86-6

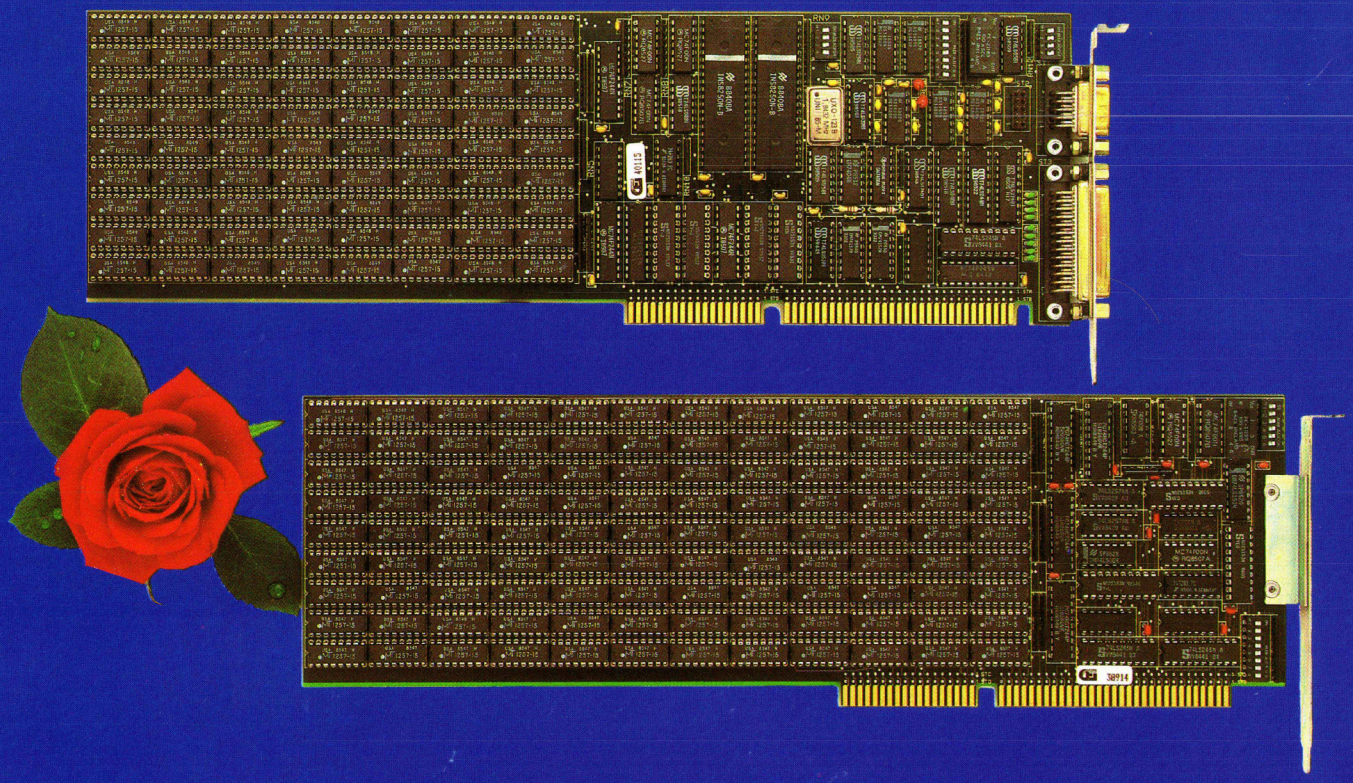
MIKRO + KLEIN COMPUTER

Programm zur schnellen Ellipsendarstellung

Laserprinter Kyocera im Test

DAS SCHWEIZER KLEINCOMPUTER-MAGAZIN ISSN 0251-0006 DM 8.– / S 60 / Fr. 7.50

Für den neuen XT - 286



MemAT + und Multi 3 AT für den neuen XT - 286 und die AT's

Qualität und Funktionalität aus deutscher Entwicklung und Fertigung sichern Ihnen den entscheidenden Vorsprung. Durch vorbildliches Design können alle unsere AT Karten problemlos in den neuen XT - 286 Rechnern eingesetzt werden.

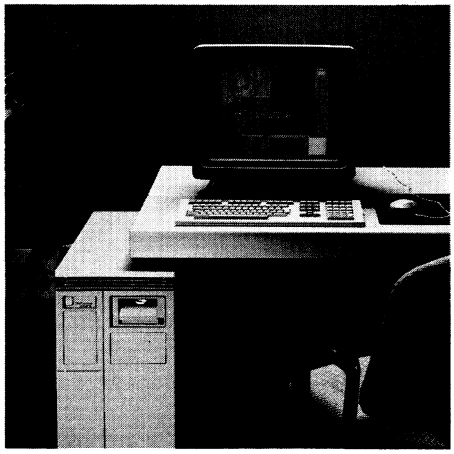
MemAT + ist die 3 MB Speicherkarte für den XT - 286 und alle AT's.

Multi 3 AT ist die Multifunktionskarte mit 2 MB Speicher, 2 seriellen Schnittstellen und einem Druckeranschluß.

Kostenlos erhalten Sie zu unseren Produkten MemAT + und Multi 3 AT den Infosys Memory Manager, zur Simulation von Expanded Memory (Lotus Intel Spezifikation), sowie weitere sinnvolle Softwareprodukte.

Infosys GmbH, Am Kümmerling 2, 6501 Bodenheim, Telefon: 0 61 35 / 30 81
Infosys AG, Industriestraße 57, CH - 8152 Glattbrugg, Telefon: 01 - 8 10 77 10

DAS LESEN SIE IN DIESER NUMMER



Mit ihren Farbgrafikfähigkeiten kennzeichnet die VAXstation II/GPX einen weiteren Fortschritt bei den technischen Arbeitsstationen. Auf der system-technologischen Grundlage der MicroVAX II verfügt die VAXstation II/GPX in bezug auf Geschwindigkeit und Leistung über aussergewöhnliche Merkmale. Im Vordergrund der Konzeption der VAXstation II/GPX steht ein VLSI-Grafikcoprozessor, welcher der Zentraleinheit der MicroVAX II die gesamte Text- und Grafikarbeit abnimmt und hervorragende Grafiken erstellt. Die VAXstation II/GPX hat die gleichen Mehrfenster-, Multitasking und hochauflösende Grafikfunktionalität wie die VAXstation II, doch zudem farbig und schneller. Zusätzlich verfügt sie über die Flexibilität, die Leistung und die Netzwerkfähigkeiten, welche die VAX-Rechnerfamilie von Digital Equipment kennzeichnen. Die VAXstation II/GPX unterstützt einen weiten Bereich von Anwendungserfordernissen, der vom Layout von VLSI- und PC-Platinen über mechanisches CAD bis zur Bildverarbeitung, technische Dokumenten- und Wetterkartenerstellung reicht. Durch die Kombination fortschrittlicher Hard- und Softwaretechnologien verbindet die VAXstation II/GPX die Multitaskingleistung der VAX, eindrucksvolle Rechenleistung und anspruchsvolle grafische Fähigkeiten mit hoher Geschwindigkeit, so wie es für eine technische Farbarbeitsstation erforderlich ist. Info: W. Moor AG, Bahnstrasse 58, 8105 Regensdorf, Tel. 01/840'66'44.

Ausgabe Dezember 1986
Erscheint 6mal pro Jahr
8. Jahrgang

A B A C U S

MULTI-USER

LOHNBUCHHALTUNG

Die Kunst der Perfektion

● Manuelle Lohnabrechnungen (direkt am Bildschirm berechnet) ○ Automatische Lohnabrechnungen ○ Lohnabrechnungen frei korrigierbar ○ Bruttolohn-, Nettolohn- und Lohnartenberechnung ○ Fixer Nettolohn (Iteration) ○ 15 Auszahlungsperioden à je 99 Unterperioden ○ Dialog-Verarbeitung (Online) ○ Vorerfassung (Batch) ○ Auszahlungslisten, Münzlisten, Bordereaux usw. ○ Lohnjournale ● Lohnausweise ○ AHV/ALV-Abrechnung ○ NBU/SUVA-Abrechnung ○ BVG/PK-Abrechnung ○ FAK-Abrechnung ○ Frei definierbare Abrechnungen ○ Frei definierbare Abrechnungsperioden ● Frei definierbare Lohnarten ○ Bedingte Lohnarten ○ Abhängige Lohnarten ○ Halbgraphische Definition der Lohnarten ○ Monatslohn, Wochenlohn, Stundenlohn usw. beliebig kombinierbar ○ Akkordlöhne, gestaffelte Provisionstarife, Ferienlohnberechnung usw. ● Personalstammdaten ○ Frei definierbare Personalstammfelder ○ Kumulierbare Personalstammfelder ○ Firmastammdaten ○ Frei definierbare Firmenstammfelder ○ Betriebe und Abteilungen ● Kostenstellen pro Arbeitnehmer, pro Lohnart usw. ○ Kostenstellen-Präferenzmatrix ○ Autom. Quellensteuerabzüge ○ DTA Datenträgeraustausch ○ Personalinformationssysteme (PIS) ○ Mandantenfähigkeit ○ Multi-User ● Integrierte Hilfsbildschirme ○ Zoom- und Window-Technik ○ FIDES- und SUVA-geprüft ● Baubranche ○ Handel und Gewerbe ○ Hotel- und Gastgewerbe ○ Produktion und Fabrikation ○ Dienstleistungs- und Treuhandsektor ● PC-DOS/MS-DOS 2.x und höher ○ Diverse Netzwerke ○ XENIX ○ UNIX V.3, DEC VAX VMS in Vorbereitung.

ABACUS RESEARCH AG

Rorschacherstrasse 170, Postfach 117, 9001 St.Gallen, Telefon 071 25 93 25, Telex 71 775 ABAC-CH, Telefax 071 253 860

Betriebswirtschaftliche Software-Entwicklung, Beratung, Schulung

Gmeiner ASG

Programmieren mit dem IBM-PC (5. Teil)

Welcher Programmierer träumt nicht davon, ein Programm zu schreiben, welches einen frei definierbaren dreidimensionalen Körper wie z.B. einen Würfel, ein Prisma, einen Kegel oder eine Kugel auf dem Bildschirm um die x-, y- oder z-Achse eines räumlichen Koordinatensystemes in Echtzeit dreht. Animation von Computergrafik ist das Zauberwort.

Marcel Sutter

Jede Bewegung eines Bildes respektive eines Körpers kann durch folgenden Algorithmus beschrieben werden:

1. Wähle die Startkoordinaten X1 und Y1 einer Ecke der zu bewegenden Figur.
2. Wiederhole die Anweisungen 2.1. bis 2.4.
   - 2.1. Zeichne die Figur an der Stelle P(X1,Y1).
   - 2.2. Berechne die neue Stelle P(X2,Y2).
   - 2.3. Lösche die Figur an der Stelle P(X1,Y1).
   - 2.4. Ersetze X1 durch X2 und Y1 durch Y2.

Dabei müssen die Anweisungen innerhalb der Schleife so schnell ablaufen, dass das Auge des Betrachters getäuscht wird und den diskreten Vorgang von Zeichnen und Löschen von Figuren als kontinuierlichen Prozess wahrnimmt.

Leider ist BASIC keine geeignete Sprache für schnelle Bildbewegungen. Falls die Figur nur aus wenigen Punkten besteht (n<10), dann schafft es der BASIC-Interpreter gerade noch. Wenn Sie aber eine ausgedehnte Figur bewegen wollen, ist der Interpreter überfordert. Bis die Figur mit zwei geschachtelten FOR...NEXT-Schleifen aufgebaut und nach einer gewissen Zeit wieder gelöscht ist, vergeht allzu viel Zeit. Die Bewegung des Bildes erfolgt ruckartig und nicht glatt.

Wer Programme in Maschinensprache schreiben kann, wird die Prozeduren «Figur zeichnen» und «Figur löschen» in Maschinensprache oder Assembler erstellen. In M+K 84-3 wurde ein BASIC-Programm veröffentlicht, mit dem ein frei wählbarer Körper auf dem Bildschirm eines Commodore CBM 4032 in real time gedreht wird. Die dabei verwendeten Prozeduren wurden in Maschinensprache für den Prozessor 6502 geschrieben und sind daher auf andere Computer wie z.B. den IBM-PC nicht übertragbar.

Damit auf Homecomputern auch allein mit BASIC Computerspiele programmiert werden können, bei denen es auf eine schnelle Bildbewegung ankommt, haben sich die Hersteller etwas einfallen lassen müssen. So wurden für den Apple II die «Shapes» und für den C-64 die «Sprites» erfunden. Mit ihrer Hilfe kann man kleine Figuren, deren Grösse je nach Computertyp fest vorgegeben ist, auf dem Bildschirm herumsausen lassen. Die Generierung und die Bewegung von Shapes und Sprites ist aber äusserst mühsam und verlangt vom Programmierer genaue Kenntnisse der Hardware und diverser exotischer POKE-Befehle. Wohl werden in den einschlägigen Computerzeitschriften immer wieder Hilfsprogramme in BASIC angeboten, die das Handling von Sprites erleichtern sollen, doch bleibt die Animation von Figuren mit Hilfe von Sprites und Shapes weiterhin eine Domäne der Computerfreaks.

Aber selbst wenn Sie sich im Umgang mit Shapes und Sprites sehr gut auskennen, können Sie immer noch nicht einen grösseren Körper um eine Raumachse rotieren lassen. Dieser Wunsch scheint daher für reine BASIC-Programmierer unerfüllbar zu sein.

Der Schreibende war daher angenehm überrascht, als er im IBM-BASIC die Grafikbefehle GET und PUT fand, welche in mittlerer und hoher Auflösung eine so schnelle Bildbewegung ermöglichen, dass tatsächlich ein beliebiger dreidimensionaler Körper allein mit einem BASIC-Programm, also ohne irgendeine Maschinenroutine und ohne irgendwelche PEEK- und POKE-Befehle, in Echtzeit bewegt und insbesondere um eine gewählte Raumachse gedreht werden kann. Der Vorgang wird vom Betrachter gerade noch als kontinuierlicher Prozess wahrgenommen.

Wir werden nun verschiedene Programme vorstellen, welche Würfel, Prismen, Zylinder, Kegel, Kegelstümpfe und Kugeln wahlweise um die x-, y- oder z-Achse rotieren lassen.

Bevor wir diese Programme entwickeln, wollen wir anhand von einfachen Beispielen zeigen, wie die Animation von Figuren mit BASIC realisiert werden kann.

1. Bewegung eines Pixels in niederer Auflösung (SCREEN 0)

Listing 1 zeigt ein einfaches BASIC-Programm, welches ein einzelnes Zeichen (im Programm ein kleines o) auf dem Bildschirm vom linken zum rechten Rand und dann wieder zurück zum linken Rand bewegt. Die Hin- und Herbewegung erfolgt so lange, bis Sie irgendeine Taste niederdrücken.

Studieren wir das Programm:

In Zeile 100 werden die Farben gesetzt. Der Bildschirm ist blau, der Rahmen schwarz und die Zeichenfarbe gelb. Die Schleife FOR X=LINKS TO RECHTS STEP SCHRITT (Zeilen 200-260) bewegt das Zeichen von links nach rechts. Die Anweisung LOCATE 12,X : COLOR 14 : PRINT«o»;

Listing 1

```
100 SCREEN 0:COLOR 14,1,0:WIDTH 80:CLS
110 LINKS=1:RECHTS=80:SCHRITT=1:DAUER=5
120 :
200 FOR X=LINKS TO RECHTS STEP SCHRITT
210  LOCATE 12,X:COLOR 14 :PRINT"o";
220    FOR PAUSE=1 TO DAUER :NEXT PAUSE
230  LOCATE 12,X:COLOR 1:PRINT"o";
240    FOR PAUSE=1 TO DAUER :NEXT PAUSE
250  IF INKEY$<>"" THEN 400
260 NEXT X
270 :
300 FOR X=RECHTS TO LINKS STEP -SCHRITT
310  LOCATE 12,X:COLOR 14:PRINT"o";
320    FOR PAUSE=1 TO DAUER :NEXT PAUSE
330  LOCATE 12,X:COLOR 1:PRINT"o";
340    FOR PAUSE=1 TO DAUER :NEXT PAUSE
350  IF INKEY$<>"" THEN 400
360 NEXT X
370 GOTO 200
400 COLOR 14,1,1:CLS:END
```

zeichnet in der 12. Zeile und in der Spalte X des Schirms das Pixel hin. Die Anweisung LOCATE 12,X : COLOR 1 : PRINT«o»; löscht das Zeichen nach einer gewissen Zeitverzögerung wieder aus, indem das Pixel in der Bildschirmfarbe auf den Schirm gezeichnet wird.

Die Zeitverzögerung erfolgt durch die Leerschleife FOR PAUSE=1 TO DAUER : NEXT PAUSE.

Die Schleife FOR X=RECHTS TO LINKS STEP -SCHRITT (Zeilen 300-360) bewegt entsprechend das Zeichen von rechts nach links. Die Tastaturabfrage IF INKEY$<>«» THEN 400 in den Zeilen 250 und 350 kontrolliert, ob Sie eine Taste gedrückt haben und dadurch das Programm beenden wollen.

Als nächstes Beispiel programmieren wir ein einfaches Computerspiel. Auf dem Bildschirm erscheint zu Beginn ein rechteckiges Bassin mit braunem Rand. Im blauen Wasser werden zufällig 40 rosarote Wasserminen verteilt. Sie sind mit einem Asterix «*» gekennzeichnet. Irgendwo wird noch willkürlich ein kleines weisses Boot hingesetzt. Es ist mit dem Zeichen »0» markiert.

Durch Drücken der Taste Num Lock schalten Sie den Zifernblock auf der rechten Seite der Tastatur ein. Mit Hilfe der Zifferntasten 1 bis 9 können Sie das Boot kontinuierlich im Bassin herumdirigieren. Sobald Sie eine Taste antippen, wird die der Taste entsprechende Bewegungsrichtung so lange beibehalten, bis Sie erneut eine andere Taste antippen. Taste 5 lässt das Boot an seinem Ort verharren, Taste 1 bewegt es diagonal nach links unten, Taste 2 senkrecht nach unten usw.

Für jede geglückte Positionsveränderung des Bootes erhalten Sie einen Punkt gutgeschrieben. Sobald aber Ihr Boot auf eine Mine stösst, ist das Spiel aus und der Computer meldet Ihren Score, also das Punktetotal.

Solche Computerspiele waren in der Frühzeit der Kleincomputer (1975-1978) recht oft anzutreffen. Heute erwecken sie höchstens noch ein müdes Lächeln, da der Konsument von Computergames inzwischen recht anspruchsvoll geworden ist.

Bevor wir das Programm analysieren, müssen wir auf zwei wichtige Anweisungen im IBM-BASIC aufmerksam machen.

Bei Homecomputern wie z.B. dem C-64 erfolgt die Steuerung eines bewegten Pixels vornehmlich mit den beiden Befehlen

```
POKE BA, C
IF PEEK(BA)<>32 THEN ...
```

Dabei ist BA die Bildschirmadresse der momentanen Stelle des Zeichens und C die POKE-Codezahl des Zeichens.

Da die Bildschirmadressen von Gerät zu Gerät verschieden sind, können solche BASIC-Programme nicht direkt auf andere Computer übertragen werden.

Beim IBM-PC dienen die folgenden geräteunabhängigen Anweisungen dem gleichen Zweck

```
LOCATE ZEILE,SPALTE:COLOR FARBE:PRINT CHR$(N)
IF SCREEN(ZEILE,SPALTE)<>32 THEN ... ELSE ...
```

Die Funktion SCREEN(ZEILE,SPALTE) meldet den ASCII-Code desjenigen Zeichens, welches an der betreffenden Bildschirmstelle vorhanden ist, z.B. die Zahl 32, wenn die Stelle unbesetzt ist.

Listing 2

```
100 KEY OFF:WIDTH 80
110 COLOR 15,1,6 :CLS
120 RANDOMIZE(TIMER)
125 PRINT:PRINT
130 PRINT TAB(20);"Das Minen - Spiel"
140 PRINT TAB(20);"--------------------"
150 PRINT:PRINT:PRINT
160 PRINT"   Der Computer setzt zu Beginn 40 Wasserminen (*) und irgendwo ein"
165 PRINT"   Boot (0). Steuern Sie das Boot mit den Zifferntasten rechts aussen
170 PRINT"   durch das Gewirr der Minen. Jede Aenderung des Ortes des Bootes"
175 PRINT"   ergibt einen Punkt. Ständig nimmt die Zahl der Minen zu."
180 PRINT"   Wenn Sie in eine Mine hineinfahren, ist das Spiel zu Ende und der"
185 PRINT"   Computer meldet Ihre Punktezahl (score).
190 LOCATE 23,1:INPUT"Haben Sie die Num-Lock-Taste gedrückt (j/n).....";A$
195 IF A$<>"j" THEN RUN
196 CLS
197 :
200 FOR J=1 TO 40
210   ZEILE=INT(24*RND)+1
220   SPALTE=INT(80*RND)+1
230   LOCATE ZEILE,SPALTE:COLOR 12:PRINT CHR$(42);
240 NEXT
250 :
300 Y=INT(24*RND)+1:X=INT(80*RND)+1
310 IF SCREEN(Y,X)=42 THEN 300
320 LOCATE Y,X:COLOR 15:PRINT CHR$(79);
330 PUNKTE=0
340 FOR PAUSE=1 TO 1000:NEXT
350 BEEP:DX=0:DY=0
360 :
400 TASTE$=INKEY$:IF TASTE$="" THEN 500
410   IF TASTE$="1" THEN DX=-1:DY= 1:GOTO 500
420   IF TASTE$="2" THEN DX= 0:DY= 1:GOTO 500
430   IF TASTE$="3" THEN DX= 1:DY= 1:GOTO 500
```

```
440    IF TASTE$="4" THEN DX=-1:DY= 0:GOTO 500
450    IF TASTE$="5" THEN DX= 0:DY= 0:GOTO 500
460    IF TASTE$="6" THEN DX= 1:DY= 0:GOTO 500
470    IF TASTE$="7" THEN DX=-1:DY=-1:GOTO 500
480    IF TASTE$="8" THEN DX=-0:DY=-1:GOTO 500
490    IF TASTE$="9" THEN DX= 1:DY=-1:GOTO 500
495 GOTO 400
500 LOCATE Y,X:COLOR 1:PRINT CHR$(79);
505 GOSUB 1000
510 X=X+DX:Y=Y+DY
520    IF X<1  THEN X=1
530    IF X>80 THEN X=80
540    IF Y<1  THEN Y=1
550    IF Y>24 THEN Y=24
560 IF SCREEN(Y,X)=42 THEN 700
570 LOCATE Y,X:COLOR 15:PRINT CHR$(79);
580 IF NOT(DX=0 AND DY=0) THEN PUNKTE=PUNKTE+1
600 ZEILE=INT(24*RND)+1:SPALTE=INT(80*RND)+1
610 IF SCREEN(ZEILE,SPALTE) <> 32 THEN 600
620 LOCATE ZEILE,SPALTE:COLOR 12:PRINT CHR$(42);
625 GOSUB 1000
630 GOTO 400
640 :
700 BEEP
710 COLOR 15,1,1
720 CLS
730 LOCATE 12,25:PRINT"Sie haben";PUNKTE;"Punkte erreicht"
740 LOCATE 20,1:PRINT"Drücken Sie jetzt wieder die Num-Lock-Taste !"
750 INPUT"Haben Sie die Num-Lock-Taste grdrückt (j/n).....";A$
760 IF A$<>"j" THEN 750 ELSE END
770 :
780 :
1000 FOR PAUSE=1 TO 30 :NEXT
1010 RETURN
```

Erläuterungen zum Programm (Listing 2):

Die Zeilen 130-195 dienen der Spielbeschreibung. Mit der Anweisung COLOR 15,1,6 in Zeile 110 wird der braune Bassinrand und das blaue Wasser initialisiert und durch CLS in Zeile 196 ausgelöst.

Die Schleife in den Zeilen 200-240 erzeugt 40 zufällige rosarote Wasserminen. Zeile 300 berechnet die willkürliche Startposition des Bootes, Zeile 310 kontrolliert, ob das berechnete Feld unbesetzt ist und Zeile 320 zeichnet das Boot hin.

Interessant ist jetzt die Steuerung des Bootes. Studieren wir den folgenden Programmausschnitt:

```
350 BEEP:DX=0:DY=0   'Verrückungen in x- und y-Richtung'
400 TASTE$=INKEY$:IF TASTE$="" THEN 500
410 IF TASTE$="1" THEN DX=-1:DY=+1:GOTO 500
... usw.
490 IF TASTE$="9" THEN DX=+1:DY=-1:GOTO 500
495 GOTO 400
500 LOCATE Y,X:COLOR 1:PRINT CHR$(79)   'Boot löschen'
505 GOSUB 1000   'Zeitverzögerung'
510 X=X+DX : Y=Y+DY 'neue Positon'
    ...
560 IF SCREEN(Y,X)=42 THEN 700 'Spiel zu Ende'
570 LOCATE Y,X:COLOR 15:PRINT CHR$(79)   'Boot zeichnen'
    ...
630 GOTO 400
```

Bei Beginn des Spieles ist DX=0 und DY=0. So lange Sie keine Taste drücken, bleibt das Boot am Ort stehen und Sie erhalten keine Punkte gutgeschrieben. Wenn Sie aber z.B. die Taste 1 antippen, wird DX=-1 und DY=+1 gesetzt. Das Boot bewegt sich darauf diagonal so lange nach links unten, bis Sie eine neue Taste drücken. Es gilt X=X+DX=X-1 und Y=Y+DY=Y+1.

Weil die IF-Abfragen in den Zeilen 410-490 nur in Aktion treten können, wenn in Zeile 400 die Variable TASTE$ vom Tastaturpuffer ein Zeichen erhalten hat, bleibt die einmal eingeschlagene Richtung erhalten.

Die Zeile 580 IF NOT(DX=0 AND DY=0) THEN PUNKTE =PUNKTE+1 sorgt dafür, dass nur dann ein Punkt erzielt wird, wenn das Boot seine Stelle verändert hat und deshalb DX und DY nicht gleichzeitig Null sein können.

Mit diesen Erläuterungen sollte das Programm in Listing 2 verständlich sein.

2. Bewegung eines Pixels in mittlerer (SCREEN 1) und in hoher Auflösung (SCREEN 2)

Wenn wir die mittlere oder hohe Auflösung einschalten, reduziert sich ein Pixel auf einen einzelnen Punkt. Mit der Anweisung PSET(X,Y),FARBE setzen wir einen Punkt und mit der Anweisung PSET(X,Y),0 löschen wir wieder diesen Punkt. Das Programm in Listing 3 bewegt einen Punkt innerhalb eines vorgegebenen Rechtecks. Der Punkt wird dabei an den Wänden genau so reflektiert wie eine Billardkugel an den Banden. Sie können die Lage der vier Wände und den Startort des Punktes selber bestimmen.

Richtung und Geschwindigkeit des Punktes sind durch die Zahlenwerte für DX, DY und DAUER in Zeile 190 festgelegt. Wählen Sie ganzzahlige Werte aus dem Intervall von -4 bis +4. Bestimmen Sie den Wert für die Variable DAUER so, dass sich der Punkt flimmerfrei über den Schirm bewegt.

Zeile 210 zeichnet das von Ihnen gewünschte Rechteck und Zeile 300 plaziert den Punkt an die verlangte Stelle. Mit

Listing 3

```
100 CLS:KEY OFF:WIDTH 80
110 LOCATE 8,1
120 INPUT"x-Koordinate der linken  Wand ..... ";XLINKS
130 INPUT"x-Koordinate der rechten Wand ..... ";XRECHTS
140 INPUT"y-Koordinate der oberen  Wand ..... ";YOBEN
150 INPUT"y-Koordinate der unteren Wand ..... ";YUNTEN
160 PRINT:PRINT
170 INPUT"x-Koordinate des Startpunktes ..... ";X
180 INPUT"y-Koordinate des Startpunktes ..... ";Y
190 DX=2:DY=2:DAUER=10
195 :
200 SCREEN 1:COLOR 0,0
210 LINE(XLINKS,YOBEN)-(XRECHTS,YUNTEN),2,B
220 :
300 PSET(X,Y),3
310 XALT=X:YALT=Y
320 X=X+DX:Y=Y+DY
330 IF X<=XLINKS OR X>=XRECHTS THEN X=X-DX:DX=-DX
340 IF Y<=YOBEN  OR Y>=YUNTEN  THEN Y=Y-DY:DY=-DY
350 PSET(XALT,YALT),0
360 FOR PAUSE=1 TO DAUER:NEXT PAUSE
370 IF INKEY$="" THEN 300
380 :
400 SCREEN 0:WIDTH 80:END
```

X=XALT und Y=YALT werden die Koordinaten des Punktes für später gespeichert. Mit X=X+DX und Y=Y+DY wird die neue Stelle berechnet. In Zeile 350 wird der Punkt mit der Anweisung PSET(XALT,YALT),0 gelöscht und nach einer gewissen Zeitverzögerung in Zeile 360 am neuen Ort mit PSET(X,Y),3 in Zeile 300 wieder hingezeichnet.

Es gibt noch andere Möglichkeiten, den gleichen Bewegungsablauf zu programmieren, z.B. Verwendung einer WHILE-Schleife, Verwendung von PRESET(X,Y) usw.

Wie Sie gesehen haben, ist die Bewegung eines einzelnen Pixels in niederer, mittlerer oder hoher Auflösung leicht zu programmieren.

Wie bewegt man aber eine ausgedehnte Figur, die aus vielen einzelnen Punkten aufgebaut ist?

Wie schon früher erwähnt, dienen dazu die Anweisungen GET und PUT im Grafikmodus SCREEN 1 und SCREEN 2.

3. Die Anweisung GET(X1,Y1)-(X2,Y2),A

Die Anweisung GET(X1,Y1)-(X2,Y2),A kopiert den rechtekkigen Bildschirmausschnitt mit der linken oberen Ecke P(X1,Y1) und der rechten unteren Ecke P(X2,Y2) in das einfach indizierte numerische Feld mit dem frei wählbaren Variablennamen A. Dieses Feld muss mit DIM A(N) vorgängig dimensioniert werden. Wir werden sofort zeigen, wie man aus der Länge und Breite des Bildschirmausschnitts die Feldgrösse N berechnen kann.

Durch die GET-Anweisung wird die Farbnummer (0,1,2,3) jedes Punktes des Rechtecks bestimmt und byteweise in das lineare Feld A(N) übertragen. Der Kopiervorgang dauert nur den Bruchteil einer Sekunde. Wichtig ist, dass das von Ihnen gewählte Rechteck die ganze Figur, die bewegt werden soll, umschliesst.

Berechnung der Grösse N des Feldes A(N)

Das Handbuch liefert folgende Berechnungsformel:

Anzahl Bytes = 4 + Y*INT((X*Pixelbits+7)/8)

Dabei bedeuten

X = (X2-X1+1) die Rechtecklänge,
Y = (Y2-Y1+1) die Rechteckbreite.

In SCREEN 1 ist Pixelbits = 2 und
in SCREEN 2 ist Pixelbits = 1

Zahlen vom Typ INTEGER belegen 2 Bytes,
Zahlen vom Typ SINGLE PRECISION 4 Bytes,
Zahlen vom Typ DOUBLE PRECISION 8 Bytes,
ein Einzelzeichen eines Strings 3 Bytes.

Wir werden in allen folgenden Programmen nur in SCREEN 1 arbeiten und das Feld A für Zahlen mit einfacher Genauigkeit dimensionieren.

Somit ergibt sich für die gesuchte Feldgrösse

N = (Anzahl Bytes nach Formel : 4) - 1

Wir dürfen deshalb eine 1 abziehen, weil die Indizierung von Feldern in BASIC bei 0 statt bei 1 beginnt.

1. Beispiel:

Eine Figur sei durch das Rechteck mit den diagonal liegenden Punkten P(135,84) und P(172,115) umschlossen. Wie gross muss N mindestens gewählt werden?

X = 172-135+1 = 38
Y = 115-84+1 = 32
Anzahl Bytes = 4 + 32*INT((38*2+7)/8) = 324
Somit ist N = (324:4) - 1 = 80

Die Anweisungen lauten

```
DIM A(80)
.....
GET(135,84)-(172,115),A
```

2. Beispiel:

Wie gross ist der Speicherbedarf, um den ganzen Bildschirm in mittlerer Auflösung in das Feld A(N) zu kopieren?

Anzahl Bytes = 4 + 200*INT((320*2+7)/8) = 16004
Also wird N = (16004:4) -1 = 4000

Wir benötigen somit 4 KBytes, um den Inhalt des ganzen Bildschirms abspeichern zu können.

4. Die Anweisung PUT(X1,Y1),A,Modus

Die Anweisung PUT(X1,Y1),A,Modus bringt das im Feld A gespeicherte Bild blitzschnell wieder auf den Bildschirm, wobei der Punkt P(X1,Y1) die linke obere Ecke des rechteckförmigen Bildausschnittes ist.

Mit dem Modus geben Sie dem Computer bekannt, wie er das zu übertragende Bild mit dem schon auf dem Schirm vorhandenen Bild verknüpfen soll.

Folgende Verbindungsarten sind möglich:

PUT(X1,Y1),A,XOR
PUT(X1,Y1),A,PSET
PUT(X1,Y1),A,PRESET
PUT(X1,Y1),A,AND
PUT(X1,Y1),A,OR

Wenn Sie den Modus bei der PUT-Anweisung weglassen, setzt IBM-BASIC die Standardvoreinstellung XOR.

Die nachfolgenden drei Regeln bewähren sich im Umgang mit der PUT-Anweisung:

1. Regel:

Mit PUT(X1,Y1),A,PSET wird der Bildausschnitt (X1,Y1)-(X2,Y2) auf dem Bildschirm gelöscht und durch das in A gespeicherte Bild ersetzt.

2. Regel:

Mit PUT(X1,Y1),A,OR wird dem Bildausschnitt (X1,Y1)-(X2,Y2) auf dem Bildschirm das in A gespeicherte Bild überlagert. Man sieht also gleichzeitig zwei Bilder!

3. Regel:

Mit PUT(X1,Y1),A,XOR wird der Bildausschnitt (X1,Y1)-(X2,Y2) auf dem Bildschirm ausgelöscht, falls das in A gespeicherte Bild dazu pixelweise kongruent ist!

Hier die genaue Wirkungsweise der Modi:

Die Farbnummern der Pixel des rechteckförmigen Bildausschnittes werden nicht als Dezimalzahlen 0,1,2,3 sondern als zweistellige Dualzahlen 00,01,10,11 im Feld A(N) gespeichert. Wird eine PUT-Anweisung durchgeführt, dann werden die Dualziffern des Pixels auf dem Schirm und des Pixels des zu übertragenden Bildpunktes einzeln gemäss den folgenden Wahrheitstabellen verrechnet:

Sei a eine Dualziffer der im Feld A gespeicherten Farbnummer des Bildpunktes und
sei b die entsprechende Dualziffer der Farbnummer des Punktes auf dem Schirm.

PUT(X1,Y1),A,XOR

a	b	a XOR b
0	0	0
0	1	1
1	0	1
1	1	0

PUT(X1,Y1),A,PSET

a	b	a PSET b
0	0	0
0	1	0
1	0	1
1	1	1

PUT(X1,Y1),A,PRESET

a	b	a PRESET b
0	0	1
0	1	1
1	0	0
1	1	1

PUT(X1,Y1),A,AND

a	b	a AND b
0	0	0
0	1	0
1	0	0
1	1	1

PUT(X1,Y1),A,OR

a	b	a OR b
0	0	0
0	1	1
1	0	1
1	1	1

Wir weisen jetzt rechnerisch unsere drei Regeln nach:

1. Regel

aa PSET bb

aa \ bb	00	01	10	11
00	00	00	00	00
01	01	01	01	01
10	10	10	10	10
11	11	11	11	11

Offensichtlich erscheint nur das in A gespeicherte Bild aus den Farbnummern aa auf dem Bildschirm.

2. Regel

aa OR bb

aa \ bb	00	01	10	11
00	00	01	10	11
01	01	01	11	11
10	10	11	10	11
11	11	11	11	11

Wir erkennen, dass zwei Bilder einander überlagert werden.

3. Regel

aa XOR bb

aa \ bb	00	01	10	11
00	00	01	10	11
01	01	00	11	10
10	10	11	00	01
11	11	10	01	00

Haben die Pixels die gleichen Farbnummern, dann kommt es zur Auslöschung. Beachten Sie die Diagonale von links oben nach rechts unten in der Tabelle.

5. Bewegung einer Figur mit Hilfe von GET und PUT

Zwei verschiedene Algorithmen sind möglich.

1. Algorithmus

1. Erstelle irgendwo auf dem Schirm die Figur, welche bewegt werden soll.
2. Speichere mit GET(X1,Y1)-(X2,Y2),A das Bild in das Feld A(N).
3. Lösche den Bildschirm.
4. Wiederhole die Anweisungen 4.1. bis 4.4.
   - 4.1. Zeichne die Figur mit PUT(X1,Y1),A,XOR an der Stelle P(X1,Y1).
   - 4.2. Berechne die neue Stelle P(X2,Y2).
   - 4.3. Lösche die Figur mit PUT(X1,Y1),A,XOR an der alten Stelle P(X1,Y1).
   - 4.4. Ersetze X1 durch X2 und Y1 durch Y2.

Dieser Algorithmus hat einen gravierenden Nachteil. Durch die rasche Aufeinanderfolge des Zeichnens und Löschens der Figur entsteht eine ruckartige Bewegung, die ausserdem unangenehm flimmert. Selbst der Einbau einer Zeitverzögerungsschleife zwischen Zeichnen und Löschen bringt keine Abhilfe.

2. Algorithmus

Wenn Sie die Figur an der Stelle P(X1,Y1) durch ein Rechteck so umschliessen, dass dieses nach der Verschiebung an die neue Stelle P(X2,Y2) die Figur an der alten Stelle P(X1,Y1) immer noch überdeckt, dann können Sie die Figur mit PUT(X1,Y1),A,PSET wesentlich schneller und optisch befriedigender bewegen.

1. Erstelle die Figur auf dem Bildschirm.
2. Speichere mit GET(X1,Y1)-(X2,Y2),A das Bild der Figur in das Feld A. Das Rechteck (X1,Y1)-(X2,Y2) muss grösser als die Figur sein.
3. Lösche den Bildschirm.
4. Wiederhole die Anweisungen 4.1. bis 4.3.
   - 4.1. Zeichne die Figur an der Stelle P(X1,Y1) mit PUT(X1,Y1),A,PSET.
   - 4.2. Berechne die neue Stelle P(X2,Y2).
   - 4.3. Ersetze X1 durch X2 und Y1 durch Y2.

Wir geben ein Beispiel:
Auf blauem Hintergrund wird ein farbiges Schweizerkreuz gezeichnet und im Feld KREUZ gespeichert. Sobald Sie eine Taste antippen, wird der Bildschirm gelöscht. Danach erscheint das Kreuz auf der rechten Seite des Schirms und beginnt nach links herum zu kreisen. Die Bewegung ist relativ langsam aber flimmerfrei. Wenn Sie irgendeine Taste niederdrücken, hört die Bewegung auf und ein weiterer Tastendruck löscht den Bildschirm. Sie finden das Programm in Listing 4.

Sie wissen jetzt, wie man eine ausgedehnte Figur mit den Anweisungen GET und PUT bewegen kann. Mit diesen Anweisungen lassen sich aber noch andere raffinierte grafische Effekte erzielen.

6. Aufbau von grafischen Darstellungen durch Ueberlagern zweier verschiedener Bilder

Erneut verwenden wir als Zeichenfigur ein Schweizerkreuz. Mit ihm wollen wir den Bildschirm überdecken.

Das Programm in Listing 5 tut dies und Abbildung 1 ist die Hardcopy vom Bildschirm. Aus drucktechnischen Gründen haben wir für die Hardcopy andere Farben als

Listing 4

```
100 SCREEN 1:COLOR 0,0:CLS
110 DIM KREUZ(166)
120 :
200 LINE (5,5)-(45,45),3,B
210 DRAW"bm20,40;c3;r10;u10;r10;u10;l10;u10;l10;d10;l10;d10;r10;d10"
220 PAINT(25,25),3,3:PAINT(8,8),2,3
230 GET(0,0)-(50,50),KREUZ
240 :
250 LOCATE 22,1:PRINT"Taste druecken";
260 IF INKEY$="" THEN 260
265 :
270 CLS:BM=3.14159/180
275 LOCATE 1,1:PRINT"Rotierendes"
276 LOCATE 2,1:PRINT"Schweizerkreuz"
280 :
300 FOR W=0 TO 357 STEP 3
310   X=180+70*COS(W*BM):Y=75-70*SIN(W*BM)
320   PUT(X,Y),KREUZ,PSET
340 NEXT W
360 IF INKEY$="" THEN 300
370 :
400 BEEP
410 LOCATE 23,1:PRINT"Taste druecken";
420 IF INKEY$="" THEN 420
430 SCREEN 0:WIDTH 80:END
```

die aus dem Programm verwendet, sonst wäre der Ausdruck sehr dunkel geworden.

Wenn Sie das Programm fahren, werden Sie überrascht sein, wie schnell der Computer den Bildschirm mit Kreuzen zudeckt. Niemals könnten Sie dieses Tempo mit LINE-, DRAW- und PAINT-Anweisungen innerhalb der beiden geschachtelten Schleifen erreichen.

Im nächsten Programm wollen wir zwei verschiedene aber gleichgrosse Figuren übereinanderlegen, um auch eine Anwendung des OR-Modus kennenzulernen.

Im Programm in Listing 6 wird zunächst ein gelbes Kreuz gezeichnet mit GET(0,0)-(39,39),KREUZ abgespeichert. Darauf wird der Schirm gelöscht. Mit der Schleife von Zeile 300-330 werden fünf verschieden farbige konzentrische Kreise gezeichnet und mit GET(0,0)-(39,39),KREIS für nachher im Feld KREIS festgehalten.

