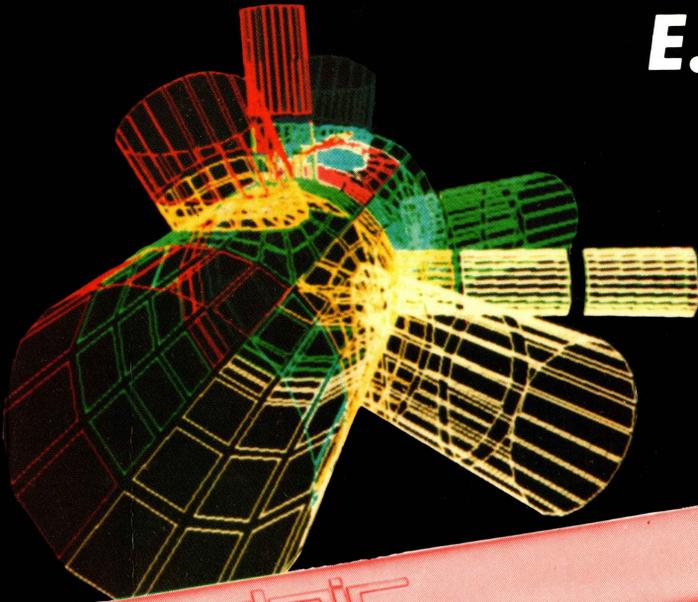


E. Floegel



Programmieren

*in Basic und
Maschinencode*

mit dem

ZX81

ISBN 3-921682-93-2

Es kann keine Gewähr dafür übernommen werden, daß die in diesem Buche verwendeten Angaben, Schaltungen, Warenbezeichnungen und Warenzeichen, sowie Programmlistings frei von Schutzrechten Dritter sind. Alle Angaben werden nur für Amateurzwecke mitgeteilt. Alle Daten und Vergleichsangaben sind als unverbindliche Hinweise zu verstehen. Sie geben auch keinen Aufschluß über eventuelle Verfügbarkeit oder Liefermöglichkeit. In jedem Falle sind die Unterlagen der Hersteller zur Information heranzuziehen.

Nachdruck und öffentliche Wiedergabe, besonders die Übersetzung in andere Sprachen verboten. Programmlistings dürfen weiterhin nicht in irgendeiner Form vervielfältigt oder verbreitet werden. Alle Programmlistings sind Copyright der Fa. Ing. W. Hofacker GmbH. Verboten ist weiterhin die öffentliche Vorführung und Benutzung dieser Programme in Seminaren und Ausstellungen. Irrtum, sowie alle Rechte vorbehalten.

COPYRIGHT by Ing. W. HOFACKER © 1982,
Tegernseerstr. 18, 8150 Holzkirchen

1. Auflage 1982

Gedruckt in der Bundesrepublik Deutschland — Printed in West-Germany —
Imprime'en RFA.

Programmieren

***in Basic und
Maschinencode***

mit dem

ZX81



Vorwort

Der ZX81 ist ein kleiner Rechner. Das Wort "klein" bezieht sich aber nur auf seine Abmessungen, nicht aber auf seine Leistung. In den letzten Wochen, in denen ich mich mit dem Programmieren dieses Rechners beschäftigte, fand ich viele Eigenschaften, die ihn als ernsthaftes Gerät und beileibe nicht als Spielzeug erscheinen lassen.

Die Programme in diesem Buch sind aufgeteilt in Spielprogramme, Schul- und andere Programme und Programme zur Datenverwaltung. Alle Programme sind abgeschlossen und lauffähig. Der Leser ist aber aufgefordert, diese Programme nicht als starr zu betrachten, sondern sie nach seinen Belieben zu erweitern oder abzuändern. Dadurch lernt er einerseits die Programme zu verstehen, andererseits sie auf seine eigenen Bedürfnisse zuzuschneiden.

Für alle diejenigen, die sich über BASIC hinaus, mit der Programmierung des Prozessors Z80 beschäftigen wollen, ist ein Kapitel über die Verwendung von Maschinencode eingeschlossen.

Danken möchte ich Herrn Gerhard Birzer von der Fa. Sinclair, der uns breitwillig Testgeräte zur Verfügung gestellt hat.

Dem Leser wünsche ich viel Spaß beim Probieren und Programmieren.

Holzkirchen, Januar 1982

Ekkehard Flögel

Inhaltsverzeichnis

1. Was heißt Programmieren ?	1
2. Programmieren in BASIC	5
2.1 Programm zur Erfassung der Kosten eines Kraftfahrzeugs	6
2.2 Korrigieren und Ändern von Programmen	12
2.3 Fehlersuche in Programmen	14
3. Spiele	17
3.1 Spielelemente	17
3.2 Schießbude	21
3.3 Käfer	23
3.4 PENG	24
3.5 Schlange	27
3.6 Der freie Fall	28
3.7 Der schiefe Wurf	30
3.8 Ente	32
4. Programme für die Schule	35
4.1 GGT	35
4.2 KGV	36
4.3 Primfaktorenzerlegung	36
4.4 Lösung der quadratischen Gleichung	37
4.5 NUSU	38
4.6 FKTW	39
4.7 POLYM	41
4.8 Darstellung von Funktionen	42
4.9 Vokabeln lernen	44
4.10 Berechnung der Mehrwertsteuer und des Nettobetrages	48
4.11 Berechnung des Wochentages	49
4.12 Dezimal Hexadezimal Wandlung	49
5. Datenverwaltungsprogramme	51
5.1 Lagerbestand	51
5.2 LISTE	53
5.3 Terminkalender	58
5.4 Schallplattenverzeichnis	62

6. Programmieren in Maschinencode	69
6.1 Einführung in die Programmierung mit Maschinencode	69
6.2 Der Maschinencode-Editor MONI (16k)	77
7. Anschluß einer PIO an den ZX81 – Einführung	87
7.1 Anschluß der PIO an den ZX81	88
7.2 Programmierung der PIO	91
8. Steuerungsprogramme	97
8.1 Ansteuerung einer Leuchtdiodenzeile	97
8.2 Lauflicht	98
8.3 Lichtbalken	99
8.4 Blinklicht	100
9. Anhang	101
9.1 Vergleich ZX81 BASIC-Befehle mit anderen Rechnern	101
9.2 HEX–DEZ Zahlen	107
9.3 Tabelle für Vorwärts- und Rückwärtsverzweigungen	107
9.4 Befelssatz für Z80 A CPU	109
9.5 Z80 PIO-Pinbelegung	122
9.6 Datenblatt 47 LS 138	123
9.7 Programme und Tips für den ZX81-Anwender	126
9.8 Literaturhinweis	128

Was heißt Programmieren ?

1. Was heißt Programmieren ?

Der ZX81 wird für viele der erste Kontakt mit einem programmierbaren Rechner sein. Deshalb soll zuerst einmal die Frage geklärt werden, was bedeutet es, einen Rechner zu programmieren.



**Der Mikrocomputer Sinclair ZX81 hat Abmessungen von
16,7 x 17,5 x 4 cm und wiegt lediglich 350 Gramm**

Ein Programm ist die Umsetzung eines Problems in eine Sprache, die der Rechner versteht. Zuerst muß also ein Problem, eine Aufgabe bestehen, die vom Rechner gelöst werden soll.

Diese Aufgaben können vielfältig sein. Es können reine mathematische Aufgaben, wie zum Beispiel Auswertung von Formeln oder wirtschaftliche Aufgaben, wie etwa die Überwachung eines Lagerbestandes oder Aufgaben der Mess- und Regeltechnik, wie zum Beispiel die laufende Überwachung einer Innen- und Außentemperatur.

Die Lösung solcher Aufgaben mit dem Rechner bedingt allerdings zweierlei. Zuerst muß ein Lösungsweg gesucht werden. Das Auffinden eines solchen Lösungsweges, auch Algorithmus genannt, ist nicht immer einfach.

Betrachten wir hierzu ein einfaches Beispiel:

Wir haben eine Liste mit vielen Namen. Aus dieser Liste wollen wir einen Namen aussuchen.

Was für Lösungsmöglichkeiten gibt es ? Für die Beantwortung dieser Frage überlegt man zuerst einmal, wie man diese Aufgabe ohne Rechner löst. Sind es nur wenige Namen, so wird man oben in der Liste anfangen, jeden Namen mit dem gesuchten vergleichen, solange, bis man den richtigen Namen gefunden hat. Dieses Suchen kann sehr schnell gehen, wenn der gesuchte Name ganz oben in der Liste steht, oder sehr lange, wenn er erst am Ende der Liste auftaucht. Dabei ist es aber gleichgültig, ob die Namen alphabetisch sortiert sind oder nicht.

Sind es viele Namen, wie zum Beispiel in einen Telefonbuch, dann dauert dieses Verfahren zu lange. Man wird dann einen anderen Weg versuchen. Sortiert man zuerst alle Namen nach dem Alphabet, so kann man hinterher viel schneller suchen. Man wird zuerst den gesuchten Namen mit einem Namen in der Mitte der Liste vergleichen, und somit feststellen, ob der gesuchte Name davor oder dahinter steht. Je nachdem wird man in der vorderen Hälfte oder in der hinteren Hälfte nachsuchen. Dieses Suchen geht viel schneller, es bedingt aber, daß die Namen sortiert sind. Dieses Sortieren kann aber ebenfalls sehr lange dauern. Seit Beginn der programmierbaren Rechner sind viele Sortierverfahren entwickelt worden, wie zum Beispiel BUBBLE, RIPPLE,

SHELL, HEAP oder QUICK-Sort (2*), um nur einige zu nennen. Ein Programmbeispiel dazu zeigt auch Programm LISTE in Kapitel 5.

An diesem Beispiel kann man aber auch eine Eigenschaft eines Computers zeigen, die sehr wichtig ist. Angenommen, wir suchen den Namen Schmidt in der Liste, dort ist aber nur ein Schmied vorhanden, so wird diesen Namen zwar ein Mensch, nicht aber der Computer finden. Dessen Auftrag lautete nämlich, Schmidt und nicht Schmied zu suchen. Kann so ein Fall auftreten, so muß auch dafür eine Lösungsmöglichkeit gesucht werden. Eine wäre zum Beispiel, als Suchwort Schmi* einzugeben. Das bedeutet, suche alle Namen, die mit Schmi anfangen. Dann wird der Rechner also Schmidt, Schmied, aber auch Schmielmayer finden.

Ein Rechner ist, entgegen allen anderen Behauptungen, nicht intelligent, sondern nur so intelligent, wie sein Programmierer. Ich halte es für absurd, wenn durch Einsatz eines Mikroprozessors, einem Rechner, aus einer Waschmaschine plötzlich eine intelligente Waschmaschine wird. Bisher haben Programmierer die Tatsache, daß die breite Öffentlichkeit keine Ahnung vom Programmieren hat, dazu benutzt, eigene Fehler einfach dem Computer zuzuschreiben.

“Dieser blöde Computer schreibt für einen Schuldbetrag von 0,00 DM eine Mahnung aus.“

Der Fehler liegt nicht beim Computer, sondern beim Programmierer.

Wenn man einen Lösungsweg gefunden hat, so braucht man ein zweites, eine Sprache in welcher dieser Lösungsweg beschrieben wird. Natürlich muß es eine Sprache sein, die der Rechner auch versteht. Unser ZX81 versteht “The Beginners All Purpose Symbolic Instruction Code” oder kurz BASIC. Von dieser, Anfang der 60er Jahre am College of Dartmouth entwickelten Sprache, gibt es viele Dialekte. Das ZX81 BASIC ist so ein Dialekt, das heißt, es weicht an einigen Stellen von anderen BASIC-Dialekten ab, aber nicht so weit, daß es mit diesen nicht mehr vergleichbar wäre. BASIC ist eine interpretive Sprache, ein Inter-

* Literatur:

(2) C. Lorenz, BASIC für Fortgeschrittene, Hofacker Verlag

preter. Das bedeutet, daß eine direkte Anweisung wie

PRINT 3*4

gleich ausgeführt wird.

Ein Programm ist eine Folge von solchen Anweisungen, die nach Zeilennummern geordnet sind. Bei einem Programmablauf wird jede Zeile interpretiert und sofort ausgeführt. Auf den Unterschied zu Compilersprachen soll hier nicht eingegangen werden. Auf die Sprach-elemente von BASIC soll ebenfalls nicht eingegangen werden. Diese sind in dem wirklich ausgezeichneten Handbuch des ZX81 beschrieben.

Fassen wir noch einmal zusammen:

Programmieren bedeutet

1. Lösungswege für ein Problem finden,
2. Beschreiben dieser Wege in einer Sprache, die der Computer versteht.



Der eingebaute 1 K -- byte RAM -- Arbeitsspeicher läßt sich durch Aufstecken eines externen Moduls auf 16 K -- byte RAM erweitern

Programmieren in BASIC

2. Programmieren in BASIC

In diesem Kapitel soll an einem Beispiel gezeigt werden, wie man von der Aufgabenstellung zum fertigen Programm fortschreitet.

Die Aufgabenstellung ist bewußt einfach gehalten, um den Aufbau des Programms leichter verständlich zu machen. Sie lautet: Erfassung der laufenden Kosten eines Kraftfahrzeugs.

