DAS ELEKTRONISCHE MAGAZIN 5/87

INPUT 64

Infos · News · Programme · Unterhaltung · Tips

Volkszählung '87
Bleibt der Bürger anonym?
Eine Simulation für den C64/C128

Byte-Compactor
Bis zu 70 Prozent Platz gespart

SpeedBackup
Diskette in 1 Minute kopiert

Spiele:
Pyramidon
Golum

Serien:
64er Tips
Assembler-Schule

Dokumentation und Bedienungshinweise

INPUT-ein Magazin aus dem Verlag Heinz Heise GmbH, Postfach 61 04 07, 3000 Hannover 61

Liebe(r) 64er-Besitzer(in)!

Titelbilder von Computerzeitschriften haben in der Regel nichts mit Politik zu tun. Warum sollten sie auch, als Themen sind Technik, Hardware, Software und Know-how angesagt. Deswegen werden diese Zeitschriften von den Lesern gekauft, für politische Nachrichten, Analysen und Meinungen sind andere zuständig — von der FAZ bis zur „tageszeitung".

Ein Titelbild zum Thema Volkszählung fällt aus diesem Schema heraus. Wird man von INPUT jetzt auch noch mit Politik belästigt? Man wird. Begrenzt. Ausnahmsweise.

Politische Entscheidungen fallen in erster Linie aufgrund von Interessen und Meinungen, haben gelegentlich (wahrscheinlich zu selten) aber auch etwas mit Fachwissen zu tun. Bei Themen wie Waldsterben oder Rhein-Vergiftung ist auch und gerade die Kompetenz von Wissenschaft und Technik gefragt. Andererseits reichen die Auswirkungen technisch-wissenschaftlicher Entwicklungen weit in politische und gesellschaftliche Bereiche hinein.

Diese Diskussion um die „Wertfreiheit" von Wissenschaft und die politische Verantwortung von Technik ist nicht neu; sie findet auch rund um den Computer und die EDV statt. Die nicht gerade emotionslos geführte Auseinandersetzung wird beherrscht von Schlagwörtern wie „Computerisierung", „Jobkiller", „gläserner Bürger". Daß die Einführung der elektronischen Datenverarbeitung Arbeitsplätze und Freizeitverhalten für viele Menschen radikal umkrempelt, läßt sich auch kaum abstreiten.

Das alles muß Sie als Besitzer eines Homecomputers nicht unbedingt interessieren. Aber: Dem Thema Volkszählung kann sich ganz praktisch niemand entziehen, eine Totalerfassung betrifft nun einmal ausnahmslos jeden. Diese Volkszählung ist, in Teilen und insgesamt, umstritten; die 1983 geplante Volkszählung scheiterte sogar am Einspruch der Verfassungsrichter. Eines der in diesem Zusammenhang aufgetauchten Probleme hängt sehr eng mit Computern zusammen: die Frage nach Datensicherheit und Datenschutz. Werden nur anonyme, statistische Daten gespeichert? Oder sind die mit dem Begriff „gläserner Bürger" zusammengefaßten Befürchtungen berechtigt?

Diese Frage läßt sich nicht ohne die nötige technische Kompetenz beantworten. Und diese technische Kompetenz ist sozusagen die „Schnittstelle" zwischen Politik und Commodore 64. Mit den in dieser Ausgabe vorgestellten Simulationsprogrammen können Sie sich an Ihrem Rechner mit den technischen Aspekten von Datenschutz, Anonymität statistischer Daten, Re-Identifizierbarkeit, kurzum: mit dem „gläsernen Bürger" beschäftigen.

Und diese Vermittlung von computerspezifischem Fachwissen zu gesellschaftlichen Problemen ist durchaus eine Aufgabe von Computerzeitschriften. Welche politischen Schlüsse Sie daraus ziehen, ist dann Ihre Sache.

Ihre INPUT-Redaktion

INHALT

Leser fragen . . .	2	64er Tips: Zeichensatz-Manipulationen	21
Volkszählung '87	3	Strategiespiel: Pyramidon	24
Assembler-Schule Teil 3	10	Assembler-Tips Teil 2: Struktur und Modularität	25
Byte-Compactor	16	Hinweise zur Bedienung	29
SpeedBackup	18	Vorschau	31
Vier Spiele in einem: Golum	19	Impressum	32
ID-Werkstatt: CAD-Patch, MiniDat-Kassetten-Version, VT-Patch, C64/Sharp	20		

Auf einen Blick: INPUT64-Betriebssystem-Befehle

Titel abkürzen	CTRL und Q
Hilfsseite aufrufen	CTRL und H
zum Inhaltsverzeichnis	CTRL und I
Bildschirmfarbe ändern	CTRL und F
Rahmenfarbe ändern	CTRL und R
Bildschirmausdruck	CTRL und B
Programm sichern	CTRL und S

Laden von Diskette:
LOAD „INPUT★",8,1
Laden von Kassette:
LOAD oder SHIFT und RUN/STOP

Ausführliche Bedienungshinweise finden Sie auf Seite 29.

Leser fragen . . .

CAD und Seikosha-Drucker

Auf der CeBIT war auch das CAD-Programm Inhalt vieler Gespräche. In diesem Zusammenhang haben wir von einem Anwender eine weitere Druckeranpassung bekommen. Es geht um den Drucker "Seikosha SP 180 VC". Sie müssen die Grundeinstellung MPS 801 vornehmen und die Sekundär-Adresse auf 7 ändern.

Mit dieser kleinen Veränderung sollte auch dieser weit verbreitete Drucker mit dem CAD-Programm zusammen harmonieren.

Steuerbescheid plotten

Das Lohnsteuerprogramm (INPUT 2/87) spielt mit kleinen Änderungen auch zusammen mit dem Commodore-Plotter 1526. Es dreht sich um folgende Zeilen:

```
3290 IF AS = 13 THEN 3325
3325 IF GA = 6 THEN 3345
3344 GOTO 3350
3345 SA=0:OPEN 4,6:OPEN
6,6,6:PRINT #6,1:CLOSE 4: CLOSE 6
```

Dadurch wird der Plotter auf ASCII-Ausgabe und Groß-/Kleinschrift initialisiert. *Bülter, Dörpen.*

Schnell, aber falsch

Der in der letzten Rätselauflösung vorgestellte "schnellste" Algorithmus ist leider von uns im Programm (und damit folgerichtig auch im Listing) nicht korrekt wiedergegeben worden. Hier nun die vom Autor eingesandte lauffähige Version:

```
7000 rem loesung schnell
7010 rem rudolf homey
7020 :
7030 x=int(n/2):ifn>x+xthene(n,n)=-1
7040 ifx=4then7070
7050 z=x+x+1
7060 fori=1tox:e(i+i,i)=-1:e(z-i-1,z-i)=-1:next:return
7070 fori=1tox:p=i+i+2-int(i/4)*8:e(p,i)=-1:e(9-p,9-i)=-1:next:return
7080 :
```

Vokabeltrainer verhaspelt sich

Wenn ich mit Ihrem Vokabeltrainer (INPUT 3/87) in der Datei „englisch all." die 132ste Vokabel ändern will, wechselt die Schriftfarbe und es erscheinen wirre Zeichen auf dem Bildschirm. Fehler im Vokabeltrainer?
(I. Turski, Düsseldorf)

Obwohl mit der veröffentlichten Version des Vokabeltrainers zwei umfangreiche Dateien erstellt wurden, ist der beschriebene Fehler nie aufgetreten. Nichtsdestotrotz hat Herr Turski recht: Bestimmte Zeichenkombinationen „verträgt" das Programm im Verbesserungsmodus nicht. In der ID-Werkstatt dieser Ausgabe befindet sich ein kleines Programm, das diesen Fehler behebt. Speichern Sie dazu dieses Patch-Programm aus dem Magazin heraus auf einen eigenen Datenträger ab. Anschließend laden Sie es von Ihrer eigenen Kassette/Diskette und starten es mit RUN. Als nächstes den Vokabeltrainer laden (nicht starten), im Direktmodus SYS 53000 eingeben und das Vokabelprogramm wieder abspeichern.
(d. Red.)

DOS-Bugs

Zur Abwechslung ein Tip aus der Redaktion, ehe eine entsprechende Leseranfrage vorliegt: Nach den 64er Tips zu „Relativen Dateien" (12/86) wurde versucht, ein kleines Auswertungsprogramm zu schreiben. Ein Record sollte genau 42 Bytes umfassen, der OPEN-Befehl lautete also
OPEN1,8,2,"DATEN,L,"+CHR$(42)

Auf diese syntaktisch einwandfreie Befehl-Sequenz reagierte das DOS mit der Fehlermeldung „Syntax error". Als niemandem etwas Vernünftiges dazu einfiel, tippte jemand eher spaßeshalber ein
OPEN1,8,2,"DATEN,L,"+CHR$(43)
also eine andere Record-Länge. Dies funktionierte. Grund: 42 ist der ASCII-Code für den Stern(*), und der Stern ist als Bestandteil eines File-Namens ungültig! Eine Record-Länge von 63 ist somit auch nicht möglich, da dies der ASCII-Code für das Fragezeichen ist. (d. Red.)

Volkszählung '87:

Bleiben Bürgerdaten anonym?

Eine Simulation auf dem C64

Die 83er Volkszählung wurde bekanntlich vom Bundesverfassungsgericht untersagt. In seinem Urteil machte es für künftige Zählungen unter anderem zur Auflage, daß die gesammelten Daten „anonymisiert" werden müßten. Das bedeutet: Die zusammengefaßten Daten dürfen nur so gespeichert und weitergeleitet werden, daß Rückschlüsse auf eine einzelne, konkrete Person nicht möglich sind. Da diese „Re-Identifizierung" mit dem nötigen technischen Aufwand prinzipiell immer möglich ist, wurde diese Auflage eingegrenzt: Die Daten müssen „faktisch anonym" sein, eine Re-Identifizierung darf nur mit einem unvertretbar hohen Aufwand an Zeit, Kosten und Personal möglich sein.

In seinem Urteil zur Volkszählung '83 machte das Bundesverfassungsgericht zur Auflage, daß die gesammelten Daten anonym sein müssen. Das heißt: Aus den Volkszählungsdaten dürfen keine Rückschlüsse auf einzelne Personen gezogen werden können. Wie groß dieses „Re-Identifizierungs-Risiko" ist, kann mit dem hier vorgestellten Simulations-Programm nachvollzogen werden.

Um diese Frage für die Volkszählung '87 beantworten zu können, wurde am Institut für Informatik der Universität Hamburg ein Modell zur Simulation und Auswertung von 100 000 Personendaten entwickelt. Die von der Informatik-Studentin Simone Fischer-Hübner unter Leitung von Prof. Dr. Klaus Brunnstein — einer der Klageführer gegen die 83er Volkszählung — erstellte Studienarbeit besteht aus zwei Teilen: Zunächst wird auf Basis bekannter Daten der Stadt Hamburg eine statistisch stimmige Datei mit 100 000 Personensätzen erzeugt. Berücksichtigt werden dabei ausgewählte Fragen der Volkszählung zu Geschlecht, Alter, Nationalität, Beruf, Konfession und so weiter.

100 000 künstliche Hanseaten

Die so generierten Daten einer „künstlichen Bevölkerung" können anschließend mit einem verbreiteten Auswertungsprogramm (wen's interessiert: dBASE) ausgewertet werden. Für jede Verknüpfung von Merkmalen (beispielsweise weiblich, 40 Jahre, Lehrerin) wird die Anzahl der Personen ausgegeben, auf die diese Merkmale zutreffen.

Sie können diesen Versuch, der an der Hamburger Universität auf einem Personalcomputer mit angeschlossener Festplatte durchgeführt wurde, auf Ihrem C64/C128 und einer Diskettenstation nachvollziehen. (Die Verarbeitung derartiger Datenmengen mit einer Datasette würde einen bestenfalls an den Rand der Verzweiflung treiben.)

Die Simulation wurde um die Möglichkeit erweitert, auch eine begrenzte Anzahl von Daten von Hand einzugeben, etwa die eigenen oder die seiner Bekannten und Arbeitskollegen. Da die Kapazität einer Diskettenstation mit der einer Festplatte bei weitem nicht mithalten kann, wurde die maximale Anzahl der Personen auf 2222 begrenzt.

Dies ist auf den ersten Blick natürlich längst nicht so beeindruckend wie die Simulation einer 100 000-Personen-Bevölkerung. Wider Erwarten macht diese Einschränkung das Experiment aber sogar realistischer. Denn bekanntlich ist die kleinste Erfassungseinheit der Volkszählung die sogenannte 'Blockseite', das heißt: alle Haushalte einer durch zwei Querstraßen begrenzten Straßenseite. Und daß die Zahl der durch eine Blockseite zusammengefaßten Personen 2000 überschreitet, dürfte allenfalls für Hochhaussiedlungen zutreffen.

Eine Straßenseite im 64er

Um das Experiment nachzuvollziehen, speichern Sie bitte beide Programme aus dem Magazin heraus auf eine eigene Diskette ab. Das eine generiert eine Datei mit cirka 2000 Personen, mit dem anderen können Sie diese Datei erweitern oder auswerten.

Um die Simulation auf das Wesentliche zu beschränken, wurden nicht alle Daten, die der Personenbogen der Volkszählung '87 abfragt, in das Experiment übernommen. Wir haben uns, ebenso wie die Hamburger Wissenschaftler, beschränkt auf die Fragen nach

- Geburtsjahr (1a)
- Geburtsmonat (1b)
- Geschlecht (2)
- Familienstand (3)
- Religion (4)
- Staatsangehörigkeit (5)
- Erwerbstätigkeit (7)
- Schulabschluß (9)
- Ausbildungsabschluß (10a)
- Schule (12)
- Wirtschaftszweig (16)
- Beruf (17)

Die eingeklammerten Zahlen bezeichnen jeweils die entsprechenden Fragen im amtlichen Personenbogen. Wir gehen davon aus, daß die fächendekkende Werbung der Statistischen Ämter mit dem Abdruck eines Musterfragebogens auch Sie erreicht hat oder daß Sie den Fragebogen schon bekommen haben. Die Frage 12 (Schule) wird nur bei Schülern und Studenten berücksichtigt. Der Einfachheit halber wurde auch nur der Schultyp und nicht, wie im Volkszählungs-Fragebogen, die Anschrift der Schule aufgenommen.

Dafür werden einige Merkmale verarbeitet, die nicht direkt den Fragen des Wohnungs- oder Personenfragebogens zuzuordnen sind, nämlich

- Einzelhaushalt
- Gebäudenummer
- Haushaltsnummer

Da der Bezugspunkt des Experiments die langfristig abgespeicherten Daten sind, erwies sich diese Erweiterung als schlüssig. Paragraph 15 Absatz 3 des Volkszählungsgesetzes schreibt nämlich vor:

„Die laufenden Nummern und die Ordnungsnummern nach § 4 sind zu löschen, sobald die Zusammenhänge zwischen Personen und Haushalt, Haushalt und Wohnung, Wohnung und Gebäude . . . festgehalten worden sind . . ."

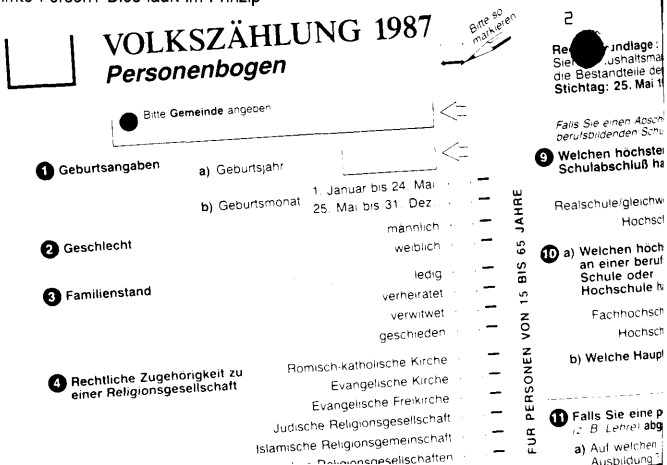
Das heißt: Innerhalb der Erfassungseinheit „Blockseite" werden außerdem Ordnungsmerkmale abgelegt, die eine Zuordnung der Daten nach Haushalten und Gebäuden ermöglichen. Wie das die Statistischen Bundesämter im einzelnen machen, ist bislang unbekannt, da die Programme zur Verarbeitung der Daten noch nicht existieren. Wir haben diese Vorschrift so umgesetzt, daß jeder Personensatz einen Hinweis enthält, ob es sich um einen Einzelhaushalt handelt, und eine Gebäude- und eine Haushaltsnummer. So sind die Daten eines Gebäudes und/oder eines Haushalts zusammenführbar.

Vor dem Isolieren . . .

Wie re-identifiziert man nun eine bestimmte Person? Dies läuft im Prinzip nach folgender Methode ab: Sie suchen beispielsweise eine 35jährige, ledige Lehrerin. Es werden also aus der vorhandenen Datei alle weiblichen Personen herausgesucht, die Jahrgang 1952 und ledig sind. Bei der Erfassungseinheit „Blockseite" trifft dies vielleicht auf zwei Personen zu; um die gewünschte Person isolieren zu können, müßten deswegen die gefundenen Daten noch einmal nach einem weiteren Merkmal durchsucht werden, beispielsweise nach der Konfession. Mit etwas Glück ist dann schon der gesuchte Datensatz isoliert.

