Einsteigen - Verstehen - Beherrschen

DM 3,80 öS 30 sfr 3,80

computer kurs

Heft 36

Sega SC3000H

Kalkulationen

Spritesteuerung

Koala-Grafiktablett

Mensch und Maschine

Spannungsrelais als Schaltuhr

Ein wöchentliches Sammelwerk

computer kurs

Heft 36

Inhalt

Peripherie

Koala-Grafik 981
Leicht, klein und einfach in der Anwendung

Betrieb mit gewissem Komfort 1003
Diskettensysteme für den Acorn B

BASIC 36

Unterwasser-Attacken 984
In die Sprites kommt Bewegung

Tips für die Praxis

Stromquelle 986
Ein Netzspannungsrelais dient als Zeitschaltuhr

Bits und Bytes

Spritesteuerung mit dem Spectrum 989
Maschinencoderoutinen für Actionspiele

Computer Welt

Zu neuen Ufern 992
Software von Psion für Heim- und Personal-Computer

Mensch und Maschine 998
Ergonomie der Rechnerarbeitsplätze

Hardware

Perfektes Spielen 993
Der Sega SC3000H für Einsteiger

Software

Haushaltskasse 996
Kalkulationssysteme im Cassettenformat

Kampf um die Krone 1005
„Lord Of Midnights" von Beyond Software

PASCAL

Eins plus zwei gleich siebzig? 1000
Strukturierte Datenklassen und Datentypen

Computer-Logik

Ist doch logisch! 1006
Zusammenfassung und Wiederholung

Fachwörter von A—Z

WIE SIE JEDE WOCHE IHR HEFT BEKOMMEN

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

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

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

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

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

Abonnement

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

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

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

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

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

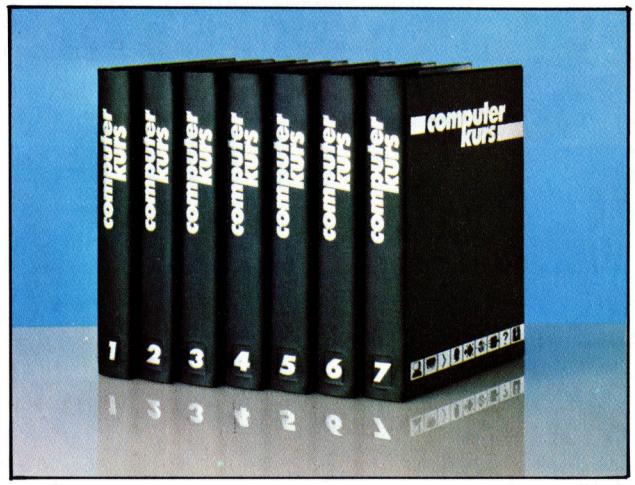
Schweiz: Der Sammelordner kostet sfr 15. Bitte wenden Sie sich an Ihren Kiosk; dort werden Sie jederzeit die gewünschten Exemplare erhalten.

INHALTSVERZEICHNIS

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

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

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



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

Koala-Grafik

Das Koala-Pad bietet eine einfache Möglichkeit, hochwertige Grafiken zu erzeugen. Leicht, klein und einfach in der Anwendung, erlaubt dieses Peripheriegerät den Aufbau ausgefeilter Bildschirmgrafiken auf Fingerdruck.

Das von der amerikanischen Firma Koala Technology entwickelte Koala-Pad gibt es für die Apple-, Atari- und Commodore-Computer. In diesem Artikel gehen wir speziell auf die C 64-Version ein, die Unterschiede zu den anderen Systemen sind nur geringfügig.

Im Gegensatz zu vielen anderen Maltafeln ist das Koala-Pad leicht und klein (20,5 x 16 cm). In der Mitte befindet sich ein 11 mal 11 cm messendes Quadrat aus Kohlefaserstoff, das auf einer berührungsempfindlichen Membran liegt. Durch einfachen Fingerdruck oder Berührung der Membran mit einem Stift kann der Anwender einen Cursor über den Bildschirm steuern oder Optionen wählen.

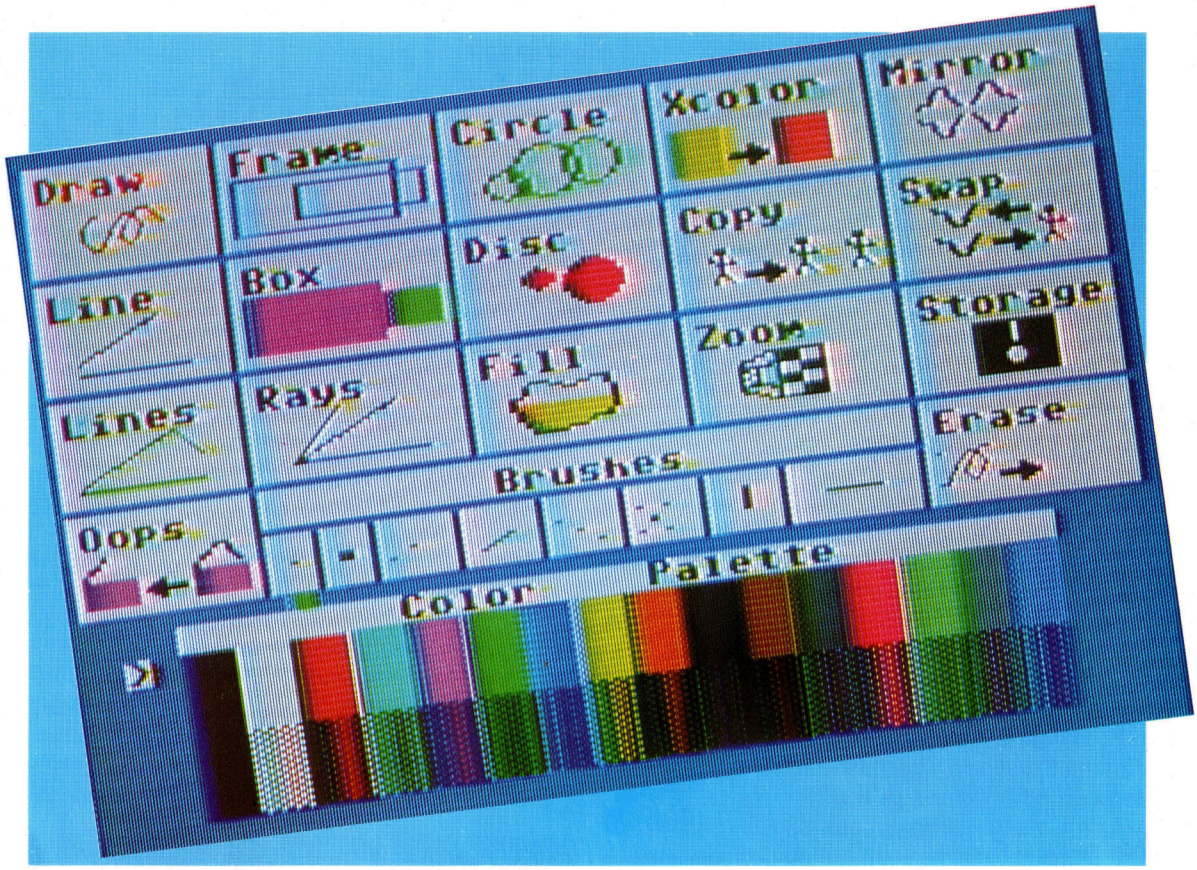
Die Membran besteht aus zwei Lagen leitender Drähte – die eine ist horizontal, die andere vertikal angeordnet. Durch Druck auf die Membran kann das Tablett „feststellen", welche Drähte Kontakt haben, und sie übermittelt die sich daraus ergebenden Koordinaten an den Computer. Oberhalb dieser Zeichenfläche befinden sich zwei Knöpfe, von denen der eine gedrückt werden muß, wenn der Anwender einen bestimmten Bildschirmpunkt kolorieren oder eine Funktion aus dem Programm-Menü wählen will.

Farbenfrohe Bilder

Das Koala-Pad ist über den Joystick-Port mit dem Computer verbunden. Die zur Betreibung des Pad erforderliche Software – der Koala Painter – wird von Diskette geladen. Nach dem Laden werden die verschiedenen Optionen auf dem Bildschirm dargestellt. Am unteren Bildschirmrand steht die „Palette" mit 16 „Echt-" und 16 „Misch-"Farben. Über der Palette befinden sich acht Kästen mit „Pinseln". Das sind verschiedene einfache Konturen, die auf den Schirm geplottet werden können und von einem Einzelpunkt bis zu Kombinationen aus Punkten und Linien reichen. Um die „Pinsel" herum sind verschiedene Optionen zum Zeichnen von Linien und Figuren auf dem Schirm angeordnet. Der Anwender wählt sie durch Druck auf die Membran, worauf ein Cursor-Pfeil auf das entsprechende Feld weist. Durch einen zweiten Druck auf den „Select"-Knopf wird das so angewählte Feld aktiviert.

Das Koala-Pad bietet die Möglichkeiten, einzelne Linien, Strahlen (Linien von einem bestimmten Punkt ausgehend), Rahmen und Kreise zu zeichnen. Bei Verwendung der „Box"- (farbige Rechtecke) oder der „Disc"-Option (farbige Kreise) können Farben „im Stück" eingebracht werden. Weitere Kolorierungsmöglichkeiten erlaubt der Befehl „FILL", mit dem ein eingegrenztes Feld mit einer Farbe nach Wahl versehen werden kann. Mit dem X-COLOUR-Befehl lassen sich die Farben verändern. Identische Formen können simultan unter Verwendung des Befehls „MIRROR" gezeichnet werden. Damit teilt man den Schirm in vier Sektoren, wobei sich der Cursor allerdings nur im oberen linken Viertel bewegen läßt. Alles was innerhalb dieses Quadran-

Umfangreiche und zugleich detaillierte Grafiken lassen sich in wenigen Stunden mit der leicht verständlichen, menügeführten Software erstellen. Anders als ähnliche Grafik-Tabletts kann man das Koala-Pad bei der Arbeit leicht in der Hand halten.



Das Hauptmenü des Koala-Pad besteht aus Rechtecken, in denen sich sowohl Namen als auch Piktogramme befinden. Die Piktogramme sollen zwar dem leichteren Verständnis dienen, sind aber aufgrund der Darstellung teilweise verwirrend. Der Cursorpfeil wird in ein Rechteck gebracht, wonach der „Select"-Knopf zu drücken ist. Die Bezeichnung des gewählten Rechtecks blinkt daraufhin und zeigt dem Anwender, in welchem Mode er arbeitet.

ten gezeichnet wird, erscheint automatisch im entsprechenden Bereich der anderen drei Bildschirmviertel.

Hochauflösende Punktgrafik erreicht man bei Aufruf des ZOOM-Befehls. Hierbei kann der Anwender jeden beliebigen Teil des Bildschirms anwählen, der daraufhin als vergrößertes Fenster am unteren Bildschirmrand erscheint. Die einzelnen Punkte werden im Format acht mal acht Punkt dargestellt. Mit dieser Option lassen sich Schriften und Formen schnell und einfach erzeugen. Mit dem COPY-Befehl kann man die Zeichnungen an jede Stelle des Bildschirms übertragen.

Nach Wahl dieser Optionen verschwindet der Cursor-Pfeil vom Schirm, und der Select-Knopf muß gedrückt werden. Der Schirm wird nun zur „Leinwand", auf der das gewünschte Bild gemalt werden kann.

Kritikpunkte

Die mit diesem Tablett erzeugten Grafiken sind qualitativ hervorragend und können mit kommerziell produzierten, hochauflösenden Bildschirmdarstellungen durchaus konkurrieren. Enttäuschend ist jedoch die Qualität des Befehls DRAW, mit dem freies Zeichnen möglich ist. Die Auflösung der Zeichenfläche entspricht bei weitem nicht der hochauflösenden Bildschirmdarstellung des C 64.

Die Finger- oder Stiftspitze des Anwenders befindet sich häufig nicht über dem exakten Rasterpunkt, sondern berührt zwei Punkte gleichzeitig. Aus der „Fehlinterpretation" ergibt sich ein ziemliches Durcheinander insofern, als daß beispielsweise statt einer Linie Gekrakel zu sehen ist. Weiterer Kritikpunkt: Außer dem ZOOM-Befehl gibt es keine Option zur Farbänderung, ohne ins Hauptmenü zurückzukehren. Doch diese Kritik ist in Relation zu der Geschwindigkeit, mit der die LINE- und FILL-Befehle ausgeführt werden, unerheblich.

Ein weiterer Nachteil liegt bei der Fehlerkorrektur. Hat man etwas falsch gemacht, beseitigt man die Fehlerquelle durch Verwendung des OOPS-Befehls, der über das Hauptmenü erreichbar ist. Aber: Dieser Befehl löscht alles, was man vom Hauptmenü ausgehend gezeichnet hat. Eine Alternative zur Fehlerbeseitigung bietet der ZOOM-Befehl, mit dem punktweise Korrektur möglich ist, was aber sehr zeitintensiv sein kann.

Bildschirminhalte können auf Diskette gespeichert werden und lassen sich leicht in eigene Programme integrieren. Damit ist die Möglichkeit der Programmentwicklung im Stile von „Hobbit" gegeben, bei denen die Grafik dominiert und der Text am unteren Bildschirmrand untergebracht wird. Bis zu 16 verschiedene Darstellungen können mit dem Koala-Pad auf den Schirm geholt werden. Wenngleich es nicht möglich ist, ein Bild direkt von Diskette in den Bildschirmspeicher zu laden, hat Koala Technology doch ein Programmbestandteil integriert, mit dem man einmal geladene Bilder in den Bildschirmspeicher übertragen kann.

Die höchste Auflösung eines gespeicherten Koala-Painter-Bildes beträgt bei Integration in ein BASIC-Programm 255 mal 255 Punkte. Bei Verwendung eines Maschinencodes läßt sich höhere Auflösung erzielen. Diese Beschränkung kann problematisch sein, da beim C 64 die Bildschirmauflösung 320 mal 200 Punkte beträgt. Das Koala-Pad ist deshalb auf die 255-Punkt-Darstellung limitiert, da dies die größte Zahl ist, die mit einem Byte adressiert werden kann. Die Adressierung eines ganzen Bildschirms würde 16-Bit-Adressen erfordern. Nachteil: Man verliert einen Teil der Darstellung. Der Grund liegt darin, daß bei nicht kompletter Bildschirmdarstellung der Inhalt irgendwo im RAM gespeichert ist. Deshalb muß der acht-KByte-Schirm gleichzeitig mit der Farbinformation vom frei verfügbaren Speicherplatz in den hochauflösenden Speicher übertragen werden.

Trotz dieser Einwände läßt sich zusammenfassend feststellen, daß das Koala-Pad eine sehr nützliche Peripherie für jeden ist, der hochauflösende Grafiken auf dem Computer erzeugen will. Schade, daß die Software lediglich auf Diskette zur Verfügung steht. Für Besitzer von Diskettenstationen jedoch, besonders für angehende Künstler oder Abenteuer-Programmautoren, lohnt die Investition allemal.

Koala-Pad

ABMESSUNGEN

210 x 165 mm

DARSTELLUNG

Die volle 320 x 200-Punkt-Darstellung des C 64 wird genutzt, wenngleich diese bei Aufruf durch ein BASIC-Programm auf 255 x 255 reduziert ist.

SCHNITTSTELLEN

Verbindung über den Joystick-Port des C 64.

DOKUMENTATION

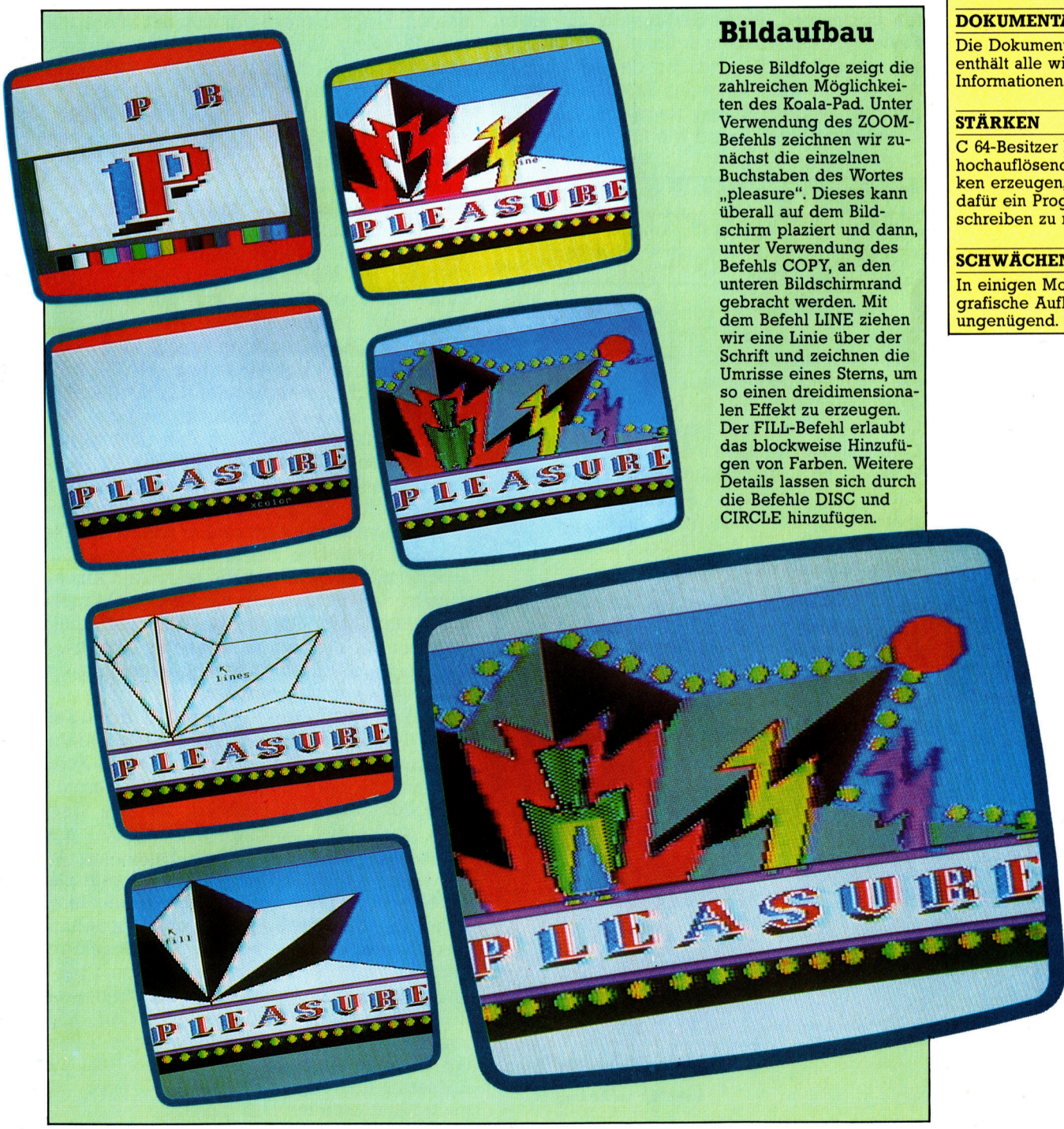
Die Dokumentation enthält alle wichtigen Informationen.

STÄRKEN

C 64-Besitzer können hochauflösende Grafiken erzeugen, ohne dafür ein Programm schreiben zu müssen.

SCHWÄCHEN

In einigen Modi ist die grafische Auflösung ungenügend.



Bildaufbau

Diese Bildfolge zeigt die zahlreichen Möglichkeiten des Koala-Pad. Unter Verwendung des ZOOM-Befehls zeichnen wir zunächst die einzelnen Buchstaben des Wortes „pleasure". Dieses kann überall auf dem Bildschirm plaziert und dann, unter Verwendung des Befehls COPY, an den unteren Bildschirmrand gebracht werden. Mit dem Befehl LINE ziehen wir eine Linie über der Schrift und zeichnen die Umrisse eines Sterns, um so einen dreidimensionalen Effekt zu erzeugen. Der FILL-Befehl erlaubt das blockweise Hinzufügen von Farben. Weitere Details lassen sich durch die Befehle DISC und CIRCLE hinzufügen.

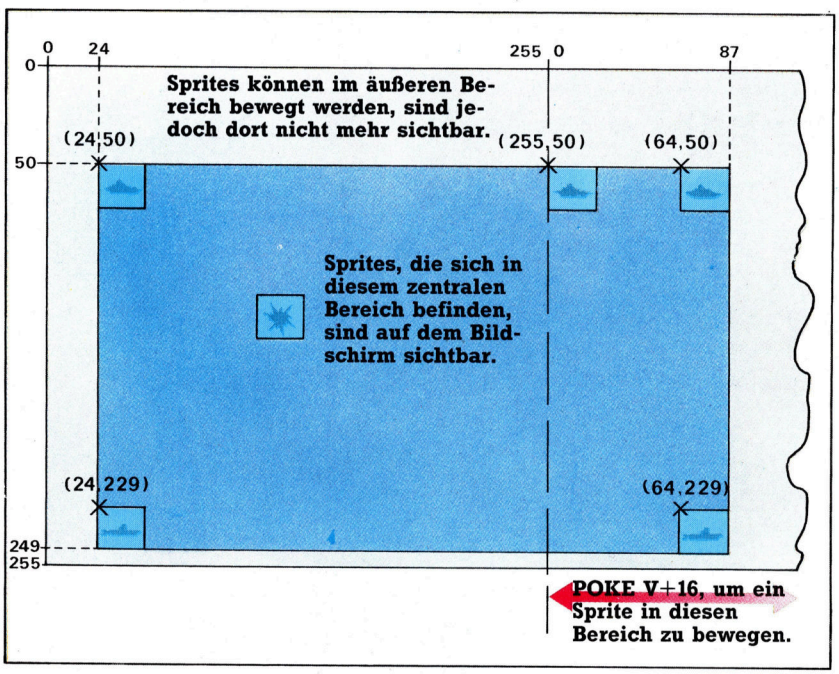
Unterwasser-Attacken

Nachdem bereits das Entwerfen, Einfärben und Vergrößern von Sprites behandelt wurde, wollen wir uns jetzt mit Bewegungs- und Kontrollroutinen beschäftigen.

Um ein Sprite auf dem Bildschirm zu positionieren, muß man die X- und Y-Koordinaten bestimmen. Jedes Sprite wird auf einem Raster von 21x24 Pixeln definiert, und der Ausgangspunkt zur Bestimmung der Koordinaten ist die obere linke Ecke des Rasters. Die Koordinaten werden in den Registern des VIC-Chips abgelegt. Sie sind wie folgt zugeordnet:

Sprite-Nummer	0		1		2		3		7	
Koordinte	X	Y	X	Y	X	Y	X	Y	X	Y
VIC-Register	V	V+1	V+2	V+3	V+4	V+5	V+6	V+7	V+14	V+15

In jedem Speichrplatz können Zahlen von 0 bis 255 abgelegt werden. Dies ist mehr, als zum Ansteuern eines der 200 Pixel in der vertikalen (Y-) Richtung notwendig ist. Da die 320 Pixel in der horizontalen (X-) Richtung den in einem acht-Bit-Register speicherbaren Wert von 255 überschreiten, braucht man ein weiteres Bit für jedes Sprite. Diese zusätzlichen Bits werden in einem Register mit der Adresse V+16 zusammengefaßt. Das Diagramm zeigt die Bildschirmgrenzen für vollständig sichtbare, nicht vergrößerte Sprites.



Dieses Diagramm zeigt die Parameter zur Positionierung der Sprites auf dem Bildschirm. Plaziert man ein Sprite innerhalb des zentralen Bereiches dieses Bildschirms, so ist das Sprite sichtbar. Wird ein Sprite in die äußeren Bereiche bewegt, so ist es nicht sichtbar. Die vier Sprites in den Ecken des zentralen Bereiches zeigen die Grenzen an, innerhalb derer ein Sprite bewegt werden kann und trotzdem noch vollständig zu sehen ist. Auf dem Diagramm sehen Sie ferner die Koordinaten der oberen linken Ecke jedes Sprites.

Die Bewegung des Schiffes wird durch die Programmzeilen 230–250 sowie 270–290 kontrolliert. Das Schiff bewegt sich nur in horizontaler Richtung, so daß die Y-Koordinate unverändert bleibt. Wird sie auf den Wert 80 gesetzt, so wird das Schiff exakt auf dem Meer positioniert. Die X-Koordinate des Schiffes, X0, wird auf den Wert 160 gesetzt, damit es einen zentralen Startpunkt hat.

Das Schiff wird über die Tastatur unter Verwendung der Tasten „Z" und „X" für Links- und Rechtsbewegungen gesteuert. Die Zeilen 230–250 der Hauptschleife rufen die Anweisung GET auf, um die Tastatur abzufragen. Wurde keine Taste gedrückt, so wird die Programmausführung fortgesetzt. Der Wert, den man durch die GET-Anweisung erhält, wird in A$ gespeichert und zum Verändern der X-Koordinate des Schiffes verwendet. Wird beispielsweise die „Z"-Taste gedrückt, so wird das Schiff ein kleines Stück nach links bewegt, und der Wert der X-Koordinate wird verkleinert. In den Zeilen 240 und 250 wird außerdem ein Test durchgeführt, ob das Schiff die Grenzen seines Bewegungsraumes erreicht hat oder nicht. Die IF...THEN-Struktur beim Commodore-BASIC arbeitet in der Form, daß die restlichen Anweisungen der entsprechenden Zeile nicht mehr ausgeführt werden, wenn die erste Bedingung nicht zutrifft. In unserem Programm sind diese Vergleiche so organisiert, daß keine Rechenzeit des Computers verschwendet wird.