Wie wir schon im ersten Kapitel gesehen haben, müssen wir unseren Rechner alle Schritte genau vorgeben, denn er kann ja nur das ausführen, was wir ihm eingegeben haben. Vor dem Programmieren machen wir eine sogenannte Systemanalyse.

Folgende Fragen müssen dabei beantwortet werden:

1. Welches sind die Eingabedaten ?
2. Wie werden diese verarbeitet ?
3. Welche Daten sollen ausgegeben werden ?
4. In welcher Form werden die Daten gespeichert ?
5. Gibt es bei Berechnungen mehrdeutige Lösungen ?
6. Welche mathematischen Verfahren werden angewendet ?

Dieser Fragenkatalog kann sich bei einzelnen Problemen noch erheblich erweitern. Es kann zum Beispiel mehrere Lösungsmöglichkeiten geben. Dann ist zu entscheiden welche die beste ist. Wir werden später bei anderen Programmen verschiedene Lösungsmöglichkeiten mit ihren Vor- und Nachteilen betrachten. Nun zu unserem Problem, der Erfassung der Kosten eines Kraftfahrzeugs.

2.1 Programm zur Erfassung der Kosten eines Kraftfahrzeugs

1. Frage: Eingabedaten.

Das sind natürlich alle Kosten, die auftreten. Erste Einschränkung, wir wollen nur alle Kosten erfassen, die als direkte Ausgaben auftreten, nicht aber Amortisation und Wertminderung.

Feste Kosten sind Steuer und Versicherung.

Laufende Kosten sind Ausgaben für Benzin, Öl, Reparaturen, Ersatzteile, Wartung und Pflege.

Damit ist für unseren Fall die erste Frage beantwortet.

Die nächsten beiden Fragen hängen eng zusammen. Wir werden unsere Daten so verarbeiten, daß die gewünschten Ergebnisse ausgegeben werden können. Dies sind die Summe aller Einzelausgaben, also die Summe der Benzinkosten, Reparaturen usw. Diese Zahlen können erfasst werden. Dabei ergibt sich jetzt die Frage, ob nicht die Kosten pro gefahrenen Kilometer eine sehr aussagekräftige Zahl ist. Unsere vorschnelle Ausnahme, das alle Eingabedaten vorhanden sind, war also nicht richtig. Es muß noch die Zahl der gefahrenen Kilometer angegeben werden. Dann sollte der Benzinverbrauch auch ermittelt werden. Also bei der Eingabe der Benzinkosten auch die Eingabe der getankten Mengen. Eventuell muß auch eine Fahrtkostenerstattung berücksichtigt werden, wenn das Fahrzeug für dienstliche Zwecke verwendet wurde. Die Ausgabedaten sind also die Summe aller Einzelkosten, die Gesamtsumme, die Kosten pro Kilometer und der Verbrauch.

Fassen wir einmal zusammen:

Eingabedaten: Kilometer
Liter
Benzinpreis
Reparaturkosten
Wartung/Pflege
Erstattungen

Ausgabedaten: Summe der Einzeleingaben
Gesamtsumme
Verbrauch
Kosten pro Kilometer

Wie werden nun die Daten gespeichert? Wenn man diese Aufgabe ohne Computer löst, so benutzt man dazu ein Fahrtenbuch, in das man diese Angaben einträgt. Die Eingaben werden dort täglich gemacht und immer für einen Monat zusammengefasst. Genauso soll es auch mit dem Computer gemacht werden. Dazu brauchen wir noch eine weitere Eingabe, den Monatsnamen. Damit haben wir alle Fakten für unser Programm beisammen und können uns nun überlegen, wie das Programm ablaufen soll. Die im Fragenkatalog angegebenen Fragen 5 und 6 brauchen nicht untersucht werden, da nur Grundrechnungsarten (Addition, Subtraktion, Multiplikation, Division) verwendet werden.

Wie soll nun das Programm ablaufen?

Nun, genauso wie im Fahrtenbuch. Wir geben den Monatsnamen ein, erhalten einen Ausdruck aller bis dahin gemachten Einträge, fügen unsere neuen Einträge hinzu und schließen die Eingabe ab.

Dann benutzen wir eine der Eigenschaften des ZX81 BASIC, die andere Rechner in dieser Form nicht haben. Wenn das Programm auf Kasette gespeichert wird, so werden auch alle Variablen mit ihren augenblicklichen Werten aufgezeichnet, und somit alle Veränderungen, die gerade gemacht wurden. Diese Daten werden allerdings gelöscht, wenn ein Programm mit RUN gestartet wird. Beim ZX81 sollte man sich angewöhnen, Programme mit GOTO zu starten, denn dann bleiben alle Werte der Variablen erhalten.

Nun zum Programm:

Für unsere Kostenerfassung legen wir ein Zahlenfeld mit 13 Zeilen und jeweils 7 Spalten fest.

DIM K (13,7)

Die 13 bezieht sich auf die Monate Januar bis Dezember, dazu kommt noch die Gesamtsumme die 7 auf die 7 möglichen Eingabedaten. Diese Dimensionsangabe darf auch nur einmal ganz zu Beginn vor den ersten Eintragungen gemacht werden, da jede neue Dimensionierung die Werte der Variablen löscht. Deshalb machen wir ein kleines Vorprogramm, das unser Kostenfeld eröffnet und alle Einträge zu Null setzt.

```

010 DIM K(13,7)
020 FOR I=1 TO 13
030 FOR J=1 TO 7
040 LET K(I,J)=0
050 NEXT J
060 NEXT I
070 GOTO 1000

```

2.1 Eröffnung des Kostenfeldes

Unser Hauptprogramm in Abbildung 2.2 beginnt in Zeile 1000. Erst wird der Bildschirm gelöscht und dann der gewünschte Monat als 3-stellige Zeichenkette eingegeben. Für Januar also JAN, Februar = FEB usw. bis Dezember = DEZ.

Das Unterprogramm in Zeile 100 bis 170 überprüft, ob die Eingabe richtig war und weist der Variablen I den Monatsindex zu. Für Januar also 1, Februar = 2 usw. Es vergleicht dazu jeweils 7 aufeinanderfolgende Buchstaben der Zeichenkette M\$ mit der Eingabe B\$. Wenn eine Übereinstimmung besteht, wird das Unterprogramm verlassen. Damit haben wir festgelegt, welche Spalten aus unserem Kostenfeld ausgegeben werden sollen. War die Eingabe falsch, so springt das Programm nach 1000 und wartet auf eine neue Eingabe. Dieses Abfangen von Fehlbedienungen kostet zwar Speicherplatz, sollte aber wenn immer möglich, in ein Programm eingebaut werden.

Im Unterprogramm 200 findet die Ausgabe der Bezeichnungen der Einzelposten statt. Auch dafür verwenden wir Abkürzungen und zwar bedeuten KM = Kilometer, LTR = Liter, BZK = Benzinkosten, REP = Reparaturen, EST = Ersatzteile, W/P = Wartung und Pflege, ERST = Erstattungen, SUM = Summe aller Kosten, L/KM ist der Verbrauch Liter pro Kilometer und DM/KM sind die aus den obigen Angaben gemachten Kosten. Durch die PRINT AT Anweisung werden die Bezeichnungen immer an die gleichen Stellen auf den Bildschirm ausgegeben.

Das Unterprogramm in den Zeilen 300 – 390 gibt die für diesen Monat gemachten Einträge aus. Sind keine Einträge vorhanden, wird nicht 0.00 ausgegeben, sondern die Ausgabe wird unterdrückt.

Für die Ausgabe wird die gespeicherte Zahl in eine Zeichenkette zurückgewandelt. Der Grund dafür ist, daß die Ausgabe auf den Bildschirm so erfolgen soll, daß alle Dezimalpunkte übereinander in der gleichen Spalte erscheinen. Der Computer würde sonst die Ausgabe folgendermaßen machen.

```
25.32
   5
31.2
```

anstatt

```
25.32
  5.00
31.20
```

In den Zeilen 340 bis 375 wird zuerst die Länge der Zahl festgestellt und dann der Dezimalpunkt gesucht. Dann können die notwendigen Ergänzungen gemacht werden. Es gibt noch einen anderen Weg, so einen Ausdruck zu erhalten, ohne diese Umsetzung in eine Zeichenkette. Dies fiel mir bei der Durchsicht einiger Krankenhausrechnungen auf, die auf einem Großcomputer erstellt wurden. Dort werden alle Einträge die auf einen vollen DM Betrag enden, durch Addieren eines Pfennigs erweitert. Statt DM 300 wird DM 300.01 berechnet.

Im Unterprogramm 400 bis 470 wird die Eingabe neuer Einträge durchgeführt. Ein Pfeil (< —) zeigt auf die Position, die als nächstes eingegeben werden soll. Eine leere Eingabe (nur Drücken auf die NL-Taste) setzt den Pfeil auf den nächsten Posten. Die Eingaben werden gleich aufaddiert. Nach der letzten Eingabe (ERST) berechnet der Computer die Gesamtsumme SUM, den Verbrauch und die Kosten pro Kilometer.

Sind alle Eingaben gemacht worden, so fragt der Computer, ob er eine Gesamtbilanz machen soll. Wird diese Frage mit J beantwortet, so addiert er alle Monatsbeträge und berechnet daraus wie bei einem einzelnen Monat die Gesamtsumme, Verbrauch und Kosten pro Kilometer.

Das Programm ist modular aufgebaut. Das heißt für die einzelnen Berechnungen, für Ein- und Ausgaben sind Unterprogramme verwendet worden, die wenn sie benötigt werden, vom Hauptprogramm aufgerufen werden. Durch die Benutzerfreundlichkeit ist es allerdings sehr lang

geworden. Es funktioniert nur mit dem 16k RAM-Modul. Es ist außerdem noch nicht vollständig. Es fehlt noch die Eingabe der festen Kosten und vielleicht die Ausgabe der Daten für einen einzelnen Monat, ohne daß Eingaben verlangt werden. Diese Erweiterungen sollen als Programmierübung für den Leser überlassen bleiben.

```
100 LET M$="JANFEBMRZAPRMAIJUNJULAUGSEPOKTNOVDEZ"  
110 FOR K=1 TO 34 STEP 3  
120 LET B$=M$(K TO K+2)  
130 IF A$=B$ THEN GOTO 160  
140 NEXT K  
150 GOTO 1000  
160 LET I= INT (K/3)+1  
170 RETURN  
200 PRINT AT 4,5;"KM:"  
210 PRINT AT 5,4;"LTR:"  
220 PRINT AT 6,4;"BZK:"  
230 PRINT AT 7,4;"REP:"  
240 PRINT AT 8,4;"EST:"  
250 PRINT AT 9,4;"W/P:"  
260 PRINT AT 10,3;"ERST:"  
270 PRINT AT 12,4;"SUM:"  
280 PRINT AT 14,3;"L/KM:"  
290 PRINT AT 15,2;"DM/KM:"  
299 RETURN  
300 FOR J=1 TO 7  
310 IF K(I,J)=0 THEN GOTO 320  
315 GOSUB 330  
320 NEXT J  
325 RETURN  
330 LET B$= STR$ K(I,J)  
340 LET L= LEN B$  
345 FOR N=1 TO L  
350 LET C$=B$(N TO N)  
355 IF C$="." THEN GOTO 370  
360 NEXT N  
365 LET B$=B$+".00"  
370 IF N=L-2 THEN GOTO 380  
375 IF N=L-1 THEN LET B$=B$+"0"  
380 LET L=8- LEN B$  
385 PRINT AT 3+J,12+L;B$  
390 RETURN
```

```

400 FOR J=1 TO 7
410 PRINT AT 3+J,9;"<--"
420 INPUT N$
425 IF N$="" THEN GOTO 450
430 LET N= VAL N$
435 LET K(I,J)=K(I,J)+N
440 GOSUB 330
450 PRINT AT 3+J,9;"  "
460 NEXT J
470 RETURN
500 LET N=0
510 FOR J=3 TO 6
520 LET N=N+K(I,J)
530 NEXT J
535 LET N=N-K(I,7)
540 LET J=9
545 LET S=N
550 LET B$= STR$ N
560 GOSUB 340
570 LET N=K(I,2)/K(I,1)*100
575 LET N= INT (N*100)/100
580 LET J=11
590 LET B$= STR$ N
600 GOSUB 340
610 LET N= INT (S/K(I,1)*1000)/1000
620 LET J=12
625 LET B$= STR$ N
630 PRINT AT 15,16;B$
690 RETURN
700 FOR J=1 TO 7
710 LET K(13,J)=0
715 NEXT J
720 FOR J=1 TO 7
730 FOR I=1 TO 12
740 LET K(13,J)=K(13,J)+K(I,J)
750 NEXT I
755 LET B$= STR$ K(13,J)
760 GOSUB 340
770 NEXT J
780 RETURN
1000 CLS
1010 PRINT AT 2,3;"MONAT:";

```

```

1020 INPUT A$
1030 PRINT A$
1040 GOSUB 100
1050 GOSUB 200
1060 GOSUB 300
1070 GOSUB 400
1080 GOSUB 500
1090 PRINT AT 17,2;"GESAMTBILANZ (J/N) "
1095 IF INKEY$="" THEN GOTO 1095
1100 IF INKEY$ <> "N" THEN GOTO 1200
1110 CLS
1120 PRINT AT 10,10;"ENDE"
1130 STOP
1200 CLS
1210 PRINT AT 2,3;"GESAMTBILANZ:"
1220 GOSUB 200
1230 GOSUB 700
1240 GOSUB 500

```

2.2 Programm Fahrtenbuch

Wenn nach neuen Einträgen das Programm mit SAVE auf Kassette geschrieben wird und immer mit GOTO 1000 gestartet wird, bleiben alle Einträge erhalten. Man hat somit die Erfassung der Daten für jeden Monat und das ganze Jahr.