Damit stehen auch sämtliche weitere Daten dieser Person zur Verfügung, aber zunächst ohne Namen und Anschrift. In der aktuellen Diskussion wird allerdings aus gutem Grund die beschriebene Isolierung eines Datensatzes mit der Re-Identifizierung einer Person gleichgesetzt. Es ist nämlich kein Problem, innerhalb einer Straßenseite den Namen und die genaue Anschrift der gesuchten Person zu recherchieren.1)

Das Programm „Generierung" erzeugt, wie der Name schon sagt, eine „künstliche Bevölkerung" von circa 2000 Personen. Die Anzahl der erzeugten Personen schwankt, da die statistischen Kriterien über die Auswertung des Zufallszahlen-Generators nachgebildet werden, und — so der Zufall es will — die 2000ste Person noch fünf Mitbewohner haben

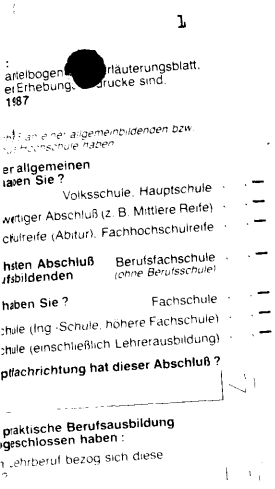


kann. Legen Sie, nachdem Sie dieses Programm vom eigenen Datenträger geladen und mit RUN gestartet haben, eine leere, formatierte Diskette in das Laufwerk und starten dann mit der Leer-Taste.

. . . ein bißchen generieren

Es wird eine relative Datei namens „PERSDAT" mit einer Länge von circa 240 Blöcken erzeugt. Das Programm benötigt dazu etwa 45 Minuten, also etwas Geduld! Ist die künstliche Bevölkerung komplett, werden Sie zum Ausschalten des Rechners aufgefordert, es gibt kein Programmende per Tastendruck oder ähnlichem.

Mit dem Programm „Auswertung" können Sie die Datei „PERSDAT" mit eigenen Daten erweitern und/oder die vorhandenen Daten auswerten. Dazu bedarf es einer kurzen Beschäftigung mit der Struktur der auf Diskette abgelegten Daten: Jeder Datensatz, das heißt jeder Record, besteht aus 30 Zeichen und einem CHR$(13) als Endemarkierung. Die ersten vier Bytes davon dienen der internen Identifikation. Die Zuordnung der restlichen 26 Zeichen entspricht auch den jeweils zulässigen Angaben für die Auswertung beziehungsweise Neueingabe von Daten (die folgende Beschreibung ist weitgehend aus der Arbeit von Frau Fischer-Hübner zitiert):


1

:
arteibogen [redacted] rläuterungsblatt.
er Erhebung[redacted] rucke sind.
1987

...ge einer allgemeinbildenden bzw.
...Hochschule haben

er allgemeinen
aben Sie ?
Volksschule, Hauptschule
vertiger Abschluß (z. B. Mittlere Reife)
chulreife (Abitur), Fachhochschulreife

hsten Abschluß Berufsfachschule
ufsbildenden (ohne Berufsschule)

haben Sie ? Fachschule
chule (Ing.-Schule, höhere Fachschule)
chule (einschließlich Lehrerausbildung)
pfachrichtung hat dieser Abschluß ?

praktische Berufsausbildung
geschlossen haben :
Lehrberuf bezog sich diese

Das **Geburtsjahr** wird durch die Zeichen-Darstellung von „00" bis „87" abgelegt, was den Jahrgängen von 1900 bis 1987 entspricht. Die wenigen Personen, die im vorigen Jahrhundert geboren sind, wurden der Einfachheit halber ausgeklammert.

Mit **Geburtsmonat** ist genaugenommen die Jahreshälfte gemeint. Entsprechend der Abfrage im Volkszählungs-Bogen nimmt dieser Wert eine „1" an, wenn die betreffende Person in den ersten sechs Monaten des Jahres geboren ist, eine "2" entspricht der zweiten Jahreshälfte. Die Differenz von etwa einem Monat zum Stichtag der Volkszählung ist problemlos zu vernachlässigen.

Das **Geschlecht** ist erwartungsgemäß „m" für männlich und "w" für weiblich.

Die **Staatsangehörigkeit** wird kodiert durch „d" für deutsch, „t" für türkisch, „g" für griechisch, „i" für italienisch, „j" für jugoslawisch, „e" für andere EG-Staaten und „s" für sonstige oder keine Staatsangehörigkeit.

Für **Familienstand** kommen „l" für ledig, „h" für verheiratet, „s" für geschieden und „w" für verwitwet in Frage.

Das Merkmal **Religion** wird mit „e" für evangelisch, „f" für freikirchlich, „k" für katholisch, „i" für moslemisch, „j" für jüdisch, „o" für ohne Religion und „s" für sonstige Religionen belegt.

Als **Schulabschluß** wird „a" für Abitur/Fachhochschulreife, „r" für Realschule (Mittlere Reife), „v" für Volksschulabschluß und "o" für Ohne Abschluß zugewiesen.

Nur bei Schülern und Studenten wird der **Schule** ein Wert zugewiesen: „k" für Kindergarten/Vorschule, „g" für Grundschule, „v" für Volksschule, „r" für Realschule, „G" für Gymnasium, „b" für Berufsfachschule, „f" für Fachschule, „h" für Fachhochschule und „u" für Universität.

Die Komponente **Ausbildungsabschluß** kann die Werte „b" für Berufsfachschulabschluß, „f" für Fachschulabschluß, „h" für Fachhochschulabschluß und „u" für Universitätsstudium annehmen. Ein „o" kennzeichnet Personen, die keinen derartigen Abschluß haben.

Die **Beschäftigung** wird kodiert durch „e" für erwerbstätig, „n" für nicht erwerbstätig, „a" für arbeitslos, „s" für Schüler und „S" für Studenten. Sie wird für Kinder, die noch nicht zur Schule gehen, mit einem „k" belegt.

Das Feld **Beruf** wird bei Erwerbstätigen mit einem zweistelligen Schlüssel für die Berufsgruppen, ebenfalls nach der Systematik des Mikrozensus, belegt. Die Entschlüsselung finden Sie in den Tabellen.

Der **Wirtschaftszweig** wird durch einen Code von „0" bis „9" dargestellt. Die Aufteilung ist ebenfalls den Tabellen zu entnehmen.

Die **Haushaltsnummer** erhält die vierstellige Zahlendarstellung des Haushaltes, in der die entsprechende Person lebt. Im Programmverlauf werden Werte ab „0000" erzeugt.

Die **Gebäudenummer** ist aus programmtechnischen Gründen intern durch eine vierstellige Zahl im Klartext abgelegt. Da aber die Zahl der Gebäude bei 2000 Personen 999 nie übersteigt, können bei der Auswertung/Neueingabe nur maximal dreistellige Zahlen eingegeben werden. Dazu Näheres später.

Die Komponente **Einzelhaushalt** wird mit „j" belegt, falls die Person einen Einzelhaushalt führt, ansonsten mit „n".

Wer sich etwas mit der Darstellung von Daten auskennt, wird feststellen, daß enorm großzügig mit dem verfügbaren Speicherplatz umgegangen

Ein Homecomputer reicht aus

Nach der Auswertung seines Modellversuchs zur Re-Identifikation von Daten mit einer künstlichen Bevölkerung von 100 000 Personen stellte das Hamburger Institut fest: „Es gibt kaum eine Person, die nicht mit etwa 10 Merkmalen eindeutig re-identifiziert werden kann." Damit sei es „höchst zweifelhaft, ob die vom Bundesverfassungsgericht geforderte faktische Anonymität der Volkszählungsdaten tatsächlich gewährleistet ist." (aus dem Vorwort Prof. Brunnsteins zur Arbeit von Frau Fischer-Hübner)

Unsere Ergebnisse auf dem C64 bestätigen die Aussagen der Hamburger Wissenschaftler, wahrscheinlich werden Sie beim Experimentieren dasselbe feststellen.

Der finanzielle Aufwand für den Versuch an der hanseatischen Universität betrug immerhin noch um die 10 000 DM — den Preis für einen IBM-AT-kompatiblen Rechner plus Festplatte und dem Auswertungsprogramm.

Eine geeignete Rechnerkonfiguration (wie der C64 mit Diskettenstation) zur Auswertung der Daten einer Blockseite ist heute für deutlich unter 1000 DM zu haben — auf der Kostenseite somit kaum der von den Verfassungsrichtern geforderte 'unverhältnismäßig hohe Aufwand', der mit der Re-Identifizierung von Bürger-Daten zwingend verbunden sein sollte.

Offen bleibt dabei die Frage, ob jemand diese Re-Identifizierung mit den echten Daten der Volkszählung tatsächlich vornimmt — sie ist im Volkszählungsgesetz ausdrücklich untersagt und mit einer Freiheitsstrafe von bis zu einem Jahr belegt. Die Überlegung, ob und wie weit sich die Behörden und ihre Mitarbeiter streng nach dem Gesetz verhalten, wird jeder, abhängig vom eigenen politischen Standort, unterschiedlich beurteilen.

Offen bleibt auch die Frage, wie sicher diese Daten vor unbefugtem Zugriff bei den Statistischen Bundesämtern lagern. In der Regel ist auf diese Daten kein On-line-Zugriff möglich, es bedürfte daher erheblicher krimineller Energie, um an diese Daten zu gelangen. Das technische Problem, die meist auf Bändern gespeicherten Daten zu lesen und auszuwerten, ist dagegen relativ leicht lösbar.

Und es bleibt noch eine Reihe anderer Fragen offen. Die nach der Notwendigkeit von Totalerfassungen, die „Was habe ich denn zu verbergen?"-Frage und und und. Die Antworten darauf sind wiederum nur auf der politischen, nicht auf der technischen Ebene zu finden und seien darum dem Leser überlassen. Nicht nur, weil in der Redaktion die Meinungen zur Volkszählung genauso unterschiedlich sind wie auch sonst in der Bevölkerung. Sondern auch, weil wir — wie schon im Editorial (Seite 1) erwähnt — uns als Computer-Magazin auf technische Probleme beziehen. Wie die Frage, ob die höchstrichterlich geforderte „faktische Anonymität" der Volkszählungsdaten überhaupt gewährleistet werden kann. Die Antwort darauf ist nicht offen. Sie heißt „Nein".

JS

wurde. Dies ist Absicht; mit entsprechenden Tricks ließe sich die Dateilänge nämlich halbieren, und alle abgefragten Daten des Personenbogens hätten Platz auf Diskette.

Vom groben Sieb . . .

Nach dem Laden des Auswertungsprogramms vom eigenen Datenträger und dem Start mit RUN prüft dieses zunächst, ob sich die Datei namens „PERSDAT" auf Diskette befindet. Ist dies nicht der Fall, erscheint eine entsprechende Fehlermeldung. Legen Sie also vor dem Start die richtige Diskette ein.

Wenn Sie im Anfangsmenü des Auswertungsprogramms den Unterpunkt „Auswerten" wählen, können Sie in der Eingabemaske die jeweils zulässige Kodierung für die einzelnen Daten eingeben. Unzulässige Zeichen werden nicht angenommen. Keine Eingabe bedeutet, daß dieses Kriterium beim Durchsuchen der Daten nicht berücksichtigt werden soll. Alle Eingaben werden mit RETURN abgeschlossen.

Beispiel: Es soll ein 40jähriger, lediger, alleinlebender Mann gesucht werden.

Sie geben als Jahrgang 46 und 47 ein, dies ist zu verstehen als zwischen 1946 und 1947 geboren. (Wenn Sie genauere Informationen über seinen Geburtstag hätten, könnten Sie auch im zweiten Jahrgangsfeld nur RETURN drücken.)

Da der Geburtsmonat unbekannt ist, wird dieses Feld mit RETURN übergangen. Als Geschlecht geben Sie „m" ein, beim Familienstand „l".

Die nächsten Felder werden wieder mit RETURN übergangen und sind somit nicht Suchkriterium. Wenn der Cursor hinter „Einzelhaushalt" blinkt, übergeben Sie mit „j" an die Such-

maske, daß die gesuchte Person alleinlebend ist. Die Abfragen nach Gebäudenummer und Haushaltsnummer werden nur mit RETURN beantwortet.

Im ersten Durchlauf werden alle Daten nach den eingegebenen Merkmalen durchsucht, Sie müssen sich deswegen circa 10 Minuten gedulden. Sind alle Datensätze verglichen, können Sie entscheiden, ob

- alle gefundenen Datensätze komplett ausgegeben werden sollen
- eine neue Suchmaske eingegeben werden soll
- die Suchmaske erweitert werden soll

. . . zum feinen Raster

Nehmen wir einmal an, wir hätten bei unserem Beispiel 20 Personen gefunden, auf die die Merkmale Ledig, 35 Jahre alt, Einzelhaushalt und Männlich zutreffen. Das macht den Rückschluß auf eine einzelne Person noch zu unbequem. Sie erinnern sich aber, daß die gesuchte Person katholisch ist. (Das ist eine einfache Eingrenzung, da die generierten Daten auf Hamburger Statistiken beruhen und dort Katholiken bekanntlich in der Minderheit sind.) Wählen Sie dann aus „Suchmaske erweitern", und es können entsprechend der oben beschriebenen Vorgehensweise weitere Kriterien eingegeben werden. In unserem Beispiel wäre das ein „k" für Religion; bereits im ersten Durchgang eingegebene Merkmale können jetzt logischerweise nicht mehr geändert werden und werden übersprungen.

Das zweite Durchsuchen der Datei geht erheblich schneller, da nur die schon gefundenen Datensätze mit der neuen Suchmaske verglichen werden. Nach diesem zweiten Pass stehen Sie vor der gleichen Entscheidung wie vor dem ersten Durchlauf. Wenn nur noch eine Person herausgefiltert wurde, können Sie sich deren komplette Daten ansehen. (Das wäre bei den echten Volkszählungsdaten natürlich interessanter, da es dort einerseits um reale Personen, andererseits um mehr Daten pro Person geht.) Ist Ihnen die Zahl der so isolierten Personen noch zu groß, könnten Sie versuchen, durch ein nochmaliges Erweitern der Suchmaske, etwa durch die Staatsangehörigkeit, den betreffenden Personenkreis weiter einzugrenzen.

Sie werden selbst feststellen, wie schwer oder wie leicht die Re-Identifizierung bestimmter Personen ist. Die Anzahl der erforderlichen Merkmale hängt sehr von der statistischen Häufigkeit dieser Merkmale ab. So ist wahrscheinlich eine Ingenieurin schon durch die Merkmale Geschlecht und Beruf innerhalb einer Blockseite identifizierbar, ein evangelischer, verheirateter Metallarbeiter eventuell nur über weitere als diese vier Merkmale.

Um Mißverständnissen vorzubeugen: Es ist nicht gesagt, daß in der bei Ihnen zu Hause erzeugten Datei die erwähnten Beispielpersonen existieren. Jede neu erzeugte Datei „PERSDAT" ist unterschiedlich, nur die statistische Verteilung der Merkmale ist dieselbe!

Ist eine Person identifiziert, kann über die Haushaltsnummer auch auf die Daten eventueller Mitbewohner zugegriffen werden; über die Gebäudenummer ebenso auf die Daten aller Einwohner desselben Hauses. Um Ihnen unnütze Suche nach nicht vorhandenen Gebäuden zu ersparen, kann in diesen beiden Felder nur die maximale Anzahl der in der Datei vorhandenen Haushalte beziehungsweise Gebäude eingetragen werden.

In der Masse untertauchen

So richtig interessant wird's natürlich erst, wenn man überprüft, wie schnell die eigenen persönlichen Daten isoliert werden können. Wählen Sie den entsprechenden Menüpunkt, und geben Sie Ihre persönlichen Daten, die ihrer Familie, Freunde oder Arbeitskollegen ein. Sie müssen alle Fragen bis zu dem Punkt „Beschäftigung" beantworten; ob Sie auch bei den weiteren Punkten Angaben machen, ist Ihnen freigestellt. Die Eingaben werden auch nicht auf Plausibilität geprüft.

Die Eingabe von Mehrpersonen-Haushalten (über gleiche Haushaltsnummern und Einzelhaushalt gleich „n") und die Zuordnung von Haushalten zu Gebäuden liegt in der Hand des Benutzers, dabei gelten folgende Vereinbarungen: Die niedrigste zulässige Haushaltsnummer ist die höchste in der Datei bereits vorhandene, Entsprechendes gilt für die Gebäude. Wird keine Haushaltsnummer angegeben, wird ein neuer Haushalt erzeugt. Wird keine Gebäudenummer angegeben, wird nach jedem neunten Haushalt ein neues Gebäude generiert.

Alle anderen Felder, in denen keine Angaben fehlen, werden mit der entsprechenden Anzahl von Leerzeichen belegt. Dies ist zumindest der Stand bei Drucklegung dieses Heftes. Eventuelle Verbesserungen werden Ihnen gegebenenfalls vom Programm mitgeteilt. JS

1) Entweder durch Zugriff auf andere Dateien, oder notfalls durch Recherchen vor Ort.

Tabellen, Tabellen . . .

