81
3

2. Jahrgang ISSN 0251-2017 ALLES ÜBER

CBM/PET NEWS

ÜBER ALLES

AZ CH-6006 Luzern / Verlag SCC AG, Seeburgstrasse 12 / Erscheint 6mal jährlich

Lieber PET - Freund,

an der neuen Schrift und der Gestaltung der vorliegenden Ausgabe sehen Sie, dass auch wir mit der Zeit gehen und heute das Heft nicht mehr mit der Schreibmaschine schreiben, sondern zu einem Wordprozessorsystem übergegangen sind. Für die Interessenten sei soviel gesagt. Die CBM/PET New's werden auf einem CBM 3032 mit Floppy 3040 geschrieben, wobei ein Wordprozessor II zur Anwendung gelangt. Der Ausdruck erfolgt auf einer Olympia Schreibmaschine ES 100 mit IEC - Interface. Sicher wird Ihnen das bessere Schriftbild sehr von Nutzen sein.

Wir hatten, wie angekündigt, vor in diesem Heft einen Teil dem Thema "Mininterrupt und Interrupt" zu widmen. Da aber die Leserbriefe, die dazu eingingen, sehr umfangreich sind, sehen wir uns gezwungen dieses Thema auf die nächste Ausgabe zu verschieben, um dann ausführlich Stellung zu diesen interessanten Themen unter der Rubrik "Tricks und Tips" zu nehmen. Sollten Sie noch interessante Beitraege zu diesen Themen haben, so senden Sie uns diese bis spätestens 20.Juni 1981 ein.

Da in diesem Heft die Rubriken Maschinensprache und Programm des Monats sehr umfangreich sind, müssen wir dies eine Mal auf die Leserbriefe verzichten. Dafür werden wir dann im nächsten Heft gerade zwei Themen aufgreifen, zu denen verschiedentlich Anfragen vorgelegen haben und zu denen uns aus dem Kreis unserer Leser eine grosse Zahl von Zuschriften zugegangen sind. Dies ist einmal der bereits erwähnte Interrupt und andererseits das Thema "Schutz vor unberechtigtem Listen".

Während uns Leserbriefe mit Tricks und Maschinenroutinen in grossem Masse zugesandt werden, vermissen wir in letzter Zeit Programme, die sich für die Rubrik "Programm des Monats" eignen. Wie Sie wissen, werden die publizierten Programme honoriert. Senden Sie uns also bitte auch zu diesem Thema Ihre Vorschläge, natürlich können Sie auch mit Ihren Wünschen zum Programm des Monats als auch mit Anregungen zur Gestaltung an uns herantreten, ebenso wie mit Ihren Fragen. Wenn immer möglich werden wir Ihren Wünschen gerecht werden und Ihre Fragen beantworten.

Mit den besten Wünschen für einen schönen Sommerurlaub verbleiben wir bis Anfang September.

Heinz Kastien

Tricks und Tips

INPUT - ROUTINEN

Im letzten Heft hatten wir unter der Rubrik "Programm des Monats Basic - Basic" das Wetterprogramm vorgestellt, das eine Subroutine zur Eingabe der Daten enthielt. Diese Subroutine unterscheidet sich wesentlich von den bisher bekannten Methoden der Eingabe von Daten via Tastatur.
Folgende Methoden sind bekannt:

1. Der INPUT - Befehl
2. Der GET - Befehl

Beide Eingabearten haben unbestrittene Vorteile, leider aber auch erhebliche Mängel.
Der INPUT Befehl erlaubt die Eingabe von numerischen und alphanumerischen Variablen, es ist jedoch nicht möglich nur ein Space einzugeben, wohl aber ein geshiftetes Space. Wird nach einem INPUT - Befehl ohne jegliche Eingabe die RETURN - Taste gedrückt, steigt der Rechner aus dem Programm aus und gibt READY. Die Aestheten unter den Programmierern empfinden sicherlich auch das Fragezeichen auf dem Bildschirm als störend, das bei einem INPUT Statement entsteht.
Das nach dem Drücken der RETURN - Taste, der Rechner das Programm verlässt, falls nicht vorher ein Wert eingegeben worden ist, lässt sich sehr einfach vermeiden. (Dieses Verfahren wurde in Mikro - und Kleincomputer 80/6 bereits eingehend beschrieben.) Wird dem ersten INPUT - Befehl POKE 14,1 vorangesetzt, so kann auch nur ein Space als Wertzuweisung erfolgen und der Rechner verlässt das Programm beim Drücken der RETURN - Taste ohne Wertzuweisung nicht mehr. Gleichzeitig wird das Fragezeichen unterdrückt. Nach dem INPUT - Befehl muss ein POKE 14,0 folgen. Mit POKE 14,1 wird die Nummer des aktive I/O Kanal auf 1 gesetzt. (Für das alte Betriebssystem muss an Stelle von POKE 14,1 POKE 3,1 und für das 8000 System POKE 16,1 benutzt werden.) Folgen mehrere INPUT - Statements aufeinander, so muss nur vor dem ersten INPUT POKE 14,1 stehen und hinter dem letzten INPUT POKE 14,0, allerding muss hinter jedem INPUT ein PRINT - Statement stehen, da sonst die einzelnen INPUT alle hintereinander erscheinen.
Es gibt eine zweite Möglichkeit, die zum gleichen Resultat fuehrt.
Das Keyboard wird als File behandelt und mit OPEN1,0 eröffnet, selbstverständlich müssen nun auch alle INPUT Befehle INPUT#1,A$ heissen. Zur Meldung des Promts muss nun ohnehin ein PRINT Statement nachgesetzt werden, das dann auch das Carriage Return erzeugt.
Der GET - Befehl zeigt den grossen Vorteil der sehr schnellen Eingabe und des Fehlens des Fragezeichens auf dem Bildschirm. Leider ist es aber nicht möglich mit einem GET - Befehl mehr als ein Zeichen zu übernehmen. Mehr muss wohl an dieser Stelle nicht über den GET - Befehl gesagt werden, da er hinlänglich bekannt sein dürfte. Nun aber zu der GET - Subroutine. Mit diesem Unterprogramm ist es möglich auch mehr als ein Zeichen mittels GET einzugeben, und darüberhinaus bestimmte Zeichen von der Eingabe auszuschliessen, z.B. nur numerische oder alphanumerische Zeichen, oder nur Gross - bzw. Kleinbuchstaben. Ebenso ist eine definierte Funktionszuweisung zu bestimmten Tasten oder eine minimale und eine maximale Länge der Eingabe programmierbar. Die nachfolgende Subroutine, die in jedes Programm eingebaut werden kann, soll dieses Verhalten demonstrieren.

```
10000 X1$="":A=0:PRINT" ▌";
10010 GETZZ$:IFZZ$=""THEN10010
10020 ZZ=ASC(ZZ$)
10030 IFZZ=20THEN10100
10040 IFA<MTHEN10060
10050 IFZZ=13THEN10160
10060 IFZZ<32ORZZ>96THEN10010
10070 X1$=X1$+ZZ$:A=A+1
10080 PRINTZZ$;"▌";
10090 GOTO 10010
10100 IFA=1THENX1$="":A=0:GOTO10010
10110 IFA<1THEN10010
10120 A=A-1
10130 X1$=LEFT$(X1$,A)
10140 PRINT"▌▌";
10150 GOTO10010
10160 PRINT" "
10170 RETURN
READY.
```

Zeile 10000 X1$ wird als Leerstring definiert. Die Variable A = 0 gesetzt und ein Space und ein Strich gedruckt sowie der Cursor um eine Stelle nach rechts geschoben.
Zeile 10010 Der übliche GET - Befehl mit Schlaufe.
Zeile 10020 Es wird der ASCII - Code des eingelesenen Zeichens ermittelt.
Zeile 10030 Wurde die DELETE - TASTE gedrueckt, wird das RETURN verweigert.
Zeile 10040 Wenn die Länge der Eingabe kleiner ist als definiert, wird ein RETURN verweigert.
Zeile 10050 Wenn die RETURN - Taste gedrueckt ist, wird die Subrou tine abgeschlossen.
Zeile 10060 Es werden nur grosse Buchstaben von der Subroutine angenommen.
Zeile 10070 Der String wird um das letzte Zeichen erweitert und a um 1 erhöht.
Zeile 10080 Es wird das letzte Zeichen geschrieben und ein Strich und ein Cursor rechts angehängt.
Zeile 10090 Rücksprung zum Anfang der Subroutine.
Zeile 10100 Verzweigung wenn A = 1 ist.
Zeile 10110 Rücksprung zum Anfang wenn a < 1 ist.
Zeile 10120 a wird um 1 erniedrigt.
Zeile 10130 X1$ ist gleich der Anzahl Zeichen des Gesamtstring definiert durch a.
Zeile 10140 Es wird ein Cursor rechts, ein Strich und ein Cursor rechts geschreiben.
Zeile 10150 Sprung zum Anfang der Subroutine.
Zeile 10160 Es wird ein Blank geschrieben.
Zeile 10170 Rückkehr in das Hauptprogramm.

Das Drucken des Striches hat die Aufgabe auf dem Bildschirm anzuzeigen, wo man sich gerade befindet, also ein Ersatz für den Cursor.
Die Anwendung dieser Subroutine zeigt das nachfolgende Beispiel.

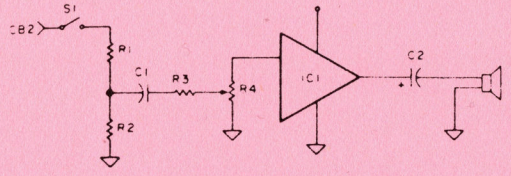
```
100 PRINT"GEBEN SIE DEN NAMEN EIN  : ":GOSUB 10000:X$=X1$
110 PRINT"GEBEN SIE DIE NUMMER EIN  : ":GOSUB 10000:X=VAL(X1$)
```

Im ersten Moment erscheint diese Routine etwas kompliziert und unnötig, wenn sie aber erst einmal eingesetzt wird und man die vielfaeltigen Möglichkeiten erkannt hat, wird sie schnell zu einem unentbehrlichen Helfer werden.