Das Programm lässt sich leicht abändern. Man könnte z. B. Haushaltskosten, Taschengeldausgaben usw. erfassen. Es sind in diesem Programm einige Programmiertechniken verwendet worden, die in ähnlicher Form noch in anderen Programmen verwendet werden.

2.2 Korrigieren und Ändern von Programmen

Die Programme des ZX81 lassen sich auch sehr leicht editieren. Das heißt, fehlerhafte Zeilen abändern, löschen usw. Schon bei der Eingabe wird die richtige Schreibweise überwacht. Selbst wenn die Zeile richtig aus dem Eingabespeicher in den Programmspeicher übernommen wurde, so ist damit noch nicht gesagt, ob sie auch sachlich richtig ist. Kein Programm ist von Anfang an fehlerfrei. Fehler müssen gesucht und beseitigt werden.

Zum Editieren von Programmzeilen benötigt man hauptsächlich die Cursortasten \leftarrow , \rightarrow , die EDIT- und die RUBOUT-Taste.

Als Beispiel für das Ändern einer Zeile nehmen wir an, wir hätten in Zeile 740 unseres Programms einen Fehler gemacht. Wir haben dort geschrieben

```
740 LET K (13,I) = K (13,J) + K (I,J)
```

Das I in K (13,I) ist falsch, dort muß ein J stehen.

Zuerst geben wir LIST 740 ein.

Damit ist die Zeile 740 die oberste Zeile auf dem Bildschirm, und der Zeiger \rangle (invers) zeigt auf diese Zeile. Damit kann sie mit EDIT in den Änderungsspeicher übernommen werden. Jetzt drücken wir die Cursortaste \rightarrow sooft bis der Cursor direkt hinter dem I steht. Wenn wir zu weit gegangen sind, so gehen wir mit \leftarrow wieder zurück. Nun löschen wir das Zeichen durch RUBOUT und fügen das neue Zeichen J ein. Wenn wir jetzt die NL-Taste betätigen wird die geänderte Zeile in das Programm übernommen.

Beim Ändern eines Zeichen muß dieses also vorher gelöscht und dann neu eingegeben werden. Sollen neue Zeichen eingefügt werden, so wird der Cursor an diese Stelle gesetzt und die Zeichen eingetippt. Der Rechner schafft sich automatisch Platz für diese neuen Zeichen.

Soll eine Zeile eine andere Zeilennummer erhalten, so wird diese Zeile, wie oben beschrieben, in den Änderungsspeicher geholt, mit RUBOUT die alte Zeilennummer gelöscht und die neue Zeilennummer eingegeben. Nach Betätigen der NL-Taste ist nun diese Zeile unter der alten und neuen Zeilennummer vorhanden. Durch Eingabe der alten Zeilennummer allein wird sie dann gelöscht.

Soll die folgende Zeile ebenfalls geändert werden so kann man den Editorcursor mit der Cursortaste \downarrow nach unten schieben. Die Verwendung dieser Taste ist aber nur dann sinnvoll, wenn nur wenige Zeilen übersprungen werden sollen. Sonst ist es schneller und einfacher LIST (Zeilennummer) einzugeben.

2.3 Fehlersuche in Programmen

Im Prinzip kann man davon ausgehen, daß kein Programm auf Anhieb fehlerfrei ist. Durch die Syntaxprüfung sind Schreibfehler schon bei der Eingabe gefunden worden. Logische Fehler werden erst beim Programmlauf gefunden. Gegenüber anderen BASIC-Dialekten weist das ZX81 BASIC eine Besonderheit auf. Es erkennt, ob auf der rechten Seite einer Anweisung eine Variable steht, der noch kein Wert zugewiesen wurde. Wenn also im Programm eine Zeile so lautet

```
220 LET Y = A * X * X + B * X + C
```

und der Variablen B wurde vorher kein Wert zugewiesen, so erfolgt die Fehlermeldung 2/220. In anderen Basic-Sprachen wird für B einfach Null angenommen und ohne Fehlermeldung weitergerechnet. Trotzdem sollten aber auch Rechenergebnisse einmal von Hand oder mit dem Taschenrechner überprüft werden.

Wenn ein Programm gestartet wurde, und es erscheint längere Zeit keine Ausgabe auf dem Bildschirm, so kann das Programm in einer unendlichen Schleife stecken. Das Programm kann dann durch die BREAK-Taste unterbrochen werden. Dann wird die Zeilennummer ausgegeben bei welcher das Programm unterbrochen wurde. In dieser Gegend muß man dann nach Möglichkeit einer unendlichen Schleifenbildung suchen. Meistens ist es ein GOTO Befehl mit einr falschen Zeilennummer, oder eine nicht erfüllte IF-THEN Bedingung wie im folgenden Beispiel:

```
10 LET I = 1
20 IF I = 100 THEN GOTO 50
30 LET I = I + 2
40 GOTO 20
50 STOP
```

Die Bedingung $I = 100$ ist nie erfüllt, da I immer eine ungerade Zahl ist. Nach $I = 99$ wird $I = 101$. Damit stellt dies eine unendliche Schleife dar. Richtig wäre in Zeile 20 die Abfrage gewesen.

```
20 IF I >= 100 THEN GOTO 50
```

Damit wäre die Schleife mit $I = 101$ verlassen worden. Meistens ist aber nicht klar ersichtlich, wo im Programm so eine

Schleife auftritt. Dann wird man versuchen, durch PRINT-Anweisungen oder durch Einfügen des STOP-Befehls den Fehler einzukreisen.

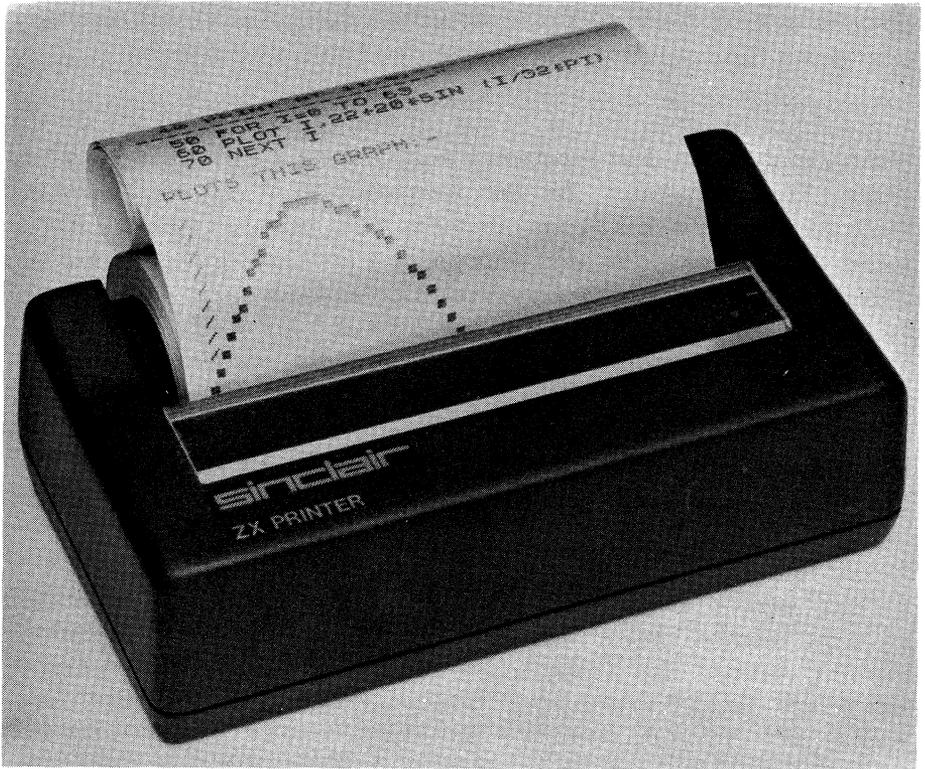
Ein anderer beliebter Fehler ist das unabsichtliche Ändern des Wertes einer Variablen. In dem Programm in Abbildung 2.2 ist I eine Variable, deren Wert den Index für den ausgewählten Monat angibt. Dieser Wert darf während des Programmlaufes nicht geändert werden. Allzu leicht vergisst man dies und verwendet I als Laufparameter in einer Schleife. Die Ergebnisse sind dann reine Zufallszahlen.

BASIC ist eine interpretative Sprache. Das heißt, das eingegebene Programm oder Teile des Programms können sofort ausgeführt werden. Dies sollte man dazu benutzen, kleine Programmabschnitte (Module) sofort zu testen. Dies ist wesentlich einfacher, als ein vollständiges Programm auf Fehler zu untersuchen.

Leider führt dies aber auch dazu, daß Programme nur am Rechner entwickelt werden und nebenher keine Notizen zum Programmablauf gemacht werden. Bevor man mit dem Schreiben eines Programms beginnt, sollten die wesentlichen Grundzüge des Programmablaufs schriftlich festgehalten werden. Dies erleichtert wesentlich die Fehlersuche.

Auf eine weitere Fehlermöglichkeit, die Instabilität der angewendeten mathematischen Verfahren, soll hier nicht eingegangen werden.

Im Allgemeinen kann man sagen, daß die Fehlersuche mindestens ebenso lange dauert, wie das Schreiben des Programms.



Der Sinclair ZX -- Drucker wurde speziell als Peripherie für die Sinclair-Mikrocomputer ZX81 und ZX80 (mit nachträglich eingebauten 8 K BASIC ROM) entwickelt

Spiele

3. Spiele

3.1 Spielelemente

In diesem Kapitel soll gezeigt werden, wie aus einigen Grundelementen Spiele entwickelt werden können. In den meisten Fällen wird man die Grafikmöglichkeiten des ZX81 ausnutzen. Deshalb werden wir zuerst einige Figuren aus den Grafikzeichen entwerfen. Dazu benutzt man am besten kariertes Papier. Bei der Grafik wird der Platz, den ein ausgegebener Buchstabe ausfüllt, nochmals in 4 kleine Quadrate aufgeteilt.

Diese Grafikzeichen sind im Handbuch auf Seite 78 dargestellt. Zuerst wollen wir in der Bildschirmmitte ein kleines Männchen zeichnen. In Abbildung 3.1 ist dieses Männchen skizziert. Diese Skizze übertragen wir nun in ein Programm. Die oberste Zeile des Männchens lautet als Programmzeile

```
PRINT " ■■■ "
```



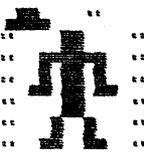
3.1 Skizze Männchen

Das vollständige Programm zeigt Abbildung 3.2.

```

100 LET H=10
110 LET V=10
120 PRINT AT V,H;" "
130 PRINT AT V+1,H;" "
140 PRINT AT V+2,H;" "
150 PRINT AT V+3,H;" "
160 PRINT AT V+4,H;" "
170 PRINT AT V+5,H;" "
180 PRINT AT V+6,H;" "

```



3.2 Männchen zeichnen

Nun lassen wir das Männchen mit einer Hand winken. Dazu löschen wir einen Arm und ersetzen ihn durch " ". Nach einer kurzen Warteschleife zeichnen wir wieder das Ausgangsbild. Dieses Programm zeigt Abbildung 3.3. Dieses Programm muß durch BREAK abgebrochen werden. Zur Verzögerung verwenden wir nicht den PAUSE-Befehl, denn er verursacht ein Bildschirmzittern. Deshalb die Warteschleife mit FOR NEXT.

```

100 LET H=10
110 LET V=10
120 PRINT AT V,H;" "
130 PRINT AT V+1,H;" "
140 PRINT AT V+2,H;" "
150 PRINT AT V+3,H;" "
160 PRINT AT V+4,H;" "
170 PRINT AT V+5,H;" "
180 PRINT AT V+6,H;" "
190 PRINT AT V+3,H+9;" "
200 PRINT AT V+2,H+9;" "
210 FOR I=1 TO 10
220 NEXT I
230 PRINT AT V+2,H+9;" "
240 PRINT AT V+3,H+9;" "
250 FOR I=1 TO 10
260 NEXT I
270 GOTO 200

```



3.3 Winkendes Männchen

Nun wollen wir Figuren über den Bildschirm bewegen. Dazu eignet sich unser Männchen nicht, sondern wir brauchen dazu einfachere Figuren. Abbildung 3.4 zeigt ein "Raumschiff". Dieses soll sich am oberen Bildschirmrand hin und her bewegen. Das Programm zeigt Abbildung

3.5. Die beiden Leerzeichen am Anfang und Ende der Zeichenkette löschen jeweils bei der Rechts- oder Linksbewegung das vorangegangene Zeichen.