Die Bewegung des U-Bootes wird in den Zeilen 300–350 und 2500–2570 gesteuert. Die Unterroutine zum Zurücksetzen der U-Boot-Koordinaten, die bei Zeilennummer 2500 beginnt, verwendet die RND-Funktion, um die Tiefe (Y3) sowie die Geschwindigkeit (DX) des U-Bootes auszuwählen. Y3 wird immer eine Integer-Zahl im Bereich von 110 bis 214 sein. Dadurch wird sichergestellt, daß das U-Boot nicht über der Meeresoberfläche oder unterhalb des Meeresbodens erscheint. Der Geschwindigkeitsfaktor DX liegt im Bereich von 1 bis 3 und bestimmt die Anzahl der Pixel, um die die X-Koordinate des U-Bootes vergrößert oder verkleinert wird. Die X-Koordinate des U-Bootes und das Bit des V+16-Registers, das die X-Koordinaten-Werte von Sprite 3 über 255 kontrolliert, sind auf Null gesetzt.

In den Zeilen 300–350 der Hauptschleife wird der für DX ausgewählte Wert zu dem der X-Koordinate des U-Bootes addiert. Dann wird überprüft, ob das U-Boot den Rand des Bildschirms erreicht hat. Die Zeilen 340 und 350 erlauben die Lösung des Problems, wenn X3 den Wert 255 überschreitet. Nimmt die X-Koordinate beispielsweise den Wert 256 an, so müssen zwei Dinge passieren – das Bit, das Sprite 3 im V+16-Register zugeordnet ist, wird auf eins gesetzt, und das normale X-Koordinaten-Register erhält wieder den Wert Null. Die folgende Tabelle zeigt diesen Vorgang:

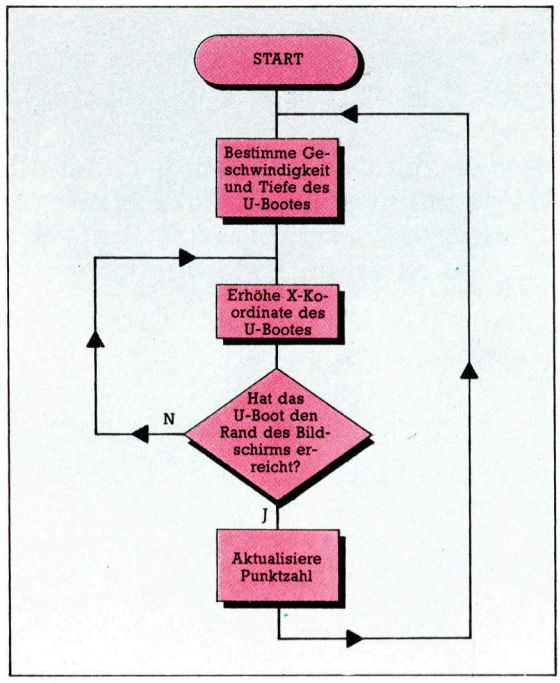
X3	V+16								V+6							
254	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0
255	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
256	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
257	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1

Die Variable H3 wird auf den Wert 1 gesetzt, wenn der Wert von X3 größer als 255 ist. Entsprechend wird L3 auf Null gesetzt, wenn H3 dem Wert 1 entspricht. Anschließend können die Werte von L3 und H3 in die Register V+6 und V+16 gePOKEt werden.

Die Hauptschleife des Programms muß in bezug auf die Wasserbomben zwei Aufgaben erfüllen: Zum einen muß registriert werden, wann die „M"-Taste gedrückt wird, und zum anderen muß, sobald eine Wasserbombe abgeworfen wurde, ihre vertikale Bewegung kontrolliert werden. Außerdem muß das Programm gewährleisten, daß keine weiteren Wasserbomben abgeworfen werden können, solange eine Bombe fällt. Dieses zuletzt genannte Problem kann mit Hilfe eines Flags gelöst werden. In diesem Programm verwenden wir zur Kennzeichnung, daß eine Wasserbombe geworfen wurde, die Variable FL. Ihr Wert ist 1, wenn eine Bombe geworfen wird, und ansonsten 0. In Zeile 100 des Programms wird der Anfangswert von FL auf Null gesetzt. Zeile 260 nimmt Zugriff auf die „SET UP DEPTH CHARGES"-Routine, wenn die „M"-Taste gedrückt und das Flag auf Null gesetzt wird.

Die „SET UP DEPTH CHARGES"-Routine muß drei Aufgaben erfüllen:
1) Setzen des Flags FL auf 1, als Kennzeichen, daß eine Wasserbombe abgeworfen wurde.
2) Setzen der Start-Koordinaten: Die X-Koordinate übernimmt ihren Wert von der des Schiffes, und die Y-Koordinate wird auf den Anfangswert 95 gesetzt, also direkt unterhalb der Meeresoberfläche.
3) Aktivierung des Wasserbomben-Sprites.

Die „MOVE DEPTH CHARGES"-Unterroutine wird verwendet, um die Wasserbombe zu bewegen. Zusätzlich müssen folgende Abfragen durchgeführt werden:
1) Ist die Wasserbombe am U-Boot vorbeigefallen, oder hat sie den Meeresboden erreicht?
2) Hat die Wasserbombe (Sprite 2) das U-Boot getroffen?



Dieses einfache Flußdiagramm zeigt, wie das Programm die Bewegungen des U-Boot-Sprites kontrolliert. Es wird eine zufällige Tiefe und Geschwindigkeit ausgewählt, wobei darauf geachtet wird, daß das U-Boot unterhalb der Meeresoberfläche und oberhalb des Meeresbodens dargestellt wird. Anschließend wird das U-Boot über den Bildschirm bewegt, bis es die andere Seite erreicht.

Tritt der erste Fall ein, kann das Wasserbomben-Sprite abgeschaltet und das Flag auf Null gesetzt werden, so daß eine neue Wasserbombe abgeworfen werden kann. Der zweite Fall kann mit Hilfe einer weiteren Funktion der C64-Sprites-Fähigkeiten ermittelt werden. Es handelt sich dabei um das sogenannte Sprite-Kollisions-Register (V+30). Ist ein bestimmtes Sprite in eine Kollision mit einem anderen Sprite verwickelt, wird das entsprechende Bit in diesem Register auf 1 gesetzt. Kollidieren also nun das U-Boot (Sprite 3) und die Wasserbombe (Sprite 2), ist der Inhalt des Registers V+30 gleich 12 (00001100=12). Indem man mit dem Befehl PEEK den Inhalt dieses Registers überprüft, kann festgestellt werden, ob die Wasserbombe das U-Boot getroffen hat.

Routinen zum Bewegen des U-Bootes

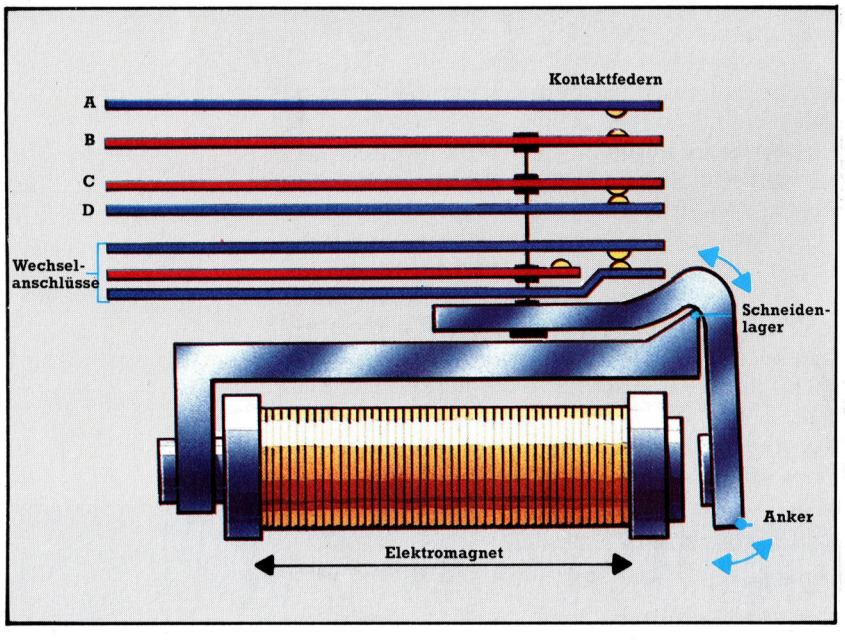
```
130 GOSUB 2500: REM SET SUB CO-ORDS
230 GET A$
240 IF A$="Z" THEN X0=X0-1.5: IF X0<24 THEN
X0=24
250 IF A$="X" THEN X0=X0+1.5: IF X0>245 THEN
X0=245
260 IF A$="M" AND FL=0 THEN GOSUB 3000:
    REM SET DEPTH CHARGES
265 :
270 REM ** MOVE SHIP **
290 POKE V,X0
295 :
300 REM ** MOVE SUB **
310 X3=X3+DX
315 :
320 REM  ** SUB AT SCREEN END? **
330 IF X3>360 THEN DS=-1:GOSUB 5500:
GOSUB 2500
340 H3=INT(X3/256):L3=X3-256*H3
350 POKE V+6,L3: POKEV+16,PEEK(V+16)OR(8*H3)
360 IF FL=1 THEN GOSUB 4000:
 REM MOVE DEPTH CHARGE
370 GOTO 200: REM RESTART MAIN LOOP
380:
390:
2500 REM **** RESET SUB COORDS ****
2510 Y3=110+INT(RND(TI)*105)
2520 X3=0:DX=RND(TI)*3+1
2530 POKE V+7,Y3:POKE V+6,X3
2540 POKE V+16,PEEK(V+16) AND 247
2550 RETURN
2560 :
2570 :
3000 REM ***SET UP DEPTH CHARGES***
3010:
3020 REM ** SET FLAG **
3030 FL=1
3040 :
3050 REM ** SET COORDS ***
3060 Y2=95:X2=X0
3070 POKE V+4,X2:POKE V+5,Y2
3080 :
3090 REM ** TURN ON SPRITE 2 **
3100 POKE V+21,PEEK(V+21)OR4
3110 RETURN
3120 :
3130 :
4000 REM ** MOVE DEPTH CHARGE **
4010 :
4020 REM ** INCREASE Y COORD **
4030 Y2=Y2+2
4040 POKE V+5,Y2
4050 :
4060 REM **TEST BOTTOM & TURN OFF**
4070 IF Y2>Y3+25 OR Y2>216 THEN
POKE V+21,PEEK(V+21) AND 251:FL=0
4080 :
4090 REM **TEST FOR SUB HIT**
4100 IF PEEK(V+30)=12 THEN GOSUB 5000:
REM HIT ROUTINE
4110 RETURN
4120 :
4130 :
```

Stromquelle

Wir setzen den Selbstbau-Kurs mit der Bauanleitung für ein Netzspannungsrelais fort. Damit kann Ihr Computer die Zimmerbeleuchtung zu vorgewählten Zeiten ein- oder ausschalten, aber auch einen Videorecorder sekundengenau steuern.

Elektrische Relais sind Schalter, die durch ein Signal gesteuert werden. In unserem Beispiel sollen große Ströme mit hoher Spannung durch ein Niedervolt-Signal an- und abgeschaltet werden. Relais gibt es in vielen unterschiedlichen Ausführungen, am häufigsten ist aber das Magnetrelais mit Kern, Spule und beweglichem Anker.

Die Relais-Kontakte werden durch Bewegung des Ankers geöffnet und geschlossen. Wenn eine ausreichende Spannung an der um den Weicheisenkern gewickelten Spule liegt, zieht der Spulenkern den Anker an sich heran: Die am Anker befestigten Kontakte bewegen sich dabei nach oben.



Im Bild ist das Relais in Ruheposition, es ist also keine Spannung an der Spule angelegt. Das Kontaktpaar AB ist dann unterbrochen, das Paar CD aber verbunden. In der Arbeitsposition (wenn Spannung an der Spule liegt) findet man umgekehrte Verhältnisse – A und B sind verbunden, C und D jedoch nicht. Durch diesen Aufbau hat man die Wahl, ob das Steuersignal einen oder zwei Stromkreise öffnen oder schließen soll.

Eine weitere Möglichkeit ist die, das Relais als Wechselschalter einzusetzen. Von den drei untersten Kontakten auf dem Bild sind im Ruhezustand nur der obere und untere Kontakt verbunden. In der Arbeitsposition des Relais drückt die mittlere Kontaktfeder ihr oberes Gegenstück hoch und unterbricht dabei die Verbindung der beiden anderen.

Liste der Bauteile

Anzahl	Bauteil
2	Relais, 250 Volt, 10 A, Steuerspannung 8–15 Volt
2	Kunststoff-Gehäuse mit angespritztem Netzstecker und eingebauter Steckdose
2	Feinsicherungshalter für Platinenmontage
2	Feinsicherungen, 6 Ampere
*	4-mm-Stecker (Bananenstecker)
*	2 Meter 2adriges Flachkabel
*	Lochplatinenreste (Veroboard)
*	Kleine Stücke isoliertes Netzkabel

Die mit * bezeichneten Teile sollten noch von Ihrer letzten Bastelarbeit übrig sein. Mit den angegebenen Teilen können Sie zwei Relaismodule herstellen – der Ausgangsbuffer kann bis zu vier Relais gleichzeitig ansteuern. Wenn Sie kein Fertiggehäuse mit angespritztem Stecker und eingebauter Steckdose bekommen können, läßt sich die Schaltung auch in ein normales Gehäuse einbauen.

Alles im Kasten

Achten Sie bei allen Verbindungen auf Sicherheit und guten Kontakt. Prüfen Sie Ihre Platine genau – zwischen Netz- und Signalkabeln darf es keine Verbindung geben!

Die Relais- und die Sicherungsplatine werden mit Epoxidharz-Kleber im Gehäuse befestigt. Einige Allzweck-Kleber leiten die Elektrizität, diese dürfen keinesfalls verwendet werden! Zur Prüfung sollten Sie etwas Klebstoff auf einem Stück Papier trocknen lassen und den Widerstand mit dem Multimeter messen. Verwenden Sie auch beim kleinsten Ausschlag einen anderen Kleber!

Nach dem Trocknen des Klebstoffes kann das Gehäuse geschlossen werden. Die Bananenstecker an die Signalkabel anschließen. Die Polung ist dabei unwichtig, das Relais arbeitet in jeder Stromrichtung gleich. Die Feinsicherung sorgt dafür, daß das Relais nicht versehentlich überlastet wird. Auch wenn es nach den aufgedruckten Daten bis zu 10 A schalten kann, sollten Sie sich aus Sicherheitsgründen auf 5 A beschränken – also auf Geräte unter 1100 Watt.

Achtung!

*** Vor jeder Arbeit muß der Netzstecker aus der Steckdose gezogen werden!**
*** Prüfen Sie vor dem ersten Einschalten alle Verbindungen sehr genau mit dem Multimeter!**
*** Bauen Sie keine provisorische Schaltung auf – NETZSPANNUNG IST LEBENSGEFÄHRLICH!**

Die Platine

Schneiden Sie die Platine so zu, daß keine Leiterbahn gleichzeitig für die Netz- und die Signalanschlüsse des Relais zuständig ist. Dann das Relais auflöten.

PRÜFEN SIE DIE PLATINE JETZT NOCH EINMAL GENAU! Mit dem Multimeter auf Verbindung zwischen den Bahnen testen – ein Fehler dabei kann tödliche Folgen haben!

Mit isoliertem Netzkabel wird die Verbindung vom einen Pol des in das Gehäuse integrierten Steckers zur Relaisplatine und von dort zu einem Anschluß des Steckdosenteils hergestellt. Der andere Pol wird – ebenfalls durch isoliertes Kabel – mit einem Stück Platine verbunden, auf dem die Feinsicherungshalter festgelötet sind. Nur die Schutzleiteranschlüsse von Stecker- und Steckdosenteil werden mit einer grün-gelben Leitung direkt verbunden. Die Signalleitungen mit den Bananensteckern werden an die Wicklungsanschlüsse des Relais gelötet.

Testprogramm

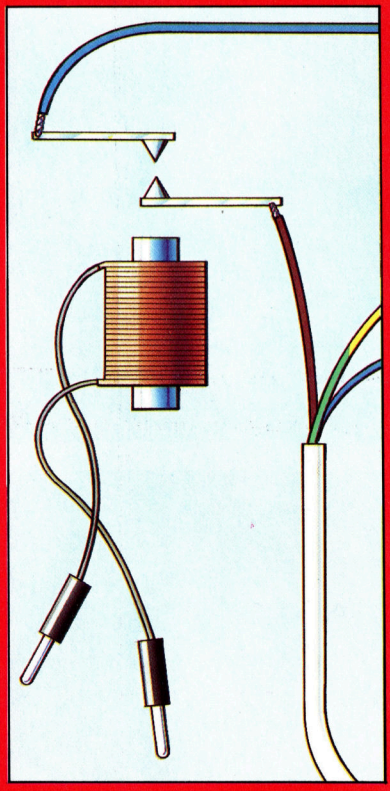
Wenn Ihr Relaismodul fertig und gründlich geprüft ist, können Sie mit diesem Programm testen, ob es die angeschlossenen Geräte richtig ein- und ausschaltet. Nehmen Sie einfach eine kleine Leselampe, und plazieren Sie deren Stecker in die Netzsteckdose des Relaisgehäuses. Die Signalleitungen werden mit dem positiven und negativen Anschluß (Kanal 0) des Niedervolt-Ausgangs verbunden. Jetzt kommt noch der Netzanschluß in die Steckdose.

Ist alles richtig angeschlossen, geben Sie dieses kurze Programm ein – es schaltet die Lampe für fünf Sekunden ein und danach sofort wieder aus.

```
10 REM NETZRELAIS-TEST ACORN B
20 DDR=&FE62:DATREG=&FE60
30 ? DDR=255:REM NUR AUSGABE
40 ?DATREG=1:REM LICHT EIN
50 TIME=0:REM STARTET TIMER
60 REPEAT
70 UNTIL TIME>500
80 ?DATREG=0:REM LICHT AUS
```

```
10 REM NETZRELAIS-TEST C 64
20 DDR=56579:DATREG=56577
30 POKE DDR,255:REM NUR AUSGABE
40 POKE DATREG,1:REM LICHT AN
50 T=TI:REM STARTET TIMER
60 IF (TI—T)<300 THEN 60
70 POKE DATREG,0:REM LICHT AUS
```

Sollte die Lampe beim Programmablauf nicht leuchten, müssen Sie die Schaltung überprüfen. Ziehen Sie zuvor sämtliche Stecker!



Funktionsprinzip

Das Netzrelaismodul schützt den Computer – und auch Sie selbst – vor der hohen Netzspannung. Dazu wird der Schaltvorgang über eine ungefährliche Signalspannung ausgelöst. Computer und Netzspannung sind nur über ein Magnetfeld, nicht aber elektrisch miteinander verbunden.
Damit das Relais nicht überlastet wird, ist in der Schaltung eine 6-Ampere-Feinsicherung vorgesehen, die in einem Sicherungshalter auf einer kleinen Extraplatine im Gehäuse sitzt. Wie die Platinen genau aussehen müssen, hängt davon ab, wo beim verwendeten Relais die Signalanschlüsse beziehungsweise die (geschalteten) Netzanschlüsse liegen.
Falls Sie bisher noch nie mit Netzspannungsgeräten gearbeitet haben, sollten Sie die Schaltung nicht allein aufbauen – das könnte gefährlich werden. Wenn Sie das niedervoltgesteuerte Netzrelais nicht fertig kaufen wollen, sollten Sie Ihre Schaltung vor der ersten Inbetriebnahme noch einmal einem Fachmann – etwa einem Elektriker oder Fachhändler – zeigen. Er sieht sofort, ob alles elektrisch sicher aufgebaut ist.

Morse-Code

Morse Code

A	.–
B	–...
C	–.–.
D	–..
E	.
F	..–.
G	––.
H	....
I	..
J	.–––
K	–.–
L	.–..
M	––
N	–.
O	–––
P	.––.
Q	––.–
R	.–.
S	...
T	–
U	..–
V	...–
W	.––
X	–..–
Y	–.––
Z	––..
.	.–.–.–
,	––..––

Morse-Code-Programm

Dieses Listing für den Commodore 64 läßt Ihren Rechner im Morse-Code „sprechen". Sie geben eine Nachricht (message) ein, und der Computer setzt diese in Beep-Töne und Leuchtzeichen um. Auf diese Art können Sie gleich ausprobieren, ob das Netzrelais die Signale richtig an die Lampe weiterleitet.

```
100 GOSUB 2000:          REM  INIT
150 FOR L=0 TO 1 STEP 0
200 PRINT"ENTER YOUR MESSAGE"
220 PRINT"TYPE 'BYE' TO QUIT"
240 INPUT"MESSAGE ";MS$
300 ML=LEN(MS$):M$=""
320 FOR K=1 TO ML
330 C$=MID$(MS$,K,1)
340 IF C$=)"A"ANDC$<="Z"THEN M$=M$+C$
350 IF C$=" " THEN M$=M$+C$
360 NEXT K:IF M$="" THEN NEXT L
400 ML=LEN(M$)
420 FOR K=1 TO ML
440 CH$=MID$(M$,K,1):CH=ASC(CH$)-64
450 IF CH=-32 THEN FORPP=1TO6*DE:NEXTPP
460 IF CH()-32 THEN GOSUB 3000
480 FOR PP=1 TO 3*DE:NEXT PP
500 NEXT K
550 IF M$="BYE" THEN L=1
600 NEXT L
900 END
2000 REM*******INIT*****************
2100 DIM M$(26)
2110 DDR=56579:DATRG=56577:POKE DDR,255
2120 DE=25:RX=2*DE
2130 V=54296:LF=54272:HF=54273:W=54276
2140 A=54277:S=54278
2150 FOR K=LF TO LF+24:POKEK,0:NEXT K
2160 POKEA,24:POKES,129:POKEV,15
2200 DATA ".-","-...","-.-.","-..","."
2220 DATA "..-.","--.","....",".."
2240 DATA ".---","-.-",".-..","--"
2260 DATA "-.","---",".--.","--.-"
2280 DATA ".-.","...","-","..-","...-"
2300 DATA ".--","-..-","-.--","--.."
2400 FOR K=1 TO 26:READ M$(K):NEXT K
2990 RETURN
3000 REM*********FLASH & BEEP********
3050 PRINT CH$,M$(CH)
3100 N=LEN(M$(CH))
3200 FOR C=1 TO N
3220 D=DE-(ASC(MID$(M$(CH),C,1))-46)*RX
3240 POKE DATRG,1          :REM FLASH
3250 POKE LF,172:POKE HF,57:REM BEEP
3260 POKE W,33:FOR PP=1 TO D:NEXT PP
3270 POKE W,32
3280 POKE LF,0:POKE HF,0   :REM UNBEEP
3290 POKE DATRG,0          :REM UNFLASH
3300 FOR PP=1 TO DE:NEXT PP
3320 NEXT C
3490 RETURN
```

Auch dieses Listing dient dazu, geschriebene Wörter in den Morse-Code zu übersetzen. Es ist für den Acorn B. Jeder beliebige Alpha-String wird mit Hilfe von Licht- und Tonsignalen wiedergegeben. Wenn Sie auch Zahlen umsetzen wollen, müßten die entsprechenden Morse-Codes noch als Datas eingegeben werden.

```
 100 PROCinitialise
 120 ALLOVER=FALSE
 150 REPEAT
 200 PRINT"ENTER YOUR MESSAGE"
 220 PRINT"TYPE 'BYE' TO QUIT"
 240 INPUT"MESSAGE",MS$
 300 ML=LEN(MS$):M$=""
 320 FOR K=1 TO ML
 330 C$=MID$(MS$,K,1)
 340 IF C$)="A"ANDC$(="Z"THEN M$=M$+C$
 350 IF C$=" " THEN M$=M$+C$
 360 NEXT K:IF M$="" THEN UNTIL FALSE
 400 ML=LEN(M$)
 420 FOR K=1 TO ML
 440 CH$=MID$(M$,K,1):CH=ASC(CH$)-64
 450 IF CH=-32 THEN PROCdelay(6*DE*IX)
 460 IF CH()-32 THEN PROCbeepflash
 480 PROCdelay(3*DE)
 500 NEXT K
 550 IF M$="BYE" THEN ALLOVER=TRUE
 600 UNTIL ALLOVER
 900 END
2000 REM**********INIT*************
2050 DEFPROCinitialise
2100 DIM M$(26)
2110 DDR=&FE62:DATREG=&FE60:?DDR=255
2120 DE=3:RX=2*DE:IX=30
2200 DATA ".-","-...","-.-.","-..","."
2220 DATA "..-.","--.",".....","..."
2240 DATA ".---","-.-",".-..","--"
2260 DATA "-.","---",".--.","--.-"
2280 DATA ".-.","...","-",".."-","...-"
2300 DATA ".--","-..-","-.--","--.."
2400 FOR K=1 TO 26:READ M$(K):NEXT K
2990 ENDPROC
3000 REM*****FLASHANDBEEP**********
3020 DEFPROCbeepflash
3050 PRINT CH$,M$(CH)
3100 N=LEN(M$(CH))
3200 FOR C=1 TO N
3220 D=DE-(ASC(MID$(M$(CH),C,1))-46)*RX
3240 ?DATREG=1             :REM  FLASH
3260 SOUND 1,-15,200,D     :REM  BEEP
3270 PROCdelay(IX*D)
3290 ?DATREG=0             :REM  UNFLASH
3300 PROCdelay(DE*IX)
3320 NEXT C
3490 ENDPROC
4000 REM*******DELAY***************
4100 DEFPROCdelay(time)
4200 FOR DD=1 TO time:NEXT DD
4490 ENDPROC
```

Spritesteuerung mit dem Spectrum

Detaillierte, schnelle Spiele werden fast ausschließlich in Maschinencode geschrieben. Dies kann für Anfänger mit Schwierigkeiten verbunden sein.

