Einsteigen - Verstehen - Beherrschen

DM 3,80 öS 30 sfr 3,80

computer kurs

Heft 76

Ein wöchentliches Sammelwerk

Unterricht am Rechner

Das Meßgerät wird fertig

Ein Kartenspiel

Unix für die Profis

Sonderteil Programmier-Service

computer kurs

Heft 76

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.

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

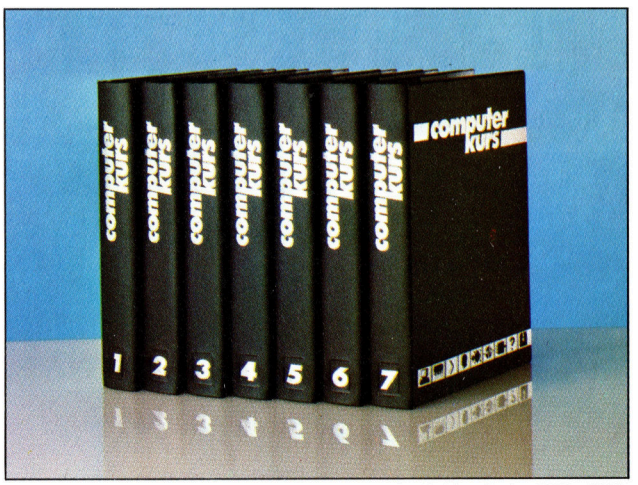
Schweiz: Der Sammelordner kostet sfr 15. Bitte wenden Sie sich an Ihren Kiosk; dort werden Sie jederzeit die gewünschten Exemplare erhalten.

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), Holger Neuhaus, Peter Aldick, 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

Trotz der Faszination moderner Technik steigt das Interesse am „reinen" Computerwissen nicht weiter an. Anstelle von Informationen über die technische Arbeitsweise von Rechnern fordern Schüler und Eltern zunehmend ein praxisbezogenes Lernen am und mit dem Computer. Die Maschine wird zum Werkzeug für andere Fähigkeiten wie Entwerfen, Planen und Berechnen.

Mehr lernen

Bis heute ist noch immer nicht eindeutig entschieden, ob Computer einen dauerhaften Platz im Schulleben bekommen sollen oder nicht. Möglich, daß Diskussionen darüber in Kürze von der Entwicklung überholt werden . . .

Noch vor weniger als 20 Jahren war der Einzug des Computers in die Schule kaum vorstellbar – Computer gehörten allein in den Bereich der Wissenschaft. Auch erste zaghafte Kontakte zwischen Schule und Computer veränderten diese Betrachtungsweise nicht. Kein Wunder – die massive Verbreitung des „Elektronengehirns" als Heimcomputer war noch nicht in Sicht. Der Computer blieb eben noch vornehmlich „Rechner".

Erst in den letzten Jahren wurde die Informatik ein eigenständiges, ernstgenommenes Fachgebiet, das sich vom Übergewicht der Mathematik, Elektrotechnik und Physik freischwimmen konnte. Trotzdem ist Informatik als Schulfach eine Seltenheit. Nur langsam dringt Computerkunde in die Schulen vor.

Fortschrittliche (und nicht selten computerbegeisterte) Lehrer machten oft den Anfang. Sie wollten ihren Schülern zumindest etwas über den Einfluß des Rechners auf den Arbeitsalltag vermitteln. Nach einem Zwischenhoch für die Vermittlung reiner Programmiertechnik läßt sich bereits heute ein Wandel der Lernziele absehen: Die Computertechnik (Logische Gatter, Halbaddierer etc.) tritt zurück. Mehr und mehr gewinnt die Betrachtung des Rechners als Hilfs- und Arbeitsmittel an Boden.

Ein Vorreiter des anwenderorientierten „Computerns" in der Schule sind nicht nur die USA, sondern auch England: Hier wurden durch Zusammenarbeit von Schulen und dem Erziehungsministeriums neue Lehrpläne entwickelt. Eine Neugestaltung des in England bereits sehr viel stärker verbreiteten Computer-Unterrichts soll jetzt die Problemlösung sein. In der Praxis heißt das, jeder Schüler soll mit Computerhilfe praktische Anwendungsfälle bewältigen. Dabei brauchen die Schüler jedoch nicht mehr jeden Arbeitsschritt selbst zu programmieren – wer will, darf auch die Verknüpfung eigener Programme mit fertigen Software-Paketen zur Lösung nutzen.

Auf höherem Niveau

Schulwissen über Computer wird an den Universitäten – jedenfalls in der BRD – heute noch selten vorausgesetzt. Auch ein Informatik-Student im Anfangssemester kann sich noch ohne Programmierkenntnisse in den Hörsaal wagen. Das könnte sich bei weiterem Vor-

dringen des Computers in das Erziehungswesen ändern. Allerdings sind auch Widerstände gegen die Ausbreitung der neuen Technologie gerade in Deutschland nicht zu unterschätzen. Viele Lehrer und noch mehr Eltern stehen dem Computer skeptisch gegenüber.

Es gibt speziell in der Elternschaft gegenläufige Tendenzen: Eine große Gruppe geht davon aus, daß Computer „das" Medium der Zukunft sein werden. Diese Eltern wünschen sich für ihre schulpflichtigen Kinder eine angemessene Vorbereitung auf die neue Technologie. Auf der anderen Seite steht die Furcht vor der Mechanisierung und Technisierung auch des Schullebens, die eine allzu einseitige Ausrichtung der Jugendlichen begünstigen könnte.

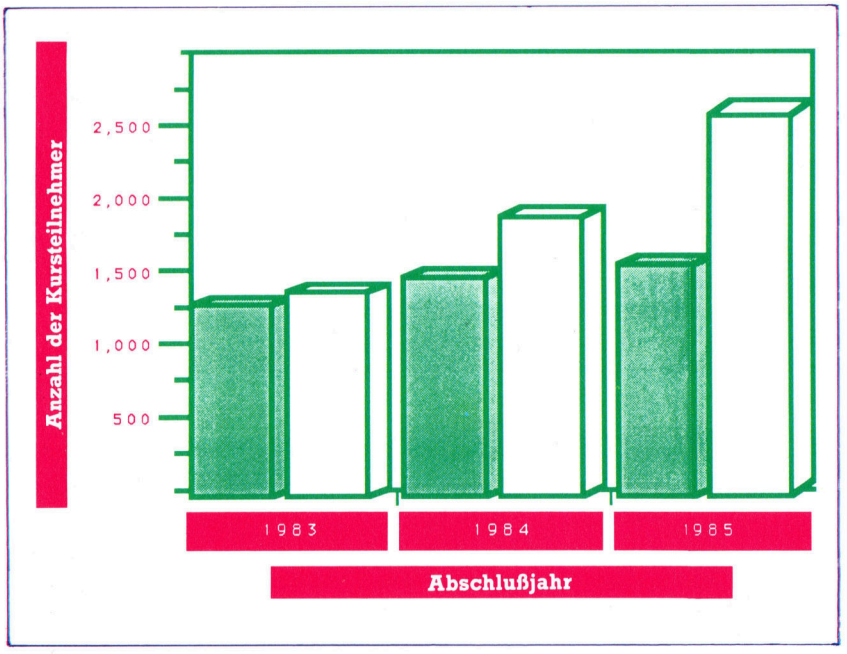
Aber auch hier scheint Abhilfe in Sicht: Im Ausland machen Schulen bereits den Versuch, die Kluft zwischen dem ausufernden privaten Gebrauch von Heimcomputern und dem in der Schule vorherrschenden wissenschaftlichen Computereinsatz zu verkleinern. Dazu sollen Datentechnik-Kurse beitragen, die das Wissen über die Informationsverarbeitung allgemeiner behandeln und dadurch neue Brücken schlagen.

Informationen nutzen

Im Idealfall sollten zukünftige (schulische) Datentechnik-Kurse ein Grundwissen über die Methoden der Datenerhebung, der Informations-Speicherung, der Datenverarbeitung und Informationsvermittlung erzeugen. Dabei müssen jedoch alle Arten von Informationen (sprachliche, schriftliche, visuelle, numerische) eingeschlossen werden. Beim Lernen kann dann auch der Computer eingesetzt werden, der Umgang mit Datenbanken, Textsystemen oder Grafiksoftware sollte Mittel zum Zweck sein. Schließlich dürfte der Computeralltag für die meisten Schüler in ihrer beruflichen Zukunft auch nicht den Umgang mit Lötkolben und Source-Codes beinhalten. Eher werden die Rechner integraler Bestandteil diverser Hilfsmittel sein.

Die bei diesem Vorgehen erlernten Strategien lassen sich auf andere Problematiken und Situationen übertragen. Das neue Wissensgebiet könnte sich zum gleichberechtigten Partner von Mathematik oder Deutsch entwickeln. Mit Erscheinen immer leistungsfähigerer Software könnten dann auch jüngere Schüler an die Informationstechnik herangeführt werden. Ein Fernziel sollte dabei festgehalten werden: den Einsatz des Computers nicht mehr als notwendiges, unvermeidliches Übel zu sehen, sondern als Werkzeug für Schüler, Lehrer und Verwaltung.

Die Zahlen zur untenstehenden Statistik stammen aus England, wo es Computer-Unterricht als Wahlfach sowohl in der Mittel- als auch in der Oberstufe gibt. Der Zuwachs an Kursteilnehmern in der Mittelstufe war dabei erheblich höher als die Zunahme in der Oberstufe, wo der Unterricht spezialisiert ist. Das untermauert die These, daß Computerwissen zwar als nützliche Ergänzung gesehen wird, als Schwerpunktfach jedoch an Bedeutung verliert.



Was soll vermittelt werden?

Empfehlungen für Lehrpläne

Fähigkeit zur Darstellung
- Darstellung und Verständnis von Lösungswegen
- Nutzen und Inhalt angemessener Dokumentation

Computer-Grundwissen
- Hardware (CPU, Speicher, Funktionsweise von Ein- und Ausgabegeräten)
- Computer-Architektur (Register, Zyklus Holen/Ausführen)
- Verarbeitung von Daten (Zahlen, Zeichen, Befehle)
- Software (maschinenorientierte und höhere Programmiersprachen, Compiler, Fehler und Fehlersuch-Algorithmen, Betriebssysteme, Anwenderprogramme)

Informationsverarbeitung
- Grundzüge der Systemanalyse
- Daten sammeln, bewerten, übertragen
- Fehlervermeidungs-Strategien
- Verschiedene Arten von Files und ihre Gestaltung
- Anforderungen an ein- und ausgegebene Informationen
- Flußdiagramme von anwenderorientierten Systemen
- Methoden zur Datensicherung
- Verarbeitungsmethoden (Batch, Echtzeit etc.)
- Auswahl des richtigen Computers für gegebene Anwendungen.
- Zwischenmenschliche Folgen (Privatsphäre, Datenschutz)

Problemlösung mit dem Computer
- Fähigkeit zur Problemdefinition, Datenermittlung und dem geeigneten Einsatz des Rechners zu ihrer Verarbeitung und Darstellung

Das Multitalent

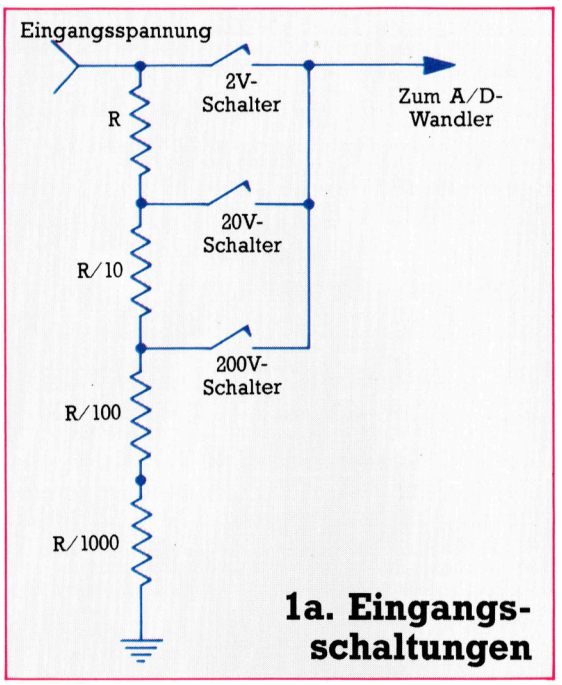
Wir stellen Schaltungen vor, die unserem Multimeter zusätzliche Fähigkeiten und Vielseitigkeit verleihen.

Im jetzigen Zustand können wir mit unserem Multimeter Spannungen zwischen 0,0000 und 1,9999 Volt messen. Durch Herunterteilen der Eingangsspannung läßt sich dieser Bereich bis auf 19999 Volt (20 kV) erhöhen. Eine solche Spannung ist jedoch extrem gefährlich! Wir wollen daher den Maximalwert auf 199,99 Volt begrenzen.

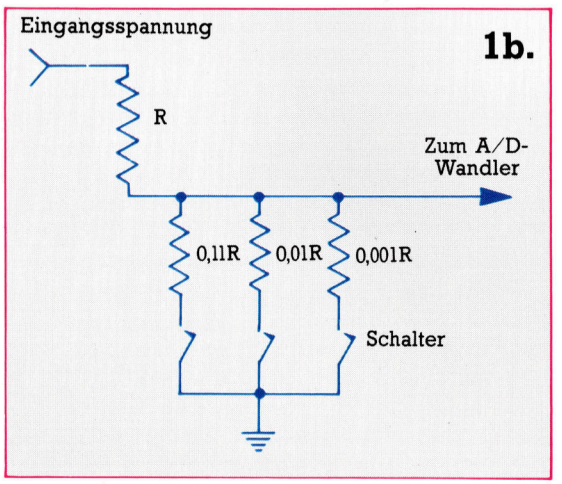
Sie werden bemerkt haben, daß auf der Zeichnung im letzten Abschnitt je drei Leitungen von drei LEDs nicht angeschlossen waren. Sie sind zum Ein- und Ausschalten der Dezimalpunkte rechts neben den Ziffern vorgesehen. Mit einem dreipoligen Schalter, der mit der Eingangsschaltung gekoppelt ist, können wir sicherstellen, daß der für den jeweils gewählten Meßbereich korrekte Punkt leuchtet.

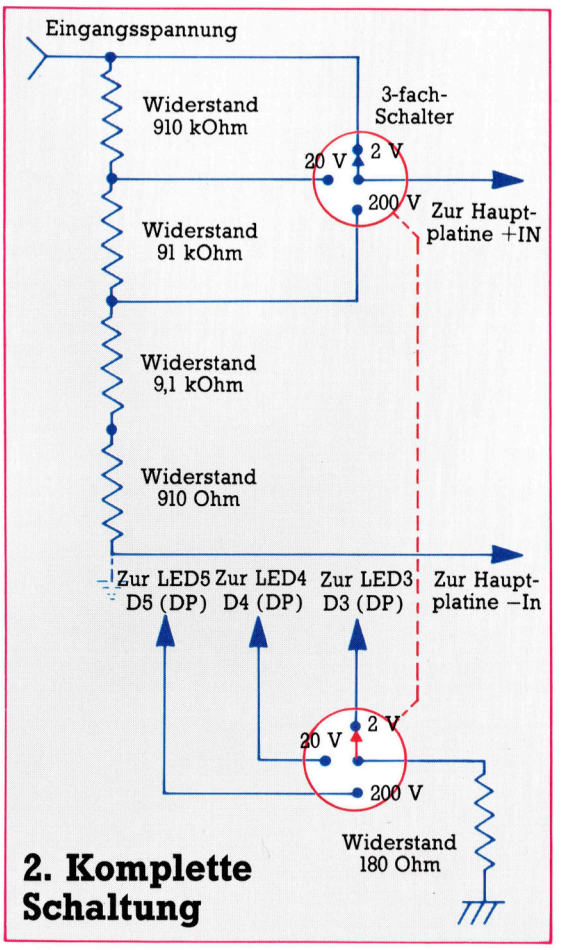
Grundsätzlich gibt es zwei verschiedene Techniken für die Eingangsschaltung. Beide nutzen ein Spannungsteiler-Netzwerk, das die Gesamtspannung über Widerstände so verteilt, daß nur ein Bruchteil der angeschlossenen Spannung vom Meßgerät abgegriffen wird. Die Schaltungsvarianten haben wir in Bild 1 dargestellt. Typ a ist besonders einfach und läßt sich zudem leicht für die Messung von Strömen und Widerständen abwandeln. Auf die zweite Schaltung trifft das nicht zu, sie könnte aber für Meßgeräte mit automatischer Bereichswahl eingesetzt werden, da sie mit elektronischen Schaltern arbeitet. Version a braucht mechanische Schalter. Wegen der leichteren Anpaßbarkeit ziehen wir die zweite Version jedoch vor.

Um den Meßbereich unseres Digitalvoltmeters zu vergrößern, benötigen wir eine Abschwächer-Schaltung. Damit kann die Eingangsspannung auf die für das Gerät verträglichen Werte zwischen 0 und 2 Volt reduziert werden. Beide Schaltungsvarianten arbeiten mit Widerstandsnetzwerken, an denen eine Teilspannung abgegriffen wird. Wir halten uns an die obere Schaltung – Schaltung b erfordert besondere Schutzmaßnahmen, könnte dafür aber auch mit elektronischen anstelle von mechanischen Umschaltern bestückt werden.



1a. Eingangsschaltungen





2. Komplette Schaltung

Die Schaltung wird auf einem kleinen Stück Lochplatine aufgebaut. Alle Widerstände sollten eine Genauigkeit von 1% (Präzisionswiderstände) aufweisen. Sie werden in einer Reihe auf die Platine gelötet, die Anschlüsse des 910-kOhm-Widerstandes sowie ein Anschluß des 91kOhm-Widerstandes mit einem doppelten 3 x UM-Schalter verbunden. Plus- und Minusausgang schließen Sie mit kurzen Stücken isolierten Drahtes an der Hauptplatine an. Die Anschlüsse auf der zweiten Ebene des Schalters werden mit den Dezimalpunkt-Anschlüssen der Siebensegment-LEDs 3,4 und 5 verbunden. Dadurch erscheint beim Umschalten in einen anderen Meßbereich der Dezimalpunkt korrekt.

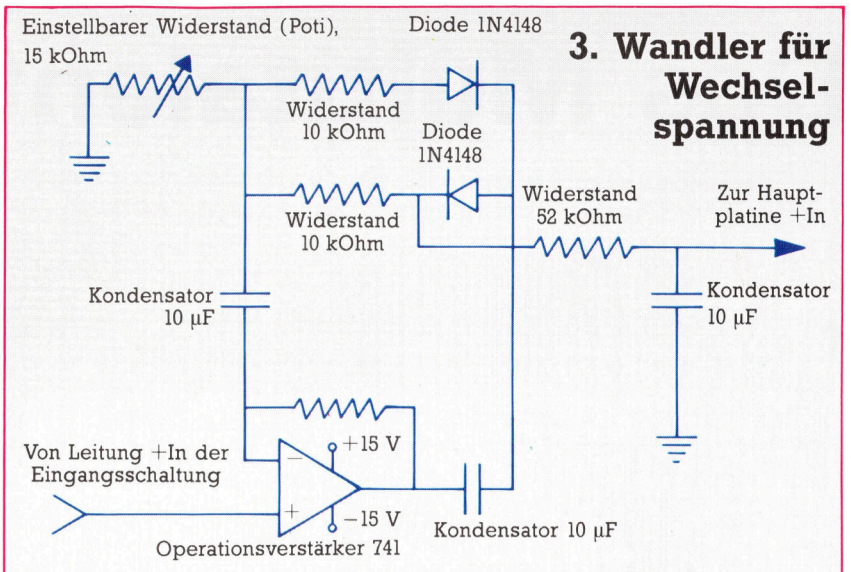
Die Schaltung nach Bild 2 wird zusammen mit der Dezimalpunkt-Steuerung auf einem Stück Lochplatine aufgebaut. Als Widerstände sollten Metallschicht-Typen mit einer Genauigkeit von mindestens einem Prozent verwendet werden. Man sollte bei der Schaltung darauf achten, die Leitungen kurz zu halten – die Eingangsimpedanz des A/D-Wandlerchips ist sehr hoch, so daß die Antennenwirkung langer Drähte zur Einstrahlung von Fremdsignalen führen könnte.

Messung in drei Bereichen

Die Abschwächerschaltung ist einfach aufgebaut und arbeitet mit Widerstandswerten aus der Normreihe nach DIN. Der Eingangswiderstand von über 1 MOhm führt zu einem ungefährlichen Stromfluß von weniger als 5 µA bei einer Spannung von 5 Volt.

Die vorgestellte Schaltung kann Gleichströme in drei Bereichen messen – 2 Volt, 20 Volt und 200 Volt. Mit einigen einfachen Zusätzen läßt sich die Schaltung aber auch für alle möglichen Sonderwünsche abwandeln, nicht nur für Widerstands- oder Wechselspannungsmessung, sondern auch für etwas Alltägliches wie Temperaturermittlung.

