Einsteigen - Verstehen - Beherrschen

DM 3,80 öS 30 sfr 3,80

computer kurs

Heft 38

Einbruch in Datennetze

Der PC Tandy 1000

Fischertechnik-Roboter

Jagd auf Chiffren

LED-Anzeige aus Dioden

Ein wöchentliches Sammelwerk

computer kurs

Heft 38

Inhalt

WIE SIE JEDE WOCHE IHR HEFT BEKOMMEN

Computer Kurs ist ein wöchentlich erscheinendes Sammelwerk. Die Gesamtzahl der Hefte ergibt ein vollständiges Computer-Nachschlagewerk. Damit Sie jede Woche Ihr Heft erhalten, bitten Sie Ihren Zeitschriftenhändler, Computer Kurs für Sie zu reservieren.

Zurückliegende Hefte
Ihr Zeitschriftenhändler besorgt Ihnen gerne zurückliegende Hefte. Sie können sie aber auch direkt beim Verlag bestellen.

Deutschland: Das einzelne Heft kostet DM 3,80. Bitte füllen Sie eine Postzahlkarte aus an: Marshall Cavendish Int. Ltd. (MCI), Sammelwerk-Service, Postgiroamt Hamburg 48064-202, Postfach 105703, 2000 Hamburg 1, Kennwort: Computer Kurs

Österreich: Das einzelne Heft kostet öS 30. Bitte füllen Sie eine Zahlkarte aus an: Computer Kurs, Wollzeile 11, 1011 Wien, Postscheckkonto Wien 7857201 oder legen Sie Ihrer Bestellung einen Verrechnungsscheck bei. Kennwort: Computer Kurs.

Schweiz: Das einzelne Heft kostet sfr 3,80. Bitte wenden Sie sich an Ihren Kiosk; dort werden Sie jederzeit die gewünschten Exemplare erhalten.

Abonnement

Sie können Computer Kurs auch alle 2 Wochen (je 2 Ausgaben) per Post zum gleichen Preis im Abonnement beziehen. Der Abopreis für 12 Ausgaben beträgt DM 45,60 inkl. Mwst., den wir Ihnen nach Eingang der Bestellung berechnen. Bitte senden Sie Ihre Bestellung an: Marshall Cavendish Int. Ltd. (MCI), Sammelwerk Service, Postgiroamt Hamburg 86853-201, Postfach 105703, 2000 Hamburg 1, Kennwort: Abo Computer Kurs. Bitte geben Sie an, ab welcher Nummer das Abo beginnen soll und ob Sie regelmäßig für jeweils 12 Folgen einen Sammelordner wünschen. Bei Bestellungen aus Österreich oder Schweiz senden Sie Ihren Auftrag bitte auch an die Hamburger Adresse. Berechnung und Zahlung erfolgen in Landeswährung zum Ladenpreis.

WICHTIG: Bei Ihren Bestellungen muß der linke Abschnitt der Zahlkarte Ihre vollständige Adresse enthalten, damit Sie die Hefte schnell und sicher erhalten. Überweisen Sie durch Ihre Bank, so muß die Überweisungskopie Ihre vollständige Anschrift gut leserlich enthalten.

SAMMELORDNER
Sie können die Sammelordner entweder direkt bei Ihrem Zeitschriftenhändler kaufen (falls nicht vorrätig, bestellt er sie gerne für Sie) oder aber Sie bestellen die Sammelordner für den gleichen Preis beim Verlag wie folgt:

Deutschland: Der Sammelordner kostet DM 12. Bitte füllen Sie eine Zahlkarte aus an: Marshall Cavendish International Ltd. (MCI), Sammelwerk-Service, Postgiroamt Hamburg 48064-202, Postfach 105703, 2000 Hamburg 1, Kennwort: Sammelordner Computer Kurs.

Österreich: Der Sammelordner kostet öS 98. Bitte füllen Sie eine Zahlkarte aus an: Computer Kurs Wollzeile 11, 1011 Wien, Postscheckkonto Wien 7857201 oder legen Sie Ihrer Bestellung einen Verrechnungsscheck bei. Kennwort: Sammelordner Computer Kurs

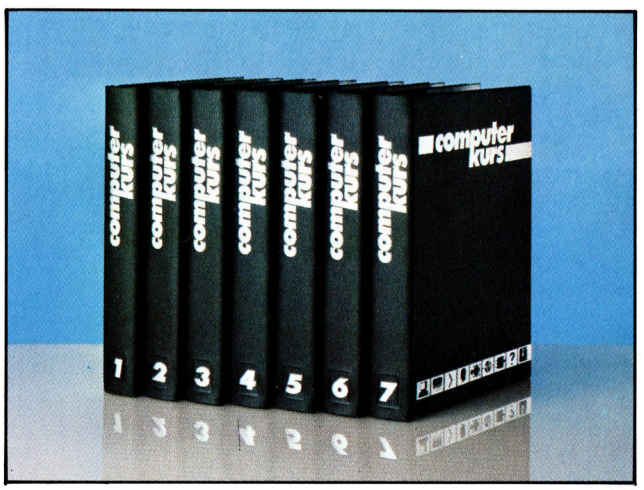
Schweiz: Der Sammelordner kostet sfr 15. Bitte wenden Sie sich an Ihren Kiosk; dort werden Sie jederzeit die gewünschten Exemplare erhalten.

INHALTSVERZEICHNIS

Alle 12 Hefte erscheint ein Teilindex. Die letzte Ausgabe von Computer Kurs enthält den Gesamtindex — darin einbezogen sind Kreuzverweise auf die Artikel, die mit dem gesuchten Stichwort in Verbindung stehen.

Redaktion: Winfried Schmidt (verantw. f. d. Inhalt), Elke Leibinger, Susanne Brandt, Uta Brandl (Layout), Sammelwerk Redaktions-Service GmbH, Paulstraße 3, 2000 Hamburg 1

Vertrieb: Marshall Cavendish International Ltd., Heidenkampsweg 74, 2000 Hamburg 1



Druck: E. Schwend GmbH, Schmollerstraße 31, 7170 Schwäbisch Hall

Der Konkurrent

Weder auf dem Markt für Heimgeräte noch im kommerziellen Bereich gelang der Tandy Corporation bisher der echte Durchbruch. Mit dem Tandy 1000 bietet die Firma nun eine preisgünstige Alternative zum IBM PC.

Obwohl die Tandy Corporation mit ihrem TRS-80 einer der ersten Anbieter von Microcomputern war, erlangte sie nie den internationalen Erfolg, den die Rivalen der ersten Stunde – Commodore auf dem Heimsektor und Apple auf dem kommerziellen Gebiet erobern konnten. Nach einer Reorganisation versucht Tandy nun zwei neue Strategien, um den kommerziellen Markt zu erobern. Durch die Partnerschaft mit ACT (die Hersteller des Apricot) brachte Tandy das Angebot seiner europäischen Computerzentren mit der Apricot-Gerätereihe auf den neuesten Stand. Außerdem stellte die eigene Entwicklungsabteilung ein neues Gerät vor.

Schon seit langem ist man sich in der Computerindustrie darüber im klaren, daß eine preisgünstige IBM-PC-kompatible Maschine große Erfolgschancen hat. Ein derartiges Gerät würde nicht nur auf kommerziellem Gebiet großen Anklang finden, sondern auch auf dem amerikanischen Heimcomputermarkt, wo Geräte angeboten werden, die man in Europa – aufgrund der hohen Preise – als rein kommerziell einstuft.

Der Tandy 1000 ist vollständig IBM-kompatibel. Die Grundausstattung mit einem Diskettenlaufwerk und 128 KByte Arbeitsspeicher kostet etwa 5000 Mark. Tandy verspricht sich von dieser rigorosen Preisunterbietung des IBM PC einen wesentlichen Marktvorteil.

In diesem Artikel untersuchen wir den Tandy 1000 mit zwei Laufwerken und 256 KByte Speicher. Die Ähnlichkeit mit dem IBM PC ist auch am Design zu erkennen. Das System besteht aus einem großen Gehäuse mit Computer, Schnittstellen und Diskettenlaufwerken, einem separaten Monitor und einer beweglichen Tastatur.

Wie bei dem IBM PC wird der Eindruck von Solidität und Zuverlässigkeit vermittelt. Die Tastatur besteht aus Schreibmaschinentasten mit Griffmulden. Die unter dem Keyboard installierten Füße ermöglichen einen Neigungswinkel bis zu 15 Grad.

Obwohl der Tandy alle Tasten besitzt, die für die Arbeit mit IBM-Programmen erforderlich sind, wurden sie anders angeordnet als auf dem IBM PC. Daraus ergeben sich Vor- und Nachteile. Trotz des gelungenen Designs gab die Tastatur des IBM Anlaß zur Kritik, da wichtige Tasten, wie Shift und Alternate, unglücklich positioniert waren und die Tasten für Return und Control durch ihre im Verhältnis geringe Größe leicht verfehlt werden können.

Der Tandy 1000 ist eine IBM PC-kompatible Maschine. Wie der IBM PC verwendet dieser Computer einen 8088-Prozessor und die Standard 5 1/4-Zoll-Diskettenlaufwerke. Das Bild zeigt eine Geräteversion mit Doppellaufwerk und zusätzlichen 128 KByte Arbeitsspeicher. Auf dem Monitor ist der Flugsimulator von Microsoft zu sehen, der für den IBM PC geschrieben wurde. Die Fähigkeit, dieses Programm fehlerlos verarbeiten zu können, ist ein Beweis für die Kompatibilität des Gerätes.

Verbesserte Tastatur

Tandy hat diese Kritik bei der Konstruktion seiner Tastatur berücksichtigt. Die -Taste, die – zusammen mit der Shift-Taste viele Probleme verursachte, wurde völlig herausgenommen. Ihre Funktionen liegen nun auf den Tasten (SHIFT) 4 und (SHIFT) 7 des Zehnerblocks.

Auch die Tasten für Alternate und Caps Lock befinden sich nicht neben der Leertaste, sondern im Bereich des Zehnerblocks bzw.

links unter den Control-Tasten. Durch diese Änderungen ist die Bedienung der Tastatur weitaus einfacher als auf dem IBM PC.

Natürlich muß man sich umstellen, wenn man den eigenwilligen Aufbau der IBM-Tastatur gewöhnt war. Die Tastatur ist jedoch auch für Computerneulinge leicht zu bedienen und gehört zu den besten des Marktes.

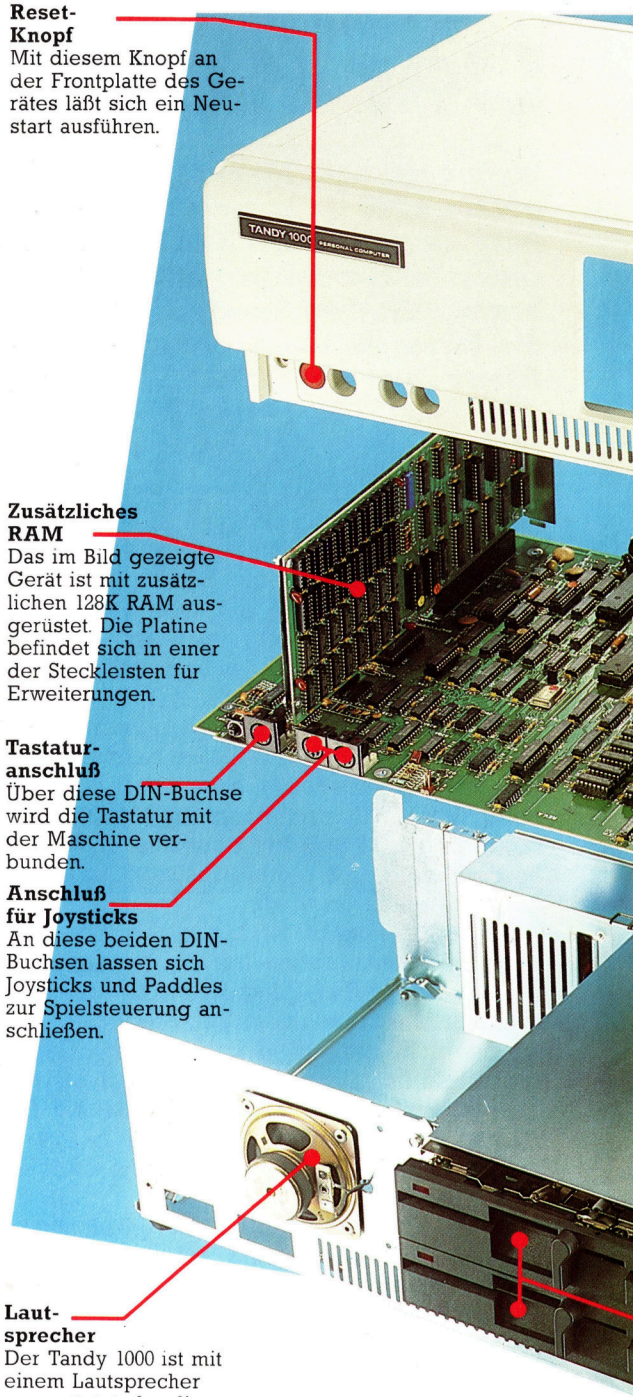
Tandy hat aber noch weitere Änderungen eingeführt. So wurden die Insert- und Delete-Tasten auf + und − verlegt und befinden sich damit über der Tastatur statt darunter. Auch wurden zwischen Tastatur und Zehnerblock zusätzliche Tasten eingebaut, darunter Hold, Print, Break und Cursortasten zum Editieren. Die Funktionstasten, die sich auf dem IBM links der Tastatur befanden, liegen bei diesem Gerät in einer Reihe oberhalb des Tastenfeldes. Zwar sind sie dort weniger gut zu erreichen, doch bietet der Tandy als Ausgleich zwei zusätzliche Funktionstasten.

Erweiterungsmöglichkeiten

Der Computer selbst befindet sich, ähnlich wie bei IBM, in einem kastenförmigen Gehäuse. Auf der rechten Seite liegen die beiden 5 1/4-Zoll-Laufwerke (die Standardversion enthält nur ein Laufwerk, die Vorrichtung für den Einbau eines zweiten sind jedoch vorhanden). Die Laufwerke arbeiten zwar leiser als die IBM-Geräte, in der Zugriffsgeschwindigkeit zeigen sich jedoch kaum Unterschiede. Lästig ist, daß die Disketten beim Entriegeln der Schächte nicht wie sonst üblich automatisch herausspringen, sondern per Hand herausgezogen werden müssen.

Unterhalb des Gehäusevorsprungs der Vorderseite befinden sich links der Lüftungsschlitze zwei sechspolige DIN-Buchsen, über die sich Joysticks und andere Steuergeräte anschließen lassen. Daneben liegt der orangefarbene Reset-Knopf für den Neustart des Systems. Obwohl er bequem zu erreichen ist,

Die Rückansicht
Außer einer Centronics-Druckerschnittstelle hat der Tandy 1000 Anschlußmöglichkeiten sowohl für RGB- als auch Composite-Video. Außerdem gibt es eine Audiobuchse, mit der sich das Gerät an eine externe HiFi-Anlage anschließen läßt.



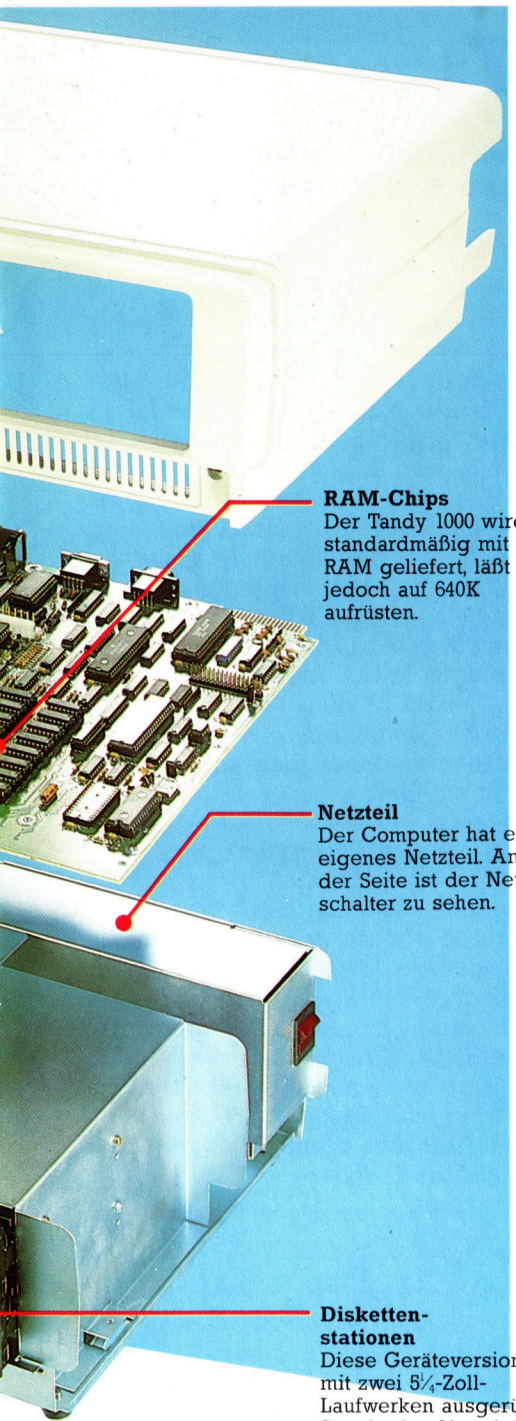
Reset-Knopf
Mit diesem Knopf an der Frontplatte des Gerätes läßt sich ein Neustart ausführen.

Zusätzliches RAM
Das im Bild gezeigte Gerät ist mit zusätzlichen 128K RAM ausgerüstet. Die Platine befindet sich in einer der Steckleisten für Erweiterungen.

Tastaturanschluß
Über diese DIN-Buchse wird die Tastatur mit der Maschine verbunden.

Anschluß für Joysticks
An diese beiden DIN-Buchsen lassen sich Joysticks und Paddles zur Spielsteuerung anschließen.

Lautsprecher
Der Tandy 1000 ist mit einem Lautsprecher ausgerüstet, der die Möglichkeiten der Klangerzeugung des MS-BASIC zum Tragen bringt.



RAM-Chips
Der Tandy 1000 wird standardmäßig mit 128K RAM geliefert, läßt sich jedoch auf 640K aufrüsten.

Netzteil
Der Computer hat ein eigenes Netzteil. An der Seite ist der Netzschalter zu sehen.

Diskettenstationen
Diese Geräteversion ist mit zwei 5¼-Zoll-Laufwerken ausgerüstet. Standardmäßig wird die Maschine jedoch nur mit einer Floppy geliefert.

sitzt er zu tief im Gehäuse, um irrtümlich bedient werden zu können.

Auf der Rückseite des Computers sind die Schnittstellen für Peripheriegeräte untergebracht. Der Netzanschluß befindet sich auf der linken Seite — oberhalb der Centronics-Schnittstelle. Über zwei siebenpolige D-Buchsen lassen sich ein Lichtgriffel und ein Farbmonitor anschließen. Neben dem Ventilator liegt ein Ausgang für Composite-Video, der für einen monochromen Monitor oder einen Videomonitor mit einem Composite-Colour-Signal geeignet ist, und ein Audioanschluß, mit dem sich die Töne des Gerätes über ein externes HiFi-System verstärken lassen. Auf der rechten Seite sind drei Erweiterungssteckleisten eingebaut, über die sich zusätzlich Peripherie-Interfaces oder Speicherplatinen anschließen lassen.

Im Vergleich

Tandy betont, daß seine Maschine über viele Anschlußmöglichkeiten verfügt, die ein Standard-IBM PC nicht bietet. Dies stimmt zwar, doch scheint die Firma ihr Gerät nur mit dem absoluten Minimum für den kommerziellen Einsatz ausgestattet zu haben. So besitzt die Standardversion des Tandy 1000 nur 128 KByte, die für einige der neueren integrierten Programmpakete längst nicht mehr ausreichen. Auch muß die RS232-Schnittstelle — ohne die eine Datenübertragung im kommerziellen Bereich kaum denkbar ist — zusätzlich nachgerüstet werden. Ein Plus ist allerdings das integrierte Softwarepaket „Deskmate", das im Lieferumfang des Tandy 1000 enthalten ist. Es beinhaltet unter anderem eine brauchbare Textverarbeitung.

Das Problem beim Bau einer IBM PC-kompatiblen Maschine liegt nicht in der Beschaffung der 8088-CPU oder des Betriebssystems MSDOS. Beide Komponenten sind frei im Markt verfügbar. Die Schwierigkeit ist das BIOS (Basic Input/Output System), das unter dem Copyright von IBM läuft. Da viele Programme die Routinen des BIOS direkt ansprechen, müssen kompatible Maschinen die gleichen Einsprungadressen enthalten, damit der Programmablauf nicht gestört wird.

Das von der Phoenix Compatibility Corporation erstellte BIOS des Tandy 1000 ist in dieser Hinsicht eine Ausnahme. Nicht nur das extrem komplizierte Lotus 1-2-3 funktioniert darauf (obwohl 256 K dafür etwas eng sind), sondern auch Wordstar 2000 und dBase II.

Gute Erfolgschancen

Mathematische Abläufe im BASIC sind auf dem Tandy jedoch ebenso langsam wie auf dem IBM PC. Das Gerät benötigt für das Zählen von 0 bis 1000 volle sechs Sekunden und ist damit langsamer als viele Acht-Bit-Micros.

Der Tandy 1000 ist zweifellos eine brauchbare Alternative für kommerzielle Anwender, die von der großen Menge guter Software profitieren wollen, die für den IBM PC geschrieben wurde. Wegen der Hardwaregrenzen, die sich aus der IBM-Kompatibilität ergaben, und der Notwendigkeit, den Preis so niedrig wie möglich halten zu müssen, bietet der Tandy 1000 kaum neue Eigenschaften. Wenn jedoch IBM seine Hochpreispolitik beibehält, ist der Tandy 1000 ein Gerät mit Erfolgschancen.

Tandy 1000

ABMESSUNGEN
420 x 335 x 150 mm

ZENTRALEINHEIT
Intel 8088 mit 4,77 MHz

SPEICHER-KAPAZITÄT
128K RAM, auf 640K erweiterbar.

BILDSCHIRM-DARSTELLUNG
Textdarstellung: 80 x 25 Zeichen; hohe Auflösung: 640 x 200 Pixel. Von 16 verfügbaren Farben kann das Gerät acht gleichzeitig darstellen.

SCHNITTSTELLEN
Buchsen für je einen Composite-Video- und RGB-Monitor; Druckerschnittstelle, Audioanschluß und drei Steckleisten für Erweiterungen.

PROGRAMMIER-SPRACHEN
BASIC und alle Sprachen, die unter MS-DOS laufen (beispielsweise FORTRAN, COBOL etc.).

TASTATUR
Insgesamt 90 Tasten, darunter ein Zehnerblock und 12 Funktionstasten.

HANDBÜCHER
Die Handbücher enthalten eine Dokumentation von MS-DOS, MS-BASIC und des integrierten Systems Desktop, das zum Lieferumfang der Maschine gehört.

