (TRACE)	$CBF1	($CBEC)
K (KONTROLLE)	$CAB7	($CAAC)	V (VERSCHIEB)	$CA43	($CA38) – U Kommando
L (LOADSAVE)	$C84E	($C83A)	W (WRITE)	$C9D3	($C9D4)
M (MEMDUMP)	$C3F9	($C3E3)	X (EXIT)	$C36E	($C369)
O (OCUPPY)	$C9C1	($C9C2)	Z (DMON)	$CE09	($CDFE)

```
 programm : smon $c000          c000 cffa
-----------------------------------------

c000 : a9 14 8d 16 03 a9 c2 8d  7d
c008 : 17 03 00 27 23 24 25 2c  c6
c010 : 3a 3b 3d 3f 41 42 43 44  db
c018 : 46 47 49 4b 4c 4d 4f 50  cb
c020 : 52 53 54 56 57 58 5a 00  9d
c028 : 00 00 00 da ca 2d c9 07  cf
c030 : c9 1b c9 fb c6 1c c4 b5  44
c038 : c3 f4 ca 99 c8 d0 c6 6b  60
c040 : c9 3c ca 5c c5 10 cb e2  37
c048 : c3 43 c8 b6 ca 4d c8 f8  e2
c050 : c3 c0 c9 3c c8 85 c3 4d  d0
c058 : c8 f0 cb 42 ca d2 c9 6d  19
c060 : c3 08 ce 00 00 00 00 00  db
c068 : 00 00 00 ff ff 01 00 41  f3
c070 : 5a 49 52 54 80 20 40 10  b8
c078 : 00 02 01 01 02 00 91 91  63
c080 : 0d 53 d9 31 37 32 0d 00  0d
c088 : 7d 4c 7d c9 0d 0d 20 20  be
c090 : 50 43 20 20 53 52 20 41  59
c098 : 43 20 58 52 20 59 52 20  a2
c0a0 : 53 50 20 20 4e 56 2d 42  f8
c0a8 : 44 49 5a 43 00 02 04 01  b2
c0b0 : 2c 00 2c 59 29 58 9d 1f  1d
c0b8 : ff 1c 1c 1f 1f 1f 1c df  cb
c0c0 : 1c 1f df ff ff 03 1f 80  f9
c0c8 : 09 20 0c 04 10 01 11 14  da
c0d0 : 96 1c 19 94 be 6c 03 13  cf
c0d8 : 01 02 02 03 03 02 02 02  08
c0e0 : 02 02 02 03 03 02 03 03  17
c0e8 : 03 02 00 40 40 80 80 20  3f
c0f0 : 10 25 26 21 22 81 82 21  bb
c0f8 : 82 84 08 08 e7 e7 e7 e7  ed
c100 : e3 e3 e3 e3 e3 e3 e3 e3  ff
c108 : e3 e3 e7 a7 e7 e7 f3 f3  41
c110 : f7 df 26 46 06 66 41 81  e5
c118 : e1 01 a0 a2 a1 c1 21 61  66
c120 : 84 86 e6 c6 e0 c0 24 4c  b7
c128 : 20 90 b0 f0 30 d0 10 50  45
c130 : 70 78 00 18 d8 58 b8 ca  a8
c138 : 88 e8 c8 ea 48 08 68 28  7a
c140 : 40 60 aa a8 ba 8a 9a 98  0c
c148 : 38 f8 89 9c 9e b2 2a 4a  af
c150 : 0a 6a 4f 23 93 b3 f3 33  d5
c158 : d3 13 53 73 52 4c 41 52  29
c160 : 45 53 53 4f 4c 4c 4c 43  ec
c168 : 41 41 53 53 49 44 43 43  d3
c170 : 42 4a 4a 42 42 42 42 42  76
c178 : 42 42 42 53 42 43 43 43  a8
c180 : 43 44 44 49 49 4e 50 50  09
c188 : 50 50 52 52 54 54 54 54  c1
c190 : 54 54 53 53 4f 53 53 4f  c9
c198 : 4f 54 42 52 44 44 44 4d  fe
c1a0 : 4e 44 54 54 4e 45 50 50  a1
c1a8 : 49 4d 53 43 43 45 4d 4e  05
c1b0 : 50 56 56 45 52 4c 4c 4c  bb
c1b8 : 4c 45 45 4e 4e 4f 48 48  d3
c1c0 : 4c 4c 54 54 41 41 53 58  ee
c1c8 : 58 59 45 45 4c 52 4c 52  f4
c1d0 : 52 41 43 41 59 58 41 50  ba
c1d8 : 44 43 59 58 43 43 58 59  82
c1e0 : 54 50 52 43 53 51 49 45  c9
c1e8 : 4c 43 53 49 4b 43 44 49  46
c1f0 : 56 58 59 58 59 50 41 50  91
c1f8 : 41 50 49 53 58 59 58 41  52
c200 : 53 41 43 44 08 84 81 22  3c
c208 : 21 26 20 80 03 20 1c 14  1e
c210 : 14 10 04 0c d8 a9 08 8d  c5
c218 : b0 02 a9 04 8d af 02 a9  66
c220 : 06 8d 20 d0 8d 21 d0 a9  87
c228 : 03 8d 86 02 a2 05 68 9d  03
c230 : a8 02 ca 10 f9 ad a9 02  46
c238 : d0 03 ce a8 02 ce a9 02  94
c240 : ba 8e ae 02 a9 52 4c ff  8c
c248 : c2 20 c2 c2 f0 0b 20 7e  08
c250 : c2 8d a9 02 a5 fc 8d a8  4d
c258 : 02 60 a2 a4 20 80 c2 20  19
c260 : 80 c2 d0 1c 20 7e c2 a9  4d
c268 : fe 85 fd a9 ff 85 fe 20  46
c270 : c2 c2 d0 0c 8d 77 02 e6  b3
c278 : c6 60 20 7e c2 2c a2 fb  56
c280 : 20 8d c2 95 01 20 9a c2  cb
c288 : 95 00 e8 e8 60 20 ca c2  2c
c290 : c9 20 f0 f9 c9 2c f0 f5  92
c298 : d0 03 20 ca c2 20 af c2  bd
c2a0 : 0a 0a 0a 0a 85 b4 20 ca  87