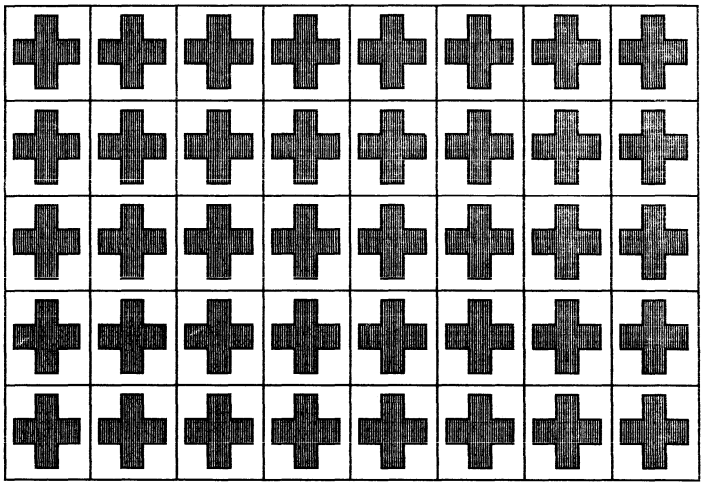


Abbildung 1

Listing 5

```
100 SCREEN 1:COLOR 1,1:CLS
110 DIM KREUZ(100)
120 :
200 LINE(0,0)-(39,39),3,B
210 DRAW"bm15,35;c3;r10;u10;r10;u10;l10;u10;l10;d10;l10;d10;r10;d10"
220 PAINT(20,20),3,3:PAINT(3,3),2,3
230 GET(0,0)-(39,39),KREUZ
240 :
250 LOCATE 22,1:PRINT"Taste druecken";
260 IF INKEY$="" THEN 260
270 CLS
280 :
300 FOR X=0 TO 273 STEP 39
310   FOR Y=0 TO 156 STEP 39
320     PUT(X,Y),KREUZ,PSET
330   NEXT
340 NEXT
360 :
400 BEEP
410 IF INKEY$="" THEN 410
420 SCREEN 0:WIDTH 80:END
```

Listing 6

```
100 SCREEN 1:COLOR 1,0:CLS
110 DIM KREUZ(100),KREIS(100)
120 :
200 LINE(0,0)-(39,39),3,B
210 DRAW"bm15,35;c3;r10;u10;r10;u10;l10;u10;l10;d10;l10;d10;r10;d10"
220 GET(0,0)-(39,39),KREUZ
230 CLS
240 :
250 FARBE=3
300 FOR R=6 TO 18 STEP 3
310   CIRCLE(20,20),R,FARBE
320   FARBE=FARBE-1:IF FARBE=0 THEN FARBE=3
330 NEXT R
340 GET(0,0)-(39,39),KREIS
350 CLS
360 :
400 FOR X=0 TO 273 STEP 39
410   FOR Y=0 TO 156 STEP 39
420     PUT(X,Y),KREUZ,PSET:PUT(X,Y),KREIS,OR
430   NEXT Y
440 NEXT X
450 :
500 BEEP
510 IF INKEY$="" THEN 510
520 SCREEN 0:WIDTH 80:END
```

Die geschachtelten Schleifen von Zeile 400-440 decken den Schirm mit Kreuzen und Kreisscharen zu, wobei die Bilder durch PUT(X,Y),KREUZ,PSET : PUT(X,Y),KREIS,OR superponiert werden. Abbildung 2 zeigt das entstehende Figurenmuster.

Im 4. Teil unserer Serie (M+K 86-5) haben wir Ihnen einen neuen Algorithmus vorgestellt, mit dem Sie beim Zeichnen von dreidimensionalen Funktionen die nicht sichtbaren Linien auslöschen können. Die entstehenden Bilder der Funktion werden als Liniengrafik einer 3D-Funktion bezeichnet.

Noch viel realistischer sehen dreidimensionale Funktionen in der sogenannten Netzgrafik aus.

Wir erweitern daher unsere Hidden-Line-Routine so, dass wir Netzbilder solcher Funktionen erstellen können.

Algorithmus

1. Feld A(4000) dimensionieren
2. Zahlenwerte für X1,X2,Y1,Y2,DX,DY,K,KX,KY festlegen
3. FOR Y=Y1 TO Y2 STEP DY
   4. FOR X=X1 TO X2 STEP 1
      4.1. XX=KX*X:YY=KY*Y:Z=K*F(XX,YY)
      4.2. XNEU und YNEU berechnen
      4.3. Lösche die Linie von P(XNEU,YNEU) bis P(XNEU,199) aus
      4.4. Wenn X>X1 dann verbinde P(XALT, YALT) mit P(XNEU,YNEU)
      4.5. Ersetze XALT durch XNEU und YALT durch YNEU
5. Speichere den Bildschirm mit GET(0,0)-(319,199),A
6. Lösche den Bildschirm
7. FOR X=X2 TO X1 STEP -DX
   8. FOR Y=Y1 TO Y2 STEP 1
      8.1. XX=KK*X:YY=KY*Y:Z=K*F(XX,YY)
      8.2. XNEU und YNEU berechnen
      8.3. Lösche die Linie von P(XNEU,YNEU) bis P(XNEU,199) aus
      8.4. Wenn Y>Y1 dann verbinde P(XALT, YALT) mit P(XNEU,YNEU)
      8.5. Ersetze XALT durch XNEU und YALT durch YNEU
9. Ueberlagere das gespeicherte Bild mit PUT(0,0),A,OR

Wollen Sie inserieren?

Media-Unterlagen

☎ 041-31 18 46

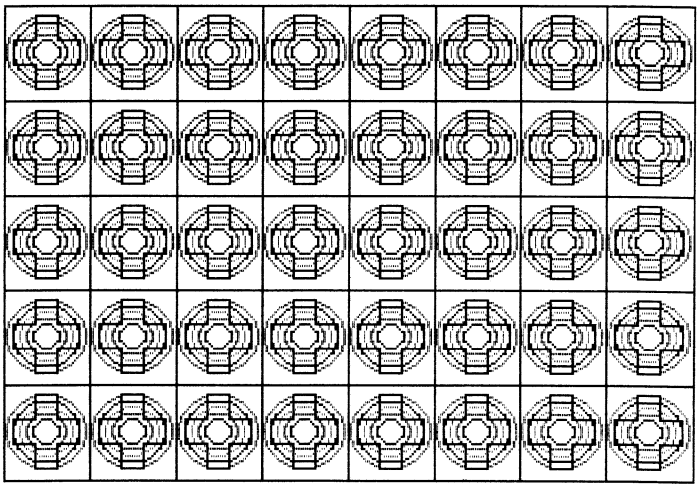


Abbildung 2

Die ersten zwei geschachtelten Schleifen (Anweisungen 3. bis 4.5.) zeichnen die Funktion von hinten links oben nach vorne rechts unten. Es entstehen die «waagrechten Linien». Wenn die Funktion fertig erstellt ist, muss der ganze Bildschirm in das Feld A(4000) kopiert werden. Darauf wird der Bildschirm gelöscht.

Die zweiten zwei geschachtelten Schleifen (Anweisungen 7. bis 8.5.) zeichnen die gleiche Funktion von hinten rechts oben nach vorne links unten. Es entstehen die «senkrechten Linien». Aus optischen Gründen haben wir für das zweite Bild eine andere Zeichenfarbe gewählt. Zum Schluss überlagern wir das in A gespeicherte Bild der zuletzt gezeichneten Figur (Anweisung 9.).

Da vermutlich viele Leser an Netzgrafiken von 3D-Funktionen interessiert sind, drucken wir vier komplette Listings samt Hardcopies ab.

1. Programm

Das Programm in Listing 7 zeichnet den Graph der Funktion z=75*exp(-x*x-y*y), siehe Abbildung 3.

2. Programm

Das Programm in Listing 8 zeichnet den Graph der Funktion z=30*cos(x)*cos(y), siehe Abbildung 4.

3. Programm

Das Programm in Listing 9 zeichnet die Hälfte eines Torus mit eingeschlossener Kugel, siehe Abbildung 5.

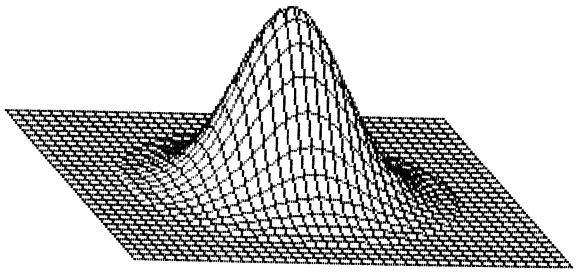
4. Programm

Das Programm in Listing 10 zeichnet den Graph der Funktion z=35* cos(r)-cos(3)/3+cos(5r)/5-cos(7r)/7), wobei r=sqr(x*x+y*y) ist. Abbildung 6 zeigt einen Programmlauf. Es entsteht der in Informatikerkreisen bekannte Hut.

Alle Programme sind nach dem gleichen Muster «gestrickt». Es müsste Ihnen daher nicht schwerfallen, selber ähnliche Programme für Ihre Lieblingsfunktion zu schreiben.

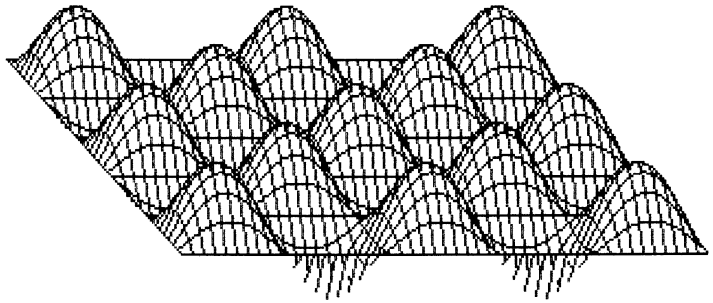
Wir sind jetzt dank den Anweisungen GET und PUT in der Lage, das Traumprogramm «Drehung eines Körpers um eine Achse in real time» zu entwickeln.

Fortsetzung in M+K 87-1



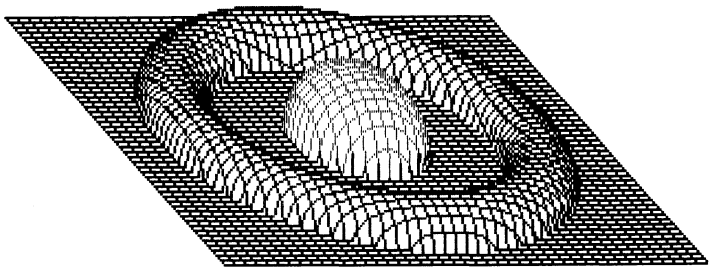
Listing 7 / Abbildung 3

```
100 SCREEN 1:CLS:KEY OFF
110 DIM A(4001)
120 COLOR 0,0
130 X1=-100:X2=100:Y1=-30:Y2=30
140 U=160:V=120:DY=2
150 KX=1/33:KY=1/10:K=75
160 :
200 FOR Y=Y1 TO Y2 STEP DY:YY=Y*KY
210   FOR X=X1 TO X2
220    XX=X*KX:Z=K*EXP(-XX*XX-YY*YY)
240    XNEU=U+Y+X:YNEU=V+Y-Z
250    LINE(XNEU,YNEU)-(XNEU,199),0
260    IF X>X1 THEN LINE(XALT,YALT)-(XNEU,YNEU),2
270    XALT=XNEU:YALT=YNEU
280   NEXT X
290 NEXT Y
295 :
300 GET(0,0)-(319,199),A
310 CLS
320 :
400 FOR X=X2 TO X1 STEP -5:XX=X*KX
410   FOR Y=Y1 TO Y2
420    YY=Y*KY:Z=K*EXP(-XX*XX-YY*YY)
430    XNEU=U+Y+X:YNEU=V+Y-Z
440    LINE(XNEU,YNEU)-(XNEU,199),0
450    IF Y>Y1 THEN LINE(XALT,YALT)-(XNEU,YNEU),3
460    XALT=XNEU:YALT=YNEU
470   NEXT Y
480 NEXT X
490 :
500 PUT(0,0),A,OR
510 LOCATE 1,1:PRINT"z = k*exp(-x*x-y*y)"
520 LOCATE 2,1:PRINT"mit Netzlinien"
530 BEEP
540 :
600 IF INKEY$="" THEN 600
610 SCREEN 0:WIDTH 80:END
```



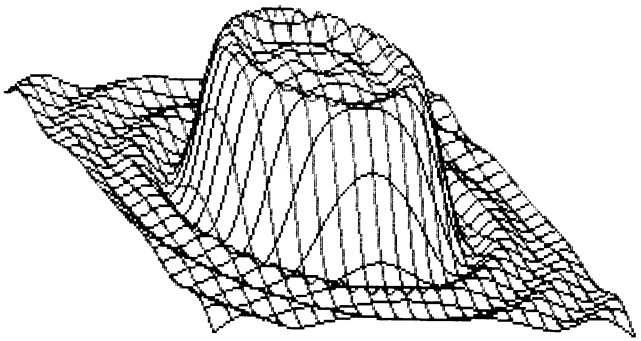
Listing 8 / Abbildung 4

```
100 SCREEN 1:CLS:KEY OFF:COLOR 0,0
110 DIM A(4000)
120 X1=-120:X2=120:Y1=-40:Y2=40
130 U=160:V=100:DY=2:DX=5
140 K=30:KX=2.5*3.14159/120:KY=2.5*3.14159/40
150 :
200 FOR Y=Y1 TO Y2 STEP DY
210  Z1=K*COS(Y*KY)
220     FOR X=X1 TO X2
230       Z=Z1*COS(X*KX)
240       XNEU=U+Y+X:YNEU=V+Y-Z
250       LINE(XNEU,YNEU)-(XNEU,199),0
260       IF X>X1 THEN LINE(XALT,YALT)-(XNEU,YNEU),3
270       XALT=XNEU:YALT=YNEU
280     NEXT X
290 NEXT Y
300 :
310 GET(0,0)-(319,199),A
320 CLS
330 :
400 FOR X=X2 TO X1 STEP -DX
410  Z1=K*COS(X*KX)
420     FOR Y=Y1 TO Y2
430       Z=Z1*COS(Y*KY)
440       XNEU=U+Y+X:YNEU=V+Y-Z
450       LINE(XNEU,YNEU)-(XNEU,199),0
460       IF Y>Y1 THEN LINE(XALT,YALT)-(XNEU,YNEU),2
470       XALT=XNEU:YALT=YNEU
480     NEXT Y
490 NEXT X
500 :
510 PUT(0,0),A,OR
520 BEEP
530 LOCATE 1,1:PRINT"z=cos(x)*cos(y)"
540 :
600 IF INKEY$="" THEN 600
610 SCREEN 0:WIDTH 80:END
```



Listing 9 / Abbildung 5

```
100 CLS:KEY OFF:SCREEN 1:COLOR 0,0
110 X1=-110:X2=110:Y1=-100:Y2=100
120 U=160:V=100:DY=4:DX=5
130 DIM A(4001)
140 :
200 FOR Y=Y1 TO Y2 STEP DY
210   FOR X=X1 TO X2
220    GOSUB 1000
230    XNEU=U+Y/2+X : YNEU=V+Y/2-Z
240    LINE(XNEU,YNEU)-(XNEU,199),0
250    IF X>X1 THEN LINE(XALT,YALT)-(XNEU,YNEU),F
260    XALT=XNEU : YALT=YNEU
270   NEXT X
280 NEXT Y
290 :
300 GET(0,0)-(319,199),A
310 CLS
320 :
400 FOR X=X2 TO X1 STEP -DX
410   FOR Y=Y1 TO Y2
420    GOSUB 1000
430    XNEU=U+Y/2+X:YNEU=V+Y/2-Z
440    LINE(XNEU,YNEU)-(XNEU,199),0
450    IF Y>Y1 THEN LINE(XALT,YALT)-(XNEU,YNEU),F
460    XALT=XNEU:YALT=YNEU
470   NEXT Y
480 NEXT X
490 :
500 PUT(0,0),A,OR
510 LOCATE 1,1:PRINT"Torus mit Halbkugel"
520 :
600 BEEP
610 IF INKEY$="" THEN 610
620 SCREEN 0:WIDTH 80:END
630 :
1000 R=SQR(X*X+Y*Y)
1010 IF R>90 THEN Z=0:F=3:RETURN
1020 IF R<60 THEN 1040
1030 R1=ABS(75-R):Z=SQR(225-R1*R1):F=2:RETURN
1040 IF R>30 THEN Z=0:F=3:RETURN
1050 Z=SQR(900-R*R):F=1:RETURN
```



Listing 10 / Abbildung 6

```
100 SCREEN 1:KEY OFF:CLS
110 COLOR 0,0
120 U=160:V=100:DX=8:DY=8
130 X1=-96 :X2=96 :Y1=-96 :Y2=96
140 KX=3.14159/100:KY=KX:K=35
150 DIM A(4000)
160 :
200 FOR Y=Y1 TO Y2 STEP DY
210  YY=KY*Y
220   FOR X=X1 TO X2
230    XX=KX*X:R=SQR(XX*XX+YY*YY)
240    Z=K*(COS(R)-COS(3*R)/3+COS(5*R)/5-COS(7*R)/7)
250    XNEU=U+Y/2+X:YNEU=V+Y/2-Z
260    LINE(XNEU,YNEU)-(XNEU,199),0
270    IF X>X1 THEN LINE(XALT,YALT)-(XNEU,YNEU),3
280    XALT=XNEU:YALT=YNEU
290   NEXT
300 NEXT
310 :
320 GET(0,0)-(319,199),A
340 CLS
350 FOR X=X2 TO X1 STEP -DX
360  XX=KX*X
370   FOR Y=Y1 TO Y2
380    YY=KY*Y:R=SQR(XX*XX+YY*YY)
390    Z=K*(COS(R)-COS(3*R)/3+COS(5*R)/5-COS(7*R)/7)
400    XNEU=U+Y/2+X:YNEU=V+Y/2-Z
410    LINE(XNEU,YNEU)-(XNEU,199),0
420    IF Y>Y1 THEN LINE(XALT,YALT)-(XNEU,YNEU),2
430    XALT=XNEU:YALT=YNEU
440   NEXT
450 NEXT
460 :
470 PUT(0,0),A,OR
500 LOCATE 1,1:PRINT"z=cos(r)-cos(3r)/3+cos(5r)/5-cos(7r)/7";
510 BEEP
520 :
600 IF INKEY$<>"e" THEN 600
610 SCREEN 0:WIDTH 80:END
```

Künstliche Intelligenz (3. Teil)

Nachdem in M+K 86-5 die Grundlagen der künstlichen Intelligenz und Expertensysteme erklärt wurden, befassen wir uns in dieser Ausgabe mit der Spracherkennung. Dadurch ist es möglich, sich in natürlicher Sprache mit dem Rechner zu unterhalten. Unter «natürlicher Sprache» wird hierbei sowohl die geschriebene als auch die gesprochene Sprache verstanden und beide Arten werden erläutert.

Man stelle sich einen Computer vor, der die Ausdrucksweise des Menschen versteht und darauf reagieren kann. Diese (zumindest bis jetzt noch) Zukunftsvision lässt weite Anwendungsbereiche erkennen, beispielsweise:

- Uebersetzung fremder Sprache
- Auskunftssysteme
- Steuerung von Maschinen über Sprache
- Einfachere Bedienung von Computern

Nachdem in der letzten Ausgabe von Expertensystemen die Rede war, die Bodenschätze aufspüren können, wird der Leser jetzt wohl erwarten,

Michael Schlingmann

eine Aufzählung sprachverstehender Software angeboten zu bekommen. Doch leider steckt die Entwicklung von Computern die «hören» können, im wahrsten Sinne des Wortes noch in den Kinderschuhen: der beste Rechner, der zur Zeit in den Laboratorien in aller Welt zu finden ist, hat in Bezug auf seine sprachlichen Fähigkeiten das Entwicklungsstadium eines dreijährigen Kindes gerade erreicht. Dem Programmierer müssen sich also ganz enorme Schwierigkeiten in den Weg stellen. Welcher Art diese Probleme sind, davon soll dieser Beitrag im Folgenden handeln.

Geschriebene Sprache

Da das Verstehen gesprochener Sprache das Begreifen des dabei verwendeten Satzgefüges voraussetzt, beginnen wir mit den Unannehmlichkeiten, die schon die geschriebene Sprache dem Rechner aufzwingt. Hierbei wollen wir bis auf weiteres unter dem Begriff «geschriebene Sprache» die Eingabe von Schrift über eine Tastatur verstehen.

Schon in den vierziger Jahren beschäftigten sich die Wissenschaftler im Rahmen militärischer Forschungen mit dem Problem der maschinellen Uebersetzung fremder Sprachen. Alle diese Versuche fielen mehr oder weniger kläglich ins Wasser. In den sechziger Jahren wurde auch in Universitäten versucht, dem Computer eine Art Intelligenz beizubringen, die sich darin äussert, dass er auf Fragen über ein bestimmtes Gebiet eine Antwort geben kann. Ein solches Programm ist z.B. ELIZA von Weizenbaum, das auch heute noch bekannteste KI-Programm. Wie schon an früherer Stelle berichtet, stellt ELIZA einen Psychiater dar, der sich mit seinen Patienten «unterhält». In einer Datei ist eine Liste von Schlüsselwörtern gespeichert, die erfahrungsgemäss häufig von den Patienten angewandt werden. Jedem Wort ist ein typisches Antwortmuster zugeordnet, das eigentlich nur darin besteht, den vom Benutzer eingegebenen Satz in geeigneter Weise umzuformen. Ist der Satz etwas länger, so werden Satzteile, in denen kein Schlüsselwort vorkommt, einfach gelöscht. Tauchen mehrere Schlüsselworte auf, kommen Prioritätsregeln zur Geltung. Tritt einmal gar kein Schlüsselwort auf, so weicht der Rechner aus, indem er z.B. sagt: «Erzählen Sie mir mehr von sich».

Mit relativ einfachen Regeln kann Verstehen also vorgetäuscht werden. Dass es sich hier aber wirklich nur um die Vorspiegelung falscher Tatsachen handelt, kann man daran erkennen, dass bei Eingabe unsinniger Sätze mit Schlüsselwort genau derselbe Unsinn wieder ausgegeben wird. Beispiel: Eingabe: «Ich kann mein urps nicht mehr leiden». Antwort: «Was stört Sie so an Ihrem urps?» Um den Rechner wirklich verstehen zu lassen, muss man also etwas mehr in die Tiefe gehen und sich in die Niederungen der Grammatik begeben. Dies haben die Forscher natürlich getan und es wurden wesentlich bessere Systeme entwickelt, die mit ELIZA ausser der Bezeichnung «Computerprogramm» nichts mehr gemein haben.

Das Grundprinzip dieser neuen Software ist in Bild 1 dargestellt. Hierbei durchläuft die Eingabe des Benutzers verschiedene Stadien, in denen die Satzstruktur und die Bedeutung der darin enthaltenen Worte über eine Reihe von Inferenzmechanismen untersucht werden. Unter anderem stehen die Ueberprüfung der grammatikalischen Richtigkeit und die Suche nach den vorkommenden Worten in einem Lexikon an. Ist diese Barriere überwunden, muss sich der Computer «Gedanken» machen über den Sinn des Satzes. Dazu ist unter anderem die Untersuchung der Vorgeschichte und, falls der Satz weitergeht, des Restinhalts des Satzes notwendig. Ist auch dies geschafft, wird über geeignete Regeln, die auch grammatikalische Gesetze beinhalten, eine Ausgabe veranlasst, die dem Benutzer (hoffentlich) seine Frage beantwortet.

Bei all dem erhebt sich natürlich die Frage, weshalb das alles so schwierig ist. Wenn Sie den Schalterbeamten am Bahnhof fragen, wann der nächste Zug nach Zürich geht, so wird er wohl kaum überprüfen, ob der von Ihnen gesprochene Satz grammatikalisch richtig ist und die einzelnen Worte überhaupt existieren.

Oder doch? Die Antwort ist ja. Wir (oder besser gesagt der Beamte) merken das nur nicht. In unserem Gehirn ist in der Regel die vollständige Grammatik unserer Muttersprache abgespeichert und wir verfügen im Erwachsenenalter leicht über einen Wortschatz von 100'000 Wörtern, obwohl wir nur etwa 4'000 davon mehr oder weniger ständig benützen. Nur geht die Ueberprüfung nach Richtigkeit des Satzes wesentlich schneller als beim Computer. Denn wir müssen uns nicht bei jedem Mal neu überlegen, was der Fragesteller denn jetzt genau meint. Dies ergibt sich in der Regel aus dem Umfeld oder aus früheren Begegnungen.

Natürlich kann man dem Rechner das alles in Form von Daten eingeben. Allerdings wird man dann sehr schnell Kapazitätsprobleme bekommen, auch wenn Festplatten mit Hunderten von Megabytes benutzt werden. Ein Beispiel gefällig? Stellen Sie sich vor, Sie wollen in einem Restaurant Fisch essen. Und jetzt probieren Sie bitte, sich an alle Tatsachen zu erinnern, die dabei beachtet werden müssen. Dazu gehört unter anderem die richtige Sitzhaltung, das Führen der Gabel, das Entgräten, ...

Schreiben Sie alles (alles!) auf, was Ihnen dazu einfällt. Mit Sicherheit werden Sie dafür mindestens einen Tag und mehrere Kugelschreiber verbrauchen. Und trotzdem wird der Computer, wenn wir ihm das alles eingegeben haben, noch keinen Fisch essen können. Denn in unserem Aufschrieb haben wir ja vergessen, dass wir den Mund aufmachen müs-

sen, um den Fisch zu verspeisen. Sie sehen an diesem kleinen Beispiel, dass es ungeheuer schwierig ist, dem Computer unser Wissen überhaupt bereitzustellen. Dazu noch ein Vergleich: das menschliche Gehirn hat eine Speicherkapazität von etwa 10 hoch 14 Bit pro Kubikzentimeter, eine Festplatte bringt es zur Zeit nur auf etwa 10 hoch 4, also zehn Zehnerpotenzen weniger.

Aus diesem Grund kann man sprachverstehende Computer in nächster Zukunft nur für ganz spezielle Aufgaben einsetzen, die kein allzu grosses Hintergrundwissen verlangen.

Gesetzt den Fall, dass diese Hürde überwunden ist, türmen sich auf dem Weg zum Verstehen noch weitere nicht minder komplizierte Probleme auf. Mit seinem Rechner tritt man normalerweise in Kontakt über eine Sprache die er versteht. Man nennt sie Programmiersprache. Deren Grammatik ist äusserst einfach, Syntaxänderungen kommen nicht in Frage. Falls doch, schickt der Computer eine Fehlermeldung. Anders als die formale Programmiersprache steht es mit unserer Umgangssprache, mittels der wir mit unseren Nachbarn kommunizieren. Diese Sprache lebt, täglich tauchen neue Worte auf, wird ein Ausdruck in einem anderen Sinn verwendet, wird die Grammatik wenigstens in kleinen Teilen verändert. Nehmen wir z.B. den Ausdruck «Null Bock», mit dem die Jugend klar machen will, dass sie zu bestimmten Tätigkeiten nicht unbedingt herangezogen werden will. Ein Computer würde dies natürlich so verstehen, dass keine männlichen Ziegen mehr da sind, und man kann ihm daraus nicht einmal einen Vorwurf machen.

Damit sind wir aber auch schon beim Kern des Problems, den sogenannten Ambiguitäten. Eine Ambiguität besteht dann, wenn ein Wort mehrfache Bedeutung hat.

Dazu ein klassisches Beispiel, gefunden von dem KI-Pionier David Waltz: Der Satz «I saw a man on the hill with a telescope» lässt, ins Deutsche übersetzt, mehrere Deutungen zu:

a) Ich sah einen Mann auf dem Hügel mit einem Fernglas in der Hand.
b) Ich sah einen Mann auf dem Hügel, auf dem sich ein Fernrohr befindet.
c) Ich sah durch ein Fernrohr und erblickte einen Mann auf dem Hügel.

Aehnlich ambiguente Sätze gibt es natürlich auch in deutsch. Die richtige

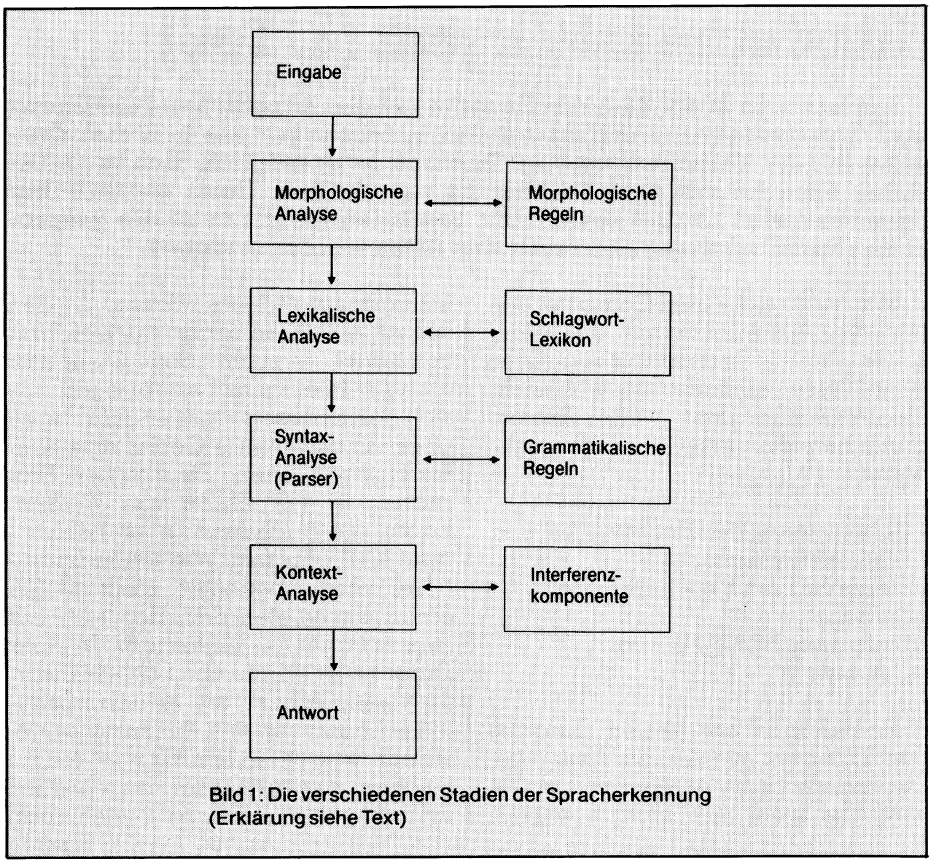


Bild 1: Die verschiedenen Stadien der Spracherkennung (Erklärung siehe Text)

Deutung lässt sich erst erahnen, wenn der Kontext des Satzes bekannt ist. Im obigen Beispiel wird allerdings auch diese Methode meist versagen.

Ein Programm behilft sich in diesem Fall damit, dass es Pronomina, Adverbe und dergleichen z.B. dem jeweils nachfolgenden Substantiv zuordnen. Eine ähnliche Vorgehensweise wird im obigen Beispiel wohl zur Interpretation a) führen.

Allerdings versagt das Verfahren bei Sätzen, die etwas verschlungener sind. Der Satz «Ich kaufte mir Apfelsinen, setzte mich zu meiner Freundin und ass sie» ist für den Menschen ohne weiteres verständlich. Der Computer hingegen wird unweigerlich zur Folgerung kommen, dass ein Menschenfresser unter uns ist, es sei denn man teilt ihm vorher mit, dass man Freundinnen im Gegensatz zu Apfelsinen nicht verzehren darf.

Auch indirekte Fragen führen zu Problemen. Wenn Sie einen Passanten fragen: «Können Sie mir die Uhrzeit sagen?», so bekommen Sie in der Regel die gewünschte Auskunft. Ein Computer würde im selben Fall nur mit einem treuherzigen «ja» antworten, was auf Dauer wohl kaum zufriedenstellen wird.

Weitere Diskrepanzen können auftreten bei Uebersetzungen. Der Satz «Ich habe ein Schloss gekauft» lässt mehrere Deutungen zu. Bei einer Uebersetzung ins Französische gibt es aber Probleme, denn die entstandene Ambiguität lässt sich nicht übersetzen. Man muss auswählen unter «J'ai acheté un château» oder «J'ai acheté une serrure». Die wahre Bedeutung ist nur aus dem Kontext ableitbar. Wenn dort Türen oder Aehnliches auftauchen, so ist die Wahrscheinlichkeit gross, dass es sich beim Schloss um ein Türschloss handelt.

Schliesslich gibt es noch eine Art von Ambiguitäten, die bisher noch nicht besprochen wurde: «Karl will einen taiwanesischen Computer kaufen» lässt wieder mehrere Möglichkeiten zu:

a) Karl will einen bestimmten Rechner kaufen, der aus taiwanesischer Produktion stammt.
b) Die einzige Forderung, die Karl an seinen neuen Computer stellt, ist die, dass er aus Taiwan kommen soll.

Es gibt noch weitere Sorten von Ambiguitäten, ich hoffe aber, dass die obigen Beispiele genügen, um einen Einblick in die aussergewöhnlichen Schwierigkeiten beim Sprachverstehen zu bekommen.

Wie sind diese Probleme zu meistern? Ein Weg bei der Uebersetzung von Texten ist der, die Ambiguitäten durch einen menschlichen Vorbearbeiter auszufiltern. Er kann das in seiner Muttersprache machen und muss

keine zwei Sprachen beherrschen. Verwirklicht wurde diese Methode unter anderem bei der Pan-Amerikanischen Gesundheitsorganisation, die auf diese Weise schon über eine Million Worte vom Spanischen ins Englische übersetzt hat. Ebenso wird zur Zeit ein Programm für das Europaparlament erprobt, mit dem Sinn, Reden in die dort zugelassenen sieben Sprachen zu übersetzen. Der Vorteil ist natürlich der, dass die Ambiguitätenauflösung nur einmal, nämlich in der Originalsprache gemacht werden muss. Danach kann in alle Sprachen übersetzt werden.

Die andere Möglichkeit wurde in ihren Grundzügen schon in Bild 1 dargestellt. Bei der morphologischen Analyse wird jedes Wort in seine Grundbestandteile zerlegt (Flexionselemente), bis eine Wurzel gefunden ist, die das Wort eindeutig charakterisiert, egal, ob es in Präsens oder Plusquamperfekt steht. Die Wurzel von «laufen» wäre z.B. «lauf». Ebenso muss aber auch erkannt werden, dass «lief» zu «lauf» gehört.

Die nachfolgende lexikalische Untersuchung ergibt, ob es sich bei dem Wort um ein Verb oder um ein Substantiv handelt, in welcher Zeitform es geschrieben ist usw. Auch hier kann es zu Problemen kommen, denn «Floh» kann die Vergangenheit von «fliehen» sein, ebenso aber eine Tierart bezeichnen. Meist geht aber aus der morphologischen Analyse hervor, was jetzt gemeint ist.

Es folgt die Syntaxanalyse, die man auch «Parsenning» nennt. Der Parser überprüft zunächst einmal, ob der eingegebene Satz aus seiner Sicht grammatikalisch richtig ist. Das heisst, in einer Datenbank sind alle Syntaxregeln gespeichert, die notwendig sind, um einen Satz korrekt aufzubauen. Kommt eine andere Möglichkeit vor, so kann der Computer den Inhalt des Satzes nicht verstehen.

Es kommt also darauf an, mit möglichst wenig Grammatikregeln alle Gegebenheiten abzudecken. Und diese müssen so exakt formuliert sein, dass sie der Rechner auch zur Sprachanalyse einsetzen kann. Dieser Mühe unterzog sich in den sechziger Jahren Noam Chomsky vom MIT. Er schuf eine Anzahl von Regeln, deren mechanische Anwendung alle zugelassenen (aber nur diese) Strukturen erzeugen. Ein Beispiel dafür ist im Bild 2 zu sehen. Hier ist die Grammatik auf 26 Sätze beschränkt, die der Rechner Wort für Wort auseinandernimmt. Realisiert wurde dies in einem Programm von Levinson und Liberman, die damit ein Auskunftssystem für Flughäfen simulieren wollten.

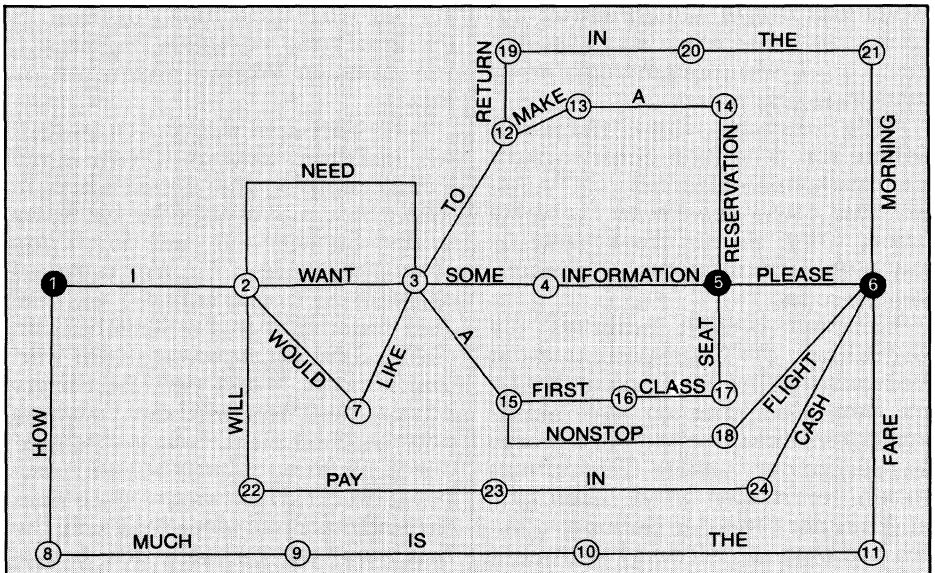


Bild 2: Grammatik mit endlich vielen Regeln
Startpunkt ist ❶, Zielpunkt ❺ oder ❻. Jede Folge möglicher Worte wird im Diagramm als einen der 26 möglichen Sätze erkannt.

Natürlich sind mit diesen Syntaxregeln noch längst nicht alle Probleme des Parsers beseitigt. Ist nämlich die Grammatik richtig erkannt, hat er die verschiedenen Worte einzelnen Satzteilen zuzuordnen. Als Beispiel sei der Satz von vorhin mit der Freundin erwähnt. Sieht man einmal von den moralischen Bedenken ab, so stellt «ich setzte mich zu meiner Freundin und ass sie» einen grammatikalisch völlig korrekten Bestandteil des Satzes dar. Die vollständige Analyse führt aber zu nichts, da der Teil «ich kaufte Apfelsinen» allein für sich im Raum stehen bleibt.

Deshalb müssen immer verschiedene Alternativen untersucht werden, bis der gesamte Satz einen Sinn bekommt. Hierfür gibt es verschiedene Möglichkeiten. Manche Parser arbeiten sich von vorne nach hinten durch den Satz, andere versuchen, mit lokalen Wortkombinationen ans Ziel zu kommen.

Angenommen, der Parser hatte Erfolg und konnte den Satz richtig zuordnen. Der Rechner ist jetzt in etwa im gleichen Stadium wie ein Student, der in der Prüfung nach einer Lösung der Einsteinschen Feldgleichungen gefragt wird. Er hat die Frage wohl verstanden, das heisst aber nicht, dass er automatisch eine Antwort darauf weiss.

Nehmen wir wieder das Beispiel von vorhin. Auf die Frage «Esse ich Obst?» kann der Computer nur dann antworten, wenn er weiss, dass Apfelsinen eine Obstsorte darstellen. Dieses Wissen könnte z.B. durch eine prädikatenlogische Verknüpfung bereitgestellt werden nach dem Schema «alles was eine Apfelsine ist, ist auch Obst». Durch einen geeigneten Inferenzmechanismus, hier beispielsweise durch ein automatisches Beweisverfahren, ist der Rechner dann in der Lage, die Frage richtig zu beantworten.

Wenn der Satz erkannt wurde, wird der Kontext herangezogen, um weitere Informationen zu gewinnen.

Bei komplizierteren Satzgefügen ist das die einzige Möglichkeit, deren Sinn zu verstehen. Aendern wir den Satz in «Wir kauften Apfelsinen...» um. Wer ist jetzt mit «wir» gemeint? Bezieht es sich auf den Schreiber dieser Zeilen und den Leser oder auf den Schreiber und dessen Freundin? Der Sinn ist nur aus dem Kontext heraus ersehbar.

Beinahe unmöglich wird der Versuch der Analyse, wenn Metaphern wie «Der Vorstand tritt zurück» gebraucht werden. Mit unserem Hintergrundwissen ist es uns völlig klar, dass es sich hier um keinen neuen Tanzschritt sondern um das Eingeständnis einer Niederlage handelt.

Die Beschränkungen, denen die Formalisierung der kontextabhängigen Bedeutung unterworfen ist, sind so gross, dass es unter Umständen für immer unmöglich ist, die menschliche Sprache auf einem Computer zu imitieren. Bisher wurden nur sprachverstehende Systeme realisiert, die in ganz speziellen Anwendungsbereichen zu Einsatz kommen. Ein Beispiel hierfür ist auch das Expertensystem SHRDLU, das schon früher angesprochen wurde. Dieses kann zwar jede

Frage aus seiner Klötzchenwelt beantworten, aber auch nur deshalb, weil der Anwendungsbereich eben sehr eingeschränkt ist.

Auch andere Expertensysteme stehen vor derselben Schwierigkeit: auch sie können nicht mit dem Benutzer über dessen seelische Probleme reden, wenn sie für die Erkennung von Lebererkrankungen eingesetzt werden sollen.

Prinzipiell besteht natürlich die Möglichkeit, die Grammatik und die Ambiguitäten bei der Kommunikation mit dem Computer so weit einzuschränken, dass er ohne allzu grosse Anstrengungen unsere Tastatureingaben verarbeiten kann. Allerdings ist dann der Sinn der Uebung, nämlich den Umgang mit dem Elektronengehirn zu erleichtern, verfehlt. Denn bis man sich an die vorgeschriebene Syntax gewöhnt hat, kann man genauso gut eine der angebotenen Computersprachen lernen, die es schon lange auf dem Markt gibt.

Gesprochene Sprache

Nachdem Sie nun die fast unüberwindlichen Schwierigkeiten erahnen können, die sich bei der Verarbeitung geschriebener Sprache auftun, werden Sie sich wohl fragen, ob es überhaupt nützt, sich mit dem noch grösseren Problem der gesprochenen Sprache zu beschäftigen.

Es stimmt zwar, dass bei der Erkennung gesprochener Sprache dieselben Bedingungen zu erfüllen sind wie bei der geschriebenen Sprache. Allerdings geht man hier von einer anderen Zielsetzung aus. Wird bei der geschriebenen Sprache Wert darauf gelegt, dass der Computer möglichst ganze Sätze, auch mit Mehrdeutigkeiten, versteht, so reicht jetzt die Erkennung von einzelnen Worten. Man denke nur an die Möglichkeit, einen Rollstuhl mittels der Kommandos «vorwärts», «rückwärts», «rechts», «links» zu steuern. Damit könnte vielen Behinderten geholfen werden, wieder ein menschenwürdiges Leben zu führen. Und in der Tat gibt es solche Apparaturen schon.

Man sieht also, dass nicht die prinzipiellen Schwierigkeiten, sondern die Anforderungen geringer sind. Und damit ergeben sich sofort völlig neue Anwendungsgebiete für die Sprachverarbeitung.

Wie die Erkennung des gesprochenen Wortes im einzelnen funktioniert, davon soll im folgenden die Rede sein.

Zu diesem Zweck lohnt es, sich erst einmal Gedanken darüber zu machen, wie ein Wort überhaupt zustande kommt und vor allem, wie es sich von anderen Worten unterscheiden lässt.

Beim Sprechen wird in den Lungen ein Luftstrom erzeugt. Dieser sucht sich seinen Weg über den Kehlkopf, in dem sich die Stimmbänder befinden, in den Rachenraum und durch den Mund nach aussen. Ein Wort kann auf dreierlei Art «modelliert» werden. Zum ersten gibt es die Stimmbänder, die zu Schwingungen angeregt werden können. Wenn sich die Stimmbänder berühren, wird der Luftstrom unterbrochen. Der Druck, der sich dadurch hinter den Stimmbändern aufbaut, drückt diese wieder auseinander und das Spiel wiederholt sich. Die Zeitdauer, mit der sich das Ganze wiederholt, ist charakteristisch für die Höhe des dabei entstandenen Tons. Ein Beispiel dafür sind die Vokale.

Die zweite Möglichkeit zur Tonvariation besteht in der Fähigkeit, die akustischen Gegebenheiten im Mund-, Nasen- und Rachenraum zu verändern, indem irgendwo an diesen Stellen Verengungen geschaffen werden, die schmal genug sind, um eine Turbulenz im Luftstrom hervorzurufen. Dazu ein Beispiel: Wird der Luftstrom durch einen Engpass zwischen der oberen Zahnreihe und der Unterlippe gezwängt, so entsteht der Laut «f».

Die Laute, die in der Mundgegend erzeugt werden, sind in der Regel aperiodisch und stimmlos. Durch eine Kombination mit Stimmbandschwingungen kommt man z.B. zum stimmhaften «s».