Die Schlüssel der Wirtschaftszweige

Code	Wirtschaftszweig
0	Land- und Forstwirtschaft, *Tierhaltung, Fischerei*
1	Energiewirtschaft, Wasserversorgung, Bergbau
2	Verarbeitendes Gewerbe (ohne Baugewerbe)
3	Baugewerbe
4	Handel
5	Verkehr und Nachrichtenübermittlung
6	Kreditinstitute und Versicherungsgewerbe
7	Andere Diensleistungen
8	Organisationen ohne Erwerbscharakter, Privathaushalte
9	Gebietskörperschaften, Sozialversicherung

Kodierung der Berufsgruppen

Berufsgruppe	Berufsbezeichnung
01	Landwirte
02	Tierzüchter,Fischereiberufe
03	Verwalter,Berater in Landwirtschaft u. Tierzucht
04	Landwirtschaftliche Arbeits kräfte,Tierpfleger
05	Gartenbauer
06	Forst-,Jagdberufe
07	Bergleute
08	Mineral,-Erdöl-, Erdgasgewinner
09	Mineralaufbereiter
10	Steinbearbeiter
11	Baustoffhersteller
12	Keramiker
13	Glasmacher
14	Chemiearbeiter
15	Kunststoffhersteller
16	Papierhersteller,-verarbeiter
17	Drucker
18	Holzaufbereiter,Holzwaren Fertiger
19	Metallerzeuger,Walzer
20	Former,Formgiesser
21	Metallverformer(spanlos)
22	Metallverformer(spanend)
23	Metalloberflächenbearbeiter, -vergüter,-beschichter
24	Metallverbinder
25	Schmiede
26	Feinblechner,Installateure
27	Schlosser
28	Mechaniker
29	Werkzeugmacher
30	Metallfeinbauer und zugeordnete Berufe
31	Elektriker
32	Montierer und Metallberufe,a.n.g.
33	Spinnberufe
34	Textilhersteller
35	Textilverarbeiter
36	Textilveredler
37	Lederhersteller,Leder- und Fellverarbeiter
39	Back-, Konditorwarenhersteller
40	Fleisch-, Fischverarbeiter
41	Fleischer
42	Getränke-,Genussmittelher steller
43	Übrige Ernährungsberufe
44	Maurer,Betonarbeiter
45	Zimmerer,Dachdecker, Gerüstbauer
46	Strassen-,Tiefbauer
47	Bauhilfsarbeiter
48	Bauausstatter
49	Tischler,Modellbauer
50	Tischler
51	Maler,Lackierer und verwandte Berufe
52	Warenprüfer,Versandfertigmacher
53	Hilfsarbeiter ohne nähere Tätigkeitsangabe
54	Maschinisten und zugehörige Berufe
60	Ingenieure
61	Chemiker,Physiker, Mathematiker
62	Techniker
63	*Technische Sonderfachkräfte*
68	Warenkaufleute
69	*Bank-,Versicherungskaufleute*
70	Andere Dienstleistungskaufleute
71	*Berufe des Landverkehrs*
72	Berufe des Wasser- und Luftverkehrs
73	Berufe des Nachrichtenverkehrs
74	Lagerverwalter,Lager-,Transportarbeiter
75	Unternehmer,Organisatoren,Wirtschaftsprüfer
76	Abgeordnete, administrativ entscheidene Berufstätige
77	Rechnungskaufleute, Datenverarbeitungsfachleute
78	Bürofach-, Bürohilfskräfte
79	Dienst-, Wachberufe
80	Sicherheitswahrer
81	Rechtswahrer
82	Publizisten,Dolmetscher,Bibliothekare
83	Künstler und zugeordnete Berufe
84	Ärzte,Apotheker
85	Übrige Gesundheitsberufe
86	Sozialpflegerische Berufe
87	Lehrer
88	Geistes- und naturwissenschaftliche Berufe,a.n.g.
89	Seelsorger
90	*Körperpfleger*
91	Gästebetreuer
92	*Hauswirtschaftliche Berufe*
93	Reinigungsberufe
97	Mithelfende Familienangehörige ausserhalb der Landwirtschaft
98	Arbeitskräfte mit noch nicht bestimmten Beruf
99	Arbeitskräfte ohne nähere Tätigkeitsangabe

Fragen, zulässige Eingaben, Bedeutung

Frage	Kodierung	Bedeutung
Geburtsjahr	00 - 87	1900 bis 1987
Geburtsmonat	1 / 2	1./2. Jahreshälfte
Geschlecht	m / w	männlich/weiblich
Staatsangehörigkeit	d	deutsch
	t	türkisch
	g	griechisch
	i	italienisch
	j	jugoslawisch
	e	andere EG-Staaten
	s	sonstige/keine
Familienstand	l	ledig
	h	verheiratet
	s	geschieden
	w	verwitwet
Religion	e	evangelisch
	f	freikirchlich
	k	katholisch
	i	moslemisch
	j	jüdisch
	o	ohne Religion
	s	sonstige
Schulabschluß	a	Abitur/Fachhochschulreife
	r	Realschule (Mittlere Reife)
	v	Volksschulabschluß
	o	Ohne Abschluß
Schule	k	Kindergarten/Vorschule
	g	Grundschule
	v	Volksschule
	r	Realschule
	G	Gymnasium
	b	Berufsfachschule
	f	Fachschule
	h	Fachhochschule
	u	Universität
Ausbildungsabschluß	b	Berufsfachschulabschluß
	f	Fachschulabschluß
	h	Fachhochschulabschluß
	u	Universitätsstudium
	o	ohne Abschluß
Beschäftigung	e	erwerbstätig
	n	nicht erwerbstätig
	a	arbeitslos
	s	Schüler
	S	Studenten
	k	Kleinkinder
Beruf	00 - 99	Kodierung der Berufsgruppen, siehe Tabelle
Wirtschaftszweig	0 - 9	Schlüssel der Wirtschaftszweige
Haushaltsnummer	0000-9999	fortlaufende Haushaltsnummer
Gebäudenummer	000-999	fortlaufende Gebäudenummer, intern vierstellig!
Einzelhaushalt	j/n	j: Einzelhaushalt n: kein Einzelhaushalt

In die Tiefe

INPUT 64-Assemblerschule, Teil 3

In der letzten Folge wurde ja schon der JSR-Befehl vorgestellt. Wir haben gesagt, daß die Abkürzung von 'Jump to SubRoutine' (Springe in ein Unterprogramm) kommt. Hin und wieder liest man auch 'Jump Saving Return-adress' (Springe und rette Rückkehr-adresse). Diese Interpretation werden Sie verstehen, wenn Sie die genaue Funktionsweise kennengelernt haben. Allerdings müssen wir dazu etwas weiter ausholen.

Der Befehl JSR benutzt den Prozessor-Stack. 'Stack' ist mal wieder ein englisches Wort, übersetzt heißt es Stapel. Dabei handelt es sich um einen besonderen, für diesen Zweck reservierten Speicherbereich. Er liegt bei 6502-Computern an den Adressen $100 bis $1FF, also in der Speicherseite 1.

Man kann sich den Stack anschaulich wie einen Stapel aus Notizzetteln vorstellen: Sie können auf einen solchen Stapel Zettel oben drauflegen oder von oben jeweils einen wegnehmen. Lesen kann man immer nur den obersten.

In dieser Lektion unseres Maschinensprache-Lehrgangs werden wir uns im doppelten Sinne in neue Tiefen wagen. Zum einen werden Sie den Kellerspeicher kennenlernen. Zum anderen steigen wir von der Ebene der Bytes noch eine Stufe tiefer und schauen uns Befehle an, mit denen einzelne Bits manipuliert werden können. Wenn Sie durch die Beschäftigung mit der Maschinensprache bereits sechzehn Finger haben, werden Sie es vielleicht auch als Abstieg empfinden, daß wir gegen Ende dieses Kapitels wieder im Dezimalsystem rechnen.

Tiefstapler

Der Prozessor-Stack beginnt an der Adresse $1FF und 'wächst' abwärts. Die CPU besitzt zur Verwaltung des Stapels ein eigenes Register, den Stackpointer (Stapelzeiger). In unserem Simulator wird sein Inhalt in der Spalte unter 'SP' angezeigt. Der Stackpointer zeigt immer auf die nächste freie Speicherstelle im Stack. Diesen Mechanismus zeigt Bild 1: Die ersten vier Einträge im Stapel ($1FF bis $1FC) enthalten bereits Informationen. Legt ein Programm nun etwas auf dem Stack ab, so wird die Speicherstelle $1FB beschrieben und gleichzeitig der Stackpointer um Eins vermindert. Beim Lesen aus dem Stack erhöht der Prozessor den Inhalt des Stackpointers um Eins und liest die Speicherstelle $1FC.

Wenn der Prozessor nun auf einen JSR-Befehl trifft, so schreibt er den augenblicklichen Inhalt des Programmzählers auf den Stack, zuerst das höherwertige und dann das niederwertige Byte. Auf diese Art 'merkt' er sich die Adresse des Unterprogrammaufrufes. Dann springt er zu der im Befehl angegebenen Adresse. (Aus Prozessor-internen Gründen wird übrigens die Adresse des dritten Bytes des JSR-Befehles abgespeichert.)

Am Ende eines jeden Unterprogrammes steht der Befehl

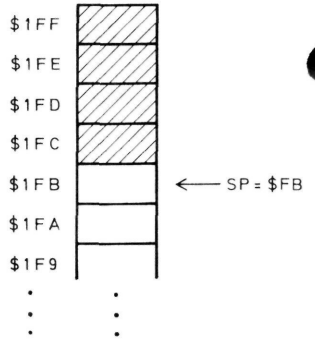
RTS

(ReTurn from Subroutine, zu deutsch Rückkehr aus Unterprogramm). Er wird folgendermaßen abgearbeitet: Die CPU holt sich die oberen beiden Bytes vom Stack und lädt sie in den Programmzähler. Der wird dann um Eins erhöht, und an dieser Adresse setzt der Prozessor seine Arbeit fort. An dieser Stelle steht ja genau der nächste Befehl nach dem JSR-Kommando.

Stapellauf

Auf diese Weise kann sich der Prozessor bis zu 128 ineinander verschachtelte Unterprogrammaufrufe merken (Je-

Bild 1: Der Prozessor-Stack wächst abwärts. Die schraffierten Kästchen stellen bereits belegte Speicherstellen dar.



der belegt zwei Bytes, der Stack kann 256 Bytes aufnehmen.). Das erste Beispielprogramm auf Ihrem Datenträger verdeutlicht diesen Mechanismus.

Es beginnt mit einem Sprung in das Unterprogramm INPT, das die Aufgabe hat, die Tastatur abzufragen, die Anzahl der Tastendrücke zu zählen und bei Eingabe der RETURN-Taste abzubrechen. Dazu benutzt es die Routine WAIT. Diese aktiviert in einer Schleife die Betriebssystem-Routine GETC so lange, bis eine Taste gedrückt wird. WAIT gibt den Wert der betätigten Taste im Akku zurück. INPT prüft dann, ob es die RETURN-Taste war. In diesem Falle bricht sie ab und gibt die Anzahl der gedrückten Tasten im X-Register an das aufrufende Programm zurück. Anderenfalls wird das X-Register inkrementiert und die Tastatur erneut abgefragt. Im Hauptprogramm wird nach Beendigung des Unterprogramms INPT die Anzahl der Tasten in der Speicherstelle ZAHL gespeichert und vor dem Programmende noch ein erneuter Tastendruck abgewartet.

Wenn Sie das Programm starten, achten Sie auf den jeweiligen Inhalt des Stackpointers. Mit der Minus-Taste können Sie sich auch den Stack ansehen. Interessant sind hier die Adressen $1F8 bis $1FF. Wie Sie sehen, wird bei der Ausführung des RTS-Befehles im Stack selber nichts geändert. Durch die Inkrementierung des Stackpointers ist die Rückkehradresse jedoch aus der Sicht des Prozessors aus dem Stack entfernt.

Bestimmt haben Sie die Bedeutung des zweiten Befehles im Beispielprogramm schon erraten, er sei hier aber der Vollständigkeit halber noch erklärt.

STX

heißt STore X-register und speichert den Inhalt des Index-X-Registers an der angegebenen Speicheradresse ab. Wie Sie völlig richtig vermuten, gibt es diesen Befehl auch für das Y-Register, er heißt dann

STY

Schiebung

Wo wir schon mal dabei sind, neue Befehle zu lernen, können wir eigentlich auch gleich damit weitermachen. Die nächsten vier Instruktionen bilden die Gruppe der Schiebe- und Rotationsbefehle. Ihre Funktionsweise ist in Bild 2 dargestellt. Links sehen Sie die beiden Schiebebefehle. Sie heißen

ASL (Arithmetic Shift Left)
und
LSR (Logical Shift Right).

Sie bewirken eine Verschiebung des angesprochenen Bytes um eine Bitstelle nach links (ASL) beziehungsweise nach rechts (LSR). Das dabei freiwerdende Bit wird mit einer Null aufgefüllt, das herausgeschobene Bit landet in der Carry-Flagge. Neben der Carry werden je nach Ergebnis noch die Zero- und die Negativ-Flagge beeinflußt.

Mit den Schiebebefehlen kann man — je nach Adressierungsart — entweder den Akku-Inhalt verändern oder auf eine Speicherstelle zugreifen. Ohne Argument (Adressierungsart Implied) fühlt sich der Akku angesprochen, ansonsten die angegebene Adresse. (Allerdings verlangen einige Assembler die Schreibweise 'ASL A', wenn die Implied-Version gemeint ist.)

Diese Befehle werden sehr oft benötigt, wenn irgendwelche Bytes bitweise verarbeitet werden sollen. Außerdem dienen sie zur Multiplikation und Division. Wenn man im Dezimalsystem eine Zahl um eine Stelle nach links verschiebt, ergibt das eine Multiplikation mit Zehn. Genauso entspricht die Verschiebung einer Binärzahl um ein Bit nach links einer Multiplikation mit Zwei. Analog ergibt das Verschieben nach rechts eine Division.

Bitkarussell

Die beiden anderen Operationen dieser Vierergruppe sind die sogenannten Rotationsbefehle. Sie sind den Schiebebefehlen sehr ähnlich. Die Abkürzungen für die Rotationsbefehle sind

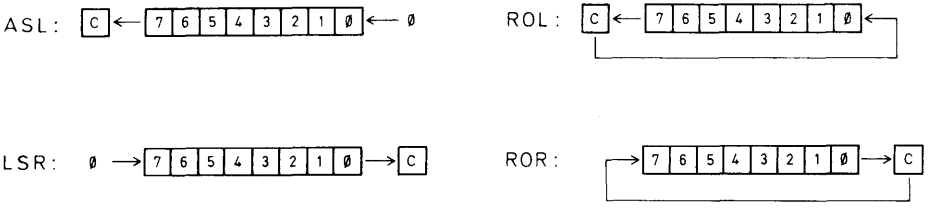


Bild 2: Die Schiebe- und Rotationsbefehle benutzen die Carry-Flagge als achtes Bit.

ROL (ROtate Left)
und
ROR (ROtate Right).

Auch bei ihnen wird das jeweilige Byte um ein Bit nach links (ROL) oder nach rechts (ROR) verschoben, wobei das überschüssige Bit in die Carry gelangt. Am anderen Ende wird jedoch nicht eine Null nachgeschoben sondern der Inhalt, den die Carry vor der Ausführung des Befehles hat.

Für die Rotationsbefehle gilt in Bezug auf die Beeinflussung der anderen Status-Flaggen und die Adressierungsarten das gleiche wie bei den Schiebebefehlen.

Diese Operationen werden benötigt, wenn der Inhalt der Carry-Flagge von Bedeutung ist. Zum Beispiel kann man eine Zwei-Byte-Zahl mit der Befehlsfolge

ASL Low-Byte

ROL High-Byte

verdoppeln. Das höchste Bit des Low-Bytes landet dabei in Bit 0 des High-Bytes.

Logisch!

Die nächste Gruppe von Befehlen, die Sie kennenlernen sollen, stellen drei verschiedene Operationen zur Verfügung. An ihnen ist — ähnlich wie bei den Befehlen ADC und SBC — der Akku und ein anderes Byte beteiligt. Alle drei kennen dieselben Adressierungsarten wie ADC und SBC, und sie beeinflussen die Negativ- und die Zero-Flagge. Es handelt sich um die logischen Verknüpfungen von Binärzahlen.

Der 6502-Prozessor kennt Maschinenbefehle für die drei Logik-Funktionen Und, Oder und Exklusiv-Oder. Die Wertetabellen der drei Verknüpfungen sind in Tabelle 1 enthalten.

A	B	A AND B	A OR B	A EXOR B
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Tabelle 1: Die Funktionsweise der logischen Operationen erkennt man aus ihren Wahrheitstafeln.

Der zur Und-Verknüpfung gehörende Maschinensprache Befehl heißt

AND (AND with akku)

Mit ihm werden der Akku und das Argument bitweise gemäß der abgebildeten Wertetabelle verknüpft. Das Ergebnis steht danach im Akku.

Mit diesem Befehl lassen sich gezielt Bits im Akku löschen. Alle Bits, die im Argument gleich Null sind, sind es auch im Ergebnis. Ein im Argument gesetztes Bit beeinflußt das entsprechende Akku-Bit nicht. Beispielsweise löscht der Befehl 'AND #$0F' das obere Nibble im Akku. Das niederwertige Nibble enthält nach der Ausführung immer noch seinen alten Wert. Diesen Vorgang nennt man Maskieren, das Byte $0F stellt eine Bit-Maske dar.

Mit dem Befehl

ORA (OR with Akku)

wird eine bitweise Oder-Verknüpfung zwischen dem Akku und dem Argument durchgeführt, dessen Ergebnis wieder im Akku steht. Mit ihm werden einzelne Bits im Akku gesetzt. Alle Eins-Bits im Argument werden als solche in den Akku übernommen.

Um eine bitweise Exklusiv-Oder-Verknüpfung zwischen dem Akku und einem Argument durchzuführen, benutzt man den Befehl

EOR (Exclusive OR with akku)

Auch bei ihm nimmt der Akku das Ergebnis auf. Alle Akku-Bits, die im Argument gesetzt sind, werden durch diesen Befehl invertiert. Mit 'EOR #$FF' kann man so auf einfache Art und Weise das Einerkomplement der im Akku stehenden Zahl berechnen lassen.

Neben dem gezielten Manipulieren einzelner Bits im Akku werden die logischen Verknüpfungen auch zum Testen von Zahlen verwendet. So ist nach einem EOR-Befehl die Z-Flagge genau dann gesetzt, wenn Akku-Inhalt und Argument gleich waren. Im Gegensatz zum SBC-Befehl bleiben die V-Flagge und die Carry jedoch erhalten.

Um die Wirkungsweise dieser drei Befehle zu verstehen, sollten Sie sie mit dem Simulator ausgiebig ausprobieren. Erinnern Sie sich dazu an Ihr allererstes Maschinensprache-Programm zurück.

Begib Dich direkt dorthin

Nach so viel Theorie wollen wir uns wieder mal einem Programm zuwenden. Das zweite Beispiel auf Ihrem Datenträger enthält eine Routine, mit der ein Byte in hexadezimaler Darstellung auf dem Bildschirm ausgegeben werden kann.

Warum das Programm mit dem Befehl

JMP

beginnt, steht einige Zeilen tiefer. Aber daß er ausgeschrieben JuMP heißt und 'springe' bedeutet, sollen Sie jetzt schon wissen. Dieser Drei-Byte-Befehl veranlaßt den Prozessor, seine Arbeit an der angegebenen Adresse fortzusetzen, und zwar ohne Rücksicht auf

den Inhalt des Status-Registers. Ein weiterer Unterschied zu den Branch-Befehlen ist, daß man mit JMP beliebige Entfernungen innerhalb des Speichers überbrücken kann. Die Zieladresse steht explizit im Befehl und nicht als Differenz zum augenblicklichen Stand des Programmzählers.

