Einsteigen - Verstehen - Beherrschen

DM 3,80 öS 30 sfr 3,80

computer kurs

Ein wöchentliches Sammelwerk

Chipgesteuerte Kameras

Ausbau-Computer: Der AIM 65

Spannendes Abenteuerspiel

Hybrids Musikmaschine

Pixelzeichnungen

Heft 35

computer kurs

Heft 35

Inhalt

WIE SIE JEDE WOCHE IHR HEFT BEKOMMEN

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

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

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

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

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

Abonnement

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

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

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

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

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

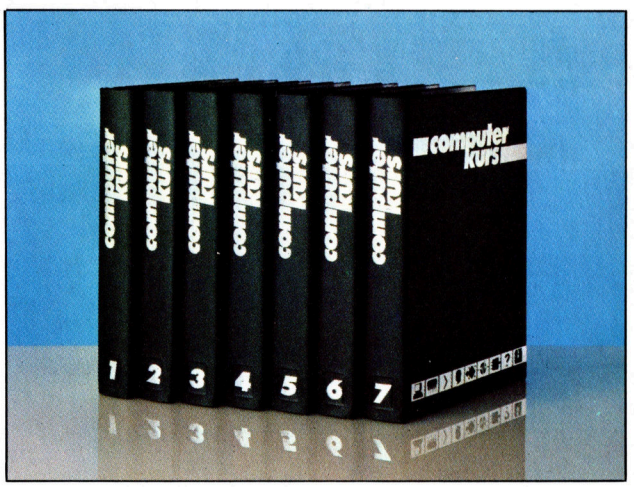
Schweiz: Der Sammelordner kostet sfr 15. Bitte wenden Sie sich an Ihren Kiosk; dort werden Sie jederzeit die gewünschten Exemplare erhalten.

INHALTSVERZEICHNIS

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

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

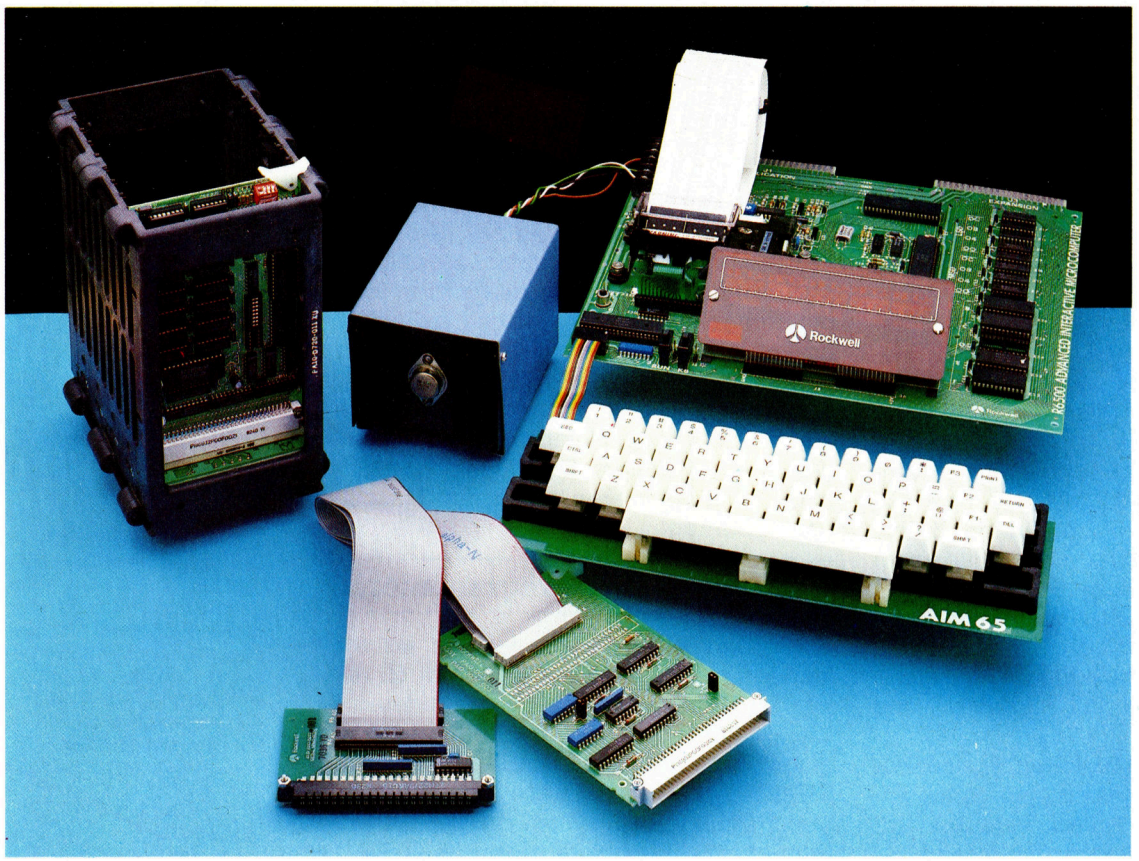
Vertrieb: Marshall Cavendish International Ltd., Heidenkampsweg 74, 2000 Hamburg 1



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

Platinencomputer

Nicht alle Microcomputer sind in sorgfältig gestylten Gehäusen untergebracht. Es gibt eine ganze Reihe von Maschinen, die in Form von einzelnen Platinen verkauft werden. Eins der flexibelsten Geräte dieser Art ist der Advanced Interactive Microcomputer von Rockwell – besser bekannt als AIM 65 –, der als Lernhilfe und zur Unterstützung der Systementwicklung konzipiert wurde.



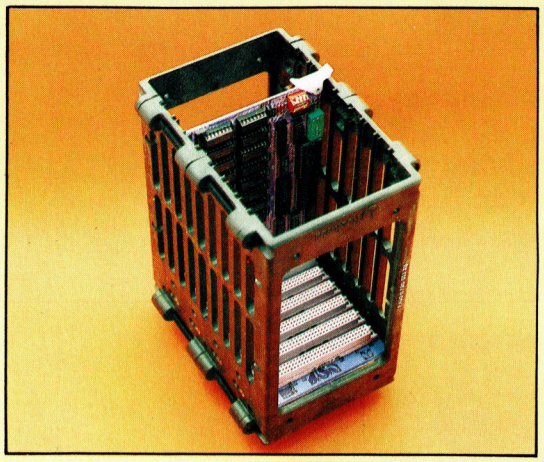
Auch ein vollständiges AIM-System macht einen recht unfertigen Eindruck, obwohl Hauptplatine und Tastatur mit Gehäuse geliefert werden. Eine Zusatzkarte läßt sich direkt auf die Rückseite des AIM aufstecken. Darüber hinausgehende Erweiterungen müssen in einem Extragehäuse untergebracht werden, das per Flachkabel und den zwei im Vordergrund gezeigten Platinen an den Computer angeschlossen wird. Der blaue Metallkasten enthält die Stromversorgung.

In seiner einfachsten Version besteht der AIM 65 nur aus einer nackten Platine, ohne jegliches Gehäuse. Seine Fähigkeiten sind jedoch bemerkenswert. Es ist die einzige Entwicklungsmaschine, die einen eingebauten Drukker besitzt. Die meisten Geräte dieser Art sind weitaus weniger großzügig ausgelegt und besitzen weder eine echte Tastatur, noch die LED-Anzeige mit 20 Zeichen zu je 16 Segmenten, mit der AIM-Besitzer einen Einblick in die Vorgänge im Inneren der Maschine erhalten.

Diese Eigenschaften sind zwar nicht unbedingt notwendig, doch ist der übliche Maschinentyp mit Hexadezimaltastatur, einer Anzeige mit acht Zeichen zu je sieben Segmenten und ohne Drucker weitaus schwerer zu bedienen. In der Grundausstattung von Ein-/Ausgabekanälen und Speicher entspricht der AIM mit seinen vier KByte RAM und zwölf KByte ROM dem Standard.

Im Vergleich mit kommerziellen Maschinen und Heimcomputern, die standardmäßig über 16 KByte und mehr RAM verfügen, scheinen vier KByte sehr mager zu sein. Der AIM 65 hat jedoch zwei solide Steckleisten, über die sich weitere Platinen wie die 32 KByte RAM-Erweiterung anschließen lassen. Entwicklungssysteme werden außerdem zumeist für Anwendungen eingesetzt, die keine großen Speicherkapazitäten erfordern.

Da der AIM 65 auf dem Microprozessor 6502 aufbaut, gibt es in der Speicherunterteilung feste Grenzen. Die 512 RAM-Bytes von $0100 bis $01FF enthalten den Maschinenstack, während die Zero-Page von $0000 bis $00FF das „Pseudo-Register" des 6502 darstellt. Da Maschinenbefehle dieser Konstruktion keine Seitenadresse (binär 00000000) zu enthalten brauchen, sind sie kürzer und damit in der Ausführung weitaus schneller als normale Register.



Speichererweiterung
Der AIM läßt sich fast unbegrenzt erweitern. Das im Bild gezeigte Erweiterungsgehäuse kann mit acht zusätzlichen Platinen bestückt werden (es gibt aber auch Versionen für vier oder sechzehn Platinen). Angeboten werden Karten für die Speichererweiterung (bis zu 128K RAM und ROM), eine IEEE-Standard-Schnittstelle und Platinen, mit denen sich der AIM an Diskettenstationen, Monitore und Drucker anschließen läßt.

Anwendungssteckleiste
Diese Steckleiste hat zwei Eingänge, über die sich Peripheriegeräte steuern lassen.

Stromversorgung
Hier werden die Stromkabel angeschlossen. Der AIM benötigt 5V für den Computer und 24V für den Drucker.

Drucker
Der eingebaute Thermodrucker des AIM ist sehr ungewöhnlich für Geräte dieser Kategorie. Mit seinen zwanzig Spalten eignet er sich für alle Anwendungen, bei denen die Vorgänge protokolliert werden müssen.

Reset-Schalter
Damit wird der Warmstart des Computers ausgelöst.

Diese Gegebenheit ist der Schlüssel zu der Flexibilität und Popularität des Chips: Dem Anwender stehen 256 frei definierbare Register zur Verfügung.

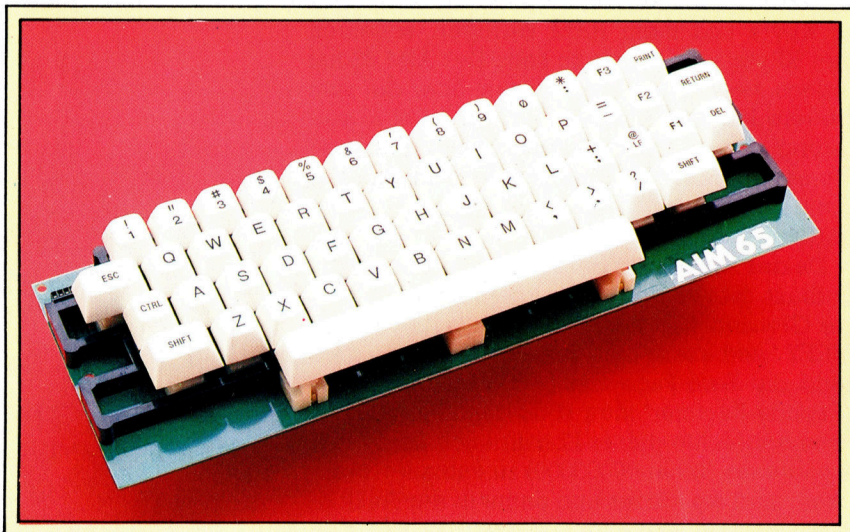
Für das Gerät existiert eine Reihe von Sprachen und Hilfsprogrammen. Wichtigstes Modul ist das Monitor-ROM, ein Block von vier KByte, der von $E000 bis $EFFF reicht und Systemmodule in „maschinennaher" Sprache enthält. Im Monitor-ROM befinden sich ein Zeileneditor, eine Ein-Schritt-Mechanik (die das Programm nach jedem ausgeführten Befehl anhält, damit der Speicherinhalt untersucht und gegebenenfalls geändert werden kann), ein Tracer (der während des Programmablaufs Schritt für Schritt den Inhalt des Programmzählers anzeigt) und schließlich die üblichen Veränderungsmöglichkeiten für Register und Speicherinhalte.

Die Maschine wird mit fünf Stecksockeln für ROMs geliefert, über die sich eine ganze Palette von festprogrammierten Modulen anschließen läßt. Unter anderem gibt es eine BASIC-Version mit allen Hauptfunktionen, Fünf-Byte-Zahlen im Fließkommaformat und einzelig festgelegten arithmetischen Funktionen.

Für den typischen Käufer des AIM 65, der an der Steuerung von Maschinenprozessen und ähnlichen Anwendungen interessiert ist, sind die weiteren ROM-Sets jedoch weitaus interessanter. So gibt es einen Assembler und die bemerkenswerte, aber wenig bekannte Sprache PL/65, die ALGOL bzw. PL/1 ähnlich ist. Sie erzeugt bei der Compilierung den Quellentext des Assemblers. Dieser Text läßt sich anschließend optimieren und in den Maschinencode übertragen.

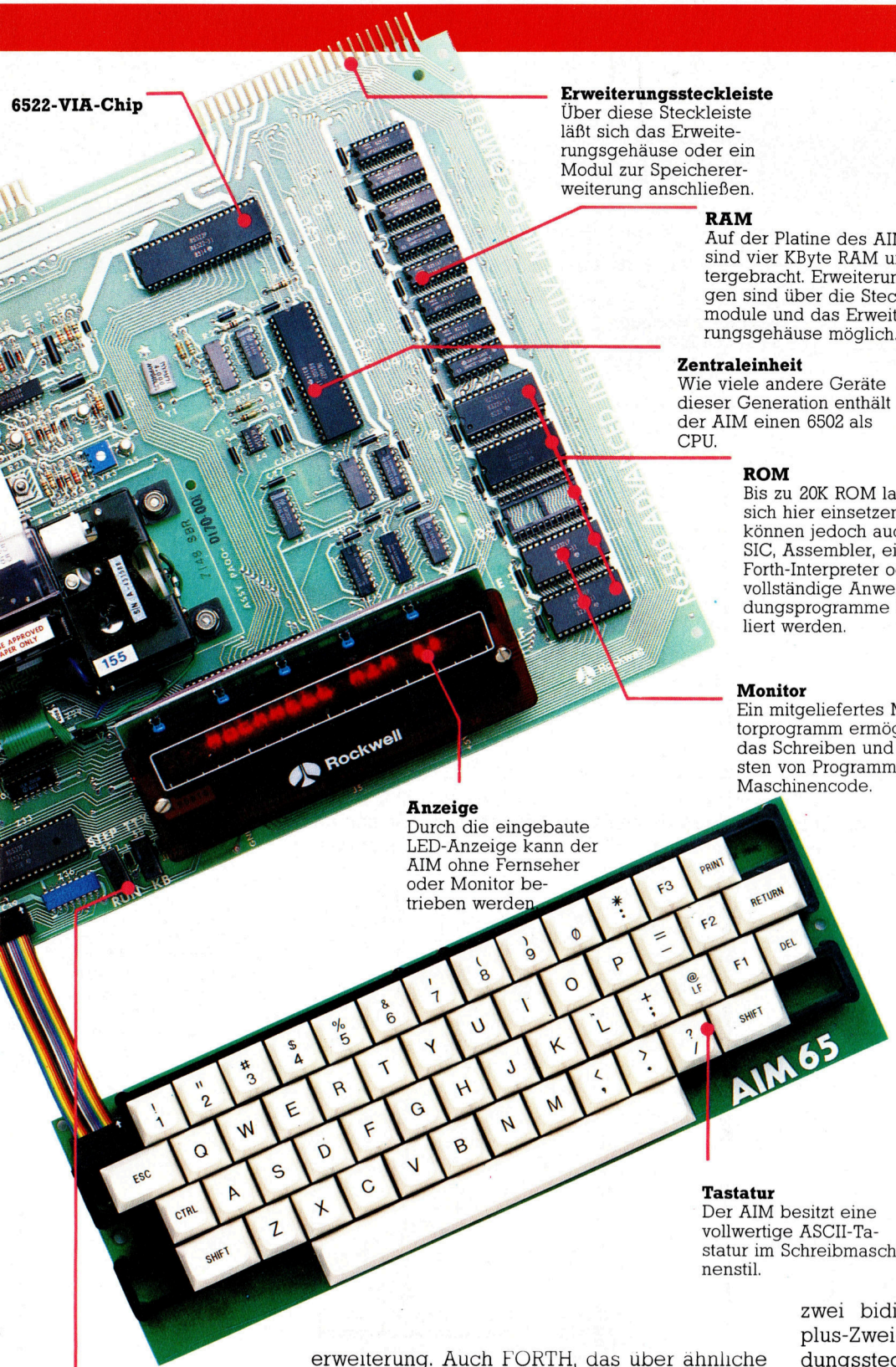
Strukturiertes Programmieren

Auch INSTANT PASCAL – eine wenig bekannte Sprache – ist auf dem AIM 65 verfügbar. Im Gegensatz zu fast allen anderen PASCAL-Dialekten ist sie interaktiv, muß allerdings vom Interpreter bearbeitet werden. INSTANT PASCAL bietet außer den Vorteilen der guten Strukturierung auch die bequeme Ausführung und Flexibilität des BASIC. Da PASCAL jedoch eine umfangreiche Sprache ist, läuft sie auf dem AIM 65 nicht ohne Speicher-



Tastatur
Der AIM ist mit einer guten Tastatur ausgestattet, über die sich das Gerät leicht in BASIC und anderen Sprachen programmieren läßt. (Viele andere Entwicklungssysteme besitzen nur Hexadezimaltasten (0–9, A–F) und sind daher weniger bequem zu bedienen.) Wenn die PRINT-Taste rechts oben im Tastenfeld gleichzeitig mit der CTRL-Taste gedrückt wird, schaltet sich der Drucker ein, und alle angezeigten Zeichen werden auf Papier ausgegeben.

Tastatursteuerung
Ein 6532-RIOT-Chip decodiert die Tastatureingaben.



AIM 65

ABMESSUNGEN
292 x 267 x 60 mm

ZENTRALEINHEIT
6502

SPEICHERKAPAZITÄT
4 K ROM, erweiterbar auf 20K; 4 K RAM, erweiterbar auf 64K.

BILDSCHIRM-DARSTELLUNG
Kein Bildschirm, jedoch eine LED-Anzeige mit 20 Zeichen zu je 16 Segmenten. Für den Anschluß eines Monitors wird eine Zusatzplatine benötigt.

SCHNITTSTELLEN
Zwei acht-Bit-bidirektionale Eingänge mit je zwei Steuerleitungen plus Systembus.

PROGRAMMIERSPRACHEN
Ein Miniassembler und ein Zeileneditor werden mitgeliefert. Ein vollwertiger Assembler, BASIC, FORTH, PL/65 und INSTANT PASCAL sind verfügbar.

TASTATUR
53 Schreibmaschinentasten, darunter drei Funktionstasten.

HANDBÜCHER
Die Handbücher für Installation und Hardware sind vorbildlich. Sie enthalten alle Informationen, die der Anwender benötigt.

STÄRKEN
Der größte Vorteil der Maschine ist die extreme Vielseitigkeit, die durch Erweiterungsmöglichkeiten und den Zugang zu mehreren Programmsprachen gegeben ist. Die Handbücher sind ausgezeichnet.

SCHWÄCHEN
Das Gerät hat keine wesentlichen Schwächen. In den Hochsprachen besteht zwar ein Mangel an Software, doch wurde der AIM 65 eher als Entwicklungssystem und weniger als Anwendungsmaschine konzipiert.

erweiterung. Auch FORTH, das über ähnliche Fähigkeiten verfügt, wird für den AIM angeboten. Es kommt jedoch mit weniger Platz aus als PASCAL und läuft daher ohne Speichererweiterung.

Außer der Steckleiste für Erweiterungen, die alle Hauptsignale der Daten- und Adreßleitungen, Taktsignale und Stromzuführungen enthält, gibt es noch zwei 6522-Versatile-Interface-Adapter-Chips. Einer steuert den Drucker, die Teletype-Schnittstelle und das Cassettenrecorderinterface. Der zweite hat keine feste Aufgabe und wurde in Form von zwei bidirektionalen Anschlüssen im Acht-plus-Zwei-Leitungsformat auf die Anwendungssteckleiste J2 gelegt. Außerdem gibt es den komplexen und hochentwickelten 6532-RAM-I/O-Timer (RIOT), der jedoch nur die Tastatur steuert.

Insgesamt ist der AIM 65 ein robuster, flexibler und gut durchkonstruierter Einplatinencomputer mit ausgezeichneten Ausbaumöglichkeiten. Seine Fähigkeiten machen ihn besonders für denjenigen interessant, der eine kleine, aber brauchbare Maschine benötigt, die zwar nicht über die Eigenschaften eines kommerziellen Gerätes verfügt, jedoch flexibler ist als ein normaler Heimcomputer.

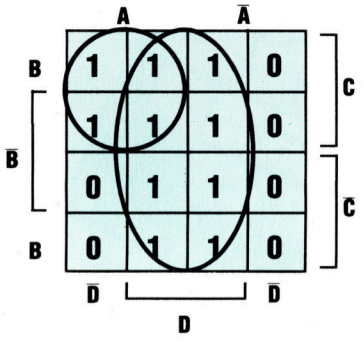
Schaltpläne

Den Nutzen von Karnaugh-Tafeln bei der Vereinfachung Boolescher Ausdrücke mit zwei oder drei Variablen kennen Sie bereits. Jetzt sind die schwierigen Fälle mit vier Variablen an der Reihe. Außerdem werden Sie mit der Anwendung von Karnaugh-Tafeln im Schaltungsentwurf vertraut gemacht.

Vier Variablen: Eine Karnaugh-Tafel für eine Booleschen Ausdruck mit vier Variablen wirkt auf den ersten Blick etwas kompliziert. Aber keine Angst, mit den schon bekannten Regeln läßt sich auch eine solche Tafel problemlos bearbeiten. Nehmen Sie einmal an, daß Sie diesen Ausdruck vereinfachen sollen:

$$AB\overline{C}\overline{D} + ABCD + \overline{A}BCD + A\overline{B}\overline{C}\overline{D} + \overline{B}CD + A\overline{B}CD + \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}CD + B\overline{C}D$$

Dafür brauchen Sie eine k-Tafel für vier Variablen. Weil der Ausdruck aber aus acht Elementen besteht, müssen insgesamt zehn Einsen in die Tafel eingetragen werden (sowohl $\overline{B}CD$ als auch $B\overline{C}D$ stellen je zwei Fälle dar). Hier die k-Tafel:



Die k-Tafel zeigt eine zentrale Gruppe von acht Einsen, die für jede mögliche Kombination mit D stehen. Eine zweite Gruppe von vier Einsen in der linken oberen Ecke stellt alle Fälle mit A AND C dar. Der Ausdruck läßt sich also zu A AND C OR D (A.C + D) vereinfachen.