3.4 Skizze Raumschiff

```

100 LET A$="    "
110 LET V=0
120 FOR H=0 TO 26
130 PRINT AT V,H;A$
140 NEXT H
150 FOR H=26 TO 0 STEP -1
160 PRINT AT V,H;A$
170 NEXT H
180 GOTO 120

```

3.5 Bewegen am oberen Rand

Der linke obere Bildschirmpunkt hat die Koordinaten 0,0, der rechte obere Bildschirmpunkt hat die Koordinaten 0,31. Unser 6 Zeichen langes Symbol kann sich also zwischen 0 und 26 bewegen (siehe auch Handbuch Seite 123).

Nun erweitern wir das Programm und lassen das kleine Rechteck in der Mitte herunterfallen. Dies soll an einer beliebigen Stelle geschehen. Zur Bestimmung dieser Stelle benutzen wir die RND-Funktion. Diese liefert eine Zufallszahl zwischen 0 und 1. Wir benötigen aber eine Zufallszahl zwischen 1 und 26, und zwar als ganze Zahl. Mit

$R = \text{INT}(26 * \text{RND})$ erhalten wir

Zufallszahlen zwischen 0 und 25, da die INT-Funktion immer die nächste kleinere Zahl als Ergebnis ausgibt. Mit

$R = \text{INT}(26 * \text{RND}) + 1$ erhalten wir die

Zufallszahlen im richtigen Bereich. In dem Programm in Abbildung 3.6 beginnt in Zeile 200 das Herunterfallen des Balles. Das Mittelstück in

unserem Raumschiff wird durch  ersetzt und das Zeichen  von V=0 bis V=18 nach unten fallen gelassen. Wenn es unten angelangt ist wird eine neue Zufallszahl bestimmt und das Raumschiff fliegt weiter.

Das Programm für das Herunterfallen zeigt Abbildung 3.6.

```

200 PRINT AT V,H+2;"■ "
210 PRINT AT 1,H+2;"■"
220 FOR V=1 TO 18
230 PRINT AT V,H+2;" "
240 PRINT AT V+1,H+2;"■"
250 NEXT V

```

3.6 Ballabwurf

Die nächste Erweiterung, die wir durchführen, ist die Einführung eines kleinen Auffangkorbes für die herunterfallenden Bälle. Dieser Auffangkorb soll durch die Tasten ← und → hin- und herbewegt werden. Dies geschieht in einem Unterprogramm ab Zeile 300. Dieses Unterprogramm muß sowohl beim Hin- und Herfliegen des Raumschiffs und auch beim Herunterfallen des Balles durchlaufen werden. Mit der Funktion INKEY\$ wird abgefragt, ob die Taste 5 (←) oder die Taste 8 (→) gedrückt wurde. Das vollständige Programm zeigt Abbildung 3.7.

```

100 LET A$=" ■■■■ "
105 LET X=0
110 LET V=0
115 LET R= INT (26*RND)+1
120 FOR H=0 TO 26
130 PRINT AT V,H;A$
132 GOSUB 300
135 IF H=R THEN GOTO 200
140 NEXT H
150 FOR H=26 TO 0 STEP -1
160 PRINT AT V,H;A$
162 GOSUB 300
165 IF H=R THEN GOTO 200
170 NEXT H
180 GOTO 120

```

```

200 PRINT AT V,H+2;"■"
210 PRINT AT 1,H+2;"■"
220 FOR V=1 TO 18
230 PRINT AT V,H+2;" "
240 PRINT AT V+1,H+2;"■"
245 IF V=16 AND X=H OR V=16 AND X=H+1 THEN
    GOTO 260
247 GOSUB 300
250 NEXT V
260 LET V=0
270 LET R= INT (26*RND)+1
280 GOTO 140
300 PRINT AT 17,X;" ■ "
310 IF INKEY$="8" THEN LET X=X+1
320 IF INKEY$="5" THEN LET X=X-1
330 RETURN

```

3.7 Vollständiges Programm Fangen

Nun müssen wir noch feststellen ob der Ball gefangen wurde. Wenn $V=16$ ist und $X = H$ oder $H + 1$, dann wurde der Ball gefangen.

Dieses Spiel ist recht einfach und sollte nur dazu dienen, einige Grundelemente von Grafikspielen aufzuzeigen. Es wurde auch darauf verzichtet Punktzahlen einzublenden. Aus diesen Grundelementen lassen sich viele andere Spiele zusammensetzen. Man kann zum Beispiel das Raumschiff mit den Tasten \leftarrow und \rightarrow steuern und den Abwurf mit der Taste \downarrow auslösen. Dazu lässt man, durch Zufallszahlen erzeugt, kleine Schiffe über den Bildschirm wandern. Der Spieler muß nun versuchen, diese zu treffen. Oder man lässt Figuren, wiederum durch Zufallszahlen erzeugt, auf dem Bildschirm erscheinen und wieder verschwinden. Der Spieler muß versuchen diese Figuren vor dem Verschwinden zu treffen.

3.2 Schießbude

Diese Elemente werden auch im Spiel "Schießbude" verwendet (Abbildung 3.8).

```

100 FOR I=0 TO 2
110 PRINT AT I,0;"■"
120 NEXT I
130 FOR I=3 TO 20
140 PRINT AT I-3,0;" "
150 PRINT AT I,0;"■"
152 LET V=1
155 IF INKEY$="8" THEN GOSUB 300
160 NEXT I
170 FOR I=20 TO 3 STEP -1
180 PRINT AT I,0;" "
190 PRINT AT I-3,0;"■"
192 LET V=2
195 IF INKEY$="8" THEN GOSUB 300
200 NEXT I
210 GOTO 130
300 GOSUB 500
305 FOR H=1 TO L
310 PRINT AT I-V,H;" ■"
320 NEXT H
325 PRINT AT 21,10;A; AT 21,14;S
327 IF S=0 THEN GOTO 570
330 RETURN
400 FOR I=0 TO 20
410 PRINT AT I,28;"▨"
420 NEXT I
430 FOR J=1 TO 3
440 LET R= INT (18*RND)+1
450 PRINT AT R,28;" "
455 LET P(J)=R
460 NEXT J
470 RETURN
500 LET L=27
510 LET K=I-V
520 IF K=P(1) OR K=P(2) OR K=P(3) THEN LET L=30
530 IF L=30 THEN LET A=A+10
540 LET S=S-1
560 RETURN
570 PRINT AT 10,10;"ENDE"
580 STOP
1000 DIM P(3)
1010 GOSUB 400
1020 LET A=0
1030 LET S=10
1040 GOTO 100

```

Das Programm wird mit GOTO 1000 gestartet. Zuerst wird an der rechten Bildhälfte eine Wand aufgebaut und mit 3 zufälligen Löchern versehen. Sind weniger als 3 Löcher zu sehen, so wurden durch die RND Funktion 2 gleiche Zufallszahlen erzeugt. Die Höhen dieser Löcher in vertikaler Richtung werden in den Variablen P(J) gespeichert.

A ist die erreichte Punktzahl, sie wird nach jedem Durchschuß um 10 erhöht. S ist die maximale Zahl der Schüsse. Im Programmteil 100 bis 210 wird das "Paddle" auf und ab bewegt. Der Schuß wird mit der → Taste ausgelöst.

Danach wird zuerst geprüft, ob in dieser Höhe ein Loch in der Mauer ist (Zeile 520). Wenn dies der Fall ist, fliegt er mit L=30 durch die Wand, sonst bleibt er mit L=27 darin stecken. Die Schußbewegung geschieht im Unterprogramm 300, in der schon früher beschriebenen Weise.

Programmänderung:

Die Löcher müssen durch möglichst wenig Schüsse verstopft werden. Die Schußlänge L ist dann immer 27. Wenn ein Loch verstopft ist wird der entsprechende Wert von P (J) auf Null gesetzt. Sind alle drei Werte von P (J) = 0 so ist das Spiel beendet.

3.3 Käfer

Im Spiel "Käfer" wird eine Figur durch die 4 Cursortasten ←, →, ↑ und ↓ über den Bildschirm bewegt. Dieser Bewegung ist aber eine, durch Zufallszahlen erzeugte Bewegung überlagert. Die Aufgabe dieses Spiels ist es, den Käfer in das Ziel, dargestellt durch ein + Zeichen in Bildmitte, zu bringen und mit der Taste 9 abzuschießen. Das Programm kann durch RUN oder GOTO 100 gestartet werden. Die Variable A bestimmt die Spieldauer. Sie wird nach jeder Bewegung des Käfers um eins erniedrigt. Ist sie Null, dann ist das Spiel beendet. Die Zahl der Treffer wird angezeigt. Das Programm zeigt Abbildung 3.9.

```

100 LET A=600
110 LET B=0
120 LET H= INT (RND*18)
130 LET V= INT (RND*26)
140 PRINT AT H-1,V+1;"      "
150 PRINT AT H,V;"  ■■  "
160 PRINT AT H+1,V;" ■■  "
170 PRINT AT H+2,V+1;"      "
180 PRINT AT 11,15;"+"
190 LET A=A-1
200 IF A=0 THEN GOTO 300
210 IF INKEY$="5" THEN LET V=V-1
220 IF INKEY$="6" THEN LET H=H+1
230 IF INKEY$="7" THEN LET H=H-1
240 IF INKEY$="8" THEN LET V=V+1
250 IF INKEY$="9" THEN GOTO 270
260 GOTO 400
270 IF H >= 10 AND H <= 11 AND V >= 13 AND V
   <= 14 THEN LET B=B+1
280 CLS
300 PRINT AT 0,0;"TREFFER=";B
310 GOTO 120
400 LET R= INT (RND*4+.5)
410 IF R=1 THEN LET H=H+1
420 IF R=2 THEN LET V=V+1
430 IF R=3 THEN LET H=H-1
440 IF R=4 THEN LET V=V-1
450 GOTO 140

```

3.9 Käfer

3.4 PENG

Im Spiel "PENG" wird wiederum ein Zeichen, der inverse Cursor auf dem Bildschirm bewegt. Die Richtung in der er sich bewegt, kann durch die Cursortasten bestimmt werden. Der Weg beginnt irgendwo auf dem Bildschirm, immer mit der Laufrichtung nach unten. Durch die Taste ← wird er nach links, durch die Taste → nach rechts und entsprechend mit den beiden anderen Tasten nach oben und unten umgelenkt. In dieser Richtung läuft der inverse Cursor dann weiter und zieht eine Spur hinter sich her. Dieser Cursor soll nun so mit den Tasten gelenkt werden, daß er möglichst lang auf dem Bildschirm bleibt, nie am Rand

anstößt oder seine eigene Spur kreuzt. Sonst wird PENG und die Zahl der Schritte ausgegeben. Starten des Programms mit RUN oder GOTO 100.

```
100 LET A=700
105 LET N=0
120 LET H= INT (RND*18)
130 LET V= INT (RND*26)
140 PRINT AT H,V;"■"
210 IF INKEY$="5" THEN LET A=600
220 IF INKEY$="6" THEN LET A=700
230 IF INKEY$="7" THEN LET A=800
240 IF INKEY$="8" THEN LET A=900
250 GOTO A
290 IF H<0 OR H>19 THEN GOTO 500
300 IF V<0 OR V>29 THEN GOTO 500
310 LET N=N+1
330 GOTO 210
500 PRINT AT 10,16;"PENG"; AT 10,20;N
510 STOP
515 CLS
520 GOTO 100
600 LET V=V-1
605 PRINT AT H,V;"■"
610 PRINT AT H,V-1;
620 GOTO 1000
700 LET H=H+1
705 PRINT AT H,V;"■"
710 PRINT AT H+1,V;
720 GOTO 1000
800 LET H=H-1
805 PRINT AT H,V;"■"
810 PRINT AT H-1,V;
820 GOTO 1000
900 LET V=V+1
905 PRINT AT H,V;"■"
910 PRINT AT H,V+1;
1000 IF PEEK ( PEEK 16398+256* PEEK 16399)
=128 THEN GOTO 500
1010 GOTO 290
```

3.10 PENG

In dem Programm in Abbildung 3.10 sind einige Besonderheiten des ZX81 BASIC verwendet worden, darunter die Möglichkeit den GOTO-Befehl mit einer Variablen als Zielparameter zu versehen. Die Zeilen 210 bis 250 entsprechen der in anderen BASIC-Sprachen vorhandenen Befehl

ON A GOTO 600,700,800,900

Das bedeutet, wenn $A = 1$ ist, wird nach 600, mit $A = 2$ nach 700 usw. gesprungen. Der ZX81 kennt diesen Befehl nicht, so daß man ihn durch einen berechneten Sprung ersetzen muß. Die gleiche Wirkung wie oben hat der Befehl

GOTO A * 100 + 500 beim ZX81.

In der Zeile 1000 wird der Inhalt einer Adresse untersucht. In den Zeilen 16398 und 16399 ist die Adresse der nächsten PRINT-Position auf dem Bildschirm enthalten (siehe Handbuch Seite 178). In der Zelle 16398 ist der niederwertige Adressteil, in der Zelle 16399 der höherwertige Adressteil gespeichert (siehe auch Kapitel 6). Diese Adresse erhält man durch

PEEK 16398 + 256 * PEEK 16399

in dezimaler Form.

