Einsteigen - Verstehen - Beherrschen

DM 3,80 öS 30 sfr 3,80

computer kurs

Heft 37

- Ataris Star Raiders
- Memotech RS218
- Acorn B-Sprites
- Minis aus Japan
- Micro-Produktion
- Der programmierte Wecker

Ein wöchentliches Sammelwerk

computer kurs

Heft 37

Inhalt

WIE SIE JEDE WOCHE IHR HEFT BEKOMMEN

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

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

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

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

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

Abonnement

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

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

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

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

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

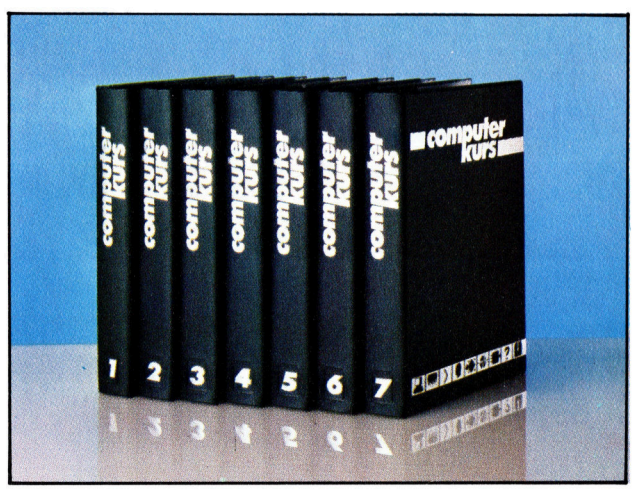
Schweiz: Der Sammelordner kostet sfr 15. Bitte wenden Sie sich an Ihren Kiosk; dort werden Sie jederzeit die gewünschten Exemplare erhalten.

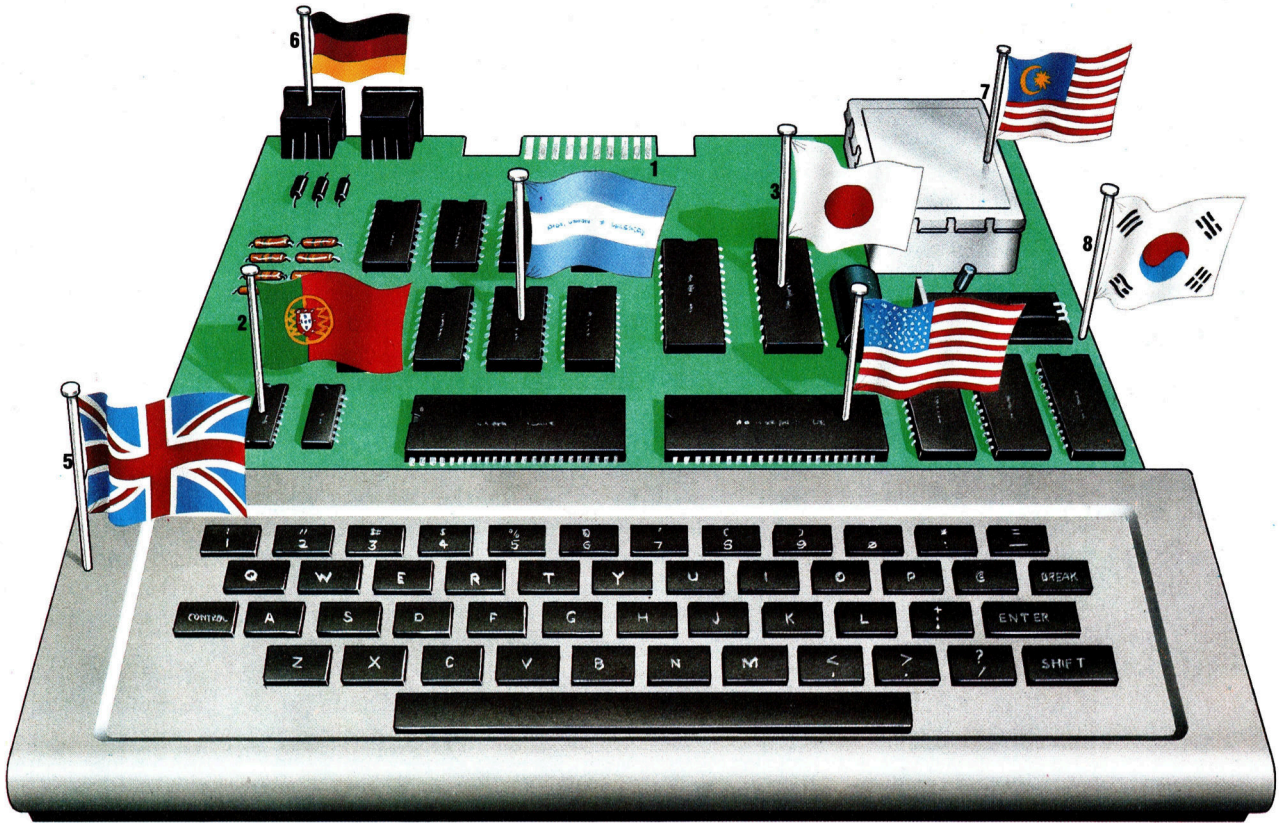
INHALTSVERZEICHNIS

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

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

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





Weltweiter Handel

Die Computer-Industrie gehört zu den wenigen Branchen, in denen das Markenzeichen des Herstellers nur wenig über den wirklichen Produzenten eines Gerätes aussagt. Die Bauteile der Computer werden in zahlreichen verschiedenen Firmen gefertigt.

Die Produktion von Computern ist ein weltumspannendes Geschäft. Der Schneider CPC 464 stammt beispielsweise aus Korea, der Acorn B größtenteils aus Hongkong. Und Sinclair versteht sich als reine Forschungs- und Entwicklungsfirma – Produktion von Bauteilen und Gerätemontage werden von Vertragsfirmen abgewickelt.

Die Ursache dafür ist der extreme Preisdruck. Die gesamte Computer-Entwicklung steht unter dem Zwang, möglichst niedrige Herstellungskosten zu sichern. Damit der Preis gering bleibt, werden die Platinen so klein wie möglich entworfen – also muß auch die Anzahl der Bauteile auf ein Minimum beschränkt werden. Dabei sind die Chips selbst nicht teuer – erst die Arbeitszeit für das Einsetzen der zahlreichen Bauteile in die Platine treibt die Kosten hoch. Und bei der großen Zahl der Kontaktstellen steigt das Risiko, Fehler „einzubauen".

Als Ausweg werden in modernen Computern zunehmend ULA- (uncommitted logic array) Chips verwendet. Sie sind zwar teuer, ersetzen aber Dutzende kleinerer Chips auf einer einzelnen Platine.

Ein großer Teil aller Microchips stammt aus Kalifornien. Der Begriff Silicon Valley ist bereits zum Markenzeichen der dort ansässigen Computer-Firmen geworden. Die Chips werden in einem Kunststoff- oder Keramikgehäuse untergebracht. Die Technik dafür ist relativ einfach, aber arbeitsintensiv. Die Folge dieses Problems ist: Die Rohware wird in die unterschiedlichsten Niedriglohn-Länder geschafft und bekommt erst dort ihr „Kleid".

Nach dem Entwurf einer Platine wird eine Vertragsfirma für die Fertigung benötigt. Wie die Herstellung von Halbleiter-Chips erfordert auch die Platinenherstellung komplizierte Technik und erhebliche Investitionen in den Maschinenpark. Dies ist eine Aufgabe für Spezialfirmen, die nach detaillierten Entwürfen die Produktion der grünen Kunstharzplatten mit ihren metallischen Leiterbahnen übernehmen.

Auch der Aufbau der Platinen selbst richtet sich nach den Herstellungskosten. Es ist zwar technisch möglich, Sandwich-Platinen aus übereinanderliegenden Metall- und Kunststoffschichten anzufertigen, aber ein guter Ingenieur vermeidet sie aus Kostengründen. Da-

Dieser imaginäre Computer besteht aus Bauteilen, die in zahlreichen verschiedenen Ländern gefertigt wurden. Die Chips stammen aus: 1) El Salvador, 2) Portugal, 3) Japan und 4) USA. Gehäuse, Tastatur und Endmontage sind aus 5) England, Steckanschlüsse aus der 6) BRD, Videomodulator aus 7) Malaysia und die Rechnerplatine aus 8) Korea.

gegen hat sich für Computer-Platinen folgende Technik durchgesetzt: Die in der Platine für die Bauteil-Anschlüsse vorgesehenen Löcher werden mit einem metallischen Überzug versehen. Der Kontakt wird dadurch verbessert, und die Betriebssicherheit nimmt zu.

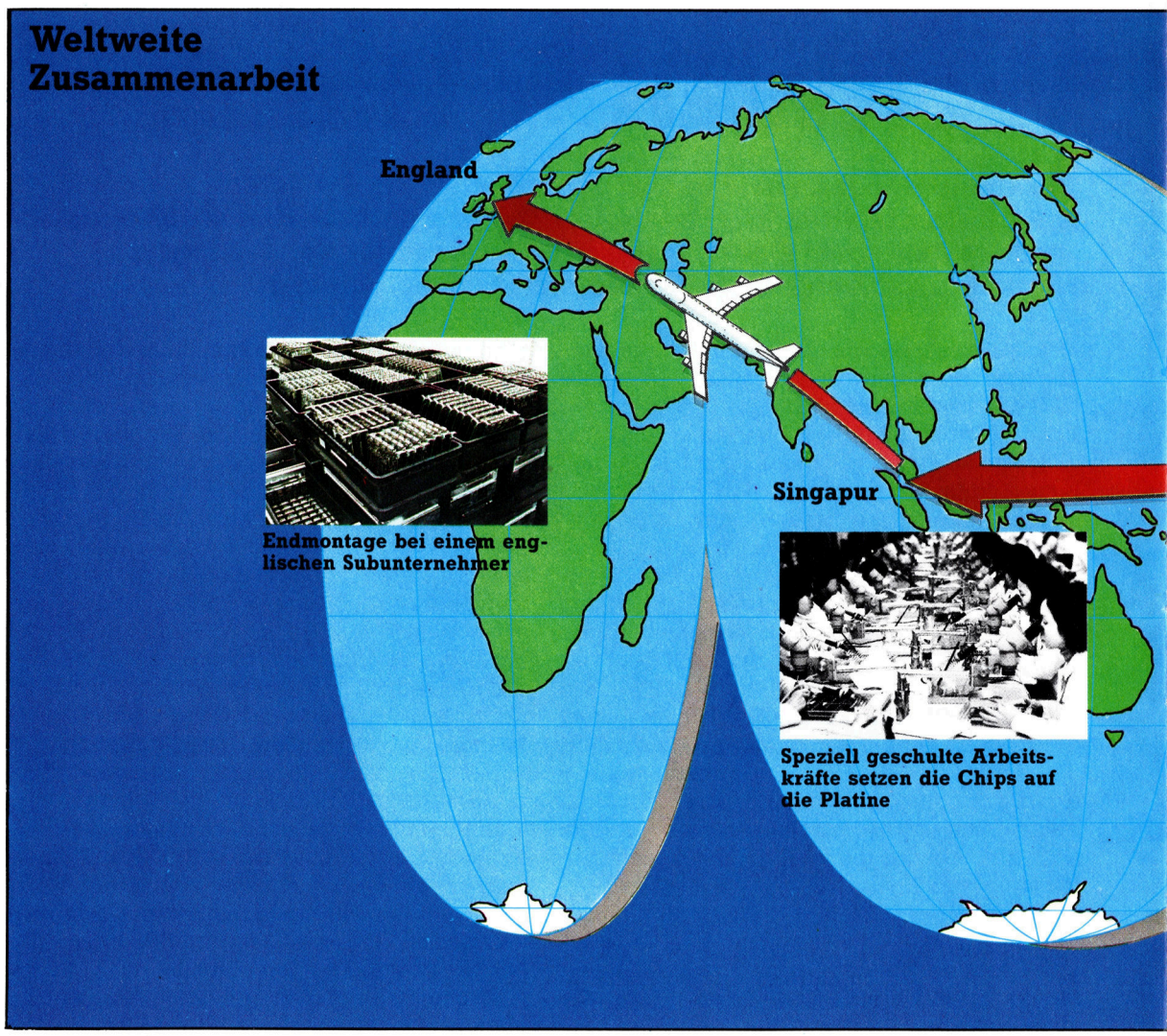
Die Endmontage

Die Suche nach günstigen Zulieferbetrieben für Netzgeräte, Tastaturen und Video-Modulatoren erstreckt sich über alle Länder. Dabei ist der Preis der wichtigste Faktor. Die Bauteile werden oft zuvor an eine Vertragsfirma geliefert, die sie komplettiert – nicht selten hat diese ihren Sitz auf einem anderen Kontinent. Selbst ein simples Plastikgehäuse wird von einem externen Zulieferbetrieb hergestellt, der über die teuren Spritzguß-Maschinen verfügt und sie kostengünstig einsetzen kann.

Für die Endmontage gibt es zwei mögliche Verfahren, mit Hilfe von zahlreichen Arbeitskräften mit niedrigen Löhnen oder durch vollautomatische Fertigung. In Singapur, Korea und Hongkong geht man den ersten Weg, Europa, Japan und die USA setzen auf die automatisierte Produktion. Industrieroboter bestükken die Platinen an Fertigungsstraßen. Die Bauteile werden direkt von großen Rollen zugeführt – nur der Rollenwechsel ist noch Aufgabe eines Arbeiters.

Nach dem Bestucken der Platine wandert sie in ein Lot-Schwallbad. Dabei werden die auf der Platinenrückseite vorstehenden Bauteil-Anschlüsse verzinnt. Durch die Kapillarwirkung gerät das flüssige Zinn auch in die metallisierten Löcher und sichert den guten elektrischen Kontakt zwischen Leiterbahn und Bauteil-Anschlußdraht. Zum Schluß werden die Platinen geprüft, verpackt und in ein Lager geschafft, wo sie auf die Weiterbeförderung bzw. ihren Verkauf an den Endabnehmer warten. Das hört sich einfach an, jeder dieser Schritte wirft jedoch spezifische Probleme auf.

Das erste Problem hat mit dem „Timing" zu tun. Viele Bauteile aus unterschiedlichen Quellen müssen gleichzeitig für die Endmontage zur Verfügung stehen. Wichtigste Person in diesem Ablauf ist der Einkäufer. Er muß die Bauteile nicht nur billig erwerben, sondern auch für pünktliche Lieferung sorgen. Das Fehlen eines einzigen Bauteils kann katastrophale



Folgen haben – stillstehende Produktionsanlagen kosten Geld, verzogerte Lieferungen kosten wichtige Kunden. Aber auch die Einzelpreise dürfen bei der Planung nicht aus dem Blickfeld geraten – ein paar Pfennige mehr oder weniger für einen simplen Stecker addieren sich bei den heute üblichen Stückzahlen zu respektablen Summen.

Ebenso anfällig für Fehler ist die Endmontage selbst, gleichgültig, ob sie per Hand oder mit Automaten vorgenommen wird. Ein falsch eingesetztes oder fehlendes Bauteil macht die gesamte Platine unbrauchbar. „Kalte Lötstellen" gibt es auch beim Schwallbad-Verfahren, und manchmal entspricht eine komplette Bauteil-Charge nicht den geforderten Ansprüchen.

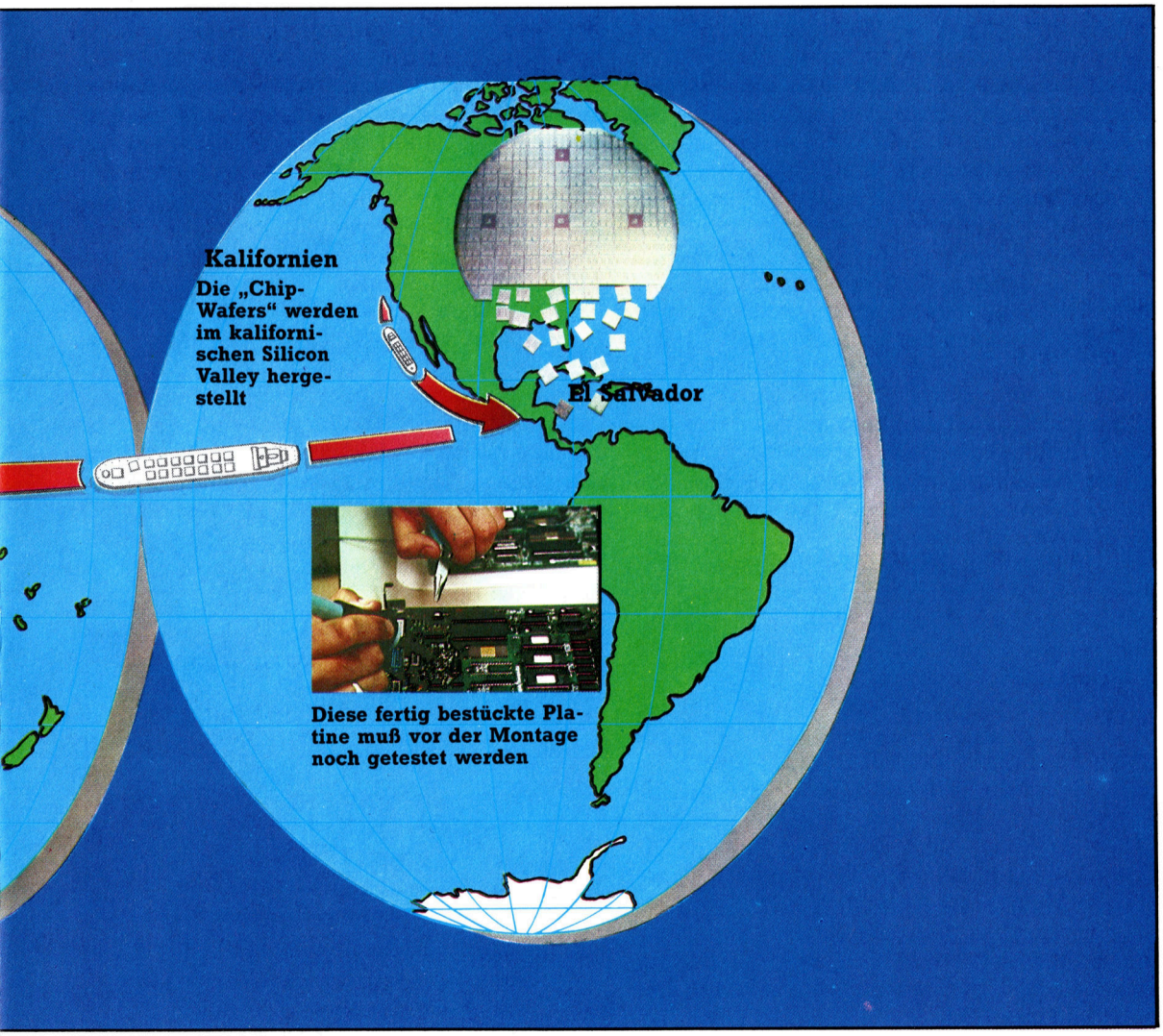
Härtetests zeigen Fehler auf

Als Ausweg bleiben nur ständige Tests, sowohl der Bauteile als auch der gesamten Platine. Stichproben bei zugelieferten Bauteilen sind bei den meisten Computer-Herstellern ohnehin die Regel, unterschiedlich schwere „Härtetests" eine Selbstverständlichkeit. Die Prüfung von Platinen ist sehr aufwendig und stellt hohe Ansprüche an die Fähigkeiten des verwendeten Test-Computers. Verzichten kann man darauf nicht – kein Abnehmer findet sich auf Dauer mit großen Ausfallraten ab. Es ist sogar üblich, daß potentielle Großkunden während der Produktion unabhängige Tests von Bauteilen und Geräten durchführen lassen.

Der wichtigste Test bleibt natürlich trotzdem der praktische Einsatz. Professionelle Computer werden oft tagelang unter festverdrahteten Prüf-Routinen oder mit speziellen Testprogrammen „eingefahren". Die Zeit bringt unbarmherzig jeden Mangel der Maschine an den Tag.

Bei der erwähnten Vielzahl von Unwägbarkeiten ist es kein Wunder, daß Heimcomputer oft verspätet ausgeliefert werden oder nicht richtig funktionieren. Die für die Endmontage zuständige Firma ist von der rechtzeitigen Belieferung durch die Bauteil-Hersteller abhängig. Werbung und Vertrieb sind wiederum davon abhängig, daß die Geräte rechtzeitig in der richtigen Form vorliegen. Auch Händler und Kunden erwarten zu Recht, daß ein Gerät pünktlich und in funktionsfähigem Zustand bei ihnen eintrifft.

Aus Gründen der Kostenminimierung werden die einzelnen Computer-Bauteile in unterschiedlichen Ländern gefertigt. Die billigeren Arbeitskräfte aus Niedriglohn-Ländern haben zu einem deutlichen Preisverfall beigetragen. Die Fortschritte bei der Fertigungsautomatisierung erlauben jedoch inzwischen auch in hochindustrialisierten Ländern eine kostengünstige Produktion. So wird etwa der Oric Atmos vollständig in England hergestellt, obwohl die Firma zur Versorgung ausländischer Märkte auch Produktionsanlagen in Singapur unterhält.



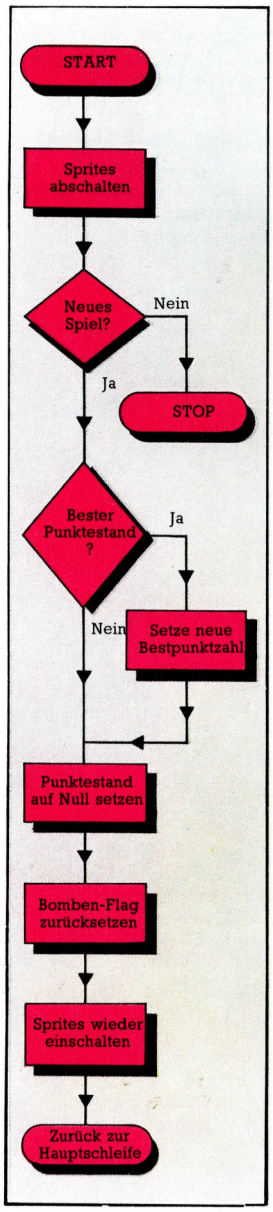
Die Weltkarte zeigt den Weg der Computer-Bauteile auf ihrer Reise von einem Fertigungsschritt zum nächsten.

Wasserspiele

In diesem Teil des Kurses werden wir die Routinen zur Darstellung einer Explosion und die Spielende-Routine erklären.

Im letzten Teil unseres Kurses haben wir gezeigt, wie das Sprite-Kollisions-Register eingesetzt wird. Die HIT-Unterroutine ab Zeile 5000 muß drei Aufgaben erfüllen. Als erstes muß sie eine Explosion an der Stelle auf dem Bildschirm darstellen, an der die beiden Sprites kollidiert sind. Danach muß die Routine den Punktestand des Spielers um den Wert des U-Bootes erhöhen, der sich aus seiner Geschwindigkeit (DX) und seiner Tiefe (Y3) errechnet. Als letztes müssen die Koordinaten für das nächste U-Boot zurückgesetzt werden.