STÄRKEN
Das Gerät ist voll IBM-kompatibel und verfügt über eine Reihe von Schnittstellen, die es für das Standard-IBM-Modell nicht gibt.

SCHWÄCHEN
Tandy hat die Maschine nicht wesentlich über den Standard des ursprünglichen IBM PC hinaus entwickelt. Durch den Einsatz des 8088-Prozessors statt des modernen 8086 arbeitet der Tandy 1000 langsamer als es nötig ist.

Wissensbaum

Mit dem folgenden Programm können Sie Ihren Computer das „Denken" lehren. Anhand von logischen Fragen und Antworten läßt sich das Listing fortlaufend ausbauen.

Tiere raten ist ein Spiel, in dem der Computer die Aufgabe hat, das Tier, das sich der Spieler ausgedacht hat, herauszufinden. Er versucht dies durch Fragen wie „Kann es fliegen?", „Hat es einen Pelz?" und so weiter. Sie dürfen nur mit „Ja" oder „Nein" antworten, worauf der Computer sich zu einem Punkt vorarbeitet, an dem er eine Antwort geben kann. Die zwei Aspekte, die das Programm besonders unterhaltsam machen, sind die Fähigkeit des Computers, in vernünftigem Deutsch zu kommunizieren (obwohl Ihre Antworten auf „Ja" und „Nein" beschränkt sind), sowie das „Wissen", auf das der Computer beim Erraten des Tieres zurückgreifen kann.

Bei diesem Spiel handelt es sich um ein sehr einfaches heuristisches Programm, das selbständig mit jedem Durchlauf dazulernt. Wenn das Programm das erste Mal gestartet wird, „kennt" es nur zwei Tiernamen und eine Frage. Abhängig davon, ob Ihre Antwort „Ja" oder „Nein" lautet, kann der Computer versuchen, das Tier zu erraten. Wenn der Computer falsch rät (was beim ersten Mal fast immer der Fall ist), fragt Sie das Programm nach dem Namen des von Ihnen gedachten Tieres, sowie nach einer Aussage, um es von dem vom Programm vorgeschlagenen Tier zu unterscheiden. Diese Informationen werden dann in der Datenbank des Programms abgelegt und bilden einen sogenannten „Baum des Wissens", der beim nächsten Spiel verwendet werden kann. Bei jedem Spiel vergrößert sich dieser Baum, bis das Programm schließlich nahezu alle Tiere richtig errät und nur noch selten ein neues beigebracht bekommt.

Datenorganisation

Ein interessanter Aspekt ist, daß das Programm nach wie vor wirklich absolut nichts über die Tiere „weiß". Es folgt blind dem roten Faden, der durch das kombinierte Wissen aller Spieler aufgebaut wurde. Die Informationen könnten sich auch auf verschiedene Biersorten, Motorräder, Medikamente oder Ihre Freunde und Ihre Familie beziehen. Die Grundfassung des Programms, die die Definition der ersten Frage sowie zwei Anworten ermöglicht, kann für die verschiedenartigsten Aufgaben verwendet werden. Es sind also nicht die Daten selbst, die den Programmablauf bestimmen, sondern die Art und Weise, in der die Daten organisiert werden.

Das Erstellen eines „lernfähigen" Programms ist nicht besonders schwierig. Meistens werden Daten-Strukturen dieser Art in BASIC-Arrays gespeichert. In unserem Beispielprogramm wird T$() für die Fragen und die Namen der Tiere, und J() sowie N() für die Verknüpfungen bestimmter Eintragungen in T$ verwendet. Diese Verknüpfungen weisen den Weg durch den Baum. Bei jeder „Ja"-Antwort ruft das Programm die entsprechende Anweisung aus T$ auf. Ähnlich verhält es sich mit den Eintragungen in N(). Am Ende des „Baumes" ist der Text ins T$ keine Frage mehr, sondern ergibt den Namen eines Tieres. Sowohl J() als auch N() werden auf 0 gesetzt, und das Programm rät ein Tier.

Tips für die Erweiterung

Die vorliegende Version des Programms wurde kurz und einfach gehalten, um Ihnen die grundlegenden Prinzipien zu zeigen. Wenn Sie es erweitern wollen, können Sie die Version für Ihren Computer mit Farbgrafiken, Tönen, zusätzlichen Fragen und dergleichen erweitern. Eine wesentliche Verbesserung wäre, wenn Sie eine Routine zum Speichern der Daten auf Diskette oder Cassette in das Programm einbauen würden.

BASIC-Dialekte

Dieses Programm ist in Microsoft-BASIC geschrieben und sollte somit in ungeänderter Form auf vielen Computern laufen können. Die einzige Änderung, die Sie vielleicht durchführen wollen, ist das Format der PRINT-Befehle, wenn Ihnen die Darstellung auf dem Bildschirm nicht gefällt.

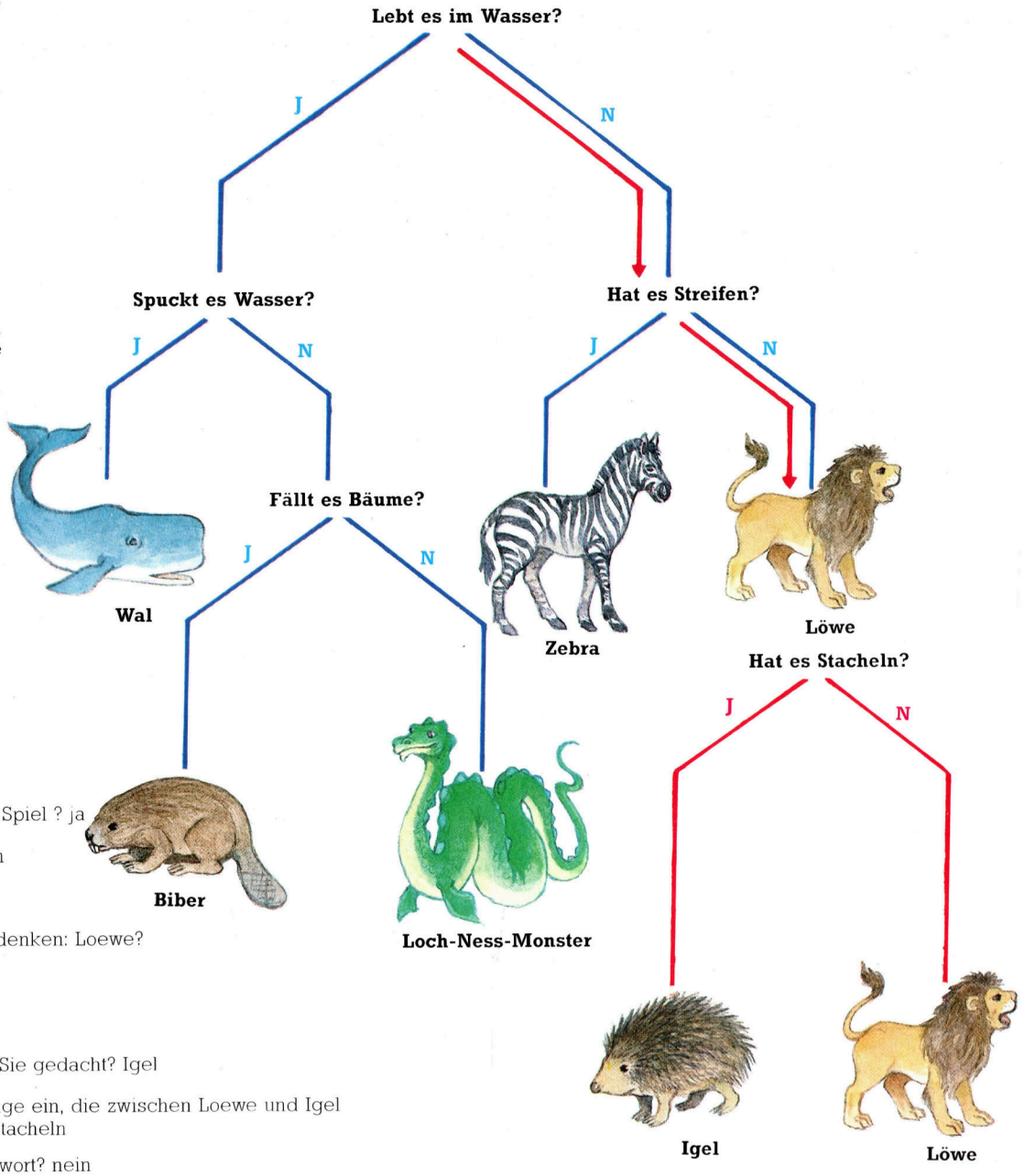
Beim Spectrum müssen alle Zuordnungs-Anweisungen mit dem Schlüsselwort „LET" beginnen. Schreiben Sie die folgenden Zeilen wie folgt neu:

```
45 LET L=40:REM Anzahl der Zeichen in
einer Frage
50 DIM J(N):DIM N(N):DIM T$(N,L)
150 LET I$=A$(1):LET P$="A"
200 IF A=30 THEN PRINT:PRINT
"TSCHUESS":STOP
230 IF J(P)=0 AND N(P)=0 THEN GOTO 290
```

LERN-BAUM

Dieses Diagramm zeigt den Tier-Baum nach diversen gespielten Spielen. Es wurden fünf verschiedene Tiere erlernt, mit vier Fragen zur Unterscheidung. Aus dem gezeigten Beispiellauf (in rot) können Sie ersehen, wie der Computer den „Baum" beim nächsten Programmlauf zur Beantwortung der Fragen des Spielers verwendet.

Wenn Sie Ihre eigene Version des Programms eingegeben haben, versuchen Sie einmal, an ein Kamel zu denken und dieses Tier dem Computer „beizubringen".



Wagen Sie ein weiteres Spiel ? ja

Lebt es im Wasser? nein

Hat es Streifen? nein

Ist das Tier, an das Sie denken: Loewe?

nein

Ich gebe auf!!

An welches Tier haben Sie gedacht? Igel

Bitte geben Sie eine Frage ein, die zwischen Loewe und Igel unterscheidet ? hat es Stacheln

Für Loewe ware die Antwort? nein

Ich kenne nun 6 verschiedene Tiere!

Wagen Sie ein weiteres Spiel?

```
10 REM TIERE RATEN
20 REM
30 REM ** INITIALISIERUNG
40 N=100: REM MAXIMALE ANZAHL AN TIEREN
50 DIM J(N), N(N), T$(N)
60 C=3:FOR I=1 TO 3:READ J(I), N(I),T$(I):NEXT I
70 PRINT:PRINT "T I E R E  R A T E N !":PRINT
80 GOTO 190
90 REM ** ANTWORTE MIT JA ODER NEIN
100 PRINT:PRINT Q$;" ";:INPUT A$
110 IF A$="j" OR A$="J" OR A$="JA" OR A$="ja" THEN A=1:RETURN
120 IF A$="n" OR A$="N" OR A$="NEIN" OR A$="nein" THEN A=0:RETURN
130 PRINT:PRINT "Antworten Sie mit JA oder NEIN": GOTO 100
180 REM ** BEGINNE NEUES SPIEL
190 Q$="Wagen Sie ein weiteres Spiel?": GOSUB 100
200 IF A=0 THEN PRINT:PRINT "Tschuess!": END
210 P=1
220 REM ** SPIELSTART
230 IF J(P)=0 AND N(P)=0 THEN 290
240 Q$=T$(P): GOSUB 100
250 IF A=1 THEN P=J(P)
260 IF A=0 THEN P=N(P)
270 GOTO 230
280 REM ** ERRATE EIN TIER
290 A$=T$(P):T$=A$
300 Q$="Ist das Tier, an das Sie denken," + A$: GOSUB 100
310 IF A=1 THEN PRINT:PRINT "Ich habe es erraten!!!!": GOTO 430
320 REM ** LERNE NEUES TIER
330 PRINT:PRINT "Ich gebe auf!!":PRINT "An welches Tier haben Sie gedacht":: INPUT N$
340 A$=N$
350 PRINT:PRINT "Bitte geben Sie eine Frage ein, die":PRINT "zwischen ";A$;" und ";T$;"
    unterscheidet.":INPUT D$
360 Q$="Fuer"+T$+"waere die Antwort": GOSUB 100
370 A$=T$(P):T$(P)=D$:T$(C+1)=A$:T$(C+2)=N$
380 IF A=1 THEN J(P)=C+1:N(P)=C+2
390 IF A=0 THEN J(P)=C+2:N(P)=C+1
400 J(C+1)=0:N(C+1)=0:J(C+2)=0:N(C+2)=0
410 C=C+2
420 REM ** BEENDE SPIEL & FRAGE NACH NEUSTART
430 A=INT(C/2)+1
440 PRINT:PRINT "Ich kenne nun ";A;" verschiedene Tiere!"
450 GOTO 190
460 REM ** INITIALISIERUNGS-DATEN
470 DATA 2,3,"Lebt es im Wasser?"
480 DATA 0,0,"Walfisch"
490 DATA 0,0,"Loewe"
```

Chiffren-Jagd

In einem Rennen gegen die Zeit müssen Sie in die Tiefen der Festung des teuflischen Elvin eindringen, um Paßworte zu sammeln und letztlich den richtigen Code zu finden. „Impossible Mission" gehört zu den neueren „Plattform"-Spielen von Epyx.

Aufgabe bei Impossible Mission ist es, mittels Lift und Korridoren die verschiedenen Räume des teuflischen Wissenschaftlers zu erreichen. Nach dem Betreten eines Raums muß nach Teilen des Paßwortes gesucht werden. Dabei muß der Held den Robotern ausweichen. Hat man das vollständige Paßwort, kann man in das Laboratorium des Professors gelangen. Gelegentlich findet man Paßwörter, die die Roboter vorübergehend abschalten.

Zu den derzeit populärsten Computerspielen gehören die „Plattformspiele". Bei diesem Programmtyp besteht die Aufgabe des Spielers (in seiner Eigenschaft als Held) darin, Gegenstände zu erlangen, die auf den vielen Ebenen der verschiedenen Bildschirme positioniert sind. Dabei tritt der Held meist vom linken oder rechten unteren Rand ins Bildschirmgeschehen und erreicht die Plattformen (oder Ebenen) über Lifte, Leitern oder Trampoline. Spielentscheidende Gegenstände werden dabei häufig in nur schwer erreichbaren Ecken versteckt, die nur durch intensives Nachdenken eingesammelt werden können.

Spiele dieser Art werden durch Außerirdische oder andere Gegner so erschwert. Man muß diesen Figuren ausweichen, weil die Berührung mit ihnen zu hohen Punktverlusten führt. Die Grafiken sind entweder statisch oder folgen einem festgelegten Muster. Überdies haben viele dieser Sprites „Schuß"-Möglichkeiten. Die Bewegungen der Gegner können je nach Spiel mehr oder weniger intelligent sein. Einige von ihnen bewegen sich auf festgelegter Strecke lediglich vor und zurück, wogegen andere darauf programmiert sind, den Helden zu orten und somit seine Bewegungsfreiheit einzuschränken.

Plattformspiele bieten dem Programmierer einige Vorteile. Hat er das Aussehen des Helden, der Wächter, Schätze, Plattformen, Leitern und Lifte erst einmal definiert, können alle anderen „Räume" durch bloßes Umordnen der Einzelelemente leicht programmiert werden.

Bei Impossible Mission werden die genannten Möglichkeiten voll genutzt. Spielziel ist es, die im Mobiliar von 32 verschiedenen Räumen versteckten Einzelstücke eines Codes zu finden. Mit dessen Hilfe kann der Spieler dann ins Machtzentrum des teuflischen Wissenschaftlers Elvin eindringen. Die Einrichtung wird durch Roboter und einen gelegentlich auftauchenden großen schwarzen Ball bewacht. In einigen Möbelstücken finden sich spezielle Paßwörter, die es dem Spieler gestatten, Fahrstühle in dem jeweiligen Raum zu steuern oder die Roboter vorübergehend zu stoppen. Mittels Lift und über verschiedene Korridore bewegt sich der Spieler von einem Raum zum anderen. Am unteren linken Rand des Bildschirms zeigt eine Karte, welche Räume bereits betreten wurden, sowie die gegenwärtige Position des Spielers.

Eingebaute Sprachsynthese

Paßwörter kann man auch in Coderäumen erlangen. Dabei wird eine Notenfolge mit unterschiedlicher Tonhöhe gespielt, und der Spieler muß die Töne in aufsteigender Folge ordnen. Je mehr Paßwörter man bekommen will, desto mehr Noten werden gespielt. Anders als bei den üblichen Spielen dieses Typs wird dem Spieler keine bestimmte Anzahl von „Leben" gegeben, sondern er bekommt jedesmal, wenn der Held getötet wird, eine festgelegte Zeitstrafe.

Der dadurch von den Programmierern gesparte Speicherraum wird optimal genutzt. Die wohl bemerkenswerteste Eigenschaft ist die Sprachsynthese, die sehr viel Speicherplatz benötigt. Ein paar gesprochene Sätze von etwa zehn Wörtern belegen bis zu vier KByte RAM. Die Sprachsynthese gehört zum Besten, was je in einem Spiel zu hören war. Die Grafiken sind ebenfalls sehr gut. Wenngleich die Räume selbst diesen Grafikstandard nicht halten, sind die Details des Helden, der Einrichtung und der Roboter exzellent. Die einzelnen Elemente ergänzen sich zu einem ungewöhnlichen Spiel, das lange Freude macht.

Impossible Mission: Für Commodore 64
Hersteller: CBS-Software
Autor: Dennis Caswell
Joystick: Erforderlich
Format: Cassette/Diskette

Texte, Texte

Die gebräuchlichste ernsthafte Anwendung für Microcomputer ist die Textverarbeitung. Fast jeder schreibt irgendwann einmal Briefe, Berichte oder Aufsätze. Ein Textverarbeitungssystem vereinfacht diese Aufgabe. Wir geben einen Überblick über das Programmangebot.

Im Grunde sind Wortprozessoren nichts weiter als computerisierte Versionen der Schreibmaschine. Der Text wird über die Computertastatur eingegeben und erscheint auf dem Bildschirm. Ohne das ganze Schriftstück neu eingeben zu müssen, kann man stellenweise Veränderungen vornehmen. Steht der Text korrekt, wird er einfach ausgedruckt.

Neben der allgemeinen Angst vor dem Computer hält eigentlich nur das Nichtvertrautsein mit der Tastatur viele Menschen vom Gebrauch eines Wortprozessors ab. Dabei ist es einfacher, auf einer Computertastatur als auf einer Schreibmaschine zu schreiben. Die unausweichlichen Fehler beim üblichen Zwei-Fingersystem bei den ersten Schreibversuchen bleiben beim Schreiben mit dem Wortprozessor erspart, da bei letzterem binnen weniger Sekunden Korrekturen und Ausdruck möglich sind. Eine solche Perfektion von Anfang an stärkt natürlich das Selbstvertrauen.

Wenngleich einige Rechner weniger gut geeignet sind, so kann doch fast jeder Heimcomputer für Textverarbeitung verwendet werden. Ursache für die geringere Eignung ist einmal, daß gute Programme für bestimmte Rechnertypen nicht verfügbar sind, und in anderen Fällen scheitert es daran, daß Rechner und Peripherie für diese Aufgaben ungeeignet sind.

Die Kosten selbst für ein einfaches Textverarbeitungsystem können hoch werden, da die erforderlichen Extras oft doppelt so teuer sind wie der Computer selbst. Teuerstes Stück dabei ist der Drucker. Ohne einen guten Drucker ist die Anschaffung eines Textverarbeitungssystems sinnlos, denn auch in absehbarer Zukunft werden Texte immer noch auf Papier gedruckt werden.

Selbst einfachste Drucker sind teuer, obwohl ihre Druckqualität ziemlich zu wünschen übrig läßt. Textverarbeitung erfordert in vielen Fällen einen hochwertigen Ausdruck. Es scheint wenig sinnvoll, viel Zeit für die Erstellung eines Schriftstücks aufzuwenden, das anschließend recht unschön auf einem Matrix-Drucker ausgegeben wird. Typenrad-Drucker liefern bessere Druckqualität, sind aber langsam und teuer, wenngleich die Preise zur Zeit stetig sinken. Einige elektrische Schreibmaschinen sind mit Schnittstellen ausgestattet, so daß sie als Computer-Drucker verwendet werden können. Das spart denjenigen Geld, die vom Schreibmaschine-Schreiben auf Textverarbeitung umsteigen wollen.

Manche Benutzer haben Probleme bei der Verbindung ihres Druckers mit dem Computer. Dieses Problem entfällt bei den Rechnern, die mit Standard-Schnittstellen ausgestattet sind. Der Benutzer benötigt lediglich ein Kabel, um beide zu verbinden. Rechner wie Commodore

Acorn B

Die Kombination des Acorn B mit dem Torch-Disketten-Paket erlaubt die Verwendung hervorragender Software, wozu auch der WordStar gehört. Die Kosten für das System sind höher als die für viele Geschäftscomputer. Man kann allerdings auch einen preisgünstigen Drucker verwenden und, statt WordStar zu kaufen, das „Perfect Writer" Programm benutzen, das im Lieferumfang von Torch enthalten ist.

oder Atari sind mit eigenen Schnittstellen ausgestattet, die – ohne Erweiterungen – nur den Betrieb der hauseigenen Drucker erlauben. Einige Micros wie etwa der Sinclair Spectrum werden ohne Drucker-Interface geliefert. Es muß zusätzlich gekauft werden.

Textformatierung

Steht ein geeigneter Drucker zur Verfügung, geht es um den Kauf der richtigen Software. Für populäre Systeme stehen viele Programme dieser Art zur Verfügung. Die Programmqualität schwankt jedoch betrachtlich. Es gibt einfache Textverarbeitungssoftware, mit der nur Einfügungen und Loschungen moglich sind. Bei anderen können ganze Textblocke innerhalb eines Artikels verschoben werden. Sie bieten eine Druckvorausschau (zeigen also den Text auf dem Bildschirm, wie er auf dem Papier ausgedruckt wurde) und verfugen uber Formatierungsmoglichkeiten (etwa die Breiteneinstellung des Textes auf einheitliche Zeilenbreite = Blocksatz).

Manche Textverarbeitungsprogramme erlauben die Suche nach bestimmten Wörtern oder Sätzen. Damit können Rechtschreibfehler einfach beseitigt werden, wenn der betreffende Begriff mehrfach im Text wiederholt wurde. Fur bestimmte Wortprozessoren gibt es Rechtschreibungs-Prüfprogramme. Ferner stehen erganzende Programme wie Datenbanken und Dateiverwaltungen zur Verfügung, die mit der Textverarbeitung kombiniert werden konnen. Höher entwickelte Programme dieser Art nutzen die Besonderheiten bestimmter Rechner. Auf Matrix-Druckern ist die Darstellung verschiedener Schriftbilder möglich (kursive, fette oder kleine Buchstaben). Bestimmte Programme erlauben das Mischen dieser Schriften. Schließlich stehen Wortprozessoren zur Verfugung, die dahingehend erweitert sind, daß die Grafikmöglichkeiten bestimmter Drukker genutzt werden können. Damit konnen viele weitere Schriftarten in den Text einfließen, wozu auch sogenannte Kapitalchen, Ornamente usw. gehören. Auf diese Weise läßt sich ein Text sehr interessant gestalten.