c2a8 : c2 20 af c2 05 b4 60 c9  ca
c2b0 : 3a 90 02 69 08 29 0f 60  a7
c2b8 : 20 ca c2 c9 20 f0 f9 c6  26
c2c0 : d3 60 20 cf ff c6 d3 c9  de
c2c8 : 0d 60 20 cf ff c9 0d d0  2b
c2d0 : f8 a9 3f 20 d2 ff ae ae  b6
c2d8 : 02 9a a2 00 86 c6 20 51  92
c2e0 : c3 a1 d1 c9 27 f0 11 c9  f3
c2e8 : 3a f0 0d c9 3b f0 09 c9  0a
c2f0 : 2c f0 05 a9 2e 20 d2 ff  3a
c2f8 : 20 ca c2 c9 2e f0 f9 85  c4
c300 : ac 29 7f a2 20 dd 0a c0  10
c308 : f0 05 ca d0 f8 f0 c2 20  aa
c310 : 15 c3 4c d6 c2 8a 0a aa  f3
c318 : e8 bd 29 c0 48 ca bd 29  65
c320 : c0 48 60 a5 fc 20 2a c3  d2
c328 : a5 fb 48 4a 4a 4a 4a 20  87
c330 : 35 c3 68 29 0f c9 0a 90  0f
c338 : 02 69 06 69 30 4c d2 ff  4e
c340 : a9 0d 20 d2 ff 8a 4c d2  fd
c348 : ff 20 4c c3 a9 20 4c d2  55
c350 : ff a9 0d 4c d2 ff 85 bb  ab
c358 : 84 bc a0 00 b1 bb f0 06  2b
c360 : 20 d2 ff c8 d0 f6 60 e6  16
c368 : fb d0 02 e6 fc 60 a9 0e  be
c370 : 8d 86 02 8d 20 d0 a9 06  ae
c378 : 8d 21 d0 a9 37 85 01 ae  00
c380 : ae 02 9a 4c 74 a4 a0 c0  d0
c388 : a9 8c 20 56 c3 a2 3b 20  c9
c390 : 40 c3 ad a8 02 85 fc ad  ce
c398 : a9 02 85 fb 20 23 c3 20  8e
c3a0 : 4c c3 a2 fb bd af 01 20  93
c3a8 : 2a c3 20 4c c3 e8 d0 f4  f6
c3b0 : ad aa 02 4c d0 c3 20 4e  05
c3b8 : c2 a2 fb 20 ca c2 20 9a  47
c3c0 : c2 9d af 01 e8 d0 f4 20  86
c3c8 : 4c c3 bd aa 02 4c d0 c3  08
c3d0 : 85 aa a9 20 a0 09 20 d2  91
c3d8 : ff 06 aa a9 30 69 00 88  19
c3e0 : d0 f4 60 20 49 c2 ae ae  09
c3e8 : 02 9a a2 fa bd ae 01 48  25
c3f0 : e8 d0 f9 68 a8 68 aa 68  15
c3f8 : 40 20 64 c2 a2 3a 20 40  b7
c400 : c3 20 23 c3 a0 20 a2 00  aa
c408 : 20 4c c3 a1 fb 20 2a c3  64
c410 : a1 fb 20 39 c4 d0 f1 20  b9
c418 : 5d c4 90 e0 60 20 7e c2  9e
c420 : a0 20 a2 00 20 ca c2 20  1d
c428 : 9a c2 81 fb c1 fb f0 03  c9
c430 : 4c d1 c2 20 39 c4 d0 ec  f0
c438 : 60 c9 20 90 0c c9 60 90  49
c440 : 0a c9 c0 90 04 c9 db 90  90
c448 : 04 a9 2e 29 3f 29 7f 91  30
c450 : d1 ad 86 02 91 f3 20 67  e2
c458 : c3 c8 c0 28 60 20 6f c4  03
c460 : 4c 66 c4 20 67 c3 a5 fb  38
c468 : c5 fd a5 fc e5 fe 60 20  4d
c470 : 94 c4 20 86 c4 f0 0e 20  8b
c478 : 86 c4 f0 fb c9 20 d0 05  07
c480 : 8d 77 02 e6 c6 60 20 e4  e0
c488 : ff 48 20 e1 ff f0 02 68  50
c490 : 60 4c d6 c2 a0 28 24 ac  59
c498 : 10 f6 84 c8 84 d0 a9 ff  d3
c4a0 : 20 c3 ff a9 ff 85 b8 85  f1
c4a8 : b9 ad af 02 85 ba 20 c0  94
c4b0 : ff a2 00 86 d3 ca 20 c9  79
c4b8 : ff 20 cf ff 20 d2 ff c9  e7
c4c0 : 0d d0 f6 20 cc ff a9 91  8d
c4c8 : 4c d2 ff a0 00 b1 fb 24  57
c4d0 : aa 30 02 50 0c a2 1f dd  2b
c4d8 : 3c c1 f0 2f ca e0 15 d0  c0
c4e0 : f6 a2 04 dd 49 c1 f0 21  8d
c4e8 : dd 4d c1 f0 1e ca d0 f3  5e
c4f0 : a2 38 dd 11 c1 f0 14 ca  d1
c4f8 : e0 16 d0 f6 b1 fb 3d fb  de
c500 : c0 5d 11 c1 f0 05 ca d0  ef
c508 : f3 a2 00 86 ad 8a f0 0f  2e
c510 : a2 11 b1 fb 3d b5 c0 5d  66
c518 : c6 c0 f0 03 ca d0 f3 bd  59
c520 : ea c0 85 ab bd d8 c0 85  f2
c528 : b6 a6 ad 60 a0 01 b1 fb  7a
c530 : aa c8 b1 fb a0 10 c4 ab  1f
c538 : d0 07 20 4a c5 a0 03 d0  ec
c540 : 02 a4 b6 8e ae 00 8d af  94
c548 : 00 60 a0 01 b1 fb 10 01  fe
c550 : 88 38 65 fb aa e8 f0 01  85
c558 : 88 98 65 fc 60 a2 00 86  4d
c560 : aa 20 64 c2 20 8c c5 a5  54
c568 : ad c9 16 f0 09 c9 30 f0  1f
c570 : 05 c9 21 d0 11 ea 20 94  ce
c578 : c4 20 51 c3 a2 23 a9 2d  5d
c580 : 20 d2 ff ca d0 fa 20 5d  83
c588 : c4 90 d9 60 a2 2c 20 40  a3
c590 : c3 20 23 c3 20 4c c3 20  58
c598 : 75 c6 20 cb c4 20 4c c3  f8