Hardware

VERSTAERKERBAUSTEIN

Nachdem wir im letzten Heft die Basisplatine und den allgemeinen Aufbau der I/O - Porterweiterung sowie die Treiberplatine für die LED - Reihe beschrieben haben, stellen wir Ihnen nun eine Verstärkerplatine vor, die zur Wiedergabe von Tönen Verwendung findet. Auch diese Platine ist so ausgelegt, das durch entsprechende Schalter sowohl eine Wiedergabe von Tönen erreicht wird, die mit dem frei schwingenden Oscillator über den USER-PORT Ausgang CB 2/M erzeugt werden, als auch komplexe Töne, die nur mittels eines D/A - Wandlers möglich sind. Hierzu ist allerdings eine zusätzliche Platine mit einem Digital-Analog-Wandler erforderlich, den wir im nächsten Heft vorstellen werden.
Die Platine enthält einen Spannungsteiler aus den Widerständen R1 und R2, die das Eingangssignal cirka um den Faktor 5 abschwächen um den Eingang des Verstärkers nicht zu übersteuern. Anschliessend wird das Signal auf ein Potentiometer gegeben, womit die Lautstärke geregelt werden kann. Der Schalter S1 zum Abschalten des Verstärkers und das Potentiometer befinden sich nicht auf der Platine sondern auf der Frontplatte des Basisgerätes. Der Verstärker selbst ist ein integrierter NF-Verstärker LM 380 mit einer Ausgangsleistung von 0.5 W bei einer Spannung von 18 V und einem Lastwiderstand von 5 Ohm. Der Lautsprecher von 5 Ohm und 0.5 W kann in das Gehäuse selbst eingebaut werden oder über einen normalen Lautsprecherstecker auf der Rückseite des Gehäuses auch extern angeschlossen werden. Kopfhörerempfang ist ebenfalls möglich. Wir glauben mit dieser Verstärkerplatine eine USER - PORT - Erweiterung vorgestellt zu haben, die vor allem durch Ihre vielseitige Anwendung besticht.
Die Platine ist unter der Nummer P 2252 zum Preis von Fr. 48.-- incl. Potentiometer und Schalter als Bausatz erhältlich oder unter der Nummer P 2253 zum Preis von Fr. 68,-- auch als Fertiggerät lieferbar. Es muss allerdings noch einmal ausdrücklich darauf hingewiesen werden, dass der Einsatz dieser Verstärkergruppe nur mit dem Basisgerät wie in Heft 2/81 der CBM - PET NEW's beschrieben, möglich ist.
Bastler, die nur an der Platinenvorlage und an der Bestückungsliste sowie dem Bestückungsplan interessiert sind, erhalten dies gegen Fr. 2.-- in Briefmarken oder 3 intern. Postantwortscheine beim Verlag SCC AG.
Damit Sie nach Fertigstellung des Gerätes nicht lange warten müssen, bis Sie auch die erforderliche Software haben, drucken wir noch zwei kleine Testprogramme ab, mit denen Sie Ihr Gerät umgehend checken können.



Schaltschema

R1	Widerstand	220 k	C1	Kondensator	.1 uF
R2	Widerstand	47 k	C2	Kondensator	470 uf
R3	Widerstand	68 k	IC	LM 380 Verstärker IC	
R4	Potentiometer	47 k	S1	Schalter	

SIRENE

```
 1000 POKE 59467,16:POKE 59466,156
 1010 FOR I=1 TO 3:FOR J=1 TO 2
 1020 POKE 59464,223:FOR K=1 TO 400:NEXT K
 1030 POKE 59464,149:FOR K=1 TO 400:NEXT K
 1040 POKE 59464,0:FOR K=1 TO 800:NEXT K
 1050 NEXT J:NEXT I:POKE 59467,0
READY.
```

LICHTER LOESCHEN

```
 100 POKE 59467,16:POKE 59466,104
 110 READ A,B:IF A=999 THEN POKE 59467,0:END
 120 POKE 59464,A:FORI=1 TO 60*B:NEXT:GOTO 110
 500 DATA 250,2,0,.2,250,1,0,.2,188,4,0,1,250,2,0,.2,188,1,0,.2
 510 DATA 149,4,0,1,250,2,0,.2,188,1,0,.2,149,3,0,.4,250,2,0,.2
 520 DATA 188,1,0,.2,149,3,0,.4,250,2,0,.2,188,1,0,.2,149,5,0,1
 530 DATA 188,2,0,.2,149,1,0,.2,125,5,0,.2,149,4,0,.2,188,3,0,.2
 540 DATA 250,5,0,.2,250,4,0,.2,250,2,0,.4,177,5,999,0
READY.
```

Für das nächste Heft haben wir für die USER - PORT - ERWEITERUNG einen Joystick vorgesehen.

CBM/PET ZUBEHOER

P 9898	CBM Umrüstsatz von Basic 1 auf Basic 2	Fr. 240.--
P 9899	CBM Umrüstsatz von Basic 2 auf Basic 4	Fr. 240.--
P 9900	CBM Umrüstsatz für Floppy von DOS 1 auf DOS 2	Fr. 240.--
P 9907	Basic Toolkit ROM TK 8000 für CBM 8032	Fr. 236.--
P 9987	Newtim Monitor/Editor-ROM zu CBM 3000/4000/8000 (unbedingt Floppy - Typ angeben !!!)	Fr. 475.--
P 9996	CBM Visicalc Deutsch ! (Floppy-Typ angeben !)	Fr. 580.--
P 9994	CBM Pascal zu Serie 3000 (Disk und ROM)	Fr. 350.--
P 9990	CBM Wordprocessor II zu Serie 3000	Fr. 280.--
P 2241	CBM TV-Modulator mit Videointerface	Fr. 155.--
P 2042	Grüner Filter zu PET 2001 mit weissem Schirm	Fr. 25.--
P 2249	8-Bit D/A-Wandler	Fr. 180.--

Programm des Monats BASIC-BASIC

HORNUSSEN

Die Liste der Spiele, die für den PET oder CBM entworfen worden sind, ist recht lang, trotzdem haben wir dieser Liste ein weiteres hinzuzufügen. Weil die Grundsubstanz aller Spiele immer die Gleiche ist, nämlich utopische Spielereien mit Raumschiffen oder ähnlichem, haben wir uns überlegt, ob wir nicht einmal eine echte Neuerung entwerfen sollten.
Als rechte Eidgenossen haben wir uns gesagt, dass das älteste Spiel der Schweiz schliesslich auch seine Berechtigung für ein CBM - Programm hat und so möchten wir Ihnen eine CBM - Version des Hornussens vorstellen.
Für unsere ausländischen Leser, die dieses Spiel nicht kennen, wollen wir das Hornussen kurz erklären.
Auf einen Bock, der aus Lehm geformt wird, legt man eine etwa 5 cm grosse Hartgummischeibe, den Hornuss. Mit einer Stahlsaite, an deren vorderem Ende sich ein Stück Rundholz, das Treff, befindet, wird der Hornuss vom Bock geschlagen. Diese Arbeit übernimmt für uns der Rechner.
Auf einem Spielfeld von ca. 20 m Breite und 180 m Länge stehen die Abtuer. Dies sind Spieler mit Holztafeln, die dem entgegenfliegenden Hornuss in den Weg gehalten oder hochgeworfen werden, um den Hornuss aufzuhalten. Wer den Hornuss möglichst früh aus der Luft heruntergeholt, erhält Punkte für seine Mannschaft. Diese alte überlieferte Sport kann mit dem Rechner gespielt werden, wobei Sie die Funktion des Abtuers übernehmen.
Die Anzahl der Hornusse, die Sie spielen wollen, können Sie vorgeben, eben so die Geschwindigkeit des Hornuss. Mit den Zahlen 0 - 9 wird die Höhe der Tafel des Abtuers bestimmt.
Sobald die gewählte Anzahl Hornusse gespielt worden ist, wird eine Bewertungstabelle ausgegeben.
Sollten Sie nach reichlicher Uebung zu dem Schluss gekommen sein, das Sie nun als Profi unter die Hornussspieler gehen können, so geben wir Ihnen gerne den nächsten zuständigen kantonalen Verein an. Nur möchten wir Sie schon jetzt warnen, Hornussen ist ein harter Sport.