Typenraddrucker können Proportionalschrift drucken: Breite Buchstaben wie beispielsweise das „w" erhalten mehr Raum als englaufende wie das „l". Damit unterscheidet sich das Schriftbild von dem einer herkömmlichen Schreibmaschine. Der Text wird dadurch lesbarer. Besonders wenn dabei auch noch – wie es einige Programme bieten – das Mischen von Schriften und das rechts- und linksbundige Ausrichten des Textes möglich ist.

Es gibt Textverarbeitungssysteme in Disketten-, Cassetten- und ROM-Form. Wichtiger aber ist, wie der erstellte Text gespeichert wird – gewöhnlich auf Cassette oder Diskette. Cassetten sind zwar billig, aber umstandlich und langsam in der Handhabung. Und die Textmenge ist dabei stark eingeschrankt. Disketten zeichnen sich durch Schnelligkeit bei der Verarbeitung und im Zugriff aus.

Zeichendarstellung

Manche Programme sind in der Handhabung sehr umständlich und verfügen uber eine merkwurdige Tastenbelegung. Besonders kompliziert ist die Texteingabe auf dem Sinclair Spectrum. Inzwischen stehen allerdings aufsetzbare Tastaturen für das System zur Verfügung, die die Arbeit wesentlich erleichtern. Tastaturen mit Funktionstasten sind besonders geeignet, da hier die zahlreichen – oftmals recht komplizierten – Befehlsfolgen auf wenigen Tasten bereitgestellt werden. Auch die Bildschirmdarstellung kann Probleme verursa-

Sinclair Spectrum

Es ist zwar das einfachste Textverarbeitungssystem, aber dennoch teuer. Die Begrenzungen sind systembedingt, so unter anderem das unbefriedigende Keyboard, das Fehlen eines Monitor-Interface und das Provisorium Microdrive statt Diskettenstationen. Man kann jedoch eine besser Tastatur aufsetzen. Tasword Two ist eines der wenigen Spectrum-Textverarbeitungsprogramme, das mit Microdrives arbeitet.

chen. Der VC 20 stellt beispielsweise nur 22 Zeichen in einer Zeile auf dem Bildschirm dar, wogegen Bürorechner üblicherweise 80 Zeichen pro Zeile zeigen. Benutzt man diese Programmart oft, empfiehlt sich ein Rechner, der mindestens 25 x 80 Zeichen darstellen kann. Zur Schonung der Augen ist ein guter Monitor zu empfehlen.

Die Buchstabendarstellung auf dem Schirm ist ebenfalls unterschiedlich. Bei einigen Rechnern sind die Lettern aus einer größeren Anzahl von Einzelpunkten zusammengesetzt, womit die Lesbarkeit erhöht wird. Die meisten Heimcomputer bauen die Lettern aus einem Acht-Punkte-Raster, wenige Ausnahmen stellen sie in einem Sechs-Punkte-Raster dar. Die meisten größeren Computersysteme liefern eine 16 x 16-Punkte-Zeichendarstellung in bester Qualität.

Für alltägliche Anwendungen wie das Briefeschreiben unf für kurze Schriftstücke reichen auch einfache Heimcomputersysteme aus. Wer sich aber als Buchautor betätigt oder größere Textmengen zu verarbeiten hat, braucht mehr. Empfehlenswert ist dann ein mit Monitor, zwei Diskettenstationen, einem guten Drucker, Schreibmaschinentastatur und perfektem Textverarbeitungssystem ausgestattetes System. Die Erweiterung eines Heimcomputers auf dieses Niveau ist teuer, meist sogar teurer als der Kauf eines von vornherein so ausgestatteten Business-Systems.

Diese bieten ohnehin weitere Vorteile, da sie für professionelle Anwendungen entwikkelt wurden. Sie zeichnen sich durch gute Tastaturen, Bildschirme, Diskettenstationen und Drucker-Schnittstellen aus. Der wichtigste Vorteil ist die zur Verfügung stehende Software. Es gibt eine Reihe ausgezeichneter Textverarbeitungsprogramme, da die meisten Computer auf einem der Standardbetriebssysteme basieren. Daraus ergibt sich, daß jedes Textverarbeitungsprogramm für verschiedene Rechner zur Verfügung steht.

Das wohl bekannteste Programm dieser Art ist WordStar, das es für CP/M, CP/M-86 und MS-DOS-Betriebssysteme gibt. Diese Software verfügt über viele Besonderheiten, ist aber teuer und kostet ein Vielfaches eines durchschnittlichen Heimcomputerprogramms. Wirtschaftlich aber wäre es falsch, einen Heimcomputer für professionelle Textverarbeitung zu verwenden. Die dabei vergeudete Zeit steht in keinem Verhältnis zu den scheinbar gesparten Kosten.

Im Kaufpreis enthalten

Die Kosten für ein Profisystem sind zwar hoch, doch die Preise sinken stetig. Einige Heimcomputer können so erweitert werden, daß Standard-Betriebsysteme wie CP/M auf ihnen lauffähig sind. Wer also bereits viel in sein System investiert hat, kann durchaus ohne große Extrakosten „ernsthafte" Software nutzen. Ein weiterer Trend trägt zur Kostenreduzierung für Textverarbeitung bei: Verschiedene Anbieter liefern ihre Systeme mit Textverarbeitungsprogrammen wie WordStar ohne Zusatzkosten aus.

Die neueste Entwicklung in diesem Bereich ist die Textverarbeitung für unterwegs. Es gibt mehrere batteriebetriebene Computer mit integrierten Textverarbeitungssystemen, die es Geschäftsleuten auf Reisen ermöglichen, Notizen und Briefe überall und jederzeit zu schreiben. Es gibt jedoch kaum andere Anwendungsbereiche für diese Rechner. Doch diese Hand-helds erfreuen sich immer größerer Beliebtheit. Vielleicht ist nicht nur das Ende der Schreibmaschine absehbar, sondern auch das von Kugelschreiber und Papier.

Commodore 64

Der C64 bietet das beste Preis/Leistungsverhältnis für einen Wortprozessor mit einer Diskettenstation. Steht nur eine Floppy zur Verfügung, wird's schwierig. Der C 64 kann ohne Erweiterungskarte nur 40 Zeichen pro Zeile darstellen. Es gibt einen preiswerten Commodore-Drucker, der aber sehr schwache Druckqualität liefert.

Fenstertechnik

Wir stellen eine Maschinencoderoutine vor, die auf dem Spectrum die Programmierung von Fenstern möglich macht. Ein Beispielprogramm in BASIC zeigt, wie der Text innerhalb der Fenster bewegt wird.

Das Wort „HELLO" kann in Pixelschritten horizontal und vertikal über das Fenster bewegt werden.

Mit unserem Maschinencodeprogramm können Sie auf dem Spectrum rechteckige Bildschirmfenster definieren, deren Inhalt sich auf und ab und nach rechts und links bewegen läßt. Die Größe der Fenster ist beliebig, sie können an jeder Position des Bildschirms stehen und sind nicht an die Zeichengröße von acht mal acht Pixeln gebunden.

Die Maschinencodetabellen mit den Fensterparametern und den Zwischenspeichern fangen bei Adresse $B004 (dezimal 45060) an. Die Tabellenadressen des ersten Fensters befinden sich in $B000 und $B001 (dezimal 45056 und 45057). Da die Tabelle für jedes Fenster 11 Bytes benötigt, fängt die nächste Tabelle bei $B00F (dezimal 45071) an und die des dritten Fensters bei $B01A (dezimal 45082).

Das Beispielprogramm in BASIC steuert nur ein Fenster. Seine Daten werden von Zeile 180 bis 230 mit POKE in den Speicher geladen und in Zeile 240 initialisiert. Jedes weitere Fenster kann auf die gleiche Weise definiert werden. Wenn Sie die Adresse der gewünschten Fenstertabelle in Speicherstelle WT und WT+1 POKEn, können Sie von einem Fenster zum anderen springen. Die Richtung des Scrollens läßt sich mit einem POKE-Befehl auf die Speicherstelle WNDWTB + DIR steuern: POKE 0 scrollt links, 1 rechts, 2 auf und 3 ab.

Das Assemblerprogramm definiert zunächst eine Reihe von Konstanten. PIXADR ist eine im ROM gespeicherte Unterroutine des Spectrum. Sie berechnet die Adresse des Bildschirmbytes und innerhalb dieses Bytes die Bitnummer des Bildschirmpunktes, der von den PLOT-Koordinaten definiert wurde. PIXADR übernimmt die Y-Koordinate aus Register B und die X-Koordinate aus C und speichert die errechnete Bildschirmadresse nach dem Ablauf in Register HL und die Bitposition in A.

Die Routine INITW überprüft zunächst, ob die Koordinaten der unteren rechten Fensterecke wirklich unterhalb und rechts von der oberen linken Fensterecke liegen. Sie stellt außerdem sicher, daß der linke und rechte Rand sich nicht im gleichen Byte des Bildschirmspeichers befinden. Damit wird sichergestellt, daß das Fenster mindestens ein Zeichen breit ist.

Der letzte Abschnitt von INITW berechnet LFTMSK und RTMASK. Diese beiden Werte werden benötigt, wenn beim Scrollen des Fensterinhalts die Bildschirmbytes teilweise innerhalb und teilweise außerhalb des Fensters liegen. Die Maskenbits, die den außerhalb des Fensters liegenden Bildschirmbits entsprechen, stehen auf Eins, die zum Fensterinneren gehörenden Bits dagegen auf Null.

Da der Ablauf für eine Links- und Rechtsbewegung fast identisch ist, gibt es dafür nur eine Routine. Dabei wird über Bit Null des Richtungsbytes festgestellt, welcher Teil des Codes ausgeführt werden soll.

Da das rechte und auch das linke Scrollen bei der obersten Pixelzeile des Fensters anfängt, kopiert HORIZ zuerst die oberste Zeile in den Zwischenspeicher. Das Linksscrollen fängt dann am rechten Rand jeder Pixelzeile des Fensters an und setzt sich nach links fort. Zur Vorbereitung dieses Ablaufs werden RMASK und LMASK auf MASK1 und MASK2 kopiert, und weiterhin wird die Adresse des Bildschirmbytes am linken Rand der bearbeiteten Pixelzeile berechnet und im Registerpaar DE gespeichert. Die Adresse des Bildschirmbytes am rechten Ende der Pixelzeile wird ebenfalls berechnet und in HL abgelegt. Danach führt HLNSCR das Scrollen dieser Pixelzeile aus. Die Routine testet dabei zunächst, ob die unterste Zeile des Fensters erreicht ist. Ist dies nicht der Fall, nimmt sie die nächste Pixelzeile und springt zum Weiterscrollen auf HORIZ3.

Scroll-Vorgänge

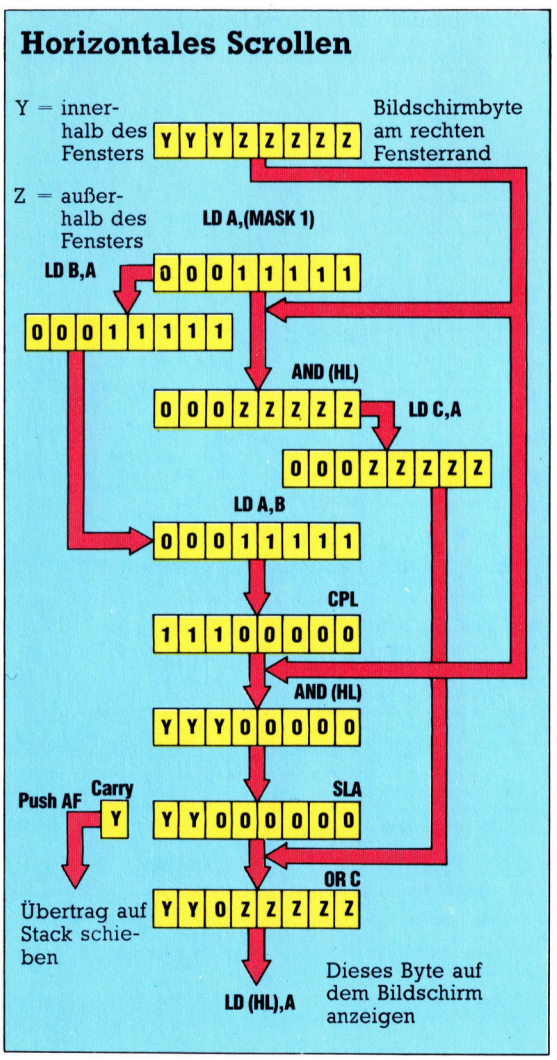
HLNSCR beginnt bei dem Randbyte, dessen Bits sowohl in als auch außerhalb des Fensters liegen können, nimmt dann die innerhalb des Fensters liegenden Bytes und hört am anderen Fensterrand auf. Unser Schaubild zeigt, wie der Code beim Linksscrollen das rechtsliegende Byte behandelt. Der Teil von HLNSCR, der bei NEXT anfängt, bewegt die Bytes innerhalb des Fensters. Das Bit, das beim Bewegen des vorigen Bytes in den Übertrag gewandert war, wird dabei mit PUSH AF auf den Stack geschoben und danach mit POPAF wieder in das Übertragsflag zurückgesetzt. Das Linksscrollen wird nun mit dem Befehl RL(HL) ausgeführt, wobei das Bildschirmbyte mit SHIFT nach links bewegt wird, das Bit des Übertragsflags linksaußen in das Byte eintritt und das rechtsaußen liegende Bit in das Übertragsflag wandert. PUSH AF speichert das Übertragsflag, so daß dieses Bit für das nächste Byte er-

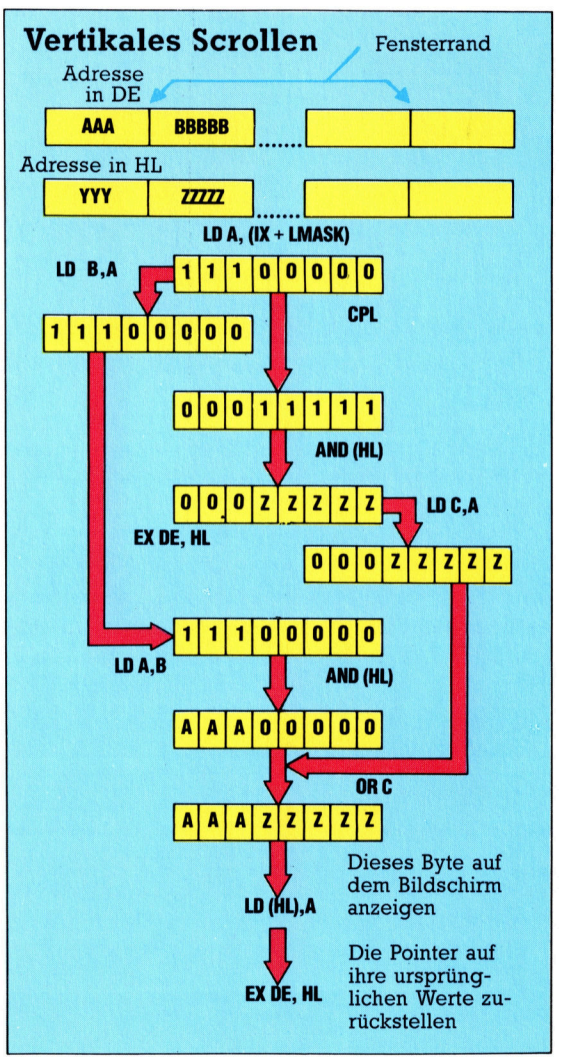
halten bleibt. Als Test für das Zeilenende braucht die Routine nur die Register L und E zu vergleichen, da das höherwertige Byte einer Bildschirmadresse bei allen Bildschirmbytes der gleichen Zeile übereinstimmt. Das Scrollen des letzten Randbytes wird wie beim ersten Randbyte durchgeführt.

Auch die Vorgänge bei der Auf- und Abwartsbewegung lassen sich mit einer Routine erledigen. Beim Aufwärtsscrollen mit VERT wird zunächst die Y-Koordinate der obersten Zeile in den Zwischenspeicher geladen, dann die Bildschirmadresse für den linken und rechten Rand der Zeile berechnet und die Zeilenlänge bestimmt. Die Routine speichert die Adresse des rechten Randes in DE und die Adresse des entsprechenden Bytes der darunterliegenden Zeile in HL. Nachdem die Unterroutine VLNSCR das Scrollen ausgeführt hat, testet VERT, ob der untere Rand des Fensters erreicht ist. Ist dies nicht der Fall, wird die darunterliegende Zeile angesprochen und VERT5 für das Scrollen dieser Zeile aufgerufen. Wenn der untere Rand erreicht ist, füllt der bei CLREDG beginnende Code die unterste Pixelzeile mit Nullen, um diese Zeile auf dem Bildschirm zu löschen.

Ebenso wie HLNSCR behandelt VLNSCR die Ränder mit einer separaten Routine. Unser zweites Bild zeigt die Bearbeitung der Randbytes. Für das Scrollen des Mittelteils der Pixelzeile werden die Register HL und DE inkrementiert, so daß sie für das erste innere Byte der laufenden Zeile auf das entsprechende Byte der darüberliegenden Zeile zeigen. VLNSCR berechnet dann die Länge des Mittelteils (d. h. der Bytes, die vollständig innerhalb des Fensters liegen), lädt diese Information in BC und bewegt den gesamten Mittelteil dieser Zeile mit dem Befehl LDIR als Block um eine Zeile nach oben.

Die Scrollroutinen sind nicht besonders schnell. Die Ursache dafür liegt in der Kombination der Links/Rechts- und Auf/Ab-Routinen und an den zahlreichen Tests, mit denen das Programm erst herausfinden muß, welchen Teil des Codes es ausführen soll. Zusätzlich verlangsamen die geteilten linken und rechten Randbytes mit ihren Spezialroutinen den Ablauf. Das Scrollen ließe sich weitaus schneller ausführen, wenn jede Bewegungsrichtung eine eigene Routine hätte und die Ränder der Fenster nur auf der Grenze zwischen zwei Bildschirmbytes liegen dürften.





Beim horizontalen Scrollen müssen die Bildschirmbytes am Fensterrand in „Masken" gesetzt werden, damit die Pixel innerhalb des Fensters von denen außerhalb trennbar sind. Der Inhalt dieser Bytes darf außerdem nur bitweise bewegt werden. Beide Vorgänge setzen für diese Aufgabe das logische AND, OR und SHIFT ein und erhalten den Inhalt des PSR durch eine Zwischenspeicherung im Stack. Das vertikale Scrollen wird durch die Memory Map des Spectrum sehr vereinfacht. Auch hier müssen die Bildschirmbytes „separiert" werden, um zwischen den Pixeln innerhalb und außerhalb des Fensters unterscheiden zu können. Der Inhalt der Register DE und HL, die die Pointer der Bildschirmadressen enthalten, werden mit dem Befehl EX ausgetauscht.

Fenster auf dem Spectum

So sollten Sie vorgehen:
1) Geben Sie das BASIC-Beispiel ein
2) SAVE „SCROLL" LINE 5
3) Tippen Sie den Maschinencode ein, und rufen Sie das Ladeprogramm für Maschinencode mit RUN auf
4) SAVE „SCROLLMC" CODE 45312,410 speichert den Code auf der Cassette direkt hinter dem BASIC-Programm
5) Spulen Sie die Cassette an den Anfang zurück, und geben sie LOAD „SCROLL" ein

Assemblerlisting

```
 10 PIXADR EQU  #22AA
 20 WT     EQU  #B000
 30 MASK1  EQU  #B002
 40 MASK2  EQU  #B003
 50 WNDWTB EQU  #B004
 60 LEFTX  EQU  0
 70 TOPY   EQU  1
 80 RIGHTX EQU  2
 90 BOTY   EQU  3
100 LBIT   EQU  4
110 RBIT   EQU  5
120 CURNTY EQU  6
130 DIR    EQU  7
140 LMASK  EQU  8
150 RMASK  EQU  9
160 LENGTH EQU  10
170        ORG  #B100
180 INIT   LD   HL,(WT)
190        PUSH HL
200        POP  IX
210        CALL INITW
220        RET
230 SCROLL LD   HL,(WT)
240        PUSH HL
250        POP  IX
260        BIT  1,(IX+DIR)
270        PUSH AF
280        CALL Z,HORIZ
290        POP  AF
300        CALL NZ,VERT
310        RET
320 HORIZ  LD   A,(IX+TOPY)
330        LD   (IX+CURNTY),A
340        BIT  0,(IX+DIR)
350        JR   Z,HORIZ1
360        LD   B,(IX+LMASK)
370        LD   A,(IX+RMASK)
380        JR   HORIZ2
390 HORIZ1 LD   B,(IX+RMASK)
400        LD   A,(IX+LMASK)
410 HORIZ2 LD   (MASK2),A
420        LD   A,B
430        LD   (MASK1),A
440 HORIZ3 LD   C,(IX+LEFTX)
450        LD   B,(IX+CURNTY)
460        CALL PIXADR
470        EX   DE,HL
480        LD   C,(IX+RIGHTX)
490        LD   B,(IX+CURNTY)
500        CALL PIXADR
510        BIT  0,(IX+DIR)
520        JR   Z,HORIZ4
530        EX   DE,HL
540 HORIZ4 CALL HLNSCR
550        LD   A,(IX+BOTY)
560        CP   (IX+CURNTY)
570        RET  Z
580        DEC  (IX+CURNTY)
590        JR   HORIZ3
600 HLNSCR LD   A,(MASK1)
610        LD   B,A
620        AND  (HL)
630        LD   C,A
640        LD   A,B
650        CPL
660        AND  (HL)
670        BIT  0,(IX+DIR)
680        JR   Z,HLN1
690        SRA  A
700        JR   HLN2
710 HLN1   SLA  A
720 HLN2   PUSH AF
730        OR   C
740        LD   (HL),A
750 NEXT   BIT  0,(IX+DIR)
760        JR   Z,HLN3
770        INC  HL
780        JR   HLN4
790 HLN3   DEC  HL
800 HLN4   LD   A,L
810        CP   E
820        JR   Z,LAST
830        POP  AF
840        BIT  0,(IX+DIR)
850        JR   Z,HLN5
860        RR   (HL)
870        JR   HLN6
880 HLN5   RL   (HL)
890 HLN6   PUSH AF
900        JR   NEXT
910 LAST   LD   C,(HL)
920        LD   A,(MASK2)
930        AND  C
940        LD   B,A
950        POP  AF
960        BIT  0,(IX+DIR)
970        JR   Z,HLN7
980        RR   C
 990        JR   HLN8
1000 HLN7  RL   C
1010 HLN8  LD   A,(MASK2)
1020       CPL
1030       AND  C
1040       OR   B
1050       LD   (HL),A
1060       RET
1070 VERT  LD   C,(IX+LEFTX)
1080       BIT  0,(IX+DIR)
1090       JR   Z,VERT1
1100       LD   B,(IX+BOTY)
1110       JR   VERT2
1120 VERT1 LD   B,(IX+TOPY)
1130 VERT2 LD   (IX+CURNTY),B
1140       CALL PIXADR
1150       PUSH HL
1160       PUSH HL
1170       LD   C,(IX+RIGHTX)
1180       LD   B,(IX+CURNTY)
1190       CALL PIXADR
1200       POP  DE
1210       AND  A
1220       SBC  HL,DE
1230       LD   A,L
1240       DEC  A
1250       LD   (IX+LENGTH),A
1260 VERT3 BIT  0,(IX+DIR)
1270       JR   Z,VERT4
1280       INC  (IX+CURNTY)
1290       JR   VERT5
1300 VERT4 DEC  (IX+CURNTY)
1310 VERT5 LD   C,(IX+LEFTX)
1320       LD   B,(IX+CURNTY)
1330       CALL PIXADR
1340       POP  DE
1350       PUSH HL
1360       CALL VLNSCR
1370       LD   A,(IX+CURNTY)
1380       BIT  0,(IX+DIR)
1390       JR   Z,VERT6
1400       CP   (IX+TOPY)
1410       JR   VERT7
1420 VERT6 CP   (IX+BOTY)
1430 VERT7 JR   NZ,VERT3
1440 CLREDG POP  HL
1450       LD   A,(IX+LMASK)
1460       AND  (HL)
1470       LD   (HL),A
1480       LD   B,(IX+LENGTH)
1490       LD   A,0
1500 CLR1  INC  HL
1510       LD   (HL),A
1520       DJNZ CLR1
1530       INC  HL
1540       LD   A,(IX+RMASK)
1550       AND  (HL)
1560       LD   (HL),A
1570       RET
1580 VLNSCR LD  A,(IX+LMASK)
1590       CALL ENDBYT
1600       INC  HL
1610       INC  DE
1620       LD   B,0
1630       LD   C,(IX+LENGTH)
1640       LDIR
1650       LD   A,(IX+RMASK)
1660 ENDBYT LD  B,A
1670       CPL
1680       AND  (HL)
1690       LD   C,A
1700       EX   DE,HL
1710       LD   A,B
1720       AND  (HL)
1730       OR   C
1740       LD   (HL),A
1750       EX   DE,HL
1760       RET
1770 INITW LD   A,(IX+RIGHTX)
1780       CP   (IX+LEFTX)
1790       JR   Z,ERROR
1800       JR   C,ERROR
1810       LD   A,(IX+TOPY)
1820       CP   (IX+BOTY)
1830       JR   Z,ERROR
1840       JR   C,ERROR
1850       LD   C,(IX+LEFTX)
1860       LD   B,(IX+TOPY)
1870       CALL PIXADR
1880       PUSH HL
1890       LD   (IX+LBIT),A
1900       LD   C,(IX+RIGHTX)
1910       LD   B,(IX+TOPY)
1920       CALL PIXADR
1930       LD   (IX+RBIT),A
1940       POP  BC
1950       LD   A,C
1960       CP   L
1970       JR   Z,ERROR
1980 LFTMSK LD  B,(IX+LBIT)
1990       LD   A,0
2000 L1    SCF
2010       RRA
2020       DJNZ L1
2030       LD   (IX+LMASK),A
2040 RTMASK LD  B,(IX+RBIT)
2050       LD   A,255
2060 L2    AND  A
2070       RRA
2080       DJNZ L2
2090       LD   (IX+RMASK),A
2100       RET
2110 ERROR RST  8
2120       DEFB 25
```

