DAS ELEKTRONISCHE MAGAZIN 7/87

INPUT 64

Infos · News · Programme · Unterhaltung · Tips

Modernes für den Bildschirm

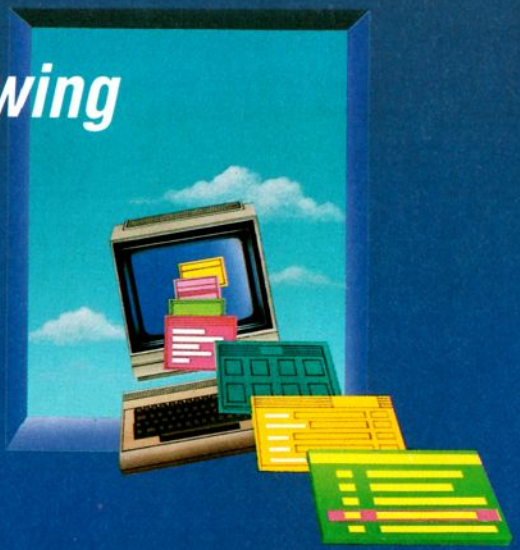
INPUT-Windowing

24 Textfenster gleichzeitig

Traditionelles für die Tastatur

Kommando-Interpreter

File-Handling wie beim PC



Spiele:
Ping-Pong Classic
Spider

Serien:
Assembler-Schule
64er Tips
Englische GRAMmatik

Neues Rätsel:
Schnelle Kreise

ID-Werkstatt

Dokumentation und Bedienungshinweise

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

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

Daß Computer wegen ihres andauernden Preisverfalls keine Wertanlage sind, hat sich wohl mittlerweile herumgesprochen. Trotzdem werden die meisten C64-Besitzer nicht schlecht gestaunt haben, daß ihr Rechner seit Ende Mai in der Ladenkette eines bekannten Billiganbieters für unter 300 DM zu haben ist. Zwar in der „alten" Form (sprich: Brotkasten), aber immerhin mit GEOS. Für den „Neuen" muß ein Hunderter mehr hingeblättert werden.

Für Insider kommt dies nicht überraschend. Commodore ließ auf der CeBIT verlauten, daß man noch einige hunderttausend 64er — neue, versteht sich — produzieren wolle; dazu muß der „Alte" vorher vom Markt sein.

Über die Marktstrategien von Commodore soll hier aber nicht weiter spekuliert werden. Sondern: Diese Niedrigpreis-Politik macht einen ernstzunehmenden Rechner erschwinglich für all die Leute, die sich bislang dessen Anschaffung nicht leisten konnten oder wollten. Also einige zehn- oder eher hunderttausend Wohnstuben und Kinderzimmer mehr, die nicht länger „computerfrei" sein werden.

Dies fällt zusammen mit einem anderen Trend, dem rapiden Preisverfall im Personalcomputer-Bereich. Ein komplett ausgerüsteter PC ist heute für weniger Geld zu haben als die 64er der ersten Stunde. Der Computer entwickelt sich so mehr und mehr zu einem Massenartikel, der ebenso selbstverständlich wird wie eine Kaffeemaschine oder ein Fernsehgerät.

Das hat natürlich in erster Linie etwas mit Geschäft und Umsatz zu tun. Es könnte aber auch einen erfreulichen Nebeneffekt haben: die Versachlichung der Diskussion über die gesellschaftspolitischen Folgen der „Computerisierung". Bekanntlich wird da gern mit den Alternativen „Wunderwerk" contra „Teufelszeug" argumentiert. Beides dürfte man Leuten, die ihren eigenen Rechner zu Hause stehen haben, kaum einreden können. Denn die haben die Chance, sich selbst ein Bild von den Möglichkeiten, aber auch den Grenzen der „Neuen Technologien" zu machen. Insofern könnte man statt von „Versachlichung der Diskussion" auch von „Demokratisierung" reden. Weil mehr Menschen mitdiskutieren können.

Redaktionelle Mitteilung

Wer bislang an seinem Kiosk vergeblich nach der Disketten-Version von INPUT64 gefragt hat, braucht ab Ausgabe 9/87 nicht mehr ersatzweise auf die INPUT-Kassette zurückzugreifen. Wegen der zunehmenden Verbreitung von Disketten-Laufwerken wird die Kassetten-Version ab September eingestellt, statt dessen wird die INPUT-Diskettenversion endlich flächendeckend vertrieben.

Jürgen Seeger

INHALT

Auf einen Blick: INPUT 64 - 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 . . .

IRAs, der MPS803 und die LFs

Ihr in Ausgabe 4/87 veröffentlichter Re-Assembler IRAs spielt nicht mit dem Drucker MPS803 zusammen. Das hängt mit der etwas unschönen Art zusammen, wie das Programm mit den Sekundär-Adressen der geöffneten Ziel-Geräte umgeht. Folgende Änderungen schaffen Abhilfe:

$0CA5 JSR $080D

Die Adresse $080D befindet sich in der BASIC-Zeile des Programms, dort muß folgendes eingegeben werden:

```
$080D   JSR $FFBD
$0810   LDA #04
$0812   CMP $BA
$0814   BNE $081A
$0816   LDA #00
$0818   STA $B9
$081A   RTS
```

Außerdem sollten die Adressen $080A bis $080C mit Nullbytes gefüllt werden. *I. Turski, Düsseldorf*

Zum Thema „Re-Assemblieren und Drucken" können wir auch noch einen Tip beisteuern: Eventuell störende Linefeeds werden durch Beschreiben der Adresse $1934 mit $2C vermieden. (d. Red.)

Volkszählung und Jahrhundertwende

Wenn ich die generierte Datei „künstlicher Bürger" mit Ihrem Auswertungs-Programm aus INPUT 5/87 (die beiden Simulations-Programme zur Volkszählung '87, d. Red.) um die Daten meiner kurz nach der Jahrhundertwende geborenen Großeltern erweitern will, stürzt das Programm mit einer Fehlermeldung ab. Woran liegt das? *(tel. Anfrage)*

Wenn der abgefragte Geburtsjahrgang ein- statt zweistellig angegeben wird, versucht das Programm, dies zu korrigieren. Die dafür eingesetzte Routine arbeitet allerdings nicht korrekt. Dieser Fehler ist leicht zu vermeiden: „01" statt „1" eingeben.

Gelegentliche Schwierigkeiten beim Auswerten einer mit eigenen Daten versehenen Bürgerdatei sind übrigens fast immer darauf zurückzuführen, daß das Programm nicht ordnungsgemäß mit der Taste 'F5' beendet wurde. Dadurch konnte die Datei nicht geschlossen werden, und der letzte Datensatz ist zerstört. Meist ist damit auch die Datei als Ganzes unbrauchbar geworden. (d.Red.)

CAD und Epson CP 80

Die folgende Einstellung gilt für den Epson CP80 mit Görlitz-Interface und ist wie folgt vorzunehmen: Gerätenummer: 04; Sekundäradresse: 04; Codefolge für Grafik an — vor: 1b, 41, 07, 1b, 55, 01, 1b, 4b; Codefolge für Zeilenvorschub: 0d, 0a; Nadelzahl: 08; oberstes Bit: 00; Drehen der Bitfolge: 01; maximale Druckbreite: 01e0. *Michael Schlag*

Sicherlich kann der eine oder andere Leser die Anzahl der angepaßten Drucker erhöhen, indem er uns seine spezifische Anpassung zusendet. Wir werden auch weiterhin alle Einsendungen veröffentlichen. (d. Red.)

Unbekannte Fehlermeldung

. . . erhielt ich die Meldung „missing file name error". Diese Fehlermeldung finde ich im Commodore-Handbuch nicht. *M. Hirsch, Mannheim*

Diese Fehlermeldung taucht in der Tat im deutschsprachigen Handbuch für den C64 nicht auf. Sie bedeutet, daß beim Versuch, ein Programm auf Diskette zu speichern oder von dort zu laden, der File-Name nicht existiert beziehungsweise ein Leer-String ist. Beispielsweise durch die Befehlsfolge:

```
N$="":SAVE N$,8
```

Diese Fehlermeldung tritt nicht beim Lesen von oder Schreiben auf Kassette auf, da für dieses Gerät ein File-Name nicht zwingend vorgeschrieben ist.

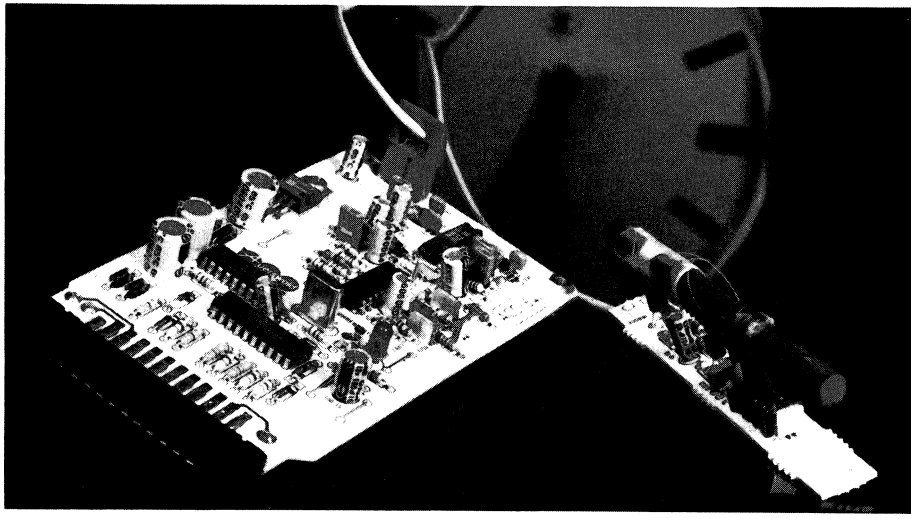
Übrigens fehlt im C64-Handbuch auch die Erklärung der Fehlermeldung „illegal device number". Diese quittiert unmögliche Anforderungen an ein Gerät, etwa

```
LOAD"$",3
```

oder

```
SAVE"TEST",2
```

Und wenn Ihnen mal die Meldung „file data error" über den Weg läuft: unter „bad data" nachsehen! Diese Fehler sind im Handbuch des 128er behoben. (d. Red.)



Hardware Review

Zeitspiele

Funkuhrmodul ACC64 für C64

Die meisten wissen sicherlich, daß der C64 eine sehr genaue Echtzeituhr enthält, doch Laufzeitgenauigkeit ist nur eine Sache. Auf welche Zeit soll man die Uhr eigentlich stellen? Die Firma CONRAD Electronic in Hirschau bietet seit kurzem den Funkuhrzusatz ACC64 an, der zusammen mit einem C64 die offizielle Zeit empfängt. Genauer geht's nimmer!

Genau ist nicht genau genug! Wenn Sie sich darüber ärgern, daß Ihre Armbanduhr ein bis zwei Sekunden im Monat abweicht, lesen Sie weiter. Nicht mehr als eine gute Quarz-Uhr kostet ein Zusatzgerät für den C64, mit dem Sie immer auf die amtliche genaue Zeit zugreifen können.

So wird die Zeit gemacht

Doch zuerst ein bißchen über die Zeit. „Produziert" wird unsere Zeit von der Physikalisch-Technischen Bundesanstalt in Braunschweig. Dort laufen mehrere Atomuhren. Mit Atomkraft oder Radioaktivität haben diese, nebenbei bemerkt, nichts zu tun. Die Zeiten dieser Uhrengruppe werden gemittelt, und so erhält man eine supergenaue Zeit mit einer Abweichung von weniger als einer Sekunde in 300 000 Jahren. Diese Zeit steuert eine weitere Uhr beim Langwellensender DCF77 in Mainflingen bei Frankfurt. Von hier wird die Zeitinformation in kodierter Form ähnlich einer Radiosendung ausgestrahlt. Seit 1978 darf jeder, der mag oder muß, Empfänger für den DCF77 gebürenfrei betreiben und sich die amtliche Zeit angeln.

Anschluß an die Zeit

Das Funkuhrmodul ACC64 besteht aus zwei Platinen, die mit einem 5 m langen Kabel verbunden sind. Auf der großen Platine, die in den Userport gesteckt wird, befindet sich alles, was zur kompletten Auswertung und Bewertung der Zeit benötigt wird. Die zweite Platine trägt die Antenne mit dem Antennenverstärker. Treten beim DCF77-Empfang gelegentliche Störungen auf, ist das unproblematisch, da eine quarzgesteuerte Uhr weiterläuft und die Zeit zur Verfügung stellt.

Im Wahrheit ist es so, daß man eine Quarzuhr hat, die sich automatisch

stellt und immer wieder mit der DCF77-Zeit abgleicht. Dies alles passiert schon allein dadurch, daß das Modul an den C64 angeschlossen ist. Nach etwa 5 Minuten ist das Modul synchronisiert und trägt die Zeit in sich.

Was tun mit der Zeit?

Da das ACC64-Modul keine eigene Anzeige besitzt und man ja zudem seinem Computer die Zeit für Programme zur Verfügung stellen will, braucht man geeignete Software. Die mitgelieferte Diskette enthält mehrere Beispielprogramme, mit denen man Zugang zur Zeit erhält. Am beeindruckendsten ist das Programm „Schnelldemo", in großen Digitalziffern erscheinen Zeit, Datum und Wochentag auf dem Bildschirm. Für Spezialisten werden auch die weiteren Kennbits der Zeit-Kodierung angezeigt. Solch eine Uhr ist zwar ganz nett, doch ist der Rechner zu nichts anderen mehr zu gebrauchen. Will man die Zeit in eigenen Programmen verwenden, hilft das kleine BASIC-Programm „Stelldemo". Dieses stellt die interne C64-Echtzeituhr auf die empfangene Zeit. Das zugehörige Maschinenprogramm legt die vom ACC64 dekodierten Zeitsignale im RAM des Rechners ab. Die Beispielprogramme lassen sich unverändert in eigene Programme einbinden oder vorher entsprechend den eigenen Bedürfnissen anpassen. Alle Maschinenprogramme liegen auch im Assembler-Source-Code vor, der benötigte Assembler ist freundlicherweise gleich mitgeliefert worden.

Zeit im Angebot

Das ACC64 ist ein sauber aufgebautes und zuverlässiges Gerät. Ohne Schwierigkeiten kann man es in Betrieb nehmen. Lange Zeit läßt sich mit dem mitgelieferten Beispielprogrammen spielen, ebenso wird alles Erforderliche für ernsthafte Anwendungen mitgeliefert. Die Diskette enthält in mehreren Text-Files jede Menge Anleitungen, angefangen bei der Hardware-Beschreibung, der Beschreibung der Beispielprogramme bis hin zur Erklärung der Zeitkodierung des DCF77-Senders.

Das Funkuhrmodul ACC64 ist bei der Firma CONRAD Electronic, Postfach 1180, 8452 Hirschau, unter der Bestellnummer 97 57 96 für 129 DM erhältlich. pan

Schnelles Zirkeln

Neues Rätsel

Ein eher „traditionelles" Problem ist diesmal zu lösen: die Entwicklung von kurzen und schnellen Kreis-Algorithmen in BASIC. Zu gewinnen gibt es zwei INPUT64-Jahresabos und zehn Computer-Bücher.