Programmbesprechung

```
Zeile  100 -  200 Titel
Zeile  210 -  320 Spielerklärung
Zeile  330 -  430 Zeichnen des Spielfeldes  mittels FOR ... NEXT und PRINT
Zeile  440        Subroutine zum Zeichen des Spielers und des Bocks
Zeile  450 -  500 Eingabe der Schlagzahl und Geschwindigkeit
Zeile  520 -  530 Subroutine Erhöhung der Schlagzahl und Anzeige der
                  Schläge sowie der Treffer
Zeile  550 -  610 Subroutine Abschlag vom Bock
Zeile  620 -  680 Subroutine für einen leeren Bock
Zeile  690 -  900 Subroutine Spieler 1. Wurfhöhe
Zeile  910 -  960 Subroutine 2.Wurfhöhe
Zeile  970 - 1020 Subroutine 3.Wurfhöhe
Zeile 1030 - 1080 Subroutine 4.Wurfhöhe
Zeile 1090 - 1150 Subroutine 5.Wurfhöhe
Zeile 1160 - 1210 Subroutine 6.Wurfhöhe
Zeile 1210 - 1270 Subroutine 7.Wurfhöhe
Zeile 1280 - 1330 Subroutine 8.Wurfhöhe
Zeile 1340 - 1390 Subroutine 9.Wurfhöhe
Zeile 1400 - 1450 Subroutine 10. Wurfhöhe
Zeile 1460 - 1470 FOR ... NEXT Schleife für Schlagzahl
```

Zeile 1480 - 1490 Erzeugen einer Zufallszahl für die Flugbahn
Zeile 1500 Subroutine Anzeige des Spielstandes
Zeile 1510 - 1540 Bestimmung der Variablen für die Flugbahn
Zeile 1520 Strecke des Hornuss zur Zeit T
Zeile 1530 Höhe des Hornuss zur Zeit T
Zeile 1540 PS=Weite, PH=Höhe für die POKE Variable in Zeile 1550
Zeile 1550 PQ=alte Variable, PP=neue Variable
Zeile 1560 Subroutine zur Darstellung des Schlages
Zeile 1570 Wenn POKE-Variable = 0, setzt dieser auf Fluganfang.
Zeile 1580 Letzten Hornusspunkt löschen und neuen setzen.
Zeile 1590 bedingte Verzweigung für Eingabe der Wurfhöhe.
Zeile 1600 - 1630 Suche nach der Beendigung der Flugbahn.
Zeile 1640 Frage nach einem Treffer.
Zeile 1660 Treffervariable um 1 erhöhen
Zeile 1670 Löschen des Hornusspunktes.
Zeile 1700 Sprung für Spielanfang
Zeile 1720 - 1770 Aus der Anzahl der Schläge AS und der Treffer TR werden die Variablen BS und CS errechnet.
Zeile 1780 - 1890 Bewertung
Zeile 1900 - 1950 Frage nach erneutem Spiel und Variablen = 0 setzen.
Zeile 1960 GET Abfrage der gedrückten Zahl für die Wurfhöhe
Zeile 1970 Berechnete Verzweigung fuer Tafelhöhe
Zeile 2000 - 2150 Sprünge zu den Wurfhöhen, damit die Tafel nicht sprung haft in die Höhe schnellt, sondern langsam über die ganze Höhe fliegt.
Zeile 2160 - 2210 blinkende Trefferanzeige

H O R N U S S E N

```
100 REM CURSOR 6 TIEF U. 11 NACH RECHTS
110 PRINT"■■■■■■■■■■■■■■■■■■■■H O R N U S S E N■
120 REM CURSOR 6 TIEF U. 7 NACH RECHTS
130 PRINT:PRINT"■■■■■■■■■■■■■EIN ALTER SCHWEIZER-SPORT
140 PRINT"■■■■PROGRAMMIERT DURCH R. KONZ LUZERN 1981"
150 REM CURSOR 6 TIEF
160 PRINT:PRINT"■■■■■WUENSCHEN SIE EINE ERKLAERUNG ? (J/N)
170 GET E$:IF E$= "" THEN 170
180 IF E$="J" THEN 210
190 IF E$="N" THEN 330
200 GOTO 170
210 PRINT"■AUF EINER SEITE IST DER SCHLAEGER, AUF
220 PRINT:PRINT"DER ANDERN SEITE STEHT DER FAENGER.
230 PRINT:PRINT"DER SCHLAEGER SCHLAEGT DEN HORNUS VON
240 PRINT:PRINT"EINEM BOCK WEG, UND DER FAENGER STOPPT
250 PRINT:PRINT"DEN FLIEGENDEN HORNUS, INDEM ER IHM
260 PRINT:PRINT"EINE HOLZTAFEL IN DEN WEG HAELT, BZW.
270 PRINT:PRINT"WIRFT FUER GROESSERE HOEHEN.":PRINT
280 PRINT:PRINT"DER SCHLAEGER IST HIER DER COMPUTER, DER
290 PRINT"FAENGER SIND SIE. SIE BESTIMMEN DIE
300 PRINT:PRINT"HOEHE DER HOLZTAFEL MIT DEN ZAHLEN 0-9.
310 PRINT:PRINT:PRINT"DRUECKEN SIE NUN DIE ■SPACE■-TASTE
320 GET F$:IF F$<>" " THEN 320
330 REM   SPIELABLAUF
340 PRINT"■";
350 FOR I=1 TO 38 :PRINT"■";: NEXT
360 PRINT"■";
370 FOR J=1 TO 22 :PRINT"■ ";
380 FOR I=1 TO 38 :PRINT"■";: NEXT I
390 PRINT" ■":NEXT J
400 PRINT"■"
410 FOR I=1 TO 38 :PRINT"■";:NEXT:PRINT"■■"
```

```
420 PRINT"
430 FOR I=1 TO 38 :PRINT"";:NEXT: PRINT""
440 GOSUB 620:GOSUB 690
450 INPUT"WIE OFT WOLLEN SIE SPIELEN";G$
460 GS=VAL(LEFT$(G$,4))
470 PRINT"WELCHE GESCHWINDIGKEIT (0-9) ?"
480 GET V$:IF V$="" THEN 480
490 G=VAL(V$)/10
500 GOTO 1460
510 REM SPIELSTAND
520 AS=AS+1
530 PRINT"SCHLAEGE: "AS"                TREFFER: "TR
540 RETURN
550 S2$="
560 S2$=S2$+"
570 S2$=S2$+"
580 S2$=S2$+"
590 S2$=S2$+"
600 PRINT"S2$
610 RETURN
620 S1$="
630 S1$=S1$+"
640 S1$=S1$+"
650 S1$=S1$+"
660 S1$=S1$+"
670 PRINT"S1$
680 RETURN
690 PRINT"TAB(34);"
700 PRINT"
710 PRINT"
720 PRINT"
730 PRINT"
740 PRINT"
750 PRINT"
760 PRINT"
770 PRINT"
780 PRINT"
790 PRINT"
800 PRINT"
810 PRINT"
820 PRINT"
830 PRINT"
840 PRINT"
850 PRINT"
860 PRINT"
870 PRINT"
880 PRINT"
890 PRINT"
900 RETURN
910 REM  ZWEITE HOEHE
920 PRINT"
930 PRINT"
940 PRINT"
950 PRINT"
960 RETURN
970 REM  DRITTE HOEHE
980 PRINT"
990 PRINT"
1000 PRINT"
1010 PRINT"
1020 RETURN
```

```
1030 REM  VIERTE HOEHE
1040 PRINT"■■■■■■■■[IIII]■■    ■■";
1050 PRINT"■■■■■■■■■■■■■■■■■■■■";
1060 PRINT"■■■■■■■■■■■■■■■■■■■■";
1070 PRINT"■■■■■■■■■■■■■■■■■■■■";
1080 RETURN
1090 REM  FUENFTE HOEHE
1100 PRINT"■■■■■■■■[IIII]■■■■■■■■■■■■■■■■■■■■";
1110 PRINT"■■■■■■■■■■■■■■■■■■■■";
1120 PRINT"■■■■■■■■■■■■■■■■■■■■";
1130 PRINT"■■■■■■■■■■■■■■■■■■■■";
1140 PRINT"■■■■■■■■■■■■■■■■■■■■";
1150 RETURN
1160 REM  SECHSTE HOEHE
1170 PRINT"■■■■■■■■[IIIII]■■■■■■■■■■■■■■■■■■■■";
1180 PRINT"■■■■■■■■■■■■■■■■■■■■";
1190 PRINT"■■■■■■■■■■■■■■■■■■■■";
1200 PRINT"■■■■■■■■■■■■■■■■■■■■";
1210 RETURN
1220 REM  SIEBENTE HOEHE
1230 PRINT"■■■■■■■■[IIII]■■■■■■■■■■■■■■■■■■■■";
1240 PRINT"■■■■■■■■■■■■■■■■■■■■";
1250 PRINT"■■■■■■■■■■■■■■■■■■■■";
1260 PRINT"■■■■■■■■■■■■■■■■■■■■";
1270 RETURN
1280 REM  ACHTE HOEHE
1290 PRINT"■■■■■■■■[IIII]■■■■■■■■■■■■■■■■■■■■";
1300 PRINT"■■■■■■■■■■■■■■■■■■■■";
1310 PRINT"■■■■■■■■■■■■■■■■■■■■";
1320 PRINT"■■■■■■■■■■■■■■■■■■■■";
1330 RETURN
1340 REM  NEUNTE HOEHE
1350 PRINT"■■■■[IIII]■■■■■■■■■■■■■■■■■■■■";
1360 PRINT"■■■■■■■■■■■■■■■■■■■■";
1370 PRINT"■■■■■■■■■■■■■■■■■■■■";
1380 PRINT"■■■■■■■■■■■■■■■■■■■■";
1390 RETURN
1400 REM  ZEHNTE HOEHE
1410 PRINT"■■■■[IIII]■■■■■■■■■■■■■■■■■■■■";
1420 PRINT"■■■■■■■■■■■■■■■■■■■■";
1430 PRINT"■■■■■■■■■■■■■■■■■■■■";
1440 PRINT"■■■■■■■■■■■■■■■■■■■■";
1450 RETURN
1460 REM -------SPIELABLAUF-------
1470 FOR J=1 TO GS
1480 Z=INT(8*RND(0)+12)
1490 V=400*Z:A=70:T=0:PP=0:V=0
1500 GOSUB 510
1510 T=T+.25+G
1520 S=INT(V*T*(SIN(A)/180*π))
1530 H=INT(V*T*(COS(A)/180*π)-9.81/2*T↑2)
1540 PS=INT(4*S/100):PH=INT(H/20)
1550 PQ=PP:PP=33653+PS-PH*40
1560 GOSUB 620
1570 IF PQ=0 THEN PQ=33653
1580 POKE PQ,32:POKE PP,87
1590 IF V=0 THEN GOSUB 1960
1600 FOR Q=1 TO G*10
1610 IF PEEK(PP+INT(Q/2+.5))<>32 GOTO 1640
1620 NEXT Q
1630 IF PEEK(PP-40)=32 GOTO 1510
```

```
 1640 IF PEEK(PP+INT(Q/2+.5))=245 GOTO 1660
 1650 GOTO 1690
 1660 TR=TR+1
 1670 POKE PP,32
 1680 PP=PP+INT(Q/2+.5)-1:I=32956:GOTO 2160
 1690 POKE PP,32
 1700 GOSUB 690:GOSUB 550
 1710 NEXT J
 1720 PRINT"■■■■■SIE HABEN VON"AS"SCHLAEGEN"
 1730 PRINT"■"TR"TREFFER GEMACHT !"
 1740 PRINT"■   ■                                    ■"
 1750 PRINT"    ■      BEWERTUNGS-SKALA              ■"
 1760 PRINT"    ■                                    ■"
 1770 BS=INT(AS/2-AS/10)*2:CS=INT(AS/5)
 1780 IF TR>=BS THEN PRINT"■";
 1790 PRINT"■"BS"■■ BIS"AS"■■ TREFFER : SEHR GUT"
 1800 IF TR>=BS-CS AND TR<BS THEN PRINT"■";
 1810 PRINT"■"BS-CS"■■ BIS"BS-1"■■ TREFFER : GUT"
 1820 IF TR>=BS-2*CS AND TR<BS-CS THEN PRINT"■";
 1830 PRINT"■"BS-2*CS"■■ BIS"BS-CS-1"■■ TREFFER : MUSS NOCH UEBEN"
 1840 IF TR>=BS-3*CS AND TR<BS-2*CS THEN PRINT"■";
 1850 PRINT"■"BS-3*CS"■■ BIS"BS-2*CS-1"■■ TREFFER : MUSS NOCH VIEL
 1860 IF TR<BS-3*CS THEN PRINT"■";                          UEBEN"
 1870 PRINT"■■■0 BIS"BS-3*CS-1"■■ TREFFER : GEBEN SIE'S"
 1880 IF TR<BS-3*CS THEN PRINT"■";
 1890 PRINT"■■■■■■■■■■■■■■■■■■■■■■■AM BESTEN AUF"
 1900 PRINT"■■WOLLEN SIE NOCHMAL SPIELEN ? (J/N)"
 1910 GETS$:IF S$="" GOTO 1910
 1920 IF S$="J" THEN TR=0:AS=0:GOTO 160
 1930 IF S$="N" THEN END
 1940 GOTO 1910
 1950 END
 1960 GET Y:IF Y=0 THEN RETURN
 1970 ON Y GOTO 1980,2000,2020,2040,2060,2080,2100,2120,2140
 1980 GOSUB 690:GOSUB 910
 1990 RETURN
 2000 GOSUB 1980:GOSUB 970
 2010 RETURN
 2020 GOSUB 2000:GOSUB 1030
 2030 RETURN
 2040 GOSUB 2020:GOSUB 1090
 2050 RETURN
 2060 GOSUB 2040:GOSUB 1160
 2070 RETURN
 2080 GOSUB 2060:GOSUB 1220
 2090 RETURN
 2100 GOSUB 2080:GOSUB 1280
 2110 RETURN
 2120 GOSUB 2100:GOSUB 1340
 2130 RETURN
 2140 GOSUB 2120:GOSUB 1400
 2150 RETURN
 2160 FOR I=1 TO 3
 2170 POKE PP,42
 2180 FOR N=0 TO 50:NEXT N
 2190 POKE PP,87
 2200 NEXT I
 2210 GOTO 1690
READY.
```