Das Spectrum-BASIC hat einige Nachteile, die besonders beim Einsatz beweglicher Grafikelemente auffallen. Actionspiele werden jedoch erst durch unterschiedlich gestaltete Sprites interessant, die sich zudem fließend bewegen sollten.

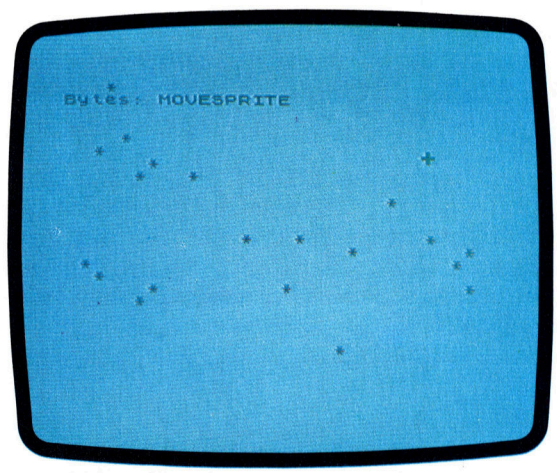
Zwar ist es nicht leicht, in der Assemblersprache Grafikroutinen zu schreiben, doch gibt das hier vorgestellte Programm immerhin einen gewissen Einblick. Das Programm zeigt einen Sternenhintergrund, in dem Sie mit den Cursortasten ein Kreuz bewegen können. Dieses läßt sich in einzelnen Pixelschritten in alle vier Himmelsrichtungen steuern, während der Hintergrund unverändert bleibt. Die Routine läßt sich auch in BASIC-Programme einbauen.

Dazu muß zunächst das BASIC-Programm eingegeben werden. Das Laden des Maschinencodes kann entweder über das BASIC-Ladeprogramm stattfinden oder über die Eingabe des Assembler-Quellentextes mit einem geeigneten Assembler. Beide Programmversionen lassen sich mit den Zeilen 9000 und 9010 auf Cassette speichern.

Funktion der Routine

Um Einblick in die Funktionsweise der Routine zu gewinnen, sehen wir uns zunächst das BASIC-Programm an. Die Unterroutine bei Zeile 1000 liest aus den DATA-Befehlen die Definition des Sprites und POKEt sie ins RAM. Die Zeilen 90 bis 110 erstellen den Hintergrund, und Zeile 120 legt einen Aufgangspunkt des Kreuzes fest. Der Befehl PRINT AT 10,16 veranlaßt den BASIC-Interpreter, die Bildschirmadresse dieser Koordinaten zu berechnen. Die Adresse wird dann in der Systemvariablen DFCC (Speicherstelle 23684 und 23685) abgelegt. Zeile 130 ruft den Initialisierungsteil des Maschinencodes auf. Die Schleife in den Zeilen 140 bis 180 fragt die Tastatur nach Eingaben ab, stellt den eingegebenen Wert über POKE dem Maschinencode zur Verfügung und ruft die Routine auf, die das Sprite um ein Pixel in die gewünschte Richtung bewegt.

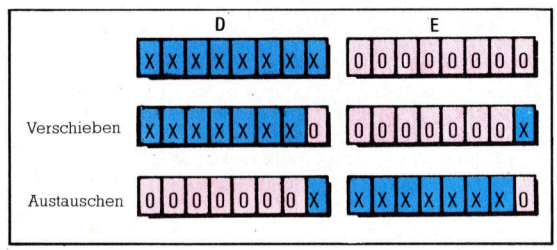
Am Anfang des Assemblerprogramms stehen die Namensdefinitionen der eingesetzten Speicherstellen. KEY enthält den Tastenwert und SPRPOS die Speicheradresse der Bildschirmposition, an der das Sprite dargestellt wird. SPRTAB ist eine Tabelle, in der die Spritedefinition und der Inhalt der Bildschirmposition, die von dem Sprite überschrieben wurde, gespeichert sind. Das Sprite kann nicht nur von einem Zeichenfeld zum anderen springen, sondern auch pixelweise über den gesamten Bildschirm bewegt werden. Die acht Bits jeder Spritezeile werden dafür so auf die beiden Tabellenbytes aufgeteilt, daß sie den zwei Bytes des Bildschirmspeichers und der Bildschirmanzeige entsprechen. Die Speicherstelle BITPOS enthält die Anzahl der Bits, um



Das BASIC-Programm bewegt das Sprite (in den DATA-Befehlen als Kreuz angelegt) mit den Cursortasten über einen Sternenhintergrund.

die sich die Spritedaten vom ersten Bit an verschoben haben.

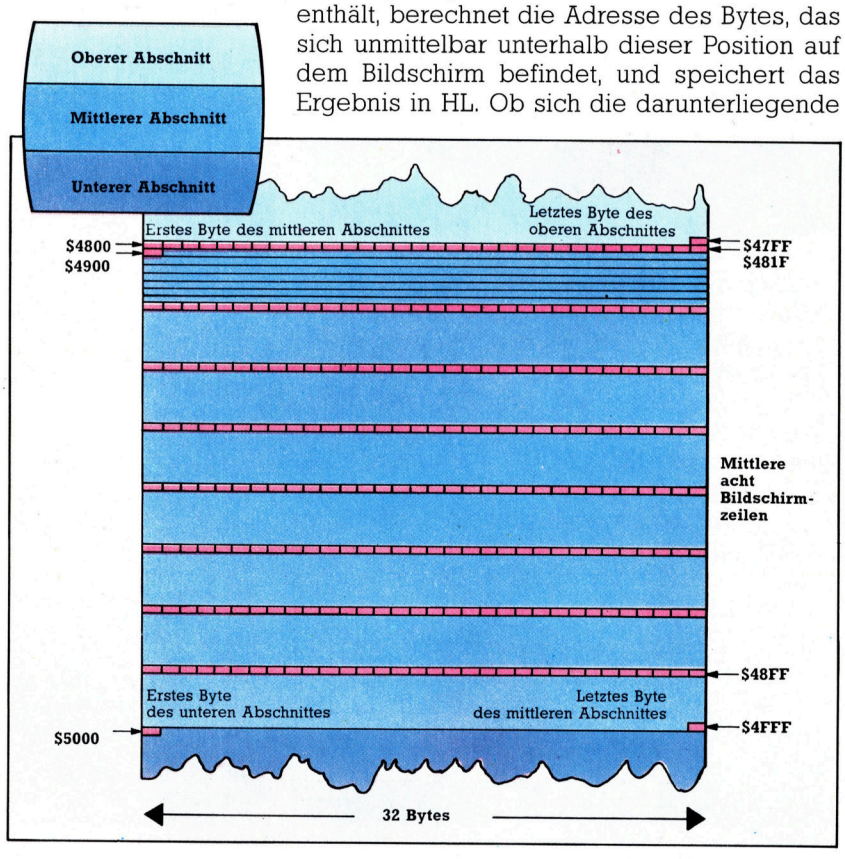
Der Initialisierungsteil des Programms liest die Anfangsadresse des Sprites aus DFCC und springt dann auf SAVSCR, der diese Adresse in SPRPOS speichert, den Wert 1 in das D-Register lädt und die Unterroutine UNDER aufruft. Wenn D auf 1 gesetzt ist, kopiert UNDER den Inhalt des Bildschirmbereichs, auf dem der Sprite erscheinen wird, damit der Hintergrund nach einer weiteren Bewegung des



Bei einer Verschiebung der Sprite-Bits (hier durch X dargestellt) nach links oder rechts kann ein Bit aus einem der beiden Speicherbytes „herausfallen". Tritt dieser Fall ein, dann werden D und E vertauscht und so die Form des Sprites wiederhergestellt.

Die 24 Zeilen des Spectrum-Bildschirms werden für die Memory Map in drei Abschnitte zu je acht Zeilen unterteilt. Die Abbildung zeigt, wie der mittlere Teil aufgebaut ist. Jede Zeile des Bildschirms ist in acht hochauflösende Linien zu 32 Bytes aufgeteilt. Jedes Bit entspricht dabei einem Pixelpunkt. Die Zeilenbytes eines Abschnittes sind sequentiell durchnumeriert, wobei die Numerierung sich in der darunterliegenden Zeile fortsetzt. Die Bytes der acht ersten Linien eines Bildschirmabschnittes haben daher fortlaufende Adressen. Die nächste Adresse ist dann das erste Byte der zweiten Pixellinie der ersten Bildschirmzeile. Das folgende Programm für den 48K-Spectrum verdeutlicht dies, indem es in hoher Auflösung per POKE eine Linie in jedes Bildschirmbyte setzt.

```
50 LET SCRNSTART=
   16384
60 LET SCRNEND=22527
100 FOR B=SCRNSTART
    TO SCRNEND
200 POKE B,255
300 NEXT B
```



KEVIN JONES

Sprites wiederhergestellt werden kann. Das Programm ruft dann die Unterroutine PRSPRT auf, die das Sprite auf den Schirm bringt.

Der Programmteil für die Spritebewegung fängt bei Label MOVSPR an. Zunächst wird 0 in das D-Register geladen und die Unterroutine UNDER aufgerufen. Da D auf 0 gesetzt ist, kopiert UNDER nun den zuvor gespeicherten Bildschirminhalt wieder auf den Schirm zurück und löscht damit das Sprite. Danach liest das Programm die Spriteposition und den Eingabewert, tastet diesen, ruft die Routine auf, die die Bewegung in die entsprechende Richtung vorbereitet, und springt auf SAVSCR, um wie zuvor den Bildschirminhalt zu speichern und das Sprite wieder darzustellen.

Um die Arbeitsweise der beiden Routinen ABOVE und BELOW, die die vertikale Bewegung des Sprites vorbereiten, zu verstehen, muß untersucht werden, wie der Spectrum Speicheradressen und Bildschirmpositionen mineinander verknüpft. An den Hexadezimaladressen läßt sich erkennen, daß das niederwertige Byte jedes der 256 Zeichen eines Bildschirmblocks mit der Zeichennummer dieses Zeichens innerhalb des Blocks übereinstimmt. Das höherwertige Byte hingegen wird mit jeder Pixelzeile zum unteren Bildschirmrand hin um eins erhöht. Die acht Pixelreihen eines Zeichens haben im oberen Bildschirmdrittel daher Adressen von $4000 bis $47FF, im mittleren $4800 bis $4FFF und im unteren $5000 bis $57FF.

Die Unterroutine BELOW setzt voraus, daß das Registerpaar HL eine Bildschirmadresse enthält, berechnet die Adresse des Bytes, das sich unmittelbar unterhalb dieser Position auf dem Bildschirm befindet, und speichert das Ergebnis in HL. Ob sich die darunterliegende Pixelzeile in einem neuen Zeichenblock befindet, können Sie daran erkennen, daß die Bildschirmadresse von H im Binärformat die drei niederwertigen Bits 111 enthält. BELOW überprüft zunächst diesen Fall. Ergibt sich der gleiche Zeichenblock, braucht nur 1 zu H addiert zu werden. Ergibt sich jedoch ein anderer Zeichenblock, dann muß $20 zu L addiert werden. Liegt der neue Wert von L zwischen 0 und $1F (die drei höherwertigen Bits sind 000), dann befinden wir uns in einem anderen Bildschirmabschnitt. Der Wert HL enthält die augenblickliche Bildschirmadresse.

Liegt jedoch der gleiche Bildschirmabschnitt vor, muß 7 von H subtrahiert werden. Der gesamte Vorgang läßt sich leichter verstehen, wenn Sie sich schrittweise ansehen, wie der Code die Adressen verarbeitet. ABOVE funktioniert ähnlich wie BELOW, berechnet jedoch die Adresse des Pixels oberhalb der Bildschirmposition.

Muster verschieben

Die Unterroutinen LMOVE und RMOVE verschieben das Pixelmuster in der Tabelle nach rechts und links. Da sich diese beiden Routinen ebenfalls ähneln, brauchen wir uns nur die Arbeitsweise von LMOVE anzusehen. Dabei wird der Akkumulator mit dem Zeiger der Bitposition (einer Ein-Byte-Zahl zwischen 0 und 7, die den Bitpositionen innerhalb eines Bytes entspricht) geladen. Für die Bewegung wird nun vom Akkumulator 1 subtrahiert. Das Ergebnis ist wiederum eine Zahl zwischen 0 und 7. (War der ursprüngliche Wert die Zahl 0, dann steht im Akkumulator 255.) Eine Schleife führt diesen Vorgang für alle acht Pixelzeilen des Sprites aus. Für jede Zeile werden nun die zwei entsprechenden Bytes dieser Pixelzeile in das Registerpaar DE geladen und eine 16-Bit-Linksverschiebung durchgeführt. Wenn hierbei kein Bit des Sprites aus D heraus- und in E wieder hineingeschoben wird, kann das verschobene Spritemuster wieder in die Tabelle zurückgespeichert und die nächste Pixelzeile geladen werden. Hat sich jedoch ein Spritebit von D nach E verschoben, dann werden D und E vor dem Rückspeichern vertauscht und von HL 1 subtrahiert, damit das Sprite weiter links dargestellt wird.

PRSPRT, die letzte Unterroutine des Programms, bringt das Sprite auf den Bildschirm. Das Modul besteht aus zwei verschachtelten Schleifen. Eine – gesteuert vom C-Register – für die acht Pixelzeilen des Sprites und die andere – gesteuert vom B-Register – für die zwei Bytes, die in die Pixelzeile eingesetzt werden. Der wichtigste Teil dieser Routine bringt das Bitmuster des Sprites auf den Bildschirm, ohne die Bits zu verändern, die nicht von dem Sprite „überdeckt" werden. Die Bildschirmadresse ist im Registerpaar HL gespeichert, und die Adresse der Spritetabelle in IX.

Spritebewegung

```
   5 REM   *Program 1*
  10 CLEAR 45055: REM AFFF HEX
  20 LOAD "MOVESPRITE"CODE
  30 LET KEY=45056
  40 LET BITPOS=45059
  50 LET SPRTAB=45060
  60 LET INIT=45312: REM B100 HEX
  70 LET MOVSPR=45317: REM B105 HEX
  80 GO SUB 1000: REM SET UP SPRITE
  90 FOR I=1 TO 20
 100 PRINT AT 21*RND,31*RND;"*";
 110 NEXT I
 120 PRINT AT 10,16;
 130 RANDOMIZE USR INIT
 140 LET X$=INKEY$: IF X$="" THEN  GO TO 140
 150 LET X=VAL X$
 160 POKE KEY,X
 170 RANDOMIZE USR MOVSPR
 180 GO TO 140
1000 POKE BITPOS,0
1010 FOR I=0 TO 7
1020 READ X
1030 POKE SPRTAB+2*I,X
1040 POKE SPRTAB+1+2*I,0
1050 NEXT I
1060 RETURN
2000 DATA BIN 00011000
2010 DATA BIN 00011000
2020 DATA BIN 00011000
2030 DATA BIN 11111111
2040 DATA BIN 11111111
2050 DATA BIN 00011000
2060 DATA BIN 00011000
2070 DATA BIN 00011000
```

```
   5 REM   *Program 2*
  10 LET A=45312
  20 FOR L=1000 TO 1420 STEP 10
  30 LET S=0
  40 FOR A=A TO A+7
  50 READ B
  60 POKE A,B
  70 LET S=S+B
  80 NEXT A
  90 READ C
 100 IF C<>S THEN  PRINT "ERROR IN LINE ";L: STOP
 110 NEXT L
 120 READ B
 130 POKE A,B
 140 READ B
 150 POKE (A+1),B
 200 PRINT "INSERT PROGRAM TAPE"
 250 SAVE "MOVESPRITE"CODE 45312,400
1000 DATA 42,132,92,24,44,22,0,205,561
1010 DATA 61,177,42,1,176,58,0,176,691
1020 DATA 254,5,32,5,205,165,177,24,867
1030 DATA 24,254,6,32,5,205,109,177,812
1040 DATA 24,15,254,7,32,5,205,136,678
1050 DATA 177,24,6,254,8,192,205,228,1094
1060 DATA 177,34,1,176,22,1,205,61,677
1070 DATA 177,205,35,178,201,42,1,176,1015
1080 DATA 221,33,4,176,14,8,229,6,691
1090 DATA 2,203,66,32,6,221,126,16,672
1100 DATA 119,24,4,126,221,119,16,35,664
1110 DATA 205,83,178,221,35,16,234,225,1197
1120 DATA 13,200,221,35,197,213,205,109,1193
1130 DATA 177,209,193,24,217,62,7,164,1053
1140 DATA 254,7,40,2,36,201,17,32,589
1150 DATA 0,25,62,224,165,32,4,205,717
1160 DATA 83,178,201,124,214,7,103,201,1111
1170 DATA 62,7,164,40,2,37,201,17,530
1180 DATA 32,0,167,237,82,62,224,165,969
1190 DATA 254,224,32,4,205,76,178,201,1174
1200 DATA 124,198,7,103,201,221,33,3,890
1210 DATA 176,221,126,0,61,230,7,221,1042
1220 DATA 119,0,79,221,35,6,8,221,689
1230 DATA 94,0,221,86,1,203,3,245,853
1240 DATA 203,11,241,203,18,203,19,62,960
1250 DATA 7,185,32,3,122,83,95,221,748
1260 DATA 115,0,221,114,1,221,35,221,928
1270 DATA 35,16,220,62,7,185,192,43,760
1280 DATA 205,76,178,201,221,33,3,176,1093
1290 DATA 221,126,0,60,230,7,221,119,984
1300 DATA 0,79,221,35,6,8,221,94,664
1310 DATA 0,221,86,1,203,11,245,203,970
1320 DATA 3,241,203,26,203,27,62,0,765
1330 DATA 185,32,3,122,83,95,221,115,856
1340 DATA 0,221,114,1,221,35,221,35,848
1350 DATA 16,220,62,0,185,192,35,205,915
1360 DATA 83,178,201,42,1,176,221,33,935
1370 DATA 4,176,14,8,229,6,2,221,660
1380 DATA 126,0,47,87,126,162,221,182,951
1390 DATA 0,119,35,205,76,178,221,35,869
1400 DATA 16,237,225,13,200,197,205,109,1202
1410 DATA 177,193,24,224,62,63,188,192,1123
1420 DATA 38,87,201,62,88,188,192,38,894
1430 DATA 64,201,265
```

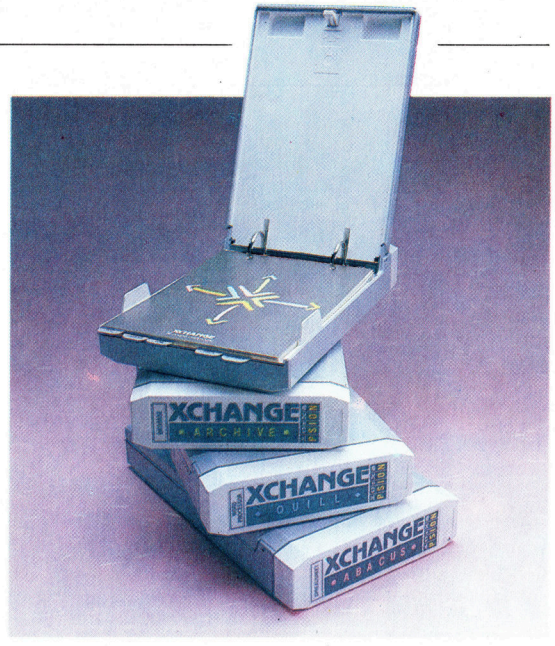
```
KEY     EQU   £B000
SPRPOS  EQU   £B001
BITPOS  EQU   £B003
SPRTAB  EQU   £B004
DFCC    EQU   £5C84
        ORG   £B100
INIT    LD    HL,(DFCC)
        JR    SAVSCR
MOVSPR  LD    D,0
        CALL  UNDER
        LD    HL,(SPRPOS)
        LD    A,(KEY)
        CP    5
        JR    NZ,L0
        CALL  LMOVE
        JR    SAVSCR
L0      CP    6
        JR    NZ,L1
        CALL  BELOW
        JR    SAVSCR
L1      CP    7
        JR    NZ,L2
        CALL  ABOVE
        JR    SAVSCR
L2      CP    8
        RET   NZ
        CALL  RMOVE
SAVSCR  LD    (SPRPOS),HL
        LD    D,1
        CALL  UNDER
        CALL  PRSPRT
        RET
UNDER   LD    HL,(SPRPOS)
        LD    IX,SPRTAB
        LD    C,8
LINE    LD    B,2
BYTE    BIT   0,D
        JR    NZ,SVESCR
WIPOUT  LD    A,(IX+£10)
        LD    (HL),A
        JR    CONT
SVESCR  LD    A,(HL)
        LD    (IX+£10),A
CONT    INC   HL
        INC   IX
        DJNZ  BYTE
        DEC   C
        RET   Z
        INC   IX
        DEC   HL
        DEC   HL
        PUSH  BC
        PUSH  DE
        CALL  BELOW
        POP   DE
        POP   BC
        JR    LINE
BELOW   LD    A,7
        AND   H
        CP    7
        JR    Z,BDIFCB
BSAMCB  INC   H
        RET
BDIFCB  LD    DE,£20
        ADD   HL,DE
        LD    A,£E0
        AND   L
        RET   Z
BSAMSB  LD    A,H
        SUB   7
        LD    H,A
        RET
ABOVE   LD    A,7
        AND   H
        JR    Z,ADIFCB
ASAMCB  DEC   H
        RET
ADIFCB  LD    DE,£20
        AND   A
        SBC   HL,DE
        LD    A,£E0
        AND   L
        CP    £E0
        RET   Z
ASAMSB  LD    A,H
        ADD   A,7
        LD    H,A
        RET
LMOVE   LD    IX,BITPOS
        LD    A,(IX+0)
        DEC   A
        AND   7
        LD    (IX+0),A
        LD    C,A
        INC   IX
RMOVE   LD    IX,BITPOS
        LD    A,(IX+0)
        INC   A
        AND   7
        LD    (IX+0),A
        LD    C,A
        INC   IX
        LD    B,8
RPAIR   LD    E,(IX+0)
        LD    D,(IX+1)
        RRC   E
        PUSH  AF
        RLC   E
        POP   AF
        RR    D
        RR    E
        LD    A,0
        CP    C
        JR    NZ,RSTORE
        LD    A,D
        LD    D,E
        LD    E,A
RSTORE  LD    (IX+0),E
        LD    (IX+1),D
        INC   IX
        INC   IX
        DJNZ  RPAIR
        LD    A,0
        CP    C
        RET   NZ
        INC   HL
        RET
PRSPRT  LD    HL,(SPRPOS)
        LD    IX,SPRTAB
        LD    C,8
PRLINE  LD    B,2
PRBYTE  LD    A,(IX+0)
        CPL
        LD    D,A
        LD    A,(HL)
        AND   D
        OR    (IX+0)
        LD    (HL),A
        INC   HL
        INC   IX
        DJNZ  PRBYTE
        DEC   C
        RET   Z
        DEC   HL
        DEC   HL
        PUSH  BC
        CALL  BELOW
        POP   BC
        JR    PRLINE
        LD    B,8
LPAIR   LD    E,(IX+0)
        LD    D,(IX+1)
        RLC   E
        PUSH  AF
        RRC   E
        POP   AF
        RL    D
        RL    E
        LD    A,7
        CP    C
        JR    NZ,LSTORE
        LD    A,D
        LD    D,E
        LD    E,A
LSTORE  LD    (IX+0),E
        LD    (IX+1),D
        INC   IX
        INC   IX
        DJNZ  LPAIR
        LD    A,7
        CP    C
        RET   NZ
        DEC   HL
        RET
```

Einsatz der Programme

1) Programm 1 eingeben und SAVE "Program 1" LINE 10 aufrufen.
2) Programm 2 eingeben und mit RUN *Program 2* aufrufen. Damit wird der Maschinencode auf Cassette gespeichert – am besten gleich hinter *Program 1*.
3) LOAD "Program 1" läßt das Programm automatisch ablaufen.