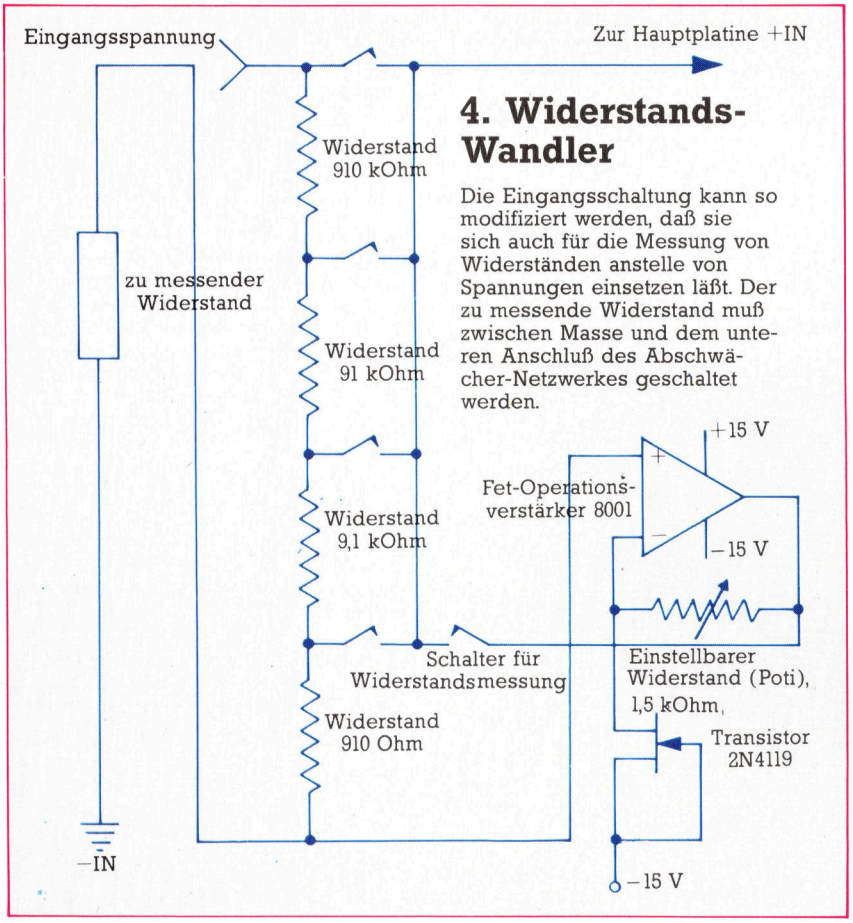
Für die Messung von Wechselspannungen mit dem DVM muß die Eingangs-Wechselspannung gleichgerichtet werden. Das geht allerdings nicht mit einem einfachen Diodengleichrichter; Sie brauchen dazu einen speziellen Meßgleichrichter. Eine mögliche Schaltung zeigt Bild 3. Sie arbeitet mit dem weit verbreiteten und preiswerten Operationsverstärker 741. Nachteilig ist, daß eine Spannungsversorgung mit +/– 15 Volt benötigt wird. Sie kann allerdings mit den 12 Volt des Netztrafos realisiert werden, da die Leistungsaufnahme des Op-Amps 741 sehr niedrig ist.



Wechselspannungs-Wandler
Das Digitalvoltmeter muß zum Messen von Wechselspannungen mit dieser Zusatzschaltung ausgestattet werden. Die Schaltung wird in die Verbindungsleitung zwischen der Eingangsschaltung und der Hauptplatine (Anschluß +IN) eingeschleift.

Die Widerstandsmessung mit einem konventionellen Analog-Voltmeter ist simpel, weil der Strom durch einen ohmschen Widerstand linear mit dem Widerstandswert verknüpft ist. Bei einem digitalen Voltmeter ist der Vorgang etwas komplizierter; wir benötigen eine Ohmmeter-Eingangsschaltung. Der Widerstandswandler nach Bild 4 speist eine Konstantspannung in die Eingangsschaltung, wodurch im Widerstand ein konstanter Strom fließt. Ein Konstantstrom über einen unbekannten Widerstand erzeugt einen zum Widerstandswert proportionalen Spannungsabfall. Dieser Spannungsabfall kann direkt über das Digitalvoltmeter gemessen werden. Als Konstantstromquelle arbeitet in dieser Schaltung ein Operationsverstärker mit sehr extrem hoher Eingangsimpedanz.

Durch Verbindung der Widerstands-Zusatzschaltung mit dem Netzwerk des Eingangs-Abschwächers lassen sich vier verschiedene Meßbereiche wählen: 2 kOhm, 20 kOhm, 200 kOhm und 2 MOhm. (Deswegen liegen unten im Netzwerk zwei Widerstände – bei nur drei Meßbereichen wären wir mit einem einzigen 10 kOhm-Widerstand ausgekommen.) Natürlich benötigen wir hierfür einen Schalter mit vier Stellungen.



Durch die Röhre

Unter MS-DOS lassen sich Software und Daten logisch organisieren, wenn man die Dateien je nach Inhalt auf unterschiedliche Ebenen der hierarchischen Sub-Directories stellt. In dieser Folge sehen wir uns einige Varianten dieser Technik an.

Die Unix-Dienstmodule pwd (Print Working Directory – Aktuelles Directory drucken) und tree (Directorystruktur anzeigen) sind nicht immer im Lieferumfang von MS-DOS enthalten. Es gibt aber auch andere Möglichkeiten, zum Erfolg zu kommen. Wenn Sie bei Aufruf von prompt die Zeichen $p angeben, erscheint der volle Pfadname (vom Rootdirectory des aktuellen Laufwerks aus) als Teil des DOS-Prompts, das nach jeder Programmbeendigung angezeigt wird.

Nehmen Sie an, Sie arbeiten in dem Sub-Directory TV (Text-Verarbeitung) und möchten eine Diskette formatieren, auf die Sie das gerade erstellte Schriftstück kopieren wollen. Die Eingabe format b: hat keine Wirkung, da das Diskettenmodul FORMAT.EXE sich in einem Sub-Directory namens system befindet. Nur mit dem vollen Pfadnamen \system\format b: erreichen Sie Ihr Ziel. Bei Aufruf des Programms WCOUNT.EXE (Wortzähler), das sich auf Laufwerk C in einem Sub-Directory befindet, ist die Eingabe noch umständlicher:

```
C:\text\tools\wcount report17.doc
```

Glücklicherweise gibt es eine Abkürzung. Mit dem integrierten Befehl path können Sie Pfade vorgeben, die MS-DOS immer dann durchsucht, wenn ein Befehl nicht im aktuellen Directory steht. Ohne Parameter zeigt path bestehende Pfade an oder gibt „No Path" aus. Wenn sich alle oft eingesetzten Programme und Dienstmodule in den aufgeführten Directories befinden, brauchen Sie nur

```
path=A:\system;C:\text\tools
```

einzugeben, um sie von jedem aktuellen Directory aus in den Sub-Directories tools und system erreichen zu können. Die Suche folgt der angegebenen Reihenfolge, bei namensgleichen Programmen wird die zuerst gefundene Übereinstimmung aufgerufen.

Zuvor noch eine Warnung: Die Argumentenliste darf keine Leerzeichen enthalten, weil sie als Endzeichen interpretiert werden.

```
pathC:\system;A:\mylib;C:\text\tools
```

würde daher nichts finden, was nicht im aktuellen Directory oder im Systemdirectory des Laufwerks C steht.

Die von UNIX beeinflußten MS-DOS-Versionen (von 2.0 aufwärts) können die Ein- und Ausgabe der Systemgeräte und Diskettendateien „umlenken". Feste Systemkomponenten wie CON (Tastatur) und PRN (Drucker) werden dabei wie Diskettendateien behandelt. Der Befehl type zeigt den Inhalt einer Textdatei normalerweise auf dem Bildschirm an. Mit „>" läßt sich die Ausgabe jedoch auf jedes beliebige Gerät (oder genauer gesagt: jede Datei) umlenken.

```
type report17.doc>prn
```

Weitere Befehle von MS-DOS

Hier weitere MS-DOS Befehle, die in COMMAND.COM – die MS-DOS Version des CP/M-Befehlszeileninterpreters (CCP) – eingebaut sind:

Befehl	Funktion
cls	Bildschirm löschen
prompt	Systemprompt ändern
pwd	Aktuelles Directory anzeigen
re (rmdir)	(leeres) Directory löschen
ver	Versionsnummer anzeigen
vol	Name von Volume anzeigen

Nicht jedes System enthält all diese Befehle. Beispielsweise ist das (von Unix entlehnte) ped nicht unbedingt notwendig, da der Befehl prompt mit $p den aktuellen Pfad anzeigt:

```
prompt $p
```

Auch andere Zeichen haben in Verbindung mit dem $ spezielle Bedeutungen:

d	Datum anzeigen
t	Zeit anzeigen
—	Mit CR/LF eine neue Zeile anfangen
e	ein Escape-Zeichen senden

Mit den Escapesequenzen lassen sich die Bildschirmeigenschaften ändern:

```
prompt $t on $d $_$e[7m $p $e[m
```

zeigt in einer Zeile Zeit und Datum an und in der nächsten den aktuellen Pfad in Negativdarstellung:

```
15:42:36 on 17—05—86
A:\SYSTEM\UTILS
```

initialisiert den Drucker und gibt die Datei aus. Der Befehl

```
dir a:>c:nov22.dir
```

bringt den Inhalt des Directories nicht wie üblich auf den Bildschirm, sondern schreibt ihn in die Datei nov22.dir auf Laufwerk C. Mit der Ausgabeumlenkung können Programme oder Dienstmodule ihre Ausgaben zu jedem Gerät oder jeder Datei des Systems senden.

Doppelte spitze Klammern (>>) lösen die gleiche Ausgabeumlenkung aus, überschreiben den Dateiinhalt jedoch nicht. Nach

```
dir b:>>c:nov22.dir
```

enthält nov22.dir eine Liste der Dateien aller Subdirectories und aktuellen Directories von Laufwerk A und B.

Schon für sich allein genommen ist die Umlenkung recht praktisch, zusammen mit etlichen einfachen „Softwarewerkzeugen" wird MS-DOS durch die E/A-Umlenkung fast schon zu einer interaktiven Programmiersprache.

Die praktischsten Softwarewerkzeuge sind einfache Programme, die die über übliche Eingabekanäle gelesenen Daten verändern, und sie dann an die normalen Ausgabekanäle übergeben. Module dieser Art werden „Filter" genannt. Sie lassen sich recht einfach schreiben, müssen vor ihrer Ausführung jedoch compiliert werden, damit sie – wie alle Systembefehle – unabhängig von einem Interpreter oder dem Quellentext funktionieren.

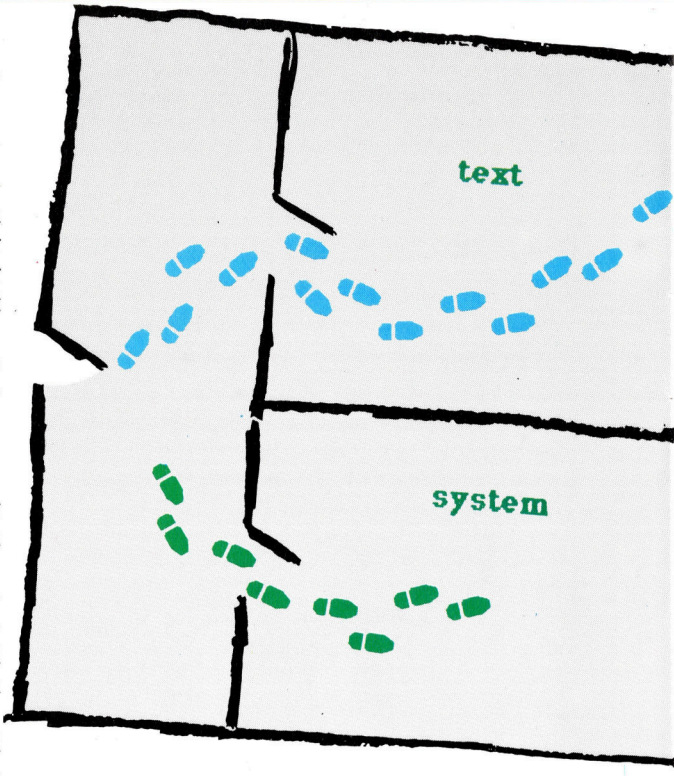
Zeichenbearbeitung

Unix oder die MS-DOS-Versionen 2/3 können mit den Zeichenbearbeitungsroutinen der normalen Ein- und Ausgabegeräte auch Dateien zusammenstellen. Die Eingabeumkehrung geschieht mit „<". Ein Programm kann dabei Eingaben, die normalerweise über die Tastatur kommen, aus einer Datei erhalten. So nimmt beispielsweise die Routine

```
date<date.now
```

ein als String gespeichertes Datum aus der Datei date.now entgegen, wobei die übliche Ausgabemeldung auf dem Bildschirm erscheint. So etwas bewährt sich besonders bei softwaregesteuerten Uhren, die bei jedem Booten des Systems neu gestellt werden müssen. Wenn Sie die Zeile in die Datei autoexec.bat einfügen, brauchen Sie das Datum nicht ständig neu einzugeben.

Einige praktische Filter werden gleich als Dienstmodule von MS-DOS mitgeliefert. So nimmt MORE.COM einen beliebigen Eingabetext und kopiert ihn in die Standardausgabe. Nach jeder dreiundzwanzigsten Zeile zeigt MORE die Meldung -- More -- an und wartet auf einen Tastendruck. Auf diese Weise läßt sich Text bequem und übersichtlich bildschirmweise abrufen.




```
more<report.doc
```

Eines der nützlichsten Dienstmodule von MS-DOS ist SORT.EXE. Das Programm arbeitet als Filter, der Textzeilen einliest und sie vor der Ausgabe in alphabetischer Reihenfolge sortiert. Ein sortiertes Inhaltsverzeichnis aller Textdateien läßt sich leicht mit

```
dir *.txt>temp.dir
sort<temp.dir
del temp.dir
```

erstellen. Wie Unix lassen sich diese Vorgänge auch über sogenannte Röhren (Pipeli-

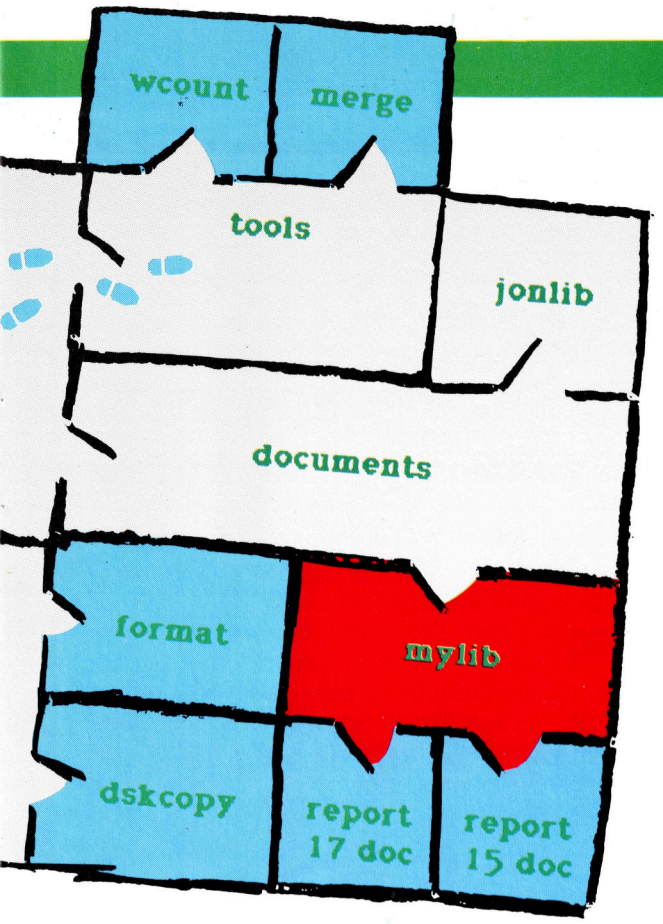
Gefilterte Dateien

„Filter" können einfache Programme sein, die Standardeingaben entgegennehmen und die Daten vor der Ausgabe ändern, erweitern oder löschen.

Mit WordStar erstellte Textdateien und bestimmte andere Programme enthalten oft Zeichen, deren höchstwertiges Bit für Formatierungszwecke gesetzt ist. Eine „saubere" Kopie ohne diese Zeichen läßt sich leicht durch ein Programm erzeugen, das mit der Umlenkung der Ein- und Ausgabekanäle arbeitet:

```
pipz<badfile.txt>goodfile.txt
```

Das folgende Programmbeispiel (in PASCAL) setzt alle Zeichen auf ihre normalen ASCII-Werte (bis 127) und führt damit den gleichen Ablauf aus wie die Option [z] des CP/M-Befehls PIP. Das Modul stellt die höchstwertigen Bits aller Zeichen auf Null. Da der MS-DOS-Befehl copy diese Möglichkeit nicht besitzt, ist dieses Modul besonders praktisch.



Die hierarchische Dateistruktur von MS-DOS läßt sich als Zimmerflucht darstellen, in der sich bestimmte Räume nur über bestimmte andere Räume betreten lassen. Die „Endräume" bezeichnen Dateien oder Programme und nicht die Directories (im Bild blau unterlegt). Nehmen Sie an, Sie arbeiten im Sub-Directory mylib an einem Text namens report17.doc. Dienstprogramme wie Wortzähler (wcount) und Module zur Diskettenformatierung „wohnen" in anderen Räumen. Damit Sie beim Aufruf eines Moduls nicht jedesmal den gesamten Pfad angeben müssen, können Sie mit dem MS-DOS-Befehl path einen Standardpfad festlegen.

nes) als ein Ablauf ausführen. Das Zeichen „|" zeigt an, daß die Ausgabe eines Vorgangs auf einen Kanal oder Strom umgelenkt werden soll, der dann als Eingabequelle des darauffolgenden Vorgangs dient. In diesem Fall genügt die Eingabe von:

```
dir *.txt|sort
```

Temporäre Dateien werden nach Ende des Vorgangs von MS-DOS automatisch gelöscht und sind für den Anwender unsichtbar.

Der Befehl sort läßt sich auch durch Optionen beeinflussen. So sortiert /r in umgekehrter Reihenfolge, während /+n (n kann jede beliebige Zahl sein) die Zeilenposition des Zeichens angibt, von dem an die Zeilen miteinander verglichen werden. Das MS-DOS-Directory gibt Auskunft über die Dateigröße in Bytes (die Information beginnt beim 15ten Zeichen) und Zeit und Datum, an dem die Datei angelegt wurde. Der folgende Befehl erzeugt den Ausdruck des Directories, das nach Dateigröße sortiert wurde:

```
dir|sort/r/+15>prn
```

Das Programm kann die Datei auch direkt an den Drucker weiterleiten:

```
pipz < badfile.txt > prn
```

Ein „Filterbeispiel" in PASCAL:

```
PROGRAMM PipZ (input, output);
{setzt chars auf ASCII-Werte zwischen 0 . .127}
VAR
  c: char;
BEGIN
  WHILE NOT EoF (input) DO
    BEGIN
      WHILE NOT EoLn (input) DO
        BEGIN
          read (input,c);
          write (output,
            chr (ord (c) MOD 128))
        END;
      Read Ln (input);
      Write Ln (output)
    END
END.
```

Flexible Filter

FIND.EXE ist ein weiterer flexibler DOS-Filter, mit dem Sie in Textdateien nach bestimmten Strings suchen können:

```
find"sort " C:\articles\MSDOS4.txt
```

Dabei wird jede Zeile, die den String „sort " enthält, zur Ausgabe gesandt (bei mehrfachem Vorkommen in einer Zeile wird die Zeile nur einmal angezeigt). Beachten Sie, daß durch das Leerzeichen hinter „sort " die Wörter sorting und sorted nicht gefunden werden. Auch Sort wird nicht gefunden, da find zwischen Groß- und Kleinschreibung unterscheidet.

Wenn Sie diese Unterscheidung nicht wünschen, können Sie leicht einen eigenen Filter schreiben (KLEIN.EXE), der alle Eingaben in Kleinbuchstaben umwandelt.

```
klein<meeting.doc|find "metal"|more
```

zeigt jede Zeile des Textes meeting.doc an, die die Buchstabenfolge metal enthält (metallisch, Metallurgie, Gourmetallüren – gleich, ob groß- oder kleingeschrieben) und unterbricht die Darstellung alle 23 Zeilen.

Die Flexibilität von Pipelines und E/A-Umleitungen zeigen Befehle wie

```
dir *.pas|find"-05-86"|sort>ptn
```

Hier werden alle im Mai angelegten Quellentexte von PASCAL (.pas) in alphabetischer Form ausgedruckt. find kann auch die Zeilennummern der Texte angeben (Option /n) oder die Ausgabe unterdrücken und nur die Gesamtzahl der Vorkommen anzeigen (Option /c). Die Option /v findet alle Zeilen, die den String nicht enthalten.

Nehmen Sie an, die Textdatei company.dat enthält von Spalte 25 an Namen und zugehörige Gehaltsdaten.

```
klein<company.dat|find"schmidt"/v|sort/
25+|more
```

zeigt alle Zeilen, die „Schmidt" nicht enthalten nach Gehalt sortiert an.

Sie sehen, daß sich mit „Röhren" und der E/A-Umlenkung schon auf Befehlsebene neue Programme erzeugen lassen, die sich aus einfachen Softwarewerkzeugen in logischer Folge zu nützlichen Routinen aufbauen.

Toolbox für alle Fälle

Machen Sie sich das Leben leichter – mit zusätzlichen „Werkzeugen" für den Computerumgang. Unser Maschinenprogramm bietet viele Befehle, mit denen Sie in Ihren BASIC-Programmen Ordnung schaffen können.