Das, was in dieser Adresse gespeichert ist erhält man dann durch

PEEK (PEEK 13398 + 256 * PEEK 16399)

In den Programmteilen 600 bis 900 ist der Cursor schon unsichtbar auf die nächste PRINT-Position gesetzt worden. Dann wird in Zeile 1000 der Inhalt dieser Zelle geholt. Wenn dieser Inhalt 128 ist, dies entspricht dem inversen Cursor, dann ist er gerade an der eigenen Spur irgendwo angestoßen. Das Programm wird dann beendet.

Programmänderung:

Die Abfrage nach dem Anstoßen kann entfallen. Das Programm kann man dann verkürzen und zum Zeichnen von Linienmustern verwenden.

Eine andere Erweiterung ist das folgende Programm

3.5 Schlange

In diesem Programm wird wieder der inverse Cursor durch die 4 Tasten über den Bildschirm bewegt. Diesmal ist es eine Schlange, die plötzlich auftauchende Zeichen fressen soll, ohne sich dabei in den eigenen Schwanz zu beißen. Die Zeichen haben verschiedene Werte, die zu der Gesamtpunktzahl addiert werden. Das Ziel des Spiels ist, möglichst viele Punkte zu sammeln. Das Programm zeigt Abbildung 3.11.

```
100 LET A=700
105 LET N=0
110 LET P=0
120 LET H= INT (RND*18)
130 LET V= INT (RND*26)
140 PRINT AT H,V;"■"
210 IF INKEY$="5" THEN LET A=600
220 IF INKEY$="6" THEN LET A=700
230 IF INKEY$="7" THEN LET A=800
240 IF INKEY$="8" THEN LET A=900
250 GOTO A
290 IF H<0 OR H>19 THEN GOTO 500
300 IF V<0 OR V>29 THEN GOTO 500
310 LET N=N+1
320 GOSUB 2000
330 GOTO 210
500 PRINT AT 10,16;"PENG"; AT 10,23;P
510 STOP
515 CLS
520 GOTO 100
600 LET V=V-1
605 PRINT AT H,V;"■"
610 PRINT AT H,V-1;
620 GOTO 1000
700 LET H=H+1
705 PRINT AT H,V;"■"
710 PRINT AT H+1,V;
720 GOTO 1000
800 LET H=H-1
805 PRINT AT H,V;"■"
810 PRINT AT H-1,V;
820 GOTO 1000
900 LET V=V+1
```

```

905 PRINT AT H,V;"■"
910 PRINT AT H,V+1;
1000 LET Q= PEEK ( PEEK 16398+256* PEEK 16399)
1010 IF Q=128 THEN GOTO 500
1020 IF Q=136 THEN LET P=P+10
1030 IF Q=6 THEN LET P=P+20
1040 IF Q=134 THEN LET P=P+30
1050 GOTO 290
2000 LET X= INT (RND*26)
2010 LET Y= INT (RND*18)
2020 IF N<3 THEN PRINT AT Y,X;"▣"
2030 IF INT (N/10)=N/10 AND N>51 THEN PRINT
    AT Y,X;"■"
2040 IF INT (N/10)=N/10 AND N<51 THEN PRINT
    AT Y,X;"■"
2999 RETURN

```

3.11 Schlange

3.6 Der freie Fall

In vielen Spielen wird geworfen oder geschossen. Dabei werden für die Flugbahn die Gleichungen des schiefen Wurfes oder des freien Falls verwendet.

Wir wollen hier mit den PLOT und UNPLOT-Befehlen den freien Fall und den schiefen Wurf simulieren. Auf unseren Bildschirm ist das Koordinatensystem in Abbildung 3.12 gezeichnet. Für die Plotbefehle haben die Ecken des Bildschirms folgende Koordinaten. Für die linke untere Ecke $X=0$, $Y=0$, für die linke obere Ecke $X=0$, $Y=43$, für die rechte untere Ecke $X=63$, $Y=0$ und für die rechte obere Ecke $X=43$, $Y=63$. In Y-Richtung soll der Punkt mit $Y=40$ der Höhe $H=2000$ m entsprechen. In X-Richtung soll der Punkt mit $X=60$ der Weite 3000 m entsprechen. Die Gleichungen für den freien Fall im Koordinatensystem von Abbildung 3.12 lauten:

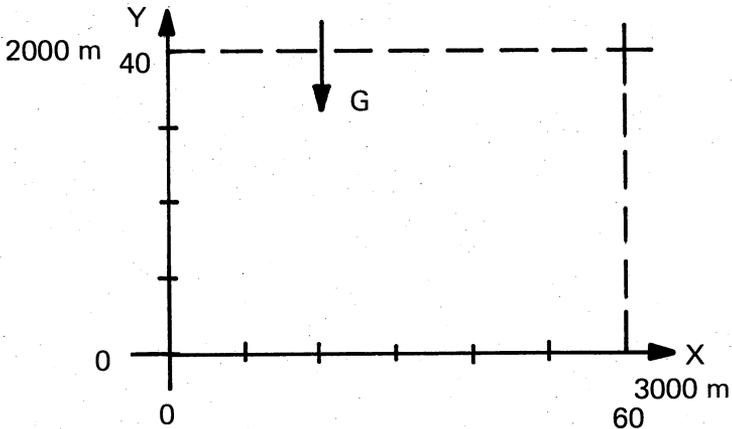
$$V = -G * T + V_0$$

und

$$Y = -G/2 * T^2 + V_0 * T + Y_0$$

Dabei ist G die Erdbeschleunigung, T die Zeit in Sekunden und V_0 die Anfangsgeschwindigkeit zur Zeit $T=0$.

Y ist die Höhe und Y_0 die Anfangshöhe zur Zeit $T=0$.



3.12 Koordinatensystem für freien Fall und schiefen Wurf

In unserem ersten Programmbeispiel simulieren wir den freien Fall aus 2000 m Höhe. Von dort lassen wir einen Körper mit der Anfangsgeschwindigkeit $V_0=0$ fallen. Das Programm dazu zeigt Abbildung 3.13.

```
100 LET Y0=2000
110 LET V0=0
120 LET G=10
130 LET T=0
140 LET H=Y0
200 LET V=-G*T+V0
210 LET Y=-G/2*T*T+V0*T+Y0
220 UNPLOT 30,H/50
230 PLOT 30,Y/50
240 LET H=Y
250 IF H <= 0 THEN STOP
260 LET T=T+0.5
270 GOTO 200
```

3.13 Freier Fall

Die Anfangswerte sind also $V_0=0$ und $Y_0=2000$. Nach jeder Berechnung einer neuen Höhe, wird die Zeit um Eins erhöht. Durch Ändern des Zeitzuwachses in Zeile 260 kann die Bewegung schneller oder langsamer gemacht werden.

Das Programm wird nun so abgeändert, das der Körper mit der Anfangsgeschwindigkeit $V_0=0$ von der Höhe $H = Y_0 = 0$ nach oben geworfen wird. Die Anfangsgeschwindigkeit wird vorgegeben. Das Programm dazu zeigt Abbildung 3.14. Für V_0 kann man die Werte 10, 100, usw. bis etwa 200 eingeben, ohne daß der Ball über den oberen Bildrand hinausfliegt.

```

100 LET Y0=0
105 PRINT "V0=";
110 INPUT V0
115 PRINT V0

120 LET G=10
130 LET T=0
140 LET H=Y0
200 LET V=-G*T+V0
210 LET Y=-G/2*T*T+V0*T+Y0
220 UNPLOT 30,H/50
230 PLOT 30,Y/50
240 LET H=Y
250 IF H <= 0 THEN STOP
260 LET T=T+0.5
270 GOTO 200

```

3.14 Hochwerfen

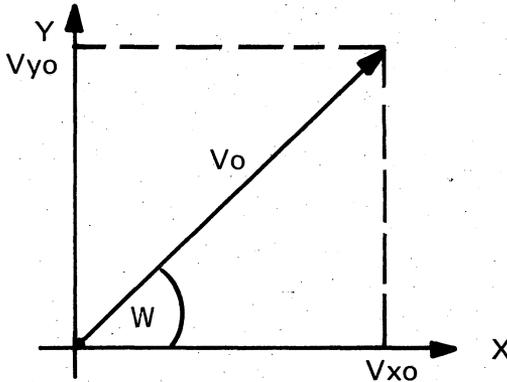
3.7 Der schiefe Wurf

Nach der nur eindimensionalen Bewegung, wollen wir nun noch eine Bewegung in X-Richtung überlagern. Dies ist eine Bewegung mit konstanter Geschwindigkeit $V_X = C$. Damit erhält man für den Weg $X = C * T + X_0$. Wir haben dann die 4 Gleichungen:

$$\begin{aligned}
 V_y &= -G * T + V_{y0} \\
 V_x &= V_{x0} \\
 Y &= -G/2 * T^2 + V_{y0} * T + Y_0 \\
 X &= V_{x0} * T + X_0
 \end{aligned}$$

Alle unsere Wurfparabeln sollen in der linken unteren Ecke bei $x = 0$, $y = 0$ beginnen. Damit sind die Anfangsbedingungen für die Ausgangswerte Y_0 und $X_0 = 0$.

Als Anfangsbedingungen für den Wurf wollen wir den Abschlußwinkel und den Betrag der Anfangsgeschwindigkeit vorgeben. Abbildung 3.15 zeigt, wie sich in Abhängigkeit vom Anfangswinkel W der Betrag der Anfangsgeschwindigkeit V_0 auf die Anfangsgeschwindigkeiten in X-Richtung (V_{xo}) und in Y-Richtung (V_{yo}) aufteilt.



3.15 Bestimmung der Anfangsgeschwindigkeit in X- und Y-Richtung

Wir erhalten für $V_{xo} = V_0 \cdot \cos W$ und $V_{yo} = V_0 \cdot \sin W$

Das Programm hierfür zeigt Abbildung 3.16. Für die Stärke können Werte zwischen 100 und 200 eingegeben werden. Die Wurfparabeln werden gezeichnet. Soll nun der Wurf gezeigt werden, so ist folgende Zeile einzufügen:

```
320 UNPLOT L/50, H/50
```

```
100 LET X0=0
110 LET Y0=0
120 PRINT AT 0,0;"WINKEL=";
130 INPUT W
```

```

140 PRINT W
145 LET W=W*2*PI/360
150 PRINT "STAERKE=";
160 INPUT V0
170 PRINT V0
180 LET VX=V0* COS W
190 LET VY=V0* SIN W
200 LET G=10
210 LET T=0
220 LET H=0
230 LET L=0
300 LET Y=-G/2*T*T+VY*T
305 IF Y<0 THEN GOTO 120
310 LET X=VX*T
330 PLOT X/50,Y/50
340 LET L=X
350 LET H=Y
355 LET T=T+0.5
360 GOTO 300

```

3.16 Wurfparabeln

3.8 Ente

Für das Programm "Ente" in Abbildung 3.17 wird das Wurfparabelprogramm verwendet. Ziel ist es, eine kleine Ente am unteren rechten Bildrand abzuschiesen. Der Abschlußwinkel und die Abschlußstärke werden eingegeben. In der Trefferabfrage in Zeile 405 wird festgestellt ob getroffen wurde. Wenn nicht, dann darf weitergeschossen werden.

```

100 LET X0=0
110 LET Y0=0
115 PRINT AT 21,27;"■"
120 PRINT AT 0,0;"WINKEL=";
130 INPUT W
140 PRINT W
145 LET W=W*2*PI/360
150 PRINT "STAERKE=";
160 INPUT V0
170 PRINT V0
180 LET VX=V0* COS W

```

```

190 LET VY=V0* SIN W
200 LET G=10
210 LET T=0
220 LET H=0
230 LET L=0
300 LET Y=-G/2*T*T+VY*T
305 IF Y<0 THEN GOTO 400
310 LET X=VX*T
320 UNPLOT L/50,H/50
330 PLOT X/50,Y/50
340 LET L=X
350 LET H=Y
355 LET T=T+0.5
360 GOTO 300
400 LET X=X/50
405 IF INT X<52 OR INT X>55 THEN GOTO 120
410 PRINT AT 10,10;"TREFFER"
420 PRINT AT 21,27;" "
430 STOP

```

3.17 ENTE

Programmänderungen:

Im vorliegenden Fall ist es nicht schwer, wenn man einmal getroffen hat, die kleine Ente wieder zu treffen. Erweiterungen sind: Die Ente wird zufällig an eine beliebige Stelle am unteren Bildrand gesetzt. Die nächste Erschwerung wäre ein Hindernis in der Bildmitte, das überschossen werden muß. Dann kann man noch Gegenwind einführen, der die Flugbahn bremst. Dies kann man durch Einfügen von Zeile

308 LET VX = VX - B, wobei B eine Zahl zwischen -3 und +3 sein kann. Ein negatives Vorzeichen bedeutet, daß kein Gegenwind herrscht, sondern der Flug des Körpers vom Wind unterstützt wird. Dieses Spiel kann leicht zu dem bekannten Spiel "Artillerie" erweitert werden, wobei 2 Gegenspieler versuchen, sich gegenseitig abzuschießen.

Diese Spiele sind nicht als vollständige Spiele gedacht, sondern sie sollen Anregungen zu Weiterentwicklung oder zur Neuprogrammierung geben.