Ausdruck verändern

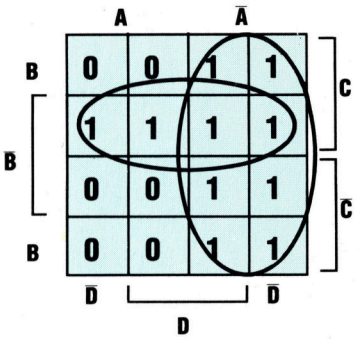
Gelegentlich muß ein Ausdruck verändert werden, damit er sich als k-Tafel schreiben läßt. Hier ein Beispiel:

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

Die entsprechende Tafel kann nur nach Anwendung des Gesetzes von de Morgan erstellt werden. Der Ausdruck heißt dann:

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

Und das ist die zugehörige Karnaugh-Tafel:



Mit der Tafel finden Sie die Vereinfachung zu $\overline{A} + \overline{B}.C$. Ein zweites Mal mit de Morgans Gesetz behandelt, ergibt sich schließlich dieser Ausdruck:

$$\overline{A\,(B+\overline{C})}$$

Beispiel 1: Dreißig Tage

Sie kennen sicher den Trick, durch Abzählen an den Fingerknöcheln zu bestimmen, ob ein Monat dreißig oder einunddreißig Tage hat. Aber wie sieht eine elektronische Schaltung für diese Bestimmung aus? Dazu ordnen wir jedem Monat einen Vier-Bit-Code zu – von 0001 für den Januar bis 1100 für den Dezember. Nach Eingabe dieses Codes soll die Schaltung für Monate mit dreißig Tagen eine 1 ausgeben.

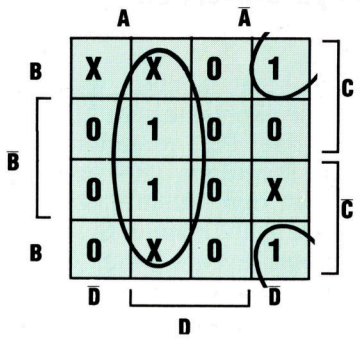
Die Wertetabelle einer solchen Schaltung:

Monat	Eingaben				Ausgabe
	A	B	C	D	S
	0	0	0	0	X
JAN	0	0	0	1	0
FEB	0	0	1	0	0
MÄR	0	0	1	1	0
APR	0	1	0	0	1
MAI	0	1	0	1	0
JUN	0	1	1	0	1
JUL	0	1	1	1	0
AUG	1	0	0	0	0
SEP	1	0	0	1	1
OKT	1	0	1	0	0
NOV	1	0	1	1	1
DEZ	1	1	0	0	0
	1	1	0	1	X
	1	1	1	0	X
	1	1	1	1	X

Das Ergebnis X ist in der Wertetabelle für ungültige Eingaben vorgesehen, wir gehen aber davon aus, das solche Eingaben nicht erfolgen. Für alle Fälle, in denen S=1 ist, läßt sich aus der Tabelle dieser Ausdruck formen:

$$S = \overline{A}.B.\overline{C}.\overline{D} + \overline{A}.B.C.\overline{D} + A.\overline{B}.\overline{C}.D + A.\overline{B}.C.D$$

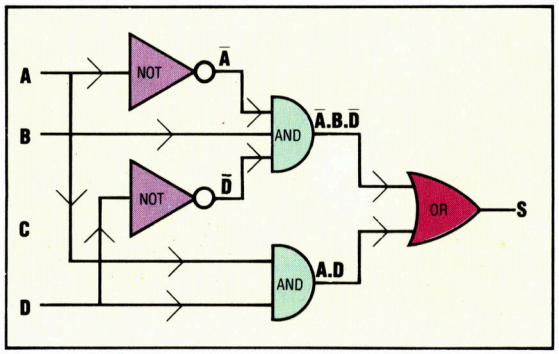
Zusammen mit den ungültigen Eingaben (X) ergibt sich die k-Tafel:



Die Tafel hilft beim Vereinfachen des Ausdrucks auf

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

Danach läßt sich die Schaltung für eine „Dreißig-Tage-Meldung" entwerfen:



Beispiel 2: Ungerade Zahlen

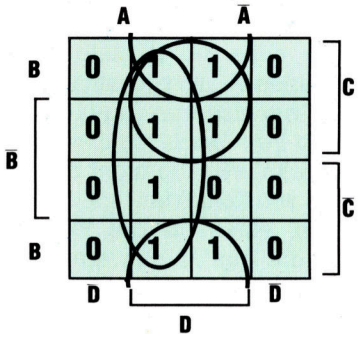
Dezimal-zahl	Eingaben A	B	C	D	Ausgabe S
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	1

Die Zahlen 0 bis 15 können auch durch vier Binärziffern (0000 bis 1111) dargestellt werden. Die geplante Schaltung soll bei einer im Binärcode eingegebenen ungeraden Zahl, die größer als zwei ist, eine 1 ausgeben.

Der erste Schritt ist das Aufstellen der Wertetafel für alle möglichen Fälle. Wieder muß für alle Fälle, in denen S wahr (=1) ist, der entsprechende Boolesche Ausdruck formuliert werden:

$$S = \overline{A}.\overline{B}.C.D + \overline{A}.B.\overline{C}.D + \overline{A}.B.C.D + A.\overline{B}.\overline{C}.D + A.\overline{B}.C.D + A.B.\overline{C}.D + A.B.C.D$$

Daraus entsteht diese k-Tafel:



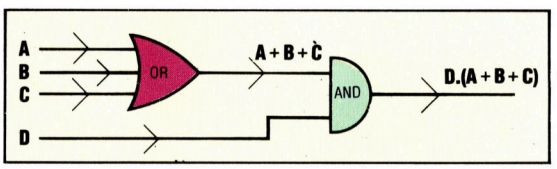
Aus der Tafel können Sie drei Vierergruppen isolieren. Dieser Ausdruck beschreibt sie:

$$S = A.D + C.D + B.D$$

Vereinfacht durch Anwendung des Distributiv-Gesetzes wird daraus:

$$S = D.(A + B + C)$$

Hier die dazugehörige Schaltung:



Der nächste Kursabschnitt wird Ihre Kenntnisse durch Wiederholung der wichtigsten Regeln und viele interessante Übungsaufgaben weiter vertiefen.

Übung 5

1) Vereinfachen Sie die mit Hilfe der Karnaugh-Tafeln folgende Boolesche Ausdrücke:

a) $\overline{A}.B.C + A.\overline{B}.\overline{C}. + \overline{A}.\overline{C} + \overline{A}.B.C + A.B.\overline{C}$

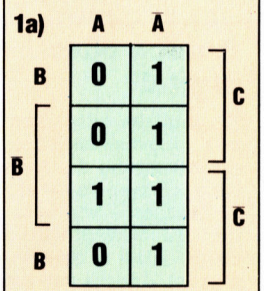
b) $\overline{B + C} + \overline{B}.\overline{C} + A.C$

c) $A.\overline{B}.D + \overline{A}.D + A.B.C.D + A.B.\overline{C} + \overline{A}.B.\overline{C}.\overline{D}$

2) Entwerfen Sie eine Schaltung für die Eingabe der Binärdarstellung von Ziffern zwischen 0 und 7. Die Schaltung soll eine 1 ausgeben, wenn die Eingabe entweder eine ungerade Zahl oder ein Vielfaches von 3 ist (3 oder 6). Vereinfachen Sie die Logikfunktion mit der Wahrheitstabelle und den k-Tafeln.

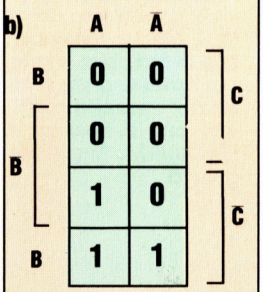
Lösungen der Übung 4

1a)



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

b)



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

Steuerung mit einem Griff

Das zweimotorige Auto wird diesmal mit einem Joystick gesteuert.

Der Commodore 64 und der Acorn B haben zwei ganz unterschiedlich konstruierte Joysticks: Beim Commodore finden wir den Standardtyp mit vier integrierten Schaltern für die unterschiedlichen Richtungen und einen Feuerknopf – alles auf digitaler Basis. Der Acorn B hat einen Analog-Joystick. Allerdings kann auch ein modifizierter Digital-Joystick am Acorn Analog-Port angeschlossen werden. Analoge Joysticks arbeiten nicht durch Öffnen und Schließen von Kontakten, sondern mit zwei Potentiometern, eines für die Rechts/Links- und eines für die Auf/Ab-Bewegung.

Commodore 64

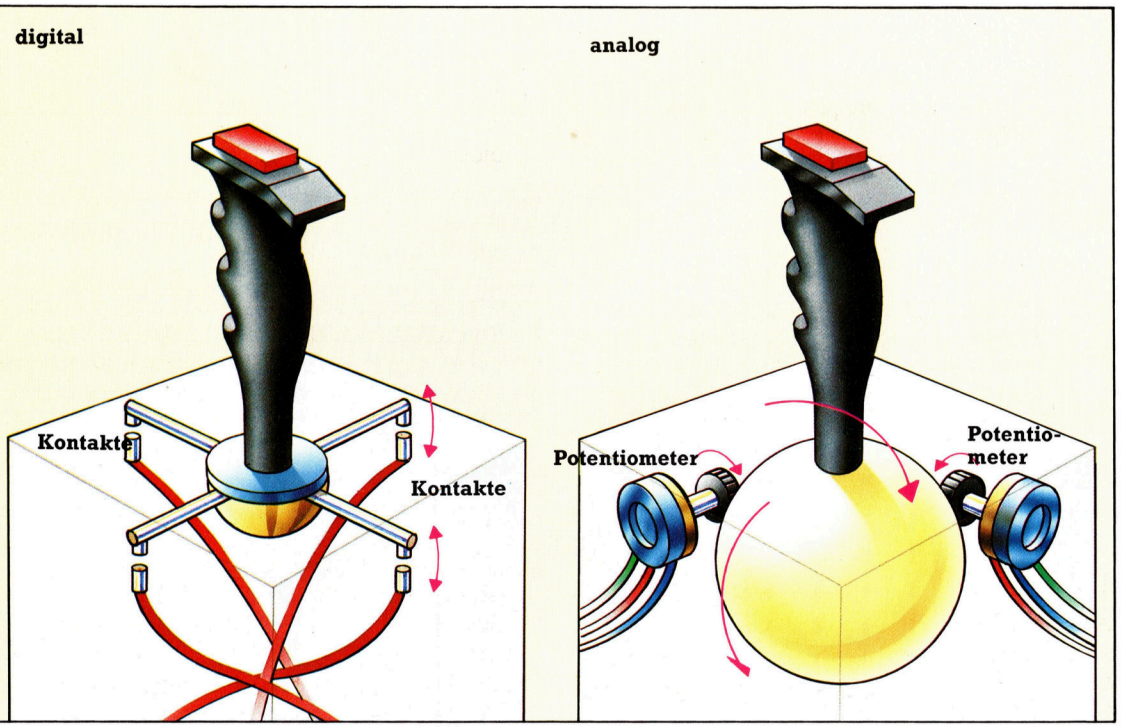
Der Commodore hat zwei Joystick-Buchsen. Bei den folgenden Programmen und Erläuterungen wird auf den Anschluß 2 Bezug genommen, verbinden Sie Ihren Joystick also mit dem Port direkt neben dem Ein/Aus-Schalter. Beim Drücken des Handgriffs nach vorn wird einer der vier eingebauten Schalter geschlossen. Die Daten des Port 2 befinden sich in der Speicheradresse 56320. Durch Schließen eines Joystick-Kontaktes geht ein Bit dieses Speicherplatzes auf „Low", ähnlich wie sich durch Schließen eines Kontaktes am Ausgangsbuffer die Bits am User Port verändern. Geben Sie das folgende Programm ein – es zeigt wiederholt den Inhalt des Datenregisters an. Wenn das Programm läuft, können Sie die Veränderung der Registerwerte beim Bewegen des Joysticks bzw. beim Drücken des Feuerknopfs auf dem Bildschirm beobachten.

```
10 REM **** C 64 JOYSTICK—ABFRAGE ****
20 PORT2=56320
30 JOY=PEEK(PORT2):GOSUB500
40 PRINT CHR$(145),JOY, B$
50 GOTO30
60:
500 REM BINAERUMSETZUNG S/R
510 B$="":N=JOY
520 FOR D=1 TO 8
530 N1=INT(N/2):R=N—2*N1
540 B$=B$+STR$(R) N=N1
550 NEXT D
560 RETURN
```

Sie werden erkennen, welche Bits im Joystick-Register den vier Richtungsschaltern entsprechen. Der Joystick in Ruheposition erzeugt die 127, digital 01111111. Druck nach vorn ergibt den Wert 126 (01111110), Bit 0 ist also offensichtlich mit dem „Oben"-Kontakt des Joysticks verbunden. Die folgende Tabelle zeigt die unterschiedlichen Registerwerte:

Analog oder digital?

Ein digitaler oder „Kontakt"–Joystick registriert die Bewegung des Griffes nach zwei von vier Richtungen – wie weit der Griff bewegt wird, bleibt dabei unberücksichtigt. Anders der Analog-Joystick: Durch zwei rechtwinklig angeordnete Potentiometer wird neben der Richtung auch meßbar, wie weit der Griff aus der Ruheposition verschoben wurde. Die Spannung am Schleifer der Potentiometer wird erst im Rechner in ein digitales Signal umgesetzt.



Joystick	Dezimal	Binär
Mittel-position	127	01111111
Auf	126	01111110
Ab	125	01111101
Links	123	01111011
Rechts	119	01110111
Feuer	111	01101111

Es ist Ihnen wahrscheinlich aufgefallen, daß bei einer diagonalen Bewegung mit dem Joystick zwei Schalter zugleich geschlossen werden. Für die Fahrzeug-Steuerung wird diese Funktion zwar nicht gebraucht, wir wollen Ihnen die resultierenden Werte aber nicht vorenthalten:

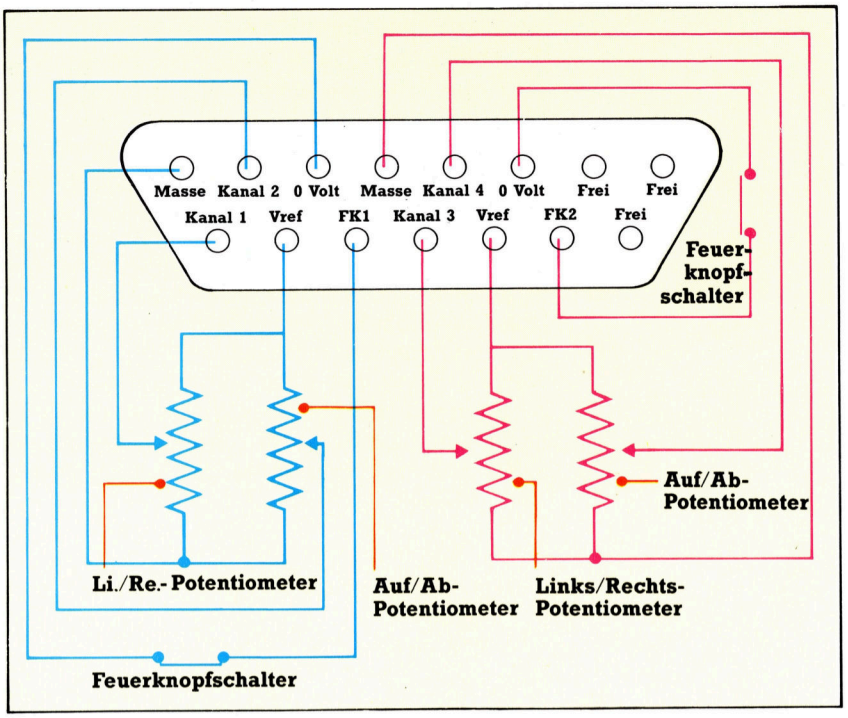
Joystick	Dezimal	Binär
Auf/Links	122	01111010
Auf/ Rechts	118	01110110
Ab/Links	121	01111001
Ab/ Rechts	117	01110101

Mit diesem Programm können Sie das zweimotorige Auto über den Joystick steuern. Dazu wird es wie bereits zuvor beschrieben am Niedervolt-Ausgang angeschlossen. Der Joystick muß mit Port 2 verbunden sein.

```
10 REM****C 64 JOYSTICK****
20 DDR=56579:DATREG=56577
30 POKEDDR,255:REM NUR AUSGABE
40 JOY=PEEK(56320):REM JOY PORT 2
50 GOSUB1000:REM JOYSTICK-TEST
60 POKEDATREG,0:GOTO40
90:
1000 REM JOYSTICK-TEST S/R
1005 IFJOY=127THEN POKEDATREG,0
1010 IFJOY=126THEN POKEDATREG,5
1020 IFJOY=125THEN POKEDATREG,10
1030 IFJOY=123THEN POKEDATREG,6
1040 IFJOY=119THEN POKEDATREG,9
1050 IFJOY=111THEN POKEDATREG,0:END
1060 RETURN
```

Acorn B

Beim Acorn-Joystick kommen anstelle von Schaltern zwei Potentiometer für die Abtastung der Auf/Ab- bzw. Links/Rechts-Bewegungen zum Einsatz – er arbeitet „analog". Im Gegensatz zu digitalen Geräten können damit nicht nur einfache Richtungsangaben, sondern auch die genauen Positionen innerhalb festgelegter Grenzen gemessen und übermittelt werden. Die Funktion der Joystick-Potentiometer soll kurz erklärt werden: Ein Potentiometer ist eigentlich nichts anderes als ein Widerstand, an dessen Anschlüssen eine Spannung liegt. Ein dritter Anschluß (Schleifer) kann auf dem Widerstand hin- und herbewegt werden, wobei er je nach seiner Stellung einen bestimmten Anteil der Spannung abgreift. Bei linearen Potentiometern ist die vom Schleifer abgegriffene Spannung in der Mitte des Widerstandes gerade halb so groß wie die Gesamtspannung. Der Schleifer kann durch Verschieben auf beliebige Werte zwischen 0 Volt und Gesamtspannung eingestellt werden – wie bei der Lautstärkeregelung eines Radios. Bei analogen Joysticks sind die Schleifer zweier Potentiometer mechanisch mit dem Handgriff verbunden, die im rechten Winkel zueinander angeordnet sind. Die Anschlußbelegung des Acorn-Analogports stellt das Schaubild auf dieser Seite dar.



Die Joysticks für den Acorn B werden meist paarweise mit einem gemeinsamen Stecker verkauft. Die Pinbelegung für den Joystick 2 ist dieselbe wie für den Joystick 1 – es können also auch Einzel-Joysticks angeschlossen werden.

Der Computer speist beide Potentiometer mit einer Spannung. Für die Messung der von den Schleifern abgegriffenen Spannungen sind zwei Eingangskanäle vorgesehen, Kanal 1 für das Links/Rechts-Potentiometer und Kanal 2 für das Auf/Ab-Potentiometer. Der Feuerknopf des Joysticks ist als normaler Schaltkontakt ausgeführt.

Die eingestellte Schleifer-Spannung wird im Computer von einem Analog/Digitalwandler „übersetzt". Dazu vergleicht der Wandler beide vom Joystick gelieferten Spannungen alle zehn Millisekunden mit einer Referenzspannung und gibt als Ergebnis einen Digitalwert aus. Dieser Wert soll hier für die Steuerung des Spielzeugautos genutzt werden.

Eingaben über den Analog-Port können vom Acorn mit dem Befehl ADVAL gelesen werden. ADVAL ergibt je nach Stellung des Joysticks Werte zwischen 0 und 65520. Eine kleine Eingangsspannung vom Schleifer gibt niedrige, eine hohe Spannung entsprechend höhere ADVAL-Werte. Für unseren Zweck sind nur die beiden Grenzwerte, 0 und 65520, von Interesse. Der von ADVAL zu lesende Kanal steht in Klammern hinter dem Befehl. ADVAL(1) liest also Kanal 1 und gibt einen Wert zwischen 0 und 65520 aus.

Mit ADVAL(0) lassen sich zwei weitere Abläufe steuern: Die beiden niedrigsten Bits ent-

sprechen den Feuerknöpfen von Joystick 1 und Joystick 2. X=ADVAL(0) AND 3 ergibt eine Eins, wenn der Feuerkopf von Joystick 1 gedrückt ist. X=ADVAL(0) DIV 256 gibt die Nummer des Kanals aus, bei dem schließlich die letzte Analog/Digital-Umwandlung vorgenommen wurde.

Die Digitalumwandlung jedes analogen Kanals dauert etwa zehn Millisekunden, die Abfrage aller Joystick-Kanäle also insgesamt 40 Millisekunden. Da wir im Moment nur Kanal 1 und 2 verwenden, läßt sich die Abfrage durch Beschränkung auf diese beiden Kanäle beschleunigen. Durch *FX16,2 werden Kanal 1 und 2 ein-, Kanal 3 und 4 jedoch ausgeschaltet.

Hier das komplette Programm zur Steuerung des zweimotorigen Spielzeugautos:

```
10 REM ACORN JOYSTICK-STEUERUNG
20 DDR=&FE62:DATREG=&FE60
30 ?DDR=255:REM NUR AUSGABE
40 REM AKTIVIERUNG A—D KANAL 1&2
50 *FX16,2
60 REPEAT
70 PROCtest_joystick
80 UNTIL fire=1
90 END
100:
110 DEF PROCtest_joystick
120 REPEAT
130 channel=ADVAL(0) DIV 256
140 UNTIL channel <>0:REM WARTEN
150 IF channel=1 THEN PROCleft_right
160 IF channel=2 THEN PROCup_down
170 ENDPROC
180:
190 DEF PROCleft_right
200 REPEAT
210 joyval=ADVAL(1)
220 IF joyval <100 THEN ?DATREG=9
230 IF joyval >64000 THEN ?DATREG=6
240 fire=ADVAL(0) AND 3
250 PRINT?DATREG, channel, joyval
260 UNTIL (joyval>100 AND joyval <64000) OR fire=1
270 ?DATREG=0
280 ENDPROC
290:
300 DEF PROCup_down
310 REPEAT
320 joyval=ADVAL(2)
330 IF joyval <100 THEN ?DATREG=10
340 IF joyval >64000 THEN ?DATREG=5
350 fire=ADVAL(0) AND 3
360 PRINT?DATREG, channel, joyval
370 UNTIL (joyval>100 AND joyval <64000) OR fire=1
380 ?DATREG=0
390 ENDPROC
```

Lösung der Übungen

1) Am einfachsten läßt sich Ihr Fahrzeug durch Messen der Zeit kalibrieren, die es für das Zurücklegen einer 10, 20, 50, 100 oder 150 cm langen Strecke braucht. Berechnen Sie die Geschwindigkeiten, und bilden Sie daraus einen Durchschnittswert. Ähnlich gehen Sie auch bei der Zeitfestlegung für das Kurvenfahren vor: Wählen Sie eine Anzahl verschiedener Winkel, und testen Sie für jeden separat die benötigte Zeit. Exakte Motorsteuerung durch Ein- und Ausschalten ist etwas kompliziert, zumal die Programmstruktur eine sehr präzise Zeitmessung notwendig macht. Abweichungen von nur wenigen Hundertstelsekunden erzeugen schon deutliche Fehler bei der zurückgelegten Strecke bzw. beim Abbiegen. Durch den Einbau einer „Untersetzung" zwischen Motor und Rädern läßt sich dieses Problem aber sehr elegant umgehen.