Zu Ihrer Arbeitserleichterung zeigen wir ein Toolkit-Programm für den Spectrum und den Commodore. Damit können Sie Ihren Rechner weit besser ausnutzen und schneller programmieren.

Das Programm wurde in Maschinensprache geschrieben, damit es parallel zu BASIC-Programmen laufen kann. In den folgenden Abschnitten finden Sie alle Informationen über das Laden, Speichern und die Anwendung dieser Arbeitshilfe.

Spectrum

Das Spectrum-Programm bietet acht Zusatzbefehle: Umnumerieren, Block löschen, Bytes frei?, Programmlänge?, automatische Zeilennumerierung, Inhaltsverzeichnis von Cassetten und Hex/Dezimal- bzw. Dezimal/Hex-Umwandlung. Alle Befehle werden mit RANDOMIZE USR gefolgt von einer Zahl aufgerufen. Nach Aufruf eines Befehls fragt das Programm nach den nötigen Eingaben, etwa der Anzahl zu löschender Zeilen oder der Zahl, die umgewandelt werden soll.

Das Programm benutzt verschiedene Routinen aus dem „Cross-Referencer", Sie müssen daher beide Programme miteinander verbinden. Hinweise dazu stehen im

Werkzeug-Programm selbst. Geben Sie das Programm also einfach ein, und folgen Sie nach dem Start den Anweisungen. Haben Sie in den DATA-Zeilen einen Fehler gemacht, meldet sich das Programm. Solche Fehler müssen Sie vor dem Abspeichern des Maschinenprogramms natürlich beseitigen. Der Cross-Referencer selbst kann eigenständig eingesetzt werden. Er durchsucht auf Tastendruck Programme nach vom Benutzer vorgegebenen Strings, Funktionen oder BASIC-Befehlen und bringt jede Programmzeile, die diese „Targets" enthält, auf den Schirm – eine Unterstützung, die beim Entwickeln und Debuggen längerer BASIC-Programme außerordentlich hilfreich ist. Als weiteren Vorteil bietet das in Maschinencode geschriebene Programm eine Replace-Option, die das automatische Austauschen ungeeigneter Strings, Variablen und BASIC-Befehle überall dort ermöglicht, wo sie im Programm auftauchen.

Stringsuche

Im Listing folgt einem kurzen BASIC-Lader der Code des Hauptprogramms. Nach dem Eintippen wird geSAVEd und RUN gedrückt. Bei korrekter Eingabe wird auch der Maschinencode gespeichert. Um den Crossreferencer zu benutzen, wird zunächst

CLEAR 64559
LOAD "CREF"CODE

eingegeben. Sodann stehen folgende Routinen zur Verfügung: Mit RANDOMIZE USR 64634 erscheinen alle Zeilen auf dem Schirm, die den vom User nach dem Prompt „Enter Target String" eingetippten String enthalten. Mit RANDOMIZE USR 64911 wird das Programm nach dem vom Benutzer vorgegebenen BASIC-Befehl durchkämmt, und RANDOMIZE USR 64713 sucht nach Zeilen, in denen DEFFN implementiert wurde. RANDOMIZE USR 64713 erlaubt den Austausch eines Strings gegen einen anderen. Target- und Replacementstring müssen gemeinsam eingegeben werden und dürfen jeweils nicht länger als meist ausreichende 20 Zeichen sein.

Das Toolkit-Programm speichert die Kombination von Werkzeug- und Cross-Referencer-Programm als „TOOLKIT"CODE. Wenn es beim Programmieren eingesetzt werden soll, geben Sie ein:

CLEAR 63488
LOAD "TOOLKIT" CODE

Der RENUMBER-Befehl (Zeilen neu numerieren) wird mit RANDOMIZE USR 63489 aufgerufen. Das Programm fragt dann nach dem gewünschten Zeilenabstand. (Minimum 1, Maximum 255.)

Zur Bestimmung einer Programmlänge dient:

RANDOMIZE USR 63889

Die Größe des freien Speicherbereichs gibt:

RANDOMIZE USR 63860

Mit RANDOMIZE USR 64154 aktivieren Sie die automatische Zeilennumerierung. Sie werden nach Anfangszeile (1–9900) und Zeilenabstand (ebenfalls 1–9900) gefragt. Um den Befehl zu annulieren, geben Sie nach dem Erscheinen einer Zeilennummer einfach 00 ein. Die darauf folgende „Nonsense in Basic"-Meldung können Sie ignorieren.

Um einen ganzen Block zu löschen, wird RANDOMIZE USR 64000 eingetippt. Sie müssen dann nur noch Anfangs- und End-Zeilennummer des Blocks angeben.

Das Inhaltsverzeichnis einer Cassette erhalten Sie mit RANDOMIZE USR 63919. Nach diesem Befehl beginnt der Rand des Bildschirms zu blinken. Bringen Sie die Cassette in Startposition und drücken Sie PLAY. Sie sehen dann die Überschriften Ihrer Datensätze.

Die Dezimal/Hex-Umwandlung erfolgt mit RANDOMIZE USR 64394, Hex/Dezimalwandlung durch RANDOMIZE USR 64453. Beim letzten Befehl brauchen Sie übrigens nach Eingabe der Hex-Zahl kein ENTER mehr einzugeben.

Commodore

Das Commodore-Programm stellt 43 neue Befehle zur Verfügung, die genau wie die vorhandenen BASIC-Anweisungen völlig problemlos eingesetzt werden können.

Das Programm wird eingetippt und gestartet. Bei Fehlern in den DATA-Zeilen wird die Nummer der falsch eingegebenen Zeile angezeigt. Ist alles in Ordnung, speichern Sie die BASIC-Version mit

SAVE "TOOLKIT"

Lassen Sie danach das Programm noch einmal laufen, und geben Sie SYS 52480 ein, um den Maschinen-Code zu erzeugen. Dieser kann gleich unter Verwendung eines der neuen Befehle blitzschnell abgespeichert werden:

@ MSAVE 49152,53247,
"MCTOOLKIT",1,1

Diese Maschinensprach-Version benötigen Sie, um mit den neuen Möglichkeiten zu arbeiten. Zum Laden dient:

LOAD "MCTOOLKIT",1,1

Wegen der Funktionsweise des Programms bestehen alle neuen Befehle aus einem alten BASIC-Befehl mit ein oder zwei Zusatz-Zeichen.

@ PPOKE Speicherplatz, Adresse. Zum Beispiel: @ PPOKE 51,16384 POKEt oberes und unteres Byte von 16384 auf die beiden Speicherplätze 51 und 52.

@ PPEEK Speicherplatz. Zum Beispiel: @ PPEEK 51 gibt PEEK(51) +256*PEEK(52) aus.

Zum Beispiel @ POKER 49152,4,12,51,34,15,21 POKEt fünf Daten in den Speicher ab Adresse 49152.

@ POKES Startadresse, Endadresse, Wert (0 bis 255). Füllt einen Speicherbereich mit dem angegebenen Wert. Man kann den Speicher löschen, indem man Null eingibt, oder einen Teil des Bildschirms füllen, indem man einen Bildschirmcode in den Bildschirmspeicher schreibt.

@ BNEW Adresse. Setzt den BASIC-Anfang auf die angegebene Adresse.

@ GETNEW. Setzt den Anfang von BASIC auf den Standardwert. Gleichbedeutend mit dem OLD-Befehl. Die beiden letzten Befehle können dazu benutzt werden, um mehrere Programme im Speicher zu halten. Sie können etwa ein BASIC-Programm laden, den BASIC-Anfang etwa mit @ BNEW 16384 verschieben und ein weiteres Programm laden. @ BNEW 2048: @ GETNEW aktiviert dann erneut das erste Programm.

@ NNEW führt einen Kaltstart aus. Alle Programme werden gelöscht.

@ GGOTO Variable – springt auf die Zeile, die durch die Variable angegeben wird.
@ LRESTORE Zeilennummer – veranlaßt, daß das nächste READ in dieser Zeile ausgeführt wird.
@ MGOTO Anfangsadresse, Endadresse, neue Startadresse – verschiebt den Speicherbereich zwischen alter Anfangs- und Endadresse zur neuen Anfangsadresse.
@ MREAD Anfangsadresse, Endadresse – gibt den Inhalt des Speicherbereichs aus und summiert. Ein typisches Ergebnis ist etwa 12,32,65,22,#131. Man kann den Befehl verwenden, um aus Maschinencode DATA-Zeilen mit Prüfsummen zu erzeugen. Das kurze Programm am Ende dieses Abschnitts demonstriert diese Möglichkeit.
@ MSAVE Anfangsadresse, Endadresse, Programmname, Gerätenummer, 1 – Savet einen Speicherbereich als Maschinencode. Ein Anwendungsbeispiel wurde bereits beim Speichern des Werkzeugprogramms vorgeführt.
@ MFRE zeigt den für BASIC verfügbaren Speicherplatz an.
@ COST unteres Byte, oberes Byte – @ COST 6,1 beispielsweise berechnet 6+1*256.
@ POST Adresse – wandelt eine Adresse in zwei Bytes um.
@ CCHR$ Anzahl (von 0 bis 255), ASCII-Code – @ CCHR$ 6,65 gibt z. B. 6mal den Buchstaben A aus.
@ D' Dezimalzahl – wandelt in Hexadezimal um.
@ H'$ vierstellige Hex-Zahl – wandelt in Dezimal um.
@ ONTO Schrittweite (0–255) – startet die automatische Zeilennumerierung. Beginnt mit der ersten eingegebenen Zeilennummer.
@ ONTO RETURN – beendet die automatische Zeilennumerierung. Vorher sollten Sie SHIFT und RETURN eingeben.
@ RLIST, erste Zeilennummer, Schrittweite – Numeriert neu. @ RLIST,10,5 fängt bei Zeile 10 mit Schrittweite 5 an. Achten Sie auf das erste Komma, das mit eingegeben werden muß.
@ WAITGET, Variable – wartet auf einen Tastendruck. Der Name der Variablen ist beliebig.
@ WAIT' 711*Anzahl Sekunden – Verzögerung um die vorgegebene Anzahl von Sekunden. (Die 711 ist ein Skalierungsfaktor.)
@ PRINT% Farbe (0 bis 15),X(0 bis 39), Y(0 bis 24), auszudruckender Text – druckt den Text an der Position X,Y in der angegebenen Farbe.
@ COR Farbe (0 bis 15), Rahmenfarbe (0 bis 15), Hintergrundfarbe (0 bis 15) – Farbzusammenstellung
@ CLR' Nummer (0 bis 39) – löscht die angegebene Zeile
@ SCLR – löscht Bildschirm
@ SNEW – schaltet Bildschirm wieder ein
@ UP – Schiebt Bildschirm eine Zeile nach oben.
@ SYS1 – Bildschirm an
@ SYS0 – Bildschirm aus
@ ASC1 – Kleinbuchstaben
@ ASC0 – Großbuchstaben
@ ON1 – SHIFT unwirksam
@ ON0 – SHIFT wirksam
@ DEF1 – RUN/STOP unwirksam
@ DEF0 – RUN/STOP wirksam
@ FN1 – Wiederhol-Funktion ein
@ FN0 – Wiederhol-Funktion aus
@ KCLR – Tastatur Eingabe-Speicher löschen (wie POKE 198,0)
@ TOP – Cursor zum Bildschirmanfang
@ SIF – löscht alle Register des Sound-Chips.
@ SON Stimme, Lautstärke, A/D, S/R, Wellenform, oberes, unteres Byte Tonhöhe – Setzt die Parameter für einen Ton.

DATAs einlesen

Beim Befehl @ MREAD wurde schon erwähnt, daß man aus Maschinencode ein BASIC-Programm mit DATA-Zeilen erzeugen kann, indem man einen Speicherbereich auswertet. Angenommen, Sie haben mit einem Assembler ein Programm ab Adresse 49152 erzeugt, dann geben Sie erst die beiden folgenden Zeilen ein und schließen sie anschließend mit RETURN ab:

```
A=0:X=49152:@CCHR$79,32:?"□□□"
A*10"DATA□";:@MREAD X+A*15,14+
X+A*15
```

Nach dem RETURN sehen Sie die erste DATA-Zeile auf dem Bildschirm. Gehen Sie einfach mit dem Cursor in die Zeile und übernehmen sie diese mit RETURN. Gehen Sie dann mit dem Cursor zu der Zeile, die mit A=0 anfängt, und ändern Sie die 0 in eine 1. Dann können Sie mit der nächsten DATA-Zeile fortfahren. Erhöhen Sie A solange, bis alle DATAs eingelesen sind. Bei dieser Routine haben die Zeilen einen Abstand von 10. Sie können das ändern, wenn Sie in der zweiten Zeile die „10" vor DATA ändern.

Das Toolkit kennt keinen direkten Befehl zum Löschen von Blöcken. Schalten Sie die Wiederholfunktion durch @ FN1 ein, und geben Sie dann @ ONTO mit der geeigneten Schrittweite ein. Solange Sie die Return-Taste drücken, werden alle Zeilen von der Startzeile ab gelöscht.

Spectrum

Wenn Sie Besitzer eines Microdrives sind, ändern Sie bitte das Wort 'Cassette' in Zeile 30 auf 'Cartridge' und Zeile 87 in 87 Save* "m"; 1; "CREF" CODE 64560, 832.

Cross-Referencer

```
10 CLEAR 64559: BORDER 0: INK 7: PAPER
   0: CLS
20 PRINT INVERSE 1;AT 0,7;"□CROSS
   REFERENCER. "
30 PRINT """"□Poking in machine code.
   Prepare a casette for saving."
40 LET L=90: RESTORE L: FOR N=64560
   TO 65343 STEP 16
50 LET T=0: FOR D=0 TO 15: READ A:
   POKE N+D,A: LET T=T+A: NEXT D
60 READ A: IF A<>T THEN PRINT FLASH
   1;"□CHECK-SUM ERROR IN
   LINE ";L;"!": PRINT "EXPECTED
   VALUE ";A;"ACTUAL VALUE□";T: STOP
70 LET L=L+10: NEXT N
80 PRINT AT 18,0;"ANY KEY TO START SAVE
   PROCEDURE"
85 LET A$=INKEY$: IF A$="" THEN GOTO
   85
87 SAVE "CREF"CODE 64560,832
88 STOP
90 DATA 203,39,95,22,0,221,33,233,253,221,
   25,221,110,0,221,102,1999
100 DATA 1,205,69,252,201,62,254,229,205,
   1,22,225,126,35,254,255,2396
110 DATA 40,4,215,195,76,252,201,207,9,
   209,225,205,229,25,201,205,2498
120 DATA 142,2,14,0,32,249,205,30,3,48,244,
   21,95,205,51,3,1344
130 DATA 254,0,201,46,25,118,45,32,252,
```

```
201,62,13,205,48,252,33,1787
140 DATA 63,255,17,85,255,205,3,253,42,83,
92,34,59,255,126,254,2081
150 DATA 64,208,17,4,0,25,126,254,13,32,3,
35,24,237,205,179,1426
160 DATA 252,56,3,35,24,240,229,42,59,255,
205,85,24,62,13,215,1799
170 DATA 225,24,240,229,17,85,255,58,63,
255,71,26,19,190,35,32,1824
180 DATA 5,16,248,225,55,201,225,191,201,
62,254,205,1,22,42,83,2036
190 DATA 92,34,59,255,126,254,64,208,17,4,
0,25,126,254,13,32,1563
200 DATA 3,35,24,237,254,168,40,7,254,206,
40,3,35,24,237,229,1796
210 DATA 42,59,255,205,85,24,62,13,215,225,
62,13,1,0,0,237,1498
220 DATA 177,24,206,175,119,213,229,205,
115,252,205,95,252,245,215,241,2968
230 DATA 225,209,254,13,40,5,18,19,52,24,
234,201,62,13,205,48,1622
240 DATA 252,17,85,255,33,63,255,205,3,253,
62,18,205,48,252,33,2039
250 DATA 64,255,17,65,255,205,3,253,42,83,
92,126,254,64,208,17,2003
260 DATA 4,0,25,126,254,13,32,3,35,24,240,
254,34,32,9,35,1120
270 DATA 1,0,0,237,177,126,24,243,205,179,
252,56,3,35,24,227,1789
280 DATA 58,63,255,79,58,64,255,145,56,25,
40,8,79,6,0,229,1420
290 DATA 205,85,22,225,58,64,255,17,65,255,
79,6,0,235,237,176,1984
300 DATA 235,24,192,237,68,79,6,0,229,205,
232,25,225,24,229,62,2072
310 DATA 13,205,48,252,17,85,255,33,63,255,
205,3,253,33,150,0,1870
320 DATA 27,235,203,254,235,1,165,90,197,
205,179,252,56,13,35,203,2350
330 DATA 126,35,40,251,193,12,16,240,195,
87,252,193,42,83,92,34,1891
340 DATA 59,255,126,254,64,208,35,35,35,35,
126,35,254,13,40,239,1813
350 DATA 185,32,247,197,62,13,1,0,0,237,
177,229,42,59,255,205,1941
360 DATA 85,24,62,13,215,225,193,24,214,15,
254,70,254,79,254,86,2067
370 DATA 254,98,254,110,254,27,254,35,254,
48,254,64,254,124,254,143,2681
380 DATA 254,160,254,184,254,206,254,229,
254,236,254,0,255,10,255,66,3125
390 DATA 89,84,69,83,32,70,82,69,69,61,255,
80,82,79,71,82,1357
400 DATA 65,77,255,78,85,77,66,69,82,32,65,
82,82,65,89,255,1524
410 DATA 67,72,65,82,65,67,84,69,82,32,65,
82,82,65,89,255,1323
420 DATA 66,89,84,69,83,255,80,82,79,71,82,
65,77,61,255,32,1530
430 DATA 66,89,84,69,83,255,70,73,76,69,32,
84,89,80,69,61,1349
440 DATA 32,255,70,73,76,69,32,78,65,77,69,
61,32,255,70,73,1387
450 DATA 76,69,32,76,69,78,71,84,72,61,32,
255,69,78,84,69,1275
460 DATA 82,32,83,84,65,82,84,32,76,73,78,
69,32,58,255,69,1254
470 DATA 78,84,69,82,32,69,78,68,32,76,73,
78,69,32,58,255,1233
480 DATA 69,78,84,69,82,32,76,73,78,69,32,
73,78,67,82,69,1111
490 DATA 77,69,78,84,83,32,58,255,69,78,84,
69,82,32,84,65,1299
500 DATA 82,71,69,84,32,83,84,82,73,78,71,
58,32,255,69,78,1301
510 DATA 84,69,82,32,68,69,67,73,77,65,76,
32,78,85,77,66,1100
520 DATA 69,82,32,58,255,72,69,88,32,61,32,
255,69,78,84,69,1405
530 DATA 82,32,72,69,88,39,32,78,85,77,66,
69,82,58,32,255,1216
540 DATA 68,69,67,73,77,65,76,61,32,255,69,
78,84,69,82,32,1257
550 DATA 78,69,87,32,83,84,82,73,78,71,58,
255,0,0,0,0,1050
560 DATA 9,23,220,10,254,21,206,11,254,80,
3,23,220,10,215,24,1583
570 DATA 177,1,0,10,0,100,0,232,3,16,39,48,
48,49,48,0,771
```