In Zeile 5010 wird eine 0 in das Kollisionsregister V+30 gePOKEt, um es zu löschen. In Zeile 5030 wird der Explosion eine X-Koordinate zugeordnet, die zehn Pixel weiter rechts von der Koordinate der Wasserbombe festgelegt wird. Diese leichte Verschiebung bewirkt, daß die Explosion zentral über der Wasserbombe erfolgt. Da X2 seinen Wert aus der X-Koordinate des Schiffes (X0) erhält, ist sein Wert auf ein Maximum von 245 begrenzt. Die Y-Koordinate für die Explosion wird direkt von der des U-Bootes übernommen.



Sprite-Explosionen

Das Explosions-Sprite wird als Sprite 1 bezeichnet. In Zeile 5040 wird Bit 1 des Registers V+21 auf eins gesetzt. Dadurch wird Sprite 1 aktiviert, ohne daß die Werte anderer Bits dieses Registers beeinflußt werden. Übrigens erscheint das Explosions-Sprite über bzw. vor den U-Boot- und Wasserbomben-Sprites. Dieser Umstand wird als Sprite-Priorität bezeichnet. Die Regeln hierfür sind sehr einfach. Sprites mit einer niedrigeren Kennziffer erscheinen vor Sprites mit einer höheren Kennziffer. Aus diesem Grund hat das Explosions-Sprite die Kennziffer 1 erhalten.

Die Farbe des Explosions-Sprites wird mit Hilfe der Speicherstelle V+40 und des VIC-Chips kontrolliert. Durch schnelles UmPOKEn der Farbwerte mittels einer FOR. . .NEXT-Schleife kann man einen sehr interessanten Effekt erzielen. Eine äußere FOR. . .NEXT-Schleife wiederholt diesen Vorgang 20mal (Zeilen 5060–5100). Sobald die Explosion erfolgt ist, müssen alle drei Sprites (Explosion, Wasserbombe und U-Boot) vom Bildschirm verschwinden. Dies wird in Zeile 5130 des Programms ausgeführt.

Wie bereits erwähnt, muß der Punktestand des Spielers mit Hilfe der Unterroutine ab Zeile 5500 aktualisiert werden. Soll der Punktestand um den Wert des U-Bootes erhöht werden (im Gegensatz zu einem Punkteabzug, wenn das U-Boot unbeschädigt den rechten Rand des Bildschirmes erreicht), wird der Wert von DS auf eins gesetzt. Abschließend müssen die Koordinaten mittels der Routine bei Zeile 2500 zurückgesetzt und das U-Boot-Sprite wieder eingeschaltet werden, bevor ein anderes U-Boot über den Bildschirm fahren kann. Außerdem ist das Flag, das den Abwurf einer Wasserbombe signalisiert, auf Null zu setzen, damit der Spieler wieder neue Wasserbomben abwerfen kann.

Nach einer Zeitspanne von drei Minuten verläßt das Programm die Hauptschleife und geht zu Zeile 400. Als wir uns das erste Mal mit der internen Uhr des Commodore 64 befaßt haben, enthielt Zeile 400 nur eine einfache END-Anweisung. Die END-OF-GAME-Routine dagegen gestattet den Beginn eines neuen Spieles und eine Aufzeichnung der besten Punktzahlen. Das Flußdiagramm zeigt die Aufgaben, die von einer solchen Routine erfüllt werden müssen. Die Zeilen 400 bis 660 führen diese Aufgaben aus. Der größte Teil des Programmcodes erklärt sich von selbst. Denken Sie daran, daß CHR$(19) den Cursor in der oberen linken Ecke des Bildschirmes positioniert und daß CHR$(144) bewirkt, daß nachfolgend gePRINTete Buchstaben schwarz gefärbt werden.

In diesem kurzen Programm für den Commodore 64 haben Sie gelernt, wie man ein einfaches animiertes (also bewegtes) Spiel konstruiert. Während der Entwicklung des Programms haben wir uns mit allen wesentlichen Aspekten der Programmierung eines derartigen Spieles in BASIC befaßt. Ein Weg zur Erweiterung des Spieles mit eigenen Ideen wäre beispielsweise die Verwendung der vier noch ungenutzten Sprites.

Tabelle der verwendeten Variablen

V	Beginn der VIC-Chip-Register
FL	Wird nach Abwurf einer Wasserbombe auf eins gesetzt
SC	Gegenwärtiger Punktestand des Spielers
HS	Bisher höchster Punktestand
TI$	Interne Uhr des Commodore 64
X0	X-Koordinate des Schiffes
X2,Y2	X- und Y-Koordinaten der Wasserbombe
X3,Y3	X- und Y-Koordinaten des U-Bootes
H3,L3	Höherw. und niederw. Byte der X-Koordinate des U-Bootes
DX	Pixelzahl, um die die X-Koord. des U-Bootes erhöht wird
DS	Flag (Kennung), ob ein Punktestand erhöht (DS=1) oder verringert werden muß (DS=−1)