Zu neuen Ufern

Das Unternehmen Psion produziert vornehmlich für Sinclair Research. Es hat das Programm „Horizons" für den Spectrum entwickelt und die vier Applikationen, die im Lieferumfang des QL enthalten sind. Unlängst wandte man sich dem Business-Software-Bereich zu und stieg in den Hardware-Bereich ein.

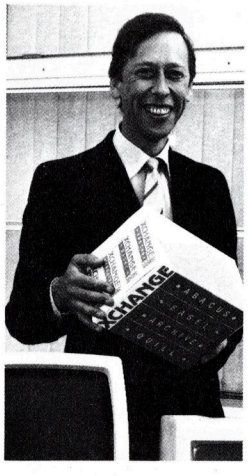


Psions Xchange ist ein Paket integrierter Business-Software, das auf Programmen basiert, die für den Sinclair QL entwickelt wurden. Zu Xchange gehören die Textverarbeitung Quill, die Datenverwaltung Archive, das Finanzplanungsprogramm Abacus und das Grafiksystem Easel. Die vier Programme können sowohl als Paket als auch einzeln erworben werden. Xchange gibt es für den IBM PC und PC XT, ACT Apricot und Apricot XI sowie den Sirius I. Zusätzliche Versionen für den Apple Macintosh und den DEC Rainbow sind in Vorbereitung.

Psion wurde 1981 von Dr. David Potter, einem Dozenten des Imperial College, London, gegründet. Erster Marketing-Coup des Jungunternehmens war die Veröffentlichung von vier Programmen für den ZX 81: Flug-Simulator, Backgammon, Vu-Calc und Vu-File. Charles Davies und Colly Myers zeichneten als Programmautoren verantwortlich. Mit diesem Angebotsspektrum hochwertiger Software wurde der Name des Hauses schlagartig bekannt. Als Sinclair Research 1982 einen Hersteller zur Entwicklung eines Demo-Programmpaketes suchte, das die Stärken des Spectrum aufzeigen sollte, war es klar, daß Psion das betreffende Unternehmen war.

Die Popularität der Sinclair-Rechner bedeutet: Es gibt ein großes Absatzgebiet für Psion-Software. Die Firma hatte eine Reihe bemerkenswerter Erfolge. So wurden über 500 000 Exemplare des „Flugsimulator" für den ZX 81 und den Spectrum verkauft.

Innovativ war Psion schon immer. Das Haus selbst trug entscheidend zur Entwicklung des sogenannten „Kreuz-kompilierens" als Software-Schreibtechnik für den Heimcomputer bei. Das Einführungspaket „Horizons" für den Spectrum wurde auf einem Tandy TRS-80 entwickelt. Heute erfolgt die Herstellung sämtlicher Software auf zwei VAX 750-Minicomputern. Psion hat zudem QLUB organisiert, eine Hotline für QL-Anwender, die bei schriftlichen Reklamationen innerhalb 48 Stunden Antwort und Lösungen garantiert.



Dr. David Potter ist Gründer und Hauptanteilseigner von Psion. Der Ex-Akademiker, der am Imperial College in London auf Computerphysik spezialisiert war, gründete Psion als Potter Scientific Investments.

Der Psion Organiser

Die Pläne für eine Reihe von Geschäftsprogrammen für den IBM PC, Apricot, Sirius, DEC Rainbow und Macintosh-Computer sind weit fortgeschritten. Wie auch beim QL-Paket gehören zur Xchange-Programmpalette Spreadsheet, Datenbank, Grafikprogramm und Textverarbeitung. Der Unterschied zwischen herkömmlichen Programmen besteht nach Auskunft von Psion darin, daß die Daten untereinander ausgetauscht werden können.

Eine weitere, große Aufmerksamkeit erregende Entwicklung von Psion ist der Organiser. Viele Leute waren überrascht, daß ein Haus, das eigentlich als Software-Unternehmen gesehen wurde, einen Taschencomputer entwickelte. Robin Kinnear, ein Sprecher der Firma, versucht, die richtige Perspektive in diese Entwicklung zu bringen: „Hintergrund ist, daß Psion ein Microcomputer-Software-Unternehmen ist. Unter dem Gesichtspunkt Software-Paket hielten wir den ‚Organiser' für eine sehr gute Idee. Doch wir stellten fest, daß es derartiges nicht gab. Also beschloß Psion, eigene Hardware zu entwickeln. Diese Entwicklung war stark durch Software geprägt."

Derzeit sind nur drei Applikationen (Wissenschaft, Mathematik und Finanzen) für den Organiser erhältlich, neben den acht-KByte- und 16-KByte-RAM-Paketen (oder „datapaks", wie sie genannt werden). Psion wird weitere Programme entwickeln und wurde bereits von vielen Interessenten angesprochen, die Software für diesen Rechner schreiben wollen.

Gegenwärtig ist Expansion im Markt eine der Prioritäten des Unternehmens: Jüngst hat man Niederlassungen in den USA und Südafrika etabliert, Verträge zur Verbesserung des Psion-Produktvertriebs in Europa wurden unterzeichnet. Weiterhin hat Sinclair Research mit dem Verkauf von Rechnern nach Osteuropa begonnen, angefangen mit einer Lieferung von 400 ZX 81 in die Tschechoslowakei. Psion, das sich mit der Entwicklung fremdsprachiger Versionen der eigenen Software befaßt, wird diesem Vorbild sicherlich bald folgen und hier ebenfalls einsteigen.

Perfektes Spielen

Die Firma Sega, ein japanisches Unternehmen, hat sich mit Arcademaschinen und Spielen weltweit einen bedeutenden Ruf geschaffen. Aufgrund dieser Erfahrung erwartet man von einem Sega-Computer hervorragende Spielqualitäten.

Der Sega SC3000H wurde auf zahlreichen Computermessen vorgestellt und erntete dabei positive Kommentare. Obwohl er hinsichtlich Gestaltung und Funktionen kaum als revolutionär bezeichnet werden kann, bietet er doch einige Vorzüge wie Erweiterungsfähigkeit und ein großes Angebot an Spiel-Software. Der SC3000H wurde mit dem Z80A-Prozessor ausgestattet.

Der Computer wiegt 1,1 Kilogramm. Das schwarze Gehäuse ist mit weißen alphanumerischen Tasten und grauen Umschalttasten (für Spezialfunktionen, Control, Shift, Return usw.) bestückt. Die schreibmaschinenähnliche Plastiktastatur enttäuscht beim Schreiben. Allerdings ist die Qualität der Tastatur für ein Gerät dieser Preisklasse durchaus akzeptabel. Das „klickende" Gefühl beim Anschlag der Tasten ist möglicherweise ein Überbleibsel der Gummitastatur, mit der die Version für Japan auf den Markt kam. Neben dem bedienungsfreundlichen Cursorsteuerungsblock enthält das Tastenfeld eine nicht programmierbare Funktionstaste für die Eingabe von BASIC-Befehlen, die „Graph"-Taste für den Aufruf von Grafiksymbolen, eine „clear screen"- sowie eine „insert/delete"-Taste. Leider ist die Tastatur des SC3000H, im Gegensatz zu der des SC3000, nicht mit den üblichen Grafiksymbolen ausgestattet.

Der SC3000H wurde mit zahlreichen Schnittstellen versehen. Auf der linken Seite liegen zwei Standard-Joystickanschlüsse, und rechts befindet sich ein ROM-Modulschacht. Auf der Rückseite wurden Fernseher- und Farbmonitoranschluß, Druckerinterface, Cassettenrecorderanschluß sowie eine Buchse für den 9-Volt-Adapter untergebracht. Ferner gehören eine Schaltbox für den TV-Betrieb und eine BASIC-Cartridge mit einem Einführungsheft zum Lieferumfang.

Glücklicherweise ist die Installation des Computers unproblematisch, obwohl die zweiseitige Broschüre lediglich Hinweise für den

Der Sega SC3000H wurde als Konkurrenz zum Sinclair Spectrum und Commodore 64 konzipiert. Für den Computer stehen eine Reihe von Peripheriegeräten wie Cassettenrecorder, Printer/Plotter sowie Joysticks zur Verfügung.

Fernseher- und den Stromanschluß gibt. Eine grüne LED zeigt an, ob der Computer eingeschaltet ist. Allerdings fehlt der Hinweis, daß das Gerät nur funktioniert, wenn sich ein Modul im Steckschacht befindet. Mit eingesetzter BASIC-Cartridge stehen dem Anwender jedoch nur noch 515 Byte RAM zur Verfügung, da das Gerät keinen integrierten BASIC-Interpreter besitzt. Will man also nicht ausschließlich auf fertige Spiele zurückgreifen und selbst programmieren, so ist der Erwerb eines Erweiterungsmoduls notwendig.

Weitere Verwirrung entsteht dadurch, daß in der Broschüre auf die mit Grafiksymbolen versehene Tastatur des SC3000 Bezug genommen wird. So muß der Anwender das beigelegte Keyboard-Diagramm zu Hilfe nehmen, um die mit Symbolen und Befehlen definierten Tasten zu finden. Aufgerufen werden die Befehle RUN, LOAD und GOTO durch Drücken der Funktionstaste zusammen mit der entsprechenden alphanumerischen Taste. Auf diese Weise lassen sich auch mathematische Funktionen wie zum Beispiel ABS, SIN, COS, TAN sowie die String-Befehle LEFT, RIGHT und MID wählen.

Volle Farbenpracht

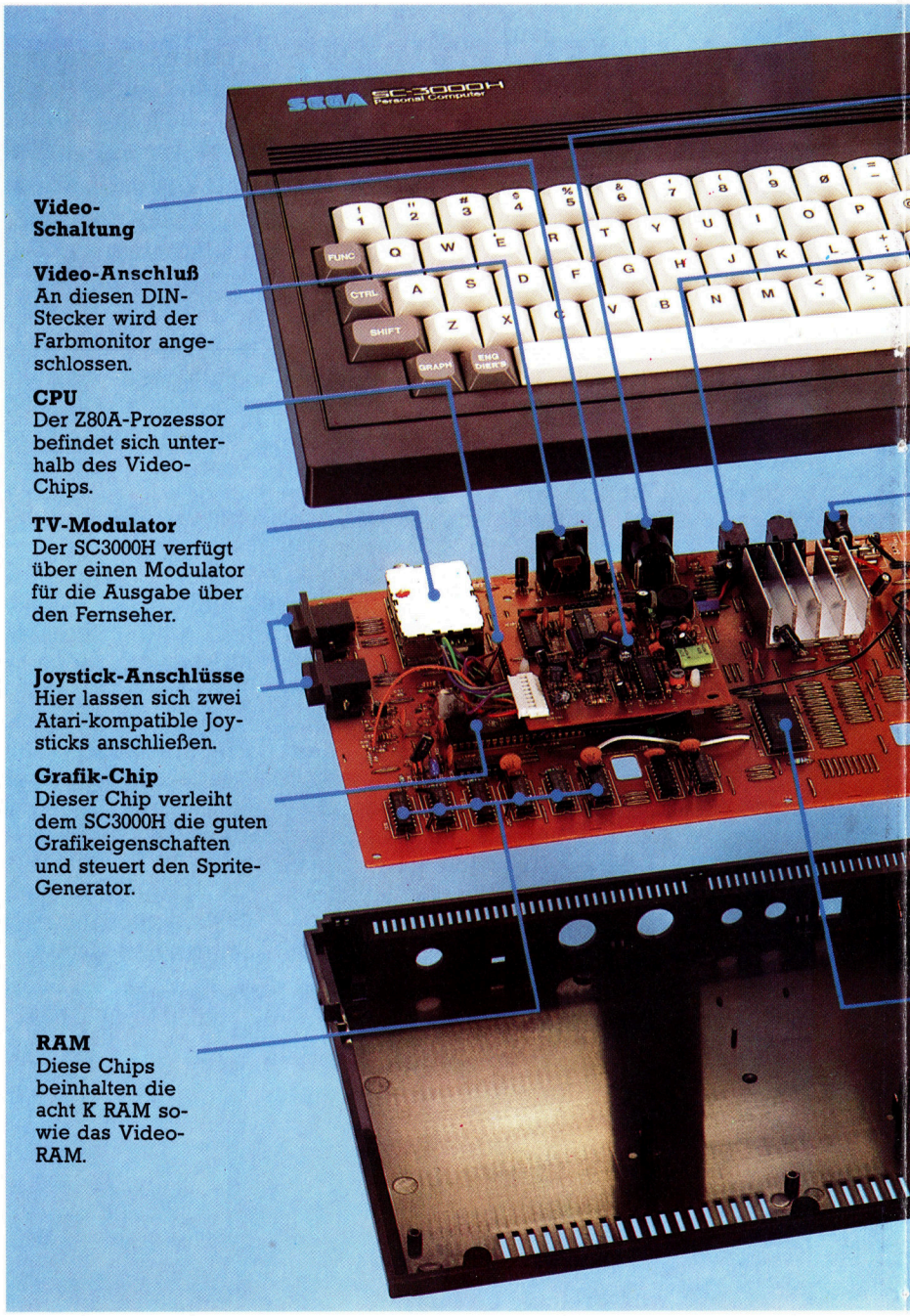
Besondere Beachtung verdienen die Grafikfähigkeiten des SC3000H. Man hat die Möglichkeit, zwischen zwei Modi zu wählen. Der erste ist für Texte im Format 24 Zeilen x 40 Zeichen mit zwei Farben gedacht, während der Grafikschirm Bilder in einer Auflösung von 256x192 Pixeln und 16 Farben darstellt. Zusätzlich können 210 Schattierungen und 32 Sprites erzeugt werden. Das Sega-BASIC, das in vielen Punkten dem Microsoft-Extended-BASIC ähnlich ist, enthält auch Befehle wie DRAW, COLOR, PAINT und SPRITE.

Der SC3000H ist mit sechs Sound-Kanälen ausgerüstet, die sich über POKE oder die BASIC-Anweisungen BEEP und SOUND ansprechen lassen. Die Musik-Cartridge unterstützt zwar das Erzeugen von Melodien, entspricht aber nicht im Entferntesten der im Handbuch angepriesenen „Synthesizer-Qualität".

Die Erfahrungen, die Sega im Bereich der Arcadespiele gesammelt hat, spiegeln sich in der Qualität der Spielsoftware wider. Die Grafik ist durchweg gut, wenngleich die gleichzeitige Darstellung von Text und Grafik anscheinend mit Schwierigkeiten verbunden ist. Der ausgezeichnete Sound unterstreicht die Wirkung der Spiele. Die RESET-Taste hat während des Spielablaufs eine Pausenfunktion.

Die von Sega angebotenen Joysticks erlauben keine schnelle und fließende Steuerung. Man kann jedoch alternativ die Cursorsteuerungstasten zur Spielkontrolle einsetzen.

Zusätzlich bietet das Unternehmen eine Reihe von eigenen Peripheriegeräten an, wie Cassettenrecorder, Farb-Printer/Plotter sowie



Die Sega-Joysticks
Sega bietet zwei verschiedene Joysticks an, die sich für unterschiedliche Spielkategorien eignen. Bei beiden Typen werden acht Positionen gelesen. Sie sind zum Atari-Stecker (neun Pins) kompatibel.

computer kurs

Heft 25–36

Computer Welt

Heft		Seite
25	Laservisionen	673
	Verkabelte Klänge	695
	Tops auf Band	700
26	Sehender Roboter	701
	Synthi-Klänge	721
27	Großer Schlußakkord	729
	Über die Sprache	747
	Spiel um Millionen	755
28	Steuern und Regeln	757
	Das Wissen	779
	Z-Prozessoren	782
29	Schach auf Chips	785
	Perfekte Kopie	802
	Chip-Geplauder	808
30	Kleinroboter	825
	Mit etwas Mut	835
31	Große Roboterarme	841
	Digital Research	859
32	Financial-Micro-Times	869
	Perfekte Zukunft?	894
33	Das gewisse Etwas	897
	Branchenwechsel	923
34	Werden Sie überwacht?	925
	Schreiben für den Schirm	949
	Imagepflege	952
35	Gebrüder Casio	961
	In Schußweite	965
	Natürliche Auslese	978
36	Zu neuen Ufern	992
	Mensch und Maschine	998

Hardware

Heft		Seite
25	Nachbauten	678
26	Jedem seinen Einstein?	713
27	Der Mephisto PHC 64	735
28	Kompakt und kompatibel?	769
29	Klein und leistungsstark	796
30	Im Taschenformat	813
31	Der ACT Apricot	853
32	Gut gestimmt	875
33	In neuem Gewand	902
34	Kompakt verpackt	937
35	Platinencomputer	953
36	Perfektes Spielen	993

Tips für die Praxis

Heft		Seite
25	Transistor-Logik	690
26	Halbaddierer mit Logik-Gattern	706
27	Wiederholung	740
28	Lösungen	765
	EPROM-Brenner	774
29	Heißer Draht	790
30	Buffer für den User Port	819
	Telekommunikation	830
31	Test des Buffers	850
32	Schaltkasten	888
33	Motorsteuerung	915
34	Doppelte Kraft	932
35	Steuerung mit einem Griff	958
36	Stromquelle	986

Software

Heft		Seite
25	Daten-Ablage	676
	Gut zuhören!	684
26	KHLHMPH CHMFLHQ VSUDFKH	704
	In Reih und Glied	716
	Geisterjäger	720
27	Serieller Zugriff	733
	Sommerspiele	754
28	Direkter Zugriff	772
	Raumfestung	778
29	Stille Wasser . . .	793
	Schlüsselposition	800
30	Tal der Trolle	818
	Cassettentricks	828
31	Kalkulationen	844
	Sternensuche	858
32	Vier Minuten Vorwarnzeit	874
	Schnelle Post	884
33	Ein Paket mit Überraschungen	907
	Kleiner Schlucker	914
34	Software-Symphonie	935
	Handel im All	948
35	Zusammenspiel	968
	Psycho-Attacke	974
36	Haushaltskasse	996
	Kampf um die Krone	1005

BASIC

Heft		Seite
25	Das Spectrum des ZX-BASIC	688
26	Funktionen und Kontroll-Strukturen	718
27	Das ABC des Acorn B	745
28	Befehls-Codes	776
29	Trigonometrie	794
30	Grade der Präzision	839
31	U-Boot-Jagd	848
32	Grafik-Entwürfe	886
33	Definitionen	904
34	Völlig aufgelöst	927
35	Rollenverteilung	970
36	Unterwasser-Attacken	984

Peripherie

Heft		Seite
25	Camera Obscura	685
	Bewegter Arm	698
26	Musik aus dem C 64	711
	Prestel-Modem	724
27	Schwarz auf Weiß	742
	Die Movits kommen!	760
	Arbeiten mit der Commodore-Floppy	783
29	Touchmaster	788
30	Sektoren-Grenze	837
31	Rasche Ratte	846
	Drucker-Grafik	864
32	Bildwiedergabe mit Videobuchse	881
33	Rundumblick	909
	Drachenfutter	918
34	Piktogramme	944
35	Die Musikmaschine	972
36	Koala-Grafik	981
	Betrieb mit gewissem Komfort	1003

LOGO

Heft		Seite
25	Neue Strukturen	692
26	Diagramme	726
27	LOGO-Geometrie	738
28	Räumliche Verstellung	766
29	Rastermuster	810
30	Fakultäten	822
31	Mosaikmuster	866

Bits & Bytes

Heft		Seite
25	Register und Speicher	681
26	Assemblerbefehle	708
27	Flagge zeigen	750
28	Arten der Adressierung	762
29	Assemblerschleifen	804
30	Universell einsetzbar	832
31	Last In First Out	861
32	Adreßregister	890
33	Alles Schiebung	920
34	Geteilte Zahlen	940
35	Pixelzeichnungen	975
36	Spritesteuerung mit dem Spectrum	989

PASCAL

Heft		Seite
32	Disziplin ist alles	872
33	Wort für Wort	912
34	Datentypen	946
35	Option und Auswahl	962
36	Eins plus zwei gleich siebzig?	1000

Computer-Logik

Heft		Seite
30	Boolesche Algebra	816
31	Rechner-Addition	856
32	De Morgans Gesetz	878
33	Bedingungsabfragen	900
34	Karnaugh-Tafeln	930
35	Schaltpläne	956
36	Ist doch logisch!	1006

A

Seite

Acorn B: als Terminal 831
AMX-Maus-Paket 944-5
-BASIC 745-6
BBC Buggy 827
Beasty 698-9, 826
Disketten 837
Diskettenlaufwerke 1003-4
EPROM-Brenner 774-5
EVI elektronische Kamera 685-7
Grafik 746
Grenzenverschiebung 681-2
Joystick 959-60
-Kabel 791
Morse-Code-Programm 988
Musiksynthesizer 972-3
ACT Apricot 853-5
ADA 873
Addierer: Halb-, mit Logik-Gattern 706-7
Adresse: Adressierung mit Assembler 762-4
AIM-65 953-5
Akkumulator 683
Alexander, Nick 700
ALGOL 873
ALU (Arithmetisch/Logische Einheit) 683
AMPLE 972-3
AMX-Maus-Paket 944-5
Apple 980
Apple IIc 796-9
Apple III 797
Lisa 797, 969
Aquarius-Drucker 743
Arithmetisch/Logische Einheit (ALU) 683
Assemblersprache: Adressierung 762-4
Befehle 708-10
Grafiken 975-7
mathematische Abläufe 750-3
Module 832-4
-Programm 949
Schleifen 804-7
Unterroutinen 834
Astronomie: „Starfinder" 858
Atari 755-6
Disketten 837
XL-Reihe 898

B

Seite

BASIC: Acorn B- 745-6
Commodore- 776-7, 848-9, 904-6, 927-9
Funktionen mit Sinclair- 718
mathematische Diagramme 886-7
Stringvariablen mit Sinclair- 688-9
Trigonometrie 794-5, 839-40
Basteln; Motorsteuerung 915-17
Netzspannungsrelais 986-7
Roboter 760-1
Rückblick 740-1
Schaltkasten 888-9
selbstgebaute Steuerung 757-9
User-Port-Buffer 819-21
BBC Buggy 827, 911
Beasty 698-9, 826
Betriebssysteme, integrierte 968-9
Bildschirm: Touch-Screen 968
Bildschirmtextsysteme: Prestel 724-5
Binärsystem: Division 940-3
Bitmuster (Bit mapping) 927
Boolesche Algebra 816-17, 1006
Addition 856-7
de Morgans Gesetz 878-80
Karnaugh-Tafeln 930-1, 956-7
Variablen 946-7
Branson, Richard 700
Breadboard (Experimentierplatte) 690
Brother EP-22-Drucker 744
Buffer: User-Port- 819-21, 850-2
Bus 681
Bushnell, Nolan 755

C

Seite

Carrier Tone (Trägerfrequenz) VS 25
Carry (Übertrag) VS 25, 750
Casio 961
Cassettenbuffer 682
Cassettengerät: Dateien mit 828-9
Cassettenrecorder 909
CCD (Charge-Coupled Devices) VS 25
Cell (Tabellenfeld) VS 25
Centronics-Schnittstelle VS 25
Chain (Kette) VS 26
Channel (Kanal) VS 26
Character Generator (Zeichen-generator) VS 26
Check Digit/Check Bit (Prüfziffer/Prüfbit) VS 26
Cheetah: RAT (Infrarot-Joypad) 846, 911
Chip: VLSI- 690, 740
siehe auch Microprozessor
Clock (Taktgeber) VS 27
CMOS VS 27
Coaxial Cable (Koaxialkabel) VS 27
COBOL VS 27, 873
Cold Start (Kaltstart) VS 28
Command Language (Kommando-sprache) VS 28
Commodore-BASIC 776-7
U-Boot-Jagdspiel 848-9, 904-6, 927-9
Commodore Communication System 830
Commodore-Floppy 783-4
Commodore 64: als Terminal 831
Cassettenbuffer 682
Disketten 837
Grafik 927-9, 970-1, 984-5
Joystick 958-9
Koala-Pad 981-3
Morse-Code-Programm 988
Music Maker 711-12
Sprites 970-1
Compaq Plus 937-9
Comparator (Komparator) VS 28
Compiler VS 28, 912
Complement (Komplement) VS 29
Computergesteuertes System 757-9
Concatenate (Verketten) VS 29
Concurrency (Mehrprogrammbetrieb) VS 29
Constant (Konstante) VS 29
Contents adressable (inhalts-adressierbar) VS 29
Control Character (Steuerzeichen) VS 30
Courseware (Lernsoftware) VS 30
CP/M (Control Program for Micro-computers) VS 30
Digital Research 859-60
CPU (Zentraleinheit) VS 30
Ein- und Zwei-Byte-Register 800-3
Innere Organisation 682
Crash (Zusammenbruch) VS 31
Cross-Assembler VS 31
Current Loop (Stromschleife) VS 31
Cursor VS 31
Cyber-310 843