Die dritte Möglichkeit besteht darin, den Luftstrom zeitweise völlig zu unterbrechen, wie dies bei «p» und «t» der Fall ist.

Um das alles für den Computer erfassbar zu machen, muss man ein mathematisches Modell dafür entwickeln. Zu diesem Zweck wird die Akustik des Sprechorgans durch eine schallharte Röhre simuliert, bei der verschiedene Querschnitte unterschiedlich hohen Tönen entsprechen. Desweiteren wird angenommen, dass ein Sprechlaut eine periodische Schwingung darstellt, was in recht guter Näherung auch korrekt ist. Damit ist die Theorie der Fourieranalyse anwendbar. Diese besagt, dass jede periodische Schwingung darstellbar ist durch eine Ueberlagerung von Sinuswellen mit verschiedenen Frequenzen und Amplituden. In der Regel genügen einige charakteristische Frequenzen zur Identifikation eines Lauts. In dieser Weise bekommt man ein Amplitudenspektrum, das weiter untersucht werden kann. Bild 3 stellt solche Spektren dar.

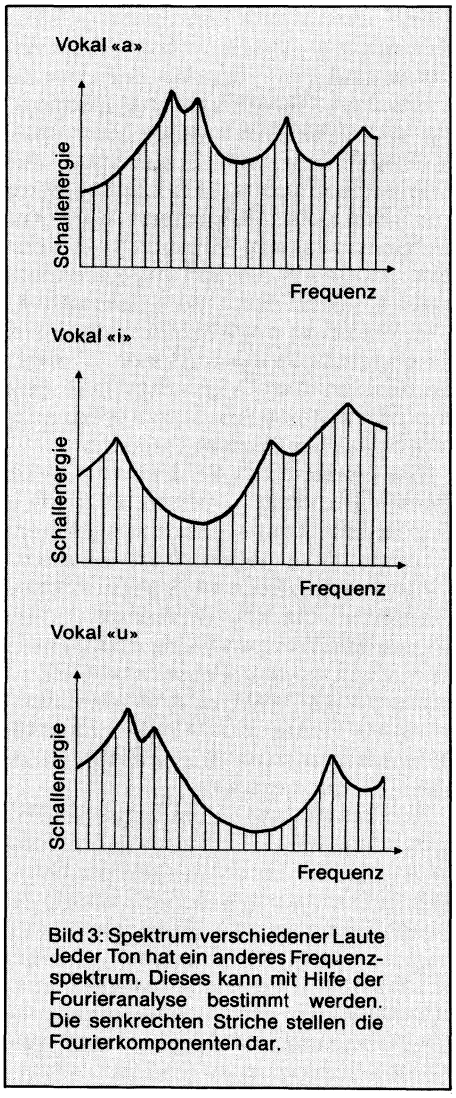


Bild 3: Spektrum verschiedener Laute
Jeder Ton hat ein anderes Frequenzspektrum. Dieses kann mit Hilfe der Fourieranalyse bestimmt werden. Die senkrechten Striche stellen die Fourierkomponenten dar.

Mit diesem mathematischen Rüstzeug lässt sich die Wirkung des menschlichen Stimmapparats gut beschreiben: er weist alle Elemente eines linearen Filters auf. Die Laute, die verschiedenen Filtern entsprechen (dargestellt durch z.B. unterschiedliche Zungenpositionen), lassen sich dann mathematisch in Form einer sogenannten Uebertragungsfunktion erfassen. Das ist die Funktion, die beschreibt, wie ein Laut auf seinem Weg zwischen Stimmbändern und Zähnen beeinflusst wird. Die Uebertragungsfunktion hat in der Mehrzahl der Fälle eine Anzahl leicht erkennbarer Maxima, in denen der Grossteil der Energie des Sprachsignals konzentriert ist.

Die Uebertragungsfunktion sei jetzt bekannt. Nun muss sie natürlich ausgewertet und den verschiedenen erlaubten Worten zugeordnet werden.

Dabei wird angenommen, dass sich die Parameter der Uebertragungsfunktion in einem Zeitraum von 5 bis 20 ms aufgrund der begrenzten Schwinggeschwindigkeit der Sprech-

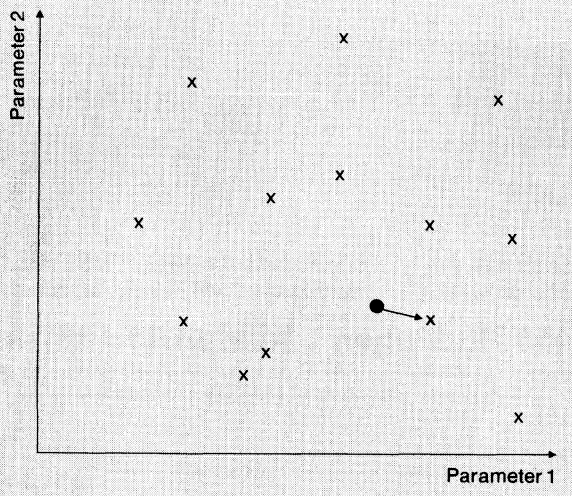


Bild 4: Vergleich mit den Referenzworten
Das eingegebene Wort wird beim Vergleich als das Referenzwort erkannt, das den kleinsten Abstand hat (siehe Pfeil).

organe. Es reicht also, nur etwa alle 10 ms die Filterparameter aus dem Sprachsignal neu zu bestimmen. Dies geschieht, wie bereits angedeutet, mit Hilfe der Fourieranalyse, für die es sehr leistungsfähige Algorithmen gibt.

Um die Aufgabe zu erleichtern, werden (mathematische) Filter nachgeschaltet, die das Sprachsignal in einzelne Frequenzbänder aufteilt. In der Praxis kommen 20 bis 30 Filter zur Verwendung. Das Verfahren ist bekannt unter der Bezeichnung «Filterbankmethode».

Bis zu dieser Stelle ist aber nur die grobe Vorarbeit geleistet, das Signal wurde mathematisch so aufbereitet, dass der Computer mit ihm arbeiten kann. Denn nun müssen die einzelnen Worte und Phoneme erkannt werden. Voraussetzung hierfür ist vor allem, dass zwischen verschiedenen Worten eine Pause eingelegt wird, denn sonst können sie nicht oder nur sehr schlecht getrennt werden. In der Tat existiert bisher noch kein Programm, das fliessend gesprochene Sprache «versteht».

Wesentlich ist vor allem die Stärke des Signals. Uebersteigt es einen bestimmten Pegel, so wird daraus auf einen Wortanfang geschlossen. Da ein Wort aber auch mit stimmlosen Lauten wie «s» in «Süden» beginnen kann, geht der Rechner zuerst einmal in die Vergangenheit des Sprachsignals und untersucht, ob sich dort ein Anzeichen für einen solchen Laut findet. Ebenso wird das Signal zeitlich in Richtung Zukunft beobachtet. Denn ein Absinken der Energie kurz nach dem vermeintlichen Wortanfang deutet eher auf ein Hintergrundgeräusch hin, das unterdrückt werden muss. Der Anfang des Wortes wird daraufhin woanders gesucht.

Dasselbe geschieht bei der Detektion eines Wortendes. Es folgt der Vergleich des Spektrums zwischen Wortanfang und -ende. Die zeitlich aufeinanderfolgende Anzahl von Filterparametern wird in Form von Vektoren festgehalten.

Es folgt der Vergleich mit Referenzworten, die vom Benutzer vorher dem Computer eingegeben werden müssen. Diesen Vorgang kann man gewissermassen als die Lernphase des Rechners bezeichnen, da er hier die Möglichkeit erhält, aufgenommene Laute mit den gespeicherten Worten zu identifizieren.

Damit steht der Computer nun vor dem Problem, mit dem Sie sicher schon lange gerechnet haben: jeder Benutzer hat eine andere Stimme und Betonung. Die Betonung ist sogar abhängig vom Kontext der der speziellen seelischen Situation. Desweiteren kann die Sprechgeschwindigkeit nicht unbeträchtlich variieren. Es können sich also für dasselbe Wort durchaus unterschiedliche Signalverläufe ergeben. Ebenso ist es aber möglich, das verschiedene Worte ähnliche Signalverläufe haben. Der Rechner muss also meist unter verschiedenen Möglichkeiten auswählen.

Es gibt aber eine Methode, um zumindest die unterschiedliche Sprechgeschwindigkeit in den Griff zu bekommen. Zu diesem Zweck wird das Referenzwort an einigen Stellen solange gedehnt oder komprimiert, bis eine im mathematischen Sinn optimale Uebereinstimmung gefunden ist.

Die Vektoren, die z.B. n verschiedene Parameter repräsentieren, werden in einen n-dimensionalen Vektorraum abgebildet. Dies ist in Bild 4 geschehen, aus abbildungstechnischen Gründen ist hier n=2. Die Kreuze bedeuten die Referenzworte, der Punkt soll das zu erkennende Wort darstellen. Die Summe der Quadrate der geometrischen Abstände zwischen Punkt und den jeweiligen Kreuzen ist ein Mass für die Wahrscheinlichkeit, dass Kreuz und Punkt dasselbe Wort verkörpern.

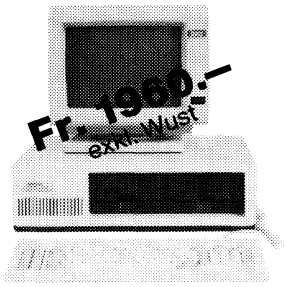
Je nach Kompliziertheit des Spracherkennungssystems wird jetzt das zu untersuchende Wort dem Referenzwort zugeordnet, welches ihm am nächsten steht, also den kleinsten Abstand hat. Die andere Möglichkeit ist die, dass verschiedene Referenzworte ausgewählt werden, die einen relativ kleinen Abstand haben. Das richtige Wort wird dann durch semantische Interpretation wie beim Erkennen geschriebener Sprache ermittelt.

Da hier, wie gesagt, mit Wahrscheinlichkeiten gearbeitet wird, liegt es auf der Hand, dass mit Vergrösserung der Anzahl von Referenzworten auch die Wahrscheinlichkeit zunimmt, ein falsches Wort zu erkennen. Und das ist auch der Hauptgrund, dass sprachverstehende Programme selten einen Wortschatz von mehr als 100 Worten haben. Erst bei einem Wortschatz von etwa zehn Worten kann man relativ sicher gehen, dass das System auch richtig reagiert. Aber wie schon betont, reicht dieser Wortschatz aus, um einen Rollstuhl zu steuern. □

PC-Software?

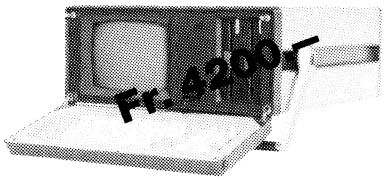
Weit über 100 in der Schweiz erhältliche PC-Programme für alle denkbaren Anwendungen wurden im COMPUTERMARKT, der Schwesterzeitschrift von M+K, bereits vorgestellt. Und Nummer für Nummer kommen neue dazu. Ueberhaupt, wenn Sie mehr wissen wollen, was in der Schweizer PC-Branche läuft und wer wo was anbietet, sollten Sie COMPUTERMARKT noch heute abonnieren. Als Abonnent von M+K erhalten Sie das CM-Abo vergünstigt für nur Fr. 15.--. Abo-Bestellkarte finden Sie auf Seite 3 in diesem Heft.

ECO-PC/XT



8088 CPU, 256 K-RAM, 2 Floppies
Tastatur, Monitor

Industrie-Portabel



XT- oder AT-kompatible Versionen,
Konfiguration nach Kundenwunsch.

403-AT

80286 CPU, 6/8 MHz, 1 MB-RAM, 20 MB
Harddisk, 1,2 MB Floppy, Tastatur, Monitor

beltronic Im Chapf 8455 Rüdlingen Telefon 01 / 867 31 41

PC-PRINTER-POINT

PC-PRINTER-POINT · Postfach 212 · CH-8100 HEILSAU 1 · Tel. 071/52 21 22

PC-PRINTER-POINT,
die mit den neuen konsumentenfreundlichen PLUSPUNKTEN:

- keine mysteriösen Grauimporte
- 12 Monate Garantie auf Drucker
- 6 Monate Garantie auf Computer und Zubehör
- 24 Monate Garantie auf CITIZEN-Produkte
- qualifizierter REPARATUR-DIENST
- SOFORTIGE Lieferung nach Geldeingang
- Lieferung «FREI HAUS CH»
- KOSTENLOSES LEIHGERÄT, falls die Garantiereparatur mehr als 6 Werktage beanspruchen sollte
- KOSTENLOSES PC-Druckerkabel inklusive,
- KOSTENLOSE Produktinformation und Auskünfte unter **071/52 21 22**

AKTIONSPAKET

KAYPRO PC **voll IBM-kompatibel,** 8088 CPU, 2×360 KB-Diskettenlaufwerke, 760 KB RAM durch Multifunktionskarte mit Echtzeituhr, 12-Zoll-Grün-Monochrom-Monitor, Multivideo-Adapter, IBM-AT-Tastatur-Auslegung, ser. und par. Schnittstelle, 5 freie Steckplätze, WORDSTAR, Mailmerge, MITE, Polywindows, GW-Basic, MS-DOS 2.11, Handbücher 4450.–jetzt **sFr. 3750.–**

CITIZEN MSP 10 **8 KB-Druckpuffer, IBM- und EPSON-kompatibel,** 160/40 CPS, voll BIT-IMAGE-graphikfähig, deutschsprachiges Handbuch, PC-Druckerkabel 1490.–jetzt **sFr. 1150.–**

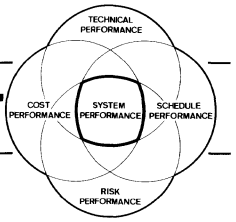
KOMPLETTPAKET KAYPRO PC **(6 Monate Garantie)** und CITIZEN MSP 10 **(24 Monate Garantie)** inkl. Kabel 5940.–jetzt **sFr. 4450.–**

Druckerprodukte		ATARI-Komplettsysteme			
		520 STM	520 STM	1040 STF	1040 STF
Marken/Modelle	Zoll/EDV/NLQ	Mono	Color	Mono	Color
SEIKOSHA:					
MP 1300 AI	10/300/100	2790	3250	3350	3790
MP 1300 AI Color	10/300/100	2950	3390	3490	3950
MP 5300 AI	15/300/100	3090	3550	3690	4150
FUJITSU:					
DMPG 9	10/180/25	1990	2490	2590	3490
DX 2100	10/220/44	2390	2850	2950	3250
DX 2200	15/220/44	2650	2990	3090	3590
STAR					
NL 10 Interface	10/120/30	2150	2590	2690	3090
NX 15 (NEU!)	15/120/30	2590	2990	3090	3490

Bitte beachten Sie auch unsere **sensationellen DRUCKER-ANGEBOTE** in den NOVEMBER-AUSGABEN der Computerzeitschriften MEIN COMPUTER und COMPUTERMARKT!
Und vergessen Sie nicht beim Preisvergleichen: ein **kostenloses PC-Druckerkabel** und die PC-PRINTER-POINT **konsumentenfreundlichen Pluspunkte!**

★★★ **Angebote nur so lange wie der Vorrat reicht** ★★★

PROJEKTMANAGEMENT



Die System Lösung mit **PMS-II:**

- NETZPLAN PROZESSOR
- EINSATZMITTEL MANAGEMENT
- MATERIAL, DIENSTLEISTUNGS, und STÖRFAKTOREN MANAGEMENT
- RELATIONALE DATENBANK mit
  - Kontrakt-Administration
  - Projekt-Strukturplan
  - Text Managemnet & Dokument Retrieval System
  - Lotus & Multiplan Schnittstellen
- GRAFIK DRUCKER + PLOTTER-PROGRAMME
- LOKALES NETZWERK

Operationell auf: MS-DOS, CP/M-86, UNIX System V

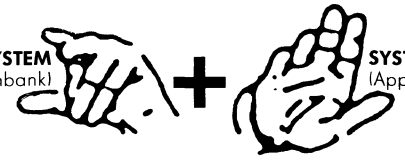
Die professionelle **SISTA** System Lösung

SISTA System Engineering SA ☎ (01) 813 26 35
Ziegeleistr. 63 A · CH-8426 Lufingen Tx 825 635 txkch

AVATECH PICK

INVESTIEREN SIE SICHER UND EFFIZIENT IN IHRE SOFTWARE-ZUKUNFT

PICK BETRIEBSSYSTEM
(Relationale Datenbank)



SYSTEM BUILDER
(Applikationsgenerator)

KOSTENREDUKTION durch einfache und effiziente Erstellung von Anwender-Software dank 4. Generation Tools
MULTIUSER auch für PC XT/AT (bis zehn Benutzer)
ERPROBT in über 60'000 Installationen weltweit
KOMPATIBEL zu Hardware von über zwanzig verschiedenen Herstellern vom PC über Minis bis Mainframes

AVATECH AG
Feldeggstr. 26, CH-8008 Zürich
Tel. 01/2515115, Tlx. 816029 AVA CH

Grafik mit FORTH (2. Teil)

Da wir uns auf die Universalität aller FORTH-Systeme berufen, die zur Zeit auf dem Markt erhältlich sind, wollen wir in diesem Teil einige grundlegende Grafiktechniken des M&T-FORTH beschreiben. M&T-FORTH ist im Prinzip ein erweitertes FIG-FORTH. Im Gegensatz zum FORTH der vorhergehenden Ausgabe sind in dieser Variante des FIG-FORTH die Grafikbefehle nicht im Standardvokabular enthalten, sondern müssen entweder von Screens nachgeladen werden, oder direkt in das Wörterbuch als neue Befehle aufgenommen werden.

Heinz Kastien

Die zur Programmierung von Grafik erforderlichen Befehle sind im Manual beschrieben, sodass einer Verwendung dieser Begriffe nichts im Wege steht. Da zur Definiton der neuen Grafikbefehle nur Ausdrücke des Standard-FORTH verwendet werden, können diese Grafikbefehle auch in jedes andere FORTH-System implementiert werden. Das Standard-FORTH wird um die nachfolgenden Worte erweitert.

GRAPHIK	Initialisiert den HRG-Grafikmode
NORMAL	Schaltet in den Textmode zurück
LOESCHEN	Löscht den Bildschirm
FARBE	Setzt die Farbe des Hintergrundes und der Punkte
INVERS	Invertiert den Bildschirm
BITADR	Definiert aus den Koordinaten X und Y ein Bitmuster
SET	Setzt einen Punkt
RESET	Löscht einen Punkt

Darüberhinaus stehen noch eine Reihe von Befehlen zum Programmieren von Sprites und sogar ein Spriteeditor zur Verfügung.

Diese acht Begriffe sollen in einem Screen mit dem Namen «Graphik» und mit dem Screen #1 gespeichert werden. Das Arbeiten mit Screens wurde schon in M+K 86-4 behandelt. Obwohl sich die Behandlung der Screens in den diversen FORTH-Systemen unterschiedlicher Worte bedient, ist die Technik immer die gleiche. Ein Screen wird über einen Editor erzeugt und nach erfolgter Editierung abgespeichert. Das Screen kann innerhalb eines Programms oder auch selbstständig aufgerufen und in den Arbeitsspeicher geladen werden, hierbei werden alle im Screen enthaltenen neuen Befehle temporär in das Wörterbuch integriert.

Das erste erstellte FORTH-Wort lautet «GRAPHIK» und initialisiert die HRG. Hierzu muss dem Rechner mitgeteilt werden, wo sich der Grafik- bzw. Videoschirm befindet. Ebenfalls muss der Bit-Map-Mode zur Erzeugung der Grafik eingeschaltet werden. In unserem Fall legen wir den Grafikbildschirm auf die Adresse $6000-$FFFF (24576-65520) und das Video-RAM, das für die Punkt- und Hintergrundfarben verantwortlich ist, auf die Adressen $5000-$5FFF (20480-24575). Die Initialisierung der Grafik erfolgt mit den beiden Speicherstellen 53265 und 53272. In die Speicherstelle 53265 ($D011) muss der Wert 59 ($3B) geschrieben werden, um von der Textdarstellung in die hochauflösende Grafik zu gelangen, mit der Speicherstelle 53272 ($D018) wird auf Grossschrift umgeschaltet. In diese Speicherstelle wird der Wert 121 ($79) geschrieben. Da alle nachfolgenden Operationen mit Hexadezimalzahlen durchgeführt werden, muss in Zeile 1 zuerst auf diese Zahlenbasis umgeschaltet und erst danach das Wort «GRAPHIK» definiert werden.

In den Zeilen 3 und 4 werden die beiden Speicherstellen definiert und schliesslich wird in Zeile 5 das Bit 14 des Video-RAM auf 1 gesetzt. Hierzu wird die Speicherstelle 56676 auf den Stack gelegt, dupliziert und der Inhalt dieser Speicherstelle mit $FE (254) AND verknüpft. Nach Vertauschen der beiden obersten Stackplätze wird der Wert abgespeichert. Mit dem Wort «Graphik», das in das Wörterbuch integriert wird, kann nun bei allen zukünftigen Grafiken die HRG initialisiert werden.

```
HEX
:_GRAPHIK
79_D018_C!          Setzen des
                    Bit-Map-Mode
3B_D011_C!          HRG
                    Einschalten
DD00_DUP_C@_FE_AND_SWAP_C!_;
```

Soll wieder in den Normaltextmode zurückgeschaltet werden, müssen lediglich die alten Parameter wieder gesetzt werden. Dies erfolgt mit der Befehlsfolge:

```
:_NORMAL
15_D018_C!
1B_D011_C!
DD00_DUP_C@_03_OR_SWAP_C!_;
```

Fast ebensowichtig wie die Initialisierung ist das Löschen des Bildschirms. Hierzu steht der Befehl «ERASE» zur Verfügung, dieser Maschinensprachebefehl löscht jeweils n Bytes ab der angegebenene Adresse. Da der Bildschirmspeicher von uns mit $2000 Bytes ab der Adresse $6000 definiert worden ist, muss der Befehl zum Löschen des Bildschirms wie folgt aussehen.

```
:_LOESCHEN
6000_2000_ERASE_;
```

Auch der Befehl «LOESCHEN» kann entweder in das Wörterbuch aufgenommen werden, oder jeweils ab dem Screen zugeladen werden.

«FARBE» ist unser FORTH-Wort zur Definition der Farbe des Bildschirmhintergrundes und der Zeichen. Die Codes für die Farben sind im Manual beschrieben und entsprechen dem im BASIC verwendeten Code. So wird mit Code 0A der Hintergrund Schwarz und die Zeichen Rot und mit Code 14 der Hintergrund Weiss und die Zeichen Violett. Der Farbencode errechnet sich im Dezimalsystem nach der Formel:

Farbencode = Hintergrundfarbe
+ 16 * Punktfarbe

Der Code muss noch in eine Hexadezimalzahl umgewandelt werden. Der Wert der Punktfarbe wird vom Stack geholt, mit 16 ($10) multipliziert und zum Wert der Hintergrundfarbe addiert. Die Werte $5C00 (23552) und

Das Symbol «_» repräsentiert einen Leerraum (Space), der sehr wichtig ist und auf keinen Fall vergessen werden darf. Ein Space zwischen zwei Worten, gibt FORTH die Möglichkeit beide zu unterscheiden und zu trennen. Bevor sie ausgeführt werden, kommen sie nämlich in einen Zwischenspeicher (Terminal-Input-Puffer) und FORTH hätte ohne einen Leerraum zwischen den einzelnen Befehlen keine Möglichkeit, sie zu interpretieren.

```
BIN 000000000000000000000000. HEX  0 EN
BIN 000000000000000000000000. HEX  1 EN
BIN 000000000000000000000000. HEX  2 EN
BIN 000000000000000000000000. HEX  3 EN
BIN 000000000000000000000000. HEX  4 EN
BIN 000000000000000000000000. HEX  5 EN
BIN 000000000000000000000000. HEX  6 EN
BIN 000000000000000000000000. HEX  7 EN
BIN 000000000000000000000000. HEX  8 EN
BIN 000000000000000000000000. HEX  9 EN
BIN 000000000000000000000000. HEX 10 EN
BIN 000000000000000000000000. HEX 11 EN
BIN 000000000000000000000000. HEX 12 EN
BIN 000000000000000000000000. HEX 13 EN
BIN 000000000000000000000000. HEX 14 EN
BIN 000000000000000000000000. HEX 15 EN
BIN 000000000000000000000000. HEX 16 EN
BIN 000000000000000000000000. HEX 17 EN
BIN 000000000000000000000000. HEX 18 EN
BIN 000000000000000000000000. HEX 19 EN
BIN 000000000000000000000000. HEX 20 EN
```

Abb. 1 Spriteeditor

```
M 000000000000. HEX  0 EN
M 000000000000. HEX  1 EN
M 000000000000. HEX  2 EN
M 000000000000. HEX  3 EN
M 000000000000. HEX  4 EN
M 000000000000. HEX  5 EN
M 000000000000. HEX  6 EN
M 000000000000. HEX  7 EN
M 000000000000. HEX  8 EN
M 000000000000. HEX  9 EN
M 000000000000. HEX 10 EN
M 000000000000. HEX 11 EN
M 000000000000. HEX 12 EN
M 000000000000. HEX 13 EN
M 000000000000. HEX 14 EN
M 000000000000. HEX 15 EN
M 000000000000. HEX 16 EN
M 000000000000. HEX 17 EN
M 000000000000. HEX 18 EN
M 000000000000. HEX 19 EN
M 000000000000. HEX 20 EN
```

Abb. 2 Multicolorsprite

$400 (1024) werden auf den Stack gelegt, vertauscht und mit dem Befehl FILL ab der Adresse $5C00 1024 ($0400) Speicherstellen mit dem Wert $07 gesetzt.

```
:_FARBE
10_*_+_5C00_400_ROT_FILL_;
```

Um einen Punkt auf dem Bildschirm zu setzen oder zu löschen, müssen seine X- und Y-Koordinaten bekannt sein. X kann Werte zwischen 1 und 320 annehmen, Y zwischen 1 und 160. Diese X- und Y-Koordinaten müssen nun in ein Bitmuster und eine Adresse umgerechnet werden. In BASIC erfolgt dies mit dem nachfolgenden Programm. Prinzipiell beschreiten wir in FORTH den gleichen Weg. Die Programmbeispiele zeigen die Berechnung der Adresse und des Bitmusters in BASIC und in FORTH.

BASIC-Programm zur Berechnung der Adresse und des Bimusters aus den X- und Y-Koordinaten.

```
60000 YK = 320 * INT(Y/8)
        + INT((Y/8-INT(Y))/8)
60010 XK = 8-INT(X/8)
60020 EX = 2↑(7-INT((X/8)
        - INT(X/8)*8)
60030 S  = 8192 + YK + XK
60040 POKE S,PEEK(S) OR EX
```

Nächsten Monat gibt's wieder

COMPUTER MARKT

mit aktuellen Informationen.

Der gleichen Befehlsfolge entspricht dieses FORTH-Programm

```
:_BITADR
DUP_7_AND_SWAP_8_/
140_*_!_6000_OVER_8
/_8_*_+_
SWAP_7_AND_7_SWAP_--
DUP_IF
1_SWAP_0
DO_DUP_+_LOOP
1+
ENDIF
SWAP_DUP_0<
ID._«_Illegal Quantity Error»
QUIT
ENDIF_;
```

Mit dem errechnete Punkt, dessen Adresse und Bitmuster mit dem Befehl «BITADR» auf den Stack gelegt worden sind, lässt sich nun ein Punkt auf dem Bildschirm setzen oder löschen. Die Worte hierfür sind:

SET mit der Befehlsfolge

```
: SET_BITADR_SWAP_OVER_C@_
                    OR_SWAP_C!_;
```

RESET mit der Befehlsfolge

```
: RESET_BITADR_SWAP_FF_XOR_
        OVER_C@_AND_SWAP_C!_;
```

Der Unterschied des Punktes Setzen (SET) oder Löschen (RESET) besteht in der Exclusiv ODER-Verknüpfung des Speicherinhaltes mit $FF und der UND-Verknüpfung mit dem alten Speicherinhalt.

Vielfach wird man den Wunsch haben, den Bildschirminhalt zu invertieren. Die Definition eines Befehls, der den Inhalt des Bildschirms umkehrt, ist relativ einfach, da hierzu lediglich alle Positionen des Bildschirmspeichers mit $FF (255) Exclusiv ODER (EXOR) verknüpft werden müssen. Hierzu bedient man sich des FORTH-Wortes «TOGGLE». Toggle vernüpft ab der Speicherstelle Adr. n1 Byte mit dem Wert n2 und schreibt den neuen Wert in Adr. TOGGLE ist mit der Befehlssequenz

```
ADR_N1_OVER_@_XOR_SWAP_!
```

gleichbedeutend. Der gesamte Wortlaut für den neuen Begriff «INVERS» lautet somit:

```
: INVERS_2000_0_DO_I_6000_+_FF_
                    TOGGLE_LOOP_;
```

Ab der Speicherstelle $6000 werden mit der Schleife 8192 Bytes mit $FF EXOR verknüpft. Bei jedem Schleifendurchlauf wird der Wert der Laufvariablen I zur Adresse $6000 addiert und der erhaltene verküpfte Wert wieder unter der alten Adresse abgespeichert.

Die Commodore-Rechner C-64 und 128 verfügen über Sprites. Sprites sind selbstdefinierte Figuren, die monochrom oder mehrfarbig in einem 21*24 Punkte Raster entworfen werden können und die sich beliebig

über den Bildschirm verschieben lassen. Gerade Spiele lassen sich mit Sprites sehr komfortabel gestalten. FORTH bietet mit seiner grossen Verarbeitungsgeschwindigkeit eine ideale Ergänzung zu den Programmiertechniken der Sprites.

Mit den hier beschriebenen FORTH Grafik-Befehlen können vier Sprites im Bereich

$02C0 - 02FE (704 - 766) Sprite 11
$0340 - 037E (832 - 894) Sprite 13
$0380 - 03BE (896 - 958) Sprite 14
$03C0 - 03FE (960 - 1022) Sprite 15

definiert werden. Für eine grössere Anzahl von Sprites muss der Videocontroller und das Bildschirm-RAM verschoben werden, da sonst zu wenig Speicherplatz zur Verfügung steht. Ein Spriteeditor in FORTH sieht dann folgendermassen aus:

```
HEX_0_VARIABLE_SPADR
:_BIN_2_BASE_!_;
:EN_3_*_SPADR_@_+_DUP_1+
SWAP_ROT_SWAP_C!_OVER_OVER
SWAP_100_/_SWAP_C!_1+_SWAP
FF_AND_SWAP_C!_;
:_SPRITE_HEX_SPADR_!_15_0_DO
CR . «_BIN»
.«_00000000000000000000000._HEX_»
I_2_._R_.«_EN»_LOOP_CR_;
```

Der Spriteeditor wird mit DECIMAL 704 SPRITE aufgerufen, worauf auf dem Bildschirm der Ausdruck gemäss Abb. 1 erscheint.

Mit dem Cursor kann nun das Sprite editiert werden, indem an allen Positionen, an denen später ein Punkt erscheinen soll, eine «1» eingetragen wird. Ist das Sprite fertig entworfen, so geht man mit der HOME-Taste in die linke obere Zeile und quittiert jede Zeile mit RETURN. Das Sprite wird hiermit ab der Adresse 704 abgespeichert.

Will man ein Multicolorsprite entwerfen, so müssen die letzten vier Zeilen des Editors wie folgt geändert werden. Den Ausdruck sehen Sie in Abb. 2.

```
:_M_4_BASE_!_;
:_MSPRITE_HEX_SPADR_!_15_0_DO
CR_._«_M_»
.«_000000000000._HEX_»
I_2_._R_.«_EN»_LOOP_CR_;
```

Es können nun an allen Positionen des Sprites Zahlen mit den Werten 0-3 eingetragen werden, hierbei bedeuten

0 = Farbe des Hintergrundes
= Spritefarbe
1 = Multicolorfarbe 1
2 = Multicolorfarbe 2
3 = Multicolorfarbe 3

N1 Befehl		N1 ist die Nummer des Sprite
:_SPRAN_D015_AN_;	(53269)	Schaltet Spritedarstellung ein
:_SPRAUS_D015_AUS_;	(53269)	Schaltet Spritedarstellung aus
:_XEXAN_D017_AN_;	(53271)	Dehnung des Sprite in der Y Achse an
:_XEXAUS_D017_AUS_;	(53271)	Dehnung des Sprite in der Y Achse aus
:_YEXAN_D01D_AN_;	(53277)	Dehnung des Sprite in der X Achse an
:_YEXAUS_D01D_AUS_;	(53277)	Dehnung des Sprite in der X Achse aus
:_MULTIAN_D01C_AN_;	(53276)	Schaltet Multicolor-darstellung ein
:_MULTIAUS_D01C_AUS_;	(53276)	Schaltet Multicolor-darstellung aus

```
:_SETMULTI_100_*_+_D015!_;
```
Setzt die Farben N1 und N2 in das Multicolorspriteregister
Stackrelation : N1 N2 SETMULTI

```
:_SETSPRCOLOR_D027_+_C!_;
```
Färbt das N2. Sprite mit der Farbe N1
Stackrelation : N1 N2 SETSPRCOLOR

```
:_MOVESPR
  OVER_FF_>_IF
  DUP_D010_AN
  ENDIF
  2_*_D000_+_ROT_OVER_C!_1+_C1_;
```

Bewegt das N1 Sprite zu den Koordinaten X und Y.
Stackrelation : X Y N1 MOVESPR

Abb. 3 Stackrelation

Ist ein Sprite erstellt und abgespeichert, muss dem Videocontroller mitgeteilt werden, auf welcher Adresse das Sprite liegt. Dies erfolgt mit dem Befehl:

```
:_SETSPADR_40_/_>R_0400_+_+_
                    R>SWAP_C_;
```

ADR1_N1_ADR2_SETSPADR teilt dem Videocontroller die Adresse und die Spritenummer mit. Hierbei ist ADR1 die Basisadresse des Bildschirms, normalerweise $0400, N1 die Nummer des Sprites und ADR2 die Spriteadresse.

Die nachfolgenden Befehle, mit denen die Spriteprogrammierung wesentlich vereinfacht wird, zeigen nochmals die Universalität des FORTH bei der Definition eigener neuer Worte. Hierzu sind jedoch einige Hilfsbefehle erforderlich.

```
:_EX2
DUP_IF
SWAP_1+_1_SWAP_1
DO_OVER_*_LOOP_SWAP
DROP
ELSE
1+
ENDIF_;
```

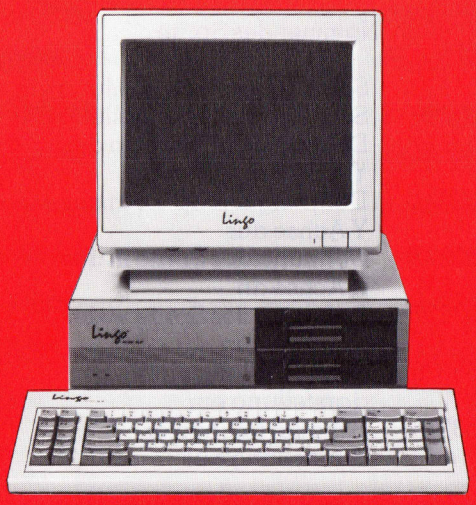
Dieses kurze Programm errechnet die Zweierpotenz von N1.

```
:_AN_SWAP_EX_2_OVER_C@_OR_
                    SWAP_C!_;
:_AUS_SWAP_EX»_FF_SWAP_--_
        OVER_C@_OVER_C@_AND
                    _SWAP_C!_;
```

Diese beiden Hilfsformen setzen oder löschen das N1 Byte der Adresse ADR. Mit diesen beiden Hilfsbefehlen können nun eine Reihe von FORTH-Worten zur Darstellung von Sprites definiert werden. Alle nachfolgend genannten Befehle haben die Stackrelation gemäss Abb. 3.

Diese einfachen Beispiele zeigen, dass auch mit FORTH-Systemen, die in ihrem Vokabular keine eigenen Befehle für eine HRG haben, diese trotzdem programmiert werden kann. In der nächsten Folge besprechen wir einige wichtige Anwendungen der HRG, wie z.B. das Zeichnen von Linien und Kreisen, um dann auf das Rechnen mit Floating Point Variablen überzuwechseln. □

Der kleine Grosse!



Lingo PC-88/XT

(Made in Singapore)

- **Standardversion mit 640K RAM NEC V20-Prozessor 4,77/7,33 MHZ, Serial- und Parallel-Schnittstelle, (2. Serial-Schnittstelle optional), Uhr mit Kalender, Game-Adapter, 135 Watt-Netzteil, Floppy-Controller, Monochrome Graphikkarte, 12 Zoll-Monitor, Tastatur VSM oder international, MS-DOS 3.2 mit Handbuch, GW-Basic.**

– Version mit 2 Floppies 360 KB	sFr. 2995.–
– Version mit 1 Floppy 360 KB und Hard Disk 20 MB 5,25 Zoll	sFr. 3995.–
– Version mit 2 Floppies 360 KB und 20 MB Hard Disk 3,5 Zoll	sFr. 4695.–

Die Preise verstehen sich ab Weiningen inkl. Wust. Schulen, Lehrer, Studenten, Grossabnehmer und Wiederverkäufer erhalten interessante Rabatte.

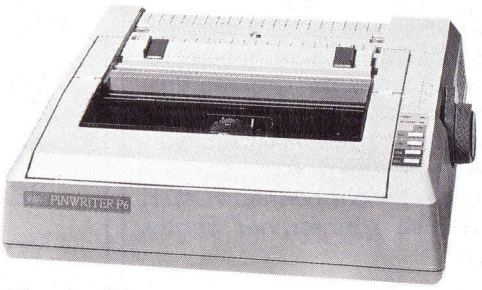
Wir führen eine grosse Auswahl an Zusatzprodukten für den PC.

swiss PC **Als Hersteller des Swiss PC verfügen wir über ein sehr grosses Know-how im Bereich PC. Sie erhalten bei uns einen einmaligen technischen Support.**

Twix Equipment AG

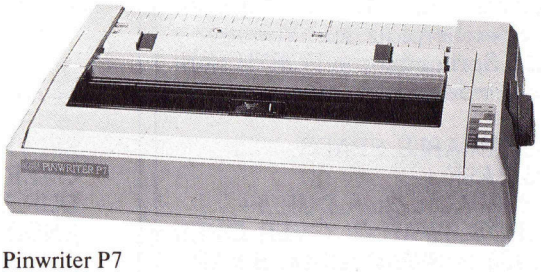
PC-Center
Gewerbezentrum
8104 Weiningen
Telefon 01/750 11 10
Telex 829 003 twxw

Twix



Pinwriter P6

Zeigt bei Grafik starke Seiten.



Pinwriter P7

NEC Pinwriter P6/P7.
Mit dem 24-Nadel-Präzisionsdruckkopf der neuen, bis zu 216 Zeichen/Sek. schnellen Punktmatrixdrucker Pinwriter P6 und P7, die sich lediglich durch die Schreibbreite (P6: 80 Zeichen pro Zeile/P7: 136 Zeichen pro Zeile) unterscheiden, bieten sich vielseitige Möglichkeiten für besseren Druck. Grafiken werden durch eine feine Auflösung von 360 x 360 Punkte/Zoll nuanciert und detailgenau wie noch nie wiedergegeben. Vielfältige Papierzuführungsoptionen, die optionale Colorausstattung und ein auf 40 KByte erweiterungsfähiger 8 KByte-Buffer zählen neben der geringen Geräuschentwicklung (53 dBA im Quiet Mode bei halber Geschwindigkeit) weiterhin zu den starken Seiten dieser neuen Drucker der Extraklasse.

SYSDAT

SYSDAT Computer Products AG
Bern – Stationsweg 5, CH-3627 Heimberg
Tel.: 0 33/37 70 40, Telex: 921 310
Telefax: 0 33-37 80 20

Zürich – Rietstr. 2
8103 Unterengstringen
Tel.: 01/7 50 51 41
Telefax: 01-7 50 55 01

NEC

Schnelle Ellipsendarstellung auf Rasterbildschirmen

Immer wieder tauchen in verschiedensten Zeitschriften kurze Artikel über eine möglichst schnelle Ausgabe von Kreisen auf. Besitzt man jedoch einen Grafikbildschirm, bei dem sich die tatsächlichen Entfernungen zweier benachbarter Bildpunkte in X- und Y-Richtung voneinander unterscheiden, was ja eigentlich der Standardfall ist, erscheint ein Kreis leider als Ellipse. Will man Kreise darstellen, müssen also Ellipsen gezeichnet werden. Der nachfolgende Artikel beschreibt einen Algorithmus zur schnellen Ausgabe einer Ellipse am Bildschirm. Bei entsprechender Wahl der Halbachsenlängen kann man auch bei unterschiedlicher XY-Auflösung optisch perfekte Kreise zeichnen.

Gerhard Piran

Als Grundlage für die Entwicklung eines schnell arbeitenden Ellipsenprogrammes dient das von J. E. Bresenham (siehe Literaturverzeichnis) entwickelte Verfahren zur Darstellung von Kreisen auf Rasterbildschirmen, welches sicherlich eines der schnellsten ist, da es nur mit Integerarithmetik arbeitet.

Um einen kompletten Kreis zeichnen zu können, braucht lediglich ein Achtelkreis berechnet zu werden. Alle weiteren Punkte werden durch Spiegelung ermittelt (siehe Abb. 1). Das Programm kann in M+K 84-1, p. 79 nachgeschlagen werden.

Ellipsenalgorithmus

Bei der Darstellung einer kompletten Ellipse müssen jedoch alle Punkte innerhalb eines Quadranten berechnet werden. Die fehlenden drei Punkte werden wiederum durch Spiegelung in die anderen Quadranten gewonnen (siehe Abb. 2). Als Ausgangspunkt dient der Punkt P(0,b). Alle weiteren Punkte werden im Uhrzeigersinn angefügt.

Wie beim Kreisalgorithmus gibt es auch beim Setzen der Ellipsenpunkte nur zwei Möglichkeiten. Zuvor muss jedoch unterschieden werden, in welchem Teilabschnitt man sich befindet. Im Teilabschnitt 1 (Ta1) gibt es nur flache Ellipsenteilstücke. Die Tangenten an die Ellipsenkurve liegen unter einem Winkel an, der grösser oder gleich -45 Grad ist (0 >= k >= -1). Ausgehend von einem bereits bekannten Punkt kann der nächste Punkt nur rechts oder rechts unterhalb folgen (aus Fall A-D folgt Punkt S oder T, siehe Abb. 3). Im Teilabschnitt 2 (Ta2) ist die Tangentenneigung kleiner als -45 Grad (-1 > k > -∞). Es darf nur mehr zwischen dem rechts unteren oder dem unteren Punkt gewählt werden (aus Fall E-G folgt Punkt T oder U).

Welcher der beiden möglichen Punkte nun tatsächlich gewählt werden muss, ist im Prinzip recht einfach zu erklären. Es ist jener Punkt, welcher der reellen Ellipsenkurve am nächsten ist. Die mathematische Formulierung dieser Aussage ist jedoch alles andere als einfach und ein Kernstück für die schnelle Ausgabe von Ellipsen. Tests haben ergeben, dass für jeden zu setzenden Punkt im Mittel 3,2 Abfragen, 3,1 Strichrechnungen und 2,5 Punktrechnungen erforderlich sind. Das ergibt im Gegensatz zur normalen Ellipsendarstellung mit Sinus-Cosinusberechnungen eine sehr grosse Laufzeitersparnis.