c5a0 : b1 fb 20 2a c3 20 4c c3  92
c5a8 : c8 c4 b6 d0 f3 a9 03 38  a3
c5b0 : e5 b6 aa f0 09 20 49 c3  f7
c5b8 : 20 4c c3 ca d0 f7 a9 20  fc
c5c0 : 20 d2 ff a0 00 a6 ad d0  eb
c5c8 : 11 a2 03 a9 2a 20 d2 ff  0f
c5d0 : ca d0 f8 24 aa 30 85 4c  a0
c5d8 : 6a c6 24 aa 50 29 a9 08  09
c5e0 : 24 ab f0 23 b1 fb 29 fc  14
c5e8 : 85 ad c8 b1 fb 0a a8 b9  d2
c5f0 : 3c 03 8d ae 00 c8 b9 3c  8d
c5f8 : 03 8d af 00 20 be c6 a4  0a
c600 : b6 20 93 c6 20 cb c4 bd  73
c608 : 5b c1 20 d2 ff bd 93 c1  66
c610 : 20 d2 ff bd cb c1 20 d2  42
c618 : ff a9 20 24 ab f0 03 20  07
c620 : 49 c3 a2 20 a9 04 24 ab  9a
c628 : f0 02 a2 28 8a 20 d2 ff  bc
c630 : 24 ab 50 05 a9 23 20 d2  b8
c638 : ff 20 2c c5 88 f0 16 a9  c7
c640 : 08 24 ab f0 07 a9 4d 20  96
c648 : d2 ff a0 01 b9 ad 00 20  ab
c650 : 2a c3 88 d0 f7 a0 03 b9  9c
c658 : ac c0 24 ab f0 09 b9 af  80
c660 : c0 be b2 c0 20 42 c3 88  78
c668 : d0 ed a5 b6 20 67 c3 38  2c
c670 : e9 01 d0 f8 60 a4 d3 a9  fb
c678 : 20 91 d1 c8 c0 28 90 f9  72
c680 : 60 e4 ab d0 04 05 ad 85  81
c688 : ad 60 b9 ad 00 91 fb d1  a9
c690 : fb d0 04 88 10 f4 60 68  00
c698 : 68 60 d0 1c 8a 05 ab 85  72
c6a0 : ab a9 04 85 b5 20 cf ff  6d
c6a8 : c9 20 f0 0d c9 24 f0 09  f3
c6b0 : c9 28 f0 05 c9 2c f0 01  2e
c6b8 : 60 c6 b5 d0 e8 60 e0 18  48
c6c0 : 30 0e ad ae 00 38 e9 02  a6
c6c8 : 38 e5 fb 8d ae 00 a0 40  91
c6d0 : 60 20 7e c2 85 fd a5 fc  11
c6d8 : 85 fe 20 51 c3 20 e4 c6  6d
c6e0 : 30 fb 10 f6 a9 00 85 d3  49
c6e8 : 20 4c c3 20 23 c3 20 4c  8d
c6f0 : c3 20 cf ff a9 01 85 d3  17
c6f8 : a2 80 d0 05 a2 80 8e b1  7b
c700 : 02 86 aa 20 7e c2 a9 25  e3
c708 : 85 c8 2c b1 02 10 08 a2  39
c710 : 0a 20 cf ff ca d0 fa a9  90
c718 : 00 8d b1 02 20 a1 c6 c9  49
c720 : 46 d0 16 46 aa 68 68 a2  f2
c728 : 02 b5 fa 48 b5 fc 95 fa  5c
c730 : 68 95 fc ca d0 f3 4c 64  a2
c738 : c5 c9 2e d0 11 20 9a c2  89
c740 : a0 00 91 fb d1 fb d0 04  0c
c748 : 20 67 c3 c8 88 60 a2 fd  38
c750 : c9 4d d0 19 20 9a c2 a0  3a
c758 : 00 c9 3f b0 ef 0a a8 a5  60
c760 : fb 99 3c 03 a5 fc c8 99  30
c768 : 3c 03 20 a1 c6 95 a9 e0  e4
c770 : fd d0 04 a9 07 85 b7 e8  59
c778 : d0 f0 a2 38 a5 a6 dd 5b  2e
```

```
c780 : c1 f0 05 ca d0 f6 ca 60  05
c788 : a5 a7 dd 93 c1 d0 f4 a5  ac
c790 : a8 dd cb c1 d0 ed bd 11  e7
c798 : c1 85 ad 20 a1 c6 a0 00  5e
c7a0 : e0 20 10 09 c9 20 d0 08  a7
c7a8 : bd 4d c1 85 ad 4c 31 c8  c0
c7b0 : a0 08 c9 4d f0 20 a0 40  83
c7b8 : c9 23 f0 1a 20 9d c2 8d  a7
c7c0 : ae 00 8d af 00 20 a1 c6  dd
c7c8 : a0 20 c9 30 90 1b c9 47  88
c7d0 : b0 17 a0 80 c6 d3 20 a1  13
c7d8 : c6 20 9d c2 8d ae 00 20  fd
c7e0 : a1 c6 c0 08 f0 03 20 be  3b
c7e8 : c6 84 ab a2 01 c9 58 20  2f
c7f0 : 9a c6 a2 04 c9 29 20 9a  b2
c7f8 : c6 a2 02 c9 59 20 9a c6  58
c800 : a5 ad 29 0d f0 0a a2 40  d2
c808 : a9 08 20 81 c6 a9 18 2c  60
c810 : a9 1c a2 82 20 81 c6 a0  2b
c818 : 08 a5 ad c9 20 f0 09 be  c3
c820 : 03 c2 b9 0b c2 20 81 c6  15
c828 : 88 d0 f4 a5 ab 10 01 c8  db
c830 : c8 20 8a c6 c6 b7 a5 b7  b4
c838 : 85 d3 4c 97 c5 20 8d c2  c6
c840 : 8d af 02 60 20 8d c2 8d  c6
c848 : b0 02 60 4c d1 c2 a0 02  55
c850 : 84 bc 88 84 b9 84 bb 88  a5
c858 : 84 b7 20 ca c2 c9 22 d0  be
c860 : ea 20 ca c2 91 bb c8 e6  4d
c868 : b7 c9 22 d0 f4 c6 b7 ad  66
c870 : b0 02 85 ba a5 ac c9 53  67
c878 : f0 13 20 c2 c2 f0 09 a2  6f
c880 : c3 20 80 c2 a9 00 85 b9  f0
c888 : a9 00 6c 30 03 a2 c1 20  df