Das ZX81 Minirechenzentrum

Programme für die Schule

4. Programme für die Schule

Der ZX81 eignet sich auch vorzüglich, einige der bei Schülern unbeliebten Hausaufgaben zu lösen. Hier sollen einige kleinere Programme vorgestellt werden.

4.1 GGT

Das Programm GGT in Abbildung 4.1 berechnet den größten, gemeinsamen Teiler zweier Zahlen A und B.

```
100 PRINT "GROESTER GEMEINSAMER TEILER"  
110 PRINT "A=";  
120 INPUT A  
130 PRINT A  
140 PRINT "B=";  
150 INPUT B  
160 PRINT B  
170 LET Q= INT (A/B)  
180 LET R=A-Q*B  
190 LET A=B  
200 LET B=R  
210 IF R>0 THEN GOTO 170  
220 PRINT "GGT=";A
```

4.1 GGT

4.2 KGV

Das Programm in Abbildung 4.2 berechnet das kleinste gemeinsame Vielfache von 3 Zahlen A, B und C.

```
100 PRINT "KLEINSTES GEMEINSAMES VIELFACHES"
110 PRINT "A=";
120 INPUT A
130 PRINT A
140 PRINT "B=";
150 INPUT B
160 PRINT B
170 PRINT "C=";
180 INPUT C
190 PRINT C
200 LET X=A
210 IF INT (X/A)=X/A THEN GOTO 240
220 LET X=X+1
230 GOTO 210
240 IF INT (X/B)=X/B THEN GOTO 270
250 LET X=X+1
260 GOTO 210
270 IF INT (X/C)=X/C THEN GOTO 300
280 LET X=X+1
290 GOTO 210
300 PRINT "KGV=";X
```

4.2 KGV

4.3 Primfaktorenzerlegung

Das Programm in Abbildung 4.3 zerlegt eine Zahl in seine Primfaktoren.

```
100 PRINT "PRIMFAKTOREN-ZERLEGUNG"
110 PRINT
120 PRINT "ZAHL=";
130 INPUT Z
140 PRINT Z
150 PRINT Z;"=";
160 IF Z<0 THEN PRINT "-";
```

```

170 PRINT "1";
200 LET Z= ABS Z
205 LET E=Z
208 LET T=1
210 FOR I=2 TO Z
220 LET N=0
230 IF Z/I <> INT (Z/I) THEN GOTO 270
240 LET Z=Z/I
250 LET N=N+1
260 GOTO 230
270 IF N=0 THEN GOTO 300
280 FOR J=1 TO N
285 PRINT "*" ; I ;
287 LET T=T*I
288 IF T=E THEN GOTO 310
290 NEXT J
300 NEXT I
310 PRINT
320 PRINT "NOCHMAL (J/N) "
330 IF INKEY$="" THEN GOTO 330
340 IF INKEY$="N" THEN STOP
350 CLS
360 GOTO 100

```

4.3 Primfaktorenzerlegung

4.4 Lösung der quadratischen Gleichung

Das folgende Programm in Abbildung 4.4 bestimmt die Nullstellen einer Quadratischen Gleichung der Form $a \cdot x^2 + Bx + C = 0$

über die Formel
$$X_{1,2} = -\frac{b}{2a} \pm \frac{1}{2a} \sqrt{b^2 - 4ac}$$

```

100 PRINT "A*X*X+B*X+C=0"
110 PRINT "A=" ;
120 INPUT A
130 PRINT A
140 PRINT "B=" ;
150 INPUT B
160 PRINT B

```

```

170 PRINT "C=";
180 INPUT C
190 PRINT C
200 LET Q=B*B-4*A*C
210 IF Q<0 THEN GOTO 300
220 LET Q= SQR Q/(2*A)
230 LET C=-B/(2*A)
240 LET X1=C+Q
245 PRINT "X1=";
250 PRINT X1
260 LET X2=C-Q
265 PRINT "X2=";
270 PRINT X2
280 STOP
300 PRINT "KEINE REELLE LOESUNG"
310 STOP

```

4.4 Lösung der quadratischen Gleichung

4.5 NUSU

Das Programm NUSU bestimmt die Nullstellen einer Funktion in einem gegebenen Intervall. Vorgegeben werden die Intervallgrenzen XA, XE die Schrittweite H und die Genauigkeit EPS, mit welcher die Nullstelle bestimmt werden soll. Die Funktion ist im Unterprogramm FKT (Zeilen 1000 – 1020) programmiert. Das im Programm verwendete Polynom hat die Nullstellen bei $x = -3$, $x = 2$ und $x = 5$.

Eingabe für die Intervallgrenzen zum Beispiel XA = - 10, XE = + 10, H = 0,1 und EPS = 0,0001 (1E - 4).

```

100 PRINT "XA=";
110 INPUT XA
120 PRINT XA
130 PRINT "XE=";
140 INPUT XE
150 PRINT XE
160 PRINT "H=";
170 INPUT H
180 PRINT H
190 PRINT "EPS=";

```

```

200 INPUT E
210 PRINT E
220 LET FKT=1000
230 LET NUSU=2000
240 LET X=XA
250 GOSUB FKT
260 LET YA=Y
270 LET XA=X
280 LET X=XA+H
290 IF X>XE THEN STOP
300 GOSUB FKT
310 IF SGN Y= SGN YA THEN GOTO 500
320 LET YN=Y
330 LET XN=X
340 GOSUB NUSU
350 PRINT "NST BEI X=";X
360 IF X<XE THEN LET X=X+E
370 GOTO 250
500 LET YA=Y
510 LET XA=X
520 GOTO 280
1000 REM FKT
1010 LET Y=X*X*X-4*X*X-11*X+30
1020 RETURN
2000 REM NUSU
2010 IF ABS (Y)<E THEN RETURN
2020 LET X=(XN-XA)/2+XA
2030 GOSUB FKT
2040 IF ABS Y<E THEN RETURN
2050 IF SGN Y= SGN YN THEN GOTO 2100
2060 LET YA=Y
2070 LET XA=X
2080 GOTO 2020
2100 LET YN=Y
2110 LET XN=X
2120 GOTO 2020

```

4.5 NUSU

4.6 FKTW

Das Programm in FKTW (Abbildung 4.6) bestimmt den Funktionswert

eines Polynoms N-ten Grades, wobei N hier auf 10 begrenzt ist. Das Polynom hat die Form:

$$a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 = y$$

Die Vorfaktoren sind in einer Matrix W(1) bis W(N+1) gespeichert. Als Lösungsverfahren wird das Hornerschema angewendet. Ist in dem Polynom ein Term nicht vorhanden, so ist bei der Eingabe des Vorfaktors 0 einzugeben.

```

100 GOTO 1000
200 REM POLYEIN
210 PRINT "N=";
220 INPUT N
230 PRINT N
240 DIM W(N+1)
250 FOR I=1 TO N+1
255 PRINT "A(";I-1;")=";
260 INPUT W(I)
270 PRINT W(I)
280 NEXT I
290 RETURN
300 LET Y=W(N+1)*X
310 FOR I=N TO 2 STEP -1
320 LET Y=(Y+W(I))*X
330 NEXT I
335 LET Y=Y+W(1)
340 RETURN
1000 GOSUB 200
1010 PRINT "X=";
1015 INPUT X
1017 PRINT X
1020 GOSUB 300
1030 PRINT "Y=";Y
1040 PRINT "NEUES X? (J/N) "
1050 IF INKEY$="" THEN GOTO 1050
1060 IF INKEY$="J" THEN GOTO 1010
1070 STOP

```

4.6 FKTW

4.7 POLYM

Das Ausmultiplizieren von Polynomen gehört immer noch zu den schwierigsten Aufgaben in der Schulmathematik. Das Programm POLYM (Abbildung 4.7) multipliziert 1 Polynom N-ten Grades mit einem Polynom M-ten Grades. N und M sind nur durch den Speicherplatz beschränkt. Die Ergebnisse werden ausgedruckt. Für die Eingabe beider Polynome wird die gleiche Eingaberoutine verwendet. Die zuerst eingegebenen Koeffizienten werden in eine Matrix A (K) umgespeichert, die als zweite eingegebenen Koeffizienten bleiben in W(N) erhalten. Das Ergebnis wird in einer Matrix E(L) gespeichert.

```
100 GOTO 1000
200 REM POLYEIN
210 PRINT "N=";
220 INPUT N
230 PRINT N
240 DIM W(N+1)
250 FOR I=1 TO N+1
255 PRINT "A(";I-1;")=";
260 INPUT W(I)
270 PRINT W(I)
280 NEXT I
290 RETURN
1000 PRINT "1.POLYNOM"
1010 GOSUB 200
1020 LET K=N
1030 DIM A(K+1)
1040 FOR I=1 TO K+1
1050 LET A(I)=W(I)
1060 NEXT I
1070 CLS
1080 PRINT "2.POLYNOM"
1090 GOSUB 200
1100 LET L=N+K+1
1105 DIM E(L)
1110 FOR I=1 TO L
1120 LET E(I)=0
1130 NEXT I
1140 FOR I=1 TO N+1
1150 FOR J=1 TO K+1
```

```

1155 LET L=I+J-1
1160 LET E(L)=E(L)+W(I)*A(J)
1170 NEXT J
1180 NEXT I
1190 CLS
1200 LET L=K+N+1
1210 FOR I=1 TO L
1220 PRINT "E(";I-1;")=";E(I)
1230 NEXT I

```

4.7 POLYM

4.8 Darstellung von Funktionen

In vielen Fällen ist es sehr nützlich, den Verlauf von Funktionen darzustellen. Der ZX81 besitzt zwar keine hochauflösende Grafik, doch mit der Bildauflösung von 64 Punkten in horizontaler und 44 Punkten in vertikaler Richtung lässt sich immerhin ein Überblick über den Funktionsverlauf geben.

Für unsere Beispiele legen wir den Nullpunkt des Koordinatensystems in die Mitte des Bildschirms. Für andere Funktionen wird man den Koordinatenursprung eventuell an eine andere Stelle legen. Das vollständige Programm zeigt Abbildung 4.8. Im Unterprogramm 100 wird das Achsenkreuz gezeichnet.

Im Unterprogramm wird der X-Bereich mit XA für den Anfangswert von X, XE den Endwert von X, YA den Anfangswert von Y und YE den Endwert von Y eingegeben. H ist die Schrittweite. Die Schrittweite H und den Wertebereich von Y muß man vorher abschätzen. Das Programm berechnet daraus die Maßstabsfaktoren MX und MY. Die Funktion ist im Unterprogramm 500 in Zeile 510 programmiert. Durch Ändern dieser Zeile können andere Funktionen gezeichnet werden. Als Laufparameter in der FOR-NEXT-Schleife kann die Variable X nicht verwendet werden, da diese in der Schleife selbst verändert wird. Die Funktion wird nur gezeichnet wenn der Bildpunkt innerhalb des Bildschirms liegt. Damit keine Fehlermeldung auftritt, erfolgt vor dem Plot-Aufruf die Abfrage ob der Bildpunkt innerhalb der Ränder liegt.

3 Beispiele:

1. $y = \sin x + \sin 2x$

$$XA = -4, XE = 4, YA = -1, YE = 1, H = 0,1$$

2. $y = (x^2 + 1) / (x^2 - 1)$

$$XA = -2, XE = 2, YA = -5, YE = 5, H = 0,1$$

3. $y = \sin x/x$

$$XA = -10, XE = 10, YA = -1, YE = 1, H = 0,3$$

Bei den Beispielen 2 und 3 ist bei der Programmierung der Funktion darauf zu achten, daß keine Division durch Null auftritt.

```
050 GOTO 1000
100 FOR I=0 TO 63
110 PLOT I,22
120 NEXT I
130 FOR I=0 TO 43
140 PLOT 31,I
150 NEXT I
160 PRINT AT 0,14;"Y"
170 PRINT AT 9,31;"X"
180 RETURN
200 LET X0=31
210 LET Y0=22
220 PRINT "XA=";
230 INPUT XA
240 PRINT XA
250 PRINT "XE=";
260 INPUT XE
270 PRINT XE
280 PRINT "YA=";
290 INPUT YA
300 PRINT YA
310 PRINT "YE=";
320 INPUT YE
330 PRINT YE
350 LET MX=63/ ABS (XE-XA)
360 LET MY=43/ ABS (YE-YA)
370 PRINT "H=";
380 INPUT H
390 PRINT H
```

```

399 RETURN
500 FOR L=XA TO XE STEP H
505 LET X=L
510 LET Y= SIN X+ SIN (2*X)
520 LET X=X*MX+X0
530 LET Y=Y*MY+Y0
540 IF X >= 0 AND X <= 63 AND Y >= 0 AND Y <=
550 NEXT L 43 THEN PLOT X,Y
560 RETURN
1000 GOSUB 200
1010 CLS
1020 GOSUB 100
1050 GOSUB 500

```

4.8 PLOT $\sin x + \sin 2x$

4.9 Vokabeln lernen

Mit dem Programm in Abbildung 4.9 kann man Vokabeln lernen und seine Lernleistung prüfen. Die Worte werden wie in ein Wörterbuch eingetragen. Erst die deutsche Bezeichnung, dann die englische (französische, lateinische usw.). Beim Lern- und Abhörteil wird durch eine Zufallszahl zwischen 1 und N, der Zahl der Einträge, ein Begriff ausgewählt. Vorher kann gewählt werden, ob dies ein deutscher oder englischer Ausdruck ist. Der anderssprachige Begriff wird eingegeben und mit dem gespeicherten Wort verglichen. Stimmen beide überein, kann man weitermachen oder aufhören. Hat man aber ein falsches Wort oder eine falsche Schreibweise eingegeben, so wird eine nochmalige Eingabe verlangt. Nach der 3. falschen Eingabe wird das richtige Wort ausgegeben.