Nimmt man den Ellipsenmittelpunkt M im Koordinatenursprung (0,0) an, berechnen sich die Entfernungen der Punkte zur Ellipsenkurve mit P(x,y) als Vorgabe (siehe Abb. 3) wie folgt - hervorgehobene Felder sind Teile des Unterprogrammes DRAW_ ELLIPSE:

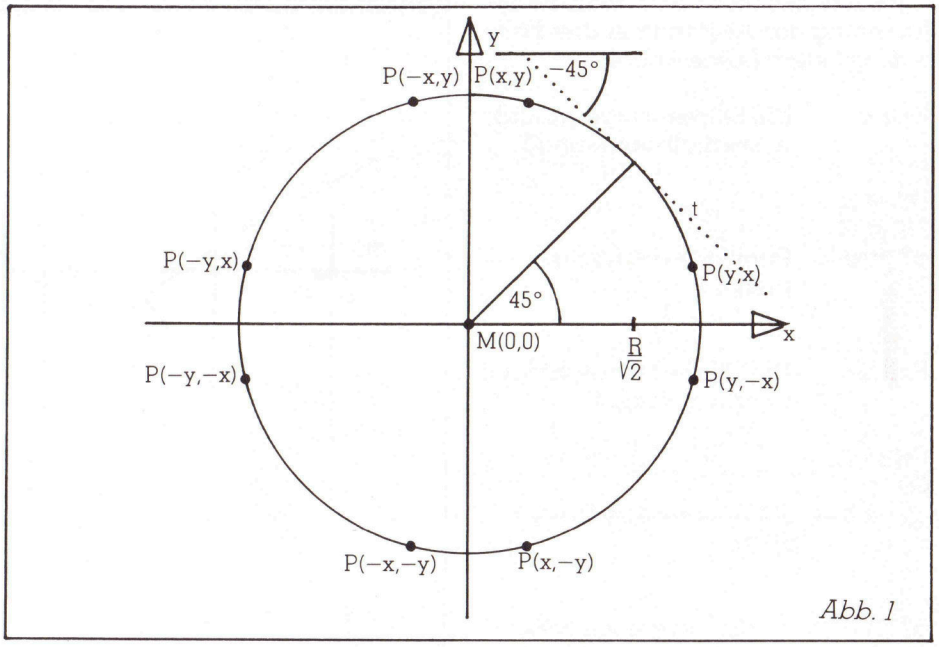


Abb. 1

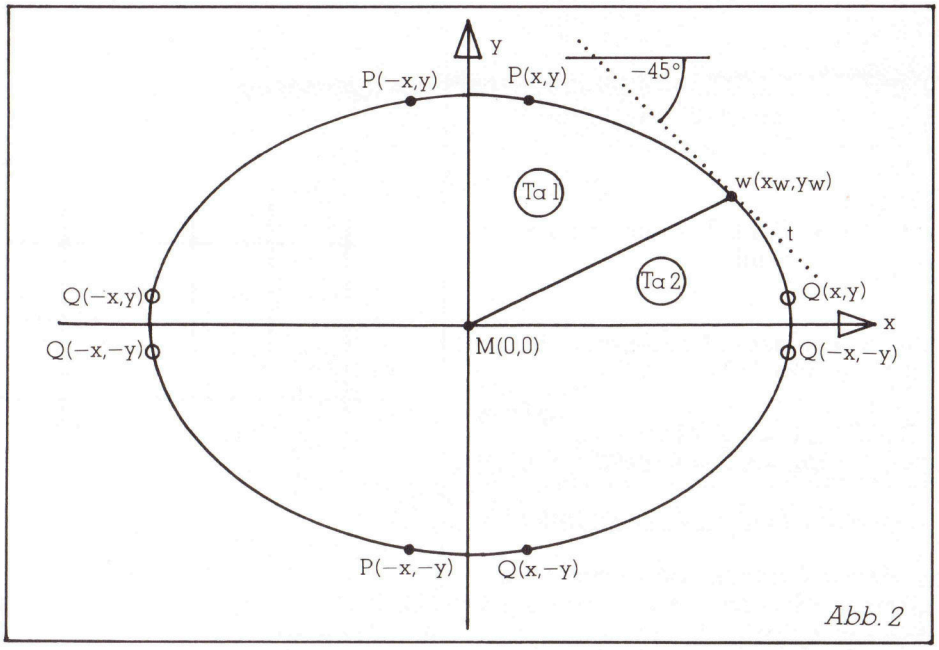


Abb. 2

$DS = zerr*(x+1)^2+y^2-b^2$

$DT = zerr*(x+1)^2+(y-1)^2-b^2$

$DU = zerr*x^2+(y-1)^2-b^2$

$zerr = b^2/a^2$

Berechnung Zerrfaktor

Wahl des nächsten Punktes

Für die weiteren Betrachtungen bilden wir die Summe der Abstände 2-er Punkte.

$d1 = DS + DT$

$d2 = DT + DU$

Im Teilabschnitt 1 kann man die Betrachtung der Abstände in drei Fälle untergliedern (siehe Abb. 4):

Fall a: Die Ellipsenkurve verläuft ausserhalb von S und T.

$DS < 0$
$DT < 0$
$d1 < 0 \rightarrow$ Punkt S ist der nächste Punkt.

Fall b: Die Ellipsenkurve verläuft zwischen S und T.

$DS > 0$
$DT < 0$
$d1 < 0 \rightarrow$ Punkt S ist der nächste Punkt.

$d1 >= 0 \rightarrow$ Punkt T ist der nächste Punkt.

Fall c: Die Ellipsenkurve verläuft innerhalb von S und T.

$DS > 0$
$DT > 0$
$d1 > 0 \rightarrow$ Punkt T ist der nächste Punkt.

Zusammenfassend ergeben sich also nur zwei Fälle:

$d1 < 0 \rightarrow$ Punkt S
$d1 >= 0 \rightarrow$ Punkt T

Punktentscheidung Teilabschnitt 1

Wendet man im Teilabschnitt 2 die selben Betrachtungen an, ergeben sich folgende zwei Fälle:

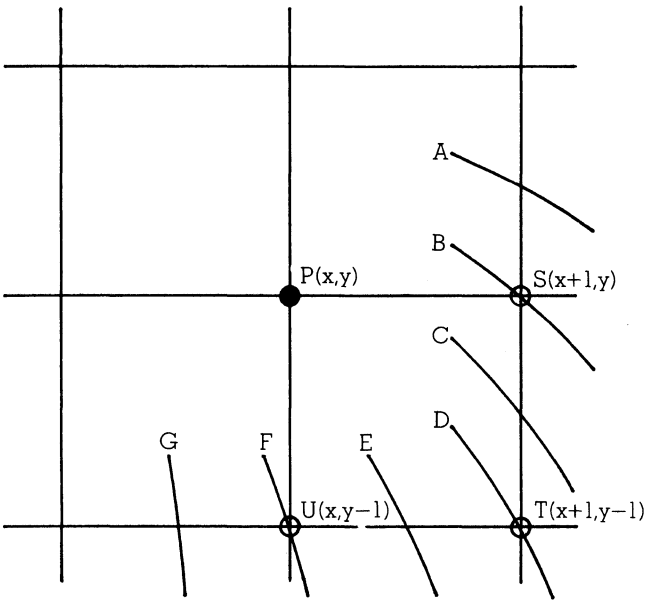


Abb. 3

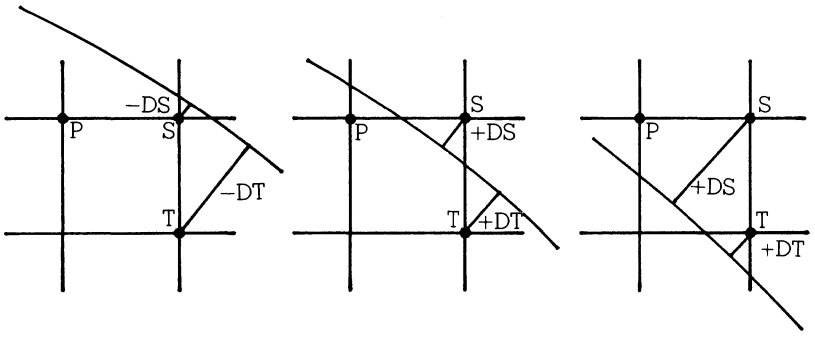


Abb. 4 a) b) c)

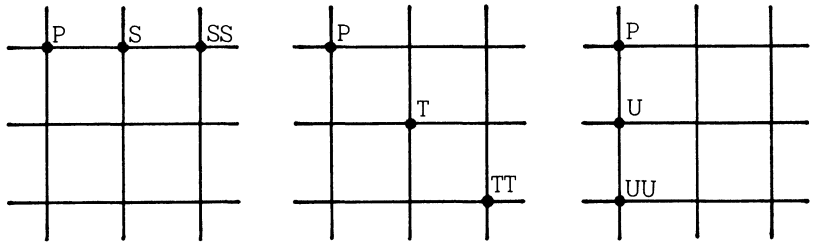


Abb. 5

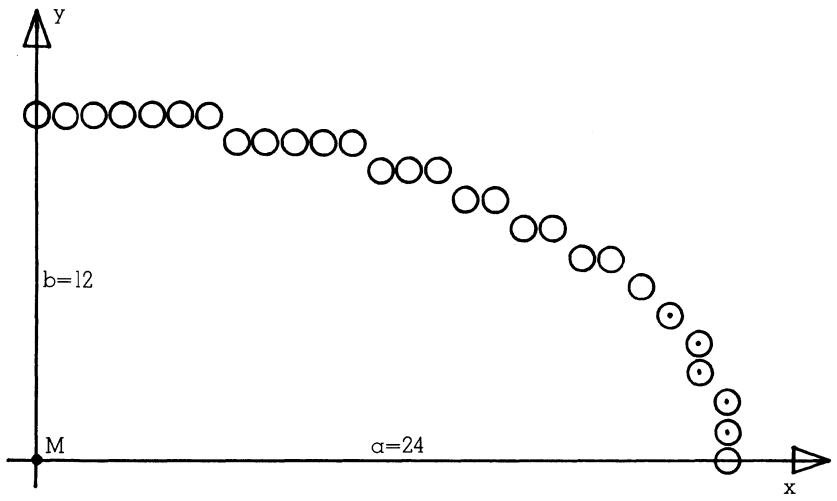
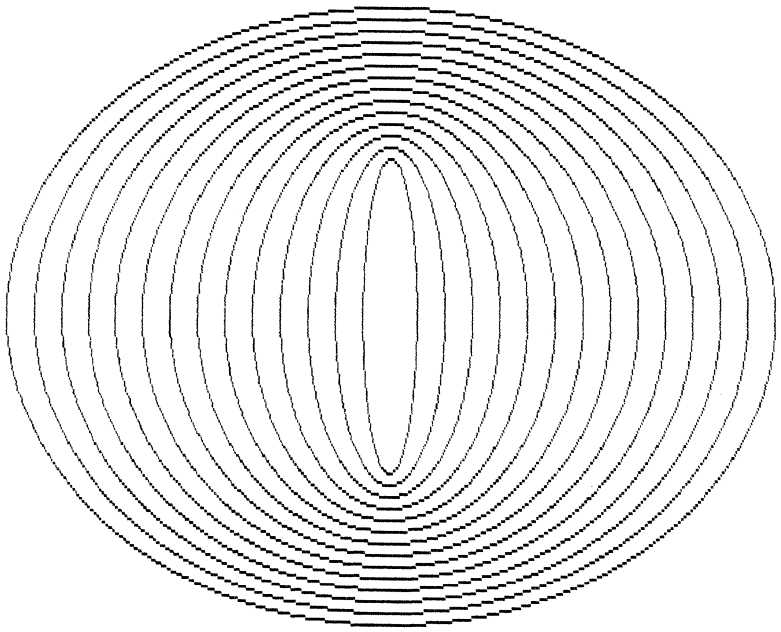
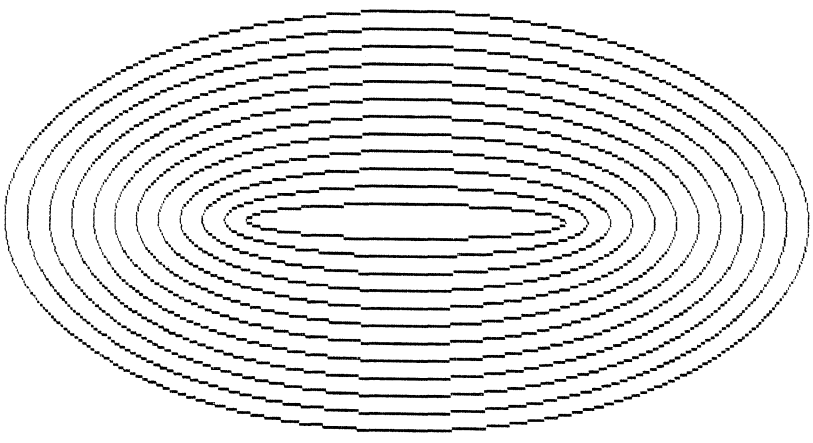


Abb. 7



Ellipsentest (0–7) : 2



Ellipsentest (0–7) : 3

d2 < 0 → Punkt T d2 >= 0 → Punkt U

Punktentscheidung Teilabschnitt

Setzen der Punkte

Im Teilabschnitt 1 wird eine Schleife durchlaufen, in der der X-Wert solange um 1 erhöht wird, bis der X-Wert des Wendepunktes W(x) erreicht ist (siehe Abb. 2). Um die Schleife mathematisch beschreiben zu können, muss der Wendepunkt berechnet werden. Er ist jener Ellipsenpunkt, dessen Tangentenneigung -45 Grad beträgt (k = -1). Ermittelt wird er durch die 1. Ableitung der Ellipsengleichung:

Ellipsengleichung:

$x^2/a^2 + y^2/b^2 = 1$

1. Ableitung mit Tangentensteigung = -1:

$y = dy/dx = -1 \rightarrow -xw/yw*zerr = -1$

woraus folgt:

$yw = zerr * xw$

Da in der X-Scheife nur Tangentenneigungen von 0 bis -45 Grad auftreten können, wird aus dieser Glei-

```
W (     21.47 /      5.37 )

1:   X=  0   Y= 12      d=     -22.50
1:   X=  1   Y= 12      d=     -21.00
1:   X=  2   Y= 12      d=     -18.50
1:   X=  3   Y= 12      d=     -15.00
1:   X=  4   Y= 12      d=     -10.50
1:   X=  5   Y= 12      d=      -5.00
1:   X=  6   Y= 12      d=       1.50
1:   X=  7   Y= 11      d=     -35.00
1:   X=  8   Y= 11      d=     -26.50
1:   X=  9   Y= 11      d=     -17.00
1:   X= 10   Y= 11      d=      -6.50
1:   X= 11   Y= 11      d=       5.00
1:   X= 12   Y= 10      d=     -22.50
1:   X= 13   Y= 10      d=      -9.00
1:   X= 14   Y= 10      d=       5.50
1:   X= 15   Y=  9      d=     -15.00
1:   X= 16   Y=  9      d=       1.50
1:   X= 17   Y=  8      d=     -13.00
1:   X= 18   Y=  8      d=       5.50
1:   X= 19   Y=  7      d=      -3.00
1:   X= 20   Y=  7      d=      17.50
1:   X= 21   Y=  6      d=      15.00
2:   X= 22   Y=  5      d=      -2.75
2:   X= 23   Y=  4      d=       4.75
2:   X= 23   Y=  3      d=      -5.25
2:   X= 24   Y=  2      d=      11.25
2:   X= 24   Y=  1      d=       9.25
3:   X= 24   Y=  0      d=      11.25
```

Ellipsentest 1: *a= 24* *b= 12*

chung direkt die Schleifenbedingung abgeleitet:

```
WHILE y >= (zerr * x) DO ...
```

Ausgehend von dem Punkt, an dem die X-Schleife abgebrochen wurde (in der Nähe des Wendepunktes), wird nun im Teilabschnitt 2 eine weitere Schleife durchlaufen, in der der Y-Wert solange um 1 erniedrigt wird, bis er die X-Achse erreicht hat.

```
WHILE y > 0 DO ...
```

Wer die Koordinaten des Wendepunktes berechnen möchte, braucht nur in die folgenden Formeln einsetzen:

$xw = a^2/sqrt(a^2+b^2)$

$yw = b^2/sqrt(a^2+b^2)$

Optimierung

Setzt man DS, DT und DU in die Gleichung von d1 und d2 ein, ergeben sich folgende Formeln:

$d1_{(i)} = DS + DT$
$= 2*zerr*(x^2+2*x+1)+2*y^2-2*y+1-2*b^2$

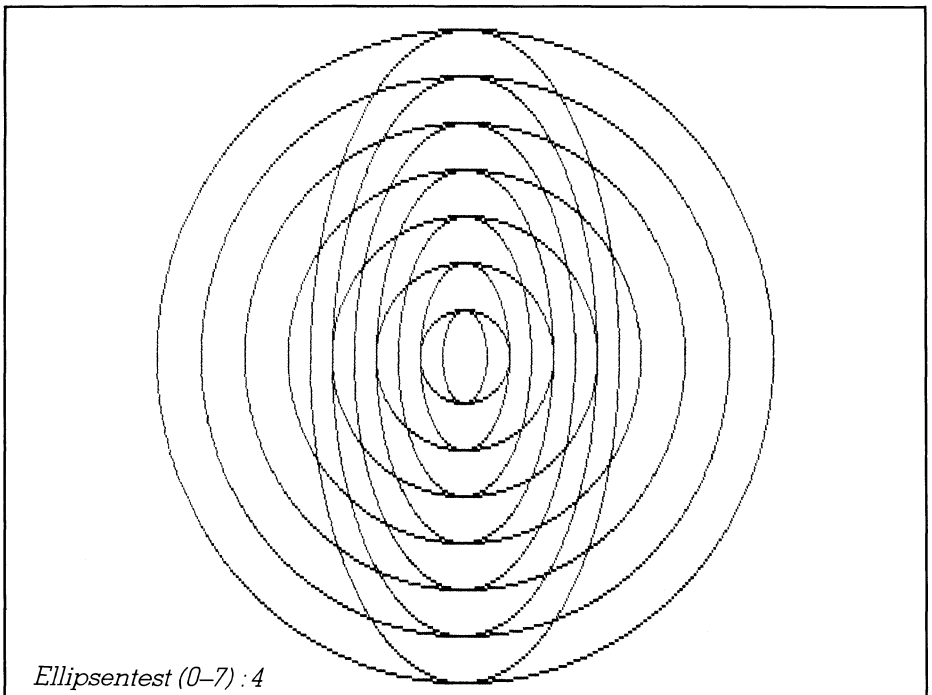
$d2_{(i)} = DT + DU$
$= zerr*(2*x^2+2x+1)+2*y^2-4*y+2-2*b^2$

Diese Formeln für die Distanzberechnungen sind jedoch immer noch zu kompliziert für einen effizienten Rechnereinsatz. Die Idee, den Distanzwert nicht bei jedem Punkt sondern nur für den 1. Punkt P(0,b) zu berechnen und dann nur mehr zu korrigieren, wird sich in der Folge als recht gut erweisen.

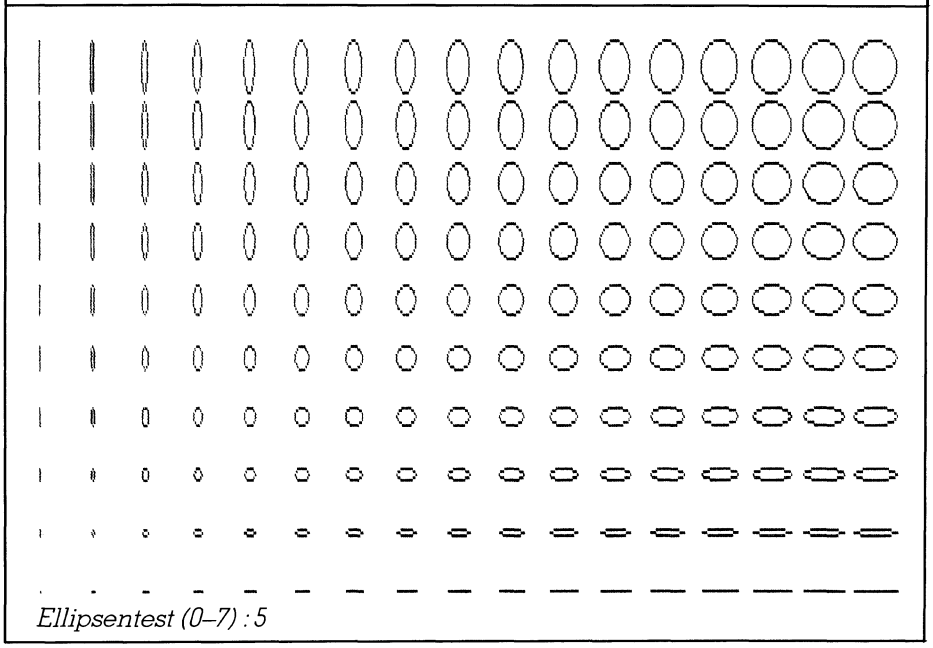
$ddl = d1_{(i+1)} - d1_{(i)}$
$dd2 = d2_{(i+1)} - d2_{(i)}$

dd?: Differenz der Distanzwerte

Nach Abb. 5 können die Distanzwerte für SS, TT und UU berechnet werden.
$d1_{(i+1)} = DSS + DTT$
$d2_{(i+1)} = DTT + DUU$



Ellipsentest (0–7) : 4



Ellipsentest (0–7) : 5

Literatur

N.I. Badler, Disk generators for a Raster Display Device Computer Grafics Image Processing, 12/77 S:589-593

J.E. Bresenham, A Linear Algorithm for Incremental Digital Display of Circular Arcs Communications of the ACM, 2/77 S:100-106

D.E. Field, Algorithms for Drawing Simple Geometric Objects on Raster Devices TR 314, Prinston University, 6/83

J.D. Foley / Adries Van Dam, Fundamentals of Interactiv Computer Grafics Addison-Welsley Publishing Company, 1982 S:441-446

B.K.P. Horn, Circle Generator for Display Divices Computer Grafics Image Processing, 1976 S:280-288

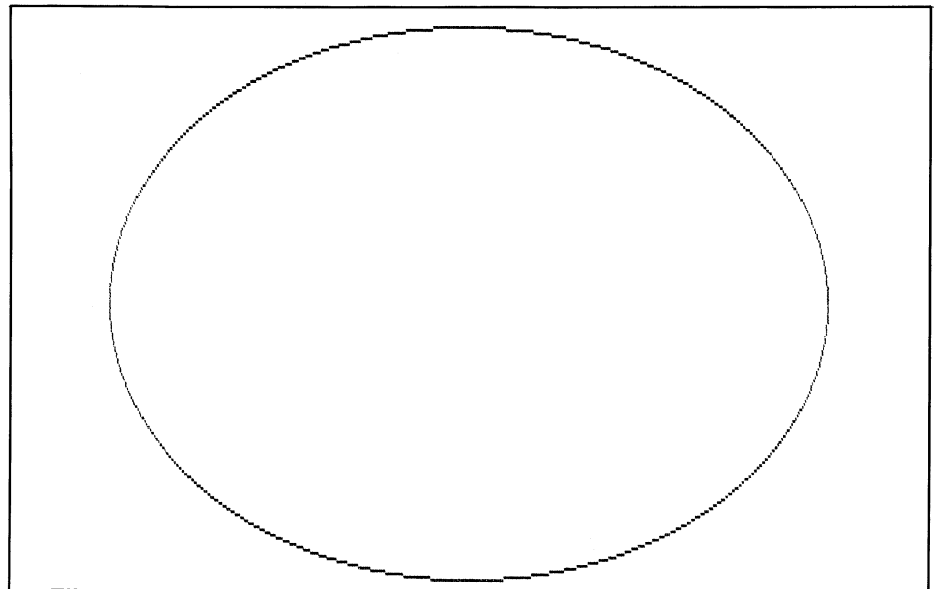
M.L.V. Pitteway, Algorithm for Drawing Ellipses or Hyperbolae with a Digital Plotter Computer J. 11/67 S:282-289

Bruno L. Stanek, Schnelle Kreise mit Pascal, M+K 83-6

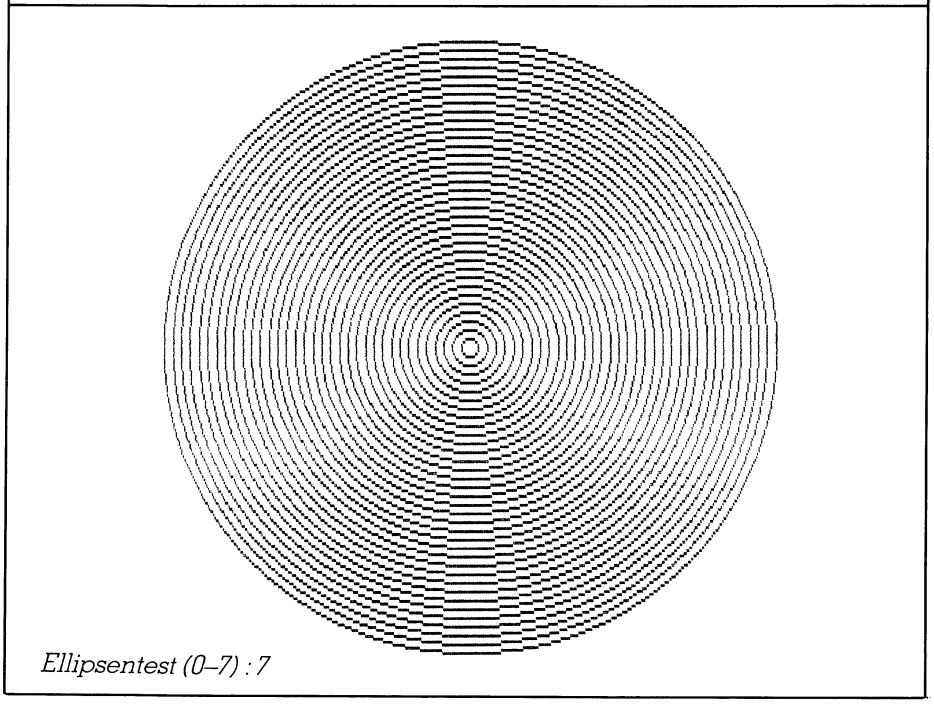
Christoph Braendle, Schnelle Kreise - ein Echo, M+K 84-1

H. Völz, Basicprogramm für schnelle Kreise und Ellipsen, M+K 84-5

Armin Bohg, Schnelle Kreise, C't 5/85



Ellipsentest (Ende = 0, Test = 1–7) : 6 a = 123,4 b = 95,2



Ellipsentest (0–7) : 7

Nach Einsetzen in die Differenzformel ergeben sich einfachere Formeln:

gewählter Punkt	Ellipsen-Formeln
1.	$d_{(1)} = 2*(zerr-b)+1$
S	$d_{(i+1)} = d_{(i)}+zerr*(4*x+6)$
T	$d_{(i+1)} = d_{(i)}+4*(zerr*x-y)+6+zerr*4$
U	$d_{(i+1)} = d_{(i)}-4*y+6$

Aus diesen Ellipsenformeln lassen sich übrigens sehr rasch die Kreisformeln von J. E. Bresenham ableiten. Setzt man a = b ergibt sich für den Zerrfaktor = 1 und weiters:

gewählter Punkt	Kreis-Formeln
1.	$d_{(1)} = 3-2*r$
S	$d_{(i+1)} = d_{(i)}+4*x+6$
T	$d_{(i+1)} = d_{(i)}+4*(x-y)+10$
U	$d_{(i+1)} = d_{(i)}-4*y+6$

Kreisdarstellung

Will man einen Kreis am Bildschirm darstellen, muss bei unterschiedlicher Bildschirmauflösung eine Ellipse gezeichnet werden, bei der sich die Hauptachse zur Nebenachse gleich verhält, wie die Entfernungen zweier Punkte in X-Richtung zu der in Y-Richtung. Nimmt man die Auflösung in Y-Richtung als Grundmass, so muss zur massstabsgetreuen Darstellung der Wert in X-Richtung korrigiert werden. Für die Darstellung eines Kreises mit dem Radius r ergeben sich daraus folgende Halbachsenlängen für die Ellipsenparametrierung:

a = r*xy_fak

b = r

Die Ermittlung der Verhältniszahl (xy_fak) ist recht einfach. Man leitet sie aus den Verhältnissen der physikalischen Abmessungen (phys) des Bildschirmes zu der Auflösung des Bildschirmes in Bildpunkten (pixels) ab (Bild 6).

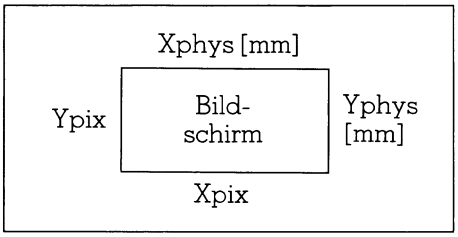


Bild 6

Xpix/XY_fak*Ypix = Xphys/Yphys

daraus ergibt sich folgende Formel:

$$XY\_fak = \frac{Yphys}{Xphys} * \frac{Xpix}{Ypix}$$

Berechnung Bildschirmfaktor

Da der Bildschirmfaktor für einen Bildschirm immer der gleiche ist, ist es sinnvoll, ihn einmal zu berechnen und als Konstante zu verwenden. In der Prozedur DRAW_REAL_CIRCLE zum Zeichnen von massstabsgetreuen Kreisen wird die Umrechnung des Kreisradius in die entsprechenden Achsenlängen der zu zeichnenden Ellipse und die Ausgabe der Ellipse durchgeführt.

Nächsten Monat gibt's wieder
COMPUTER MARKT
mit aktuellen Informationen.

```
+-------------------------------------[procedure DRAW_ELLIPSE]-+
| Versorgung:  mx,my  Koordinaten Mittelpunkt                  |
|              a,b    Halbachsenlaengen                        |
|              color  Zeichenfarbe                            |
+==============================================================+
|  Ausgangspunkt P(0,b) initialisieren                         |
+--------------------------------------------------------------+
|  Zerrfaktor zerr berechnen                                   |
+--------------------------------------------------------------+
|  Distanzwert d initialisieren                                |
+--------------------------------------------------------------+
|  solange Wendepunkt nicht erreicht                 [repeat]  |
|  +-----------------------------------------------------------+
|  | 4 Punkte in angegebener Farbe setzen                      |
|  +-----------------------------------------------------------+
|  | Distanzwert < 0 ?                                  [if]   |
|  +-------------------------------------------------[then]--+
|  | Auswahl von Punkt S mit Distanzwertkorrektur:             |
|  | d = d + zerr * (4 * x + 6)                                |
|  +---+---------------------------------------------[else]--+
|  |   | Auswahl von Punkt T mit Distanzwertkorrektur:         |
|  |   | d = d + 4 * (zerr * x - y) + 6 * zerr + 4             |
|  |   +-------------------------------------------------------+
|  |   | Y-Wert um 1 erniedrigen                               |
|  +---+-------------------------------------------------------+
|  | X-Wert um 1 erhoehen                                      |
+---+-----------------------------------------------------------+
|  Distanzwert wegen neuer Richtung korrigieren                |
|  d = d - zerr * (2 * x + 1) - 2 * y + 1                      |
+--------------------------------------------------------------+
|  solange X-Achse nicht erreicht                    [repeat]  |
|  +-----------------------------------------------------------+
|  | 4 Punkte in angegebener Farbe setzen                      |
|  +-----------------------------------------------------------+
|  | Distanzwert > 0 ?                                  [if]   |
|  +-------------------------------------------------[then]--+
|  | Auswahl von Punkt U mit Distanzwertkorrektur:             |
|  | d = d - 4 * y + 6                                         |
|  +---+---------------------------------------------[else]--+
|  |   | Auswahl von Punkt T mit Distanzwertkorrektur:         |
|  |   | d = d + 4 * (zerr * x - y) + 6 * zerr + 4             |
|  |   +-------------------------------------------------------+
|  |   | X-Wert um 1 erhoehen                                  |
|  +---+-------------------------------------------------------+
|  | Y-Wert um 1 erniedrigen                                   |
+---+-----------------------------------------------------------+
|  Abschlusspunkte setzen                                      |
+--------------------------------------------------------------+
```

Struktogramm des Unterprogrammes DRAW_ELLIPSE

Ellipsendarstellung

Um Ellipsen mit richtigen Achsenverhältnissen darstellen zu können, wurde die Prozedur DRAW_REAL_ELLIPSE geschaffen. Auch hier wird durch die Korrektur der Hauptachsenlänge eine massstabsgetreue Ausgabe der Ellipse erzielt.

Programmanpassung

Die Konstanten XPIXMAX und YPIXMAX sind der entsprechenden Auflösung des verwendeten Bildschirmes anzupassen.

Die Prozedur CLRSCREEN ist durch die Standardprozedur von Turbo-Pascal CLRSCR zu ersetzen, falls keine spezielle Initialisierung der grafischen Ausgabe erforderlich ist, wie zum Beispiel das Aktivieren des Video-RAM-Zugriffs. Die Anweisung {$I CLRSCR.INC} entfällt dabei.

Die Prozedur SETPIX setzt einen Punkt in angegebener Farbe auf dem Bildschirm. Für den IBM-PC zum Beispiel muss sie durch den Befehl PLOT ersetzt werden. Die Anweisung {$I TPIXEL.INC} entfällt dabei.

Der Nullpunkt des Koordinatensystems liegt in der linken unteren Ecke des Bildschirmes.

Tests

Hat man alles an seinen Rechner angepasst, steht den verschiedenen Tests nichts mehr im Wege. Mit Test 1 erhält man ein Datenprotokoll aller Punkte. Bild 7 zeigt den berechneten Ellipsenabschnitt als vergrösserten Ausschnitt. Die punktierten Kreise sind Punkte, die in der Y-Schleife berechnet wurden. Test 2-4 dient zur visuellen Kontrolle von flachen bzw. schmalen Ellipsen. Im Test 5 werden Sonderfälle wie z.B. $a = 0$ oder $b = 0$ getestet. Für die Prüfung von anderen Ellipsenalgorithmen ist dieser Test sehr aussagekräftig und schwer zu bestehen. Test 6 zeichnet eine massstabsgetreue Ellipse, Test 7 massstabsgetreue Kreise. Sind alle Tests bestanden, steht einer Implementierung der Ellipsenroutine in ein Zeichenprogramm oder in ein CAD-Programm nichts mehr im Wege. □

COMPUTER SPLITTER

Erfolgreiche Zukunft für 3D-Systeme

(149/ih) In einer Vielzahl amerikanischer Unternehmen haben CAD/CAM-Systeme den Zeichnungstisch endgültig verdrängt. Während CAD/CAM bis anhin vor allem für zeichnerische Anwendungen eingesetzt wurde, wird CAD/CAM in nächster Zeit, wie einer neuen Studie von Frost Sullivan zu entnehmen ist, vermehrt für analytische Zwecke Verwendung finden. Bis Anfang der 90er Jahre wird der CAD/CAM-Markt ein Volumen von 8 Mia. Dollar aufweisen. Besonders im Bereiche der dreidimensionalen Gestaltung und anderer anspruchsvoller Anwendungen wird ein beträchtliches Wachstum erwartet. Als Hauptbenutzer von CAD/CAM gelten die Hersteller von integrierten Schaltungen, aber auch bei der Bauindustrie wird ein Wechsel zu CAD/CAM als Konstruktionswerkzeug erwartet. Die Studie vermittelt einen Ueberblick über die verschiedenen Anwendungen, verfügbare Sofware, CAD/CAM-Benutzergruppen, spezifische Produkte, Lieferanten und ihre Marktanteile. Mechanische Anwendungen dominieren (mit einem Anteil von 40%). Unter den Lieferanten führt IBM vor Intergraph und Computervision. Info: Frost Sullivan Inc., 106 Fulton Street, New York, NY 10038, USA.□

```
 1:  0
 2:  0 {*************************************************************************}
 3:  0 {*                                                                       *}
 4:  0 {*  Include-File:   E L L I P S E . I N C                                *}
 5:  0 {*                                                                       *}
 6:  0 {*  Sprache:        T u r b o - P a s c a l     V 2.0                    *}
 7:  0 {*                                                                       *}
 8:  0 {*  Author:         Gerhard Piran            09.06.86                    *}
 9:  0 {*                                                                       *}
10:  0 {*-----------------------------------------------------------------------*}
11:  0 {*                                                                       *}
12:  0 {*  Aufruf:         DRAW_ELLIPSE (mx,my, a,b, color);                    *}
13:  0 {*                                                                       *}
14:  0 {*  Versorgung:     mx,my: integer;   Koordinaten Mittelpunkt            *}
15:  0 {*                                                                       *}
16:  0 {*                  a,b: integer;     Halbachsenlaengen >= 0             *}
17:  0 {*                                                                       *}
18:  0 {*                  color: integer;   Farbe, mit der Ellipse             *}
19:  0 {*                                    dargestellt werden soll            *}
20:  0 {*                                                                       *}
21:  0 {*************************************************************************}
22:  0 PROCEDURE DRAW_ELLIPSE (mx,my, a,b, color: integer);
23:  0
24:  0 VAR   x,y  : integer;                    {Punktzeiger}
25:  0       zerr : real;                       {Zerrfaktor = b^2 / a^2 }
26:  0       d    : real;                       {Distanz Punkt <--> opt. Ellipsenlinie}
27:  0
28:  0 BEGIN
29:  1   x := 0;                                {Ausgangspunkt initialisieren}
30:  1   y := b;
31:  1   IF a = 0                               {Hauptachse = 0 ?}
32:  1   THEN BEGIN
33:  2     zerr := 0.0;                        {Zerrfaktor unwichtig}
34:  2     d := 1.0E37;                        {immer Punkt U waehlen !}
35:  1   END
36:  1   ELSE BEGIN                            {Hauptachse gueltig}
37:  2     zerr := sqr (b / a);                {Zerrfaktor berechnen}
38:  2     IF b = 0                            {Nebenachse = 0 ?}
39:  2     THEN d := -1.0                      {immer Punkt S waehlen !}
40:  2     ELSE d := 2 * (zerr - b) + 1;       {Distanzwert initialisieren}
41:  2
42:  2     {--- X-Schleife ---}
43:  2     WHILE (y >= zerr * x) and (x < a)   {solange Tangentensteigung >= -1 }
44:  2     DO BEGIN                            { und X-Wert < halbe Hauptachse}
45:  3       IF prot THEN writeln (lst,'1:  X=',x:3,'  Y=',y:3,'     d=',d:10:2); {Test}
46:  3
47:  3       SETPIX (mx + x, my + y, color);   {aktuellen Punkt und }
48:  3       SETPIX (mx + x, my - y, color);   { 3 Spiegelpunkte ausgeben}
49:  3       SETPIX (mx - x, my - y, color);
50:  3       SETPIX (mx - x, my + y, color);
51:  3
52:  3       IF d < 0
53:  3       THEN d := d + zerr * (4 * x + 6) {Auswahl von S}
54:  3       ELSE BEGIN                       {Auswahl von T}
55:  4         d := d + 4 * (zerr * x - y) + 6 * zerr + 4;
56:  4         y := y - 1;
57:  3       END;
58:  3       x := x + 1;
59:  2     END;                               {Distanzwert korrigieren}
60:  2     d := d - zerr * (2 * x + 1) - 2 * y + 1;
61:  1   END;
```

```
 62:  1
 63:  1   {--- Y-schleife ---}
 64:  1   WHILE (y > 0) and (x <= a)          {solange X-Achse nicht erreicht}
 65:  1   DO BEGIN
 66:  2     IF prot THEN writeln (lst,'2:  X=',x:3,'  Y=',y:3,'     d=',d:10:2); {Test}
 67:  2
 68:  2     SETPIX (mx + x, my + y, color);    {aktuellen Punkt und }
 69:  2     SETPIX (mx + x, my - y, color);    { 3 Spiegelpunkte ausgeben}
 70:  2     SETPIX (mx - x, my - y, color);
 71:  2     SETPIX (mx - x, my + y, color);
 72:  2
 73:  2     IF d > 0
 74:  2     THEN d := d - 4 * y + 6            {Auswahl von U}
 75:  2     ELSE BEGIN                         {Auswahl von T}
 76:  3       d := d + 4 * (zerr * x - y) + 6 * zerr + 4;
 77:  3       x := x + 1;
 78:  2     END;
 79:  2     y := y - 1;
 80:  1   END;
 81:  1
 82:  1   REPEAT
 83:  2     IF prot THEN writeln (lst,'3:  X=',x:3,'  Y=',y:3,'     d=',d:10:2); {Test}
 84:  2
 85:  2     SETPIX (mx + x, my, color);        {Pr ( a,0)}
 86:  2     SETPIX (mx - x, my, color);        {Pl (-a,0)}
 87:  2     x := x + 1;
 88:  1   UNTIL x > a;
 89:  0 END;
 90:  0
 91:  0 {*************************************************************************}
 92:  0 {*                                                                       *}
 93:  0 {*  Aufruf:          DRAW_REAL_ELLIPSE (mx,my, a,b, color);              *}
 94:  0 {*                                                                       *}
 95:  0 {*  Versorgung:      mx,my: integer;   Koordinaten Mittelpunkt           *}
 96:  0 {*                                                                       *}
 97:  0 {*                   a,b: real;        Halbachsenlaengen >= 0.0          *}
 98:  0 {*                                                                       *}
 99:  0 {*                   color: integer;   Farbe, mit der Ellipse            *}
100:  0 {*                                     dargestellt werden soll           *}
101:  0 {*                                                                       *}
102:  0 {*************************************************************************}
103:  0 PROCEDURE DRAW_REAL_ELLIPSE (mx,my: integer; a,b:real; color: integer);
104:  0
105:  0 CONST xy_fak = 2.12;                   {Verhaeltniszahl Bildschirm}
106:  0
107:  0 BEGIN
108:  1   DRAW_ELLIPSE (mx,my, round (a * xy_fak), round (b), color);
109:  0 END;
110:  0
111:  0 {*************************************************************************}
112:  0 {*                                                                       *}
113:  0 {*  Aufruf:          DRAW_REAL_CIRCLE (mx,my, r, color);                 *}
114:  0 {*                                                                       *}
115:  0 {*  Versorgung:      mx,my: integer;   Koordinaten Mittelpunkt           *}
116:  0 {*                                                                       *}
117:  0 {*                   r: real;          Kreisradius >= 0.0                *}
118:  0 {*                                                                       *}
119:  0 {*                   color: integer;   Farbe, mit der Kreis              *}
120:  0 {*                                     dargestellt werden soll           *}
121:  0 {*                                                                       *}
122:  0 {*************************************************************************}
```

```
123:  0 PROCEDURE DRAW_REAL_CIRCLE (mx,my: integer; r:real; color: integer);
124:  0
125:  0 CONST xy_fak = 2.12;                    {Verhaeltniszahl Bildschirm}
126:  0
127:  0 BEGIN
128:  1   DRAW_ELLIPSE (mx,my, round (r * xy_fak), round (r), color);
129:  0 END;
```

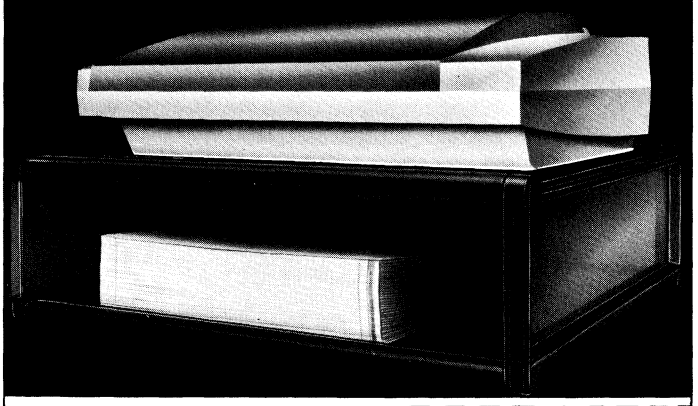
```
  1:  0
  2:  0 {*************************************************************************}
  3:  0 {*                                                                       *}
  4:  0 {*  ELLIPSENTEST:   Testrahmen fuer die Prozedur DRAW_ELLIPSE            *}
  5:  0 {*                                                                       *}
  6:  0 {*                  Sprache:  T U R B O - P A S C A L   V 2.0            *}
  7:  0 {*                                                                       *}
  8:  0 {*                  Author:   Gerhard Piran          09.06.86            *}
  9:  0 {*                                                                       *}
 10:  0 {*************************************************************************}
 11:  0 PROGRAM test_ellipse;
 12:  0
 13:  0 CONST Xpixmax = 640;  Ypixmax = 225;   {Aufloesung Bildschirm}
 14:  0
 15:  0 VAR a,b, test, I1,I2 : integer;
 16:  0     ar,br            : real;
 17:  0     C1               : char;
 18:  0     prot             : boolean;
 19:  0
 20:  0 {$I clrscr.inc}                        {Bildschirm loeschen}
 21:  0 {$I tpixel.inc}                        {Punkt setzen}
 22:  0 {$I ellipse.inc}                       {Ellipse zeichnen}
 23:  0
 24:  0 BEGIN
 25:  1   REPEAT
 26:  2     CLRSCREEN;
 27:  2     prot := false;
 28:  2     write ('Ellipsentest (Ende = 0,  Tests = 1-7): ');
 29:  2     read (kbd,C1);
 30:  2     write (C1);
 31:  2     test := ord (C1) - 48;
 32:  2     CASE test of
 33:  3       1: BEGIN
 34:  4         prot := true;
 35:  4         gotoxy (1,2);
 36:  4         write   ('a = ');    readln (a);
 37:  4         write   ('b = ');    readln (b);
 38:  4         writeln (lst);
 39:  4         writeln (lst,'Ellipsentest 1:    a=',a:3,'   b=',b:3);
 40:  4         writeln (lst);
 41:  4         write   (lst,'W (',a*a/sqrt(a*a+b*b):9:2,' /');
 42:  4         writeln (lst,b*b/sqrt(a*a+b*b):9:2,' )');
 43:  4         writeln (lst);
 44:  4         DRAW_ELLIPSE (Xpixmax div 2, Ypixmax div 2, a,b, 7);
 45:  3       END;
 46:  3
 47:  3       2: FOR I1 := 1 TO 14 DO
 48:  3       DRAW_ELLIPSE (Xpixmax div 2, Ypixmax div 2, 20 * I1, 50 + 4 * I1, 7);
```

```
49:  3
50:  3        3: FOR I1 := 1 TO 12 DO
51:  3        DRAW_ELLIPSE (Xpixmax div 2, Ypixmax div 2, 100 + 16 * I1, 6 * I1, 7);
52:  3
53:  3        4: FOR I1 := 1 TO 7
54:  3        DO BEGIN
55:  4          DRAW_ELLIPSE (Xpixmax div 2, Ypixmax div 2, 16 * I1, 16 * I1, 7);
56:  4          DRAW_ELLIPSE (Xpixmax div 2, Ypixmax div 2, 32 * I1, 16 * I1, 7);
57:  3        END;
58:  3
59:  3        5: FOR I1 := 0 TO 16
60:  3        DO FOR I2 := 0 TO 9
61:  3        DO DRAW_ELLIPSE (10 + I1 * 38, 10 + I2 * 20, I1, I2, 7);
62:  3
63:  3        6: BEGIN
64:  4          gotoxy (1,2);
65:  4          write   ('a = ');    readln (ar);
66:  4          write   ('b = ');    readln (br);
67:  4          DRAW_REAL_ELLIPSE (Xpixmax div 2, Ypixmax div 2, ar,br, 7);
68:  3        END;
69:  3
70:  3        7: BEGIN
71:  4          FOR I1 := 1 TO 35
72:  4          DO DRAW_REAL_CIRCLE (Xpixmax div 2, Ypixmax div 2, I1 * 3, 7);
73:  3        END;
74:  2      END;
75:  2      IF (test > 0) and (test <= 7) THEN read (kbd,C1);
76:  1    UNTIL test = 0;
77:  0  END.
```


... Ordnung aktuell mit

multi form Druckerplattform ...