BASIC-Beispiel

```
  5 CLEAR 32767
 10 LOAD ""CODE
 20 LET WT=45056
 30 LET WINDOWTABLE=45060: REM B004 HEX
 40 LET LEFTX=0
 50 LET TOPY=1
 60 LET RIGHTX=2
 70 LET BOTY=3
 75 LET DIR=7
 80 LET SCROLL=45322
 90 LET INIT=45312
100 BORDER 6
110 PAPER 4: INK 2
120 CLS
180 POKE WT,4: POKE WT+1,176
190 REM WT & WT+1 NOW CONTAIN ADDRESS 45060
IN LO,HI FORMAT
200 POKE WINDOWTABLE+LEFTX,5
210 POKE WINDOWTABLE+TOPY,80
220 POKE WINDOWTABLE+RIGHTX,250
230 POKE WINDOWTABLE+BOTY,35
240 RANDOMIZE USR INIT
250 FOR Y=0 TO 175
260 PLOT 0,Y
270 DRAW 255,0
280 NEXT Y
290 POKE WINDOWTABLE+DIR,2
300 FOR Y=0 TO 45
310 RANDOMIZE USR SCROLL
320 NEXT Y
400 PRINT AT 12,20;"HELLO";
410 POKE WINDOWTABLE+DIR,0
420 FOR I=1 TO 130
430 RANDOMIZE USR SCROLL
440 NEXT I
470 POKE WINDOWTABLE+DIR,3
480 FOR I=1 TO 35
490 RANDOMIZE USR SCROLL
500 NEXT I
510 POKE WINDOWTABLE+DIR,1
520 FOR I=1 TO 130
530 RANDOMIZE USR SCROLL
540 NEXT I
550 POKE WINDOWTABLE+DIR,2
560 FOR I=1 TO 35
570 RANDOMIZE USR SCROLL
580 NEXT I
590 GO TO 410
999 STOP
```

Ladeprogramm für Maschinencode

```
 100 LET a=45312
 110 FOR l=1000 TO 1500 STEP 10
 120 LET s=0
 125 FOR a=a TO a + 7
 130 READ b
 140 POKE a,b
 150 LET s=s+b
 160 NEXT a
 170 READ c
 180 IF s<>c THEN PRINT "ERROR IN LINE "; l : STOP
 190 NEXT l
1000 DATA 42,0,176,229,221,225,205,69,1167
1010 DATA 178,201,42,0,176,229,221,225,1272
1020 DATA 221,203,7,78,245,204,29,177,1164
1030 DATA 241,196,184,177,201,221,126,1,1347
1040 DATA 221,119,6,221,203,7,70,40,887
1050 DATA 8,221,70,8,221,126,9,24,687
1060 DATA 6,221,70,9,221,126,8,50,711
1070 DATA 3,176,120,50,2,176,221,78,826
1080 DATA 0,221,70,6,205,170,34,235,941
1090 DATA 221,78,2,221,70,6,205,170,973
1100 DATA 34,221,203,7,70,40,1,235,811
1110 DATA 205,103,177,221,126,3,221,190,1246
1120 DATA 6,200,221,53,6,24,215,58,783
1130 DATA 2,176,71,166,79,120,47,166,827
1140 DATA 221,203,7,70,40,4,203,47,795
1150 DATA 24,2,203,39,245,177,119,221,1030
1160 DATA 203,7,70,40,3,35,24,1,383
1170 DATA 43,125,187,40,16,241,221,203,1076
1180 DATA 7,70,40,4,203,30,24,2,380
1190 DATA 203,22,245,24,226,78,58,3,859
1200 DATA 176,161,71,241,221,203,7,70,1150
1210 DATA 40,4,203,25,24,2,203,17,518
1220 DATA 58,3,176,47,161,176,119,201,941
1230 DATA 221,78,0,221,203,7,70,40,840
1240 DATA 5,221,70,3,24,3,221,70,617
1250 DATA 1,221,112,6,205,170,34,229,978
1260 DATA 229,221,78,2,221,70,6,205,1032
1270 DATA 170,34,209,167,237,82,125,61,1085
1280 DATA 221,119,10,221,203,7,70,40,891
1290 DATA 5,221,52,6,24,3,221,53,585
1300 DATA 6,221,78,0,221,70,6,205,807
1310 DATA 170,34,209,229,205,40,178,221,1286
1320 DATA 126,6,221,203,7,70,40,5,678
1330 DATA 221,190,1,24,3,221,190,3,853
1340 DATA 32,209,225,221,126,8,166,119,1106
1350 DATA 221,70,10,62,0,35,119,16,533
1360 DATA 252,35,221,126,9,166,119,201,1129
1370 DATA 221,126,8,205,58,178,35,19,850
1380 DATA 6,0,221,78,10,237,176,221,949
1390 DATA 126,9,71,47,166,79,235,120,853
1400 DATA 166,177,119,235,201,221,126,2,1247
1410 DATA 221,190,0,40,67,56,65,221,860
1420 DATA 126,1,221,190,3,40,57,56,694
1430 DATA 55,221,78,0,221,70,1,205,851
1440 DATA 170,34,229,221,119,4,221,78,1076
1450 DATA 2,221,70,1,205,170,34,221,924
1460 DATA 119,5,193,121,189,40,25,221,913
1470 DATA 70,4,62,0,55,31,16,252,490
1480 DATA 221,119,8,221,70,5,62,255,961
1490 DATA 167,31,16,252,221,119,9,201,1016
1500 DATA 207,25,0,0,0,0,0,0,232
```

Roboter-Bausatz

Die Firma Fischer hat einen „Computing"-Baukasten herausgebracht, mit dem man zehn verschiedene Robotermodelle anhand von Vorlagen zusammensetzen kann. Technisch Interessierte können dabei erste Erfahrungen im Bereich der Robotik sammeln.

Die Heimcomputerbranche wartet einstweilen noch vergeblich auf das Zutreffen der Prophezeiung, Heimroboter würden den nächsten großen Wachstumsschub bringen – es ist schwer zu sagen, woran das liegt. Zwar ist in den letzten Jahren eine ganze Reihe solcher Geräte auf den Markt gekommen, aber es traten dabei zahlreiche Probleme auf.

Einerseits gibt es sehr einfache Roboter wie die Movits, die man auch ohne technische Vorkenntnisse in ein paar Stunden zusammensetzen kann. Der Nachteil dabei ist, daß diese Geräte nur sehr eingeschränkte Einsatzmöglichkeiten bieten, so daß der Benutzer bald die Lust daran verliert. Auf der anderen Seite werden teurere Roboter verkauft, deren Betrieb im allgemeinen zumindest Programmierkenntnisse voraussetzt.

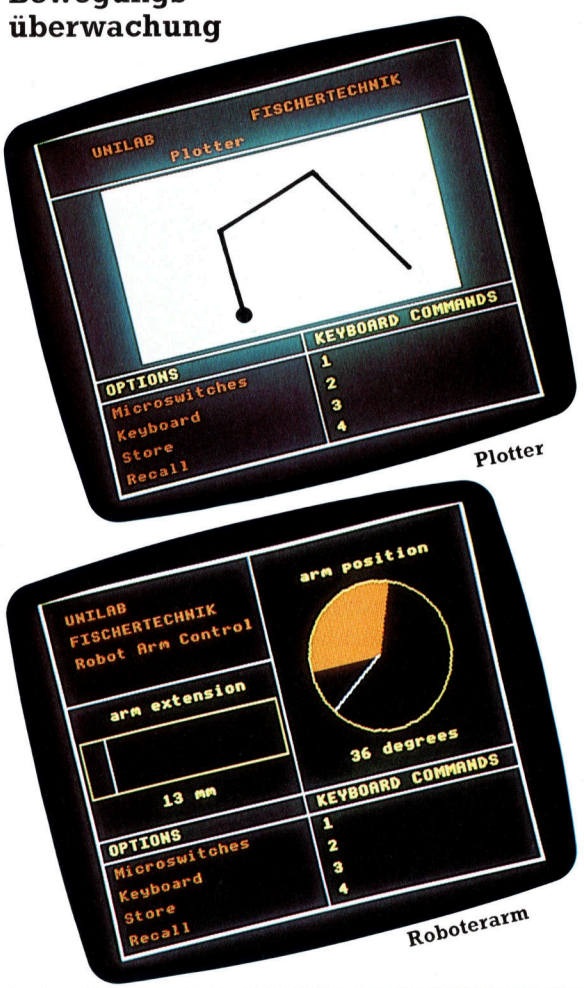
Eigentlich braucht man ein System, das einfach aufzubauen ist und mit dem der Anwender dennoch eine große Anzahl verschiedener Funktionen ausprobieren kann. Dieser Gedanke führte zur Entwicklung des Roboterbaukastens von Fischertechnik. Schon seit Jahren gibt es Elektronik-Experimentierkasten für Kinder, die damit zum Beispiel ein einfaches Radio basteln können. Bereits seit Generationen gibt es die Lego-Bausteine (vorwiegend für statische Modelle) und Mechanik-Baukasten wie die von Fischer, bei denen aus relativ wenigen Funktionselementen die verschiedensten Figuren und Maschinen gebaut werden können. Der Computing-Baukasten vereint nun Mechanik und Elektronik und dazu noch Rechnertechnik. Das Ergebnis ist ein System, das vielleicht den lange vorhergesagten Durchbruch der Roboter im Heimcomputerbereich bedeuten konnte.

Mit dem Baukasten lassen sich verschiedene computergesteuerte Geräte aufbauen. Das kann ein Roboterarm sein oder ein Apparat zum Sortieren verschiedener Dinge nach Länge, aber auch ein Plotter oder ein Grafiktablett. Die Modelle lassen sich nach dem Zusammenbau einfach zerlegen und zu einem anderen Gerät montieren. Für den Betrieb am Rechner ist ein Interface erforderlich, das entsprechend den Computerdaten die Leistung für die Antriebselemente schaltet und die Rückmeldungen des Roboters an den Rechner weitergibt.

Die Bauteile aus Kunststoff werden mit Verbindungszapfen zu den verschiedenen Modellen zusammengesteckt. Die Einzelblöcke können so aneinandergefügt werden, daß zum Beispiel ein längerer Arm oder eine Halterung für eine Antriebseinheit entstehen. Weiter enthält der Kasten eine 260 x 187 mm große Kunststofftafel, die entweder als Grundplatte für den Roboter oder, beim Aufbau des Plotters und des Grafiktabletts, als Papierunterlage dient. Auch die elektrische Versorgung und die Steuerung sind aus Einzelteilen zusammenzusetzen. Der Bausatz umfaßt zahlreiche Stecker und Buchsen, acht Schalter und zwei Motoren, dazu einen Meter 20adriges Flachbandkabel und ein Stück Leitung für kurze Schleifen.

Die Stabilität des Fischertechnik-Systems

Bewegungs-überwachung



Links ist gezeigt, wie sich beim Acorn B die Software dem Benutzer vorstellt, oben zunächst für den Betrieb des Plotters. Bei anderen Computern, wie etwa dem C64, entfällt diese hervorragende grafische Umsetzung leider völlig. Das darunter befindliche Torten-Diagramm der „arm position" gibt die Winkelstellung des Arms wieder, ausgehend von der Nullstellung. Bei einer Rechtsdrehung bewegt sich die Linie im Kreis simultan, und der entsprechende Winkel wird darunter ausgegeben. Am unteren Bildschirmrand sind die möglichen Steuerungsvarianten angegeben, nämlich Schalter- oder Tastatursteuerung (Microswitches bzw. Keyboard), Speicherung von Bewegungsabläufen beim „Teach-In" (Store) und deren späterer Aufruf (Recall).

Das gleiche Befehlsschema wird beim Plotter verwendet. Hier kommen allerdings noch Kommandos für das Heben und Senken des Zeichenstifts hinzu.

laßt vermuten, daß die Teile viele Jahre halten – mit Ausnahme der Potentiometer

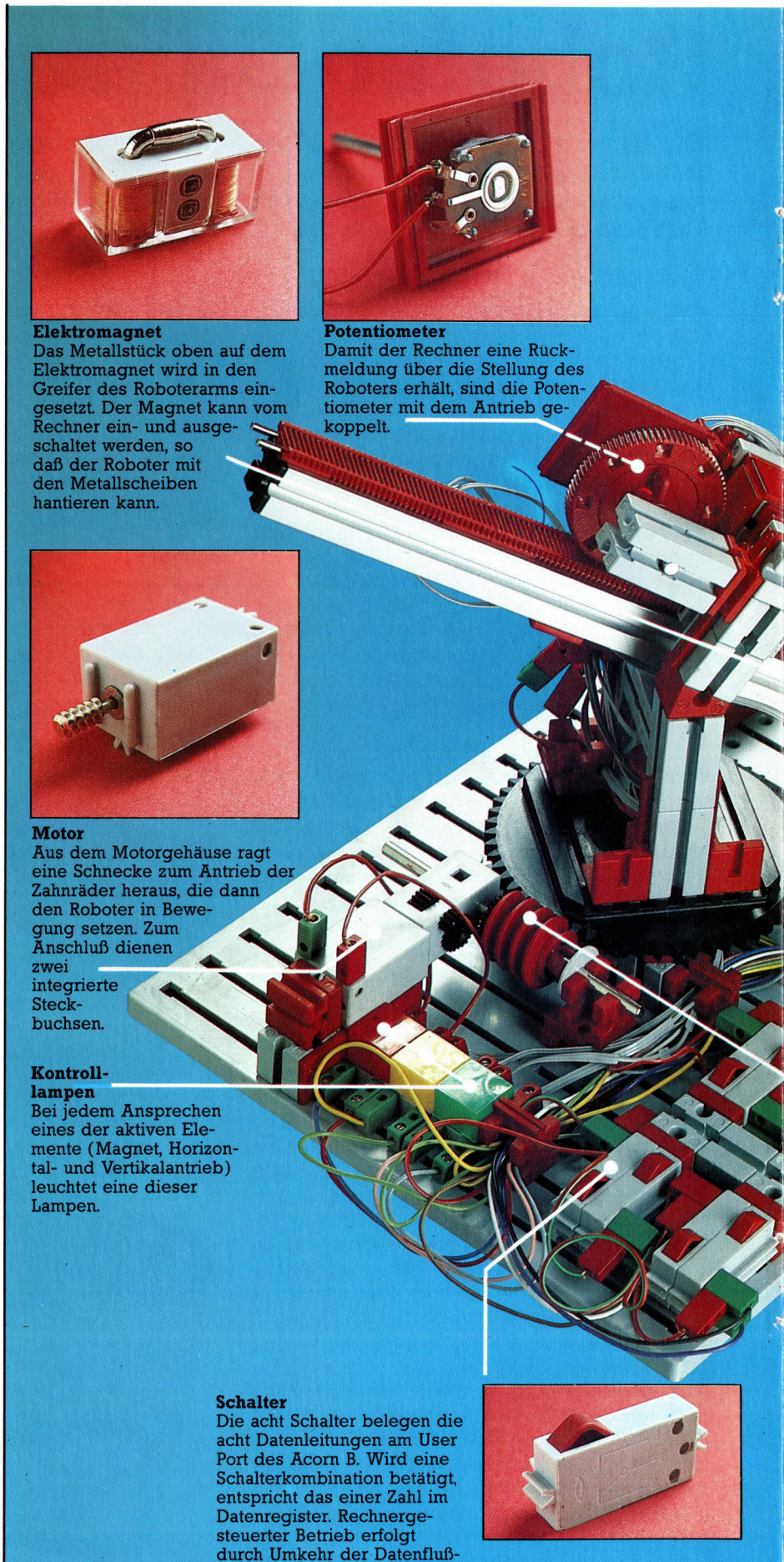
Der Aufbau der Modelle ist relativ einfach, und die meisten Elemente sind so griffig, daß auch ungeübte Finger damit umgehen können. Schuld an etwaigen Schwierigkeiten ist eigentlich nur die Anleitung – der schwachste Punkt an dem ganzen System. Sie hat zwar ein paar gute Seiten, etwa die leicht nachvollziehbaren Schaltpläne, stützt sich aber im übrigen auf nur wenige Fotos der verschiedenen Aufbaustadien Zusätzlich werden die Einzelteile für den jeweiligen Bauabschnitt abgebildet, und zuweilen findet man noch eine Art Explosionszeichnung. Trotzdem sind manche Details schwer auszumachen. Außerdem zeigt das Foto den Aufbau nur aus einer Perspektive, so daß der Hobby-Bastler oftmals nur erahnen kann, wie die nicht dargestellten Bauteile der restlichen Seiten zusammenzufügen sind.

Unzureichende Anleitung

Erschwerend kommt hinzu, daß es sich um Schwarzweißfotos handelt. Wenn Sie sich bis zu den elektrischen Verbindungen vorgearbeitet haben, werden Sie feststellen, daß die richtige Verdrahtung aus der Bauanleitung allein nicht zu ersehen ist – dabei sind nämlich die Kabelfarben entscheidend. Nur begrenzt helfen die vereinzelten Hinweise in der Programmieranleitung – beim Roboterarm beispielsweise wird daraus zwar die Schalterbelegung klar, nicht aber die Zuordnung der Potentiometer und Motoren.

Wenn der Roboter fertig ist, erfolgt der Anschluß des Interface. Das ist glücklicherweise viel einfacher. Die Schnittstellenkarte trägt Gruppen von Steckbuchsen für den Anschluß des Flachbandkabels. In der Abbildung oben rechts ist das Interface für den Acorn B gezeigt; für andere Rechner sieht die Karte ähnlich aus. Links auf der Leiterplatte liegen die Schalter-Eingänge. Deren Logik wird vom Datenregister und vom Datenflußregister des Rechners bestimmt und funktioniert ähnlich wie die der selbstgebauten Buffereinheit aus unserem Bastelkurs. Darunter werden die Potentiometer angeschlossen, an denen der Rechner die Position des Roboters abliest und entsprechend die Motoren und Lampen ein- und ausschaltet. Die Potentiometerspannungen gelangen über den Analogeingang in den Rechner.

Rechts auf der Interfacekarte sind die Buchsen für die Motoren und den Elektromagneten (mit dem der Roboter Eisenteile aufnehmen kann), außerdem ein Anschluß für die Betriebsspannung von sechs bis acht Volt. Das Interface belegt am Acorn B gleichzeitig den User Port, den Analogeingang und den Drukkerausgang, über den die Motoren und der Magnet gesteuert werden. Um das Ganze „idiotensicher" zu machen, sind auf der



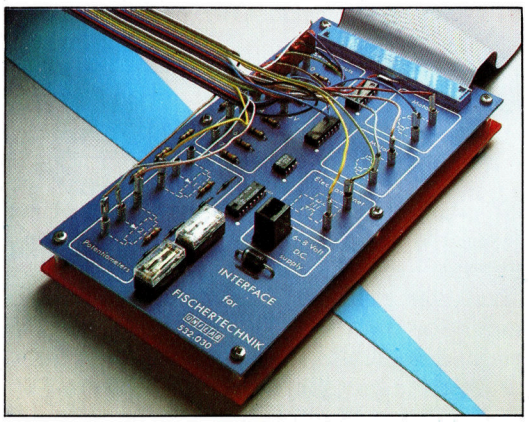
Elektromagnet
Das Metallstück oben auf dem Elektromagnet wird in den Greifer des Roboterarms eingesetzt. Der Magnet kann vom Rechner ein- und ausgeschaltet werden, so daß der Roboter mit den Metallscheiben hantieren kann.

Potentiometer
Damit der Rechner eine Rückmeldung über die Stellung des Roboters erhält, sind die Potentiometer mit dem Antrieb gekoppelt.

Motor
Aus dem Motorgehäuse ragt eine Schnecke zum Antrieb der Zahnräder heraus, die dann den Roboter in Bewegung setzen. Zum Anschluß dienen zwei integrierte Steckbuchsen.

Kontrolllampen
Bei jedem Ansprechen eines der aktiven Elemente (Magnet, Horizontal- und Vertikalantrieb) leuchtet eine dieser Lampen.