U-Bootjagd – Das komplette Listing

```
10 REM ****************************
30 REM ** 64 PROGRAMMING PROJECT **
70 REM ****************************
90 POKE55,0:POKE56,48:CLR:
REM LOWER MEMTOP
100 V=53248:FL=0:SC=0
110 GOSUB1000: REM SCREEN SETUP
120 GOSUB2000: REM SPRITE CREATION
130 GOSUB2500: REM SET SUB COORDS
140 TI$="000000"
200 REM **** MAIN LOOP ****
210 REM   ** TIMER **
220 PRINTCHR$(19);:PRINTTAB(14)CHR$(5);
"TIME "MID$(TI$,3,2)":"RIGHT$(TI$,2)
225 IFVAL(TI$)>259 THEN 400:REM END GAME
230 GET A$
240 IF A$="Z" THEN X0=X0-1.5:IF X0<24
THENX0=24
250 IF A$="X" THEN X0=X0+1.5:IF X0>245
THENX0=245
260 IF A$="M" AND FL=0 THEN GOSUB3000:
REM SET UP DEPTH CHARGES
270 REM  ** MOVE SHIP **
290 POKE V,X0
300 REM ** MOVE SUB **
310 X3=X3+DX
320 REM**IF SUB REACH EDGE OF SCREEN **
330 IF X3>360 THEN DS=-1:GOSUB5500:
GOSUB2500
340 H3=INT(X3/256):L3=X3-256*H3
350 POKE V+6,L3
360 IF H3=1 THEN POKE V+16,PEEK(V+16)OR8:
GOTO380
370 POKE V+16,PEEK(V+16)AND247
380 IF FL=1 THEN GOSUB4000:
REM MOVE DEPTH CHARGE
390 GOTO 200:REM RESTART MAIN LOOP
400 REM **** END OF GAME CONDITIONS ****
410 REM   ** TURN OFF SPRITES **
420 POKE V+21,0
430 REM   ** RESET SUB & SHIP COORDS **
440 X0=160:GOSUB 2500
450 INPUT"ANOTHER GAME (Y/N)";AN$
460 IF AN$<>"Y"THENEND
480 REM   ** RUBOUT MESSAGE **
490 PRINT CHR$(19):REM HOME CURSOR
500 FOR I=1 TO 120
510 PRINT" ";
520 NEXT I
540 REM   ** SET HI SCORE **
550 IF SC>HS THEN HS=SC
560 PRINT CHR$(19);CHR$(144);" SCORE 000":
SC=0
570 PRINT CHR$(19);
580 PRINT TAB(26);CHR$(144);"HI SCORE";HS
600 REM   ** RESET TIMER AND FLAG **
610 TI$="000000":FL=0
630 REM   ** TURN ON SUB & SHIP **
640 POKE V+21,9
660 GOTO200: REM RESTART LOOP
1000 REM **** SCREEN SETUP ****
1010 PRINT CHR$(147):REM CLEAR SCREEN
1030 REM   ** COLOUR SEA **
1040 POKE 53281,14:POKE 53280,6
1050 FOR I=1264 TO 1943
1060 POKE I,160:POKE I+54272,6
1070 NEXT
1090 REM   ** SEA BOTTOM **
1100 FORI=1944 TO 2023
1110 POKE I,102:POKE I+54272,9
1120 NEXT
1130 POKE 650,128:REM REPEAT KEYS
1150 REM   ** SCORE **
1160 PRINT CHR$(19);CHR$(144);" SCORE 000"
;SPC(16);"HI SCORE 000"
1170 RETURN
2000 REM **** SPRITE CREATION ****
2020 REM   ** READ SHIP DATA **
2030 FOR I= 12288 TO 12350
2040 READ A:POKE I,A:NEXT I
2060 REM  ** READ SUB DATA **
2070 FOR I=12352 TO 12414
2080 READ A:POKE I,A:NEXT
2100 REM   ** READ CHARGES DATA **
2110 FOR I = 12416 TO 12478
2120 READ A:POKE I,A:NEXT
2140 REM   ** READ EXPLOSION DATA **
2150 FOR I = 12480 TO 12542
2160 READ A:POKE I,A:NEXT
2180 REM   ** SET POINTERS **
2190 POKE 2040,192:POKE 2041,193:POKE 2042
,194
2200 POKE2043,195
2220 REM   ** SET COLOURS **
2230 POKE V+39,0:POKE V+40,1:POKE V+41,0
2240 POKE V+42,0
2260 REM   ** SET INITIAL COORDS **
2270 POKE V+1,80:X0=160: REM SHIP COORDS
2280 POKE V+29,15:POKE V+23,2
2300 REM   ** TURN ON SPRITES 0 & 3 **
2310 POKE V+21,9
2320 RETURN
2500 REM **** RESET SUB COORDS ****
2510 Y3=110+INT(RND(TI)*105)
2520 POKE V+7,Y3:POKE V+6,0
2530 X3=0:DX=RND(TI)*3+1
2540 POKE V+16,0
2550 RETURN
3000 REM **** SETUP DEPTH CHARGES ****
3020 REM   ** SET FLAG **
3030 FL=1
3050 REM   ** SET COORDS **
3060 Y2=95:X2=X0
3070 POKE V+4,X2:POKE V+5,Y2
3090 REM   ** TURN ON SPRITE 2 **
3100 POKE V+21,PEEK(V+21)OR4
3110 RETURN
4000 REM **** MOVE DEPTH CHARGE ****
4020 REM   ** DECREASE Y COORD **
4030 Y2=Y2+2
4050 REM   ** TEST SEA BTM & TURN OFF **
4060 IF Y2>Y3+25 OR Y2>216 THEN POKEV+21,
PEEK(V+21)AND251:FL=0
4070 POKE V+5,Y2
4090 REM   ** TEST FOR HIT ON SUB **
4100 IF PEEK(V+30)=12 THEN GOSUB 5000:
REM HIT ROUTINE
4110 RETURN
5000 REM **** HIT ROUTINE ****
5010 POKE V+30,0:REM CLR COLLISION REG.
5020 REM   ** TURN ON EXPLOSION SPRITE **
5030 POKE V+2,X2+10:POKE V+3,Y3
5040 POKE V+21,PEEK(V+21)OR2
5060 REM   ** FLASH COLOURS **
5070 FOR I=1 TO 20
5080 FOR J=1TO 15
5090 POKE V+40,J
5100 NEXT J:NEXT I
5120 REM   ** TURN OFF SPRITES 1,2 & 3**
5130 POKE V+21,PEEK(V+21)AND241
5150 REM   ** UPDATE SCORE **
5160 DS=1:GOSUB 5500
5180 REM   ** RESET SUB COORDS & FLAG **
5190 FL=0:GOSUB 2500
5210 REM   ** TURN SUB BACK ON **
5220 POKE V+21,PEEK(V+21)OR8
5230 RETURN
5500 REM **** UPDATE SCORE ****
5510 SC=SC+INT(Y3+DX*30)*DS
5520 IF SC<0 THEN SC=0
5530 PRINT CHR$(19);CHR$(144)" SCORE";SC;
CHR$(157);"     "
5540 RETURN
6000 REM **** SHIP DATA ****
6010 DATA0,0,0,0,0,0,0,0,0
6020 DATA0,128,0,0,192,0,0,192,0
6030 DATA0,192,0,1,224,0,1,224,0
6040 DATA13,224,0,3,248,128,3,253,8
6050 DATA15,254,16,31,255,48,255,255,255
6060 DATA127,255,254,63,255,254,31,255,252
6070 DATA0,0,0,0,0,0,0,0,0
6100 REM **** EXPLOSION DATA ****
6110 DATA0,0,0,0,0,0,0,16,0,0,8,0,4,16
6120 DATA0,3,2,64,1,56,128,12,255,144
6130 DATA1,238,40,5,151,0,11,121,0,1
6140 DATA183,0,25,214,96,0,236,48,6,24
6150 DATA152,3,98,0,8,51,0,0,96,128,0
6160 DATA64,0,0,0,0,0,0,0
6170 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
6200 REM **** DEPTH CHARGES DATA ****
6210 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
6220 DATA0,0,0,32,0,0,32,0,0,32,0,0,32,0
6230 DATA0,0,0,0,0,0,0,0
6240 DATA2,0,0,2,0,0,2,0,0,2,0,0
6250 DATA0,0,0,0,0,0,0,0
6260 DATA0,0,0,0,0,0,0,0
6300 REM **** SUBMARINE DATA ****
6310 DATA0,0,0,0,0,0,0,0,0,0,0,0
6320 DATA0,8,0,0,12,0,0,12,0
6330 DATA0,12,0,0,28,0,0,60,0
6340 DATA0,126,0,199,255,255
6350 DATA239,255,255,127,255,255
6360 DATA255,255,254,199,255,254
6370 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

Hier ist das komplette Listing für unser Spiel zusammen mit einer Tabelle der verwendeten Variablen. Das Listing enthält viele REM-Anweisungen, um es leichter verständlich zu machen. Sie können diese REM-Anweisungen weglassen, wenn Sie das Programm in Ihren Computer eingeben. Nehmen wir als Beispiel an, Sie wollten die REM-Anweisung in Zeile 400 löschen. Diese Zeilennummer wird jedoch als Bestandteil einer GOTO-Anweisung in Zeile 225 verwendet. Löschen Sie Zeile 400 vollständig, so bewirkt dies beim Programmlauf die Fehlermeldung "UNDEF'D STATEMENT ERROR AT LINE 225" und der Programmlauf wird abgebrochen. Um solche Fehler zu vermeiden, ist es am besten, wenn Sie nur solche REM-Anweisungen weglassen, die am Ende einer Zeile stehen, sowie Zeilen, die lediglich Doppelpunkte (:) zur Trennung von Programmteilen enthalten.

Zinsberechnung

In der ersten Folge dieser Serie haben wir uns das Kalkulationssystem Vu-Calc für den Sinclair Spectrum und den Acorn B angesehen. Heute untersuchen wir, wie Vu-Calc die monatlichen Raten für Hypotheken und Bankkredite berechnet.

Die große Stärke auch einfacher Kalkulationsprogramme wie Vu-Calc von Psion ist die problemlose Verknüpfung von Daten und komplizierten Formeln. Obwohl Vu-Calc keine eingebauten Formeln enthält, lassen sich damit doch interessante und praktische Modelle aufbauen. Die einzige mathematische Hilfe von Vu-Calc ist die Summierung von Zellenblöcken, deren Adressen mit einem @ gekennzeichnet sind.

Hochentwickelte Systeme enthalten oft mathematische Formeln, die dem Anwender auf Abruf zur Verfügung stehen. Der Vorteil ist hierbei, daß die Formeln eingesetzt werden können, ohne daß ihre Funktionsweise bekannt sein muß. Wenn Sie mit Multiplan beispielsweise eine Hypothekenformel einsetzen wollen, die die monatlichen Raten für 15, 20 und 25 Jahre berechnet, brauchen Sie nur die Formel aufzurufen und die entsprechenden Daten einzugeben. Multiplan liefert automatisch alle gewünschten Ergebnisse.

Die gleichen Berechnungen nehmen mit Vu-Calc weitaus mehr Zeit in Anspruch. Dabei muß zunächst die Formel zusammengestellt und in die Maschine eingegeben werden. Vu-Calc gibt dem Anwender außerdem recht enge Grenzen vor. Wenn jede Spalte die Daten eines Monats enthalten soll, kann das größte Modell nur etwas mehr als zwei Jahre umfassen, da das System nur über 28 Spalten verfügt. Auch die Genauigkeit ist ein Problem – Vu-Calc arbeitet nur mit Ganzzahlen (Integer) und ignoriert alle Nachkommastellen, so daß 99,9 als 99 interpretiert wird.

Bequem ist die Möglichkeit, Werte und Formeln jederzeit in alle Felder eintragen zu können. Wenn der Cursor beispielsweise auf dem leeren Feld H5 steht und Sie in der Befehlszeile 500 * 2 eingeben, erscheint das Ergebnis – 1.000 – sofort nach dem Drücken der ENTER-Taste in H5.

Das Editieren von Formeln ist leider sehr umständlich. „Intelligente" Pakete wie Lotus 1-2-3 arbeiten mit einer Funktionstaste, die den gesamten Feldinhalt in die Befehlszeile überträgt. Zwar hat Vu-Calc einen EDIT-Befehl (#E), doch wenn Sie in einer langen Formel eine Klammer vergessen haben, gibt es keine Möglichkeit, etwas einzufügen. Da der EDIT-Befehl dem Programm nur mitteilt, daß der alte Inhalt eines Feldes gelöscht und durch einen neuen ersetzt werden soll, mussen Sie die Formel neu eingeben.

Mit dem Befehl REPLICATE (#R) lassen sich recht komplizierte Modelle aufbauen. Nehmen wir an, Sie möchten die in der vorigen Folge erwähnte Haushaltskasse erweitern, um die Auswirkungen der Inflation auf die Lebensmittelkosten zu berechnen. Sie setzen dabei eine konstante Teuerungsrate von 0,5 Prozent pro Monat voraus. Mit Papier und Bleistift sind die entsprechenden Berechnungen recht aufwendig. Mit der richtigen Formel und dem Befehl REPLICATE erledigt Vu-Calc diese Aufgabe jedoch in kürzester Zeit.

Relative und absolute Daten

Zunächst muß die Information eingegeben werden, daß der Anfangswert der Kosten (zum Beispiel 800 Mark) pro Monat um 0,5 Prozent wachsen soll. In hochentwickelten Kalkulationssystemen gibt es dafür den Befehl GROW BY. Vu-Calc verlangt jedoch die Eingabe der entsprechenden mathematischen Vorgänge. Am Anfang aller Formeln mit Feldadressen muß ein $ oder ein % stehen. Diese zwei Symbole informieren das Programm, daß in dieser Formel relative (%) oder absolute ($) Feldadressen enthalten sind. Bei absoluten Feldadressen spricht Vu-Calc den Inhalt des angegebenen Feldes an.

Sehen wir uns den Einsatz „relativer Adressen" einmal genauer an. Die Formel, die die Ausgaben um 0,5 Prozent wachsen läßt, lautet %B3*100.5/100, wobei % eine relative Feldadresse anzeigt und B3 den Anfangswert der monatlichen Lebensmittelausgaben enthält. Nachdem Sie die Formel in Feld B4 eingegeben haben, brauchen Sie sie nur zu kopieren, um den Jahreswert zu erhalten. B4 zeigt dabei den numerischen Wert der Formel an – die Formel erscheint nur dann am unteren Rand der Tabelle, wenn sich der Cursor in B4 befindet. Der Befehl REPLICATE #R,B4,B5:B14 kopiert die Formel. Die Tabelle sieht nun folgendermaßen aus:

	1	2	3	4	5
A			JAN	FEB	MAR
B	Lebensmittel				

Die Bildschirmanzeige macht deutlich, warum der Anfangswert erst in Feld B3 steht: Spalte 1 und 2 sind mit dem Wort „Lebensmittel" belegt. Beachten Sie, daß alle Werte ganzzahlig sind. Der Wert für März ist eigentlich 202,005, durch die Abrundung wird jedoch 202 angezeigt. Aus dem gleichen Grund erscheint im April statt 203,01502 die Zahl 203.

Dieses einfache Beispiel zeigt, wie REPLICATE mit relativen Feldadressen arbeitet. Bei jedem Eintrag in ein neues Feld verändert das Programm die Formel automatisch: Der Eintrag in B4 lautete %B3*100.5/100; in B5 erscheint er als %B4*100.5/100; in B6 als %B5*100.5/100 etc. Die Spaltennummern der Feldadressen werden dabei jeweils um eins erhöht. Eine Übertragung in vertikaler Richtung hat ähnliche Wirkung auf die Feldadressen (aus E1 wird F1). Werden statt relativer Adressen absolute Adressen ($) angegeben, verändern sich die Feldadressen nicht. Die Formel würde dann ohne Veränderung in alle angegebenen Felder kopiert, und der Wert jedes Feldes wäre mit dem in B4 angezeigten Wert identisch.

Das gleiche Modell läßt sich auch für die Vorausberechnung der monatlichen Kosten einer Firma für Rohmaterialien einsetzen. Nehmen wir an, die monatlichen Kosten von 100 000 Mark erhöhen sich für die Dauer von zwei Jahren um 0,5 Prozent pro Monat. Wieviel kosten die Materialien, wenn sie zur Hälfte des zweiten Jahres gekauft werden?

Ändern Sie zunächst den Wert von B3 in 100 000 um, indem Sie den Cursor in Feld B3 stellen und die neue Zahl eingeben. Übertragen Sie nun mit REPLICATE die Formel auf B14 bis B26, um zwei volle Jahre zu erhalten. Höherentwickelte Kalkulationssysteme zeigen das Ergebnis in dem Augenblick an, in dem der neue Wert eingetragen ist. Vu-Calc muß das Ergebnis (das im Augenblick noch auf der alten Formel basiert) jedoch erst mit dem Befehl CALCULATE – #C – neu berechnen. Danach steht das gewünschte Ergebnis in Feld B20. Der Betrag ist 109 931 Mark – es ergeben sich also Mehrkosten von fast 10 000 Mark. Zwar ist dieser Wert nicht völlig exakt, da er gerundet wurde, er kann jedoch immerhin einen Einblick geben, wie sich die Geldentwertung in diesem Zeitraum auswirkt.

Berechnung der Raten

Unser letztes Beispiel enthält eine kompliziertere Formel. Nehmen Sie an, Sie möchten 1000 Mark Bankschulden, auf die Ihnen 27 Prozent Zinsen pro Jahr berechnet werden, mit 80 Mark pro Monat zurückzahlen. Wann haben Sie den gesamten Betrag – mit Zinsen – ausgeglichen? Für diese Information werden zu der Hauptsumme des Kredites die monatlichen Zinsen hinzugezählt und davon die monatlichen Zahlungen abgezogen. Wenn B1 die Hauptsumme von 1000 Mark enthält, lautet die Formel %B1+%B1*.27/12–80. Übertragen Sie diese Formel auf alle 28 Spalten des Modells und sehen Sie nach, in welchem Feld die Summe ins Positive übergeht. Zu diesem Zeitpunkt ist der Kredit zurückgezahlt. In unserem Modell würde die Rückzahlung 16 Monate in Anspruch nehmen.

In der nächsten Folge werden wir den Abacus – das Kalkulationsprogramm für den Sinclair QL – genauer untersuchen.

Relativ – Absolut

In diesem einfachen Modell wurden Formeln mit dem Befehl REPLICATE in Felder der Zeilen C, D und E übertragen. Da Feld C3 mit absoluten Feldadressen versehen ist, erscheint die gleiche Formel – A5/12 – und auch das gleiche Ergebnis – 450 – in allen Feldern der Reihe C. Die Formeln der Felder D4 und E4 enthalten jedoch relative Feldadressen. Der Bezug auf die Adressen ändert sich in einer Reihe von einem Feld zum nächsten, und es werden unterschiedliche Ergebnisse dargestellt.

	1	2	3	4	5
A		Jahreseinkommen =			5400
B			JAN	FEB	MAR
C	Einkommen		A5/12 450	A5/12 450	A5/12 450
D	Ausgaben		DATA 400	D3*1.01 404	D4*1.01 408
E	Monatssaldo		C3-D3 50	C4-D4 46	C5-D5 42

Das Erfolgsteam

Fast jede Woche präsentiert sich auf dem Markt ein neues, auf Spielprogramme spezialisiertes Softwarehaus. Doch nur wenige dieser Senkrechtstarter haben Bestand oder landen gar Bestseller. Eine der Ausnahmen ist die 1981 gegründete Firma Quicksilva.

Die Produktion des Sinclair ZX 80 war ausschlaggebend für die Entstehung von Quicksilva. Der Erfolg des Rechners ermutigte den Ingenieur Nick Lambert, ein Drei-KByte-Erweiterungsmodul für den mit einem einzigen KByte ausgestatteten ZX 80 zu entwickeln. Er verkaufte es recht erfolgreich im Versand. Nachdem Sinclair den ZX 81 herausgebracht hatte, gründete Lambert gemeinsam mit John Hollis und Mark Eyles Quicksilva, um Erweiterungen für den neuen Rechner zu produzieren. 1981 war auch das Jahr, in dem „Defender" auf den Markt kam, ein von Lambert geschriebenes Spiel, mit dem Quicksilva sich erstmals im Software-Markt versuchte.

Der Erfolg von Defender veranlaßte Quicksilva zur Entwicklung weiterer Programme im Spielhallenstil, und man verzichtete auf weitere Hardware-Entwicklungen.

Der zweite große Software-Erfolg des Unternehmens trug den Titel „Timegate". Diesem Adventure folgten weitere, so daß Weihnachten 1982 bereits zehn Quicksilva-Programme auf dem Markt waren. Die Nachfrage nach Quicksilva-Produkten stieg dank der Qualität und des Namens schlagartig an. Nach einem Start mit rund 800 Mark Kredit konnte die Gesellschaft in ihrem ersten Jahr einen stolzen Umsatz von 250 000 Mark vorweisen. Heute sind Quicksilva-Produkte in allen großen Warenhäusern und bei zahlreichen Einzelhändlern zu finden.

Eine derartige Entwicklung hat natürlich zur Folge, daß drei Mitarbeiter zuwenig sind. Heute sucht Quicksilva mit Anzeigen in der Computerpresse nach Programmautoren, und die Programmschreiber haben die Chance, bis zu 15 Prozent aus Lizenzerträgen von jeder verkauften Cassette zu verdienen. Inzwischen produziert das Unternehmen auch Programme für den Acorn B, den Dragon, den Commodore 64 und den VC 20.

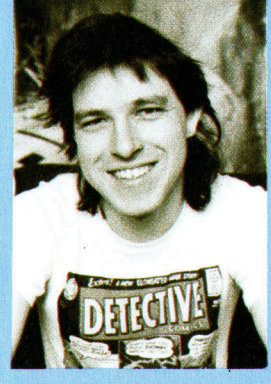
Perfekte Grafik: Ant Attack

Quicksilva gilt derzeit als eines der führenden britischen Software-Häuser. Mit „Ant Attack" wurde der Name noch bekannter. Dieses Spiel mit überragenden grafischen Elementen haben wir bereits vorgestellt. Rod Cousens, der die Position des Managing Director übernahm, nachdem Lambert sich entschlossen hatte, wieder auf dem kreativen Sektor zu arbeiten, wurde zum Vizepräsidenten der „Guild of Software Houses". 1983 wurde er von der Computer Trade Association zur „Persönlichkeit des Jahres" gewählt.

Quicksilva ist weiterhin auf der Suche nach neuen Produktbereichen, die das Angebot vergrößern. Unlängst wurde ein „gewaltloses" Spiel mit dem Titel „The Snowman" veröffentlicht, das auf einer Erzählung für Kinder von Raymond Briggs basiert. Es wird als willkommenes Gegenstück zu den üblichen „Ballerspielen" betrachtet. Und auch auf andere Weise ist das Unternehmen Vorreiter: Man erhofft sich schon bald eine starke Position im nordamerikanischen Software-Markt.

Hier sind einige der neueren Veröffentlichungen von Quicksilva. Die Firma produziert Programme für viele Systeme.

Rod Cousens, geschäftsführender Direktor von Quicksilva



Mark Eyles, Anzeigenleiter und einer der Gründer des Unternehmens.

Stark verbessert

Fehlende Schnittstellen für Diskettenstationen mögen Ursache für den geringen Verkauf des ansonsten ausgezeichneten Memotech 500 gewesen sein. Jüngst stellte das Unternehmen den RS128 mit vollen Interfacemöglichkeiten vor.

Obwohl die beiden Rechner der Memotech-500-Computerserie, der MTX 500 und der MTX 512, sehr gelobt wurden, fanden sie bei Heimcomputer-Käufern kaum Beachtung. Das lag sicher nicht an ihren ausgezeichneten Eigenschaften – wie der hochauflösenden Grafik, eingebautem Assembler, ausgefeiltem BASIC und einer speziellen Textverarbeitungssprache namens NODDY. Die Gründe sind eher darin zu suchen, daß die Rechner in ihren Eigenschaften nicht auf eine bestimmte Geräteklasse abgestimmt wurden.

Mit einem Preis von rund 1000 Mark sind sie für den Nur-Spieler zu teuer, dem die Besonderheiten der Rechner nicht den hohen Preis wert sind. Der „ernsthafte" Anwender (Memotech hatte diese Serie ursprünglich für Kleinunternehmer entwickelt) vermißt eingebaute Schnittstellen, die die Verbindung mit Diskettenstationen erst möglich machen. Es gab sie zwar, doch wurden sie auf separaten Platinen geliefert, die in den Computer eingesetzt werden mußten. Für ein Unternehmen, das sich mit der Entwicklung von Erweiterungen für den ZX81 einen Namen machte, ist das nicht weiter verwunderlich. Den Anwender dagegen stört so etwas, wenn er einen Rechner haben will, der einfach eingeschaltet wird und dann läuft. Memotech scheint mittlerweile das Problem erkannt zu haben und baut nun den RS128 mit eingebauten Interfaces.

Auf den ersten Blick stimmt der RS128 mit der 500er-Serie optisch überein. Wie seine Vorgänger verfügt er über ein Aluminiumgehäuse statt der sonst üblichen „Plastikverpakkung". Dadurch ist der Memotech erheblich schwerer als andere Micros. Eine Standard-QWERTY-Tastatur und eine numerische Tastatur stehen zur Verfügung, auf die auch einige Befehle der NODDY-Textprogrammiersprache gelegt sind. Rechts neben der Tastatur finden sich weitere acht programmierbare Funktionstasten. Die Tasten sind griffig und wirken sehr professionell.

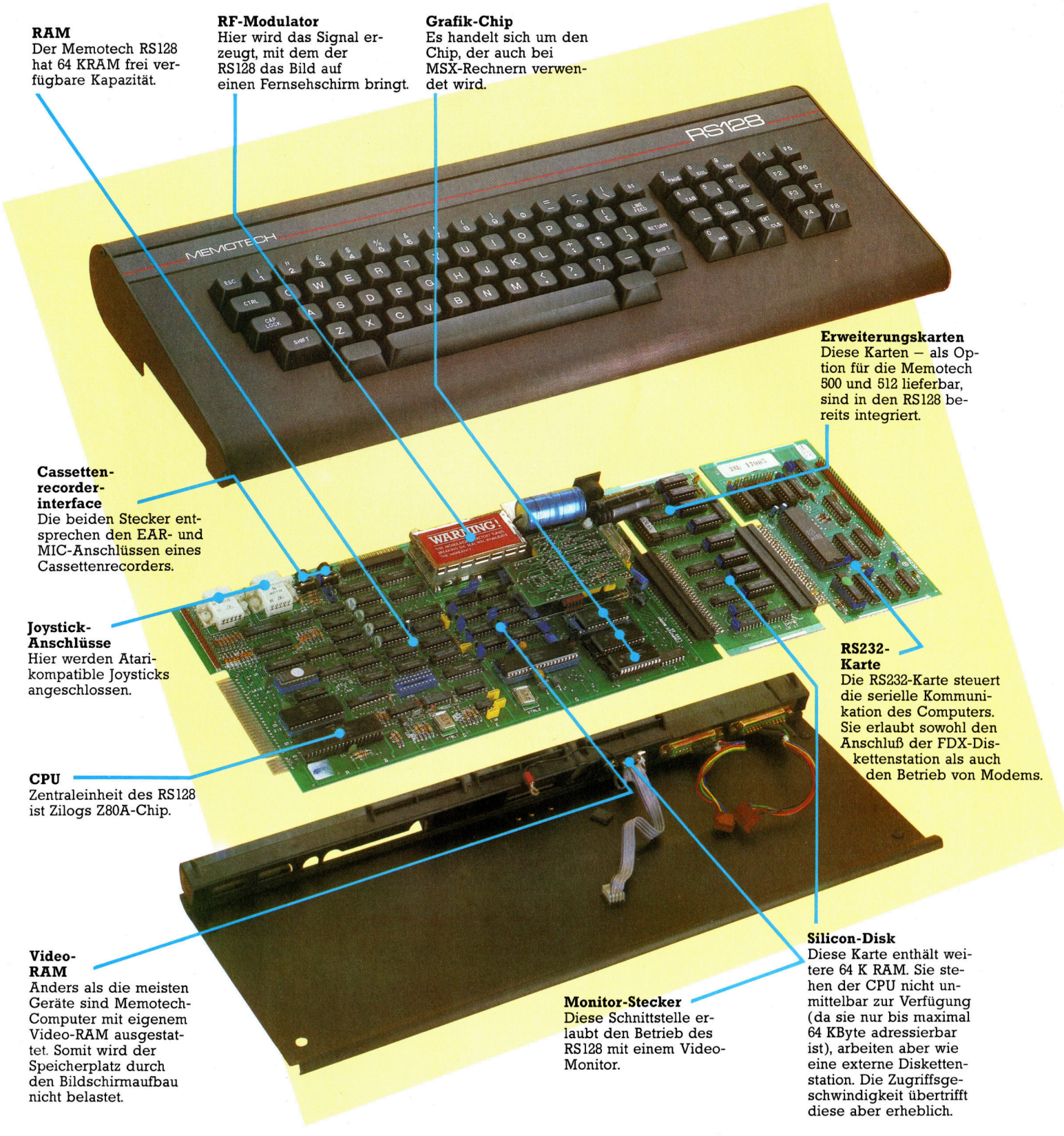
Kritik gibt es dabei jedoch auch: So ist die RETURN-Taste kaum größer als die anderen Tasten, und Leute, die blind schreiben, werden zunächst Schwierigkeiten haben, diese Taste schnell zu finden. Die DELETE-Taste ist nicht Bestandteil der Schreibmaschinentastatur, sondern in die numerische Tastatur integriert. In der oberen rechten Ecke des Feldes liegt die BACKSPACE-Taste. Anders als bei den meisten Computer-Tastaturen, bei denen die Backspace-Taste zugleich die Funktion des Löschen-nach-links (bekannt als „destruktives Backspace") hat, dient sie hier lediglich zur Steuerung des Cursors nach links.

Der Memotech RS128 ist eine verbesserte Version der MTX-500er-Serie. Das neue Modell wurde mit zwei RS232-Schnittstellen ausgestattet, die den Anschluß von FDX-Diskettenstationen erlauben. Damit ist der Rechner für den ernsthaften Heimcomputeranwender oder den Gewerbetreibenden sehr interessant.

Stereoton möglich

Auf der Rückseite des Gehäuses befinden sich mehrere Schnittstellen. Ganz links außen liegen zwei RS232-Ports, die die Verbindung des Rechners mit FDX-Diskettenstationen ermöglichen. Sie dienen zugleich als Anschluß für beispielsweise serielle Drucker und Netzwerk-Kommunikation. Rechts daneben liegen ein Composite-Video- und ein HiFi-Stecker. Mit letzterem kann der Ton des Computers über ein normales Stereosystem ausgegeben werden. Darauf folgen Stromanschluß und RF-Stecker sowie das Centronics-Druckerinterface. Das Cassettenrecorder-Interface besteht aus zwei Buchsen für EAR (Ohr) und MIC (Mikrofon). Schließlich liegen dort neunpolige Joystick-Anschlüsse für Atari-kompatible Sticks.

Die Interfaces sind weiß beschriftet, so daß man ihre Funktionen leicht ablesen kann. Memotech hat die Anschlüsse allerdings in Ver-



RAM
Der Memotech RS128 hat 64 KRAM frei verfügbare Kapazität.

RF-Modulator
Hier wird das Signal erzeugt, mit dem der RS128 das Bild auf einen Fernsehschirm bringt.

Grafik-Chip
Es handelt sich um den Chip, der auch bei MSX-Rechnern verwendet wird.

Erweiterungskarten
Diese Karten – als Option für die Memotech 500 und 512 lieferbar, sind in den RS128 bereits integriert.

Cassettenrecorder-interface
Die beiden Stecker entsprechen den EAR- und MIC-Anschlüssen eines Cassettenrecorders.

Joystick-Anschlüsse
Hier werden Atarikompatible Joysticks angeschlossen.

RS232-Karte
Die RS232-Karte steuert die serielle Kommunikation des Computers. Sie erlaubt sowohl den Anschluß der FDX-Diskettenstation als auch den Betrieb von Modems.

CPU
Zentraleinheit des RS128 ist Zilogs Z80A-Chip.

Video-RAM
Anders als die meisten Geräte sind Memotech-Computer mit eigenem Video-RAM ausgestattet. Somit wird der Speicherplatz durch den Bildschirmaufbau nicht belastet.

Monitor-Stecker
Diese Schnittstelle erlaubt den Betrieb des RS128 mit einem Video-Monitor.

Silicon-Disk
Diese Karte enthält weitere 64 K RAM. Sie stehen der CPU nicht unmittelbar zur Verfügung (da sie nur bis maximal 64 KByte adressierbar ist), arbeiten aber wie eine externe Diskettenstation. Die Zugriffsgeschwindigkeit übertrifft diese aber erheblich.

tiefungen gelegt, so daß man sich etwas verrenken muß, um Peripheriegeräte an der Rückseite des Rechners anzuschließen.

Der BASIC-Screen (24 mal 40 Zeichen) ist dreigeteilt. Die oberen 19 Reihen stellen den Hauptbildschirmteil dar, auf dem Programm-Listings gescrollt werden. Unter diesem Bildschirm befindet sich der EDIT-Teil, in den neue Zeilen eingegeben werden. Und ganz unten stellt eine einzelne Zeile Fehlermeldungen dar. Wie bei den Sinclair-Rechnern werden Programmzeilen durch Verwendung des EDIT-Befehls geändert. Das Betriebssystem läßt nicht zu, daß eine Zeile vom EDIT-Screen eingefügt wird, solange diese einen Syntax-Fehler enthält.

Das BASIC selbst ist dem MSX-BASIC eng verwandt und enthält Befehle wie SOUND, PA-

PER, INK und CIRCLE. Allerdings enthält dieses BASIC auch einige sehr nützliche Befehle, die es im MSX-BASIC nicht gibt. Der Befehl CSR x,y positioniert den Cursor auf den Bildschirmpunkt mit den Koordinaten (x,y). Ein weiterer sinnvoller Befehl ist CRVS, der dem Anwender erlaubt, ein Fenster an beliebiger Stelle des Bildschirms zu definieren. Text und Grafik können in solchen Fenstern dargestellt werden.

Als Zentraleinheit dient der Z80A, durch den der Rechner CP/M-fähig ist. Viele kleinere Computerhersteller entschieden sich deshalb für den Z80, da das Problem der eigenen umfangreichen Softwareherstellung umgangen werden kann und der Anwender sofort über eine große Softwarebibliothek verfügt. Voraussetzung für eine optimale CP/M-Nutzung ist eine 80-Zeichen-Darstellung. Ungewöhnlich ist, daß Memotech die 80-Zeichen-Karte in der Diskettenstation untergebracht hat.

Software-Paket gratis

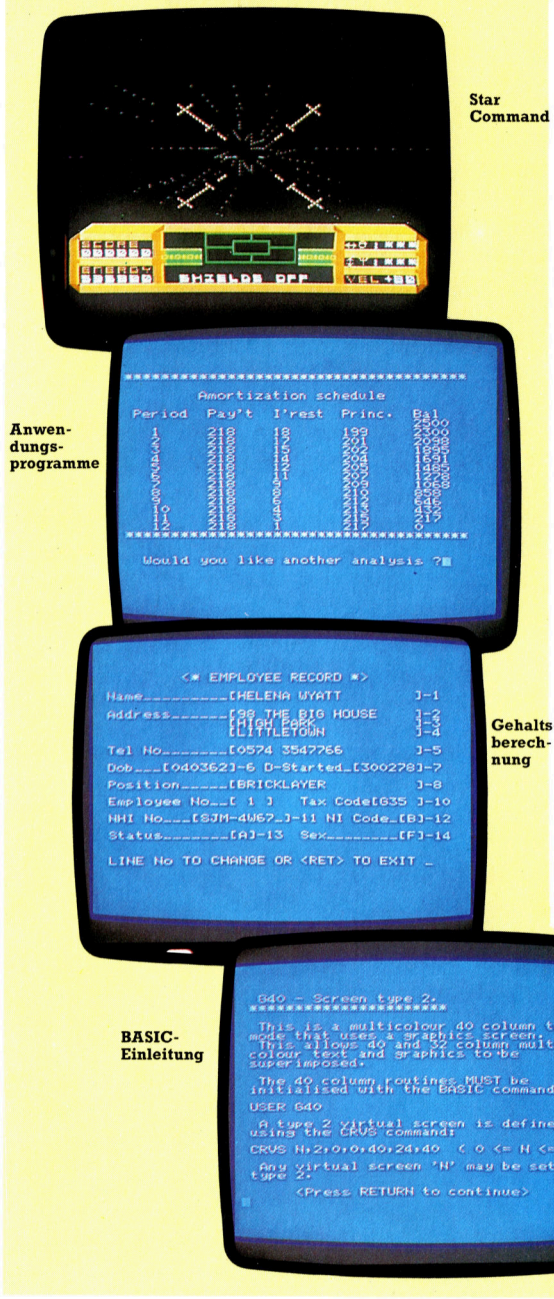
Mit der Floppy wird ein Software-Paket geliefert. Neben der CP/M-System-Diskette bekommt man die „New Word"-Textverarbeitung, das „SuperCalc"-Spreadsheet, „Compact" und „Televideo" – womit man die Floppy Disketten, die in anderen Formaten geschrieben sind, lesen kann. Dazu gehören laut Memotech auch IBM-Disketten. Und schließlich „Contact", mit dem der zweite RS232-Anschluß an ein Netzwerk gehängt werden kann.

Der RS128 verfügt über 128 KByte RAM. Da aber ein Acht-Bit-Prozessor verwendet wird, mit dem nur 64 KByte adressierbar sind, stehen die restlichen 64 KByte als „Silicon Disk" zur Verfügung. Diese Silizium-Diskette speichert Dateien und Programme genauso wie eine normale Diskette – mit dem Unterschied, daß die Informationen auf Chips abgelegt sind und so bis zu 50mal schneller verarbeitet werden können als die von herkömmlichen Floppy-Disks. Die auf der Diskette gespeicherte Information wird in adressierbares RAM umgewandelt. Ist die Arbeit beendet, können die Daten dauerhaft auf Floppy-Disk gespeichert werden.

Das mitgelieferte Handbuch hat weit mehr Seiten als sonst bei Heimcomputern üblich. Allerdings liegt dies nur daran, daß es nicht richtig gesetzt, sondern getippt wurde. Es enthält nur wenige weiterführende Informationen. Memotech hat aber alle für den Anwender nötigen technischen Einzelheiten dargestellt, einschließlich der Schaltpläne, Pinbelegungen und Aufrufe für das Betriebssystem.

Mit der Weiterentwicklung der Memotech-500-Serie in Richtung RS128 versucht das Unternehmen, einen Standard-Business-Rechner zu etablieren. Es bleibt allerdings abzuwarten, ob sich die Maschine gegen ihre Konkurrenten behaupten kann.

FDX-Doppel-Disk-Drive
Mit diesem FDX-Doppel-Disk-Drive kann der Computer das CP/M-Betriebssystem verarbeiten. Die Speicherkapazität beträgt 500 KByte.



Memotech RS 128

ABMESSUNGEN
488x202x56 mm

ZENTRALEINHEIT
Z80A mit 4 MHz

SPEICHER-KAPAZITÄT
64 K RAM und 24 K ROM

BILDSCHIRM-DARSTELLUNG
40x24 im Textmodus; Text mit Grafik-Mode: 32x24 Text und 256x192 Punkte in 16 Farben; bis zu 32 unabhängig voneinander kontrollierbare Sprites.

SCHNITTSTELLEN
Cassette (MIC und EAR), I/O-Interface, zwei Joystick-Anschlüsse, zwei RS232-Anschlüsse, HiFi-Buchse, Composite-Video-Buchse, TV-Buchse, Parallel-Schnittstelle.

SPRACHEN
BASIC, FORTH, PASCAL

TASTATUR
57 Schreibmaschinentasten.

HANDBÜCHER
Das Handbuch entspricht dem der MTX-500er-Serie.

STÄRKEN
Durch die RS232-Karten ist der RS128 für ernsthafte Heimanwender und den kleinen Gewerbebetrieb sehr interessant.

SCHWÄCHEN
Kaum Programme.

Software-Unterstützung

Derzeit gibt es eine Reihe von Geschäfts- und Spielprogrammen für Memotech-Rechner. Im Vergleich zu anderen Systemen ist die Zahl der erhältlichen Programme begrenzt. Doch die Verarbeitung von CP/M-Software löst dieses Problem sicher bald.

Krieg der Sterne

„Star Raiders“ ist eine Weiterentwicklung des populären Computerspiels „Star Trek“. Im Gegensatz zu den anderen Spielen der Atari-Klassiker-Serie wurde es speziell für Heimcomputer entwickelt und bietet spannende Unterhaltung.

Begegnung mit den Zylonen

Galaktische Karte

Der Spieler übernimmt die Rolle des Kommandanten des Raumschiffs Star Raider und durchquert die Galaxis, um feindliche Raumschiffe der Zylonen zu verfolgen. Die Spielsteuerung erfolgt über Joystick und Tastatur, und durch Druck auf die „F“-Taste wird der Blick aus dem Cockpit freigegeben. Die Position des Zylonen-Schiffs wird durch Maßeinheiten am unteren Bildschirmrand angezeigt. Mit der „L“-Taste aktiviert man den „Long range Scanner“ – eine Art Radar, der einen Überblick der nächsten Umgebung rund um das Raumschiff gibt.

Die Aufgabe des Spielers besteht darin, sich dem Feind zu nähern, entweder mit Normaltriebwerken, was die Gefahr birgt, daß die Zylonen entkommen, oder mit „Hyperspace“ (diesen Raumsprung leitet man durch Druck auf die Taste „H“ ein). Vor Aktivierung der Hypergeschwindigkeit muß der Suchcomputer aktiviert werden, was mit der Taste „T“ erfolgt. Geschieht dies nicht, kann es sein, daß das Raumschiff in einem völlig unbekannten Sektor der Galaxis wieder auftaucht. Andere dabei zu berücksichtigende Faktoren sind die Anwendung des Angriffscomputers (der über „C“ aufgerufen wird) sowie die Schutzschilde des Star Raider, die durch Druck auf die Taste „S“ aktiviert bzw. ausgeschaltet werden.

Hat man den Hyperraum wieder verlassen, blinkt ein rotes Alarmlicht und der Kampf beginnt. Die Zylonen greifen den Star Raider nun von allen Seiten an. Durch Joysticksteuerung kann der Spieler sein Raumfahrzeug in alle Richtungen bewegen. Die Geschwindigkeit wird über die numerischen Tasten reguliert.

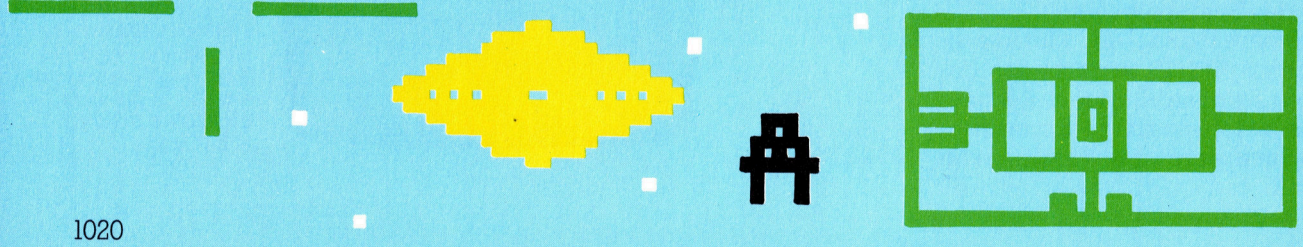
Diese Kampftechnik erfordert aber viel Treibstoff, zumal der Star Raider dabei häufig getroffen wird, was ein Aufsuchen der nächstgelegenen Raumstation erforderlich macht, um aufzutanken und Reparaturen durchführen zu lassen. Der Spieler muß sich dazu in ein Gitter bewegen, in dem sich ein Stern befindet. Während der Star Raider näher kommt, wird der Stern langsam als gelbe große fliegende Untertasse erkennbar. Das schwerste Manöver im Spiel folgt nun. Es geht darum, das Raumfahrzeug in die richtige Umlaufbahn zur Station zu bekommen. Dazu muß die Raumstation ins Zielkreuz des Raumschiffs gebracht und dort gehalten werden. Der Spieler muß die Schiffsgeschwindigkeit so lange verlangsamen, bis der Entfernungsmesser Null anzeigt. Ist dies geschehen, kann der Star Raider gestoppt werden, und am oberen Bildschirmrand leuchtet die Information "ORBIT ESTABLISHED" auf. Dieses Manöver ist sehr sorgfältig durchzuführen, da man leicht über das Ziel hinausschießen kann. Befindet man sich in der Umlaufbahn, startet ein Tankraumschiff von der Basis und dockt am Star Raider an. Danach ist das Schiff wieder einsatzfähig.

Sehr häufig wird die Meldung "STARBASE SURROUNDED" angezeigt. Das bedeutet für den Spieler, daß er schnellstmöglich dorthin gelangen muß, um die Zerstörung der eingekreisten Station zu verhindern.

Star Raider hat vier verschiedene Schwierigkeitsgrade, die von „Novice“ bis „Commander“ reichen. In den unteren Schwierigkeitsgraden muß der Spieler keine Angst vor größeren Schäden am Star Raider haben, da es dabei nur wenige Zylonen-Schiffe gibt, die auch nicht sehr genau schießen. In den höheren Ebenen ist es äußerst schwierig, auch nur wenige Spielminuten zu überstehen.

1980 wurde Star Raiders in den USA als „Spiel des Jahres“ ausgezeichnet. Es ist leider nur für Atari-Rechner am Markt.

Star Raiders: Für alle Atari-Computer
Hersteller/Vertrieb: Atari Corp. (Deuschland) GmbH
Programm: Atari
Joystick: erforderlich
Format: Cartridge



Gutes Gedächtnis

Der Sinclair Spectrum wurde anfangs wegen des fehlenden Massenspeichers für ernsthafte Anwendungen kaum in Betracht gezogen. Inzwischen wurde neben dem Sinclair Microdrive das Wafadrive von Rotronics entwickelt.

Der Sinclair Spectrum wird vielerorts als reiner Spielcomputer ohne Chancen bei „ernsthaften" Anwendungen angesehen. Schuld daran ist größtenteils die Tastatur, die nicht gerade zu ausgiebigem Gebrauch bei der Textverarbeitung oder Datenbankorganisation einlädt. Hier bemüht sich Sinclair allerdings um Abhilfe und bietet mittlerweile den Spectrum+ mit Schreibmaschinentasten an.

Damit ist das Problem aber noch nicht gelöst. Andere Vorwürfe, mit denen die Spectrum-Verteidiger zu kämpfen haben, betreffen das Fehlen von normgerechten Schnittstellen und mehr noch von schnellen, zuverlässigen Massenspeichern, die für den kommerziellen Einsatz und für den ernsthaften Heimgebrauch unerläßlich sind. Mit der Einführung von Interface 1 und Microdrive wurden zwar diese Kritikpunkte beseitigt, jedoch lassen Geschwindigkeit und Verläßlichkeit des Microdrive immer noch zu wünschen übrig. Außerdem ist bisher nur sehr wenig vom umfangreichen Programmangebot für den Spectrum auch auf Microdrive-Cassetten lieferbar. In solchen Fällen liegt es für Fremdhersteller nahe, hierfür eigene Produkte zu entwickeln. In diesem Artikel wird zunächst das Wafadrive von Rotronics vorgestellt. In einem der folgenden Hefte besprechen wir das Discovery 1 der Firma Opus Supplies.

Anders als das Interface 1 mit den Microdrives ist das Wafadrive ein Kompaktgerät – Schnittstellen und Laufwerke stecken in einem gemeinsamen Gehäuse. Das erspart gegenüber dem Microdrive zwar einigen Kabelsalat, aber andererseits geht auch Flexibilität verloren, weil ein Anschluß weiterer Einheiten zur Kapazitätserhöhung hier unmöglich ist.

Das Wafadrive befindet sich in einem schwarzen Plastikgehäuse, aus dem ein 54poliges Flachbandkabel herausragt, das an die Peripherie-Steckleiste des Spectrum angeschlossen werden muß. An der Vorderseite sehen Sie die beiden Schlitze für die Wafer-Cartridges, dazwischen drei Leuchtdioden (eine als Betriebsspannungskontrolle und die anderen zur Bandlaufanzeige).

Hinten links ist ein Bus-Erweiterungsstecker für das Interface 2 eingebaut, daneben ein Centronics-kompatibler Anschluß für einen Drucker mit Parallelschnittstelle und hinten rechts noch ein Stecker für Modems oder andere seriell arbeitende Geräte nach RS232-Norm. Damit haben Sie mehr Einsatzmöglichkeiten als beim Interface 1, bei dem Sie beispielsweise für den Betrieb eines Paralleldruckers eine zusätzliche Centronics-Schnittstelle anschließen müssen. Leider ist es zur Zeit noch schwierig, einen Centronics-Drucker oder ein Modem zu finden, dessen Stecker in das Wafadrive paßt.

Höhere Lebensdauer

Die speziellen Wafadrive-Stringy-Floppies haben große Ähnlichkeit mit den beim Microdrive verwendeten. Jede Cartridge enthält eine Endlosschleife aus 1,8 mm breitem Magnetband, das gegenüber dem normalen Toncassettenband eine höhere Lebensdauer und bessere Speichereigenschaften besitzt. Nach dem Formatieren kann das Band etwa 128 KByte an Daten aufnehmen. Daneben liefert Rotronics auch 64- und 16-KByte-Cartridges.

Einschließlich Schutzhülle sind die Cartridges etwa so lang und breit wie die Microdrive-Cassetten, aber ungefähr doppelt so dick. Bei den Wafadrive-Cartridges ist die Hülle entbehrlich, weil das empfindliche Bandmaterial durch eine selbstschließende Abdeckung ge-

Anders als beim Microdrive-System von Sinclair sind beim Wafadrive von Rotronics die Schnittstellen (RS232 und Centronics) und zwei Laufwerke in einem Gehäuse untergebracht. Das Flachbandkabel wird mit der rückwärtigen Peripherie-Steckleiste des Spectrum verbunden.

gen Berührung geschützt ist, ähnlich wie bei den 3½-Zoll-Microfloppies von Sony (allerdings verwendet Rotronics einen Plastik- statt eines Metallschutzes). An der linken Seite der Cartridge befindet sich eine Schreibsperre, deren Abbrechen ein neuerliches Beschreiben unmöglich macht – es sei denn, der Benutzer überlistet die Abtastmechanik durch kleine Tricks.

Die Wafadrive-Befehle sind ähnlich wie die des Microdrive. In beiden Fällen gehört hinter jeden Befehl ein „*", um anzuzeigen, daß der externe Speicher angesprochen ist – etwa in der Form SAVE*, LOAD* und VERIFY*. Trotzdem gibt es einige Unterschiede, weil beim Wafadrive anders als beim Microdrive nicht mehr als zwei Laufwerke angeschlossen werden können. Zum Formatieren dient bei den Microdrives der Befehl FORMAT* 'm'; 0; „name", wobei sich "m"; 0 auf das gewünschte Laufwerk bezieht. Bei Benutzung des Wafadrive heißt es statt dessen nur FORMAT*'a: name'. Dabei bezeichnet a: (alternativ b:) das aufgerufene Laufwerk, denn hier gibt es nur zwei Möglichkeiten, während beim Microdrive zum Ansprechen der Laufwerke die Ziffern 0–7 in Frage kommen.

Das Wafadrive macht wie das Microdrive davon Gebrauch, daß beim Sinclair Spectrum softwaremäßig 16 „Streams" oder Kanäle für die Ein- und Ausgabe von Daten eingerichtet werden können. Einige Kanäle sind für Bildschirm und Drucker reserviert, aber Nummer vier bis 15 stehen für andere Peripheriegeräte zur Verfügung. Ausgabekanäle für das Wafadrive werden mit OPEN# eröffnet. Das Wafadrive selbst besitzt noch zwei zusätzliche Kanäle. Es sind r und c (wahlweise auch Großbuchstaben) für die RS232- bzw. die Centronics-Schnittstelle, die ähnlich wie die Kanäle t und b beim Programmieren der RS232-Schnittstelle des Interface 1 angesprochen werden.

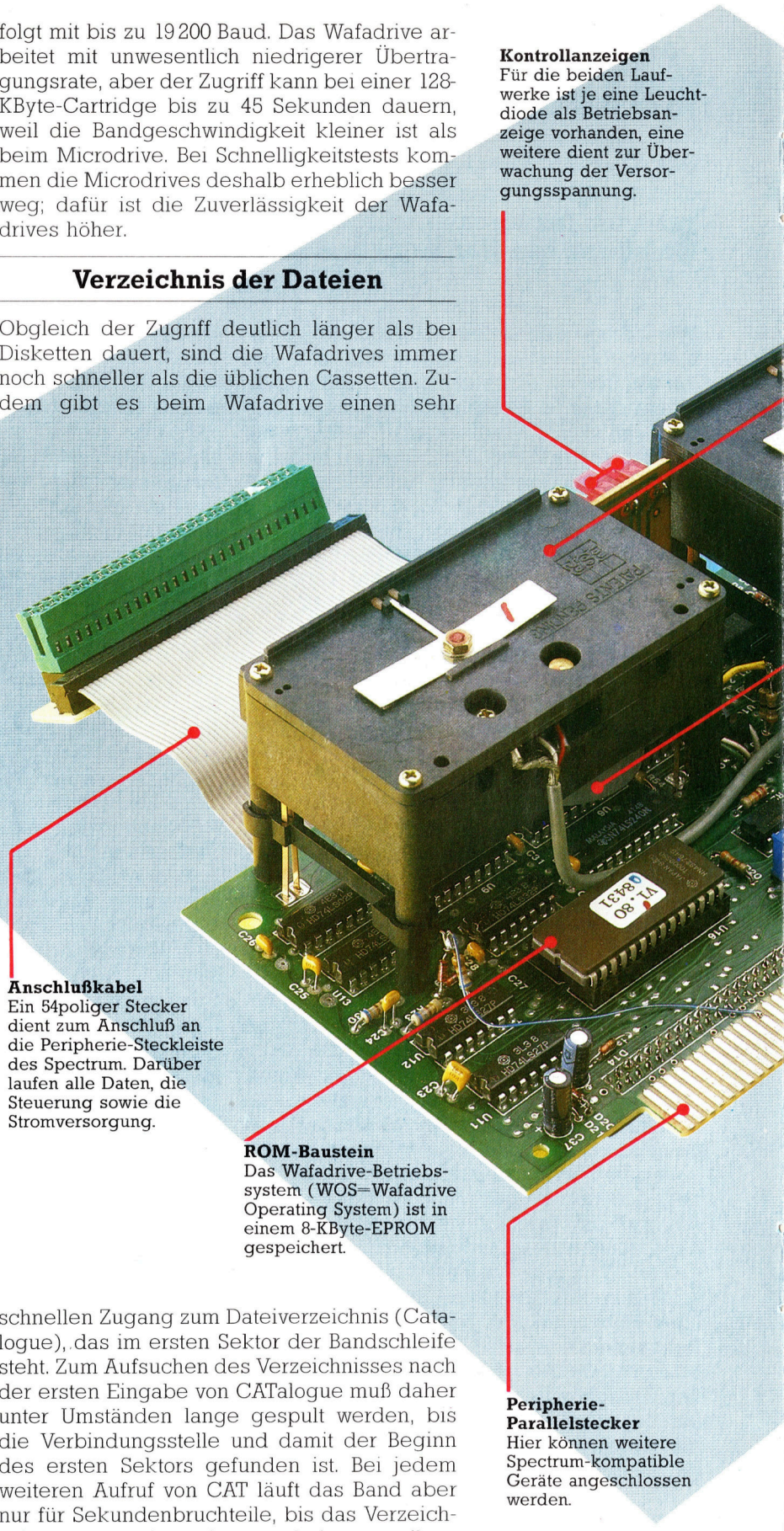
Wafadrive Operating System

Der erweiterte BASIC-Befehlssatz für die Systemsteuerung ist als „Wafadrive Operating System" (WOS) in einem 8-KByte-ROM untergebracht. Das WOS arbeitet mittels Überlagerung der untersten acht KByte des Spectrum-ROMs (weitgehend wie beim Interface 1). Der Befehl LOAD* beim Spectrum führt beispielsweise zunächst zum Aufruf der Fehlerroutine durch den BASIC-Interpreter. Das WOS fängt diesen Aufruf aber ab und ruft das Wafadrive-ROM auf, das nun seinerseits die Behandlung des Fehlers übernimmt und LOAD* als Wafadrive-Kommando interpretiert.

Das Wafadrive ist etwas langsamer als ein Microdrive. Das Aufsuchen einer bestimmten Information erfordert bei einer 100-KByte-Microdrive-Cassette im Schnitt 3,5 Sekunden, und die Datenübertragung zum Rechner erfolgt mit bis zu 19200 Baud. Das Wafadrive arbeitet mit unwesentlich niedrigerer Übertragungsrate, aber der Zugriff kann bei einer 128-KByte-Cartridge bis zu 45 Sekunden dauern, weil die Bandgeschwindigkeit kleiner ist als beim Microdrive. Bei Schnelligkeitstests kommen die Microdrives deshalb erheblich besser weg; dafür ist die Zuverlässigkeit der Wafadrives höher.

Verzeichnis der Dateien

Obgleich der Zugriff deutlich länger als bei Disketten dauert, sind die Wafadrives immer noch schneller als die üblichen Cassetten. Zudem gibt es beim Wafadrive einen sehr schnellen Zugang zum Dateiverzeichnis (Catalogue), das im ersten Sektor der Bandschleife steht. Zum Aufsuchen des Verzeichnisses nach der ersten Eingabe von CATalogue muß daher unter Umständen lange gespult werden, bis die Verbindungsstelle und damit der Beginn des ersten Sektors gefunden ist. Bei jedem weiteren Aufruf von CAT läuft das Band aber nur für Sekundenbruchteile, bis das Verzeichnis auf dem Bildschirm erscheint – weil es nämlich nach dem ersten Aufruf im RAM abgelegt worden ist. Das WOS wartet daher bei er-

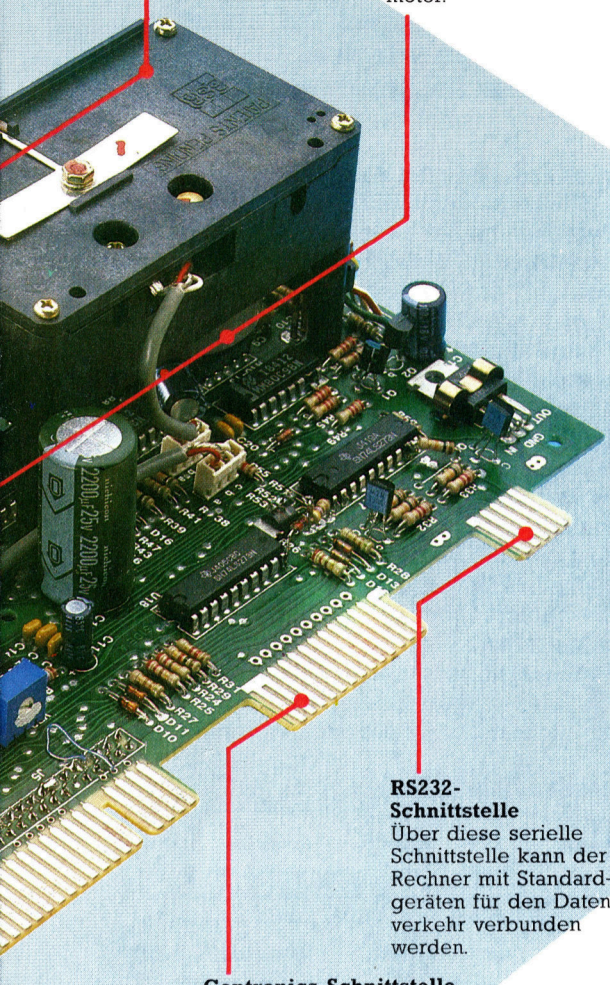


Kontrollanzeigen
Für die beiden Laufwerke ist je eine Leuchtdiode als Betriebsanzeige vorhanden, eine weitere dient zur Überwachung der Versorgungsspannung.

Anschlußkabel
Ein 54poliger Stecker dient zum Anschluß an die Peripherie-Steckleiste des Spectrum. Darüber laufen alle Daten, die Steuerung sowie die Stromversorgung.

ROM-Baustein
Das Wafadrive-Betriebssystem (WOS=Wafadrive Operating System) ist in einem 8-KByte-EPROM gespeichert.

Peripherie-Parallelstecker
Hier können weitere Spectrum-kompatible Geräte angeschlossen werden.



Bandlaufwerke
Das Wafadrive enthält zwei Laufwerke, die in getrennten Gehäusen nebeneinander angeordnet sind. Rotronics liefert dafür 16-, 64- und 128-KByte-Cartridges.

Motoren
Jedes Laufwerk hat einen eigenen Bandantriebsmotor.

RS232-Schnittstelle
Über diese serielle Schnittstelle kann der Rechner mit Standardgeräten für den Datenverkehr verbunden werden.

Centronics-Schnittstelle
Das Centronics-Interface dient zum Anschluß eines Druckers.

neutem Aufruf nur das Kennfeld des nächsten Sektors ab, um zu überprüfen, ob noch die gleiche Cartridge eingeschoben ist und listet dann sofort das Verzeichnis aus dem RAM.

Im Bemühen, dieses Massenspeicher-Konzept für ernsthafte Anwendungen auf dem Markt zu etablieren, wird der Hardware von Rotronics ein ziemlich umfangreiches Textverarbeitungsprogramm (Spectral Writer) beigepackt, das das Wafadrive voll ausnutzt. Funktionen wie das Umstellen von Abschnitten, Einfügen von Wörtern und Löschen von Zeilen werden durch Drücken von „Symbol Shift" in Verbindung mit einer weiteren Rechnertaste aufgerufen. Andere Routinen für das Hantieren mit Wafadrive-Dateien wie das Abspeichern (SAVE) und Laden (LOAD) von Text-Files werden über Sonderbefehle zugänglich. Der Spectral Writer ist ein sehr brauchbares Textverarbeitungspaket, obwohl sich am Bildschirm nicht die Zeilenlänge vorgeben läßt.

Kaum Programme auf Wafas

Natürlich steht und fällt jedes neue Speichermedium mit seiner Aufnahme bei den Softwarehäusern. Das ist auch ein Handicap beim Wafadrive, weil zur Zeit kaum Programme auf Wafadrive-Cartridges zur Verfügung stehen – ein Problem, das für die Besitzer des Spectrum Microdrive ebenso zutrifft. Damit ist für den Wafadrive-Benutzer aber noch nicht alles verloren: Zumindest eine Firma vertreibt jetzt ein Programm zum Überspielen kommerzieller Software auf Wafadrive-Datenträger. Das nötigt zwar jedesmal gleich zum Kauf einer zusätzlichen Leercassette, aber der Aufwand lohnt sich zweifellos wegen der erheblich kürzeren Zugriffszeit.

Ein gewisses Problem stellen auch die nicht normgerechten Steckverbindungen am Wafadrive dar. Wenn der Benutzer nicht bereit ist, die Kabelstecker selbst zu bauen, muß er unter Umständen lange nach Peripheriegeräten mit passenden Anschlußmöglichkeiten suchen. Trotz dieser Nachteile ist das Wafadrive aber ein leistungsfähiges Gerät und ohne Frage eine vollwertige Alternative zum Interface 1 und dem Microdrive von Sinclair.

Rotronics Wafadrive

ABMESSUNGEN
230x110x80mm

SCHNITTSTELLEN
seriell RS232, parallel Centronics, Peripherie-Steckleiste

SPEICHERMEDIUM
Endlose Magnetbandschleife

KAPAZITÄT
Lieferbar sind Cartridges mit 16, 64 und 128 KByte

GESCHWINDIGKEIT
Übertragungsrate 16 000 Baud, maximale Zugriffszeit 6,5 s bei 16-KByte- und 45 s bei 128-KByte-Cartridges.

Wafadrive-Cartridge
Das Gehäuse erinnert in seinen Größenverhältnissen an eine übliche Cassette, es enthält aber eine endlose Bandschleife. Daher entfällt das Zurückspulen beim Aufsuchen von Daten, die schon am Schreib/Lesekopf vorbeigelaufen sind.

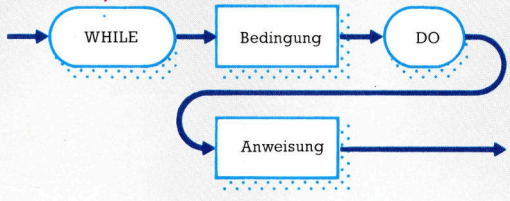
Mach's noch einmal, RAM

In unserer Einführung in PASCAL wurden bisher die strukturierten Anweisungen gezeigt, mit denen sich Programme exakt auf einzelne Probleme zuschneiden lassen. Heute untersuchen wir drei verschiedene Strukturen für die Steuerung von Programmschleifen.

Zwei der drei Möglichkeiten zur Strukturierung von PASCAL-Anweisungen haben wir bereits dargestellt. Sequentielle Strukturen sind in die Wörter BEGIN und END eingeschlossen, während bedingte Anweisungen mit IF und CASE aufgebaut werden. Für den dritten Strukturtyp – die Wiederholung – stellt PASCAL drei Konstruktionsmöglichkeiten zur Verfügung. Die FOR-Anweisung kann eine feste Anzahl von Schleifendurchgängen aufrufen, während WHILE und REPEAT durch Bedingungen gesteuert werden.

In der WHILE-Schleife wird zunächst der Boolesche Ausdruck zwischen den beiden reservierten Wörtern WHILE und DO bewertet.

Die WHILE-Anweisung:



Ist das Ergebnis wahr („true"), dann wird die folgende Anweisung (die jede nur mögliche PASCAL-Anweisung mit beliebig gestaffelter Tiefe sein kann) so lange wiederholt, wie die Bedingung zutrifft. Nach jeder Ausführung der Anweisung wird die Boolesche Bedingung des WHILE-Befehls neu bewertet. Das bedeutet aber auch, daß mindestens einer der in der Anweisung enthaltenen Werte durch die Abläufe innerhalb der Schleife verändert werden muß. Hier ein Beispiel:

```
WHILE MaxInt > 1 DO
  WriteLn ('Schleife wird ausgefuehrt')
```

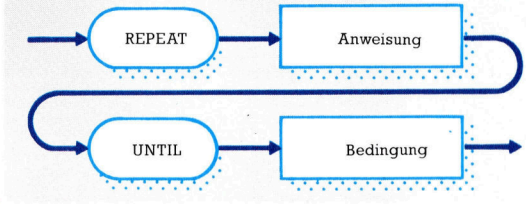
Da diese Schleife sich nur schwer beenden läßt, hier ein realistischeres Beispiel:

```
read ( Abhebung );
WHILE Abhebung > KontoGuthaben DO
  BEGIN
    WriteLn ('Nicht genug Geld vorhanden —
      Neuer Versuch:');
    write ('Abhebung DM ?');
    read ( Abhebung )
END
```

Dieser Programmteil verdeutlicht das wichtigste Merkmal der WHILE-Struktur: Wenn die Abhebung kleiner ist als das augenblickliche Kontoguthaben, wird die WHILE-Schleife nicht ausgeführt. Die Ausführung der Schleife wird beendet, wenn die Bedingung unwahr („false") ist. Die Steuerung wird dann wieder von dem sequentiellen Ablauf übernommen, und die erste Anweisung wird nach der WHILE-Struktur ausgeführt.

Beachten Sie, daß im Gegensatz zu WHILE die Endbedingung für REPEAT..UNTIL am Ende der Struktur liegt. Die REPEAT-Schleife wird beendet, wenn der Boolesche Ausdruck „true" ist, während die WHILE-Bedingung nur dann aufhört, wenn die Bedingung „false" ist. Sind innerhalb der REPEAT-Struktur mehrere Anweisungen enthalten, können sie in BEGIN . .END eingeschlossen werden. Dies ist jedoch nicht nötig, da die reservierten

Die REPEAT-Anweisung:



Wörter REPEAT und UNTIL als Begrenzung ausreichen. BEGIN und END haben die gleiche Funktion wie ein zusätzliches Semikolon vor einem reservierten Wort, wobei jedoch keine leere Anweisung entsteht.

Der folgende Programmteil zählt, wie oft der Buchstabe e in einem per Tastatur eingegebenen Text vorkommt. Die Eingabe muß mit einem Punkt abgeschlossen werden, damit die REPEAT-Schleife beendet und das Ergebnis dargestellt werden kann.