2) Mit dem Programm aus der letzten Folge kann das Auto durch eine kurvenreiche Strecke gesteuert werden. Der Rückweg ist schwieriger: Jeder Richtungsvariablen muß der entgegengesetzte Wert zugeordnet werden. Vor- und Rückwärts bilden ein solches Paar:

Commodore 64

```
17 A(1)=5:B(1)=10:A(2)=10:B(2)=5
18 A(3)=6:B(3)=9:A(4)=9:B(4)=6
```

Wenn Sie dieses Programm hinzufügen, fährt das Auto auf der gespeicherten Route zurück.

```
92 GOSUB2000:REM ZURUECK ZUM START
2000 REM ZURUECK ZUM START S/R
2010 FOR I=C TO 1 STEP —1
2020 FOR J=1 TO 4
2030 IF DR(I,1)=A(J) THEN POKE DATREG, B(J):J=4
2040 NEXT J
2050 T=TI
2060 IF(TI—T)<DR(I,2) THEN 2060
2070 NEXT I
2080 STOP
2090 RETURN
```

Acorn B

```
1020 DIM DR(100,2),A(10),B(10)
1025 A(1)=5:B(1)=10:A(2)=10:B(2)=5
1026 A(3)=6:B(3)=9:A(4)=9:B(4)=6
1115 PROCreverse_replay
2000 DEF PROCreverse_replay
2010 FOR I=C TO 1 STEP—1
2020 FOR J=1 TO 4
2030 IF DR(I,1)=A(J) THEN?DATREG=B(J):J=4
2040 NEXT J
2042 TIME=0
2045 REPEAT UNTIL TIME>DR(I,2)
2047 ?DATREG=0
2050 NEXT I
2055 PRINT"TASTE C FUER WEITER"
2060 REPEAT A$=GET$
2070 UNTIL A$="C"
2080 ENDPROC
3000 FORI=1TOC:PRINTDR(I,1),DR(I,2)
3010 NEXT
```

3) Wenn Anschluß 7 Vorwärts, 6 Rückwärts, Anschluß 5 Links und Anschluß 4 Rechts sind:

```
10 REM ACORN B EXTERNE SCHALTER
20 DDR=&FE62:DATREG=&FE60
30 ?DDR=15:REM EINGABE—ANSCHLUESSE 4—7
40 ?DATREG=0
50 PROCtest_input
60 GOTO50
70:
80 DEF PROCtest_input
90 IF(?DATREG AND 240)=240 THEN ?DATREG=0
100 IF(?DATREG AND 128)=0 THEN ?DATREG=5
110 IF(?DATREG AND 64)=0 THEN ?DATREG=10
120 IF(?DATREG AND 32)=0 THEN ?DATREG=6
130 IF(?DATREG AND 16)=0 THEN ?DATREG=9
140 ENDPROC

10 REM C64 EXTERNE SCHALTER
20 DDR=56579:DATREG=56577
30 POKEDDR,15:REM EINGABE ANSCHLUSS 4—7
40 POKEDATREG,0:REM MOTOREN AUS
50 REM A7 VOR,A6 RUECK,A5 LI,A4 RE
55 REM IST EINE EINGANGSLEITUNG LOW?
60 IF(PEEK(DATREG)AND240)<>240THENGOSUB1000:GOTO60
70 POKEDATREG 0:GOTO60
80:
1000 REM EINGAENGE ABFRAGEN S/R
1005 IF (PEEK(DATREG)AND128)=0THENPOKEDATREG,5
1010 IF(PEEK(DATREG)AND64)=0THENPOKEDATREG,10
1020 IF(PEEK(DATREG)AND32)=0THENPOKEDATREG,6
1030 IF(PEEK(DATREG)AND16)=0THENPOKEDATREG,9
1040 RETURN
```

Gebrüder Casio

Casio ist als Hersteller von Uhren, Taschenrechnern und Musikinstrumenten bekannt. In den vergangenen Jahren brachte das Unternehmen zahlreiche Hand-held- und Taschencomputer heraus, um so auf diesen speziellen Markt zu gelangen.

Casio behauptet, den Taschenrechner-Markt weltweit mit 50 Prozent zu beherrschen, beschäftigt aber nur etwa 3300 Angestellte. 1983 machte Casio einen Gesamtumsatz von 29 Millionen US-Dollar – für einen Elektronikhersteller dieser Größenordnung nicht besonders viel.

Casio wurde von den fünf Brüdern Kashio nach dem Zweiten Weltkrieg gegründet. Damals hieß die Firma Kashio Seisakujo und stellte Büromaterialien her. Anfang der fünfziger Jahre entwickelte man den 14-A-Relay-Calculator (Relaisrechner). Er war einer der ersten elektrischen Rechner, hatte die Größe eines Schreibtisches und wog 130 kg. 1957 erfolgte dann die Namensänderung in Casio Computer Company Ltd.

Mißerfolg des FX-702P

Im Jahre 1982 führte Casio den ersten Taschencomputer ein, als FX-702 P bezeichnet. Er war für wissenschaftliche Anwendungen konzipiert und verfügte über eine ungewöhnliche Tastatur-Anordnung: Die Tasten waren nämlich alphabetisch aufeinanderfolgend gesetzt und nicht im üblichen QWERTY- bzw. QWERTZ-Standard angelegt. Aufgrund der enttäuschenden Verkaufsergebnisse ersetzte man den Rechner durch das Modell FX-700 P, das mit QWERTY-Tastatur ausgestattet war.

Darauf folgte der PB-100, ein Taschencomputer für Geschäftsleute. Ähnlich gestaltet wie der FX-702 P, war er mit einer Flüssigkristall-Anzeige und numerischer Tastatur versehen. Dazu lieferte Casio eigene Cassettenrecorder, Drucker/Plotter und RAM-Erweiterungen für beide Modelle.

FX-750 P ist das jüngste Nachfolgemodell des PB-100. Dieser Computer verfügt über zwei „RAMcard"-Schächte, in die schmale Metallmodule (in Streichholzschachtelgröße) passen, auf denen bis zu vier KByte gespeichert werden können. Jede Karte enthält eine Drei-Volt-Batterie, so daß die Daten nicht verlorengehen, wenn die Karte aus dem Computer entfernt wird. Programme lassen sich auf diese Weise beliebig speichern oder laden.

Casio stellt außerdem den FP-200 her, einen Hand-held-Computer mit acht KByte RAM, der auf 32 KByte erweitert werden kann. Der FP-200 verfügt über eine LCD-Anzeige. Im Text-Mode werden acht Zeilen mit je 20 Zeichen dargestellt. Die Grafikauflösung beträgt 160 x 64 Punkte. Der neueste Rechner der Casio-Produktpalette ist der SL-800. Es handelt sich um einen preiswerten Taschenrechner von Größe und Gewicht einer Kreditkarte. Die extrem schlanke Form ist das Entwicklungsergebnis eines Herstellungsprozesses, den man „Filmen" nennt. Die Bauteile werden dabei auf einen speziell beschichteten Film gedruckt und nicht wie bei den herkömmlichen Verfahren auf eine Platine gelötet.

Casio vertreibt in vielen europäischen Ländern und dem Fernen Osten Großrechner und MSX-Computer. In England aber bietet das Unternehmen ausschließlich Taschenrechner und kleine Computer an. Auf die Frage, warum Casio nicht versucht, in den englischen Heimcomputer- und Geschäftscomputer-Markt zu gehen, verweist Tony Manton, Verkaufsleiter für Taschenrechner, auf die Gefahren, die dieser Industriebereich birgt: „Wir wollen langsam wachsen", sagt er. „Wir müssen die Leute erziehen und ihnen verdeutlichen, daß Taschen- und Hand-held-Computer mehr als nur Taschenrechner sind."

Tadao Kashio, Präsident von Casio, ist eines der fünf Familienmitglieder der Kashios, die das Unternehmen Casio führen.

Casio Forschungs- und Entwicklungszentrum in Tokio

Option und Auswahl

In diesem PASCAL-Kurs werden die Strukturen der Vergleiche erklärt. Dabei behandeln wir die IF-Anweisung und die CASE-Anweisung, eine Struktur, die Mehrfach-Auswahlen ermöglicht.

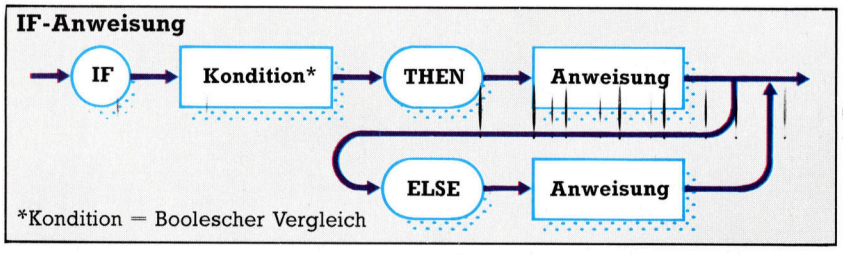
Die IF-Anweisung ist bei PASCAL ähnlich zu handhaben wie bei den meisten anderen Programmiersprachen. Durch Einrücken untergeordneter Begriffe wird die logische Struktur einer solchen IF-Anweisung erkennbar werden. Im folgenden sehen Sie zwei IF-Anweisungen:

```
IF zahl=grenze
   THEN
      WriteLn ('Kein Platz')
   ELSE
      write ('Naechste Eingabe?');
IF nummer > maximum THEN
   maximum : = nummer
```

Im zweiten Beispiel bedeutet das Fehlen von einer ELSE-Bedingung:

```
ELSE
   {mache nichts}
```

Das Schreiben der reservierten Wörter THEN und ELSE mit derselben Einrückung hilft Ihnen beim Nachvollziehen der Programmstruktur.



PASCAL war die erste Programmiersprache, die die Verwendung vorbestimmter Skalen einführte. Diese Methode ist sehr hilfreich, da sie einen umfassenden Überblick über die verschiedenen Daten-Typen in einem Programm ermöglicht, ohne daß man ständig Daten in numerische Codes umwandeln muß. Wir haben bereits gesehen, wie eine einfache Konstanten-Definition uns dabei helfen kann:

```
CONST
   breite =80;
```

Der Konstanten-Identifikator „breite" kann dann im gesamten Programm als Referenz in bezug auf die Anzahl der Spalten des Bildschirms oder des Druckers verwendet werden. Um das Programm für einen Drucker umzuschreiben, der nur eine Spaltenbreite von 40 hat, braucht man nur eine Zeile in den Definitionen am Anfang des Programms zu ändern. Mit dieser Änderung werden automatisch alle Berechnungen der Datenausgabe entsprechend angepaßt.

Wenn man Farbgrafiken verwendet, könnten die verfügbaren Farben durch eine entsprechende Skala von Zahlen dargestellt werden (beispielsweise rot = 1, gruen = 2). Doch dann besteht die Gefahr, daß man die Wurzel aus gruen ziehen oder blau mit gelb multiplizieren müßte! Dies ist nicht nur unlogisch, sondern auch eine potentielle Fehlerquelle. Der TYPE-Definitionsabschnitt eines PASCAL-Programms kann jedoch dazu verwendet werden, einen völlig neuen Skalar-Typ zu definieren. Hierzu braucht nur eine Liste von Identifikatoren, die alle konstanten Werte der Skala repräsentieren, bestimmt zu werden. Zum Beispiel:

```
TYPE
   farbe=(rot,gruen,gelb,blau,magenta,cyan);
```

Da es eine Definition und keine Deklaration ist, wird das Gleichheitszeichen zur Zuordnung der in Klammern befindlichen geordneten Werte zum Identifikator (farbe) verwendet. Die Identifikatoren werden intern durch Integer-Werte (mit der normalen Zählweise beginnend bei Null) repräsentiert. Diese numerische Repräsentation wird automatisch durch den Compiler organisiert und entspricht in etwa den Codes der Computer-Zeichensätze. Jeder Farbwert hat eine Ordnungs-Zahl, die man mit der Skalar-Funktion "ord" bestimmen kann. Im hier gezeigten Beispiel würde ord(rot) den Wert 0 und ord(cyan) den Wert 5 als Ergebnis ausgeben. Wir können nun eine Variable in der bereits bekannten Weise deklarieren:

```
VAR
   farbwert : farbe;
```

Hiermit wird ein Identifikator (farbwert) als Datenbestandteil des Datentyps "farbe" deklariert. Ähnlich verhält es sich mit der folgenden Deklaration:

```
VAR
   buchstabe : zeichen;
```

Diese Deklaration gibt die Zeichenart eines Datenobjektes mit dem Namen "buchstabe" an. Die einzigen Operationen, die für selbstbestimmte Datentypen definiert sind, sind Vergleiche und die Skalar-Funktionen. So könnte man beispielsweise schreiben:

```
IF farbwert < cyan THEN
   farbwert := succ(farbwert)
```

Zeichen und Zahlen können als Parameter in "write-" und "WriteLn"-Anweisungen verwendet werden, doch

```
WriteLn (farbwert)
```

wäre illegal. Wenn man die Namen ausdrukken will, muß man die Farbwerte in Form von Zeichen-Strings darstellen. Dies ist eine ideale Anwendung für die andere bereits erwähnte PASCAL-Struktur, die CASE-Anweisung.

Bis jetzt haben Sie gesehen, daß die Handhabung der IF-Anweisung in PASCAL der anderer Programmiersprachen emtspricht. Durch die freie Programmtexteingabe von PASCAL wird sie sogar noch einfacher nachvollziehbar. Trotzdem kann es vorkommen, daß man mehrere Entscheidungen auf einmal treffen muß. Betrachten Sie das folgende Beispiel:

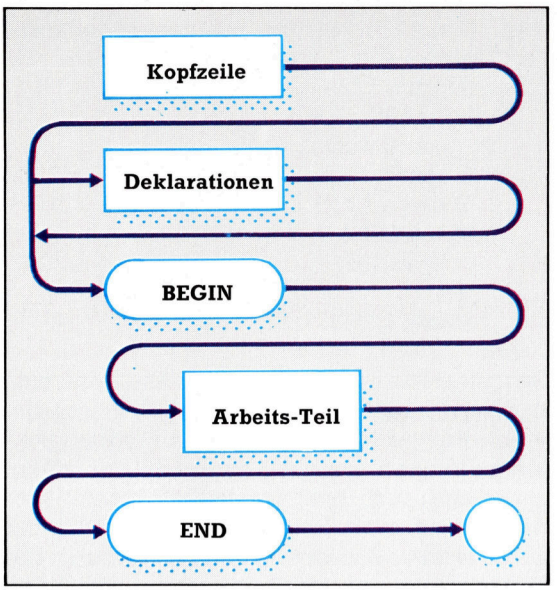
```
IF N = 1
   THEN
      write ('st.')
   ELSE
      IF N = 2
         THEN
            write ('nd.')
         ELSE
            IF N = 3
               THEN
                  write ('rd.')
               ELSE
                  write ('th.')
```

Wann immer die auszuführende Anweisung vom Wert einer einfachen Skala in einem begrenzten Bereich abhängt, kann man die CASE-Anweisung verwenden. Für das hier gezeigte Beispiel sieht das dann so aus:

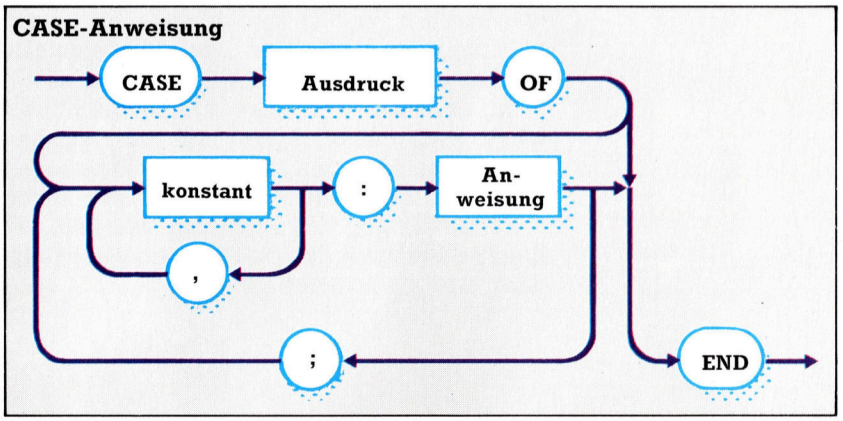
```
CASE N OF
   1                  : write ('st.');
   2                  : write ('nd.');
   3                  : write ('rd.');
   4,5,6,7,8,9        : write ('th.')
END {CASE}
```

Beachten Sie, daß nur die Werte von N Gültigkeit haben, die auch innerhalb der CASE-Anweisung spezifiziert wurden – in diesem Beispiel die Werte 1 bis 9. Alle Werte, die N während der Programmausführung eventuell beinhalten kann, müssen angegeben werden. Für das hier gezeigte Beispiel wäre somit die Eingabe 0 für N nicht möglich.

Viele PASCAL-Compiler berücksichtigen auch die zusätzlichen reservierten Worte OTHERWISE oder OTHERS, die zur Aktivierung eines alternativen Programmteils verwendet werden können. Ausführliche Informationen über diese Wörter entnehmen Sie bitte dem Handbuch Ihrer PASCAL-Version.



Dies ist das vereinfachte Flußdiagramm eines PASCAL-Programms, in dem alle wesentlichen Elemente enhalten sind. Beachten Sie, daß die reservierten Wörter von PASCAL in Großbuchstaben geschrieben sind. Diese Form der Darstellung soll dabei helfen, sie von den nicht reservierten Wörtern zu unterscheiden. Die meisten PASCAL-Versionen akzeptieren jedoch auch reservierte Wörter in Kleinbuchstaben.



CASE ist die einzige Anweisung in PASCAL, die das Wort END zur Begrenzung einer Struktur verwendet, die nicht mit BEGIN gestartet wurde. Es ist daher allgemein üblich, das Ende jeder CASE-Anweisung in der hier gezeigten Form zu kennzeichnen.

Die Struktur arbeitet in der Form, daß sie den Skalar-Ausdruck zwischen den reservierten Wörtern CASE und OF berechnet. (Ein einfacher Variablenname ist ein Ausdruck, der keine Berechnungen erfordert.) Der ermittelte Wert wird mit den folgenden Konstanten verglichen. Trifft ein Vergleich zu, wird die dem Doppelpunkt folgende Anweisung ausgeführt. Die Programmkontrolle überspringt dabei keinen Teil. Dadurch bleibt der Zusammenhang der Konstruktion gewahrt. Wenn mehrere Operationen ausgeführt werden müssen, kann selbstverständlich auch noch eine zusammengesetzte Anweisung zwischen den Worten BEGIN und END verwendet werden.

Unter gewissen Umständen kann es vorkommen, daß einige Werte keinerlei Aktivität erfordern. In solchen Fällen kann die einfachste aller PASCAL-Anweisungen verwendet wer-

den. Diese Anweisung ist die sogenannte "Null"-Anweisung, die soviel bedeutet wie "mache nichts". Dazu ein Beispiel:

```
CASE N MOD 4 OF
  0: {mache nichts};
  1,3 : begin
    write (N MOD 4: 1, 'viertel');
    if N MOD 4 > 1 then
      write ('s')
    end;
  2 : write ('1/2')
END {CASE}
```

Beachten Sie, daß nach wie vor ein Semikolon benötigt wird, um auch diese nicht existierende Anweisung von folgenden Programmteilen zu trennen. Bei der Anweisung der letzten Zuordnung muß kein Semikolon gesetzt werden, da ein reserviertes Wort folgt (END) und keine weitere Zuordnung oder Anweisung. Die Verwendung des MOD-Operators gewährleistet, daß der Wert im Bereich von 0 bis 3 liegt. MOD gibt den Rest einer Integer-Division aus, ähnlich wie bei einigen BASIC-Versionen.

Im nächsten Teil dieses PASCAL-Kurses werden wir uns mit diesen und allen anderen Operatoren sowie mit eingebauten Funktionen befassen. Zum Abschluß zeigen wir Ihnen noch die Lösung des Problems, wie man die Zeichen-Strings für jeden Wert unseres selbstdefinierten Datentypes ausdruckt:

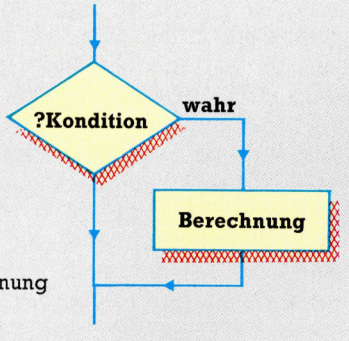
```
CASE farbwert OF
  rot          : write ('Rot');
  gruen        : write ('Gruen');
  gelb         : write ('Gelb');
  blau         : write ('Blau');
  magenta      : write ('Magenta');
  cyan         : write ('Cyan');
END {CASE}
```

Vergleichs-Konstruktionen

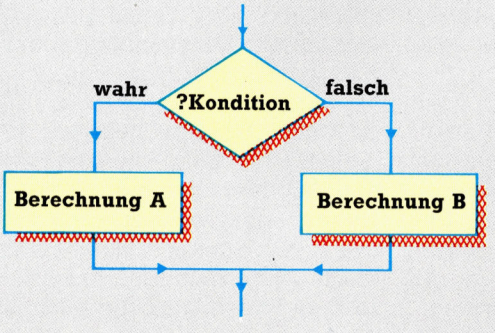
IF ...THEN führt den Programmteil „Berechnung" dann aus, wenn der Vergleich wahr ist. Ist der Vergleich falsch, so wird die Programmausführung einfach mit der nächsten Anweisung fortgesetzt.

Im Gegensatz zur IF ...THEN-Konstruktion, die eine „Option" anbietet, stellt IF ...THEN ...ELSE eine „Auswahl" zur Verfügung. Abhängig vom Ergebnis des Vergleichs wird eine von zwei Berechnungen ausgeführt.

Option
IF Kondition
  THEN Berechnung



Auswahl
IF Kondition
  THEN Berechnung A
  ELSE Berechnung B



Wieviele Tage bis Ultimo?

Das folgende Programm berechnet lediglich die Anzahl der Tage zwischen einem vom Anwender eingegebenen Datum und dem Ende eines Monats. Trotzdem können Sie hier einige Grundsätze der PASCAL-Programmierung erkennen.