D

Seite

Daisy-chain VS 26
Daisy Wheel (Typenrad) VS 32
Database (Datenbank) VS 32
Data Compression (Datenverdichtung) VS 32
Data Processing (Datenverarbeitung) VS 32
Datei: binäre 676-7
Hashing- 800-1
mit Cassettensystemen 828-9
Random-Access- 772-3, 800-1
sequentielle 716-17, 733-4, 828-9
Datenbank (Database) VS 32
Bild-, mit Laserplatte 673-5
Daten-Richtungs-Register (DDR) 790-2
Datenverarbeitung (Data Processing) VS 33
Datenverdichtung (Data Compression) VS 32
Debugging (Fehlerbeseitigung) VS 33
Decision Table (Entscheidungstabelle) VS 33
Decision Tree (Entscheidungsbaum) VS 33
Declaration Statement (Vereinbarung) VS 33
Decrement (Dekrementieren) VS 34
Degaussing (Entmagnetisierung) VS 34
Delimiter (Trennzeichen) VS 34
de Morgans Gesetz 878-80
Diagnostic Routine (Diagnoseprogramm) VS 34
Digital VS 34
Digital Plotter VS 35
Digital Research 859-60
Digitise (Digitalisieren) VS 35
Digitiser (Digitalisiergerät) VS 35
DIL (Dual In-Line) VS 35
Dimension VS 35
Direct Access (Direktzugriff) VS 35
Disassembler VS 36
Diskette: Formatierung 837-8
Diskettenstation 909
Acorn B 1003-4
Commodore 783-4
Dragon 32/64 918-19
Diskettenverwaltungssystem (DOS) 676
DMA (Direct Memory Access) VS 36
Doppelte Aufzeichnungsdichte (Double Density) VS 36
Doppelte Genauigkeit (Double Precision) VS 36
DOS (Disk Operating System), siehe Diskettenverwaltungssystem
Double Density (Doppelte Aufzeichnungsdichte) VS 36
Double Precision (Doppelte Genauigkeit) VS 36
Download (Laden) VS 36
Dragon: Disketten 837, 918
Diskettensystem 918-19
Drucker 911
anschlagfrei 742-4
elektrostatischer 742-4
Matrix- 864-5
Thermo- 742
Typenrad- VS 32

E

Seite

Edinburgh Turtle 827
Emu Drumulator-Schlagzeugmaschine 729
Entmagnetisierung (Degaussing) VS 34
Entscheidungstabelle (Decision Table) VS 33
Entscheidungsbaum (Decision Tree) VS 33
Entwicklungssysteme 949-50
EPROM-Brenner 774-5
Epson CX-21-Modem 830
FX-80-Drucker 864, 865
P-40-Thermodrucker 911
Ergonomie 998-9
Experimentierplatte (Breadboard) 690

F

Seite

Fairlight CMI (Computer Musical Instrument) 729, 730-1
Fakultäten mit LOGO 822-4
Feedback (Rückkopplung) 758-9
Fehlerbeseitigung (Debugging) VS 33
Feld 996
Tabellen- VS 25
Fernseh/Monitor-Kombination 881-3
Flag 750
FORTH 999
FORTRAN 873

G

Seite

Gatter: logische 691, 706-7, 816-17, 1006
NOR- 741, 765
Geheimcodes 704-5
Genesis P101 842, 843
Geometrie mit LOGO 738-9
Grafik: -hilfsmittel 910

mathematische Diagramme 886-7
mit Acorn B 746
mit Assembler 975-7
mit Commodore 64 904-6, 927-9, 970-1
mit Koala-Pad 981-3
mit Matrix-Drucker 864-5
Grafiktablett 910
„Touchmaster" 788-9

H

Seite

Halbaddierer mit Logik-Gattern 706-7
Hashing 800-1
Haushaltskasse 996-7
Hebot II 826
Hero 842, 843
Hobbit 1003
HRA 933/934 843
Hybrid Technology Music-500 972-3

I

Seite

IC (Integrated Circuit) 690, 740
Inhaltsadressierbar (contents adressable) VS 29
Intel 4004 979
8008 978, 979
8080 978, 979
8086 979, 980
8088 979, 980
Intelligent Software (IS): Entwicklungssystem 950

J

Seite

Jellinghaus Music Systems MIDI-Paket 695
Joystick 911
RAT Infrarot-Joypad 846-7, 911
Steuersystem mit 958-60

K

Seite

Kalkulationssysteme 844-5
Haushalts- 996-7
Kaltstart (Cold Start) VS 28
Kamera: Chip-gesteuerte elektronische 965-7
Kanal (Channel) 685-7
Karnaugh-Tafeln VS 26
Kashio, Tadao 930-1, 956-7, 1006
Kette (Chain) 961
Kildall, Gary VS 26
Koala-Pad 859, 978
Koaxialkabel (Coaxial cable) 981-3
Kommandosprache (Command Language) VS 27
Kommerzielle Programme: VS 28
Haushaltskasse
PIPS 996-7
Serienbriefprogramme 835-6
Tabellenkalkulation 884-5
Verkaufsanalyse 844-5
Komparator (Comparator) 935-6
Komplement (Complement) VS 28
Kondensatoren VS 29
Konstante (Constant) 740
Kontrollstrukturen mit LOGO VS 29
Kurzweil Musiksystem 692-4
Kyocera Portables 731
678-80

L

Seite

Laden (Download) VS 36
Ladungsgekoppelte Bauelemente (CCD) VS 25
Laserplatte 673-5
Laser-Spieler 674-5
Lernsoftware (Courseware) VS 30
Lichtgriffel (Light pen) 910
Logik: Addition 856-7
Bedingungsabfragen 900-1
Boolesche Algebra 816-17, 1006
de Morgans Gesetz 878-80
Karnaugh-Tafeln 930-1, 956-7, 1006
logische Gatter 691, 706-7, 816-17, 1006
-Schaltung mit Transistoren 690-1, 1006
Rückblick 1006-7
LOGO: Diagramme 726-8
Fakultäten 822-4
Geometrie 738-9
Kontrollstrukturen 692-4
Mosaikmuster 866-8
Rastermuster 810-12
räumliche Umwandlung 766-8
Vor- und Nachteile 694
Lotus Symphony 935, 936
1-2-3 935-6

M

Seite

Marketing 897-9
Maschinencode: Assemblerbefehle 708-10
binäre Division 940-3
Routinen für Actionspiele 989-91
Speicherbereiche für 681-3
Mathematische Operationen: Addition 856-7
binäre Division 940-3
Boolesche Algebra 816-17, 1006
Fakultäten mit LOGO 822-4
mathematische Diagramme 886-7
mit Assembler 750-3
Multiplikation 920, 921-2
Subtraktion 920
Trigonometrie 794-5, 839-40
Maus 968
AMX-Paket 944-5
Mehrprogrammbetrieb (Concurrency) VS 29
Melbourne House 952
Memocon 827
Memory Map 940
Mentor 841, 842
Mephisto PHC64 735-7
Microdrive 909
Microprozessor: Entwicklungsgeschichte 978-80
Zilog 782
MIDI (Musical Instrument Digital Interface) 721-3
Hard- und Software 695-7
Milgrom, Alfred 952
Modem 910
Commodore Communication System 830
Epson CX-21 830
Micronet- 724-5
Prism-1000 830
Protek-1200 830
Software für 830-1
Monitor 911
Fernseh/Monitor-Kombination 881-3
Morse-Code-Programm 988
MOS Technology 6502 979, 980
Motorola 6800 978, 979
6809 979, 980
68000 979, 980
Movits 760-1, 827
Music-500 972-3
Musiksynthese: Commodore Music Maker 711-12
Hallgerät 731-2
MIDI 721-3
MIDI-Hard- und Software 695-7
Music-500 972-3
Musiksysteme 697, 729-32
Sampling 730
Yamaha CX5M 875-7
Mustererkennung 925-6
My Talking Computer 808-9

N

Seite

NEC PC-8201A 678-80
Neptune 841, 842, 843
New England Digital Synthesizer 729

O

Seite

Olivetti M10 678-80
Osborne 769
Executive 769
Vadem 769-71

P

Seite

PASCAL 872-3
Datentypen 1000-2
IF- und CASE-Anweisung 962-4
INSTANT PASCAL 954-5
Syntax und Vokabular 912-13
Variablen 946-7
Pascal, Blaise 872
PDSG (Programmable Digital Sound Generator) 696-7
Peripheriegeräte 909-11
bewegter Arm 698-9
Drucker 742-4, 864-5, VS 32, 911
Drucker/Plotter 911
Fernseh/Monitor-Kombination 881-3
Grafikhilfsmittel 910
Grafiktablett 788-9, 910
Joystick 911
Kamera, elektronische 685-7
Koala-Pad 981-3
Maus 944-5
Modem 830-1, 910
Monitor 911
Music Maker 711-12
selbstgebaute 757-9
Speichersysteme 909
Sprach-Synthesizer 910
Piktogramme 944-5
Pixel 702
-zeichnungen 975-6
Platinencomputer 953-5
Plotter 911
digital VS 35
Portables, siehe tragbarer Computer
Potter, David 992
Prestel 724-5
Prism 1000 Modem 830
Programm, integriertes 907-8, 935-6
siehe auch kommerzielle Programme
Programmiersprache: ADA 873
AMPLE 972-3
C-Sprache 860
COBOL VS 27
FORTH 999
FORTRAN 873
INSTANT PASCAL 954-5
MCL (Music Composition Language) 721
Stammbaum 873
siehe auch BASIC; LOGO; PASCAL
Programmzähler 683
Protek 1200 Modem 830
Prüfziffer/Prüfbit (Check Digit/Check Bit) VS 26
Psion 950-1, 992
Vu-Calc 996-7
Xchange 936, 992

Q

Seite

Quantec Hallgerät 732

R

Seite

Rastermuster mit LOGO 810-12
RAT Infrarot-Joypad 846, 911
Register 683
Allzweck- 683
Ein- und Zwei-Byte- 890-3
Index- 683
Status- 683
Relais 758, 986-7
Roboter: Arme 841-3
BBC Buggy 827, 911
Beasty 698-9, 826

Boden- 825-7, 911
Cyber-310 843
Edinburgh Turtle 827
Genesis P101 842, 843
Hebot II 826
Hero 842, 843
HRA 933/934 843
intelligenter 779-81
künftige Entwicklungen 894-6
Memocon 827
Mentor 841, 842
Movits 760-1, 827
Neptune 841, 842, 843
sehender 701-3
Simulationen 802-3
Sprache 747-9
Valiant Turtle 827, 911
Rockwell AIM-65 953-5
Roland GR700 Synthesizer 722
MSQ-700 Sequenzer 729
Rückkopplung (Feedback) 758-9

S — Seite

Sampling 730
Schaltkasten 888-9
Schaltung: Transistor-Logik- 690-1
Schildkröte 825-7
Schnittstelle: Centronics- VS 25
Schrittmotor 759
Sega SC3000H 993-5
Servomotor 759
Sharp PC-1251 813, 815
PC-1500A 813-15
PC-2500 814
Shiina, Takayoshii 835
Sinclair-BASIC: Funktionen und Kontroll-Strukturen 718-19
String-Variablen 688-9
Sinclair-Spectrum: als Terminal 831
Memory Map 940
Micon 695
Micronet-Modem 724-5
RAT Infrarot-Joypad 846-7
Spectrum+ 902-3
Spritegrafik 989-91
Sinclair ZX-Drucker 742, 743
Sinclair ZX-81: Hebot II 826
„Slave" 758
Software: Betriebssystem 968-9
Entwicklungssystem 949-51
integrierte 907-8, 935-6
Tele- VS 36
Softwarehäuser: Digital Research 559-60
Melbourne House 952
Psion 950-1, 992
Virgin Games 700
Sony Microlaufwerk 854
PCM (Pulse Code Modulator) 730
SORD 835-6
PIPS 835-6
Speicher: -bereiche für Maschinensprache 681-3
Stack- 861-3
-systeme 909
Spiele: „Elite" 948
Geschäfts- 869-71
„Ghostbusters" 720
Hör-Spiel 684
„Lord of Midnights" 1005
„Missile Command" 874
„Pacman" 914
„Psytron" 974
„River Rescue" 793
Schach 785-7
„Summer Games" 754
„Twin Kingdom Valley" 818
U-Boot-Jagdspiel für Commodore 848-9, 904-6, 927-9, 970-1, 984-5
„ZAXXON" 778
Spracherkennung 749
Sprachsynthese 748-9, 910
My Talking Computer 808-9
Sprite-Grafiken: Kontrollroutinen 984-5
mit Commodore 64 970-1
mit Maschinencode 989-91
Stack-Speicher 861-3
Stapel (Stack) 861-3
Stapelzeiger (Stack Pointer) 683, 861
Statusregister 750
Steuersystem 911, 915-17, 932-4
mit Joystick 958-60
selbstgebauter 757-9
Steuerzeichen (Control Character) VS 30
Stromschleife (Current Loop) VS 31
Synclavier Synthesizer 731
Synthese: Musik 695-7, 721-3, 729-32, 972-3
Sprach- 748-9, 910
Synthesizer 729-32
-Konzepte 696-7
Music-500 972-3

T — Seite

Tabellenfeld (Cell) VS 25
Tabellenkalkulationen 844-5
Taktgeber (Clock) VS 27
Tandy 923-4
Modell-100 678-80, 924
TP-10-Drucker 744
TRS-80 als Terminal 831
Taschencomputer: Sharp 813-15
Tatung Einstein 713-15
Terminal, Heimcomputer als 831
Textverarbeitung: Serienbriefprogramme 884-5
Touchmaster 788-9
Touch-Screen 968
Tracer, digital 910
Tragbarer Computer (Portables)
Apple IIc 796-9
Compaq Plus 937-9
Sharp-Taschencomputer 813-15
Tandy Modell-100 678-80, 924
Trägerfrequenz (Carrier Tone) VS 25
Transistoren 740
Bipolar- VS 27
Feld-Effekt VS 27
Logik-Schaltungen mit 690-1
Turtle: Edinburgh 827
Hebot II 826
Valiant 827, 911
Typenrad (Daisy Wheel) VS 32

U — Seite

Übertrag (Carry) VS 25, 750
Überwachung 925-6
User Port 758, 790-2
-Buffer 819-21, 850-2

V — Seite

Valiant Turtle 827, 911
Variablen: PASCAL 946-7
String-, mit Sinclair-BASIC 688-9
Vax-750 951
Vereinbarung (Declaration Statement) VS 33
Verkaufsanalyse 935-6
Verketten (Concatenate) VS 29
Vermarktung 897-9
Verschlüsselung 704-5
Verzweigung (Sprung: relativer und absoluter) 807
Videoplatte 673-5
Virgin Games 700

W — Seite

Wahrheitstabellen 816-17
Widerstände 740

X — Seite

XRI-Systems Micon 695

Y — Seite

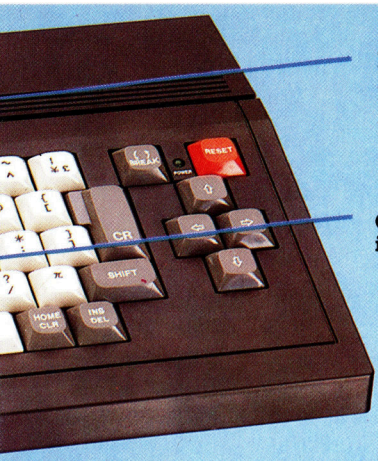
Yamaha Synthesizer CX5M 875-7
DX7 721
KX5-MIDI-Interface 729

Z — Seite

Zeichengenerator (Character Generator) VS 26
Zentraleinheit (CPU) VS 30
innere Organisation 682
Zilog 782
Z80 978-80
Z8000 979, 980
Zusammenbruch (Crash) VS 31

Stichwörter, die auf die vorletzte Heftseite hinweisen, sind als VS und mit der betreffenden Heftnummer gekennzeichnet.

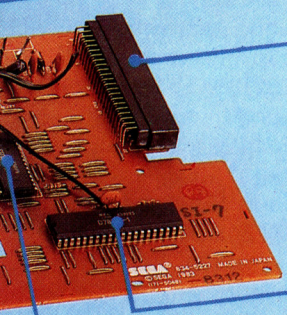
Alle zwölf Hefte erscheint ein solcher Teilindex. Der Gesamtindex erscheint mit dem letzten Heft – darin einbezogen sind Kreuzverweise auf die Artikel, die mit dem gesuchten Stichwort in Verbindung stehen.



Druckeranschluß
Über dieses Interface läßt sich der Farb-Printer/Plotter anschließen.

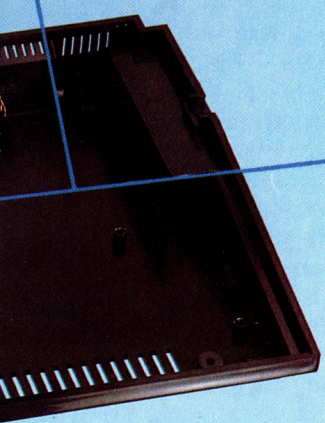
Cassettenrecorderinterface

Haupt-Adapter



Cartridge-Sockel
Hier werden die ROM-Cartridges eingesteckt.

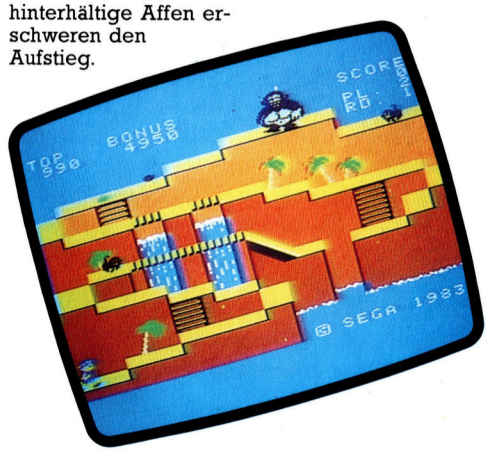
Ein/Ausgabe-Chip
Dieser Chip steuert die meisten Ein- und Ausgaben des Computers.



ROM
Da der SC3000H nicht über ein integriertes BASIC verfügt, wird das ROM hauptsächlich für interne Arbeitsabläufe und die Bildschirmausgabe genutzt.

Safari
Man muß auf gefährliche Tiere mit einem Betäubungsgewehr schießen.

Congo Bongo
Sie sind auf der Jagd nach einem Gorilla. Fallende Kokosnüsse und hinterhältige Affen erschweren den Aufstieg.



Baseball
Ein Kampf um Punkte und Zeit. Die Gegner sind unerbittlich.

Sinbad Mystery
Ein kombiniertes Labyrinth- und Abenteuer-Spiel.

Sega SC3000H

ABMESSUNGEN
353 x 210 x 46 mm

CPU
Z80A, 4 MHz

SPEICHERKAPAZITÄT
8 K RAM, extern erweiterbar auf 48 K; 18 K ROM, extern erweiterbar auf 32 K; 16 K Video-RAM

BILDSCHIRMDARSTELLUNG
24 Reihen mal 40 Spalten im Text-Modus, Grafikauflösung 256 x 192 mit 32 Sprites und 16 Farben in 15 Helligkeitsstufen.

SCHNITTSTELLEN
Cartridge-Steckplatz, zwei Joystick-Anschlüsse, Fernseher- und Monitoranschluß, Lautsprecher-, Cassettenrecorder- und Druckerschnittstelle

SPRACHEN
BASIC, LOGO

TASTATUR
64 schreibmaschinenähnliche Tasten und Cursorsteuerungsfeld, Funktionstaste, Grafik- und Sonderzeichentasten

HANDBÜCHER
Eine zweiseitige Broschüre und ein BASIC-Handbuch, das mit der Cartridge ausgeliefert wird. Es fehlen wichtige Hinweise für die problemlose Inbetriebnahme des Computers.

STÄRKEN
Der Computer bietet gute Grafik- und Soundmöglichkeiten sowie ein komfortables BASIC. Er eignet sich besonders für Spiele und als Gerät für Einsteiger.

SCHWÄCHEN
Die Dokumentation ist unzulänglich. Die geringe Speicherkapazität reicht für selbstgeschriebene Programme nicht aus. Die Tastatur sollte mit den Symbolen und Befehlen versehen sein.

Sega-Software
Bei der Software für den SC3000H spürt man die Erfahrung, die Sega bei seinen Arcadespielen gesammelt hat. Gute Grafik- und Soundeffekte sowie 3D-Grafiken zeichnen diese Programme aus.

die Erweiterungseinheit. Diese stellt 64 KByte RAM zur Verfügung und enthält ferner eine eingebaute Diskettenstation.

Alles in allem bietet der Sega SC3000H einige bemerkenswerte Fähigkeiten zu einem akzeptablen Preis. Wünschenswert wären eine bessere Dokumentation sowie die Beschriftung der definierten Tasten. Der größte Minuspunkt ist jedoch der winzige RAM-Bereich, der die Maschine – ohne zusätzliche Erweiterung – als reinen Spielcomputer abstempelt.

Haushaltskasse

Nicht nur Manager und Buchhalter mit teuren kommerziellen Computern können von einem Kalkulationssystem profitieren. In dieser Serie zeigen wir Ihnen, wie sich Kalkulationssysteme im Cassettenformat auf Heimcomputern einsetzen lassen.

Auf dem Markt gibt es etliche preisgünstige Kalkulationssysteme im Cassettenformat, die für viele Heimcomputer angeboten werden. Diese Artikelserie gibt eine schrittweise Einführung, wie Sie Kalkulationssysteme für alltägliche Aufgaben einsetzen können, darunter die Haushaltskasse, Hypothekenzinsberechnungen und der Vergleich zwischen Bareinkauf oder Leasing bei größeren Anschaffungen.

Das „Herz" jedes Kalkulationssystems ist eine elektronische Tabelle, die – ähnlich wie Rechenpapier – in Zeilen und Spalten unterteilt ist. Der Bildschirm funktioniert dabei wie ein bewegliches Fenster, in dem jeder Teil des gesamten Blattes dargestellt werden kann.

Der Schnittpunkt von Zeile und Spalte wird Feld genannt. Ein Feld kann Zahlen, Text oder Formeln enthalten. Der Tabellencursor ist normalerweise ein invers dargestellter Block, der sich mit den Steuertasten über den Bildschirm bewegen läßt. Dabei sind Tastatureingaben jeweils für das Feld bestimmt, auf dem sich der Cursor gerade befindet.

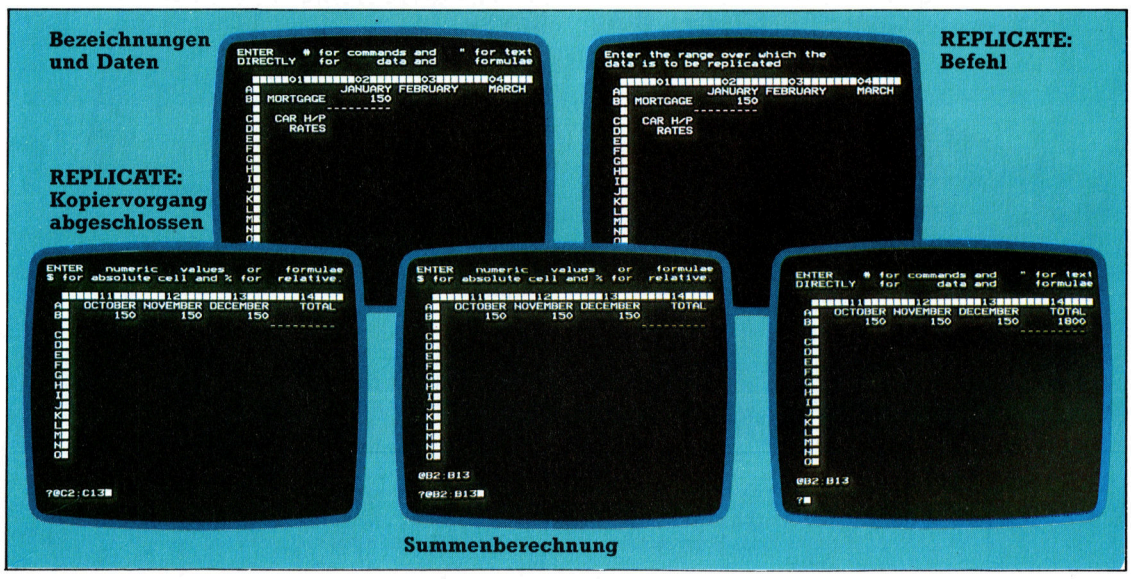
In diesem ersten Artikel werden wir genauer auf das Programmpaket „Vu-Calc" in der Acorn-B-Version eingehen. Es wird von der Firma Psion vertrieben und eignet sich auch für den Spectrum. Die Versionen sind bis auf kleine Unterschiede in den Befehlsnamen identisch.

Vu-Calc ist ein gutes Beispiel dafür, wie flexibel und praktisch ein einfaches Tabellensystem sein kann, obwohl es nicht über die hochentwickelten Fähigkeiten von Paketen wie etwa Lotus 1-2-3 verfügt. Es hat zum Beispiel nicht die Möglichkeit, den Bildschirm vertikal oder horizontal zu teilen, um unterschiedliche Teile der Tabelle gleichzeitig anzuzeigen (eine Fähigkeit, die auf allen „größeren" Kalkulationssystemen vorhanden ist), und es fehlt der Bedienungskomfort. Dennoch kann Vu-Calc einige sehr praktische Modelle aufbauen und diese auf Cassette speichern.