Toolkit

```
5 CLEAR 63488: BORDER 0: PAPER 0: INK 6:
CLS
10 PRINT AT 0,10; INVERSE 1;" TOOLKIT "
12 PRINT AT 8,2;"□Press any key to load
cross-□□referencer machine code.":
PAUSE 0
14 LOAD "CREF"CODE
15 CLS : PRINT "Poking TOOLKIT machine
code.□□□□Please prepare a cassette for
saving."
20 LET L=100: RESTORE L: FOR N=63489
TO 64560 STEP 16
30 LET T=0: FOR D=0 TO 15
40 READ A: POKE (N+D),A: LET T=T+A:
NEXT D
50 READ A: IF A<>T THEN PRINT
"CHECKSUM ERROR IN LINE□";L: STOP
60 LET L=L+10
70 NEXT N
100 DATA 62,12,205,48,252,205,60,250,237,
67,155,248,42,83,92,1,2019
110 DATA 0,0,126,254,128,40,9,197,205,184,
25,193,3,235,24,242,1865
120 DATA 205,43,45,58,155,248,205,40,45,
239,4,56,205,162,45,33,1788
130 DATA 15,39,167,237,66,48,2,207,5,33,
145,248,126,60,40,32,1470
140 DATA 35,229,237,91,83,92,42,75,92,167,
237,82,68,77,235,237,2079
150 DATA 177,197,229,245,204,157,248,241,
225,193,234,80,248,225,24,220,3147
160 DATA 42,83,92,58,155,248,54,0,35,119,
205,40,45,239,49,192,1656
170 DATA 56,42,83,92,205,184,25,42,75,92,
43,167,237,82,216,235,1876
180 DATA 229,239,224,15,49,56,205,162,45,
225,112,35,113,43,24,228,2004
190 DATA 201,224,228,235,236,239,246,255,
0,0,10,0,229,6,4,35,2148
200 DATA 126,254,14,40,4,16,248,225,201,
197,35,35,35,78,35,70,1613
210 DATA 42,83,92,217,1,1,0,217,205,149,22,
43,235,167,237,66,1777
220 DATA 235,48,11,217,3,217,197,205,184,
25,193,235,24,234,217,197,2442
230DATA217,209,42,155,248,205,169,48,235,
42,104,92,35,35,115,35,1986
240 DATA 114,235,62,0,167,1,9,0,237,66,56,
17,60,1,90,0,1115
250 DATA 237,66,56,9,60,1,132,3,237,66,56,
1,60,209,225,229,1647
260 DATA 245,130,214,4,245,6,0,56,9,79,40,
12,205,85,22,35,1387
270 DATA 24,6,237,68,79,205,232,25,193,241,
197,79,6,0,9,65,1666
280 DATA 229,197,35,35,235,42,104,92,1,5,0,
237,176,193,225,229,2035
290 DATA 197,239,224,164,5,58,193,164,4,
224,1,3,225,192,2,56,1951
300 DATA 205,213,45,193,225,198,48,119,43,
16,228,229,42,104,92,35,2035
310 DATA 35,126,225,198,48,119,241,193,
245,42,83,92,167,229,237,66,2346
320 DATA 225,48,8,197,205,184,25,193,235,
24,242,235,35,35,193,126,2210
330 DATA 128,119,201,62,0,205,48,252,205,
26,31,33,0,0,62,0,1372
340 DATA 237,66,229,193,205,43,45,62,254,
205,1,22,205,227,45,201,2240
350 DATA 42,75,92,237,75,83,92,62,0,237,66,
229,62,1,205,48,1606
360 DATA 252,193,205,43,45,205,227,45,62,
2,205,48,252,201,221,33,2239
370 DATA 32,255,17,17,0,175,55,205,86,5,62,
3,205,48,252,221,1638
380 DATA 33,32,255,221,126,0,198,6,221,229,
205,48,252,62,13,215,2116
390 DATA 62,4,205,48,252,221,225,221,35,
221,126,0,254,255,40,10,2179
400 DATA 6,10,221,126,0,221,35,215,16,248,
62,13,215,62,5,205,1660
410 DATA 48,252,221,33,32,255,221,78,11,
221,70,12,195,163,249,205,2266
420 DATA 142,250,62,10,205,48,252,205,60,
250,205,110,25,229,62,13,2128
430 DATA 215,62,11,205,48,252,205,60,250,
205,110,25,193,32,16,229,2118
440 DATA 35,35,126,35,95,126,87,225,237,90,
```

```
    17,4,0,237,90,229,1668
450 DATA 197,62,0,237,66,218,87,252,195,89,
    252,6,0,197,205,95,2158
460 DATA 252,205,115,252,193,254,13,40,26,
    254,58,48,240,214,48,56,2268
470 DATA 236,245,4,120,254,6,32,4,5,241,24,
    225,241,245,198,48,2128
480 DATA 215,24,218,221,33,49,255,120,254,
    0,40,207,33,0,0,221,1890
490 DATA 94,0,221,86,1,241,254,0,40,7,237,
    90,56,13,61,32,1433
500 DATA 249,221,35,221,35,5,32,231,229,
    193,201,207,5,42,75,92,2073
510 DATA 237,75,83,92,237,66,192,207,9,62,
    10,205,48,252,205,60,2040
520 DATA 250,34,30,255,62,13,215,62,12,
    205,48,252,205,60,250,34,1987
530 DATA 28,255,33,48,48,34,59,255,34,61,
    255,237,75,30,255,205,1912
540 DATA 115,251,62,2,50,107,92,50,107,92,
    205,149,23,205,176,22,1708
550 DATA 62,0,205,1,22,33,59,255,6,4,126,
    229,197,205,129,15,1548
560 DATA 193,225,35,16,245,205,44,15,205,
    23,27,221,33,58,92,221,1858
570 DATA 203,0,126,32,13,42,89,92,205,167,
    17,62,255,50,58,92,1503
580 DATA 24,206,42,89,92,34,93,92,205,251,
    25,120,177,32,10,223,1715
590 DATA 254,13,40,174,205,176,22,207,1,
    237,67,73,92,42,93,92,1788
600 DATA 235,33,85,21,229,42,97,92,55,237,
    82,229,96,105,205,110,1953
610 DATA 25,32,6,205,184,25,205,232,25,
    193,121,61,176,40,47,197,1774
620 DATA 3,3,3,3,43,237,91,83,92,213,205,
    85,22,225,34,83,1425
630 DATA 92,193,197,19,42,97,92,43,43,237,
    184,42,73,92,235,193,1874
640 DATA 112,43,113,43,115,43,114,237,75,
    28,255,205,115,251,241,195,2185
650 DATA 195,250,33,62,255,126,60,254,58,
    40,8,119,11,121,128,176,1896
660 DATA 200,24,239,62,48,119,43,24,236,
    62,14,205,48,252,205,60,1841
670 DATA 250,62,13,215,197,62,15,205,48,
    252,193,46,2,96,124,203,1983
680 DATA 31,203,31,203,31,203,31,230,15,
    205,189,251,215,124,230,15,2207
690 DATA 205,189,251,215,97,45,32,230,62,
    13,215,201,198,48,254,58,2313
700 DATA 216,198,7,201,62,16,205,48,252,
    17,85,255,6,4,213,197,1982
710 DATA 205,95,252,205,115,252,215,245,
    241,193,209,18,19,16,239,62,2581
720 DATA 13,215,62,17,205,48,252,221,33,85,
    255,17,0,16,33,0,1472
730 DATA 0,14,4,221,126,0,221,35,214,48,
    218,87,252,254,10,56,1760
740 DATA 2,214,7,254,16,210,87,252,71,254,
    0,40,3,25,16,253,1704
750 DATA 203,58,203,27,203,58,203,27,203,
    58,203,27,203,58,203,27,1964
760 DATA 13,32,208,229,193,205,43,45,205,
    227,45,62,13,215,201,203,2139
800 CLS : PRINT AT 5,5;"□
    COMPILATION COMPLETE.□"
810 PRINT AT 7,2;"PREPARE A CASSETTE
    FOR SAVING."
820 PRINT AT 9,4;"FILENAME IS "
    "TOOLKIT""□CODE "
830 SAVE "TOOLKIT"CODE 63489,2000
```

Commodore

```
100 DATA 32,158,183,142,134,2,32,253,174,
    32,158,183,138,72,32, #1725
101 DATA 253,174,32,158,183,104,168,24,32,
    240,255,32,253,174,32, #2114
102 DATA 164,170,96,32,158,183,142,134,2,
    32,253,174,32,158,183, #1913
103 DATA 142,32,208,32,253,174,32,158,183,
    142,33,208,96,32,247, #1972
104 DATA 183,32,253,174,32,235,183,142,19,
    3,169,0,133,2,32, #1592
105 DATA 253,174,32,158,183,138,164,2,145,
    20,204,19,3,240,5, #1740
106 DATA 230,2,76,74,192,96,32,138,173,32,
    247,183,165,20,133, #1793
107 DATA 251,165,21,133,252,32,253,174,32,
    138,173,32,247,183,32, #2118
108 DATA 253,174,32,158,183,134,2,165,21,
    197,252,144,35,208,6, #1964
109 DATA 165,20,197,251,144,27,165,2,160,
    0,145,251,165,251,197, #2140
110 DATA 20,208,6,165,252,197,21,240,9,
    230,251,208,234,230,252, #2523
111 DATA 76,141,192,96,32,138,173,32,247,
    183,165,20,133,251,165, #2044
112 DATA 21,133,252,32,253,174,32,138,173,
    32,247,183,165,20,133, #1988
113 DATA 253,165,21,133,254,32,253,174,32,
    138,173,32,247,183,165, #2255
114 DATA 254,197,252,144,41,208,6,165,253,
    197,251,144,33,160,0, #2305
115 DATA 177,251,145,20,165,251,197,253,
    208,6,165,252,197,254,240, #2781
116 DATA 15,230,251,208,2,230,252,230,20,
    208,228,230,21,76,223, #2424
117 DATA 192,96,32,138,173,32,247,183,165,
    20,133,251,165,21,133, #1981
118 DATA 252,32,253,174,32,138,173,32,247,
    183,160,0,165,20,145, #2006
119 DATA 251,165,21,200,145,251,96,32,158,
    183,32,255,233,96,32, #2150
120 DATA 138,173,32,247,183,160,0,177,20,
    170,200,177,20,32,205, #1934
121 DATA 189,96,32,138,173,32,247,183,165,
    20,133,251,165,21,133, #1978
122 DATA 252,32,253,174,32,138,173,32,247,
    183,165,21,197,252,144, #2295
123 DATA 61,208,6,165,20,197,251,144,53,
    169,0,133,253,133,254, #2047
124 DATA 24,160,0,177,251,170,101,253,133,
    253,165,254,105,0,133, #2179
125 DATA 254,169,0,32,205,189,169,44,32,
    210,255,165,251,197,20, #2192
126 DATA 208,6,165,252,197,21,240,9,230,
    251,208,214,230,252,76, #2559
127 DATA 104,193,169,35,32,210,255,166,
    253,165,254,76,205,189,32, #2338
128 DATA 138,173,32,247,183,162,0,232,208,
    253,198,20,169,255,197, #2467
129 DATA 20,208,245,198,21,197,21,208,239,
    96,32,158,183,160,0, #1986
130 DATA 224,1,208,2,160,7,224,2,208,2,
    160,14,132,2,32, #1378
131 DATA 253,174,32,158,183,142,24,212,32,
    253,174,32,158,183,138, #2148
132 DATA 164,2,153,5,212,32,253,174,32,158,
    183,138,164,2,153, #1825
133 DATA 6,212,32,253,174,32,158,183,138,
    164,2,153,4,212,142, #1865
134 DATA 19,3,32,253,174,32,158,183,138,
    164,2,153,1,212,32, #1556
135 DATA 253,174,32,158,183,138,164,2,153,
    0,212,206,19,3,173, #1870
136 DATA 19,3,164,2,153,4,212,96,162,0,138,
    157,0,212,232, #1554
137 DATA 224,25,208,248,96,76,68,229,76,24,
    229,76,234,232,169, #2214
138 DATA 0,141,138,2,96,169,128,141,138,2,
    96,169,0,133,198, #1551
139 DATA 96,169,237,141,40,3,96,169,251,
    141,40,3,96,76,102, #1660
140 DATA 229,169,27,141,17,208,96,169,11,
    141,17,208,96,169,21, #1719
141 DATA 141,24,208,96,169,23,141,24,208,
    96,169,9,76,210,255, #1849
142 DATA 169,8,76,210,255,32,138,173,32,
    247,183,76,163,168,32, #1962
143 DATA 138,173,32,247,183,169,0,168,145,
    20,24,165,20,105,1, #1590
144 DATA 133,43,165,21,105,0,133,44,76,154,
    227,169,62,32,210, #1574
145 DATA 255,169,18,32,210,255,165,55,56,
    229,45,170,165,56,229, #2109
146 DATA 46,32,205,189,169,96,160,228,76,
    30,171,169,0,133,198, #1902
147 DATA 165,198,201,1,208,250,76,146,171,
    169,8,160,1,145,43, #1942
148 DATA 32,51,165,24,165,34,105,2,133,45,
    133,47,133,49,165, #1283
149 DATA 35,105,0,133,46,133,48,133,50,96,
    32,138,173,32,247, #1401
150 DATA 183,165,20,133,63,165,21,133,64,
    32,19,166,56,165,95, #1480
151 DATA 233,1,133,65,165,96,233,0,133,66,
    96,162,0,181,43, #1607
152 DATA 149,251,232,224,4,208,247,32,138,
    173,32,247,183,165,20, #2305
```

```
153 DATA 133,43,165,21,133,44,32,253,174,
    32,138,173,32,247,183,#1803
154 DATA 165,20,133,45,165,21,133,46,32,86,
    225,162,0,181,251,#1665
155 DATA 149,43,232,224,4,208,247,96,32,
    158,183,134,2,32,253,#1997
156 DATA 174,32,158,183,142,19,3,165,2,201,
    0,240,11,173,19,#1522
157 DATA 3,32,210,255,198,2,76,79,195,96,
    76,154,227,32,138,#1773
158 DATA 173,32,247,183,170,169,72,32,210,
    255,169,39,32,210,255,#2248
159 DATA 169,36,32,210,255,138,32,139,195,
    138,32,144,195,152,32,#1899
160 DATA 139,195,152,32,144,195,96,24,106,
    106,106,106,41,15,24,#1481
161 DATA 105,48,201,58,144,2,105,6,32,210,
    255,96,169,68,32,#1531
162 DATA 210,255,169,39,32,210,255,32,186,
    195,133,34,32,186,195,#2163
163 DATA 170,165,34,32,205,189,76,228,167,
    32,203,195,10,10,10,#1726
164 DATA 10,133,35,32,203,195,101,35,133,
    35,96,32,115,0,201,#1356
165 DATA 58,41,15,144,2,105,8,96,32,138,
    173,32,247,183,169,#1443
166 DATA 91,32,210,255,169,0,166,20,32,
    205,189,169,44,32,210,#1824
167 DATA 255,169,0,166,21,32,205,189,169,
    93,32,210,255,96,32,#1924
168 DATA 158,183,134,2,32,253,174,32,158,
    183,138,166,2,76,205,#1896
169 DATA 189,5,0,0,32,121,0,208,6,169,0,
    141,14,196,96,#1177
170 DATA 169,1,141,14,196,169,53,141,4,3,
    169,196,141,5,3,#1405
171 DATA 32,138,173,32,247,183,165,20,141,
    12,196,165,21,141,13,#1679
172 DATA 196,96,173,0,2,201,48,144,59,201,
    58,176,55,173,14,#1596
173 DATA 196,240,50,32,124,165,132,2,173,
    12,196,24,101,20,133,#1600
174 DATA 99,173,13,196,101,21,133,98,162,
    144,56,32,73,188,32,#1521
175 DATA 223,189,133,254,132,255,160,0,
    177,254,240,6,153,119,2,#2297
176 DATA 200,208,246,132,198,164,2,96,76,
    124,165,0,0,0,0,#1611
177 PRINT"■":X=49152:C=76:
    GOSUB 200
178 DATA 169,11,141,8,3,169,205,141,9,3,96,
    32,115,0,201,#1303
179 DATA 64,240,3,76,231,167,160,1,177,122,
    133,255,160,2,177,#1968
180 DATA 122,133,2,162,0,189,128,206,197,
    255,208,9,232,189,128,#2160
181 DATA 206,197,2,240,16,202,201,0,240,6,
    232,232,224,128,208,#2334
182 DATA 230,162,11,108,0,3,134,2,32,115,0,
    32,115,0,32,#976
183 DATA 115,0,166,2,189,255,205,133,252,
    189,0,206,133,253,169,#2267
184 DATA 76,133,251,32,251,0,76,174,167,96,
    0,0,0,0,0,#1256
185 X=52480:C=6:GOSUB 200
186 DATA 0,192,33,192,64,192,96,192,169,
    192,1,193,36,193,43,#1788
187 DATA 193,61,193,163,193,189,193,36,194,
    48,194,51,194,54,194,#2150
188 DATA 57,194,63,194,69,194,74,194,80,
    194,86,194,89,194,95,#1971
189 DATA 194,101,194,107,194,113,194,118,
    194,123,194,132,194,159,194,#2405
190 DATA189,194,202,194,233,194,8,195,65,
    195,97,195,100,195,159,#2415
191 DATA 195,215,195,251,195,15,196,73,197,
    0,0,0,0,0,0,#1532
192 X=52736:C=5:GOSUB 200
193 DATA 153,37,67,176,151,82,151,83,77,
    137,80,151,156,39,80,#1620
194 DATA 194,77,135,146,39,83,145,83,139,
    83,156,83,162,85,80,#1690
195 DATA 165,48,165,49,75,156,150,48,150,
    49,164,80,158,48,158,#1663
196 DATA 49,198,48,198,49,145,48,145,49,71,
    137,66,162,77,184,#1626
197 DATA 146,161,161,162,76,140,77,148,67,
    199,78,162,68,39,72,#1756
198 DATA 39,185,84,190,84,145,164,82,155,0,
    0,0,0,0,0,#1128
199 X=52864:C=5:GOSUB 200:GOTO 205
200 FOR Z=0 TO C:T=0:FOR ZZ=0 TO
    14:READ M:POKE X,M
201 PRINT"■LINE"PEEK(63)+PEEK(64)*
    256:T=T+M:X=X+1
202 NEXT ZZ:READ X$:IF VAL(RIGHT$(X$,
    LEN(X$)-1))<>T THEN 204
203 NEXT Z:PRINT"OK■■":RETURN
204 PRINT"ERROR IN LINE!":END
205 K=50505:T=0
206 READA:IFA=-1THEN209
207 POKEK,A:K=K+1
208 T=T+A:GOTO206
209 IFT<>52549THENPRINT"■■
    CHECKSUM ERROR":END
210 IFK<>50928 THENPRINT
    "■■NUMBER OF VALUES ERROR":END
211 PRINT"USE SYS 52480 TO EXECUTE
    MACHINE CODE"
212 END
213 DATA 32,253,174,32,167,169,165
214 DATA 20,133,53,165,21,133,54
215 DATA 32,253,174,32,167,169,165
216 DATA 20,133,49,165,21,133,50
217 DATA 32,142,166,32,201,198,32
218 DATA 201,198,208,33,32,2,198
219 DATA 32,201,198,32,201,198,208
220 DATA 3,76,212,198,32,201,198
```

```
221 DATA 165,99,145,122,32,201,198
222 DATA 165,98,145,122,32,13,198
223 DATA 240,226,32,201,198,32,201
224 DATA 198,32,201,198,201,34,208
225 DATA 11,32,201,198,240,197,201
226 DATA 34,208,247,240,238,170,240
227 DATA 188,16,233,162,4,221,235
228 DATA 198,240,5,202,208,248,240
229 DATA 221,165,122,133,59,165,123
230 DATA 133,60,32,115,0,176,211
231 DATA 32,107,169,32,32,198,165
232 DATA 60,133,123,165,59,133,122
233 DATA 160,0,162,0,189,0,1
234 DATA 240,17,72,32,115,0,144
235 DATA 3,32,82,198,104,160,0
236 DATA 145,122,232,208,234,32,115
237 DATA 0,176,8,32,97,198,32
238 DATA 121,0,144,248,201,44,240
239 DATA 186,208,152,165,53,133,99
240 DATA 165,54,133,98,76,142,166
241 DATA 165,99,24,101,49,133,99
242 DATA 165,98,101,50,133,98,32
243 DATA 201,198,208,251,96,32,2
244 DATA 198,32,201,198,32,201,198
245 DATA 208,8,169,255,133,99,133
246 DATA 98,48,14,32,201,198,197
247 DATA 20,208,16,32,201,198,197
248 DATA 21,208,12,162,144,56,32
249 DATA 73,188,76,223,189,32,201
250 DATA 198,32,13,198,240,209,32
251 DATA 114,198,230,251,32,165,198
252 DATA 230,45,208,2,230,46,96
253 DATA 32,114,198,198,251,32,141
254 DATA 198,165,45,208,2,198,46
255 DATA 198,45,96,32,124,198,160
256 DATA 0, 132, 17, 132, 251, 96, 165
257 DATA 122, 133, 34, 165, 123, 133, 35
258 DATA 165,45,133,36,165,46,133
259 DATA 37,96,164,17,200,177,34
260 DATA 164,251,200,145,34,32,190
261 DATA 198,208,1,96,230,34,208
262 DATA 236,230,35,208,232,164,17
263 DATA 177,36,164,251,145,36,32
264 DATA 190,198,208,1,96,165,36
265 DATA 208,2,198,37,198,36,76
266 DATA 165,198,165,34,197,36,208
267 DATA 4,165,35,197,37,96,160
268 DATA 0,230,122,208,2,230,123
269 DATA 177,122,96,32,51,165,165
270 DATA 34,166,35,24,105,2,133
271 DATA 45,144,1,232,134,46,32
272 DATA 89,166,76,116,164,0,137
273 DATA 138,141,167,-1
```

Hintertürchen

Über das ROM des Interface 1 können Sie den Sinclair Spectrum mit eigenen BASIC-Befehlen versehen. In der vorigen Folge hatten wir uns die „Mechanik" dieser neuen Befehle angesehen, diesmal zeigen wir, wie sie angelegt werden.

Die neuen BASIC-Befehle müssen bei den Syntax- und Runtimeprüfungen des Haupt-ROMs und des Interface-1-ROMs zunächst einmal Fehlermeldungen erzeugen, da sonst die Steuerung nicht an die von der Variablen VECTOR (bei 23735 und 23736) angegebene Routine übergeben wird. Die Fehler werden mit zwei Techniken hervorgerufen:

1. Sie verändern ein Schlüsselwort, das die Tests dann nicht mehr bestehen kann. Dabei genügt die Eingabe einer falschen Parameterzahl, einer veränderten Syntax oder eines zusätzlichen Zeichens. Wir geben Ihnen dafür Beispiele:

BORDER* (ein an den Befehl angehängtes Zeichen) LINE 10,10,1,19 (zu viele Parameter)

2. Wenn Sie ein Zeichen vor den Befehl stellen, erzeugen Sie einen neuen Befehl, beispielswiese *BEEP oder *PRINT.

Da Befehle dieser Art beliebig auf BASIC-Variablen zugreifen können, erstellen wir unsere neuen Befehle mit der zweiten Methode.