Die CASE-Anweisung wird zur Umsetzung der eingegebenen numerischen Daten in das interne Datenformat verwendet. Anschließend werden die Daten wieder umgesetzt, damit sie als String auf dem Bildschirm ausgegeben werden können. Beachten Sie die Zeichen-Variable (symbol), die nur dazu dient, die beliebigen Zeichen zu lesen, die zur Trennung der Eingabe verwendet werden.

```
PROGRAM          Datum          (input,output);
CONST
     Punkt='.';
TYPE
    Kalender=(Jan, Feb, Mrz, Apr, Mai,
      Jun, Jul, Aug, Sep, Okt, Nov, Dez);
VAR
    Monatname      :Kalender;
    tag,
    monat,
    jahr,
    rest           :integer;
    symbol         :zeichen;
    Schaltjahr     :boolean;
BEGIN
  WriteLn ('Geben Sie das Datum wie folgt ein:');
  WriteLn ('TT/MM/JJ':40);
  WriteLn;
  write ('Datum?');
  read (tag, symbol, monat, symbol, jahr);
  Schaltjahr:=jahr MOD 4 = 0;
      {Jahrhunderte werden nicht beachtet}
IF (monat > 0) AND (monat < = 12)
  THEN
    CASE monat OF

    1  :MonatName:=Jan;     7  :MonatName:=Jul;
    2  :MonatName:=Feb;     8  :MonatName'=Aug;
    3  :MonatName:=Mrz;     9  :MonatName:=Sep;
    4  :MonatName:=Apr;     10 :MonatName:=Okt;
    5  :MonatName:=Mai;     11 :MonatName:=Nov;
    6  :MonatName:=Jun;     12 :MonatName:=Dez

    END              {CASE}
  ELSE
    BEGIN
      WriteLn ('Aeh?');
      WriteLn (' — das Programm kann nicht', 'weiter ausgefuehrt
            werden!')
    END; {MonatName ist nicht initialisiert}

  CASE MonatName OF
  Jan,Mar,
  Mai,Jul          :Rest:=31—tag;
  Apr,Jun
  Sep,Nov          :Rest:=30—tag;
  Feb              :IF Schaltjahr
                      THEN
                        Rest:=29—tag
                      ELSE
                        Rest:=28—tag

  END; {CASE}

  WriteLn;
  write('Es sind noch', Rest:1,
              'Tage im Monat');

  CASE Monat Name OF
    Jan            : write ('Januar');
    Feb            : write ('Februar');
    Mrz            : write ('Maerz');
    Apr            : write ('April');
    Mai            : write ('Mai');
    Jun            : write ('Juni');
    Jul            : write ('Juli');
    Aug            : write ('August');
    Sep            : write ('September');
    Okt            : write ('Oktober');
    Nov            : write ('November');
    Dez            : write ('Dezember');

  END; {CASE}

  WriteLn (Punkt)

  END.
```

Seit den siebziger Jahren steuern Microprozessoren die miteinander verknüpften Funktionen von Lichtmessung und Blendenwahl. Chip-gesteuerte Kameras von heute können weit mehr: Sie steuern Verschluß-, Belichtungs- und Blitzfunktion, womit selbst ein Anfänger hochwertige Fotos machen kann. Zwei dieser Kameras sind die Nikon FA und die Pentax Super A, die beide über verschiedene Kontrollprogramme verfügen, um unterschiedlichen Bedingungen gerecht zu werden.

In Schußweite

Eines der Hauptprobleme für den Fotoneuling ist, die technischen Details seiner Ausrüstung zu verstehen. Die neuesten Kameras sind mit Microprozessoren ausgestattet, die das Fotografieren erheblich vereinfachen. Hier stellen wir einige vor.

Beim Fotografieren besteht die erste Aufgabe des Fotografen darin, die richtige Belichtungszeit zu wählen. Dabei wird die Lichtmenge festgelegt, die beim Aufnehmen einer bestimmten Szene den Film in der Kamera erreichen soll. Zur Bestimmung der Belichtungszeit muß die richtige Balance zwischen Blendeneinstellung und Verschlußgeschwindigkeit gefunden werden. Dabei müssen die Größe des „Lochs" in der Linse, das den Lichteinfall bestimmt, und die Dauer der Belichtung in ein passendes Verhältnis gebracht werden.

Deshalb muß zunächst die Menge des einfallenden Lichtes gemessen werden. Entsprechend der Filmempfindlichkeit werden anschließend Blende und Verschlußgeschwindigkeit eingestellt. Im Laufe der Zeit wurden immer bessere Belichtungsmesser entwickelt, die den Fotografen die Lichtintensität angaben. Später wurden Belichtungsmesser direkt in die Kameras eingebaut, wenngleich es dem Fotografen immer noch überlassen blieb, Verschlußgeschwindigkeit und Blende per Hand einzustellen.

Dank der Elektronikentwicklung in den siebziger Jahren wurde es möglich, die vom Belichtungsmesser ermittelten Werte direkt in Eingaben für Blende und Verschlußzeit umzusetzen. So kann man heute vergleichsweise gute Fotoergebnisse erzielen, indem man die Kamera einfach auf ein Objekt richtet und den Auslöser betätigt. – Eine nützliche Einrichtung, sowohl für den Anfänger, der Fotos machen will, ohne die Kamerafunktionen kennenlernen zu müssen, wie auch für den professionellen Fotoreporter, der unter schwierigen Bedingungen gute Ergebnisse erzielen will.

Sechs Aufnahmemodi

Die Canon A1 ermöglicht sechs verschiedene Aufnahmemodi. Das sind:

1. Verschluß-Priorität: Der Fotograf legt eine Verschlußgeschwindigkeit fest, und die Kamera wählt die entsprechende Blende.

2. Blenden-Priorität: Der Fotograf wählt die Blende, die Kamera legt die Verschlußgeschwindigkeit fest.

3. Programm: Die Kamera ermittelt sowohl Blende als auch Verschlußgeschwindigkeit unter Verwendung eines Programms, das beide Faktoren optimal kombiniert.

4. Automatischer Blitz: Mit speziellen Blitzgeräten versehen, wählt die Kamera automatisch die richtige Verschlußgeschwindigkeit für Blitz (1/60 Sekunde) und stellt die Blende entsprechend ein. Ein in das Blitzgerät integrierter Belichtungsmesser beendet den Blitz, sobald genug Licht vom Objekt reflektiert worden ist.

5. Unterdrückte Linsen-Priorität: Sie findet bei älteren Linsensystemen und bei bestimmten Objektiven Anwendung, die mit einem gleichbleibenden Blendenwert arbeiten.

Neuere Linsen bleiben beim Suchen in der maximalen Blendeneinstellung, die sich von der unterscheidet, die bei der Aufnahme verwendet wird. Dadurch wird das Bild im Sucher so hell wie möglich.

6. Manuell: Sowohl Verschlußgeschwindigkeit als auch Blende werden vom Fotografen eingestellt. Das ist nützlich, wenn man besondere Effekte erreichen will.

Im Sucherfeld der Canon A1 wird elektronisch dargestellt, in welchem Modus die Kamera arbeitet und welche Werte für Verschlußgeschwindigkeit und Blende gewählt wurden. Diese Darstellung erfolgt mit LEDs, so daß sie auch im Dunkeln zu erkennen ist.

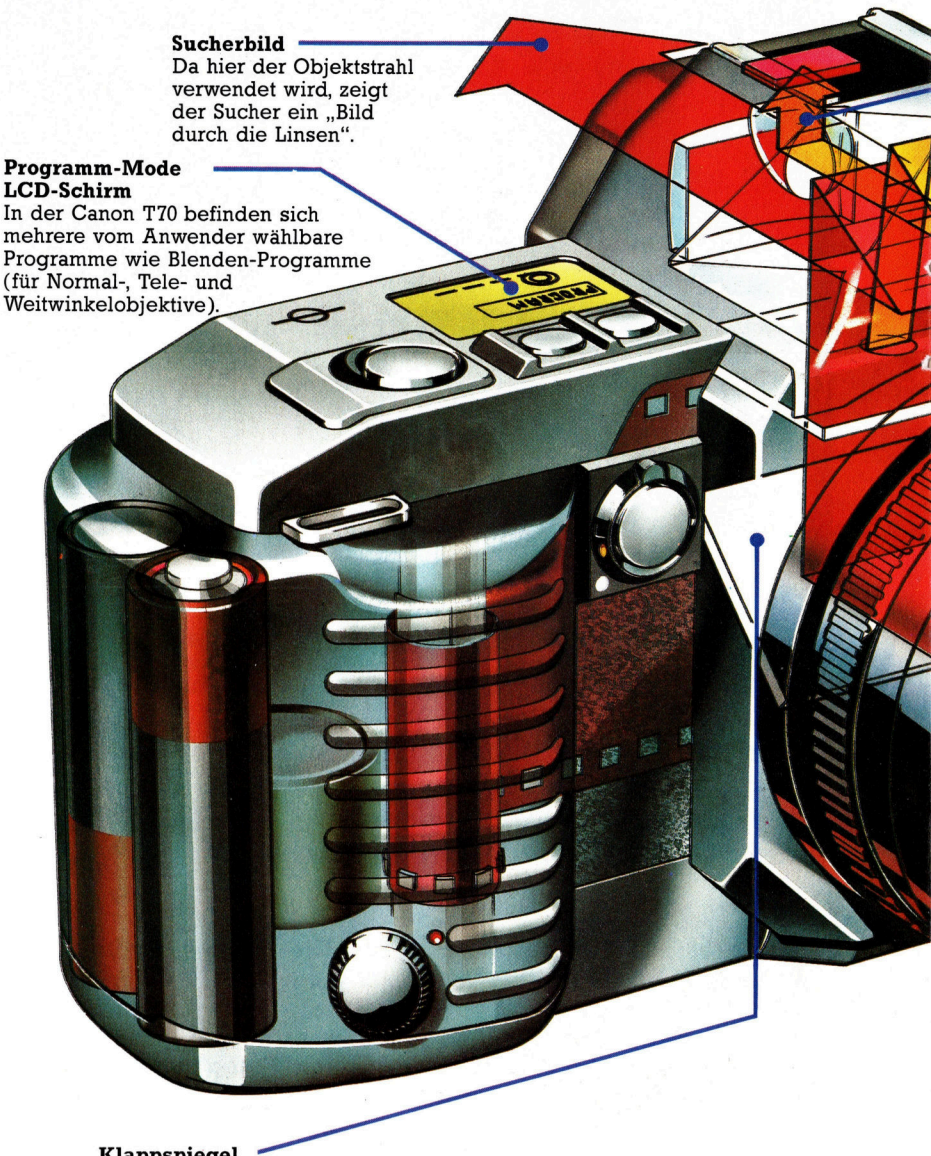
„Verwackelte" Bilder

Trotz des generell nützlichen Programm-Modes der Canon A1 arbeitet die Kamera mit einer sehr einfachen Technik. Das bedeutet in bestimmten Fällen, daß die Kamera nicht die bestmögliche Kombination wählt. Fotografiert man etwa am späten Abend, wählt die Kamera eine Verschlußgeschwindigkeit von 1/30 und eine Blende von f 2.8. Fotos, die mit mehr als 1/60 Sekunde aufgenommen werden, sind wegen der leichten Bewegungen der Hand des Fotografen meist „verwackelt". Die Kamera zeigt durch ein Blitzen an, daß bei Belichtungszeiten über 1/60 Sekunde Verwackelungsgefahr besteht, doch das Programm wählt keine größere Blende, um eine höhere Verschlußgeschwindigkeit zu ermöglichen.

Viele Konkurrenzunternehmen haben Multi-Mode-Kameras mit eingebauten Microprozessoren auf den Markt gebracht. Die Pentax Super A ist mit einem leistungsfähigeren Programm als die Canon A1 ausgestattet. So sind bessere Kombinationen zwischen Verschlußgeschwindigkeit und Blende möglich, sowohl bei sehr viel als auch bei sehr wenig Licht. In der zuvor beschriebenen Situation „spätabends" würde auch die Pentax Super A eine Verschlußgeschwindigkeit von unter 1/60 Sekunde wählen. Verwackeln wäre damit weitestgehend ausgeschlossen. Und die von Canons Erzrivalen gefertigte Nikon FA, „weiß" automatisch, ob sie mit einem Teleobjektiv ausgestattet ist und setzt ein entsprechendes Programm ein.

Dem Nikon-Beispiel folgend integrierte Canon in seiner T70 drei Programme. Eines wird bei normalen Objektiven aktiviert, eines bei Teleobjektiven und das dritte bei Weitwinkel-Objektiven. Die Kamera kann allerdings nicht automatisch feststellen, welches Objektiv aufgesetzt ist. Der Anwender muß also das entsprechende Programm selbst wählen. Das bedeutet keinen Rückschritt, da dem Fotografen auf diese Weise kreative Kontrollmöglichkeiten geboten werden.

Ein Problem bei Automatik-Kameras ist, daß sie eine Durchschnittsbelichtungszeit für das ganze Bild wählen. Der Belichtungsmesser ist durch Objekte mit extremen Lichtverhältnissen leicht zu täuschen. Wird beispielsweise ein Motorrad im Gegenlicht fotografiert, ermittelt die Kamera die richtige Belichtungszeit für die Sonne. Das Motorrad ist dann dunkel und kaum zu erkennen. Würde andererseits das Motorrad vor einem dunklen Hintergrund fotografiert werden, betrachtete die Kamera das Objekt als noch dunkler, womit es zu einer Überbelichtung käme.

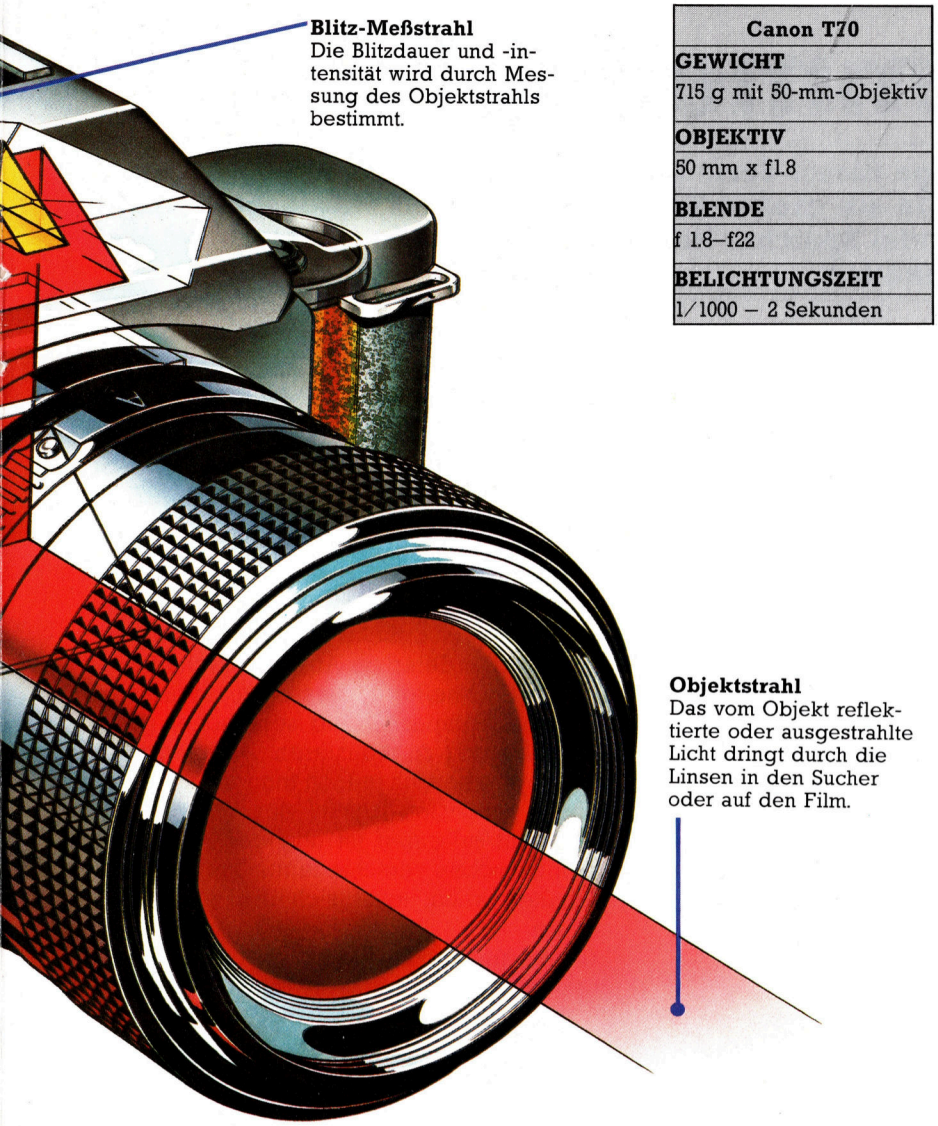


Micro-Lichtmessung

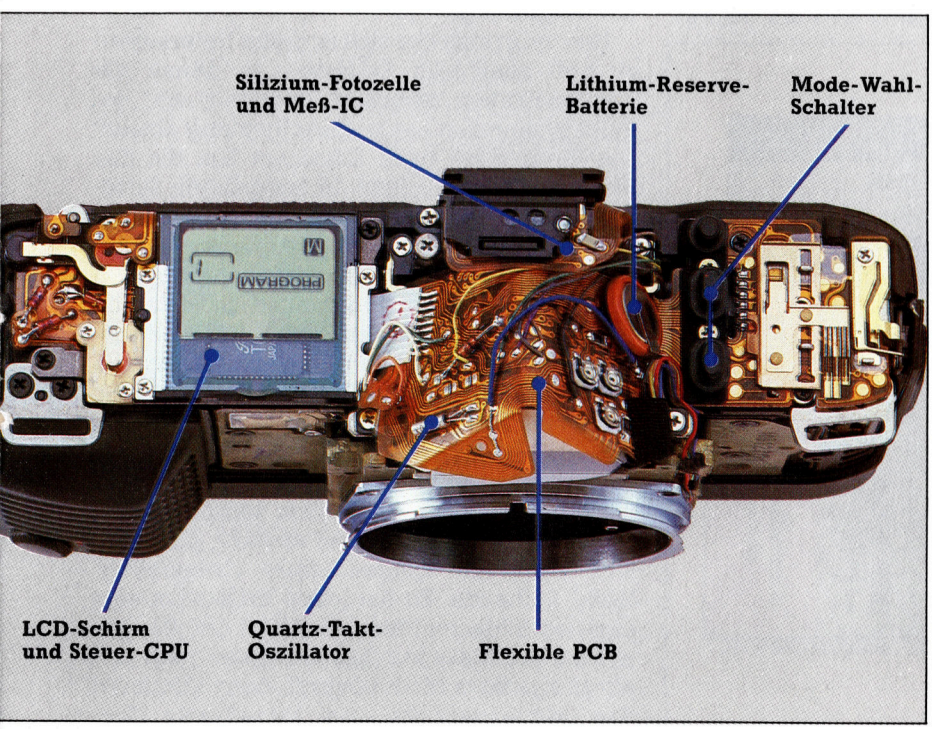
Bei der Nikon FA wird ein neues Verfahren zur Lösung dieses Problems genutzt. Statt nur einer Lichtmessung finden hier Messungen in fünf verschiedenen Bildteilen statt. Mit Hilfe eines Microprozessors werden die fünf Werte dann mit verschiedenen „Standard-Szenen"

Micro-Fotos

Eine speziell entwickelte 8-Bit-CPU steuert sämtliche Funktionen der Canon T70, unterstützt von Meß- und ISO-Chips. Der Quartz-Takt-Oszillator erzeugt die Synchronisationspulse und steuert die Belichtungszeit. Das Meß-IC setzt über sprunggefederte Kontakte die Blende. Wahlweise kann ein Befehlsmodul eingesetzt werden, mit dem eine automatische Intervall-Belichtung möglich ist (zwischen einer Sekunde und einem Tag). Die verwendeten Daten werden direkt auf das belichtete Negativ geschrieben.



Canon T70
GEWICHT
715 g mit 50-mm-Objektiv
OBJEKTIV
50 mm x f1.8
BLENDE
f 1.8–f22
BELICHTUNGSZEIT
1/1000 – 2 Sekunden



verglichen, die die Kamera als Programm enthält. Jede dieser Szenen ist das Ergebnis einer Analyse von Tausenden von Fotografien.

Doch die Kamera, die Elektronik am meisten nutzt, ist die Canon T70. Die T70 verfügt über keinerlei mechanische Kontrollen. Die acht Modi werden durch Druck auf den Knopf an der linken oberen Seite der Kamera gewählt. Daraufhin erscheint eine Information in gut sichtbarer LCD-Darstellung auf der rechten oberen Kameraseite. Sie sehen unter anderem die Verschlußgeschwindigkeit, die Blende und den Modus.

Bei der Wahl der Verschlußgeschwindigkeit kann korrigiert werden, indem auf zwei Knöpfe gedrückt wird. Die Filmempfindlichkeit (als ASA oder ISO-Zahl bekannt) wird auf ähnliche Weise eingegeben. Außerdem erscheint in der Flüssigkristallanzeige der Bildzähler, mit dem die Anzahl der gemachten Fotografien angegeben wird.

Die Kamera verfügt über einen eingebauten Motor, der den Film vor- und rückwärts wickelt. Betrieben wird die T70 mit zwei handelsüblichen Batterien deren Statusanzeige im LCD-Register mit drei Balken erfolgt. Wird der Selbstauslöser der Canon T70-Kamera benutzt, zeigt ein Programm über LCD-Display die zu zählenden Sekunden bis zur Auslösung des Verschlusses.

Energiesparende Chips

Die in Kameras verwendeten Microprozessoren sind weniger leistungsfähig als die in Computern. Die T70 ist mit einem speziell gefertigten Chip auf CMOS-Basis ausgestattet, womit der Energieverbrauch äußerst gering gehalten wird. Er ist auf nur 32KHz getaktet. Microcomputer beispielsweise arbeiten rund hundertmal schneller. Ist die Kamera nicht in Betrieb, sinkt der Takt auf etwa acht KHz, um Energie zu sparen.

Der Microprozessor ist in ein 60-poliges flaches Kästchen eingebettet und verfügt über einen großen ROM-Bereich, doch lediglich 16 Byte RAM. Vier weitere Chips sind dem Microprozessor zugeordnet, von denen der Input/Output-Chip der wichtigste ist. Er steuert über Magnete und einen Motor die mechanischen Abläufe in der Kamera. Ferner wandelt er das analoge elektrische Signal des Belichtungsmesser-Chips in ein digitales Signal um, das der Microprozessor verarbeiten kann.

Dank der Microelektronik macht das Fotografieren mehr Spaß, weil man hochwertige Fotos machen kann, ohne die Feinheiten des Fotografierens verstehen zu müssen. Und dennoch wird es immer Fälle geben, in denen die elektronische Kamera falsche Werte ermittelt. Der Fotograf, der die Funktionsweise einer Kamera tatsächlich versteht, wird mit einer Hochleistungskamera immer im Vorteil gegenüber dem Anfänger sein.