... der richtige Platz für Ihren Bürodrucker
stabil – funktionsgerecht – preiswert



JA, die **multiform** Druckerplattform interessiert mich, bitte um ausführliche Dokumentation!

multi form

Name
Firma
Adresse

M+K 6/86

Registra AG · CH-8132 Egg/ZH
Gewerbestrasse 16 · Tel. (01) 98 42 424

DATA MAIL 1

NEU! DATA MAIL 1 AG

Versand sämtlicher Standard Soft- und Hardware.
Data Mail-Preise sind Ihr Gewinn! ...

* dBase III Plus D	**1535.–**
dBase III Plus E	**1535.–**
TEX-ASS-Window-Plus D	**1990.–**
Lotus 1-2-3 E/D/F	**790.–**
Page Maker PC-Version	**1440.–**
Reflex E	**260.–**
Turbo Pascal 3.0 E	**190.–**
GEM Collection E	**360.–**
Hardcard Plus (20 MB)	**1960.–**
MS Mouse Serial	**395.–**
Paradise AutoSwitch EGA Card	**1060.–**
Advantage 1 MB AT	**1660.–**

Auf Anfrage senden wir Ihnen gerne unsere vollständige Preisliste (IBM, Apple).

* Ab Lager lieferbar! Exkl. WUST 6,2%, exkl. Versandspesen.

DATA MAIL 1 AG
Güterstrasse 187, 4002 Basel
Telefon: **061/35 20 90**

Gerettete Daten auf der Commodore-Floppy

Wem ist dies nicht schon einmal passiert: Daten auf einer Diskette werden gelöscht, und noch bevor der Irrtum bemerkt wird, sind Daten verschwunden, die eigentlich doch noch gebraucht wurden. Aber Daten, die auf einer Diskette gelöscht werden, müssen nicht unbedingt verloren sein. Wir zeigen Ihnen am Beispiel des Commodore-Laufwerkes 1541 für den C64, was zur eventuellen Rettung schon gelöschter Daten unternommen werden kann.

Oliver Rosenbaum

Dem verzweifelten Benutzer kommt hier eine technische Besonderheit der Diskettenorganisation zugute: Daten werden beim «Löschen» nicht eigentlich eliminiert, sondern ihr Speicherplatz auf der Diskette wird lediglich zum neuerlichen Beschreiben freigegeben. Hieraus ist abzuleiten, dass nach einem (versehentlichen) Löschvorgang kein Schreibvorgang auf der Diskette folgen darf, denn der freigegebene Platz könnte dann mit neuen Daten belegt werden und die ursprünglich vorhandenen Daten sind endgültig verloren. Um die «gelöschten» Daten wieder zugänglich zu machen, sind verschiedene Schritte notwendig. Man benötigt zunächst einen Disk-Editor, also ein Programm, welches es erlaubt, einzelne Sektoren der Diskette zu lesen, bzw. zu ändern. Dieser neue Disk-Editor muss zunächst in den Rechner geladen werden. Im Bildschirmdialog können beliebige Sektoren, bzw. bestimmte Bytes gelesen oder geändert werden. Ohne Editor geht es auch mit den entsprechenden Hilfsprogrammen (sind auf der Demo-Diskette zum Laufwerk 1541, welche beim Kauf mitgeliefert werden), jedoch nicht so komfortabel.

Geändert werden sollen diejenigen Stellen auf der Diskette, welche anzeigen: «Sektor belegt» oder »Sektor frei». Dazu muss man natürlich etwas über den Aufbau einer Diskette und der System-seitig belegten Sektoren wissen (siehe Abb. 1).

Bei der Commodore-Floppy der Laufwerke 1540, 1541 und 4040 liegt das Inhaltsverzeichnis (BAM) der Diskette auf der Spur 18, Sektor 0 und belegt einen Speicherplatz von 128 Bytes (siehe Abb. 2). BAM ist die Abkürzung für **B**lock **A**vailability **M**ap (Inhaltsverzeichnis der verfügbaren Blöcke).

Bei gewöhnlicher Diskettenbenutzung, also Speichern, bzw. Lesen von Programmen und Daten, muss der Benutzer nichts über die BAM und deren Aufbau wissen, da sie automatisch vom DOS (Disk Operating System) verwaltet wird. Beim Speichern von Daten wird die BAM vom DOS abgefragt, ob und wieviel Speicherplatz auf der Diskette noch frei ist und auch, wo dieser Speicherplatz zu finden ist. Wenn die Diskette belegt ist, oder der angeforderte Speicherplatz nicht mehr verfügbar ist, meldet das DOS dies dem Benutzer über den Bildschirm und der «Speichervorgang» wird unterbrochen (bzw. erst gar nicht begonnen). Im Falle einer Speicherung von Daten wird die BAM auf den aktuellen Stand gebracht durch Eintrag der entsprechenden Werte.

Beim «Löschen» einer Datei wird in die BAM lediglich die Information geschrieben, dass der betreffende Speicherbereich auf der Diskette wieder «frei» ist, also neu belegt werden kann (siehe Abb. 5).

Mit Hilfe der BAM lässt sich also jede Datei auf der Diskette genau lokalisieren. Denn die Dateien und Programme werden natürlich nicht so abgespeichert, wie sie im Inhaltsverzeichnis «$» aufgeführt werden. Das kann auch nicht so sein: Wird z.B. ein bestehendes Programm geladen, im Arbeitsspeicher erweitert (vergrössert) und unter dem gleichen Namen neu abgespeichert, benötigt es ja auch mehr Platz auf der Diskette. Nun werden die im Inhaltsverzeichnis folgenden Programme keineswegs «verschoben» um dem nun grösseren Programm Platz zu machen, sondern in der BAM wird notiert, an welcher Stelle der Diskette das Programm oder die Datei fortgesetzt wird (siehe Abb. 6).

Sektorbelegung der Spuren

SPUR	SEKTORNUMMER	SEKTOREN
1 bis 17	0 bis 20	21
18 bis 24	0 bis 18	19
25 bis 30	0 bis 17	18
31 bis 35	0 bis 16	17

Bild 1

Aufbau der BAM (Spur 18, Sektor 0)

BYTE	INHALT	DEFINITION
0,1	18,01	Spur und Sektor des ersten Directoryblocks
2	65	ASCII-Zeichen «A» (=1540/1541/4040-Format)
3	0	0-Flag für zukünftige DOS-Erweiterungen
4-143		Bitmuster der freien bzw. belegten Blocks (0=belegt, 1=frei)

Bild 2

Directory Header (Vorspann) Spur 18, Sektor 0

BYTE	INHALT	DEFINITION
144-161		Name der Diskette (ergänzt mit «geshiften Spaces»)
162,163		ID-Kennzeichnung der Diskette
164	160	«geshifte Space»
165,166	50,65	ASCII-Darstellung von «2A» (DOS-Version und Format)
166,167	160	«geshiftetes Space»
177-255	0	mit «0» aufgefüllt, nicht benutzt

Bild 3

Directory Format (Spur 18, Sektor 1)

BYTE	INHALT
0,1	Spur und Sektor des nächsten Blocks der Directory
2-31	Eintrag des 1. Files
34-63	Eintrag des 2. Files
66-95	Eintrag des 3. Files
98-128	Eintrag des 4. Files
130-159	Eintrag des 5. Files
162-191	Eintrag des 6. Files
194-223	Eintrag des 7. Files
226-255	Eintrag des 8. Files

Bild 4

Format eines Directory Eintrags

BYTE	INHALT
0	Filetyp verknüpft (OR) mit $80 0 = DELeted 1 = SEQuential 2 = PROGram 3 = USeR 4 = RELative
1,2	Spur und Sektor des 1. Datenblocks
3-18	Filename (ergänzt mit «geshifteten Spaces»)
19,20	Spur und Sektor des ersten Blocks (nur für Relatives)
21	Blocklänge (nur für Relatives)
22-25	nicht benutzt
26,27	Spur und Sector des neuen Files beim Ueberschreiben mit β
28,29	Anzahl der Blocks im File (Low Byte, High Byte)

Bild 5

Format eines Sequentiellen Files

BYTE	INHALT
0,1	Spur und Sektor des nächsten Datenblocks
2-255	254 Bytes Daten

Format eines Programmfiles

BYTE	INHALT
0,1	Spur und Sektor des nächsten Programmblocks
2-255	254 Bytes Programm im VC-Speicherformat. Das Ende des Files ist durch 3 Nullen gekennzeichnet.

Bild 6

Das Dienstprogramm VIEW-BAM (auf der Demo-Diskette) liest die BAM von der Diskette und zeigt sie auf dem Bildschirm an. Hierbei werden die Sektorennummern vertikal und die Spurnummern horizontal gezählt.

Die belegten Blöcke erscheinen dunkel und die freien Blöcke hell. Das erste Byte jedes Dateneintrages enthält die Information über den Block. Findet man hier eine Null, so ist dieser Block freigegeben. Soll der ursprüngliche Eintrag des Blocks wieder zugängig gemacht werden, muss hier ein Code eingetragen werden:

0 = frei
1 = Sequentielles File
2 = Programmfile
3 = User-File
4 = Relatives File

Hier im ersten Block einer Datei findet man ebenfalls die Adresse des folgenden Blockes und so kann man Stück für Stück die verloren gegelaubte Datei (Programm) wieder aktivieren. (Die Adresse des ersten Blocks steht natürlich im Inhaltsverzeichnis) siehe Abb. 3 und 4.

Aufgrund dieser Manipulation ist die BAM der Diskette jedoch nicht mehr aktuell, da «von Hand» geändert wurde. Daher muss nun auch die BAM »von Hand» aktualisiert werden. Im Normalfall werden diese Schritte vom DOS erledigt. Zuletzt sei noch auf das Handbuch verwiesen, das zwar keine Anleitung für den hier beschriebenen Vorgang macht, aber die einzelnen Dienstprogramme und Blockzugriffe sind dort nochmal ausführlich beschrieben. Im Prinzip kann diese Methode bei allen Disketten durchgeführt werden, wenn diese nach dem gleichen System aufgebaut sind. □

COMPUTER SPLITTER

Ende der Röntgenstrahltechnik?

(148/ih) Bis 1990, so besagt eine neue Studie von Frost Sullivan Inc., New York, werden neue digitale Technologien zur Speicherung, Uebermittlung und Rekonstruierung von Bildern das diagnostische Werkzeug der vergangenen Jahrzehnte, die Röntgenstrahltechnik, ergänzen und vielleicht sogar ersetzen. Das traditionelle Röntgenstrahlverfahren wird im Vergleich zu digitalen Techniken bald immer weniger flexibel und leistungsfähig sein. Bis 1992 sollen digitale Bildanalysegeräte für den medizinischen Bereich in den USA ein jährliches Marktvolumen von bis zu 347 Mio. Dollar erreichen. Dies schliesst sowohl Geräte, die als Zusatz zu bestehenden diagnostischen Einrichtungen angeboten werden, sowie komplette Systeme ein. Noch ist die neue Technologie nicht ausgereift. Ausserdem ziehen viele Radiologen immer noch den Röntgenfilm als Analysewerkzeug vor. Langfristig gesehen jedoch werden die beschränkten Möglichkeiten der Röntgenstrahltechnik ein wachsendes Problem darstellen. Nebst einer Analyse der Benutzergruppen dieser neuen digitalen Systeme (allen voran die Spitäler), enthält die erwähnte Studie auch eine Zusammenfassung der Anbieter, geführt von Raytel, Colorado Video und DataSpan. Mehr Informationen sind erhältlich bei: Frost Sullivan Inc., 106 Fulton Street, New York, NY 10038, USA. □

Berechnung elektrischer Filter in Turbo-Pascal

Bei der Entwicklung elektronischer Schaltungen müssen oft komplizierte Berechnungen durchgeführt werden. Dies betrifft vor allem das Gebiet der analogen Schaltungstechnik und speziell der Filtertechnik. Viele der Berechnungen können durch einen Arbeitsplatzrechner ausgeführt werden, wodurch der Entwickler von zeitraubender Routinearbeit entlastet wird. Dank müheloser Berechnung ist es auch möglich, verschiedene Fälle zu untersuchen, um das Optimum zu finden.

Ernst Pfenninger

Die Aufgaben bei der Entwicklung elektronischer Schaltungen bestehen einerseits aus der Synthese, anderseits aus der Analyse von Schaltungen. Das vorliegende Programm unterstützt die Synthese von Bandpassfiltern 4. Ordnung. Die Synthese ist meistens erheblich schwieriger als die Analyse, weshalb für die Synthese von Schaltungen noch wenig Software zur Verfügung steht. Für die Analyse jedoch steht schon heute ein Programm zur Verfügung, mit dessen Hilfe sich in beliebigen Netzwerken mit beliebiger Anordnung der Elemente alle Ströme und Spannungen bei jeder Frequenz berechnen lassen. Da für die Synthese meist idealisierte Annahmen über die verwendeten Bauelemente vorliegen, empfiehlt es sich in jedem Fall, nach der Entwicklung der Schaltung eine Analyse unter Berücksichtigung von Nebeneffekten vorzunehmen. Meist kann dann in mehreren Schritten ein optimiertes Ergebnis gefunden werden.

Für die Synthese muss eine exakte Aufgabenstellung vorliegen, z.B. wird ein Bandpassfilter mit gewissen Daten verlangt, und für jede Aufgabenstellung muss mit Hilfe der speziellen Theorie ein Programm erstellt werden.

Der Bandpassfilter

An dieser Stelle soll nur kurz umrissen werden, was ein Bandpassfilter für Eigenschaften hat. In [1] ist die Theorie ausführlich dargestellt.

Ein Bandpassfilter 4. Ordnung hat einen typischen Amplituden- und Phasengang wie aus Bild 1 ersichtlich. Bei der Mittenfrequenz ist die Dämpfung minimal. Der Durchlassbereich ist definiert als der Bereich zwischen den beiden Frequenzen oberhalb und unterhalb der Mittenfrequenz, bei denen die Dämpfung drei Dezibel beträgt. Die Bandbreite ist gleich der Breite des Durchlassbereichs. Oberhalb und unterhalb des Durchlassbereichs werden die Frequenzen gedämpft, wobei die Amplitude mit dem Faktor 100 abnimmt bei Veränderung der Frequenz um den Faktor 10.

Doch genug der grauen Theorie, jetzt muss noch erwähnt werden, wo solche Filter Verwendung finden. Sie dienen beispielsweise zur Erkennung einer bestimmten Frequenz in einem Nachrichtensignal, zur Wegfilterung von Störungen bei Telefon- und Funkverkehr oder zur Trennung von Modem-Kanälen.

Einen Bandpassfilter 4. Ordnung kann man mit Hilfe von zwei hintereinander geschalteten Bandpassfiltern 2. Ordnung aufbauen. Diese werden gebildet aus einem Kondensator, einer Induktivität und einem Widerstand (passives Filter) oder mit einem Operationsverstärker, Kondensatoren und Widerständen (aktives Filter).

Das Pascal-Programm

Die Berechnung von Bandpassfiltern 4. Ordnung von Hand ist zeitraubend und umständlich. Die Gefahr von Fehlern ist gross. Eine Gleichung 4. Grades ist unter anderem aufzulösen, was nur iterativ möglich ist, da keine geschlossene Lösung existiert. All dies nimmt uns das Programm ab. Es ist in Turbo-Pascal geschrieben und mit einer benutzerfreundlichen Bedienung ausgerüstet. Es sind Menüs vorhanden, um den Ablauf des Programmes zu beeinflussen. Die Auswahl geschieht dabei so, dass mit den beiden Cursortasten für Auf- und Abbewegung der richtige Menüeintrag gewählt und mit der ENTER-Taste aktiviert wird. Falsche Eingaben werden im ganzen Programm erkannt und durch einen Warnton gekennzeichnet.

Nach dem Start des Programmes wird zuerst das Menü zur Auswahl des Filtertyps angezeigt (näheres zum Filtertyp siehe in [1]) sowie die Eingabe von Mittenfrequenz, Bandbreite und Verstärkung verlangt. Als Ergebnisse sind abrufbar: Die Koeffizienten der Uebertragungsfunktion, die Daten der Teilfilter, die Komponentenwerte bei der Realisierung mit Operationsverstärker sowie mit LRC-Filtern. Bild 2 zeigt die Schaltung des Filters mit Induktivitäten, Bild 3 den aktiven Filter mit Operationsverstärkern. Das Programm ist im Betrieb selbsterklärend und berechnet in etwa einer

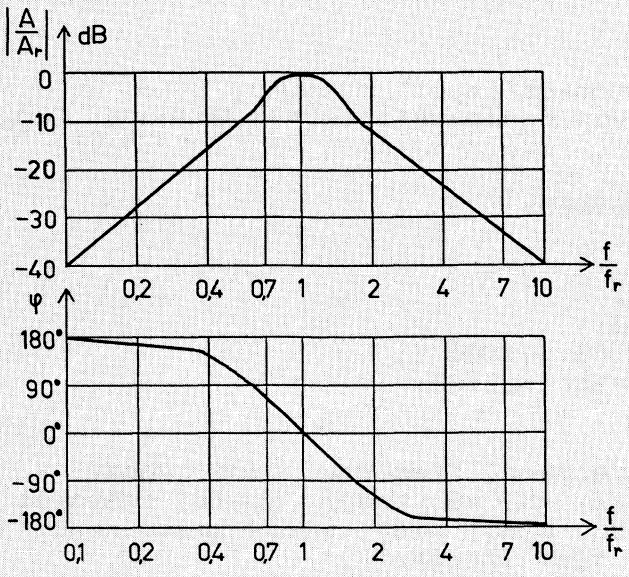


Bild 1: Amplituden- und Phasengang eines Bandpassfilters 4. Ordnung

Literaturhinweis

[1] U. Tietze, Ch. Schenk: Halbleiter-Schaltungstechnik. pringer-Verlag, Berlin.

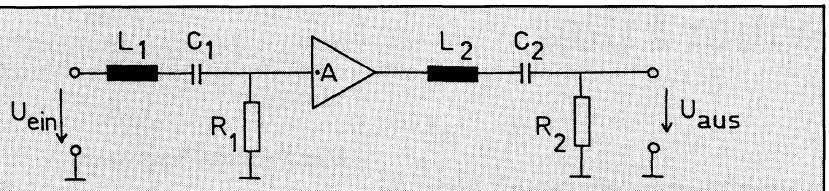


Bild 2: Passiver Bandpassfilter 4. Ordnung. Die Komponenten werden vom Programm berechnet.

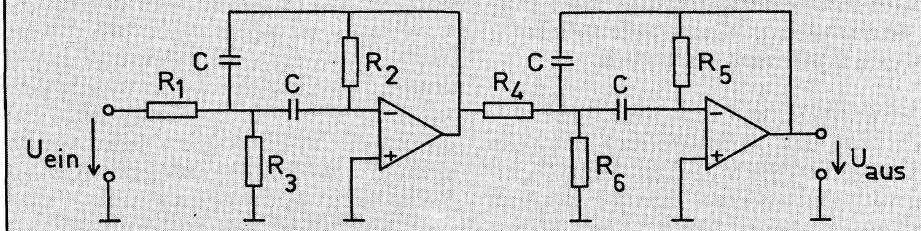


Bild 3: Aktiver Bandpassfilter 4. Ordnung. Die Widerstände R_1 bis R_6 werden vom Programm berechnet.

Sekunde, was von Hand mindestens eine Stunde dauern würde.

Das Programm in Turbo-Pascal kann grundsätzlich auf allen Rechnern mit dieser Programmiersprache angewandt werden. Es wird weder ein grafischer noch ein Farbbildschirm vorausgesetzt. Die Bildschirmgrösse sollte die üblichen 24 Zeilen zu 80 Zeichen umfassen. Einzige Anpassung bei anderen Rechnern könnte die beiden Konstanten «up» und «down» betreffen. Diese sind gleich dem von der Tastatur kommenden Wert bei Betätigung von Cursor-Tasten.

Analyse des Filters

Die Berechnung aller Komponenten des Filters ist mit Hilfe des Programmes eine leichte Sache. Hat man die Werte, sollte nun noch die Analyse mit Hilfe eines Netzwerkanalyse-Programmes vorgenommen werden. Damit lässt sich feststellen, wie sich die Nichtidealitäten von Operationsverstärker, Kondensatoren usw. auswirken. Ausserdem können damit die Auswirkungen der Bauelemente-Toleranzen abgeklärt werden. Meist berechnet der Computer ja Werte, welche ausserhalb einer Normreihe liegen, so dass der nächstliegende Wert verwendet werden muss. All dies hat Auswirkungen auf den Filter, und erst nachdem der Frequenzgang bei verschiedenen Toleranzverhältnissen berechnet worden ist, kann der Filter in einer Applikation verwendet werden.

Schlussfolgerung

Arbeitsplatzrechner bieten bei der Entwicklung elektronischer Schaltungen viele Anwendungsmöglichkeiten. Das vorliegende Programm wurde in der Praxis oft angewendet und brachte einen grossen Zeitgewinn. Mit Hilfe des Netzwerkanalyse-Programms konnte das Ergebnis überprüft und optimiert werden. □

COMPUTER SPLITTER

ASUP

(723/ro) Das «Academic Support Program» ist eine langjährige Aktion der Firma Olivetti um das Betriebssystem UNIX zu verbreiten. Das Sponsor-Programm der in Deutschland mittlerweile grosse Marktanteile gewinnenden Olivetti-Gruppe unterstützt europäische Universitäten mit der kostenlosen Lieferung von Computern nebst UNIX-Betriebssystemen und Programmen. Darüber hinaus wird ein überregionales Kommunikationsnetz unterhalten. Olivetti vermittelt Studentenpraktika in den konzerneigenen Forschungseinrichtungen und denen der Bell Laboratories in den USA. In Deutschland kam nun als erste Universität die Uni Bremen in den Genuss des ASUP. Gegenleistung der Hochschule ist die kostenfreie Veröffentlichung der Projektergebnisse mit den UNIX-Systemen. Das Betriebssystem soll in Forschung und Lehre Einsatz finden. Dafür hat Olivetti beste Voraussetzungen geschaffen: Zum Einsatz kommen die neuen, mit 32-bit Prozessoren bestückten Olivetti AT T3 B2. □

```
program bandpass(input, output);
{Synthese von Bandpassfiltern 4.Ordnung. Copyright by E. Pfenninger}

const
  pi=3.14159265;
  bell=^G;
  up=#5;
  down=#24;
  enter=#13;
  version='24.6.1986';
type str13=string[13];

var
  a1, b1, relBand, frequenz, band, verst, x: real;
  k1, k2, k3, k4, k5, omega, guete, resVerst: real;
  ind1, ind2, wid1, wid2, kapazitaet: real;
  i, wahl, typ: integer;
  wid: array[1..6] of real;
  zahl: str13;
```

```
procedure format(argument:real);
var
  code,exponent: integer;
  rest:byte;
  wert: real;
  hilfsStr: string[11];
begin
  wert:=abs(argument);
  str(wert:11,hilfsStr);
  if hilfsStr[1]='0' then zahl:='       0'
  else
  begin
    if argument<0 then zahl:='-' else zahl:='';
    val(copy(hilfsStr,10,2),exponent,code);
    rest:=exponent mod 3;
    if hilfsStr[9]='-' then exponent:=-exponent;
    if (exponent<0) and (rest<>0) then rest:=3-rest;
    if rest=0 then zahl:=zahl+copy(hilfsStr,1,7)
    else
    begin
      zahl:=zahl+hilfsStr[1]+hilfsstr[3];
      if rest=1 then zahl:=zahl+'.'+copy(hilfsStr,4,4)
      else zahl:=zahl+hilfsStr[4]+'.'+copy(hilfsStr,5,3);
      exponent:=exponent-rest;
    end;
    if abs(exponent)>9 then str(exponent:3,hilfsStr)
      else str(exponent:2,hilfsStr);
    if exponent<>0 then zahl:=zahl+' E'+hilfsStr;
  end;
end; {formatReal}

procedure holeTabellenwerte;
begin
  case typ of
   1: begin a1:=1.2872; b1:=0.4142; end;
   2: begin a1:=1.3617; b1:=0.6180; end;
   3: begin a1:=1.4142; b1:=1.0000; end;
   4: begin a1:=1.3614; b1:=1.3827; end;
   5: begin a1:=1.3022; b1:=1.5515; end;
   6: begin a1:=1.1813; b1:=1.7775; end;
   7: begin a1:=1.0650; b1:=1.9305; end;
  end; {case}
end; {holeTabellenwert}

function wertA(y:real):real;
begin
  wertA:=y*relBand*a1/(b1*(1+sqr(y)))
end; {wertA}

function wertB(y:real):real;
begin
  wertB:=sqr(y)+sqr(wertA(y))+1/sqr(y)-2-sqr(relBand)/b1;
end; {wertB}

function menuWahl(anz:integer):integer;
var zahl, altZahl: integer;
    eingabe: char;
begin
  writeln; writeln; writeln('Cursor fuer auf, ab    ENTER fuer Wahl');
  zahl:=1; altZahl:=zahl;
  repeat
    gotoXY(2,altZahl+2); write('   '); altZahl:=zahl;
    gotoXY(2,zahl+2); write('-->'); gotoXY(40,zahl+2);
    read(kbd, eingabe);
    case eingabe of
```

```
      up:   if zahl>1 then zahl:=pred(zahl);
      down: if zahl<anz then zahl:=succ(zahl);
      enter: begin end;
    else write(bell);
    end;  {case}
  until eingabe=enter;
  menuWahl:=zahl;
end;  {menuWahl}

procedure eingabeFiltertyp;
begin
  clrScr;
  gotoXY(1,24); write('Copyright E. Pfenninger. Version ',version);
  gotoXY(1,1);
  writeln('Wahl des Filtertyps fuer Berechnung von Bandpassfilter',
          ' 4. Ordnung:'); writeln;
  writeln('     Kritisch gedaempft');
  writeln('     Bessel');
  writeln('     Butterworth');
  writeln('     Chebychew mit 0.5 dB Rippel');
  writeln('     Chebychew mit 1 dB Rippel');
  writeln('     Chebychew mit 2 dB Rippel');
  writeln('     Chebychew mit 3 dB Rippel');
  typ:=menuWahl(7);
end;  {eingabeFiltertyp}

procedure wahlDerAusgabe;
begin
  clrScr; writeln('Ausgabe:'); writeln;
  writeln('     Uebertragungsfunktion');
  writeln('     Filterdaten');
  writeln('     Komponentenwerte aktives Filter');
  writeln('     Komponentenwerte passives Filter');
  writeln('     Neustart');
  writeln('     Ausgang zum Betriebssystem');
  wahl:=menuWahl(6);
end;  {wahlDerAusgabe}

procedure writeAusgangsDaten;
begin
  clrScr;
  format(frequenz); writeln('Mittenfrequenz  ',zahl,' Hz');
  format(band);     writeln('Bandbreite      ',zahl,' Hz');
  format(verst);    writeln('Verstaerkung    ',zahl);
  writeln; writeln;
end; {writeAusgangsDaten}

procedure eingabeFilterdaten;
var ok: boolean;
    antwort: char;
begin
repeat
  repeat
    clrScr;
    write('Mittenfrequenz in Hz (positiv):  ');
    {$I-}  read(frequenz)  {$I+}; ok:=(frequenz>0) and (IOresult=0);
    if not ok then write(bell);
  until ok;
  repeat
    clrScr;
    format(2*frequenz);
    write('Bandbreite in Hz (0 < Bandbreite < ',zahl,'):  ');
    {$I-}  read(band)  {$I+};
    ok:=(band<2*frequenz) and (IOresult=0) and (band>0);
    if not ok then write(bell);
  until ok;
```

```
  repeat
    clrScr;
    write('Verstaerkung:  ');
    {$I-}  read(verst)  {$I+};
    ok:=(IOresult=0);
    if not ok then write(bell);
    verst:=abs(verst);
  until ok;
  writeAusgangsDaten;
  write('In Ordnung ? (j/n)  ');
  repeat read(kbd,antwort) until upCase(antwort) in ['J','N'];
until upCase(antwort)='J';
clrScr;
end;  {eingabeFilterdaten}

function sign(wert: real):integer;
begin
  if wert<0 then sign:=-1 else sign:=1;
end; {sign}

procedure berechneX;
var unterArg, oberArg: real;
begin
  relBand:=band/frequenz; unterArg:=1E-6; oberArg:=1;
  if (a1=1.2872) and (b1=0.4142) then x:=1
  else
    repeat
      x:=(unterArg+oberArg)/2;
      if sign(wertB(x))=sign(wertB(unterArg)) then unterArg:=x else oberArg:=x;
    until abs(unterArg-oberArg)<1E-5;
end;  {berechneX}
procedure berechneUebertrFkt;
begin
  omega:=2*pi*frequenz; guete:=1/wertA(x);
  k1:=-1*verst*sqr(relBand)/b1/sqr(omega); k2:=x/guete/omega;
  k3:=sqr(x/omega); k4:=1/guete/x/omega; k5:=1/sqr(x*omega);
  resVerst:=relBand*sqrt(verst/b1)*guete;
end;  {berechneUebertrFkt}
procedure warteAufZeichen;
var zeichen: char;
begin
  writeln; writeln; write('Weiter ? (j/n) ');
  repeat read(kbd, zeichen) until zeichen in ['j','J'];
end; {warteAufZeichen}
procedure ausgabe;
var ok: boolean;
begin
  case wahl of
   1:  begin
         writeAusgangsDaten;
         writeln('Uebertragungsfunktion:'); writeln;
         format(k1); writeln('K1= ',zahl); format(k2); writeln('K2= ',zahl);
         format(k3); writeln('K3= ',zahl); format(k4); writeln('K4= ',zahl);
         format(k5); writeln('K5= ',zahl);
         writeln;
         writeln('                      K1 * w^2');
         writeln('A(w) = ----------------------------------');
         writeln('       (1+j*K2*w-K3*w^2)*(1+j*K4*w-K5*w^2)');
         warteAufZeichen;
       end;
   2:  begin
         writeAusgangsDaten;
         writeln('Filterdaten'); writeln;
         writeln('1. Teilfilter: Bandpass 2. Ordnung');
         format(frequenz*x); writeln('Frequenz: ',zahl,' Hz');
```

```
        format(guete); writeln('Polguete: ',zahl);
        format(resVerst); writeln('Verstaerkung: ',zahl); writeln;
        writeln('2. Teilfilter: Bandpass 2. Ordnung');
        format(frequenz/x); writeln('Frequenz: ',zahl,' Hz');
        format(guete); writeln('Polguete: ',zahl);
        format(resVerst); writeln('Verstaerkung: ',zahl);
        warteAufZeichen;
      end;
  3:  begin
        repeat
          clrScr;
          write('Kondensator (Farad):  ');
          {$I-}  read(kapazitaet)  {$I+};
          ok:=(kapazitaet>0) and (IOresult=0);
          if not ok then write(bell);
        until ok;
        wid[2]:=guete/x/pi/frequenz/kapazitaet;
        wid[5]:=guete*x/pi/frequenz/kapazitaet;
        wid[1]:=wid[2]/2/resVerst; wid[4]:=wid[5]/2/resVerst;
        wid[3]:=resVerst*wid[1]/(2*sqr(guete)-resVerst);
        wid[6]:=resVerst*wid[4]/(2*sqr(guete)-resVerst);
        writeAusgangsDaten;
        writeln('Komponenten des aktiven Filters:');
        writeln;
        for i:=1 to 6 do
        begin
          format(wid[i]); writeln('R',i,'=',zahl,' Ohm');
        end;
        format(kapazitaet); writeln('Kondensatoren: ',zahl,' Farad');
        writeln; writeln('Die open-Loop-Verstaerkung der',
          ' Operationsverstaerker muss gross sein');
        format(2*sqr(guete)); writeln('gegenueber ',zahl);
        warteAufZeichen;
      end;
  4:  begin
        repeat
          clrScr;
          write('Kondensator (Farad):  ');
          {$I-}  read(kapazitaet)  {$I+};
          ok:=(kapazitaet>0) and (IOresult=0);
          if not ok then write(bell);
        until ok;
        ind1:=1/sqr(omega*x)/kapazitaet; ind2:=sqr(x/omega)/kapazitaet;
        wid1:=sqrt(ind1/kapazitaet)/guete; wid2:=sqrt(ind2/kapazitaet)/guete;
        writeAusgangsDaten;
        writeln('Komponentenwerte fuer passives Filter'); writeln;
        format(ind1); writeln('L1=',zahl,' Henry');
        format(wid1); writeln('R1=',zahl,' Ohm');
        format(ind2); writeln('L2=',zahl,' Henry');
        format(wid2); writeln('R2=',zahl,' Ohm');
        format(kapazitaet); writeln('C=',zahl,' Farad');
        format(sqr(resVerst)); writeln('Verstaerkung A=',zahl);
        warteAufZeichen;
      end;
  6:  begin clrScr; writeln('Ende der Bandpassfilterberechnung'); end;
  end; {case}
end;  {ausgabe}

begin  {Hauptprogramm}
  repeat
    eingabeFiltertyp; holeTabellenwerte; eingabeFilterdaten;
    berechneX; berechneUebertrFkt;
    repeat wahlDerAusgabe; ausgabe until wahl in [5,6];
  until wahl=6;
end.  {program Bandpass}
```

Mandelbrot-Explorer

Vor einiger Zeit präsentierten die Zeitschriften «GEO» und «Spektrum der Wissenschaft» ihren Lesern faszinierende Computergrafiken. Die bizarren Bilder zeigten grafische Entsprechungen sogenannter mathematischer Rückkopplungen. Zuden wurden auch die Grundzüge eines Programmes beschrieben, mit dem sich solche Grafiken erzeugen lassen. Das hier vorgestellte Programm basiert auf dieser Beschreibung. Selbstverständlich lassen sich die damit auf einem PC erzeugten Bilder nicht mit denjenigen vergleichen, welche die beiden Artikel illustrierten. Doch schon dieses relativ einfache Programm liefert Bilder von solch suggestiver Wirkung, dass wohl der eine oder andere nimmermüde Computer-Forscher am Ende des Monats überrascht sein wird, wie hoch seine Stromrechnung ausgefallen ist. Denn selbst ein IBM-PC/AT mit 80287 Co-Prozessor benötigt mehr als eine halbe Stunde für die Berechnung einer einzelnen Grafik.

Andreas Pichler

Theoretische Grundlage des Programmes sind mathematische Rückkopplungen. Von einer mathematischen Rückkopplung spricht man dann, wenn eine bestimmte Vorschrift immer wieder mit ihrem eigenen Ergebnis «gefüttert» wird. Betrachten wir dazu die komplexe Zahlenebene und die Vorschrift:

$$\underline{z} \leftarrow -\underline{z}^2 + \underline{c}$$

Wir setzen die Konstante $\underline{c}$ gleich dem Wert der zu untersuchenden Zahl und die Variable $\underline{z}$ auf $\underline{z}^0 = 0 + i0$. Dann lassen wir die Vorschrift laufen. Das Ganze wiederholen wir für jede Zahl der Ebene. Untersuchen wir daraufhin das Resultat, so stellen wir fest, dass es im Zentrum der Ebene eine Menge von Zahlen gibt, bei denen der Wert von $\underline{z}$ auch nach beliebig vielen Iterationen einen bestimmten Grenzwert nicht überschritten hat, während er bei Zahlen ausserhalb dieser Menge schnell einmal gegen Unendlich gewachsen ist. Die Menge wird nach Benoit B. Mandelbrot, einem Wissenschaftler am IBM Thomas J. Watson Forschungszentrum, «Mandelbrot-Menge» benannt. Grafisch umgesetzt erhält man von dieser Menge eine Struktur, die in der deutschsprachigen Literatur oftmals als **«Apfelmännchen»** bezeichnet wird. Ein «Apfelmännchen» sieht aus wie eine arg deformierte, auf der Seite liegende Acht mit knospenartigen Ausbuchtungen. Sein Umfang ist umsäumt von einem filigranen Halo, einer Grenzfläche unendlicher Komplexität, auf die sich das eigentliche Interesse der Mathematiker und Physiker konzentriert. Vergrössert man Teile davon, offenbaren sich neue feine und feinste Strukturen, die im groben Raster verborgen blieben. Sie können bis an die Grenzen Ihrer Gleitkomma-Arithmetik (und Ihrer Geduld) gehen, ohne je auf ein Ende zu stossen. Und immer wieder werden Sie auch auf Miniaturausgaben des «Apfelmännchens» stossen, hundertmal kleiner, tausendmal kleiner, unendlich (?) mal kleiner.