Bei diesem Programmierwettbewerb geht es wieder einmal um zwei Kriterien: Tempo oder Kürze. Doch erst einmal zur Aufgabenstellung.

Gegeben ist ein Zeichenbrett mit einem Koordinatensystem, dessen Nullpunkt in der linken oberen Ecke liegt und dessen Ausdehnung 320 x 200 Punkte (Breite mal Höhe) beträgt; der übliche High-Resolution-Grafikbildschirm des 64ers also. Sie brauchen sich um die Programmierung von Grafikbefehlen keine Gedanken zu machen, zum „Rahmenprogramm" der Aufgabe gehört eine Grafikerweiterung (siehe Kasten „HiRes-Paket als Zugabe"). Das Rahmenprogramm, das mit CTRL-S auf eine eigene Kassette/Diskette abgespeichert werden kann, schaltet auch die hochauflösende Grafik ein und initialisiert die Variablen, die den Kreis beschreiben. Die x-Koordinate ist in der Variablen „X" abgelegt, die y-Koordinate dementsprechend in „Y", der Radius in „R". Um dieses Rahmenprogramm zusammen mit dem Grafikpaket abzuspeichern, muß vor dem SAVE-Befehl im Direktmodus der BASIC-Anfang durch POKE 44,8 zurückgesetzt werden.

Alles im Rahmen

Somit lautet die Aufgabe: Zeichnen Sie unter Verwendung des SET-Befehls (Punktsetz-Befehl) einen Kreis mit dem Mittelpunkt x/y und dem Radius r! Benutzen Sie dazu das Unterprogramm ab Zeile 40000 im Rahmenprogramm, das Sie aus dem Magazin heraus abgespeichert haben; schließen Sie diesen Programmteil mit RETURN ab.

Alles Weitere übernimmt das Rahmenprogramm, wie die Auswertung der Zeit und die Überprüfung der Qualität. Sie können sich entscheiden, ob Sie Ihren Algorithmus unter der Kategorie „Schnell" oder der Kategorie „Kurz" bewerten lassen wollen. Die Bewertung nach Zeit ist wei-

HiRes-Paket als Zugabe

Das Rahmenprogramm zum Rätsel enthält, damit Sie sich nur mit dem Algorithmus beschäftigen müssen, auch ein komplettes Grafikpaket. Es handelt sich dabei um eine sogenannte Programmierer-Version des in Ausgabe 4/85 veröffentlichten HIRESSPEED. Das in diesem Programm benutzte HIRESSPEED enthält die Tools Print At und Inkey und ist innerhalb des INPUT 64-Betriebssystems lauffähig. Die Video-Basisadresse liegt bei $8000 (d 32768), und der Grafikbildschirm beginnt bei $A000 (d 40960), also unter dem BASIC-ROM. Sie können das gesamte Tool abspeichern, indem Sie Folgendes beachten:

Das Rahmenprogramm von Ihrem eigenen Datenträger laden, Programm starten, NEW und eine REM-Zeile eingeben, POKE 44,8 eintippen, abspeichern.

HIRESSPEED-Befehle:

CIRCLE x,y,xr,yr
CLS
COLOR zf,hf
FILL x,y
GRAPH
HIRES zf,hf
LINE x1,y1,x2,y2
MODE mo
NORM
SET x,y
TEXT x,y,b,"text"
GLOAD "name",ga
GSAVE "name",ga,sa
PRAT x,y,"Text"
KEY x,y,l,df$,z$(,fl:)

Nähere Hinweise zu den Grafikbefehlen in Ausgabe 4/85. PRAT und KEY ist in Ausgabe 6/85 beschrieben. Die meisten Befehle sprechen für sich, wir werden sie hier auch nicht näher erklären. Da es sich bei dieser Fassung von HIRESSPEED um eine Programmierer-Version handelt, sind nicht alle Fehlbedienungen abgefangen. Nach Drücken von RUN/STOP-RESTORE scheint der Rechner zu „hängen". Meist schafft ein POKE 648,4 Abhilfe. Dadurch werden die Tastatureingaben wieder in das Video-RAM ab $0400 (d 1024) geschrieben.

ter kein Problem, das Rahmenprogramm gibt die benötigte Zeit aus. Mit „Kurz" ist die Anzahl der benötigten Befehle gemeint, dabei gilt als ein Befehl alles, was nicht durch Doppelpunkte getrennt werden muß und in eine normale BASIC-Zeile paßt. Beispiel:

A=SIN(SQR(TAN(B*C*3/2)*100))
ist ein Befehl, die Zeile

A=1:B=2

enthält zwei Befehle, obwohl sie wesentlich kürzer ist. Überlange Zeilen sind verboten, als solche gelten Zeilen, die bei der Ausgabe durch den LIST-Befehl länger als zwei Bildschirmzeilen sind.

Entweder schnell . . .

Schicken Sie uns Ihre Lösung bitte nicht auf Datenträger (werden nicht zurückgeschickt!), sondern als Listing ein! Vermerken Sie auf diesem Listing

— ihren Namen mit **vollständiger Anschrift**, eventuell auch Telefonnummer,

— die Kategorie, unter der Sie sich beteiligen wollen, also „Schnell" oder „Kurz",

— und, unabhängig von der gewählten Kategorie, die Anzahl der benötigten Befehle oder die benötigte Zeit.

Die Lösung muß außerdem bestimmten Qualitätsanforderungen entsprechen, die das von uns mitgelieferte Programmgerüst testet.

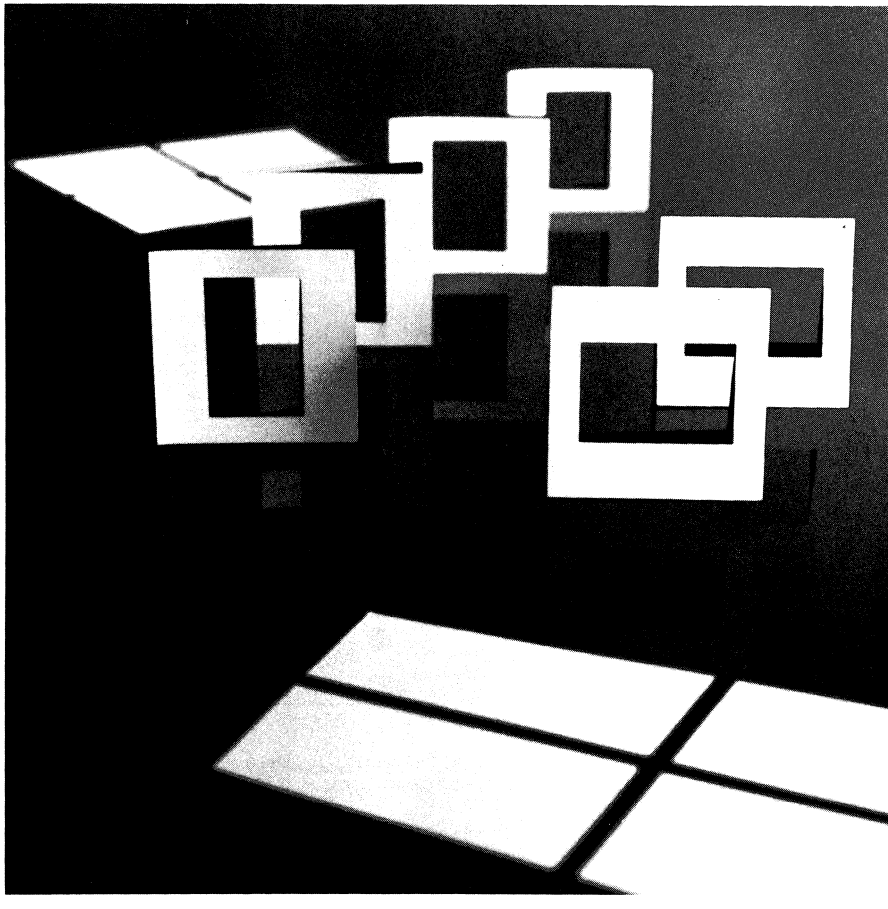
. . . oder kurz

Zum einen geht es dabei um die Genauigkeit des Kreis-Algorithmus, er soll nämlich Kreise erzeugen, keine Ovale. Ihr Kreis darf an keiner Stelle die beiden durch den im Grafikpaket eingebauten CIRCLE-Befehl gezogenen Kreise berühren. (Es versteht sich wohl von selbst, daß die Lösung CIRCLE X,Y,R,R nicht gilt.)

Zum anderen geht es um die Dichte der von Ihrem Programm gesetzten Punkte. Diese wird mittels des ebenfalls im Grafikpaket enthaltenen FILL-Befehls getestet, dieser darf aus Ihrem Kreis keinen Ausweg finden.

Aber genau!

Einsendeschluß ist der 31. Juli 1987, es gilt das Datum des Poststempels. Die Sieger in den beiden Kategorien erhalten je ein Jahresabo, für die zweiten bis sechsten Plätze gibt's ein Computer-Fachbuch. Bei mehreren gleichen Einsendungen entscheidet das Los, und wie immer ist der Rechtsweg ausgeschlossen. JS



Fenster auf und Fenster zu

INPUT-Windowing

Für manche Computer-Anwender ist der Begriff „Windows" nichts Neues. Andere dagegen lassen nur die Ohren hängen und wissen nicht, um was es dabei geht. Um die hängenden Ohren wieder aufzurichten, bieten wir Ihnen „INPUT-Windowing" an, eine Betriebssystem-Erweiterung, die es auch denjenigen erlaubt, über dieses Thema ein Wörtchen mitzureden, die bislang noch nichts von der Fenstertechnik gehört hatten. Die Wettbewerbssieger Oliver Mülhens und Thorsten Blaudeck haben sich mit diesem Programm die 3000 DM wirklich verdient.

In Insider-Kreisen wird von „Windowing" und von „Desktops" und solchem Zeug geredet. Was ist das überhaupt? Kommen wir zum ersten Begriff. Windows sind nichts anderes als Fenster. Ganz einfach, nicht wahr? Aber was bedeutet Desktop? Desk kommt wie Window aus dem Englischen und heißt Schreibtisch, und top heißt oben. Mit Desktop ist nichts anderes gemeint als Schreibtisch-Oberfläche.

Stellen Sie sich also vor, Sie sitzen vor solch einem Schreibtisch, auf dessen Oberfläche Sie in unserem Fall bis zu 24 beliebig große Arbeitsblätter liegen haben. Jetzt kommen wir zu den Windows. Jedes dieser Arbeitsblätter entspricht nämlich einem sogenannten Fenster.

Unbürokratisches Verwalten

Ordnungsliebende Menschen haben diese 24 Blätter meist zu einem Stapel zusammengefaßt. Von diesem Stapel sieht man nur das oberste Blatt, denn alle darunterliegenden Blätter werden ja von diesem verdeckt. Nimmt man das oberste herunter, kommt das nächste Blatt zum Vorschein und so weiter.

Bei nicht so ordnungsliebenden Menschen liegen die Arbeitsblätter mehr verstreut auf dem Schreibtisch herum. Dabei kann es vorkommen, daß man einige Blätter nur zum Teil, andere wiederum ganz betrachten kann. Wollen Sie den Inhalt eines dieser verdeckten Blätter lesen oder sogar auf ein anderes verdecktes Blatt etwas notieren, müssen Sie es unter den anderen Zetteln hervorholen und obendrauf legen. Jetzt können Sie auch sehen, daß die darauf enthaltene Information nicht einfach verschwunden, sondern noch vorhanden ist. Nach diesem Verfahren arbeitet auch INPUT-Windowing mit den „Arbeitsblättern", die wir in Zukunft Fenster oder Windows nennen wollen.

Damit wir uns nicht um die Verwaltung dieser Fenster kümmern müssen, überlassen wir sie dem Computer. Auch die verdeckten oder zum Teil verdeckten Inhalte der Fenster muß er kennen und legt sie zu diesem Zweck im Arbeitsspeicher ab. Dafür werden 12 KByte Platz reserviert. Dieser befindet sich unter dem Kernal- und dem Charakter-ROM. INPUT-Windowing selbst belegt die freien 4 KByte ab $C000. Wie Sie sehen, bleibt dadurch das komplette BASIC-RAM für Programme erhalten. Die reservierten 12 KByte werden in je 6 KByte für Farbe und 6 KByte für Zeichencode aufgeteilt. Sind alle Fenster geöffnet, darf deren Gesamtgröße 6144 - 1, also insgesamt 6143 Bytes nicht überschreiten. Das entspricht 23 Fenstern der Größe 16 * 16 und einem Fenster der Größe 15 * 17 (inklusive Rahmen). Für die meisten Anwendungen ist dies mehr als genug. Da wir schon einmal bei den Rahmen sind, noch ein Wörtchen dazu. Natürlich fehlen diese auch bei INPUT-Windowing nicht. Die Variationsmöglichkeiten sind nahezu unbegrenzt, denn Sie haben die Wahl zwischen 10 Haupttypen:

- 8 vorgefertigte Rahmen, per Nummer ansprechbar
- frei definierbare Rahmen
- kein Rahmen (dazu später mehr)

Einige wesentliche Merkmale einer Window-Verwaltung:

- schnell
- komfortabel
- von BASIC aus ansteuerbar
- leicht erlernbar
- speicherplatzsparend
- flexibel

Was Windows alles sein sollten

Um dies alles zu erreichen, muß das Programm natürlich in Maschinencode geschrieben sein und dem BASIC neue Befehle zur Verfügung stellen. Diese wiederum sollten aber leicht erlernbar sein und die Ablaufgeschwindigkeit der „alten" Befehle nicht nachteilig beeinflussen (was nicht selten der Fall ist). Die Lösung der Probleme ist genauso einfach wie effektiv: Um die neuen von den alten Befehlen unterscheiden zu können, erhalten sie eine Präfix, ein vorangestelltes Zeichen (bei INPUT-Windowing ein „W"), und enden größtenteils mit schon bekannten Befehlen, z.B. „OPEN".

INPUT-Windowing versteht Byte-Sparsamkeit im doppelten Sinne. Das heißt, daß auch die Verwaltung verdeckter Bildflächen, die bei Windows nun einmal dazugehören, nicht übermäßig viel Platz benötigt. Damit verbietet sich die Verwaltung hochauflösender Grafiken von selbst. Wer nun meint, daß die normale Zeichensatzgrafik mehr Nach- als Vorteile bietet, wird hier eines Besseren belehrt, denn die Tatsache, daß das Betriebssystem und auch das komplette BASIC auf Zeichensatzgrafik abgestimmt sind, vereinfacht die Handhabung solcher Fenster wesentlich. Das bekommt auch der Anwender zu spüren, denn stellen Sie sich 'PRINT', 'INPUT', 'LINE INPUT' und ähnliches im hochauflösenden Grafikmodus vor!

Nun aber zu den technischen Daten von INPUT-Windowing:

- 24 (!) voneinander unabhängige, einzeln verwaltbare Fenster
- die maximale Fenstergröße erstreckt sich über den gesamten Bildschirm
- sowohl Zeichen als auch Farben werden verarbeitet
- Fenster einzeln ein- und ausblendbar
- keine Geschwindigkeitsabnahme des BASIC durch Zusatzbefehle
- Datenablage unter den ROMs, das heißt: 38911 BASIC Bytes frei!