Sichtbarer Erfolg

In der nächsten Programmzeile wird ein Label eingeführt, unter dem unsere Routine das auszugebende Byte finden soll.

Die folgende Zeile ist das Sprungziel des JMP-Befehls und der Anfang des eigentlichen Programmes. Die ersten beiden Befehle sorgen dafür, daß ein Dollarzeichen als Kennung für 'Hexadezimal' ausgegeben wird.

Das auszugebende Byte wird alsdann in den Akku geladen und viermal nach rechts geschoben. Dadurch gelangt der Inhalt des oberen Nibbles — das ja zuerst auf dem Bildschirm erscheinen soll — in die unteren vier Bits des Akkus, und das obere Nibble wird gleich Null.

So wird es auch von der Routine PNIB erwartet, die dazu dient, eine Zahl zwischen $00 und $0F auszugeben. Nach ihrer Beendigung lädt das Programm den Inhalt von WERT erneut in den Akku und maskiert mit dem Befehl 'AND #$0F' das untere Nibble aus. So kann es ebenfalls mittels PNIB ausgegeben werden. Zu dem folgenden RTS kommen wir noch.

PNIB soll eine Hex-Ziffer ausgeben. Aus einer ASCII-Tabelle entnehmen wir: Die Ziffern 0 bis 9 haben die Codes $30 bis $39, zu den Buchstaben A bis F gehören die ASCII-Werte $41 bis $46.

Die Routine PNIB arbeitet nun folgendermaßen: Wie bereits gesagt, erwartet sie im Akku eine Zahl mit gelöschtem High-Nibble. Durch den ORA-Befehl werden die unteren vier Bits nicht angetastet (Im Argument sind sie gleich Null.), und das höherwertige Nibble erhält eine Drei. Für die Ziffern 0 bis 9 enthält der Akku also schon den richtigen Wert zur Übergabe an die Betriebssystem-Routine PRCH. Das wird durch die nächsten beiden Befehle überprüft. Sie bewirken einen Sprung zum Label OKAY, wenn der Akku eine ASCII-Ziffer enthält.

Den Befehl „CMP #'9+1" wollen wir etwas genauer unter die Lupe nehmen: Der Assembler erkennt beim Übersetzen an dem Apostroph, daß ein ASCII-Zeichen folgt und setzt dafür den entsprechenden Wert ein. Die Anweisung „+1" läßt ihn noch Eins addieren. Im Simulator liest sich der Befehl somit als „CMP #$3A", also Vergleich mit dem der 9 folgenden Zeichen. Und so soll es auch sein, weil die CMP-BCC-Sequenz auf 'kleiner' und nicht auf 'kleiner oder gleich' prüft.

Wird der Sprung nicht ausgeführt, so enthält der Akku mindestens $3A. Zu $41, dem Wert für ein A, fehlen also noch 7. Da beim Erreichen des ADC-Befehls die Carry in jedem Falle gesetzt ist, kann man sich sparen, sie zu löschen. Man gibt im Argument einfach einen weniger an. So erklärt sich der Befehl „ADC #6".

Das Folgende ist wieder simpel: Mit „JSR PRCH" wird das Zeichen ausgegeben, und der RTS-Befehl führt in das aufrufende Programm zurück.

Von BASIC nach Maschine

Das besprochene Programm eignet sich sehr gut als kleines Hilfsprogramm, das man auch von BASIC aus aufrufen kann. Dazu überspielen Sie es aus dem Hauptmenue der Assemblerschule auf einen eigenen Datenträger. Dann schalten Sie den Rechner kurz aus und wieder ein, laden den INPUT-Ass (Den Assembler aus INPUT 64, Ausgabe 6/86) und mit ihm das abgespeicherte Beispielprogramm. Assemblieren Sie es entweder in den Speicher oder, wenn Sie es für später lauffähig aufheben wollen, auf einen Datenträger. Dann können Sie es mit LOAD"name",8,1 einfach wieder laden.

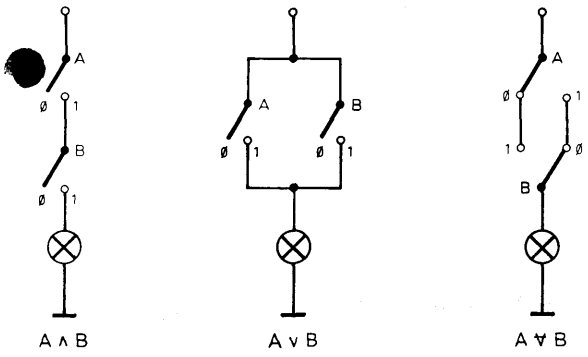


Bild 3: Die logischen Verknüpfungen AND, OR und EXOR lassen sich auch schaltungstechnisch darstellen.

Gestartet wird das assemblierte Programm mit SYS 49152 (49152 ist gleich $C000, der Startadresse). Vorher können Sie mit POKE 49155,zahl den auszugebenden Wert festlegen.

Nun zu dem oben angekündigten tieferen Sinn des JMP-Befehles am Anfang des Programmes: Bislang haben wir Zahlen immer hinter einem Programm aufgehoben. Solange man nur in Assembler programmiert, ist das auch in Ordnung. Bei einem Programm, das man aus BASIC heraus aufrufen will und das Werte übergeben bekommt, ist die hier vorgestellte Reihenfolge jedoch sinnvoller. Wenn Sie nämlich Änderungen an dem Programm vornehmen, etwa um hinter dem Byte noch ein RETURN auszugeben (Das sollten Sie übrigens mal probieren!), ändert sich automatisch die Programmlänge, und Sie müßten sich eine andere POKE-Adresse für die Wertübergabe merken.

. . . und wieder zurück

Der erste RTS-Befehl (der an der Adresse $C01B) bewirkt eine Rückkehr nach BASIC. Der BASIC-Interpreter legt nämlich bei der Ausführung des SYS-Befehles automatisch eine Rückkehr-Adresse auf dem Stack ab. Dadurch ist es möglich, auch aus BASIC-Programmen heraus eine Maschinensprache-Routine aufzurufen und anschließend mit dem folgenden BASIC-Befehl weiterzumachen.

Sie können in den bisherigen Beispielprogrammen den abschließenden BRK-Befehl auch durch ein RTS ersetzen, wenn Sie sie im BASIC-Direktmodus ausprobieren wollen. Dadurch vermeiden Sie, daß der Bildschirm nach der Abarbeitung gelöscht wird.

Innerhalb des Kurses liegt übrigens beim Start des Simulators immer die Startadresse des Editors auf dem Stack. Diese können Sie auch unter dem vordefinierten Label EXIT erreichen.

Zehn statt sechzehn

Zum Schluß wollen wir uns noch — wie in der letzten Folge angekündigt — die Wirkung der D-Flagge ansehen. Wenn sie gesetzt ist, arbeitet der Prozessor im sogenannten Dezimal-Modus. Für die Additions- und Subtraktionsbefehle werden dann Zahlen im BCD-Format verwendet. BCD heißt 'Binary Coded Decimal', also binär kodierte Dezimalziffer.

Tabelle 2: Der besseren Lesbarkeit und Genauigkeit der BCD-Darstellung fallen sechs Bit-Kombinationen zum Opfer.

Binär	BCD
%0000	0
%0001	1
%0010	2
%0011	3
%0100	4
%0101	5
%0110	6
%0111	7
%1000	8
%1001	9
%1010	ungültig
%1011	ungültig
%1100	ungültig
%1101	ungültig
%1110	ungültig
%1111	ungültig

Normalerweise kann man mit einem Nibble sechzehn verschiedene Zahlen darstellen. Im BCD-Code verwendet man nur zehn davon (Siehe Tabelle 2), also jeweils ein Nibble für eine Dezimalziffer.

Die gesetzte D-Flagge im Status-Register bewirkt nun, daß das Ergebnis einer Addition oder einer Subtraktion automatisch im BCD-Code dargestellt wird. Ein Überlauf in das nächsthöhere Nibble entsteht nicht erst beim Überschreiten von 15, sondern bereits wenn das Resultat größer als neun wird. Anschauliche Beispiele dazu finden Sie in den Erklärungen auf Ihrem Datenträger.

Der Befehl, mit dem man die D-Flagge setzt, heißt

SED (SEt Decimal-flag)

In den 'normalen' Rechenmodus können Sie den Prozessor mit dem Befehl

CLD (CLear Decimal-flag)

zurückschalten. Diesen Befehl sollten Sie auch zu Anfang eines jeden Pro-

Zum Programm

Die INPUT 64-Assembler-Schule setzt sich aus mehreren Teilen zusammen. Nach dem Laden sehen Sie ein Titelbild, von dem aus Sie mit einem beliebigen Tastendruck in das Hauptmenü gelangen.

Wenn Sie nun F1 drücken, gelangen Sie in ein Menü, das Ihnen verschiedene Themen zur Auswahl stellt. Die Erklärungen, die Sie jetzt abrufen können, sollten Sie parallel zum Beiheft lesen. Beide Medien ergänzen sich hier. Sie können die Erklärungen auch mit CTRL-b ausdrucken. Ins Hauptmenü gelangen Sie jederzeit mit der STOP-Taste zurück.

Mit F3 gelangen Sie aus dem Hauptmenü zu einer Auswahl verschiedener Beispielprogramme. Sie können eines davon auswählen, das Sie sich dann im Editor anschauen oder auch verändern können. Wenn Sie an dieser Stelle eine Null eingeben, enthält der Editor das zuletzt bearbeitete Programm, beim ersten Aufruf ist der Textspeicher leer.

Wenn Sie ein Beispielprogramm bearbeitet haben und - mit der STOP-Taste - wieder ins Hauptmenü springen, können Sie Ihren Text auch auf einen Drucker ausgeben lassen oder auf einen eigenen Datenträger abspeichern. Abgespeicherte Programme können Sie direkt mit dem INPUT-Ass (Ausgabe 6/86) laden und weiterbearbeiten.

Vom Editor aus gelangen Sie mit F7 in einen integrierten Simulator. Hier können Sie unsere Programmbeispiele oder Ihre selbstentworfenen Programme ablaufen lassen und testen, ob sie sich erwartungsgemäß verhalten.

Ausführliche Hinweise zur Bedienung des Editors und des Simulators sind im Programm enthalten. Sie können Sie von dort aus jeweils mit der Funktionstaste F6 aufrufen. Es wird empfohlen, diese Seiten vor der Benutzung des Programmpakets einmal gründlich zu lesen. Besitzer eines Druckers können sie auch mit CTRL-b zu Papier bringen.

Die INPUT 64-Assemblerschule ist eine Serie, die in der Ausgabe 3/87 begonnen hat. Die einzelnen Lektionen bauen aufeinander auf. Wer noch keine Erfahrungen mit der Maschinensprache-Programmierung hat, tut gut daran, mit der ersten Folge anzufangen.

grammes ausführen lassen, wenn unklar ist, in welchem Zustand sich der Prozessor beim Programmstart befindet.

Verwandlung

Das dritte Beispielprogramm in dieser Folge benutzt den Dezimal-Modus der CPU zur Umrechnung einer Zwei-Byte-Hexadezimalzahl in eine fünfstellige BCD-Zahl. Das Programm beginnt wieder mit einem Sprung, der die verwendeten Datenspeicher überbrückt.

Das eigentliche Programm beginnt mit dem Löschen des Ergebnis-Puffers. Dann wird das Index-Y-Register als Zähler für sechzehn Schleifendurchläufe — für jedes Bit einen — initialisiert und die Dezimal-Flagge gesetzt.

Die ersten beiden Befehle innerhalb der Schleife bewirken eine Verdoppelung der umzuwandelnden Zahl und eine Übertragung des höchstwertigen Bits in die Carry-Flagge.

Die folgenden neun Befehle bewirken beim ersten Schleifendurchlauf noch nicht viel. Jedoch handelt es sich hierbei um die Verdoppelung des Inhaltes des Ergebnisspeichers im BCD-Format mit gleichzeitiger Addition des Überlaufs aus ARGU. Dadurch, daß diese Schleife sechzehnmal durchlaufen wird, wird jedes Bit aus ARGU stellenrichtig zu ERGB addiert.

Nach dem letzten Schleifendurchlauf löscht das Programm die D-Flagge wieder und kehrt zum Aufrufpunkt zurück. Dort könnte beispielsweise eine Routine folgen, die den Inhalt von ERGB als fünfstellig formatierte Dezimalzahl auf dem Bildschirm ausgibt.

Diese Routine sollten Sie übrigens mit dem in dieser Folge Gelernten erstellen können. Das wäre dann auch die Hausaufgabe. Dazu ist zwar eine ganze Menge Tipparbeit nötig, aber immerhin entwickeln Sie auf diese Weise ein Programm, das — ähnlich wie das zweite Beispiel — einen praktischen Nutzen hat. Wie gewohnt werden wir Ihnen in der nächsten Folge eine Musterlösung vorstellen.

Außerdem sollten Sie auch im Juni wieder dabeisein, wenn Sie wissen wollen, warum die Indexregister Indexregister heißen und wie man mit Maschinensprache Ruck-Zuck Riesen-Speicherbereiche verschieben kann. Hajo Schulz

Überflüssiges merken

Byte-Compactor für alle Programme

Natürlich muß das nicht sein, wenn es gelingt, die Wiederholungs-Bytes vor dem Sichern auf Datenträger quasi zu zählen und nach dem Laden (unmittelbar nach dem Anstarten) wieder zu rekonstruieren. Wie das der Compactor genau macht, können Sie in dem Abschnitt "Compactor intern" nachlesen.

Demonstration

Wenn Sie in dem Menü die Demonstration angewählt haben, erscheint eine hochauflösende Grafik, die nun kompaktiert wird. Um dieses anschaulich zu gestalten, wurde der Speicherbereich für das kompaktierte Bild ebenfalls (in den oberen) Bereich der HiRes-Seite gelegt.

Durch Drücken einer beliebigen Taste können Sie die Demonstration steuern. Kurze Texte in der untersten Zeile weisen Sie auf die wesentlichen Informationen hin. Während dieser Vorführung ist ausnahmsweise die INPUT-Funktion CTRL + I umgelenkt. Sie führt nicht wie gewohnt zum Inhaltsverzeichnis, sondern auf die Menüseite zurück. Von hier aus ist Input wieder wie gewohnt zu bedienen.

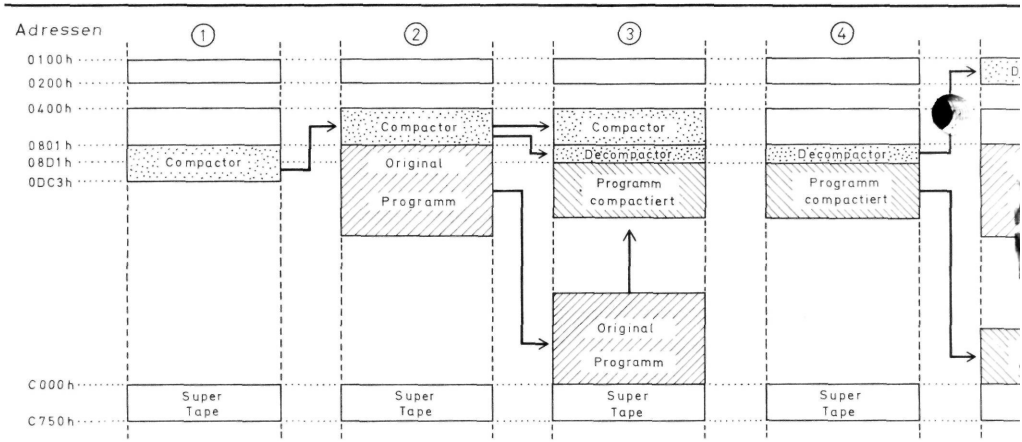
In vielen Programmen kommen an einigen Stellen gleiche Bytes mehrfach hintereinander vor. Am auffälligsten tritt diese Erscheinung bei einem Großteil der Grafikbilder auf: In der Mitte befindet sich eine Figur, rundherum gähnende Leere, zahllose Null-Bytes. Dieser Ballast muß nun auf den Datenträger geschrieben und auch wieder gelesen werden. Das muß doch nicht sein, oder?

Einfache Bedienung

Sie laden — natürlich außerhalb von INPUT 64 — den Compactor von Ihrem Datenträger in den Rechner und starten diesen mit RUN. Sie werden zunächst gefragt, wie das zu kompaktierende Programm heißt und das fertig kompaktierte Programm heißen soll (hier sollten Sie einen anderen Namen eingeben). Danach geben Sie die Geräteadressen für das Quell- und das Zielprogramm ein.

Sie können jetzt noch entscheiden, ob das Speicherende bei C000h (49152dez) „festgenagelt" werden soll. Sofern Sie mit SuperTape arbeiten oder vielleicht in diesem Bereich

Speicherbelegung Compactor



einen Monitor liegen haben, müssen Sie diese Abfrage mit "j" (für ja) beantworten. Ansonsten geben Sie "n" (für nein) ein und können dann auch sehr lange Programme bearbeiten.

Mit RETURN starten Sie jetzt den Lade- und den Kompaktiervorgang. Wenn das Kompaktieren abgeschlossen ist, weisen Sie den Compactor mit einem weiteren RETURN an, das kompaktierte Programm wieder abzuspeichern. Erscheint die Meldung „Kürzen nicht möglich" auf dem Bildschirm, war das „kompakte" nicht mindestens 1 Byte kürzer als das Original.

Das war's auch schon! Wenn Sie das kompaktierte Programm nun wieder laden, werden Sie außer einer kurzen „Denkpause" unmittelbar nach dem Anstarten keinen Unterschied feststellen; außer der Tatsache natürlich, daß das Laden viel schneller ging.