c890 : 80 c2 a2 ae 20 80 c2 6c  da
c898 : 32 03 20 7e c2 20 ca c2  02
c8a0 : 49 02 4a 4a 08 20 80 c2  cf
c8a8 : 20 51 c3 28 b0 0c a5 fd  65
c8b0 : 65 fb aa a5 fe 65 fc 38  f2
c8b8 : b0 09 a5 fb e5 fd aa a5  1a
c8c0 : fc e5 fe a8 8a 84 fc 85  4f
c8c8 : fb 84 62 85 63 08 a9 00  6c
c8d0 : 85 d3 20 75 c6 a5 fc d0  25
c8d8 : 0f 20 49 c3 a5 fb 20 2a  d1
c8e0 : c3 a5 fb 20 d0 c3 f0 03  6e
c8e8 : 20 23 c3 20 4c c3 a2 90  1d
c8f0 : a5 01 8d b1 02 a9 37 85  05
c8f8 : 01 28 20 49 bc 20 dd bd  fe
c900 : ae b1 02 86 01 4c 56 c3  2c
c908 : 20 8d c2 aa a4 d3 b1 d1  48
c910 : 49 20 f0 a3 8a a8 20 9a  bd
c918 : c2 38 b0 a9 20 b8 c2 a0  6c
c920 : 08 48 20 ca c2 c9 31 68  be
c928 : 2a 88 d0 f5 f0 eb 20 b8  e9
c930 : c2 a2 00 8a 86 fb 85 fc  ed
c938 : a8 20 cf ff c9 3a b0 84  1e
c940 : e9 2f b0 04 38 4c c4 c8  f8
c948 : 85 fd 06 fb 26 fc a5 fc  a8
c950 : 85 fe a5 fb 0a 26 fe 0a  1f
c958 : 26 fe 18 65 fb 08 18 65  db
c960 : fd aa a5 fe 65 fc 28 69  ad
c968 : 00 4c 34 c9 20 7a c2 a9  09
c970 : 37 85 01 a2 04 bd 87 c0  cc
c978 : 95 aa ca 10 f8 20 51 c3  74
c980 : a6 aa a5 ab 20 cd bd e6  8f
c988 : aa d0 02 e6 ab a9 44 20  51
c990 : d2 ff a9 c1 20 d2 ff a0  de
c998 : 00 b1 fb 84 62 85 63 20  20
c9a0 : d1 bd 20 63 c4 a2 03 b0  93
c9a8 : 0a a9 2c a6 d3 e0 49 90  f1
c9b0 : e3 a2 09 86 c6 bd 7d c0  c9
c9b8 : 9d 76 02 ca d0 f7 4c 6e  45
c9c0 : c3 20 7a c2 20 8d c2 a2  49
c9c8 : 00 81 fb 48 20 63 c4 68  92
c9d0 : 90 f7 60 20 5a c2 a5 a6  18
c9d8 : d0 02 c6 a7 c6 a6 20 30  d2
c9e0 : ca 86 b5 a0 02 90 04 a2  69
c9e8 : 02 a0 00 18 a5 a6 65 ae  c0
c9f0 : 85 aa a5 a7 65 af 85 ab  6a
c9f8 : a1 a4 81 a8 41 a8 05 b5  3a
ca00 : 85 b5 a5 a4 c5 a6 a5 a5  d1
ca08 : e5 a7 b0 1d 18 b5 a4 79  45
ca10 : 6b c0 95 a4 b5 a5 79 6c  1c
ca18 : c0 95 a5 8a 18 69 04 aa  90
ca20 : c9 07 90 e8 e9 08 aa b0  99
ca28 : cf a5 b5 f0 0f 4c d1 c2  75
ca30 : 38 a2 fe b5 aa f5 a6 95  50
ca38 : b0 e8 d0 f7 60 20 62 ca  b5
ca40 : 4c d6 c9 4c 62 ca c5 a7  d6
ca48 : d0 02 e4 a6 b0 13 c5 a5  2d
ca50 : d0 02 e4 a4 90 0b 85 b4  d0
ca58 : 8a 18 65 ae aa a5 b4 65  93
ca60 : af 60 20 5a c2 20 7a c2  2f
ca68 : 20 30 ca 20 cb c4 c8 a9  b0
ca70 : 10 24 ab f0 26 a6 fb a5  6e
ca78 : fc 20 46 ca 86 aa b1 fb  ec
ca80 : 85 b5 20 4a c5 a0 01 20  d7
ca88 : 46 ca ca 8a 18 e5 aa 91  b6
ca90 : fb 45 b5 10 19 20 51 c3  fd
ca98 : 20 23 c3 24 ab 10 0f b1  9a
caa0 : fb aa c8 b1 fb 20 46 ca  c8
caa8 : 91 fb 8a 88 91 fb 20 6a  39
cab0 : c6 20 66 c4 90 b5 60 20  31
cab8 : 64 c2 a2 27 20 40 c3 20  5e
cac0 : 23 c3 a0 08 a2 00 20 4c  31
cac8 : c3 a1 fb 20 39 c4 d0 f9  50
cad0 : a2 00 20 5d c4 f0 03 4c  9f
cad8 : ba ca 60 20 7e c2 a0 03  9a
cae0 : 20 cf ff 88 d0 fa 20 ca  f4
cae8 : c2 c9 2e f0 02 91 fb c8  67
caf0 : c0 20 90 f2 60 20 7a c2  b9
caf8 : a2 00 a1 fb c1 fd d0 0b  e7
cb00 : 20 67 c3 e6 fd d0 f3 e6  a5
cb08 : fe d0 ef 20 4c c3 4c 23  c9
cb10 : c3 a9 ff a2 04 95 fa ca  6a
cb18 : d0 fb 20 ca c2 a2 05 dd  58
cb20 : 6e c0 f0 45 ca d0 f8 86  f7
cb28 : a9 20 b4 cb e8 20 cf ff  57
cb30 : c9 20 f0 f3 c9 2c d0 03  0b
cb38 : 20 7a c2 20 51 c3 a4 a9  63
cb40 : b1 fb 20 d6 cb d0 18 88  86
cb48 : 10 f6 20 23 c3 20 4c c3  36
cb50 : a4 d3 c0 24 90 09 20 94  8d
cb58 : c4 20 72 c4 20 51 c3 20  3d
cb60 : 63 c4 90 da a0 27 4c 96  46
cb68 : c4 bd 73 c0 85 a8 bd 78  85
cb70 : c0 85 a9 aa f0 06 20 b4  dc
cb78 : cb ca d0 fa 20 7a c2 20  5d
cb80 : cb c4 20 2c c5 a5 a8 24  af