— Jokerzeichen beschleunigen die Abarbeitung und vereinfachen die Eingabe
— Funktionen zur Umwandlung von Bildschirmcode in CBM-ASCII sowie umgekehrt
— u.v.m.

Außerdem ist in INPUT-Windowing ein Maschinenprogramm zum Bewegen von Sprites enthalten, das von BASIC aus mit SYS928 aufgerufen wird und im Demonstrationsprogramm ab Zeile 5000 zu finden ist.

Demos mal anders

An dieser Stelle ist noch einiges zu der im INPUT 64-Betriebssystem laufenden Version von INPUT-Windowing zu sagen. Innerhalb von INPUT 64 läuft ein Demo-Programm, das in BASIC geschrieben ist. Dieses Demo-Programm bezieht sich auf die neuen Befehle von INPUT-Windowing und kann genauso wie die Erweiterung selbst mit CTRL-S auf Ihren eigenen Datenträger abgespeichert werden. Dazu können Sie mit der Cursor-Taste für 'rauf/runter' das Programm bestimmen, welches gesichert werden soll. Speichern Sie auf jeden Fall das Demo-Programm mit ab, denn anhand dieses Programmes ist sehr anschaulich zu sehen, wie Sie alle Möglichkeiten von INPUT-Windowing ausschöpfen können.

Bisher unerwähnt blieb die Parametertabelle, die INPUT-Windowing intern führt. Falls Sie dies nicht interessiert, überspringen Sie einfach den nun folgenden Abschnitt.

INPUT-Windowing holt sich alle wichtigen Informationen aus einer Parameterliste. Diese enthält die genaue Beschreibung und den augenblicklichen Zustand der Fenster. Auch wenn Sie sich nicht darum zu kümmern brauchen, sei dennoch verraten, daß hier die aktuelle Cursor-Position im Fenster (für den „WPRINT"-Befehl), die Startadresse des Fensters im RAM für die unsichtbaren Zeichen, die Startposition im Bildschirmspeicher, die Höhe, die Breite, das REVERS-Flag, das Hochkommamodus- und das Ein- Ausblend-Flag abgelegt sind. Weiterhin erfährt man dort auch, ob das Fenster überhaupt geöffnet ist und nicht zuletzt die aktuelle Cursor-Farbe (ebenfalls für den „WPRINT"-Befehl). Um das Verwirrspiel komplett zu machen: auch das Rahmen-Flag ist dort vertreten, welches besagt, ob ein Rahmen angewählt wurde oder nicht. Das ist wichtig zur richtigen Positionierung des Cursors und bei 'CLR/HOME', da sonst der Rahmen überschrieben würde.

Grundregeln bei der Bedienung

a) Es ist immer sinnvoll, sich mit 'WCLR' und 'PRINT‹CLR/HOME›' aller eventuell noch vorhandenen Fenster zu entledigen.

b) Gewünschtes Fenster mit 'WOPEN' definieren (noch unsichtbar).

c) Ab hier können (fast) alle neuen Befehle auf dieses Fenster angewendet werden. „Fast" bedeutet, daß ein nachträgliches „WOPEN" selbstverständlich nicht mehr möglich ist.

d) 'WCLOSE' schließt das Fenster wieder. Dazu müssen Sie nur die bei 'WOPEN' verwendete Nummer angeben. Das entspricht dem Verfahren, mit dem das C64-BASIC Files verarbeitet. Daher ist keine besondere Eingewöhnung notwendig.

Wie in a) beschrieben, können Sie natürlich auch alle Fenster auf einen Schlag entfernen. Der Unterschied zu 'WCLOSE' liegt darin, daß die Fenster nicht vom Bildschirm genommen werden, was aber ein anschließendes 'PRINT‹CLR/HOME›' sofort besorgt. Bis hierhin alles klar? Wenn ja, dann schreiten wir nun zur detaillierten Beschreibung der einzelnen Anweisungen, wobei vorher noch einige Abmachungen getroffen werden müssen.

Kürzel	Bezeichnung
PRT	Fensternummer
BRE	Fensterbreite
HOE	Fensterhöhe
RFA	Rahmenfarbe
RNR	Rahmennummer
RSG	Rahmen-String
*	Joker
ALE	Ausgabeliste
XKO	x-Koordinate
YKO	y-Koordinate
RUX	x-Koordinate rechts unten
RUY	y-Koordinate rechts unten
ZCE	Zeichencode
LOX	x-Koordinate links oben
LOY	y-Koordinate links oben
MOD	Modus
AZL	Anzahl
FAB	Farbe
ZNR	Zeilennummer
SGV	String-Variable
SPE	Spalte
LGE	Länge
DSG	Definitions-String
ZSG	Ziel-String
ZWV	Zeilenwechselvariable

Tabelle 1: Kürzel über Kürzel

Alle in der folgenden Befehlsbeschreibung verwendeten Kürzel beziehen sich auf Tabelle 1. **WOPEN** PRT,LOX,LOY,BRE,HOE,RFA (,RNR bzw. RSG)

Öffnet ein Fenster mit der Priorität PRT an den Koordinaten LOX und LOY mit der Breite BRE, der Höhe HOE und der Rahmenfarbe RFA,

stellt es aber nicht auf dem Bildschirm dar. Ein bereits geöffnetes Fenster darf kein zweites Mal geöffnet werden. Für PRT sind Werte von 1 bis 24 erlaubt. Ein Fenster mit der Priorität 1 wird von allen anderen überdeckt. 1 hat die kleinste, 24 die höchste Priorität. LOX und LOY beziehen sich auf die Koordinaten der linken oberen Ekke des Fensters. LOX akzeptiert Werte von 0 bis 39, LOY Werte von 0 bis 24. RFA kann Werte für die Rahmenfarbe von 0 bis 15 annehmen. Die Werte in der Klammer müssen nicht angegeben werden. RNR bezieht sich auf eine Nummer der acht vordefinierten Rahmen, RSG steht für Rahmen-String.

Der Rahmen-String muß mindestens aus acht Zeichen bestehen ("#-#!!#-#"). Diese Zeichen stellen den gesamten Rahmen dar (siehe Bild 1). Sie bedeuten von links nach rechts: linke obere Ecke, obere Kante, rechte obere Ecke, linke Seite, rechte Seite, linke untere Ecke, rechte untere Ecke.

```
#------#
!      !
!      !
!      !
!      !
!      !
!      !
!      !
!      !
#------#
```

Bild 1: So kann ein selbstgemachter Rahmen aussehen.

Zusätzlich zu den Rahmenzeichen kann der Rahmen-String auch Steuerzeichen enthalten, zum Beispiel 'RVSON' oder 'RVSOFF'.

WCLOSE PRT

Schließt das Fenster mit der Priorität PRT und entfernt es vom Bildschirm und aus dem Speicher. Statt PRT kann auch das Jokerzeichen (*) angegeben werden. Dann wird das zuletzt bediente Fenster geschlossen. Wird ein Fenster angesprochen, das vorher nicht geöffnet wurde, erscheint ein 'FILE NOT OPEN ERROR'.

WCLR

Diese Anweisung schließt alle geöffneten Fenster, entfernt sie aber nicht vom Bildschirm.

WPRINT PRT

Schreibt in das mit PRT bezeichnete Fenster. Weiteres siehe unter dem normalen BASIC-PRINT-Befehl. Auch hier ist der Joker zugelassen.

WAT PRT,LOX,LOY

Setzt den Cursor im Fenster mit der Priorität PRT an die durch LOX und LOY bezeichnete Koordinate. Der Joker ist erlaubt.

WONOFF PRTm

Macht ein mit PRT oder dem Joker bezeichnetes unsichtbares Fenster auf dem Bildschirm sichtbar und umgekehrt. Das bezeichnete Fenster muß vorher geöffnet sein.

WMOVE PRT,LOX,LOY

Dieser Befehl verschiebt das Fenster PRT zu den neuen, mit LOX und LOY bezeichneten Koordinaten. Der Joker ist erlaubt.

WDEF PRT,MODm

Ändert die Schreibsperre für den Fensterrahmen des Fensters PRT.
MOD=0: Schreibsperre aufgehoben. Es kann jetzt mit **WPRINT** auf den Rahmen geschrieben werden.

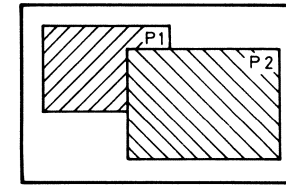
MOD=1: Schreibsperre aktiv. Es kann jetzt nicht auf den Rahmen geschrieben werden.

WFRAME PRT,RNR

Dieser Befehl erzeugt für das Fenster PRT einen vorgefertigten Rahmen mit der Rahmennummer RNR oder einen selbstdefinierten Rahmen RSG.

WSWAP PRT 1,PRT 2

Tauscht die Priorität des Fensters PRT 1 mit der Priorität des Fensters PRT 2. Außerdem werden auch die Inhalte der Fenster vertauscht. Dieser Befehl ist mit Vorsicht zu genießen. Bei einem erneuten Ansprechen eines der beiden vertauschten Fenster sollte man bedenken, daß jeweils das andere Fenster gemeint ist.



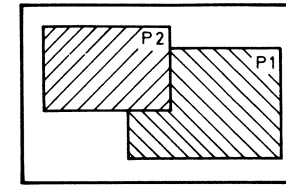


Bild 2: Es geht immer hin und her

WDROT PRT,AZL

Läßt die Prioritäten einer Anzahl AZL von Fenstern rotieren. Fenster PRT erhält dabei die höchste Priorität. Beispiel: **WDROT**1,3

Fenster eins erhält die Priorität drei, Fenster zwei erhält die Priorität eins und Fenster drei die Priorität zwei. Die drei Fenster werden auch ihrer Priorität entsprechend dargestellt, wobei sich die Koordinaten der Fenster nicht ändern (siehe Bild 2 und 3).

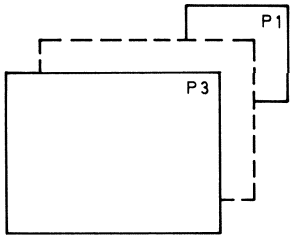


Bild 3: Vor dem . . .

WUROT PRT,AZL

Ähnlich dem Befehl **WDROT**, wobei aber das Fenster mit der höchsten Priorität die kleinste Priorität erhält.

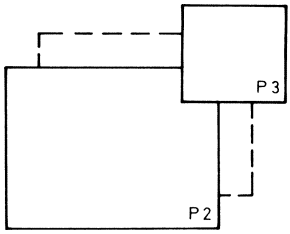


Bild 4: . . .und nach dem Rotieren

WREV PRT,LOX,LOY,RUX,RUY

Invertiert einen Ausschnitt des Fensters PRT. Der Auschnitt hat die Koordinaten, die durch LOX,LOY,RUX und RUY angegeben sind.

WPAINT PRT,ZCE,FAB

Füllt das Fenster PRT mit dem Zeichencode ZCE, der die Farbe FAB hat.

WPOKE PRT,XKO,YKO,ZCE (,FAB)

Dieser Befehl schreibt in das Fenster PRT an den Koordinaten XKO,YKO das Zeichen ZCE. Die in Klammern stehende Farbe FAB für das zu schreibende Zeichen muß nicht, kann aber mit angegeben werden.

WSTR$ PRT,ZNR,SGV

Aus dem Fenster PRT wird der Inhalt der Zeile mit der Nummer ZNR in die String-Variable SGV übertragen.

WINPUT PRT,SPE,ZNE,LGE,DSG, ZSG (,ZWV)

Ein INKEY-Befehl für das Fenster PRT. Die Eingabe beginnt in der Spalte SPE und der Zeile ZNR. Das Eingabefeld hat eine Länge von LGE Zeichen. Welche Zeichen erlaubt sind, kann im Definitions-String DSG angegeben werden. Die eingegebenen Zeichen werden in den Ziel-String ZSG übernommen.

Wird die Zeilenwechselvariable mit angegeben, können Unterschiede beim Beenden der Eingabe gemacht werden. ZVG enthält dabei die Werte

−1 bei Beenden mit 'Cursor rauf'
0 bei Beenden mit 'RETURN'
1 bei Beenden mit 'Cursor runter'

WSPEEK (PRT,XKO,YKO)

Liest aus dem Fenster PRT den Zeichencode, der an den Koordinaten XKO und YKO steht. Alle drei Werte müssen in der Klammer stehen.

WCPEEK (PRT,XKO,YKO)

Liest aus dem Fenster PRT den Farbwert, der an den Koordinaten XKO und YKO steht. Alle drei Werte müssen in der Klammer stehen.

WCHR$ (NAG)

Das in den Klammern stehende numerische Argument wird in C64-ASCII gewandelt.

WASC (String)

Wandelt das erste Zeichen des in Klammern stehenden Strings oder den Inhalt der String-Variablen in ein numerisches Argument.

WAT SPE,ZNR

Der Cursor wird an die Spalte SPE und die Zeile ZNR des zuletzt bedienten Fensters gesetzt. Dieser Befehl kann auch innerhalb des **WPRINT**-Befehls benutzt werden. Beispiel:

WPRINT 1,"DAS IST";**WAT** 1,1; "EIN";**WAT** 2,2;"TEST !"

Es bleibt nur noch nachzutragen, daß bei einer 'IF . . .THEN'-Konstruktion der für Befehlserweiterungen typische Doppelpunkt nach dem 'THEN' nicht fehlen darf. Also:

IF ‹Ausdruck› THEN: ‹INPUT-Windowing-Befehl›

Wer nach diesen Ausführungen noch Fragen zu den einzelnen Kommandos hat, der findet die Antworten darauf sicherlich beim Ausprobieren oder im Demonstrationsprogramm.

O. Mülhens/T. Blaudeck/kfp

ID – Werkstatt

Wandeln, Üben und Wundern

Vokabeln wandeln zum zweiten

Ein **VT-Konverter** befand sich schon einmal in dieser Rubrik (4/87). Versprochen hatten wir, daß dieses Programm Dateien des im Februar 85 veröffentlichten Programms DICTIONARY in solche des VOKABELTRAINERs (3/87) verwandelt. Alles Lüge!

Zwar konvertierte dieses Tool die Daten verschiedenster Vokabelprogramme, aber nicht die von DICTIONARY. Deswegen hier also der richtige Wandler. Hinweise zur Bedienung erübrigen sich wegen der ausführlichen Kommentierung in den REM-Zeilen.

Die letzten Ausgaben dieser Rubrik waren der Programmpflege gewidmet. Wir hatten Ihnen Hilfsprogramme angeboten, die bereits veröffentlichte Programme von erkannten Fehlern befreit haben. Diesmal stellen wir Ihnen unter anderem zwei kleine Programme vor, deren Grundideen durchaus ausbaufähig sind; also echte Werkstatt-Produkte.

Programmieren gut – Kopfrechnen schwach

Wenn ein Byte-Wert größer als 127 ist, weiß jeder Programmierer, daß das siebte Bit gesetzt ist. Sollte die Aufgabenstellung aber jenseits der Byte-Grenze liegen, ist oftmals der Griff zum Taschenrechner oder der Direktmodus des Rechners angesagt. Aber auch für Nichtprogrammierer (und nicht nur für die „Kleinen") kann ein gelegentliches — oder auch regelmäßiges — Trainieren der kleinen grauen Zellen nicht schädlich sein.