Die Vu-Calc-Version des Acorn B hat maximal 28 Spalten (1 bis 28) und 52 Zeilen (mit alphabetischer Kennung, wobei allen Zeilen nach „Z" doppelte Buchstaben wie „AA", „BB" zugeordnet sind). Dies ist ein vergleichsweise kleines Format, die Grenzen legt jedoch der Arbeitsspeicher des Acorn B (32 KByte) fest.

Nützlich: Haushaltsplan

Eine Jahresübersicht der Haushaltskasse ist einer der nützlichsten Einsatzbereiche dieses Kalkulationssystems. Es läßt sich zudem auch noch leicht aufbauen. Haben Sie das Modell einmal eingerichtet, dann können Sie auf einen Blick sehen, welche Wirkung zusätzliche Ausgaben wie ungewöhnlich hohe Telefonrechnungen oder Reisen auf Ihr Gesamtbudget haben. Auch läßt sich damit leicht verfolgen, welche Bereiche von der Änderung



eines Zahlenwertes angesprochen werden.

Zunächst schreiben Sie alle Haushaltsausgaben auf, die Ihnen einfallen, und notieren die monatlich dafür aufgewendete Summe. Die Ausgabenkategorien werden nun nacheinander in die erste Spalte der Tabelle eingetragen und die darauf folgenden Spalten mit den Bezeichnungen „JAN", „FEB", „MRZ" etc. versehen. Alle Texteinträge müssen in Anführungszeichen eingeschlossen sein, die jedoch nicht in der Tabelle erscheinen. Die ersten Reihen und Zeilen könnten beispielsweise folgendermaßen aussehen:

	1	2	3	4
A		JAN	FEB	MRZ
B	AUTO			
C	HYPOTHEK	600		
D	MIETE			

Dies ist der Grundaufbau des Modells. Die Eingabe der entsprechenden Werte läßt sich mit dem Befehl REPLICATE steuern.

Alle Befehle von Vu-Calc beginnen mit einem Doppelkreuz(#). Wenn Sie daher dieses Zeichen in ein leeres Feld eingeben, schaltet das Programm auf den „Befehlsmodus" und erwartet die Eingabe eines Kommandos. Eines der vielseitigsten Kommandos ist REPLICATE, das dem Anwender die lästige Arbeit abnimmt, alle Werte und Formeln eines Modells einzeln eingeben zu müssen.

Die Feldreihe für feste monatliche Ausgaben (zum Beispiel Miete, Strom, Hypothek) enthält konstante Werte. Wenn die Hypothek beispielsweise 600 Mark beträgt, dann kann REPLICATE diesen Wert in die zwölf zugehörigen Felder eintragen.

REPLICATE wird mit #R aufgerufen. Oberhalb der Tabelle erscheint dann eine Promptzeile mit folgendem Inhalt:

Replicate — Ursprungsfeld eingeben, RETURN spricht das aktuelle Feld an.

Das Programm fragt hier, welches Feld kopiert werden soll (es läßt sich nur jeweils ein Feld und kein Felderblock kopieren — hier ist eine der Einschränkungen von Vu-Calc). Das Feld wird durch seine Koordinaten festgelegt, wobei zuerst der Buchstabe der Zeile und dann die Spaltennummer angegeben wird — etwa C2. Nach dieser Eingabe gibt die Promptzeile folgende Aufforderung:

Bereich angeben, auf den die Daten kopiert werden sollen.

Ein Feldbereich läßt sich durch die Angabe des ersten (oder linksstehenden) und des letzten (oder rechts- bzw. untenstehenden) Feldes definieren. (Stellen Sie sich einen Felderblock als Rechteck vor, dessen linke obere und rechte untere Ecke Sie angeben.)

In unserem Beispiel wollen wir den Wert von 600 Mark in einen Felderblock der Zeile C von C3 bis C13 (die mit „FEB" bis „DEZ" überschriebenen Spalten) kopieren. Die Eingabe sieht folgendermaßen aus: #R, C2, C3:C13. Damit erhält jedes der angegebenen Felder automatisch und fast augenblicklich den Wert 600. (REPLICATE entfaltet seine eigentlichen Fähigkeiten jedoch erst beim Kopieren von Formeln. Dieses Gebiet ist jedoch etwas komplizierter, da Formeln entweder „relativ" oder „absolut" aufgebaut sein können — ein Thema, auf das wir in der nächsten Folge genauer eingehen.)

Die anderen Ausgabenbereiche können auf die gleiche Weise vervielfältigt werden. Wenn der Wert eines bestimmten Monats größer oder kleiner sein soll als der kopierte Standardwert, bewegen Sie den Cursor einfach auf dieses Feld und überschreiben den alten Wert mit einem neuen.

In unserem Modell soll Spalte 14 die Gesamtsummen anzeigen. Da Vu-Calc die Fähigkeit hat, alle Werte einer Zeile oder Spalte zu addieren, brauchen Sie keinen Taschenrechner einzusetzen. Zeilen- oder Spaltenbereiche lassen sich mit dem @-Befehl addieren. Der Bereich wird dabei mit der gleichen Methode angegeben wie bei REPLICATE. Sie brauchen nur den Cursor in C14 zu stellen, @C2:C13 einzugeben, und das Feld C14 zeigt die Gesamtsumme der Hypothekenzahlungen an — in diesem Fall 7200.

Verschiedene Methoden möglich

Statt die Werte auf diese Weise zusammenzuzählen, können Sie in Feld C14 aber auch C2*12 eintragen. Da C kein Anführungszeichen vorangeht, interpretiert Vu-Calc diese Eingabe als Formel und zeigt automatisch das Ergebnis (7200) an. Sie sehen, es gibt unterschiedliche Wege, Ergebnisse zu erzielen.

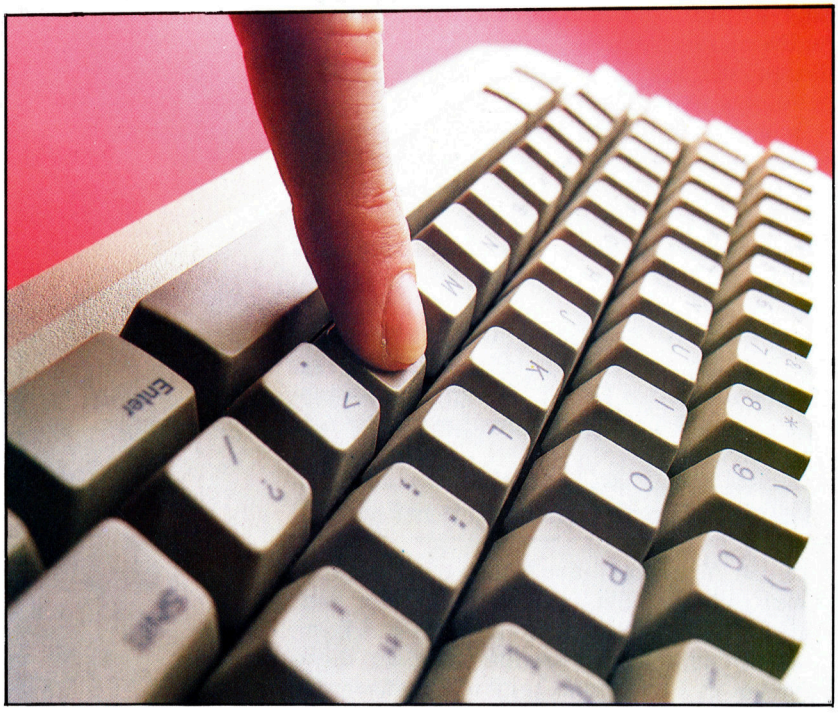
In der nächsten Folge dieser Serie werden wir Ausgaben um feste Prozentsätze „wachsen" lassen und uns ansehen, wie Formeln mit absoluten und relativen Feldbezügen kopiert werden.

Die Programme

Commodore 64
Busicalc: Cassette/Diskette von Supersoft, Winchester House, Canning Road, Harrow HA3 7SJ.
Insta-Calc Graphic: Cartridge/Diskette von Dataview Wordcraft Ltd., Radix House, East Street, Colchester CO1 2XB.
Acorn B
Vu-Calc: Cassette von Psion Ltd., 2 Huntsworth Mews, Gloucester Place, London NW1 6DD.
Spectrum
Vu-Calc: Cassette von Psion Ltd.

Mensch und Maschine

Die Ergonomie befaßt sich mit der Anpassung von Handwerkszeug und Umfeld an die Bedürfnisse des arbeitenden Menschen, unter Berücksichtigung unterschiedlichster Kriterien. Dabei sind physikalisch-technische Aspekte ebenso wie die der Physiologie und Psychologie zu beachten.



Bei der „Schnittstelle Benutzer" denkt jeder zunächst an die Tastatur, die Gegenstand vieler ergonomischer Verbesserungsversuche war und ist – Konturtasten, gewölbtes Tastenfeld, numerischer Tastenblock und integrierte LCD-Anzeige. Das eigentliche Handicap, nämlich die unzweckmäßige QWERTY- bzw. QWERTZ-Anordnung, ist aber anscheinend nicht auszurotten, weil niemand umlernen will. Solche gefühlsmäßigen Abneigungen bilden oft das entscheidende Hindernis beim Versuch, technische Geräte besser an die menschlichen Bedürfnisse anzupassen.

Durch das Vordringen der Computer am Arbeitsplatz und im privaten Bereich rückt die Ergonomie für viele ins Blickfeld. Farbliche und akustische Effekte beispielsweise können die Arbeitsbereitschaft stark beeinflussen: Bei der Telefonvermittlung wird wartenden Anrufern mancherorts Musik zur Besänftigung vorgespielt, und ein grüner Bildschirm wird zumeist als angenehm empfunden, während sich beim Arbeiten mit den alten Schwarzweißschirmen nach ein paar Stunden Streßsymptome einstellen.

Es ist eine alte Erfahrung der Ergonomie, daß Mißvergnügen an der Umgebung (oft ohne bewußtes Erkennen der direkten Ursache) die Anfälligkeit für Seh- und Rückenbeschwerden erhöht und einen erhöhten Ausfall durch Krankheit zur Folge hat. Rechnerbenutzer leiden oft unter den Raumverhältnissen oder einer nicht durchdachten Tätigkeitsgestaltung. Bei intensiver ergonomischer Untersuchung eines Büros mit vielen Bildschirm-Terminals, die einfach längs der Wand aufgestellt worden waren, erwiesen sich Streß- und Isolationsgefühle, die durch diese Anordnung bedingt waren, als Ursache für die niedrige Effizienz und die hohe Fehlerquote. Nachdem die Arbeitnehmer in der Raummitte an gegeneinander gestellten Tischen mit niedrigen Aufbauten untergebracht waren und somit eine freundlichere Atmosphäre geschaffen worden war, wurde die Arbeitsqualität nun sofort bedeutend besser.

Die Einführung von Computern hat häufig zu einer Tätigkeitsabwertung geführt, weil schematische und reizlose Arbeit die Menschen zu wenig forderte. Heute gehört es bei einer Umstellung auf Datenverarbeitung zu den selbstverständlichen Pflichten des Systemanalytikers, eine hinreichend anspruchsvolle und befriedigende Beschäftigung sicherzustellen. Das erfordert sachverständige ergonomische Beratung.

Bei einem guten ergonomischen Konzept wird der arbeitende Mensch als wesentliche Systemkomponente mit spezifischem Verhalten betrachtet. Humanitäre und betriebswirtschaftliche Überlegungen gebieten gleichermaßen, daß das Wohlbefinden der Mitarbeiter im Rang nicht hinter Adreßbuskapazität und Taktfrequenz liegen darf.

Auch der Programmierer kann aus einer sorgfältigen Analyse der besonderen Anforderungen und Probleme bei seiner Arbeit Nutzen ziehen. Ständige Konzentration, logisches Vorgehen und Akribie liegen von Natur aus den wenigsten – die meisten Menschen wehren sich sogar unbewußt gegen solche Auflagen. Das ist eine der Ursachen für die unvermeidlichen Programmierfehler.

Die Gestaltung neuer Programmiersprachen oder Betriebssysteme unter ergonomischen Aspekten ist äußerst reizvoll und bietet dem Anwender viele Vorteile. Psychologen haben den Menschen als „informationsverarbeitendes System" ausgiebig studiert, insbesondere Gedächtnisstruktur, Aufnahmefähigkeit und Zeitverhalten. Diese Erkenntnisse unterstützen die Entwicklung benutzerfreundlicher Systeme, die ohne viel Übung wie selbstverständlich zu handhaben sind.

Bereits seit längerer Zeit werden psychologische Überlegungen in der Industrie bei Trainigs-, Auswahl- und Organisationsmethoden berücksichtigt. Beispielsweise kann man auf Konstrukteure so einwirken, daß sie ihren Entwurf anwendergerecht gestalten, statt vorauszusetzen, der Benutzer werde sich schon an das neue System gewöhnen.

Neuerdings befaßt sich die Ergonomie verstärkt mit der „Schnittstelle Benutzer", das heißt mit der Bedienung komplexer Softwaresysteme. Früher waren die Benutzer fast ausschließlich versierte, stark motivierte Profis, die für eine hohe Rechnerleistung ohne weiteres Unbequemlichkeiten und den Zwang zur Aneignung spezifischer Kenntnisse akzeptierten. Der heutige Benutzer ist dagegen ein Durchschnittsmensch mit begrenzter Nachsicht gegenüber anspruchsvollen und „eigenwilligen" Apparaten. Und nicht nur Gelegenheitsbenutzer haben Probleme mit moderner Technologie: Zahlreiche Datenbanken, die megabyteweise Management-Information zur Verfügung stellen können, werden falsch oder gar nicht genutzt, weil dafür die Beherrschung komplexer Sprachen wie SQL Voraussetzung ist. Entwickler wie Anwender setzen auf die künftigen „Umgangssprachen"-Terminals, die hoffentlich auch eine Kommunikation mit Datenbanken und Finanzprogrammen im Alltagsdeutsch zulassen. Dabei muß das Terminal alle Eingaben in Rechnerbefehle umwandeln und umgekehrt die Antworten des Systems in verständlicher Form ausgeben.

Programme erklären sich

Es gibt viele Wege, dem Benutzer die Arbeit zu erleichtern, etwa mit HELP-Routinen. Forschungsarbeiten über „künstliche Intelligenz" haben zur Entwicklung von erfahrungsorientierten Programmen geführt, die ihre Entscheidungsfindung selbst erklären. Ähnlich könnte man zu intelligenten Beratungs-Modulen für Softwarepakete kommen, allerdings nicht unter Umgehung der Frage, wann und für wen eine Erklärung wichtig ist. Während den Programmierer Datenstrukturen und Verarbeitungsprozesse interessieren, möchten kommerzielle Anwender eigentlich nur wissen, wie man mit der Software etwas geschäftlich Verwertbares zuwege bringt.

Eine Rolle spielen bei der „Schnittstelle Benutzer" sicher auch klischeehafte Vorstellungen. Viele gehen mit Vorurteilen an Computer heran, wobei der Rechner oft für gescheiter gehalten wird als ein Mensch, der die gleiche Arbeit verrichtet. Oder der Rechner gilt wegen der knappen Kommentare vieler Sofwarepakete (speziell bei Eingabefehlern) als unfreundlich, sogar feindselig – eine unsinnige Vermenschlichung des Apparats, die die Arbeit unter Umständen belasten kann.

Unüberlegte Methoden zur Hebung der Benutzerfreundlichkeit, die den Computer intelligenter erscheinen lassen als er ist, machen das nur schlimmer. Launige Dialogformeln wie HALLO FRITZ, ICH BIN DER GUTE GEIST DER DATENBANK ermuntern den Benutzer geradezu, im gleichen Stil zu antworten – im allgemeinen mit niederschmetterndem Ergebnis und entsprechendem Frustrationseffekt.

Echte Benutzerfreundlichkeit erreicht man nur mit viel Geschick: Man muß sich in die Situation des Benutzers einfühlen und versuchen, sein Verhältnis zu seiner Tätigkeit und zum Rechner zu deuten. Ergonomische Gestaltung zeigt sich nicht darin, daß Sie den Bildschirm schräg stellen und die Plattenlaufwerke violett anmalen!

Es ist allgemein bekannt, daß die Leistungsfähigkeit des arbeitenden Menschen durch grobphysiologische Einflußfaktoren wie Tischhöhe, Lufttemperatur oder Geräuschpegel beeinträchtigt werden kann. Aber auch die mittelbaren Effekte von Farbgebung, Lichtwirkung und Raumgefühl gelten inzwischen als gleich wichtig bei der Beurteilung von Arbeitssystemen, insbesondere in der Datenverarbeitung. Neben der Auswahl von Hardware und Software muß der Systemanalytiker, der eine neue EDV-Anlage aufstellt, eine ganze Reihe derartiger Gesichtspunkte berücksichtigen.

Flexibel in der Anwendung

Festgelegte Denkmodelle können genauso frustrierend und beengend wirken wie eine ungemütliche Umgebung. Die Sprache FORTH, die vielerorts als künftiger BASIC-Ersatz angesehen wird, hat der Astronom Charles Moore entwickelt. Er arbeitete am Kitts Peak-Observatorium in Arizona an der Steuerung von Teleskopbewegungen und merkte bald, wie wenig FORTRAN als rein datenverarbeitende Sprache für Steuerungszwecke brauchbar ist. Deshalb schuf er mit FORTH etwas Neues, was sich von starr strukturierten Sprachen wie FORTRAN dadurch unterscheidet, daß für jede spezielle Programmieraufgabe praktisch ein neuer Dialekt entwickelt werden kann. Als Preis für die hohe Flexibilität ist FORTH nicht leicht erlernbar – effiziente Systeme sind nicht immer bequem in der Handhabung.

Eins plus zwei gleich siebzig?

Mit seinen strukturierten Datenklassen und vielen Datentypen eignet sich PASCAL ideal zur Arbeit mit der Logik komplizierter mathematischer Abläufe.

Wieviel ist eins plus zwei? Ohne zusätzliche Information wird die Antwort automatisch „drei" lauten. Wir setzen dabei voraus, daß wir es mit natürlichen Zahlen zu tun haben. Wenn wir jedoch ein Fünfzigpfennigstück und zwei Zehnpfennigstücke in einen Automaten einwerfen, kann das Ergebnis durchaus siebzig Pfennig sein – oder, wenn man es anders betrachtet, eine Tasse Kaffee. Die genaue Angabe der Datenklasse hat folglich große Bedeutung.

In PASCAL können Sie nun eine ganze Reihe von Datentypen selbst definieren und haben damit die Möglichkeit, Ihre Programmalgorithmen exakt auf die eingesetzte Datenstruktur zuzuschneiden. Der PASCAL-Compiler meldet dabei sofort einen Fehler, wenn beispielsweise einer Booleschen Variablen über die Tastatur ein Wert zugeordnet werden soll. Diese Hilfe kann Stunden frustrierender Fehlersuche ersparen, da der Compiler kein Quellenprogramm ausführt, das noch Ablauffehler enthält. Einer der Vorteile von PASCAL ist, daß es zahlreiche unterschiedliche Datentypen verarbeitet und man diese auf verschiedene Arten beschreiben kann.

Die einfachen Variablentypen (die nur einen einzigen Wert enthalten können) arbeiten nach folgenden Regeln: Unterschiedliche Skalartypen sind untereinander nicht kompatibel. Es gibt jedoch zwei numerische Variablen, die viele gemeinsame Eigenschaften besitzen. So können sich reale Zahlen (Real) Ganzzahlen (Integer) annähern und Ganzzahlen in reale Zahlen umgewandelt werden. Reale Zahlen können jedoch nie völlig zu Ganzzahlen werden. Hier einige Beispiele:

```
Program Compatibility (input, output);
VAR
  intA,
  intB :integer;
  Xreal,
  Yreal :real;
BEGIN
  read (intA,Xreal); (*erst Integer, dann einen
    Real eingeben*)
  Yreal := intA; (*real := ist möglich*)
  intB := Xreal; (*FEHLER : Umwandlung
    nicht möglich*)
  (*etc.
```

Arithmetische Vorgänge sind logischerweise nur für numerische Variablen definiert. Die folgenden mathematischen Zeichen können mit numerischen Variablen vom Typ Integer und Real eingesetzt werden:

+ Addition
– Subtraktion
* Multiplikation
/ Divison (Fließkommaformat)

Diese Zeichen werden „dyadische" (zum Zweiersystem gehörende) Operatoren genannt, da sie zwei Operanden (Real oder Integer) benötigen. Ist einer der Operanden ein Real, ergibt sich ein Real: 2 + 2.0 ist 4.0 (nicht 4). Das Ergebnis einer Division ist in jedem Fall ein Real, selbst wenn beide Operatoren Integer sind: 3/4 ist 0.6 und 8/4 ist 2.0.

Bei einer Division zweier Integers läßt sich mit einem Real als Ergebnis oft nichts anfangen. Verteilt man beispielsweise zwölf Stücke Schokolade an zehn Personen, so erhält jeder ein Stück, während zwei Stücke übrig bleiben. Mit den zwei Operatoren DIV und MOD (beides Schlüsselwörter in PASCAL) ist die Ganzzahlenteilung jedoch möglich. Das Ergebnis und der Rest werden dabei im Integerformat angezeigt: 15 DIV 5=3 und 15 MOD 5=0; 31 DIV 7=4 und 31 MOD 7=3. Dabei dürfen keine negativen Werte eingesetzt werden, da die Integerdivision für negative Nenner nicht definiert ist. Bei beiden Divisionsarten – Real und Integer – ist die Teilung durch Null ein Fehler, da noch niemand herausgefunden hat, wie die Unendlichkeit bewertet werden kann.

Multiplikation und Division werden vor Addition und Subtraktion ausgeführt. Mit Klammern läßt sich diese Rangfolge jedoch ändern. (8+4) DIV 2=6 aber 8+4 DIV 2=10.

Obwohl Reals nicht unmittelbar in Integers umgewandelt werden können, führen „Trunc" und „Round" diese Funktion indirekt aus. Trunc schneidet bei einem Real alle Stellen hinter dem Komma ab: trunc (3.999) = 3 und trunc (–123,456) = –123. Die Funktion round rundet eine Zahl gegen Null ab, wenn die Nachkommastellen kleiner sind als 0.5. Liegen sie darüber, wird in die andere Richtung gerundet: round (1234.5) = 1235 und round

(−0.49237) = 0. Ein Fehler tritt auf, wenn das Ergebnis der beiden Funktionen außerhalb des für den Typ Integer verfügbaren Zahlenbereiches liegt (−MaxInt bis MaxInt). Fangen Sie diese Möglichkeit sicherheitshalber vorher mit einer IF-Abfrage ab. Die in Klammern stehende Zahl muß ein Real sein, da Integers keine Nachkommastellen besitzen und eine Rundung nicht möglich ist.

Die Funktion odd bietet in PASCAL eine weitere Möglichkeit, Zahlenwerte zu überprüfen. Damit läßt sich feststellen, ob ein Integer gerade oder ungerade ist. Odd erzeugt ein Ergebnis vom Typ der Booleschen Variablen: Ist die in Klammern stehende Zahl gerade, ergibt odd den Wert False; bei ungerade den Wert True.

```
IF odd(N)
  THEN
    WriteLn ("ungerade")
  ELSE
    WriteLn ("gerade")
```

oder:

```
IF odd(N) THEN
  IF N MOD 2=0 THEN
    WriteLn ("Der Compiler spinnt!")
```

In der kleinen Tabelle auf der nächsten Seite sind alle arithmetischen Funktionen von PASCAL aufgeführt. Die jeweils in Klammern stehende Variable kann jeden Wert vom Typ Real und Integer annehmen, wobei sich das Ergebnisformat von abs und sqr dem Klammerwert anpaßt: abs(−19.372) ist 19.372 und abs(255) ist 255. Alle anderen Funktionen erzeugen Reals: sqrt(16) ist 4.0, nicht 4. Der Grund dafür sind die Berechnungsalgorithmen. Sie bauen auf der Summierung von Reihen auf und arbeiten mit Kommazahlen.