cb88 : ab d0 09 a8 d0 21 a5 ad  fb
cb90 : d0 1d f0 0d a4 a9 b9 ad  a6
cb98 : 00 20 d6 cb d0 11 88 d0  31
cba0 : f5 84 aa 20 8c c5 20 6f  dc
cba8 : c4 20 66 c4 90 d1 60 20  08
cbb0 : 6a c6 f0 f5 20 c0 cb 9d  eb
cbb8 : cc 03 bd 3c 03 9d 6c 03  d2
cbc0 : 20 ca c2 a0 0f c9 2a d0  94
cbc8 : 02 a0 00 20 af c2 9d 3c  1e
cbd0 : 03 98 9d 9c 03 60 85 b4  cd
cbd8 : 4a 4a 4a 4a 59 6c 03 39  9b
cbe0 : cc 03 29 0f d0 0a a5 b4  b7
cbe8 : 59 3c 03 39 9c 03 29 0f  ec
cbf0 : 60 68 68 20 cf ff c9 57  75
cbf8 : d0 03 4c 56 cd c9 42 d0  fe
cc00 : 03 4c d0 cd c9 51 d0 03  87
cc08 : 4c 4f cd c9 53 f0 03 4c  0a
cc10 : d1 c2 20 8d c2 48 20 8d  06
cc18 : c2 48 20 49 c2 a0 00 b1  c4
cc20 : fb 8d bc 02 98 91 fb a9  ab
cc28 : 36 8d 16 03 a9 cc 8d 17  70
cc30 : 03 a2 fc 4c ec c3 a2 03  ca
cc38 : 68 9d aa 02 ca 10 f9 68  40
cc40 : 68 ba 8e ae 02 ad a8 02  b3
cc48 : 85 fc ad a9 02 85 fb ad  83
cc50 : bc 02 a0 00 91 fb a9 14  fd
cc58 : 8d 16 03 a9 c2 8d 17 03  e1
cc60 : a9 52 4c ff c2 20 51 c3  3f
cc68 : ad 11 d0 09 10 8d 11 d0  46
cc70 : 60 8d ab 02 08 68 29 ef  0a
cc78 : 8d aa 02 8e ac 02 8c ad  15
cc80 : 02 68 18 69 01 8d a9 02  11
cc88 : 68 69 00 8d a8 02 a9 80  99
cc90 : 8d bc 02 d0 10 20 e5 cd  4b
cc98 : 20 dd fd d8 a2 05 68 9d  70
cca0 : a8 02 ca 10 f9 ad 14 03  61
cca8 : 8d bb 02 ad 15 03 8d ba  5e
ccb0 : 02 ba 8e ae 02 58 ad aa  78
ccb8 : 02 29 10 f0 08 20 65 cc  22
ccc0 : a9 52 4c ff c2 2c bc 02  2a
ccc8 : 50 1f 38 ad a9 02 ed bd  49
ccd0 : 02 8d b1 02 ad a8 02 ed  49
ccd8 : be 02 0d b1 02 d0 67 ad  b0
cce0 : bf 02 d0 5f a9 80 8d bc  0f
cce8 : 02 30 12 4e bc 02 90 cd  0a
ccf0 : ae ae 02 9a a9 cc 48 a9  3f
ccf8 : 70 48 4c ba cd 20 65 cc  04
cd00 : a9 a8 85 fb a9 02 85 fc  99
cd08 : 20 4c c3 a0 00 b1 fb 20  11
cd10 : 2a c3 c8 c0 07 f0 09 c0  04
cd18 : 01 f0 f2 20 4c c3 d0 ed  54
cd20 : ad a9 02 ae a8 02 85 fb  a1
cd28 : 86 fc 20 49 c3 20 cb c4  54
cd30 : 20 c7 c5 20 e4 ff f0 fb  b3
cd38 : c9 4a d0 0a a9 01 8d bc  ee
cd40 : 02 d0 2f ce bf 02 a5 91  16
cd48 : c9 7f d0 26 4c bd cc 20  f0
cd50 : f2 cd a9 40 d0 0a 20 f2  5f
cd58 : cd 08 68 8d aa 02 a9 80  57
cd60 : 8d bc 02 ba 8e ae 02 20  ca
cd68 : 49 c2 20 65 cc ad bc 02  f8
cd70 : f0 37 a2 00 ad 11 d0 a8  9c
cd78 : 29 10 f0 10 98 29 ef 8d  95
cd80 : 11 d0 ea ea a0 0c ca d0  48
cd88 : fd 88 d0 fa 78 a9 47 8d  6a
cd90 : 04 dc 8e 05 dc ad 0e dc  74
cd98 : 29 80 09 11 8d 0e dc a9  76
cda0 : 95 a2 cc 8d bb 02 8e ba  e7
cda8 : 02 ae ae 02 9a 78 ad bb  89
cdb0 : 02 ae ba 02 8d 14 03 8e  9b
cdb8 : 15 03 ad a8 02 48 ad a9  3c
cdc0 : 02 48 ad aa 02 48 ad ab  17
cdc8 : 02 ae ac 02 ac ad 02 40  4d
cdd0 : 20 8d c2 8d be 02 20 8d  b1
cdd8 : c2 8d bd 02 20 8d c2 8d  a5
cde0 : bf 02 4c d6 c2 ad b8 02  0f
cde8 : ae b9 02 8d 14 03 8e 15  63
cdf0 : 03 60 ad 14 03 ae 15 03  11
cdf8 : 8d b8 02 8e b9 02 a9 95  b1
ce00 : 8d 16 03 a9 cc 8d 17 03  2a
ce08 : 60 a9 07 8d 20 d0 a9 36  4c
ce10 : 85 01 a2 00 bd e4 cf 9d  3c
ce18 : d0 02 e8 e0 0d 90 f5 a2  b2
ce20 : 2a 20 40 c3 20 cf ff c9  f7
ce28 : 2a f0 f9 a2 06 dd d2 cf  d7
ce30 : d0 11 8e c1 02 8a 0a aa  57
ce38 : e8 bd d8 cf 48 ca bd d8  b2
ce40 : cf 48 60 ca 10 e7 4c 1f  54
ce48 : ce a9 00 85 fb a9 bf 85  b3
ce50 : fc 85 fe a5 fb 69 04 85  a9
ce58 : fd 20 fc c3 20 e1 ff f0  10
ce60 : 0f ad 8d 02 f0 f6 a9 00  57
ce68 : 85 c6 a5 fc c9 c0 90 e3  06
ce70 : 4c 1f ce 20 7e c2 a0 20  c4
ce78 : a2 00 20 ca c2 20 9a c2  99