BILDSCHIRMAUSDRUCK MUEHLESPIEL

Im Heft 1/81 haben wir ein Mühleprogramm abgedruckt, das zwar hervorragend in seiner Spielstrategie ist, mit dem aber nur manipuliert werden konnte, wenn man ein entsprechendes Mühlebrett vor sich liegen hat, da es keinen Bildschirmausdruck ergibt. In der gleichen Zeitschrift haben wir nach Lesern gefragt, die bereit wären, das Programm gerade um diesen Bildschirmausdruck zu erweitern. Herr P. Somm aus Weinfelden hat uns die dazugehörige Programmerweiterung zugesandt. Gleichzeitig hat auch Herr Kast aus Balgach sein Mühlespiel noch etwas revidiert. Diese kleine Aenderung, wie auch die Erweiterung um den Bildschirmausdruck sehen Sie im nachfolgenden Programm.
Das eigentliche Mühleprogramm kann also sehr leicht ergänzt werden, da nur die Zeilen aufgeführt sind, die gegenüber dem Originalprogramm Aenderungen aufweisen. Sofern Sie Ihr Mühleprogramm durch unseren Kassettenservice bestellen, so enthält er selbstverständlich bereits diese Ergänzungen.
Vom Aussehen des Spielfeldes auf dem Bildschirm können Sie sich gemäss dem folgenden Hardcopyausdruck leicht überzeugen, die Verschiebung Länge - Breite ergibt sich aus der verschiedenenartigen Matrix des Bildschirms 8*8 gegenüber dem Drucker 5 * 7.

PROGRAMMAENDERUNG

```
    3 DIMP%(52):GOTO10:REM VERSION #3,  1. TEIL
  600 M=0:ONUPGOTO630,650,670,690,730,810,850,910
  830 N=1:IZ=RND(-TI)
  840 UP=7:IZ=0
  850 IZ=IZ+1:I=INT(8*RND(1))+2:J=INT(3*RND(1))*20:P=I+J
  860 IFP%(P)=0THENM=8:RETURN
  870 IF 20THEN850
  900 UP=8:I=1:J=0
  910 I=I+1:IFI>9THENI=2:J=J+20:IFJ>40THEN940
  920 P=I+J:IFP%(P)=0THENM=8:RETURN
  930 GOTO 910
  940 IF N 3THENN=N+1:GOTO840
  950 RETURN
 3750 B=P%(I+F1)+P%(I+F1+20)+P%(I+F1+40)
 3755 IFN=1ANDB=27ORN=1ANDB=11THENRETURN
 3760 IFN=2ANDB=11THENRETURN
 3860 B=P%(A-1)+P%(A+1)
 3870 IFN=1ANDB=18ORN=1ANDB=2THENRETURN
 3880 IFN=2ANDB=2THENRETURN
 3890 PN=P:PA=A:GOTO2600
 3900 B=P%(P+2*F1)+P%(P+3*F1)
 3910 IFN=1ANDB=18ORN=1ANDB=2THEN3810
 3920 IFN=2ANDB=2THEN3810
 3930 GOTO 3770
READY.
```