Sie werden bemerkt haben, daß wir bei Ergebnissen vom Typ Real das Wort „ist" verwenden und nicht das Gleichheitszeichen; „ist" bedeutet hier „ist dasselbe wie" statt „ist gleich". Reals sollten nie auf exakte Gleichheit hin überprüft werden, da schon der kleinste Berechnungsfehler bewirken kann, daß zwei formal „gleiche" Reals sich um 1.0E-27 unterscheiden. Für uns ist dieser Unterschied zwar sehr klein, für den Computer jedoch nicht. Statt IF X = Y THEN ... können Sie die abs-Funktion folgendermaßen verwenden:

```
IF abs(X—Y) < Kleinste Differenz
THEN (* etc.*)
```

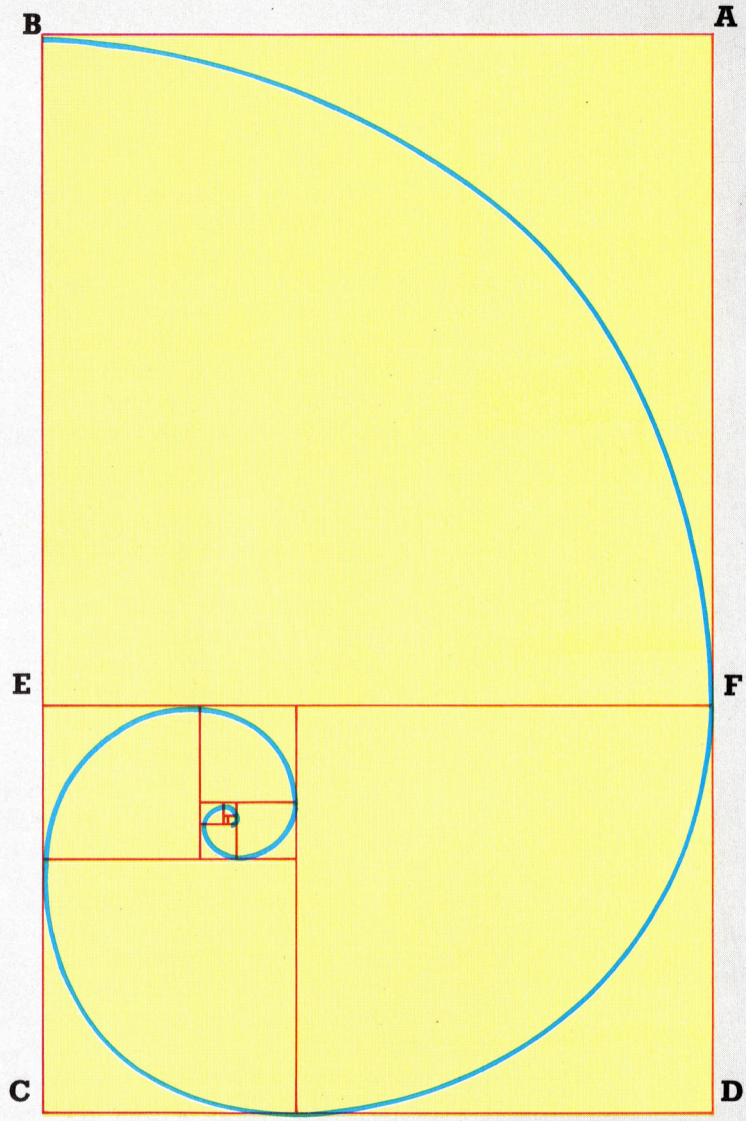
Hierbei wird getestet, ob die Differenz zwischen X und Y unter einem bestimmten Minimalwert liegt, der am Anfang des Programms über CONST definiert ist. Logarithmen werden zur Basis (e) angegeben, nicht zur Basis 10, während exp den Wert e (2,71828...) um die in Klammern stehende Zahl potenziert. Wirth entschied sich ganz bewußt dafür, in PASCAL keine Potenzfunktion aufzunehmen. In BASIC-Programmen findet man oft Zeilen wie

```
500 LET D = B ^ 2+4*A*C
```

Der goldene Schnitt

Eine Gerade läßt sich nur an einem einzigen Punkt derart teilen, daß das Verhältnis der kürzeren Strecke zur längeren Strecke dem Verhältnis der längeren Strecke zur ganzen Linie gleicht. Dieses Verhältnis wird „sectio aurea" oder „goldener Schnitt" genannt und ist – wie Pi – eine irrationale Zahl mit unendlich vielen Nachkommastellen.

Der goldene Schnitt wurde besonders von griechischen Künstlern und Architekten eingesetzt. Einige der interessantesten Beispiele finden sich jedoch in der Natur. So baut die „Seespirale" ihre Schale nach dem goldenen Schnitt. Die Radiusvektoren – nach jeweils 90 Grad aufgezeichnet – teilen einander in diesem Verhältnis.



Die Seiten des Rechtecks ABCD werden nach dem goldenen Schnitt geteilt. Ein Quadrat ABEF kann darin derart untergebracht werden, daß sich ein Rechteck (EFDC) mit dem gleichen Seitenverhältnis wie ABCD bildet. Dieser Vorgang läßt sich bis ins Unendliche fortsetzen.

die das Quadrat einer Zahl nach der langsamsten und ungenauesten Methode bilden. B * B eignet sich dafür weitaus besser. In PASCAL ergibt sqr(N) das Quadrat von N im Integerformat (vorausgesetzt, N ist ein Integer); sqr(X/3) erzeugt ein Real. Und noch eine BASIC-Gewohnheit muß aufgegeben werden: sqrt(x) und nicht sqr(x) ergibt die Quadratwurzel einer Zahl.

Alle Skalartypen haben einen festen und geordneten Wertebereich. Von ihnen können Untertypen abgeleitet werden, die einen Teilbereich des ursprünglichen Skalartyps umfassen. Bei der Definition dieser Untertypen werden die Unter- und Obergrenzen in Klammern angegeben:

```
TYPE byte = 0..255; (*Unterbereich von
   Integer*)
   alpha = 'A'..'Z'; (*Unterbereich von char*)
   farbe = (kreuz, pik, herz, karo); (*ein
   neuer Typ*)
   hauptfarbe =(herz . . karo);(*Unterbereich
   von farbe*)
```

Außer den erwähnten Vorteilen kann ein guter PASCAL-Compiler Variablen vom Typ byte beispielsweise in acht Bytes (bei „gepackter" Darstellung) statt in 32 Bytes unterbringen. Eine Matrix dieser Art belegt nur 25 Prozent der Speicherkapazität, die normale Integerelemente einnehmen würden. Beachten Sie, daß der Typ hauptfarbe zwei mögliche Werte enthält, die nicht vom Booleschen Typ sind, wie:

```
TYPE
   boolean = (false, true);
```

Alle Untertypen übernehmen Datenstruktur und Abläufe des Haupttyps.

Die Definition von alpha kann außer dem Alphabet natürlich auch andere Zeichen umfassen. So enthält der ASCII-Zeichensatz zwischen „Z" und „a" sechs weitere Symbole, darunter die eckigen Klammern („[" und „]"), die zu den 23 Schlüsselwörtern von PASCAL gehören. Sie dienen – wie in vielen anderen Sprachen – als Grenzmarkierung für die Komponenten mehrerer strukturierter Datentypen. Auch BASIC hatte diese Symbole ursprünglich für Matrixindizes eingesetzt. Da jedoch einige ältere Computer nur über den „halben" ASCII-Zeichensatz ohne Kleinbuchstaben, eckige und geschweifte Klammern und einige andere Symbole verfügten, wurde die Syntax auf runde Klammern umgestellt. Der ursprüngliche Apple II war auch an diese Grenzen gebunden. PASCAL bietet folgende Alternativen: (. und .) lassen sich für [und] einsetzen und (* und *) statt { und }. Diese alternative Kennzeichnung von Kommentaren wird oft verwandt, wobei nur bestimmte Compiler die eckigen Klammern verarbeiten können. Die Begrenzungen lassen sich auch mischen: (* Dies ist ein korrekter Kommentar }.

Funktion	Ergebniswert
abs(K)	Absolutwert von K
sqr(K)	Quadrat von K
sqrt(K)	Quadratwurzel aus K
sin(A)	Sinus von A (A im Bogenmaß)
cos(A)	Cosinus von A (A im Bogenmaß)
arctan(T)	Arcus Tangens von T (Ergebnis im Bogenmaß)
ln(K)	Logarithmus von K zur Basis e
exp(L)	Wert von e hoch L

Das goldene Band

Das hier aufgeführte Programm erzeugt Fibonacci-Einheiten und zeigt sie mit den entsprechenden Verhältniszahlen an. Interessanterweise nähert sich dabei das Verhältnis jedes aufeinanderfolgenden Einheitenpaars mehr und mehr den Werten des goldenen Schnittes.

Das Programm enthält aber auch einige Grundlagen, die wir in früheren Folgen dieser Serie untersucht hatten, darunter weitere Beispiele für die REPEAT-Struktur von PASCAL. Versuchen Sie einmal, die Endbedingung der Schleife so zu verändern, daß das Programm endet, wenn die nächste zu berechnende Fibonacci-Einheit größer als MaxInt ist.

```
PROGRAM     Golden (output);

CONST   Epsilon          = 1.0E-08;

TYPE    Fibonacci        = 1.. MaxInt;

VAR
first,
second,
next        : Fibonacci;
ratio,
Gold        : real;
count       : integer;

BEGIN
  WriteLn ('Der goldene Schnitt' : 30);
  WriteLn;
  WriteLn ('Fibonacci Serie:');
  first:=1; (* per Definition*)
  second :=1;
  ratio:=first/second;
  count:=0;

REPEAT
  IF count MOD 10=0 THEN
    BEGIN (*alle 10 Zeilen = Druck der Ueberschrift*)
      WriteLn;
      WriteLn ('First':10,
        'Second' : 10, 'Ratio' : 14);
      WriteLn
    END;

  WriteLn (first : 10, second : 10,
    ratio : 16 : 8);
  count := count + 1;

  Gold : = ratio; (* alter GS*)
  next := first + second;
  first := second; (* nächstes Element der *)
  second := next; (*Serie aufrufen*)
  ratio := first/second

UNTIL abs (ratio - Gold) < Epsilon;

WriteLn;

WriteLn ('die goldene Mitte ist :',
    100 * ratio : 10 : 5 '%')

END.
```

Betrieb mit gewissem Komfort

Der Acorn B bietet vielseitige Möglichkeiten für den Diskettenbetrieb. Da Acorn einer ganzen Reihe von Herstellern detaillierte Informationen über das Betriebssystem und die Steuerung zur Verfügung gestellt hat, gibt es auf dem Markt diverse Laufwerke für diesen Computer.

Die Acorn-kompatiblen Diskettenlaufwerke arbeiten durchweg mit den üblichen $5^1/_4$-Zoll-Floppies. Demnächst wird es auch 8-Zoll-Laufwerke geben.

Da das Disk Filling System (DFS) im ROM gespeichert ist, wird bei der Ausführung der meisten Diskettenbefehle das RAM nicht beansprucht. Das DFS wird durch eine zum Laufwerk gehörige Diskette mit Dienstprogrammen ergänzt, die herstellerspezifisch sind und Formatier- und Kontrollfunktionen beinhalten.

Die Diskettenstation wird mittels Flachbandkabel und 34poligem Disk-Drive-Stecker an der Unterseite des Rechners angeschlossen, über den sich der ganze Datenverkehr mit Einzel- und Doppellaufwerken abspielt.

Steuerung durch Chip

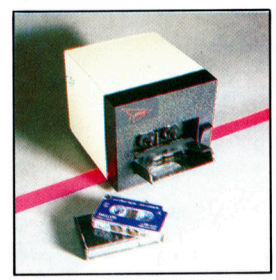
Die Steuerung besorgt der Disk-Controller-Chip 8271, der die Parallel/Seriell-Umwandlung der 8-Bit-Daten vom Rechner für die Übergabe an das Laufwerk und die Rückumsetzung ausführt. Die Formatierung ist 40- oder 80spurig mit zehn Sektoren à 256 Byte vorgesehen, nach Wahl ein- oder zweiseitig, aber immer nur mit einfacher Aufzeichnungsdichte. Pro Seite ergibt sich eine Kapazität von circa 100 KByte bei 40 und 200 KByte bei 80 Spuren.

Die Laufwerke haben Identifikationsnummern (eine 0 für das erste, eine 1 für das zweite). Bei doppelseitiger Aufzeichnung werden die Diskettenseiten als getrennte Laufwerke behandelt (Ziffern 0 und 2 für die erste bzw. 1 und 3 für die zweite Diskette).

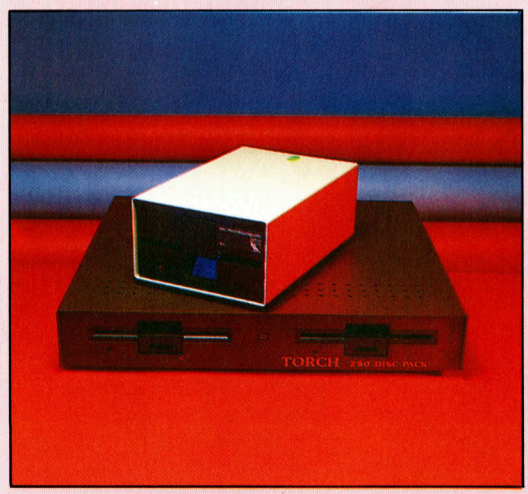
Für jede Diskette (oder jede Seite) wird zu Beginn der ersten Spur ein Dateiverzeichnis angelegt. Das Verzeichnis enthält Informationen zur Disketten- und Datei-Identifizierung und kann in bis zu 27 getrennt ansprechbare „Directories" unterteilt werden, so daß Dateien nach Kategorien geordnet werden können. Es gibt jedoch keine Belegungstabelle (Block Availability Map=BAM), aber dafür das Kommando *COMPACT, das alle beim Löschen entstandenen Lücken findet und die verbleibenden Dateien so zusammenschiebt, daß alle freien Blöcke hinter dem letzten File liegen.

Programm- und Datenfiles sind auf Diskette genau wie auf Cassette zu speichern. Für den Cassettenbetrieb ist zunächst RETURN-*TAPE einzugeben und RETURN-*DISK, um auf Diskette umzustellen. Neben den üblichen Befehlen für die Dateiverwaltung sind beim DFS viele zusätzliche Routinen vorgesehen.

Wie das Acorn-BASIC bietet auch das DFS zahlreiche Möglichkeiten zur Datenmanipulation. Viele Hilfsprogramme, die eigentlich bei einem Heimcomputer-DOS gar nicht zu erwarten sind, wurden hier benutzerfreundlich realisiert. Das System fördert eine effiziente Diskettenorganisation, und die Datenübertragung erfolgt zügig – das Laden eines 20-KByte-Programms dauert rund fünf Sekunden. Nachteilig ist neben dem relativ hohen Preis der Floppy, daß beim DFS höchstens 21 Dateien pro Seite gespeichert werden können. Angesichts der flexiblen Directory-Struktur und der Maximalkapazität von 200 KByte je Seite bedeutet das eine echte Einschränkung.



Der „Hobbit" ist ein speziell für den Acorn B als Floppy-Ersatz entwickeltes Magnetcassettengerät. Das Laufwerk ist vollständig softwaregesteuert; schneller Vor- und Rücklauf, Aufnahme- und Wiedergabefunktionen werden automatisch ausgelöst.



Laufwerke für den Acorn B
Wer den Acorn voll nutzen will, kommt um eine Diskettenstation nicht herum. Das Foto zeigt die beiden meistgekauften Laufwerke – Acorn 100 und Torch Z80. Für ihren Betrieb ist ein ROM mit dem zugehörigen DOS erforderlich, das auf die Platine des Computers gesteckt wird.

Diskettenbefehle des Acorn

Wenn bei einem Kommando eine Datei zu spezifizieren ist, lautet die vollständige Angabe:

BEFEHL:DV.DR.DATEINAME

Dabei ist BEFEHL das betreffende Kommando, DV (=DriVe) die Laufwerknummer (0–3), DR die DiRectory-Identifikation (A–X oder $). Der DATEINAME kann aus maximal sieben Zeichen (die Zeichen #,*, .und : dürfen nicht verwandt werden) bestehen. Bei fehlender Laufwerk- und Directory-Angabe werden standardmäßig die Laufwerknummer 0 und die Directory-Identifikation $ angenommen. Die Standardeinstellung ist mit den Befehlen *DRIVE, *DIR und *LIB zu ändern. Das DFS läßt auch die Verwendung der Sonderzeichen „#“ und „*“ als „Wildcards“ zu. Damit sind bei einigen Befehlen alle Laufwerke, Directories oder Dateinamen mit gleichem Anfangsbuchstaben erreichbar. Für den Parametersatz wird im folgenden Text die Abkürzung <FSP> (File-Spezifikations-Parameter) gebraucht.

Neben den üblichen Cassetten-Befehlen und der Direktzugriff-Routine kennt das DFS die folgenden Diskettenkommandos:

***FORM40,*FORM80 und *VERIFY** (Formatierung und Kontrolle)
dienen zum Formatieren mit 40 oder 80 Spuren bzw. zur Kontrolle. Diese Dienstprogramme befinden sich auf einer laufwerkspezifischen Diskette.

***ACCESS** (Zugangssperre)
sperrt in der Form *ACCESS<FSP> L eine Datei gegen Löschen oder Überschreiben; *ACCESS <FSP> hebt die Sperre wieder auf.

***BACKUP, *DESTROY und *ENABLE** (Kopieren, Löschen und Freigabe)
Mit *BACKUP Quell-DV Ziel-DV wird die Diskette im „Quell“-Laufwerk vollständig auf eine Diskette im „Ziel“-Laufwerk kopiert. *DESTROY <FSP> löscht eine Datei; wenn <FSP> „Wildcards“ enthält, können mehrere Dateien auf einmal gelöscht werden. Wegen der nicht korrigierbaren Wirkung von BACKUP und DESTROY werden diese Befehle erst nach zusätzlicher Eingabe von *ENABLE vom DFS ausgeführt.

***BUILD <FSP>** (ASCII-Datei-Aufbau)
übernimmt die folgenden Tastenanschläge bis zum Drücken der ESCAPE-Taste in ein ASCII-File mit der angegebenen Spezifikation.

***CAT DV**(Katalog)
listet das Dateiverzeichnis des betreffenden Laufwerks auf.

***COMPACT DV** (Verdichten)
verlagert den freien Speicherraum auf der Diskette in einen zusammenhängenden Block hinter der letzten Datei.

***COPY** (Übertragen)
dupliziert eine (oder bei Wildcard-Namen mehrere) Datei(en) auf eine zweite Diskette.

***DELETE <FSP>** (Streichen)
löscht eine einzelne Datei im Dateiverzeichnis; diese Datei kann dann überschrieben werden.

***DIR DR** (Directory-Standardeinstellung)
dient zur Festlegung der Standard-Directory. Alle im folgenden mit *SAVE oder SAVE gespeicherten Dateien werden dieser Directory zugeordnet.

***DRIVE DV** (Laufwerk-Standardeinstellung)
definiert das Standard-Laufwerk.

***DUMP <FSP>**(Speicherabzug)
bewirkt die Ausgabe der genannten Datei im Hexadezimal-Code.

***EXEC <FSP>** (Ausführung)
interpretiert den Inhalt einer Datei wie eine Tastatureingabe. Das ist sehr nützlich für häufig verwendete Befehlsfolgen; derartige Dateien sind mit *BUILD aufzubauen.

***HELP** (Hilfsinformationen)
gibt es in der Form *HELP DFS zur Erzeugung eines Auszugs aus der Befehlsliste (mit Angaben zum Befehlsaufbau) und als *HELP UTILS zur Auflistung der übrigen DFS-Standardroutinen.

***INFO <FSP>** (Zusatzinformation)
dient zum Einholen von Informationen über eine oder (bei Wildcard-Namen) mehrere Datei(en). Sie erfahren Speicheradresse, Ausführungsadresse, Länge in Bytes und Sektorlage.

***LIV:DV.DR** (Bibliothek-Definition)
erklärt die angegebene Directory zur „Bibliothek“. Das ermöglicht den Gebrauch des Kurzbefehls *DATEINAME, wodurch in der betreffenden Directory das namentlich gekennzeichnete Maschinenprogramm aufgesucht, geladen und sofort ausgeführt wird.

***LIST** <FSP> (Auslisten)
bewirkt die Ausgabe der genannten ASCII-Datei mit Zeilennumerierung.

***LOAD <FSP>**
lädt eine Datei unter der RAM-Adresse, von der aus sie gespeichert wurde.

***OPT 1** (Info-Option)
veranlaßt bei jedem Zugriff auf eine Datei die Bereitstellung der gleichen Zusatzinformationen wie unter *INFO (Aktivierung durch *OPT 1 1, Ausschaltung durch *OPT 1 0).

***OPT 4** (Start-Option)
beeinflußt die nach dem Einschalten bzw. nach (SHIFT) BREAK ausgelöste Programmstart-Automatik: *OPT 4 0 unterdrückt den automatischen Start, *OPT 4 1 bewirkt LOAD für die !BOOT-Datei, *OPT 4 2 deren Start und *OPT 4 4 einen Start mit Interpretation von !BOOT.

***RENAME <old FSP> <new FSP>** (Umbenennen)
dient zur Änderung eines Dateinamens und zur Übertragung in eine andere Directory, aber nicht zum Austausch zwischen zwei Laufwerken.

***RUN <FSP>** (Laden mit Starten)
bewirkt das Einlesen eines Maschinenprogramms ins RAM und sofortigen Start.

***SAVE** (Speichern)
kopiert den Inhalt eines RAM-Bereichs auf Diskette (Standardeinstellung für Laufwerk und Directory). Befehlsaufbau:

```
*SAVE "NAME" SSSS FFFF EEEE
RRRR
```

oder

```
*SAVE "NAME" SSSS+LLLL EEEE
RRRR
```

Dabei ist SSSS die Anfangsadresse des Speicherbereichs und FFFF die Endadresse. EEEE ist die Startadresse des gespeicherten Programms und RRRR die Adresse, unter der es wieder geladen werden soll; LLLL ist die Dateilänge in Bytes (alle Angaben hexadezimal). Fehlen RRRR und EEEE, erfolgt Laden und Starten mit dem Adreßwert SSSS.

***SPOOL <FSP>**
(Textzeichen-Speicherung)
eröffnet eine Datei, in der die eintreffende Information als Text-File abgelegt wird. So kann beispielsweise ein BASIC-Programm als ASCII-Datei gespeichert werden.

***TITLE "DISKETTENNAME"** (Umtaufen)
versieht die Diskette im Standardlaufwerk mit dem angegebenen neuen Namen.

***TYPE <FSP>** (Ausdrucken)
dient zur Ausgabe einer ASCII-Datei ohne Zeilennumerierung.

***WIPE <FSP>**(Löschen)
hat die gleiche Wirkung wie *DESTROY, ohne die zusätzliche Angabe von *ENABLE.

Kampf um die Krone

„Lord Of Midnights" von Beyond Software wird als neues Konzept in der Software-Entwicklung betrachtet, da darin Elemente von Kriegs- und Abenteuerspielen kombiniert sind. Bemerkenswert sind die herausragenden Grafik-Techniken, die Tausende verschiedener Perspektiven erlauben.

Computer-Kriegsspiele, die als Nachahmung der Brettspiele entwickelt wurden, unterscheiden sich beträchtlich von Abenteuerspielen, von denen die meisten – wenn auch nur in Ansätzen – auf dem klassischen „Dungeons & Dragons" aufbauen. Bei einem Kriegsspiel sind Strategie und taktische Planung erforderlich, wobei die erforderlichen Bestandteile wie Armeen und Waffen auf einer Karte des Kriegsschauplatzes dargestellt werden. Ein Abenteuer dagegen basiert auf Überraschung und Findigkeit: Der Spieler muß eine Reihe von Problemen lösen, die nacheinander im Spielverlauf offenbar werden. In Lords Of Midnight für den Spectrum mit 48 K hat Beyond Software diese beiden Techniken zur Entwicklung eines neuen Spieltyps kombiniert.

Zum Lieferumfang gehört ein 30seitiges Handbuch, das eine Karte des Land Of Midnight enthält, in dem die Handlung stattfindet. Der Spieler übernimmt die Rolle von Luxor dem Mondprinzen, Lord der Freien. Luxor besitzt den Mondring, mit dem er seine vier Gefährten und jeden Lord der Freien, den er rekrutieren kann, lenkt. Mit Hilfe des Rings kann Luxor durch die Augen dieser Helfer sehen. Die Freien beherrschen den Großteil des Südens, der gegen Angriffe von Doomdark, dem Hexenkönig von Mitternacht, zu verteidigen ist.

Doomdark besitzt die „Eiskrone", eine magische Waffe, die „Eisfurcht" über seine Feinde bringt. Sie schwächt jeden der Freien, der sich dem Land im Norden nähert. Einer der Gefährten aber, Morkin, ist gegen Eisfurcht immun. Seine Aufgabe besteht darin, zu Doomdarks Zitadelle zu gelangen, um die Krone zu zerstören. Die anderen sind währenddessen in den offenen Kampf mit den Armeen Doomdarks verstrickt.

Herkömmliche Kriegsspiele basieren auf einer Karte, mit deren Hilfe der Spieler feststellen kann, wie die feindlichen Armeen aufgestellt sind. Diese bei Lords Of Midnight ständig aktualisierte Karte steht dem Spieler nicht zur Verfügung, sondern ist im Speicher des Spectrum verborgen. Man sieht das Geschehen mit den Augen eines Lord Of The Free. Dieser kann in acht Richtungen sehen, die mittels Tastatur gewählt werden. So ist es möglich, daß eine feindliche Armee hinter den vor ihm liegenden Hügeln liegt, die aber erst dann zu sehen ist, wenn er die Hügel überschritten hat und in die betreffende Richtung schaut.

Am beeindruckendsten sind jedoch die hervorragenden Grafiken. Beyond Software sagt dazu, daß über 32 000 verschiedene Szenen verfügbar sind, die mit den Augen von 32 verschiedenen Personen gesehen werden können. Natürlich kann der Spectrum diese Fülle von Bildschirmen nicht auf einmal im Speicher haben. Deshalb entwickelte Beyond eine als „landscaping" bezeichnete Technik. Die vielen tausend verschiedenen Bildschirme sind aus 15 Formen zusammengesetzt, die in jeweils vier verschiedenen Größen zur Verfügung stehen. Diese Grundformen lassen sich miteinander kombinieren, um detaillierte Ansichten zu gestalten – Hügel, Wälder, Berge, Armeen, Dörfer und Festungen werden gezeigt.