Programmbeschreibung

Eigentlicher Kern des Programmes ist die unscheinbare Prozedur «mandelbrot()». Sie untersucht komplexe Zahlen einer definierten Ebene auf ihre Zugehörigkeit zur Mandelbrot-Menge. Selbstverständlich kann dabei eine einzelne Zahl nicht unendlich lange untersucht werden; gewisse Abbruchkriterien müssen eingeführt werden, um das Verfahren abzukürzen. Glücklicherweise garantiert ein Resultat aus der Iterationstheorie (vgl. Literatur [2]), dass $\underline{z}$ gegen Unendlich strebt, wenn der Betrag von $\underline{z}$ irgendwann grösser als 2 wird. Das zweite Abbruchkriterium ist weitaus profaner. Eine obere Grenze bestimmt, wieviele Iterationen maximal für eine Zahl ausgeführt werden. Erreicht $\underline{z}$ nämlich auch nach dieser Anzahl Iterationsdurchgänge den Betragswert von 2 nicht, wird schlicht angenommen, dass diese Zahl zur Mandelbrot-Menge gehört. Dies zur Theorie. Praktisch erhält «mandelbrot()» via Parameterschnittstelle ein Byte-Array «map», den Ursprung einer komplexen Ebene «origin» und deren Seitenlänge «side» sowie die maximale Anzahl Iterationsschritte «step» für eine Untersuchung. Aus der Seitenlänge der Ebene und des Byte-Arrays errechnet die Prozedur dann einen Rasterabstand für die Abbil-

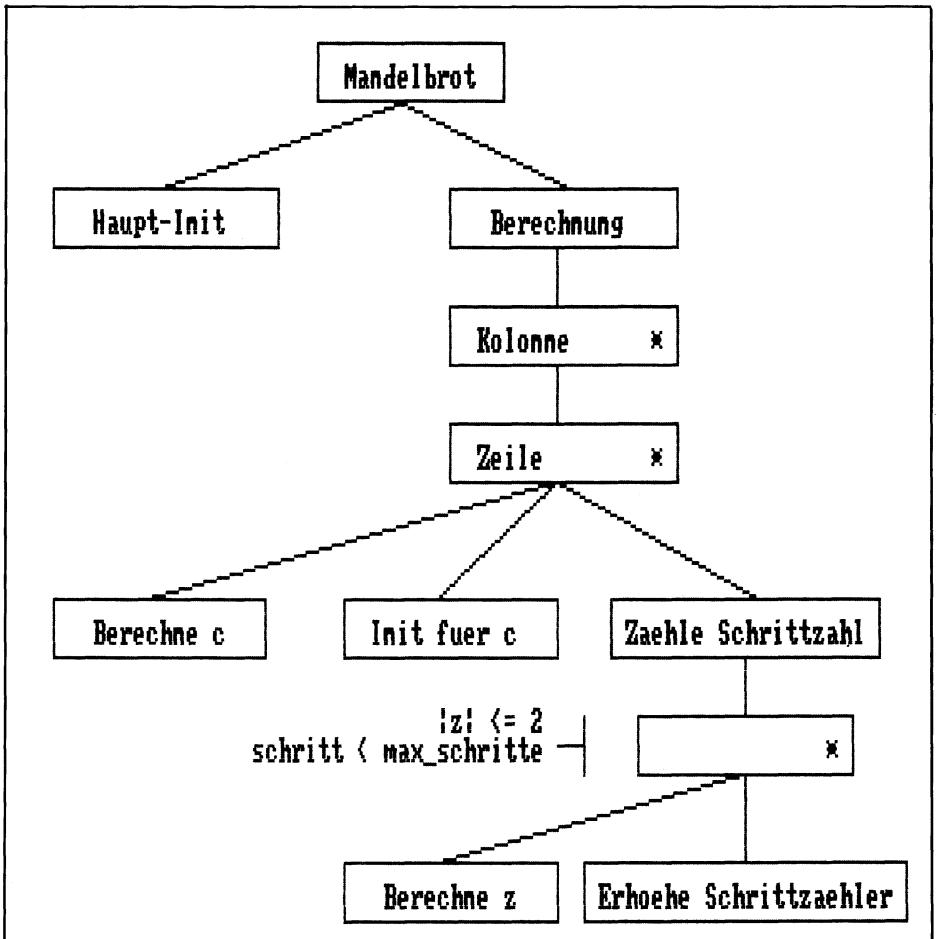


Abb. 1: Michael Jackson-Struktogramm des Kernalgorithmus

```
Generate Map ..........<G>
Retrieve Map ..........<R>
Examine Map ...........<E>
Store Map .............<S>

Exit ..................<Q>



Select item :
```

Abb. 2: Mandelbrot-Set Exploser

```
Parameter Input
---------------


Enter real term of origin :


Invalid Entry in -1.2 e-4
                     ^-- Error
```

Abb. 3: Mandelbrot-Set Exploser

dung der kontinuierlichen Ebene auf das Array. Jedes Element des Arrays steht also für einen diskreten Wert der Ebene, wobei «origin», die linke obere Ecke der Ebene, dem Element [0,0] entspricht und RE(«origin» + «side»)/ IM(«origin» - «side«), die rechte untere Ecke, dem Element [«max_point», «max_point»]. Dabei ist zu beachten, dass der Realteil zunimmt, während der Imaginärteil abnimmt. Grund dafür ist die spätere grafische Darstellung. Auf die so gewonnenen Rasterwerte wird nun die obige Vorschrift angewendet. Die Variable «node» entspricht dabei c und «eval» entspricht z. Abb. 1 zeigt den dafür verwendeten Algorithmus in Form eines Struktogrammes. Das ansonsten zu vermeidende Realzahlen-Abbruchkriterium ist hier zulässig, da die Schleife dank des zweiten Abbruchkriteriums auch dann terminiert, wenn die Betragsfolge von «eval» gegen einen Grenzwert unterhalb von 2 konvergieren sollte. Nach Abschluss der Berechnung enthält das Array die für jeden Rasterwert benötigte Anzahl Iterationsschritte. Bedingt durch den Datentyp des Arrays können für eine Zahl höchstens 255 Iterationsdurchgänge ausgeführt werden. Dem Mathematiker mag dieses Kriterium zu recht als etwas niedrig erscheinen, doch für die spätere Darstellung ist es bei weitem genügend. Genügend auch, wenn man bedenkt, dass im Extremfall immer noch 79,5 Millionen Gleitkomma-Operationen durchgeführt werden müssen! Selbst für einen Arithmetik-Prozessor ein hartes Stück Arbeit.

Das mit «mandelbrot()» erzeugte Array kann nun mittels der Prozedur «assign_color()» einer weiteren Bearbeitung unterzogen werden. Als deren Ergebnis erhält man die grafische Ausgabe der untersuchten komplexen Ebene. Dazu wird zuerst analysiert, in welchem Bereich sich die Werte des Arrays bewegen. Dann wird dieser Bereich gemäss der Anzahl der vom Graphics Adapter zur Verfügung gestellten Farben aufgeteilt. Zuletzt wird das gesamte Array auf den Bildschirm ausgegeben, wobei jedes Element, sprich Rasterwert der Ebene, einem Graphics Pixel entspricht. Zahlen der Mandelbrot-Menge werden dabei schwarz, die anderen gemäss einer Farbe entsprechend der Anzahl benötigter Iterationsdurchgänge eingefärbt. Leider erlaubt der Color Graphics Adapter keine grossen Sprünge: gerade vier Farben lassen sich gleichzeitig darstellen. Bei einer maximalen Iterationsgrenze von 255 müssen mit einer Farbe schlimmstenfalls über 80 Werte abgedeckt werden. Dass dabei die suggestive Wirkung verloren geht, scheint einzuleuchten. Ich ziehe deshalb Grenzen zwischen 30 und 100 vor. Zwar zähle ich dann einzelne Zahlen zur Mandelbrot-Menge, obwohl sie dieser gar nicht angehören. Doch eine Auswertung zeigte, dass die daraus resultierende Fehlinterpretation - wenigstens in dem von mir untersuchten Bereich - klein ist, da Nichtmitglieder der Menge die Betragsgrenze von 2 meist schon nach sehr wenigen Durchgängen überschritten. Es liegt an Ihnen zu entscheiden, ob Sie ästhetisch ansprechende oder mathematisch exaktere Bilder generieren wollen. Ganz exakt werden Sie sowieso niemals sein, denn dazu müssten Sie jedes potentielle Mitglied der Mandelbrot-Menge unendlich lange untersuchen.

Die restlichen Prozeduren des Programmes bilden die Benutzeroberfläche. «store()» und «retrieve()» besorgen, unterstützt von Hilfsprozeduren, das File-Handling. «get_parameter()» liest die benötigten Parameter ein und «examine()» erlaubt die Erkundung der generierten Bilder mit Hilfe eines Fadenkreuzes. Das gesamte I/O-Handling ist gegen Fehleingaben geschützt und informiert den Benutzer über gemachte Fehler. Insbesondere sorgt die Prozedur «store()» dafür, dass bereits existierende Dateien nicht überschrieben werden. Für die Konvertierung der Eingabe in den entsprechenden Datentyp wird die Standardprozedur «val()» verwendet. Damit gelten für Parametereingaben die gleichen Syntaxregeln wie in Pascal.

In Abb. 2 sehen Sie das Hauptmenü. Es umfasst vier Auswahlpunkte sowie das EXIT-Kommando zum Verlassen des Programmes. Als Eingabe werden sowohl Klein- als auch Grossbuchstaben akzeptiert. Bei den Auswahlpunkten handelt es sich im Einzelnen um:

- *Generate Map*

Erlaubt Ihnen die Eingabe von Parametern und führt anschliessend automatisch die Untersuchung der von Ihnen spezifizierten komplexen Ebene durch.

- *Retrieve Map*

Liest ein durch Sie spezifiziertes Datenfile ein.

- *Examine Map*

Ermöglicht Ihnen die interaktive Untersuchung der durch Parameter definierten oder von einem File eingelesenen Ebene.

- *Store Map*

Speichert sämtliche benötigte Parameter in ein durch Sie spezifiziertes Datenfile ab.

Wie Sie sich die Fehlerbehandlung vorstellen können, demonstriert Abb. 3. Verlangt wurde die Eingabe einer REAL-Zahl. Der Anwender fügte ein Leerzeichen zwischen Argument und Exponent ein. Gemäss Syntaxbeschreibung darf eine REAL-Zahl aber nur die monadischen Operatoren + und -, die zehn Ziffern sowie gegebe-

nenfalls das Symbol E oder e für den Exponenten enthalten. In der Statuszeile wurde die ungültige Zeichenkette noch einmal ausgegeben und die fehlerhafte Stelle markiert. Das Eingabefeld wurde für weitere Versuche bereits wieder gelöscht. Beachten Sie bitte auch, dass es keine Abbruchmöglichkeit gibt. Haben Sie einmal gewählt, müssen Sie bis zum Schluss durchziehen. Einzig <CTRL><C> kann Sie während einer I/O-Operation von der Eingabe befreien, bricht dann aber auch das Programm ab.

Für die Erkundung können Sie auch eine Maus verwenden. Allerdings sind im Programm keine Schutzmechanismen für deren Kontrolle implementiert. Bei zu schnellen Bewegungen kann es deshalb zu Programmabstürzen kommen. Bei meinem System wurde beispielsweise das Programm abgebrochen, wobei jedoch schon zuvor der DOS-Speicherverfolgungsbereich zerstört worden ist, so dass der Befehlsinterpreter COMMAND.COM nicht mehr geladen werden konnte und das ganze System stoppte (Memory Allocation Error Cannot load COMMAND, system halted).

Auf Forschungsreise

Wer weiss, was es noch alles zu entdecken gäbe, würde man sich in der komplexen Ebene auch noch an anderen Orten versuchen. Doch für alle diejenigen, denen es die Apfelmännchen angetan haben, hier ihre Heimat (siehe Abb. 4).

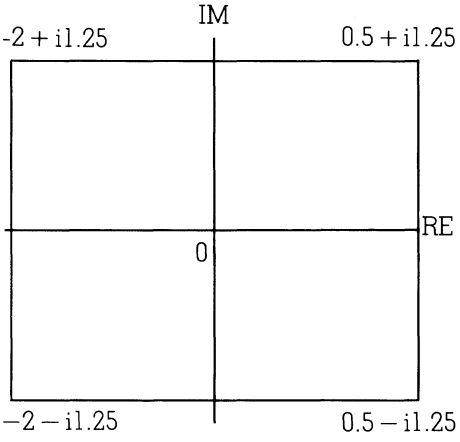


Abb. 4

Versuchen Sie es zuerst einmal mit diesem Gebiet, wobei Sie «step» auf etwa 30 setzen. Sie sollten dann, ohne Stunden darauf zu warten, das Original-Apfelmännchen als grafisches Resultat erhalten (vgl. Abb. 5). So im Gebiet um RE(-1.79) und IM(0) herum werden Sie eine Art Hellebardenspitze erkennen. Diese Spitze enthält

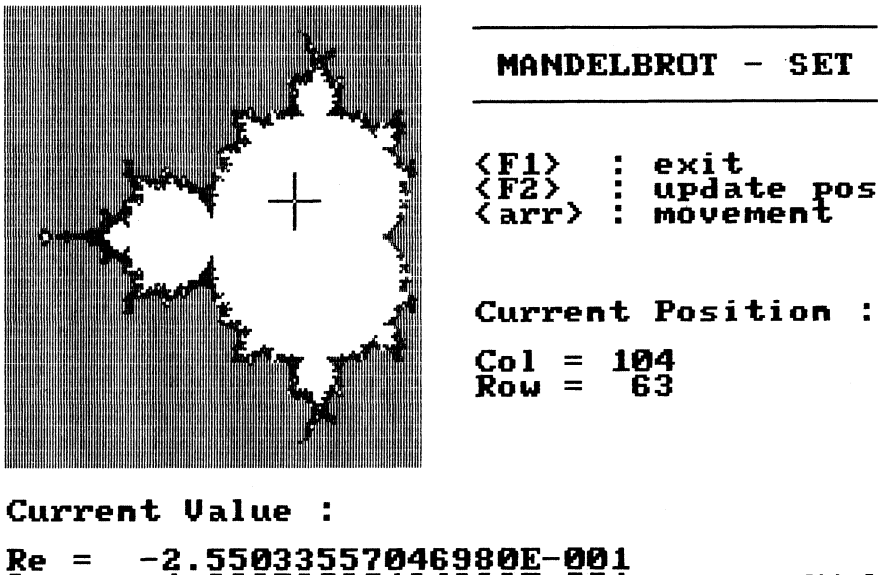


Abb. 5

eine Miniaturausgabe des Apfelmännchens. Mit der momentanen Auflösung ist es kaum auszumachen. Doch dass dem tatsächlich so ist, davon können Sie sich leicht selbst überzeugen. Sie brauchen dazu nämlich nur die Linse Ihres (Programm-) Mikroskopes auf dieses Gebiet zu richten. Die Werte dafür sind ungefähr:

origin = $-1.799 + i2.517\ 10^{-2}$
side = 0.068

Aufgrund der begrenzten Farbmöglichkeiten des Graphics Color Adapters werden Sie an manchen Stellen, wo die Bilder in [1] und [2] filigrane Muster zeigten, nur einfarbige Flächen erhalten. Doch keine Sorge, die Muster sind vorhanden, Sie können sie nur nicht sehen. Spielen Sie verschiedene Werte von «step» durch - Mathematiker mögen mir verzeihen - und die Strukturen werden sich Ihnen offenbaren!

Vergleich

Um den Zeitbedarf der Berechnungen abzuschätzen, liess ich das Programm je auf einem IBM-PC/AT sowie auf einem IBM-PC-kompatiblen Rechner (MULTITECH MPF-PC 522) ablaufen. Beide Rechner sind mit den entsprechenden Co-Prozessoren ausgerüstet. Berechnet wurde jeweils die gleiche komplexe Ebene mit der maximalen Anzahl unterstützter Iterationsdurchgänge. Implementiert wurde das Programm in Turbo-Pascal [3]. Die folgende Tabelle gibt Auskunft über die dabei erzielten Resultate:

Parameter: origin side step	=	$-2 + i1.25$ 2.5 255
PC mit iAPX88/10	:	178 min
PC mit iAPX88/20	:	62 min
AT mit iAPX286/10	:	64 min
AT mit iAPX286/20	:	36 min

Verbesserungen

Der wohl gravierendste Mangel des Programmes ist, wie bereits schon mehrfach angedeutet, die äusserst bescheidene Farbauswahl. Besitzer von Enhanced oder Professional Graphics Adapters, die darüber hinaus noch Kenntnisse über deren Ansteuerung haben, können die Möglichkeiten des Programmes enorm steigern, indem sie diese Karten anstelle des doch schon ziemlich veralteten Color Graphics Adapter verwenden. Mit einer Palette von 16 oder mehr Farben sollten Sie Resultate erzielen können, die denen des «Grafiklabors Dynamischer Systeme» der Universität Bremen nicht mehr viel nachstehen.

Verbesserungsmöglichkeiten finden Sie auch in der Prozedur «examine()». Auch könnte beispielsweise mit Hilfe der Turbo-Graphix Toolbox der Bildschirm um einiges besser gestaltet werden. Als hinderlich erwies sich ferner, dass aus der Erkundung hervorgehende Parameter nicht automatisch an andere Prozeduren übergeben werden können.

Literatur

[1] GEO, 6/84, «Mathematik: Die unendliche Reise», H.O Peitgen/P. Richter

[2] Spektrum der Wissenschaft, 10/85 «Computer-Kurzweil», A.K Dewdney

[3] Borland Int. Turbo-Pascal V3.0 (8087) - Reference Manual

[4] Logitech SA Modula-2/86 8087 R2.0 - User's Manual

Was nützt die schönste Grafik auf dem Bildschirm, wenn man sie nicht zu Papier bringt. Die vorliegende Programmversion unterstützt keinerlei Druckmöglichkeiten. Behelfen kann man sich mit dem DOS-Programm «GRAPHICS» (ab PC-DOS V3.0). Abb. 5 zeigt einen derartigen Ausdruck. Besser ist allerdings eine Integration. Auch hier kann die Graphix Toolbox mit der entsprechenden Prozedur weiterhelfen.

Bleibt zum Schluss die Frage nach Steigerung der Ausführungsgeschwindigkeit. Mehr aus Neugier implementierte ich den Kernalgorithmus, also die Prozedur «mandelbrot()» zusammen mit einem kleinen Umgebungsprogramm, in Modula-2. Ich verwendete dazu das Logitech-System BLS/87 R2.0 [4]. Zu meinem Erstaunen stellte ich dabei fest, dass die 8087-Implementation von Modula-2/86 auf einem AT mehr als doppelt, auf einem PC sogar fast dreimal so schnell wie das entsprechende Turbo-Pascal-Programm ausgeführt wurde. Leider konnte ich noch nicht alle benötigten Bibliotheksmodule schreiben, sonst würde Ihnen dieses Programm sicher in Modula-2 vorliegen. Aber vielleicht versuchen Sie selber eine Implementierung in Modula-2 oder C. Es würde mich interessieren, welche Zeiten etwa M2SDS oder Microsoft-C benötigen. □

Laserprint-Service

für *Macintosh*™-Benützer.

Sie sparen Kosten für Satz und Layout, und Sie sparen Zeit, denn während wir Ihre Daten ab Diskette ausdrucken, können Sie sich schon wieder anderem widmen.

Natürlich *lasern* wir auch auf Ihr mitgeliefertes (Brief-) Papier!

Wir können Ihre Laserkopien zudem als Druckvorlagen für die Anfertigung von **Drucksachen** übernehmen, und wir liefern preisgünstige **Repros** und **Raserfilme** ab Ihren Vorlagen.

Verlangen Sie die Unterlagen für *Desktop Publisher* von

mesa graphix

Thomas Marti, Rugenaustrasse 4

3800 Interlaken 036 22 40 63

```
PROGRAM mandelbrot_explorer;

{------------------------------------------------------------------------

  Author : Andreas Pichler
  Date   : June 1986
  File   : mandelb.pas

  System Requirements :

   HW : IBM-PC/XT/AT or strict compatibles
        Color Graphics Adapter
   SW : PC- or MS-DOS V2.11 or higher
        TurboPascal V3.0 (8087)

   Using an iAPX88/20 (8088 + 8087) is highly recommended !

  Description :

   Provides a program for exploring the so-called MANDELBROT set
   according to an algorithm described by  A. K. Dewdney  in the
   October issue of "Spektrum der Wissenschaft".

------------------------------------------------------------------------}

{$I a:\turbo3\graph.p}  {include extended graphics facilities}

CONST max_point    = 149;
      head_line1   = 2;
      head_line2   = 4;
      head_line3   = 5;
      item1_line   = 10;
      item2_line   = 11;
      item3_line   = 12;
      item4_line   = 13;
      exit_line    = 15;
      message_line = 15;
      return_line  = 19;
      input_line   = 20;
      error_line   = 24;

TYPE  complex  = RECORD re,im : REAL END;
      matrix   = ARRAY [0..max_point,0..max_point] OF BYTE;
      buffer   = RECORD
                   origin : complex;side : REAL;
                   step : INTEGER;map : matrix
                 END;
      item     = (re_origin,im_origin,map_side,max_step);
      file_rec = FILE OF buffer;
      entry    = SET OF CHAR;
      name     = STRING[40];

VAR   proc_rec    : buffer;   {MANDELBROT set parameter}
      reply       : CHAR;     {query reply}
      select      : item;     {item to be input}
      data        : file_rec; {file containing map and info}
      command     : CHAR;     {contains user entry}
      valid       : entry;    {valid command selectors}
      quit        : entry;    {valid exit selectors}
      file_name   : name;     {name of file to be used}
      map_exists  : BOOLEAN;  {map exists yes/no}

{--------- Auxiliary Subroutines ---------------------------------}

PROCEDURE clr_field (start,stop : INTEGER);

{Clear a field defined by <start,stop>}

 VAR index : INTEGER; {loop variable}

 BEGIN
  FOR index := start TO stop DO
   BEGIN
    gotoxy(1,index);clreol
   END
 END {clr_field};
```

```
FUNCTION exist (VAR f : file_rec; file_name : name): BOOLEAN;

{Check existence by opening a file <f> of name <file_name>}

 BEGIN
  assign(f, file_name);
  {$I-} reset(f); {$I+}
  IF ioresult = 0
   THEN exist := TRUE
   ELSE exist := FALSE
 END {exist};


FUNCTION create (VAR f : file_rec; file_name : name): BOOLEAN;

{Create a file <f> of name <file_name>}

 BEGIN
  assign(f,file_name);
  {$I-} rewrite(f); {$I+}
  IF ioresult = 0
   THEN create := TRUE
   ELSE create := FALSE
 END {create};


PROCEDURE header;

{Write header above menu}

 BEGIN
  normvideo;
  gotoxy(5,head_line1);
  write('_____________________________________________');
  gotoxy(5,head_line2);
  write('M A N D E L B R O T - S E T   E X P L O R E R');
  gotoxy(5,head_line3);
  write('_____________________________________________');
  lowvideo
 END {header};


{--------- Subroutines for menu item  - Generate Map ---------------}

PROCEDURE get_parameter(select : item; VAR origin : complex;
                        VAR side : REAL; VAR step : INTEGER);

{Get parameter specified by <select> from keyboard and put it into
 the appropriate global variable <origin,side, or step>.
 GET-PARAMETER() provides full I/O checking}

 CONST error_msg1 = 'Invalid Entry in ';

 VAR   okay   : BOOLEAN;      {control variable}
       answer : STRING[20];   {entry string}
       fault  : INTEGER;      {error position}

 BEGIN
  okay := TRUE;
  gotoxy(5,message_line);write('Parameter Input');
  gotoxy(5,message_line + 1);write('---------------');
  REPEAT
   clr_field(input_line,input_line); {clear previous entries}
   gotoxy(5,input_line);
   CASE select OF
    re_origin : BEGIN {get origin real term}
                 write('Enter real term of origin : ');
                 readln(answer);
                 val(answer,origin.re,fault); {convert to REAL}
                END;
    im_origin : BEGIN {get origin imaginary term}
                 write('Enter imaginary term of origin : ');
                 readln(answer);
                 val(answer,origin.im,fault); {convert to REAL}
                END;
    map_side  : BEGIN {get side length of map}
                 write('Enter side length of map : ');
                 readln(answer);
                 val(answer,side,fault); {convert to REAL}
                END;
    max_step  : BEGIN {get maximum count of iteration steps}
                 okay := TRUE; {re-initialize}
                 write('Enter max count of iteration steps [1..255]: ');
                 readln(answer);
                 val(answer,step,fault); {convert to INTEGER}
                 IF (step < 1) OR (step > 255)
                  THEN okay := FALSE;
                END;
   END;
   clr_field(error_line,error_line+1); {reset error field}

   IF fault <> 0
    THEN {invalid entry}
     BEGIN
      gotoxy(5,error_line);normvideo;write(error_msg1,answer);
      gotoxy(length(error_msg1) + fault + 4,error_line + 1);
      write('^-- Error');lowvideo
     END
    ELSE {check for range error}
     IF NOT okay
      THEN {range error}
       BEGIN
        fault := 1; {force repetition}
        gotoxy(5,error_line);normvideo;write(error_msg1,answer);
        gotoxy(length(error_msg1) + 5,error_line + 1);
        write('^-- Out of range');lowvideo
       END
  UNTIL (fault = 0);
  clr_field(message_line,error_line)
 END {get_parameter};


PROCEDURE mandelbrot (VAR map : matrix; origin : complex;
                      side : REAL; step : INTEGER);

{Implementation of the MANDELBROT algorithm. Maps a complex area
 defined by <origin,side> to a graphics area <map> and  executes
 for each complex node value the algorithm}

 CONST max_value    = 4.0;

 VAR   col,row     : INTEGER; {loop variables}
       node        : complex; {grid node value}
       eval        : complex; {computed value}
       sqr_re_eval : REAL;    {square of eval-real term}
       sqr_im_eval : REAL;    {square of eval-imaginary term}
       save_re     : REAL;    {used for intermediate storage}
       distance    : REAL;    {normalized node distance}
```

```
BEGIN
 lowvideo;
 distance := side / max_point; {compute node distance}
 FOR col := 0 TO max_point DO {scan columns}
  BEGIN
   gotoxy(5,message_line);
   write('Generate Map - Current Column is ',col:0);
   FOR row := 0 TO max_point DO {scan rows}
    BEGIN {process each node}
     BEGIN {compute value of node(col,row) }
      node.re := origin.re + distance * col;
      node.im := origin.im - distance * row
     END;
     BEGIN {initialization}
      map[col,row] := 0;
      eval.re := 0;eval.im := 0;
      sqr_re_eval := 0; sqr_im_eval := 0
     END;
     BEGIN {get count of node[col,row] }
      WHILE ((map[col,row] < step) AND
            ((sqr_re_eval + sqr_im_eval) <= max_value)) DO
       BEGIN
        sqr_re_eval := eval.re * eval.re;
        sqr_im_eval := eval.im * eval.im;
        save_re := sqr_re_eval - sqr_im_eval + node.re;
        eval.im := 2 * eval.re * eval.im + node.im;
        eval.re := save_re;
        map[col,row] := map[col,row] + 1 {count steps}
       END
     END
    END
  END; {map scanning}
  clr_field(message_line,message_line)
END {mandelbrot};

{--------- Subroutines for menu item  - Examine Map ----------------}

PROCEDURE assign_color (VAR map : matrix; step : INTEGER);

{Assigns each node in <map> a color depending on <step>. Node values
 which needed more than <step> iterations are said to  be members of
 the MANDELBROT set and will be plotted black.   The range of itera-
 tions below <step> is divided into three parts and each part is as-
 signed a color}

 CONST max_color = 2; {3 different colors available}

 TYPE  table = ARRAY [0..max_color] OF INTEGER;

 VAR   max_cnt     : INTEGER; {max iteration steps}
       min_cnt     : INTEGER; {min iteration steps}
       limit       : table;   {start value for each color}
       delta_color : INTEGER; {count range covered by one color}
       col,row,i   : INTEGER; {loop variables}
       color_index : INTEGER; {color to be used}
 BEGIN
  BEGIN {get count range}
   max_cnt := 1;min_cnt := step; {initialization}
   FOR col := 0 TO max_point DO {scan columns}
    FOR row := 0 TO max_point DO {scan rows}
     BEGIN {process each node}
      IF map[col,row] < min_cnt THEN min_cnt := map[col,row];
      IF ((map[col,row] < step) AND
         (map[col,row] > max_cnt)) THEN max_cnt := map[col,row]
     END
  END;
  BEGIN {evaluate color ranges}
   delta_color := (max_cnt - min_cnt + 1) DIV (max_color + 1);
   IF (((max_cnt - min_cnt + 1) MOD (max_color + 1)) <> 0)
    THEN delta_color := delta_color + 1; {adjust range}
   FOR i := 0 TO max_color DO {generate range table}
    limit[i] := min_cnt + i * delta_color
  END;
  BEGIN {plot mandelbrot map}
   graphcolormode; {set mode to 320 * 200 color}
   graphbackground(black);
   palette(2); {use black,lightgreen,lightred,yellow}
   FOR col := 0 TO max_point DO {scan columns}
    BEGIN
     FOR row := 0 TO max_point DO {scan rows}
      BEGIN {assign each node a color and plot it}
       IF map[col,row] >= step
        THEN {is supposed to be a MANDELBROT member}
         plot(col,row,0)
        ELSE {not a member of the MANDELBROT set}
         BEGIN
          i := 0;
          WHILE (i <= max_color) DO {determine range}
           IF ((map[col,row] >= limit[i]) AND
              (map[col,row] < (limit[i] + delta_color)))
            THEN {count is in this range}
             BEGIN
              plot(col,row,i+1);
              i := max_color + 1 {force leaving loop}
             END
            ELSE i := i + 1; {try next range}
         END
      END
    END
  END
END {assign_color};


PROCEDURE examine (VAR map : matrix; origin : complex;
                   side : REAL);

{Lets you travelling inside the MANDELBROT picture in  order to find
 interesting places worth to zoom. On request information of current
 position and associated complex node value are provided}

 CONST left_arr  = #75; {return code for left arrow  #27/#75}
       right_arr = #77; {return code for right arrow #27/#75}
       up_arr    = #72; {return code for up arrow    #27/#72}
       down_arr  = #80; {return code for down arrow  #27/#80}
       confirm   = #59; {confirmation = <F1>         #27/#59}
       update    = #60; {update position = <F2>      #27/#60}
       x_start   = 75;  {x-axis center}
       y_start   = 75;  {y-axis center}
       l         = 9;   {length of a crosshair arm }
       header1   = 1;
       header2   = 3;   {position of header}
       header3   = 4;
       key1      = 7;
       key2      = 8;   {position of key information}
       key3      = 9;
       pos_head  = 13;
       col_line  = 15;  {position of position information}
       row_line  = 16;
```

Dataland GmbH
Postfach 212
CH-9100 Herisau 1

Telefon 071 / 52 21 21

dataland
PRINTER & COMPUTER

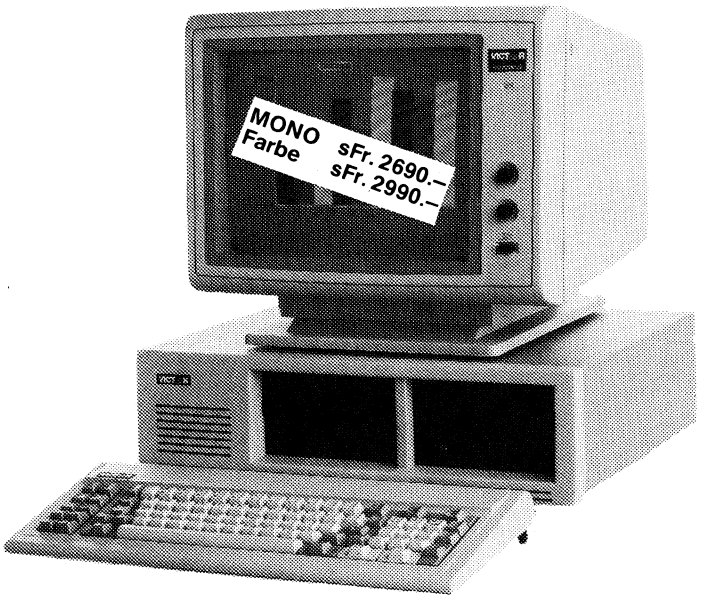
Ein «Professional Computer (PC)» ohne Kompromisse zum SUPERPREIS!

Die Zentraleinheit:
16-Bit, Intel 8088, 4.77 MHz, voll IBM-PC/XT-kompatibel, 256 KB RAM, erweiterbar auf 640 KB RAM, 2×360 KB Diskettenlaufwerke, 7 Erweiterungsslots, davon 5 frei, serielle und parallele Schnittstelle.

Die Tastatur:
Standard-Tastatur (DIN 2137), frei beweglich, Tasten programmierbar, LED-Anzeigen, Zehnerblock.

Die Ausstattung:
MS-DOS 2.11, GW-Basic-Interpreter, Benutzerinterface VICTOR VU, deutschsprachige Handbücher.

VICTOR VPC



Die DATALAND -Komplett-Version:

MONOCHROM-Bildschirm-Version
14 Zoll, grün, dreh- und kippbar, 25×80 Zeichen/Zeilen, 7×9 Matrix, hochauflösend, Helligkeits- und Kontrasteinstellung.

FARB-Bildschirm-Version
14 Zoll, Farb-Bildschirm, 16 Farben, dreh- und kippbar, 520×240 Bildpunkte, 8×8 Matrix.

Die DATALAND-VICTOR-Optionen:
15 oder 30 MB-Festplatte

VICTOR VPC, Mono, Kabel, **STAR NL 10** **sFr. 3390.–**
VICTOR VPC, Mono, Kabel, **STAR NX 15** **sFr. 3890.–**
VICTOR VPC, Color, Kabel, **STAR NL 10** **sFr. 3690.–**
VICTOR VPC, Color, Kabel, **STAR NX 15** **sFr. 4190.–**

DATALAND-Komplett-Pakete inklusive Druckerkabel und STAR NL 10 sowie Interface:

VICTOR VPC, Monochrom, komplett	sFr. 3390.–	VICTOR VPC, Farbe, komplett	sFr. 3690.–
ATARI 1040 STF, Mono, komplett	sFr. 2790.–	ATARI 1040 STF, ATARI-Color	sFr. 3190.–
ATARI 520 STM, Mono, komplett	sFr. 2490.–	ATARI 520 STM, ATARI-Color	sFr. 2690.–
BONDWELL 8, Handheld sowie DICONIX 150			sFr. 3990.–

DATALAND DRUCKER-Programm inkl. Druckerkabel:
CITIZEN (Preissenkung!), DICONIX, EPSON, FUJITSU, C. ITOH, NEC, OKI und STAR.
DATALAND-Drucker sind **immer günstiger** als der schweizerische Richtpreis. **Garantiert keine Grauimporte:** DATALAND vertritt die obigen Herstellermarken als authorisierter Fachhändler!

WO? DATALAND-Verkaufsdepots finden Sie in **neun Kantonen der Deutschweiz**
WAS? DATALAND verkauft nur Markenprodukte der offiziellen schweizerischen Importeure
WIE? inklusive **«DATALAND-3-Tage-Reparaturfrist-Garantie»,** Druckerkabel und 6 Monate Garantie auf Computer, 12 Monate auf alle Drucker, CITIZEN sogar 24 Monate!

Dataland GmbH
Postfach 212
CH-9100 Herisau 1

dataland
PRINTER & COMPUTER

Tel. 071 / 52 21 21

DATALAND-Verkaufsdepots finden Sie in neun Kantonen der Deutschschweiz!

```
      val_head  = 21;
      re_line   = 23;  {position of value information}
      im_line   = 24;

TYPE  pic_buffer = ARRAY [0..196] OF BYTE;
      position   = RECORD x,y : INTEGER END;

VAR   prev_pos  : position;   {prev. crosshair position}
      next_pos  : position;   {next crosshair position}
      picture   : pic_buffer; {save picture here, assuming 1 = 9}
      entry     : CHAR;       {movement or conf command}
      distance  : REAL;       {normalized node distance}
      index     : INTEGER;    {loop variable}
      p_dx,p_dy : INTEGER;    {positive crosshair extent}
      m_dx,m_dy : INTEGER;    {negative crosshair extent}

FUNCTION esc_sequence (VAR chr : CHAR) : BOOLEAN;

{Check for escape sequences}

 CONST ESC = #27; {ISO-Code for <Escape> }

 BEGIN
  read(KBD,chr); {read character}
  IF ((chr = ESC) AND keypressed)
   THEN {ESC-sequence encountered}
    BEGIN
     read(KBD,chr); {read identifier}
     esc_sequence := TRUE
    END
   ELSE {just a character}
    esc_sequence := FALSE
 END {esc_sequence};

FUNCTION clip (VAR m_dx,m_dy,p_dx,p_dy : INTEGER;
               pos : position) : BOOLEAN;

{Check whether crosshair could be drawn in its full extent,
 and return border values according to this check}

 BEGIN
  WITH pos DO
   BEGIN
    IF ((x > max_point) OR (y > max_point)
        OR (x < 0) OR (y < 0))
     THEN {at border}
      BEGIN
       sound(100);delay(100);nosound;
       clip := TRUE
      END
     ELSE {still in range}
      BEGIN
       IF (x - 1) >= 0 THEN m_dx := 1 ELSE m_dx := x;
       IF (y - 1) >= 0 THEN m_dy := 1 ELSE m_dy := y;
       IF (x + 1) <= max_point
        THEN p_dx := 1
        ELSE p_dx := max_point - x;
       IF (y + 1) <= max_point
        THEN p_dy := 1
        ELSE p_dy := max_point - y;
       clip := FALSE
      END
   END
 END {clip};
```



```
BEGIN
 BEGIN {init}
  gotoxy(22,header1);write('____________________');
  gotoxy(22,header2);write(' MANDELBROT - SET ');
  gotoxy(22,header3);write('____________________');
  gotoxy(22,key1);write('<F1>  : exit');
  gotoxy(22,key2);write('<F2>  : update pos');
  gotoxy(22,key3);write('<arr> : movement');
  gotoxy(1,val_head);write('Current Value :');
  gotoxy(22,pos_head);write('Current Position :');
  p_dx := 1; p_dy := 1; m_dx := 1; m_dy := 1;
  {full crosshair can be drawn because of center pos}
  WITH prev_pos DO
   BEGIN {save picture that will contain crosshair}
    x := x_start;y := y_start;
    getpic(picture,x - m_dx,y - m_dy,x + p_dx,y + p_dy)
   END;
  WITH next_pos DO

   BEGIN
    x := x_start;y := y_start
   END;
  distance := side / max_point;
  gotoxy(22,col_line);write('Col = ',next_pos.x:3);
  gotoxy(22,row_line);write('Row = ',next_pos.y:3);
  gotoxy(1,re_line);
  write('Re = ',origin.re + distance * next_pos.x);
  gotoxy(1,im_line);
  write('Im = ',origin.im - distance * next_pos.y)
 END;
 REPEAT
  WITH prev_pos DO
   BEGIN
    BEGIN {draw crosshair using color translation}
     colortable(3,0,0,0); {crosshair is black/yellow}
     draw(x - m_dx,y,x + p_dx,y, - 1);
     draw(x,y - m_dy,x,y + p_dy, - 1)
    END;
    BEGIN {save picture location to restore}
     x := x - m_dx; y := y + p_dy
    END;
    IF esc_sequence (entry)
     THEN {ESC-sequence encountered}
      CASE entry OF
       left_arr  : BEGIN
                     next_pos.x := next_pos.x - 1;
                     IF clip(m_dx,m_dy,p_dx,p_dy,next_pos)
                      THEN {reached border}
                       BEGIN {un-do movement}
                        next_pos.x := next_pos.x + 1
                       END
                    END;
       right_arr : BEGIN
                     next_pos.x := next_pos.x + 1;
                     IF clip(m_dx,m_dy,p_dx,p_dy,next_pos)
                      THEN {reached border}
                       BEGIN {un-do movement}
                        next_pos.x := next_pos.x - 1
                       END
                    END;
       up_arr    : BEGIN
                     next_pos.y := next_pos.y - 1;
                     IF clip(m_dx,m_dy,p_dx,p_dy,next_pos)
                      THEN {reached border}
```

SICOS®

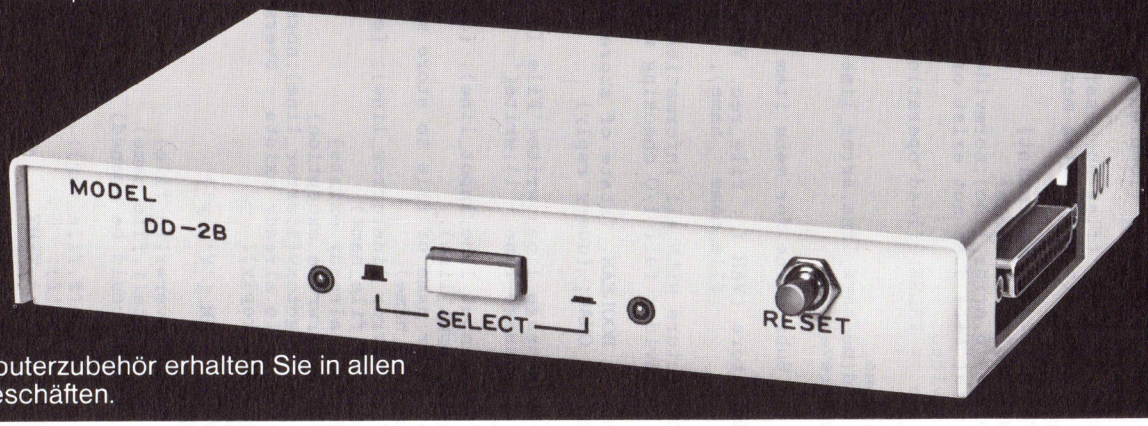
Auch rund um den Computer muss es stimmen. Darum hat SICOS eine Zubehör-Linie entwickelt, die sich auszeichnet durch formschönes Styling, durch ausgeklügelte Details und attraktive Preise.

Schön, praktisch und preiswert:

Computerzubehör von SICOS

SICOS-Switch-Box *Die SICOS-Switch-Box ermöglicht das kostengünstige Zusammenschalten mehrerer Hardware-Komponenten.*

Modell DD1A für 1 Computer und 2 Drucker parallel
Modell DD1B für 2 Computer und 1 Drucker parallel
Modell DD1C für 1 Computer und 2 Drucker seriell
Modell DD2A für 1 Computer und 2 Drucker parallel, 32-kB-Speicher
Modell DD2B für 2 Computer und 1 Drucker parallel, 32-kB-Speicher



SICOS-Computerzubehör erhalten Sie in allen guten Fachgeschäften.

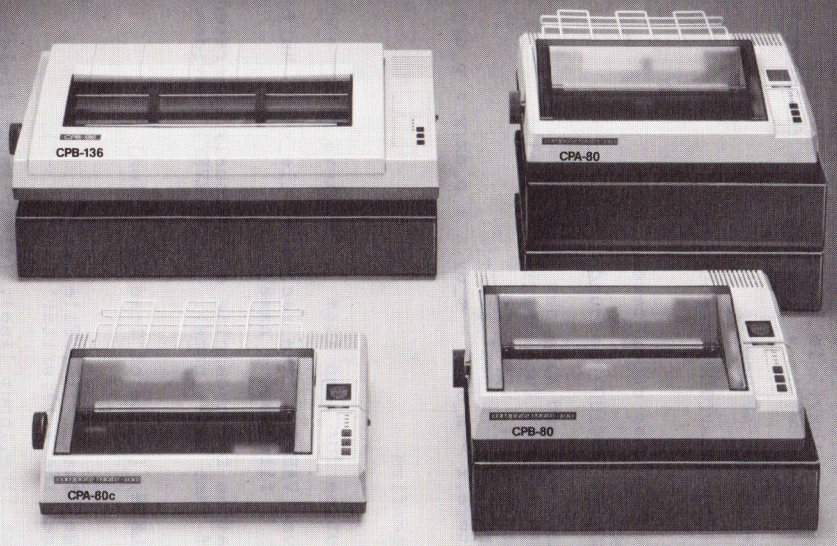
Generalvertretung für die Schweiz:
EXCOM AG, Einsiedlerstrasse 31, 8820 Wädenswil,
Telefon 01/780 74 14

Erhältlich im
EPSON☐INFORMATION☐CENTER am Stauffacher
Stauffacherstrasse 35, 8004 Zürich

ROTRONIC COMPUTER PERIPHERIE

ROTRONIC COMPUTER PERIPHERIE

Damit sich Ihr Computer gut ausdrucken kann!