Zusammenspiel

In der vorigen Folge dieser Serie beschäftigten wir uns mit Systemen, deren Funktionen in ein großes Programm integriert waren. Da Pakete dieser Art jedoch viel Speicher belegen, untersuchen wir hier flexiblere Möglichkeiten.

Es gibt eine Methode, integrierte Software-Pakete zu erstellen, die völlig anders aufgebaut sind als die bisher erwähnten Systeme. Sie ist abhängig von der Art des Betriebssystems, das die grundlegenden Voraussetzungen zur Integration liefern muß. Der Aufbau derartiger Systeme ist jedoch nicht einfach, da die Ansprüche an die Software sowie an die Hardware – das betrifft unter anderem die Speicherkapazität – viel höher sind als bei herkömmlichen Computern.

Die Programme dieser Betriebssysteme unterscheiden sich von denen herkömmlicher Computer erheblich. Ein großer Teil der Programme steuert dabei die „Anwenderschnittstelle", das sind Routinen, die Befehle und Informationen vom Anwender entgegennehmen und Ergebnisse anzeigen.

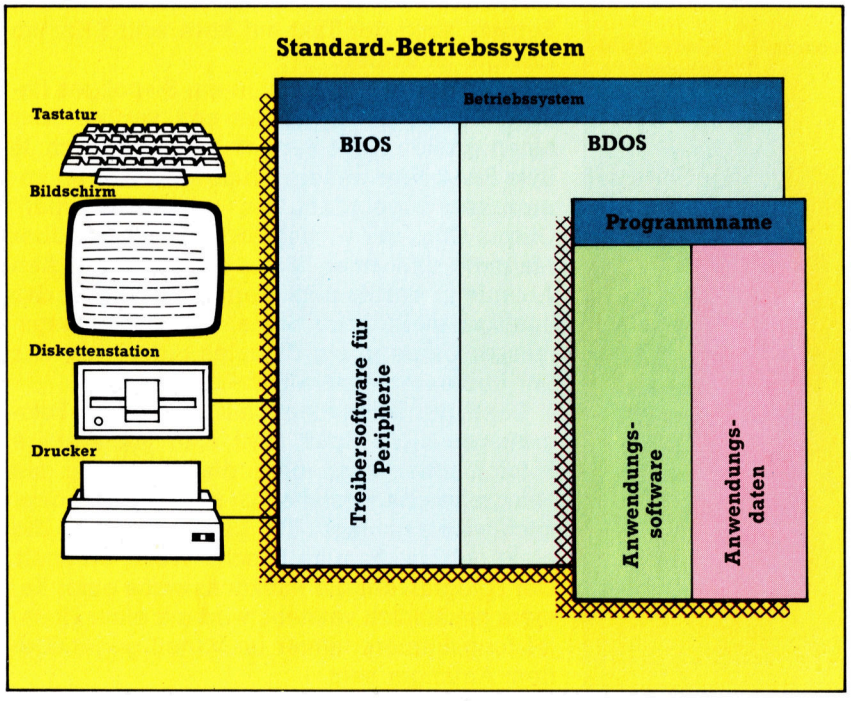
Die Anwenderschnittstelle eines integrierten Betriebssystems unterhält für jedes Programm einen eigenen Satz an Routinen. Soll ein Programm beispielsweise eine Reihe von Auswahlmöglichkeiten auf dem Bildschirm darstellen (ein Menü), so erledigt dies ein speziell dafür erstelltes Betriebssystemmodul. Das hat den Vorteil, daß alle unter diesem System geschriebenen Programme ähnliche Betriebsabläufe enthalten. Außerdem gewinnt der Anwender durch das Erlernen eines Programms leicht Zugang zu allen anderen.

Die Maus ist eine spezielle Anwenderschnittstelle für solche Systeme. Sie steuert den Bildschirmcursor und kann einzelne Programmodule aufrufen. Ein Alternative ist der Touch-Screen, bei dem die Lichtstrahlenmatrix des Bildschirms auf Berührung reagiert. In beiden Fällen ist der Bildschirm in Fenster aufgeteilt, die unterschiedliche Aufgaben anzeigen. Technisch sind derartige Anwenderschnittstellen nur mit schnellen Prozessoren, umfangreicher Speicherkapazität und sehr hoher grafischer Auflösung zu realisieren. Da integrierte Betriebssysteme sich jedoch mit fast allen Programmen einsetzen lassen, der Umgang mit ihnen leicht erlernbar ist und der Anwender schnell zwischen verschiedenen Programmen umschalten kann, lohnt sich die zusätzliche Ausgabe.

Die Integrationsmethode dieser Systeme ist bemerkenswert. Programm und Anwender sind dabei nie in direktem Kontakt – alle Aufgaben werden über das Betriebssystem geleitet, das die Steuerung übernimmt. Die Anwenderprogramme werden zu „Betriebssystemerweiterungen" und der Computer zur „integrierten Umgebung".

Hier zeigt sich der zweite große Unterschied zu herkömmlichen Systemen, in denen die Kommunikation zwischen Programm und Betriebssystem fast nur in einer Richtung abläuft: Die Programme rufen Vorgänge auf, die das Betriebssystem ausführt. In integrierten Systemen übernimmt das Betriebssystem die Steuerung und veranlaßt die Programme zur Ausführung von Aufgaben. Die Kommunikation läuft in beiden Richtungen: Programme müssen die vom Betriebssystem verlangten Abläufe ausführen können und das Betriebssystem muß auf Aufforderungen der Programme reagieren.

Bei herkömmlichen Betriebssystemen übernimmt das laufende Programm die Steuerung. Seine Logik bestimmt, was auf dem Bildschirm erscheint, auf welche Diskettenstation zugegriffen werden soll und wie die Tastatur abgefragt wird. Seine Befehle werden an das Betriebssystem übermittelt, das dann die Steuerung der angesprochenen Hardware übernimmt.



Interne Organisation

Ist dieser Grad an Integration zwischen Software und Maschine einmal erreicht, dann läßt sich auch die entsprechende „Umgebung" leicht aufbauen. So hat jedes Programm sein eigenes Bildschirmfenster. Wenn ein Anwender mit der Maus auf dieses Fenster zeigt und ein bestimmtes Modul aufruft, dann informiert das Betriebssystem das betreffende Pro-

gramm und führt mit ihm zusammen alle notwendigen Vorgänge aus.

Soll beispielsweise ein Fenster auf dem Bildschirm versetzt werden, braucht der Anwender nur das entsprechende Modul aufzurufen. Die Betriebssystemroutinen führen die Arbeit aus. Aber auch das Programm wird über die Veränderung informiert und kann so seinen grafischen Aufbau verändern. Wird die Maus in ein anderes Fenster bewegt, dann hört das ursprüngliche Programm auf zu arbeiten, solange das Betriebssystem mit dem neuen Programm arbeitet. Der Wechsel zwischen Programmen gestaltet sich damit so einfach wie die Bewegung der Maus.

Datenaustausch

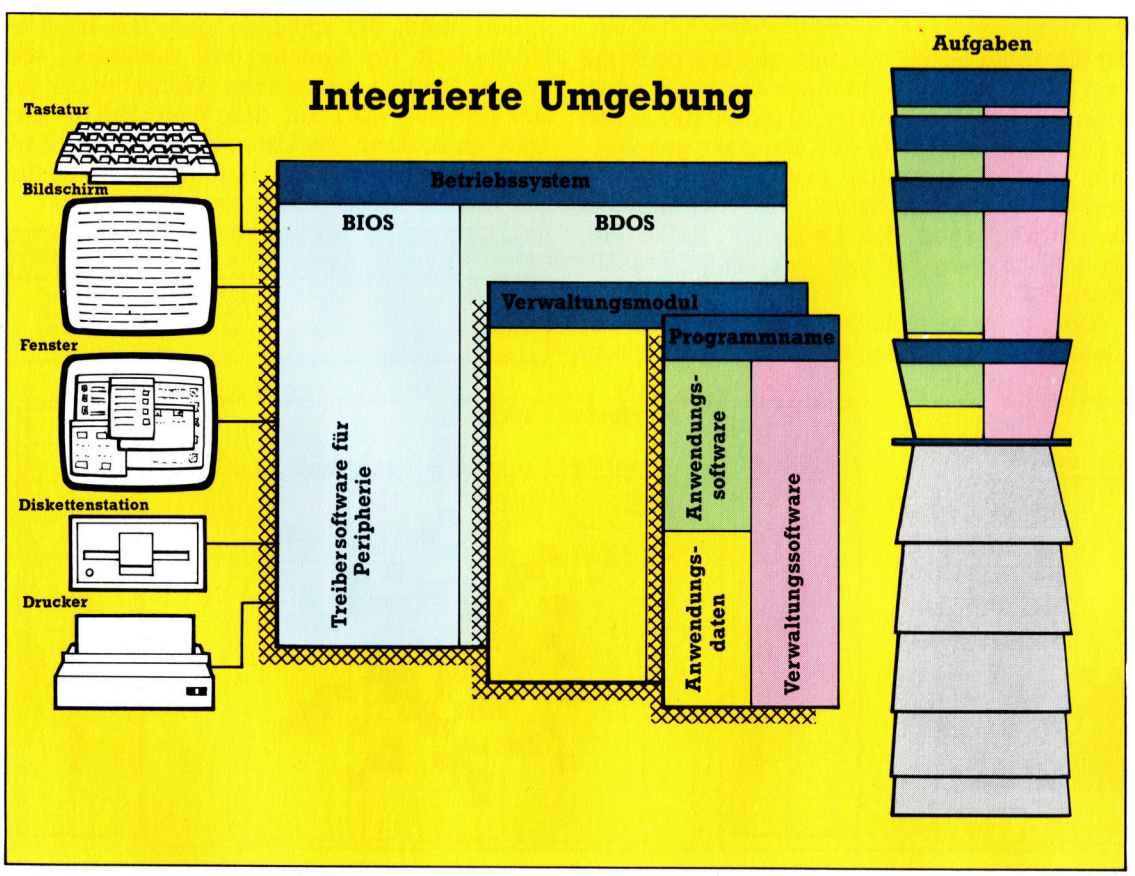
Wie bei den großen „allumfassenden" Programmen versuchen auch die hier vorgestellten Systeme, Programme und Bildschirminformationen im Speicher und damit schnell abrufbar zu halten. Doch auch bei Computern mit großer Speicherkapazität muß das Betriebssystem gelegentlich Informationen und Programme mit Hilfe von Disketten austauschen, da der Speicher nicht alles gleichzeitig aufnehmen kann. Um annehmbare Arbeitsgeschwindigkeiten zu erreichen, werden bei derartigen Systemen oft Festplatten verwendet.

Die Betriebssysteme enthalten Formate und Routinen für den Datenaustausch zwischen Programmen. Wenn Daten eines Programms in ein anderes übertragen werden sollen, hält das Betriebssystem das erste Programm an, startet das zweite und veranlaßt es, die Informationen des ersten zu lesen und zu speichern. Vorgänge dieser Art können auch automatisch ablaufen, so daß sich zum Beispiel ein Diagramm verändert, wenn die Zahlen des entsprechenden Kalkulationssystems modifiziert werden. Die beiden Programme laufen dabei nicht gleichzeitig ab – das Betriebssystem jongliert lediglich je nach Bedarf zwischen beiden Anwendungen.

Beim Lisa von Apple wurde dieses System noch verfeinert. Die Informationen eines Programms können dabei an ein schwarzes Brett „geheftet" und dann in jedes andere Programm übertragen werden. Mit den Daten wird auch die Formatinformation weitergegeben, so daß ein Diagramm, das mit der Grafiksoftware erstellt wurde, auch in anderen Anwendungen als Diagramm erscheint.

Diese Methode, Software zu integrieren, bietet zahlreiche Möglichkeiten, da an allen Programmen des Systems gleichzeitig gearbeitet werden kann, die einzelnen Module sich leicht aufrufen lassen und auch die Datenübertragung zwischen den Anwendungen problemlos funktioniert. Ein Nachteil ist die hochentwickelte – und damit im Augenblick noch sehr teure – Hardware.

Jede technische Neuerung dieser Größenordnung braucht Zeit, bis sie sich durchsetzt. So hatte Xerox zum Beispiel die Maus und die Fensterschnittstelle schon vor zehn Jahren entwickelt, doch erst jetzt sind sie gebräuchlich.



In einem integrierten System wird das Betriebssystem von einem Verwaltungsmodul unterstützt, das alle laufenden Programme und Daten als „Aufgaben" versteht, die in eine Reihenfolge gebracht und ausgeführt werden müssen. Das darunterliegende Betriebssystem wird als Systemsoftware angesprochen, dieses Modul lädt die Aufgaben je nach Anforderung des Anwenders in den Hauptspeicher oder legt sie auf der Diskette ab. Die zwischen unterschiedlichen Anwendungen ausgetauschten Informationen haben ein Standardformat, das eine problemlose Übertragung ermöglicht.

Rollenverteilung

Die Erstellung von Sprites ist eines der interessantesten Details der Grafik-Möglichkeiten des Commodore 64. In diesem Abschnitt werden wir alle Arbeitsschritte zur Erstellung von Sprites durchführen.

Ein Sprite wird auf ähnliche Art und Weise erstellt wie normale Zeichen, die wir bereits in einem früheren Teil des Kurses besprochen haben. Der Unterschied besteht lediglich darin, daß ein Sprite auf einem größeren Raster konstruiert wird. Ist ein Sprite einmal definiert, können Optionen wie Farbe und Position auf dem Bildschirm mit Hilfe spezieller Register des Video-Kontroll-Chips (VIC) bestimmt werden.

Das Raster, auf dem ein Sprite erstellt wird, hat eine Größe von 21 Reihen mit je 24 Pixeln. Jede Reihe besteht aus drei acht-Pixel-Elementen und wird durch drei Bytes im Speicher repräsentiert. Somit werden also insgesamt 63 Bytes zum Speichern der Daten eines Sprites benötigt. Die hier gezeigten Diagramme zeigen die vier Sprites, die bei unserer U-Boot-Jagd verwendet werden. Die Zahlen an der Seite jeder Zeichnung stellen die Dezimalwerte dar, die später in DATA-Anweisungen für die Sprites abgelegt werden.

Ein Sprite besteht aus 21 Reihen mit je drei Bytes. Diese Bytes repräsentieren Bit-Muster, werden jedoch im BASIC-Programm in Form ihrer Dezimal-Werte in DATA-Anweisungen gespeichert. Diese Werte können Sie neben den Sprite-Diagrammen und im Programm-Listing sehen. Das Programm POKEt diese Werte in einen angegebenen Bereich des Speichers (RAM), wo der Video-Kontroll-Chip auf sie als Sprite-Daten Zugriff nimmt. Er stellt sie dar und ermöglicht eine Bewegung auf dem Bildschirm mit minimalem Programmieraufwand.

Speichern der DATA-Anweisungen

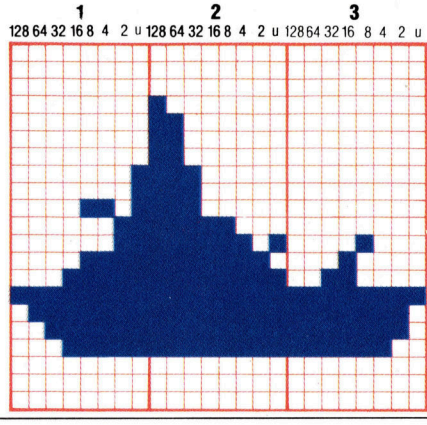
Ist ein Sprite einmal definiert und in eine Reihe von DATA-Anweisungen umgewandelt worden, müssen die Daten mit den Befehlen READ und POKE gelesen und in den Speicher geschrieben werden. Verwendet man dafür beispielsweise einen Bereich mit der Startadresse 12288, so liegen die Daten im BASIC-Programmspeicher, der sich von 2048 bis 40960 erstreckt.

Um ein versehentliches Überschreiben der Daten auszuschließen, wird die obere Grenze des BASIC-Speicherbereiches auf einen Wert festgelegt, der unterhalb der Sprite-Daten liegt. Der Wert der oberen Grenze des BASIC-Speicherbereiches wird in den Adressen 55 (niederes Byte) und 56 (höherwertiges Byte) gespeichert. Normalerweise beinhalten diese beiden Adressen die Werte 0 und 160, was der Adresse 40960 entspricht. Umgewandelt in niederes und höherwertiges Byte, entspricht die Adresse 12288 den Werten 0 und 48. Indem man diese beiden Werte in die Adressen 55 und 56 POKEt, kann man die obere Grenze des BASIC-Speicherbereiches bei Programmstart neu festlegen (siehe Programmzeile 90).

Da die Sprite-Daten in verschiedenen Bereichen des Speichers abgelegt werden können, braucht man einen Zeiger, der angibt, ab welcher Adresse die Daten abgelegt wurden. Insgesamt gibt es acht Sprite-Zeiger, die in den Adressen 2040 (Sprite 0) bis 2047 (für Sprite 7) abgelegt sind. Die Daten für das Schiff des Programms beginnen bei Adresse 12288. Das Schiff bezeichnen wir als Sprite 0, so daß der Zeiger in Adresse 2040 den Wert 192 (12288 dividiert durch 64) enthalten muß. Der nächste Datenblock repräsentiert die Explosion, die wir als Sprite 1 bezeichnen. Wenn der Zeiger bei Adresse 2041 auf den Wert 193 gesetzt wird, so müssen die Daten ab Adresse 12352 abgelegt werden. In der folgenden Tabelle finden Sie die Werte, die verwendet werden:

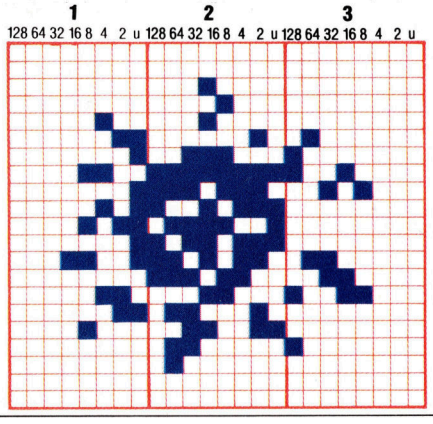
Sprite-Nummer	0	1	2	3
Sprite-Zeiger	192	193	194	195
63 Bytes für Sprite-Daten	12288 bis 12350	12352 bis 12414	12416 bis 12478	12480 bis 12542

Schiff - Sprite 0



1	2	3
0	0	0
0	0	0
0	0	0
0	128	0
0	192	0
0	192	0
0	192	0
1	224	0
1	224	0
13	224	0
3	248	128
3	253	8
15	254	16
31	255	48
255	255	255
127	255	254
63	255	254
31	255	252
0	0	0
0	0	0
0	0	0

Explosion - Sprite 1



1	2	3
0	0	0
0	0	0
0	16	0
0	8	0
4	16	0
3	2	64
1	56	128
12	255	144
1	238	40
5	151	0
11	121	0
1	183	0
25	214	96
0	236	48
6	24	152
3	98	0
8	51	0
0	96	128
0	64	0
0	0	0
0	0	0

Beachten Sie, daß ein Byte ungenutzt übrig bleibt. Die Programmteile, die die Sprite-Daten lesen und in den Speicher schreiben, befinden sich in den Zeilen 2000 bis 2210.

Der Video-Kontroll-Chip verfügt über verschiedene Spezialregister, die zur Kontrolle der Sprites verwendet werden. Die erste Adresse des VIC ist 53248. Wenn wir V den Wert 53248 zuordnen, so kann die nächste Adresse des VIC als V+1 bezeichnet werden und so weiter.

Die Farbe jedes Sprites wird durch POKEn eines Farbcodes (im Bereich von 0 bis 15) in ein spezielles Register bestimmt. Jedes der acht Sprites hat sein eigenes Farbregister. Sie befinden sich bei den Adressen von V+39 bis V+46. Wenn wir beispielsweise das Schiff schwarz einfärben wollen, so braucht man nur den Farbcode 0 in Adresse V+39 zu POKEn. Entsprechend können die anderen Sprites eingefärbt werden (siehe Programmzeilen 2220 bis 2250).

X- und Y-Koordinaten

Die X-Koordinate für Sprite 0 befindet sich in Adresse V, und die Y-Koordinate für Sprite 0 wird in Adresse V+1 gespeichert. Die X- und Y-Koordinaten für Sprite 1 werden in den Adressen V+2 und V+3 abgelegt, bis zu Adresse V+15 (siehe Zeilen 2260 bis 2280).

Sprites können sowohl horizontal als auch vertikal um den Faktor 2 gestreckt werden. Die Sprites für das Schiff und das U-Boot sehen zwar schon jetzt recht breit aus, doch wollen wir nun ihre Breite verdoppeln. In unserem Programm werden alle vier Sprites horizontal verdoppelt. Das VIC-Register, das hierfür zuständig ist, hat die Adresse V+29. Dabei werden nicht acht verschiedene Register benötigt, sondern lediglich diese eine Adresse, mit der die Funktion ein- oder ausgeschaltet wird. Es wird also nur ein Bit je Sprite benötigt, um seine horizontale Verbreiterung zu kontrollieren. In der folgenden Tabelle zeigen wir Ihnen die Bit-Kobination für den POKE, mit dem alle vier definierten Sprites verbreitert werden:

Sprite-Nummer	7	6	5	4	3	2	1	0
Inhalt von V+29	0	0	0	0	1	1	1	1

= 15 (dezimal)

Eine Erweiterung in vertikaler Richtung wird über Adresse V+23 gesteuert. Die Explosion in unserem Spiel, Sprite 1, wird sowohl in horizontaler als auch in vertikaler Richtung erweitert, seine Größe also insgesamt verdoppelt (Zeilen 2290 bis 2310).

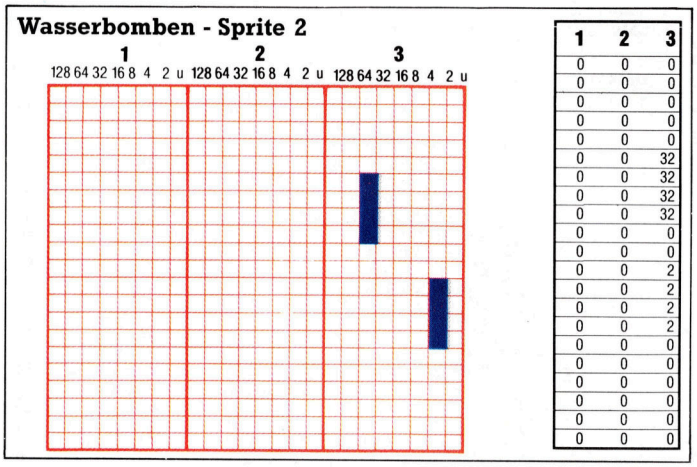
Die abschließende Aufgabe ist, die benötigten Sprites zu aktivieren. Zu diesem Zweck wird ein einziges Bit des VIC-Registers (V+21) verwendet. Mit diesem Bit wird jedes Sprite ein- bzw. ausgeschaltet. Im Spiel sind nur das Schiff und das U-Boot von Anfang an aktiviert (Zeilen 2310 bis 2360).