Den Tiger gibt's dazu

Da wir den Kompaktierungs-Vorgang am Beispiel einer HiRes-Grafik veranschaulichen, bot es sich an, Ihnen dieses Bild auch zur Verrfügung zu stellen. Sie können also das Bild — selbstverständlich in kompaktierter Form — auf den eigenen Datenträger überspielen.

Nachdem Sie das Bild von Ihrem Datenträger geladen haben, wird es eben mal schnell dekompaktiert und in den richtigen Speicherbereich geschoben und angezeigt. Nach einem beliebigen Tastendruck wird ein Reset ausgelöst, um alle Zeiger wieder in den Normalzustand zurückzusetzen.

F. Rosenschein / WM

Compactor intern

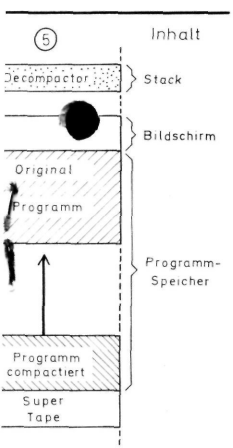
Wie bereits angedeutet, ist das Grundprinzip relativ simpel. Der Compactor durchsucht das Original-Programm nach sich unmittelbar wiederholenden Bytes, merkt sich die Anzahl und verschlüsselt diese Information in Steuer-Byte, Anzahl der Wiederholungen und Darstellungsbyte. Beim Dekompaktieren wird diese verschlüsselte Information wieder aufgelöst, und das Original-Programm ist damit wiederhergestellt.

Als Steuer-Byte wird ein Byte verwendet, das in dem Programm gar nicht oder nur sehr selten vorkommt, da immer dann, wenn dieses Byte im Programm selber steht, dieses logischerweise ebenfalls verschlüsselt werden muß, um dem Decompactor mitzuteilen, daß es sich eben nicht um ein Steuer-Byte handelt.

Soweit zu der Theorie. In der Praxis stellt sich das Problem, daß der Compactor, der Decompactor, das Original-Programm und das kompaktierte Programm ja irgendwo im Speicher stehen müssen. Hinzu kommt noch die Bedingung, daß ein Programm, das mit RUN startfähig sein soll, am Anfang des Programmspeichers — also bei 0801h (2049dez) — stehen muß.

Die Grafik zeigt für jedes Stadium die Speicherbelegung. Wenn Sie den Compactor geladen haben (1), liegt dieser selbstverständlich am Anfang des Programmspeichers. Nach den entsprechenden Eingaben (siehe Bedienungshinweise) verschiebt sich der Compactor in den Bildschirmbereich ab 0400h und lädt das Original-Programm nach 0801h (2). Bevor der eigentliche Kompaktierungs-Vorgang beginnt, wird das Original-Programm an das Speicherende geschoben und der Decompactor (war im Compactor enthalten) an die Adresse 0801h kopiert. Unmittelbar dahinter beginnt das kompaktierte Programm, welches nun Stück für Stück erzeugt wird (3). Nachdem die Kompaktierung abgeschlossen ist, wird das kompaktierte Programm mit dem Decompactor abgespeichert.

Bild (4) zeigt den Zustand nach dem Laden des kompaktierten Programms. Wird dieses mit RUN gestartet, kopiert sich der Decompaktor in den Prozessor-Stack (ab Adresse 0100h), schiebt das kompaktierte Programm an das Ende des Programmspeichers und erzeugt am Anfang des Programmspeichers das Original-Programm (5).



Die Speicherbelegung während der einzelnen Ablaufphasen.

683 Blöcke in 60 Sekunden

Eine randvolle Diskette File für File zu kopieren kann beinahe in eine abendfüllende Beschäftigung ausarten. Mit SpeedBackup wird die Datensicherung zur Minutensache.

SpeedBackup ist ein Programm zum Kopieren von Disketten, also kein Filecopy-Programm. Es werden ein oder zwei Laufwerke unterstützt. Kopiert werden „normale" C64-Disketten, das heißt einseitige Disketten mit 35 Spuren.

SpeedBackup wird nach dem Abspeichern auf Diskette mit LOAD"name",8 geladen und mit RUN gestartet. („name" ist natürlich der Name, den Sie beim Abspeichern mit CTRL und s eingegeben haben.) Zunächst müssen Sie entscheiden, ob das Programm mit einem oder mit zwei Laufwerken arbeiten soll. Wird die Frage mit 2 beantwortet, müssen zuerst die beiden Laufwerke initialisiert werden. Hierbei kann das Ziel-Laufwerk eine Gerätenummer von 8 bis 11 haben, die vom Programm automatisch festgestellt wird. Das Quell-Laufwerk muß dagegen immer die Geräteadresse 8 haben, ebenso das angeschlossene Laufwerk im Modus 1 (ein Laufwerk). Da die Laufwerke beim Initialisieren nacheinander eingeschaltet werden, können auch beide die gleiche Gerätenummer haben.

Nach der Bestätigung mit RETURN erwartet SpeedBackup noch einige Parameter. Gängige Werte werden jeweils vorgegeben, die im Normalfall mit RETURN übernommen werden können. Es geht um Formatieren — Ist die Zieldiskette bereits formatiert, kann hier mit Nein geantwortet werden, was natürlich eine Zeitersparnis mit sich bringt. An dieser Stelle kann auch $ eingegeben werden, um sich das Inhaltsverzeichnis der Diskette auf dem Bildschirm anzusehen. Achtung: sämtliche Daten der Zieldiskette gehen auf jeden Fall unwiderbringlich verloren, da immer sämtliche Blöcke kopiert werden.

Verifizieren — Dies ist bei einem zuverlässigen Laufwerk und Marken-Disketten im allgemeinen nicht notwendig, so daß hier mit Nein geantwortet werden kann, was ebenfalls eine Zeitersparnis bringt.

Lesefehler mitkopieren — Wird diese Frage mit Ja beantwortet, werden Lesefehler mitkopiert. Wird mit Nein geantwortet, werden die defekten Blöcke zwar mit übernommen, die Prüfsumme wird jedoch angeglichen, so daß auf der Zieldiskette keine Fehlermeldung mehr erscheint.

Die letzte Frage erscheint nur bei der Vorgabe „zwei Laufwerke". Es wird geprüft, mit wie vielen Laufwerken die folgende Kopie gemacht werden soll (1 oder 2).

Sind alle Eingaben komplett, werden Sie zum Einlegen der Diskette(n) aufgefordert. Nach der Quittierung mit RETURN wird aus Timing-Gründen der Bildschirm abgeschaltet, und die Laufwerke beginnen beziehungsweise das Laufwerk beginnt zu arbeiten. Durch Betätigen der Funktionstaste F1 kann der Kopiervorgang abgebrochen werden. Mit einem Laufwerk muß die Diskette während des Kopierens vier Mal gewechselt werden, bei zwei Laufwerken erspart man sich dies Discjockey-Training. Während des Zugriffs darf die Restore-Taste nicht berührt werden, da sonst das Timing gestört werden kann. Während des Kopierens aufgetretene Lese- und Schreibfehler erscheinen auf dem Bildschirm. Ist die zu lesende Diskette defekt, kann sich der Lesevorgang erheblich verzögern, da die Fehler analysiert werden müssen. Im Normalfall dauert eine Kopie umgefähr 60 Sekunden, die Zeit zum Diskettenwechsel nicht mitgerechnet. Falls weitere Disketten dupliziert werden sollen, können Sie nach Betätigen von RETURN erneut die Parameter eingeben. Das Programm hat keinen Ausgang in Richtung BASIC, nach dem Ende der Sitzung muß der Rechner ausgeschaltet werden.

SpeedBackup ist speziell für die Floppy 1541 geschrieben worden. Auf den redaktionseigenen 1570/1571 lief das Programm (fast) nie — ausprobieren! Martin Friedl/JS

GOLUM^4

Vier Spiele in einem

Können Sie so aus dem Stegreif fünf Gesellschaftsspielarten nennen? Wie wäre es zum Beispiel mit:

- Strategie-
- Taktik-
- Geschicklichkeits-
- Reaktions-
- Knobelspiel

Mit GOLUM möchten wir Ihnen ein Gesellschaftsspiel für Ihren C 64 vorstellen, das alle eben genannten Spieleigenschaften in sich vereint. Sie versammeln sich am Rechner in gemütlicher Runde am „Bildschirmbrett". Es geht darum, Spielfiguren mit mehr oder weniger Würfelglück vom Start ins Ziel zu bringen, wobei Blockaden und mißgünstige Mitspieler Ihre Absicht durchkreuzen können.

Gelingt es einem, durch geschicktes Ausnutzen der geworfenen Augenzahl mit einer Spielfigur eine Blockade zu erreichen, stellt GOLUM neue Aufgaben. Je nach zufälliger Neigung muß der Spieler eine Geschicklichkeitsaufgabe (Labyrinthspiel), eine Reaktionsaufgabe (Ballspiel) oder eine Knobelaufgabe (Wortratespiel) bewältigen. Nur wer das Ziel in der vorgegebenen Zeit erreicht, kann mit Taktik und Strategie versuchen, die Mitspieler durch geschicktes Positionieren der gewonnenen Blockade aufzuhalten. Ihnen stehen drei Spielfiguren zur Verfügung. Sie können zu zweit, zu dritt oder zu viert spielen. Bevor Sie sich ans Brett wagen, hier zuerst die Spielbeschreibung zu GOLUM.

Spielgeschick

Fast alle Steuerfunktionen im Spiel sind entweder mit dem JOYSTICK (in PORT 2) oder den CURSOR und RETURN-Tasten zu bedienen.

Öffnen Sie die Computerspielkiste. Es gibt Würfel, die nicht vom Tisch fallen können, Spielfiguren, die nicht im Laufe der Zeit verschwinden, und Bälle, die auf dem Spielfeld bleiben. Ein Modul, vollgepackt mit vier Spielen, jedes aus einer anderen „Spielecke". Hier bietet sich die Chance, mit Freunden den Ärger über einen verregneten Nachmittag zu vergessen.

Auf der Titelseite können Sie und Ihre Mitspieler Ihre Namen eingeben. Tippfehler werden ebenso gnädig wie beim normalen BASIC-Editor behandelt. Mit der Reihenfolge der Namenseingabe legen Sie zu Beginn des Spieles die Spielerreihenfolge fest. Nach diesen kleinen Vorarbeiten geht es auch schon los.

Inder Hauptspielebene von GOLUM bewegen sich alle Spieler in Richtung des Kreises, welcher im oberen Bereich des Spielfeldes liegt. Um eine Spielfigur bewegen zu können, muß der Spieler erst eine Zahl würfeln. Nach Ausführen der verlangten Steueranweisung wird der Würfel gestoppt. Der Spieler wählt seine Spielfigur (Richtungsauswahl —› links oder rechts ‹—) und 'nimmt' sie mit einem Druck auf die FEUER- oder RETURN-Taste mit. Nun kann sich der Spieler mit seiner Figur auf dem ganzen Spielfeld bewegen. Die Laufrichtung kann sogar während des Zuges geändert werden, wobei jeder Schritt zählt. Damit lassen sich sehr überraschende Winkelzüge ausführen, mit denen man seine Mitspieler um eine Figur bringt oder sich einer Blockade bemächtigt.

Doch der Spielleiter, der im Spielprogramm seiner Aufgabe nachgeht, achtet auf unerlaubte Züge. Solchen Irrtümern begegnet er mit einer deutlichen Bildschirmmitteilung. Erst wenn Sie mit einer Taste Ihre Einsicht kundtun, stellt GOLUM die Spielfigur zurück zum Ausgangspunkt des Zuges, und das Spiel geht weiter. Gelangen Sie in einem Zug genau auf das Feld, das mit einer Blockade besetzt ist, stellt GOLUM Sie vor eine der drei anderen Spielaufgaben. Werden diese Aufgaben von Ihnen erfüllt, so dürfen Sie Ihre 'besetzte' Blockade auf dem Spielfeld verschieben.

Auf allen Ebenen

Testen Sie Ihre Geschicklichkeit im LABYRINTHSPIEL.

Bewegen Sie eine Schlange vom Start zum Zielpunkt durch den Irrgarten. Vermeiden Sie es jedoch, gegen eine Wand zu prallen oder den Rückzug anzutreten. In solchen Fällen haben Sie das Spiel verloren.Ebenso geht es Ihnen, wenn Sie allzu gemütlich um die Ecken kurven. Zu Beginn können Sie unter neun verschiedenen Schwierigkeitsstufen wählen. Daran anschließend müssen Sie sich für eine der Anfangsstellungen entscheiden und dann geht es auf Knopfdruck los. Zum Steuern benutzen Sie bereits bekannten Steuerfunktionen: CURSOR-Tasten oder Joystick (in Port-2).

Fordern Sie Ihr Reaktionsvermögen im BALLSPIEL.

Ähnlich wie bei den bekannten BREAKKOUT-Spielen bewegt sich ein

Ball innerhalb eines Feldes. Sie versuchen seine Flugbahn mit einem Fangnetz so zu beeinflussen, daß er innerhalb der gesetzten Zeitspanne den wandernden Ausgang trifft. Auch hier wird mit den CURSOR-Tasten oder Joystick in Port-2 gesteuert.

Setzen Sie Ihre Knobelfähigkeiten im RATEFIXSPIEL ein.

Innerhalb einer festen Zeitspanne müssen Sie versuchen, ein Wort durch richtiges Einsetzen von Buchstaben zu erraten. Mit den CURSOR-Tasten oder dem Joystick wählen Sie einen Buchstaben aus, den Sie dann mit der Feuer- oder RETURN-Taste übernehmen. Haben Sie richtig geraten, erscheint der Buchstabe an den entsprechenden Stellen im Anzeigefeld, war es falsch, riskieren Sie den Verfall eines alten Gemäuers.

Das sind also die Sonderaufgaben des GOLUM-Spiels, denen Sie sich bei Erreichen einer Blockade stellen müssen. Was auf Sie zukommt, entscheidet ein Zufallsgenerator. Nur wenn Sie die Aufgabe bestanden haben, dürfen Sie Ihre Blockade einem anderen „vor die Füße" legen. Gewonnen hat, wer als erster eine seiner Spielfiguren ins Zielfeld des Brettspieles gebracht hat. Wie alle Gesellschaftsspiele gewinnt GOLUM natürlich an Spielreiz, je mehr Spieler(innen) mitwirken.

Damit Sie die Spielfreuden auch außerhalb von INPUT 64 auskosten können, läßt sich GOLUM mit CTRL+S bei der ersten Bildschirmseite auf Datenträger abspeichern. GOLUM ist vollständig in Maschinensprache geschrieben und belegt über 100 Blöcke. Wer also im Commodore-Kassetten-Format speichern will, sollte eine entsprechende Kassette mit langer Laufzeit einlegen.

Th. Stahmer/RH

ID-Werkstatt

Beziehungen verbessert

Im launigen April kamen gleich drei Programme in die Werkstatt zur Untersuchung. Während INPUT-CAD und MiniDat, Baujahr 1986, dem Rückruf folgten und einer kurzen Reparatur unterzogen wurden, lieferte unsere Schwesterzeitschrift c't ein Sondermodell zur Übernahme ins INPUT 64-Angebot.

Kreise gezogen

Patch für INPUT-CAD liefert den letzten Schliff. Bereits in der letzten Ausgabe haben wir die uns bekanntgewordenen Fehler in INPUT-CAD dokumentiert und Hilfe versprochen. Neben den dort aufgeführten drei Fehlern berichtigt das — aus der ID-Werkstatt abzuspeichernde — Patch-Programm einen weiteren kleinen Fehler bei der Druckausgabe. Bisher war es nämlich nicht möglich, tatsächlich alle eingestellten Spalten zu Papier zu bringen. Die letzte Spalte wurde immer unterschlagen. Wenn Sie das Patch-Programm auf Ihren eigenen Datenträger gesichert haben, verfahren Sie bitte genauso wie bei der Ihnen inzwischen vertrauten Verbindung der einzelnen CAD-Teile. Das Patch-Programm lädt "CAD V4" nach und erzeugt das Programm "CAD V4 1". Der genaue Vorgang ist in den entsprechenden Beiheften mehrmals detailliert beschrieben worden.

Ans Band gehängt

MiniDat V2 aus INPUT 64 12/86 sträubte sich hartnäckig gegen jedwede Zusammenarbeit mit SuperTape oder MultiTape II. Die Verbesserungen gegenüber der Version V1 aus INPUT 64 2/86 führten zur Zerstörung wichtiger Zeropage-Adressen, über die MiniDat und SuperTape beziehungsweise MultiTape II Informationen austauschen. Diesmal können wir Ihnen ein bereits korrigiertes Programm anbieten. Ein Patch-Programm hätte MiniDat selbst an Blocklänge glatt überrundet. Kassettenbenutzer können also aufatmen. MiniDat V2.C kann aus der ID-Werkstatt ausgewählt und abgespeichert werden. Außerdem ist MiniDat nur noch 27 Blöcke lang, ganze 9 Blöcke weniger. Die Bedienung, wie sie in Heft 12/86 beschrieben ist, hat sich dadurch aber nicht geändert. Rückwärtsblättern wird nach wie vor über die F2-Taste erreicht (und nicht über F3, wie's druckfehlerhaft im Heft stand). RH

Kurz angebunden

c't-Leser und PC-1401-Besitzer werden nach diesem Stichwort Ausschau gehalten haben. Getreu unserer Devise „Software ohne Abtippen" finden Sie diesmal als drittes Angebot das Maschinenprogramm zur PC-1401-Schnittstelle in seiner neusten Version. Dem Rest der Welt sei kurz erklärt, worum es geht: Das Programm stellt ein leistungsfähiges BASIC-Entwicklungssystem für den programmierbaren PC-1401 von Sharp dar. Dieser Rechner hat etwa die Größe eines älteren Taschenrechners und versteht sogar BASIC. Da sich jedoch auf dem C64-Bildschirm Programm-Entwicklungen besser überschauen lassen als auf dem Winz-Display des PC-1401 mit seinen 16 Zeichen, kamen die Autoren Huth und Möllemann auf die Idee, mit etwas Hardware und der oben genannten Software eine Kopplung zwischen diesen beiden Rechnern herzustellen. Auf dem C64 Entwickeltes läßt sich damit mühelos im PC-1401 in die Tasche stecken. Wer neugierig geworden ist: ein Blick in c't 3/87 auf Seite 162 ff. schafft Klarheit.