Sollten Ihnen die Aufgaben des **Kopfrechentrainers** zu einfach sein, können Sie an den geeigneten Stellen den Wertebereich der Zahlenräume einfach verändern. Die Grenze des C64 übersteigt sicherlich auch die Fähigkeiten des geübtesten Kopfrechners.

Wundersames Zahlenraten

Wir entnehmen dem Begleitschreiben des Autors, daß das kleine Spiel **Bitmaster** als Kartenspiel eine lange Tradition hat. „Vor 60 Jahren bekam ich als 10jähriger ein Zahlenratespiel mit 7 Spielkarten geschenkt . . ."

Auch mit der Umsetzung auf den Computer hat dieser alte „Kartentrick" nichts von seiner Faszination eingebüßt. Das Spiel erklärt sich selbst, und wenn Sie hinter die Programm-Logik kommen wollen, können Sie einfach LIST sagen und das kurze Programm analysieren. WM

Lernen im Dialog

Englische GRAMmatik

Der feine Unterschied, ob mit einer Wenn-Dann-Konstruktion eine allgemeingültige Beziehung oder eine logische Konsequenz beschrieben wird, liegt Ihnen in der deutschen Sprache wahrscheinlich „im Blut". Auf Englisch können Sie derlei in dieser Folge unserer Serie üben.

Thema des siebten Teils der Grammatik-Serie sind zum einen die bei allen Englischschülern beliebten „If-clauses", unter verschiedenen Zeiten natürlich, zum anderen die Form „Reported Speech".

Die Bedienung des Lernprogramms ist denkbar einfach: Die richtigen Eingaben müssen jeweils in die Textlücken der Beispielsätze eingegeben werden (mit RETURN abschließen); nach jeder Eingabe können Sie entweder eine der auf dem Bildschirm gezeigten Möglichkeiten wählen oder mit einer beliebigen anderen Taste mit der nächsten Frage fortfahren.

Die Bezeichnung „Lernprogramm" ist übrigens etwas irreführend. Es werden nämlich nicht die grammatischen Regeln oder gar deren Hintergrund vermittelt, sondern es geht darum, vorhandene oder im Lauf der Zeit verschüttete Kenntnisse einzuüben beziehungsweise aufzufrischen. Es handelt sich also eher um ein sogenanntes „Repetitorium". JS

Stapelweise Befehle

INPUT 64-Assembler-Schule, Teil 5

Wenn Sie sich die Opcode-Tabelle in der Mitte dieses Heftes ansehen, werden Sie feststellen, daß Sie die meisten Befehle schon kennen. Wir haben zwar nicht jeden Befehl mit jeder Adressierungsart verwendet. Die Arbeitsweise der Befehle sollte Ihnen aber trotzdem bekannt sein. Es ist übrigens ziemlich überflüssig, diese Tabelle auswendig zu lernen, Sie sollten aber wissen, welche Befehle mit welchen Adressierungsarten kombinierbar sind.

Wenn Sie die bisherigen Folgen dieses Kurses aufmerksam durchgearbeitet haben, sind Sie in Maschinensprache schon einigermaßen fit. Die wenigen Befehle, die Sie noch nicht kennengelernt haben, werden diesmal besprochen. Außerdem wollen wir einen Blick auf Eigenheiten des C64 werfen.

Faulenzer

Der einfachste unter den Befehlen, die wir noch nicht vorgestellt haben, heißt

NOP

Er benötigt keine Argumente und tut auch nichts (NOP heißt No OPeration — keine Operation). Auf den ersten Blick scheint es überflüssig, einen solchen Befehl zur Verfügung zu haben. Aber auch das Nichtstun kostet Zeit. Und manchmal kommt es eben darauf an, die CPU ein wenig warten zu lassen, ohne Registerinhalte zu verändern. Der Befehl findet häufig in zeitkritischen Routinen Anwendung, zum Beispiel bei Programmen zur Bedienung des Kassettenrecorders oder der seriellen Schnittstelle.

Bit-Tester

Ebenfalls bei Programmen, die für Peripherie zuständig sind, wird der Befehl

BIT

gern verwendet. Mit ihm kann abgefragt werden, ob bestimmte Bits in der angesprochenen Speicherstelle gesetzt sind. Er führt eine bitweise AND-Verknüpfung zwischen dem Akku und dem Inhalt der Adresse durch. Im Gegensatz zum AND-Befehl wird das Resultat jedoch nicht in den Akku übernommen. Sein Inhalt bleibt unverändert — etwas Ähnliches kennen Sie ja schon von dem Befehl CMP.

Je nach dem Ergebnis der AND-Verknüpfung wird die Z-Flagge im Statusregister gesetzt. Die Flags N und V enthalten nach der Abarbeitung des BIT-Befehls den Inhalt der Bits 7 beziehungsweise 6 der getesteten Adresse.

Mit diesem Befehl kann man also unabhängig vom Akku-Inhalt die beiden obersten Bits eines Speicher-Bytes testen. Will man andere Bits überprüfen, so muß der Akku vorher mit einer entsprechenden Maske geladen werden.

Speicher als Zähler

Auch die beiden Befehle, die jetzt zur Sprache kommen sollen, sind nicht schwer zu verstehen; sie arbeiten ähnlich wie schon bekannte Kommandos. Gemeint sind

INC (INCrement memory) und
DEC (DECrement memory).

Der Befehl INC tut dasselbe wie INX oder INY, nur wird nicht ein Indexregister, sondern eine Speicherstelle um eins erhöht. Die Flaggen des Statusregisters verhalten sich dabei wie gehabt.

Der DEC-Befehl vermindert den Inhalt der angesprochenen Adresse um eins, auch er beeinflußt die N- und die Z-Flagge wie die Befehle DEX und DEY.

Diese beiden Befehle werden benutzt, wenn ein Schleifenzähler benötigt wird, aber die Register X und Y als Indexregister verwendet werden müssen. Eine andere Anwendung dieser Befehle zeigt Bild 1. Hier wird mit dem INC-Befehl Zeit und Speicherplatz gespart. Bei der Subtraktion einer Ein-Byte-Zahl von einem Zwei-Byte-Wert können Sie mit dem DEC-Befehl den gleichen Trick anwenden.

Datenkeller

In der vorletzten Folge haben Sie den Prozessor-Stack, auch Kellerspeicher genannt, kennengelernt. Sie kennen seine Bedeutung für die Aufbewahrung von Rückkehradressen bei Un-

```
C018:  ;PNTR=PNTR+7
C018:        CLC
C019:        LDA PNTR
C01C:        ADC #7
C01E:        STA PNTR
C021:        LDA PNTR+1
C024:        ADC #0
C026:        STA PNTR+1
C029:        ...
```

```
C018:  ;PNTR=PNTR+7
C018:        CLC
C019:        LDA PNTR
C01C:        ADC #7
C01E:        STA PNTR
C021:        BCC OKAY
C023:        INC PNTR+1
C026:  OKAY  ...
```

Bild 1: Mit dem INC-Befehl wird die Adition eines Bytes zu einem Zwei-Byte-Wert schneller und kürzer.

terprogrammaufrufen. Das ist aber nur die halbe Wahrheit. Man kann den Stack nämlich auch selbst für die Zwischenspeicherung beliebiger Daten nutzen. Dazu gibt es das Befehlspaar

PHA (PusH Accu) und
PLA (PulL Accu).

Der erste dieser beiden Befehle „schiebt" den Inhalt des Akkus auf den Stapel. Dabei wird der Stackpointer um eins vermindert, diese Operation belegt also nur ein Byte im Stack. (Beim JSR-Befehl sind es jedesmal zwei Bytes.)

Durch den PLA-Befehl wird der so zwischengespeicherte Wert wieder in den Akku zurückgeholt und der belegte Speicherplatz im Stapel freigegeben.

Diese beiden Befehle sind mit äußerster Vorsicht zu genießen. Vergißt man nämlich, einen gePUSHten Wert zurückzuholen, oder läßt den Prozessor einen überflüssigen PLA-Befehl ausführen, so befindet sich der Stack in einem nicht ordnungsgemäßen Zustand — und das nächste RTS kommt bestimmt. Der Prozessor kann ja nicht wissen, ob die beiden oben auf dem Stack liegenden Bytes durch einen PHA- oder durch einen JSR-Befehl dort hingekommen sind. Er interpretiert sie bei einem RTS immer als Return-Adresse. Und daß dieser Sprung „in die Wüste" geht, wenn ein Byte zuviel oder zuwenig auf dem Stack liegt, dürfte wohl klar sein. Diesem Programmierfehler ist besonders schwer beizukommen, da der Prozessor dabei jedesmal woanders landen kann und sich unter Umständen völlig chaotisch verhält.

Leider gibt es die Möglichkeit der Ablage auf dem Stack nicht für die Indexregister. Braucht man sie doch einmal, muß man sich mit einer TXA-PHA- beziehungsweise PLA-TAX-Folge behelfen. (Achtung, dabei geht der Akku-Inhalt verloren!)

Zum Trost haben die Entwickler des 6502 die Möglichkeit vorgesehen, den Inhalt des Prozessor-Status-Registers auf dem Stack abzulegen und wiederherzustellen. Dazu gibt es die Befehle

PHP (PusH Processor status) und
PLP (PulL Processor status).

Auch diese beiden Befehle sind nicht ganz ungefährlich. Zusätzlich zu den bei PHA und PLA besprochenen Problemen kann ein unüberlegtes PLP-Kommando beispielsweise dazu führen, daß plötzlich die D-Flagge gesetzt ist und keiner weiß, warum das Programm nur noch Müll berechnet.

Andererseits sind diese beiden Befehle die einzigen, mit denen man direkt an das Statusregister herankommt. Um zu testen, ob die D-Flagge gesetzt ist, muß man zum Beispiel die Befehlsfolge

```
PHP
PLA
AND #%00001000
BEQ HEXA
BNE DEZI
```

verwenden. Es gibt nämlich keinen BDS- („Branch on Decimal Set"-) Befehl.

Das Unterste zuoberst kehren

Wo wir gerade beim Stack sind, sollen hier noch zwei Befehle vorgestellt werden, die auch mit dem Keller zu tun haben. Sie heißen

TSX (Transfer Stack pointer to X) und
TXS (Transfer X to Stack pointer).

Letzterer dient dazu, den Stackpointer zu initialisieren. Das muß zum Beispiel beim Einschalten des Rechners oder nach einem Reset geschehen. Dazu wird das X-Register normalerweise mit $FF geladen und dann der TXS-Befehl ausgeführt. Dadurch sind aus der Sicht des Prozessors alle Eintragungen aus dem Stack entfernt, und es stehen wieder 256 freie Plätze zur Verfügung.

Der TSX-Befehl ist die einzige Möglichkeit, den Inhalt des Stackpointers zugänglich zu machen. Der BASIC-Interpreter prüft zum Beispiel vor jedem GOSUB-Befehl mit einer TSX-CPX-Folge, ob noch genug Platz auf dem Stapel frei ist.

Mit diesem Befehl kann man auch die Kellerstruktur überlisten und zum Beispiel die drittletzte Eintragung aus dem Stack lesen:

```
TSX
LDA $103,X
```

Dabei wird weder der Stackpointer noch der Inhalt des Stapels selbst beeinflußt.

Anlasser

Alle Befehle und Programme, die wir bislang besprochen haben, arbeiten genau voraussehbar. Das heißt, wenn man die Registerinhalte des Prozessors und das Programm kennt, kann man genau voraussagen, welche Aktionen in welcher Reihenfolge durchgeführt werden. Es gibt aber auch Möglichkeiten, diese Abarbeitung eines Programmes von außen zu beeinflussen. Dazu gibt es die sogenannten Unterbrechungen oder Interrupts.

Beim 6502 gibt es drei Arten, von außen eine Programmunterbrechung herbeizuführen. Eine davon ist der Reset. Der Prozessor hat dafür eine spezielle Leitung. Wird an dieser ein Impuls angelegt, so bricht der Prozessor sofort seine Arbeit ab. Er lädt dann den Programmzähler mit dem Inhalt der Adressen $FFFC und $FFFD. Diese Adressen liegen im ROM. Das ist bei jedem 6502-Rechner so, denn auch beim Einschalten wird der Programmzähler mit dieser Adresse geladen, und die soll ja jedesmal denselben Wert enthalten, damit der Computer in einen definierten Ausgangszustand kommt.

Das Laden des Programmzählers mit der Reset-Adresse bewirkt nichts anderes als einen Sprung zu dieser Adresse. Beim C64 ist es die Adresse $FCE2 (64738). An dieser Stelle (auch noch im ROM) steht eine Routine, die zunächst den Stackpointer initialisiert, die D-Flagge löscht und überprüft, ob ein Autostart-Modul im Extension-Port steckt. Falls nicht, werden als nächstes die Peripherie-Chips initialisiert. Dazu gehört zum Beispiel, daß Hintergrund- und Rahmenfarbe gesetzt werden, die Lautstärke des SID auf null geht und die Schnittstellen (Userport, serieller Bus, Kassette etc.) in einen definierten Zustand versetzt werden.

Dann werden diverse Pointer und Vektoren gesetzt. (Zu dem Begriff Vektoren kommen wir in wenigen Augenblicken.) Zum Schluß wird die NEW-Routine angesprungen, und von da aus geht es in den BASIC-Interpreter, der nun auf Ihre Eingaben wartet.

Unterbrechungen . . .

Die beiden anderen erwähnten Hardware-Unterbrechungen heißen „nicht maskierbarer Interrupt" (NMI) und Interrupt-Anfrage („Interrupt-Request", IRQ). Sie laufen ganz ähnlich ab wie ein Reset, dienen aber anderen Zwecken. Mit ihnen kann ein laufendes Programm unterbrochen werden, damit der Prozessor zwischendurch etwas anderes tut und dann zum unterbrochenen Programm zurückkehrt.

Beim C64 wird zum Beispiel regelmäßig ein IRQ ausgelöst, und zwar sechzigmal pro Sekunde. Das heißt, jedes Programm wird alle sechzigstel Sekunde unterbrochen. Der Prozessor führt dadurch regelmäßig ein Programm aus, das überprüft, ob eine Taste gedrückt ist, den Cursor blinken läßt, die Zeit (TI$) hochzählt und beim Drücken der STOP-Taste ein BASIC-Programm abbricht. Diese Routine läuft quasi „im Hintergrund", der Anwender merkt überhaupt nicht, daß sein Programm laufend unterbrochen wird.

Die Rückkehr zum unterbrochenen Programm wird dadurch bewerkstelligt, daß die CPU bei einem Interrupt ähnlich reagiert wie bei einem Unterprogrammaufruf. Die Adresse des nächsten Befehles wird nämlich auf dem Stack abgelegt.

Zusätzlich speichert der Prozessor auch den Inhalt des Statusregisters auf dem Stack ab. Denn im Gegensatz zu einem JSR-Befehl weiß man ja als Programmierer nicht, wann während des Programmlaufes ein Interrupt ausgelöst wird. Kommt er zufällig zwischen einem CMP- und einem BEQ-Befehl und verändert die Interrupt-Routine die Zero-Flagge, könnte das fatale Folgen haben.