Alle eingegebenen Begriffe können auf Bildschirm ausgegeben werden.

```

010 DIM D$(100,15)
020 DIM A$(100,15)
030 LET N=0
040 LET E$="ENGLISCH"
050 LET B$="DEUTSCH"
060 DIM I$(1,15)

```

```

100 CLS
105 PRINT AT 3,20;"E) INGEBEN"
110 PRINT AT 4,20;"L) ERNEN"
115 PRINT AT 5,20;"B) EENDEN"
117 PRINT AT 6,20;"A) USGEBEN"
120 IF INKEY$="" THEN GOTO 120
130 IF INKEY$="E" THEN GOTO 1000
140 IF INKEY$="L" THEN GOTO 2000
150 IF INKEY$="B" THEN GOTO 200
160 IF INKEY$="A" THEN GOTO 3000
199 GOTO 100
200 CLS
210 PRINT AT 10,10;"ENDE"
220 STOP
1000 LET N=N+1
1002 CLS
1005 PRINT AT 10,2;B$;
1010 INPUT D$(N)
1020 PRINT AT 10,12;D$(N)
1030 PRINT AT 11,2;E$
1040 INPUT A$(N)
1050 PRINT AT 11,12;A$(N)
1052 GOSUB 1060
1054 GOTO 1000
1060 PRINT AT 20,10;"WEITER (J/N) "
1070 IF INKEY$="" THEN GOTO 1070
1080 IF INKEY$="N" THEN GOTO 100
1090 RETURN
2000 CLS
2010 PRINT AT 3,20;"D) EUTSCH"
2020 PRINT AT 4,20;"E) ENGLISCH"
2030 LET B=0
2040 IF INKEY$="" THEN GOTO 2040
2050 IF INKEY$="D" THEN LET B=1
2060 CLS
2065 LET Z=0
2070 LET R= INT (RND*N)+1
2080 IF B=0 THEN LET V$=A$(R)
2090 IF B=1 THEN LET V$=D$(R)
2095 PRINT AT 10,9;V$
2100 PRINT AT 11,0;"EINGABE"
2110 INPUT I$(1)
2120 PRINT AT 11,9;I$(1)

```

```

2130 LET W=0
2132 IF B=0 THEN LET V$=D$(R)
2134 IF B=1 THEN LET V$=A$(R)
2136 IF I$(1)=V$ THEN LET W=1
2140 IF W=0 THEN PRINT AT 12,9;"IST FALSCH"
2150 IF W=1 THEN GOTO 2270
2170 PAUSE 100
2180 PRINT AT 11,9;" "
2190 LET Z=Z+1
2200 IF Z=3 THEN GOTO 2250
2210 GOTO 2100
2250 PRINT AT 11,9;A$(R)
2260 LET W=1
2270 PRINT AT 12,9;"IST RICHTIG"
2280 GOSUB 1060
2290 GOTO 2060
3000 CLS
3010 PRINT AT 0,0;B$
3020 PRINT AT 0,16;E$
3025 LET I=2
3030 FOR L=1 TO N
3040 LET I=I+1
3050 PRINT AT I,0;D$(L)
3060 PRINT AT I,16;A$(L)
3070 IF I=18 THEN GOTO 3200
3080 NEXT L
3200 GOSUB 1060
3210 IF L>N THEN GOTO 100
3220 GOTO 3080

```

4.9 Vokabeln lernen

Programmbeschreibung:

Die Zeilen 10 bis 60 sind das Vorprogramm. Dies wird nur dann mit RUN gestartet, wenn eine neue Kassette angelegt werden soll. Sonst wird das Programm immer mit GOTO 100 gestartet. Die Zahl der Einträge ist auf 100, die Buchstabenzahl ist auf 15 begrenzt. Dabei wird eine Eigenschaft des ZX81 ausgenutzt. Programm und Daten gleichzeitig auf Kassette aufzuzeichnen. Der Inhalt der Daten geht bei einem Programmbeginn mit RUN oder mit einer neuen DIM-Anweisung verloren.

Die Zeilen 100 bis 199 enthalten das Menüprogramm mit

E)INGEBEN
L)ERNEN
B)EENDEN
A)USGEBEN

Durch Drücken des Buchstabens wird in den entsprechenden Programmteil gesprungen. Bei der Eingabe wird erst der deutsche, dann der englische Begriff eingegeben. Die Eingabe erfolgt in den Programmzeilen 1000 bis 1090. Für das Lernen und Abhören sind die Zeilen 2000 bis 2290 zuständig. Die Variable B entscheidet zwischen deutsch und englisch, die Variable W ob der eingegebene Begriff richtig ist oder nicht.

Die Ausgabe auf den Bildschirm erfolgt in den Zeilen 3000 bis 3220. nach der Ausgabe von jeweils 16 Begriffen wird angehalten, und, wenn die Frage nach dem Weitermachen mit J beantwortet wurde, die Ausgabe fortgesetzt, wenn noch Begriffe vorhanden waren. Wenn nicht, dann wird ins Menüprogramm gesprungen. Bei Beendigung des Programms wird nach 200 gesprungen und damit das Programm beendet.

Achtung !

Wenn neue Eingaben gemacht wurden, dann muß nach dem Programmende das Programm wieder auf Kassette gespeichert werden, sonst sind die Eingaben verloren.

Programmänderungen:

Der Ausdruck "englisch" im Programm kann durch die beliebige andere Sprache ersetzt werden.

Der Bereich der erzeugten Zufallszahlen kann eingeschränkt werden, so daß nur Teile des Wortschatzes abgefragt werden.

Ein Nachteil des Programms ist es, daß eingegebene Begriffe nicht gelöscht werden können. Dazu braucht man einen Programmteil, der den Begriff sucht und wenn er ihn gefunden hat, ihn durch seinen Nachfolger überschreibt und so die Liste verkürzt (siehe auch Kapitel 5).

Man kann trotzdem einen Eintrag ändern. Man muß nur durch Abzählen herausbekommen, welche Nummer er hat. Nach Beendigung des Programms lässt man sich, wenn z. B. der 22. te deutsche Begriff falsch ist, zuerst durch PRINT D\$(22) den Begriff ausdrucken und dann mit LET D\$="NEUER EINTRAG" ändern.

Weitere Programme nicht nur für die Schule:

4.10 Berechnung der Mehrwertsteuer und des Nettobetrages

Die beiden kleinen Programme in Abbildung 4.10 und 4.11 berechnen die Mehrwertsteuer und den Nettobetrag. Bei der INT-Funktion wird immer auf die nächste kleinere Zahl abgerundet. Aus 10.21 wird 10, aus 10.99 wird auch 10. Um nun eine kaufmännische Rundung zu erzielen, wird vor der Bildung der INT-Funktion 0.5 hinzuaddiert.

```
100 PRINT "BETAG: ";
110 INPUT B
120 PRINT B
125 PRINT " MWST: ";
130 LET M= INT (B+0.5*113)/100-B
140 PRINT M
150 LET S=M+B
160 PRINT " ZUS: ";
170 PRINT S
```

4.10 MWST

```
100 PRINT "BETRAG:";
110 INPUT B
120 PRINT B
130 LET M=B/1.13+0.005
140 LET M= INT (M*100)/100
150 PRINT "NETTO:";M
160 LET N=B-M
170 PRINT "MWST:";N
```

4.11 NETTO

4.11 Berechnung des Wochentages

Das Programm in Abbildung 4.12 berechnet den Wochentag zu einem gegebenen Datum. Aus dem eingegebenen Datum berechnet das Programm in den Zeilen 230 bis 280 eine Kennzahl N für den Wochentag. Aus der Zeichenkette W\$ wird dann die entsprechende Buchstabenkombination herausgesucht.

```
100 PRINT "WOCHENTAG"
110 PRINT "EINGABE:DD.MM.JJJJ"
120 PRINT " DATUM:";
125 INPUT A$
130 PRINT A$
200 LET D= VAL A$(1 TO 2)
210 LET M= VAL A$(4 TO 5)
220 LET J= VAL A$(7 TO 10)
230 IF M>2 THEN GOTO 260
240 LET M=M+12
250 LET J=J-1
260 LET N=D+2*M+ INT (.6*(M+1))+J+ INT (J/4)-
    INT (J/100)+ INT (J/400)+2
270 LET N= INT ((N/7- INT (N/7))*7+.5)
280 LET N=2*N+1
290 LET W$="SASOMODIMIDOFR"
300 PRINT "TAG:";W$(N TO N+1)
```

4.12 WOCHENTAG

4.12 Dezimal Hexadezimal Wandlung

Das Programm in Abbildung 4.13 zeigt die Umwandlung einer Dezimalzahl zwischen 0 und 65535 in eine Hexadezimalzahl.

Das Verfahren, das dabei angewandt wird ist recht einfach. Zuerst wird abgezählt, wie oft 4096 dann 256 und zuletzt 16 in der Zahl vorhanden ist. Beim ZX81 Code folgen auf die Zahlen, ohne Unterbrechung wie bei den ASCII-Zeichen, die Buchstaben. Deshalb ist die einfache Umwandlung in Zeile 900 bis 920 möglich.

Das gleiche Verfahren lässt sich dazu verwenden, Dezimalzahlen in römische Zahlen umzuwandeln.

```

005  REM DEZ-HEX WANDLUNG
010  PRINT "DEZIMALZAHL: ";
020  INPUT N
025  PRINT N
030  LET A=0
032  LET B=0
034  LET C=0
036  IF N>65535 THEN PRINT "ZU GROSS"
038  IF N>65535 THEN GOTO 80
040  IF N >= 4096 THEN GOSUB 500
050  IF N >= 256 THEN GOSUB 600
060  IF N >= 16 THEN GOSUB 700
070  GOSUB 800
080  STOP
500  REM
510  LET N=N-4096
520  LET A=A+1
530  IF N >= 4096 THEN GOTO 510
540  RETURN
600  REM
610  LET N=N-256
620  LET B=B+1
630  IF N >= 256 THEN GOTO 610
640  RETURN
700  REM
710  LET N=N-16
720  LET C=C+1
730  IF N >= 16 THEN GOTO 710
740  RETURN
800  PRINT "HEX-ZAHL: ";
810  LET M=A
820  GOSUB 900
830  LET M=B
840  GOSUB 900
850  LET M=C
860  GOSUB 900
870  LET M=N
880  GOTO 900
900  LET M=M+28
920  PRINT CHR$ M;
930  RETURN

```

4.13 Wandlung DEZ – HEX

Datenverwaltungs- programme

5. Datenverwaltungsprogramme

Die folgenden Programme sind für die Verwaltung von Daten geeignet. Der ZX81 verfügt als externen Speicher über den Kassettenrecorder und das 16k-RAM Modul. Wenn wir für Programm und interne Verwaltung etwa 4k Speicherplatz von den 16k abziehen, so bleiben uns doch immerhin 12k Speicherplatz für unsere Daten. Dies entspricht 12000 Zeichen (Buchstaben, Zahlen). Rechnet man dies in Textzeilen um, so sind dies 200 Zeilen mit 60 Zeichen, und dies ist eine ganze Menge Text. Daß das ZX81 BASIC diese Daten gleichzeitig mit dem Programm auf Kassette speichert, macht die Sache wesentlich einfacher. Bevor wir uns mit der Bearbeitung von Listen beschäftigen, wollen wir ein ganz einfaches Lagerbestandsprogramm betrachten.

5.1 Lagerbestand

Alle Artikel (100 Stück) sind in einem Zeichenfeld von 100 Einträgen zu 15 Zeichen erfasst. Der Zugriff auf einen Eintrag erfolgt durch den Index des Eintrags. A\$(10) ist der 10.te Eintrag. Dieser Index kann also als Lagerbestandsnummer oder als Bestellnummer angesehen werden. In den 15 Zeichen eines Eintrags sind 10 Zeichen für die Bezeichnung und 5 Zeichen für die Menge reserviert.