Um etwa den Befehl *B zu definieren, der einen Ton von der Dauer einer Sekunde erzeugen soll, muß zunächst entschieden werden, wo der neue Code untergebracht werden soll. Ein REM-Befehl kommt nicht in Frage, da die Systemvariablen des Interface 1 nicht immer mit den gleichen Speicheradressen arbeiten. Mit dem Befehl CLEAR nn schaffen Sie jedoch den notwendigen Platz.

Auf Fehler reagiert der Spectrum mit RST #08. Sobald die Hardware des Interface 1 feststellt, daß der Programmzähler des Z80 diese Adresse enthält, wird das ROM des Interface 1 aktiviert und ein Sprung auf die von VECTOR angezeigte Routine ausgeführt.

Systembaukasten

Jeder neue Befehl muß bestimmte „Bauteile" enthalten (siehe Bild). Wenn nun der BASIC-Interpreter einen Fehler erkennt und die von VECTOR angegebene Routine aufruft, sollte man zuerst untersuchen, welches Zeichen des BASIC-Programms das Problem verursacht hat.

Praktischerweise zeigt die Systemvariable CHADD (bei 23755 und 23756) auf dieses Zeichen. Mit zwei Routinen des Haupt-ROMs – GETCHAR und NEXTCHAR (siehe Kasten „Nützliche Adressen") – holen Sie sich das Zeichen. Die Routinen werden aus dem Bereich des Schatten-ROMs mit RST #10 gefolgt von #0018 (GETCHAR) oder #0020 (NEXTCHAR) angesprochen.

Beim Aufruf der Routine, auf die VECTOR zeigt, enthält CHADD die Adresse des Zeichens, das den Fehler ausgelöst hat. Der Befehl *B besteht aus folgendem Code:

```
D7      vector: rst  #10
1800            defw #0018              ;address of GETCHAR
FE2A            cp   "*"                ;is it a "*"?
C2F001          jp   nz,#01F0           ;if not, error
D7              rst  #10
2000            defw #0020              ;address of NEXTCHAR
FE42            cp   "B"                ;is it a "B"
C2F001          jp   nz,#01F0           ;if not, error
D7              rst  #10                ;get NEXTCHAR, should...
2000            defw #0020              ; ...be end of statement
CDB705          call #05B7              ;check end of statement
        runtime:
```

Vor Aufruf der Schatten-ROM-Routine bei #05B7 muß das Zeichen, auf das CHADD zeigt, im A-Register gespeichert werden. Das gleiche Ergebnis läßt sich auch mit NEXTCHAR erreichen, doch veranlaßt die Syntaxprüfung in der Routine bei #05B7 einen Rücksprung zum Haupt-ROM, um dort die nächste Anweisung zu testen. Während des Ablaufs (Runtime) steht hier jedoch ein normaler Subroutinenrücksprung, der den Code beim Label RUNTIME anspricht.

Beachten Sie, daß die Befehle Zeichen für Zeichen geprüft werden. Sie können die Routine leicht umbauen und so die Befehlseingabe mit Groß- und Kleinbuchstaben möglich machen. Wenn ein Zeichen den Test nicht besteht, wird wie bei einer Fehlermeldung das Schatten-ROM über #01F0 (Version 1) angesprochen. Sie müssen das aktuelle von CHADD angegebene Zeichen hier allerdings mit GETCHAR statt mit NEXTCHAR holen.

Der neue Befehl erzeugt den Ton über die BEEP-Routine des Haupt-ROMs:

```
!!0501 runtim: ld   de,#0105           ;duration of 1 second
210007         ld   hl,#0700           ;pitch
D7             rst  #10
B503           defw #03B5              ;execute BEEP routine
C3C105         jp   #05C1              ;exit
```

Eine Verzweigung zu #05C1 gibt die Steuerung an das Haupt-ROM zurück.

Ihre neuen BASIC-Befehle können – über RST #10 – natürlich auch mit anderen Routinen des Haupt-ROMs und den Routinen des Grafik-ROMs (wir gehen später genauer darauf ein) arbeiten. So setzt das folgende Listing des *B-Befehls die Routinen bei #1C82 und #1E94 (siehe Kasten) zur Bewertung eines numerischen Parameters ein, der vom Anwender angegeben wurde. Das Befehlsformat lautet *B n,

wobei n zwischen 0 und 255 liegen kann. Der Parameter gibt die Länge des BEEP in Sekunden an; *B 255 erzeugt einen Ton von 255 Sekunden.

```
                org  60000
        VECTOR: equ  #5CB7              ;VECTOR address
CF      init:   rst  #8
31              defb 49                 ;set up IF1 variables
2169EA          ld   hl,newcom          ;add of new command
22B75C          ld   (VECTOR),hl        ;get it in VECTOR
C9              ret                     ;end of initialisation

D7      newcom: rst  #10                ;VECTOR jumps here
1800            defw #0018              ;GETCHAR
FE2A            cp   "*"
C2F001          jp   nz,#01F0
D7              rst  #10
2000            defw #0020              ;NEXTCHAR
FE42            cp   "B"
C2F001          jp   nz,#01F0
D7              rst  #10
2000            defw #0020
D7              rst  #10
821C            defw #1C82              ;evaluate parameter
CDB705          call #05B7              ;check statement end
D7      runtim: rst  #10
941E            defw #1E94              ;get param in a
47              ld   b,a
C5      loop:   push bc                 ;save counter
110501          ld   de,#0105           ;duration
210007          ld   hl,#0700           ;pitch
D7              rst  #10
B503            defw #03B5              ;call BEEP routine
C1              pop  bc                 ;restore counter
10F3            djnz loop
C3C105          jp   #05C1              ;exit - all done
```

Da die Routine bei #1C82 nur Acht-Bit-Werte akzeptiert (sie hat für diesen Wert nur die acht Bits des A-Registers zur Verfügung), erzeugen Zahlen außerhalb des Bereichs von 0 bis 255 einen Fehler. Denken sie auch daran, daß der Zähler FRAMES während der Tonerzeugung nicht inkrementiert wird und damit auch die Tastaturabfrage abgeschaltet ist. Bei Angabe des Parameters 255 sollten Sie sich daher auf einige Wartezeit einrichten.

Bisher wurden die Fehler durch Eingaben erzeugt, die nicht dem Standard entsprachen. Sie können aber auch die normalen Schlüsselworte des Sinclair-BASIC so modifizieren, daß dadurch Fehler hervorgerufen werden. So rufen die Zusatzparameter der folgenden Beispiele einen Sprung auf die von VECTOR angegebene Routine hervor:

```
CIRCLE x,y,z,n
LINE x
BORDER x,y,z
```

Hier zeigt CHADD beim Einsprung in die VECTOR-Routine auf das „fehlerhafte" Zeichen, das mit dem entsprechenden Tokencode verglichen wurde.

Trotz der recht komplizierten Technik lassen sich in das Sinclair-BASIC leicht neue Befehle einbauen. Sie müssen nur genau wissen, welches ROM zur Zeit aktiv ist. Das Schatten-ROM läßt sich vom Haupt-ROM aus jederzeit mit dem Hakencode 50 (und der gewünschten Adresse) ansprechen.

Interface-1-Versionen

Wenn Sie direkt auf Routinen des Schatten-ROMs zugreifen wollen, sollten sie zuvor feststellen, mit welcher Version des Interface 1 Sie arbeiten. Generell sind alle IF1 mit Seriennummern über 87315 mit der ROM-Version 2 ausgestattet. Die Eingabe des folgenden Befehl ist jedoch zuverlässiger:

```
CLOSE #0;PRINT PEEK 23729
```

Bei einem Ergebnis 0 liegt Version 1 vor, bei 80 die Version 2. Glücklicherweise haben alle ROM-Routinen, die wir zur Erstellung neuer Befehle angesprochen haben, in beiden Versionen die gleichen Adressen. Andere Routinen dürften jedoch Schwierigkeiten machen. Mit einem geeigneten Disassembler und unserem Ladeprogramm für das Schatten-ROM (siehe letzte Folge) können Sie sich den Inhalt der Routinen anzeigen lassen.

Das ROM der Version 2 enthält zwei neue Hakencodes. Hakencode #33 lädt die Datensatzbeschreibung einer Datei und Hakencode #34 eröffnet einen RS 232 „b" Kanal.

Nützliche Adressen

Im Schatten-ROM:

#01F0 Verläßt das Schatten-ROM über die Fehlerroutine des Haupt-ROMs

Verläßt das Schatten-ROM, wenn das A-Register #0D (ENTER) oder ein Semikolon enthält. Sonst: Fehler.

#05C1 Kehrt bei neuer Befehlsannahme zum Interpreter zurück.

Im Haupt-ROM:

#0018 GETCHAR – Holt CHADD (siehe unten) aus HL und speichert das Zeichen, auf das CHADD zeigt, in A. Prüft, ob das Zeichen angezeigt werden kann und veranlaßt gegebenenfalls einen Rücksprung.

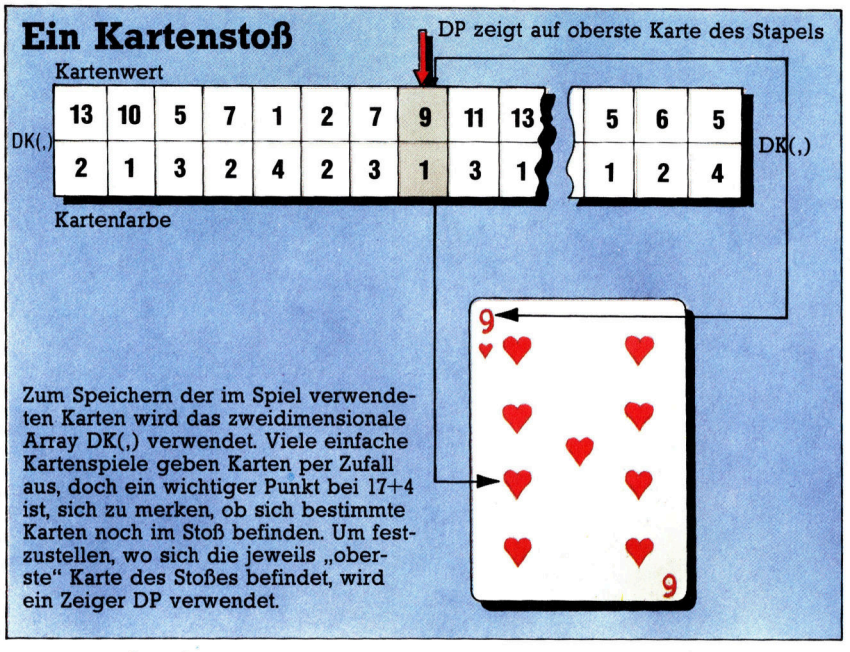
#0020 NEXTCHAR – Inkrementiert CHADD und speichert den Wert in HL. Holt das Zeichen, auf das CHADD zeigt, und prüft, ob es angezeigt werden kann. Ist dies der Fall, wird das Zeichen in A gespeichert und ein Rücksprung veranlaßt.

#1C82 Bewertet oder prüft einen numerischen Ausdruck, auf dessen erstes Element CHADD zeigt. Beim Rücksprung zeigt CHADD auf das erste Zeichen hinter dem Ausdruck, während das Ergebnis sich auf dem Berechnungsstack des Haupt-ROMs befindet.

#1E94 Nimmt einen Wert vom Berechnungsstack des Haupt-ROMs und stellt ihn in A, wenn er zwischen 0 und 255 liegt.
Sonst: Fehler.
Die Systemvariable CHADD (bei 23755 und 23756) zeigt auf das interpretierte Zeichen.

Gut gemischt

Unser neues Programmprojekt beschäftigt sich mit dem Kartenspiel 17+4. Zunächst entwickeln wir die Routinen zur Darstellung und zum Mischen des Blattes. Sie können sie, wenn Sie Spaß am Computerkartenspiel bekommen, auch in eigenen Programmen auf verschiedene Weise einsetzen.



Zum Speichern der im Spiel verwendeten Karten wird das zweidimensionale Array DK(,) verwendet. Viele einfache Kartenspiele geben Karten per Zufall aus, doch ein wichtiger Punkt bei 17+4 ist, sich zu merken, ob sich bestimmte Karten noch im Stoß befinden. Um festzustellen, wo sich die jeweils „oberste" Karte des Stoßes befindet, wird ein Zeiger DP verwendet.

Kartenspiele eignen sich gut für die Übertragung auf einen Computer. Ihre Spielregeln und Strategien finden ihren Ursprung oftmals in der Mathematik. 17+4 ist besonders einfach zu programmieren, denn die Regeln sind klar definiert.

Zu Beginn wollen wir untersuchen, wie man einen Kartenstoß generiert und einzelne Karten darstellt. Dabei entwickeln wir zunächst das Listing für den C64; Sinclair Spectrum, Acorn B und die Schneider CPC-Computer werden in der nächsten Folge bedient.

Der Einfachheit halber speichern wir die Karten in einem zweidimensionalen Array. In unserem Spiel dient dazu DK(,) mit 52 (Anzahl der Karten) mal zwei Elementen DIMensioniert. Jede Karte verfügt über zwei Merkmale, die sie im Stoß einzigartig machen: ihr Wert (As, 2, 3 usw.) und die Farbe (Herz, Karo, Pik und Kreuz).

Damit man ein numerisches Array verwenden kann, haben wir für die Kartenwerte die Zahlen 1 bis 13 (1=As – 13=König), sowie in der zweiten Dimension die Zahlen 1 bis 4 für die Farbe verwendet. Ab Zeile 500 findet die Dimensionierung statt.

Die Methode zum Kartenmischen arbeitet mit zwei verschachtelten Schleifen. Die äußere Schleife zählt die vier Farben und die innere die 13 Karten jeder Farbe. Man braucht nur einen Platz für die Karte innerhalb des Stoßes auszuwählen.

Da der Stoß ja gemischt werden soll, wählt unsere Routine einen zufälligen Platz aus. Dabei gibt es ein Problem: Die gewählte Position könnte belegt sein. Daher wird mit einer Routine in der inneren Schleife die gewählte Position überprüft. Ist sie nicht frei, wird sie um eins erhöht, bis ein freier Platz gefunden ist.

In die Ecke gestellt

Um eine Karte des Stoßes darzustellen, benötigt man lediglich Kartenwert und Farbe. Durch diese Angaben werden das Kartenmuster und die Kartenecken generiert. Obwohl bei den meisten Karten die Ecken dem Wert der Karte entsprechen, benötigt man für As, Bube, Dame und König die Bezeichnungen A, J, Q und K. Außerdem müssen wir die 10 als T darstellen, damit die Bezeichnung in einer Spalte der Karte Platz hat. Am einfachsten ist es, ein String-Array CN$() für Kartennummern zu initialisieren, in dem die 13 Bezeichnungen gespeichert werden.

Das Kartenmuster ist schwieriger zu realisieren. zur Vereinfachung stellen wir alle Bildkarten so dar wie das As: Auf ihnen ist also nur ein einzelnes Farbensymbol zu sehen.

In drei Spalten und sieben Reihen sind die Farb-Symbole darstellbar. Es stehen also 21 Positionen zur Verfügung. Zum Speichern der Muster gibt es mehrere Möglichkeiten. Wir haben uns entschlossen, jedes Kartenmuster in 21 Elementen eines Binär-Strings zu speichern, in dem jede 1 ein darzustellendes Symbol repräsentiert. Die 13 Kartendefinitionen befinden sich in DATA-Anweisungen bei Zeile 2100 und werden in das Array CD$() eingelesen.

Auf den Punkt gebracht

Bei der C64-Version gibt es keine Möglichkeit, den Cursor direkt an einer bestimmten Bildschirmstelle zu positionieren. Daher verwendet das Programm für diese Aufgabe eine Unterroutine bei Zeile 900, um den Cursor an einem durch die Variablen TX und TY definierten Punkt zu positionieren.

Die Unterroutine bei Zeile 1050 druckt mit Hilfe vordefinierter Zeichen den Kartenrand. Die Kartendetails werden dann durch die Unterroutine bei Zeile 1100 hinzugefügt, wobei die Kennung für Kartenwert und Farbe über CN und SU übergeben wird. Erste Aufgabe ist nun, das Farbsymbol aus dem entsprechenden String auszuwählen. Ob eine Karte schwarz oder rot ist, kann einfach durch Überprüfung der Farbzahl ermittelt werden.

Daraufhin können die Muster mittels der Binärdaten aus CD(CN) gedruckt werden. Wieder werden zwei verschachtelte Schleifen verwendet, mit denen die sieben Reihen durchlaufen werden, wobei der Binärstring in die entsprechenden Farbsymbole umgewandelt und dargestellt wird. Anschließend kann die Eckbezeichnung hinzugefügt werden, indem der Cursor entsprechend positioniert und CN$(CN) ausgegeben wird.

Während des Spieles werden Karten an den Spieler und den Computer ausgegeben. Dabei werden sie so plaziert, daß sie übereinanderliegen, horizontal jedoch um zwei und vertikal um eine Einheit verschoben sind. Um die Position festzustellen, an der die nächste Karte dargestellt wird, werden die Arrays X() und Y() verwendet. Das erste Element enthält die nächste Kartenposition für den Spieler, das zweite die für den Computer. Der Cursor wird also vor der eigentlichen Ausgaberoutine auf X(PL),Y(PL) gesetzt, wobei PL die Spielernummer ist (1 oder 2). Am Ende der Routine werden X(PL) und Y(PL) um Zwei bzw. Eins erhöht, damit die nächste Karte wieder richtig positioniert werden kann.

In unserer Version des Spieles übernimmt der Computer auch die Bank, so daß seine erste ausgegebene Karte verdeckt aufgelegt wird. Daher müssen wir auch eine Kartenrückseite darstellen. Hierzu dient die Routine ab Zeile 1200, die zuerst die Routine für den Kartenrand aufruft und danach die Innenfläche mit einem karierten Muster füllt.

Die vorerst letzte Routine erlaubt uns, Karten vom Stoß auszugeben und darzustellen. Hierfür ist die Unterroutine ab Zeile 1300 zuständig. Sie nimmt die Elemente des Kartenstoß-Arrays, die der obersten Karte entsprechen und weist sie CN und SU zu.

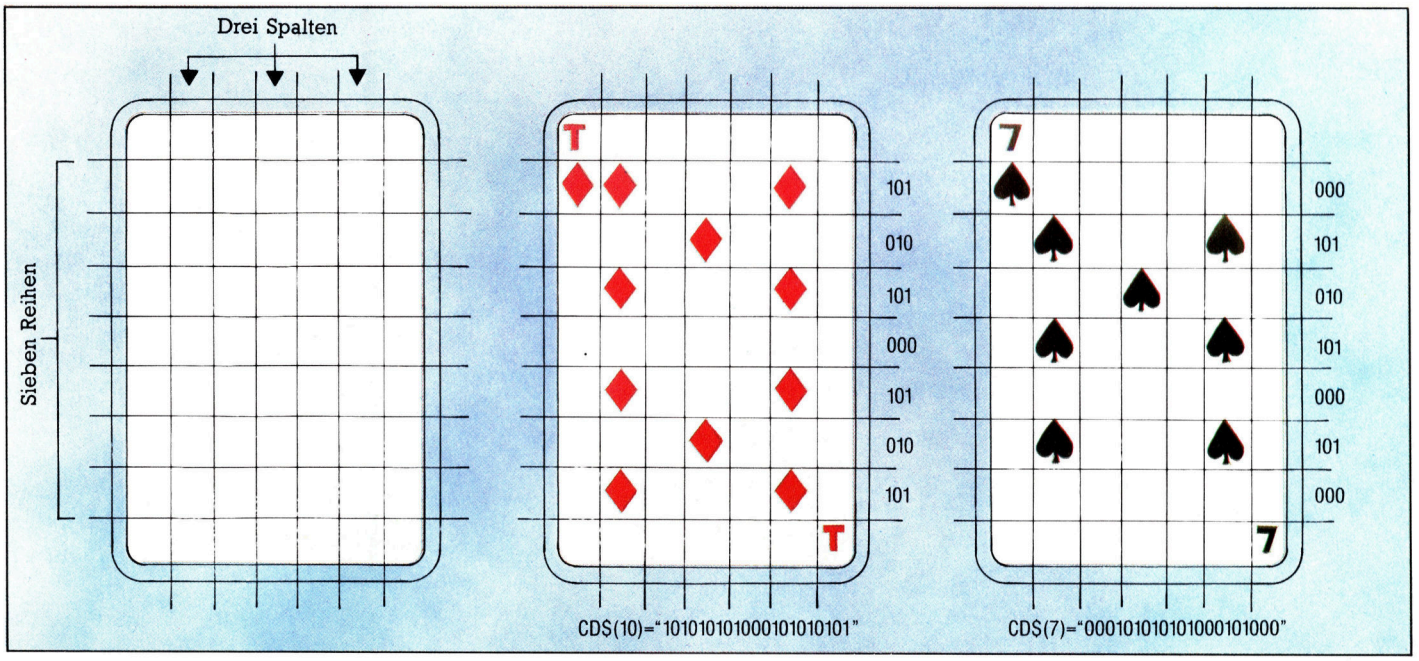
Wandernder Zeiger

Der offensichtlichste Weg zum Ausgeben der Karten wäre, die ersten Elemente des Stoß-Arrays zu nehmen, alle anderen Elemente einen Platz nach vorne zu schieben und die gerade entfernten Elemente in DK(52,1) und DK(52,2) abzulegen. Diese ganze Verschieberei ist jedoch sehr zeitintensiv und letztendlich nicht nötig. Stattdessen kann man die Variable DP verwenden, die auf die jeweils „oberste" Karte zeigt. Es wird somit Element DK(DP,1) für den Kartenwert und DK(DP,2) für die Farbnummer verwendet.