BILDSCHIRMAUSDRUCK

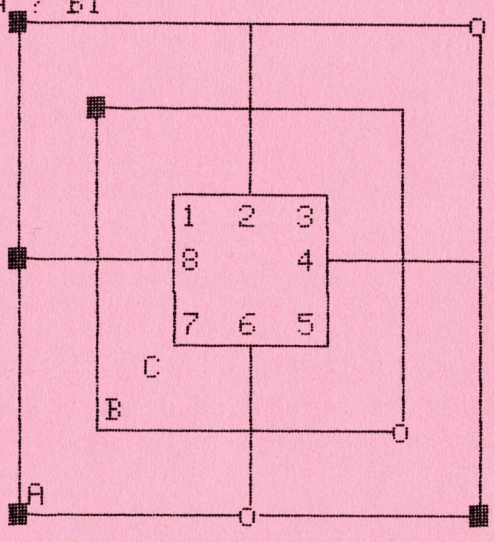
```
1 REM ALLE 'GOSUB 5' UND 'GOSUB 20040' MUESSEN GELOESCHT WERDEN !!!
4 DIMP%(52)
5 GOSUB30000
10 PRINT"■"TAB(13)"■■■■■■■WER BEGINNT ?"
11 PRINTTAB(13)"■■■(1) DU":PRINTTAB(13)"■■(2) PET"
12 GETW$:IFW$<>"1"ANDW$<>"2"THEN12
13 W=VAL(W$):PRINT"■":GOSUB40000
110 IFA1<3THEN GOSUB41000:PRINT"■■DU HAST VERLOREN.":END
```

```
 120 POKE14,1:GOSUB41000:INPUT"■DEIN ZUG (VON-NACH) ";Z$:POKE14,0:
 121 PRINT:P$=LEFT$(Z$,2):W=1 GOSUB9800:IFF1=1THEN120       GOTO31000
 145 GOSUB42000
 240 GOSUB41000:PRINT"■■HILFE - ICH BIN EINGEKLEMMT !":END
 400 POKE14,1:GOSUB41000:INPUT"■DEINE POSITION ? ";P$:POKE14,0:
 401 PRINT:W=0:GOSUB9800:IFF1=1THEN400                    GOTO32000
 405 GOSUB 43000
 450 POKE14,1:GOSUB41000:INPUT"■WELCHEN STEIN NIMMST DU MIR WEG ? "
GOTO34000                                                ;P$:POKE14,0:
 451 GOSUB44000:PRINT
 610 GOSUB41010:PRINT"■■■ICH SUCHE...":PRINT"I";
 9360 GOSUB41020:PRINT"■■■■ICH NEHME ";P$:GOSUB44000:RETURN
 20080 IFA1<3THENGOSUB41000:PRINT"■■DU HAST VERLOREN.":END
 20090 GOSUB41000:INPUT"■DEIN ZUG (VON-NACH) ";Z$:P$=LEFT$(Z$,2):
 20091 W=1:GOSUB21880:IFF1=1THEN20090                    GOTO33000
 20105 GOSUB 42000
 20170 GOSUB41000:PRINT"■■ICH GRATULIERE. DU HAST GEWONNEN.":END
 20280 GOSUB41010:PRINT"■■■ICH SUCHE...":PRINT"I";
 21770 GOSUB41020:PRINT"■■■■ICH NEHME ";P$:GOSUB44000:RETURN
 21800 GOSUB41010:PRINT"■■■EIN ZUG ":Z$:GOSUB45000:IFF3=1THENGOSUB
 29999 REM TEST AUF '@'                                  21410
 30000 PRINT"■■SOLL DAS SPIELFELD MIT DER CODIERUNG
 30010 PRINT"■VERSEHEN WERDEN ?  (J/N)
 30020 PRINT"■■■WAEHREND DES SPIELS KANN DIE CODIERUNG
 30030 PRINT"■MIT '@' EIN- UND AUSGESCHALTET WERDEN.
 30040 GETAN$:IFAN$<>"J"ANDAN$<>"N"THEN30040
 30050 DIMBL(12):DIMCO(12):FORQ9=1TO11:READBL(Q9):READCO(Q9):NEXT
 30060 DATA33151,49,33154,50,33157,51,33237,52,33357,53,33354,54,
 30070 DATA55,33231,56,33663,1,33507,2,33429,3              33351,55
 30080 RETURN
 31000 IFLEFT$(P$,1)<>"@"ANDLEFT$(Z$,1)<>"@"THENGOTO121
 31010 IFAN=1THENFORQ9=1TO11:POKEBL(Q9)+40,32:NEXT:AN=0:GOTO120
 31020 FORQ9=1TO11:POKEBL(Q9)+40,CO(Q9):NEXT:AN=1:GOTO120
 32000 IFLEFT$(P$,1)<>"@"ANDLEFT$(Z$,1)<>"@"THENGOTO401
 32010 IFAN=1THENFORQ9=1TO11:POKEBL(Q9)+40,32:NEXT:AN=0:GOTO400
 32020 FORQ9=1TO11:POKEBL(Q9)+40,CO(Q9):NEXT:AN=1:GOTO400
 33000 IFLEFT$(P$,1)<>"@"ANDLEFT$(Z$,1)<>"@"THENGOTO20091
 33010 IFAN=1THENFORQ9=1TO11:POKEBL(Q9)+40,32:NEXT:AN=0:GOTO20090
 33020 FORQ9=1TO11:POKEBL(Q9)+40,CO(Q9):NEXT:AN=1:GOTO20090
 34000 IFLEFT$(P$,1)<>"@"ANDLEFT$(Z$,1)<>"@"THENGOTO451
 34010 IFAN=1THENFORQ9=1TO11:POKEBL(Q9)+40,32:NEXT:AN=0:GOTO450
 34020 FORQ9=1TO11:POKEBL(Q9)+40,CO(Q9):NEXT:AN=1:GOTO450
 39999 REM ZEICHNEN DES SPIELFELDES
 40000 QZ=14:PRINT"■■";
 40004 REM            1234567890123456789012345
 40005 PRINTTAB(QZ)"┌─────────┬─────────┐
 40010 PRINTTAB(QZ)"│         │         │
 40020 PRINTTAB(QZ)"│         │         │
 40030 PRINTTAB(QZ)"│         │         │
 40040 PRINTTAB(QZ)"│   ┌─────┼─────┐   │
 40050 PRINTTAB(QZ)"│   │     │     │   │
 40060 PRINTTAB(QZ)"│   │     │     │   │
 40070 PRINTTAB(QZ)"│   │     │     │   │
 40080 PRINTTAB(QZ)"│   │   ┌─┼─┐   │   │
 40090 PRINTTAB(QZ)"│   │   │   │   │   │
 40100 PRINTTAB(QZ)"│   │   │   │   │   │
 40110 PRINTTAB(QZ)"├───┼───┘   └───┼───┤
 40120 PRINTTAB(QZ)"│   │   │   │   │   │
 40130 PRINTTAB(QZ)"│   │   │   │   │   │
 40140 PRINTTAB(QZ)"│   │   │   │   │   │
 40150 PRINTTAB(QZ)"│   │   └─┬─┘   │   │
 40160 PRINTTAB(QZ)"│   │     │     │   │
 40170 PRINTTAB(QZ)"│   │     │     │   │
```

```
40180 PRINTTAB(QZ)"┃    ┃        ┃        ┃    ┃
40190 PRINTTAB(QZ)"┃    ┗━━━━━━━━┻━━━━━━━━┛    ┃
40200 PRINTTAB(QZ)"┃                            ┃
40210 PRINTTAB(QZ)"┃                            ┃
40220 PRINTTAB(QZ)"┃                            ┃
40230 PRINTTAB(QZ)"┗━━━━━━━━━━━━━━━━━━━━━━━━━━━━┛";
40235 IFAN$="J"THENAN=1:FORQ9=1TO11:POKEBL(Q9)+40,CO(Q9):NEXT
40239 REM EINLESEN DER ECKKOORDINATEN
40240 DIMPL(30):DIMOW(30)
40250 FORQ9=0TO2:FORQ8=1TO9
40260 READPL(10*Q9+Q8):READOW(10*Q9+Q8)
40270 NEXT:NEXT
40280 DATA32782,112,32794,114,32806,110,33246,115,33726,125,33714
40281 DATA113,33702,109,33222,107
40290 DATA32946,112,32954,91,32962,110,33242,91,33562,125,33554
40291 DATA91,33546,109,33226,91
40300 DATA33110,112,33114,113,33118,110,33238,107,33398,125,33394
40301 DATA114,33390,109,33230,115
40999 REM LOESCHEN DER ALTEN ANWEISUNG
41000 PRINT"■                                          ":RETURN
41010 PRINT"■■■■                    "
41020 PRINT"■■■■■■                    ":RETURN
41999 REM SETZEN UND BEWEGEN EINES STEINES (ICH/ZIEHEN)
42000 Q2=160:Q6$=Z$:GOTO45010
42999 REM SETZEN UND BEWEGEN EINES STEINES (ICH/SETZEN)
43000 Q2=160:Q6$=P$:GOTO45010
43999 REM LOESCHEN EINES STEINES
44000 Q6$=P$:GOSUB46000
44010 POKEPL(Q7)+40,OW(Q7):RETURN
44999 REM SETZEN UND BEWEGEN EINES STEINES (PET)
45000 Q2=87:Q6$=Z$
45010 GOSUB46000
45020 IFAS>9THEN45100
45030 POKEPL(Q7)+40,Q2:RETURN
45100 POKEPL(Q7)+40,OW(Q7)
45110 POKEPL(Q3)+40,Q2:RETURN
45999 REM BERECHNEN DER BILDSCHIRMADRESSE
46000 Q5$=LEFT$(Q6$,2):Q4$=MID$(Q6$,4,2)
46010 Q7=VAL(RIGHT$(Q5$,1))+10*(ASC(LEFT$(Q5$,1))-65)
46020 IFQ4$=""THENQ3=0:RETURN
46030 Q3=VAL(RIGHT$(Q4$,1))+10*(ASC(LEFT$(Q4$,1))-65)
46040 RETURN
```

```
DEINE POSITION ? B1

ICH SUCHE ...
```



Maschinensprache

ASSEMBLER — PROGRAMMIERUNG
FORTSETZUNG

Heute wollen wir Ihnen in groben Zügen schildern, wie eine Assemblersprache aufgebaut, auf welche Details zu achten ist und wie ein Assembler gesteuert wird.
Wenn wir ein Vorhaben z.B. Berechnungen durchführen oder Steueraufgaben einem Mikrocomputer übertragen wollen, so müssen wir uns mit ihm in seiner Sprache unterhalten, denn nur dies versteht er. Die Sprache wird uns beim Kauf des Mikroprozessors (Zentraleinheit vom Mikrocomputer) mitgegeben, resp. aufgezwungen. Sprachelemente sind es meist wenige, 50 - 150 je nach Mikroprozessortyp.
Solche Sprachelemente bezeichnet man auch als Maschinenbefehl. Jeder Maschinenbefehl wird durch ein ganz bestimmtes Wort dargestellt, d.h. ein solches Codewort setzt sich aus mehreren Bits zusammen. Ein Bit ist bekanntlich die kleinste Einheit um einen Zustand zu charakterisieren. Ein elektrischer Zustand kann stromdurchflossen oder nicht, elektrisch leitend oder isolierend sein. Ein Bit hat also zwei Zustände und in der Elektronik stellt man das Bit als 1 oder 0 dar. Die heute am meisten verbreiteten Mikroprozessoren arbeiten mit 8 Bits, zusammengefasst in einem Codewort. Dann spricht man von einem Byte. Dies gilt auch für die CPU 6502 der cbm Computer.
Aus den 8 Bit mit ihren zwei Zuständen könnte man 2 hoch 8 Kombinationen herstellen. Dies ergäbe 256 verschiedene Maschinenbefehle, die in einem Byte codiert werden könnten. Nicht alle Kombinationen werden ausgenützt. Bei der CPU 6502 sind nur 56 Maschinenbefehle bekannt, einige sind aber mehrfach codiert. Man denke dabei nur an die verschiedenen Adressierungsarten für denselben Befehl.
Der Mensch denkt auf die Dauer nicht gerne abstrakt, er hat's lieber praktisch. So auch bei den Maschinenbefehlen. Wenn man bedenkt, dass wir die vielen Kombinationen der Maschinenbefehle als Bitpakete merken müssten, so würden wir fast wahnsinnig. Man hat sich deshalb geeinigt, die Befehle nicht im Dualsystem, sondern im Hexadezimalsystem darzustellen. Dabei ist man noch weiter gegangen: die Maschinenbefehle werden nicht nur durch Hexadezimalzahlen sondern durch verständliche und logische Bezeichnungen dargestellt. Jetzt sind wir bei den mnemonischen Codebezeichnungen angelangt. Die Bezeichnungen können wir Menschen uns leichter einprägen. Wir verständigen uns in der Mikrocomputerei also mit mnemonischen Begriffen. An dieser Stelle tritt der Assembler in Aktion. Er vollzieht nämlich nichts anderes als unsere Bezeichnungen in den Maschinencode zu transponieren.