Schalter
Die acht Schalter belegen die acht Datenleitungen am User Port des Acorn B. Wird eine Schalterkombination betätigt, entspricht das einer Zahl im Datenregister. Rechnergesteuerter Betrieb erfolgt durch Umkehr der Datenflußrichtung.



Interface
Über das Unilab-Interface laufen sämtliche Steuerungsvorgänge. Die Roboter-Leitungen werden an Buchsen auf der Karte geführt, die ihrerseits mit dem User Port verbunden ist.

Metallscheiben
Mit Hilfe des Elektromagneten nimmt der Roboter die Metallscheiben auf.

Getriebe
Die beiden Motoren setzen den Roboter über Schnecken, Zahnräder und Zahnstangen in Bewegung, was trotz der mehrfachen Umlenkung erstaunlich gut funktioniert. Andere Übersetzungen geben die Rückmeldungen an die Potentiometer.

Schnittstellenkarte Schutzschaltungen vorgesehen, die die Rechnereingänge auch bei Verdrahtungsfehlern vor zu hohen Spannungspegeln bewahren.

Die Anleitung für das Interface bereitet dagegen kaum Probleme. Die Installation der Schnittstelle wird ausführlich erläutert, ebenso der Gebrauch der mitgelieferten Software. Außerdem steht darin eine kurze Zusammenstellung der System-Steueradressen, was eine Programmierung des Roboters in BASIC ermöglicht. Schnittstellen stehen außer für die Acorn-Rechner für die Commodore VC20/C64, für die Apple II-Versionen, für den Kosmos CP1, den Busch Microtronic 2090 und den NDR-Computer zur Verfügung, außerdem für die meisten Z80-Systeme. Schnittstellen für den Sinclair Spectrum und den Schneider CPC464 sind in Vorbereitung, ebenso zwei Spezial-Baukästen (Greifhand-Roboter und x,y-Koordinatentisch), die mit den gleichen Schnittstellen zu betreiben sind.

Programmierung der Modelle

Zu jeder der Schnittstellen wird eine Diskette geliefert, die Programme zur Unterstützung der Steuerung enthält. Auf der Diskette befindet sich zunächst ein Grundprogramm mit den einzelnen Steuerkommandos, auf das alle anderen Programme zugreifen und mit dessen Hilfe Sie beliebige eigene Programme aufbauen können. Es handelt sich im wesentlichen um eine Reihe von Zusatzbefehlen zur BASIC-Erweiterung. Das zweite universelle Programm auf der Diskette ist ein Diagnoseprogramm zur Unterstützung der Fehlersuche.

Ferner enthält die Diskette für jedes der Baukastenmodelle ein fertiges Betriebsprogramm, das in der mitgelieferten Programmieranleitung aufgelistet und erläutert ist (wie das Grund- und das Diagnoseprogramm). Beim Roboter ist auch Teach-In-Betrieb vorgesehen, das heißt, der Rechner kann einen schaltergesteuerten Ablauf für die automatische Ausführung speichern.

Trotz der erwähnten Schwächen der Anleitung für den mechanischen Aufbau ist der Computing-Baukasten mit einer der Schnittstellen eine lohnende Anschaffung. Die Modelle funktionieren allgemein erstaunlich gut und sind in wenigen Stunden zusammengesetzt. Dabei sind Sie nicht an die Vorlagen gebunden – wenn Sie erst mit dem System vertraut sind, können Sie beliebige Roboter basteln. Bei einem Preis von knapp 450 Mark (einschließlich Schnittstelle) ist ja auch etwas mehr als nur ein Kinderspielzeug zu erwarten. Eine gründliche Überarbeitung der Anleitung vorausgesetzt, ist der Fischertechnik-Baukasten ein idealer Einstieg für jeden, der Einblick in die Robotertechnik gewinnen möchte. Demnächst werden wir im „Bastelkurs" zeigen, wie man einen Roboterarm selbst bauen kann.

Fischertechnik-Computing

ANTRIEB
Ein oder zwei Gleichstrommotoren.

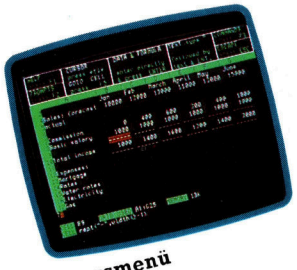
STEUERUNG
Über die Tastatur (bzw. vom Programm) oder über die acht Schalter am Roboter selbst.

STROMVERSORGUNG
nicht beigefügt; 6-Volt-Batterie.

HERSTELLER
Fischer-Werke, Weinhalde 14–18, 7244 Tumlingen/Waldachtal; Lieferung durch den Spielwaren- und Computer-Fachhandel

Modellfall

Wir untersuchen Abacus, ein Kalkulationspaket, das zum Lieferumfang des Sinclair QL gehört.



Anfangsmenü

Befehlsmenü

Bei der Vorstellung des Sinclair QL erregte besonders das mitgelieferte Programmpaket – Quill (Text), Archive (Datenbank), Easel (Grafik) und Abacus (Kalkulationsystem) – viel Aufsehen, da es Merkmale der integrierten Software enthielt. Die Daten der einzelnen Programme sind untereinander kompatibel, so daß beispielsweise Daten des Kalkulationssystems in Diagrammform dargestellt und in ein mit Quill erstelltes Schriftstück übernommen werden können. Auch der Bildschirmaufbau ist derart abgestimmt, daß einige Befehle in allen Programmen identische Funktionen ausführen. So rufen drei der fünf Funktionstasten des QL in allen Anwendungen die gleichen Vorgänge ab. F1 spricht Help an, F2 steuert den Promptbereich und F3 schaltet den Befehlsmodus ein. Die Programme müssen jedoch einzeln aufgerufen werden.

Es gibt zwei Möglichkeiten, Abacus (und auch die anderen QL-Programme) zu laden. Beim ersten wird die Abacus-Cartridge in den Microdrive 1 eingesetzt und über F1 der Monitor oder über F2 der Fernseher gewählt. Danach laden die Boot-Routinen des QL-Paketes das Programm automatisch. Das Anfangsmenü erscheint aber auch nach der Eingabe von lrun mdv1_boot, wenn die Anzeigeart bereits angegeben wurde (vorausgesetzt, das Programm ist in Laufwerk 1).

Der Bildschirm zeigt die obere linke Ecke der Kalkulationstabelle, die im Handbuch als Raster (Grid) bezeichnet wird. Anfangs werden nur die Spalten A bis F und die Zeilen 1 bis 15 angezeigt. (Abacus kann maximal 64 Spalten und 255 Zeilen verarbeiten, im Vergleich: Vu-Calc hat maximal 28 Spalten und 55 Zeilen.) Oberhalb dieses Rasters befinden sich der Promptbereich, eine Dateneingabezeile und Statusinformationen. Die Prompts sind besonders für den Anfänger hilfreich, da sie die verfügbaren Befehle anzeigen. Mit F2 lassen sich die Prompts löschen. Der Promptbereich enthält Informationen darüber, welche Funktionstasten welche Abläufe steuern, wie der Cursor bewegt wird, wie sich Felder direkt ansprechen lassen, wie Daten und Texte eingegeben werden und wie der Befehlsaufruf funktioniert. Die Abläufe lassen sich jedoch nicht direkt per Cursor anwählen, sondern müssen vom Anwender über die Tastatur eingegeben werden.

Für einfache Aufgaben kann Abacus fast sofort eingesetzt werden. An einige Befehle und Formeln für höher entwickelte Modelle muß man sich jedoch zuerst noch gewöhnen. Unser Beispiel – wiederum das Haushaltsbudget – zeigt die Arbeitsweise.

Zunächst muß der Titel „Haushaltsbudget" eingegeben werden. Dazu stellen Sie den Cursor mit den Steuertasten in Feld D1 und geben ein Anführungszeichen und den Text ein. (Wie bei Vu-Calc muß allen Texteingaben ein doppeltes Anführungszeichen vorangehen.) Bei Abacus – wie auch in vielen anderen Kalkulationssystemen – kann der Text eines Feldes in das dahinterliegende „überfließen", wenn dieses Feld leer ist. In die Tabelle lassen sich daher beliebig lange Titel eintragen.

Der Titel soll nun durch eine Unterstreichung hervorgehoben werden. Dazu stellen Sie den Cursor in das Feld unterhalb des Titels (D2) und geben rept("=",len(d1)) ein. Rept entspricht dem Befehl REPLICATE von Vu-Calc. Das Gleichheitszeichen (=) ist das Symbol, das wiederholt werden soll (in diesem Fall als doppelte Unterstreichung). Der Rest des Befehls – len(d1) – gibt an, daß das Zeichen (=) über die Länge des in Feld D1 enthaltenen Textes kopiert werden soll.

Formatänderungen

Anders als bei Vu-Calc mit neun Zeichen pro Feld können in Abacus unterschiedliche Feldbreiten angegeben werden. Für unser Modell wollen wir Spalte A verbreitern und – um die Daten für sechs Monate anzeigen zu können – die anderen Spalten verkleinern. Die Funktionstaste F3, gefolgt von G (der Kennung für den Befehl GRID – Raster) und W (für den Befehl WIDTH – Breite) ruft diese Funktion auf. Dabei zeigt die Eingabezeile die aktuelle Spaltenbreite (10) an. Um Spalte A auf 15 zu erweitern, geben Sie hier die Zahl 15 ein. Das Programm fragt nun nach dem Bereich, für den die neue Feldbreite gelten soll. Mit den beiden Parametern A, A geben Sie an, daß nur Spalte A diese Breite haben soll. Auf die gleiche Weise können die Spalten B bis G auf die Breite von 6 gesetzt werden. Auf dem Monitor erscheinen so die Daten von sechs Monaten.

Monatsnamen können als Texte in die entsprechenden Spalten eingetragen oder von Abacus automatisch aufgerufen werden. Setzen Sie den Cursor auf A3 und geben Sie row=month (col()–1) ein. Tippen Sie B und G, wenn die Eingabezeile nach dem Bereich fragt – die Spalten werden dann automatisch mit den Monatsnamen versehen.

Als nächsten Schritt geben Sie nun die Titel der einzelnen Zeilen ein, wobei der Cursor nach jeder Eintragung um eine Zeile nach unten bewegt werden muß (Abacus führt dies nicht automatisch aus). Unser Beispiel bezieht sich auf einen Vertreter, dessen Einkommen sich aus einer Provision und dem Festgehalt errechnet. Am Ende jedes Monats werden die aktuellen Verkaufszahlen eingetragen und die Zukunftswerte entsprechend korrigiert. Das fertige Modell zeigt negative Werte in Klammern an. Hier die Titel der einzelnen Reihen: Verkäufe, geschätzt und real; Provision; Grundgehalt; Gesamteinkommen; Ausgaben: Hypothek, Grundsteuer, Wasser, Elektrizität, Gas, Telefon; Gesamtausgaben; Nettoeinkommen; Bankkonto: Anfangs- und Endsaldo.

Nach Eingabe der Titel können Sie die Tabelle mit Werten füllen. Zunächst die geschätzten Verkäufe für die nächsten sechs Monate: 10 000, 12 000, 13 000, 11 000, 12 000 und 15 000. Die Zahlen werden ohne Punkt oder andere Spezialzeichen eingegeben. Da noch keine Realwerte existieren, wird die Provision aus den geschätzten Zahlen errechnet. Sie beträgt 20% aller Verkäufe über 10 000 Mark. Die Formel für Feld B10 lautet daher row=(verkauf−10000)*.2. Nachdem der Bereich B bis G angegeben wurde, werden die Provisionen sofort berechnet und angezeigt.

Das monatliche Grundgehalt von 2000 Mark wird in B11 als row=2000 eingegeben und damit für die Spalten B bis G wiederholt. In B13 kann nun die Formel für das Gesamteinkommen angegeben werden: row=sum(col) für Zeile 0 bis 11 und Spalte B bis G. Damit ist die Berechnung des Einkommens beendet. Das Gesamteinkommen läßt sich nun durch Striche hervorheben. Setzen Sie den Cursor auf B12 und geben Sie row=rept("−", width()−2) ein. Die Tabelle zeigt nun vier Striche unter dem Wert für Grundgehalt an. Die einfachste Methode, die Striche auch in Zeile 14 – unter das Gesamteinkommen – zu setzen, ist der Befehl ECHO. Das System fragt nun nach dem Bereich, für den die Unterstreichung wiederholt werden soll – geben Sie B14:G14 an. (Freie Zeilen können Sie mit dem GRID-Befehl einsetzen.) Nun sollten sämtliche Werte von B2 bis G14 mit dem Befehl J rechtsbündig gestellt werden, damit alle Zahlen in einer Reihe untereinander stehen.

Die konstanten Ausgaben (beispielsweise Hypothekenraten) können mit einer „row"-Formel eingetragen werden. Zahlungen in unterschiedlicher Höhe (Elektrizität und Gas) müssen Sie jedoch einzeln eingeben. Die Gesamtausgaben lassen sich nun – wie die Gesamteinnahmen – als Summe all dieser Werte berechnen und definieren.

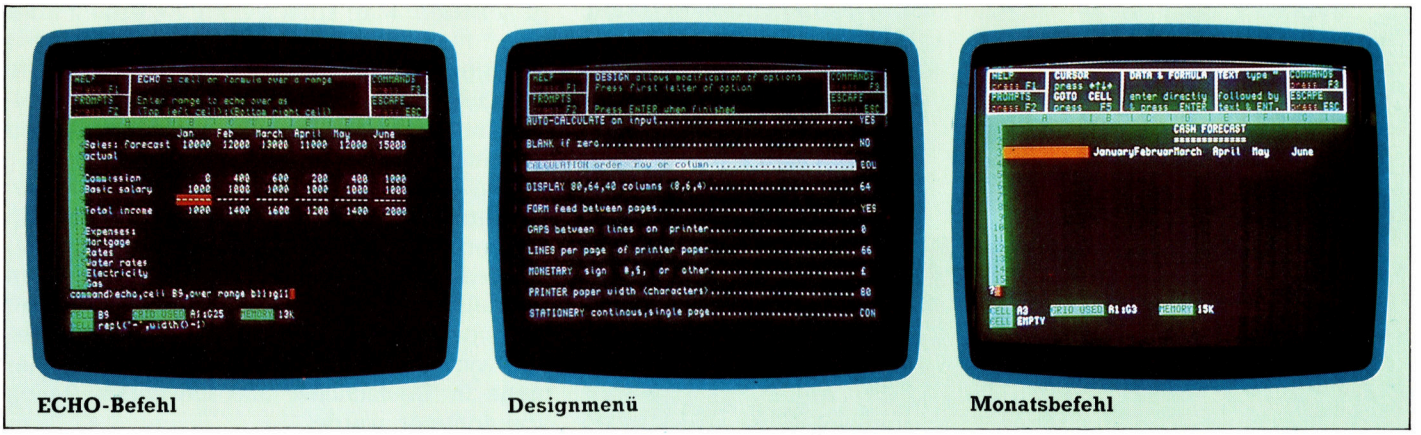
Die Endberechnung

Mit „row" kann auch das Nettoeinkommen aus Gesamteinkommen minus Gesamtausgaben berechnet werden. Die Formel lautet: row=gesamteinkommen−gesamtausgaben. Beachten Sie bei der Benennung, daß der Abacus bei einem Befehl alle Zeichen ignoriert, die hinter einem Leerzeichen stehen.

Als letzter Schritt wird der Banksaldo berechnet. Geben Sie zunächst den Anfangssaldo (beispielsweise eine Überziehung von 200 Mark) als −200 ein. Kalkulieren Sie den Endsaldo mit der Formel row=nettoeinkommen+anfangssaldo. Der Cursor sollte dabei in B28 stehen, und der Bereich B bis G muß angesprochen werden. Der angezeigte Wert ist der Endsaldo für Januar. Die Anfangssalden der anderen Monate werden mit der Formel row=B28 errechnet, die in C27 eingegeben wird. Damit dieser Vorgang in allen Spalten funktioniert, muß zunächst mit dem Befehl DESIGN die Berechnungsreihenfolge von Reihe auf Spalte geändert werden. Nach dieser Umstellung werden alle Anfangs- und Endsalden automatisch neu berechnet, wenn auch nur ein Wert geändert wird. Unser Modell zeigt negative Salden im Augenblick noch mit einem vorangestellten Minuszeichen an. Mit dem Befehl UNITS erscheinen nach der Eingabe von B alle Minuswerte in Klammern.

Das fertige Modell kann nun mit SAVE und einem geeigneten Dateinamen gespeichert werden. Anschließend können Sie die berechneten Daten ausdrucken.

Attraktiver Bildschirmaufbau, viele Bearbeitungsmöglichkeiten und ein gut ausgebauter Prompt- und Help-Bereich machen Abacus zu einem leicht und angenehm zu bedienenden Kalkulationssystem. Das Bild zeigt den Befehl ECHO, mit dem sich eine Zeile in eine andere kopieren läßt, weiterhin das Menü der DESIGN-Befehle zur Formatierung der Tabelle. Der Monatsbefehl generiert automatisch alle Monatsnamen. Das Anfangsmenü enthält im oberen Viertel des Bildschirms den Promptbereich. Das Befehlsmenü zeigt eine leere Tabelle.



Wege des Erfolgs

Xerox, einer der größten Hersteller reprografischer Einrichtungen, setzt alles daran, den Bereich der Büro-Automatisierung zu erobern. Das Unternehmen hat einen hervorragenden Ruf – in den USA geht das soweit, daß man dort Fotokopieren als „Xeroxen" bezeichnet.

Einer der größen PARC-Erfolge war die Entwicklung von STAR, einem auf der Sprache SMALL-TALK basierenden Programmiersystem. STAR vereint bei der Arbeit Programme und Daten in einem File. Diese Entwicklung war Grundlage für Apples Lisa-Technologie – das Gros der Entwickler aus dem Lisa-Team hatte zuvor bei PARC gearbeitet.

Anfang der siebziger Jahre plante das Xerox-Unternehmen ein umfangreiches Forschungsprojekt, um einen Traum zu verwirklichen: Man wollte Informationen mit Hilfe von Leitungen in jedes Büro legen. So, wie man sonst einen Strom- oder Wasseranschluß einsetzt. Xerox stellte ein Forschungsteam zusammen, das über alle nur denkbaren Freiheiten verfügte. Das wurde auch durch die Ansiedlung des Teams in Palo Alto, Kalifornien, betont. Dieser Ort liegt extrem weit von der Xerox-Zentrale in Rochester, New Hampshire, entfernt.

Die Ansiedlung im kalifornischen Santa Clara County erwies sich als außerordentlich fruchtbar. Nahe der Stanford Universität gelegen, die wegen ihrer computerwissenschaftlichen Leistungen und der Forschungen im Bereich der künstlichen Intelligenz einen hervorragenden Ruf genießt, fanden sich viele der Top-Wissenschaftler im Palo Alto Research Center (PARC) zusammen. PARC entwickelte sich zum Zentrum der Computerkultur. Dort entstand eine Fachsprache, die nur Eingeweihte verstehen. Zahlreiche Xerox-Produkte wurden in der Entwicklungsphase mit Spitznamen benannt. Die 820er-Microcomputerserie beispielsweise lief unter der Bezeichnung „Wurm". Hintergrund war, daß sie den „Apfel (Apple) schlucken" sollte.

Vorrangige Aufgabe des Forschungsteams war die Entwicklung eines Local Area Network (LAN). Dieser Begriff wird heute im Computeralltag wie selbstverständlich benutzt, doch als Xerox Ende der sechziger Jahre auf Hawaii das erste experimentelle Netzwerk installierte, war das ein revolutionärer Schritt. Voraussetzung für solche Verbindungen zwischen Großrechnern und Terminals waren kostspielige Verkabelungen, damit schnelle Datenübertragung gewährleistet werden konnte. Doch schon bei Kabeln, die länger als 20 Meter waren, gab es Probleme. Zwar ließen sich Telefonleitungen verwenden, doch die Netzwerkleistung wurde durch die Übertragungsrate von nur 9600 Baud erheblich eingeschränkt.

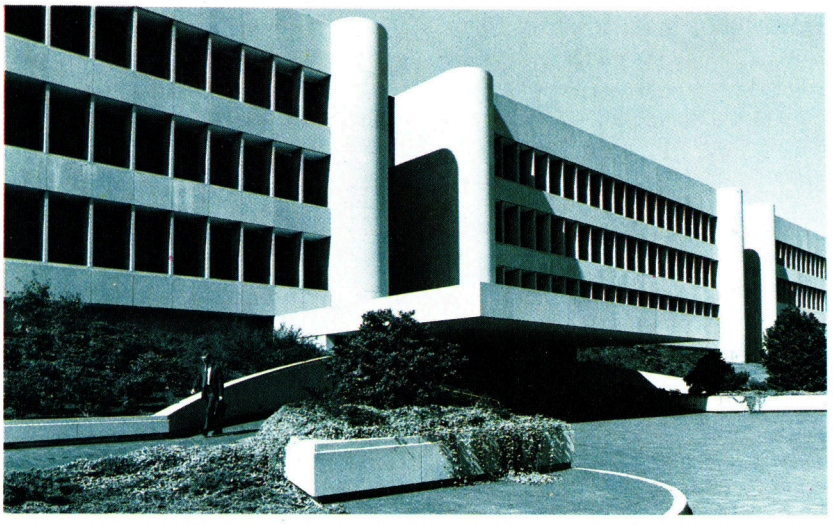
Das Ethernet LAN-Konzept

Ziel der Arbeiten in Palo Alto war die Entwicklung eines schnell arbeitenden Netzwerks, mit dem kleinere Computer miteinander verbunden werden konnten. Der Anwender sollte sowohl auf größere Computerkapazitäten zurückgreifen können, als auch Zugriff zu größeren Rechnern, Festplatten und anderen kostspieligen Peripheriegeräten wie Plotter oder Drucker haben. Dies war die Basis für das Ethernet LAN-Konzept.

Innerhalb dieses Systems wurden Verbindungen mit normalen Koaxialkabeln hergestellt. Sie ermöglichen eine Übertragungsgeschwindigkeit von zehn Millionen Bits pro Sekunde und sind neben der Datenübertragung auch für die Übertragung digitalisierter Grafik- und Sound-Informationen geeignet.

Mitte der siebziger Jahre war Ethernet funktionsfähig. Xerox sah die Notwendigkeit, andere Hersteller einzubeziehen, um damit einen Kommunikationsstandard für Computer zu schaffen. Man sprach mit IBM, erhielt aber einen negativen Bescheid. Digital Equipment dagegen war mit von der Partie. 1975 sicherte sich Xerox die Kooperation des Chip-Herstellers Intel, der die Produktion der speziellen Empfänger-Chips aufnahm.

Man experimentierte mit Ethernet in einem zu Versuchszwecken eingerichteten Büro in Schweden. Nach erfolgreicher Beendigung der Testphase wurde das System von anderen Herstellern adaptiert. Heute ist es ein offizieller internationaler Standard, der unter anderem von Hewlett-Packard, ICL in England, Siemens in Deutschland und Olivetti in Italien übernommen wurde.



LED-Anzeige

In dieser Folge des Selbstbau-Kurses wollen wir das User-Port-System durch zwei Sieben-Segment-Anzeigen weiter ausbauen. Sie stellen die Daten des User Port in hexadezimaler Form dar.