```
Zahl:= 0 ;
REPEAT
  read (Zeichen) ;
  IF Zeichen = 'e' THEN
    Zahl :=Zahl + 1
UNTIL Zeichen = '.';
WriteLn ('In dem Satz sind ', Zahl : 1,
  ''''e''s enthalten.')
```

Antworten zu den Übungen

In der vorigen Folge stellten wir die Aufgabe, bei der Generierung der Fibonacci-Einheiten und der Verhältniszahlen der aufeinanderfolgenden Paare die Schleife so zu verändern, daß die Endbedingung eintritt, wenn die nächste Fibonacci-Einheit, die berechnet werden soll, MaxInt überschreitet.

Eine Ganzzahl, die bei der Darstellung im Zweierkomplement MaxInt überschreitet, zerstört das eigene Vorzeichenbit. Da die Zahl dann negativ erscheint, würde das Programm „abstürzen". Es kann keine negativen Werte verarbeiten und würde selbst mit ganzzahligen Werten die Schleife niemals beenden können:

```
UNTIL first+second >
  MaxInt
```

kann nie eintreten, da es für den Computer keine Zahl gibt, die größer als MaxInt ist. Mit einem kleinen Trick läßt sich das Problem jedoch lösen:

```
UNTIL second >
  MaxInt-first
```

Jetzt werden natürlich auch die Real-Arithmetik und die Konstante Epsilon nicht mehr benötigt.

Für diesen Ablauf lassen sich beide Strukturen einsetzen. Die WHILE-Version sieht folgendermaßen aus:

```
Zahl:= 0 ;
read (Zeichen) ;
WHILE Zeichen <> '.' DO
  BEGIN
    IF Zeichen = 'e' THEN
      Zahl := succ (Zahl);
    read (Zeichen) ;
  END;
write ('In dem Satz sind ', Zahl : 1, );
WriteLn ('''e''s enthalten.')
```

WHILE-Befehlseingabe

Der Unterschied zwischen beiden Strukturen ist leicht zu erkennen. In der WHILE-Struktur muß ein Befehl unter Umständen zweimal eingegeben werden (die read-Anweisung), wenn die Endbedingung eintreten soll, – einmal unmittelbar vor der WHILE-Struktur und einmal in der letzten Anweisung der Schleife.

Obwohl die WHILE-Struktur sehr viele Anforderungen erfüllt, ist es manchmal notwendig, daß ein Ablauf nach einer festen Anzahl von Wiederholungen – beispielsweise nach Durchlauf einer begrenzten Werteskala – beendet werden kann. PASCAL bietet dafür die FOR-Struktur, die jedem BASIC-Programmierer vertraut ist. Zwischen den FOR-Schleifen beider Sprachen bestehen jedoch wesentliche Unterschiede. Zunächst kann jeder Skalar die Steuerung der Schleife übernehmen, nicht nur eine Ganzzahl. Wichtiger ist jedoch die Präzision der Anweisung. Wie die WHILE-Schleife kann auch FOR . .DO nicht ausgeführt werden, wenn beispielsweise der Anfangswert größer ist als der Endwert. Außerdem sind die Steuervariablen in PASCAL sehr starken Einschränkungen unterworfen. Insbesondere läßt sich ihr Wert nicht innerhalb der Schleife verändern – schon der Versuch wird von PASCAL als Fehler gemeldet. Der Vorteil dieser Sicherheitsmaßnahmen wird noch deutlich werden. Zunächst jedoch ein illegales Beispiel:

```
FOR N := 1 TO 10 DO
  IF N = 10 THEN
    N := 1
```

Diese Struktur ist fehlerhaft, da die IF-Anweisung nach zehnfacher Ausführung der Schleife die Variable N wieder auf 1 setzt und so eine unendliche Schleife erzeugt.

Die Syntax der FOR . .DO-Anweisung verlangt zwischen den beiden Begrenzungen (FOR und TO) die Zuweisung des Anfangswertes der Steuervariablen. Der Endwert wird von den reservierten Wörtern TO und DO eingeschlossen. Hier einige Beispiele:

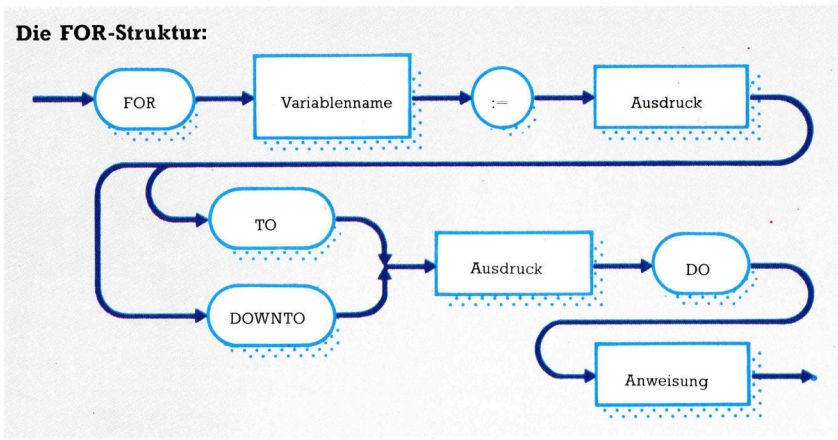
```
FOR Zeichen := 'A' TO 'Z' DO (* etc. *)
FOR Monat := 'Jan' TO 'Dez' DO (* etc. *)
FOR N := N TO succ(MaxInt DIV 1000) DO
  (* etc. *)
```

Das letzte Beispiel setzt voraus, daß N zuvor schon ein geeigneter Wert zugeordnet wurde. Wenn dieser Wert den höchstmöglichen Wert überschreitet, wird die Anweisung überhaupt nicht ausgeführt. Für eine absteigende Wertskala wird anstelle von TO das reservierte Wort DOWNTO verwandt. So könnte der Start einer Rakete folgendermaßen programmiert werden:

```
FOR CountDown := 10 DOWNTO 0 DO
  WriteLn (Countdown : 32 — 3 *
    CountDown);
WriteLn ('Wir haben LIFT OFF !')
```

Die Ausführung der FOR-Schleife wird von PASCAL sehr streng gehandhabt. Es gibt keine Entsprechung für den BASIC-Befehl STEP, der die Inkrementierung und Dekrementierung modifizieren kann.

Die FOR-Struktur:



Bierpreise

```
PROGRAM     BierPreise     (output);
Type
  Bier = ( Hell, Dunkel, Bock, Alt );
VAR
  Preis,
  Halbe : real ;
  Typ   : Bier ;
BEGIN
  WriteLn ( 'Halbe' :7,
    'Hell' :8, 'Dunkel' :9
      'Bock' :7, 'Alt' :7 ) ;
  WriteLn;
  Halbe :=0.5 ; (* Anfang bei einem Viertel *)
REPEAT
  IF Halbe — trunc ( Halbe ) > 0.4
    THEN (* als ##.# ausgeben *)
      write ( Halbe : 6 : 1, ' ' )
    ELSE (* als Ganzzahl ausgeben *)
      write ( round ( Halbe ) : 4, ' ' : 3 ) ;
  FOR Bier := Hell TO Alt DO
    BEGIN
      CASE Bier OF
        Hell   : Preis := 1.55;
        Dunkel : Preis := 1.65;
        Bock   : Preis := 1.75;
        Alt    : Preis := 1.85
      END; (* CASE *)
        (* runden und in Mark umwandeln *)
      write ( round ( Preis * Halbe )
        / 100 : 8 : 2 )
  END;
      (* neue Zeile anfangen *)
    WriteLn;
    Halbe := Halbe + 0.5
  UNTIL Halbe > 10
END.
```

Das Programm Bierpreise druckt in Viertellitersprüngen eine Preisliste für vier verschiedene Biersorten. Jede Biersorte ist in der Typendeklaration TYPE als Typenname aufgeführt. Die Auswahl der entsprechenden Preise geschieht mit der CASE-Anweisung. „Ganzzahlige" Halbe werden als Integer angezeigt und mit zwei Leerzeichen versehen. Durch die Formatierung der realen Zahlen (reals) verschwinden unerwünschte Dezimalstellen.

Morgenstund' hat Gold im Mund

Zusammen mit der geeigneten Software schaltet unser Relais-Modul aus dem letzten Kursabschnitt ein beliebiges Gerät ein oder aus. Diesmal liefern wir Ihnen Anregungen – und die nötigen Programme –, um Ihr Netzrelais sinnvoll einzusetzen.

Das Relais-Modul kann jedes angeschlossene Gerät programmgesteuert mit der Netzspannung versorgen. Dazu ist nur eine Signalspannung notwendig, die unser bereits entwickelter Niedervolt-Ausgang gemeinsam mit dem Ausgangsbuffer zur Verfügung stellt. Das Relais schaltet nur dann die Netzspannung ein, wenn der Niedervolt-Ausgang das Signal dazu gibt. Bei der Programmentwicklung können Sie sich auf dieselben Techniken stützen, die bei der Steuerung unserer Spielzeugautos und anderer Geräte mit niedriger Versorgungsspannung zum Einsatz kamen.

Verbinden Sie die Signalanschlüsse des Netzrelais mit dem positiven und dem negativen Anschluß von Kanal 0 Ihres Niedervolt-Ausganges. Das Relais schaltet dann die Netzspannung nur ein, wenn Bit 0 im Register des User Port auf „High" liegt. Ist Bit 0 aber „Low", gelangt keine Spannung mehr zum angeschlossenen Gerät. Über den Niedervolt-Ausgang können Sie bis zu vier Relaismodule unabhängig voneinander steuern.

Der Haushalt bietet ein umfangreiches Einsatzgebiet für die Möglichkeiten Ihres neuen Steuerungs-Systems – es kann viele Tätigkeiten vereinfachen und manche umständlichen Abläufe bequemer machen. Zum Auftakt und als Vorübung soll Ihr Heimcomputer durch einen relaisgesteuerten Cassettenrecorder und einen Trittschalter das „Sprechen" lernen.

Bevor der Computer auf die Signale reagiert, müssen Sie einige Sätze aufnehmen, etwa „Warum trittst du mich?", danach „Und jetzt schon wieder!", „Ich warne dich!" und so weiter. Wenn der Fußschalter und der Cassettenrecorder über die User-Port-Steuerung angeschlossen sind, werden diese Sätze einzeln nach jedem neuen Tritt abgespielt – vorausgesetzt, das Programm dazu „stimmt".

So werden die Geräte angeschlossen:

1) Signalleitungen des Relaismoduls mit dem positiven und negativen Anschluß von Kanal 0 des Niedervolt-Ausgangs verbinden.
2) Relaismodul ans Netz anschließen.
3) Die Zuleitungen des Fußschalters mit dem positiven und negativen Anschluß von Kanal 7 des Ausgangsbuffers verbinden.

Etwas schwierig ist es, beim dazugehörigen Programm die Schaltzeiten so einzurichten, daß der Recorder exakt vor und hinter einem Satz ein- und ausgeschaltet wird. Sie müssen also die Dauer der einzelnen Nachrichten genau messen, entweder mit einer Stoppuhr oder mit der eingebauten Uhr Ihres Computers. Bei drei Sätzen mit der Dauer T(1), T(2) und T(3) (in Sekunden) muß der Recorder nach jedem Druck auf den Schalter für die vorgegebene Zeit laufen und danach auch wieder anhalten. Je genauer Sie die Zeit stoppen, um so „spontaner" reagiert der Recorder dann später auf jeden „Fußtritt".

Die folgenden Programme für den Commodore 64 und den Acorn B schalten den Casset-

Acorn B

```
 10 REM BBC TREADING PROGRAM
 20 DIM T(3)
 30 DDR=&FE62:DATREG=&FE60
 40 ?DDR=127:REM L7 INPUT
 50 ?DATREG=0:REM ALL OFF
 60 CLS
 70 FORI=1TO3
 80 INPUT"TIME INTERVAL (SECS)";T(I)
 90 NEXT I
100 :
110 FORL=1TO3
120 CLS
130 REPEAT
140 UNTIL(?DATREG AND 128)=0:REM L7 LOW
150 ?DATREG=1:REM TURN ON TAPE
160 TIME=0:REM START TIMER
170 REPEAT
180 UNTIL TIME>T(L)*100
190 ?DATREG=0:REM TURN TAPE OFF
200 NEXT L
210 END
```

Commodore 64

```
10 REM CBM 64 TREADING PROGRAM
20 DD=56579:DATREG=56577
30 POKEDDR,127:REM L7 INPUT
40 POKEDATREG,0:REM ALL OFF
50 PRINTCHR$(147):REM CLEAR SCREEN
60 FORI=1 TO 3
70 INPUT"TIME INTERVAL (SECS)";T(I)
80 NEXT I
90 :
100 FOR L=1TO3
110 IF(PEEK(DATREG)AND128)<>0 THEN 110
115 POKEDATREG,1:REM TURN TAPE ON
120 T=TI:REM INITIALISE TIMER
130 IF T(L)>(TI-T)/60 THEN130
140 POKEDATREG,0:REM TURN TAPE OFF
150 NEXT L
160 END
```

tenrecorder für die drei aufeinanderfolgenden Zeitintervalle T(1), T(2) und T(3) ein – vorausgesetzt, daß vorher der Fußschalter betätigt wurde. Nur die Länge der einzelnen Nachrichten müssen Sie natürlich noch eingeben.

Nachdem Sie Ihren Computer mit den Füßen treten durften, kann er sich jetzt in früher Morgenstunde dafür rächen – als Wecker, den Sie aber – zum Glück – nach Ihren eigenen Wünschen programmieren können. Die Programme für den Commodore 64 und den Acorn B sind variabel und erlauben folgende Eingaben:

1) Uhrzeit,
2) Anzahl der gewünschten „Schlummer"-Intervalle (Zeiten zwischen zwei Wecksignalen bzw. Musik),
3) die Länge jedes „Schlummer"-Intervalls sowie die Wahl zwischen Wecksignal, Musik oder Pause,
4) Angabe, ob während bestimmter Zeiträume das Licht eingeschaltet sein soll,
5) den spätesten Zeitpunkt aufzustehen.

Programmierbarer Wecker

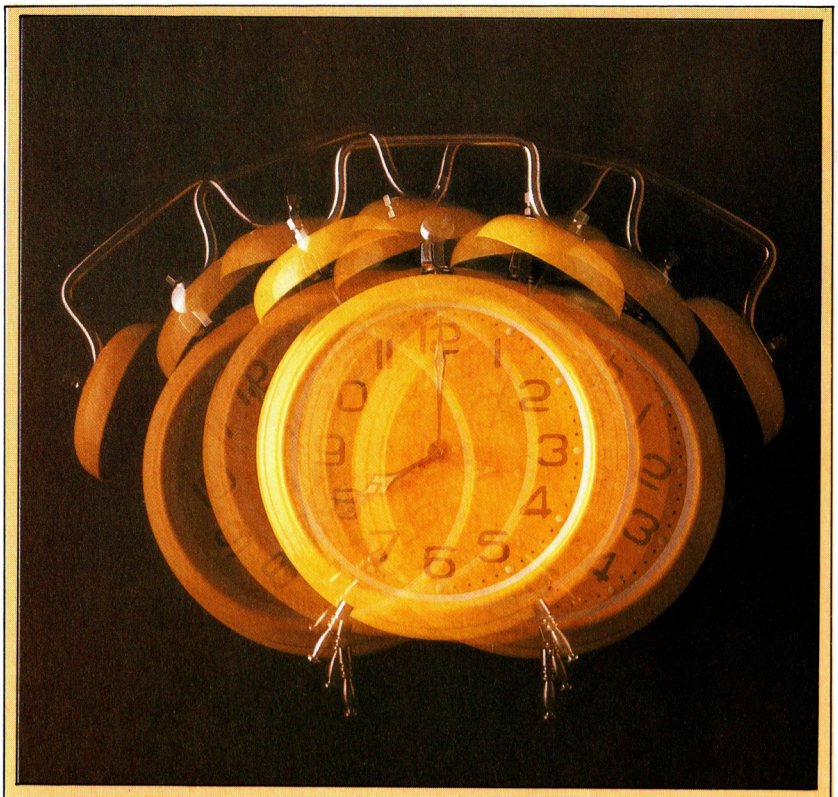
Zum korrekten Programmablauf müssen folgende Geräte am Niedervolt-Ausgang angeschlossen werden:

1) Ein Cassettenrecorder über das Netzrelais-Modul auf Kanal 0,
2) eine Tischlampe über das Netzrelais-Modul auf Kanal 1,
3) eine Klingel für 9 Volt, die direkt mit Kanal 3 verbunden wird.

Das Programm rechnet von der letztmöglichen Aufsteh-Zeit zurück und bestimmt so den Startzeitpunkt jedes einzelnen Intervalls. Die Informationen darüber, welche Geräte zu welchem Zeitpunkt eingeschaltet sein sollen, werden in Datenfeldern gespeichert. Die Feldvariablen entsprechen dem zum Einschalten der Geräte nötigen Bit-Wert im Datenregister. Mit dem logischen OR-Befehl kann der gesamte in das Datenregister einzulesende Wert für jede Gerätekombination schnell bestimmt werden.

Die größte Schwierigkeit bei der Programmerstellung ist es, die Stringvariablen so zu manipulieren, daß sich numerische Berechnungen damit anstellen lassen. Mit dem Commodore 64 ist dies besonders problematisch, weil ihm die praktischen MOD- und DIV-Befehle des Acorn B fehlen.

Sie haben jetzt ein wirklich universelles Ein- und Ausgabesystem für den Heimcomputer entwickelt. Nicht nur LEDs, sondern auch Niedervoltgeräte und Geräte für Netzspannung können damit gesteuert werden. Die Dateneingabe über unterschiedliche Sensoren ist ebenfalls möglich. Damit eröffnet sich eine Vielzahl individueller Einsatzmöglichkeiten, die von der hier vorgestellten Variante einer programmierbaren Zeitschaltuhr über die Regelung der Heizung bis zur Steuerung einzelner Lichtquellen im Haushalt reicht.



Commodore 64

```
100 REM **** CBM 64 ALARM CLOCK ****
110 DDR=56579:DATREG=56577
120 POKE DDR,255:POKEDATREG,0
130 PRINTCHR$(147):REM CLEAR SCREEN
140 INPUT"NUMBER OF SNOOZE INTERVALS";N
150 M=N+1
160 DIM A(M),M(M),L(M),TS$(M),T(M)
170 :
180 REM **** INPUT INTERVAL DATA ****
190 FOR C=1 TO N
200 PRINT:PRINT"INTERVAL NUMBER";C
210 INPUT"MUSIC,ALARM OR SILENCE (M/A/S)";AN$
215 AN$=LEFT$(AN$,1)
220 IF AN$<>"M"ANDAN$<>"A"ANDAN$<>"S"THEN 210
230 IF AN$="M" THEN M(C)=1:A(C)=0
240 IF AN$="A" THEN A(C)=8:M(C)=0
250 IF AN$="S" THEN A(C)=0:M(C)=0
260 INPUT"LIGHT ON (Y/N)";AL$
270 L$=LEFT$(AL$,1)
280 IF L$<>"Y" AND L$<>"N" THEN 260
290 IF L$="Y" THEN L(C)=2:GOTO310
300 L(C)=0
310 INPUT"TIME INTERVAL (MINS)";T(C)
320 NEXT C
330 :
340 INPUT"LATEST RISING TIME (HHMM)";LT$
350 LT$=LT$+"00":REM ADD SECONDS
360 TS$(N+1)=LT$:REM LAST TIME
370 REM CONVERT LATEST TIME TO MINUTES
380 LM=60*VAL(LEFT$(LT$,2))+VAL(MID$(LT$,3,2))
390 :
400 INPUT"TIME NOW (HHMM)";TN$
410 TI$=TN$+"00":REM START TIMER
420 :
430 REM **** ANALYSE AND CALCULATE ****
440 REM ** CALC INTERVAL START TIMES **
450 FOR C=N TO 1 STEP -1
460 LM=LM-T(C):REM START TIME IN MINS
470 HR=INT(LM/60)
480 MN=INT(60*(LM/60-HR+.000001))
490 HR$=STR$(HR):REM HOURS
500 MN$=STR$(MN):REM MINS
510 MN$=MID$(MN$,2,LEN(MN$))
520 HR$=MID$(HR$,2,LEN(HR$))
530 REM ** ADD LEADING ZEROS **
540 SP$="00"
550 HR$=LEFT$(SP$,2-LEN(HR$))+HR$
560 MN$=LEFT$(SP$,2-LEN(MN$))+MN$
570 TS$(C)=HR$+MN$+"00"
580 NEXT C
590 :
600 REM **** GO ****
610 PRINTCHR$(147)
620 FOR C=1 TO N+1
630 IF TI$<TS$(C)THENGOSUB710:GOTO630
640 DN=M(C) OR A(C) OR L(C):REM DATREG DATA
650 POKE DATREG,DN
670 NEXT C
680 POKE DATREG,0
690 END
700 :
710 REM **** DISPLAY TIMER S/R ****
720 PRINTCHR$(145);:REM CRSR UP
730 PRINTLEFT$(TI$,2)":"MID$(TI$,3,2);
740 PRINT" "RIGHT$(TI$,2)
750 RETURN
```

Acorn B