Beispiel:

Formale Bedeutung:	mnemonic	hexadecimal	Binär
setze das X-Register auf 7	LDX 7	A2 07	10100010 00000111

Nun ist zur Einleitung genug gesagt, wir wollen ja hier nicht die Maschinenbefehle kennenlernen, dazu gibt es genug Bücher. Wir wollen zeigen wie die Befehle angeordnet sein müssen, damit sie der Assembler richtig versteht und verarbeiten kann.
Die Gesamtheit aller Maschinenbefehle nennt man die Assembler-Sprache. Die einzelnen Sprachsätze sind in den Sourcezeilen (Quellencodezeilen) abgelegt. Wie in jederSprache, so gibt es auch hier bestimmte Regeln.

Beim Assembler gibt es im wesentlichen drei Sourcezeilentypen: die Instruktions-, die Direktiv- und die Steuerzeilen (engl. types of source lines: instructions, directives and controls). Hier wollen wir nur die ersten beiden besprechen, die Steuerbefehle sind in jedem Assembler operator's manual beschrieben.

In den Instruktionszeilen sind die mnemonischen Codes zu finden. Die Direktiven werden in separaten Zeilen geschrieben und können entweder an den Assembler Anweisungen geben, oder es können damit Speicherplätze reserviert oder sogar mit Werten belegt werden. Diese Direktiven nennt man Pseudo-Instruktionen. Sie sind also nicht im Instruktionsrepertoir des Mikroprozessors enthalten, sondern als Ergänzung oder Erweiterung zu betrachten. Diese Pseudo-Opcodes können die verschiedensten Bezeichnungen haben und sind assemblerspezifisch. Nicht jeder 6502 Assembler versteht die gleichen Direktiven. Hier ein paar Beispiele von Pseudo-Instruktionen:

BYTE Mit dieser Anweisung legt der Assembler einzelne Bytes (8 Bit) fest. Durch Komma abgetrennt können mehrere hintereinander geschrieben werden.

```
00 01 04        BYTE  0,1,4,16,64
10 40
```

WORD Damit werden Datenworte festgelegt. Weil diese Reservierung 2 Bytes (16 Bits) ablegt, wird vielfach auch von Zeiger- oder Pointerreservierung gesprochen. Sprung- und Pointertabellen werden damit gebildet. Der Assembler legt von einem 16 Bit Wort immer zuerst das lower Byte und erst dann das higher Byte ab.

```
1D 01           WORD  285
00 80           WORD  $8000
```

ASCI Mit dieser Anweisung kann ein ganzer Text reserviert, resp. definiert werden. Die einzelnen Charakter werden nach dem ASCII-Format festgelegt. Ein Text braucht nur durch zwei Zeichenbegrenzer angegeben werden:

```
ASCI  'S 18 KURT'
ASCI  'FAKTOR E ?'
```

Will man einer symbolischen Konstanten einen Wert zuweisen, so kann dies mit SET oder EQU oder einfach mit = geschehen, z.B.

```
ROT = 1, GELB SET 2.
```

Damit ist die Farbe ROT zum Zahlenträger geworden.
Auf diese Art können aber auch Zeropage-Variablen definiert werden.

Eine ganz andere Direktive ist zum Beispiel NOGEN oder NOBJ, je nach Assembler. Diese erzeugt nicht direkt Codes, sondern gibt an den Assembler die Anweisung ab dieser und für die folgenden Sourcezeilen keinen Maschinencode mehr zu erzeugen, bis diese Anweisung durch eine andere Direktive wieder aufgehoben wird.

Natürlich gibt es noch weitere solche Pseudo-Instruktionen, je nach Mächtigkeit vom Assembler. Die genauen Ausführungen sind in jedem Assemblerhandbuch zu finden. Dasselbe gilt auch für die weiteren Ausführungen.

Nun kommen wir noch zu den Assemblerinstruktionen. Wie Sie wissen, werden diese in den Instruktionszeilen oder auch Sourcezeilen abgelegt. Diese Sourcezeilen haben ein ganz bestimmtes Format und sind in fast allen Assemblern gleich. Das

heisst, es gibt eine Vorschrift wie die Instruktionen in der Zeile angeordnet sein müssen. Dies gilt auch für die Direktiven. Das Sourcezeilenformat wird in die folgenden 4 Felder unterteilt: Label, Opcode, Operand, Comment.
Im ersten Feld sind Labels oder Symbols definiert. Diese müssen immer linksbündig und mit einem Buchstaben beginnend geschrieben werden.
Im Opcodefeld werden alle Assemblerinstruktionen gesetzt. Beim 6502 kennen wir genau 56 verschiedene und alle sind dreistellig (3 Buchstaben). Damit der Assembler die Instruktionen von den Labels unterscheiden kann, müssen die Opcodes mit mindestens einem Leerzeichen rechts eingerückt werden. Am besten schreibt man alle Opcodes untereinander, eingerückt um das ganze Labelfeld.
Im Operandenfeld wird angegeben mit welcher Memorylocation eine Operation ausgeführt wird. Denken Sie dabei z.B. an die verschiedenen Adressierungsarten beim Maschinenbefehl LDA. Aber nicht jeder Befehl hat auch einen Operanden: DEX, CLC usw. Operanden werden vom Opcode durch ein Leerzeichen getrennt.
Im letzten Feld können wir noch Kommentar einfügen. Dieser Kommentar bezieht sich meist auf die unmittelbar in der Nähe stehenden Maschineninstruktionen.
Wichtig: Ein Kommentar soll auf keinen Fall die Instruktionen erklären, sondern was mit den Instruktionen ausgeführt wird.

Haben Sie also ein Problem in Sourcezeilen zerlegt, logisch aneinandergereiht und übergeben Sie das Programm einem Assembler zur Uebersetzung, so wird dieser meistens vorne noch drei weitere Felder ansetzen.
Im ersten Feld ist die Memoryadresse dargestellt, wo die Maschinenbefehle abgespeichert sind.
Im zweiten Feld sind dann die eigentlichen Maschinencodes zu finden, zugehörig zur mnemonischen Sourceinstruktion. Memoryadressen und Maschinencodes werden meistens in hexadezimaler Form dargestellt.
Als dritte Spalte findet man noch die Zeilennumerierung.
Damit Sie das ganze auch noch vertiefen können, haben wir ein komplettes Beispiel abgedruckt. SCREENCHAR: Den Bildschirm vom CBM mit verschiedenen Zeichen vollschreiben.

```
* CBM * ASSEMBLER 6502 *                    SCREENCHAR.LST        PAGE    1

0384                1  *=900
0384                2  !  ------------------------------------------------  !
0384                3  !  DIESES BEISPIEL ZEIGT WIE MIT 2 INEINANDER-    !
0384                4  !  LIEGENDEN SCHLEIFEN (LOOP)  5 * 200  CHAR-      !
0384                5  !  AKTER IN DEN BILDSCHIRM GESCHRIEBEN WERDEN.     !
0384                6  !  ------------------------------------------------  !
0384                7  ;
0021                8  ZPT = 33     ;ZEROPAGE-POINTER FUER INDIREKTE
0384                9                ; ADRESSIERUNG (2 BYTES: LOW, HIGH)
0384               10  ;
0384  ADAB03       11  START LDA SADR      ;DIE BILDSCHIRMADRESSE IN DEN
0387  8521         12        STA ZPT       ; ZEROPAGE-POINTER SETZEN
0389  ADAC03       13        LDA SADR+1
038C  8522         14        STA ZPT+1
038E  A205         15        LDA #5        ;1. SCHLEIFENZAEHLER SETZEN
0390  A000         16  LOOPX LDY #0         ;2. SCHLEIFENZAEHLER SETZEN
0392  98           17  LOOPY TYA            ;DEN CHARAKTER 0..200 IN DEN
0393  9121         18        STA (ZPT),Y   ; BILDSCHIRM AUSGEBEN
0395  C8           19        INY           ;DEN 2. ZAEHLER ERHOEHEN UND
0396  C0C8         20        CPY #200      ; PRUEFEN OB ENDWERT ERREICHT
0398  D0F8         21        BNE LOOPY     ;SPRINGE WENN NEIN
```

```
039A  18        22          CLC
039B  A521      23          LDA ZPT       ;DEN ZEROPAGE-POINTER UM 200
039D  69C8      24          ADC #200      ; ERHOEHEN (BILDSCHIRMADRESSE)
039F  8521      25          STA ZPT       ; (ALS DOPPELBYTE ADDITION
03A1  A522      26          LDA ZPT+1     ; AUSFUEHREN)
03A3  6900      27          ADC #0
03A5  8522      28          STA ZPT+1
03A7  CA        29          DEX           ;DEN 1. ZAEHLER ZURUECKNEHMEN
03A8  D0E6      30          BNE LOOPX     ; UND SPRINGEN WENN NICHT 0
03AA  60        31          RTS           ;DAS PROGRAMM VERLASSEN
03AB            32  ;
03AB  0080      33  SADR    .WORD 32768   ;BILDSCHIRM STARTADRESSE
03AD            34  ;
03AD            35  .END

      5 SYMBOLS    0 ERRORS    41 BYTES


LOOPX   0390      LOOPY   0392      SADR    03AB      START   0384
ZPT     0021

--- Listefile 'SCREENCHAR' mit dem Assembler erstellt. ---
```

In der heutigen Vorstellung wollen wir Ihnen das Entwicklungspaket "CBMASS 65" kurz erläutern. Dieses Programmpaket ist für Floppy-Disk Betrieb geschrieben, sowohl für CBM als auch für Computer Think Floppy. Es kann also auf dem alten wie auf dem neuen PET gearbeitet werden.
Da die Programme mit Floppys als Massenspeicher arbeiten, ist eine sehr schnelle Maschinenprogrammentwicklung möglich. Die Handhabung ist komfortabel, die Bedienungsanleitungen sind in Deutsch, ausführlich und leicht verständlich.
Das Entwicklungspaket umfasst die Programme: EDITOR, ASSEMBLER, LISTER, HEX-LOADER und MPDATA.