- **Überlegene Druckqualität**
- **Hochauflösende Grafik**
- **Einmalige Verwendung von Filmfarbband**
- **verstellbarer Formtraktor**

CPA-80
Der Universaldrucker für alle PC's und Terminals

CPB-80 + CPB-136
speziell auf die Bedürfnisse der IBM-PC Anwender abgestimmt

CPA-80C
speziell für Commodore C64 entwickelt

ROTRONIC COMPUTER PERIPHERIE

rotronic ag

8040 Zürich · Badenerstrasse 435
Telefon 01-492 32 11 · Telex 822530

```
                    BEGIN {un-do movement}
                     next_pos.y := next_pos.y + 1
                    END
                  END;
       down_arr   : BEGIN
                     next_pos.y := next_pos.y + 1;
                     IF clip(m_dx,m_dy,p_dx,p_dy,next_pos)
                      THEN {reached border}
                       BEGIN {un-do movement}
                        next_pos.y := next_pos.y - 1
                       END
                     END;
       confirm    : BEGIN {nothing to do}
                     END;
       update     : BEGIN {update position information}
                     gotoxy(22,col_line);
                     write('Col = ',next_pos.x:3);
                     gotoxy(22,row_line);
                     write('Row = ',next_pos.y:3);
                     gotoxy(1,re_line);
                     write('Re = ',origin.re + distance * next_pos.x);
                     gotoxy(1,im_line);
                     write('Im = ',origin.im - distance * next_pos.y)
                     END
       ELSE {ilegal ESC-sequence}
        BEGIN
         sound(440);delay(100);nosound
        END
      END
     ELSE {any other char encountered}
      BEGIN {beep}
       sound(440);delay(100);nosound
      END;
    BEGIN {restore old and save new picture}
     colortable(0,1,2,3); {use original color}
     putpic(picture,x,y); {restore}
     x := next_pos.x; y := next_pos.y;
     getpic(picture,x - m_dx,y - m_dy,x + p_dx,y + p_dy)
    END
   END
  UNTIL entry = confirm
 END {examine};

{--------- Subroutine for menu item  - Retrieve Map -----------------}

PROCEDURE retrieve (VAR f : file_rec; VAR file_var : buffer;
                    file_name : name);

{Retrieve information for examining a previously generated picture
 from file <f> of name <file_name>.   RETRIEVE() provides full I/O
 checking}

 VAR  done : BOOLEAN; {state of load operation}

 BEGIN
  gotoxy(5,message_line);write('File Input');
  gotoxy(5,message_line + 1);write('----------');
  REPEAT
   clr_field(input_line,input_line); {clear input field}
   gotoxy(5,input_line);
   write('Enter name of file to be loaded : ');
   readln(file_name);
   clr_field(error_line,error_line); {clear error field}
   IF exist (f,file_name)
    THEN {file already exists}
     BEGIN {load map}
      read(f,file_var); {load map}
      close(f);
      done := TRUE; {flag successful loading}
      map_exists := TRUE {flag map being existent}
     END
    ELSE {file does not exist}
     BEGIN {error message}
      gotoxy(5,error_line);normvideo;
      write('File does not exist or any other error - Try again !');
      lowvideo;
      done := FALSE; {load operation failed}
     END
  UNTIL done;
  clr_field(message_line,error_line)
 END {retrieve};

{--------- Subroutine for menu item  - Store Map -------------------}

PROCEDURE store (VAR f : file_rec; VAR file_var : buffer;
                 file_name : name);

{Store a picture with all information in file <f> of name <file_name>.
 STORE() provides full I/O checking with overwrite protection}

 VAR done  : BOOLEAN; {state of store operation}
     reply : CHAR; {query reply}
BEGIN
 gotoxy(5,message_line);write('File Input');
 gotoxy(5,message_line + 1);write('----------');
 REPEAT
  clr_field(input_line,input_line); {clear input field}
  gotoxy(5,input_line);
  write('Enter name of file to store map : ');
  readln(file_name);
  clr_field(error_line,error_line); {clear error field}
  IF exist (f,file_name)
   THEN {file already exists}
    BEGIN {overwrite exception}
     close(f);gotoxy(5,error_line);normvideo;
     write('File already exists - overwrite <y/n>');
     read(KBD,reply);
     lowvideo;
     IF reply IN ['Y','y']
      THEN
       BEGIN {overwrite file}
        IF create(f,file_name)
         THEN {could be opened}
          BEGIN
           write(f,file_var); {store map onto file}
           close(f);
           done := TRUE
          END
         ELSE {opening failed}
          BEGIN
           clr_field(error_line,error_line);
           gotoxy(5,error_line);normvideo;
           write('File creation error - Try again !');
           done := FALSE;
           lowvideo
          END
       END
      ELSE
```

```
        BEGIN {do not overwrite}
         done := FALSE
        END
      END
     ELSE {file does not exist, assuming no error}
      BEGIN {create file}
       close(f);
       IF create(f,file_name)
        THEN {could be opened}
         BEGIN
          write(f,file_var); {store map onto file}
          close(f);
          done := TRUE
         END
        ELSE {opening failed}
         BEGIN
          clr_field(error_line,error_line);
          gotoxy(5,error_line);normvideo;
          write('File creation error - Try again !');
          done := FALSE;
          lowvideo
         END
      END
   UNTIL done;
   clr_field(message_line,error_line)
 END {store};

{--------- MAIN ----------------------------------------------------}

BEGIN
 clrscr;
 valid := ['G','R','E','S','Q','g','r','e','s','q'];
 quit := ['Q','q'];
 map_exists := FALSE;
 REPEAT
  BEGIN {display menu}
   header;
   clr_field(item1_line,input_line);
   gotoxy(5,item1_line);
   write('Generate Map ..........<G>');
   gotoxy(5,item2_line);
   write('Retrieve Map ..........<R>');
   gotoxy(5,item3_line);
   write('Examine Map ...........<E>');
   gotoxy(5,item4_line);
   write('Store Map .............<S>');
   gotoxy(5,exit_line);
   write('Exit ..................<Q>');
  END;
  REPEAT
   gotoxy(5,input_line);
   write('Select item : ');read(KBD,command);
   clr_field(error_line,error_line); {clear error field}
   IF NOT (command IN valid)
    THEN {invalid entry}
     BEGIN
      gotoxy(5,error_line);normvideo;
      writeln('Invalid Entry - Try again !');
      lowvideo
     END
  UNTIL command IN valid;
  IF NOT (command IN quit)
   THEN {there is somthing to do}
    BEGIN
     clr_field(item1_line,error_line); {clear input field}
     CASE command OF {execute selected operation}
      'G','g' : BEGIN {generate map}
                 WITH proc_rec DO
                  FOR select := re_origin TO max_step DO
                   get_parameter(select,origin,side,step);
                 clr_field(item1_line,error_line+1);
                 {clear input field}
                 WITH proc_rec DO
                  mandelbrot(map,origin,side,step);
                 map_exists := TRUE; {flag map being existent}
                END;
      'R','r' : BEGIN {retrieve map}
                 retrieve(data,proc_rec,file_name)
                END;
      'E','e' : BEGIN {examine map}
                 IF map_exists
                  THEN
                   BEGIN
                    WITH proc_rec DO
                     BEGIN
                      gotoxy(5,message_line);
                      write('Please wait a moment');
                      assign_color(map,step);
                      examine(map,origin,side)
                     END;
                    BEGIN {return}
                     WHILE NOT keypressed DO
                      BEGIN
                       gotoxy(22,return_line);
                       write('Hit any key');
                       delay(400);gotoxy(22,return_line);
                       write('            ');
                       delay(400)
                      END;
                     read(KBD,reply);
                     textmode
                    END
                   END
                  ELSE
                   BEGIN {there is nothing to see}
                    gotoxy(5,error_line);normvideo;
                    write('Operation aborted - Nothing to examine');
                    lowvideo;
                    command := 'x' {force repetition}
                   END
                END;
      'S','s' : BEGIN {store map}
                 IF map_exists
                  THEN store(data,proc_rec,file_name)
                  ELSE
                   BEGIN {there is nothing to store}
                    gotoxy(5,error_line);normvideo;
                    write('Operation aborted - Nothing to store');
                    lowvideo;
                    command := 'x' {force repetition}
                   END
                END
      ELSE {must be the exit command}
       clrscr
     END
    END
  UNTIL command IN quit
END {mandelbrot_explorer}.
```

COMPUTER-BÖRSE

Zu verkaufen

Apricot mit zwei Floppies, 256 RAM, TextProgramm + SuperCalc + Basic + Basiccompiler + Fibu + Drucker, Fr. 2'990.-. MP Software, Baselstr. 4, 4132 Muttenz, ☎ 061/61'91'83 oder 61'49'79

Zu HP-71B ein **HP-41 Translation/Forth-ROM** zu verkaufen. Inklusive Textfile-Editor VP Fr. 275.- (NP Fr. 356.-). ☎ 031/41'56'74

Original **IBM-Programme**, Writing, Filing, Reporting, Assistant, alle auf 5 1/4 Zoll-Disketten inkl. Manual. Wegen Nichtgebrauchs sehr günstig. ☎ 054/22'25'45

BERATUNG UND UNTERSTUETZUNG
NEUE PC's ALLER MARKEN
DRUCKER UND PERIPHERIE
GEBRAUCHTE COMPUTER
STANDARD SOFTWARE
EINTAUSCHOFFERTEN
COMPUTER MARKET
COMACON AG
MEINRAD-LIENERT-STRASSE 15
(BEIM LOCHERGUT) 8003 ZUERICH 01/462 19 57
DONNERSTAG 17⁰⁰-21⁰⁰ / SAMSTAG 10⁰⁰-16⁰⁰

IBM PPC-Portable, 640 KB, zwei Slimline zu 360 KB, DIN-Tastatur, Centronics-Adapter, DOS 2.1 mit BASICA, Turbo-Pascal, 15 Monate alt, IBM geprüft. VP Fr. 4'200.-. ☎ 033/43'17'63 abends

Epson-Printer RX-100 und RX-80 mit Centronics-Schnittstelle. Preis günstig. ☎ 01/432'56'22 nach 18 Uhr

Tektronix Inkjet, neuwertig, inkl. diversen Farbpatronen, Papier und Kabel. Wegen Kauf eines Calcomp Colormasters günstig abzugeben. VP Fr. 650.- (NP Fr. 4'600.-). ☎ 052/22'52'92

Stocker «Soft»

Tiefstpreise – Beratung – Qualität
IBM- und APPLE-kompatible Computer, Erweiterungskarten und Zubehör
10/20 MB Winchester (IBM/APPLE)
Disketten für alle Computermodelle ab 10 Stück/Mengenrabatt
Programme nach Mass – direkt vom Hersteller (APPLE/IBM)

Stocker «Soft» **(01) 935 43 83**
8614 Bertschikon Im Müselacher 22

21-MB-Hardcard, die einzige mit der 2-Jahresgarantie. Top-Qualität, z.B. 65ms/30'000MTBF, Einführungspreis nur Fr. 1'280.-. Händler Spezial-Preis. Free-Soft, Steinstrasse 62, 8106 Adlikon.

Toshiba 1100 portable IBM-komp. Schnittstelle RS232, Mono oder Farb-Monitor, Drucker, ext. Floppy, inkl. diverser Software und BStamm-Kommunikations-PGM, günstig, ☎ G: 085/6'15'27

JUKI 5520 Farb-Matrix-Drucker. Neu ☎ 071/22'82'63

HP-71B mit Mathe-Modul, 1 Jahr alt, Preis Fr. 1'000.-. Originalsoftware für Rainbow 100 für CP/M-86, MBasic-86, Fr. 50.-. Mark-Wiliams «C» Compiler, Fr. 300.-. ☎ 053/4'38'12 abends

Hercules-kompatible Grafikkarte, Fr. 400.-. ☎ 065/55'21'43

Fujitsu 16S, wenig gebraucht, 1 Jahr, mit Monochrom-Bildschirm, Keyboard, Drucker, Fujitsu MB27406D. Auskunft: R. Dunkel, ☎ G: 061/81'27'61, P: 061/94'18'83

Epson-Geneva (PX8), 60 KRAM, P-40, 1 MBSW Fr. 1'750.-. Bondwell 2, 512 K RAM Fr. 1'800.-. Plotter Epson HI-80, Fr. 850.-. Siemens Computer CCP/M, MS-DOS, 640 K, zwei Laufwerke, zwei serielle, zwei Centronics, Fr. 4'000.-. ☎ 072/69'24'63 nach 18 Uhr

SATREF-Basic-Programmhilfe, getestet in M+K 86-4 für IBM-kompatible PCs. Für GWBasic und BASICA, Fr. 210.-. Satusoft, ☎ 01/312'04'11 (Mo-Fr 8.00 - 12.00 und 13.00 - 17.00 Uhr)

Apricot portable, 512 KB RAM, Floppy 720 KB, MS-DOS 2.11, Infrarot-Maus und Keyboard, Sprachsteuerung, MS-Cobol, VB Fr. 3'000.-. ☎ G: 01/488'24'96

SRS Micro Computer Products
Box 1207 8213 NEUNKIRCH Tel. 053/6 25 93
TULIP PC-Compact 60% schneller, 100% IBM kompatibel aus europäischer Fertigung 512K RAM Preis SUPER!! INFO anf.!
MSX, der Standard im Homecomputersektor wählen S I E den Standard der Zukunft. Wählen Sie M S X ! INFO anfordern!
SHARP Pocketcomputer der Standard unter den Pocket's! Katalog anfordern !

Commodore PC-10 komplett, Mono-/Farb-Karte, Game-Adapter, DOS, Turbo-Pascal, Windows, PFS-File + Report, alles original und viele Programme. Neu, Fr. 5'600.-, jetzt nur Fr. 2'400.-. ☎ 041/57'47'33 mittags

Apple II+ (komp.) Computer, zwei Drives, Monitor, sieben Karten, achtzig Disketten mit Programmen und Bücher, ca. Fr. 3'500.-, auch Einzelkauf möglich. ☎ 01/491'40'71 abends

Commodore-Geräte: PC 710 mit HRG 512x512 Punkte, Floppy 4040, Drucker 8023P inkl. Handbücher und Kabel, wenig gebraucht, VP Fr. 2'100.-. ☎ 061/63'14'36

Neuwertiger Typenraddrucker **Quen-Data dwo 1120**, Schweizer Zeichensatz, Einzelblatteinzug, als Peripherie zu IBM-PC, Commodore usw. Fr. 600.-. R. Faess, Zürich, ☎ 01/252'83'16

Epson FX-85, 160 Z/s, IBM- und ESC/P-kompatibel, NLQ, Fr. 1'000.-. M. Scheuner, 3400 Burgdorf, ☎ 034/22'46'31

HP-75C, tragbarer Basic-Computer mit Kassettendrive 82161A, Handbüchern und diversen Programmen günstig zu verkaufen. ☎ 071/87'19'49 abends

SPARANGEBOT !!!
IBM kompatible
COMPUTER
PC/XT & AT
Komplette Anlagen
ab Fr. 1990.--
ADD/ON Cards
Monitoren usw.
12 Monate Garantie
Tel:057/34'26'56
Montag - Samstag

ACHTUNG !!!
An alle PC Besitzer.
SUPERANGEBOT !!!
Kein Risiko !!!
Harddisk - Kit
20MB mit Kontroller
12 Monate Garantie
GRATIS Einbau
SPARPREIS Fr.1200.-
Tel:057/34'26'56
9.00 - 22.00

HP87, 412 KB RAM, I/O und Plotter ROM, 5 1/4 Zoll Doppelfloppy HP82901M, Matrixdrucker HP82905B, sechs Farben A3/A4-Plotter HP7475A, HPIB-Interface. Auch einzeln, günstig. ☎ 01/720'78'94

Ab Mitte Januar: **Commodore 8032**, Drucker 4022, Floppy 8050, Drucker-Puffer 64 K inkl. diverser Software und Handbücher, VP Fr. 3'000.-. ☎ P: 042/77'15'73, G: 042/21'20'53

Sharp MZ-80B, 64 KB (inkl. CRT, Kassetten-Deck, Erweiterungs-Rahmen), Basic-Interpreter, Spiele, diverse deutsche Handbücher, neuwertiger Zustand, Fr. 480.-. ☎ G: 01/258'16'57, P: 01/463'90'73

IBM-PC/XT, komplett mit Farbmonitor, 10 MB Harddisk, Multifunct, 512 MB, ser. +par. Ports, Clock, Maus und viel Software (WordStar, Framwork, dBase III ...), ev. mit Drucker, Preis sehr günstig. ☎ 041/61'57'52

Hardcard Plus, 10 Mega Harddisk auf einer Karte für IBM-PC und Kompatible, Fr. 1'000.-. Epson HX-20 mit 32 K-intern und Kassettenleser, Fr. 1'000.-. Beide wenig gebraucht. ☎ 022/55'25'64 ab 19 Uhr

Kalkumat PC (Tabellenkalkulationsprogramm mit Grafik für IBM-PC und Kompatible) mit allen Unterlagen und Garantie, Fr. 250.- (NP Fr. 500.-). Testbericht anfordern. ☎ 01/780'13'33 H. Schärer

PC-Spiele zum Denken Spielsammlung mit Serata, Mühle, Backgammon und Versi, lauffähig unter MS-DOS 2.11. Senden Sie frank. Retourcouvert und Fr. 30.- an R. Suter, Meisenweg 1, 3634 Thierachern

Epson PX-8 Word- und CalcStar d, Printer PX-80, Kabel, Tech. + Lehrbuch, neu Fr. 4'330.- jetzt Fr. 2'500.-, ungebraucht. ☎ G: 038/51'21'61, P: 038/51'26'46

Da wir und Sie fest davon überzeugt sind, dass das **WordPerfect 4.1** das beste Textverarbeitungsprogramm der Welt ist, gilt nachfolgendes Super-Weihnachts-Angebot:

Sie senden uns Ihr altes MS/DOS-Textprogramm, gleich welcher Marke und erhalten dafür die professionelle Version von

WordPerfect 4.1

zum 1/2 Preis von Fr. 740.– statt Fr. 1480.– inklusive Wust

(Für Schulen gelten spezielle Konditionen. Bitte fragen Sie uns.)

Software-Post

Autorisierte WordPerfect-Vertretung

Solothurnstr. 12, 2540 Grenchen, Tel. 065/53022

Texterfassung auf Ihrem PC – Satzproduktion auf der UD-Lichtsatzanlage

Einmalige Texterfassung spart Satzkosten, verhindert Übertragungsfehler. Alle Modifikationen wie Preis-, Text- oder Aufbau-Änderungen, zum Beispiel in Periodika, können problemlos wieder auf Ihrem PC vorgenommen werden. Sie erhalten die Texte wahlweise als Papierspalten oder als Film. Beides können Sie auf Wunsch selber umbrechen, montieren und maquettieren. Auch für diese Zeitschrift wird der Satz und nachher der Druck in dieser kostengünstigen Art hergestellt.

Setzen Sie jetzt Ihre Texte selber

Damit können Sie die Leistungspalette Ihres Computers um eine kostensenkende Möglichkeit erweitern.

Verlangen Sie unseren ausführlichen Prospekt oder lassen Sie sich von unseren Spezialisten in die kostensenkende Satzproduktion einführen.

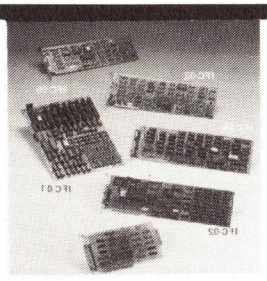
Tel. 041/44 24 44

Unionsdruckerei Luzern
6005 Luzern, Kellerstrasse 6

UD

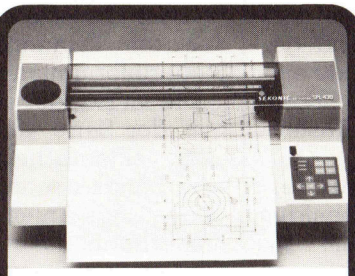
SUNNY - 1600 PC, XT, AT
– 100% IBM kompatibel, 8 slot
– XT-Turbo Board, 640 k RAM
– 8088, 4,77/8 MHz, 20 M Harddisk
– AT, 80286 CPU, (80287 Co-proc.)

ab Fr. 1600.–



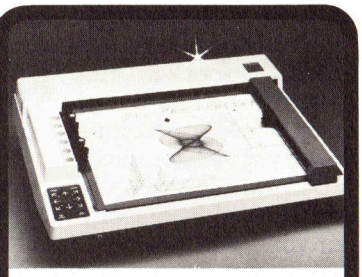
Main Board + Add on Cards
– Multifunktionskarten, Hercules V.II,
– PC-Apple-Transfer, A/D-D/A-Card
– PC-Mouse, Tablets, Light pen
– Key boards, Gehäuse, Power supp'lies

Low Cost



AUTOCAD/X-Y Plotter, DIN A 3
– extrem schnell + leise
– HP-GL-kompatibel
– RS 232 C + Centronics, 6 Farben
– hochgenaues Plotten in Steps von 0,025 mm

Fr. 2990.–



Intelligenter DIN A3-Plotter
6 Farben, extrem robust + zuverlässig an alle PC's anschliessbar Centronics, RS 232 C, HP-GL unterstützt Auto Cad, Lotus 1-2-3, Symphonie usw.

ab Fr. 1800.–

SILTRON AUTOMATION

Oberlandstrasse 297
5444 Künten

057 / 34 26 56

BACKUP-Peripherie-Geräte mit bis zu 40 MB Kapazität

Irwin Magnetics hat mit den Auslieferungen neuer Bandlaufwerke der «Serie 400» begonnen. Diese Peripherie-Geräte wurden als BACKUP-Systeme für IBM-PC/AT/XT und kompatible Personal Computer entwickelt, die mit Magnetplatten als Massenspeicher arbeiten. Mit ihren Speicherkapazitäten von 10, 20 und 40 MB entsprechen die neuen «Serie 400» Bandlaufwerke den Leistungen heutiger PC-Magnetplatten.

Neben dieser Leistungsfähigkeit ist die 'Miniaturisierung' das hervorstehende Merkmal der neuen Irwin Serie 400. Das neue Peripheriegerät ist nur 5,1 cm breit und 18,4 cm tief, die Stellfläche auf dem Schreibtisch ist etwa sechsmal kleiner als bei vergleichbaren Systemen auf dem Markt.

Irwin setzt auf allen BACKUP-Bandlaufwerken ein patentiertes Servokopf-Positionierungs- und das EC / Tape-Fehlerkorrektur-Verfahren ein. Mit der Servokopf-Positionierung wird die angesteuerte Spur auf dem Band elektronisch gesucht und gefunden. Dabei wird eine Methode eingesetzt, wie sie in ähnlicher Form auch bei Magnetplatten zur Ansteuerung verwendet wird.

Das EC/Tape-Fehlerkorrektur-Verfahren wird in dieser Form ebenfalls nur von Irwin eingesetzt. Mit ihm können sogar Daten wiedergewonnen werden, die sich auf beschädigten Teilen eines Bandes befanden.

Für die Irwin Serie 400 stehen die allgemein gebräuchlichen 3M DC-Bänder (DC1000 und DC2000) in Mikrokassetten zur Verfügung. In ihren Aussenmassen nicht grösser als ein Kartenspiel, eignet sich die Serie 400 ideal zur Ergänzung bestehender PC-Magnetplatten-Installationen.

Die Sicherheit und Kompatibilität der Irwin BACKUP-Systeme geht so weit, dass auf einem Gerät mit hoher Kapazität auch Mikro-Datenkassetten verarbeitet werden können, die auf Irwin Bandlaufwerken mit niedrigerer Kapazität erstellt wurden. Irwin hat dazu ein Verfahren entwickelt, mit dem automatisch beim Arbeitsbeginn der Bandtypus überprüft wird. Der Schreib-/Lesekopf wird dabei nach der am Bandanfang enthaltenen Steuerinformation ausgerichtet.

Die im Handel erhältlichen Versionen der Serie 400 umfassen neben dem eigentlichen Peripheriegerät eine Bandkassette, eine ausführliche Bedienungs-Anleitung sowie eine bedienerfreundliche Transfer-Software. Für Personal Computer, deren eigene Stromversorgung nicht ausreicht, wird eine unabhängige Stromversorgung für das Bandlaufwerk angeboten. Ausserdem stellt Irwin eine ganze Reihe von Verbindungskabeln zum Anschluss an verschiedene Personal Computer-Modelle zur Verfügung.

Der Anwender wird von der Irwin Software «per Menü» durch den BACKUP-Vorgang eingeführt. Die Software erlaubt, entweder den gesamten Platteninhalt oder auch nur ausgewählte Files zu kopieren. Bei der File-by-file-Datensicherung können entweder nur bestimmte Verzeichnisse, ausgewählte File-Gruppen, oder auch nur die Files gesichert werden, die seit der letzten Datensicherung geändert wurden. Die Irwin Software ist kompatibel mit PC-DOS und MS-DOS (Version 2.0 und höher). Info: CPA Computer Peripherie AG, Täfernstrasse 11, 5405 Baden-Dättwil, Tel. 056/84'01'51. □

30 MB Speichererweiterung für IBM-Kompatible

Verbatim, eine Tochtergesellschaft der Kodak, macht die integrierte Speichererweiterung um 30 MB für IBM-kompatible PCs möglich. Auf einer Zusatzkarte, die einfach in die Erweiterungs-Slots des PCs eingebaut werden kann, ist eine Harddisk mit Controller integriert. Die 3,5 Zoll Harddisk weist eine Kapazität von 20 oder 30 MBytes auf. Bestehende Laufwerke für Floppies oder eine schon vorhandene Harddisk werden von der Erweiterung nicht betroffen.

Die Harddisk ist primär für die IBM-Modelle XT und AT ausgelegt, eignet sich aber auch für den Einbau in alle kompatiblen Geräte. Sie ist zudem für lokale Netzwerke (LAN) wie die SCOM Ether-Serie einsetzbar und kann für die Betriebssysteme DOS 2.1 und höhere Formen sowie PC IX, UNIX und Xenix verwendet werden. Die enorme, integrierte Speichererweiterung dient insbesondere für Datenbanken und die Textverarbeitung.

Die Harddisk wird mit einer umfassenden Garantie von Verbatim geliefert. Innerhalb der ersten sechs Monate erfolgt der sofortige Ersatz, es schliesst sich für weitere sechs Monate eine Garantie für Ersatzteile und Arbeitsaufwand an. Die Zusatzkarte weist eine Dimension von 12,7 cm auf 35,5 cm auf und hat ein Gewicht von 1,04 kg. Info: Verbatim S.A., Postfach 3, 1211 Genève, Tel. 022/98'74'44. □

Hochleistungs-PC/AT: Der EPSON PC AX

Ein neuer Standard? Zumindest neue Massstäbe setzt der neue EPSON PC AX: Kraftvoller Inhalt in kompaktem Design. Herzstück bildet ein Intel-80286-Prozessor, welcher mittels leicht zugänglichem Schalter auf der Frontplatte zwischen den Taktfrequenzen 6/8/10 MHz wählen lässt.

Fünf halbhohe Slimline-Einschübe, die zwei internen mit einem 40 MB Harddisk bestückt, einer der drei externen mit einem 1.2 MB Laufwerk ausgestattet, lassen beliebige zusätzliche Einbaumöglichkeiten offen. Ausgehend von 640 KB RAM Grundausstattung lässt sich das Memory auf 15.5 MB ausbauen. Von neun verfügbaren Slots (sechs 16 Bit, drei 8 Bit) sind deren zwei belegt: Der eine für den Controller zweier Harddisks und eines Floppy Drives, der andere für die AT-Multifunktionskarte (RAM-/seriell/parallel-Karte).

Der Zugang zu dieser imponierenden Leistung erfolgt über eine neue AT-Tastatur bestehend aus 102 Tasten (davon 12 Funktionstasten) und mit drei LED's versehen. Die separate Anordnung von Cursor- und Zahlenblock versteht sich von selbst. Im Lieferumfang ist das EPSON MS-DOS 3.2 und eine Diagnostik-Diskette inbegriffen. Sämtliche für den AT-Standard verfügbaren Software-Anwendungen sind lauffähig. Bei der Arbeit fällt die geringe Harddisk-Zugriffszeit auf, welche je nach Konfiguration zwischen 25 bis 38 ms beträgt. Die Floppy-Laufwerke können sowohl das 1.2 MB- wie das 360 KB-Format schreiben und lesen. Info: Excom AG, Einsiedlerstrasse 31, 8820 Wädenswil, Tel. 01/780'74'14. □

Aus einem IBM-PC/XT wird ein AT

Die Mini-AT Karte mit 80286-8 Prozessor ermöglicht Ihren PC oder XT in einen AT umzuwandeln, dies mit minimalen Kosten und unter Beibehaltung von bestehenden Peripheriegeräten wie Disketten oder Disklaufwerke. Ein umgebauter PC oder AT ist somit vergleichbar mit dem neuen IBM X286 Modell.

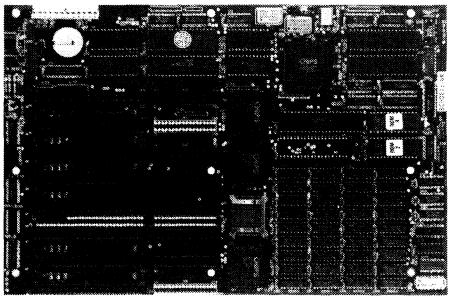
Die meisten Original-PC oder XT Controller und Interfacekarten können mit der Mini-AT-Karte verwendet werden, mit Ausnahme von Zusatzkarten, welche Systemspeicher enthalten: z.B. Memorykarten oder Multifunktionskarten.

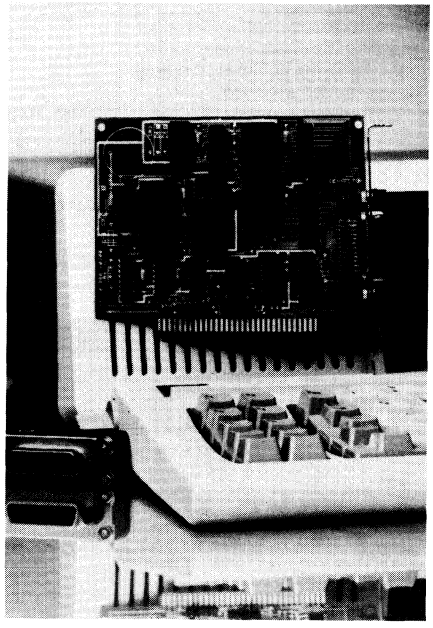
Die Mini-AT-Karte hat das gleiche Format wie das XT Motherboard und ist voll AT kompatibel. Sie funktioniert mit Taktfrequenzen von 6, 8 oder 10 MHz. Die Mini-AT-Karte unterstützt MS-DOS 3.1 und XENIX 286.

Die Harte hat standardmässig 512 KB Memory (erweiterbar auf 1 MB), 6 AT-Slots und 2 XT Slots sowie einen Sockel für einen 80287-8 Coprozessor.

Ein Test mit dem Programm SI von Norton Utilities zeigte, dass ein PC oder XT ausgerüstet mit einer Mini-AT-Karte bei 8 MHz die 7-fache und bei 10 MHz die 9-fache Geschwindigkeit von einem PC/XT erreicht.

Der Einbau ist dank den identischen Montagelöcher und Steckerpositionen mit der Original-PC/XT-Karte auch durch Nichttechniker realisierbar. Info: Marli Distribution S.A., Chemin Taverney 3, 1218 Genève, Tel. 022/98'55'66. □





Machen Sie aus Ihrem PC einen leistungsfähigen IEEE-488-Bus-Kontroller!

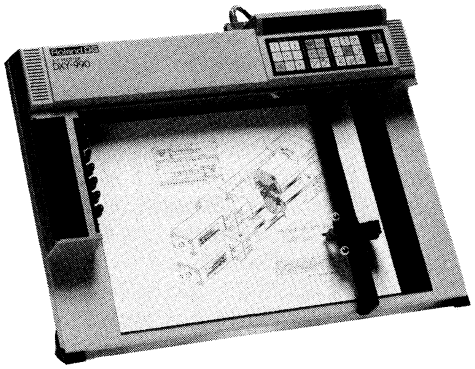
EPROM-residente Treiber-Software ermöglicht einfachste Programmierung in allen gängigen Programmiersprachen.
Kein Laden von Disketten-Software notwendig.
Einfachste Befehlsstruktur, eigene Kommandos sind möglich.
Sehr hohe Übertragungsgeschwindigkeit, Blockgrösse bis 64 kB.
Benötigt keinen Speicherplatz im Arbeitsspeicher Ihres Rechners.
SRQ-Interrupts auch in BASIC möglich (Option).
Zurzeit werden folgende Programmiersprachen ohne Aufpreis unterstützt: BASIC, BASIC-Compiler, MS/IBM PASCAL, TURBO-PASCAL, IBM/MS FORTRAN, Professional FORTRAN, Lattica - C, MS/IBM C.
Die mitgelieferte Diskette enthält alle nötigen Treiber und viele Programmbeispiele.
Fordern Sie ein PC-488 zur unverbindlichen Prüfung an!

KEITHLEY
Keithley Instruments S.A.

Kriesbachstrasse 4, 8600 Dübendorf
Telefon 01 / 821 94 44
Telex 57 536

Roland DG

Roland XY-Plotter die sorgfältigen Zeichner



Jetzt in 6 verschiedenen Modellen lieferbar.

- Zeichnet echt Europaformate A-3 (416mm × 279mm)
- Federverschluss-Automatik
- Voll HP-GL-kompatibel

Leistungen zu unvergleichlichen Preisen! Ab Fr. 2660.–

Ihr Partner für Computer-Grafik Peripherie!

PCA

Polygraph Computer AG

Geissraistrasse 30 CH-5452 Oberrohrdorf AG Tel. 056/96 47 48

Unsere Devise: Soviel Computer wie möglich für so wenig Geld wie nötig!

PC-AT 03: 640 KB Ram, 1,2 MB Laufwerk, DIN oder VSM Tastatur, 22 MB Festplatte, DOS 3.1, Utility Software, 6/8 Mhz. opt. 10 Mhz. **Fr. 5950.–***

PC-XT: 640 KB Ram, 4,77/8 Mhz, 2×360 KB Laufwerk, DIN oder VSM Tastatur, DOS 2.11, Multi I/O mit Uhr, Kalender, Game-, Parallel-, Seriell-Port, Utility Software usw. **Fr. 2590.–***

PC-XT 20: Wie PC-XT jedoch mit 20 MB Harddisk **Fr. 3890.–***

*** inkl. 14"-TTL Monitor auf dreh- und schwenkbarem Ständer und einem Hercules-kompatiblen Graphikadapter. (720×348 P.)**

Alle Geräte in der Schweiz zusammengestellt und getestet. 100% kompatibel zum Industriestandard IBM.

Festplatten:	10 MB Slimline	Fr. 920.–
	20 MB Slimline	Fr. 1120.–
Festplatten-kontroller:	max. 2 Platten,-8 Mhz	Fr. 330.–
Maus:	(MS-Kompatibel) mit Treibersoftware	Fr. 235.–

Allgemeines:
- Kurze Lieferfristen
- Alle Preise inkl. Wust.
- 12 Monate Garantie auf Geräte und Zubehör

Niederhauser & Co., Gaswerkstrasse 33, 4900 Langenthal, Telefon 063 22 27 88 (auch samstags)

COMPUTERSCHULE

Chance für (Radio-/TV-)Verkäufer mit guten Kenntnissen der Mikrocomputerwelt:

Computer verkaufen

Beratung und Verkauf von Hobby- und Home-Computern, Peripherie, Zubehör ...

Administration, Lager, Werbemassnahmen sind Aufgaben, die in unserem dynamisch wachsenden Computer Center täglich anfallen.

Zur Bewältigung dieser Aufgaben suchen wir einen weiteren aufgestellten Mitarbeiter mit:

- kaufm. oder Verkaufsausbildung
- Verhandlungsgeschick
- Englisch- und Programmierkenntnissen
- Vertrautheit in der Commodore- und Atari-Szene
- einem Alter zwischen 25 und 35 Jahren

Für ein persönliches Gespräch wenden Sie sich bitte an

SYSAG

HOLEESTRASSE 87 · 4054 BASEL · TELEFON 061 39 25 25

COMPUTERSCHULE

EcoSoft
Economy Software

6981 Astano, Tel. 091 / 73 28 13

Frei-Programm- und Shareware-Zentrale
Über 25 000 Programme für IBM-PC/Kompat., Apple II Macintosh, Atari ST, Commodore C 64/C 128, Amiga. Programme für Beruf, Geschäft, Heim und Schule.

Zum Kennenlernen guter Frei-Programme:

10 beliebte Programme für Fr. 10.–

Dazu **gratis:**

- Katalog über Frei-Programme (Public Domain) und professionelle Shareware auf Diskette(n) einschl. Sachgebiets-Verzeichnis im Wert von Fr. 10.–.

Dieses Kennenlern-Angebot erhalten Sie gegen Einsendung oder Angabe dieses Inserates und von Fr. 10.– (bar oder Scheck). Bitte unbedingt Computermodell angeben.

Manche brauchen

Der PCA von Tandon ist ein vollkompatibler AT

ein Weltunternehmen auf

--- und kostet mit 1,2 MB-Floppy-Laufwerk nur 5.995 Franken. --- Mit

dem Schreibtisch. Andere

20-MB-Festplatte kostet der PCA nur 500 Franken mehr. ----------

nur einen leistungs -

Der PCA 30 kostet 7.995 Franken und der PCA 40 -mit einer 40-MB-

faehigen Computer.

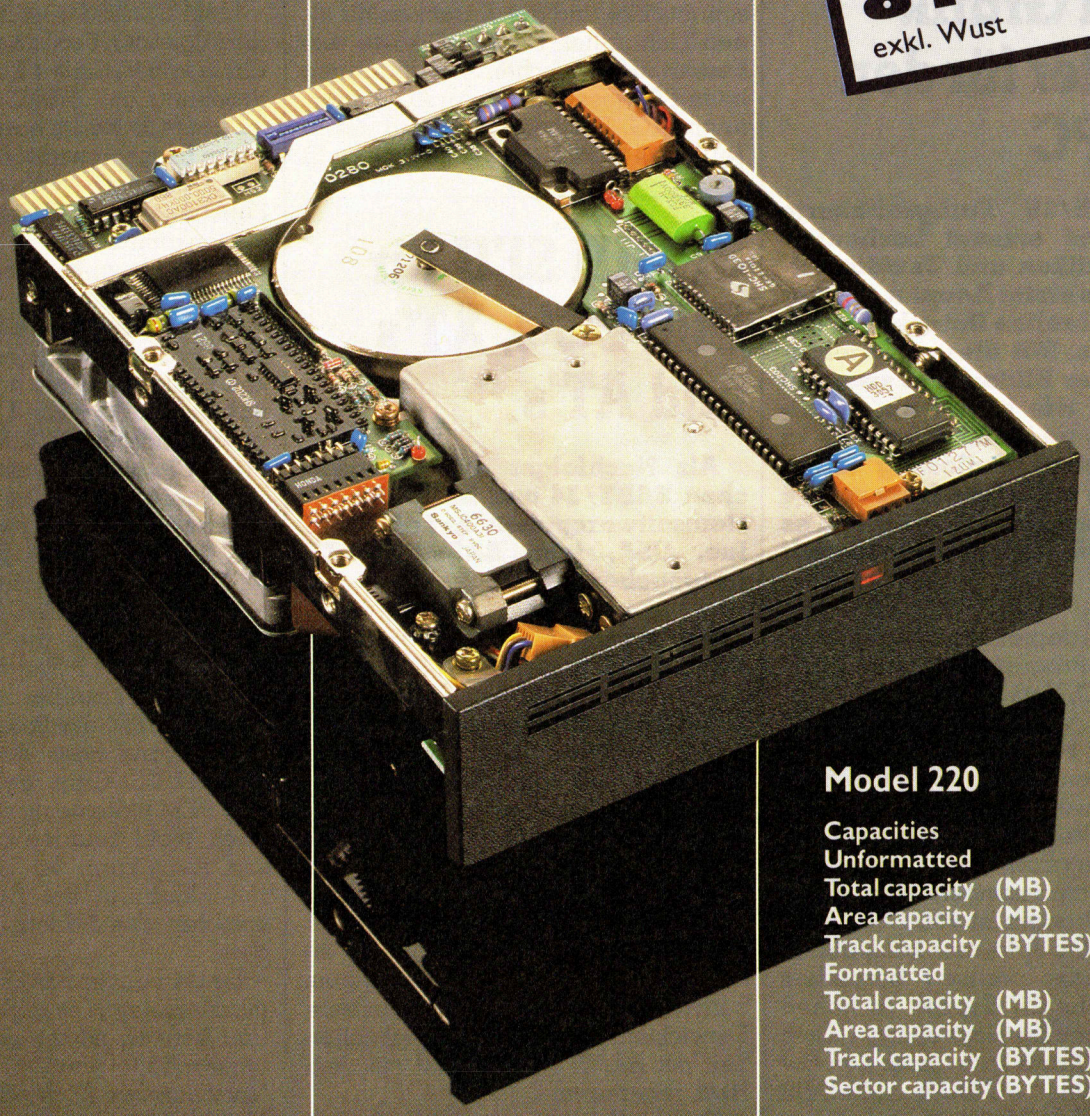
Festplatte (40 MB formatiert) nur 1000 Franken mehr.

STAECH-COMPUTER, Seilerstr. 25, 3011 Bern, Tel. 031/25.05.31 oder 031/25.58.60

j. schuitter
laserkonzepte

5 1/4" HARD-DISK

845.-
exkl. Wust

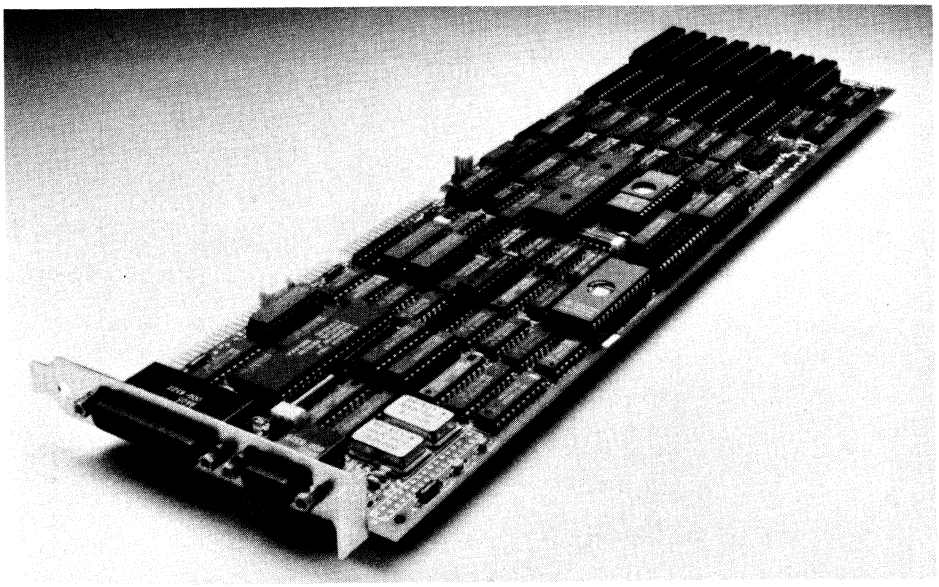


Model 220

Capacities		
Unformatted		
Total capacity	(MB)	25.5
Area capacity	(MB)	6.37
Track capacity	(BYTES)	10,416
Formatted		
Total capacity	(MB)	20.05
Area capacity	(MB)	5.01
Track capacity	(BYTES)	8,192
Sector capacity	(BYTES)	256
Disk		
Number of disks		2
Number of cylinders		612
Number of tracks		2,448
Magnetic heads		
Number of heads		4

PANATRONIC AG Zürich

Industriestrasse 59 CH-8152 Glattbrugg Telefon 01/810 32 10 Telefax 01/810 89 03 Telex 58 353



16-Bit Grafikadapter für den IBM-PC

Die BoB/16 Farbgrafikkarte von Persyst erzeugt hochauflösende Grafiken und Textdarstellungen. Als erster Adapter nutzt er den 16-Bit breiten Datenkanal des PC/AT aus. Mit dieser Kartenlogik läuft der Bildaufbau zweimal so schnell wie bisher ab. Grafiken, Texte und Bilder werden in der doppelten Auflösung abgebildet. Das volle Spektrum von 16 Grundfarben ist verfügbar und es können 136 Farbtöne dargestellt werden.

Mit dem Adapter können aktuellen CAD/CAM-Programme, wie Auto-CAD, P-CAD, Versa-CAD und Grafikprogramme wie beispielsweise PC/Paintbrush betrieben werden. Die hohe Auflösung von 640x400 Bildpunkten bewährt sich auch bei kommerziellen Programmpaketen wie Lotus 1-2-3, Symphony und Framework.

Softwareanpassungen sind keine nötig, das Produkt ist voll CGA (IBM Color Graphic Adapter) kompatibel. Der Anwender profitiert besonders von der mehr als doppelt so hohen Auflösung bei Textdarstellung (10x16 Matrix). Die Karte kann unter den Betriebssystemen PC-DOS, MS-DOS und Xenix eingesetzt werden.