Wenn Sie das komplette Listing eingegeben haben, sollten Sie überprüfen, ob die Sprite-Daten korrekt eingelesen wurden. Starten Sie deshalb das Programm, und verwenden Sie RUN oder STOP zum Abbrechen, sobald die Spieluhr am oberen Rand des Bildschirmes erscheint. Geben Sie dann die folgenden Anweisungen, ohne Zeilennummern, ein. Durch diese Zeilen werden die durch die Routine erstellten Sprites auf dem Bildschirm positioniert und dargestellt.

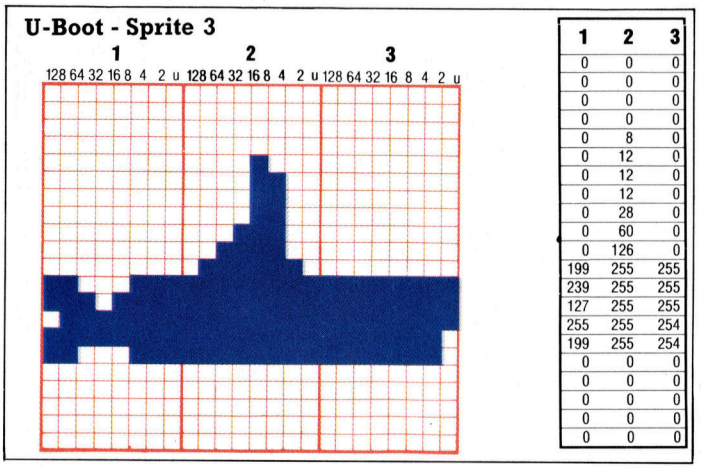
POKE V,160	(Koordinate Schiff)
POKE V+2,240:	(X- und Y-Koordinaten
POKE V+3,100	der Explosion)
POKE V+4,160:	(X- und Y-Koordinaten
POKE V+5,100	der Wasserbomben)
POKE V+6,100:	(X- und Y-Koordinaten
POKE V+7,100	des U-Bootes)
POKE V+21,15	(Aktiviert Sprites 0-3)

Wenn das Programm mit einer „OUT OF DATA ERROR"-Meldung stoppt, überprüfen Sie die Anzahl der Zahlen in den DATA-Anweisungen. Insgesamt müssen es 63 für jedes Sprite sein. Vergewissern Sie sich, daß V in Zeile 100 definiert wurde. Zum Abschluß noch ein Tip: Es ist empfehlenswert, das Programm vor einem Start abzuspeichern.

```
1 REM***C64 GRAPHICS*********
90 POKE 55,0:POKE 56,48:CLR
:REM LOWER MEMTOP
100 V=53248:FL=0:SC=0
110 GOSUB 1000
:REM SCREEN SETUP (see p215)
120 GOSUB 2000
:REM SPRITE CREATION
2000 REM***SPRITE CREATION***
2020 REM** READ SHIP DATA  **
2030 FOR I=12288 TO 12350
2040 READ A:POKE I,A:NEXT I
2060 REM** READ EXP  DATA  **
2070 FOR I=12352 TO 12414
2080 READ A:POKE I,A:NEXT I
2100 REM** READ CHRG DATA  **
2110 FOR I=12416 TO 12478
2120 READ A:POKE I,A:NEXT I
2140 REM** READ SUB DATA   **
2150 FOR I=12480 TO 12542
2160 READ A:POKE I,A:NEXT I
2180 REM** SET POINTERS    **
2190 POKE 2040,192:POKE 2041,
193:POKE 2042,194:POKE
2043,195
2220 REM** SET COLOURS      **
2230 POKE V+39,0:POKE V+40,1:
POKE V+41,0:POKE V+42,0
2260 REM**INIT SHIP COORDS **
2270 POKE V+1,80:X0=160
2290 REM** EXPAND SPRITES  **
2300 POKE V+29,15:POKE V+23,2
2320 REM** TURN ON SPRITES **
2330 POKE V+21,9
2340 RETURN
2350:
6000 REM** SHIP DATA       **
6010 DATA 0,0,0,0,0,0,0,0,0,0
6020 DATA 0,128,0,0,192,0,0,
192,0
6030 DATA 0,192,0,1,224,0,1,
224,0
6040 DATA 13,224,0,3,248,128,
3,253,8
6050 DATA 15,254,16,31,255,48,
255,255,255
6060 DATA 127,225,254,63,255,
254,31,255,252
6070 DATA 0,0,0,0,0,0,0,0,0,0
6100 REM** EXPLODE DATA    **
6110 DATA 0,0,0,0,0,0,0,16,0,
0,8,0,4,16
6120 DATA 0,3,2,64,1,56,128,
12,255,144
6130 DATA 1,238,40,5,151,0,11,
121,0,1
6140 DATA 183,0,25,214,96,0,
236,48,6,24
6150 DATA 152,3,98,0,8,51,0,
0,96,128,0
6160 DATA 64,0,0,0,0,0,0,0,0
6200 REM** DEPTH CHRG DATA **
6210 DATA 0,0,0,0,0,0,0,0,0,
0,0,0,0,0
6220 DATA 0,0,0,32,0,0,32,0,
0,32,0,0,32,0
6230 DATA 0,0,0,0,0,0,0
6240 DATA 2,0,0,2,0,0,2,0,0,
2,0,0
6250 DATA 0,0,0,0,0,0,0,0,0
6260 DATA 0,0,0,0,0,0,0,0,0
6300 REM** SUBMARINE DATA  **
6310 DATA 0,0,0,0,0,0,0,0,0,
0,0,0
6320 DATA 0,8,0,0,12,0,0,12,0
6330 DATA 0,12,0,0,28,0,0,60,0
6340 DATA 0,126,0,199,255,255
6350 DATA 239,255,255,127,
255,255
6360 DATA 255,255,254,199,
255,254
6370 DATA 0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0
```



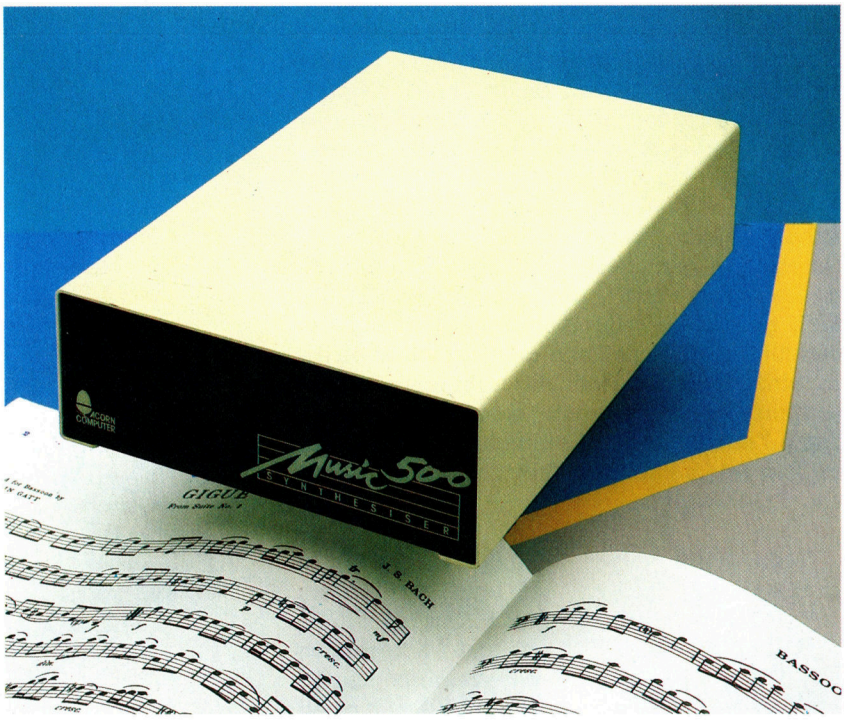
1	2	3
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	32
0	0	32
0	0	32
0	0	32
0	0	0
0	0	0
0	0	2
0	0	2
0	0	2
0	0	2
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0



1	2	3
0	0	0
0	0	0
0	0	0
0	0	0
0	8	0
0	12	0
0	12	0
0	12	0
0	28	0
0	60	0
0	126	0
199	255	255
239	255	255
127	255	255
255	255	254
199	255	254
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

Die Musikmaschine

Music 500 bietet Möglichkeiten des Musikmachens, wie sie bisher mit Heimcomputern nicht zu erreichen waren. Dank der Musik-Programmsprache, AMPLE und dem Stimmumfang einer 49 Töne umfassenden Klaviertastatur bietet Music 500 Ungewöhnliches.



Hybrids Music 500 ist wohl eines der fortschrittlichsten Computer-Musikgeräte für Heimcomputer. Mit 16 Oszillatoren und digitaler Wellenmodulation ausgestattet ist es weitaus teureren Synthesizersystemen durchaus ebenbürtig.

Das von Acorn für den Acorn B vertriebene Music-500-Paket wurde von Hybrid Technology entwickelt. Für einen Verkaufspreis von rund 800 Mark bekommt man ein Stück Elektronik in Kompaktbauweise, das in einem Diskettenstationsgehäuse steckt. Es wird mit dem 1-MHz-Bus des Acorn B und dem Stereo-Verstärker durch einen DIN-Standardstecker (mit 5 Pins) verbunden.

Im Gehäuse befinden sich 16 Oszillatoren, die über den Computer mit Frequenz (Pitch), Lautstärke, Balance und Wellenform einzeln programmiert werden können. Dank der mitgelieferten Software ist Music 500 im Preis-Leistungsverhältnis vielen derzeit angebotenen Synthesizern weit überlegen.

Die meisten Musiksynthesizer bedienen sich bei der Klangerzeugung einer der zwei folgenden Methoden. Die billigeren Geräte nehmen lediglich einen Ton auf – etwa eine Sinuswelle – und verzerren und filtern ihn dann, um den gewünschten Effekt zu erzielen. Wenngleich auf diese Art ein beachtliches Klangspektrum erzeugt werden kann, ist diese Methode doch sehr eingeschränkt, selbst wenn Experten damit arbeiten. Bei teuren Synthesizern bedient man sich mehrerer Sinuswellen, die einander modulieren. Die Amplitude der einen Welle wird dabei zur Frequenzkontrolle der anderen verwendet.

Dagegen erfolgt die Klangerzeugung beim Music 500 auf völlig andere Weise. Die durch das Gerät erzeugten Wellenformen werden digital zu einem Programmuster aufgebaut und „gesampled". Ergänzend zu dieser Möglichkeit bietet Music 500 auch Modulationstechniken, die nicht allein auf Frequenzmodulation beschränkt sind, sondern auch Kanäle mit Hilfe von Ring- und Phasenmodulation miteinander kombinieren können. Die spezielle Software wird auf Cassette geliefert, kann aber mit dem beigefügten Programm leicht auf Diskette überschrieben werden. Zum Lieferumfang gehört AMPLE, eine Musik-Programmiersprache, die alles bietet, um das Beste aus dem Music 500 herauszuholen. In vielen Bereichen ähnelt diese Sprache FORTH oder LOGO. Besonders bei der Anwendung, da es sich um eine Programmiersprache auf Wort-Basis handelt, mit der die Programmierer Paßwörter und Prozeduren definieren können. Eine Komposition könnte folgendermaßen aussehen:

```
10 .% FLIGHT OF THE BUMBLE BEEB
20 .setup
30 .play
```

Das Wort „setup" ist vom Anwender definiert und folgendermaßen programmiert worden:

```
10 .'setup'
20 .[
30 .setsynth
40 .setbass
50 .setdrums
60 .setcymbal
70 .]
```

Die Begriffe, mit denen „setup" definiert wurde, sind jeweils zur Definition unterschiedlicher Klänge vorprogrammiert, so beispielsweise „setdrums" oder „setcymbal". Einige vorgegebene Wellenformen sind im Programm enthalten, neue Wellenformen können jedoch leicht geschaffen werden. Sie lassen sich dabei entweder durch Begriffe der relativen Proportionen der ersten 16 Harmonien eines Tones definieren oder in mathematischer Form „auszeichnen". Letztere Methode gibt Music 500 die Möglichkeit, „Rauschen" zu erzeugen, indem eine zufällige Wellenform erstellt wird.

Die Definition für den Cymbal-Klang (Cymbal=Becken) könnte etwa folgendermaßen aussehen:

```
10 .'cymbal'
20 .[
30 .3 WMOD 0 rand! 128 FOR(RAND?
   INDEX WG!) FOR WGC
40 .1 CHAN
50 .3 WAVE ON RM—1POS
60 .2 CHAN
70 .3 WAVE 2000 OFFSET 1 POS
80 .]
```

Die mit der FOR . . .NEXT-Schleife vergleichbare AMPLE-Anweisung setzt jeden Punkt der Welle auf einen Zufallswert. Diese Wellenform wird für zwei klangerzeugende Kanäle (zwei Oszillatoren) benutzt, wobei die Höhe des zweiten etwa über der des ersten liegt. Das Ergebnis ist ein rauschendes, krachendes Geräusch.

Alternativ läßt sich eine melodischere Wellenform definieren:

```
10 .'synth'
20 .[
30 .WZERO 181 1 WH! 135 2 WH! 181 3 WH!
          62 4 WH! 76 5 WH! 37 6 WH!
          14 7 WH! 3 8 WH!
40 .WHG WGC 3
50 .]
```

Diese Wellenform entsteht durch variierte Proportionierung der ersten acht Harmonien. Höhe und Amplitudenhüllkurve werden ähnlich erzeugt, wobei entweder einfache Auslassungsformen benutzt werden oder das, was der Anwender spezifiziert hat.

Mit dem Wort „play" kann jeder der definierten Klänge aufgerufen werden.

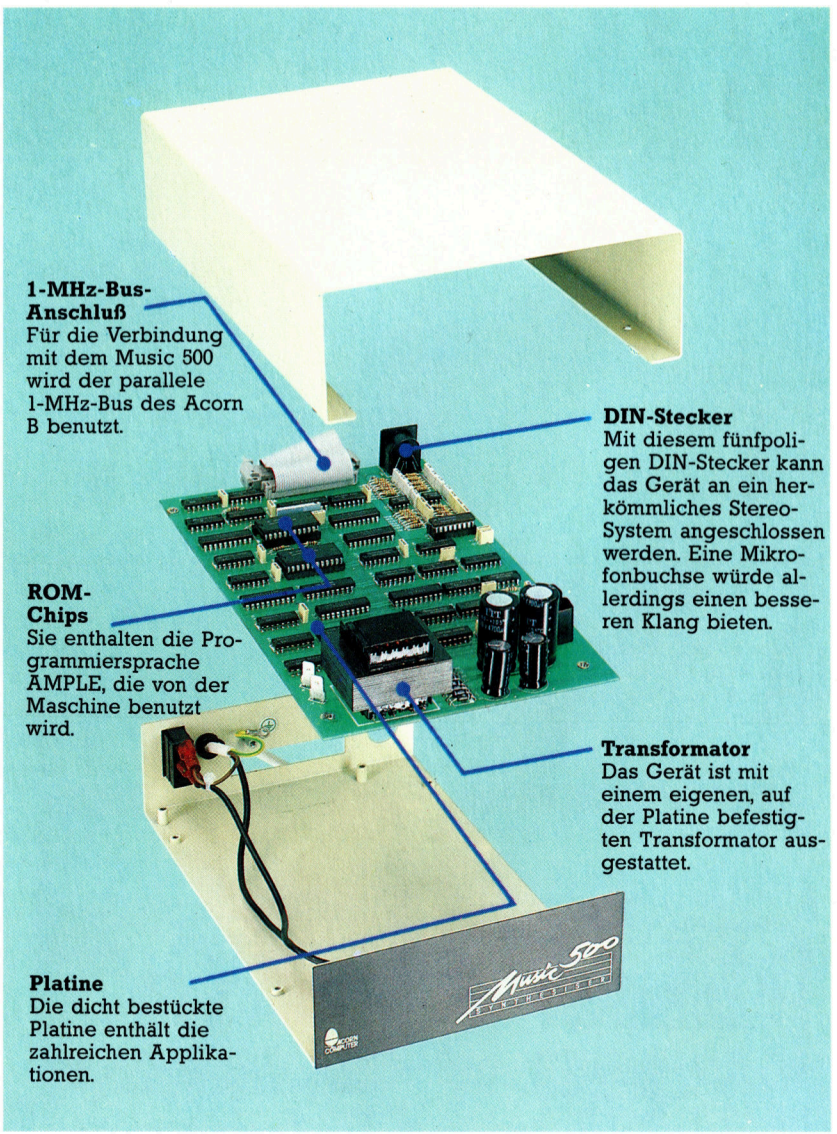
```
10 ."play"
20 .[
30 .4 PLAYERS
40 .1 PLAY (piano melody)PLAY
50 .2 PLAY (bass bassline)PLAY
60 .3 PLAY(drums rhythm)PLAY
70 .4 PLAY (cymbal crash)PLAY
80 .GO
90 .]
```

Dem System wird die Zahl der benötigten „Spieler" eingegeben. Jeder Spieler bekommt einen Klang und eine Partitur zugewiesen, und schließlich werden sie aufgefordert, gemeinsam zu spielen. Diese Mehrspur-Möglichkeit von AMPLE erlaubt eine Nutzung des Music 500 als schriebe man eine Partitur für ein Orchester. Die Sprache insgesamt ist modular und hierarchisch aufgebaut. Selbst integrierte Wörter können neu definiert werden. Die Anwendung dieser Option ist allerdings recht kompliziert. Es ist aber möglich, beispielsweise einen Taktstrich neu zu definieren, so daß die zweite Note in jedem Takt etwas lauter gespielt wird.

Acorn beabsichtigt, demnächst eine 49 Töne umfassende Klaviertastatur auf den Markt zu bringen, die mit dem User Port des Acorn B verbunden wird. Zum Lieferumfang gehört auch AMPLE-Software wie die „Notenverarbeitung", mit der das Spiel auf der Tastatur aufgezeichnet wird. So kann man es anschließend abspielen und editieren.

Music 500

Oszillatoren	16
Frequenzbreite	0-20KHz
Tonauflösung	16tel Halbton
Wellenformen	
harmonisch	16 Harmonien
geometrisch	128 Punkte
Hüllkurven	
(Höhe/Amplitude)	17 Segmente
Schrittlänge	0-320 Sekunden
Zeitschritte	10 ms
Stereopositionen	7
Wellenformgenauigkeit	8-bit logarithmisch
Samplerate	46875/sec
Frequenzauflösung	0,0056 Hz
Time Base	255ms-655 ms/zyklisch
Zusammenspiel	1–9 Stimmen



Psycho-Attacke

In erfolgreichen Programmen für Heimcomputer sind Spielhallenelemente ebenso enthalten wie Teile aus strategischen und Abenteuerspielen; denn heute ist mehr gefragt als nur schnelle Reaktion.

Das Programm „Psytron" ist ein sehr vielseitiges und aufregendes Spiel mit schnell bewegter Grafik. Der Spieler übernimmt die Rolle des Psytron, eines Wesens – halb Mensch, halb Computer –, das auf dem Planeten Betula 5 eine Raumkolonie kommandiert. Diese besteht aus mehreren Gebäuden, die jeweils besondere Funktionen haben. Lebensversorgungssysteme sind für die menschlichen Bewohner der Kolonie wichtig, und für den Computerbetrieb werden Kraftwerke benötigt. Wichtigster Bestandteil aber ist die zentrale Energieversorgung, bei deren Ausfall der gesamte Betrieb der Kolonie zum Erliegen käme. Über Tastatur oder mittels Joystick hat der Spieler einen Rundumblick von 360 Grad.

Auf den unteren Spielebenen ähnelt Psytron anderen Arcadespielen. Der Spieler verfügt über eine Reihe von Waffen, mit denen er die Angriffe Außerirdischer abwehrt. Erst ab der vierten Spielebene werden die strategischen Elemente des Programms offensichtlich.

Auf Ebene eins steuert der Psytron einen „Droiden", der benutzt wird, um die dreibeinigen Saboteure zu beseitigen, die in die Luftschleusen „teleportiert" wurden. Sie sollen die Verbindungsgänge zu den Gebäuden sprengen. In Ebene zwei und drei hat der Spieler Gelegenheit, Ufos abzuschießen, entweder einzeln oder bei Verwendung des „Disruptors" alle auf einmal. Allerdings ist der Disruptor nicht sehr haltbar. Zu zehn Prozent besteht die Möglichkeit, daß er beim Einsatz explodiert.

Nach dem Erreichen der vierten Ebene muß der Spieler strategische Entscheidungen fällen, basierend auf der Schwere der Schäden, die die einzelnen Einrichtungen erlitten haben. Während dieser Phase setzen die Außerirdischen ihren Angriff fort, bombardieren strategisch wichtige Bereiche der Station und setzen terroristische Selbstmord-Kommandos ab. In dieser Ebene gibt es die Option „Freezetime". Durch Druck auf die Return-Taste kann der Spieler den Ablauf „einfrieren", um so Schadensmeldungen aufzunehmen. Dabei ist eine Reihe von Faktoren zu berücksichtigen: Die Zahl der verletzten oder toten Einwohner, Zustand der verbliebenen Versorgungseinrichtungen und der Kraftstation. In dieser Freezetime können Reparaturen durchgeführt und die Bewohner der Station dorthin gebracht werden, wo sie sicherer sind. Auch dabei ist geschickt zu planen: Der Psytron muß überlegen, daß Reparaturtrupps mehr Nahrung und Sauerstoff benötigen als nicht arbeitende Menschen. Es kann also sinnvoll sein, einige Gebäude aufzugeben, um Treibstoff, Sauerstoff und Vorräte zu sparen.

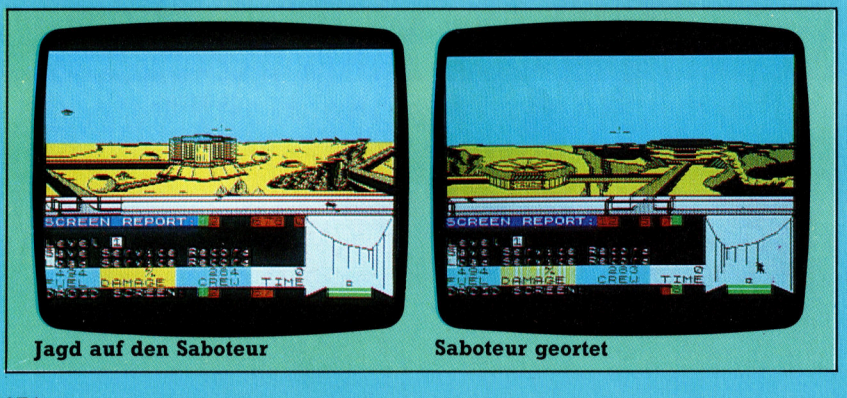
Wirtschaftlich handeln!