Um den Quellencode für die Maschinenprogramme zu erstellen, haben Sie mit dem EDITOR die folgenden Möglichkeiten:

```
( Hauptmenue )

LIST      L    Auflisten des Textes
INPUT     I    Eingabe von neuen Textzeilen
DELETE    D    Löschen von Textzeilen
CHANGE    C    Korrigieren von Textzeilen
READ      R    Ganzer Text lesen ab Massenspeicher
WRITE     W    Ganzer Text auf Massenspeicher schreiben
PRINT     P    Ganzer Text ausdrucken auf Printer
MERGE     M    Texte ab Massenspeicher zusammenfügen
DIR       Q    Inhaltsverzeichnis der Texte vom Massenspeicher
EXIT      X    Den Texteditor verlassen
```

Die genauen Ausführungen dazu sind in der Bedienungsanleitung zu finden.

Mit dem ASSEMBLER übersetzen Sie den Quellentext. Dabei können Sie aus den drei Möglichkeiten: Listfile, Hexfile und Memorycode kombinatorisch auswählen. Nach dem Einlesen des Sourcefile fragt der Rechner nacheinander:

```
DO YOU WANT A LISTFILE       (Y/N) ?
DO YOU WANT A HEXFILE        (Y/N) ?
DO YOU WANT A MEMORYCODE     (Y/N) ?
```

Anschliessend können Sie die erzeugten Files mit dem Programm LISTER ansehen und auf Wunsch mit dem Kommando "Print" ausdrucken.

Möchten Sie auch noch den Maschinencode dokumentarisch festhalten, so können Sie dies mit dem Programm HEXLOADER tun. Lesen Sie den Hexfile ein und geben Sie diesen mit dem Kommando "Print" auf den Drucker aus.

```
* CBM * ASSEMBLER  6502 *                SCREENCHAR.HEX          PAGE    1

STARTADR :   $0384          900
ENDADR   :   $03AC          940
BYTES    :                   41

0384 ADAB03852104D4C0038522A52A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A
03A4 0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A0A
```

Das HEXLOADER Programm hat im übrigen noch weitere Funktionen:

READ	R	Lesen eines Hexfile ab Floppy
LIST	L	Auflisten des Hexfile
PRINT	P	Ausdrucken des Hexfile
POKE	O	Den Hexcode in den Rechnerspeicher geben
PEEK	E	Memory auslesen und auflisten
DIR	Q	Directory vom Floppy holen und ausgeben
EXIT	X	Das Programm verlassen

MPDATA ist ein Hilfsprogramm und wandelt Maschinenprogramme aus ihrem Rechner automatisch in BASIC-Programme um. Dabei werden die einzelnen Codes in DATA-Statements abgelegt.

Mit diesem Beitrag wäre der Kurs über die Assemblerprogrammierung eigentlich abgeschlossen. Wir werden in den nächsten 3 Heften komplette Assemblerprogramme mit ausführlichen Kommentaren dazu abdrucken, wobei wir den EDASS und COMMODORE ASSEMBLER verwenden. Wir hoffen Ihnen hiermit einen guten Einstieg in diese komplexe Materie zu geben.
Nach Abschluss dieses Kapitels über Assemblerprogrammierung werden wir uns intensiev mit PASCAL auseinandersetzen, das inzwischen über den entsprechenden Compiler auch für den CBM zugänglich geworden ist.

Neuheiten

BASIC — ERWEITERUNG

Auf dem Gebiet der Programmierhilfen können wir Ihnen heute drei Neuheiten vorstellen.

Zuerst wären hier zwei verschiedene Disk mit Basicerweiterungen zu nennen.

1. BASIC - PROGRAMMIERHILFE FUER CBM 3001

Mit diesem Maschinenprogramm stehen dem Programmierer 18 zusätzliche Befehle zur Verfügung. Das Programm belegt 2.5 kByte im oberen Teil des Rechners und ist vor der Ueberschreibung mit Basic geschützt.
Das Programm enthält den üblichen Befehlssatz des Toolkit wie FIND, DELETE (KILL), RENUMBER, TRACE, STEP, OFF, AUTO, HELP, DUMP (VAR) usw.
Darueberhinaus enthält das Programm noch folgende Zusatzbefehle:

MERGE Lädt ein Programm ab Floppy und hängt es an ein im Speicher befindliches Programm an.

DCMD Ausgabe eines Befehls auf Floppy (zB. Initialisieren)

DST Liest den Floppy - Fehlerstatus und druckt ihn auf dem Bildschirm aus.

DIR Auflisten des Floppy - Direktorys

PLIST Liest ein Programm ab Floppy und listet es auf dem Bildschirm auf. Ein im Speicher befindliches Programm wird nicht gelöscht.

TYPE Die gleiche Funktion wie PLIST aber Auflisten auf Printer mit Primär adresse 4.

SLIST Liest ein sequentielles File ab Floppy und listet es auf dem Bildschirm auf.

EXEC Laden und Starten eines Programms ab Floppy.

BYE Ausschalten der Zusatzbefehle. Das Programm bleibt im Speicher und kann jederzeit neu initialisiert werden.
Die Initialisierung erfolgt mit SYS 64721. Das Laden des Programms ab Disk wird mit LOAD"*",8 vorgenommen. Alle Floppy - Befehle beziehen sich auf die Primär-adresse 8.

2.BASIC - SUPERERWEITERUNG

Dieses Programm beinhaltet die gleichen Funktionen wie sie unter der BASIC Erweiterung beschrieben sind.
Zusätzlich stehen noch folgende Befehle zur Verfügung.

*** + *** ruft das Hilfsprogramm CSHORT auf, mit dem alle Befehle in Kurzform eingegeben werden können zB., durch Drücken der $-Taste erscheint LIST auf dem Bildschirm.

*** * *** ruft das Hilfsprogramm CSHORT auf und zusätzlich REPEAT für alle Tasten.

*** - *** CSHORT und REPEAT werden ausgeschaltet.

Als sehr wichtige Neuerung steht mit *** PRINT 'xx', Ausdruck 1, Ausdruck 2, ***
der PRINT USING Befehl zur Verfügung, er erlaubt die formatierte Ausgabe von Daten.

*** !format *** dient der Definition des Formats und muss als erster in der Zeile stehen.
Das Format selbst setzt sich aus den Formatzeichen 9,Z,.,S,M,A zusammen. Es ist auf dem Bildschirm nun eine formatierte Ausgabe möglich, die bisher nur mit dem CBM - Printer realisierbar war.

Auch hier erfolgt das Laden des Programms mit LOAD"*",8 und die Initialisierung mit SYS 32512 für die 32 - k Version und mit SYS 16128 für die 16 k - Version. Das Programm belegt 6144 Byte im oberen RAM Bereich und ist gegen Ueberschreibung mit BASIC geschützt.

Die Programme BASIC ERWEITERUNG und BASIC SUPER ERWEITERUNG sind für die 16 und 32 k - Version lieferbar. Diese Angabe muss bei der Bestellung gemacht werden. Die Lieferung der Diskketten erfolgt durch den SCC für die Schweiz.

BASIC - ERWEITERUNG	Fr. 55.00
BASIC - SUPER - ERWEITERUNG	Fr. 70.00

Für Deutschland und alle anderen Länder werden die Disketten direkt durch den Ersteller geliefert.

Thomas Burkhart
Lämmerweg 25
7900 ULM-Einsingen
Tel. 07305-5407

DISK — O — PRO

Im letzten Heft haben wir den neuen Toolkit TK 8000 für den CBM 8000 vorgestellt, der sich vor allem durch seine erweiterten BASIC - Befehle auszeichnet. Eine Umrüstung der 3000-Serie auf BASIC 4.0 (4000-Serie) ist mit einem entsprechenden ROM-Satz möglich. Für die 4000-Serie ist beim SCC das TK 4000 erhältlich. Allerdings gibt es hier eine überlegenswerte Alternative.
Durch das Umrüsten wird eine erhebliche Mehrarbeit durch Umschreiben der Programme erforderlich, vor allem dann, wenn die Programme PEEK oder POKE-Befehle oder Maschinensprachroutinen enthalten, da die Zeropage und Systemadressen geändert worden sind und der 2. Kassettenbuffer nicht mehr benutzt werden kann, er wird vom System beansprucht.
Hier hilft der neue DISK-O-PRO-Toolkit, welcher beim BASIC 2.0 (3000-Toolkit) die normalen Befehle des Toolkit ermöglicht, sofern dieser eingebaut ist und darüberhinaus die Zusatzbefehle des TK 8000 und das erweiterte und vereinfachte DOS 2.0 der 8000-Serie enthält.

KILL	schaltet den DISK-O-PRO ab.
SCROLL	Repeat-Funktion und Aufrollen des Bildschirms beim Listen.
OUT	Schaltet SCROLL ab.
SET	Ausgabe von String mit definierter Taste.
PRINT USING	formatierte Ausgabe.
INITIALIZE	initialisiert die Drive.
SEND	sendet Commandostring an Floppy.
EXECUTE	lädt und startet Programm ab Floppy.
MERGE	lädt Programm ab Floppy und hängt es an ein bestehendes Programm an.