Der Anwender kann 510 verschiedene Zeichen anwählen; 256 Zeichen aus dem Standard-Zeichensatz und 256 frei bestimmbare Zeichen, die mit einem Programm (Soft-Set-Utility) definiert werden. Weitere Extras sind eine eingebaute parallele Druckerschnittstelle und der Anschluss für einen Lichtgriffel. Die Grafikkarte verbessert die grafischen Anwendungen beim PC wesentlich. Info: ACU Informatik AG, Postfach 74, 6005 Luzern, Tel. 041/44'05'22. □

RPGIII SEU für IBM PC's, XT's und AT's

Als Nachfolger der erfolgreichen BABY/34 und BABY/36 Systemsoftwarepakete hat die CSPI Inc. USA, jetzt BABY/38 SEU freigegeben.

Wie die anderen BABY-Pakete hat BABY/38 SEU zum Ziel, Minicomputer Funktionen auf den PC zu transferieren. BABY/38 ist ein Bildschirmeditor, welcher die Entwicklung und Wartung von RPGIII-Programmen und Bildschirmformaten auf einem PC erlaubt. BABY/38 SEU enthält die gleichen Funktionen wie das IBM S/ 38 SEU und spart sowohl Entwicklungszeit wie Systemresourcen.

Beim Einsatz in Verbindung mit einem IBM S/38 wird das Host-System beträchtlich entlastet, wenn mehrere Entwickler Quellencode zur gleichen Zeit mit BABY/38 SEU statt mit dem Host generieren.

Softwarehäuser, welche selber über kein IBM S/38 verfügen, können jetzt ihre /38 Software auf dem PC entwickeln und warten.

Mit BABY/38 SEU erzeugte Programme können zu einem IBM S/38 übertragen und dort umgewandelt, getestet und ausgeführt werden. Info: Informatik, Beratung und Vertriebs AG, Volkmarstrasse 10, 8033 Zürich, Tel. 01/36'36'360. □

Editor und Terminalemulator in einem: LinkPCS für IBM

LinkPCS für IBM und IBM-kompatible Rechner ist speziell für die Kommunikation zwischen einem PC und dem 8052-BASIC Chip ausgelegt. Als Betriebssystem benötigt LinksPCS/IBM entweder das MS-DOS 2.11 oder ein entsprechend kompatibles wie zum Beispiel das PC-DOS. Die Hardware muss PC/XT/AT-kompatibel sein.

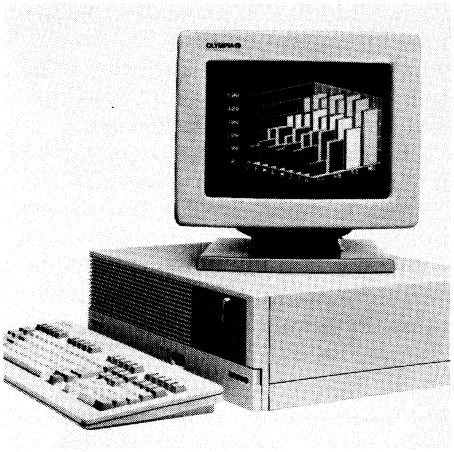
LinkPCS/IBM fasst im wesentlichen die folgenden Funktionen zusammen: Einen komfortablen Editor für die Bearbeitung des BASIC Programmtextes auf PCs. Dabei handelt es sich um einen Bildschirmeditor mit einem oder zwei Fenstern (Windows) dessen Funktionen mit WordStar-ähnlichen Tastenkombinationen oder mit Hilfe von «Pulldown-Menüs» gewählt werden.

Und ausserdem einen Terminalemulator, der aus dem IBM- oder kompatiblen PC ein einfaches Terminal macht, dessen Uebermittlungsgeschwindigkeit 110..9'600 Baud beträgt. Die hier eingesetzte Kommunikation arbeitet mit Interrupts und die Grösse des Eingangsspeichers kann frei gewählt werden. Er beträgt 2 K.. 32 K. Diese Eigenschaften erlauben auch bei MCS-BASIC V1.0 sehr hohe Baudraten und die Unterstützung vom XON/XOFF Protokolls des MCS-BASIC V1.1. Darüber hinaus kann mit LinkPCS/IBM der Programmtext vom Editor ganz oder teilweise in den 8052-BASIC Chip geladen werden und REM Statements werden von ihm nach Wahl geladen oder weggelassen. Auch kann der Text vom Terminal-Bildschirm bei Bedarf im Textspeicher des Editors abgelegt werden.

Beide Funktionen (Editor und Terminalemulator) sind gleichzeitig aktiv und können mit einem Tastendruck umgeschaltet werden. Info: Arbit AG, Lindenstrasse 22, 8153 Rümlang, Tel. 01/817'07'57. □

Olympia Olystar 60

Der seit einigen Monaten auch in der Schweiz lieferbare Mikrocomputer Olympia Olystar 60 erlangt hierzulande immer mehr Beliebtheit. Er ist modular aufgebaut, IBM AT-kompatibel und arbeitet mit dem Hochleistungsprozessor Intel 80286 bei einer Taktfrequenz von 8/6 MHz. Er kann wahlweise als Einzelplatz unter MS-DOS und Prologue oder als Mehrplatzsystem unter Prologue genutzt werden und zeichnet sich durch hohe Geschwindigkeit, Ausbaufähigkeit und grosse Flexibilität aus.



In der Grundversion wird der Olystar 60 mit einem 512 KB Hauptspeicher, einem 5.25-Zoll Diskettenlaufwerk mit 1,2 MB Speicherkapazität und einer Winchester-Festplatte mit 20 MB angeboten.

Als Bildschirm dient ein 15-Zoll Monochrom- oder ein 14-Zoll Farbmonitor mit Auflösung von 640x400 Punkten. Dargestellt werden 25 Zeilen zu je 80 Zeichen. Beide Bildschirme sind dreh- und kippbar sowie blend- und reflexionsfrei mit guter Farb- und Schriftwiedergabe.

Für die Dateneingabe steht eine ergonomisch richtig aufgeteilte, frei bewegliche Flachtastatur mit 39 programmierbaren Funktionstasten, separatem Cursor- und Textverarbeitungsblock, numerischem Tastenfeld und LED-Anzeige zur Verfügung. Der Olympia Olystar 60 ist mit einer seriellen Schnittstelle RS232C/V 24 und einer parallelen Schnittstelle Centronics ausgerüstet.

In der Grundversion sind MS-DOS, Prologue und GW-BASIC enthalten. Durch das Prologue Betriebssystem ist eine Aufwärtskompatibilität vom bereits bekannten Mikrocomputer Olympia People gewährleistet. Als Drucker können unter anderem die Olympia Typenraddrucker ESW 1000, ESW 2000, ESW 3000 oder einem Modell aus der Palette der Nadelprinter angeschlossen werden.

Als Optionen sind Speichererweiterungen, ein zweites 5.25-Zoll Diskettenlaufwerk, eine 20/30/60 MB Winchester-Festplatte und zur Datensicherung Streamer-Tapes erhältlich. Info: Olympia Büromaschinen AG, Ifangstrasse 91, 8153 Rümlang, Tel. 01/817'11'41. □

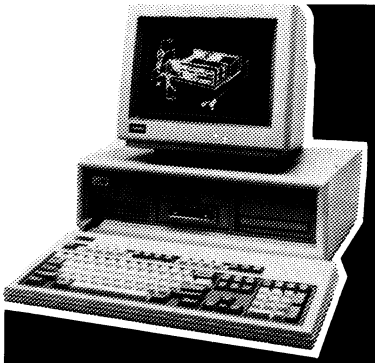
Desktop Publishing für den PC

Software Publishing Corp führt neu ihr erstes Softwarepaket für den Desktop Publishing Bereich auf dem IBM-PC ein. Die Software ermöglicht die Erstellung professioneller, typografisch einwandfreier kurzer oder langer Schriftstücke auf dem Personal Computer. Die Dokumente sind auf dem Bildschirm in der Form ersichtlich, wie sie auf dem Drucker endgültig erscheinen (das Programm ist WYSIWYG = What You See Is What You Get!).

Professional Publisher wurde durch die Firma BestInfo Inc. entwickelt, die auch SUPERPAGE II, das für den professionellen Markt geschaffene DPS Produkte erstellt hat.

Professional Publisher eignet sich zur Herstellung von Newsletters, Prospekten, Handbüchern etc. Die Software formatiert Texte, die über eine Textverarbeitungs-Software erstellt wurden, auf dem Bildschirm zu Dokumenten, wobei der Benutzer volle Kontrolle über die Seitengestaltung hat. Ueber «Stylesheets» kann die globale Seitengestaltung vordefiniert werden.

Professional Publisher kann Texte in ASCII- oder DCA-Format einlesen. Grafiken aus Harvard Presentation Graphics, Lotus 1-2-3, PC Paintbrush,



Das sind Weltrekorde!

- 3 x schneller als die bisher Schnellsten
- 2 x grössere Speicherkapazität als die bisher grösste Harddisk
- 40 x grössere MS-DOS-Verwaltung als die bisher Grösste

trotzdem nicht teurer . . . und kompatibel

von COMPAQ . . . dem zweitgrössten 16 Bit-Rechner-Hersteller der Welt. Und von dieser amerikanischen Spitzenmarke wurden nacheinander zwei Modelle zum «Computer des Jahres» gewählt von der zuständigen internationalen Jury . . . nämlich für 1984/85 und 1985/86. . . wegen fortschrittlichem Konzept und Zuverlässigkeit. Das ist noch keiner anderen Marke gelungen.

COMPAQ Deskpro 386
32 Bit ● INTEL 80 386 ● 16 MHz, umschaltbar ● 1 – 14 MB RAM
● 40 – 130 MB Festplatte
● MS-DOS-Verwaltung bis 14 MB!

Bevor wir einen anderen Personal Computer anschaffen, möchten wir die bewährten Höchstleistungs-Rechner von COMPAQ kennenlernen. Senden Sie deshalb unverbindlich die COMPAQ-Dokumentation an:

Firma: _____
zuständig: _____ Tel.: _____
Strasse: _____
PLZ/Ort: _____

COMPAQ-Import, Beratung, Service und Verkauf für die ganze Schweiz:
ComputerLand Zürich: Zentralstrasse 18, 8036 Zürich, Tel. 01/461 42 33
Basel: Emil Frey-Strasse 85, 4142 Münchenstein, Tel. 061/46 47 77
(Niederlassungen in Genf, Lausanne und Neuenburg – neues Kommunikations- Netzwerkzentrum in Zürich)

Dr. Halo und MS Windows Paint, sowie Bilder, die mit Scannern eingelesen wurden, lassen sich mit dem Text mischen. Der Text kann auch automatisch um die Bilder herum verteilt werden.

Professional Publisher unterstützt den HP Laserjet, Apple Laserwriter und andere Laserdrucker mit Postscript, arbeitet mit Scannern von Datacopy, Dest und Compuscan.

Lauffähig ist die Software auf einem IBM-PC/AT mit 640 KB Arbeitsspeicher, Hercules oder EGA Grafik-Adapter und einer Maus. Info: Computer-Graphix AG, Giessereistrasse 1, 8620 Wetzikon, Tel. 01/932'34'82. □

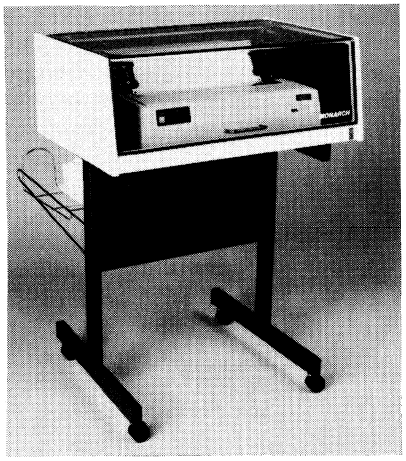
Hochleistungsfähiger Prozessor für den IBM-XT

Eine Beschleunigung der Rechnerleistung um bis zu 600 Prozent kann mit dem Einbau einer CPU-Zusatzkarte von Verbatim in die IBM-Modelle PC und PC/XT erzielt werden. Die Karte ist mit einem Intel 80286-Prozessor ausgerüstet und kann mit einem zusätzlichen 80287-Prozessor als Option erweitert werden. Der Prozessor hat eine Taktfrequenz von 7,2 MHz, der ebenfalls einsetzbare 80287-3 Math-Prozessor weist einen Takt von 4,8 MHz auf.

Die Karte mit dem weit leistungsfähigeren Prozessor lässt sich problemlos in die Erweiterungs-Slots der PCs einbauen. Die Verbindung mit der Speichereinheit wird am Sockel des zu entfernenden 8088-Prozessors hergestellt. Der Intel-Prozessor ermöglicht eine gegenüber dem PC AT um 20 Prozent höhere Rechnergeschwindigkeit, die besonders bei Spreadsheet-Applikationen und mathematischen Einsätzen ins Gewicht fällt. Die Kompatibilität mit IBM-Zusatzprodukten bleibt gewährt. Verbatim, eine Kodak-Tochter, liefert den Prozessor mit Garantieleistungen, die während der ersten sechs Monate den sofortigen Ersatz der Karte und während weiterer sechs Monate die Uebernahme von Material- und Reparaturkosten umfasst.

Verbatim ist einer der führenden Hersteller flexibler Speichermedien und ist in letzter Zeit auch mit einer Optical Disk als bedeutender Neuerung hervorgetreten. Info: Verbatim S.A., Postfach 3, 1211 Genève 19, Tel. 022/98'74'44. □

MONARCH
Schallschluckhauben
... bringen jeden Printer zum Schweigen!



– mit Hochleistungslüfter gegen Wärmestau
– modernes Design braun/beige
– Papierein-/Papieraustritt optimal positioniert (Handling).
– **99%** der Lärmenergie wird absorbiert.
– **GRATISTEST** 5 Tage

TRESOR DATA

TRESORDATA Zürich
Tresore, EDV-Mobiliar, EDV-Zubehör
Luchswiesenstrasse 191, 8051 Zürich
Tel. 01 - 41 57 57

Victor PC mit 3.5- und 5.25-Zoll Floppy

Sehr elegant und exklusiv löst Victor Technologies mit dem VPC II-D ein längst bekanntes Problem: Mit ihrem Tischmodell (mit oder ohne Harddisk bzw. Hardcard) können Benutzer von z.B. portablen PCs ihre 3.5-Zoll Disketten zusammen mit 5.25-Zoll Floppies auf dem gleichen Gerät verarbeiten; in der Grundeinheit ist für beide Disketten-Typen je ein Laufwerk verfügbar.

Die optionalen 3.5-Zoll Disketten können für 360 KB und 720 KB formatiert werden (5.25-Zoll Disketten für 360 KB). Für den VPC II-D ist es gleichbedeutend ob er von/zu 3.5-Zoll oder 5.25-Zoll Disketten liest/schreibt. Es kann entweder das 3.5-Zoll oder das 5.25-Zoll Laufwerk als Drive A zugeordnet werden; das andere wird automatisch Drive B. Diese Zuordnung ist von der Front her manuell umschaltbar. Ein Laufwerk kann ebenfalls ganz weggeschaltet werden, so dass ein Laufwerk sowohl den Drive A wie B darstellt. Selbstverständlich kann auch von einem 3.5-Zoll Laufwerk her aufgestartet werden.

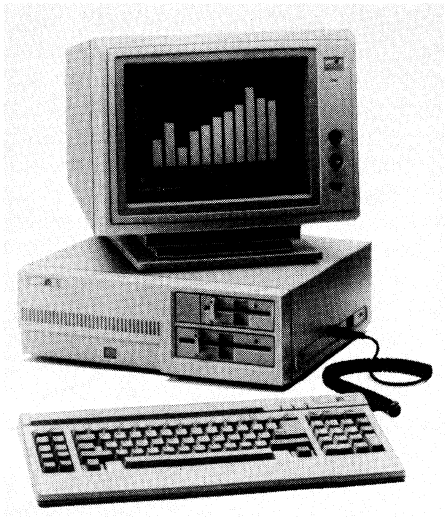
Diese Problemlösung stammt von der Firma Urs Zogg, 7220 Schiers, die die Integration ins Grundgerät VPC II-D vornimmt. Info. Victor Technologies AG, Heimstrasse 27, 8953 Dietikon, Tel. 01/741'01'44. □

10 MHz Turbo Card

«Surprise» heisst die neue Zauberformel, um Ihren PC bis zu zweieinhalbmal zu beschleunigen - und dies ohne zusätzlichen Steckplatz!

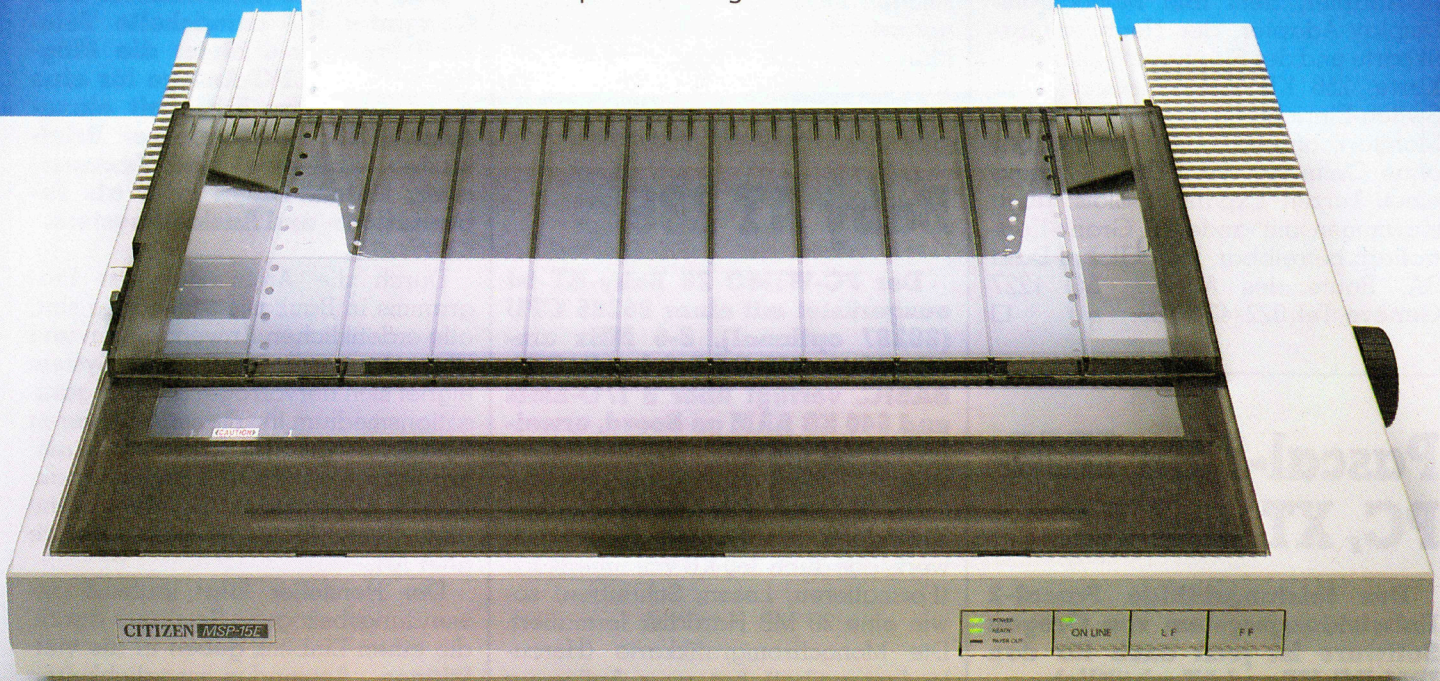
Ein einfacher Keyboard-«Toggle»-Befehl genügt, um Ihren PC von 4.77 MHz auf 10 MHz zu bringen. Dank dem Dual Speed Design ist eine 100% IBM-Kompatibilität gewährleistet.

Die Surprise-Karte ist einfach zu installieren und passt in jedem IBM PC/XT oder Kompatiblen. Die Karte kostet weniger als Fr. 700.--. Die Garantieleistungen des Herstellers erstrecken sich auf fünf Jahre. Info: se-data computer systems, Neugasse 14, 8810 Horgen, Tel. 01/725'05'35. □



EIN GANZES JAHR BESSER.

- Volle Zwei-Jahres-Garantie
- 80 bzw. 136 Zeichen pro Zeile
- 160 CPS bei Datenverarbeitung
- 40 CPS bei Schönschreibqualität
- IBM und Epson kompatibel (per Schalter anwählbar)
- Volle Grafikfähigkeit
- 8K-Druckerspeicher
- Ladbare Zeichen
- Proportionalschrift
- Schneller Zeilenvorschub
- Schiebetraktor
- Einfache Papierzuführung



Ein Jahr Garantie ist normal. Citizen ist um ein Jahr besser: Zwei Jahre ohne Ärger sind bei Citizen Computer-Druckern garantiert. Und dann der Preis. Sie könnten auch mehr bezahlen als für ihren Citizen MSP-10e oder MSP-15e – und vielleicht weniger für ihr Geld bekommen.

Wollen Sie noch mehr über die Citizen-Vorteile wissen?

Rufen Sie Laurent Poitrine, CPI Zürich, Telefon 01-740 91 71 oder Liliane Eckerlin, CPI Genêve, Telefon 022-43 68 00 an.

CITIZEN
COMPUTER DRUCKER

Es gibt Dinge, auf die Sie sich immer verlassen können.

AutoSwitch EGA-Karte

Neu im Taxan-Sortiment befindet sich ein EGA-Grafikkontroller für IBM- und kompatible Personal Computer. Taxan's EGA-Karte ermöglicht dem Benützer mehr Komfort und dem PC mehr Leistung.

Die Taxan EGA-Karte bietet: Alle Funktionen, wie sie der IBM Enhanced-Grafikadapter bietet; die EGA-Karte erkennt die Modi der gerade laufenden Programme und stellt sich automatisch auf die geforderten EGA-, CGA- oder Hercuels-Standards ein (dies erübrigt Anpassungssoftware für bestehende und zukünftige Anwenderprogramme); volle Kompatibilität mit dem IBM Farbgrafik-Adapter, dem IBM Monochrom-Display-Adapter, der Hercules-Grafikkarte und der Plantronic Colorplus-Karte; 256 KB Video-Memory; Dip-Switch-Einrichtung für verschiedene Monitorkonfigurationen (einstellbar ohne Oeffnung des PCs); Belegung eines kurzen Expansion-Slots; auch zusammen mit anderen Grafik-Kontrollern betreibbar. Info: Global Data SA, Route des Acacias 54, 1227 Genève, Tel. 022/43'87'89. □

Pascal-2 für IBM-PC, XT und AT

Das leistungsfähige Pascal-2 Entwicklungssystem von Oregon Software ist jetzt auch für den IBM-PC, XT und AT erhältlich.

Bisher war Pascal-2 vor allem in der DEC-Welt und bei Benutzern von UNIX/68000-Systemen bekannt. Pascal-2 machte sich für kompakten Programmcode und schnelle Programmlaufzeiten einen Namen. Unter Pascal-2 entwickelte Programme sind portabel, sie können einfach und oft ohne Aenderungen auf andere Hardware übertragen werden. Zudem sind Cross-Compiler erhältlich: von VAX auf 68000, 32000 und MS-DOS und von MS-DOS auf 68000.

Pascal-2 ist voll in das MS-DOS Betriebssystem und in die bisher erhältlichen Compiler integriert. Pascal-2 Programme können Microsoft For-

tran, C, Pascal und Assembler-Routinen aufrufen und so von vorhandenen Programmbibliotheken Gebrauch machen.

Den Turbo-Pascal-Benutzern bietet Pascal-2 echte Wachstumsmöglichkeiten, denn der von Pascal-2 generierte Code ist zwei- bis dreimal schneller. Die Strings sind kompatibel, und es können die gleichen vordefinierten Prozeduren und Funktionen eingesetzt werden. Zusätzlich erlaubt Pascal-2 den Zugang zu zahlreichen DOS Service-Routinen und unterstützt Netzwerk-Files. Info: Mühletaler AG, Kirchstrasse 2, 4536 Attiswil, Tel. 065/77'29'11. □

Baby AT 286

Der PC-WIMO 20 Baby-AT ist ausgerüstet mit einer 80286 CPU (80287 optional), 8-6 MHz umschaltbar, MS-DOS 3.1 mit GW-BASIC, verfügt über 8 I/O-Slots und 640 KB RAM on Board, erweiterbar auf 1 MB.

An Diskettenlaufwerken stehen zur Verfügung ein 1,2 MB Floppylaufwerk, das auch 360 KB voll unterstützt (Formatieren, Lesen, Schreiben) sowie eine 20 MB Harddisk formatiert. Die Monochromgrafikkarte (Hercules-kompatibel) hat eine Auflösung von 720x348 Punkten mit Printerport.

Der Baby AT 286 besticht durch sein kompaktes Design mit attraktiven Front Panel, Resettaste sowie Turboschalter und Schlüssel. Die im modernen Design gehaltene Slim-Line Tastatur mit 105 Tasten, 20 Funktionstasten, Beep, cls und Reset ist als deutsche DIN- oder ASCII-Tastatur lieferbar.

Ein monochromer hochauflösender Amber-Monitor (1000 Lines) vervollständigt die Standard-Ausrüstung. Selbstverständlich sind Farbgrafik und EGA ebenfalls erhältlich. Info: Max Moser Management AG, Lättichstrasse 8a, 6340 Baar, Tel. 042/31'55'53. □

Telemail für IBM-PC/XT/AT

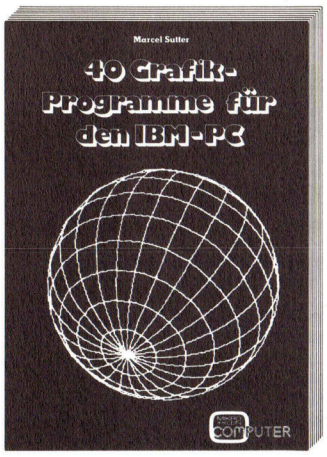
Das von der Schaffhauser PIM Computer AG entwickelte Telemail-Programm bietet die Möglichkeit, Ihre PC-Anlage für eine ganz neuartige Tätigkeit einzusetzen: Als elektronischer Briefkasten, Software- und Textspeicher, Datenbanken oder als Informations- und Auskunftsystem.

Durch die Architektur des Programms in Baukasten-Ideologie sind alle erdenklichen Anwendungen und Wünsche realisierbar. Das System eignet sich hervorragend als Organisationsmedium für Firmen und deren Filialen, Telearbeit, Aussendienstmitarbeiter, Verbände, Zeitungsredaktionen, Uebersetzer, Service-Hotline und Vermittlungs-/Auskunftsdienste aller Art.

Der Hersteller führt laufend anwendungsbezogene Seminare durch, die einen Einblick geben in die vielfältigen Anwendungsmöglichkeiten dieses neuen Mediums. Info: PIM Computer AG, Lochstrasse 18, 8200 Schaffhausen, Tel. 053/4'54'50. □

M+K 87-1
erscheint am
17. Februar
Inserateschluss
ist am 21. Januar
☎ 041-31 18 46

Das aktuelle Fachbuch

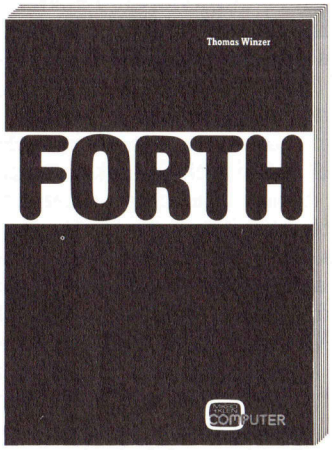


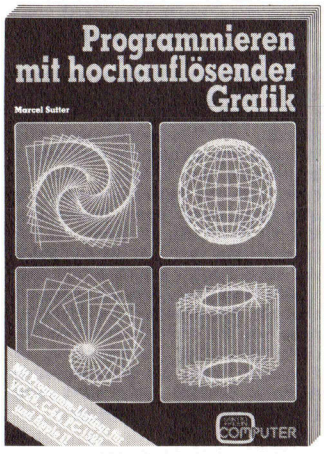
40 Grafik-Programme für den IBM-PC

Die grafische Datenverarbeitung nimmt ständig an Bedeutung zu. Wir begegnen der Computergrafik nicht nur bei allen Computerspielen, sondern in zunehmendem Masse am Arbeitsplatz des Technikers und Ingenieurs in Form von CAD. Im kaufmännischen Bereich hat die Business-Grafik mit ihren anschaulichen Diagrammen längst die Zahlentabellen abgelöst. Der Autor zeigt anhand von 40 Beispielen für den IBM-PC und seine kompatiblen Brüder, dass die Erzeugung von Computergrafiken erstaunlicherweise gar nicht so schwierig ist wie der interessierte PC-Anwender glaubt. Schritt um Schritt wird der Grafik-Neuling in die faszinierende Welt der Computergrafik eingeführt. Dem Leser gelingt es in ganz kurzer Zeit und in ständigem Vergleich von Aufgabe und Lösungsweg mit kurzen Programmlistings, die ihm neben dem beabsichtigten Aha-Effekt auch sofort ein Erfolgserlebnis vermitteln, das grundlegende Prinzip der Grafikprogrammierung zu erlernen.

Eine Einführung in das Programmieren mit FORTH

FORTH hat gegenüber den populären Programmiersprachen BASIC oder Pascal bisher eher ein Schattendasein geführt. Das hängt zum Teil damit zusammen, dass FORTH keine Programmiersprache im klassischen Sinn ist, sondern sich eher mit dem Begriff «Programmierumgebung» beschreiben lässt. Markanter Pluspunkt: FORTH ist maschinennäher als andere höhere Programmiersprachen, dadurch ergibt sich eine wesentlich gesteigerte Ausführungsgeschwindigkeit der Programme, sowie eine bessere Speicherausnutzung. Ziel des Buches ist, dem interessierten Computer-Anwender den Grundgedanken und die enormen Möglichkeiten von FORTH in der Weise näher zu bringen, dass er sich ein Bild von dieser Sprache machen und entscheiden kann, ob sie sich für eigene Anwendungen eignet. Dazu werden die wichtigsten Befehle anhand zahlreicher Beispiele ausführlich und leicht verständlich erklärt und durch zusammenfassende kurze Programme ergänzt.





Programmieren mit hochauflösender Grafik

Wer kennt sie nicht, die raffinierten Demo-Programme, die in Computershops oder auf Computer-Ausstellungen stets die Aufmerksamkeit auf sich ziehen. Wer den Wunsch hat, ähnliche Programme auf seinem Computer selbst zu entwickeln, kommt sehr rasch in Schwierigkeiten. Das bereits als Standardwerk geltende Buch «Programmieren mit hochauflösender Grafik» führt schrittweise in das Programmieren mit HRG ein. Die vorgestellten vierzig Grafikprogramme sind nicht wie üblich für ein spezielles Computersystem geschrieben: Die kurzen BASIC-Programme verwenden nur einen Grafikbefehl in metasprachlicher Form (keine PEEK- und POKE-Anweisungen), sind selbsterklärend, können top-down gelesen werden und lassen sich problemlos auf jedes Computersystem anwenden. Ein unentbehrliches Buch also für jeden, der sich mit der Computergrafik näher befassen will und bisher nirgends einen systematischen Einstieg dafür finden konnte.

Bestell-Coupon

Ich/Wir bestelle(n) das Buch (Gewünschtes bitte ankreuzen)

☐ **Eine Einführung in das Programmieren mit FORTH**
ISBN 3-907007-04-2, 140 S., Preis Fr. 39.50

☐ **40 Grafik-Programme für den IBM-PC**
ISBN 3-907007-03-4, 168 S., 75 Abb., Preis Fr. 35.--

☐ **Programmieren mit hochauflösender Grafik**
ISBN 3-907007-02-6, 288 S., 72 Abb., Preis Fr. 45.--

Bestellungen über die nächste Buchhandlung oder direkt beim Verlag:

Mikro+Kleincomputer Informa Verlag AG
Postfach 1401, 6000 Luzern 15

Eilbestellungen
041 / 311846

Meine/unsere Anschrift:

Name/Vorname/Firma

Beruf

Strasse

PLZ/Ort

X

Unterschrift Datum

Die EXEGA-Karte - ein Kleeblatt für Anspruchsvolle

Wer mit einem IBM PC/XT/AT oder einem kompatiblen Computer arbeitet, wird schon oft über den Color Graphic Adapter, kurz CGA, gestöhnt haben. Seine Darstellung von Schriftzeichen in einer Box aus 8x8 Punkten ist eine arge Zumutung an die Augen, seine Monochromgrafik in 640x200 Punkten nicht gerade das Nonplusultra und die Darstellung von vier Farben bei 320x200 Punkten gerade noch für Bildschirmspiele tauglich. Da trumpfen manche Homecomputer mit augenfreundlicheren Tatsachen auf. Es ist deshalb nicht überraschend, dass der Enhanced Graphic Adapter von IBM allmählich den CGA verdrängt. Beschleunigt wurde dieser Umstand durch die technische Entwicklung, durch Speicherschaltkreise mit hoher Bitdichte einerseits und durch ambitionierte Firmen andererseits, die die EGA-Logik in wenige Grossschaltkreise pressten. Das Resultat: Heute existiert eine Vielzahl EGA-kompatibler Karten, die klein und preisgünstig sind und zudem noch zwei andere Karten ersetzen, den Monochrom Display Adapter (MDA) und den CGA. Wir haben die EXEGA-Karte für Sie getestet.

Programmiersprachen

Es gibt weltweit mehr als hundert höhere Programmiersprachen, wovon sich aber nur etwa 15 % erheblich voneinander unterscheiden. Von einigen Programmiersprachen existieren teilweise mehrere Versionen, sogenannte «Dialekte». Hinzu kommen Besonderheiten der verschiedenen Computeranlagen und deren Compiler, auf denen mit den höheren Programmiersprachen gearbeitet wird, was die Entwicklung weiterer Dialekte fördert. In unserer Uebersicht soll im wesentlichen nur auf die Hauptunterschiede der verschiedenen Sprachen eingegangen werden. Die gebräuchlichsten Programmiersprachen werden etwas ausführlicher behandelt. Bei den Beschreibungen der einzelnen Sprachen werden teilweise Probleme angesprochen, die auch für andere Sprachen gelten. So ist die Besprechung einer bestimmten Programmiersprache hier nicht ausschliesslich dieser gewidmet, sondern greift über zu anderen Sprachen und allgemeinen Problemen.

Einführung in dBASE III

dBase III von Ashton Tate ist ein relationales Datenbanksystem, mit dem Daten in nahezu jeder Form verarbeitet werden können. Es unterscheidet sich von seinem Vorgänger, dBase II, durch seinen wesentlich grösseren Datenumfang. Bei den Befehlen des dBase III handelt es sich um Worte, die aus dem Englischen entlehnt worden sind, die aber so einleuchtend und leicht verständlich sind, dass selbst ein Laie den Sinn dieser Befehle sehr schnell erfasst. In mehreren Fortsetzungen wollen wir den ungewöhnlichen Versuch wagen, dBase III in Form eines Kurzlehrgangs so zu erklären, wie normalerweise eine Programmiersprache gelehrt würde.

LOGITECH C7 - die Supermaus

Unter den manigfaltigen Zusatzgeräten, die es zu einem Computer gibt, ist wohl ein Drucker am unentbehrlichsten. An nächster Stelle sollte dann eine Maus stehen, denn wer auf dieses Eingabegerät verzichtet, nutzt nur wenige Möglichkeiten der Arbeitserleichterung aus, die der Computer bietet. Dort, wo es um die Eingabe eines umfangreichen Text- oder Zahlenmaterials geht, kommt man um die Tastatur nicht herum, zahlreiche Programme lassen sich jedoch mühelos und mit weniger Fehler mit Hilfe einer Maus bedienen. Unter den Eingabegeräten hat sich deshalb die Maus eine dominierende Stellung erobert. Ausser ihr stehen vor allem Lichtstift und Digitalisiertablett im Einsatz. Als universelle Eingabegeräte weisen sie aber Nachteile auf, welche die Maus nicht hat.

Back-up

M+K 86-5
MSX zum Zweiten
Der EPSON PC Plus
Der Matrixdrucker STAR NL-10
Printmastern Sie schon?
Sharp PC-1600 - kompatibler «PC» im Taschenformat
Textverarbeitung mit WORDCRAFT
Schriften gestalten nach Setzerart
CASIO's Taschencomputer PB-1000
Programmieren mit dem IBM-PC (4)
Künstliche Intelligenz (2)
PRINTDIR unter MS-DOS
Dokumentation von MAKE-UP
Der Computer konjugiert französische Verben
DSI-020 - ein 68020 Coprozessor für den IBM-PC

MIKRO+KLEIN COMPUTER

Das Schweizer Kleincomputer-Magazin

8. Jahrgang ISSN 0251-0006

© Mikro+Kleincomputer Informa Verlag AG gegr. 1979 by Ernst Erb, Luzern

Im gleichen Verlag erscheint im 4. Jahrgang der aktuelle COMPUTERMARKT

Verlag, Redaktion, Inserate
Mikro+Kleincomputer Informa Verlag AG
Seeburgstrasse 12, 6000 Luzern 15
Postanschrift: Postfach 1401, 6000 Luzern 15
Telefon 041 - 31 18 46, Tx 72 227

Postcheck-Konten:
Luzern 60 - 27181-0
Stuttgart 3786-709 (BLZ 600 100 70)
Wien PSK 7975.035

Verlagsleitung
Hans-Jürgen Ottenbacher

Redaktion
Eric Hubacher, El. Ing. HTL; Peter Fischer

Ständige Mitarbeiter
Leopold Asböck; Thomas Gutekunst; Heinz Kastien, Dipl. Ing.; Oliver Rosenbaum, Dipl. Ing.; Marcel Sutter

Manuskripte und Copyright
Manuskripte werden von der Redaktion entgegengenommen. Die Zustimmung zum Abdruck wird vorausgesetzt. Für unverlangt eingesandte Manuskripte wird keine Haftung übernommen. Mit der Zustellung von Manuskripten anerkennt der Autor die Copyrightbestimmungen des Verlages. Eine Verpflichtung zum Abdruck besteht nicht. Mit der Annahme von Manuskripten durch die Redaktion und der Bestätigung durch den Verlag hat dieser das Recht zur exklusiven Veröffentlichung der entsprechenden Beiträge in seinen verlagseigenen Publikationen sowie zur Übersetzung in andere Sprachen erworben. Veröffentlichte Beiträge werden Eigentum des Verlages. Pressetexte werden nicht bestätigt. Die Publikation von Pressemitteilungen über neue oder wesentlich verbesserte Produkte ist eine Dienstleistung des Verlages. Über die Auswahl der Texte und Bilder, Kürzungen und Umformulierungen sowie deren Präsentation entscheidet die Redaktion. Ein Recht auf Veröffentlichung besteht nicht. Für die Veröffentlichung wird keine Gewähr oder Garantie übernommen, auch nicht dafür, dass die verwendeten Schaltungen, Firmennamen und Warenbezeichnungen usw. frei von Schutzrechten Dritter sind. Die Verwendung der Information erfolgt auf eigenes Risiko. Mit Verfassernamen gekennzeichnete Beiträge geben nicht unbedingt die Meinung der Redaktion wieder.

Jeder Nachdruck, auch auszugsweise, sowie Vervielfältigungen oder sonstige Verwertung von Texten aus MIKRO+KLEINCOMPUTER nur mit schriftlicher Genehmigung des Verlages und unter voller Quellenangabe.

Erscheinungsweise
zweimonatlich (gerade Monate)

Bezug
Jahresabonnement Fr. 42.– (inkl. Versand und Porto). Ausland (Europa) Fr. 49.–. Abbestellung ist durch schriftliche Kündigung jeweils 8 Wochen vor Ablauf des laufenden Bezuges möglich. Der Abonnementsbetrag ist nach Erhalt der Rechnung zur Zahlung fällig.
Nachbezug SFr. 8.– pro Heft

Inserate
☎ 041-31 18 46

Druck Unionsdruckerei AG Luzern

Printed in Switzerland

COMPUTERMARKT berichtet gezielt über alles, was auf dem Schweizer Computermarkt laufend angeboten wird. Die handverlesenen Produkt-informationen decken das gesamte Spektrum der Computerangebote ab.

ISSN 0254-5012 Fr. 4.–

COMPUTER MARKT

DIE AKTUELLE COMPUTERINFORMATION 4/86

SORD Unibox bietet echte UNIX-Umgebung

CIM - mehr als ein Schlagwort

Einführung in die Informatik (4. Teil)

Ein Lehrgang für PC-Einsteiger und PC-Interessierte

Zwei Drucker an einem Computer

Erfolgreicher Schweizer Start für den Amiga

Die neuen «offenen Systeme» AP und APEX sind da

Above Board für bis zu 8 MB-RAM

Erste Hilfe bei Initialisierungspannen neuer Disketten

Gehören Sie zum Kreis der Kleincomputeranwender, die ihre Disketten selbst initialisieren können? Wenn ja, ist für Sie die Diskettenbeschaffung relativ einfach und meist sehr preisgünstig. Dafür können aber beim erstmaligen Initialisieren Probleme auftreten. Und wenn der Kleincomputer bei der soeben neu gekauften Diskette streikt, weil er die Initialisierung nicht durchführen kann, vermutet man meist die Floppy als den Störenfried. Die Angabe des Diskettenherstellers, wonach die vorliegende Scheibe zu «100 % fehlerfrei» sei, scheint falsch zu sein. Der Disketten- und Laufwerkhersteller BASF klärt diesen Widerspruch auf und erläutert Ursachen und Abhilfemassnahmen bei Initialisierungspannen.

PC-Software kurz vorgestellt

- Wer bietet wo was an
- Wo gibt's welche Software
- Was läuft in der Schweizer Computer-Branche
- u.v.a.m.

COMPUTERMARKT gibt's sechsmal im Jahr, mit PC-Neuheiten aus erster Hand, exklusiven Hintergrund-informationen und brandheissen Insider-Meldungen.

MIKRO+KLEINCOMPUTER INFORMA VERLAG AG
Postfach 1401, 6000 Luzern 15, Tel. 041-31 18 46

Die leisen Riesen von peco

LASER Printer
für kühle Rechner!

mit dem sagenhaft günstigen

Blatt-Preis (siehe unten)

Quadlaser der Vollprofi

- Speicher 2.5 MB
  für schnellste Grafikaufbereitung
- Bis zu 60 Fonts gleichzeitig
- Netzwerkgeeignet, mit Mini-Link
  Anschluss an IBM-Grosscomputer (Option)
- Emulationen, Fonts etc. softwaregesteuert
- Font-Editor, Formulargenerator,
  Fonts und vieles mehr wird mitgeliefert
- Sehr hohe Lebensdauer (600'000 Blatt)

vollausgerüstet

Fr. 10'900.–

Betriebskosten:

PECO-Modell:	
Centronics Compact	11 Rp. p. Blatt
PECO-Modell:	
Quad-Laser	12 Rp. p. Blatt
Konkurrenz.-Mod.:	
Die meistverkauften	24–32 Rp. p. Blatt

Einsparung bei:
50'000 Blatt p. J.

Fr. 6900.–
pro Jahr

Centronics Compact – der Sparsame

- sehr einfache Bedienung
- Papiergrösse: Postkarte bis B4
- Geeignet für alle PC's
- Epson-, Diablo 630-, HP-Kompatibel

Fr. 7995.–

star
. . . die Drucker seit Gutenberg

peco

PECO AG Die Druckerspezialisten

Luzernerstr. 32 + 42 5620 Bremgarten Tel. 057/33 03 05