Zur Ansteuerung von Hexadezimal-Displays werden vier Bits benötigt – diese ermöglichen 16 Kombinationen von 0 und 1. Für die Anzeige eines Acht-Bit-Wertes sind demzufolge zwei Hexadezimalziffern nötig, eine für die vier niedrigen, eine für die vier höherwertigen Bits. Ein LED-Display besteht zwar aus sieben Einzel-LEDs, kann aber durch entsprechende Decodierschaltungen mit nur vier Leitungen (Bits) gesteuert werden.

Decodierer „übersetzen" die Anweisungen des Computers für die angeschlossenen Peripheriegeräte in elektrische Signale und umgekehrt. Wir haben zwar bereits selbst Decodier-Schaltungen gebaut, in diesem Fall soll aber eine fertige Logikschaltung verwendet werden – der Chip 7447 aus der Bauteil-Liste.

Aus je vier Dateneingängen werden durch die Logikschaltung in den Decodierern sieben Steuerausgänge für die LEDs. Die Schaltung würde etwa bei Ansteuerung mit 0111 am Eingang genau die Segmente einschalten, die gemeinsam eine 7 – hexadezimale Entsprechung von 0111 – darstellen.

Hexadezimalziffern über neun werden durch die ersten sechs Buchstaben des Alphabets – A bis F – dargestellt. Die Decodierschaltung erzeugt bei der Buchstabendarstellung etwas ungewohnte Zeichen. Eine Darstellung mit den üblichen Zeichen hätte jedoch zusätzlicher Logikfunktionen bedurft, die der Hersteller eingespart hat, um den Preis niedrig halten zu können.

Liste der Bauteile

Anzahl	Bauteil
14	Widerstände, 330 Ohm, 0,5 Watt
2	7447 BCD-7-Segment-Decodierer
1	Zweistellige 7-Segment-LED-Anzeige
2	IC-Sockel, 16 Pins
1	12-poliger Minicon-Winkelanschluß
1	10-poliger Minicon-Winkelstecker
*	8-poliges Flachbandkabel
*	7-poliges Flachbandkabel
*	Abisolierte, verzinnte Litze
1	Lochplatine (Veroboard), 36 Streifen à 50 Löcher
1	Kunststoff-Gehäuse, 116 x 61 x 31 mm

* Diese Bauteile haben Sie vielleicht noch von früheren Arbeiten übrig. Den 12poligen Minicon-Anschluß brauchen Sie nur, wenn der System-Bus weiter ausgebaut werden soll.

Das Multiplex-Verfahren

Wenn Sie die Anzeige-Schaltung fertig aufgebaut haben, zeigt sie fortlaufend den Zustand des User-Port-Datenregisters in hexadezimaler Form an. Wegen der acht Datenleitungen können beide Anzeigen unabhängig voneinander betrieben werden. Das ist nicht immer der Fall, oft teilen sich mehrere Sieben-Segment-Anzeigen eine geringere Anzahl von Datenleitungen. Damit mehrere Anzeigen gleichzeitig dargestellt werden können, verwendet man das „Multiplex-Verfahren". Dabei werden die einzelnen Anzeigen nacheinander vom Decodierer angesteuert, wobei auch die anzuzeigenden Werte umgeschaltet werden. Wenn dieser Vorgang schnell genug abläuft, scheinen alle Anzeigen gleichzeitig zu leuchten – jede Sieben-Segment-Anzeige zeigt nur den Wert, der ihr zugeordnet wurde.

Das Prinzip der Multiplex-Technik läßt sich auch mit unseren beiden Sieben-Segment-Displays verdeutlichen: Die dezimale 15 stellt der Decodierer durch eine Leerstelle dar, damit läßt sich also ein Display abschalten, während das andere leuchtet. Das folgende Programm fragt nach einer darzustellenden Ziffer und scheint diese dann auf beiden Displays gleichzeitig anzuzeigen. Wenn Sie die Leertaste am Rechner drücken, wird eine Routine aktiviert, die die Schwingungen zwischen den beiden Anzeigen künstlich verzögert – jetzt zeigt sich, daß die Ziffer in Wirklichkeit zwischen den Anzeigen hin- und herspringt. Nach Loslassen der Leertaste läuft das Programm wieder mit der normalen Geschwindigkeit – jetzt sieht es wieder so aus, als ob die Ziffer auf beiden Anzeigen ohne Unterbrechungen dargestellt wird.

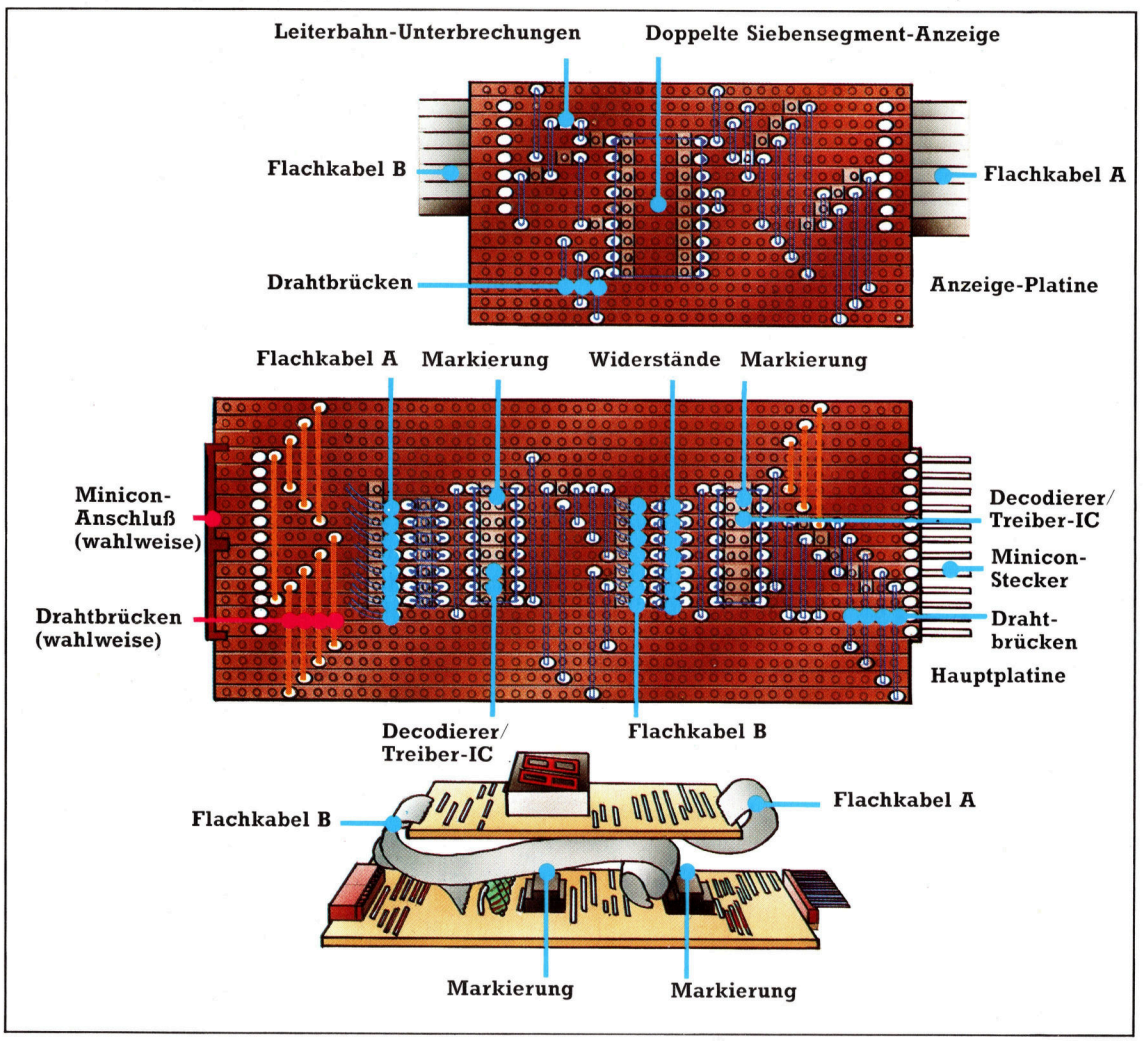
```
10 REM BBC MULTIPLEXING
20 DDR=&FE62:DATREG=&FE60
30 ?DDR=255
40 left_blank=15*16
50 right_blank=15
55 :
60 REPEAT
70 INPUT"DATA TO BE MULTIPLEXED";data
80 ?DATREG=data+left_blank
90 PROCslower
100 ?DATREG=data*16+right_blank
110 PROCslower
120 GOTO80
130 END
140 :
150 DEF PROCslower
155 REM IS SPACE BAR PRESSED ?
```

```
160 IF INKEY(-99)=-1 THEN PROCdelay
170 ENDPROC
180 :
190 DEF PROCdelay
200 FOR I =1 TO 500:NEXT
210 ENDPROC

10 REM CBM 64 MULTIPLEXING
30 DDR=56579:DATREG=56577
40 POKEDDR,255
50 LB=15*16:RB=15
60 INPUT"DATA TO BE MULTIPLEXED";DT
70 POKEDATREG,DT+LB
80 GOSUB1000:REM SLOWER
90 POKE DATREG,DT*16+RB
100 GOSUB1000:REM SLOWER
110 GOTO70
120 :
1000 REM SLOWER S/R
1010 GETA$
1020 IFA$=" "THENGOSUB2000:REM DELAY
1030 RETURN
1999 :
2000 REM DELAY S/R
2010 FORI=1 TO 250:NEXT
2020 RETURN
```

Unsere beiden Sieben-Segment-Anzeigen lassen sich sehr einfach als Hexadezimal-Zähler nutzen. Damit kann etwa die Anzahl der Schaltvorgänge eines am User Port angeschlossenen Tasters festgestellt werden. Auf den ersten Blick erscheint diese Anwendung ganz einfach, jedoch stellt sich die Frage, wie neben den acht Steuerleitungen für die Anzeige weitere Leitungen mit dem User Port verbunden werden sollen. Ist etwa Kanal 0 auf Eingabe und damit auf „High" geschaltet, kann das entsprechende Bit im Datenregister nicht mehr zur Steuerung der LEDs eingesetzt werden. Falls man 128 (Binär 10000000) ins Datenregister schreiben will, verwandelt sich die Zahl durch die Eingabe „High" von Bit 0 sofort in 129 (10000001). Ein der Multiplex-Technik ähnliches Vorgehen hilft uns hier aus der Klemme: Kanal 0 wird immer nur ganz kurz von Aus- auf Eingabe geschaltet.

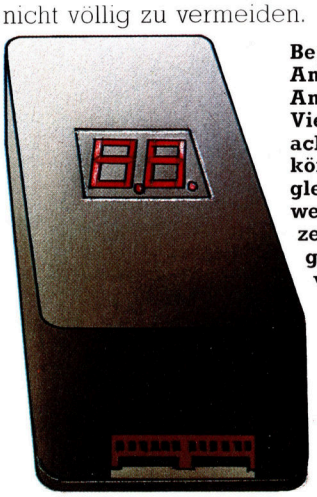
```
10 REM BBC COUNTER
20 DDR=&FE62:DATREG=&FE60
50 count=0
55 :
60 REPEAT
70 PROCinput
72 PROCadd
73 FORI=1TO40
75 PROCdisplay
77 NEXT I
80 UNTIL count>255
90 END
999 :
1000 DEF PROCadd
1010 IF flag =1 THEN count=count+1
1050 ENDPROC
```

Zuerst schneiden Sie die Lochplatinen zu – einmal mit 19 Streifen à 46 Löchern, einmal 15 Streifen à 28 Löcher. Als nächstes die Leiterbahnen unterbrechen. Danach IC-Sockel, Drahtbrücken und Widerstände einlöten. Die rot gedruckten Drahtbrücken können Sie fortlassen, wenn die Bus-Erweiterung nicht eingebaut werden soll. Minicon-Stecker und (bei Bedarf) -Sockel auf die Hauptplatine löten. Die LED-Anzeigen werden mit dem Punkt in Richtung zur Anschlußseite der Platine eingesetzt. Jetzt fehlen noch die Verbindungskabel – sie müssen ohne Verdrehung von der einen zur anderen Platine führen. Ganz zuletzt die Decodierer-ICs einsetzen.



```
1499 :
1500 DEF PROCinput
1510 ?DDR=254
1515 flag=0
1520 IF(?DATREG AND 1)=0 THEN flag=1
1525 REPEAT UNTIL (?DATREG AND 1)=1
1530 ENDPROC
1999 :
2000 DEF PROCdisplay
2010 ?DDR=255
2030 ?DATREG=count
2040 ENDPROC

10 REM CBM 64 COUNTER
30 DDR=56579:DATREG=56577
40 CC=0:REM INIT COUNT
50 :
60 GOSUB1000:REM INPUT
70 GOSUB2000:REM ADD
80 FOR I=1TO20
90 GOSUB3000:REM DISPLAY
100 NEXT I
110 IF CC<255 THEN60
120 END
999 :
1000 REM INPUT S/R
1010 POKEDDR,254
1020 FL=0
1030 IF(PEEK(DATREG)AND 1)=0 THEN FL=1
1040 IF(PEEK(DATREG)AND 1)<>1 THEN 1040
1050 RETURN
1999 :
2000 REM ADD S/R
2010 IF FL=1 THEN CC=CC+1
2020 RETURN
2999 :
3000 REM DISPLAY S/R
3010 POKEDDR,255
3020 POKEDATREG,CC
3030 RETURN
```

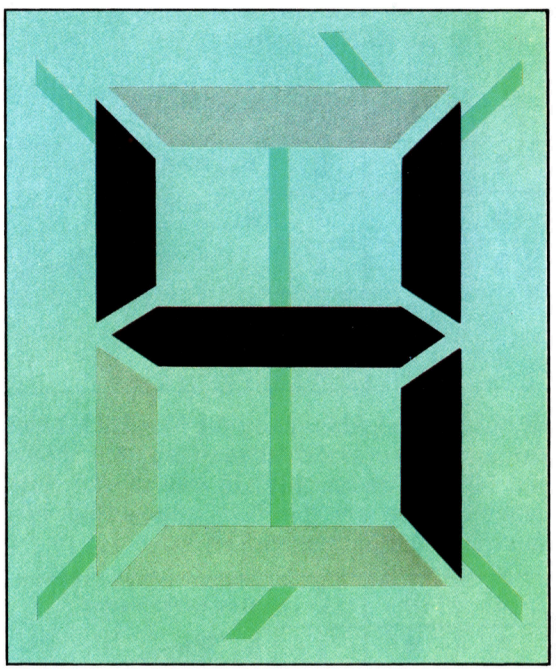
Bei jedem Programmdurchlauf wird Kanal 0 einmal auf Eingabe, durch eine FOR . . . NEXT-Schleife aber sehr häufig auf Ausgabe geschaltet. Beim Commodore 64-Programm kommen auf eine Eingabe 20 Anzeige-Zyklen, beim Acorn B sind es sogar 40 Ausgaben pro Eingabe. Dabei ist immer noch ein leichtes Flackern der LED-Anzeigen zu erkennen. Es sollte aber nicht noch weniger Zeit für die Eingabe zur Verfügung stehen – ein Signal auf dem Eingangskanal könnte andernfalls einfach übersprungen werden. Leider sind Kompromisse zwischen Wunsch und Möglichkeit beim „Real-Time"-Betrieb eines Computers nicht völlig zu vermeiden.

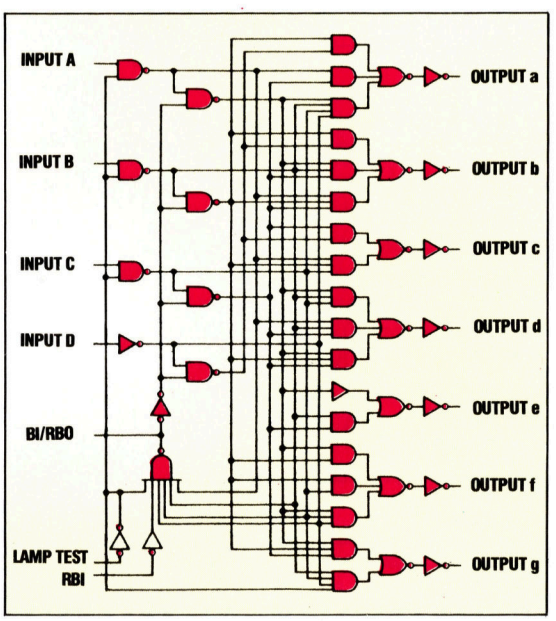


Beide Sieben-Segment-Anzeigen brauchen zur Ansteuerung je ein Vier-Bit-Signal. Mit den acht User-Port-Kanälen können also beide gleichzeitig betrieben werden. Die neue Anzeige ist mit den früher gebauten Zusatzgeräten voll kompatibel.

Dezimal	Binär				Ausgabe	Anzeige
	D3	D2	D1	D0	a b c d e f g	
0	0	0	0	0	0000001	0
1	0	0	0	1	1001111	1
2	0	0	1	0	0010010	2
3	0	0	1	1	0000110	3
4	0	1	0	0	1001100	4
5	0	1	0	1	0100100	5
6	0	1	1	0	1100000	6
7	0	1	1	1	0001111	7
8	1	0	0	0	0000000	8
9	1	0	0	1	0001100	9
10	1	0	1	0	1110010	c
11	1	0	1	1	1100110	J
12	1	1	0	0	1011100	u
13	1	1	0	1	0110100	S
14	1	1	1	0	1110000	t
15	1	1	1	1	1111111	

Die Digital-Anzeige wird vom User Port über eine Vier-Bit-Binärzahl angesteuert. Jeder Wert zwischen 0000 und 1111 entspricht einer Sieben-Bit-Zahl, deren einzelne Ziffern für je ein Segment der Anzeige zuständig sind. Eine 0 bedeutet, daß das zugehörige Segment leuchtet, bei einer 1 ist es ausgeschaltet.





Die interne Schaltung des ICs ist ganz unkompliziert – einige Logik-Gatter wandeln das Vier-Bit-Eingangssignal so um, daß damit die sieben Steuerleitungen der Anzeige versorgt werden können. Mit dem Eingang „Lamp Test" lassen sich zu Prüfzwecken alle Segmente gleichzeitig einschalten.

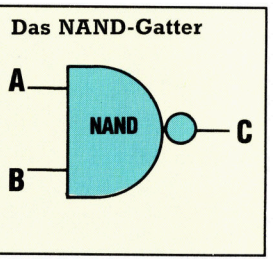
Alternative Wege

Theoretisch lassen sich alle Logik-Aufgaben mit einer Kombination der drei Gatterbausteine (AND, OR und NOT) lösen. Zusätzlich gibt es die NAND- und NOR-Gatter.

Wenn jede Logikschaltung allein mit AND-, OR- und NOT-Gattern aufgebaut werden kann – wozu dienen dann weitere Bauteile? Ganz einfach: Der Einsatz von NAND- und NOR-Gattern senkt die Herstellungskosten einer Schaltung – Leitungen können eingespart werden, und auch die Verwirklichung eleganterer Lösungen ist keine Seltenheit. Damit stehen vier verschiedene Techniken bereit, um eine Logikschaltung zu realisieren:

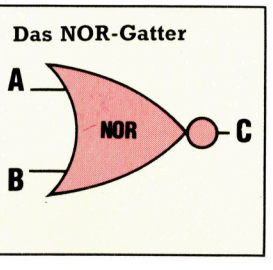
a) Mit AND-, OR- und NOT-Gattern
b) Nur mit NAND-Gattern
c) Nur mit NOR-Gattern
d) Aus allen Gattertypen zusammen

A	B	C
0	0	1
0	1	1
1	0	1
1	1	0



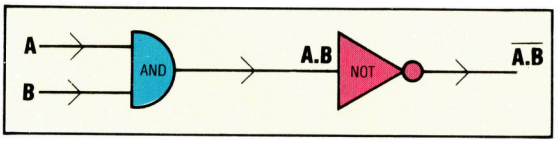
NAND ist eine Abkürzung für Not AND. Beim Vergleich der Wahrheitstabellen von AND- und NAND-Gattern fällt auf, daß in den Ausgangs-Spalten nur die Eins gegen Null vertauscht ist und umgekehrt.

A	B	C
0	0	1
0	1	0
1	0	0
1	1	0

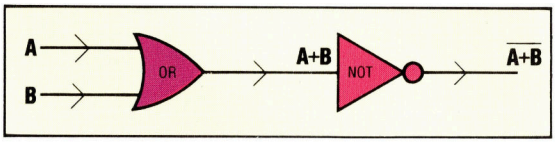


Bei NOR – der Abkürzung von Not OR – ist es ganz ähnlich. Im Vergleich mit dem OR-Gatter zeigt sich auch hier, daß die Einsen und Nullen ihre Plätze miteinander getauscht haben.

Spezielle Zeichen gibt es in der Booleschen Algebra weder für die NAND- noch für die NOR-Funktion – beide lassen sich als Kombination aus AND-, OR- und NOT-Symbolen darstellen. Dem NAND-Gatter entspricht diese Schaltung:

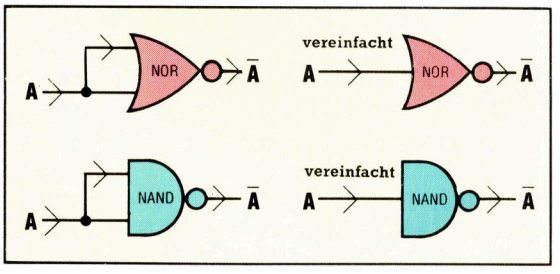


Das NOR-Gatter ist ein OR-Gatter, dem ein NOT-Gatter angehängt ist:



Genau wie sich NAND- und NOR-Gatter durch Kombination von AND/OR/NOT-Schaltungen nachbilden lassen, ist auch die Simulation der drei wichtigsten Gatterbausteine durch mehrere NAND- bzw. NOR-Gatter möglich.