Trotz dieser Vereinfachung bleibt der wichtige Gesichtspunkt berücksichtigt, daß der Abbau des Stapels vom Spieler im Geiste nachvollzogen werden kann. Sind im Laufe des Spieles schon die meisten niedrigen Karten-

Aufnehmen einer Karte

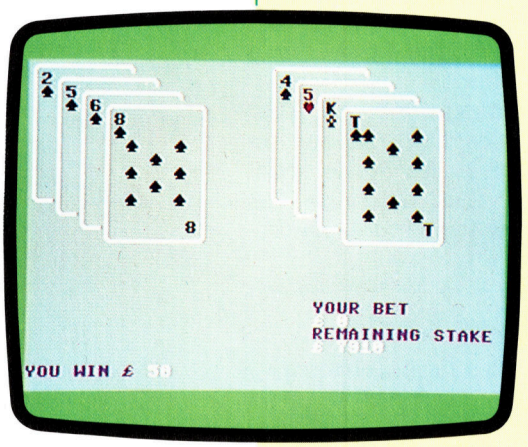
Die Muster aller 13 Karten sind als Binärstrings gespeichert. Unter Verwendung von drei Spalten und sieben Reihen zeigt das Diagramm, wie die Binärstrings für eine „Zehn" und eine „Sieben" erstellt wurden. Die Kartendarstellungsroutine verarbeitet den zugehörigen Binärstring in Dreiergruppen. Dabei wird bei einer 1 das entsprechende Farbsymbol ausgegeben und eine Leerstelle bei einer 0.



werte ausgeteilt worden, so empfiehlt es sich nicht, noch darauf zu hoffen, mit einer niedrigen Karte vom Geber bedient zu werden. Hier setzt die Spieltaktik ein, die wir, um das Spiel auch wirklich interessant gestalten zu können, schon bei den Routinen zum Kartenausteilen in Anschlag bringen müssen.

Um ein Ergebnis der bisherigen Arbeit zu sehen, können Sie die folgenden Zeilen in das Listing integrieren. Dadurch werden die einzelnen Routinen aufgerufen und fünf Karten an den Spieler und den Computer ausgegeben.

```
60 PL=1: FOR C = 1 TO 10
70 PL = 3—PL
80 FL = 0:GOSUB 1300
90 INPUT 'DRUECKE RETURN FUER DIE
   NAECHSTE KARTE'; AN$
100 NEXT C
```



Unser Programm stellt die Karten in der Hand des Spielers und der Bank (= des Computers) auf der linken und rechten Seite des Bildschirms dar.

Mischen und Darstellung der Karten

Hauptprogramm:

```
10 REM **** COMMODORE 64 PONTOON ****
20 GOSUB 500:REM INIT ARRAYS ETC
50 REM **** GAME LOOP ****
55 GOSUB 600:REM INIT GAME
```

Initialisierung:

```
500 REM ***** INIT ARRAYS ETC ****
510 SP$="":DW$="":FORI=1TO25:DW$=DW$+CHR
$(17):SP$=SP$+" ":NEXT I
511 SP$=SP$+LEFT$(SP$,14)
512 BK$="":LI$="":FORI=1 TO 7:LI$=LI$+CH
R$(195):BK$=BK$+CHR$(166):NEXT I
513 BR$=CHR$(194)
515 DIM X(2),Y(2):REM NEXT CARD POSITION
S
520 SU$=CHR$(211)+CHR$(218)+CHR$(216)+CH
R$(193):REM SUIT TYPES
530 DIM CN$(13):FORI=1TO13:READ CN$(I):N
EXT:REM READ NUMBER DATA
540 DIM CD$(13):FORI=1 TO 13:READ CD$(I)
:NEXT:REM READ PATTERN DATA
560 DIM DK(52,2):REM SET UP CARD DECK AR
RAY
570 GOSUB3000:REM SHUFFLE PACK
580 POKE53280,5:POKE53281,15:REM SCREEN
COLOURS
590 RETURN
600 REM **** INIT GAME ****
605 PRINT CHR$(147):REM CLEAR SCREEN
620 X(1)=0:Y(1)=0:X(2)=20:Y(2)=0
630 RETURN
```

Routine zur Darstellung der Karten:

```
900 REM **** PRINT AT ****
910 PRINTCHR$(19);:PRINTLEFT$(DW$,TY);TA
B(TX);:RETURN
1000 REM **** DISPLAY CARD ****
1010 GOSUB 1050:GOSUB 1100:RETURN
1050 REM **** CARD BLANK ****
1055 TX=X(PL):TY=Y(PL):GOSUB 900:REM POS
ITION
1060 PRINT TAB(TX);CHR$(5);CHR$(213);LI$
;CHR$(201)
1070 FORI=1 TO 9:PRINT TAB(TX);BR$;"
   ";BR$:NEXT I
1080 PRINT TAB(TX);CHR$(202);LI$;CHR$(20
3)
1090 RETURN
1100 REM **** CARD DETAIL ****
1120 TX=X(PL)+2:TY=Y(PL)+2:GOSUB 900:REM
 POSITION
1125 CT$=MID$(SU$,SU,1):REM SELECT SUIT
TYPE
1127 CO$=CHR$(28):IF SU>2 THEN CO$=CHR$(
144):REM SELECT COLOUR
1128 PRINTCO$;
 130 FOR I=1 TO 19 STEP 3
 140 CC$=MID$(CD$(CN),I,3):CL$=""
 142 FOR J=1 TO 3:C$=CHR$(29)+CHR$(29)
 144 IF MID$(CC$,J,1)="1" THEN C$=CT$+CH
R$(29)
1146 CL$=CL$+C$:NEXT J
1150 PRINT TAB(X(PL)+2);CL$:NEXT I
1160 REM **** ADD CORNER LABELS ****
1170 TX=X(PL)+1:TY=Y(PL)+1:GOSUB900:PRIN
TCN$(CN):REM NUMBER
1180 TY=TY+1:GOSUB900:PRINTCT$:REM SUIT
1190 TX=X(PL)+7:TY=Y(PL)+9:GOSUB900:PRIN
TCN$(CN):REM BOTTOM NUMBER
1192 X(PL)=X(PL)+2:Y(PL)=Y(PL)+1
1195 RETURN
1200 REM **** DISPLAY CARD BACK ****
1210 GOSUB 1050:GOSUB 1250:RETURN
1250 REM **** CARD BACK ****
1255 TX=X(PL):TY=Y(PL)+1:GOSUB 900:REM P
OSITION
1260 FORI=1 TO 9:PRINT TAB(TX);BR$;CHR$(
156);BK$;CHR$(5);BR$:NEXT I
1270 X(PL)=X(PL)+2:Y(PL)=Y(PL)+1
1280 RETURN
1300 REM **** DEAL A CARD ****
1310 CN=DK(DP,1):SU=DK(DP,2)
1320 DP=D+DP+1:IFDP>52 THEN DP=1:REM BUM
P DECK
1330 IF FL=1 THEN GOSUB 1200:RETURN:REM
DISPLAY BACK OF CARD
1335 GOSUB 1000:RETURN:REM DISPLAY CARD
2000 REM **** CARD NUMBER DATA ****
2010 DATA A,2,3,4,5,6,7,8,9,T,J,Q,K
2100 REM **** CARD DISPLAY DATA ****
2110 DATA"00000000001000000000":REM A
2120 DATA"00001000000000001000":REM 2
2130 DATA"00001000001000001000":REM 3
2140 DATA"00010100000000101000":REM 4
2150 DATA"00010100001000101000":REM 5
2160 DATA"00010100101000101000":REM 6
2170 DATA"00010101010100101000":REM 7
2180 DATA"00010101010101010000":REM 8
2190 DATA"10100010101010001010":REM 9
2200 DATA"10101010100010101010":REM 10
2210 DATA"00000000001000000000":REM J
2220 DATA"00000000001000000000":REM Q
2230 DATA"00000000001000000000":REM K
```

Mischen der Karten:

```
3000 REM **** SHUFFLE THE DECK ****
3005 R=RND(-TI):DP=1
3007 FORI=1 TO52:DK(I,1)=0:NEXT I
3010 FORI=1TO4:FORJ=1TO13
3020 EP=INT(RND(1)*52)+1:REM SELECT ENTR
Y POINT
3030 IF DK(EP,1)=0 THEN 3050
3040 EP=EP+1:IF EP>52 THEN EP=1
3045 GOTO 3030
3050 DK(EP,1)=J:DK(EP,2)=I:NEXT J,I
3060 RETURN
```

Universalgenie

Wir beginnen eine Serie über das Betriebssystem Unix und beschreiben einleitend die Unix-Philosophie und die Entwicklungsgeschichte. Nach einem Blick auf die Unix-„Shell" als Benutzerschnittstelle werden die Vorgänge beim Einloggen erläutert.

Ein Betriebssystem hat nicht nur die elementare Ein/Ausgabe und die Speicher- und Dateiverwaltung zu besorgen, sondern muß außerdem eine Reihe von Dienstprogrammen oder „Tools" bieten, die dem Benutzer effiziente Lösungen ermöglichen. Die Palette der Tools reicht von Compilern für diverse Sprachen bis zu Texteditier- und Druckformatierungsprogrammen. Die Stärke von Unix liegt gerade im vielseitigen Angebot an Tools und der Möglichkeit, sie einfach und elegant miteinander zu verbinden: Durch das „Pipeline"-Verfahren wird die Ausgabe des einen Dienstprogramms dem nächsten unmittelbar als Eingabe zugeleitet. Auf dieser Verknüpfung vieler simpler Routinen zur Bewältigung kompliziertester Aufgaben beruht wesentlich die Leistungsfähigkeit des Systems. Die Unix-Philosophie läßt sich in drei Leitsätze fassen:

1. Schreibe für jede Teilaufgabe ein eigenes kurzes und effizientes Programm.
2. Richte jedes Programm so ein, daß seine Ausgabe als Eingabe für ein weiteres verwendbar ist, auch wenn dieses noch nicht existieren sollte.
3. Entwickle für jede neue Aufgabe ein neues Programm, statt ein altes umzubauen, damit die eigene Tool-Bibliothek wächst.

Die Sache hat natürlich einen Haken: Alten Hasen erscheint es völlig logisch, mit zahllosen Tools zu hantieren und sie zur Lösung komplexer Probleme zu verknüpfen, aber Einsteiger tun sich damit nicht leicht. Wer die Möglichkeiten von Unix voll nutzen will, muß viele Dinge beachten.

Hinter der Fassade

Der einzige Weg, der auch Unerfahrenen den Zugang eröffnet, ist die Zwischenschaltung einer benutzerfreundlichen „Fassade", hinter der die eigentliche Vielseitigkeit, aber auch ein Teil der Leistungsfähigkeit des Systems verborgen bleibt. In der Originalform ist Unix deshalb weit öfter an der Hochschule und in der Forschung zu finden als im Büro- oder Heimcomputer-Bereich.

Wie viele technische Neuerungen wurde auch Unix eher zufällig in die Welt gesetzt. In den Bell Laboratories, der Forschungsabteilung des riesigen US-Konzerns AT&T (American Telephone & Telegraph, von Graham Bell gegründet), arbeitete Ken Thompson an der Simulation von Planetenbewegungen auf einem GE645-Computer mit einem der ersten Mehrplatz-Betriebssysteme namens „Multics". Die Softwareentwicklung gestaltete sich dabei so umständlich, daß Thompson sich lieber einer kleinen PDP-7 von Digital Equipment zuwandte. Um damit angemessen arbeiten zu können, schrieb er eigens ein Betriebssystem, das viele der Multics-Funktionen auf den kleinen Rechner übertrug. Dies System schlug so gut ein, daß sich dafür bei Bell mehrere Leute interessierten, u. a. Dennis Ritchie, und es bis 1971 zur ersten Unix-Version erweiterten.

Wie die meisten damaligen Betriebssysteme wurde auch Unix zunächst in Assembler geschrieben und lief daher nur auf den DEC-Minicomputern richtig. Dennis Ritchie entwikkelte aber gerade eine neue Sprache namens B – ein Abkömmling der Hochsprache BCPL, die fast ebenso maschinennahes Programmieren wie Assembler gestattet. Aus dem B-Entwurf entstand bald die Sprache C, und in C wurde Unix dann neu abgefaßt. Da nur ein Minimalanteil assemblerabhängig blieb, ist das System sehr leicht auf andere Computer übertragbar, die über einen C-Compiler verfügen. Unix läuft daher heute auf fast jeder Maschine,

Das Betriebssystem Unix ist vorwiegend bei größeren Rechnern anzutreffen, Hewlett-Packard hat aber auch auf ihrem PC-Portable eine Version mit der Bezeichnung HP-UX implementiert. Dieser leistungsfähige Computer verfügt über einen Lumineszenz-Flachbildschirm, integrierten Tintenstrahldrucker und bis zu 2,5 MByte RAM. Das ROM-gespeicherte Unix läßt sich über die Benutzerschnittstelle ‚PAM' (Personal Applications Manager) anhand von Piktogrammen und Fenstermenüs mit der Maus steuern.

vom Großrechner bis zum 16-Bit-Micro, sogar auf 8-Bit-Computern.

Bis vor kurzem war es recht schwierig und kostspielig, Unix für den kommerziellen Gebrauch zu erwerben. An akademische Einrichtungen vergibt die Firma AT&T dagegen die Lizenzen fast gratis, so daß sich Unix dort sehr stark durchsetzte. Im kommerziellen Bereich fand Unix dann Eingang in Gestalt „Unix-ähnlicher" Systeme wie Xenix (IBM-XT/AT), Cromix, Onyx, Idris, Coherent und OS-9 (für 6809-Rechner). Sie entsprechen in vielem dem Original-Unix-System, genau wie die neueren CP/M und MS-DOS-Versionen.

Die bisherigen Unix-ähnlichen Systeme sind an der Bell-Version 7 orientiert, die neueren (z. B. HP-UX für den ‚Integral PC' von Hewlett-Packard) an der Version V. Bei den abgeleiteten Fassungen, die an der kalifornischen Berkeley-Universität entstanden, ist jetzt die Version 4.2 aktuell, und auf dieses Berkeley 4.2 wird auch im folgenden Bezug genommen.

Eine wesentliche Besonderheit von Unix ist die „Shell" – das ist der Kommandointerpreter des Systems als Benutzerschnittstelle. Im Unterschied zu anderen Betriebssystemen ist der Kommandointerpreter hier seitens des Rechnerlieferanten wie des Benutzers anpaßbar. Einzelne Kommandos oder, bei Bedarf, die ganze ‚Shell' können neu implementiert werden. In Verbindung mit der reichen Auswahl an Tools lassen sich mit der Unix-Shell komplexe Programme entwickeln.

Bevor die Umbenennung und Neudefinition von Kommandos an der Reihe ist, sind einige Hinweise zu den Standardbefehlen angebracht – die meisten haben das Format:

kommando_name optionen
datei_oder_directory_name

Jede Option beginnt mit einem Minuszeichen. Es ist zu beachten, daß bei Unix Groß- und Kleinbuchstaben nicht gleichwertig sind; die Namen „HANS", „Hans" und „hans" haben also unterschiedliche Bedeutung.

Der erste Schritt beim Einstieg in ein Unix-System ist wie bei jedem anderen Mehrplatzsystem das „Einloggen". Dazu werden Sie zu Beginn der Sitzung automatisch aufgefordert, und auch später können Sie den Vorgang jederzeit durch Eingabe von ‚login' wieder in Gang setzen. Das System fragt mit

login:

nach der Benutzerkennung in der vereinbarten Form und verlangt nach deren Eintippen mit

password:

Ihr Paßwort, das beim Eingeben nicht auf dem Schirm erscheint, damit es niemand erspäht.

Sesam, öffne dich

Paßwörter bestehen meistens aus sechs Zeichen. Wenn Sie beim Eintippen einen Fehler machen, können Sie es nochmal versuchen, aber nicht beliebig oft – das System schreitet nach mehreren Anläufen zur Aussperrung, damit Unbefugte die Paßwörter nicht einfach durch Probieren herausbekommen. Nach dem korrekten Einloggen erscheint eine Systemmeldung.

Von hier an verfährt Unix nach Ihrem maßgeschneiderten Shell-Programm, das in einem speziellen .login-File steht und in dem auch Ihr persönliches Promptzeichen statt des standardmäßigen ‚%' vereinbart ist. Wenn Sie als rechtmäßiger Benutzer akzeptiert sind, stellt Unix selbstätig die Verbindung mit der Directory her, unter der Ihre Dateien abgelegt sind. Auch der Zugriff auf fremde Dateien ist möglich, wenn sie als „öffentlich" gekennzeichnet sind. Sie können bei Bedarf aber auch Dateien oder Directories für die Mitbenutzer der Anlage sperren. Wollen Sie Ihr Paßwort ändern, was in regelmäßigen Zeitabständen zu empfehlen ist, so tippen Sie das Kommando „passwd" ein, worauf zunächst die Eingabe des alten und dann zweimal die des neuen Paßworts von Ihnen verlangt wird.

Ein neues System lernt man am besten kennen, indem man es einfach ausprobiert, und dazu bietet Unix zwei besondere Hilfsmittel an: Eins davon ist das ‚On-line-Manual', das auf Wunsch zu praktisch jedem Befehl oder jedem interessierenden Thema einen Bildschirmauszug des Handbuchs liefert. Die Unterstützungsanforderung lautet

man topic_name

Das zweite ist der in Unix enthaltene Trainingskurs für Neulinge, der mit ‚learn' aufgerufen wird und zu den verschiedensten Fragen Lektionen erteilt.

Die Unix-Story

Viele Benutzer, die mit Betriebssystemen unzufrieden waren, sahen in Unix eine interessante Alternative. Das System wurde von einer kleinen Programmierergruppe entwickelt und ist verhältnismäßig kompakt. Vieles hat auch in neuen Versionen anderer Betriebssysteme Eingang gefunden.

1969 Dennis M. Ritchie und Ken Thompson verfassen in den Bell Laboratories (USA) für den eigenen Gebrauch ein PDP-7-Betriebssystem, das sie ‚Unix' taufen – eine Anspielung auf das damals verbreitete Mammutsystem ‚Multics'

1971 Unix wird auf die PDP–11-Rechner übertragen.

1973 Während die ursprünglichen Unix-Versionen als Assemblercode vorlagen, wird nun fast das ganze System in die Sprache C umgeschrieben, so daß es auf jeden anderen Rechner mit C-Compiler übertragbar ist.

1974/75 Die PWB-Version kommt heraus – im wesentlichen ein stark erweitertes Unix (‚Workbench'- oder ‚Programmierer'-Version).

1977 Vorstellung der Version 7, auf der die meisten derzeitigen Unix-Systeme beruhen; sie ist weitgehend von maschinenabhängigen Elementen befreit.

Endlich komplett

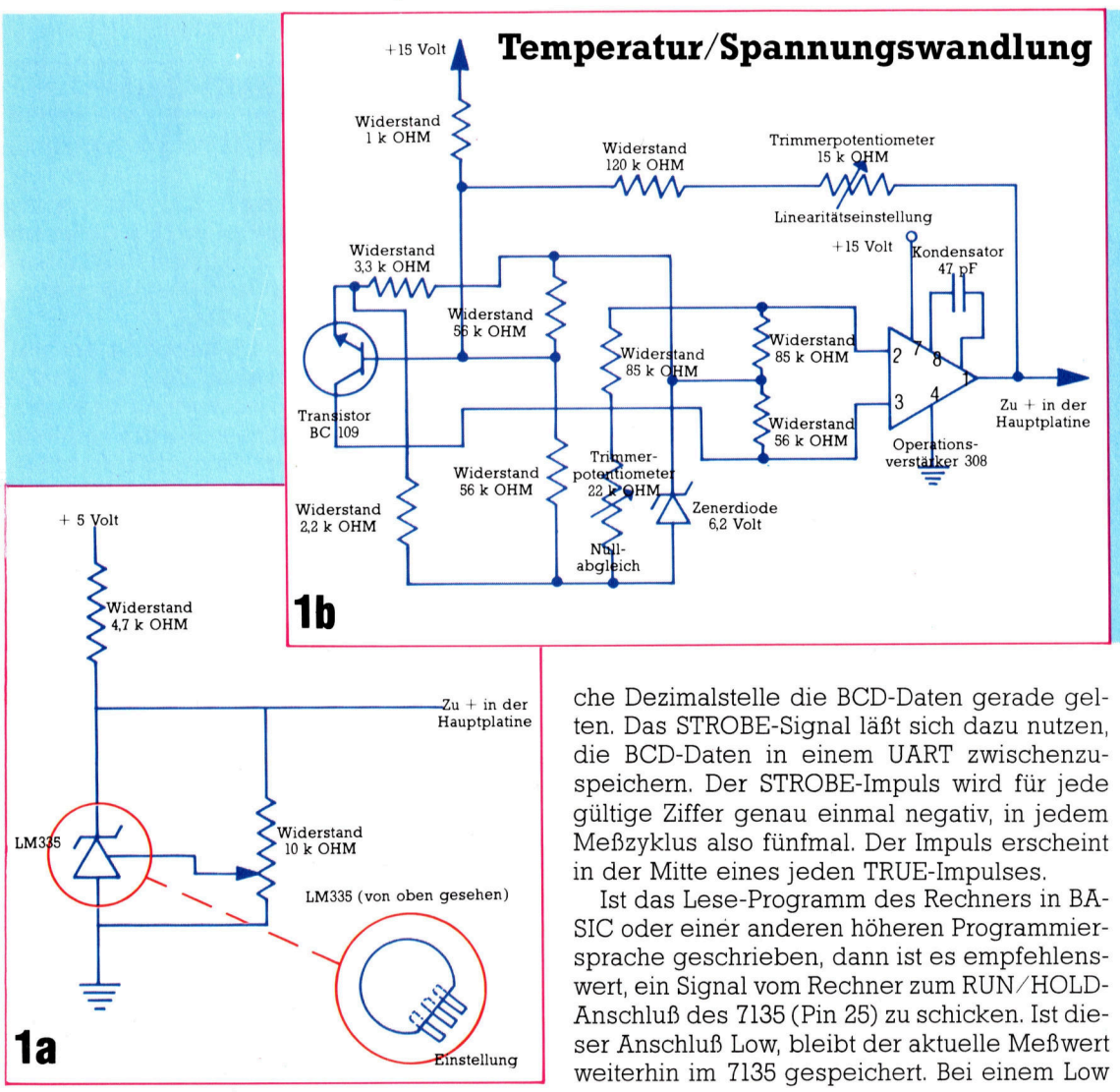
Wir schließen unser Digitalvoltmeter-Projekt mit der Vorstellung zweier Zusatzgeräte ab: einem digitalen Thermometer und einer Computer-Schnittstelle zur Realisation von Spezialaufgaben.