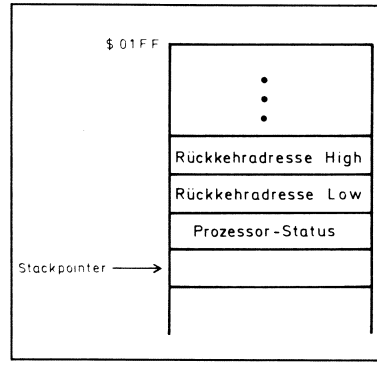
Zur Rückkehr in das unterbrochene Programm dient der Befehl

RTI (ReTurn from Interrupt),

der die Rückkehradresse und den Inhalt des Statusregisters wieder vom Stack holt.

. . .und wie man sich dagegen wehrt

Der Unterschied zwischen einem NMI und einem IRQ besteht darin, daß ein NMI in jedem Falle zu einer Programmunterbrechung führt, während man den IRQ unterbinden kann. Dazu dient die I-Flagge des Statusregisters. Sie kann mit dem Befehl



Bei einer Unterbrechung wird zuerst die Rückkehradresse, dann das Statusregister auf den Stack gerettet.

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 ÇText 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-Assembler-Schule ist eine Serie, die in 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.

SEI (SEt Interrupt disable)

gesetzt werden. Dadurch wird die Annahme des Interrupt-Requests verhindert. Um den Interrupt wieder zuzulassen, muß das I-Flag gelöscht werden. Dazu dient der Befehl

CLI (CLear Interrupt disable).

Ist die I-Flagge gelöscht, so ist die Reaktion auf einen IRQ die gleiche wie auf einen NMI. Im einzelnen passiert dabei folgendes: Zuerst wird der gerade in der Abarbeitung befindliche Befehl vollständig ausgeführt. Dann rettet der Prozessor die Rückkehradresse und das Statusregister auf den Stack und setzt die I-Flagge. Damit ist zunächst die Annahme eines weiteren IRQ gesperrt. Abschließend wird die Einsprungadresse der Interrupt-Routine aus dem ROM in den Programmzähler geladen und damit — genau wie bei einem Reset — die Interrupt-Behandlungs-Routine angesprungen.

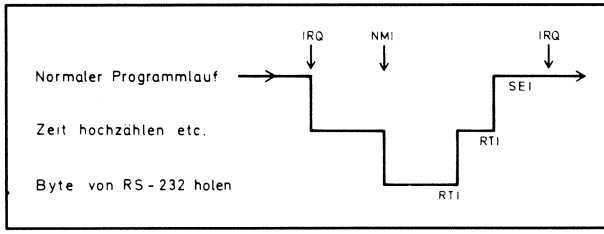
Die Adresse, an der die IRQ-Routine beginnt, steht in den Speicherzellen $FFFE und $FFFF, die für den NMI bei $FFFA und $FFFB.

Beim C64 wird ein NMI durch den gleichzeitigen Druck auf die Tasten RUN/STOP und RESTORE ausgelöst. Er dient als „weicher Reset". Außerdem kann über eine am User-Port angeschlossene RS-232-Schnittstelle ein NMI ausgelöst werden.

Wie schon erwähnt, wird ein IRQ regelmäßig durch einen Timer in einem der beiden Schnittstellenbausteine (CIAs) ausgelöst. Aber es gibt im 64er noch andere Interrupt-Quellen. So können die CIAs so programmiert werden, daß sie beim Empfangen eines Bytes am seriellen Port oder beim Anlegen eines bestimmten Spannungspegels an einer speziellen Leitung einen IRQ auslösen. Das wird zum Beispiel beim Laden von der Datassette ausgenutzt. Auch der VIC (der für die Bilderzeugung zuständige Chip) kann IRQs auslösen, beispielsweise, wenn zwei Sprites sich berühren oder wenn der Elektronenstrahl auf der Bildröhre eine bestimmte Position erreicht.

Umleitung

Nun wäre es unsinnig, auf all diese Ereignisse in derselben Art und Weise zu reagieren. Darum enthalten die NMI- und die IRQ-Routine ziemlich am Anfang einen indirekten Sprungbefehl. Das ist ein JMP-Befehl, dessen Argument nicht direkt die Adresse ist, zu der gesprungen wird, sondern eine Adresse, an der diese Adresse steht.



Ein NMI kann die IRQ-Routine unterbrechen, aber nicht umgekehrt. Er hat die höhere Priorität.

Die IRQ-Routine enthält zum Beispiel den Befehl

JMP ($0314).

Wenn die CPU auf diesen Befehl trifft, springt sie nicht zu der Adresse $314, sondern holt sich die Sprungadresse aus den Speicherzellen $314 und $315 (788/789). Man sagt, diese Adressen enthalten den IRQ-Vektor. Normalerweise steht dort die Adresse $EA31, der Beginn der IRQ-Routine im ROM.

Will man nun Interrupts für eigene Anwendungen einsetzen, so muß man diesen Vektor auf seine eigene Interrupt-Behandlungsroutine „verbiegen". Wie das geht und was dabei zu beachten ist, können Sie einem ausführlich kommentierten Assembler-Quelltext entnehmen, den Sie aus dem Programm heraus abspeichern und mit dem INPUT-ASS wieder laden können. Es handelt sich dabei um ein Programm, das den VIC so programmiert, daß er beim Erreichen einer bestimmten Rasterzeile einen Interrupt auslöst. Die Interrupt-Routine schaltet dann die Hintergrund-Farbe um.

Auf der Speicherseite drei, genauer: auf den Adressen von $300 bis $33E, stehen übrigens noch andere Sprungvektoren. Mit Ihnen ist es möglich, alle wichtigen Routinen des C64-Betriebssystems auf eigene Programme umzuleiten. Damit kann man BASIC-Erweiterungen einbinden oder die Drucker-Ausgabe auf eine Centronics-Schnittstelle am Userport umleiten. Einige dieser Vektoren sollen in der nächsten Folge der Assembler-Schule vorgestellt werden.

Dann werden wir uns auch mit der Einbindung von Assembler-Routinen in BASIC-Programme beschäftigen und einen Blick auf die Programmierung von C64-Spezialitäten wie Joystick-Abfrage und Grafik werfen. Hajo Schulz

Literatur

Christian Persson: 6502/65C02-Maschinensprache, Verlag Heinz Heise GmbH, Hannover 1983

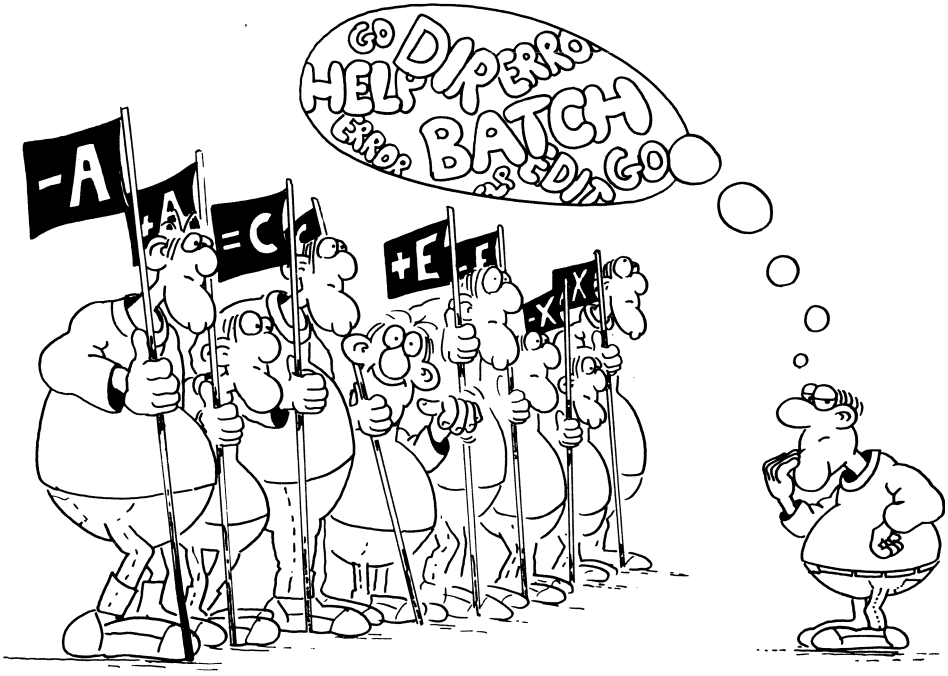
Rodnay Zaks: Programmierung des 6502, Sybex-Verlag GmbH, Düsseldorf 1981

Alle 6502-Befehle und Adressierungsarten

Adressierungsart	Implied	Immed.	Absolut	Zero-P.	ZP,X	Abs,X	Abs,Y	(ZP,X)	(ZP),Y	Relativ	Beeinflußte Flags
Befehlslänge	1	2	3	2	2	3	3	2	2	2	NV-BDIZC
ADC		69	6D	65	75	7D	79	61	71		NV....ZC
AND		29	2D	25	35	3D	39	21	31		N.....Z.
ASL	0A		0E	06	16	1E					N.....ZC
BCC										90	....
BCS										B0	....
BEQ										F0	....
BIT			2C	24							NV....Z.
BMI										30	....
BNE										D0	....
BPL										10	....
BRK	00										...1.1..
BVC										50	....
BVS										70	....
CLC	18										0
CLD	D8										....0...
CLI	58										.....0..
CLV	B8										.0......
CMP		C9	CD	C5	D5	DD	D9	C1	D1		N.....ZC
CPX		E0	EC	E4							N.....ZC
CPY		C0	CC	C4							N.....ZC
DEC			CE	C6	D6	DE					N.....Z.
DEX	CA										N.....Z.
DEY	88										N.....Z.
EOR		49	4D	45	55	5D	59	41	51		N.....Z.
INC			EE	E6	F6	FE					N.....Z.
INX	E8										N.....Z.
INY	C8										N.....Z.
JMP			4C						6C[2]		....
JSR			20								....
LDA		A9	AD	A5	B5	BD	B9	A1	B1		N.....Z.
LDX		A2	AE	A6	B6[1]		BE				N.....Z.
LDY		A0	AC	A4	B4	BC					N.....Z.
LSR	4A		4E	46	56	5E					0.....ZC
NOP	EA										....
ORA		09	0D	05	15	1D	19	01	11		N.....Z.
PHA	48										....
PHP	08										....
PLA	68										N.....Z.
PLP	28										NV-BDIZC
ROL	2A		2E	26	36	3E					N.....ZC
ROR	6A		6E	66	76	7E					N.....ZC
RTI	40										NV-BDIZC
RTS	60										....
SBC		E9	ED	E5	F5	FD	F9	E1	F1		NV....ZC
SEC	38										1
SED	F8										....1...
SEI	78										.....1..
STA			8D	85	95	9D	99	81	91		....
STX			8E	86	96[1]						....
STY			8C	84	94						....
TAX	AA										N.....Z.
TAY	A8										N.....Z.
TSX	BA										N.....Z.
TXA	8A										N.....Z.
TXS	9A										....
TYA	98										N.....Z.

[1] Zero-Page,Y

[2] Indirekt, Drei-Byte-Befehl



C'est ICI

Der INPUT Command Interpreter

Der 64er hört aufs Kommando - für geübte Freaks kein Problem. Doch manch einer tut sich schwer damit, dem Computer sein Anliegen bezüglich des Disketten-beziehungsweise File-Handlings verständlich zu machen. Hier kann ICI Abhilfe schaffen, und auch dem Profi wird eine Menge Interessantes geboten.

In der Januar-Ausgabe dieses Jahres wurde JAM unter IOS veröffentlicht. JAM bietet eine grafisch orientierte Benutzeroberfläche, die Ihnen auf komfortable Weise den Umgang mit dem Computer erleichtert, weil es hiermit nicht mehr nötig ist, sich komplexe Befehlssequenzen zu merken. Was man machen möchte, wird durch anschauliche Symbole und einfache Auswahlmöglichkeit deutlich vereinfacht. Mit dieser Methode wird eine Dialogform mit dem Computer möglich, wie sie auf neueren Rechnern bereits zum Alltag gehört. Aber auch dort findet man häufig zusätzlich zu einer grafischen Benutzeroberfläche einen sogenannten Kommandomodus, der es ermöglicht, die Funktionen auch auf dem herkömmlichen Weg über die Tastatur mit handgetippten Befehlen aufzurufen.

Anachronismus? — Wir werden sehen. Schon ein Klassiker ist das Betriebssystem CP/M, für das bereits unzählige Programme auf dem Markt vorhanden sind. Auch CP/M arbeitet wie ICI kommandoorientiert. Dasselbe gilt für das in PC-Kreisen verbreitete Betriebssystem MS-/PCDOS.

JAM oder ICI

Ein nicht zu unterschätzender Vorteil eines Kommando Interpreters gegenüber einer grafischen Benutzeroberfläche ist der wesentlich geringere Speicherplatzbedarf, da auf aufwendige Grafiken verzichtet werden kann. Deshalb lassen sich hier sehr viel

mehr Funktionen auf gleichem Raum einbauen. Da sich dieses auch bei der Entwicklung von ICI zeigte, lag es nahe, noch zusätzliche Routinen zu entwerfen. So wurde aus dem kleinen ICI ein größeres Projekt, als zunächst geplant war, doch Sie werden sehen: Die Wartezeit hat sich gelohnt.

Trotz der Unterschiede zwischen JAM und ICI wurde natürlich die schon von JAM her bekannte Kompatibilität gewahrt. Auch ICI liegt ab der Adresse $C000 (hex) im Speicher und läßt den BASIC-Speicher damit unberührt. Bei einigen Routinen werden zwar Puffer benötigt, doch deren Adreßlage können Sie mit entsprechenden Kommandos selbst bestimmen. Ansonsten werden lediglich einige Zeropage-Adressen und der Sprite-Block Nummer 11 zur Zwischenspeicherung von Daten benutzt. Und: Selbstverständlich ist auch ICI kompatibel zu Super-Disk (unser schneller Diskettenlader aus Ausgabe 1/87).

Kommandos und Flags

Die Funktionsweise eines Kommando-Interpreters ist schnell erklärt. Man nehme eine Reihe von Befehlen (die sogenannten Kommandos), die alle irgend etwas bewirken. Diese werden über die Tastatur eingegeben, und die Eingabe wird dann mit der RETURN-Taste beendet. Das Kommando wird sofort(!) ausgeführt, woraufhin ICI auf Ihr nächstes Kommando wartet. (Eine interessante Ausnahme — die Batch-Dateien — werden wir weiter unten beschreiben.) Ein einfaches Kommando wäre zum Beispiel „dir". Wie der Name vermuten läßt, wird unmittelbar das Directory angezeigt. Außer den Kommandos gibt es noch sogenannte „Flags", womit bestimmte Grundeinstellungen gesetzt werden können, zum Beispiel „+e", um ein Druckerecho einzuschalten.

Sie finden in diesem Artikel drei Tabellen, in denen in Kurzform die Befehle von ICI aufgeführt sind. (Die eigentlichen Befehle sind hier nur aus Gründen der besseren Lesbarkeit in Großbuchstaben gesetzt worden.) Dem informierten Leser wird sicher auffallen, daß sich einige am Wortlaut der CP/M-Kommandos orientieren.