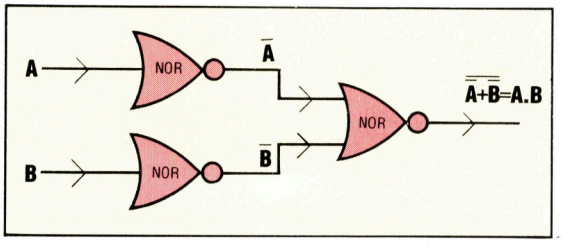
NOT-Gatter: Ein NOT-Gatter erhält man, wenn die beiden Eingänge eines NOR- oder NAND-Gatters miteinander wie folgt verbunden werden:



AND-Gatter: Nach der Booleschen Algebra heißt die Ausgabe eines AND-Gatters mit den Eingängen A und B A.B. Dieser Ausdruck kann jedoch auch umformuliert werden:

$$A.B = \overline{\overline{A}.\overline{B}} \qquad \text{(weil } A = \overline{\overline{A}}\text{)}$$
$$= \overline{\overline{A}+\overline{B}} \qquad \text{(Gesetz von de Morgan)}$$

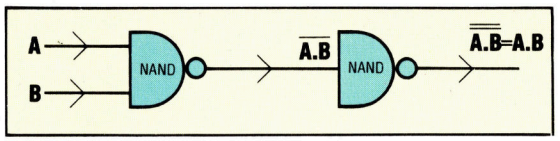
Also kann die Schaltung auch durch die Eingabe von NOT(A) und NOT(B) in ein NOR-Gatter umgesetzt werden:



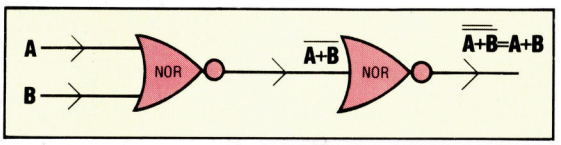
Auch die AND-Funktion läßt sich mit einem NAND-Gatter verwirklichen. Die Ausgabe des NAND-Gatters ist A.B. Verneint ergibt sich:

$$\overline{\overline{A.B}} = A.B$$

Hier die Schaltung dazu:



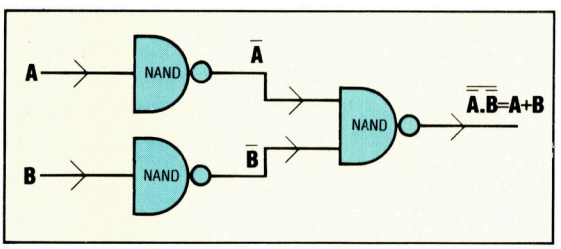
OR-Gatter: Der Verbindung zweier NAND-Gatter zu einem AND-Gatter entspricht die Verkettung zweier NOR-Gatter zu einem OR-Gatter:



Ausgabe eines OR-Gatters ist A+B. Mit den Regeln der Booleschen Algebra kann dieses Ergebnis auch in NAND-Form gebracht werden:

$$A + B = \overline{\overline{A}} + \overline{\overline{B}}$$
$$= \overline{\overline{A}.\overline{B}}$$

Die entsprechende Schaltung mit NAND-Gattern sieht daher folgendermaßen aus:

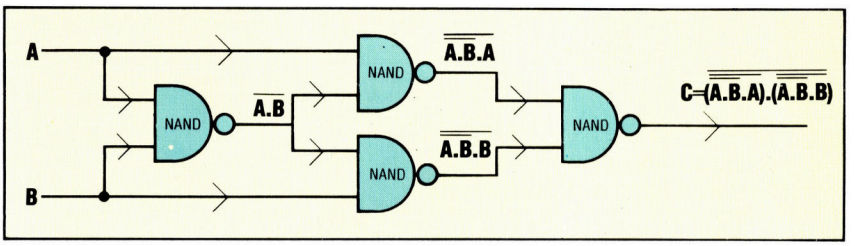


Schaltungen mit NAND-Gattern werden zur Vereinfachung zuerst als Gruppen von AND- und OR-Gatterkombinationen formuliert. Durch mehrfache Anwendung des Gesetzes von de Morgan können sie danach vollständig in NAND-Form gebracht werden. Ein Beispiel zeigt die entsprechenden Regeln für NOR-Gatter. Ein Rückblick auf das ausschließende OR-Gatter (XOR) hilft zum Verständnis der Methode: Die Ausgabe eines XOR-Gatters konnte mit dem Ausdruck $C = \overline{A}.B.(A + B)$ beschrieben werden.

Dieser Ausdruck soll nun so umgestellt werden, daß sich die Schaltung des XOR-Gatters nur mit NAND-Bausteinen realisieren läßt. Dazu wird die Formel in mehrere Gruppen von mit OR verbundenen AND-Funktionen umgewandelt.

$C = \overline{A}.B.(A + B)$ (Klammern
$= (\overline{A}.B.A) + (\overline{A}.B.B)$ ausmultiplizieren)
$= \overline{\overline{(\overline{A}.B.A)}.\overline{(\overline{A}.B.B)}}$ (Gesetz von de Morgan)

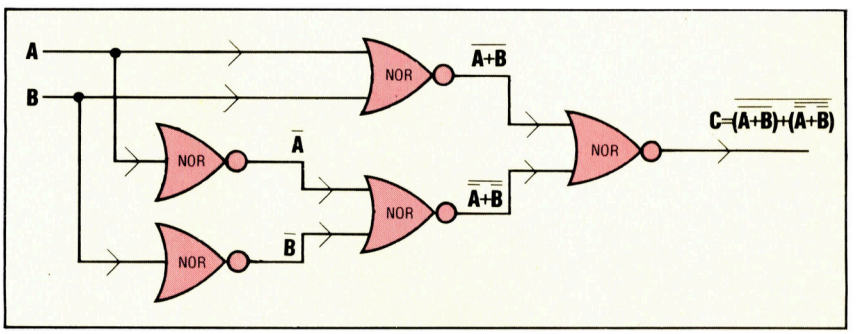
Bei einem so schwierigen Ausdruck wie diesem beginnt man beim Entwurf der Schaltung mit dem Ausgang und arbeitet sich zu den Eingängen vor. Verfolgen Sie auch unseren Schaltungsaufbau in dieser Richtung:



Um die NOR-Form zu bekommen, müssen wir wieder mit dem Ausdruck für das XOR-Gatter anfangen und es zu Gruppen aus OR-Funktionen arrangieren, die mit AND verbunden sind. Dazu wird das Gesetz von de Morgan auf die linke Seite angewendet:

$C = \overline{A}.B.(A + B)$
$= (\overline{A} + \overline{B}).(A + B)$
$= \overline{\overline{(\overline{A} + \overline{B})} + \overline{(A + B)}}$

Auch hier läßt sich die dazugehörige Schaltung am besten vom Ausgang her zu Eingängen hin entwerfen.



Lösung der Übungsaufgaben

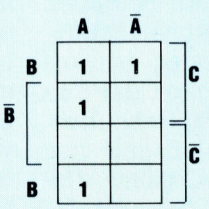
1a)

Eingänge			Ausgang
A	B	C	P
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

1b) Der Boolesche Ausdruck für P lautet:

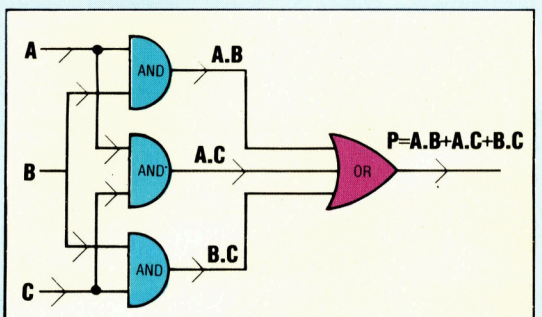
$P = \overline{A}.B.C + A.\overline{B}.C + A.B.\overline{C} + A.B.C$

Mit einer Karnaugh-Tafel kann der Ausdruck vereinfacht werden:



Dann ergibt sich $P = A.B + A.C + B.C$

1c) So sieht die Schaltung aus:



Knacken und Hacken

Der illegale Zugriff zu Großrechnern mit Heimcomputern und Modems wird als „Hacken" bezeichnet. In jüngerer Zeit wurden mehrere solcher Fälle bekannt, bei denen auch Anlagen der Regierung und multinationaler Konzerne betroffen waren. „Hacken" ist nunmehr ein öffentlich diskutiertes Thema.

Illegaler Vertrieb

Pepsi Cola Nordamerika dementierte zwar, doch auch dieses Unternehmen gehört zu den „Hakker"-Opfern der letzten Jahre. Berichten zufolge gelang der illegale Zugriff auf den in Kanada aufgestellten Computer dieser Firma. Die Hacker schickten ganze Fuhren von Pepsi an bestimmte Orte, womit parallel auch gewaltige Geldbeträge für den Transport unrechtmäßig auf ihren Konten landeten.

Der Film „War Games" entfesselte die Phantasie vieler Computer-Freaks. Unter Verwendung eines Micros und eines Modems verschaffte sich der Filmheld illegal Zugang zu einer Reihe von Computern, um seine High-School-Prüfungsergebnisse zu verändern, Flugtickets zu buchen und die neuesten Spielprogramme in sein System zu holen. Allerdings unterlief ihm ein Fehler, als er unwissentlich den Rechner NORAD anzapfte, der die Luftverteidigung für Nordamerika steuert. So wurde im Film fast ein atomarer Weltkrieg entfacht. – Spannend gemachte Unterhaltung, aber doch sicherlich viel zu überzogen?

Computer-„Einbrüche" haben tatsächlich stattgefunden (und finden statt), wobei sich als Schuldige häufig Teenager mit Heimcomputern und Modems entpuppten. Zu den „Opfern" gehören leistungsfähige Großrechner von Universitäten und Großunternehmen, aber auch „Mailboxen" und Datenbanken, die von Computerfans mit Microcomputern betrieben werden. Jeder Computer, der externe Zugriffsmöglichkeiten über Telefon bietet, ist verwundbar. 1983 kam die Wirklichkeit der Fiktion doch sehr nahe: Zwei amerikanische „Hakker" wurden verdächtigt, illegal Zugriff zum NORAD-Computer-System in Omaha, Nebraska, gewonnen zu haben.

Die beiden Teenager aus Los Angeles waren ins Arpanet gelangt, das geheime Computernetzwerk, das vom Verteidigungsministerium der Vereinigten Staaten betrieben wird. Mit einem VC 20 und einem Tandy TRS-80 hatten die beiden in den Verzeichnissen mehrerer mit Arpanet verbundener Computer herumgeblättert, unter anderem in Computer von Forschungsunternehmen und Universitäten. Detailinformationen waren auf diesem Wege zwar nicht zu erhalten – das System wird vornehmlich für den Austausch wissenschaftlicher Daten benutzt –, doch die Leichtigkeit, mit denen den beiden Teenagern das „Einbrechen" gelungen war, sorgte im Verteidigungsministerium für beträchtliche Aufregung.

Grund für den vergleichsweise einfachen Zugriff der beiden Jungen war nicht ein Systemfehler, sondern menschliche Nachlässigkeit. Alle registrierten Arpanet-Benutzer haben Paßwörter, die leider oftmals nicht sehr geschickt gewählt werden. In diesem Fall nahmen die beiden Jungen an, daß die Universität von Kalifornien (California) in Berkeley Arpanet-Benutzer sein könne. Folgerichtig gaben sie als Paßwort „UCB" ein, um ins Netzwerk zu gelangen, und hatten so freien Zugang zu allen mit Arpanet verbundenen Computern, wovon einer NORAD ist, der sich in Omaha im unterirdischen Hauptquartier befindet.

Ungenügende Schutzroutinen

Nun mögen zwar NORAD-Computer immun gegen die Hackwut sein, viele andere Systeme aber sind es nicht. In einem anderen Fall, der im Juni 1983 bekannt wurde, brach eine Gruppe von in Milwaukee ansässigen Jugendlichen in mehr als 60 Computersysteme ein, die zu Hochschulen, Firmen und dem Los Alamos National Laboratory gehörten. Letzteres entwickelt Waffensysteme. Laut Auskunft der Behörden waren auch hier keine Detailinformationen zu erhalten – aber doch Aufzeichnungen, Berichte und Botschaften. Das FBI wurde zur Aufklärung eingeschaltet und fand die Gruppe, die sich „414s" nannte. Die 414s sagten zwar aus, daß sie in keinem der angezapften Computer geheime Informationen ge-

funden hätten, doch ist der Vorgang an sich alarmierend genug.

Dies sind nur einige der bekannt gewordenen Fälle. Viele andere Fälle wurden nicht publik gemacht, da die meisten Organisationen keinen Wert darauf legen, daß bekannt wird, daß ein 17jähriger mit einem 300 Mark-Rechner ins betreffende Großrechensystem eingedrungen ist. Andere Unternehmen haben gar nicht gemerkt, daß sie angezapft wurden: Oft ist sehr schwer festzustellen, ob ein nicht registrierter Benutzer am Werk war. Viele der raffinierteren Hacker jedoch hinterlassen Botschaften wie „Ihr erwischt mich sowieso nicht" und unterschreiben mit „System Kracker" oder „Captain Zap".

Wie wird nun eigentlich gehackt? Hacker benötigen einen Heimcomputer, ein Modem und ein gewisses Maß an Einfallsreichtum. Die erste Aufgabe ist, die Telefonnummer eines Computers zu finden. Das fällt bei öffentlich zugänglichen Netzwerken wie „The Source" in den USA leicht, da diese Nummern bekannt sind. Bei Privatcomputern wird's komplizierter. Weiß man aber ungefähr, wo der Computer aufgestellt ist, findet die vom Hauptdarsteller in „War Games" praktizierte Technik Anwendung: Er programmierte seinen Computer so, daß dieser jede nur denkbare Telefonnummer in seiner Heimatstadt anwählte. Antwortete ein Computer – identifizierbar durch einen speziellen Signalton –, machte das Programm einen entsprechenden Vermerk hinter der Nummer. War ein Mensch an der Leitung, „hängte" das Modem ein und wählte die nächste Nummer. Mit einem Auto-Wahl-Modem (Selbstwahl-Modem) erfolgt das tatsächlich automatisch. Manuelles Wählen wäre sehr zeitraubend.

Einmal mit dem Computer verbunden, erfolgt die Frage nach dem Paßwort. Einige Netzwerke erlauben nach Eingabe von „GUEST" oder „NEWUSER" den beschränkten Zugriff auf bestimmte Informationen. Echte Hacker aber versuchen, das Paßwort herauszufinden. Viele Computer verfügen nicht über ausreichende Schutzroutinen und erlauben mehrere Versuche, bevor sie die Verbindung abbrechen. Selbst danach kann man das System wieder anwählen und es von neuem versuchen, ohne daß der Computer nun etwa „mißtrauisch" wird.

Ist der Hacker erst einmal in ein System gelangt, kann er in sämtlichen Dateien herumstöbern, Spiele – soweit vorhanden – finden und sogar mit anderen „Hackern" kommunizieren. Leute mit zerstörerischen Absichten löschen Dateien, hinterlassen obszöne Nachrichten oder versuchen, das ganze System „zusammenbrechen" zu lassen.

Warum tun Hacker das? Gewöhnlich liegt der Reiz in der Herausforderung, ein System „zu knacken". Viele sind an den Datei-Inhalten nicht interessiert. Ihnen verschafft allein das Decodieren Genugtuung. Eine Gesetzesübertretung liegt dennoch vor, da sie illegal und ohne zu zahlen Computerzeit nutzen.

Banken sind schon seit längerem von Computerverbrechen betroffen. Bis vor kurzer Zeit wurden diese jedoch hauptsächlich von Angestellten ausgeführt, die hohe Geldsummen auf eigens dafür eingerichtete Konten überwiesen. Verständlicherweise treten Banken und Unternehmen mit solchen Fällen selten an die Öffentlichkeit, deshalb ist die Ermittlung der exakten Schadenshöhe sehr schwer. Aufgrund der weiten Verbreitung von Heimcomputern und durch immer intensivere Verwendung von Netzwerken steigt die Computer-Kriminalität. Ob es sich dabei nun um Teenager handelt, die nur Dateien löschen und Computerzeit stehlen, oder um professionelle Kriminelle, die Gelder auf ihre Konten transferieren – die dabei angewandten Methoden sind gleich.

Versuchen und Irren

Der Erfolg des Hackers hängt im wesentlichen von seiner Ausdauer ab. Selbst wenn es einem Hacker gelingt, ein bestimmtes System zu lokalisieren, wird er beim Versuch einzu„loggen" mit einem Dialog konfrontiert, wie er hier aufgeführt ist.

```
CONNECT QX001001 14.32
     12/7/84
>
USER?
>HELP
USER?
>£$OFF
USER?
>UK001
PASSWORD?
>GUEST
PASSWORD?
>NEWUSER
PASSWORD?
>QWERTY
LOGOFF 15.13 CONNECT
     TIME = 0.13 MINS
```

RETURN drücken: Abfrage nach Benutzer ID
No help: Nicht benutzerfreundlich
1. Versuch, ID einzugeben
ID-Zugang nicht erlaubt
2. Versuch, ID einzugeben
ID akzeptiert: Paßwort-Abfrage
Erster Versuch
Nicht akzeptiert: 2. Abfrage
Zweiter Versuch
Nicht akzeptiert: 3. Abfrage
Verzweiflungs-Versuch
Die Verbindung wird vom Computer beendet, da der Benutzer kein Paßwort gefunden hat

Zuweilen gelingt es, entweder durch Glück oder mit sehr viel Geduld, Paßwörter zu finden und in das System zu kommen. Folgendes Beispiel zeigt in Dialogform, wie ein Benutzer Informationen vertaulicher Art über einen legalen Benutzer erfragen kann:

```
CONNECT BYF990
     15.14.02 12/07/84
>
USER?
>UK001
PASSWORD?
>SYSOP
LOGON 15.15.07 12/07/84
HOST: BYF990/SPYLOM
USER: UK001
SERIAL NO: ZA180-7
PRIORITY: SUPERUSER
STATUS: ACTIVE
YOU ARE SYSOP
7 USER(S)
APP01 APP02 BYF7 BTY04
     BZX88 BZX02 SYSOP
>REMOVE ARP01
USERS(S) ARP01
     DISCONNECTED
>WHO IS BTY04
BTY04      CAREY DIMMIT,
     BTY LOPP,  742
     SILICON DV
HERTS 07662-093164
```

RETURN drücken
ID akzeptiert: Paßwort-Abfrage
1. Versuch: System-Operator
Serien-Nummer der Software
Offene Files
Benutzer können inaktiv gemacht werden, wenn sie das System nicht regelmäßig benutzen
Bestätigung
Auflistung aller Benutzer
Befehl für System-Operator reserviert
Zugangsberechtigter Benutzer entfernt

Sets und Mengen

In PASCAL gibt es außer den Datenstrukturen Array und File noch Sets und Records. In dieser Folge untersuchen wir, wie Sets funktionieren, und geben einen Überblick, welche der PASCAL-Operatoren Vorrang vor anderen haben.

Wir haben uns zwar oft mit dem Zeichensatz eines Computers beschäftigt, aber nie genau definiert, was der Unterschied zwischen einem Satz (Set) und einem Array ist. Ein Set (oder auch Menge) ist eine Ansammlung von Objekten, die als Einheit angesprochen wird und nicht über die einzelnen Elemente. Da Sets keine strukturierte Ordnung haben müssen, ist oft nur die Frage wichtig: Ist ein bestimmtes Objekt Element des Sets oder nicht?

Aus Gründen der Einsatzfähigkeit sind Sets in PASCAL bestimmten Beschränkungen unterworfen. So können Sets nur einfache Skalare als Elemente haben und keine Arrays oder andere Datenstrukturen. Außerdem wird dem „Basistyp" eines Sets beim Einsatz in einem Programm eine Obergrenze gesetzt. Die Definition eines Sets ist einfach und direkt:

```
TYPE
  Zahlen   = SET OF 0..127;
  Alphabet = SET OF 'A'..'Z';
  Farben   = SET OF (Rot, Gruen, Blau);
```

Der Wertebereich, der für den ordinalen Basistyp eines Sets möglich ist, wird wie ein normaler Unterbereichstyp angegeben.

Set-Variablen werden wie alle anderen Variablen mit VAR deklariert. Ihre Elemente können – in eckige Klammern eingeschlossen – direkt angegeben werden:

```
VAR
  Codes  : Zahlen;
  Palette : Farben;
BEGIN
  Codes  := [0..2,4,8,16,32,64];
  Palette := [Rot..Blau]; (* etc. *)
```

Dieser Programmteil ordnet dem Set „Codes" die Zahlen 0 bis einschließlich 2 und 4,8,16,32 und 64 zu. Da dieser Set keine bestimmte Ordnung enthält, könnten wir ihn auch als [64,32,16,8,4,0..2] schreiben. Die Angabe 2..0 wäre als Definition eines Unterbereichstyps zwar illegal, würde ansonsten aber eine Leermenge bezeichnen. Es ist auch möglich, Sets über die Anweisung JederSet :=[] mit der Leermenge dieses Typs zu initialisieren. Dies ist in PASCAL die einzige Ausnahme der Regel, daß sich der Typ jedes Elements per Test feststellen läßt. Ohne mindestens ein Set-Element kann der Compiler den Typ allerdings nicht bestimmen.

Einer der praktischsten Operatoren für Set-Strukturen ist das reservierte Wort „IN". Mit IN läßt sich testen, ob ein Objekt (oder Objekte) Element eines Sets ist. IN ist ein Vergleichsoperator, der zwei Operanden benötigt. Dabei muß auf der linken Seite ein Ausdruck stehen, der zu den Set-Elementen gehört, und auf der rechten Seite eine Set-Variable oder eines oder mehrere Elemente des gleichen Set-Typs. Das Ergebnis ist der Boolesche Wert „true", wenn das Objekt ein Element der Vergleichsmenge ist, und „false", wenn dies nicht der Fall ist.

„N IN Codes" und „Gruen IN Palette" ergeben daher legale Boolesche Werte. Ein Test, ob ein Zeichen eine Zahl ist, wie:

```
IF (c>='0') AND (c< ='9') THEN . . .
```

läßt sich in PASCAL viel einfacher durchführen. Es muß nur festgestellt werden, ob c Element eines bestimmten Zeichensatzes ist:

```
IF c IN ('0'..'9') THEN . . .
```

So könnte der Anfang eines Kartenspielprogramms folgendermaßen aussehen:

```
TYPE
  Rang = ( Zwei, Drei, Vier, Fuenf, Sechs,
           Sieben, Acht, Neun, Zehn, Bube,
           Dame, Koenig, As);
  Kartenspiel : SET OF Rang:
VAR
  Karte : Rang;
  Bilder : Kartenspiel;
BEGIN
  Bilder := (Bube. . As);
  IF Karte IN Bilder THEN (* etc. *)
```

Vergleichen Sie diese Anordnung mit:

```
IF (Karte = Bube) OR (Karte = Dame) OR
  (Karte = Koenig) OR . .
```

Ebenfalls definiert sind Operationen mit vollständigen Sets: Vereinigungsmenge, Mengendifferenz und Schnittmenge. Wenn B beispielsweise der Set aller BASIC-Programmierer ist und P der aller PASCAL-Programmierer, dann ist die Schnittmenge beider Sets der Set aller Programmierer, die sowohl BASIC als auch PASCAL beherrschen. Die Vereinigungsmenge von P und B ist der Set aller Menschen, die entweder in PASCAL oder in BASIC programmieren – also die Kombination beider Sets.

Eine Mengendifferenz ergibt sich aus der Subtraktion aller Elemente eines Sets von einem anderen. P–B stellt daher alle Personen dar, die in PASCAL programmieren, aber nicht

in BASIC. Die Vereinigungsmenge wird als P+B geschrieben und die Mengendifferenz als P*B. Die Symbole der Set-Operatoren gleichen zwar den bekannten arithmetischen Operatoren, die unterschiedlichen Vorgänge sollten jedoch nicht durcheinandergebracht werden. Da diese Symbole den Vorgang des Bit-Testens darstellen, werden sie auch für Sets eingesetzt.