ce80 : 81 fb 20 39 c4 d0 f3 20  11
ce88 : 51 c3 4c 24 ce 20 55 cf  35
ce90 : ad c1 02 c9 02 d0 03 4c  23
ce98 : eb ce a2 00 bd 00 bf 8d  89
cea0 : c3 02 e8 bd 00 bf 8d c4  14
cea8 : 02 8a 4c cb ce 20 c2 c2  fa
ceb0 : d0 03 4c 8d ce 20 8d c2  70
ceb8 : 8d c3 02 20 8d c2 8d c4  5a
cec0 : 02 20 55 cf ad c1 02 c9  a6
cec8 : 02 f0 20 20 0d cf a2 0d  42
ced0 : 20 c6 ff a0 00 20 cf ff  a7
ced8 : ea ea ea ea 99 00 bf c8  79
cee0 : d0 f3 20 cc ff 20 bc cf  df
cee8 : 4c 49 ce 20 40 cf a2 0d  b8
cef0 : 20 c9 ff a0 00 b9 00 bf  56
cef8 : 20 d2 ff a6 90 d0 03 c8  83
cf00 : d0 f3 20 cc ff a9 32 20  c2
cf08 : 0d cf 4c b6 cf 8d d1 02  9b
cf10 : ad c3 02 20 79 cf 8e d8  25
cf18 : 02 8d d9 02 ad c4 02 20  e1
cf20 : 79 cf 8e db 02 8d dc 02  a4
cf28 : a2 0f 20 c9 ff a2 00 bd  24
cf30 : d0 02 20 d2 ff e8 e0 0d  49
cf38 : 90 f5 20 cc cf 4c 8c cf  99
cf40 : a2 0f 20 c9 ff a2 00 bd  3c
cf48 : f2 cf 20 d2 ff e8 e0 08  5f
cf50 : 90 f5 4c cc ff a9 0f a8  62
cf58 : a2 08 20 ba ff a9 00 20  eb
cf60 : bd ff 20 c0 ff a9 0d a8  10
cf68 : a2 08 20 ba ff a9 01 a2  04
cf70 : f1 a0 cf 20 bd ff 4c c0  38
cf78 : ff a2 30 38 e9 0a 90 03  13
cf80 : e8 b0 f9 69 3a 60 20 8c  ac
cf88 : cf 4c b6 cf a9 00 85 90  f7
cf90 : 20 51 c3 a9 08 20 b4 ff  d3
cf98 : a9 6f 20 96 ff 20 a5 ff  6b
cfa0 : c9 30 d0 06 4c ab ff 20  d9
cfa8 : a5 ff 20 d2 ff c9 0d d0  d3
cfb0 : f6 20 ab ff 68 68 20 bc  65
cfb8 : cf 4c 1f ce a9 0d 20 c3  5a
cfc0 : ff a9 0f 4c c3 ff a9 06  d0
cfc8 : 8d 20 d0 a9 37 85 01 4c  0b
cfd0 : d6 c2 3a 52 57 4d 58 40  a2
cfd8 : 72 ce ac ce ac ce 48 ce  b6
cfe0 : c5 cf 85 cf 55 31 3a 31  12
cfe8 : 33 20 30 20 31 38 20 30  f1
cff0 : 30 23 42 2d 50 20 31 33  19
cff8 : 20 30 b7 00 ff 00 ff 00  1e
```

Listing 1.
»SMON-komplett«-Hauptprogramm.
Bitte beachten Sie die Eingabehinweise auf Seite 110.

```
 programm : ndisass        ce09 cf3d
-------------------------------------

ce09 : 2b 4b 6b 8b 9b ab bb cb  c4
ce11 : eb 89 93 9f 0b 9c 9e 4e  46
ce19 : 53 52 53 52 53 4c 44 49  f0
ce21 : 43 4f 4c 4c 52 52 41 41  e8
ce29 : 43 53 52 50 4f 41 45 41  4b
ce31 : 58 58 50 43 41 25 26 20  48
ce39 : 21 82 80 81 22 21 82 81  24
ce41 : 03 13 07 17 1b 0f 1f 97  48
ce49 : d7 bf df 02 02 02 02 03  76
ce51 : 03 03 02 02 03 03 a2 02  6e
ce59 : d0 28 a6 ad d0 2b a2 01  90
ce61 : b1 fb c9 9c f0 38 c9 80  0f
ce69 : f0 ec c9 89 f0 e8 29 0f  8c
ce71 : c9 02 f0 16 c9 0a f0 0a  ff
ce79 : e8 c9 04 f0 05 e8 c9 0c  3c
ce81 : d0 1c 86 b6 a2 01 8e c5  d0
ce89 : 02 60 b1 fb 29 90 49 80  e4
ce91 : d0 04 a2 02 d0 ec 86 b6  48
ce99 : a2 0a 8e c5 02 60 a0 02  46
cea1 : 84 b6 a0 00 8c c5 02 b1  0b
cea9 : fb a2 0f dd 08 ce f0 d9  e3
ceb1 : ca d0 f8 29 01 f0 d2 b1  8d
ceb9 : fb 4a 4a 4a 4a 4a 18 69  df
cec1 : 02 8d c5 02 a2 0b b1 fb  7d
cec9 : 3d 40 ce dd 40 ce f0 03  da
ced1 : ca d0 f3 bd 35 ce 85 ab  ef
ced9 : bd 4b ce 85 b6 60 a0 00  91
cee1 : a6 ad f0 06 20 4c c3 4c  67
cee9 : da c5 ae c5 02 d0 06 20  09
cef1 : 4c c3 4c c9 c5 a9 2a 20  fe
cef9 : d2 ff bd 17 ce 20 d2 ff  56
cf01 : bd 21 ce 20 d2 ff bd 2b  81
cf09 : ce 4c 16 c6 a9 00 8d 6b  03
cf11 : c0 8d 6c c0 a9 4c 8d 29  50
cf19 : c5 8d be c5 a9 20 8d 30  3f
cf21 : cd a9 5b 8d 2a c5 a9 ce  60
cf29 : 8d 2b c5 a9 df 8d bf c5  e7
cf31 : 8d 31 cd a9 ce 8d c0 c5  e7
cf39 : 8d 32 cd 00 4c 8c cf a2  00
```

Listing 2. Mit dieser Erweiterung lassen sich illegale Opcodes disassemblieren.