Simulation

Innerhalb von INPUT 64 können Sie sich eine Simulation(!) ansehen. Einige der ICI-Kommandos werden hier vorgestellt. Innerhalb der Demonstration können Sie ICI jederzeit mit der Tastenkombination CTRL-S auf Ihren Datenträger überspielen. Außerhalb von INPUT 64 stehen Ihnen dann alle Kommandos zur Verfügung. Daß ICI sich allerdings nur sinnvoll mit einem (oder zwei) Diskettenlaufwerk(en) einsetzbar läßt, ist naheliegend.

Sie laden das aus INPUT 64 abgespeicherte Programm (mit LOAD"name",8) und starten es einfach mit RUN. ICI meldet sich mit einer Erkennungszeile und darunter mit '8>' und einem blinkenden Cursor. ICI ist bereit, Ihre Kommandos entgegenzunehmen. Bevor wir nun die Befehle im einzelnen beschreiben, eine Anmerkung zu der bereits verzierten Eingabezeile.

Die '8' bedeutet, daß das Laufwerk mit der Geräteadresse 8 aktiviert ist und daß sich alle folgenden Kommandos auf dieses Laufwerk beziehen. Das folgende Zeichen '>' (auch „Prompt" genannt) fordert Sie freundlich auf, eine Eingabe zu tätigen. Ihre Eingaben beziehen sich immer auf eine Zeile. (Über die Editiermöglichkeiten des Zeileneditors klärt Sie die Tabelle 3 auf.)

Die Kommandos . . .

Wir werden im folgenden die eigentlichen Befehle in alphabetischer Reihenfolge beschreiben. Die „eigentlichen" deshalb, weil jede Eingabe als Befehl interpretiert wird. Stellt die Zeichenfolge keinen gültigen Befehl dar, unterstellt ICI, daß Sie ein Programm laden wollen, das unter der eingegebenen Zeichenfolge auf der Diskette steht. Damit haben Sie schon den ersten Befehl, der nicht einmal eingegeben werden muß. Wollen Sie also ein Programm laden, geben Sie einfach den Programmnamen ein und bestätigen diese Eingabe (wie alle anderen auch) mit RETURN.

Da Sie sich nur entweder innerhalb von ICI oder im BASIC-Interpreter befinden können, gibt es das Kommando **basic** , das von ICI aus zum BASIC-Interperter verzweigt. Sie können ICI jederzeit durch einfaches Drücken der **RESTORE-Taste** erneut aufrufen. Der Rücksprung zu ICI kann natürlich nur funktionieren, wenn dessen Speicherbereich zwischenzeitlich nicht überschrieben und der 'NMI-Vektor' nicht zurückgesetzt wurde. Letzteres ist in BASIC-Programmen nicht zu erwarten.

Mit dem Kommando **bsave testprogramm,p** speichern Sie unter dem Namen „testprogramm" den Speicherbereich, der mit SETICI (siehe dort) festgelegt wurde, als Programm-File auf Diskette. Zwei Anmerkungen, die auch bei den folgenden Beschreibungen wichtig sind:

1. Befehle werden von den nachfolgenden Parametern immer durch (mindestens) ein Leerzeichen getrennt.

2. ICI unterstützt folgende File-Typen:
p = Programm-File
s = sequentielles File
u = User-File
Die Kennung sollte dem File-Namen durch ein Komma getrennt angehängt werden.

Der Befehl **dir** wurde schon genannt. Dieses Kommando benötigt keine Parameter. Wenn das Disketten-Inhaltsverzeichnis auf dem Bildschirm er-

scheint, können Sie mit **SHIFT** die Bildschirmausgabe kurzfristig anhalten, mit **SHIFT/LOCK** quasi einfrieren oder mit **RUN/STOP** abbrechen.

Auch die Commodore-DOS-Befehle sind von ICI aus erreichbar. Mit der Eingabe **dos n:tools,ab** können Sie beispielsweise eine neue Diskette formatieren, die dann den Diskettennamen „tools" und die ID-Kennung „ab" erhält. Diese Schreibweise der DOS-Befehle ist im Handbuch der Diskettenstation beschrieben (dort natürlich ohne die ICI-Kennung „dos"). Beachten Sie bitte, daß die Anführungszeichen um den Befehls-String bei ICI nicht mit eingegeben werden.

Wollten Sie bisher den Inhalt einer Datei anschauen, mußten Sie ein kleines Programm schreiben. ICI bietet mehrere Befehle hierfür an. Der erste listet einen sogenannten Hex-Dump der Datei auf dem Bildschirm. Mit **dump testdatei,s** wird die Datei „testdatei" eröffnet und auf den Bildschirm ausgegeben. Ihre Einflußmöglichkeiten sind in diesem Fall die gleichen wie bei der Direktory-Anzeige.

Auch für das Auslesen des Fehlerkanals brauchen Sie keine Programmzeile mehr zu schreiben. Sie geben einfach **error** ein, und ICI gibt postwendend den Fehlerstatus der aktiven Diskettenstation aus.

Wenn Sie ein Maschinenprogramm starten wollen, brauchen Sie ICI nicht zu verlassen, um innerhalb des BASIC-Interpreters „sys" einzugeben, sondern können dieses mit **go $$$$** aus ICI direkt vornehmen, wobei hier (und im folgenden) alle Eingaben von Adressen als vierstellige hexadezimale Zahl einzutippen sind (also gegebenenfalls mit führenden Nullen). Geben Sie nur **go** ein, verzweigt ICI zu der mit SETADR (siehe dort) festgelegten Adresse.

In der Einarbeitungsphase werden Sie nicht alle Kommandos von ICI direkt parat haben. Mit **help** können Sie sich deshalb alle Befehle angezeigen lassen.

ICI ist von der Konzeption her auf Erweiterungen ausgelegt. In einer der nächsten Ausgaben werden wir die Interna von ICI veröffentlichen. Hier nur soviel: ICI verfügt über eine eigene Befehls-Sprungleiste und die Möglichkeit, weitere Befehle zu verarbeiten. Um diese Erweiterungen einzubinden, ist das Kommando **ici name,typ** vorgesehen. Dieser Befehl wird die Erweiterung von der Diskettenstation einlesen und starten. Also noch Zukunftsmusik!

Alle Kommandos von ICI auf einen Blick

BASIC	= verzweigt zum BASIC-Interpreter
BATCH filename(,typ)	= ruft Batch-File auf und startet es an [1]
BSAVE filename(,typ)	= speichert den ICI-Pufferbereich
DIR	= zeigt Directory an
DOS befehl:(extension)	= ruft DOS-Befehle auf, als Beispiele: DOS R:neuname=altname DOS N:discname,id DOS S:filename
DUMP filename(,typ)	= listet File als Hex-Dump
EDIT filename(,typ)	= eröffnet Batch-File [1]
ERROR	= liest den Fehlerkanal aus
GO ($$$$)	= startet ADR (bzw. $$$$) [2]
HELP	= zeigt Kommandos an
ICI filename(,typ)	= liest File in ICI-Pufferbereich und startet es
PIP filename(,typ)(=filename(,typ))	= kopiert File
POKE $$$$ $$	= entspricht dem BASIC-POKE [2]
PRINT text	= gibt Text aus [1]
RUN	= startet ein BASIC-Programm
SETADR $$$$	= setzt Adresse für ↑ A und GO [2]
SETBAT $$$$ $$$$	= setzt Pufferbereich für Batch-Dateien [2]
SETICI $$$$ $$$$	= setzt ICI-Pufferbereich [2]
SAVE filename(,typ)	= speichert ein File
STATUS	= zeigt Flags und Adressen an
TYPE filename(,typ)	= zeigt File als ASCII-Code
VTYPE filename(,typ)	= zeigt File als Bildschirm-Code
WAIT	= wartet auf Tastendruck [1]
; text	= erlaubt Kommentare [1]
%	= Platzhalter, erwartet Eingabe [1]

1) Diese Befehle sind nur sinnvoll im Rahmen von Batch-Dateien.
2) Alle Ziffern-Eingaben müssen in hexadezimaler Schreibweise erfolgen.

Tabelle 1: Die Kommandos von ICI

Alle Flags und Adressen von ICI

+A	= Files werden absolut geladen
-A	= Files werden an den BASIC-Anfang geladen
↑ A	= Files werden nach ADR geladen
=C	= ein Laufwerk bei PIP
/C	= zwei Laufwerke bei PIP
+E	= Drucker-Echo an
-E	= Drucker-Echo aus
+X	= Direktmodus ausschalten
-X	= Direktmodus zulassen
8:	= Floppy 8 wird aktiviert
9:	= Floppy 9 wird aktiviert
ADR $$$$	= Adresse für ↑ A und GO [1]
ICI $$$$ $$$$	= Anfangs- und Endadresse für ICI-Pufferbereich [1]
BAT $$$$ $$$$	= Anfangs- und Endadresse für Batch-Pufferbereich [1]

1) Die Ziffern-Anzeige wird in hexadezimaler Schreibweise dargestellt.

Tabelle 2: Die Flags und die Adressen von ICI

Einen sehr leistungsfähigen Befehl stellt ICI zum Kopieren von Files zur Verfügung. Sie werden weiter unten lesen, daß sogar zwei Laufwerke unterstützt werden. Das Kopierkommando lautet: **pip quell-filename(,typ) (=ziel-filename(,typ))** . Das sieht auf den ersten Blick vielleicht verwirrend aus, ist aber extrem flexibel. An diesem Beispiel können wir Ihnen auch gleich die Syntax-Beschreibung der Tabelle 1 verdeutlichen. Unter „filename" ist ein beliebiger Name zu verstehen (der erste sollte in diesem Fall aber auch auf der Diskette vorhanden sein) und unter „typ" die Kennung p,s oder u. Sollten Sie in der Lage sein, mit zwei Laufwerken zu arbeiten — und das entsprechende Flag gesetzt haben (siehe unten) —, brauchen Sie nur **pip filename** einzugeben. Das File wird von dem einen Laufwerk auf das andere unter dem gleichen Namen kopiert. Arbeiten Sie nur mit einem Laufwerk, werden Sie zwischendurch zum Diskettenwechsel aufgefordert. Wie der Syntax zu entnehmen ist, können Sie mit dem Kopiervorgang auch gleichzeitig das File umbenennen. Eine dritte Variante besteht in der Möglichkeit, beim Kopiervorgang den File-Typ zu ändern. So können Sie beispielsweise aus einem Programm ein sequentielles File erzeugen, indem Sie unterschiedliche File-Typen angeben.

ICI und sein Zeilen-Editor

CRSR RECHTS	= CRSR ein Zeichen nach rechts
CRSR LINKS	= CRSR ein Zeichen nach links
DEL	= löscht Zeichen vor CRSR
INS	= schiebt Text ab CRSR ein Zeichen nach rechts
HOME	= CRSR zum Zeilenanfang
CLR/HOME	= löscht ganze Zeile
CTRL + E	= CRSR auf letztes Zeichen
CTRL + W	= letzte Zeile wiederholen
RETURN	= schließt Eingabe ab
← + RETURN	= bricht die Erstellung einer Batch-Datei ab
@ + RETURN	= beendet eine Batch-Datei

Tabelle 3: Der Zeilen-Editor von ICI

Auch an die direkte Speicherbeeinflussung haben wir gedacht. Mit **poke $$$$ $$** können Sie analog dem BASIC-Befehl POKE in eine Speicheradresse einen bestimmten Wert schreiben. Daß die Eingabe in hexadezimaler Schreibweise erfolgen muß, haben wir schon erwähnt. Die wie immer durch (mindestens) ein Space getrennte zweite Werteingabe ist aber in diesem Ausnahmefall nur zweistellig. Richtig spannend wird dieser Befehl erst im Rahmen von Batch-Dateien (siehe unten).

Ein im BASIC-Bereich schlummerndes Programm können Sie mit **run** direkt starten. Dieses Kommando benötigt weder Parameter noch weitere Erläuterungen.

Mit den Befehlen **setadr $$$$** , **setbat $$$$ $$$$** und **setici $$$$**

$$$$ werden ICI-typische Adressen gesetzt, deren Bedeutung wir bei den Flags näher beschreiben werden.

Um ein BASIC-Programm zu speichern, brauchen Sie unter ICI nur das Kommando **save name(,typ)** einzugeben. Dieser Befehl ist identisch mit dem BASIC-Befehl SAVE. Natürlich können Sie für „name" jeden beliebigen Namen eingeben. Sie sollten aber berücksichtigen, daß Sie unter ICI einige wenige Namen nicht wieder laden können. Wir haben bereits geschildert, daß ICI Eingaben zuerst als Befehl interpretiert und nur, wenn es sich nicht um einen Befehl handelt, versucht, ein Programm zu laden. Wenn Sie Ihr Programm beispielsweise „dir" nennen, können Sie es von ICI aus nicht mehr laden, da die Eingabe **dir** als Befehl ausgeführt würde.

Auch wenn wir die Flags bisher nur angedeutet haben, wollen wir Ihnen doch nicht verschweigen, daß Sie mit **status** den aktuellen Wert der Flags angezeigt bekommen. Auch dieses Kommando benötigt keine Parameter.

Wir hatten Ihnen versprochen, daß ICI weitere Möglichkeiten bietet, ein File direkt zur Anzeige zu bringen. Mit **type testdatei,s** wird die Datei im Klartext (ASCII-Format) auf dem Bildschirm ausgegeben. Mit diesem Befehl lassen sich unter anderem sehr einfach Batch-Dateien listen. Selbstverständlich können Sie dieses Kommando auch für andere File-Typen verwenden.

Sollte Ihre Datei im Bildschirmcode abgespeichert sein (zum Beispiel benutzt Vizawrite dieses Format), erhalten Sie eine lesbare Darstellung mit **vtype testdatei,p** . In diesem Fall sollten Sie die File-Kennung 'p' verwenden, weil Vizawrite die Dateien als Programme ablegt.

ICI kennt noch viele weitere Kommandos. Da diese aber nur in Verbindung mit Batch-Dateien interessant sind, werden wir diese erst weiter unten beschreiben.

. . . und die Flags

Die Flags (Flaggen) beeinflussen fast jedes Kommando. Die Befehle sind eigentlich auch nur in Verbindung mit den Flags zu beschreiben; aber irgendwo mußten wir ja anfangen. Sollte also ein Kommando nicht das erwartete Ergebnis bringen, schauen Sie sich die Grundeinstellungen genau an.

Zuerst zu der Frage, wohin ein Programm geladen wird. Mit **+a** sagen Sie ICI, daß das Programm absolut (in BASIC würde man ,8,1 sagen) geladen wird, mit **-a** wird das Programm an den BASIC-Anfang geladen (entspricht ,8 in BASIC), wobei ein veränderter BASIC-Anfang erkannt und berücksichtigt wird, und mit **†a** wird das Programm an die Adresse, die mit SETADR festgelegt wurde, geladen (einen entsprechenden BASIC-Befehl gibt es nicht).

Wir hatten schon angedeutet, daß gerade beim PIP-Kommando die Frage wichtig ist, ob Sie mit einem oder mit zwei Laufwerken arbeiten. Dieses teilen Sie ICI mit **=c** für ein Laufwerk beziehungsweise mit **/c** für zwei Laufwerke mit.