```
10 REM BBC ALARM CLOCK
15 MODE7
20 DDR=&FE62:DATREG=&FE60
30 CLS
40 INPUT"NUMBER OF SNOOZE INTERVALS";N
45 M=N+1
50 DIM A(M),M(M),L(M),T(M),TS(M)
70 REM **** INPUT INTERVAL DATA ****
80 FORC=1 TO N
90   PRINT"INTERVAL NUMBER";C
96   REPEAT
100     PRINT "MUSIC,ALARM OR SILENCE";
103     INPUT " (M/A/S)";AN$
105    AN$=LEFT$(AN$,1)
110    UNTIL AN$="M"ORAN$="A"ORAN$="S"
120  IF AN$="M"THEN M(C)=1:A(C)=0
130  IF AN$="A"THEN M(C)=0:A(C)=8
140  IF AN$="S"THEN M(C)=0:A(C)=0
150  REPEAT
160    INPUT"LIGHT ON (Y/N)";AL$
170    AL$=LEFT$(AL$,1)
180    UNTIL AL$="Y" OR AL$="N"
190  IF AL$="Y" THEN L(C)=2 ELSE L(C)=0
200  INPUT"TIME INTERVAL (MINS)";T(C)
210  NEXT C
220 :
230 INPUT"LATEST RISING TIME (HHMM)";LT$
232  TS(N+1)=6000*(60*VAL(LEFT$(LT$,2)))
234   TS(N+1)=TS(N+1)+VAL(RIGHT$(LT$,2))
236 REM CONVERT LATEST TIME TO MINS
237 LM=60*VAL(LEFT$(LT$,2))
239  LM=LM+VAL(RIGHT$(LT$,2))
240 INPUT"TIME NOW (HHMM)";TN$
250  TIME=6000*(60*VAL(LEFT$(TN$,2)))
255   TIME=TIME+VAL(RIGHT$(TN$,2))
260 :
270 REM ANALYSE AND CALCULATE
280 FORC=N TO 1 STEP -1
290   LM=LM-T(C):REM INTERVAL START
300   TS(C)=6000*LM
310    NEXT C
320 :
330  REM **** GO ****
340 CLS
350 FOR C=1 TO N+1
360   REPEAT
370     PROCtimer
380     UNTIL TIME>=TS(C)
390   REGDATA=M(C) OR A(C) OR L(C)
400   ?DATREG=REGDATA
420   NEXT C
430 ?DATREG=0
440 END
999 :
1000 DEF PROCtimer
1020 MIN=(TIME DIV 6000) MOD 60
1030  HR=(TIME DIV 6000) MOD 60
1040 MIN$=STR$(MIN):HR$=STR$(HR)
1042 REM ADD LEADING ZEROS
1043 SP$="00"
1044 HR$=LEFT$(SP$,2-LEN(HR$))+HR$
1045 MIN$=LEFT$(SP$,2-LEN(MIN$))+MIN$
1050 PRINTTAB(18,12)HR$;":";MIN$
1060 ENDPROC
```

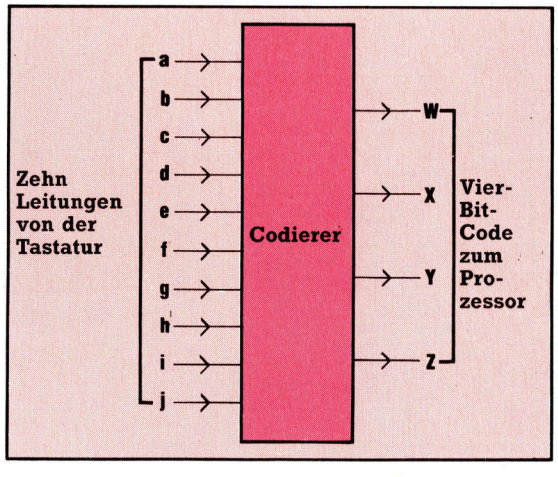
Leitungswechsel

Die CPU eines Computers übermittelt die Befehle an die Verarbeitungseinheiten in Form elektrischer Signale. Diese Signale werden vor dem „Abschicken" verschlüsselt und beim „Empfänger" wieder zurückübersetzt. Die dabei verwendeten Codier- und Decodierschaltungen sollen etwas gründlicher erläutert werden.

Der Prozessor schickt Anweisungen sowohl an interne (Akkumulator, ALU) als auch an externe Peripheriegeräte – etwa an den Drukker. Durch Verschlüsselung der Befehle ist dieser Datentransfer mit einem Minimum an Leitungen zwischen Prozessor und Peripherie möglich. Ein Beispiel dafür ist die Tastaturabfrage. Eine Tastatur hat eine bestimmte Anzahl von Ausgangsleitungen, von denen immer nur eine einzige eine Signalspannung führt. Das Tastatur-Signal kann daher auch als sechsstellige Binärzahl verschlüsselt werden – zur Übertragung sind dann nur noch sechs Leitungen erforderlich. Dazu kommen allerdings noch zwei Leitungen, eine für das Prüf-Bit und eine zweite für Fälle, in denen die Shift- oder Control-Taste mit einer anderen Taste zusammen gedrückt wird.

Vier Ausgänge

Um das Prinzip des dafür eingesetzten Codierers zu verdeutlichen, wählen wir das Beispiel einer einfacheren Tastatur mit nur zehn Tasten, auf der sich die Ziffern von 0 bis 9 eingeben lassen. Mit einem Drei-Bit-Wort hätten wir nur acht mögliche Bit-Kombinationen zur Verfügung, bei zehn Eingangsleitungen muß die Schaltung also vier Ausgänge haben. Die folgende Tabelle zeigt die entsprechenden Leitungen einer Zehner-Tastatur:

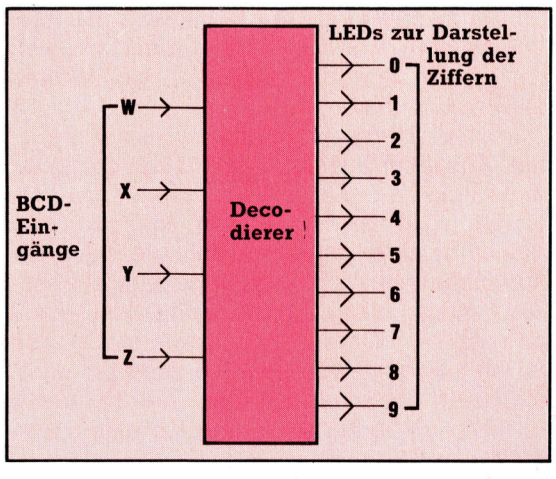


Weil jeweils nur eine der zehn Leitungen aktiv (High) sein kann, sieht die Wahrheitstabelle des Codierers folgendermaßen aus:

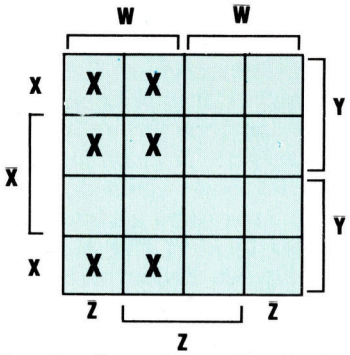
Dezimal	Eingänge										Ausgänge			
	a	b	c	d	e	f	g	h	i	j	W	X	Y	Z
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	1	0	0	0	0	0	0	0	0	0	1	0
3	0	0	0	1	0	0	0	0	0	0	0	0	1	1
4	0	0	0	0	1	0	0	0	0	0	0	1	0	0
5	0	0	0	0	0	1	0	0	0	0	0	1	0	1
6	0	0	0	0	0	0	1	0	0	0	0	1	1	0
7	0	0	0	0	0	0	0	1	0	0	0	1	1	1
8	0	0	0	0	0	0	0	0	1	0	1	0	0	0
9	0	0	0	0	0	0	0	0	0	1	1	0	0	1

Das Rückübersetzen aus einem Code nennt man „Decodieren". Dabei bedienen nur wenige Leitungen (meist direkt über den Binärcode des Prozessors) die Ausgabe einer Vielzahl verschiedener Datenkanäle. Dieses Verfahren wird zum Beispiel häufig zur Drucker- oder Plotteransteuerung eingesetzt, ist aber auch für die Positionierung des Schreib/Lesekopfes in einer Diskettenstation oder für die Wahl eines bestimmten Ausgabekanals zu einem Peripheriegerät gebräuchlich.

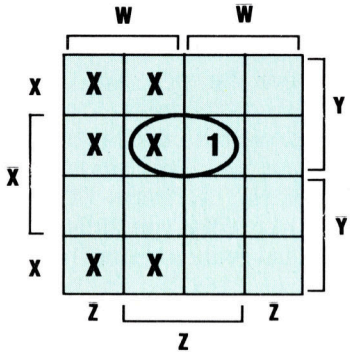
Die Arbeitsweise eines einfachen Decodierers aus AND-, OR- und NOT-Gattern läßt sich am besten mit einem Beispiel verdeutlichen: Der Decodierer soll einen BCD-Code (Binary-Coded-Decimal) so umwandeln, daß damit zehn einzelne LEDs entsprechend dem dezimalen Wert des Codes angesteuert werden können. Das wäre genau die Umkehrung der Aufgabenstellung des Codierers aus dem Tastatur-Beispiel.



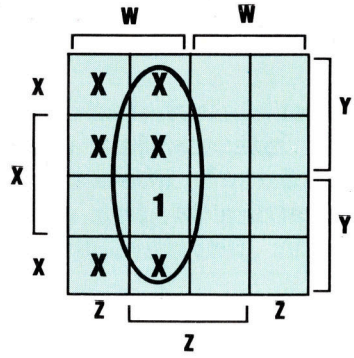
Im BCD-Code werden die dezimalen Ziffern 0 bis 9 durch je ein Vier-Bit-Wort dargestellt, der Decodierer hat also vier Eingänge. Damit sind 16 unterschiedliche Kombinationen von „Highs" in den Leitungen möglich. Da wir nur zehn Kombinationen brauchen, werden die restlichen sechs als „ungültige Eingabe" (X) betrachtet. Mit Booleschen Ausdrücken können wir die Abhängigkeit jedes der zehn Ausgänge vom Zustand der Eingabeparameter W,X,Y und Z einzeln angeben (Tabelle 2).



Wenn Sie die Ausgaben einzeln betrachten, finden sich weitere Möglichkeiten der Vereinfachung. Durch die Eintragung des Booleschen Ausdrucks für die Ausgabe der Zahl 3 ergibt sich eine verkürzte Darstellungsmöglichkeit: X.Y.Z.



Ähnlich wird im zweiten Beispiel mit dem Ausdruck für die Zahl 9 verfahren:

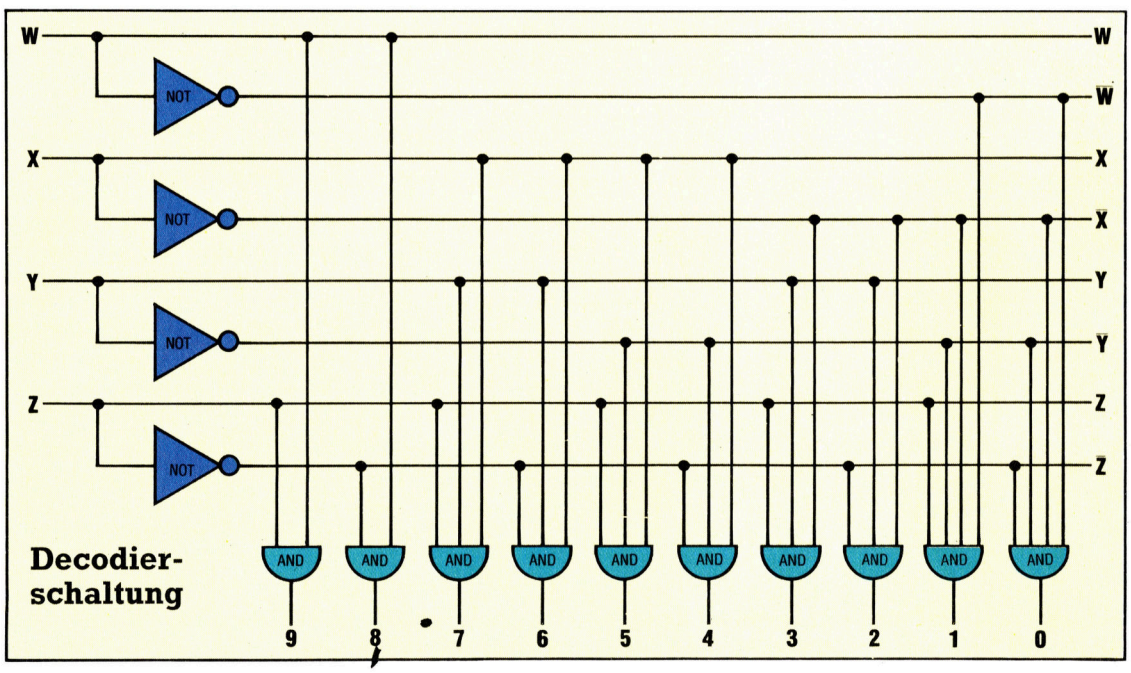


Die Schleife umschließt neben dem Ausdruck für die 9 drei Fälle von ungültiger Eingabe und steht für den Booleschen Ausdruck W.Z. Auf die gleiche Weise können noch einige andere Ausdrücke verkürzt werden.

Jetzt muß nur noch eine geeignete Schaltung zur Realisierung der zehn Booleschen Ausdrücke konstruiert werden. Da jede der vier Eingaben sowohl in normaler als auch in verneinter Form gebraucht wird, arbeitet man am besten mit acht parallelen Leitungen. Dann müssen für alle zehn Ausgänge nur Abzweigungen von den entsprechenden Leitungen hergestellt und mit AND-Gattern zusammengefaßt werden.

Übung 6

1) Entwerfen Sie eine Codierschaltung mit drei Eingängen, die bei Eingabe von 011,101,110 und 111 eine 1 und bei allen anderen Eingaben eine 0 ausgibt.
a) Zeichnen Sie dazu die Wahrheitstabelle.
b) Entwickeln Sie den Booleschen Ausdruck für die Ausgabe; vereinfachen Sie ihn.
c) Entwerfen Sie die Schaltung dazu.



Decodier-schaltung

Tabelle 1

Eingaben				BCD Zahl
W	X	Y	Z	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Tabelle 2

BCD Zahl	Boolescher Ausdruck
0	$\bar{W}.\bar{X}.\bar{Y}.\bar{Z}$
1	$\bar{W}.\bar{X}.\bar{Y}.Z$
2	$\bar{W}.\bar{X}.Y.\bar{Z}$
3	$\bar{W}.\bar{X}.Y.Z$
4	$\bar{W}.X.\bar{Y}.\bar{Z}$
5	$\bar{W}.X.\bar{Y}.Z$
6	$\bar{W}.X.Y.\bar{Z}$
7	$\bar{W}.X.Y.Z$
8	$W.\bar{X}.\bar{Y}.\bar{Z}$
9	$W.\bar{X}.\bar{Y}.Z$

Tabelle 3

BCD Zahl	Boolescher Ausdruck
0	$\bar{W}.\bar{X}.\bar{Y}.\bar{Z}$
1	$\bar{W}.\bar{X}.\bar{Y}.Z$
2	$\bar{X}.Y.\bar{Z}$
3	$\bar{X}.Y.Z$
4	$X.\bar{Y}.\bar{Z}$
5	$X.\bar{Y}.Z$
6	$X.Y.\bar{Z}$
7	$X.Y.Z$
8	$W.\bar{Z}$
9	$W.Z$

Spezialdruck

Selbst beim einfachsten Matrixdrucker läßt sich mit Spezialeffekten der Druck interessanter gestalten – beispielsweise durch besonders breite Zeichen. In diesem Artikel zeigen wir, wie diese Effekte erzeugt werden können und wie sich Computer und Drucker miteinander „unterhalten".

Elite

```
ABCDEFGHIJKLMNOP
QRSTUVWXYZabcdef
ghijklmnopqrstuv
wxyz0123456789 !
"£$%&'()*+,-./:;
<=>?@[\]^_`{|}~
```

Pica

```
ABCDEFGHIJKLM
NOPQRSTUVWXYZ
abcdefghijklm
nopqrstuvwxyz
0123456789 !"
£$%&'()*+,-./
:;<=>?@[\]^_`
{|}~
```

Pica kursiv halbfett

```
ABCDEFGHIJKLM
NOPQRSTUVWXYZ
abcdefghijklm
nopqrstuvwxyz
0123456789 !"
#$%&'()*+,-./
:;<=>?@[¥]^_`
{/}~
```

Elite gedehnt

```
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 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 0 1 2 3
4 5 6 7 8 9   !
" £ $ % & ' ( )
* + , - . / : ;
< = > ? @ [ \ ]
^ _ ` { ; } ~
```

Pica komprimiert mit Doppeldruck

```
ABCDEFGHIJKLMNOPQRSTUVW
XYZabcdefghijklmnopqrst
uvwxyz0123456789 !"£$%&
'()*+,-./:;<=>?@[\]^_`{
|}~
```

Nadeldrucker mit einer Punktmatrix bieten außer Schrifttypen wie Pica, Elite und Kursiv oft noch Gestaltungsmöglichkeiten wie Komprimieren, Dehnen und Hervorheben. Alle hier gezeigten Beispiele wurden auf dem Epson FX-80 erstellt.

Matrixdrucker können weit mehr, als nur Programme ausdrucken. Schon der kurze Blick in ein Benutzerhandbuch zeigt, wie vielfältig sich das Druckbild variieren läßt. Selbst der einfachste Matrixdrucker bietet die Möglichkeit, die Größe der Zeichen zu verändern, indem man die Anzahl der Zeichen pro Zeile (normalerweise 80) erhöht („komprimierter" Druck) oder herabsetzt („gedehnter" Druck). Auch die Zeilenzwischenräume können verändert werden. So entsteht beispielsweise ein dichtes Druckbild, wenn statt vier Zeilen acht Zeilen pro Zoll gedruckt werden.

In diesem Artikel sehen wir uns den FX-80 von Epson genauer an, da er über eine breite Palette von Druckmöglichkeiten verfügt. Der „hervorgehobene" Druck, bei dem Texte in fetter Schrift erscheinen, und die Kursivschrift sind nur zwei seiner Standardfähigkeiten. Außerordentlich praktisch ist auch, daß die Zeichen im Speicher des Druckers neu definiert und somit nach Belieben gegen die Buchstaben anderer Sprachen oder wissenschaftliche Symbole ausgetauscht werden können. Bevor wir darauf eingehen, werden wir jedoch untersuchen, wie einfache Programmlistings erzeugt werden.

Steuerungsbefehle

Der Informationsaustausch zwischen Computer und Drucker ist auf jeder Maschine anders. So verwendet der Dragon eine Variante des LIST-Befehls – LLIST –, um den Drucker zu veranlassen, eine Kopie des Programms auszugeben. Andere Geräte müssen dazu erst einen „Kanal" oder „Stream" eröffnen. Die Einzelheiten für Ihre Maschine finden Sie am ehesten im Benutzerhandbuch des Computers – Druckerhandbücher enthalten diese Information nur selten.

Doch auch nachdem die Kommunikation zwischen Computer und Drucker hergestellt ist, entspricht der erste Ausdruck nur selten den Erwartungen. Dabei tritt anfangs oft eins der folgenden Probleme auf: Entweder erscheint der gesamte Text in einer einzigen schwarzen Zeile, die nicht mehr zu entziffern ist, oder alle gedruckten Zeilen sind durch mehrere Leerzeilen voneinander getrennt. Diese Schwierigkeiten entstehen durch die Befehle für „Zeilenvorschub" (Line Feed) und „Wagenrücklauf" (Carriage Return). Der Computer stellt an das Ende jeder für den Drucker bestimmten Zeile das Zeichen für den Wagenrücklauf und veranlaßt damit den Druckkopf, an den linken Papierrand zurückzukehren. Einige Computer senden hier zusätzlich noch das Zeichen für den Zeilenvorschub, das das Papier eine Zeile hochschiebt. Viele Maschinen setzen jedoch voraus, daß der Drucker dies automatisch erledigt. In den meisten Druckern gibt es daher einen Schalter, mit dem eingestellt werden kann, ob ein eigener Zeilenvorschub erzeugt werden soll oder nicht. Wenn also eins der Probleme auftritt, sehen Sie am besten im Druckerhandbuch nach, wo dieser Schalter liegt, und stellen ihn um.

Außer für Programmlistings lassen sich Matrixdrucker auch für den Ausdruck der auf dem Bildschirm dargestellten Zeichen einsetzen. Auch hier variiert die Methode von Maschine zu Maschine. Der Standardbefehl des BASIC ist LPRINT. Auf dem Commodore 64 druckt OPEN1,4 gefolgt von PRINT#1,"HALLO" das Wort HALLO, während der Dragon für den gleichen Ablauf PRINT#-2,"HALLO" benötigt. Der Acorn B braucht ein VDU2 gefolgt von PRINT"HALLO" und dem Befehl VDU3. Beim folgenden Programmbeispiel wird LPRINT verwendet. Sie können es aber leicht für Ihre Maschine ändern:

```
10 LPRINT "MR JOHN SMITH"
20 LPRINT "7 THE PARADE"
30 LPRINT "ANYTOWN"
40 LPRINT "ABC 123"
50 FOR I=1 TO 7
60 LPRINT
70 NEXT I
100 GOTO 10
```

Mit diesem Programm lassen sich Adressen auf Etiketten ausdrucken. Selbstklebende Adreßaufkleber werden auf Endlosrollen geliefert. Die Trägerfolie ist an beiden Seiten mit Löchern versehen, die zum Transport an die Traktorführung des Druckers angepaßt werden. Das Programm enthält keine Spezialcodes und sollte auf allen Druckertypen funktionieren. Dabei wird zwar immer die gleiche Adresse gedruckt, doch können Sie es problemlos mit den entsprechenden Eingaben ergänzen oder Namen und Adressen aus Da-

teten einlesen. Die FOR. . .NEXT-Schleife zwischen Zeile 50 bis 70 druckt sieben Leerzeilen und setzt damit den Druckkopf für jede neue Adresse auf die korrekte Anfangsposition. Die genaue Anzahl Leerzeilen muß für jede Maschine individuell gesetzt werden.

Für einfache Aufkleber reicht unser Programm völlig aus. Sollen allerdings kompliziertere Schriftstücke wie Rechnungen oder Briefköpfe gedruckt werden, brauchen wir Spezialeffekte, die im Drucker mit Steuerzeichen eingeschaltet werden.

Zusätzliche Spezialeffekte

Außer den Zeichen der Tastatur enthält der ASCII-Code eine Reihe „unsichtbarer" bzw. „nicht druckbarer" Zeichen, die nicht auf Papier und Bildschirm erscheinen. Mit diesen Codes lassen sich die Spezialeffekte des Druckers einschalten. Der Standard-ASCII-Satz sieht zur Steuerung von Peripheriegeräten vier Codes (17, 18, 19 und 20) vor. Da diese jedoch nicht ausreichen, um die zirka 70 verschiedenen Spezialeffekte des Epson FX-80 abzurufen, erhält der FX-80 die meisten Steuerzeichen als sogenannte „Escape-Codes", die mit dem ESC-Zeichen (ASCII-Code 27) anfangen und aus zwei oder mehr Zeichen bestehen. Für das Einschalten der Proportionalschrift wird beispielsweise ein ESC-p gesandt.

In BASIC sieht das folgendermaßen aus:

```
LPRINT CHR$(27);"p"
```

Da das ESC-Zeichen normalerweise nicht durch das Drücken der ESC-Taste erzeugt werden kann, wird hier die CHR$-Funktion eingesetzt.

Auf dem Acorn B sieht der Ablauf so aus:

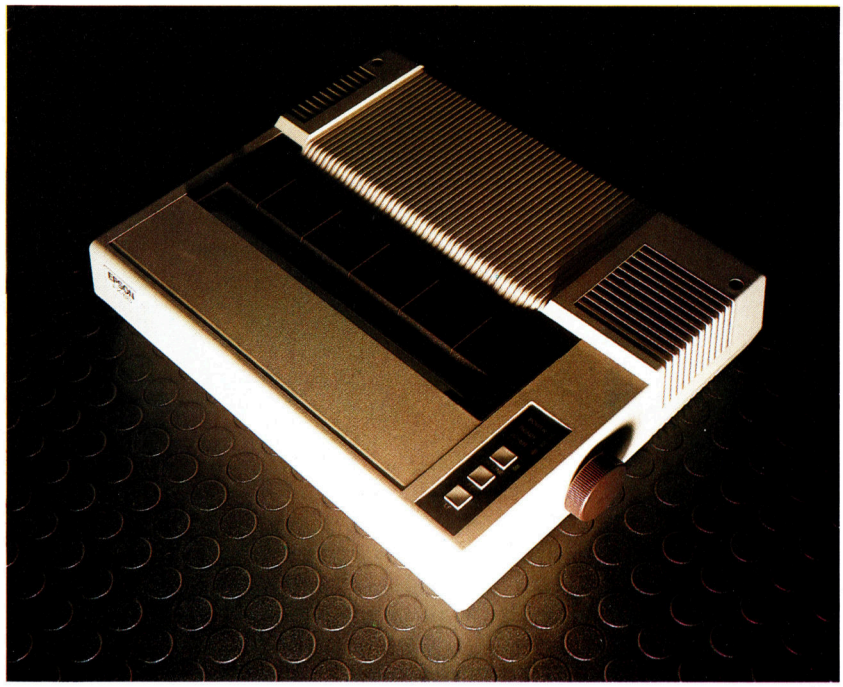
```
VDU2
VDU1,27,1,112
VDU3
```

Der Befehl VDU2 schaltet den Drucker an (enable), VDU1 bedeutet „Sende die folgenden Zeichen zum Drucker", und VDU3 schaltet den Drucker ab (disable).

Diese Steuerfolgen gelten nur für den Epson FX-80. Die gleichen Zeichen haben bei anderen Druckern entweder keine Auswirkung, lösen unerwartete Reaktionen aus oder „hängen den Drucker auf" (d. h. er ignoriert die Befehle des Computers).

Unser zweites Programm führt einige Fähigkeiten des Epson vor und druckt eine Rechnung, die von einer kleinen Autoreparaturwerkstatt stammen könnte. Da Epson-Drucker sehr weit verbreitet sind, haben andere Hersteller Epson-kompatible Geräte mit ähnlichen Steuerfolgen herausgebracht. Wenn Ihr Drukker nicht in diese Kategorie fällt, müssen Sie die Codes entsprechend ändern.

```
999 REM INVOICE HEADING
1000 LPRINT CHR$(12)
1010 LPRINT CHR$(14);TAB(12);"HCAC
MOTORS LTD."
1020 LPRINT CHR$(13);CHR$(13)
1030 LPRINT CHR$(27);"E";
1040 LPRINT TAB(36);
1050 LPRINT CHR$(27);"-";CHR$(1);
1060 LPRINT "INVOICE";
1070 LPRINT CHR$(27);"-";CHR$(0);
1080 LPRINT CHR$(27);"F";
1090 LPRINT CHR$(13);CHR$(13);CHR$(13)
1100 REM INVOICE DETAILS PRINTED
```



Trotz des hohen Preises ist dieser Drucker im kommerziellen und im Hobbybereich weit verbreitet. Der Druckkopf des FX-80 ist mit neun Nadeln ausgerüstet und druckt 160 Zeichen pro Sekunde. Die meisten Softwarepakete (zum Beispiel Textsysteme) unterstützen Epson- oder Epson-ähnliche Drucker.

Der ASCII-Code 12 in Zeile 1000 löst den „Seitenvorschub" (Form Feed) aus, der das Papier hochrollt, bis der Druckkopf am Anfang einer neuen Seite steht. Darauf folgt Code 14 (auch „Shift Out" – SO-Zeichen – genannt), der veranlaßt, daß alle folgenden Zeichen in gedehnten Buchstaben gedruckt werden – in diesem Fall der Name der Werkstatt. Die TAB-Funktion setzt den Rechnungskopf in die Zeilenmitte.

CHR$(13) ist das Zeichen für den Wagenrücklauf (Return). Allein eingesetzt erzeugt es eine Leerzeile. Von Zeile 1020 bis 1090 ziehen mehrere dieser Zeichen den Rechnungskopf auseinander. ESC-E in Zeile 1030 schaltet die hervorgehobene Schrift ein und veranlaßt, daß alle folgenden Zeichen in fetten Buchstaben erscheinen (die gleichen Buchstaben werden mehrfach überdruckt). Zeile 1050 schaltet die Unterstreichung ein, während Zeile 1070 sie wieder ausschaltet, nachdem das Wort „INVOICE" (Rechnung) unterstrichen wurde. ESC-F schaltet die hervorgehobene Schriftart aus. Ein vollständiges Rechnungsprogramm enthält natürlich einen Rechnungstext mit den ausgeführten Arbeiten, der Bankverbindung sowie den üblichen Daten. Diese Einzelheiten können durch Variablen am Anfang des Programms eingegeben werden.

Die beiden Programmbeispiele geben einen kleinen Einblick in die vielfältigen Möglichkeiten eines Matrixdruckers.

Sprite-Routinen

Mit Hilfe der Sprite-Grafik lassen sich auch in BASIC schnelle Spiele programmieren. Einige Computer haben sogar Chips, die speziell auf die Steuerung von Sprites ausgerichtet sind. In diesem Artikel wird die Sprite-Erzeugung auf dem Acorn B erläutert.

Jedes Sprite, egal welche Form oder Größe es hat, wird auf einem Raster entworfen. Zwar gibt es in der Rastergröße keine festen Grenzen, doch empfiehlt es sich, Raster zu wählen, dessen Ausmaße ein Byte oder das Vielfache eines Bytes betragen (8, 16, 24 Bits). Unsere Routine setzt ein Raster ein, das 24 Pixel breit und 21 Pixel hoch ist und im Speicher 63 Bytes belegt. Für die Konstruktion des Sprites wird zunächst die Form auf das Raster gezeichnet und dann in Binärcode umgewandelt. Nachdem die Binärwerte zu Bytes zusammengestellt sind, werden diese in Dezimal- oder Hexadezimalzahlen umgewandelt und gespeichert. Der Kasten zeigt dies an einem Beispiel.

Ursprünglicher Inhalt	C	1 1 0 1 1 1 0 0
Nach einer Rotation	1	1 0 1 1 1 0 0 C
Nach zwei Rotationen	1	0 1 1 1 0 0 C 1
Nach drei Rotationen	0	1 1 1 0 0 C 1 1
Nach vier Rotationen	1	1 1 0 0 C 1 1 0
Nach fünf Rotationen	1	1 0 0 C 1 1 0 1
Nach sechs Rotationen	1	0 0 C 1 1 0 1 1
Nach sieben Rotationen	0	0 C 1 1 0 1 1 1
Nach acht Rotationen	0	C 1 1 0 1 1 1 0
Nach neun Rotationen	C	1 1 0 1 1 1 0 0

Das Sprite besteht aus 24x21 Pixeln und wird mit dem normalen Bit-Map-Verfahren in 63 Bytes gespeichert – ein Pixel pro Bit. Die Farbe des gesamten Sprites wird von der Variablen LOGCOL festgelegt. Die Spritegröße bestimmen die beiden Faktoren XSCALE und YSCALE, die gradzahlig sein sollten.

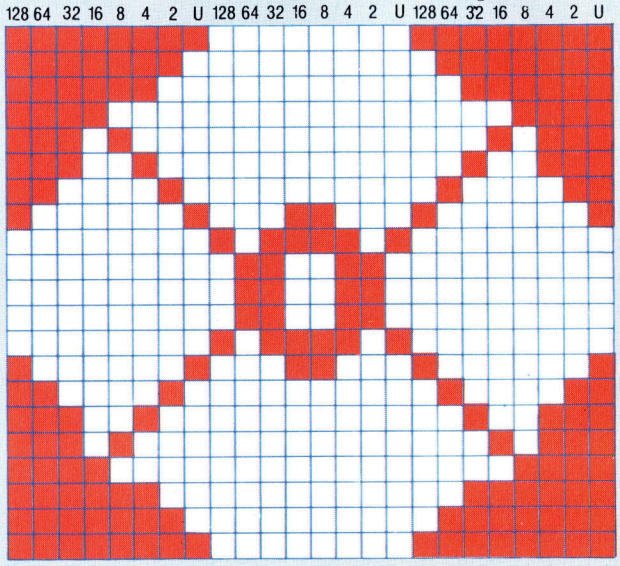
Die 63 Zahlen mit der Spritedefinition lassen sich über DATA- und READ-Befehle in den BASIC-Teil des Programms einsetzen. Die Zahlen müssen in einem speziell dafür reservierten Speicherbereich abgelegt sein. Dieser Bereich kann an einer beliebigen Stelle des RAM liegen, darf jedoch während des Programmablaufs nicht überschrieben werden. Der beste Platz für Sprite-Daten ist die Obergrenze des BASIC-Programmbereichs. Diese Grenze ist in der Variablen HIMEM festgelegt. Damit die Daten nicht überschrieben werden können, wird HIMEM in den Programmzeilen 220 und 230 herabgesetzt. Darin wird auch die Adresse des ersten Sprite-Bytes SPRDAT auf das erste Byte über den neuen Wert von HIMEM gelegt. Die Zeilen 1740 bis 1770 lesen die Daten ein und legen sie in den 63 Bytes ab, die bei SPRDAT anfangen.

Die Hauptfunktion der Maschinencoderoutine ist es, den Aufbau des Sprites zu analysieren und alle Vorgänge auszuführen, die die Daten in eine Bildschirmanzeige umsetzen. Dazu muß der Maschinencode zunächst alle Bits der 63 Datenbytes einzeln untersuchen und entscheiden, ob ein Punkt (wenn das Bit auf Eins steht) dargestellt werden soll oder nicht (wenn das Bit Null ist). Mit den Rotate-Befehlen lassen sich die Bits eines Bytes am einfachsten analysieren. So setzt Zeile 1100 des Quellencodes „ROtate Left" (ROL) ein. Dieser Befehl bewegt jedes Bit des Bytes um eine Stelle nach links, wobei der Wert des Übertragsflags rechts in das Byte eingesetzt wird und das Bit, das links aus dem Byte „herausfällt", in das Übertragsflag wandert.

Aus diesem Beispiel läßt sich leicht ersehen, daß der Anfangsinhalt des Übertragsflags (C) nicht beachtet werden muß, da es nach der Rotation durch das Byte wieder im Übertragsflag steht. Beachten Sie, daß ROL in Zeile 1100 im absoluten indizierten Adressiermodus eingesetzt wird, damit alle 63 Bytes nacheinander analysiert werden.

Sobald die Routine festgestellt hat, ob sie einen Punkt oder ein Leerzeichen plotten soll, beginnt der Aufbau der Bildschirmanzeige. Der Acorn B verwendet dafür zwei Methoden. Bei der ersten werden die entsprechenden

Acorn-Sprite



DATA

Sp. 1	Sp. 2	Sp. 3
255	0	255
254	0	127
252	0	63
240	0	15
232	0	23
228	0	39
194	0	67
129	24	129
0	189	0
0	102	0
0	102	0
0	102	0
0	189	0
129	24	129
194	0	67
228	0	39
232	0	23
240	0	15
252	0	63
254	0	127
255	0	255

Werte mit POKE direkt in den Bildschirmspeicher gesetzt. Leider ist das nicht so einfach, wie es sich anhört. Zunächst sind die Bildschirmspeicher bei den verschiedenen Modi unterschiedlich angelegt, und die mathematische Beziehung zwischen Pixeln und Bits des Bildschirms ist sehr komplex. So steuert beispielsweise jedes Speicherbyte im Modus 2 nur zwei Pixel des Bildschirms, da hier 16 Farben unterstützt werden und jedes Pixel vier Bits für die Farbkennung benötigt. Im Modus 0 (Zwei-Farben) belegt jedes Pixel jedoch nur ein Bit. Somit läßt sich ein im Modus 2 entworfenes Sprite auch nur in diesem Modus darstellen.

Sprites für alle Modi

Glücklicherweise gibt es eine weitere Methode. Bei der Grafikprogrammierung in BASIC muß das Betriebssystem genau die Vorgänge ausführen, die wir brauchen. Die entsprechende Routine kann für uns die komplizierte Arbeit erledigen. Außerdem hat der Einsatz dieser Routine den Vorteil, daß unser Programm in allen Modi funktioniert. Die Routine arbeitet auf die gleiche Weise wie der VDU-Befehl im Acorn-BASIC. Der folgende VDU-Befehl bildet beispielsweise einen einzigen Punkt auf dem Bildschirm ab:

```
VDU25,68,300;700;
```

Die Zahlen 300 und 700 geben hier die x- und y-Koordinaten an. Vom Maschinencode aus läßt sich der VDU-Befehl über den Systemaufruf von OSWRCH ansprechen. Für jede Zahl, die im Akkumulator gespeichert wird, muß der Aufruf wiederholt werden. Da der Akkumulator nur jeweils ein Byte enthalten kann, werden die x- und y-Koordinaten wie folgt in das LO-Byte HI-Byte-Format umgewandelt:

```
VDU25,68,44,1,188,2
```

Im Maschinencode muß für den gleichen Vorgang die OSWRCH-Routine sechsmal aufgerufen werden. Da der Vektor für die Anfangsadresse in Speicherstelle &FFEE untergebracht ist, wird die Routine mit JSR &FFEE angesprochen. Mit dieser Methode lassen sich auch alle anderen VDU-Befehle aufrufen. In unserem Programm ist OSWRCH mehrfach eingesetzt. Sie hat zwar keine Auswirkung auf die Register X, Y und A, verändert jedoch den Inhalt des Übertragsflags. Wenn der Übertrag noch benötigt wird, muß der Inhalt dieses Flags vor jedem Aufruf von OSWRCH gespeichert werden. Bei der ROL-Routine tritt genau dieser Fall ein. Der Übertrag läßt sich am einfachsten erhalten, wenn das Prozessor-Status-Register vor dem Aufruf von OSWRCH auf den Stapel geschoben (PHP) und danach wieder heruntergezogen (PLP) wird.

Sehen wir uns die Struktur des Maschinencodes genauer an. Die Analyse der Sprite-Daten und das Plotten erledigt die Unterroutine SPRPLT, die im Quellcode bei Zeile 890 anfängt. Diese Routine definiert zunächst die Plotmethode als „Exclusive OR", die dem GCOL-Befehl des Acorn-BASIC entspricht. Da damit die geplotteten Punkte durch einfaches Überschreiben gelöscht werden können, bleiben die Bildschirmdaten unterhalb des Sprites erhalten. Am Anfang des Maschinencodes werden mit einer absoluten Bewegung die linke obere Ecke des Sprites positioniert und danach die drei in einer Zeile liegenden Bytes der Sprite-Daten mit der Rotationsmethode analysiert.

Der Faktor für die horizontale Ausdehnung (XSCALE) gibt – abhängig vom aktuellen Inhalt des Übertragsflags – die Distanz an, über die eine relative Bewegung ausgeführt werden soll. Am Ende jeder Zeile von drei Bytes wird wiederum eine absolute Bewegung auf die gleiche x-Koordinate ausgeführt, die die linke obere Ecke des Sprites anzeigt. Die y-Koordinate wird dabei jedoch von dem Faktor für die vertikale Ausdehnung (YSCALE) verändert. Dieser Vorgang wird so lange wiederholt, bis alle 63 Bytes analysiert sind.

Die SPRPLT-Routine wird an zwei Stellen eingesetzt. Sie löscht zunächst das alte Sprite durch Überschreiben und plottet danach das neue Sprite. Die Koordinaten des neuen Sprites werden nach der Darstellung an OLDX und OLDY übergeben, und die Routine wird für den nächsten Aufruf vorbereitet.

Aufruf der Maschinencoderoutine durch BASIC-Befehle

Auch wenn die Funktionsweise der Maschinencoderoutine nicht bekannt ist, läßt sie sich leicht von BASIC aus einsetzen. Hier die notwendigen Schritte dazu:

1) Entwerfen Sie Ihr Sprite und setzen Sie die Daten mit DATA-Befehlen an die dafür vorgesehene Stelle des Programms.

2) Legen Sie die Anzeigeart fest.

3) Setzen Sie die Werte von XSCALE und YSCALE, wie in Zeile 1870 gezeigt.

4) Setzen Sie die logische Farbe des Sprites, wie in Zeile 1890 gezeigt.

5) Legen Sie die x- und y-Koordinaten der Position fest, an der das Sprite erscheinen soll, und wandeln Sie die absoluten Koordinaten mit dem Ablauf zwischen den Zeilen 2010 bis 2060 in das LO-Byte/HI-Byte-Format.

6) Rufen Sie die Routine mit der Anweisung CALL SPRITE auf.

Die Maschinencoderoutine kann – wie hier gezeigt – in das BASIC-Program eingegliedert werden. Die Assemblierung läßt sich verhindern, wenn Zeile 260 folgendermaßen geändert wird:

```
FOR opt%=0 TO 2 STEP 2
```

Die Routine kann auch nach der Assemblierung (d. h. nach dem ersten Ablauf) mit dem Befehl *SAVE gespeichert werden. Merken Sie sich dabei aber Anfangs- und Endadresse des Codes, die beim Anzeigen des Assemblerlistings dargestellt werden.

Das in 63 DATA-Bytes definierte Sprite kann mit den Cursortasten über den Bildschirm bewegt werden. Die Bewegung ist zwar relativ langsam, da die ROM-Routine OSWRCH eingesetzt wird, doch dafür arbeitet das Programm in allen Modi. Beim Aufruf des Programms erscheint auf dem Bildschirm zunächst das Assemblerlisting und dann die Grafik.

Acorn-Sprite