```
 programm : floppymon m&t cbf1 cdcb
-------------------------------------

cbf1 : a9 36 85 01 a2 00 bd 90  79
cbf9 : cd f0 06 20 d2 ff e8 d0  36
cc01 : f5 20 48 c3 a2 3e 20 37  9c
cc09 : c3 20 cf ff c9 3e f0 f9  16
cc11 : c9 20 f0 f5 a2 05 dd a6  fc
cc19 : cd f0 09 ca d0 f8 20 48  e0
cc21 : c3 4c 05 cc 8a 0a aa e8  5b
cc29 : bd aa cd 48 ca bd aa cd  99
cc31 : 48 60 20 b9 c2 d0 0a a9  17
cc39 : 00 8d 9e cd 8d 9f cd f0  50
cc41 : 1a 20 84 c2 8d 9f cd 20  32
cc49 : b9 c2 d0 07 a9 00 8d 9e  86
cc51 : cd f0 08 20 84 c2 29 f8  91
cc59 : 8d 9e cd 20 77 cd a2 0f  3b
cc61 : 20 c9 ff a2 00 bd 9b cd  b2
cc69 : 20 d2 ff e8 e0 06 90 f5  7c
cc71 : 20 cc ff a2 0f 20 c6 ff  59
cc79 : a0 00 20 cf ff 99 00 bf  68
cc81 : c8 d0 f7 20 cc ff 4c c4  3b
cc89 : cf a9 bf 85 fc a9 00 85  f5
cc91 : fb 60 20 33 cc 20 8a cc  bc
cc99 : a2 3a 20 37 c3 ad 9f cd  0b
cca1 : 20 21 c3 ad 9e cd 20 21  13
cca9 : c3 a0 20 a2 00 20 43 c3  ae
ccb1 : 20 43 c3 a1 fb 20 21 c3  65
ccb9 : a1 fb 20 20 c4 d0 f1 a9  52
ccc1 : 08 18 6d 9e cd 8d 9e cd  64
ccc9 : 08 c9 f8 d0 06 20 5c cc  7a
ccd1 : 20 8a cc 28 90 09 ee 9f  bb
ccd9 : cd 20 5c cc 20 8a cc 20  31
cce1 : 87 cd 20 e1 ff d0 b1 4c  79
cce9 : 02 cc 20 75 c2 a5 fb 8d  6c
ccf1 : a4 cd a5 fc 8d a5 cd 20  02
ccf9 : 19 cd a0 20 a2 00 20 c1  53
cd01 : c2 20 c1 c2 20 91 c2 20  76
cd09 : d2 ff 20 20 c4 d0 f2 20  c6
cd11 : cc ff 20 c4 cf 4c 07 cc  93
cd19 : 20 77 cd a2 0f 20 c9 ff  d6
```

Listing 3. Komfortabler Disketten-Monitor. Bitte beachten Sie die Eingabehinweise auf Seite 110. (Fortsetzung)

```
cd21 : a2 00 bd a1 cd 20 d2 ff  90
cd29 : e8 e0 06 90 f5 60 20 75  e3
cd31 : c2 a2 fd 20 77 c2 a5 fd  e8
cd39 : 8d a4 cd a5 fe 8d a5 cd  cf
cd41 : a9 20 8d a6 cd 20 19 cd  10
cd49 : a0 00 b1 fb 20 d2 ff c8  ff
cd51 : c0 20 90 f6 18 a9 20 6d  4e
cd59 : a4 cd b0 0c 8d a4 cd a9  1a
cd61 : 20 65 fb 85 fb 4c 46 cd  ba
cd69 : 20 cc ff 20 c4 cf a9 08  75
cd71 : 8d a6 cd 4c 02 cc a9 0f  9a
cd79 : a8 a2 08 20 ba ff a9 00  cb
cd81 : 20 bd ff 4c c0 ff 20 e4  60
cd89 : ff f0 fb 60 4c fe cd 0d  19
cd91 : 3e 46 4c 4f 50 2d 4d 4f  32
cd99 : 4e 00 4d 2d 52 00 00 ff  05
cda1 : 4d 2d 57 00 00 08 3a 4d  1f
cda9 : 56 58 40 ea cc 92 cc 2e  8a
cdb1 : cd 8c cd 76 cf a9 00 8d  6c
cdb9 : 22 c0 a9 46 8d df cf a9  d9
cdc1 : cb 8d eb cf a9 f0 8d ea  76
cdc9 : cf 00 56 40 58 ea cc 92  6b
```

Listing 3. (Schluß)

```
 programm : ndisass m&t   ce09 cf3d
-------------------------------------

ce09 : 2b 4b 6b 8b 9b ab bb cb  c4
ce11 : eb 89 93 9f 0b 9c 9e 4e  46
ce19 : 53 52 53 52 53 4c 44 49  f0
ce21 : 43 4f 4c 4c 52 52 41 41  e8
ce29 : 43 53 52 50 4f 41 45 41  4b
ce31 : 58 58 50 43 41 25 26 20  48
ce39 : 21 82 80 81 22 21 82 81  24
ce41 : 03 13 07 17 1b 0f 1f 97  48
ce49 : d7 bf df 02 02 02 02 03  76
ce51 : 03 03 02 02 03 03 a2 02  6e
ce59 : d0 28 a6 ad d0 2b a2 01  90
ce61 : b1 fb c9 9c f0 38 c9 80  0f
ce69 : f0 ec c9 89 f0 e8 29 0f  8c
ce71 : c9 02 f0 16 c9 0a f0 0a  ff
ce79 : e8 c9 04 f0 05 e8 c9 0c  3c
ce81 : d0 1c 86 b6 a2 01 8e c5  d0
ce89 : 02 60 b1 fb 29 90 49 80  e4
ce91 : d0 04 a2 02 d0 ec 86 b6  48
ce99 : a2 0a 8e c5 02 60 a0 02  46
cea1 : 84 b6 a0 00 8c c5 02 b1  0b
cea9 : fb a2 0f dd 08 ce f0 d9  e3
ceb1 : ca d0 f8 29 01 f0 d2 b1  8d
ceb9 : fb 4a 4a 4a 4a 4a 18 69  df
cec1 : 02 8d c5 02 a2 0b b1 fb  7d
cec9 : 3d 40 ce dd 40 ce f0 03  da
ced1 : ca d0 f3 bd 35 ce 85 ab  ef
ced9 : bd 4b ce 85 b6 60 a0 00  91
cee1 : a6 ad f0 06 20 43 c3 4c  1e
cee9 : be c5 ae c5 02 d0 06 20  ed
cef1 : 43 c3 4c ad c5 a9 2a 20  71
cef9 : d2 ff bd 17 ce 20 d2 ff  56
cf01 : bd 21 ce 20 d2 ff bd 2b  81
cf09 : ce 4c f8 c5 a9 00 8d 60  86
cf11 : c0 8d 61 c0 a9 4c 8d 10  5c
cf19 : c5 8d a2 c5 a9 20 8d 28  28
cf21 : cd a9 5b 8d 11 c5 a9 ce  cf
cf29 : 8d 12 c5 a9 df 8d a3 c5  ea
cf31 : 8d 29 cd a9 ce 8d a4 c5  73
cf39 : 8d 2a cd 00 4c 8c cf a2  fc
```