RH

Zeichen und Wunder

Grafische Änderungen im Zeichensatz

Wer Zeichen setzen möchte, weil er den Standard-Zeichensatz des C64 allzu bezeichnend findet, sollte den „Tauchkurs" dieser 64er Tips mitmachen, um zu verstehen, aus welchen Tiefen sich der Video-Controller, kurz VIC genannt, seine Bildschirminformationen hochholt.

Die einfache Zeichenausgabe läßt sich beim C64 in BASIC ohne Probleme verwirklichen. Und da das alles so schön einfach zu sein scheint, entsteht meist erst nach einiger Zeit der Wunsch, bei einer speziellen Problemstellung über Zeichen verfügen zu können, die es auf dem C64 nun mal nicht gibt. Als Produkt aus dem anglo-amerikanischen Raum fehlt dem C64 der deutsche Zeichensatz. Aber mathematische oder wissenschaftliche Sonderzeichen wären bei manchen Darstellungen auch ganz hübsch. Ja, es lassen sich sogar ganze Bilder und Figuren mit einem veränderten Zeichensatz erzeugen.

Also Zeitschriften und Bücher durchforsten und ran an den Zeichensatz. Nach wüstem Hin- und Herblättern merkt man dann aber, daß eine ganze Menge im Rechner geschehen muß, bis ein einzelnes Zeichen in der gewünschten Form den Bildschirm ziert.

Damit der C64 auf dem angeschlossenen Bildschirm für Erleuchtung sorgen kann, verfügt er über einen besonderen Baustein, der ausdrücklich für diese Aufgabe hergestellt wurde, den Videochip 6569, den man im geöffneten Rechner meist unter einem separaten Metallkästchen findet. Dieses Kästchen dient nicht nur der Abschirmung elektromagnetischer Streufelder, sondern auch der Wärmeableitung, da diesem Baustein in seinem Arbeitseifer ganz schön heiß werden kann. Ältere C64-Rechner verunsichern den Benutzer manchmal durch unruhig wackelnde Bildschirmanzeigen bis hin zu sich langsam auflösenden Zeichendarstellungen. Ein Grund kann in der unzureichenden Kühlung des Chips zu suchen sein. Mit einem entsprechenden Kühlkörper, der mit Wärmeleitpaste aufmontiert wird, können Sie Abhilfe schaffen (gibt es in fast jeder Elektronik-Fachhandlung).

Der Bezeichung 6569 läßt sich entnehmen, daß dieser Chip ebenfalls zur 65xx-Familie gehört. Er erledigt nicht nur sämtliche Aufgaben, die bei der Bildschirmsteuerung anfallen, sondern entlastet den Prozessor 6510 bei einigen Aufgaben, indem er selbständig dynamisch auf Speicherbereiche zugreift. Um die „Intelligenz" dieser Konstruktion ausnutzen zu können, stehen 47-Register bereit, die die Aufgabenverteilung steuern und gleichzeitig Informationen zurückgeben können, quasi 47 Telefonleitungen zum Video-Controller.

Hintergründe

Der VIC kennt fünf verschiedene Betriebsarten, in denen er etwas auf den Bildschirm zaubern kann:

- Zeichendarstellung normal
- Zeichendarstellung Multicolor
- Zeichendarstellung Extended Color
- Einzelpunktdarstellung (HiRes)
- Sprites

Wenn Sie mit einem schlichten PRINT "A" den VIC veranlassen wollen, diesen Buchstaben an der Cursor-Position auf den Bildschirm zu zeichnen, laufen die Leitungen im C64 „heiß". Der VIC stellt zuerst einmal fest, wo der Video-RAM liegt. Von dort holt er sich das betreffende Byte und bildet daraus einen Zeiger auf die passende Position im Zeichengenerator. Von dort übernimmt der VIC das Bit-Muster des Zeichens und steuert den Elektronenstrahl des Bildschirms so, daß für jedes gesetzte Bit im Muster ein Punkt auf der Mattscheibe erscheint.

Geben Sie in Ihren Rechner POKE 1024,1 ein, erscheint, spätestens wenn Sie den Cursor mit HOME positionieren, ein „A" auf dem Bildschirm. Damit aber nicht nur hellblaue Buchstaben auf blauem Grund erscheinen, bezieht der VIC zusätzlich Informationen einmal aus dem Farb-RAM, das fest ab $D800 (55296) liegt, zum anderen stehen zusätzlich Register für Hintergrund- und Rahmenfarbe zur

Verfügung. Mit POKE 55296,7 können Sie die Farbe direkt bestimmen. Hier können Sie auch mit anderen Werten herumspielen.

Aufmerksame Leser dürften gemerkt haben, daß der Bereich ab $D000 (53248) offensichtlich gut belegt ist. Wer sich noch an die Speicherbelegungspläne der 64er Tips 2/87 und 3/87 erinnert, weiß, daß gerade dieser Bereich dreistöckig angelegt ist. An dieser Adresse liegen RAM, VIC und Zeichengenerator übereinander!

Damit der VIC überhaupt auf diesen Bereich zugreifen kann, erzeugt der C64 ein sogenanntes ROM-Image, eine Art Abbild des Zeichensatzgenerators, an den Adressen 4096-8191 ($1000-$1FFF) oder bei 36864-40959 ($9000-$9FFF). Da der VIC selbst nur Adressen von $0000 bis $3FFF erzeugen kann — die restlichen Adreßleitungen sind einfach nicht verschaltet — wird er zusätzlich noch von dem „Complex Interface Adapter" CIA 2 unterstützt. Der VIC kann von sich aus nur 16 KByte adressieren. Über den CIA 2 kann jeweils in 16-KByte-Schritten weitergeschaltet werden. Die Konsequenz daraus ist aber, daß Video-RAM, Zeichensatz-RAM und SPRITE-Daten immer nur im gleichen 16K-Bereich liegen können.

Natürlich ist der Zeichensatzgenerator fest in einem ROM-Baustein untergebracht, denn sonst gäbe es böse Überraschungen, wenn der Rechner einmal abstürzt. Damit dürfte aber klar sein, daß man nicht einfach Änderungen am Zeichensatz vornehmen kann.

Um einen veränderten Zeichensatz zu erzeugen, führt man am besten folgenden Dreisprung aus:

Orginalzeichensatz ins RAM verschieben
Umschalten auf den RAM-Zeichensatz
Gezieltes Ändern der gewünschten Zeichen

Verlagerungen

Das Verschieben des Zeichensatzes läßt sich grundsätzlich auch von BASIC aus erreichen. Schneller geht es natürlich mit einem Programm in Maschinensprache. Während dieses Umkopierens muß der VIC abgeschaltet werden. In den 64er Tips finden Sie ein kleines Tool zum Abspeichern, das Ihnen die Programmierarbeit abnimmt. Anschließend ist es sinnvoll, den VIC erst einmal auf die Zeichensatz-Kopie umzuschalten, damit man während der Änderungen auch zu sehen bekommt, was man tut. Diese Umschaltung wird ebenfalls vom Tool erledigt. Anhand der Tabelle können Sie ablesen, welche Adressen mit welchen Werten versorgt werden müssen, wenn Sie andere RAM-Bereiche benutzen wollen.

Basisadresse

CIA 2
$DD00 (56578)

dez	hex	Bits	Basisadresse dez	hex
0	$00	xxxxxx00	49152	$C000
1	$01	xxxxxx01	32768	$8000
2	$02	xxxxxx10	16384	$4000
3	$03	xxxxxx11	0	$0000

Zeichensatzlage

Positionswert $D018 (53272) dez	hex	Bits	Lage des Zeichensatzes dez	hex
0	$00	xxxx 000x	0	$0000
2	$02	xxxx 001x	2048	$0800
4	$04	xxxx 010x	4096	$1000
6	$06	xxxx 011x	6144	$1800
8	$08	xxxx 100x	8192	$2000
10	$0A	xxxx 101x	10240	$2800
12	$0C	xxxx 110x	12288	$3000
14	$0E	xxxx 111x	14336	$3800

(* Basisadresse)

Video-RAM-Lage

Positionswert $D018 (53272) dez	hex	Bits	Lage des Video-RAM dez	hex
0	$00	0000 xxxx	0	$0000
16	$10	0001 xxxx	1024	$0400
32	$20	0010 xxxx	2048	$0800
48	$30	0011 xxxx	3072	$0C00
64	$40	0100 xxxx	4096	$1000
80	$50	0101 xxxx	5120	$1400
96	$60	0110 xxxx	6144	$1800
112	$70	0111 xxxx	7168	$1C00
128	$80	1000 xxxx	8192	$2000
144	$90	1001 xxxx	9216	$2400
160	$A0	1010 xxxx	10240	$2800
176	$B0	1011 xxxx	11264	$2C00
192	$C0	1100 xxxx	12288	$3000
208	$D0	1101 xxxx	13312	$3400
224	$E0	1110 xxxx	14336	$3800
240	$F0	1111 xxxx	15360	$3C00

(* Basisadresse!)

Video-RAM-Register $0288 (648) = Video-RAM-Adresse/256

Tabelle 1

darangehen, die Bit-Muster nach eigenem Bedarf zu verändern. Jedes Zeichen ist aus 8x8 Punkten aufgebaut. Jeder sichtbare Punkt entspricht einem gesetzten Bit. Jeweils 8 Bit sind zu einem Byte zusammengefaßt. Ein Zeichen verbraucht demnach 8 Speicherzellen zu je 8 Bit im RAM. Wollen Sie ein bestimmtes Zeichen verändern, müssen Sie ein wenig rechnen.

Basis-Adresse des Zeichen-RAM
Radr
Bildschirmcode des Zeichens Zcod
Zeichenadresse im RAM Zadr

Nach der Formel: Zadr=Radr+Zcod*8

finden Sie das erste Byte des Bit-Musters zum gesuchten Zeichen. Von BASIC aus können Sie durch acht POKE-Befehle in die Adressen von Zadr bis Zadr+7 ein Zeichen vollständig ändern. Damit das neue Zeichen aber auch den eigenen Vorstellungen entspricht, verwenden Sie am besten ein Entwurfblatt, wie es in Bild 2 abgebildet ist. Dabei ist darauf zu achten, daß das niederwertigste Bit — Bit 0 — jeweils den Punkt ganz links im Zeichen darstellt, also entweder 0 oder 1 sein kann, während der jeweils rechte Punkt mit dem höchsten Bit — Bit 7 —dargestellt wird. Setzen Sie diesen Punkt, müssen Sie den Wert $128=2^7$ zuaddieren. In Bild 3 können Sie ein Beispiel für eine solche Berechung nachvollziehen.

Nützliches

Mit dem Tool, das Sie sich aus den Tips heraus abspeichern können, haben Sie es leichter. Sie brauchen nur den entsprechenden Bildschirmcode anzugeben (C64-Handbuch, Seite 133/134) und anschließend acht Zahlen von 0 bis 255. Geben Sie dieses Zeichen vorher oder nachher mit dem PRINT-Befehl auf den Bildschirm aus, können Sie den Effekt Ihrer Aktivitäten direkt beobachten.

Das Zeichen-Tool stellt insgesamt vier Funktionen zur Verfügung.

Byte	Bit 7	6	5	4	3	2	1	0
1								
2								
3								
4								
5								
6								
7								
8								

Bild 2

Initialisieren
Abschalten des Tools
Änderungen im 1. Zeichensatz
Änderungen im 2. Zeichensatz

Mit **SYS 2105** wird die Erweiterung initialisiert. Der VIC wird auf den vierten 16K-Bereich bei $C000-$EFFF umgeschaltet; CIA 2 wird also auf BANK 3 gesetzt, die sozusagen im Schatten des KERNAL-ROM steht. Der Zeichensatz wird in den RAM-Bereich unter dem KERNAL nach $E000-$EFFF kopiert. Genaugenommen liegen dort sogar zwei Zeichensätze: Großbuchstaben/Commodore-Grafik und Groß-/Kleinbuchstaben. Zwischen diesen beiden Zeichensätzen können Sie wie gewohnt, zum Beispiel mit der SHIFT&COMMODORE-Taste, hin- und herschalten.

Das Tool garantiert also die weitere Verwendung von SuperDisk oder SuperTape. Wollen Sie mit Sprites arbeiten, müssen Sie diese natürlich auf diesen neuen Bereich anpassen. Zusätzlich wird der USER-Vektor auf eine Funktion umgelenkt, mit der Sie analog zu PEEK(X) jetzt auch Werte aus dem „versteckten" RAM-Bereich auslesen können. PRINT USER (57352) gibt das 9. Byte aus dem RAM-Bereich ab $E000 aus.

Die Erweiterung wird mit **SYS 2108** abgeschaltet und die gewohnte Konfiguration des C64 wiederhergestellt.

MIT
SYS 2111,Bc,W0,W1,W2,W3,W4,W5,-W6,W7 können Sie einzelne Zeichen direkt ändern. Für Bc geben Sie den gewünschten Bildschirmcode des Zeichens ein (Seite 133 im C64-Handbuch), gefolgt von den acht Werten W0 bis W7.

Wollen Sie im Zeichensatz 2 ein Zeichen ändern, können Sie dies erreichen mit:

SYS 2144,Bc,W0,W1,W2,W3,W4,W5,-W6,W7

Die Erweiterung liegt wie üblich am BASIC-Anfang. Vor dem Abspeichern Ihrer Experimente sollten Sie unbedingt vorher POKE 44,8 eingeben, da sonst nur der BASIC-Teil abgespeichert werden würde. Frank Börnke/RH

Raumschiffe und Pyramiden

Pyramidon: Spiel mit Stapeln

Sie als Astronom haben bemerkt, daß in kurzer Zeit ein riesiger Komet die Erde treffen wird. Genau an der Stelle, wo Ihre Pyramide steht. Sie wissen nicht mehr, wie Sie die Pyramide retten sollen. Doch da kommt Ihnen ein außerirdisches Wesen mit seinem Raumschiff aus dem All zu Hilfe. Sie müssen diesem Wesen mittels Joystick oder Tastatur die richtigen Befehle geben, damit es sein Raumschiff dementsprechend steuert.

Diese Befehle lauten: Dezimalpunkt (.) für links, Schrägstrich (/) für rechts, die Taste A für rauf und Taste Z für runter. Die Leertaste entspricht dem Feuerknopf. Oder verwenden Sie statt dessen den Joystick in Port zwei.

Die Sonne versinkt langsam am Horizont. Die Schatten der Palmen werden länger. Auch der weiße Sand kühlt langsam aus. Sie sitzen in der Nähe einer Pyramide in Ihrem Zelt. Je dunkler es wird, desto näher kommt Ihre Zeit. Ein Löwe schleicht einsam um das Lager und schreckt Sie aus Ihren Gedanken. Aber nicht der Anblick des Löwen macht Ihnen Angst.

Wiederaufbau . . .

Nach dem Titelbild können Sie mit der F1-Taste den Schwierigkeitsgrad auswählen. Die kleinste Pyramide besteht aus drei, die mittlere Pyramide aus vier und die große Pyramide aus fünf Plattformen. Wählen Sie eine davon aus, und betätigen anschließend die Leertaste oder den Feuerknopf.

Zwischen zwei Palmen sehen Sie die von Ihnen ausgewählte Pyramide. Sie müssen versuchen, diese an der jetzigen Stelle Stück für Stück ab- und an einer anderen Stelle (links oder rechts daneben) wieder aufzubauen. Nehmen Sie Beispielsweise die kleine Pyramide. Heben Sie den obersten Stein mit dem Raumschiff ab, und stellen ihn auf die linke Seite. Fahren Sie wieder zur Mitte, holen den nächsten Stein und stellen ihn auf die rechte Seite. Jetzt gehen Sie wieder ganz nach links und holen den kleinen Stein. Diesen stellen Sie auch auf die rechte Seite. Auf die nun frei gewordene linke Seite wird der noch in der Mitte verbliebene größte Stein gesetzt. So werden alle drei Steine hin und her gesetzt, bis die Pyramide an einer anderen Stelle wieder vollkommen aufgebaut ist.

. . . mit Stolpersteinen

Zu beachten ist dabei, daß kein großer Stein auf einem kleinen zu liegen kommt. Das würde den unteren kleinen Stein zerdrücken und ein Teil der Pyramide wäre unwiederbringlich zerstört.

Am unteren Bildschirmrand wird angezeigt, wieviel Zeit Ihnen noch bis zum Aufschlag des Kometen bleibt. Bis dahin sollten Sie die Pyramide an einem anderen Platz stehen haben. Der Komet schlägt erbarmungslos zu und zerstört alles, was ihm im Weg steht. kfp

Redaktionelle Mitteilung

Betrifft: Englische Grammatik. Aus technischen Gründen fällt diese allseits beliebte Serie in dieser Ausgabe aus. Die für dieses Mal geplante Folge mit Pronomina oder, für Nicht-Lateiner, „Fürwörtern" erscheint in der nächsten Ausgabe.

Tips zur Assemblerprogrammierung

Assembler als Hochsprache

Teil 2: Struktur statt Spaghetti

Parallel zum Maschinensprachekurs begann diese Artikelfolge, die Anfängern und Fortgeschrittenen Hilfen zur Erstellung überschaubarer Assemblerpogramme gibt. Nach dem Thema „Symbole statt Zahlensalat" in Ausgabe 3/87 geht es diesmal um Struktur und Modularität. Überlegungen, die keineswegs nur Assemblerprogrammierung betreffen, sondern sich im Prinzip auf alle Programmiersprachen übertragen lassen.