Lord Of Midnights: Für ZX Spectrum mit 48K
Hersteller: Beyond Software
Vertrieb: Rushware
Autor: Mike Singleton
Joysticks: Nicht erforderlich
Format: Cassette

Lord Of Midnights wird mit einem Überleger für die Tastatur geliefert. Damit ist es für den Spieler einfacher, die richtige Befehlstaste zu finden.

Laut Beyond Software enthält das Programm 32 000 verschiedene Ansichten. Bleibt zu fragen, ob sich jemals jemand die Mühe gemacht hat, sie zu zählen. Bemerkenswert ist der „gotisch" geschriebene Text. Beyond hat den Zeichensatz des Spectrum neu definiert, um diesen „teutonischen" Effekt zu erzielen.

Ist doch logisch!

In der Logik haben wir jetzt festen Boden unter den Füßen, speziell auf dem Gebiet der Booleschen Algebra und bei den logischen Grundfunktionen. Dieser Kursteil faßt das bisher Behandelte noch einmal zusammen und enthält darüber hinaus einige Wiederholungsaufgaben und die dazugehörigen Lösungen.

Lassen Sie uns kurz auf den Inhalt der früheren Kursabschnitte zurückblicken: In einem Computer gibt es zahlreiche für spezielle Zwecke entworfene Logikschaltungen. Dazu zählt der von uns gebaute Halbaddierer, der zusammen mit anderen Addierschaltungen die Berechnung einer Summe aus zwei Binärzahlen ermöglicht. Die Schaltung imitiert das Rechnen „per Hand" – bei einer Addition nimmt sie einen Übertrag auf die nächste Spalte vor.

Die Grundelemente, die sogenannten „logischen Gatter", kamen in der Schaltung zum Einsatz. Die Bezeichnung logisches Gatter korrespondiert mit ihrer Funktion (AND, OR und NOT). Durch Wahrheitstabellen konnten wir jedem Gatter die einer bestimmten Kombination von Eingabewerten entsprechenden Ausgaben zuordnen. Zwei Eingänge ergaben vier (2^2), drei Eingänge acht (2^3) Kombinationen usw. Es wurde ferner die Verknüpfung mehrerer Gatter zu einer Schaltung mit der gewünschten Entsprechung zwischen Ein- und Ausgabe gezeigt und durch Wahrheitstabellen beschrieben. Die Fünf-Gatter-Schaltung der ausschließenden OR-Funktion etwa gab nur dann eine 1 aus, wenn ein Eingang 1 war.

Mit speziellen Symbolen kann die Kombination logischer Funktionen wie beim gewohnten Rechnen auf dem Papier vorgenommen werden. Dieser Bereich der mathematischen Logik wird nach seinem Begründer George Boole (1815–64) „Boolesche Algebra" genannt. Jedes der drei logischen Grundelemente hat in der Booleschen Algebra sein besonderes Symbol:

A AND B	A.B
A OR B	A+B
NOT A	$\overline{A}$

Wie in der Zahlenmathematik gibt es auch in der Logik feste Regeln für den Umgang mit Buchstaben und Funktionen, die zur Vereinfachung logischer Ausdrücke genutzt werden können. Diese „Rechenvorschriften" haben wir hier in der rechten Spalte zu einer Tabelle zusammengefaßt.

Die Anwendung dieser Regeln hilft bei der Vereinfachung einer logischen Funktion und vermindert dadurch die Anzahl der zu ihrer Realisierung notwendigen Gatterbausteine. Dem gleichen Zweck dienen auch die bereits erwähnten Karnaugh-Tafeln. Diese ersetzen zwar nicht die algebraischen Methoden, können jedoch beim Versuch der Vereinfachung einer logischen Funktion eine große Arbeitserleichterung darstellen. Mit diesen Tafeln lassen sich innerhalb einer Wahrheitstabelle Gruppen von zwei, vier oder acht Ausdrücken bilden. Diese können zu einfacheren Ausdrükken zusammengefaßt werden. In der Praxis wird oft mit einer Kombination aus k-Tafeln und algebraischen Methoden gearbeitet.

Sonderfälle	
Beziehung	**Entsprechung**
$A.A = A$	$A + A = A$
$A.\overline{A} = 0$	$A + \overline{A} = 1$
$A.0 = 0$	$A + 1 = 1$
$A.1 = A$	$A + 0 = A$
$A.(A + B) = A$	$A + A.B = A$
$A.(\overline{A} + B) = A.B$	$A + \overline{A}.B = A + B$
Gesetze von de Morgan	
$1)\ \overline{A + B} = \overline{A}.\overline{B}$	
$2)\ \overline{A.B} = \overline{A} + \overline{B}$	
Assoziativgesetz	
$A.(B.C) = (A.B).C = A.B.C$	
$A + (B + C) = (A + B) + C = A + B + C$	
Kommutativgesetz	
$A.B = B.A$	
$A + B = B + A$	
Distributivgesetz	
$A.(B + C) = A.B + A.C$	

Auch den Nutzen der logischen AND- und OR-Funktion für die BASIC-Programmierung haben wir schon behandelt – sie können etwa zum Setzen einzelner Bits eines Registers hilfreich sein.

Als Schlußpunkt des ersten Kursabschnittes konnten wir durch die Kombination unserer Kenntnisse von k-Tafeln, Boolescher Algebra und logischen Gattern Schaltungen entwerfen, deren Ausgabe sich nach dem von uns festgelegten Zweck richtete. Wenn Sie sich auch beim Lösen der folgenden Wiederholungsaufgaben sicher fühlen, steht der nächste Schritt an: die fortgeschrittene Theorie der Logik.

Wiederholungsaufgaben

1) Einer Rockband wird die Produktion einer Hit-Single versprochen, wenn sie einen guten Video-Clip liefert UND die Plattenfirma Werbegeschenke zur Platte dazugibt ODER wenn die Gruppe im Fernsehen auftritt. Zeichnen Sie dazu eine Wahrheitstabelle. Wie stehen die Chancen für die Gruppe, einen Hit zu landen, wenn alle Möglichkeiten gleich wahrscheinlich sind?

2) Für den Schul-Schachklub sollen Mitgliedskarten ausgegeben werden. Die Mitgliedschaft soll beschränkt sein auf:

i) Bedienstete der Schule
ii) Schüler des vierten und fünften Semesters mit Leistungskurs Mathematik und Physik
iii) Schüler des sechsten Semesters mit Leistungskurs Mathematik oder Physik

Entwerfen Sie ein Gerät, das die Mitgliedskarten auf Knopfdruck ausgibt. Das Gerät hat folgende Wahltasten:

A= Schüler des vierten Semesters
B= Schüler des fünften Semesters
C= Schüler des sechsten Semesters
D= Schulbediensteter
E= Leistungskurs Mathematik
F= Leistungskurs Physik

Zeichnen Sie die Schaltung dazu.

3) Ein Computer steuert den Bildschirm über das Register 23148(dezimal). Bit 0 ist das niedrigste, Bit 7 das höchste Bit im Register. Der Bildschirm wird auf hohe Auflösung umgestellt, wenn Bit 4 und 5 auf 1 gesetzt sind. Dabei dürfen die anderen Bits nicht verändert werden. Schreiben Sie ein BASIC-Programm, das

a) auf hohe Auflösung umschaltet,
b) die hohe Auflösung wieder ausschaltet.

4) Ein Tresor-Schloß hat mehrere Schalter, die mit Schlüsseln bedient werden müssen. Der Filialleiter, sein Stellvertreter und der Hauptkassierer haben je einen Schlüssel. Mit zwei Schlüsseln kann der Tresor geöffnet werden, allerdings gibt es dabei Bedingungen:

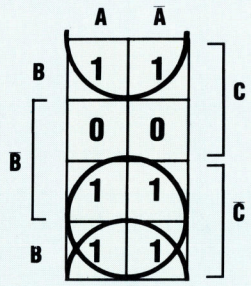
a) In der Hauptgeschäftszeit zwischen 9 und 17 Uhr kann der Tresor nur geöffnet werden, wenn der Hauptkassierer dabei ist.
b) Zu anderen Zeiten läßt sich der Tresor nur vom Filialleiter zusammen mit einer der beiden anderen Personen öffnen.

Entwerfen Sie die entsprechende Schaltung!

5) Um in einem digitalen Datenübertragungs-System besonders hohe Zuverlässigkeit zu gewährleisten, wird jedes Datenbit gleichzeitig in drei unabhängige Leitungen gespeist. Bei richtiger Übertragung liefern alle Leitungen beim Empfänger denselben Wert. Entwerfen Sie eine Schaltung, die bei falschen Daten einer Leitung deren Werte nach dem Mehrheitsprinzip korrigiert.

Lösungen der Übung 5

1a)



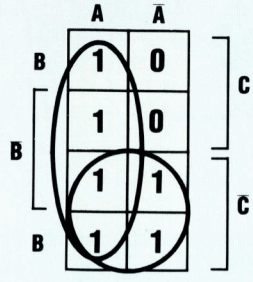
$= \bar{C} + B$

(alternativ: $= \overline{C.\bar{B}}$)

b)

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

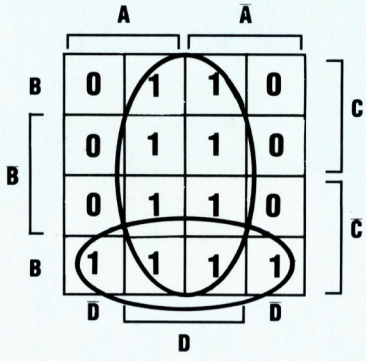
$= \bar{B}.\bar{C} + B.\bar{C} + A.C$ (Gesetz von de Morgan)



$= A + \bar{C}$

(alternativ: $= \bar{A}.C$)

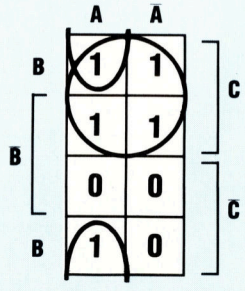
c)



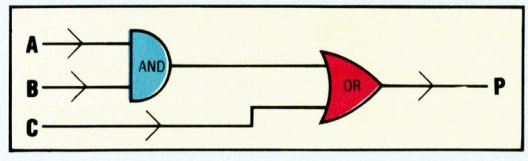
$= D + B.\bar{C}$

2) Die Wahrheitstabelle:

Dezimal	A	B	C	P
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1



Aus der k-Tafel ergibt sich der Ausdruck P=C+A.B.. Hier die Schaltung dazu:



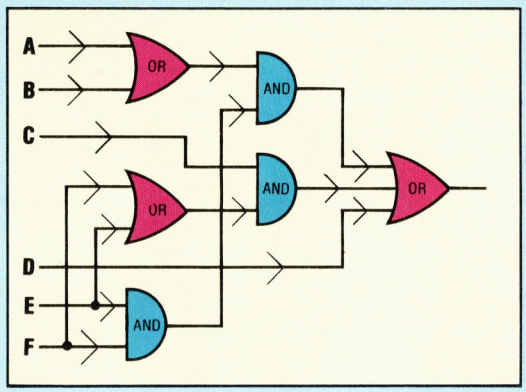
Lösungen der Wiederholungsaufgaben

1)

Guter Video-Clip	Werbe-geschenk	In der Hitliste	Single wird produziert
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Wahrscheinlichkeit des Erfolgs= 5/8 = 62,5%

2)



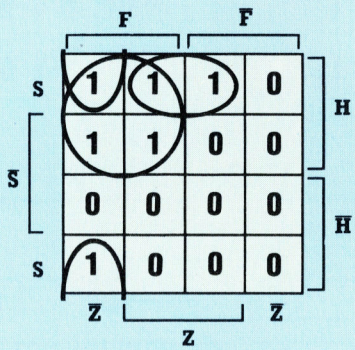
3a) Zum Einschalten der hohen Auflösung:
POKE23148,PEEK(23148)OR48

b) Zum Ausschalten der hohen Auflösung:
POKE23148,PEEK(23148)AND207

4) Die Wahrheitstabelle:

Filial-leiter	Stell-vertreter	Haupt-kassierer	Zeit v. 9–17 Uhr	Tresor geht auf
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

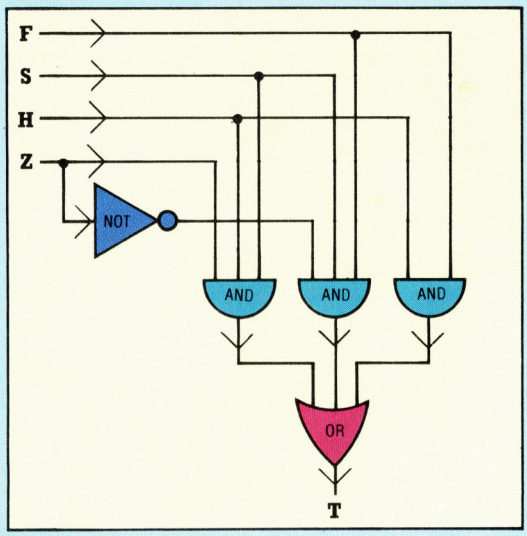
Die k-Tafel:



Hieraus ergibt sich der Ausdruck
T=F.H+F.S.Z̄+S.H.Z. – die Richtigkeit wird bei der Übertragung in Langform klarer, er besagt nämlich: Der Tresor läßt sich von

i) Filialleiter und Hauptkassierer jederzeit,
ii) Filialleiter und Stellvertreter außerhalb der Geschäftszeit und von
iii) Stellvertreter und Hauptkassierer in der Zeit von 9 bis 17 Uhr öffnen.

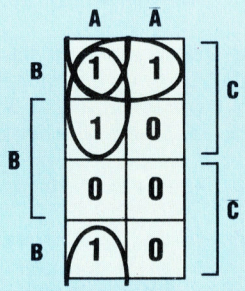
Die Schaltung dazu muß so aussehen:



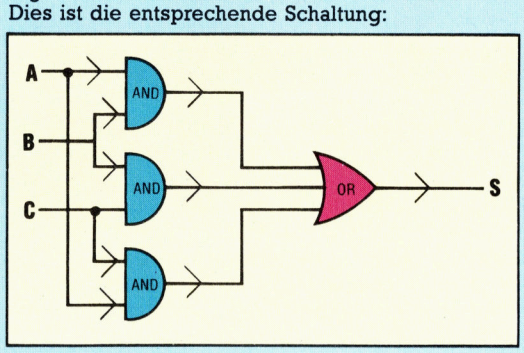
5) Wahrheitstabelle:

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

Aus der k-Tafel:



ergibt sich der Ausdruck: S=A.B+A.C+B.C.
Dies ist die entsprechende Schaltung:



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.

Disassembler = Disassembler
Ein Disassembler ist ein Programm, das Maschinencode in Assemblersprache zurückübersetzt. Es wandelt die Bytes in mnemotechnische Abkürzungen wie LDA oder JMP um, analysiert die Adressierungsart und gibt auch den Operanden mit aus, der in den nächsten Bytes steht.

Disassembler sind sehr nützlich, wenn Maschinenprogramme zu testen oder umzuschreiben sind.

DMA = DMA
DMA heißt „Direct Memory Access", auf deutsch „direkter Speicherzugriff". Darunter versteht man schaltungstechnische Maßnahmen, die mehreren Systemkomponenten gleichzeitig den Zugang zum selben Speicher erlauben. Eine wichtige Anwendung ergibt sich bei der Grafikprogrammierung: Wenn der Video-Steuerchip den Bildspeicher selbst auslesen kann, ohne für jedes Byte die CPU zu befragen, wird die Sache wesentlich effizienter.

Double Density =
Doppelte Aufzeichnungsdichte
Die Kapazität einer Diskette hängt einmal davon ab, ob sie einseitig (Single Sided = SS) oder doppelseitig (Double Sided = DS) beschichtet und damit bespielbar ist, außerdem aber von der Aufzeichnungsdichte (Spurendichte und Bitdichte in der Spur). Bei Disketten mit doppelter Dichte (Double Density = DD) ist auf einer Seite die zweifache Datenmenge unterzubringen wie bei einfacher Dichte (Single Density = SD). Die DD-Aufzeichnung erfordert eine wesentlich präzisere Magnetkopfsteuerung, was anfänglich mit merklich höheren Kosten und geringerer Zuverlässigkeit verbunden war.

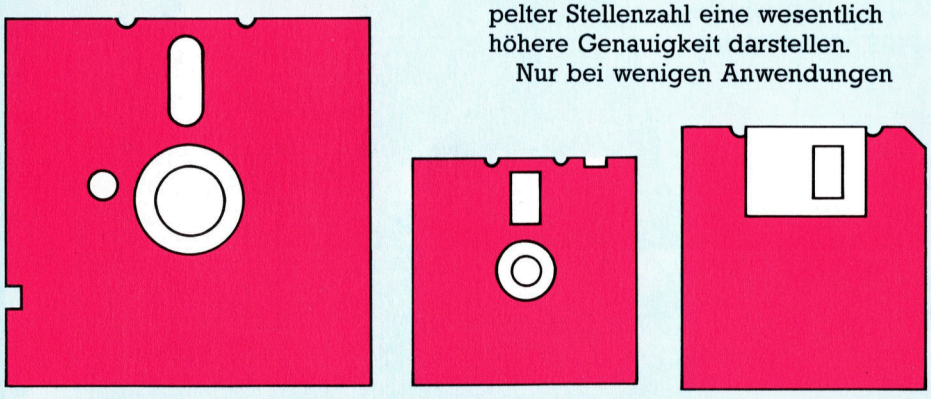
Infolge des Fortschritts der Diskettentechnologie in den letzten Jahren ist doppelte Dichte inzwischen praktisch Standard. Auf einer 5 1/4-Zoll-Diskette sind je nach Formatierung und System 90 KByte bis 1 Megabyte an Daten unterzubringen und auf den neueren 3 1/2-Zoll-Disketten bis zu 700 KByte.

Double Precision =
Doppelte Genauigkeit
Reale Zahlenvariablen, wie sie in BASIC am häufigsten sind, werden im allgemeinen mit acht oder neun Dezimalstellen gespeichert. Dabei spricht man von „Single Precision" (einfacher Genauigkeit), wogegen Double-Precision-Variablen mit doppelter Stellenzahl eine wesentlich höhere Genauigkeit darstellen.

Nur bei wenigen Anwendungen benötigt man Ergebnisse, die mehr als acht Stellen haben – wozu also 16 Stellen? Deren Notwendigkeit ergibt sich aus den Rundungsfehlern, die bei jeder Rechenoperation wie Addieren, Subtrahieren usw. entstehen. Bei umfangreichen Berechnungen, die beispielsweise beim Maschinenbau, bei Statistiken und Wetterprognosen durchgeführt werden, muß der Programmierer genau darauf achten, daß keine Rundungsfehler entstehen, die das Ergebnis unkontrollierbar verfälschen. Das Rechnen mit doppelter Genauigkeit entschärft dieses Problem, aber ohne es aus der Welt zu schaffen.

Download =
Laden (aus Fremdsystemen)
Unter „Telesoftware" versteht man Programme, die aus einer Zentrale über Telekommunikation abrufbar sind, und als Downloading werden die Übernahme und das Abspeichern im RAM oder auf Diskette bezeichnet. Ursprünglich wurde dieser Begriff bei der Übertragung von Daten aus einem Großrechner zu den angeschlossenen Terminals oder kleineren Rechnern verwendet. In England können Sie über das PRESTEL-Bildschirmtextsystem bereits telefonisch (gegen Gebühr) viele Programme für Ihren Heimcomputer abrufen.

Das Downloading kann auch den Softwareverkauf in Zukunft ganz anders gestalten. Zur Zeit ist die Lagerhaltung für die Händler ein echtes Problem, weil sich die Popularität von Spielprogrammen ständig ändert. Ein Lösungsversuch sind nachprogrammierbare Cartridges, die der Benutzer im Geschäft preisgünstig neu laden lassen kann. Ein spezielles Terminal überträgt dabei das Programm von einer Zentrale auf die Cartridge.

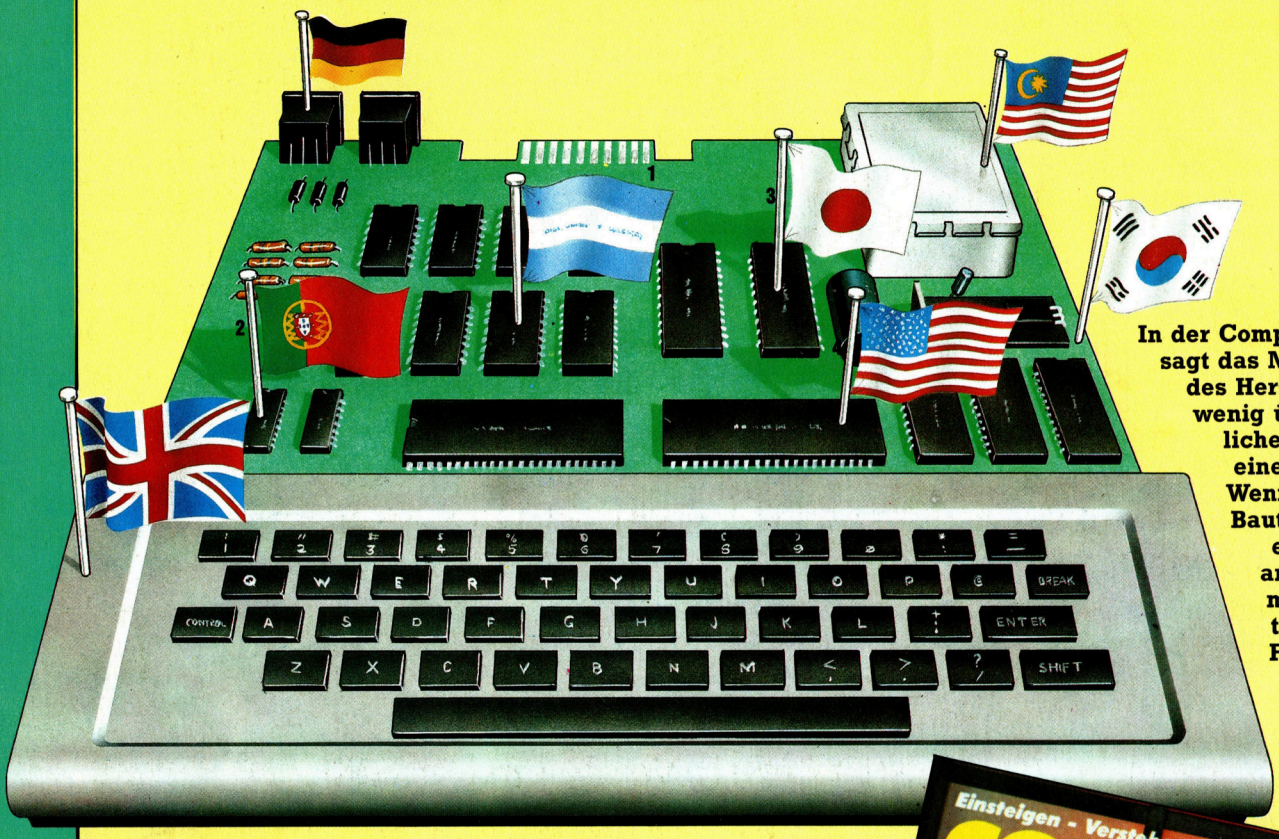


Die Spurendichte und die Bitdichte innerhalb einer Spur sind vom Diskettendurchmesser völlig unabhängig. Abgebildet sind hier im gleichen Maßstab (1) die 5 1/4-Zoll-Minifloppy, (2) die 3-Zoll- und (3) die 3 1/2-Zoll-Microfloppy.

Bildnachweise
981, 982, 983, 992, 996, 998, 1003: Ian McKinnell
984, 985, 990: Kevin Jones
989: Liz Heaney, Kevin Jones
993, 994: Chris Stevens
1001, 1007, 1008, U3: Liz Dixon

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

computer kurs Heft 37



In der Computer-Industrie sagt das Markenzeichen des Herstellers oft nur wenig über den wirklichen Produzenten eines Gerätes aus. Wenn man sich die Bauteile im Innern eines Rechners ansieht, erkennt man an den unterschiedlichen Firmenbezeichnungen, daß Computer aus internationalen Komponenten bestehen.

Spezialdruck

Selbst mit dem einfachsten Matrixdrucker lassen sich Ausdrucke interessanter gestalten. Wir erklären, wie solche Effekte erzeugt werden.

Wasserspiele

Weiter geht es mit unserem Basic-Spiel „U-Boot-Jagd". In dieser Folge wird u. a. gezeigt, wie die Darstellung einer Explosion programmiert wird.

Weckprogramm

Das im letzten Kursabschnitt erstellte Modul kommt nun zum praktischen Einsatz, z. B. als programmierbarer Wecker.

Leitungswechsel

Diese Folge der Logik-Serie beschäftigt sich mit Codier- und Decodierschaltungen.

Taschencomputer

Täglich werden Taschencomputer benutzt, um vor Ort Berechnungen durchzuführen. Wir vergleichen mehrere Rechner.