```
10 REM **** BBC SPRITES ****
20 :
30 REM ** SET UP ZERO PAGE VARIABLES **
40 TYPE   =&70
50 OLDXLO =&71:?OLDXLO=0
60 OLDXHI =&72:?OLDXHI=0
70 OLDYLO =&73:?OLDYLO=0
80 OLDYHI =&74:?OLDYHI=0
90 NEWXLO =&75
100 NEWXHI =&76
110 NEWYLO =&77
120 NEWYHI =&78
130 XSCALE =&79
140 YSCALE =&7A
150 logcol =&7B
160 ROW    =&7C
170 YTMPLO =&7D
180 YTMPHI =&7E
190 XTMPLO =&7F
200 XTMPHI =&80
210 :
220 HIMEM=HIMEM-150
230 SPRDAT =HIMEM+1
240 OSWRCH =&FFEE
250 DIM MC% &01FF
260 FOR opt% =0TO 3 STEP 3
270   P% =MC%
280   [
290   OPT opt%
300   \ **** MOVE TO OLD X,Y ****
310   \
320   .SPRITE  LDA #25
330            JSR OSWRCH
340            LDA #68
350            JSR OSWRCH
360            LDA OLDXLO
370            STA XTMPLO
380            JSR OSWRCH
390            LDA OLDXHI
400            STA XTMPHI
410            JSR OSWRCH
420            LDA OLDYLO
430            STA YTMPLO
440            JSR OSWRCH
450            LDA OLDYHI
460            STA YTMPHI
470            JSR OSWRCH
480   \
490   \ **** RUBOUT OLD SPRITE ****
500   \
510            JSR SPRPLOT
520   \
530   \ **** MOVE TO NEW X,Y ****
540   .NEWMOV    LDA #25
550            JSR OSWRCH
560            LDA #68
570            JSR OSWRCH
580            LDA NEWXLO
590            STA XTMPLO
600            JSR OSWRCH
610            LDA NEWXHI
620            STA XTMPHI
630            JSR OSWRCH
640            LDA NEWYLO
650            STA YTMPLO
660            JSR OSWRCH
670            LDA NEWYHI
680            STA YTMPHI
690            JSR OSWRCH
700   \
710   \ **** PLOT NEW SPRITE ****
720   \
730            JSR SPRPLOT
740   \
750   \ **** TRANSFER NEW X,Y TO OLD X,Y ****
760            LDA NEWXLO
770            STA OLDXLO
780            LDA NEWXHI
790            STA OLDXHI
800            LDA NEWYLO
810            STA OLDYLO
820            LDA NEWYHI
830            STA OLDYHI
840   \
850   \ **** RETURN TO BASIC ****
860   \
870            RTS
880   \
890   \ **** SPRITE PLOTTING SUBROUTINE ****
900   \
910   \ ** SET EXCLUSIVE OR PLOT **
920   .SPRPLOT LDA #18
930            JSR OSWRCH
940            LDA #3
950            JSR OSWRCH
960            LDA logcol
970            JSR OSWRCH
980   \
990   \
1000  \ ** INITIALISE COUNTS **
1010  \ ** X COUNTS BYTES, Y COUNTS BITS**
1020  \ ** ROW COUNTS ROWS OF THREE BYTES **
1030          LDX #&00
1040  .NEWROW LDA #&00
1050          STA ROW
1060  \
1070  .BYTE   LDY #&09
1080  .BIT    LDA #65
1090          STA TYPE
1100          ROL SPRDAT,X
1110          PHP          \STORE CARRY ON STACK
1120          BCS DOPLOT
1130          LDA #64
1140          STA TYPE
1150  \ ** VDU PLOT COMMAND **
1160  .DOPLOT LDA #25
1170          JSR OSWRCH
1180          LDA TYPE
1190          JSR OSWRCH
1200          LDA XSCALE
1210          JSR OSWRCH
1220          LDA #&00
1230          JSR OSWRCH
1240          JSR OSWRCH
1250          JSR OSWRCH
1260  \ ** END OF PLOT COMMAND **
1270          PLP          \RETRIEVE CARRY
1280          DEY
1290          BNE BIT
1300  \ ** IF BYTE FINISHED **
1310          INX
1320          CPX #63
1330          BEQ FINISH
1340  \ ** CHECK FOR END OF ROW **
1350          INC ROW
1360          LDA ROW
1370          CMP #3
1380          BNE BYTE
1390  \ ** IF END OF ROW SUBTRACT YSCALE FROM Y **
1400  \
1410          LDA YTMPLO
1420          SEC
1430          SBC YSCALE
1440          STA YTMPLO
1450          BCS NOSUB
1460          DEC YTMPHI
1470  \ ** ABSOLUTE MOVE TO START OF NEXT ROW **
1480  \
1490  .NOSUB  LDA #25
1500          JSR OSWRCH
1510          LDA #68
1520          JSR OSWRCH
1530          LDA XTMPLO
1540          JSR OSWRCH
1550          LDA XTMPHI
1560          JSR OSWRCH
1570          LDA YTMPLO
1580          JSR OSWRCH
1590          LDA YTMPHI
1600          JSR OSWRCH
1610  \
1620  \ ** NEXT ROW **
1630          JMP NEWROW
1640  \
1650  \ ** END OF SUBROUTINE **
1660  .FINISH RTS
1670  \
1680 ]
1690 NEXT
1700 :
1710 REM **** BASIC PROGRAM STARTS HERE ****
1720 :
1730 REM ** READ SPRITE DATA **
1740 FOR address = SPRDAT TO SPRDAT+62
1750   READ data:?address = data
1760 NEXT address
1770 :
1780 REM **** SET M/C PARAMETERS ****
1790 :
1800 MODE1
1810 GCOL0,129
1820 CLG
1830 ?XSCALE =4:?YSCALE =4
1870 ?logcol=1
1880 :
1890 X=700:Y=800
1900 PROCCOORDS (X,Y)
1910 CALL SPRITE
1920 :
1930 REM **** WAIT FOR CRSR KEYS ****
1940 FOR S=0 TO 1 STEP 0
1950 PROCKEYS
1960 PROCCOORDS (X,Y)
1970 CALL SPRITE
1980 NEXT S
1990 END
2000 :
2010 DEF PROCCOORDS (X,Y)
2020 XH=X DIV 256:XL=X MOD 256
2030 YH=Y DIV 256:YL=Y MOD 256
2040 ?NEWXLO=XL:?NEWXHI=XH
2050 ?NEWYLO=YL:?NEWYHI=YH
2060 ENDPROC
2070 REM **** SCAN KEYBOARD ****
2080 DEF PROCKEYS
2090 LOCAL LT,ZZ:LT=2
2100 FOR ZZ=0 TO 1 STEP 0
2110 IF INKEY(-58) THEN Y=Y+50:ZZ=LT
2120 IF INKEY(-42) THEN Y=Y-50:ZZ=LT
2130 IF INKEY(-26) THEN X=X-50:ZZ=LT
2140 IF INKEY(-122) THEN X=X+50:ZZ=LT
2150 NEXT ZZ
2160 ENDPROC
2170 REM **** SPRITE DATA ****
2180 DATA 255,0,255,254,0,127,252,0,63
2190 DATA 240,0,15,232,0,23,228,0,39
2200 DATA 194,0,67,129,24,129,0,189,0
2210 DATA 0,102,0
2220 DATA 0,102,0
2230 DATA 0,102,0
2240 DATA 0,189,0,129,24,129,194,0,67
2250 DATA 228,0,39,232,0,23,240,0,15
2260 DATA 252,0,63,254,0,127,255,0,255
```

Kleine Japaner

Taschencomputer werden für viele ernsthafte Anwendungen verwendet. Ingenieure benutzen sie zwischendurch, um an Ort und Stelle Berechnungen auszuführen. Börsenmakler ermitteln mit ihnen zu erwartende Gewinne, und Verkäufer können damit genaue Kostenkalkulationen durchführen. Wir vergleichen mehrere Rechner aus dem Casio-Angebot.

Programmierbare Taschenrechner konnen auf vielerlei Art verwendet werden, doch die eigentlichen Taschencomputer haben einen Vorteil, weil sie sowohl mit BASIC arbeiten als auch Text verarbeiten können. Taschencomputer sind ideal für Aufgaben, bei denen regelmaßig dieselben Formeln benotigt werden. Diese Berechnungen wiederholen sich oft und sind zeitraubend. Benutzer von Taschencomputern konnen ihre eigenen Programme schreiben, um die jeweilige Eingabe zu vereinfachen.

Der preiswerteste Rechner der Casio-Reihe ist der FX-720P. Er ist 165x85x15 mm klein und wiegt 210 Gramm. Die Tastatur des FX-720P ist halb so groß wie eine normale Tastatur.

Mit jeder Buchstabentaste können zwei weitere Zeichen (wie etwa Punkte und Kommas) erzeugt werden, sowie BASIC-Befehle, wenn gleichzeitig die Shift- oder Funktions-Taste betätigt wird. Anders als sonst üblich werden die Tasten nicht gleichzeitig mit dem Auslosen der Buchstabentaste gedruckt, sondern vor Drukken der betreffenden Taste betätigt. So kann man den Rechner mit einer Hand bedienen.

Beim FX-720P wurde eine LCD-Anzeige eingebaut, auf der pro Zeile zwolf Zeichen zu sehen sind. Längere Zeilen sind ebenfalls darstellbar. Dazu werden diese über Cursortasten nach links oder rechts gerollt. Seitlich des Schirms befindet sich ein kleines Rad, mit dem die Display-Helligkeit eingestellt werden kann. Wird der Computer sechs Minuten lang nicht benutzt, schaltet er den Bildschirm automatisch ab, um Energie zu sparen.

Geliefert wird der FX-720P mit nur zwei KByte Speicherkapazität. Allerdings ist der gesamte RAM-Speicher in Steckmodulen untergebracht, die beim Entfernen aus der Maschine die Daten behalten. Für die Erweiterung werden 2-KByte- und 4-KByte-Cartridges angeboten.

Da auf den Cartridges nur fur wenige Daten Platz ist, sind sie schnell voll – nicht zuletzt deshalb, weil der Computer gleichzeitig bis zu zehn BASIC-Programme speichern kann. Um eines eingeben zu konnen, muß ein spezieller Mode gewählt werden, ein weiterer ist notig, um das Programm laufen zu lassen. Das FX-720P-BASIC ist uberraschend gut. Es enthalt fast alle Standard-Befehle und -Funktionen mit Ausnahme von CHR$ und ASC. Mit dem einfachen BEEP-Befehl lassen sich zwei verschiedene Töne erzeugen.

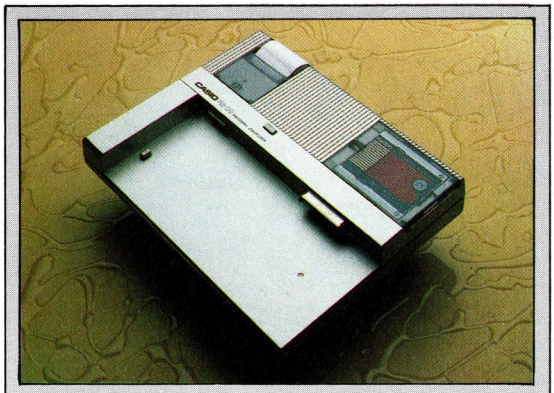
Der FX-720P gilt als der „niedrigste" Rechner der Casio-Reihe. Fast alle anderen Modelle haben verbesserte Funktionen. Eine Besonderheit hat dieser Computer jedoch. Das ist die „Data Bank", ein kleines Datenbank-Programm, in dem Informationen wie Namen und Adressen, Ausgaben oder Termine leicht gespeichert und nach verschiedenen Kriterien gesucht werden konnen. Damit wird das Gerat zum direkten Konkurrenten des „Organiser" von Psion. Der FX-720P kostet jedoch weniger und ist vielseitiger.

FX-750P für Ingenieure

Während der FX-720P mehr fur Geschäftsleute gedacht ist, steht mit dem FX-750P ein Modell zur Verfugung, das fur Wissenschaftler und Ingenieure geeignet ist. Er bietet eine 24-Zeichen-Darstellung und eine leicht verbesserte Tastatur. Wie der FX-720P verfügt er über zwei KByte Speicherkapazitat und ist ausbaufahig.

Er hat namlich einen Schacht fur ein zweites

Der FX-720P Taschencomputer von Casio verfügt über 2K Speicher, eine einzeilige Flüssigkristall-Anzeige und eine komplette alphanumerische Tastatur. Erweiterungsmodule lassen sich in das System stecken.



FA-20-Interface
Jeder Casio-Taschencomputer kann mit einem Interface dieser Art ausgerüstet werden. Der FA-20 ist mit aufladbaren Batterien und einem Trafo ausgestattet.

Modul. Damit kann die Speicherkapazität des Kleincomputers bis auf acht KByte erweitert werden. Der Rechner ist 185 mm lang und wiegt lediglich 250 Gramm. Anders als beim FX-720P müssen die Shift- und Funktions-Tasten jeweils gleichzeitig mit einer Buchstabentaste gedrückt werden.

Zehn Tasten sind mit wichtigen Konstanten belegt: Lichtgeschwindigkeit, Anziehungskraft der Erde, Plancksche Konstante, Boltzmanns Konstante, Avogadrosche Konstante, Elementargewicht, Atomgewicht, Elektronenmasse, Gravitationskonstante sowie Molarvolumen. (Physiker und Chemiker benutzen diese Konstanten ständig bei Berechnungen.)

Jüngstes und drittes Casio-Produkt ist der PB-700. Die LCD-Darstellung erfolgt mit 20 Zeichen auf vier Zeilen. Er hat vier KByte Speicherkapazität. Der Speicher kann bis auf 16 KByte durch Module erweitert werden, die auf der Unterseite angebracht werden. Er ist fast 200 mm lang und wiegt 315 Gramm.

Die Tastatur des PB-700 ist der des FX-750P ähnlich, nur daß er statt einer Funktionstaste mit einer „Caps"-Taste ausgestattet ist. Drückt man diese, stellt der Rechner kleine Buchstaben dar. Das BASIC ist dem des FX-750P ähnlich, verfügt aber über zusätzliche Befehle, um auf dem LCD Grafik zu erzeugen. Die Bildschirmauflösung beträgt 32 mal 160 Punkte.

BASIC-Bestandteile sind Befehle, die das Zeichnen von Grafiken ermöglichen, wenn der Printer/Plotter und das als FA-10 bezeichnete Cassetteninterface verwendet werden. Beide werden an den Computer gesteckt. Text und Grafik werden auf 115 mm breitem Papier vierfarbig dargestellt. Mit den BASIC-Befehlen können auf dem Printer/Plotter Linien, Kreise sowie Achsen zur Kurvendarstellung gezeichnet werden.

Der FA-10 verfügt über ein integriertes Cassetteninterface, das verwendet werden kann, um auf einem normalen Cassettenrecorder Programme und Daten zu speichern. Unter Berücksichtigung der geringen Speicherkapazität des Rechners ist das eine sehr sinnvolle Möglichkeit.

Zusätzlich läßt sich die Kombination mit einem winzigen Recorder ausstatten, so daß man über ein komplettes transportables Computersystem verfügt. Ferner steht ein separates Centronics-Interface zur Verfügung, mit dem der Betrieb von Standard-Druckern am PB-700 möglich ist.

Für den FX-720P und den FX-750P stehen ebenfalls Cassetteninterfaces und Drucker im Programm. Beim FX-720P wird der FP-12S-Drucker verwendet. Eine separate Einheit, der FA-3, übernimmt die Funktion eines Interfaces für einen Cassettenrecorder. Der FX-750P arbeitet mit dem FA-20, einem winzigen Drucker, kombiniert mit Cassetten-Interface. Er hat Platz für die Aufnahme einer Erweiterungskarte und wird mit wiederaufladbaren Batterien, einem Trafo und einem Schutzgehäuse geliefert.

Alle drei Computer sind mit umfangreichen Handbüchern ausgestattet, die alle wichtigen Informationen liefern. Offensichtlich aber handelt es sich um eine Übersetzung aus dem Japanischen, die teilweise sehr schlecht formuliert ist und so besonders den Anfänger geradezu verwirrt.

Ein weiteres Problem ist, daß es kaum Software für diese Computer gibt. Besitzer müssen ihre eigenen Programme schreiben oder Musterprogramme aus den Begleitbüchern eingeben, die aber für einen wirklich professionellen Einsatz kaum ausreichen dürften.

Typ	Maße	Gewicht	Speicher	Darstellung	Besonderheiten
FX-720P	165x35x15mm	210g	2K	1x12 Zeichen	Eingebaute Datenbank
FX-750P	185x85x15mm	250g	2K	1x24 Zeichen	Zweiter Modulschacht, physikalische Konstanten fest programmiert, erweitertes BASIC
PB-700	200x85x15mm	315g	4K	4x20 Zeichen	Grafikfähigkeit, Printer/Plotter verfügbar

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.

Drop-In = Störsignal
Von einem Drop-In spricht man, wenn auf einem Magnetträger ein Zeichen erscheint, wo keines hingehört. Die Ursache für Drop-Ins sind meist defekte Stellen in der Magnetschicht. Genau wie Drop-Outs bringen auch einzelne Drop-Ins nicht umbedingt den Datenverkehr durcheinander, weil die meisten Betriebssysteme Lesekontrollen beinhalten, die gelegentliche Bitfehler identifizieren und zum Teil auch korrigieren können.

Drop-Out = Signalausfall
Ein Drop-Out tritt auf, wenn sich aus der Beschichtung einer Diskette oder eines Magnetbands winzige Partikel lösen – Billigcassetten sind dafür besonders anfällig. Durch Drop-Outs kann ein gespeichertes Programm völlig unbrauchbar werden. Deshalb sollten Magnetträger stets mit besonderer Sorgfalt behandelt werden.

Disketten und Cassetten, die zu Drop-Outs neigen, werfen Sie am besten weg. Aber was machen Sie bei einer Winchesterplatte, die ein paar tausend Mark gekostet hat? Die Frage löst für Sie das DOS, das bei Festplatten wesentlich raffinierter arbeitet als bei Disketten: Wenn durch Fehler beim Prüflesen ein „Bad Block" erkannt wird, merkt sich das DOS dessen Lage und sperrt ihn für die weitere Benutzung.

Dump = Speicherabzug
Mit einem Dump wird der Inhalt eines Speichers auf dem Bildschirm oder durch Ausdrucken auf Papier sichtbar gemacht. Ein Binary Dump liefert die Information Byte für Byte binär verschlüsselt, ein Hex Dump hexadezimal usw. Ein Dump ist nicht nur nützlich, wenn Sie ein Maschinencode-Protokoll für die Eingabe in Steuerungssoftware brauchen, sondern hilft auch bei der Fehlersuche.

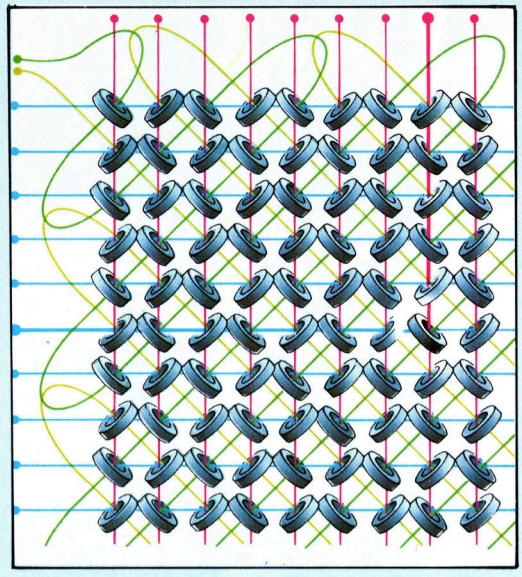
Duplex = Duplex
Beim Duplex-Betrieb können auf einem Kanal gleichzeitig in beiden Richtungen Informationen übertragen werden. Von Halbduplex spricht man, wenn der Hin- und Rückverkehr nur im Wechsel von Senden und Empfangen möglich ist. Viele Gegensprechanlagen und Funkgeräte arbeiten im Halbduplex-Betrieb, wobei jeweils von Sprechen auf Hören umgeschaltet wird. Auch der Datenverkehr zwischen Computern auf Telefonleitungen erfolgte früher halbduplex, und einen Voll-/Halbduplex-Umschalter haben auch heute noch einige Modems – der Vollduplex-Betrieb erfordert eine größere Übertragungs-Bandbreite.

Gleichzeitiges Senden und Empfangen von Informationen wird selten praktiziert. Der Vollduplexbetrieb erlaubt in Form einer simultanen Rückmeldung über den Datenempfang aber eine wirksame Übertragungskontrolle. Werden Fehler festgestellt, etwa infolge eingekoppelter Störungen, kann der Empfänger die entsprechenden Zeichen noch einmal anfordern.

Dynamic RAM = Dynamisches RAM
Bei Microcomputern werden statische und dynamische RAMs eingesetzt. Beide sind flüchtig, der Inhalt geht also bei Stromabschaltung verloren. Ein statisches RAM besteht aus zahllosen Flipflops (eins für jedes Speicherbit) und benötigt für den Verkehr mit dem Prozessor keine externe Beschaltung.

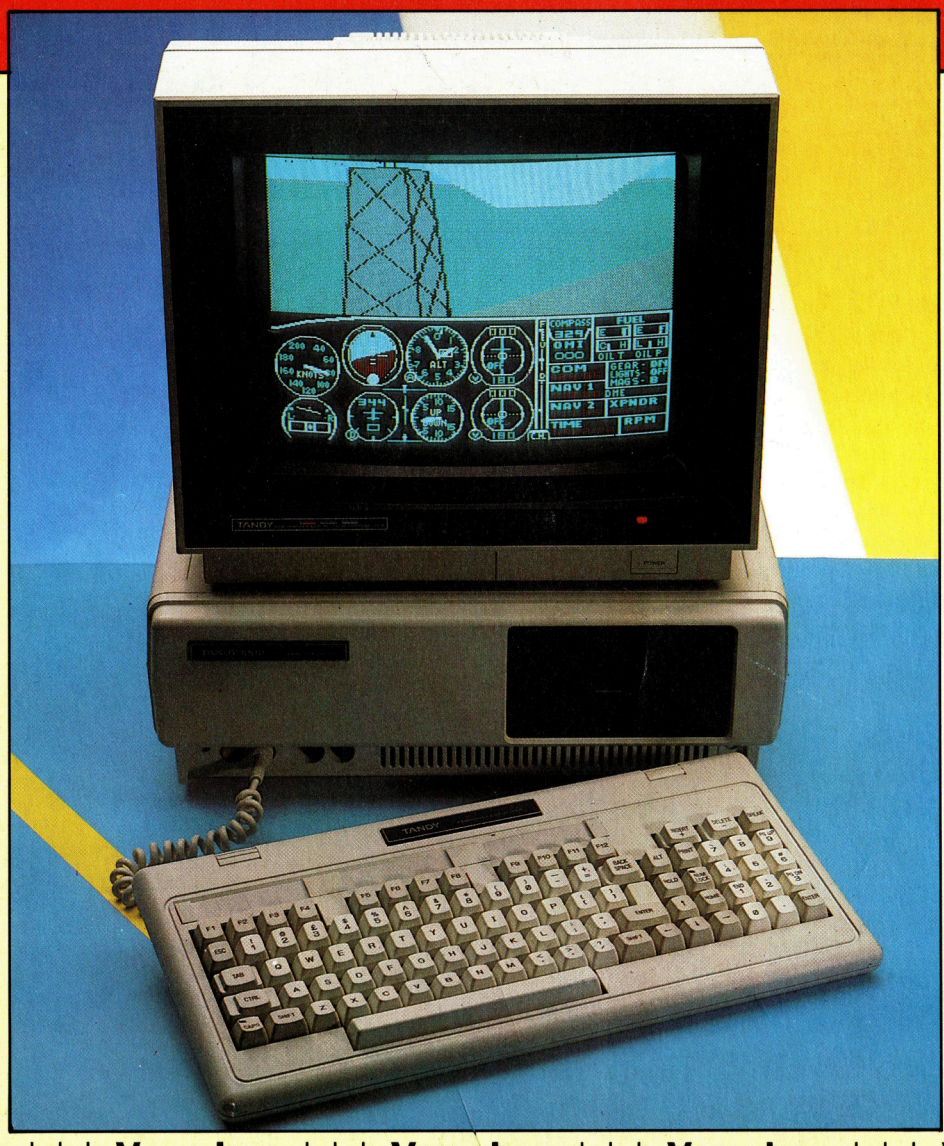
Dynamische RAMs erfordern mehr Aufwand, sind aber schneller, billiger und bieten in einem Baustein mehr Kapazität. Für jedes Bit ist ein kleiner Kondensator (mit nachgeschaltetem Transistor) vorgesehen, der je nach Ladezustand eine 1 oder eine 0 darstellen kann. Da die Ladung abfließt, muß das RAM durch eine spezielle Schaltung alle paar Millisekunden „aufgefrischt" werden. Die „Refresh"-Schaltkreise werden neuerdings gleich in die RAM-Bausteine integriert. Das erleichtert den Systemaufbau erheblich.

Der Kollektorstrom eines Transistors mit geöffnetem Gehäuse ändert sich entsprechend der Belichtung. Auch ältere Ausführungen dynamischer RAMs zeigen eine solche Lichtempfindlichkeit und eignen sich daher zum Aufbau von preiswerten visuellen Sensoren für Roboter.



Ferritkernspeicher
RAMs sind heute durchweg Halbleiterspeicher, aber lange Zeit waren die hier gezeigten Kernspeicher mit kleinen Ferritringen die preisgünstigsten und schnellsten Speicher für direkten Zugriff. 1 und 0 werden durch entgegengerichtete Magnetisierung der Kerne definiert, die auf Schreib- und Lesedrähte aufgefädelt sind.

Bildnachweise
1009: Steve Cross
1010, 1011, 1012: Kevin Jones
1015, 1020, 1024, 1025, 1028, 1029, 1032: Liz Dixon
1016: Ian McKinnell, Quicksilva, Kevin Jones, Your Spectrum
1017, 1018, 1021, 1022, 1023, 1035, 1036: Chris Stevens
1019, 1027, 1031: Ian McKinnell



Mit dem Tandy 1000 erhofft sich die Tandy Corporation eine Rückeroberung von Marktanteilen. Der Tandy 1000 ist IBM PC-kompatibel und verfügt – wie der IBM PC – über einen 8088-Prozessor und die 5¼-Zoll-Diskettenlaufwerke. Das abgebildete Modell ist mit einem Doppellaufwerk und einem zusätzlichen 128 KByte Arbeitsspeicher ausgerüstet. Zudem wurden in das Gerät weitere, wichtige Schnittstellen eingebaut.

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

computer kurs

Heft 38

Roboter — selbst gebaut?

Mit dem Computing-Baukasten der Firma Fischer kann man in wenigen Stunden die verschiedensten Peripherie-Geräte zusammenbauen.

Knacken und Hacken

In jüngster Zeit häufen sich die illegalen Einbrüche in Datenbanken von Großkonzernen. Die Schäden gehen teilweise in die Millionen.

Sets und Mengen

Sie bieten zahlreiche Möglichkeiten, Daten in PASCAL übersichtlich zu strukturieren.

Chiffren-Jagd

Das Epyx-Spiel „Impossible Mission“ fordert zu einem Wettlauf gegen die Zeit heraus.