Vor Zeiten war das möglichst virtuose Eintippen von Hexdumps unabdingliches Kennzeichen für den guten Programmierer; den, der auch die Bedeutung der hinterletzten Bits in seinem Rechner im Schlaf aufsagen konnte. Heutzutage gilt derlei als hinterwäldlerisch, man operiert im Assembler[1] mit Symbolen und Mnemonics, *Qualitätsmerkmale eines Programms* sind Überschaubarkeit, Lesbarkeit und Wartbarkeit. Symbole waren letztes Mal dran, Wartbarkeit ist eigentlich übergreifendes Motto der ganzen Serie. Mit diesem wenig verbreiteten Begriff ist übrigens die Möglichkeit gemeint, daß ein Programm auch von anderen als dem Programmierer selbst gewartet, also gegebenenfalls verändert, von Fehlern bereinigt oder anderen Anforderungen angepaßt werden kann.

Struktur und Modularität sind in den letzten Jahren in Mode gekommen, im Gegensatz zu anderen Moden zu Recht. Worum geht es dabei eigentlich? Nehmen wir uns ein relativ simples Programmierproblem, an dem wir konkret werden können. Stellen Sie sich vor, Sie haben eine Diskette voller Texte, die alle mit einer bestimmten Textverarbeitung erstellt wurden. Kürzlich haben Sie die in allen Fachzeitschriften gepriesene brandneue Textverarbeitung XYZ erstanden. Dummerweise kennt diese nicht das interne Ablageformat Ihrer vorhandenen Texte, also muß gewandelt werden. Solche Wandlerprogramme, die beispielsweise den Code für deutsche Umlaute verändern, werden „Filter" genannt, und so ein Filter soll programmiert werden.

Die — häufigere — Methode, an dieses Problem heranzugehen, ist, sich an den Rechner zu setzen und alles Erforderliche hintereinanderweg zu programmieren. Dies ist, milde ausgedrückt, unklug. Der prinzipielle Programmablauf ist dann nur überschaubar, wenn man den gesamten Programmtext Zeile für Zeile verfolgt; jede eventuell notwendige Änderung hat unüberschaubare Folgen für das Gesamtprogramm.

Die bessere Vorgehensweise ist, ein Programm „von oben nach unten" (Neudeutsch: Top-Down) zu entwikkeln und in möglichst kleine Funktionsblöcke aufzuteilen. Diese Funktionsblöcke erledigen abgegrenzte, nachvollziehbare Aufgaben unter klar definierten Bedingungen.

Bei näherer Betrachtung kann man nämlich die Aufgaben des besagten Filters klar unterteilen. Zum Programmanfang sollte eine kleine Meldung ausgegeben werden, vielleicht auch Hintergrund- und Rahmenfarbe gesetzt werden. Da die zu wandelnden Texte auf Diskette vorliegen, ist der nächste Schritt das Lesen der Daten aus der Ursprungsdatei in den Speicher. Anschließend werden die Daten gewandelt und in die Zieldatei geschrieben. Freundlicherweise wird auch das Ende des Programmlaufs dem Benutzer mit einer entsprechenden Meldung kundgetan. Somit ergeben sich fünf Aufgabenbereiche — Anfangsmeldung, Text lesen, Text wandeln, Text schreiben, Endebehandlung.

Das praktische Ergebnis ist in Listing 1a zu sehen. Das Hauptprogramm besteht aus sechs Zeilen, und es ist auf einen Blick zu erkennen, was das Programm alles tut und wie es untergliedert ist. Das läßt sich in BASIC (fast) genauso schön machen, statt der Namen der Unterprogramme werden nur kommentierte Zeilennummern eingeführt (Listing 1b).

Jeder der Funktionsblöcke kann jetzt weiter aufgegliedert werden. INIT ist sehr kurz und wird nicht weiter unterteilt (Listing 2). Voraussetzung für eine fehlerfreie Assemblierung ist na-

```
;-------------------------------------
;hauptprgramm
:begin jsr init      ;meldung ausgeben
       bcs ende      ;fehler, raus
       jsr lies      ;file/alt lesen
       bcs ende      ;war nichts
       jsr wandeln   ;daten wandeln
       bcs ende      ;
       jsr schreib   ;file/neu schreib
:ende  jsr endrout   ;ende melden
       rts           ;fertig
;---ende des hauptteils--------------
```

Listing 1a: Viel länger sollte der Hauptteil eines Programms nicht sein. Die Einzelheiten erledigen Unterprogramme.

```
1000 rem hauptteil
1010 gosub 10000:rem meldung ausgeben
1020 gosub 20000:rem file/alt lesen
1030 gosub 30000:rem daten wandeln
1040 gosub 40000:rem file/neu schreiben
1050 gosub 50000:rem ende melden
1060 end         :rem das wars
1090 rem ende hauptteil---
```

Listing 1b: Auch in BASIC kann überschaubar programmiert werden.

```
;init: farben setzen,
;      meldung an user ausgeben
;veraendert accu und y-register
;texte fuer init:
;gruss b 13,"Der File-Filter",13,0
:init  lda #farbe    ;richtige farbe
       sta bgrund    ;auf schirm
       sta hgrund    ;und hintergrund
       lda #<gruss   ;akku und
       ldy #>gruss   ;y-reg. laden
       jsr cprint    ;fuer textausgabe
       rts
;---ende von init--------------------
```

Listing 2: Die Anfangs-Initialisierung

türlich, daß FARBE, BGRUND und auch alle weiteren Symbole in den Listing-Auszügen am Programmanfang deklariert wurden. (Näheres dazu war in der letzten Folge zu erfahren. Die Namen der wichtigsten Commodore-Sytemroutinen finden Sie in Tabelle 1.)

Das Unterprogramm INIT ist trotz einer gewissen Trivialität ein gutes Beispiel. Zum einen, was die Kommentierung angeht — da die Prozessor-Register nicht gerettet werden, ist dokumentiert, welche Register dieseSubroutine verändert. Zum anderen, was die Länge angeht. Das Listing von INIT paßt vollständig auf eine Bildschirmseite. Diese Faustregel, Unterprogramme nicht länger als eine Bilschirmseite zu gestalten, muß man allerdings beim C64 wegen des 40-Zeichen-Bildschirms leider allzuoft ganz undogmatisch umgehen.

Der Funktionsblock LIES (Listing 3) ist aus mehreren Gründen weiter unterteilt. Der Benutzer muß einen File-Namen eingeben können, eine entsprechende Routine wird auch später für das Ziel-File benötigt. Deswegen wird daraus das Unterprogramm NAMASK.

Wir setzen voraus, daß NAMASK in einem Buffer an der symbolischen Adresse *FILNAM einen String hinterlegt, der* als File-Name gelten soll, und in einer anderen Speicherzelle (FNLEN) die Länge dieses Namens hinterläßt. In BASIC hieße das, beispielsweise einer Variablen namens FI$ diesen Namen zuzuweisen. Es soll hier nicht weiter interessieren, wie das im einzelnen funktioniert. Thema der Erörterung ist ja nicht „Wie realisiere ich eine Eingabe?" (das kommt demnächst in der Assembler-Schule), sondern Programm-Struktur und Modularität.

Nach NAMASK wird FOPEN aufgerufen. Die Routine FOPEN öffnet das File, dessen Name in NAMASK vom Benutzer eingegeben wurde. FOPEN soll etwas näher betrachtet werden. Der modulare (zu deutsch: aufgeteilte) Aufbau von Programmen hat neben dem Vorteil der größeren Überschaubarkeit vor allem den Sinn, sich mit der Zeit eine Sammlung fertiger Unterprogramme für immer wieder erforderliche Aufgaben anzulegen. Das Öffnen einer Floppy-Datei zum Lesen oder Schreiben ist so ein Standardproblem.

Wenn Sie einen Blick auf Listing 4 werfen, werden Sie leicht selbst die verschiedenen Anforderungen bemerken, die solch eine „Library-Routine" erfüllen muß: Sie darf die Prozessor-Register nicht verändern, deswegen werden Akku, X- und Y-Register zu Anfang der Routine auf den Stack gerettet und nach getaner Arbeit wieder restauriert. Um möglichst flexibel einsetzbar zu sein, sind alle benötigten Parameter vom Namen des Files bis zur Geräteadresse „gelabelt". In der Assemblerprogrammierung heißt das, daß diesen Symbolen im Zuweisungsteil Werte zugewiesen wurden. In BASIC würde man statt dessen Variablen verwenden, allerdings gibt das Beispiel für BASIC nicht viel her, da sich das ganze Unterprogramm durch einen einzigen Befehl ersetzen läßt
(OPEN EIN,DEV,SEK,FI$).

```
;-----------------------------------
;lies: quellfile oeffnen und daten
;      in den speicher lesen
;      aendert alle register
:lies  jsr namask    ;namen holen
       jsr fopen     ;file oeffnen
       bcc filok     ;gutgegangen
       jsr errmess   ;fehler meldung
       jsr ask       ;'weiter' fragen
       cmp #'j       ;'j' ist nochmal
       beq lies      ;versuchen
       bne filerr    ;fataler fehler
:filok jsr read      ;daten einlesen
       jsr fclose    ;file schliessen
       clc           ;ok uebergeb
       bcc filend    ;unbedingt
:filerr sec          ;fehler melden
:filend rts
;---ende von lies--------------------
```

Listing 3: Das Unterprogramm LIES verwaltet auch den Umgang mit I/O-Fehlern.

```
;-----------------------------------
;hauptprgramm
:begin jsr init      ;meldung ausgeben
       jsr lies      ;file/alt lesen
       jsr wandeln   ;daten wandeln
       jsr schreib   ;file/neu schreib
       jsr endrout   ;ende melden
       rts           ;fertig
;---ende des hauptteils--------------
```

Listing 4: So geht das Hauptprogramm auf übergebene Fehler ein.

Im Kommentar zu FOPEN ist außerdem zu lesen, daß im Carry eine Mitteilung über Erfolg oder Mißlingen übergeben wird. Dies ist die übliche Art und Weise, diese Information an die aufrufende Routine zu übergeben. Daß das IO-Handlung beim C64 seine Tücken hat, was die Feststellung von Fehlern angeht, ist aus dem Listing ersichtlich. Die Routine zum Auslesen des Fehlerkanals (DEREAD) wurde ebenfalls modularisiert, da man diese wahrscheinlich öfter braucht.

Diese Methode, Fehler nach „oben" zu übergeben, findet auch in der übergeordneten Routine LIES Anwendung. Auf das gesetzte Carry hin wird dem Benutzer der Fehler mitgeteilt. Er kann dann den Versuch wiederholen, zum Beispiel einen neuen File-Namen eingeben. Tut er dies nicht, wird der Fehler an die Hauptroutine übergeben. Listing 5 zeigt die endgültige Version des Hauptprogramms; nach jeder Routine wird abgefragt, ob ein Fehler aufgetreten ist. Wenn ja, wird direkt die Endroutine angesprungen. Nach INIT hat diese Abfrage eher einen theoretischen Grund, was sollte da schiefgehen? Hinter SCHREIB ist die Auswertung des Carry-Flags nicht notwendig, da sich ENDROUT direkt dahinter anschließt.

Daran läßt sich auch die unterschiedliche Behandlung von behebbaren und sogenannten „fatalen" Fehlern studieren. Ein behebbarer Fehler wäre beispielsweise die Angabe eines falschen Dateinamens durch den Benutzer, die zur „File not found"-Meldung führt. Erinnert er sich nach dem Hinweis durch die Routine ASK an den richtigen Namen und gibt ein J ein (etwa auf die Frage „Versuch wiederholen?"), kann dieser Fehler innerhalb des Funktionsblocks behoben werden. Ist dies nicht möglich, wird ein nicht behebbarer, also „fataler" Fehler angenommen. Der Fehler wird nach „oben" weitergereicht, und das Hauptprogramm verzweigt daraufhin zur Endebehandlung.

Beachtenswert ist noch ein wichtiger Gesichtspunkt bei der Programmierung von Unterprogrammen: Es gibt nur einen Ein- und einen Ausgang! Programme, die diese Bedingung nicht erfüllen, verdienen nicht die Bezeichnung strukturiert. Routinen mit mehreren Ausgängen sind nämlich mit vertretbarem Aufwand nicht mehr kontrollierbar. Bei allen hier abgedruckten Unterprogrammen könnte statt des abschließenden RTS (beziehungsweise des RETURNS in BASIC) ein BRK-Befehl stehen (BASIC: STOP), der den Programmlauf unterbricht und eine Auswertung aller Variablen und Register gestattet. Verboten gehört deswegen folgende BASIC-Konstruktion, bei der Zeile 10000 der Beginn eines Unterprogramms ist:

```
10000 IF A=0 THEN RETURN
. . . (Rest des Unterprogramms)
10900 RETURN
```

```
; systemadressen
:stat     = $90 ;i/o-status
;
;rom-routinen
;
:cprint   = $ab1e ;string-ausgabe
:setfls   = $ffba ;file param. setzen
:setnam   = $ffbd ;file namen setzen
:open     = $ffc0 ;file oeffnen
:close    = $ffc3 ;file schliessen
:chkin    = $ffc6 ;eingabe aus datei
:ckout    = $ffc9 ;ausgabe auf datei
:clrch    = $ffcc ;eingabe auf standart
:basin    = $ffcf ;zeichen holen bis CR
:bsout    = $ffd2 ;zeichen ausgeben
:load     = $ffd5 ;file laden
:save     = $ffd8 ;file speichern
:get      = $ffe4 ;zeichen holen
:setcur   = $fff0 ;cursor neu setzen
```

Tabelle 1: Die Zuweisungen für die wichtigsten C64-Systemadressen gehören an den Anfang jedes Assemblerprogramms.

```
;fopen: library-routine zum oeffnen
;       eines files. name wird in
;       FILNAM erwartet, laenge des
;       namens in FNLEN, DEVICE und
;       EIN muessen versorgt sein
;       in Carry wird erfolgsmeldung
;       zurueckgegeben
:fopen pha              ;akku retten
       txa              ;x-register
       pha              ;retten
       tya              ;y-register
       pha              ;retten

       lda #ein         ;kanal
       ldx #device      ;geraet
       ldy #sek         ;u. sek.adrr.
       jsr setfls       ;setzen
       lda fnlen        ;laenge und
       ldx #<filnam     ;adresse des
       ldy #>filnam     ;filenamens
       jsr setnam       ;uebergeben
       jsr open         ;file oeffnen
       bcc fcnt1        ;floppy ist da
       ;fehler aufgetreten
       jsr fclose       ;file schliessen
       sec              ;fehlerflag setzen
       jmp fend         ;und raus

:fcnt1 jsr deread       ;fehlerkanal lesen
     ;setzt carry nach ergebnis

:fend  pla              ;vom stack
       tay              ;y zurueck
       pla              ;dasselbe
       tax              ;mit x-register
       pla              ;und akku
       rts              ;fertig
;--ende von fopen----------------------
```

Listing 5: FOPEN ist eine typische Bibliotheks-Routine

```
;-------------------------------------
;endrout: endemitteilung mit erfolgs-
;         meldung.
:endrout bcc isgood   ;Carry=0 ist ok
         jsr errmess  ;fehler ausgeben
         jmp endend   ;raus
:isgood  jsr endok    ;'ok' melden
:endend  rts          ;fertig
;--ende von endrout--------------------
```

Listing 6: In ENDROUT wird abhängig vom Zustand des Carry-Flags eine entsprechende Meldung ausgegeben.

Um dieses sogenannte Unterprogramm zu testen, müßten schon an mindestens zwei Punkten Break points gesetzt werden — falls Ihnen zwei Wochen nach der Programmierung das RETURN in Zeile 10000 noch auffällt. Sonst setzt das große Staunen ein, warum das getestete Programm nur gelegentlich definiert abbricht. Das alles gilt für die Assemblerprogrammierung umso mehr, weil das Austesten von Maschinenprogrammen ohnehin kein Zuckerschlecken ist.

Die letzte Betrachtung gehört der Routine zur Endebehandlung (ENDROUT). Eingangs wurde erwähnt, daß dort eine Meldung ausgegeben werden sollte. Die Endroutine ist aber auch der geeignete Ort, die erwähnten fatalen Fehler zu behandeln; genauer gesagt, dem Benutzer mitzuteilen. Da in dieser Routine möglicherweise auch eine Aussage über die Art des aufgetretenen Fehlers denkbar wäre — die diversen Unterprogramme müßten sich dann darüber „verständigen", wie eine Fehlernummer oder dergleichen übergeben wird —, ist der Programmteil zum Vermelden des Mißerfolgs ausgegliedert worden. Listing 6 zeigt das endgültige ENDROUT.

Zum Schluß sei noch bemerkt, daß die Summe der abgedruckten Listings keinen vollständigen Filter ergibt.[2] Ziel dieses Artikels ist nicht das Abtippen von Listings, sondern ein gewisses Verständnis für die Regeln strukturierter Programmierung. JS

1) Es wird also vorausgesetzt, daß Sie im Besitz eines der zahlreich vorhandenen 6502-Assembler für den C64 sind. Einer der derzeit schnellsten Assembler wurde in INPUT 6/86 veröffentlicht; die Beschreibung spezieller Assemblerbefehle bezieht sich auf den INPUT-ASS.

2) Es fehlen die Routinen WANDELN, SCHREIB, NAMASK, ERRMESK, ASK, READ, FCLOSE, DEREAD, ENDOK; die Zuweisungen an FARBE, BGRUND, HGRUND, EIN, DEVICE, SEK; die Buffer für FILNAM und FNLEN.

Hinweise zur Bedienung

Bitte entfernen Sie eventuell vorhandene Steckmodule. Schalten Sie vor dem Laden von INPUT 64 Ihren Rechner einmal kurz aus. Geben Sie nun zum Laden der Kassette

LOAD und RETURN

beziehungsweise bei der Diskette

LOAD"INPUT",8,1 und RETURN

ein. Alles weitere geschieht von selbst.

Sollten Sie ein Laufwerk haben, das zusammen mit dem Schnellader der Diskettenversion Schwierigkeiten macht, geben Sie bitte ein

LOAD"LADER*",8,1 und RETURN.

Nach der Titelgrafik springt das Programm in das Inhaltsverzeichnis des Magazins. Dieses können Sie nun mit SPACE (Leertaste) durchblättern. Mit RETURN wird das angezeigte Programm ausgewählt und geladen. Im Fenster unten rechts erhalten Kassetten-Besitzer weitere Hinweise („Bitte Band zurückspulen" und so weiter ...).