Auf der fünften Ebene hat der Spieler Gelegenheit, mit einem Versorgungsschiff in Verbindung zu treten, das Vorräte zur Kolonie „herunterbeamen" kann. Dabei ist zu berücksichtigen, ob bei vorangegangenen Angriffen die Dockanlagen beschädigt worden sind.

Auf der sechsten und letzten Ebene sind allein strategische Faktoren ausschlaggebend. Einziges Ziel dieses Stadiums ist es, eine Stunde zu überleben und die Station unter Ausnutzung aller zuvor erlernten Möglichkeiten in Betrieb zu halten. Die Anzahl der angreifenden Schiffe nimmt zu, das Tempo wird schneller, und es scheint bald unmöglich, angesichts der zahlenmäßig gewaltigen Angreifer die lebenswichtigen Einrichtungen der Kolonie intakt zu halten.

Die Computerspiele haben sich seit „Space Invaders" enorm weiterentwickelt. Psytron ist ein hervorragendes Beispiel dafür, welche Alternativen es zu den „Ballerspielen" gibt, die bisher den Markt beherrschten.

Diese Szenen aus Psytron zeigen die Aktion auf Ebene eins. Während der Droid die Saboteure durch die Gänge jagt, gleitet der Blick über die gesamte Basis. Mit Hilfe des Fensters in der unteren rechten Bildecke sieht der Spieler, was der Droid sieht.



Jagd auf den Saboteur — Saboteur geortet

Psytron: Für ZX Spectrum (48 K) und C 64
Hersteller: Beyond Software
Vertrieb: Rushware
Autoren: Tayo Olowu und Paul Voysey
Joystick: Als Option
Format: Cassette (Spectrum), Cassette oder Diskette (C 64)

Pixelzeichnungen

Diese Serie zeigt anhand vieler Beispiele, wie sich die Microprozessoren 6502 und Z80 mit Hilfe der Assemblersprache für grafische Anwendungen einsetzen lassen.

Die hochauflösende Grafik des Commodore 64 haben wir bereits in früheren Artikeln untersucht. Dabei werden zuerst die hohe Auflösung über den Video Interface Chip (VIC) eingeschaltet und der Zeiger der Basisadresse des Zeichensatzes geändert. Nachdem der Bildschirmspeicher (ein Block von acht KByte, der bei Speicherstelle 8192 anfängt) gelöscht ist, schließt die Initialisierung der Memory-Map des Bildschirms als Farbspeicher (Speicherstelle 1024 bis 2033) die vorbereitenden Schritte ab. Bei einer Hintergrundfarbe ist diese Aufgabe relativ einfach, bei vielfarbigen Darstellungen kann sie jedoch recht kompliziert werden.

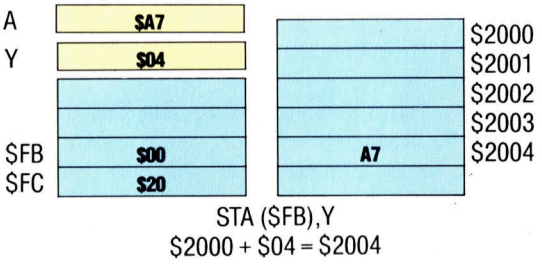
Unser Programm schaltet die hohe Auflösung ein bzw. aus und führt alle Berechnungen für die Darstellung eines Punktes durch. Der erste Teil enthält vorbereitende Schritte: In den Bytes des normalen Bildschirmspeichers wird die Farbinformation gespeichert, und die Bit-Map von acht KByte wird gelöscht.

Durch den Einsatz des HRSFLAG läßt sich diese Routine sowohl beim Ein- als auch beim Ausschalten der hohen Auflösung aufrufen. Damit die Bit-Map jedoch nicht bei jedem Aufruf gelöscht wird – beispielsweise wenn eine Figur auf dem Bildschirm erhalten bleiben soll – zeigt CLRFLG an, ob der Bildschirm gelöscht werden soll oder nicht. Das Ablaufdiagramm zeigt, wie der Maschinencode diese beiden Flags einsetzt.

Die Speicherblöcke von 256 und weniger Bytes lassen sich unkompliziert über eine Maschinencodeschleife ansprechen. Der folgende Programmteil setzt mit der Methode der absoluten indizierten Adressierung in jede Speicherstelle der Adressen BASE bis BASE+255 die Zahl $03 ein (insgesamt 256 Speicherstellen).

```
        LDY $00
        LDA $03
NEXT    STA BASE,Y
        DEY
        BNE NEXT
```

Beachten Sie, daß bei dieser Technik zunächst BASE angesprochen wird und dann die restlichen Bytes des Blocks von BASE+255 hinunter bis BASE+1. Da das Programm jedoch nicht nur auf einen Block von 256 Bytes zugreifen soll, müssen wir eine Adressiermethode einsetzen, die als „nachindiziert und indirekt" bezeichnet wird und über die Zero-Page Adressen im gesamten Speicher ansprechen kann. Das Betriebssystem des Commodore 64 belegt zwar den größten Teil der Zero-Page, doch wurden für Maschinencodeprogramme einige Bytes freigehalten. Zwei Bytes dieser Art sind 251 und 252 ($FB und $FC). Bei dieser Adreßmethode setzt der Computer voraus, daß das niederwertige Adreßbyte in dem angegebenen Byte der Zero-Page enthalten ist und das höherwertige Adreßbyte in der darauffolgenden Speicherstelle. Ein Befehl wie STA($FB), Y, bei dem Y $04 enthält und in $FB und $FC $00 und $20 gespeichert sind, berechnet die Adresse folgendermaßen:

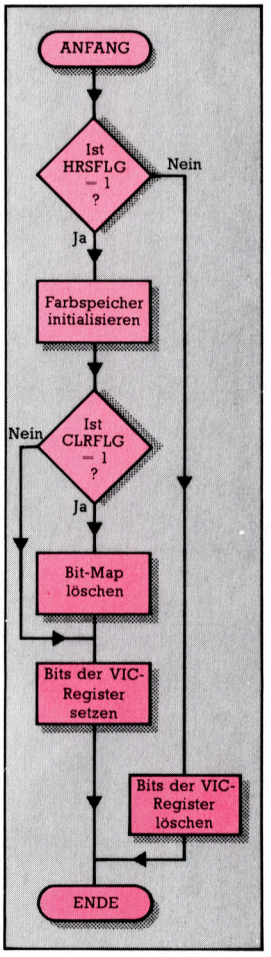


Mit einer ähnlichen Methode läßt sich ein ganzer Block von 256 Bytes adressieren. Da auch das höherwertige Byte der Adresse BASE angesprochen werden kann, ist diese Adressierungsart sehr flexibel. Eine Inkrementierung des höherwertigen Bytes erhöht BASE um 256 (bzw. setzt sie an den Anfang des nächsten Speicherblocks). Damit lassen sich Farbinformationen im normalen Bildschirmspeicher unterbringen und die Bit-Map löschen.

Der Bildschirmspeicher reicht von $0400 bis $07E7. Er besteht aus drei Blöcken zu je 256 Bytes und einem Rest von $E7 Bytes. Der mit „Farbspeicher" bezeichnete Assemblerblock (siehe Programmliste) lädt mit der nachindizierten indirekten Adressierung jedes einzelne Byte mit der Farbinformation. Dabei enthalten die Variablen SCBLO und SCBHI die nieder- und hochwertigen Bytes der Anfangsadresse des normalen Bildschirmspeichers, SCBLK die Nummer der 256-Byte Blöcke und SCREM den Rest. Auf der Zero-Page speichert P-TR das niederwertige Byte der Basisadresse.

Der zweite Teil des Maschinencode-Listings errechnet, welches Bit der Bit-Map einer bestimmten (X,Y)-Koordinate entspricht. X kann bei hoher Bildschirmauflösung Werte von 0 bis

Diese Routine führt drei Aufgaben aus: Sie initialisiert den Farbspeicher, löscht die Bit-Map und setzt oder löscht die Register für hohe Auflösung je nach Status der Flags HRSFLG und CLRFLG.



319 annehmen und Y von 0 bis 199. Dabei lassen sich alle Y-Werte in einem Byte speichern, die X-Werte über 255 jedoch nur in zwei Bytes. Mit BASIC werden die Bits der X- und Y-Koordinaten folgendermaßen errechnet:

```
1000 HB=XAND248:VBYTE=INT(Y/8)
1010 RMY=YAND7:RMX=XAND7
1020 ROW=VBYTE*320+HB
1030 BYTE=BASE+ROW+RMY
1040 POKEBYTE,PEEK(BYTE)OR(2↑(7-RMX))
```

Der entsprechende Maschinencode für die gleichen Berechnungen ist komplizierter, da HB, ROW, BASE und BYTE aus je zwei Bytes bestehen. Der größte Teil des Codes erklärt sich von selbst. Zwei Abschnitte verdienen jedoch besondere Beachtung – die Teilung von Y durch acht und die Multiplikation von VBYTE mit 320. Im Maschinencode ist die Teilung durch ein Vielfaches von Zwei einfach:

```
45      =00101101
45/2=22=00010110
22/2=11=00001011
11/2= 5=00000101
 5/2= 2=00000010
 2/2= 1=00000001
 1/2= 0=00000000
```

Jede Zweierteilung verschiebt die Bits der geteilten Zahl um eine Stelle nach rechts. Eine Teilung durch acht läßt sich daher mit drei logischen Rechtsverschiebungen (LSR – Logical Shift Right) ausführen. Da wir das Ergebnis als Ganzzahl benötigen, sind die nach rechts „herausfallenden" Bits nicht wichtig. Sie stehen bei Bedarf jedoch zur Verfügung, da sie von LSR in den Übertrag übernommen werden. Eine Multiplikation mit Zwei läßt sich ähnlich einfach durchführen: Die Bits des Multiplikanden brauchen nur um eine Stelle nach links verschoben zu werden (ASL). Auf genau diese Weise wird VBYTE mit 320 multipliziert. 320 läßt sich in 5x64 zerlegen und 64 in 2x2x2x2x2x2. Wenn wir VBYTE fünf Mal mit sich selbst addieren und dann sechs ASLs ausführen, ist die Zahl mit 320 multipliziert. Allerdings ergeben sich Schwierigkeiten, weil das Ergebnis über 255 liegt und zwei Bytes benötigt.

Beide Routinen lassen sich über den SYS-Befehl eines BASIC-Programms aufrufen. Dabei muß jedoch berücksichtigt werden, daß das BASIC-Programm nach Ausführung des Maschinencodes fortgesetzt werden soll. Da BASIC-Interpreter wie Maschinencode die Register X, Y und A einsetzen, müssen sie vor Aufruf des Maschinencodes gespeichert und danach wieder zurückgeladen werden. Diese Aufgabe läßt sich am leichtesten mit dem Stack ausführen. Dem Maschinencode können die Werte der Koordinaten, Farben und Flags per POKE übermittelt werden (siehe BASIC-Programm).

```
1 GOTO 200
2 POKE 53265,PEEK(53265)AND223
3 POKE 53272,PEEK(53272)AND240OR4
4 STOP
200 REM  **** C64 HI-RES DEMO    ****
210 :
220 POKE 56,32:CLR:      REM LWR MEMTOP
250 GOSUB 3000:      REM INITIALISE S/R
350 :
360 REM  **** SET UP HIRES MODE ****
370 :
380 PRINT CC$:PRINT :PRINT
390 INPUT"FOREGROUND COLOUR";FG
400 INPUT"BACKGROUND COLOUR";BG
410 TT=FG*16+BG:     REM CALC. COLOUR
420 POKE COLOUR,TT: REM COL TO M/C S/R
430 POKE HRSFLG,1:  REM SET HIRES ON
440 POKE CLMFLG,1:  REM CLRHIRES SCRN ON
450 SYS BEGIN:       REM ENTER M/C S/R
460 :
500 REM  **** DRAW PATTERN       ****
510 :
515 Z=0:Y1=50:Y2=150:X1=160:SP=6
520 FORY=Y1 TO Y2 STEP SP
530 FORX= Y2-Z TO Y2+Z STEP SP
540 GOSUB1000:       REM PLOT POINT
550 NEXT X
555 Z=Z+SP
557 NEXT Y
560 :
565 GETJ$:IFJ$<>""THEN 565
570 GETA$:IF A$=""THEN 570:REM AWAIT KPRESS
580 :
600 REM  **** CLEAR HIRES SCREEN ****
605 POKE HRSFLG,1:POKE CLMFLG,1
610 SYS BEGIN
620 :
630 REM  ****  LINE DEMO         ****
640 X1=0:X2=300:Y1=0:Y2=190:SP=1
670 GOSUB1500:           REM LINE PLOT
690 :
695 GETJ$:IFJ$<>""THEN695
700 GETA$:IF A$=""THEN700:REM AWAIT KPRESS
710 :
720 REM  ****   RESTORE SCREEN   ****
730 :
740 POKE HRSFLG,0:   REM HRES OFF
750 SYS BEGIN
760 PRINT CC$:PRINT :PRINT
770 PRINTTAB(9)"****END OF PROGRAM****"
780 END
999 :
1000 REM *** HIRES PLOT SUBROUTINE ***
1010 :
1020 XHI=INT(X/HX):XLO=X-XHI*HX
1030 POKE XBYTE,XLO:POKE XPAGE,XHI:POKE YBYTE,Y
1035 SYS PLOT:        REM ENTER PLOT S/R
1040 RETURN
1500 REM *** LINE  PLOT SUBROUTINE ***
1550 C9=(Y2-Y1)/(X2-X1):C8=C9*X1-Y1
1600 FORX=X1 TO X2 STEP SP
1650 Y=X*C9-C8
1700 GOSUB 1000:          REM PLOT POINT
1750 NEXT X
1800 RETURN
3000 REM *** INITIALISE SUBROUTINE ***
3020 CC$=CHR$(147):       REM CLEAR SCRN
3025 HX=256
3030 HRSFLG=49408:        REM $C100
3040 CLMFLG=49409:        REM $C101
3050 COLOUR=49410:        REM $C102
3060 XBYTE =49411:        REM $C103
3070 XPAGE =49412:        REM $C104
3080 YBYTE =49413:        REM $C105
3085 BEGIN =49422:        REM $C10E
3090 PLOT  =49539:        REM $C183
3095 PRINT CC$:PRINT:PRINT
3100 PRINTTAB(9)"****M/CODE LOADER****"
3150 PRINTTAB(9)"1) M/CODE IS ON TAPE"
3200 PRINTTAB(9)"2) M/CODE IS IN DATA"
3250 PRINTTAB(9)"3) M/CODE IS IN MEM."
3300 PRINT"HIT OPTION NUMBER"
3350 FOR LP=0 TO 1 STEP 0
3400 GET OP$
3450 IF OP$)"0" AND OP$<"4" THEN LP=1
3500 NEXT LP
3600 ON VAL(OP$) GOSUB 4000,5000,6000
3900 RETURN
4000 REM** LOAD MCODE FROM TAPE S/R **
4100 PRINT "INSERT TAPE CONTAINING MACHINE CODE S/R"
4200 IF A=0 THEN A=1:LOAD "PLOTSUB.HEX",1,1
```

```
4900 RETURN
5000 REM** LOAD MCODE FROM DATA S/R **
5005 PRINTTAB(11)"*****LOADING*****"
5010 FOR I=HRSFLG TO HRSFLG+312:READ A
5020 POKE I,A:S=S+A:NEXT I
5030 READ CC:IF CC<>S THEN PRINT"CHECKSUM
ERROR":END
5040 DATA 2,0,255,255,2,2,255,255,2,18
5050 DATA 255,255,2,2,72,138,72,152,72
5060 DATA 173,0,193,240,83,169,0,133,251
5070 DATA 169,4,133,252,162,3,160,0,173
5080 DATA 2,193,145,251,136,208,251,230
5090 DATA 252,202,48,8,208,244,145,251
5100 DATA 160,231,208,238,173,1,193,240
5110 DATA 24,169,0,133,251,169,32,133
5120 DATA 252,162,32,160,0,169,0,145,251
5130 DATA 136,208,251,230,252,202,208
5140 DATA 246,173,24,208,41,240,9,8,141
5150 DATA 24,208,173,17,208,9,32,141,17
5160 DATA 208,76,125,193,173,24,208,41
5170 DATA 240,9,4,141,24,208,173,17,208
5180 DATA 41,223,141,17,208,104,168,104
5190 DATA 170,104,96,72,138,72,152,72
5200 DATA 173,4,193,141,7,193,173,3,193
5210 DATA 41,248,141,6,193,173,3,193,41
5220 DATA 7,141,8,193,173,5,193,41,7,141
5230 DATA 10,193,162,3,78,5,193,202,208
5240 DATA 250,173,5,193,141,9,193,169,0
5250 DATA 141,11,193,141,12,193,162,5
5260 DATA 173,11,193,24,109,9,193,141,11
5270 DATA 193,202,208,243,162,6,14,12
5280 DATA 193,14,11,193,144,3,238,12,193
5290 DATA 202,208,242,173,11,193,24,109
5300 DATA 6,193,141,11,193,173,12,193
5310 DATA 109,7,193,141,12,193,173,11
5320 DATA 193,24,105,0,141,11,193,173,12
5330 DATA 193,105,32,141,12,193,173,11
5340 DATA 193,24,109,10,193,141,11,193
5350 DATA 173,12,193,105,0,141,12,193
5360 DATA 173,11,193,133,251,173,12,193
5370 DATA 133,252,169,1,141,13,193,56
5380 DATA 169,7,237,8,193,170,14,13,193
5390 DATA 202,208,250,160,0,177,251,13
5400 DATA 13,193,145,251,76,125,193
5410 DATA 38698:REM*CHECKSUM*
5900 RETURN
6000 REM** MCODE ALREADY IN MEM S/R **
6100 RETURN
```

Der Einsatz von PLOTSUB

Das BASIC-Programm zeigt die verschiedenen Stufen, die beim Einsatz von Maschinencoderoutinen für hohe Auflösung ablaufen:
1) Wenn Sie einen Assembler besitzen, können Sie das Assemblerprogramm eingeben, assemblieren, den Quellentext speichern und den Op-code unter dem Namen „PLOTSUB.HEX“ zwischen $C100 und $C238 ablegen.
2) Wenn diese Routinen von einem Programm aus aufgerufen werden sollen, müssen MEMTOP heruntergesetzt (siehe Programmzeile 220) und der Code von der Cassette geladen werden (siehe Unterprogramm 4000).
3) Das Unterprogramm bei Zeile 5000 läßt sich auch als BASIC-Programm speichern (zum Beispiel mit dem Namen MCODELOAD). Setzen Sie MEMTOP herunter und rufen Sie MCODELOAD auf, der nun den Maschinencode in den Speicher lädt. Geben Sie NEW ein und laden Sie die Programme, mit denen Sie arbeiten wollen. Die Routinen für die hohe Auflösung sind nun im Speicher und lassen sich mit entsprechenden SYS-Befehlen abrufen.
4) Der letzte DATA-Befehl enthält eine Prüfsumme – die Summe aller vorangehenden DATA-Befehle. Wenn das Programm mit der Meldung CHECKSUM ERROR abbricht, zeigt es an, daß die DATA-Befehle Fehler enthalten. Die Fehler müssen vor einem Neustart korrigiert werden.

```
;************************
;************************
;**                    **
;**  HIRES PLOTTING    **
;**                    **
;**    ROUTINE FOR     **
;**                    **
;**    COMMODORE 64    **
;**                    **
;************************
;************************
;
;
PTR     EQU $FB
MPBLO   EQU $00
MPBHI   EQU $20
SCBLO   EQU $00
SCBHI   EQU $04
SCBLK   EQU $03
SCREM   EQU $E7
MPBLK   EQU $20
        ORG $C100

HRSFLG DB   $00
CLRFLG DB   $00
COLOUR DB   $00
XLO    DB   $00
XHI    DB   $00
YLO    DB   $00
HBLO   DB   $00
HBHI   DB   $00
REMX   DB   $00
VBYTE  DB   $00
REMY   DB   $00
ROWLO  DB   $00
ROWHI  DB   $00
BPOS   DB   $00
;++++SWITCH TO HIRES MODE+++++
;++++AND CLEAR BIT MAP AREA+++
        PHA
        TXA
        PHA
        TYA
        PHA
        LDA HRSFLG
        BEQ RESET
;++++COLOUR SCREEN AREA+++++++
        LDA #SCBLO
        STA PTR
        LDA #SCBHI
        STA PTR+1
        LDX #SCBLK
        LDY #$00
        LDA COLOUR
AGAIN   STA (PTR),Y
        DEY
        BNE AGAIN
        INC PTR+1
        DEX
        BMI CLTEST
        BNE AGAIN
        STA (PTR),Y
        LDY #SCREM
        BNE AGAIN
CLTEST LDA CLRFLG
        BEQ HRESON
;++++CLEAR BIT MAP AREA+++++++
        LDA #MPBLO
        STA PTR
        LDA #MPBHI
        STA PTR+1
        LDX #MPBLK
        LDY #$00
        LDA #$00
NEXT    STA (PTR),Y
        DEY
        BNE NEXT
        INC PTR+1
        DEX
        BNE NEXT
;++++SET BIT MAP MODE+++++++++
HRESON LDA $D018
        AND #$F0
        ORA #$08
        STA $D018
        LDA $D011
        ORA #$20
        STA $D011
        JMP EXIT
;++++RESET TO NORMAL SCREEN+++
RESET   LDA $D018
        AND #$F0
        ORA #$04
        STA $D018
        LDA $D011
        AND #$DF
        STA $D011
;++++EXIT M/C+++++++++++++++++
EXIT    PLA
        TAY
        PLA
        TAX
        PLA
        RTS
;++++HIRES PLOT CALCULATION+++
        PHA
        TXA
        PHA
        TYA
        PHA
;++++CALCULATE HORIZ.BYTE+++++
        LDA XHI
        STA HBHI
        LDA XLO
        AND #$F8
        STA HBLO
        LDA XLO
        AND #$07
        STA REMX
;++++CALCULATE VERT.BYTE++++++
        LDA YLO
        AND #$07
        STA REMY
        LDX #$03
SHIFT   LSR YLO
        DEX
        BNE SHIFT
        LDA YLO
        STA VBYTE
;++++CALCULATE ROW++++++++++++
        LDA #$00
        STA ROWLO
        STA ROWHI
        LDX #$05
        LDA ROWLO
        CLC
        ADC VBYTE
        STA ROWLO
        DEX
        BNE FIVE
        LDX #$06
        ASL ROWHI
        ASL ROWLO
        BCC NCARRY
        INC ROWHI
        DEX
        BNE MULT
;++++ADD HORIZONTAL BYTE++++++
        LDA ROWLO
        CLC
        ADC HBLO
        STA ROWLO
        LDA ROWHI
        ADC HBHI
        STA ROWHI
;++++ADD HIRES MAP BASE+++++++
        LDA ROWLO
        CLC
        ADC #MPBLO
        STA ROWLO
        LDA ROWHI
        ADC #MPBHI
        STA ROWHI
;++++ADD REMAINDER OF YLO+++++
        LDA ROWLO
        CLC
        ADC REMY
        STA ROWLO
        LDA ROWHI
        ADC #$00
        STA ROWHI
;++++COMBINE 2 BYTES OF+++++++
;++++ADDRESS ON ZERO PAGE+++++
        LDA ROWLO
        STA PTR
        LDA ROWHI
        STA PTR+1
;++++CALCULATE PIXEL POSN.++++
        LDA #$01
        STA BPOS
        SEC
        LDA #$07
        SBC REMX
        TAX
        ASL BPOS
        DEX
        BNE POWER
;++++TURN ON PIXEL++++++++++++
        LDY #$00
        LDA (PTR),Y
        ORA BPOS
        STA (PTR),Y
        JMP EXIT
```

Natürliche Auslese

Viele Millionen Microprozessoren sind weltweit in Gebrauch, bei Videorecordern und Microwellengeräten ebenso wie bei Computern. Dabei wird der Heimcomputermarkt im wesentlichen von nur zwei Prozessortypen beherrscht: dem Z80 und dem 6502.