Hinter einem weiteren Flag versteckt sich eine sehr nützliche Eigenschaft von ICI. Sie können nämlich verlangen, daß alle Ausgaben, die auf dem Bildschirm erscheinen, parallel auch auf den Drucker ausgegeben werden. Das gilt unter anderem natürlich auch für DUMP und TYPE. Mit **+e** aktivieren Sie dieses Echo, und mit **-e** schalten Sie es wieder aus.

Sie können auf den BASIC-Direktmodus auch ganz verzichten. Erwarten Sie das von ICI, sollten Sie **+x** eingeben. ICI wird immer dann, wenn ein Programm abbricht oder beendet wird, sich selbständig einschaltet. Den Direktmodus wieder zulassen können Sie mit **-x** . Wenn Sie das Kommando BASIC eingeben, wird dieses Flag automatisch auf '-x' gesetzt, da der Befehl sonst wirkungslos wäre.

ICI unterstützt die Geräteadressen 8 und 9. Umschalten können Sie mit **8:** beziehungsweise mit **9:** . Beachten Sie bitte, daß eine Umschaltung bis auf Widerruf gilt und alle Kommandos sich auf die eingestellte Diskettenstation beziehen.

Mit dem Kommando SET können Sie verschiedene Adressen beeinflussen. Hier nun die Beschreibung dieser Adressen. Mit **adr $$$$** legen Sie die Adresse fest, die für den Befehl GO (ohne Adresse) und für †a relevant ist, mit **ici $$$$ $$$$** den ICI-Pufferbereich und den Speicherbereich für das Kommando BSAVE, und schließlich gibt es noch einen Pufferbereich für die Batch-Dateien, den Sie mit **bat $$$$ $$$$** beeinflussen können. Die Batch-Dateien müssen ja irgendwo im Speicher liegen. Damit sie nicht allzu störend sind, dürfen diese auch unter dem Kernal, also ab der Adresse E000 (hex) stehen. Daß '$$$$' jeweils für eine vierstellige Hexzahl steht, hatten wir schon erwähnt.

Batch-Dateien

Nun zu einem ganz besonderen Lekkerbissen: den Batch-Dateien und deren Verarbeitung. In der Computer-Literatur ist auch der Begriff der Stapelverarbeitung geläufig, der die ganze Angelegenheit recht gut beschreibt. Batch-Dateien vereinfachen den Umgang mit wiederholbaren oder öfter gebrauchten ganzen Befehlssequenzen. Vereinfacht: Sie erstellen eine Ansammlung von Befehlen, die in die-

```
print start.b aktiv
print
print taste = farbe.b
print STOP bricht ab
wait
batch farbe.b
```

Listing 1: Batch-File start.b

sem Fall nicht ausgeführt, sondern in eine Datei geschrieben werden. Erst zu einem späteren Zeitpunkt werden unter einem Namen alle Befehle aufgerufen und der Reihenfolge nach abgearbeitet.

Bei der Erstellung der Batch-Datei können Sie den ICI-Zeilen-Editor oder einen geeigneten ASCII-Editor (zum Beispiel den von INPUT-ASS) verwenden. Wir beschreiben zuerst die Handhabung mit dem ICI-Zeilen-Editor.

Sie müssen ICI mitteilen, daß Sie eine Batch-Datei schreiben wollen. Hierzu dient das Kommando **edit name,s** . Unter "name" wird die Datei später auf Diskette geschrieben und auch zur Abarbeitung wieder aufgerufen.

Das eingestellte Laufwerk läuft kurz an, und statt des '>' erscheint nun ein ':' vor dem Cursor. Sie können nun wie gewohnt die bereits bekannten (oder die noch folgenden) Kommandos oder Flag-Beeinflussungen eingeben, nur mit dem Unterschied, daß die Befehle nicht ausgeführt werden, sondern nach jedem RETURN eine weitere Zeile erscheint. Haben Sie alle Befehle eingegeben, teilen Sie ICI durch die Eingabe des @ mit anschließendem RETURN mit, daß Sie die Batch-Datei abschließen wollen. Möchten Sie die Erstellung abbrechen, also das bis dahin Eingetippte ungültig machen, geben Sie einfach ← ein, und Sie befinden sich augenblicklich wieder im "ICI-Normal-Modus".

Bevor wir Ihnen weitere Kommandos erklären, zwei Anmerkungen zum prinzipiellen Abarbeiten der Batch-Dateien. Batch-Dateien können sowohl weitere Batch-Dateien als auch (wie im Listing 5) sich selber erneut aufrufen und, was ungeahnte Möglichkeiten eröffnet, Batch-Dateien werden weiterverarbeitet, auch wenn zwischendurch ein Programm eben aus dieser Batch- Datei gestartet und wieder beendet (oder abgebrochen) wurde.

Nun die bislang vorenthaltenen weiteren Befehle:

Wollen Sie eine Batch-Datei zur Abarbeitung aufrufen, geben Sie **batch name,s** ein. Wenn sich die Batch-Datei auf dem eingestellten Laufwerk befindet, wird sie in den Batch-Pufferbereich geladen und gestartet. Der Lauf der Abarbeitung kann durch Drücken der **STOP-Taste** abgebrochen werden. Diese Tatsache machen sich die Beispielprogramme (Listing 1 bis 4) zunutze, um aus der Verkettung der gegenseitigen Aufrufe ausbrechen zu können.

Wollen Sie während der Abarbeitung einige Hinweise auf dem Bildschirm haben, können Sie das Kommando **print text** verwenden, wobei für "text" jede beliebige Zeichenfolge — begrenzt nur durch die Zeilenlänge — stehen kann. (Von dieser Möglichkeit machen alle aufgeführten Listings Gebrauch.)

Da es sinnvoll sein kann, den Automatismus der Stapelverarbeitung hier und da anzuhalten, haben wir den Befehl **wait** vorgesehen. Hier passiert nichts anderes, als daß der Computer auf einen beliebigen Tastendruck wartet.

```
print farbe.b aktiv
print farbeinstellung
print alle eingaben
print 2stellig hex
print
print rahmen
poke d020 %
print hintergrund
poke d021 %
print zeichen
poke 0286 %
print
print taste = basic.b
print STOP bricht ab
wait
batch basic.b
```

Listing 2: Batch-File farbe.b

Ein weiteres Kommando dient der Dokumentation. Zeilen, die mit einem Semikolon beginnen, werden bei der Abarbeitung einfach nicht beachtet. Dies entspricht sinngemäß dem BASIC-Befehl REM.

Der letzte Befehl ist sehr mächtig. Eigentlich ist das Folgende kein Befehl, sondern ein Teil von einem Befehl, oder ein File-Name zum Laden, oder doch ein Befehl, eine Flag-Beeinflussung, oder alles zusammen oder nichts davon. Haben wir Sie jetzt gründlich verwirrt? Gemeint ist ein Platzhalter, der für alles(!), was inner-

```
print basic.b aktiv
print BASIC-speicher
print auf normal
poke 002b 01
poke 002c 08
print
print taste = ici.b
print STOP bricht ab
wait
batch ici.b
```

Listing 3: Batch-File basic.b

```
print ici.b aktiv
print ICI-werte
      setzen
print
-a
+x
print
print taste = lader.b
print STOP bricht ab
wait
batch lader.b
```

Listing 4: Batch-File ici.b

halb von ICI möglich ist, stehen kann. Wenn ICI bei der Interpretation der Zeile auf diesen Platzhalter (der übrigens als einfaches **%** eingegeben wird und an jeder Stelle stehen kann) stößt, erwartet er eine Eingabe, die genau dort eingefügt wird, wo dieses Zeichen stand. Sinnlos, aber logisch wäre folgende Eingabe: **d%r** (egal ob im Direktmodus oder innerhalb einer Batch-Datei). ICI würde sich mit einem Doppelpunkt und einem blinkenden Cursor melden, Sie könnten ein 'i' eingeben, und ICI würde das Directory anzeigen. Richtig, das ergibt keinen Sinn! Aber es macht deutlich, was wir meinen. Sinnvoller wäre es, innerhalb einer Batch-Datei, nachdem das Disketten-Inhaltsverzeichnis auf dem Bildschirm steht, das zu ladende Programm eingeben zu können (dies wird in Listing 5 dokumentiert).

Eingabe mit Komfort

Mit dem Zeileneditor können Sie natürlich keine einmal abgeschlossene Zeile editieren. Haben Sie sich verschrieben, müssen Sie die ganze Eingabe abbrechen und von vorne beginnen. Bei umfangreichen Dateien ist diese Methode nicht sehr glücklich; wer tippt schon 30 oder 40 Zeilen fehlerfrei.

Für die Erstellung einer Batch-Datei kann aber jeder Editor verwendet werden, der eine sequentielle Commodore-ASCII-Datei erzeugen kann; also nicht Vizawrite, dafür aber zum Beispiel INPUT-ASS. Sie müssen darauf achten, daß keine Leerzeilen in der Datei sind, da die Abarbeitung der Batch-Datei sonst an dieser Stelle abgebrochen wird. Auch darf der Doppelpunkt, der beim ICI-Zeileneditor erscheint, ebensowenig eingegeben werden, wie die Endemarkierung mit dem „Klammeraffen". Wenn Sie die Beispiele in diesem Artikel betrachten, haben Sie im Prinzip genau die syntaktisch richtige Schreibweise für einen Fullscreen-Editor.

Die Wahl der Qual

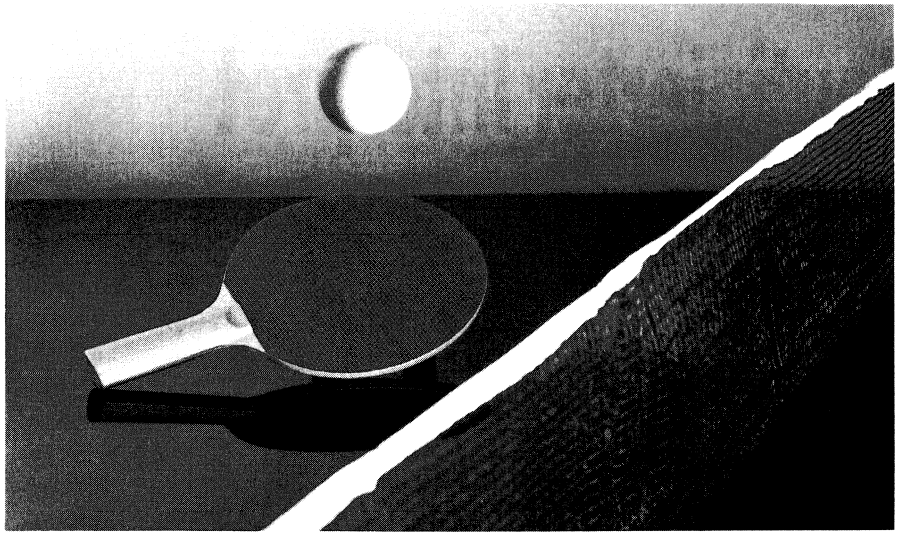
Sie haben nun die Wahl zwischen dem Kommando Interpreter ICI und der grafische Benutzeroberfläche JAM. Welche von beiden Methoden für Sie größere Vorteile bringt, können Sie selbst entscheiden. Beide Hilfsmittel haben Vor- und Nachteile. So ist JAM sicherlich einfacher zu bedienen, hat aber auch einen geringeren Befehlsumfang als ICI. Die besondere Stärke von ICI ist dagegen die Abarbeitung von Batch-Dateien, wobei ehrlicherweise eingestanden werden muß, daß die vorhandene Software nur bedingt für ein Batch-Processing geeignet ist. Aber: Was nicht ist, kann ja noch werden.

```
print lader.b aktiv
dir
print welches PRG
print laden?
%
batch lader.b
```

Listing 5: Batch-File lader.b

Grundsätzlich müßte durch ein direktes Patchen der im Speicher vorhandenen Programme (mit dem POKE-Kommando) in Verbindung mit einem absoluten Nachladen von Dateien das eine oder andere möglich sein. Wir arbeiten zum Beispiel daran, unserem INPUT-ASS die Stapelverarbeitung beizubringen. Eine Lösung können wir aber nicht versprechen.

Frank Börncke/WM



Schmettern und Schnippeln

Ping-Pong Classic

Klassisch, praktisch, gut: Video-Tischtennis. Eines der allerersten real existierenden Computerspiele wurde von unserem holländischen Autor Gerrit Knoef für den 64er neu aufgelegt.

Daß es bei diesem Spiel darum geht, den kleinen gelben Ball mit dem mehr oder weniger träge reagierenden Schläger auf dem Spielfeld zu halten, hat sich wahrscheinlich mittlerweile herumgesprochen. Wie das Spiel zu bedienen ist, entnehmen Sie am besten den Anweisungen im Menü. Falls Sie die Laufschrift nicht bis zum Ende lesen wollen, hier noch einmal die wichtigsten Hinweise: Der linke Spieler kann den Schläger mit einem Joystick in Port 2 bewegen, dem rechten Spieler ist Port 1 zugeordnet. Ersatzweise können beide Schläger per Tastatur gesteuert werden, dann gilt für den linken Spieler

w = hoch, s = runter,
für den rechten Spieler
* = hoch, ; (Semikolon) = runter.

Mit der Feuertaste beziehungsweise den Tasten x (links) oder / (rechts) kann dem Ball ein besonderer „Drive" gegeben werden. Das klappt aber nicht immer, schließlich gelingen einem auch im wirklichen Leben die Schmetterbälle und das „Schnippeln" nicht unbedingt auf Anhieb. Die RUN/STOP-Taste führt nach kurzer Bedenkzeit zurück ins Menü.

Der eigentliche Clou dieses Spieles liegt in einem gewissen Maß an Eigenaktivität. Betrachtet man zu lange das Eingangsmenü, spielt der Computer erst einmal gegen sich selbst. Dieser Modus „Computer gegen Computer" ist natürlich auch ausdrücklich wählbar und stellt unseres Erachtens eine an theoretischer Konsequenz nicht zu überbietende Weiterentwicklung der Idee des Videospiels an sich dar. JS

Routinierter Interpret

Der BASIC-Interpreter

Der Interpreter des C64 stellt mit einer Länge von 9 KByte schon ein recht komplexes Maschinensprache-Programm dar. Der Interpreter kann in zwei Bereiche geteilt werden. Das ist erstens der Bereich, der die zentrale Organisation verwaltet, und zweitens jener, der für jeden BASIC-Befehl eine Maschinenroutine zur Verfügung stellt. Wir wollen hier anhand des einfachen 'PRINT'-Befehls nur die prinzipielle Funktionsweise des ersten Teils betrachten. Ein Programm-Ablauf-Plan (PAP) soll uns dabei helfen, den Direkt-Modus und den Programm-Modus besser zu verstehen.

Maschinensprache ist nicht jedermanns Sache. Da aber ein Computer nur diese Sprache versteht, hat man sich bemüht, etwas zu entwickeln, das es erlaubt, sich dem Computer auch in einer anderen Sprache verständlich zu machen. Unter anderem hat man einen Interpreter entwickelt. Er wertet das Eingegebene aus, und wenn es sinnvoll ist, handelt er dementsprechend.

Sofort oder gleich?