Am einfachsten läßt sich ein Digitalvoltmeter durch Anschluß eines käuflichen Temperatur-Tastkopfes für die Wärmemessung ausrüsten. Diese Geräte erzeugen eine zur Temperatur proportionale Spannung. Für einen Bruchteil des Ladenpreises läßt sich ein Taster aber auch im Eigenbau herstellen. In Bild 1 haben wir zwei Schaltungsvarianten dafür zusammengestellt. Die erste, einfachere Schaltung erzeugt eine Spannung, die pro Grad Temperaturerhöhung um 10 mV anwächst. Allerdings muß ein konstanter Spannungsanteil abgezogen werden, um die gemessenen Werte richtig zu interpretieren. Falls das DVM an einen Computer angeschlossen ist, macht das keine Schwierigkeiten – die Subtraktion wird programmgesteuert durchgeführt. Der Meßbereich umfaßt -10 bis +100 Grad Celsius.

Die zweite Schaltung nutzt die Temperaturabhängigkeit eines normalen PNP–Transistors (Typ BC 109), die von einem Operationsverstärker gebuffert wird. Dem Meßbereich 0 bis 100 Grad entspricht eine linear anwachsende Spannung zwischen 0 und +5 Volt.

Wir haben den Wandler-Chip 7135 deshalb ausgesucht, weil er sich besonders leicht mit Computern koppeln läßt. Die Ausgangssignale des ICs stehen im BCD-Code (Binär codierte Dezimalwerte). Mit den zusätzlichen Ziffern-(digital-) Leitungen wird angegeben, für welche Dezimalstelle die BCD-Daten gerade gelten. Das STROBE-Signal läßt sich dazu nutzen, die BCD-Daten in einem UART zwischenzuspeichern. Der STROBE-Impuls wird für jede gültige Ziffer genau einmal negativ, in jedem Meßzyklus also fünfmal. Der Impuls erscheint in der Mitte eines jeden TRUE-Impulses.

Ist das Lese-Programm des Rechners in BASIC oder einer anderen höheren Programmiersprache geschrieben, dann ist es empfehlenswert, ein Signal vom Rechner zum RUN/HOLD-Anschluß des 7135 (Pin 25) zu schicken. Ist dieser Anschluß Low, bleibt der aktuelle Meßwert weiterhin im 7135 gespeichert. Bei einem Low



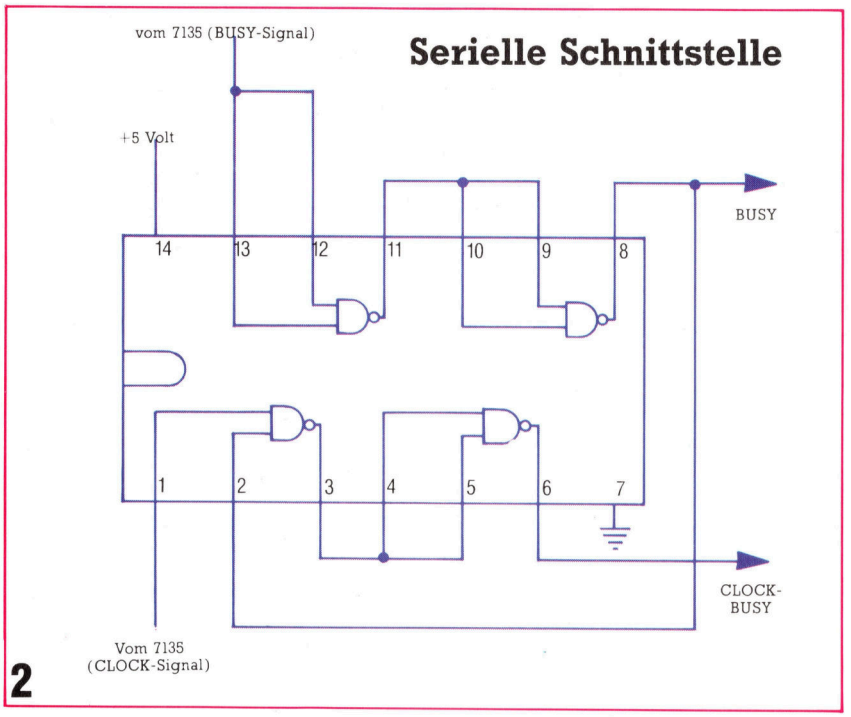
Diese Schaltpläne zeigen zwei alternative Möglichkeiten für die Umwandlung von Temperaturen in Spannungswerte. Schaltung 1b ist komplizierter als Schaltung 1a. Die erzeugte Spannung ist der Temperatur direkt proportional. Mit den beiden Trimmerpotentiometern können Nullpunkt und Linearität der Schaltung genau eingestellt werden.

Über einen 74LS00-Chip lassen sich die Signale des A/D-Wandlers 7135 so koppeln, daß ein CLOCK.BUSY-Signal entsteht. Das Signal wird zum Computer geleitet, der das Zählen der ankommenden Impulse übernimmt.

am STROBE-Anschluß (Pin 26) sendet das Programm dann ein Low zum RUN/HOLD-Pin und hat nun beliebig viel Zeit zum Auslesen der BCD-Daten, wobei die Ziffern-Treiberpulse (Pin 20,19,18,17 und 12) als Strobe fungieren.

Einfacher läßt sich das Voltmeter über den BUSY-Ausgang (Pin 21) mit dem Rechner verbinden. Am Anfang jeder Integrationsphase geht dieser Pegel auf High, einen Taktimpuls nach deren Abschluß wieder auf Low (bei Zwischenspeicherung der digitalen Datenausgabe). Eine Integrationsphase dauert 10000 Taktpulse (einen Taktpuls nach dem Nulldurchgang geht BUSY auf Low). Wenn nun CLOCK und BUSY durch eine logische AND-Schaltung miteinander verknüpft und die Anzahl der Pulse gezählt werden, die durch das AND-Gatter zum Computer gelangen, ergibt sich der Meßwert von 10001 Pulsen.

Eine noch einfachere Möglichkeit zum Zählen: Warten, bis der BUSY-Ausgang auf High geht und dann eine Schleife durchlaufen, bis BUSY wieder Low ist. Dabei erhöht man COUNT bei jedem Durchlauf um 1. Für einen solchen Ablauf muß die Taktfrequenz der CPU genau bekannt sein sowie die genaue Anzahl von Maschinentakten, die für die Ausführung eines Befehls nötig sind. Auch die Voltmeter-Taktfrequenz muß man kennen. Dadurch kann man die Länge eines BUSY genau bestimmen und muß diesen Wert von der 10001-fachen Taktimpulslänge abziehen, um die Dauer der Referenzintegrationsphase zu berechnen. Das Resultat wird durch die Dauer von CLOCK geteilt und ergibt die Anzahl der CLOCK-Pulse bei der Referenzintegration. Jeder Zählpuls entspricht dann genau 0,0001 Volt.



2

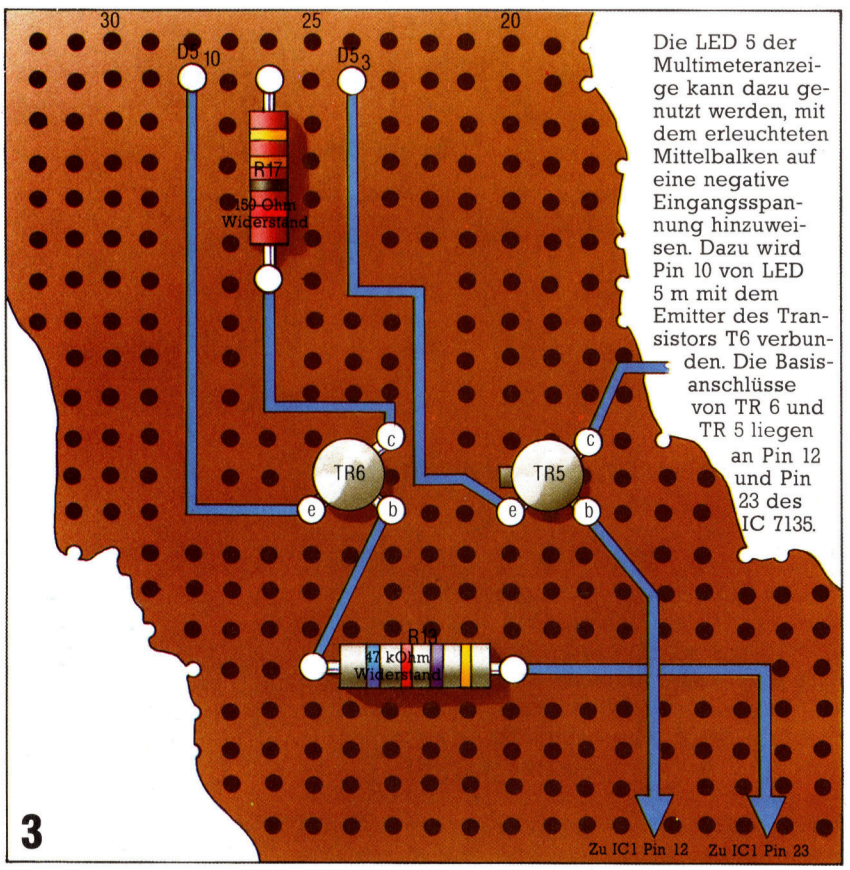
Nützliche NAND-Gatter

Das logische AND von BUSY- und Taktsignal (CLOCK) läßt sich über ein IC 74LS00 mit vier NAND-Gattern realisieren. Bild 2 zeigt, wie das BUSY.CLOCK-Signal erzeugt wird. Mit diesem Chip läßt sich auch ein gepuffertes BUSY herstellen, falls Sie nur die Dauer der Referenzintegrationsphase messen wollen.

Der Prototyp unseres Digitalvoltmeters war mit einem LED-Anzeigefeld für die fünfte Stelle ausgestattet, das über einen besonderen Plus- und Minuszeichenanschluß verfügte. Wir hatten das wegen möglicher Beschaffungsprobleme abgeändert und auch hier eine normale Siebensegment-Anzeige verwendet.

Die Schaltung (siehe S. 2081) war von der Änderung mitbetroffen. Da die Siebensegment-LED eine gemeinsame Anode für alle Segmente hat, erscheint das Minuszeichen nur bei vollständig erleuchteter Ziffer. Um die Anzeige für die Darstellung des Minuszeichens zu verwenden, sind folgende Veränderungen vorzunehmen: TR5 bleibt unverändert, der Kollektor wird aber an +5 Volt (Versorgungsspannung) angeschlossen und der Emitter mit der Anode von LED 5 (Pin 3) verbunden.

TR6 wird folgendermaßen angeschlossen: Basis an Pin 23 (Polarität) von IC1, Emitter an die Kathode von Segment g der LED5 (Pin 10). Der Kollektor wird über einen 150 Ohm-Widerstand auf Masse gelegt. In der Zeichnung ist der Schutzwiderstand des TR6-Kollektors an der Platinenkante festgelötet. Dieser Punkt muß mit der digitalen Masse verbunden sein.



3

Charles Babbage

Obwohl die von Charles Babbage konstruierten Rechenmaschinen nie richtig fertiggestellt wurden, sind sie als die unmittelbaren Vorfahren des heutigen programmierbaren Computers anzusehen.

Wenn sich diese Rechnungen doch um Himmelswillen mit der Dampfmaschine erledigen ließen!" soll Charles Babbage gerufen haben, als er sich mit den Tabellen für den Nautischen Almanach abquälte. Das 19. Jahrhundert hatte der Menschheit zwar die Dampfmaschine beschert, aber die exakte Navigation war in der christlichen Seefahrt immer noch ein Problem: Die Schiffsposition wurde aus der Beobachtung von Sonne, Mond und Sternen anhand mathematischer Tafeln berechnet, die oft ziemlich fehlerhaft waren.

Im Jahr 1812 kam Babbage die Idee, eine Maschine für die umständliche Berechnung der nautischen Tafeln zu bauen. 1823 hatte er ein kleines Modell seiner „Difference Engine" fertig und bat die Regierung um einen Zuschuß für die Finanzierung einer betriebsfähigen Anlage. Der Schatzkanzler gab ihm 1500 Pfund, und Babbage machte sich an den Bau des Rechners.

Dieses Projekt bestimmte Babbages weiteren Lebensweg. Es verschlang Unsummen, denn rein technisch lag es absolut an der Grenze des damals Machbaren. Die erforderlichen Gelder kamen durch Unterstützung des Premierministers, des Duke of Wellington, zusammen. Aber trotz Babbages fester Zusicherung, die Maschine würde ‚alles, was sie tue, garantiert fehlerfrei erledigen' zog sich die Regierung zurück, nachdem sie 17 000 Pfund riskiert hatte. Auch Babbages Mechaniker verschwand bald darauf nach einer heftigen Auseinandersetzung und nahm sämtliche Werkzeuge mit, die speziell für den Bau der Maschine angefertigt worden waren.

Statt zu resignieren, wandte sich Babbage kurzerhand einem noch ehrgeizigeren Projekt zu, nämlich der „Analytical Engine", die noch weit mehr als die „Difference Engine" leisten sollte. Das Konzept wies bereits entscheidende Merkmale des modernen Computers auf: Es waren ein großer Zahlenspeicher und eine Rechen-‚Mühle' (Mill) vorgesehen, die etwa der CPU entsprach, ferner ein Druckwerk für die Ausgabe und vor allem die Möglichkeit der Programmierung unter Verwendung bedingter Verzweigungen.

Für die Befehlssteuerung waren zunächst Stachelwalzen wie bei der Drehorgel gedacht. Später wurde das Lochkartensystem übernommen, das Joseph Jacquard für Webstühle eingeführt hatte. Babbage experimentierte auch mit unterschiedlichen Basiswerten für das Zahlensystem der Maschine, aber da er ganz auf Mechanik fixiert war, konnte er im Binärsystem (schon 1703 von Leibniz vorgeschlagen) keinen wesentlichen Vorteil sehen.

Zusammenarbeit mit Ada

Babbage arbeitete bei diesem Projekt mit der Gräfin Ada Lovelace zusammen, einer begab ten Mathematikerin. Beide hatten mit ungeheuren Schwierigkeiten zu kämpfen, nicht zuletzt mit Geldmangel. Die Gräfin verlor auch noch einen großen Teil ihres Besitzes durch ein „unfehlbares" Wettsystem für Pferderennen. Nachdem sie im Alter von 36 Jahren verstarb, setzte Babbage das Werk allein fort.

Als Mann von erstaunlicher Energie erfand er nebenher das Ophtalmoskop für die ärztliche Augenspiegelung, eine Technik für die Signalübermittlung auf See und ein Bühnen-Beleuchtungssystem; außerdem besorgte er die Choreografie für ein Ballett. In seinen letzten Lebensjahren wurde er eigensinnig: Da er den Adelstitel ‚Peer' erwartet hatte, lehnte er die Würde eines „Baron" ab, die man ihm in Anerkennung seiner Verdienste anbot.

Babbage nahm mit seinen Arbeiten das Konzept der heutigen programmgesteuerten Rechenmaschine vorweg, ohne sich der Tragweite seiner Ideen bewußt zu sein.

1792 Geboren in Totnes/Devon am 26. Dezember
1810 Geht ans Trinity College in Cambridge, um dort Mathematik zu studieren
1814 Heiratet Georgina Whitemore
1822 Veröffentlicht einen Artikel mit dem Titel ‚Überlegungen zum Einsatz von Maschinen für die Berechnung mathematischer Tafeln' und erhält die erste Goldmedaille der Astronomischen Gesellschaft, deren Mitbegründer er war
1827 Cambridge beruft ihn auf den gleichen Lehrstuhl, den einst Newton innehatte, bei 80 Pfund Gehalt jährlich. Babbage läßt sich in Cambridge jedoch nicht nieder und hält auch nie Vorlesungen
1833 Parlamentskandidat in Finsbury
1834 Aufgabe der ‚Difference Engine', nachdem sich der Mechaniker Joseph Clement zurückgezogen hat
1862 Ausstellung der teilweise fertiggestellten ‚Difference Engine' in South Kensington/London
1871 Gestorben am 18. Oktober

Perfekt im Bild

Das Betriebssystem des Spectrum bietet dem Maschinencodeprogrammierer interessante Möglichkeiten zur Steuerung des Bildschirms. Wir untersuchen die Dateien für Bildschirmdarstellung und fassen die Systemroutinen zusammen.

Die Grafik- und Bildschirmsteuerung des Spectrum ist nicht besonders kompliziert. Wir sehen uns zunächst die ROM-Routinen an, die das OS des Spectrum dafür bietet.

Um den Bildschirm adressieren zu können, muß als erstes Kanal S durch die Anwahl von Strom 2 eröffnet werden. Für einen einfachen Anzeigevorgang ist RST#10 bei weitem die einfachste Methode. Für die Darstellung eines ganzen Zeichenstrings müssen Sie jedoch die Zeichen mit einer kurzen Routine einlesen und einzeln an die RST#10-Routine übergeben.

Die entsprechende Routine ist im ROM abgelegt. Für den Aufruf laden Sie DE mit der Adresse des Strings, BC mit der Stringlänge und rufen dann die „Print String"-Routine bei #203C auf. Unser erstes Listing zeigt den Ablauf in der Praxis:

```
                ORG  60000
1172EA          LD   DE,MSG
3E02            LD   A,2              ;select stream 2
CD0116          CALL #1601            ;open channel S
1172EA          LD   DE,MSG           ;point DE to string
012600          LD   BC,38            ;load BC with string len
CD3C20          CALL #203C            ;call PRINT-STRING
C9              RET
160A06  MSG:    DEFB 22,10,6          ;codes for AT 10,5
54484520        DEFM "THE HOME COMPUTER"
160C07          DEFB 22,12,7
41445641        DEFM "ADVANCED COURSE"
```

Praktisch ist auch die ROM-Routine bei #2DE3, die eine Zahl im Fließkommaformat anzeigt. Wir haben bereits gesehen, daß die Routine bei #1C82 einen numerischen Ausdruck bewertet und das Ergebnis auf den „Calculator Stack" schiebt. Ohne diese nützliche Routine ist die Darstellung einer Zahl vom Maschinencode aus recht kompliziert. Die Zahl wird zunächst in das Registerpaar BC oder in das A-Register geladen und dann entweder mit #2D28 die STACK-A-Routine oder mit #2D2B die STACK-BC-Routine aufgerufen, die die Zahl auf den Stack schiebt. Danach brauchen Sie nur noch die Routine der Fließkommaanzeige aufzurufen.

```
                ORG  60000
013930          LD   BC,12345         ;get number in BC
CD2B2D          CALL #2D2B            ;BC onto calc stack
3E02            LD   A,2
CD0116          CALL #1601            ;open channel S
CDE32D          CALL #2DE3            ;print no from stack top
C9              RET
```

Die Vielseitigkeit der hochauflösenden Grafik des Spectrum zeigt sich jedoch erst beim Einsatz der Befehle PLOT und DRAW. PLOT hatten wir früher schon behandelt. DRAW ist ebenso einfach, benötigt aber mehr Parameter. Der Befehl zeichnet die kürzeste Linie zwischen dem zuletzt gezeichneten Punkt und einem Punkt, der relativ zum Ursprung definiert wurde. Für das Zeichen einer Linie 20 Pixel nach oben und 40 Pixel nach links lautet der Befehl ganz einfach DRAW −40,20.

Vom Maschinencode aus hat der DRAW-Befehl das Format DRAW x,y, wobei x und y separat behandelt werden müssen. Zur Vereinfachung lassen wir Kurvenfaktoren erst einmal beiseite. Wenn eine Rückkehr zum BASIC geplant ist, muß der Wert des alternativen Registerpaares H'L' erhalten bleiben, da die DRAW-Routine diese Daten verändert.