Stellen Sie sich ein Set mit acht Elementen vor. Dabei wird die Anwesenheit eines Elementes durch das Setzen eines bestimmten Bits (Eins) in einem Byte angezeigt, seine Abwesenheit dagegen durch Null. Mit einer Maske (oder einem Bit-Test) kann nun leicht festgestellt werden, ob ein Byte Element des Sets ist. OR ergibt die Vereinigungsmenge und AND die Durchschnittsmenge. Da diese Abläufe auf Maschinenebene vorhanden sind, kann der PASCAL-Compiler Sets und ihre Operationen direkt in den Maschinencode umsetzen. Dafür wird nur wenig Speicher benötigt. Daher gehören Set-Operationen zu den effektivsten Abläufen in PASCAL.

Behandlung der Operator

Die Prioritätsregeln der PASCAL-Operatoren lassen sich auf vier Ebenen zusammenfassen, wobei die Operatoren, auf die wir nicht ausführlich eingegangen sind, dem in anderen Programmiersprachen üblichen Gebrauch entsprechen. Die „monadischen" (oder einwertigen) Vorzeichenoperatoren + und − und der logische Operator NOT haben erste Priorität. Auf der zweiten Prioritätsebene befinden sich alle Multiplikationsoperatoren (darunter auch die Divisionssymbole). Die dritte Ebene enthält Addition und Subtraktion, und auf der letzten Ebene werden alle Vergleichsoperationen − darunter auch IN − durchgeführt.

Beachten Sie, daß die beiden logischen Operatoren AND und OR wie Multiplikations- bzw. Additionsoperatoren behandelt werden und damit die Boolesche Algebra korrekt umsetzen. Viele Vergleichstests müssen die Prioritäten daher mit Klammern umgehen. So erzeugt etwa IF N>0 AND N< 10 THEN . . . eine Fehlermeldung, da der Ausdruck 0 AND N zuerst bewertet wird und als Versuch gilt, zwei ganzzahlige Operanden mit einem logischen Operator zu verknüpfen.

Alle Operatoren der gleichen Priorität werden von links nach rechts abgearbeitet. Das Zuordnungssymbol (:=) hat eine geringere Priorität als die anderen Operatoren, da der Ausdruck auf der rechten Seite einer Zuweisung vor der Zuordnung erst vollständig bewertet werden muß. Noch ein Wort der Vorsicht: Gehen Sie niemals davon aus, daß ein Ausdruck nicht bewertet wird. Für den Fall N=0 läßt IF (N>0) AND K N<19) THEN . . . das Programm absturzen, da eine Teilung durch Null nicht möglich ist.

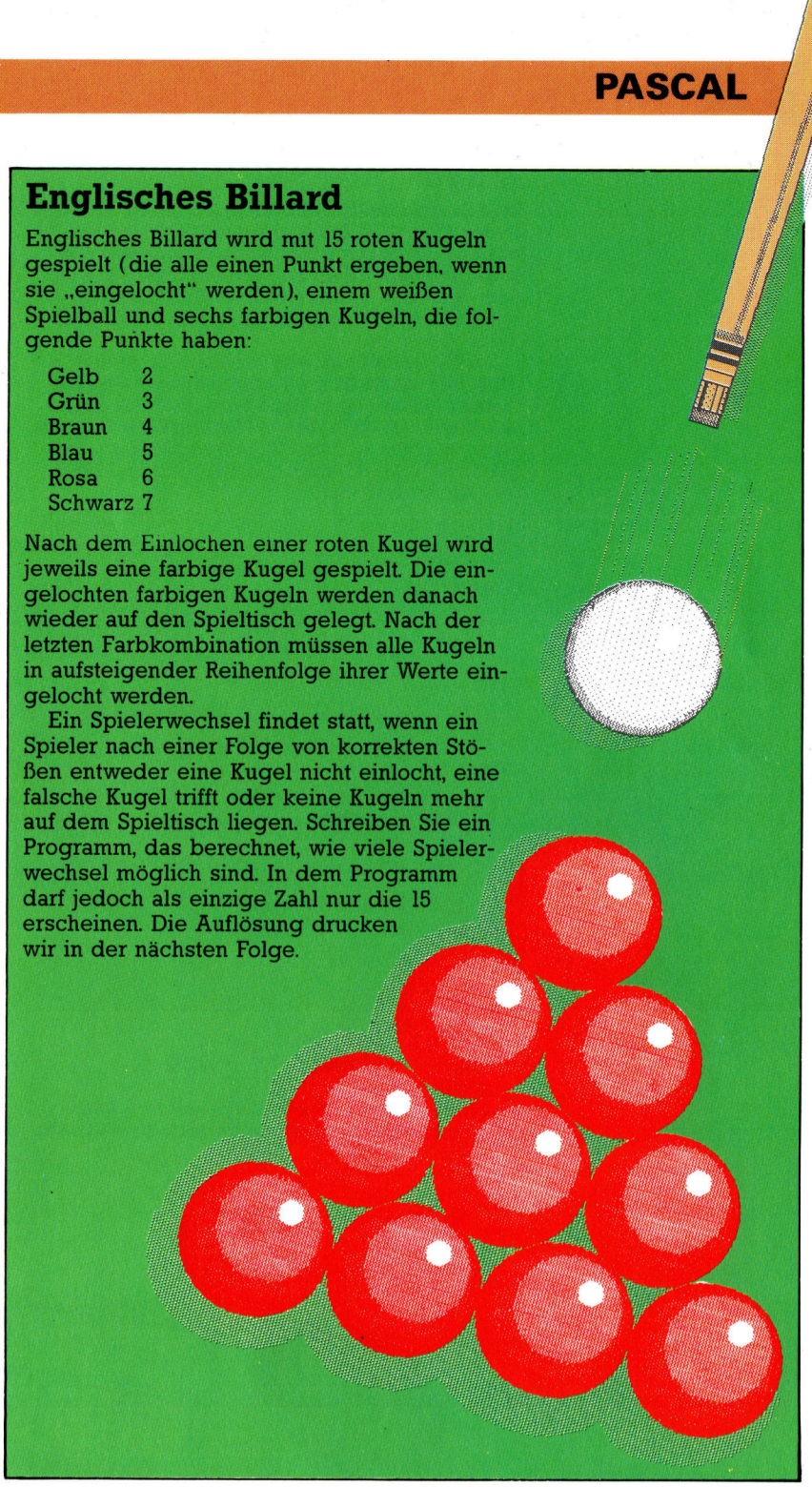
Englisches Billard

Englisches Billard wird mit 15 roten Kugeln gespielt (die alle einen Punkt ergeben, wenn sie „eingelocht" werden), einem weißen Spielball und sechs farbigen Kugeln, die folgende Punkte haben:

Gelb	2
Grün	3
Braun	4
Blau	5
Rosa	6
Schwarz	7

Nach dem Einlochen einer roten Kugel wird jeweils eine farbige Kugel gespielt. Die eingelochten farbigen Kugeln werden danach wieder auf den Spieltisch gelegt. Nach der letzten Farbkombination müssen alle Kugeln in aufsteigender Reihenfolge ihrer Werte eingelocht werden.

Ein Spielerwechsel findet statt, wenn ein Spieler nach einer Folge von korrekten Stößen entweder eine Kugel nicht einlocht, eine falsche Kugel trifft oder keine Kugeln mehr auf dem Spieltisch liegen. Schreiben Sie ein Programm, das berechnet, wie viele Spielerwechsel möglich sind. In dem Programm darf jedoch als einzige Zahl nur die 15 erscheinen. Die Auflösung drucken wir in der nächsten Folge.



Prioritätsregeln

Die Tabelle zeigt die Prioritäten der PASCAL-Operatoren. In runde Klammern eingeschlossene Ausdrücke werden separat bewertet und können so die normalen Prioritäten umgehen.

Priorität	Operator	Typ
Höchste	NOT + −	monadisch
	AND * / DIV MOD	Multiplikation
	OR + −	Addition
Niedrigste	< <= <> >=>IN	Vergleich

Bingo!

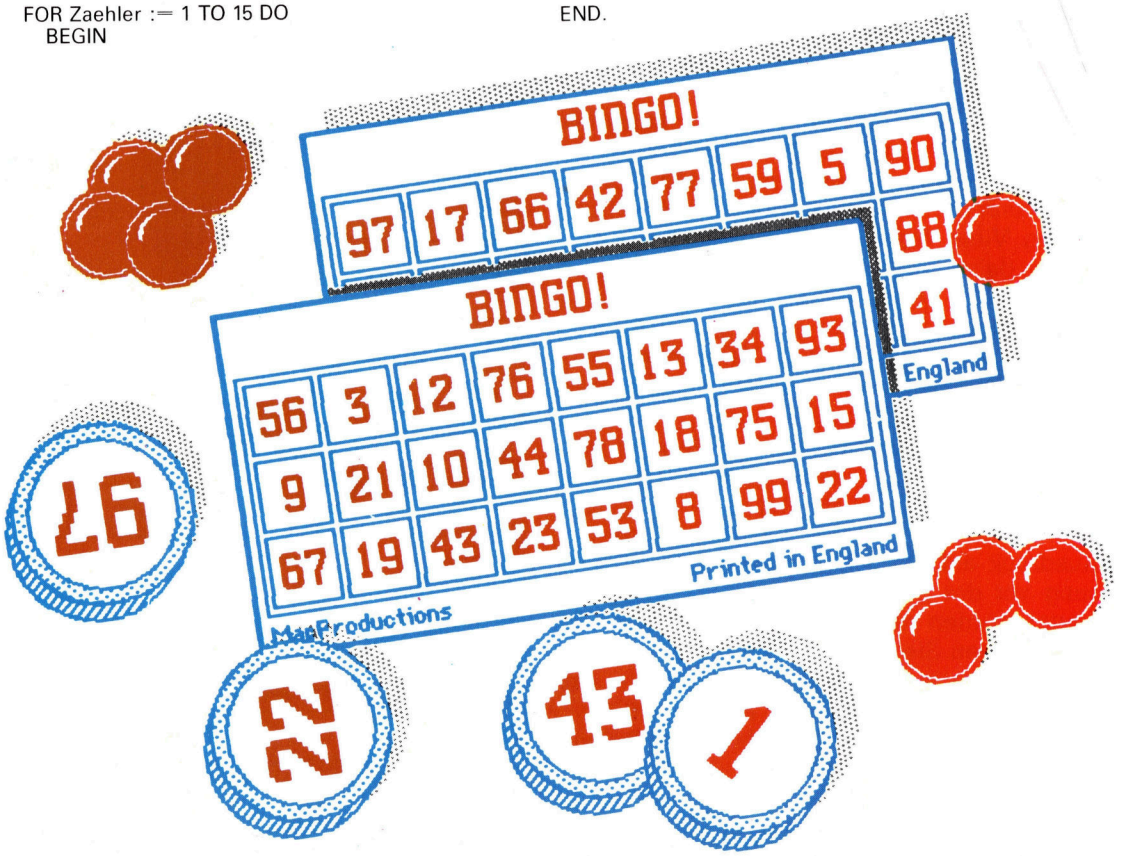
Eine Bingokarte läßt sich leicht mit einem Set darstellen. Obwohl die Grundelemente (Ganzzahlen von Eins bis 90) eine Struktur haben, ist hier das einzig wichtige Kriterium, ob die Nummer auf der Karte enthalten ist oder nicht. In PASCAL hieße das: „Nummer IN Karte" ist entweder „true" oder „false".

Das folgende Programm simuliert ein Bingospiel. Es liest zuerst die Kartennummern über die Tastatur ein und testet dann bei Aufruf der einzelnen Nummern, ob in diesen Nummern der Unterset Karte enthalten ist. Der Aufruf wird zur Leermenge, wenn sich jedes Element von Karte ebenfalls im Set „Aufruf" befindet.

Wenn zu einem Set ein Element hinzugefügt wird, das darin bereits enthalten ist, oder ein Element herausgenommen wird, das es darin nicht gibt, bleibt dieser Set unverändert. Beachten Sie, daß sich Elemente nicht direkt in einen Set einfügen lassen. Durch den Aufbau eines zusätzlichen Sets, der nur ein Element enthält, kann jedoch die Vereinigungsmenge dieser beiden Sets hergestellt werden.

Das Programm nimmt doppelte Nummerneingaben an und erzeugt einen Ablauffehler, wenn Zahlen außerhalb von Eins bis 90 eingegeben werden. Fügen Sie als Übung Schleifen ein, die dies verhindern und die die Eingabe bereits aufgerufener Nummern ablehnen.

```
PROGRAM                          Bingo (input, output);

CONST

                        Spalten = 40; (* Monitoranzeige *)
                        Haelfte = 25; (* Monitoranzeige *)
TYPE

                                 Bingo = SET OF 1 . . 90;

VAR
          Zaehler  : 1 . . 15;
          moeglich,
          Aufruf,
          leer,
          Karte    : Bingo;
          Nummer   : integer;
          Haus     : boolean;

BEGIN

  Leer := [ ] ;
  moeglich := [1 . . 90] ;
  WriteLn ( '*** BINGO ***' : Haelfte );
  WriteLn;
  WriteLn ( 'Geben Sie 15 Nummern der Karte ein,' );
  WriteLn ( 'Nach jeder Nummer bitte ein RETURN) :' );
  Karte := Leer;

  FOR Zaehler := 1 TO 15 DO
    BEGIN
      write ( Zaehler : 10, ': ? ' );
      ReadLn ( Nummer );
      Karte := Karte + [Nummer]
    END;

  WriteLn;
  WriteLn ( 'AUGEN NACH UNTEN !' : Haelfte );
  WriteLn;
  WriteLn ( 'Jetzt die Nummern aufrufen : ' );
  Aufruf := Leer;

REPEAT
  write ( '? ' : Haelfte );
  ReadLn ( Nummer );
  Aufruf := Aufruf + [Nummer] ;
  Haus := Karte — Aufruf = Leer

UNTIL Haus;

  WriteLn;
  WriteLn ( 'Glückwunsch !' : Haelfte );
  WriteLn ( 'Ihre Nummern sind :' );
  WriteLn;

  FOR Nummer := 1 TO 90 DO
    IF Nummer IN Karte THEN
      write ( Nummer : Spalten DIV 8 )

END.
```



Fachwörter von A bis Z

EAROM = EAROM
Der elektrisch veränderbare Festwertspeicher (Electrically Alterable Read-Only Memory) gehört zu den neueren Halbleiterspeichern. Seine Funktion ist am besten anhand einer Gegenüberstellung der ROM-Typen zu verstehen. Ein gewöhnliches ROM ist „maskenprogrammiert", das heißt, sein Inhalt wird beim Herstellungsprozeß durch die Belichtungsmaske festgelegt. Ein PROM (Programmierbares ROM) ist ein leeres ROM, das vom Benutzer mit einem PROM-Brenner beschrieben, aber nicht mehr geändert werden kann. Beim EPROM (Erasable = löschbares PROM) ist über dem Siliziumchip ein Quarzfenster angebracht, durch das sich der Speicher mit UV-Licht löschen und dann neu programmieren läßt.

Ein EEROM (Electrically Erasable = elektrisch löschbares ROM) funktioniert ähnlich wie ein EPROM, wird aber elektrisch und nicht mit UV-Licht gelöscht. Im Gegensatz zum EEROM können bei einem EAROM die Speicherplätze einzeln gelöscht bzw. überschrieben werden, was allerdings einigen Aufwand voraussetzt und wesentlich langsamer geht als das Lesen. Deshalb werden EAROMs vor allem dann verwendet, wenn Speicherinhalte geringfügig verändert werden müssen.

Edge Connector = Platinenstecker
Der Platinenstecker ist die einfachste Form des Rechneranschlusses und findet sich hauptsächlich bei Heimcomputern der unteren Preisklasse. Die Kontaktbahnen werden bei der Platinenherstellung im normalen Ätzverfahren auf einem etwa 1 cm überstehenden Randstreifen angelegt. Hochwertige Stecker haben vergoldete Kontakte, denn Kupfer oxidiert an der Luft, was Übergangsschwierigkeiten verursachen kann. Bei teureren Rechnern werden statt Platinensteckern spezielle Interfacebuchsen verwendet.

Hier werden einzelne Fachausdrücke eingehend behandelt. Da bei der Kommunikation mit dem Computer meist die englische Sprache verwendet wird, werden hier zunächst die englischen Begriffe genannt, dann die deutsche Übersetzung. In den Gesamtindex werden sowohl deutsche als auch englische Stichwörter aufgenommen, damit Sie es leichter haben, das von Ihnen Gesuchte zu finden.

Editor = Editor
Ein Editor ist ein Systemprogramm, das den Benutzer bei der Erzeugung und Abänderung von Zeichenfolgen unterstützt. Dabei kann es sich um Grafik-Symbole, BASIC-Anweisungen oder (bei der Textverarbeitung) um Sätze handeln. Zum Editieren benötigt man eine Reihe elementarer Funktionen wie Einfügen und Löschen von Zeichen, Überschreiben usw. Hochwertige Systeme bieten jede dieser Funktionen in verschiedenen Varianten – der Benutzer kann zum Beispiel Einzelbuchstaben, Wörter, Sätze, Abschnitte oder ganze Seiten löschen. Beim „bildschirmorientierten" Editor können Sie die Änderungen an jeder Stelle des Schirms durchführen, beim „zeilenorientierten" dagegen nur in der vorher definierten Zeile.

Electronic Mail = Elektronischer Briefkasten
Die Bürokommunikation entwickelt sich zu einem immer bedeutenderen Einsatzfeld für Microcomputer – zunehmend wird von der Datenübermittlung über Fernsprechleitungen Gebrauch gemacht. Ein verbreitetes Standard-System ist die elektronische Post, die den Austausch von Schriftstücken und Nachrichten über Rechner ermöglicht. Jeder Teilnehmer braucht neben dem Heimcomputer ein Modem und spezielle Software. Kurze Meldungen können über die Rechnertastatur direkt auf die Leitung gehen. Längere Schreiben speichert man üblicherweise auf Diskette oder Cassette ab und übergibt die Übertragung erst anschließend an ein Kommunikationsprogramm. Durch dieses Verfahren kann man Telefongebühren sparen.

Electrosensitive Printer = Elektrosensitiver Drucker
Die elektrosensitiven oder Thermo-Drucker sind billig in der Herstellung, arbeiten leise und ziemlich schnell. Ein Nachteil ist, daß sie nur mit teurem aluminiumbeschichtetem Spezialpapier funktionieren, auf dem die Zeichen oft schwer zu entziffern sind. Deshalb wird dieses Verfahren kaum für kommerzielle Anwendungen eingesetzt. Für den Ausdruck von Listings sind diese Geräte jedoch ausreichend – ein gutes Beispiel ist der ZX-Drucker von Sinclair, der mit Rollenpapier und nur 32 Zeichen pro Zeile arbeitet.

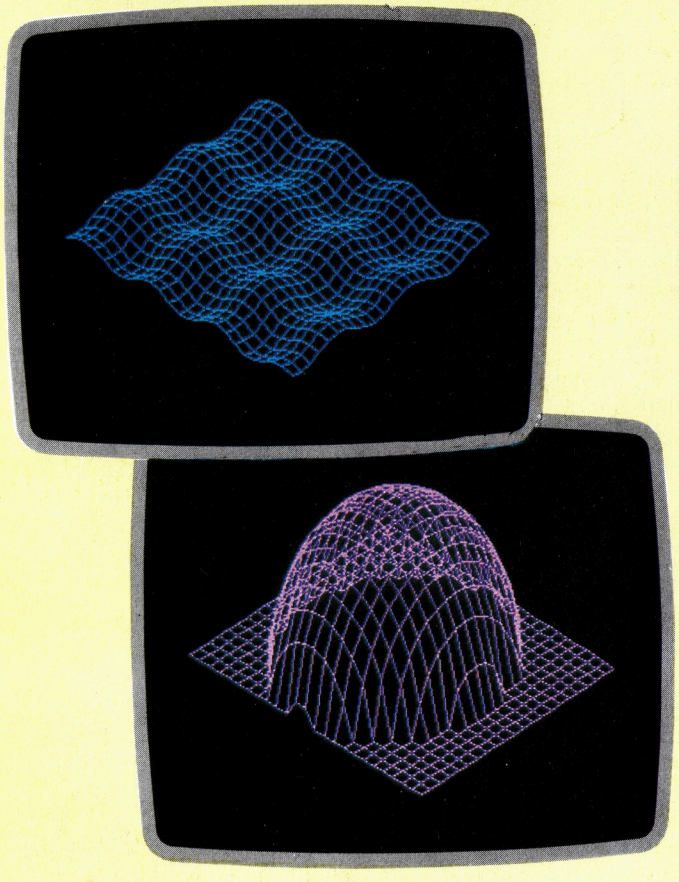
Die Zeichen werden punktweise aufgebaut. Der Druckkopf ist mit feinen Elektroden bestückt, die unmittelbar vor dem Papier stehen. An diese wird Spannung angelegt. Der resultierende Funke brennt in die Metallisierung auf der Schwarzpapier-Unterlage ein kleines Loch, das als dunkler Punkt erscheint.

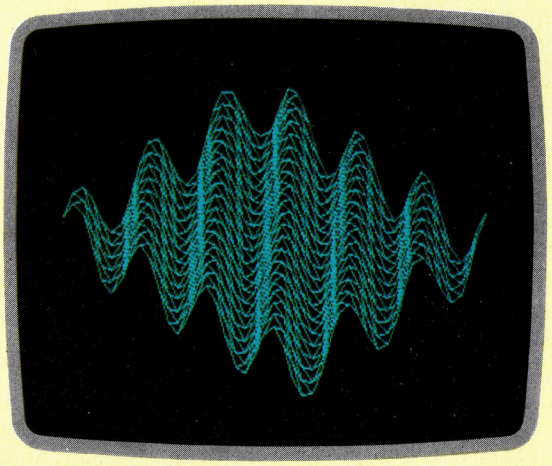
Beim Spectrum wird die zu editierende Listingzeile am unteren Bildschirmrand ausgegeben.

Bildnachweise
1037, 1038, 1051, 1054: Chris Stevens
1041: David Higham
1042: Liz Heaney
1043, 1044, 1045, 1050, 1053, 1060, 1061, 1063, 1064: Ian McKinnell
1058, 1059: Liz Dixon

computer kurs

Heft 39





Die Grafik-Fähigkeiten der meisten Heimcomputer sind durchaus gut genug, um mathematische Gleichungen darzustellen. Man kann sogar dreidimensionale Formen auf den Bildschirm bringen, wenn man verschiedene mathematische Funktionen in ein Programm integriert.

Comeback-Versuch

Ataris 130 XE ist die Antwort dieser Firma auf den rückläufigen Computer-Markt. Das Gerät weist 128 K RAM auf.

Digital/Analog-Wandler

Das User-Port-System soll um einen Analog/Digital-Wandler erweitert werden – zum Ansteuern analoger Geräte und zur Erzeugung mehrerer Effekte.

Kleine Scheibe

Die Schneider-Floppy wird mit einer Diskette geliefert, die drei Programme enthält.

Record-Befehl

Unsere Pascal-Folge beschäftigt sich diesmal mit dem „Record"-Befehl, der unterschiedliche Daten zusammenfaßt.

Leichtgewichte

Mit einem Handheld kann auch unterwegs „computert" werden. Eine Übersicht der wichtigsten Geräte.