Haben Sie bei der Auswahl eines Programms eventuell nicht weit genug zurückgespult und es wurde nicht gefunden, spulen Sie bis zum Bandanfang zurück.

Auf der zweiten Kassetten-Seite befindet sich eine Sicherheitskopie. Sollten Sie eventuell mit einem Programm Ladeschwierigkeiten haben, versuchen Sie es auf der zweiten Seite. Führt auch dies nicht zum Erfolg, lesen Sie bitte die entsprechenden Hinweise im Kapitel „Bei Ladeproblemen"!

Neben der Programmauswahl mit SPACE und dem Ladebefehl mit RETURN (im Inhaltsverzeichnis) werden die übrigen 'System-Befehle' mit der Kombination aus CTRL-Taste und einem Buchstaben eingegeben. Sie brauchen sich eigentlich nur CTRL und H zu merken (Aufruf der Hilfsseite), denn dort erscheinen die jeweils möglichen 'System-Befehle'. Nicht immer sind alle Optionen möglich (eventuell werden Sie zu Beginn des Programms auf Einschränkungen hingewiesen). Hier nun alle INPUT-64-Systembefehle:

CTRL und Q

Sie kürzen die Titelgrafik ab; INPUT 64 geht dann sofort ins Inhaltsverzeichnis.

CTRL und H

Es wird ein Hilfsfenster angezeigt, auf dem alle verfügbaren Befehle aufgeführt sind.

CTRL und I

Sie verlassen das Programm und kehren in das Inhaltsverzeichnis zurück.

CTRL und F

Ändert die Farbe des Bildschirm-Hintergrundes (auch im Inhaltsverzeichnis erreichbar)

CTRL und R

Ändert die Rahmenfarbe (auch im Inhaltsverzeichnis erreichbar).

CTRL und B

Sie erhalten einen Bildschirmausdruck — natürlich nicht von Grafikseiten oder Sprites! Angepaßt ist diese Hardcopy für Commodore-Drucker und kompatible Geräte. Das Programm wählt automatisch die richtige Geräteadresse (4,5 oder 6) aus.

CTRL und S

Wenn das Programm zum Sichern vorgesehen ist, erscheinen weitere Hilfsfenster. Sie haben die Wahl, ob Sie:

im Commodore-Format C
im SuperTape-Format S
auf Diskette D

sichern wollen. Beachten Sie bitte, daß Sie die Programme von Ihrem Datenträger immer als normale BASIC-Programme mit LOAD"NAME",1 bzw. LOAD"NAME",8 laden müssen. Wenn Sie das Programm im SuperTape-Format aus INPUT 64 abgespeichert haben, müssen Sie vor dem Laden selbstverständlich Super-Tape in Ihren Rechner geladen und initialisiert haben. (SuperTape DII haben wir in der Ausgabe 4/85 veröffentlicht.) Außerdem wird in diesem Fenster die Programmlänge in Blöcken angegeben. Kassetten-Benutzer können diese Disketten-Blockzahl nach folgender Faustregel umrechnen: Im Commodore-Format werden pro Minute neun Blöcke abgespeichert, SuperTape schreibt die gleiche Anzahl von Blöcken in cirka sechs Sekunden aufs Band.

Bei Ladeproblemen

Diskette: Bei nicht normgerecht justiertem *Schreib-/Lesekopf oder bei bestimmten* Serien wenig verbreiteter Laufwerke (1570) kann es vorkommen, daß das im INPUT-Betriebssystem eingebaute Schnelladeverfahren nicht funktioniert. Eine mögliche Fehlerursache ist ein zu geringer Abstand zwischen Floppy und Monitor/Fernseher. Das Magazin läßt sich auch im Normalverfahren laden, eventuell lohnt sich der Versuch:

LOAD"LADER",8,1

Sollte auch dies nicht zum Erfolg führen, senden Sie bitte die Diskette mit einem kurzen Vermerk über die Art des Fehlers und die verwendete Gerätekonstellation an den Verlag (Adresse siehe Impressum).

Kassette: Schimpfen Sie nicht auf uns, die Bänder sind normgerecht nach dem neusten technischen Stand aufgezeichnet und sorgfältig geprüft. Sondern: Reinigen Sie zuerst Tonköpfe und Bandführung Ihres Kassettenrecorders. Die genaue Vorgehensweise ist im Handbuch der Datassette beschrieben. Führt auch dies nicht zum Erfolg, ist wahrscheinlich der Tonkopf Ihres Gerätes verstellt. Dieser Fehler tritt leider auch bei fabrikneuen Geräten auf.

Wir haben deshalb ein Programm entwickelt, mit dessen Hilfe Sie den Aufnahme-/Wieder gabekopf justieren können. Tippen Sie das Programm JUSTAGE ein und speichern Sie es ab. Dieses Programm wertet ein etwa 30 Sekunden langes Synchronisationssignal aus, das sich am Ende jeder Kassettenseite befindet. Starten Sie das JUSTAGE-Programm mit RUN, jetzt sollte die Meldung PRESS PLAY ON TAPE kommen, drücken Sie also die PLAY-Taste. Nach dem Drücken der Taste geht der Bildschirm zunächst wie immer aus. Wird das Synchro-Signal erreicht, wechselt die Bildschirmfarbe, und zwar — bei nicht total verstellter Spurlage — völlig regelmäßig etwa dreimal pro Sekunde. Liegt die Spur des Tonkopfes grob außerhalb der zulässigen Toleranzgrenzen, geschieht entweder nichts, oder die Farben wechseln unregelmäßig. Nehmen Sie jetzt einen kleinen Schraubenzieher und werfen Sie einen Blick auf Ihre Datassette. Über der REWIND-Taste befindet sich ein kleines Loch. Wenn Sie bei gedrückter PLAY-Taste durch dieses Loch schauen, sehen Sie den Kopf der Justierschraube für die Spurlage. Drehen Sie diese Einstellschraube. Aber Vorsicht: ganz langsam drehen, ohne dabei Druck auszuüben! Drehen Sie die Schraube nicht mehr als eine Umdrehung in jede Richtung. Nach etwas Ausprobieren wird der Bildschirm gleichmäßig die Farbe wechseln. Zur Feinabstimmung lassen Sie das Synchro-Signal noch einmal von Anfang an laufen. Die Schraube jetzt nach links drehen, bis der Farbwechsel unregelmäßig wird. Diese Stellung genau merken, und die Schraube jetzt langsam wieder nach rechts drehen: Der Farbwechsel wird zunächst gleichmäßig, bei weiterem Drehen wieder unregelmäßig. Merken Sie sich auch diese Stellung, und drehen Sie die Schraube nun in Mittelstellung, das heißt zwischen die beiden Randstellungen. Denken Sie daran, daß während der Einstellung kein Druck auf den Schraubenkopf ausgeübt werden darf! Der Tonkopf Ihres Recorders ist jetzt justiert.

Sollte sich auch nach dieser Einstellung INPUT 64 nicht laden lassen, erhalten Sie von uns eine Ersatzkassette. Schicken Sie bitte die defekte Kassette mit einem entsprechenden Vermerk an den Verlag ein (Adresse siehe Impressum).

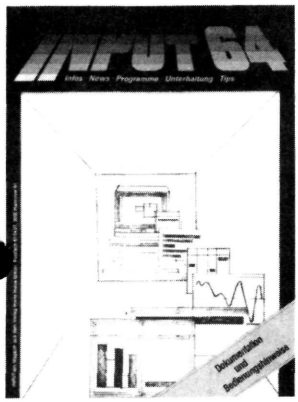
PS! In der Ausgabe 6/85 haben wir das Programm RECORDER-JUSTAGE veröffentlicht, das die Einstellung des Datenrecorders zum Kinderspiel macht.

Listing Justage

```
800 for i=49199to49410:read d:ps=ps+d:poke i,d:next
900 ifps<>24716thenprint"falsch abgetippt - fehler korrigieren!":end
950 print"o.k."
970 sys49338
1000 rem von 49199 bis 49410
1010 data173, 13,220,169,217,174,  4,220,172,  5,220,141, 14,220, 48, 44, 56
1020 data102, 88, 36, 89, 48, 12,144, 10,165, 88,133, 90,169,128,133, 88,133
1030 data 91,192,121,144,  4,224,115,176,  7,169,  0,133, 92, 56,176, 11,165
1040 data 92, 73,128,133, 92, 36, 92, 16, 19, 24,102, 88, 36, 89, 48, 12,144
1050 data 10,165, 88,133, 90,169,128,133, 88,133, 91,104,168,104,170,104, 64
1060 data 96, 36, 91, 16,252,132, 91,165, 90, 96,160,128,132, 89,165, 88,201
1070 data 22,208,250,132, 88,160, 10,132, 89,132, 91, 36, 91, 16,252,132, 91
1080 data165, 90,201, 22,208,226,136,208,241, 32,133,192,201, 22,240,249, 96
1090 data 32,147,252,120, 32, 23,248,165,  1, 41, 31,133,  1,133,192,169, 47
1100 data141, 20,  3,169,192,141, 21,  3,169,127,141, 13,220,169,144,141, 13
1110 data220,173, 17,208, 41,239,141, 17,208,169, 70,141,  4,220,169,129,141
1120 data  5,220, 88, 32,142,192,201, 42,208,249,173, 32,208, 41, 15,168,200
1130 data140, 32,208, 76,237,192,208, 76

ready.
```

Am 1. Juni 87 auf Diskette und Kassette an Ihrem Kiosk: INPUT64, Ausgabe 6/87



Wir bringen unter anderem:

INPUT-SCE

Ein Sprite- und Zeichensatz-Editor in einem. Dieses Werkzeug bietet selbstverständlich Austausch-, Kopier- und Simulationsfunktionen.

Metal Ball

Eiskalt (wie Metall) müssen Sie reagieren, um bei diesem Spiel nicht immer wieder von vorn zu beginnen. Kugeln, Würfel, Bälle und andere Dinge gilt es bei diesem Spiel zu zerstören oder wie beim Tennis zu retournieren.

CD-Manager

Der C64 hält auch mit der Lasertechnik Schritt, denn der CD-Manager verwaltet problemlos Ihre Compact-Disketten. Von Klassik bis Pop.

und außerdem:

Englische GRAMmatik Teil 6, 64er Tips, Assembler-Schule Teil 4 u.v.a.m.

Das aktuelle c't-Projekt

16-Bit-Power im C64

Der Prozessor 65C816 ist eine Weiterentwicklung der bewährten 6502-CPU mit erweitertem Befehlssatz und 16 Bit breiten Registern. Da er auch 6502-Programme ausführen kann, ist er der ideale Prozessor, um den betagten C64 in neue Leistungs-Dimensionen vorstoßen zu lassen. Mit einer Taktfrequenz von 4 MHz bewältigt das 65C816-Projekt die C64-Software mit vierfacher Geschwindigkeit und eignet sich bei einem Grundausbau von 192 KByte RAM auch als professionelles Entwicklungssystem für 16-Bit-Software. Denn selbstverständlich gehört auch ein Assembler für den 16-Bit-Befehlssatz mit dazu.

elrad–Magazin für Elektronik

Ausgabe 5/87–ab 27.4.1987 am Kiosk

Hochfrequenz-Praxis: Hf-Baukasten mit AM- und FM-Demodulation * Schaltungstechnik aktuell: SENSFET—ein neuer MOS-Leistungstransistor * Bauanleitung: UKW-Frequenzanzeige „Skalen-Sandwich" * Wochenendprojekt: Türklingel mit Telefon-Sound * Bauanleitung: MIDI to Drum * u.v.a.m.

c't–Magazin für Computertechnik

Ausgabe 6/87–ab 14.5.1987 am Kiosk

Projekte: 4-MHz-65C816-Karte für C64 * RGB-FBAS-Wandler * Software-Know-how: Spline-Interpolation * Reportage: Stand der Expertensysteme * Programme: Lösen von Redox-Gleichungen * Kontrollprogramm für Drucker * u.v.a.m

IMPRESSUM:

INPUT 64
Das elektronische Magazin

Verlag Heinz Heise GmbH
Bissendorfer Straße 8
3000 Hannover 61
Postfach 61 04 07
3000 Hannover 61
Telefon: (05 11) 53 52 - 0

Technische Anfragen:
nur dienstags von 9.00 — 16.30 Uhr

Postgiroamt Hannover, Konto-Nr. 93 05 - 308
(BLZ 250 100 30)
Kreissparkasse Hannover, Konto - Nr. 000 - 01 99 68
(BLZ 250 502 99)

Herausgeber: Christian Heise

Redaktion:
Christian Persson (Chefredakteur)
Ralph Hülsenbusch
Wolfgang Möhle
Karl-Friedrich Probst
Jürgen Seeger
Redaktionsassistent: Wolfgang Otto

Ständige Mitarbeiter:
Peter S. Berk
Irene Heinen
Peter Sager
Hajo Schulz
Eckart Steffens

Vertrieb: Anita Kreutzer

Grafische Gestaltung:
Wolfgang Ulber, Dirk Wollschläger

Herstellung: Heiner Niens

Lithografie:
Reprotechnik Hannover

Druck:
Leunismann GmbH, Hannover
CW Niemeyer, Hameln

Konfektionierung:
Lettershop Brendler, Hannover

Kassetten- und Diskettenherstellung:
SONOPRESS GMBH, Gütersloh

INPUT 64 erscheint monatlich.
Einpelpreise Kassette DM 16,80
Jahresabonnement Inland Kassette DM 140,—
Diskette DM 198,—
Einzelpreis Diskette DM 19,80

Redaktion, Abonnementverwaltung:
Verlag Heinz Heise GmbH
Postfach 61 04 07
3000 Hannover 61
Telefon: (05 11) 53 52 - 0

Abonnementverwaltung Österreich:
Evb-Verlag GmbH & Co KG
Abt. Zeitschriftenvertrieb
z. Hd. Frau Pekatschek
Amerlingstraße 1
A-1061 Wien
Jahresabonnement: Kassette DM 152,—
Diskette DM 210,—

Vertrieb (auch für Österreich, Niederlande, Luxemburg und Schweiz):
Verlagsunion Zeitschriften-Vertrieb
Postfach 57 07
D-6200 Wiesbaden
Telefon: (0 61 21) 2 66 - 0

Verantwortlich:
Christian Persson
Bissendorfer Straße 8
3000 Hannover 61

ISSN 0177 - 3771

Titelidee: **INPUT 64**
Titelillustration: Michael Thiele, Dortmund
Titel - Grafik und -Musik: Tim Pritlove, Fabian Rosenschein
Bertiebssystem: Hajo Schulz

hier abtrennen

INPUT 64-Abonnement Abruf-Coupon

Ja, übersenden Sie mir bis auf Widerruf alle künftigen INPUT64-Ausgaben ab Monat

_____

(Kündigung ist jederzeit mit Wirkung ab der jeweils übernächsten Ausgabe möglich. Überzahlte Abonnementsgebühren werden sofort anteilig erstattet.)

Das Jahresabonnement kostet: ○ auf Kassette DM 140,— inkl. Versandkosten und MwSt.

○ auf Diskette DM 198,— inkl. Versandkosten und MwSt.

(Bitte ankreuzen/Nichtzutreffendes streichen.)

Absender und Lieferanschrift

Bitte in jedes Feld nur einen Druckbuchstaben (ä = ae, ö = oe, ü = ue)

Vorname/Zuname

Beruf/Funktion

Straße/Nr.

PLZ Wohnort

_____

Datum/Unterschrift

Von meinem Recht zum schriftlichen Widerruf dieser Order innerhalb einer Woche habe ich Kenntnis genommen. Zur Wahrung der Frist genügt die rechtzeitige Absendung.

Unterschrift _____

Bitte beachten Sie, daß diese Bestellung nur dann bearbeitet werden kann, wenn beide Unterschriften eingetragen sind.

INPUT 64-Abonnement Abruf-Coupon

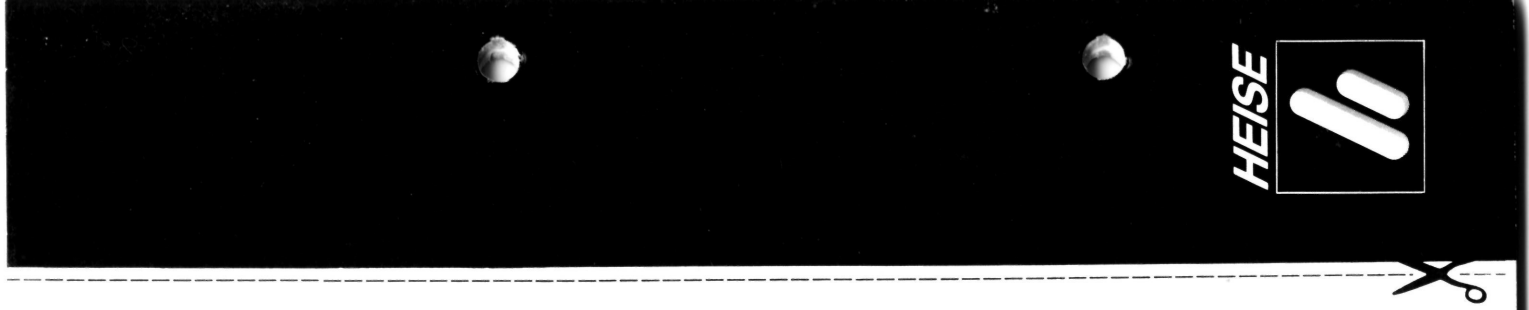
Ich wünsche Abbuchung der Abonnement-Gebühr von meinem nachstehenden Konto. Die Ermächtigung zum Einzug erteile ich hiermit.

Name des Kontoinhabers

Bankleitzahl , Konto-Nr.

Geldinstitut Ort des Geldinstituts

Bankeinzug kann nur innerhalb Deutschlands und nur von einem Giro- oder Postscheckkonto erfolgen.



Bitte im (Fenster-)Briefumschlag einsenden.
Nicht als Postkarte verwenden!

INPUT64

Vertriebsabteilung
Verlag Heinz Heise GmbH
Postfach 61 04 07

3000 Hannover 61