Nachdem Sie den C64 eingeschaltet haben, erscheint die bekannte Einschaltmeldung und darunter ein 'READY'. Diese Meldung, auch Prompt genannt, sagt Ihnen, daß der C64 bereit ist, irgendwelche Eingaben von Ihnen zu empfangen. Was passiert eigentlich zu dieser Zeit? — Der Rechner befindet sich nach dem Einschalten in der sogenannten Interpreter-Eingabe-Warteschleife. Hier wird nur darauf gewartet, daß Sie Ihre Zeile eingeben. Diese Zeile wird Zeichen für Zeichen im Eingabepuffer abgelegt und dadurch auch für Sie sichtbar. Haben Sie alles eingegeben, teilen Sie das dem Rechner durch einen Druck auf die 'RETURN'-Taste mit. Daraufhin beginnt er Ihre Eingabe zu bearbeiten. Dazu kopiert er sie in einen speziellen Puffer und markiert das Ende mit einem Null-Byte.

Nun holt sich der Interpreter das erste Zeichen aus dem Eingabepuffer und überprüft, ob dieses Zeichen eine Ziffer ist (Bild 1). Ist es eine Ziffer, so wird die Zeile in den Interpreter-Code gewandelt. Die Umwandlungsroutine macht nichts anderes, als daß sie, wie in unserem Fall, den Befehl 'PRINT' mit einer Tabelle vergleicht. Diese Befehlstabelle enthält alle Befehle, die der C64 kennt. Findet der Interpreter also in dieser Tabelle den gesuchten Befehl, so wird für das mehrere Zeichen lange Wort ein Byte eingesetzt, das sogenannte Token. Anschließend wird die BASIC-Zeile in den Programmspeicher übernommen. Danach geht's wieder ab zur Eingabe-Schleife.

Dann lieber sofort . . .

War das erste Zeichen aus dem Puffer keine Ziffer, so wird die Zeile zwar auch erst in den Interpreter-Code gewandelt, aber anschließend direkt abgearbeitet (Direkt-Modus). Der Interpreter springt dazu in einen Programmteil, der den Interpreter-Code wieder Zeichen für Zeichen liest (Bild

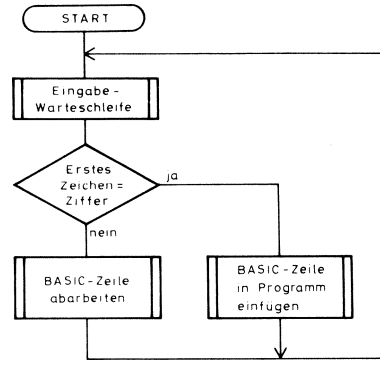


Bild 1: Nach Eingabe muß der Interpreter unterscheiden, ob die Zeile eingefügt oder ausgeführt werden soll.

2). Er fragt, nachdem er das erste Zeichen geholt hat, ob es eine Null ist. Falls ja, haben Sie nämlich nichts eingegeben, sondern nur 'RETURN' gedrückt. Das Ende der Zeile ist erreicht. Da der Rechner sich im Direkt-Modus befindet, gibt er „einfach" ein 'READY' aus und geht wieder zur Eingabe-Warteschleife.

Haben Sie aber etwas Sinnvolles eingegeben, wie zum Beispiel 'PRINT "HALLO"', ist das erste Zeichen das Token für den 'PRINT'-Befehl (das Token für den 'PRINT'-Befehl ist die Zahl 153). Anhand dieses Zeichens sucht er in einer Tabelle nach der Adresse, wo der 'PRINT'-Befehl zu finden ist, und springt anschließend dort hin. Der Befehl wird abgearbeitet.

Der Interpreter prüft nun, ob der Befehl durch einen Doppelpunkt abgeschlossen wurde. Dann wäre das Zeilenende noch nicht erreicht, und der Interpreter müßte noch weitere Befehle bearbeiten. In unserem Fall ist es aber eine Null, denn Sie haben ja nur den einen Befehl eingegeben. Das bedeutet, daß das Zeilenende erreicht ist. Der Interpreter weiß ja, daß er sich im Direkt-Modus befindet, gibt ein 'READY' aus und springt wieder zur Eingabe-Schleife.

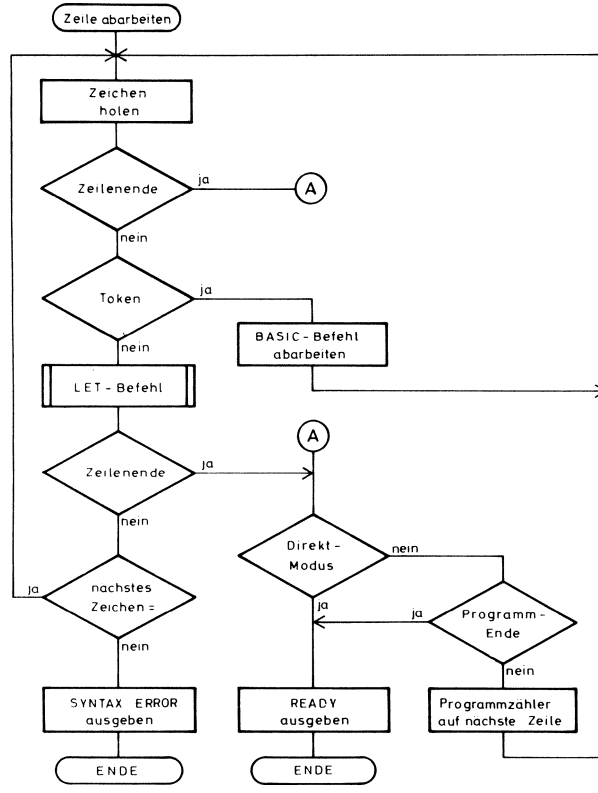
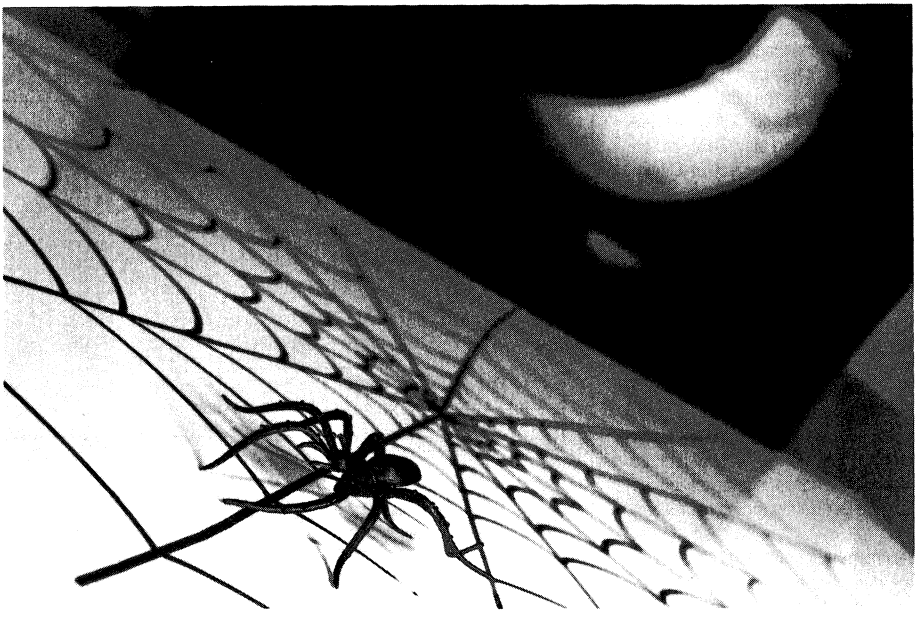


Bild 2: Nach diesem Schema wird eine „tokenisierte" BASIC–Zeile abgearbeitet.

. . . oder doch lieber gleich?

Was geschieht aber, wenn man ein Programm schreibt und es anschließend mit 'RUN' startet? Geben Sie ein: '10 PRINT "HALLO"'. Auch hier wird die eingegebene Zeile erst in den Interpreter-Code gewandelt. Danach wird sie in das aktuelle BASIC-Programm eingefügt, und wie schon erwähnt springt der Interpreter dann erneut in die Eingabe-Schleife. Erst wenn Sie den Befehl 'RUN' eingegeben haben, wird der Befehl 'RUN' abgearbeitet (im Direkt-Modus). Auch für diesen Befehl ist eine Maschinensprache-Routine zuständig. Sie enthält unter anderem einige Maschinenbefehle, die dem Interpreter sagen, daß er sich jetzt im Programm-Modus befindet.

Doch auch in diesem Modus springt der Interpreter in den Programmteil, in dem eine BASIC-Zeile abgearbeitet wird. Wiederum wird, nachdem das erste Zeichen geholt wurde, gefragt, ob das Zeilenende schon erreicht ist (Zeichen=Null), was ja nicht der Fall ist. Auch hier findet der Interpreter als erstes Zeichen eine 153 für den 'PRINT'-Befehl, das er als Token erkennt. Die Abarbeitung einer Programmzeile wird also von derselben Interpreter-Routine bewerkstelligt, die auch für den Direkt-Modus zuständig ist. Der einzige Unterschied besteht darin, daß beim Erkennen des Zeilenendes erst mal kein 'READY' ausgegeben werden darf. Vielmehr muß nachgesehen werden, ob nach dieser Zeile noch eine weitere im Speicher steht, die natürlich nicht vergessen werden darf. In unserem Beispiel gibt es aber keine nächste Zeile, der Interpreter erkennt das Programmende und kann dann endlich „sein" 'READY' auf den Bildschirm bringen. kfp



Spinnen Sie mal

Spiel: Spider

„Letzte Nacht habe ich wieder von Spinnen geträumt!"—„Igittigitt!"— Geht es Ihnen auch so, daß sie Spinnen als ekelhaft empfinden? Zumindest lassen Sie einen erschauern, diese behaarten, kleinen Vielbeiner. Nicht so „Spider".

Sie steuern eine kleine Spinne — mittels Joystick oder Tastatur — durch ein verwirrendes Labyrinth. Auf dem Weg zum Endpunkt müssen Sie 24 Schlüssel aufsammeln, denn die brauchen Sie, um eine Belohnung zu erhalten. Stellen Sie sich diese Sammelaktion aber nicht zu einfach vor, denn auf dem gesamten Weg sind schlimme Fallen in das Labyrinth eingebaut, die die Spinne nicht überspringen kann. Dabei hilft nur ein Zurück, sofern das möglich ist, denn an einer bestimmten Stelle landet sie sogar in einer Sackgasse. Da geht dann nichts mehr. An einigen Stellen sind kleine Löcher, durch die sie schlüpfen kann und so an eine andere Position im Labyrinth gelangt.

Taste	Funktion
Z	Spinne läuft nach links
/	Spinne läuft nach rechts
SPACE	Spinne springt

Mehr brauchen Spinnen nicht zu können.

Falls zu irgendeinem Zeitpunkt überhaupt nichts mehr läuft, Überreden Sie die Spinne einfach mal zu springen. In einigen Bereichen sind Überhänge oder Treppen eingebaut, an denen sie emporkrabbeln kann. Wo sich diese Stellen im Labyrinth befinden, wird nicht verraten. Im übrigen ist „Spider" ein nettes Spiel mit einer niedlichen kleinen Spinne, die Sie bestimmt bald alle gern haben. kfp

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).

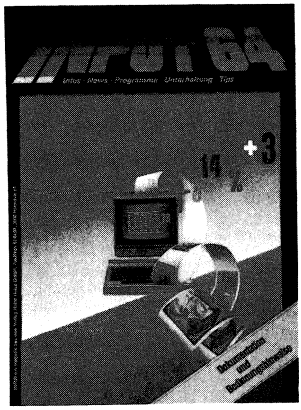
PS! In der Ausgabe 6/85 haben wir das Programm RECORDER-JUSTAGE veröffentlicht, das die Einstellung des Datenrecorders zum Kinderspiel macht.

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 3. August 87 auf Kassette und Diskette an Ihrem Kiosk: INPUT 64, Ausgabe 8/87



Wir bringen unter anderem:

INPUT-Calc 64/128

Tabellenkalkulation für C64 UND C128

Ein professionelles Anwenderprogramm, mit dem Sie sowohl auf dem C64 als auch auf dem C128 im 80-Zeichen-Modus arbeiten können. Das Programm ermittelt selbständig, welche Rechnerkonfiguration vorliegt. Sie brauchen dieses Anwenderprogramm nur zu laden und zu starten. Beim C64 verfügen Sie über 10 KByte, beim C128 sogar über 24 KByte Speicherkapazität. INPUT-Calc 64/128 ist zu INPUT-Calc aus INPUT 64-Ausgabe 10/86 voll kompatibel. Als Anwender werden Sie sich über die verbesserte Benutzerführung freuen. Neben den üblichen Rechenfunktionen können Sie auch auf Funktionen wie Maximum, Minimum, Mittelwert, Matrizen, Sortieren und Blockoperationen zugreifen. Ein professionelles Arbeitsmittel im Preis einer INPUT 64-Ausgabe enthalten.

BINGO

Spiel um Zahlen

Kenner der englischen Leidenschaften erinnern sich der mysteriösen „Bingo"-Rufe, die zur Geräuschkulisse englischer Seebäder gehören. Ein fesselndes Spiel, das in deutschsprachigen Gebieten bislang weniger bekannt ist. In der nächsten Ausgabe können Sie die auf den ersten Blick undurchschaubaren Regeln dieses Spiels kennenlernen. Nach dieser Ausgabe können Sie bei Ihrem nächsten Englandbesuch mitreden.

und außerdem:

64er Tips, Englische GRAMmatik Teil 8, Assembler-Schule Teil 6 u.v.a.m.

c't – Magazin für Computertechnik

Ausgabe 8/87 – ab 17.7.1987 am Kiosk

Projekte: Der schnellere 68000 – Austauschplatine mit 68020 und 68881 * Software-Know-how: Können Computer Sprache verstehen? – Syntaxanalyse in Lisp * Musikalischer Rechner: Stochastische Kompositionen * Paßwort-Knobeleien * Praxistip: AT-Floppies am PC * Grafik-Toolbox * u.v.a.m.

elrad – Magazin für Elektronik

Ausgabe 7/87 – ab 29.6.1987 am Kiosk

Elektrostat – allerbeste Audio-Qualität im Selbstbau * Marktfeature: Hf-Bauelemente – Gewußt Wo * Remixer – 14-Kanal-Mischpult für Homerecording-Fans * elrad-Laborblätter: A/D-Wandler * Bauanleitung: EPROM-Codeschloß * Sonderteil „Messen & Testen" mit über 20 Schaltungen * 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 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:
Erb-Verlag GmbH & Co KG
Abt. Zeitschriftenvertrieb
z. Hd. Frau Pekatschek
Amerlingstraße 1
A-1061 Wien
Telefon: (00 43 2 22) 56 62 09
(00 43 2 22) 57 94 98
(00 43 2 22) 57 05 25
Jahresabonnement: 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: S. Wustmann, Dortmund
Titel - Grafik und -Musik: Tim Pritlove, Fabian Rosenschein
Betriebssystem: 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 Diskette DM 198,— inkl. Versandkosten und MwSt.

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

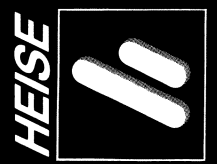
Ich wünsche Abbuchung der Abonnement-Gebühr von meinem nachstehenden Konto. Die Ermächtigung zum Einzug erteile ich hiermit.

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