Listing 4. Illegale Opcodes disassemblieren mit der M&T-Version des SMON

```
 programm : floppymon     cbf1 cded
-------------------------------------

cbf1 : a9 36 85 01 a2 00 bd b2  bd
cbf9 : cd f0 06 20 d2 ff e8 d0  36
cc01 : f5 20 51 c3 a2 3e 20 40  f0
cc09 : c3 20 cf ff c9 3e f0 f9  16
cc11 : c9 20 f0 f5 a2 05 dd c8  40
cc19 : cd f0 09 ca d0 f8 20 51  f2
cc21 : c3 4c 05 cc 8a 0a aa e8  5b
cc29 : bd cc cd 48 ca bd cc cd  32
cc31 : 48 60 20 c2 c2 d0 0a a9  38
cc39 : 00 8d c0 cd 8d c1 cd f0  ea
cc41 : 1a 20 8d c2 8d c1 cd 20  85
```

Listing 5. Komfortabler Disketten-Monitor für die M&T-Version (Fortsetzung)

```
cc49 : c2 c2 d0 07 a9 00 8d c0  d4
cc51 : cd f0 08 20 8d c2 29 f8  22
cc59 : 8d c0 cd 20 77 cd a2 0f  4c
cc61 : 20 c9 ff a2 00 bd bd cd  3a
cc69 : 20 d2 ff e8 e0 06 90 f5  7c
cc71 : 20 cc ff a2 0f 20 c6 ff  59
cc79 : a0 00 20 cf ff 99 00 bf  68
cc81 : c8 d0 f7 20 cc ff 4c bc  2b
cc89 : cf a9 bf 85 fc a9 00 85  f5
cc91 : fb 60 20 33 cc 20 8a cc  bc
cc99 : a2 3a 20 40 c3 ad c1 cd  b5
cca1 : 20 2a c3 ad c0 cd 20 2a  cc
cca9 : c3 a0 20 a2 00 20 4c c3  d2
ccb1 : 20 4c c3 a1 fb 20 2a c3  0d
ccb9 : a1 fb 20 39 c4 d0 f1 a9  75
ccc1 : 08 18 6d c0 cd 8d c0 cd  31
ccc9 : 08 c9 f8 d0 06 20 5c cc  7a
ccd1 : 20 8a cc 28 90 09 ee c1  ff
ccd9 : cd 20 5c cc 20 8a cc 20  31
cce1 : 87 cd 20 e1 ff d0 b1 4c  79
cce9 : 02 cc 20 7e c2 a5 fb 8d  8e
ccf1 : c6 cd a5 fc 8d c7 cd 20  35
ccf9 : 19 cd a0 20 a2 00 20 ca  65
cd01 : c2 20 ca c2 20 9a c2 20  00
cd09 : d2 ff 20 39 c4 d0 f2 20  e9
cd11 : cc ff 20 bc cf 4c 07 cc  92
cd19 : 20 77 cd a2 0f 20 c9 ff  d6
cd21 : a2 00 bd c3 cd 20 d2 ff  d4
cd29 : e8 e0 06 90 f5 60 20 7e  f5
cd31 : c2 a2 fd 20 80 c2 a5 fd  78
cd39 : 8d c6 cd a5 fe 8d c7 cd  68
cd41 : a9 20 8d c8 cd 20 19 cd  55
cd49 : a0 00 b1 fb 20 d2 ff c8  ff
cd51 : c0 20 90 f6 18 a9 20 6d  4e
cd59 : c6 cd b0 0c 8d c6 cd a9  4d
cd61 : 20 65 fb 85 fb 4c 46 cd  ba
cd69 : 20 cc ff 20 bc cf a9 08  f4
cd71 : 8d c8 cd 4c 02 cc a9 0f  ab
cd79 : a8 a2 08 20 ba ff a9 00  cb
cd81 : 20 bd ff 4c c0 ff 20 e4  60
cd89 : ff f0 fb 60 20 c2 c2 d0  d0
cd91 : 03 4c 86 cf a9 08 20 b1  15
cd99 : ff a9 6f 20 93 ff 20 cf  a6
cda1 : ff 20 a8 ff c9 0d d0 f6  11
cda9 : 20 ae ff 4c 02 cc 4c 09  74
cdb1 : ce 0d 3e 46 4c 4f 50 2d  39
cdb9 : 4d 4f 4e 00 4d 2d 52 00  c9
cdc1 : 00 ff 4d 2d 57 00 00 08  3f
cdc9 : 3a 4d 56 40 58 ea cc 92  7d
cdd1 : cc 2e cd 8c cd ae cd a9  96
cdd9 : 00 8d 22 c0 a9 46 8d d7  f3
cde1 : cf a9 cb 8d e3 cf a9 f0  6f
cde9 : 8d e2 cf 00 03 8e 15 03  da
```

Listing 5. (Schluß)

```
 programm : illegal-code 4000 40f4
-------------------------------------

4000 : 87 87 c7 c7 e7 e7 a7 a7  e1
4008 : 27 27 67 67 07 07 47 47  de
4010 : d7 d7 f7 ff 37 aa 17 60  b6
4018 : 57 20 97 13 b7 20 8f 0f  a0
4020 : cf cf 01 8f ef 20 0c af  99
4028 : 19 20 2f 24 30 6f 60 60  62
4030 : 0f 0c 04 4f 20 2a df 05  0d
4038 : 06 ff 0f 04 3f 60 60 7f  fa
4040 : 03 0d 1f 20 23 5f 32 30  ec
4048 : db 60 20 fb 05 11 bf 01  b5
4050 : 04 3b 03 0f 7b 12 33 1b  e0
4058 : 60 01 5b 12 03 83 12 c3  6e
4060 : 32 e3 0c a3 18 23 01 63  61
4068 : 01 03 31 43 31 d3 19 f3  9e
4070 : 60 b3 0c 33 01 73 01 13  e9
4078 : 01 53 32 53 32 ea 6d 54  f3
4080 : 6b 03 81 09 a6 07 b9 02  7c
4088 : 43 27 9d 06 ae 07 a1 1a  65
4090 : 5b 3f 85 1e b6 0f a9 12  5f
4098 : 53 37 8d 16 be 2f 91 2a  ad
40a0 : 6b 0f b5 2e 86 67 99 22  14
40a8 : 63 07 bd 26 8e 67 81 3a  62
40b0 : 7b 1f a5 3e 9b 43 b2 21  cd
40b8 : 57 bd b6 b0 4a b5 20 43  0b
40c0 : ef 36 ce 95 59 ee 49 c4  ec
40c8 : 3f c1 ee 4d e7 3a f0 54  0a
40d0 : 72 d1 00 f7 1e f7 f0 f0  71
40d8 : 5a 7b aa 60 fc f7 c9 60  1e
40e0 : 42 95 59 05 cb f2 ea e4  ae
40e8 : 92 a4 e1 94 a1 c2 03 fb  0b
40f0 : 00 54 20 00 00 00 00 00  23
```

Listing 6. Mit dem Befehl D 4000 erscheinen alle illegalen Opcodes disassembliert auf dem Bildschirm