Bei den meisten Heimcomputern beschränkt sich die Frage nach dem Prozessor darauf, ob nun ein 6502 oder ein Z80 eingebaut ist. Nur bei wenigen Rechnern kommen andere Prozessoren zum Einsatz; der Dragon arbeitet beispielsweise mit einem 6809.

Im Jahr 1972 wurde der Baustein-Hersteller Intel von Datapoint gebeten, einen Chip herzustellen, der die Vielzahl von TTL(Transistor-Transistor-Logik)-Schaltkreisen in den damaligen Computer-Terminals ersetzen sollte. Intel entwickelte daraufhin den Microprozessor 8008, der acht Bits parallel verarbeitete und der ideale „logische" Ersatz für die Gatterbausteine in den Datapoint-Terminals gewesen wäre – aber er arbeitete zu langsam und kam nicht wie geplant zum Einsatz. Entwickler und Bastler stellten jedoch schon bald fest, daß der 8008 eine vielseitige Rechner-Zentraleinheit ist. Das war die Geburtsstunde der preisgünstigen Schreibtisch-Rechner.

Die Grenzen des 8008 hinsichtlich Leistung und Geschwindigkeit waren aber doch so eng gesetzt, daß Intel wenig später als Weiterentwicklung den 8080 herausbringen mußte. Er dominierte sehr schnell den Markt.

Fast zeitgleich mit der Ankündigung des 8080 durch Intel stellte Motorola unter der Bezeichnung 6800 einen ganz anders konzipierten 8-Bit-Microprozessor vor. Sein Leistungsvermögen war durchaus ebenbürtig, und beide Prozessoren schienen gleich gut als Basis für Microcomputer geeignet.

1974 entwickelte Gary Killdall, der spätere Chef von Digital Research, bei Intel ein Disketten-Betriebssystem (DOS) namens CP/M. Dieses System ermöglichte den Einsatz der neuen Floppy-Laufwerke bei Rechnern mit 8080-Prozessor. Es wurde jedoch von Intel nicht übernommen, weil man vorrangig Großrechner im Auge hatte, für die ohnehin ausreichend Software vorhanden war.

„Geniestreich" von Zilog

Die kleinen Rechner fanden aber zunehmend Verbreitung, und CP/M unterstütze den Umgang mit Dateien sehr wirksam. Das sicherte dem 8080 lange Zeit die Position als Marktführer und verhinderte, daß sich der 6800 durchsetzen konnte. Trotz zahlreicher Ansätze, ein vergleichbares DOS für den 6800 zu entwikkeln, war der Anschluß an die Umsatzzahlen des 8080 nicht zu schaffen.

Im Verlauf des weiteren Elektronik-Booms versuchten zwar etliche Hersteller, neue Prozessortypen einzuführen; sie scheiterten aber meist am Widerstand des Marktes, auf dem sich nur behaupten konnte, was eindeutige Vorteile aufwies. Auch die in Hard- und Software-Entwicklung getätigten Investitionen verhinderten das Aufkommen neuer inkompatibler Prozessoren.

Nur dem Z80 verhalf eine Art Geniestreich zum Druchbruch. Bei der Firma Zilog, einer Neugründung früherer 8080-Entwickler von Intel, nutzte man die Chance, den Befehlssatz des 8080 zu erweitern. Der 8080 machte nämlich nicht von allen verfügbaren Bitmustern Gebrauch. Unter Verwendung der „freien" Kommandos entstand bei Zilog ein Prozessor mit erheblich höherer Leistung, der aber noch alle 8080-Befehle verstehen und daher mit der vorhandenen Software arbeiten konnte.

Neben dem erweiterten Befehlssatz bot der Z80 ein weiteres wichtiges Plus: Während der 8080 zusätzlich einen externen Taktgeber und einen System-Steuerchip benötigte, hatten die Zilog-Entwickler die gesamte Prozessor-Logik

Ahnentafel

Die gängigen Microprozessoren lassen sich zwei Entwicklungslinien zuordnen, die von den Intel-Chips 4004 und 8008 bzw. vom Motorola 6800 ausgehen.
Diese Tafel veranschaulicht die Entwicklungsstufen und zeigt einige der zugehörigen Rechner, wobei die unbekannteren Prozessoren vielfach auch in wenig verbreiteten Rechnern stekken. Der Apple III ist wohl der einzige Bürorechner mit einem 6502 und der Olivetti M20 die einzige Universalmaschine mit einem Z8000. Beide Computer konnten sich aufgrund des geringen Softwareangebots nicht durchsetzen.

Intel

4004

8008

4004 und 8008: Ursprünglich als kompakter Ersatz für ganze Platinen voller TTL-Gatter gedacht, war der 4004 nur für 4-Bit-Parallelverarbeitung ausgelegt. Intel schaffte mit dem 8008 aber bald den Schritt in den 8-Bit-Bereich. Dieser Chip wurde von Bastlern und Profis schnell als ideale Basis für den Selbstbau von Microcomputern erkannt.

Motorola

6800

6800: Der große Konkurrent des 8080, mit einer völlig anderen Struktur bei vergleichbarer Leistung. Daher prägten sich auch bei den Anwendern zwei Richtungen aus: Die einen gaben dem Konzept des Intel 8080 den Vorzug, die andern schlossen sich der 6800er-Philosophie von Motorola an.

6809

6809: Motorolas eigene Weiterentwicklung des 6800 gilt als der leistungsfähigste 8-Bit-Prozessor. Leider kam er zu spät und wurde nur bei wenigen Rechnern eingesetzt.

Dragon 32

68000

68000: Dem 16-Bit-Prozessor von Motorola blieb der große Erfolg versagt, weil der 8088 das Feld bereits beherrschte. Immerhin hat sich Sinclair beim QL für die bescheidenere Version 68008 entschieden.

68008

Sinclair QL

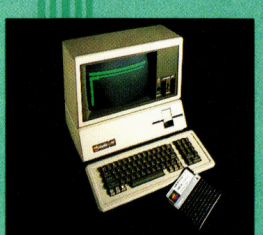
Apple Lisa

MOS Technology

6502

Commodore PET

6502: Der 8-Bit-Chip von MOS-Technology ist mit dem 6800 zwar eng verwandt, aber nicht kompatibel. Der niedrige Preis machte den 6502 für Hobbyisten und Entwickler interessant. Er ist unter anderem bei den erfolgreichen PET- und Apple-Computern anzutreffen. Auch heute wird er bei Heimcomputern viel eingesetzt, etwa von Oric und Atari.



Apple III

IBM PC

8080

8085

8080: Der erste Prozessor, der nach Erscheinen des CP/M-Betriebssystems die Entwicklung von Heimcomputern und Bürorechnern möglich machte. Daraus entstand bei Intel selbst der 8085, der mehr leistet und weniger externe Chips benötigt. Davon gibt es eine stromsparende CMOS-Version für tragbare Geräte.

Tandy TRS80 Modell 100

8086

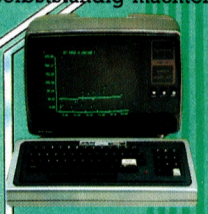
8088

8086 und 8088: Der 8088 ist eine abgemagerte Version des 8086, die mit älteren Zusatzbausteinen funktionsfähig ist und deshalb eine Zeitlang sehr beliebt war. Der Einsatz beim Sirius und beim IBM PC machte den 8088 zum verbreitetsten 16-Bit-Prozessor. Bei den meisten neuen Rechnern wird inzwischen der leistungsfähigere Chip 8086 verwendet.

Zilog

Z80

Z80: Eine verbesserte 8080-Version von Intel-Entwicklern, die sich unter dem Namen Zilog selbstständig machten.



Tandy TRS80 Modell 1

Z8000

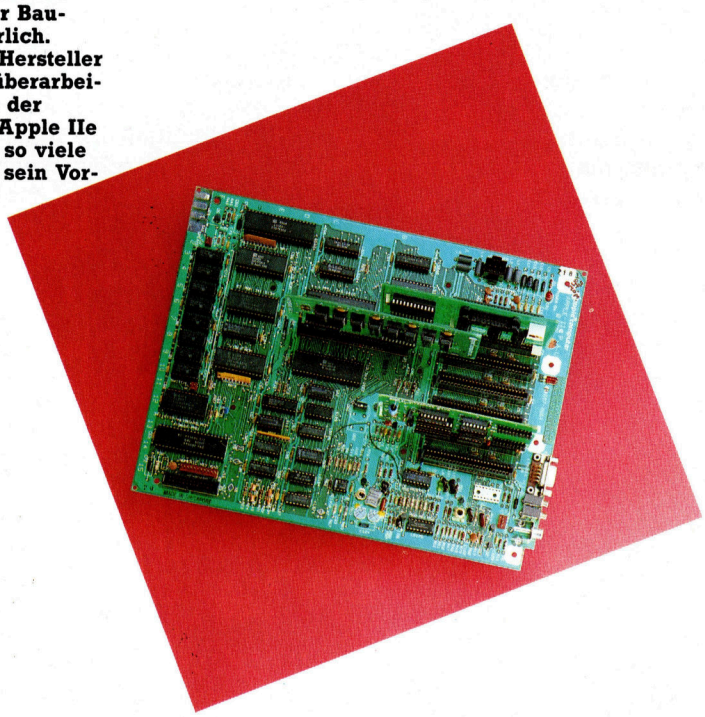
Z8000: Zilogs erster 16-Bit-Prozessor hat sich auf dem Markt nicht durchgesetzt, auch durch den Einsatz des 8088 bei IBM bedingt. Zilogs zweiter 16-Bit-Anlauf ist der Z800, der volle Kompatibilität mit dem Z80 (also auch dem 8080) verspricht.

Olivetti M20

in einen einzigen Chip gesteckt. Das machte den Z80 für Rechner-Hersteller trotz des relativ hohen Preises sehr attraktiv.

Obwohl sein Marktanteil sich in Grenzen hielt, hatte der 6800 bei einer Reihe von Entwicklern und Programmierern durchaus seinen festen Platz. Das veranlaßte Motorola schließlich zum Bau eines „hochgezüchteten" 8-Bit-Nachfolgers mit der Bezeichnung 6809. Unglücklicherweise kam fast gleichzeitig ein Konkurrent namens MOS-Technology ebenfalls mit einer verbesserten 6800-Version heraus, nämlich dem 6502. Dieser Prozessor ist der bekannteste aus der 6500er-Reihe, deren Vertreter trotz unterschiedlicher Ausstattung und Leistung alle mit dem gleichen Befehlssatz arbeiten.

Der 6502 ist zwar vom Konzept her eng mit dem 6800 verwandt, aber nicht mit ihm kompatibel – weder in bezug auf die Hardwareanforderungen noch auf die Software. Dagegen beinhaltet der Z80 den gesamten Befehlssatz des 8080 und kann ihn – vorausgesetzt, die CPU-Platine wurde entsprechend modifiziert – auch softwaremäßig ersetzen.



Bei der Verwendung hochentwickelter Chips sind zur Erfüllung der gleichen Funktionen immer weniger Bausteine erforderlich. Nachdem der Hersteller den Apple II überarbeitet hatte, wies der Nachfolgetyp Apple IIe nur noch halb so viele Chips auf wie sein Vorgänger.

Mit dem Befehlssatz des 6502 ist jeder 6800-Programmierer sofort vertraut. Der 6502 bietet gegenüber dem 6800 mehr Möglichkeiten und bereitet weniger Schnittstellenprobleme, ist aber eben nicht kompatibel. Deshalb läßt sich die gegenwärtige Marktposition des 6502 nur so erklären: Der 6502 hatte das Glück, als „Mitreisender" vom Senkrechtstart der Apple-Rechner zu profitieren.

Bei Erscheinen des ersten Apple war für Schreibtisch-Rechner noch das Steckkarten–Konzept vorherrschend: Die Systemkomponenten wurden über S-100-Stecker (100polig) auf ein „Motherboard" mit Bus- und Versorgungsleitungen aufgesteckt. Das Grundsystem bestand aus Netzteil, Motherboard, CPU-Karte, Speicherkarte, Videokarte, dazu meist noch Drucker-Interface-Karte und Floppy-Steuerkarte – kein Wunder, daß solch ein Rechner viel teurer sein mußte als ein Ein-Platinen-Computer wie der Apple.

Trotz des günstigen Preises kam der eigentliche Durchbruch für die Apple-Mannschaft unter Stephan Wozniak erst mit dem VisiCalc-Softwarepaket. Dieses Programm fand bei Geschäftsleuten großen Anklang, weil sich finanzielle Planungen damit viel schneller und müheloser als mit Taschenrechner, Bleistift und Papier durchführen ließen. VisiCalc war so erfolgreich, daß es Apple enorme Verkaufszahlen einbrachte und gleichzeitig den 6502 als einen der führenden Microprozessoren etablierte. Auch Commodore wählte für den PET und dessen Nachfolger den 6502.

Während sich bei den 8-Bit-Rechnern der 6502 behaupten konnte, hob sich schon der Vorhang für die 16-Bit-Computer. Intel offerierte dafür den 8086 und 8088, Motorola den 68000 und Zilog den Z8000. Alle drei Prozessoren haben ihre Stärken, aber keiner ist mit dem jeweiligen 8-Bit-Vorläufer kompatibel. Zum Glück für Intel kamen Digital Research und Microsoft sehr schnell mit Betriebssystemen für den 8086 und 8088 heraus (CP/M–86 bzw. MS-DOS), während Zilog und Motorola von den Sotwarehäusern arg vernachlässigt wurden. Vorteilhaft für die Intel-Prozessoren wirkte sich auch aus, daß IBM seinen Personal Computer mit dem 8088 bestückte.

Neuer Kampf um den Markt

Der Kampf um den 16-Bit-Prozessormarkt scheint eine Neuauflage der 8-Bit-Geschichte zu werden. Intels 8086 und der 8088 sind ähnlich wie zuvor der Z80 und der 6502 zum Standard geworden, wofür die Software-Unterstützung durch CP/M-86 und MS-DOS sowie der Einsatz in Bestsellern wie dem IBM PC und dem Sirius entscheidend waren. Der Z8000 von Zilog wurde nur bei einem Universalrechner, nämlich dem Olivetti M20 verwendet. Olivetti gelang es aber nicht, die entsprechende Software zu erstellen, und mußte schließlich das Gerät mit einer 8086-Steckkarte aufrüsten, um darauf MS-DOS und CP/M-86 fahren zu können. Daraufhin nahm Zilog eine 16-Bit-Neuentwicklung namens Z800 in Angriff, die mit dem Z80 softwarekompatibel ist.

Trotz der jetzigen Expansion im 16-Bit-Bereich arbeitet noch die Mehrheit der zur Zeit gehandelten Rechner mit Z80- oder 6502-Prozessoren. Gegenüber ihren Vorgängern bieten die 16-Bit-Maschinen zweifellos mehr Leistung und Geschwindigkeit, aber auch in den 8-Bit-Rechnern steckt noch sehr viel Leben, wenn man das vorhandene, sehr große Softwareangebot berücksichtigt.

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.

Digital Plotter = Digitalplotter
Bei den Zeichengeräten für Heimcomputer handelt es sich fast durchweg um Digitalgeräte. Sie werden über die Druckerschnittstelle betrieben und interpretieren die empfangenen Daten als x,y-Koordinaten von Punkten, die zu Papier gebracht werden sollen. Früher gab es nur Analogplotter, vorwiegend für den Einsatz in der Meßtechnik, etwa um die Temperatur im Innern eines Schmelzofens im genauen Zeitablauf zu zeichnen.

Digitise = Digitalisieren
Digitalisieren ist das Umsetzen einer analogen Größe in einen entsprechenden Digitalwert. Bei der Sprachdigitalisierung wird das Ausgangssignal eines Mikrofons einem Analog/Digital-Wandler zugeführt und digital auf Diskette gespeichert. Dieses Verfahren wird allerdings meist nicht als Digitalisierung, sondern als „Abtastung" (Sampling) bezeichnet, weil das Analogsignal periodisch abgefragt und gewandelt wird. Wie sich mathematisch beweisen läßt, muß zur getreuen Wiedergabe eines Signals die Abtastrate mindestens doppelt so hoch sein wie die höchste im Signal enthaltene Sinusfrequenz. Wenn beispielsweise wie beim Telefon ein Frequenzbereich von 300–3400 Hz erfaßt werden soll, muß die Abtastrate 6,8 kHz betragen.

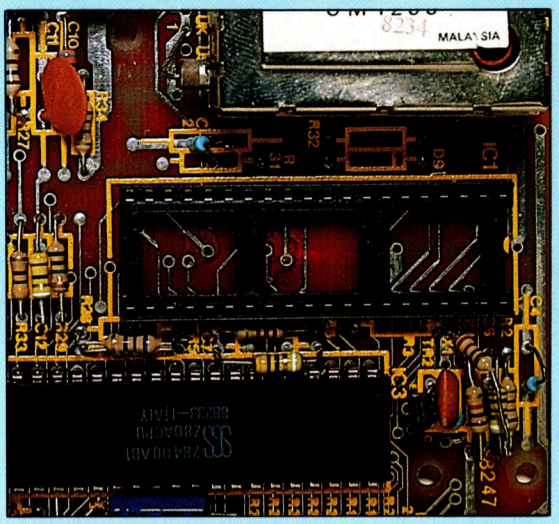
Digitiser = Digitalisiergerät
Bei vielen Rechneranwendungen, insbesondere bei der Konstruktion, Kartographie und Entwurfsgestaltung sind Zeichnungen vom Papier auf den Computer zu übertragen. Dazu brauchen Sie ein Digitalisiergerät mit schnellem Analog/Digital-Wandler. Es genügt aber in vielen Fällen auch ein Digitalisiertablett, bei dem die Zeichnung von Hand mit einem speziellen Stift nachgefahren wird.

Zur Ermittlung der Stiftposition gibt es unterschiedliche Verfahren. Bei einigen Geräten ist in das Tablett ein Drahtgitter eingebettet, über das der Stift „kapazitiv" oder „induktiv" geortet wird. Andere Systeme arbeiten mit Ultraschall, indem sie mit zwei Detektoren unter verschiedenen Winkeln die Entfernung zur Stiftspitze messen.

DIL = DIL
Dual In-Line kennzeichnet die übliche Gehäusebauform für integrierte Schaltungen mit „zweireihiger" Anordnung der Steckerstifte. Die Chips werden heute meist direkt eingelötet. So können die Herstellkosten gesenkt werden. Die Leiterplatten-Unterseite wird dazu in flüssiges Zinn getaucht, nachdem die Pins zuvor mit Bestückungsmaschinen in die entsprechenden Bohrungen der Platine gesteckt wurden.

Früher wurden nicht die Chips selbst eingelötet, sondern in DIL-Sockel gesetzt, die danach festgelötet wurden. Das erhöhte zwar die Fertigungskosten, defekte Chips ließen sich aber schnell ohne zu löten austauschen. Die Bausteine sind jedoch in den letzten zehn Jahren so viel zuverlässiger geworden, daß häufig der gelegentliche Ersatz einer ganzen Karte für den Hersteller billiger ist als die Verwendung von Sockeln und der Austausch einzelner Chips.

Bildnachweise
953, 954, 972, 973: Chris Stevens
958, 968, 969: Kevin Jones
962, 963, 964, 970, 971: Liz Dixon
964, U3: Ian McKinnell
974: Your Spectrum



Das Foto zeigt eine Heimcomputer-Platine mit einem gängigen 40poligen DIL-Sockel oberhalb des Prozessorchips.

Dimension = Dimension
Die Anzahl der Dimensionen eines Systems gibt an, wieviele Parameter zur eindeutigen Definition eines Systemelements erforderlich sind. Allgemein geläufig sind dreidimensionale Systeme, bei denen drei Achsenkoordinaten einen Raumpunkt bestimmen.

In der Datenverarbeitung spielen Dimensionen bei Variablenfeldern eine wichtige Rolle. Das Feld A(5,6) beispielsweise ist zweidimensional, weil zwei Indizes zur Festlegung eines Elements nötig sind.

Direct Access = Direktzugriff
Ein direkter Zugriff auf Daten ist (insbesondere bei Diskettendateien) dann gegeben, wenn jede Informationseinheit unmittelbar angesprochen werden kann. Bei seriellem Zugriff müssen Sie dagegen jedesmal eine Menge nicht benötigter Daten durchgehen. Wenn eine Datei (wie bei einer Datenbank) aus einzelnen Datensätzen aufgebaut ist, muß das Betriebssystem für den Direktzugriff eine ständig aktualisierte Liste führen, in der die Anfangsadressen aller Datensätze der Datei stehen. Zum Abruf einer speziellen Information steuert das DOS anhand dieser Tabelle den Lesekopf unmittelbar auf die entsprechende Spur und sucht den entsprechenden Sektor auf.

computer kurs Heft 36

Perfektes Spielen

Der Sega SC3000H ist zwar in Gestaltung und Funktion nicht gerade revolutionär, hat aber hervorragende Spielqualitäten.

Mensch und Maschine

Die Ergonomie befaßt sich mit der Anpassung von Werkzeug und Umfeld an die Bedürfnisse des arbeitenden Menschen.

Spritesteuerung

Schnelle Spiele sind fast ausschließlich im Maschinencode geschrieben. Unsere Folge erklärt die Spritesteuerung.

Netzspannungs-Relais

In dieser Folge beschreiben wir ein Gerät, das durch ein Niedervolt-Signal Strom an- oder abschaltet.

Hochwertige Grafik

Das Koala-Pad bietet viele Möglichkeiten, hochwertige Grafiken zu erzeugen. Für Apple, Atari und Commodore.