Zusätzlich sind die 14 Befehle des erweiterten DOS 2.0 möglich. Dies sind die folgenden Befehle:

Header	entspricht dem Formatierungsbefehl.
Directory	lädt den Directory ab Floppy.
Backup	dupliziert eine Diskkette.
Copy	kopiert ein Programm von einer Disk auf eine andere.
Collect	löscht alle Blöcke, die nicht durch SEQ, PRG oder REL - Dateien belegt sind.
Dsave	schreibt Programm auf Disk.
Dload	lädt Programm ab Disk.
Rename	eine Datei wird umbenannt.
Scratch	löscht Programm oder Daten von der Disk.
Concat	Dateien können zusammengehängt werden.
Dopen	Eröffnet die Dateien.
Dclose	schliesst die Dateien.
Append	hängt zusätzliche Daten an bestehende Datenfile an.
Record	positioniert den Recordzeiger einer REL-Datei auf ein Byte im angegebenen Record.

Das DISK-O-PRO wird im rechten freien Sockel eingesetzt und mit SYS 36864 initialisiert.
Es funktioniert mit allen Commodore DISK 2040, 3040, 4040, 8050 wobei jedoch beim 2040 und 3040 der Befehl Record nicht ausgefuehrt wird. Will man auch diesen Befehl ausführen, muss der ROM-Satz des Floppy gegen den des 4040 ausgetauscht werden.

Der Umrüstsatz für das Floppy 2040 und 3040 auf DOS 2 kann unter der Bestellnummer P 1103 zum Preis von Fr. 240.-- durch den SCC bezogen werden. Bezugquellen für das DISK-O-PRO teilen wir Ihnen auf Anfrage hin ebenfalls gerne mit.

SCC - KASSETTENSERVICE

Wir haben schon verschiedentlich darauf hingewiesen, dass Programme, die unter dem Titel "Programm des Monats BASIC - BASIC" erscheinen, als Kassette zum Preis von Fr. 18.-- erhältlich sind.
Alle Programme entsprechen genau dem Listing in den CBM/PET NEW's und sind alle auf dem CBM 3000 programmiert. Die Programme laufen ebenfalls auf dem PET 2001 neues Betriebssystem. Auf dem alten Betriebssystem laufen diese Programme nur dann, wenn Sie keine POKE oder PEEK - Befehle enthalten, oder nicht mit Kleinschrift arbeiten. Ausgenommen sind die POKE Befehle 32768 - 33767, die den Bildschirmausdruck bewirken.
Das Gleiche gilt für die CBM 8000 Serie. Die Programme können aber meist einfach umgeschrieben werden.
Programme, die Maschinensprachroutinen enthalten, laufen nur auf der 3000 Serie. Um Irrtümern vorzubeugen, geben wir ab sofort auf den Programmen bekannt, auf welchen Geräten Sie erstellt worden sind.

In absehbarer Zeit erscheinen auch umgeschriebene Programme für die CBM 8000 Reihe, wir werden dann eine separate Liste dieser Programme erstellen.

BAUSAETZE

Von den ständig steigenden Preisen in der Elektronikbranche sind leider auch wir betroffen worden. Dies macht sich vor allem bei den Hardwarebausätzen und Fertiggeräten bemerkbar. Wir sehen uns daher gezwungen, die Preise den neuen Gegebenheiten anzupassen, dies bedeutet eine Erhöhung von ca. 20 %.
Die Bausätze, die in den Heften der PET NEW's 1-6/1980 beschrieben worden sind, können wir dank noch vorhandener Lager, vorläufig noch zu den alten Preisen liefern, daher resultiert eine Preiserhöhung nur bei den Bausätzen des Heftes 2/1981 der USER - PORT - Erweiterung.
Der Bausatz P 2250 (bisher Fr. 180.--) kostet neu Fr. 215.-- .
Das Fertiggerät P 2251 (bisher Fr. 340.-- kostet neu Fr. 390.--.
Wir sind sicher, das Sie für unsere Situation das nötige Verständnis aufbringen werden.

SCC PREIS - SENTATIONEN

C 2779 Centronic 779 Matrixdrucker, 120 Zeichen/Sek. Fr. 2300.--
C 1730 Centronic 730 Matrixdrucker, 100 Zeichen/Sek. Fr. 1555.--
SCC, Seeburgstr. 18, 6002 LUZERN, Tel. 041 31 45 45

Best.Nr.	Artikel	SFr.
P 9990	Wordprocessor II (Diskette Rom)	280.--
P 2240	TV Modulator Bausatz	65.--
P 2241	TV Modulator mit Videointerface, fertig	155.--
P 2248	8-Bit D/A-Wandler Bausatz	120.--
P 2249	8-Bit D/A-Wandler fertig	180.--
P 9898	CBM Umrüstsatz von Basic 1.0 auf Basic 2.0	240.--
P 9899	CBM Umrüstsatz von Basic 2.0 auf Basic 4.0	240.--
P 9900	CBM Floppy Umrüstsatz Dos 1.0 auf Dos 2.0	240.--
P 9907	CBM Basic Toolkit-Rom TK 8000 (zu CBM 8032)	236.--
P 9987	Newtim SV 2.0 Monitor, Editor (Gerätetyp angeben!)	475.--
P 9988	Monjana Monitor (Rom)	198.--
P 9994	CBM-Pascal (Diskette und Rom)	350.--
P 9996	CBM-Visical (Deutsch) (Gerätekonfiguration angeben!)	580.--
P 2042	CBM Filter grün zu PET 2001 mit weissem Bildschirm	25.--
P 9113	RTTY HAM Interface mit Software zu 3032	240.--
P 9995	Assembler Rom II (CBM 8000)	480.--

Bestellungen bitte an SCC, Seeburgstrasse 18, 6002 Luzern

Korrespondenz und Manuskripte bitte an

CBM/PET NEWS
Verlag SCC AG
Seeburgstrasse 12
CH-6006 Luzern

Das Jahresabonnement (6 Ausgaben) kostet für Mitglieder des Schweizer Computer Club Fr. 18.–/DM 21.–, für Abonnenten von **Mikro- und Kleincomputer** Fr. 21.–/DM 24.– und für Nichtabonnenten Fr. 48.–/DM 55.–. Bereits erschienene Ausgaben des Jahrganges werden nachgeliefert. Verlag SCC AG, CH-6002 Luzern, Postkonto Luzern 60-27181; Stuttgart 3786-709 (BLZ 600 100 70) oder Euroscheck.

Commodore: Wegbereiter des Jedermann-Computer

commodore

Commodore AG · Dufourstraße 9 · 4010 Basel · Tel. 061/23 78 00 · Telex 64 961

Autorisierte Commodore-Wiederverkäufer mit technischem Kundendienst

Aarau Dahms Computersysteme · Tel. (064) 22 77 66. **Basel** BD-Electronic · Tel. (061) 35 36 37. Geiger-Microcomputer · Tel. (061) 44 13 13. Leobag Computer AG · Tel. (061) 35 31 14. SYSAG Systems & Services AG · Tel. (061) 38 21 20. Thürlemann Discount · Tel. (061) 22 41 66. **Bern** Computerland AG · Tel. (031) 24 25 54. Radio TV Steiner AG · Tel. (031) 55 45 81. **Biel** EIM Computer AG · Tel. (032) 23 15 88. **Brugg** Megos AG · Tel. (056) 41 34 17. **Fontainemelon** Urs Meyer Electronic · Tel. (038) 53 43 43. **Freiburg** Sovitrel SA · Tel. (037) 22 78 37. **Genf** Centre Informatique Gesmarco · Tel. (022) 21 11 75. Egg-Telsa SA · Tel. (022) 20 06 00. Gesmarco SA (Thônex) · Tel. (022) 49 88 44. Irco Electronic · Tel. (022) 20 33 06. Radio TV Steiner AG · Tel. (022) 28 52 22. **Gossau** Pius Schäfler · Tel. (071) 85 13 87. **Huttwil** Compu-Life, Rüfenacht AG · Tel. (063) 72 11 13. **Interlaken** Datatechnik · Tel. (036) 22 10 21. **Lausanne** Mafioly SA · Tel. (021) 22 00 44. Schaer Informatique · Tel. (021) 23 55 55. **Lugano** Luigi Chiodoni · Tel. (091) 23 23 09. **Luzern** Dialog Computer Treuhand AG · Tel. (041) 31 45 45. Helfenstein + Bucher AG · Tel. (041) 22 13 43. Schweizer Computer Club · Tel. (041) 31 45 45. **Magliaso** Marah SA · Tel. (091) 71 14 28. **Mellingen** Instant-Soft AG · Tel. (056) 91 20 21. **Niederrohrdorf** Nöthiger Electronic · Tel. (056) 96 28 96. **Rüti/ZH** Logon AG · Tel. (055) 31 72 30. **Schaffhausen** PIM Systems · Tel. (053) 45 45 0. Syntron Electronic · Tel. (053) 5 33 77. **Sion** Sphère Corporation · Tel. (027) 22 68 14. **St.Gallen** LASYS Tel. (071) 28 39 05. **Thun** HMB electronic · Tel. (033) 22 66 88. **Wettingen** Elbatex AG · Tel. (056) 26 98 27. **Winterthur** Nowak AG · Tel. (052) 22 08 03. **Wohlen/AG** SYSAG Systems & Services AG · Tel. (057) 6 36 50. **Zürich** Fürrer Büro-Computer · Tel. (01) 202 49 92. Hannes Keller AG · Tel. (01) 69 36 33. Logon AG · Tel. (01) 62 59 22. Microspot AG · Tel. (01) 241 20 30. Erhard Wipf AG · Tel. (01) 221 21 00.