Der absolute Wert von x (ABS x) wird in das C-Register gestellt, ABS y in das B-Register und die Vorzeichenwerte von x und y entsprechend in D und E. Die Werte dürfen über −1, 0 und 1 liegen, da die Routine sie als Inkrement zur letzten geplotteten Pixelposition addiert, um die nächste Position zu berechnen.

Zwei Linientypen

Bei einem Vorzeichenwert von 2 oder 3 ist die Linie dann gepunktet, aber auch zwei- bzw. dreimal länger als bei Inkrement 1. Beachten Sie, daß die Vorzeichenwerte das Format des Zweierkomplements haben müssen. Unser drittes Beispiel zeigt, wie die beiden Linientypen gezeichnet werden.

```
                ORG  60000
D9              EXX
E5              PUSH HL               ;save H'L'
D9              EXX
0658            LD   B,88             ;Y coordinate
0E64            LD   C,100            ;X coordinate
CDE522          CALL #22E5            ;PLOT 100,88
0E28            LD   C,40             ;ABS x=40
0614            LD   B,20             ;ABS y=20
1EFF            LD   E,255            ;SGN y=-1
1601            LD   D,1              ;SGN x=1
CDBA24          CALL #24BA            ;DRAW -40,20
3E58            LD   A,88
327D5C          LD   (23677),A        ;X org=88
3E32            LD   A,50
327E5C          LD   (23678),A        ;Y org=50
0E28            LD   C,40
0614            LD   B,20
1602            LD   D,2              ;SGN x=2 - stipple line
1E02            LD   E,2              ;SGN y=2 - stipple line
CDBA24          CALL #24BA
D9              EXX
E1              POP  HL               ;restore H'L'
D9              EXX
C9              RET
```

Unsere beiden Tabellen auf Seite 2128 enthalten in Kurzform, was wir eben über Grafik und Bildschirmanzeige gesagt haben. Eine Tabelle zeigt die Routinen und die andere praktische Systemvariablen für den Einsatz der Routinen.

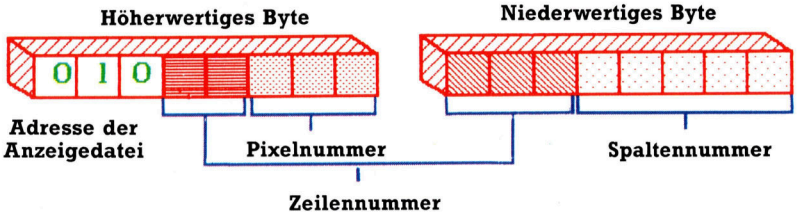
Abteilungsleiter

Der Bildschirmspeicher des Spectrum besteht aus zwei Abteilungen. Der erste Teil heißt „Anzeigedatei" und enthält Daten über jedes einzelne Pixel. Der zweite Teil ist die „Attributdatei" mit der Farbinformation. Die Anzeigedatei ist 6144 Bytes lang und erstreckt sich von 16384 bis 22527. Geben Sie das folgende Programm ein, und rufen Sie es auf:

```
10 FOR F=16384 TO 22527
20 POKE F,255
30 NEXT F
```

Der Bildschirm füllt sich langsam in drei Abschnitten. Wenn Sie nahe herangehen, können Sie erkennen, daß die Linien pixelweise gezeichnet werden, aber nicht aufeinander folgen: Die erste Pixelzeile erscheint am oberen Bildschirmrand, die nächste acht Linien darunter, die dritte sechzehn Linien unter der ersten etc. Nach der vierundsechzigsten Linie beginnt der Vorgang wieder von oben und füllt die zweite Linie, danach die neunte usw. Dieser Ablauf wiederholt sich auch im zweiten und dritten Drittel des Schirms (siehe Diagramm auf dieser Seite ganz unten).

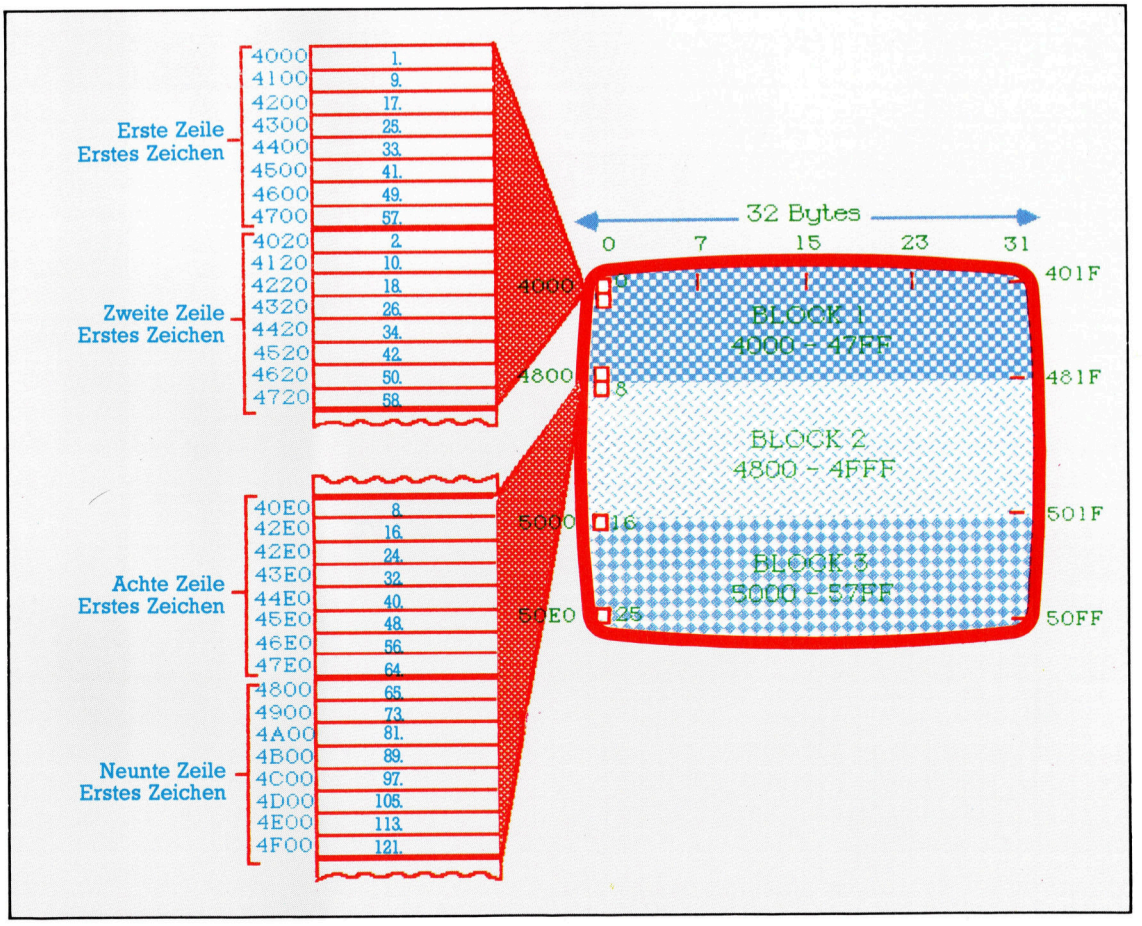
Für den Einsatz der Anzeige- bzw. Displaydatei (auch „DFile" genannt) müssen Sie berechnen, welche Adressen welche Pixelpositionen ansprechen. Das Bild zeigt, wie die Bits einer DFile-Adresse angeordnet sind. Dabei bezieht sich die Pixelnummer auf die Pixelzeile innerhalb eines Zeichenfeldes. Das oberste Pixel eines Zeichens hat die Pixelnummer 0.



Die Struktur der einzelnen Bits einer DFile-Adresse sieht zwar kompliziert aus, doch lassen sich die Bildschirmdaten damit schnell durch einzelne Registerbefehle verändern. Die Verarbeitung mit Registerpaaren wäre weit umständlicher und langsamer.

Auf die Schnelle

Wenn Sie sich in einem Zeichenfeld von einer Pixelzeile auf die darunterliegende Zeile bewegen möchten, brauchen Sie nur das Register H zu inkrementieren (INC H). Für die Bewegung um ein Zeichen nach rechts wird INC HL eingesetzt. Die Position einer PRINT-AT-Koordinate läßt sich mit der nächsten Routine berechnen. Dafür wird die Spaltennummer in E und die Zeilennummer in D gespeichert. Nach Ablauf der Routine steht die gewünschte Adresse in HL.



So ist die Anzeigedatei des Spectrum aufgebaut. Der Bildschirm unterteilt sich in drei Blöcke, die je 256 Zeichenfelder (acht Zeilen zu je 32 Zeichen) darstellen. In jedem Block sind die Pixelzeilen wie im Bild gezeigt untergebracht: Der erste Block von 256 Bytes enthält die Daten der ersten Pixelzeile der ersten 256 Zeichenfelder, die zweiten 256 Bytes die Daten der zweiten Pixelzeile etc. Im vergrößerten Bildausschnitt sind die vier Zeichen auf der linken Bildschirmseite zu erkennen. Die Pixelzeilen jedes Zeichenfeldes sind durch die entsprechenden Adressen der Anzeigedatei festgelegt.

Spectrum Variable

Name	Adresse	Beschreibung
COORDS	23677	Enthält X-Koordinate des zuletzt geplotteten Punktes
COORDS	23678	Enthält Y-Koordinate des zuletzt geplotteten Punktes
DF CC	23684	Eine Zwei-Byte-Variable mit der Speicheradresse des ersten Anzeigedateibytes, das angezeigt werden soll.
S POSN	23688	Die aktuelle Anzeigeposition für Spalten
S POSN	23689	Die aktuelle Anzeigeposition für Zeilen
ATTR P	23693	Speichert die permanenten Farben, die von INK, PAPER, FLASH und BRIGHT gesetzt wurden. Die Daten haben das ATTR-Format
MASK P	23694	Maskiert ATTR P, wenn transparente Farben (INK 9 etc.) eingesetzt werden. Für jedes gesetzte Bit wird das entsprechende Bit von ATTR P ignoriert.
ATTR T	23695	Wie ATTR P, jedoch nur für temporäre Farben – z. B. bei Farbänderungen innerhalb eines PRINT-Befehls
MASK T	23696	Wie MASK P, jedoch für temporäre Farben
P FLAG	23697	Verschiedene Flags für den Status der temporären und permanenten Farben. Jedes Bit zeigt, ob INK oder PAPER temporär oder permanent ist.

ATTR P, MASK P und P FLAG haben wir früher schon ausführlich beschrieben.

Spectrum ROM Routinen

Name	Adresse	Einsprungsbedingungen	Ergebnis
RST #10	#0010	In A muß der ASCII-Code des anzuzeigenden Zeichens stehen	Das Zeichen wird angezeigt
PR STRING	#203C	IN BC muß die Stringlänge stehen und in DF die Position des Strings im Speicher	Der String wird auf der nächsten verfügbaren Schirmposition angezeigt
PRINTED FP	#2DE3	Die anzuzeigende Zahl muß oben auf dem Calculatorstack liegen	Die oberste Zahl des Calculatorstacks wird angezeigt
STK-A	#2D28	A muß den Wert enthalten, der auf den Stack geschoben werden soll	Der Wert von A wird oben auf den Calculatorstack geschoben
STK-BC	#2D2B	BC muß den Wert enthalten, der auf den Stack geschoben werden soll	Der Wert von BC wird auf Calculator geschoben
PLOT	#22E5	In B und C müssen die Y- und X-Koordinaten gespeichert sein	Der Punkt (x,y) wird geplottet und SV COORDS aktualisiert
DRAW	#24BA	B muß ABS (y) enthalten C muß ABS (x) enthalten D muß SGN (y) enthalten E muß SGN (x) enthalten	Wenn die Werte über 1 liegen, wird eine gepunktete Linie gezeichnet
CHAN-OPEN	#1601	A muß die Nummer des zu eröffnenden Streams enthalten	Eröffnet den angegebenen Kanal

```
                ORG  60000
3EAF            LD   A,175
92              SUB  D
57              LD   D,A            ;D holds 175 - y coord
A7              AND  A
1F              RRA
37              SCF
1F              RRA
A7              AND  A
1F              RRA                 ;shuffle bits along
AA              XOR  D
E6F8            AND  248
AA              XOR  D
67              LD   H,A            ;H becomes...
7B              LD   A,E            ;64+8*INT(B/64)+B(MOD 8)
CB07            RLC  A
CB07            RLC  A
CB07            RLC  A
AA              XOR  D
CB07            RLC  A
CB07            RLC  A              ;shuffle bits along
6F              LD   L,A            ;L now has 32*INT(...
7B              LD   A,E            ;...B(MOD64)/8)+INT(x/8)
E607            AND  7              ;lo-byte is x(MOD 8)
C9              RET
```

Pixeladressen können Sie jedoch nur mit der folgenden Routine berechnen. Dabei wird wie beim vorigen Listing D mit der Y-Koordinate und E mit der X-Koordinate geladen. Nach Ablauf der Routine steht die Adresse in HL. Sie enthält einen Wert zwischen 0 und 7, der dem gesetzten Pixelbit entspricht.

```
                ORG  60000
7A              LD   A,D
E6F8            AND  248
CBF7            SET  6,A
67              LD   H,A
7A              LD   A,D
E607            AND  7              ;mask off bits 3-7
CB0F            RRC  A
CB0F            RRC  A
CB0F            RRC  A              ;move bits up
83              ADD  A,E            ;add on column bits
6F              LD   L,A
C9              RET
```

Denken Sie beim DFile auch daran, daß jedes mit POKE abgerufene Byte sich auf acht Pixel bezieht. Die Pixelbytes haben das gleiche Format wie die der anwenderdefinierten Grafik.

Die Attributdatei – der zweite Teil des Bildschirmspeichers – ist einfacher aufgebaut als die Anzeigedatei. Die Datei ist 768 Bytes lang und fängt bei 22528 an. Mit dem Programm

```
10 FOR F=22528 TO 22528 + 767
20 POKE F,0
30 NEXT F
```

können Sie sehen, daß der Bildschirm sich Zeile für Zeile füllt. Hier wird allerdings nach Abschluß der ersten Zeile gleich das erste Byte der zweiten Zeile angesprochen und Zeile für Zeile fortgefahren.

Jedes Byte der Attributdatei bezieht sich auf die INK-, PAPER-, BRIGHT- und FLASH-Werte der einzelnen Zeichenfelder. Die ersten drei Bits enthalten die Farbe von INK, Bit 3, 4 und 5 die von PAPER. Bit 6 gibt Auskunft über BRIGHT und Bit 7 über FLASH.

Zum Schluß noch ein Hinweis: Für eine Änderung der BORDER-Farbe vom Maschinencode aus müssen Sie A mit der Farbnummer laden und #229B aufrufen.

Fachwörter von A bis Z

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.

POKE = POKE
Der Befehl POKE erlaubt es, in einer höheren Programmiersprache, etwa in BASIC, eine vorzeichenlose Zahl auf eine beliebige Adresse (meist ein Register) zu laden. Dazu gibt man nacheinander POKE, die Adresse und schließlich die gewünschte Zahl ein.

POKE ist sehr vielseitig verwendbar; es wird vorwiegend zur Ausführung maschinennaher Operationen benutzt, für die es in der höheren Sprache keine entsprechenden Befehle gibt. Der Commodore 64 verfügt beispielsweise in seinem BASIC nicht über Kommandos, die die weitreichenden Ton- und Grafikmöglichkeiten unterstützen. Die Hardware läßt sich daher nur richtig nutzen, indem bestimmte VIC (Video)- und SID(Klang)-Bausteinadressen mit den entsprechenden Parametern geladen werden.

Polish Notation =
Polnische Notation
Von dem polnischen Mathematiker Jan Lukasiewicz stammt der Vorschlag, bei arithmetischen Verknüpfungen den Operator nicht zwischen, sondern vor die Operanden zu setzen (daher auch „Präfix"-Schreibweise, z. B. +xy statt wie üblich x+y). Die polnische Notation kommt der Arbeitsweise des Computers entgegen, weil dort auch zuerst der Operator (+) und dann erst die Zahlenwerte verlangt werden. Das beschleunigt die Fehlerkontrolle und die Abwicklung der Arithmetik: Nach der Decodierung des Operationsteils ‚+' weiß die CPU, daß zwei Werte folgen, die zu addieren sind.

Verwandt mit dieser Schreibweise ist die von FORTH bekannte umgekehrte polnische oder „Postfix"-Notation. Dabei steht der Operator hinter den Operanden.

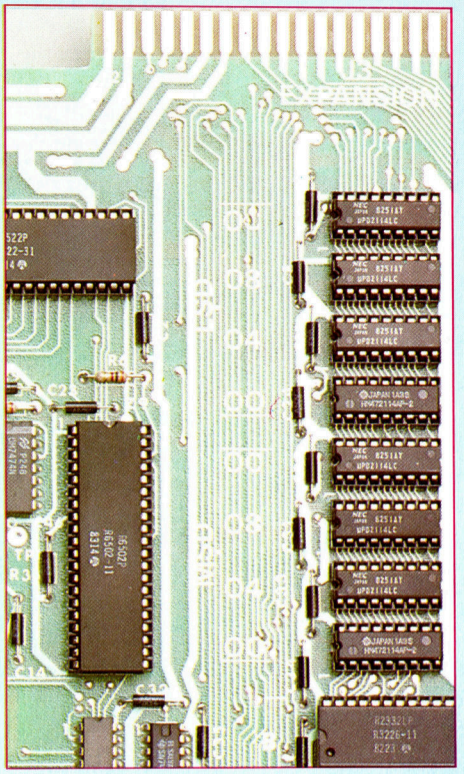
Bildnachweise
2101 Tony Sleep
2102, 2106, 2127 Caroline Clayton
2103, 2104, 2118, 2119, 2123 Kevin Jones
2120, 2121, 2125 Liz Heany
U 3 Chris Stevens

Polynomial = Polynom
Ein Polynom ist eine mathematische Funktion y=f(x) in Gestalt einer Summe von Potenzen der Variablen x, die jeweils noch mit einer Konstanten multipliziert sind, z. B.
$y=5x^3 + 2x^2 + 2x + 8$
Polynomdarstellungen spielen in der numerischen Mathematik und daher auch in der Datenverarbeitung eine wesentliche Rolle, weil sich damit andere Funktionen in beliebig genauer Näherung berechnen lassen.

Port = Anschlußstelle
Die Peripherieanschlüsse eines Rechners für die Ein- oder Ausgabe von Daten werden auch als Port bezeichnet. Die externen Ports hängen im allgemeinen nicht unmittelbar am Prozessor-Chip, sondern es sind spezielle Schnittstellen-Bausteine zur Signalumsetzung zwischengeschaltet. Diese laufen unter Namen wie „PIA" (Peripheral Interface Adaptor = Peripherieschnittstellen-Anpassung) und vermitteln zwischen der CPU und den Peripheriegeräten etwa durch Parallel/Serien-Wandlung oder durch Baudratenangleich bei der Übertragung von Daten.

Printed Circuit =
Gedruckte Schaltung
Eine ‚gedruckte' Schaltung ist das Leiterbahnensystem zur Herstellung elektrischer Verbindungen zwischen einzelnen Bauelementen auf einer Isolierstoff-Trägerplatte. Dabei sind die Leiterbahnen aber nicht aufgedruckt, sondern aus einer flächigen Kupferkaschierung mit lichtempfindlicher Beschichtung herausgeätzt, nachdem das Leiterbahnenbild darauf optisch übertragen, entwickelt und fixiert wurde. In die Platine lassen sich Löcher für die ‚Pins' der Bauelemente bohren, die anschließend durch Bestückungsmaschinen automatisch aufgesteckt und montiert werden können. Bei doppelseitigen Platinen werden auf Vorder- und Rückseite Leiterbahnen angeordnet und punktweise „durchkontaktiert", das heißt leitend verbunden.

Ohne gedruckte Schaltungen wäre die Massenproduktion von Geräten wie Transistorradios oder auch Computern wohl kaum denkbar. Bei neueren Entwicklungen werden sog. ‚Multilayer'-Platinen eingesetzt, die durch Verkleben mehrerer dünner einfacher Leiterplatten und Durchkontaktieren realisiert wurden.



Die technologische Entwicklung bei den gedruckten Schaltungen führt zu immer feineren Leiterbahnen, bis in den Zehntelmillimeterbereich. Die Umsetzung von Schaltplänen in Leiterbahnenmuster (‚Entflechtung') erfolgt heute durchweg mit CAD-Verfahren am Bildschirm.

+ Vorschau +++ Vorschau +++ Vorschau +++ Vorschau +++ Vorschau +

computer kurs

Eine Nummer größer
ist der Mini-Computer von VAX. Er verfügt immerhin über 8 MByte RAM.

Blätterrauschen
Im BASIC-Kurs geht es diesmal um Struktur und Anwendung von „Spreadsheets", den idealen Hilfsmitteln.

Funktionsfähig
MS-DOS-Funktionen können von der Assemblersprache des 8088 und von Hochsprachen angesprochen werden.

Musik und MIDI
Im Selbstbau-Kurs geht es um die Konstruktion eines MIDI-Interfaces für die Schneider CPC-Computer.