Das Programm LVV in Abbildung 5.1 wird das erste Mal mit RUN, später nur noch mit GOTO 100 gestartet. Das Programm fragt nach der Nummer des Eintrags. Dies ist eine Zahl zwischen 1 und 100. Es ist durchaus möglich noch mehr Artikel zu speichern. Theoretisch sind dies 800 Einträge. Dies bedingt eine Änderung der DIM-Anweisung in Zeile 10 vor dem ersten Start.

```

010 DIM A$(100,16)
020 DIM E$(1,10)
030 DIM Z$(1,5)
100 CLS
110 PRINT AT 2,2;"NUMMER:";
115 INPUT N
120 PRINT N
130 PRINT AT 5,1;"BEZEICHNUNG:";
140 PRINT A$(N,1 TO 10)
150 INPUT E$(1)
155 IF CODE E$(1,1) <> 0 THEN PRINT AT
    5,13;E$(1)
160 PRINT AT 6,1;" ANZAHL:";
170 PRINT A$(N,11 TO 16)
175 PRINT AT 7,1;" +/-      :";
180 INPUT Z$(1)
185 PRINT Z$(1)
190 IF CODE Z$(1,1)=0 THEN GOTO 270
200 LET Z= VAL Z$(1)
205 IF CODE A$(N,11)=0 THEN GOTO 220
210 LET Z=Z+ VAL A$(N,11 TO 16)
220 LET Z$(1)= STR$ Z
230 IF CODE E$(1,1)=0 THEN LET E$(1)=A$(N,1
    TO 10)
240 PRINT AT 6,13;"      "
250 PRINT AT 6,13;Z$(1)
260 LET A$(N)=E$(1)+Z$(1)
270 PRINT AT 20,4;"W)EITER B)EENDEN"
280 IF INKEY$="" THEN GOTO 280
290 IF INKEY$="B" THEN STOP
300 IF INKEY$="W" THEN GOTO 100
310 GOTO 270

```

5.1 LVV

Nach der Eingabe erscheint die Bezeichnung. Ist in diesem Feld noch nichts gespeichert, so wird die neue Bezeichnung eingegeben. Danach wird die Anzahl der vorhandenen Artikel eingegeben und mit W wird für einen weiteren Eintrag im Programm zurückgesprungen, oder aber mit B das Programm beendet. Vor dem Abschalten des Rechners, muß

nun der neue Bestand auf Kassette gespeichert werden.

Wird nach der Eingabe einer Nummer eine Bezeichnung ausgegeben, so wird normalerweise mit NL weitergemacht. Soll dieser Eintrag allerdings geändert werden, so kann jetzt eine neue Bezeichnung eingegeben werden, die die alte überschreibt. Bei der Eingabe der Anzahl muß eine positive Zahl ohne Vorzeichen eingegeben werden.

Programmänderungen:

Das Programm kann durch eine Variable D\$ erweitert werden, in der das jeweilige Datum der letzten Änderung gespeichert wird. Dies geschieht am besten im Programmteil "Beenden", das jetzt nur aus der Anweisung STOP besteht. Nach der Eingabe des Datums kann man noch einen Hinweis ausgeben, das Programm auf Kassette zu speichern. Das Datum kann beim Programmstart als erste Zeile auf den Bildschirm ausgegeben werden.

5.2 LISTE

Im letzten Beispiel wurden die Daten in einem Feld gespeichert. Für das Suchen und Auffinden der Daten wurde nur ein Index, eine Zahl, angegeben. Es wurden keine Such- oder Sortierprogramme benötigt.

Das Programm LISTE in Abbildung 5.2 ist ein Demonstrationsprogramm zum Aufstellen, Sortieren und Suchen in Listen. Die Elemente dieses Programms sind Bestandteile der noch folgenden Programme. Als Liste definieren wir ein Textfeld mit 40 Zeilen zu je 20 Zeichen. Das Menüprogramm in Zeile 200 bis 310 enthält die Angaben.

E)INFUEGEN
L)OESCHEN
D)RUCKEN
B)EENDEN

Die Eingabe erfolgt in den Zeilen 500 bis 550. Der neu eingegebene Begriff wird im Programmteil 1200 bis 1399 in die vorhandene, schon alphabetisch sortierte Liste eingetragen. Dazu wird der neue Begriff mit dem Eintrag in der Mitte der Liste verglichen. Je nach dem, ob er größer oder kleiner ist, wird in der oberen oder unteren Hälfte der Liste weitergesucht. Dabei wird diese Hälfte wiederum halbiert und der neue

Eintrag wiederum mit diesem Eintrag verglichen. Dies wird solange fortgesetzt bis nicht mehr halbiert werden kann. Das ist dann der Fall, wenn nur noch ein Eintrag vorhanden ist. Dann wird durch Umspeichern in den Zeilen 1295 bis 1365 Platz für den neuen Eintrag geschaffen, der in Zeile 1370 an den gefundenen Platz geschrieben wird. Wird bei diesem Suchen ein gleicher Eintrag gefunden, wird mit der Fehlermeldung "Eintrag schon vorhanden" dieser Programmteil verlassen.

```
050 DIM A$(40,20)
060 LET N=1
070 DIM D$(1,20)
200 CLS
210 LET M=18
215 LET K=0
220 PRINT TAB M;"E) INFUEGEN"
230 PRINT TAB M;"L) OESCHEN"
240 PRINT TAB M;"D) RUCKEN"
250 PRINT TAB M;"B) EENDEN"
260 IF INKEY$="E" THEN GOSUB 1200
265 IF INKEY$="D" THEN GOSUB 1100
270 IF INKEY$="B" THEN GOTO 900
280 IF INKEY$="L" THEN GOSUB 1500
300 IF K=1 THEN GOTO 200
310 GOTO 260
500 CLS
510 PRINT "EINGABE: ";
520 INPUT D$(1)
522 PRINT D$(1)
525 LET B$=D$(1)
530 IF CODE B$(1)=0 THEN RETURN
540 GOSUB 1205
550 GOTO 500
900 CLS
910 PRINT AT 11,13;"ENDE"
920 STOP
1100 CLS
1110 FOR I=1 TO N-1
1120 PRINT A$(I)
1125 NEXT I
1130 PAUSE 4E4
1135 LET K=1
```

```

1140 RETURN
1200 GOTO 500
1205 LET J=1
1210 IF N=1 THEN GOTO 1330
1220 IF B$<A$(1) THEN GOTO 1330
1230 LET J1=1
1235 LET J2=N
1240 LET J= INT ((J1+J2)/2)
1250 LET C$=A$(J)
1260 IF C$=B$ THEN GOTO 1390
1270 IF B$<C$ THEN LET J2=J
1280 IF B$>C$ THEN LET J1=J
1290 IF J <> INT ((J1+J2)/2) THEN GOTO 1240
1295 LET J=J+1
1330 LET J1=N+1
1340 IF J1<J+1 THEN GOTO 1370
1350 LET A$(J1)=A$(J1-1)
1360 LET J1=J1-1
1365 GOTO 1340
1370 LET A$(J)=B$
1380 LET N=N+1
1382 LET K=1
1385 RETURN
1390 PRINT AT 3,1;"EINTRAG SCHON VORHANDEN"
1392 PAUSE 4E4
1395 LET K=1
1399 RETURN
1500 REM LOESCHEN
1505 CLS
1510 PRINT "WELCHEN EINTRAG";
1515 INPUT D$(1)
1517 LET B$=D$(1)
1520 PRINT B$
1525 LET J=1
1530 IF N <> 1 THEN GOTO 1550
1535 PRINT "LISTE LEER"
1540 PAUSE 4E4
1545 GOTO 500
1550 LET J1=1
1555 LET J2=N
1560 LET J= INT ((J1+J2)/2)
1565 LET C$=A$(J)
1570 IF C$=B$ THEN GOTO 1600

```

```

1575 IF B$<C$ THEN GOTO 1585
1580 LET J1=J
1582 GOTO 1590
1585 LET J2=J
1590 IF J <> INT ((J1+J2)/2) THEN GOTO 1560
1595 PRINT "EINTRAG NICHT GEFUNDEN"
1597 LET K=1
1598 PAUSE 4E4
1599 RETURN
1600 LET J1=J
1610 IF J1=N THEN GOTO 1650
1620 LET A$(J1)=A$(J1+1)
1630 LET J1=J1+1
1640 GOTO 1610
1650 LET N=N-1
1655 LET K=1
1660 RETURN

```

5.2 LISTE

In den Zeilen 1500 bis 1599 wird ein Eintrag gelöscht. Wenn die Liste leer ist, erscheint sofort die Fehlermeldung "Liste leer". Die folgende Programmzeile PAUSW 4E4 ist dem Buch (1*) entnommen und bedeutet Pause forever (Pause four e four). Der Rechner wartet hier bis eine Taste gedrückt wird. Auf die gleiche Weise wie beim Einfügen, wird nun der Eintrag gesucht, der gelöscht werden soll. Wird er nicht gefunden, so erscheint die Fehlermeldung "Eintrag nicht gefunden". Ist er gefunden worden, so wird er in den Zeilen 1600 bis 1660 überschrieben und die Liste um ein Element verkürzt.

Bei der Eingabe längerer Listen ist es nicht sinnvoll jeden Eintrag sofort einzusortieren, denn dies beansprucht Zeit und derjenige, der diese Einträge macht, muß unter Umständen längere Zeit warten, bis er den nächsten Eintrag machen kann. Dann ist es besser, zuerst alles einzugeben und dann vom Rechner sortieren zu lassen. Dies zeigt das Programm SORT in Abbildung 5.3. Als Sortierverfahren wird das SHELL-METZNER Verfahren verwendet, das wesentlich schneller ist als RIPPLE- oder BUBBLE Sort (2*).

*Literatur:

- (1) T. Toms, The ZX81 Pocket Book, Phillips Associates, Epsom
- (2) C. Lorenz, BASIC für Fortgeschrittene, Hofacker Verlag

```

010 DIM A$(100,10)
020 LET N=1
100 CLS
110 PRINT AT 1,3;"EINGABE:"
120 LET I=4
130 INPUT A$(N)
140 IF CODE A$(N,1)=0 THEN GOTO 500
142 PRINT AT I,3;"      "
145 PRINT AT I,3;A$(N)
150 LET N=N+1
160 LET I=I+1
170 IF I<15 THEN LET I=4
190 GOTO 130
500 CLS
502 PRINT AT 10,5;"ICH SORTIERE"
505 GOSUB 1000
507 CLS
510 PRINT AT 1,3;"AUSGABE:"
515 LET I=4
520 FOR J=1 TO N
525 PRINT AT I,3;"      "
530 PRINT AT I,3;A$(J)
540 LET I=I+1
550 IF I>15 THEN LET I=4
560 NEXT J
570 STOP
1000 REM SHELL-METZNER SORT
1010 LET M=N
1020 LET M= INT (M/2)
1030 IF M=0 THEN RETURN
1040 LET J=1
1045 LET K=N-M
1050 LET I=J
1060 LET L=I+M
1070 IF A$(I) <= A$(L) THEN GOTO 1120
1080 LET H$=A$(I)
1083 LET A$(I)=A$(L)
1086 LET A$(L)=H$
1090 LET I=I-M
1100 IF I<1 THEN GOTO 1120
1110 GOTO 1060
1120 LET J=J+1
1130 IF J>K THEN GOTO 1020
1140 GOTO 1050

```

5.3 SORT

Beim SHELL-Sort wird die Liste in 2 Teillisten gleicher Größe zerlegt, und jeweils die beiden ersten Elemente dieser Listen miteinander verglichen. Ist das Element in der 2. Liste kleiner als das Element in der 1. Liste, so werden beide vertauscht und dies vermerkt. Dann werden die 2 Elemente verglichen, usw., bis das jeweilige letzte Element einer Liste erreicht ist. Nach diesem ersten Durchlauf befinden sich alle Elemente die kleiner als das erste Element der 2. Liste sind in der Liste 1.

Jetzt werden beide Listen wiederum halbiert, und das Verfahren solange fortgesetzt, bis eine Liste nur noch aus einem Element besteht. Dann ist die Sortierung beendet. Bei der Erweiterung des SHELL-METZNER Sortierverfahrens wird bei einer Vertauschung nachgesehen, ob einer Vertauschung mit einem Element einer unteren Teilliste möglich ist.

Bei der Verwendung von Sortierverfahren sollte man während des Sortierens anzeigen, daß der Rechner beschäftigt ist. Das Sortieren dauert meistens einige Zeit, und beim Anblick eines blanken Bildschirms neigt man zu der Annahme, daß der Rechner sich aufgehängt hat.

Im Demonstrationsprogramm SORT sind die Zeilen 10, 20 die Initialisierung vor dem ersten Lauf. Das Programm beginnt mit der Eingabe in Zeile 100. Wird die Eingabe mit einer Leereingabe abgeschlossen, so beginnt die Sortierung und die Ausgabe. Wird das Programm mit GOTO 100 gestartet, so werden die neu eingegebenen Begriffe mit den schon vorhandenen sortiert.

5.3 Terminkalender

Das Programm in 5.1 wird in einen Terminkalender (Abbildung 5.4) umgewandelt. Die Eintragungen werden in ein Feld A\$(50,50) gemacht. Es sind also 50 Termine gespeichert, deren Eintrag 8 + 41 Zeichen lang ist. Die ersten 8 Zeichen geben das Datum in der Form TT.MM.JJ (z. B. 22.10.81) an. Dieses Datum ist zugleich das Schlüsselwort für das Einsortieren, Suchen und Löschen. Die restlichen 41 Zeichen sind für den Eintrag frei.

Das Menü dieses Programmes sieht folgendermaßen aus:

TERMIN	E)INGEBEN
	A)USGEBEN
	L)OESCHEN
	T)ERMINE
	B)EENDEN

Beim Eingeben kann nach der Eingabe des Datums, Text bis 41 Zeichen gespeichert werden. Dieser Text wird später bei der Ausgabe, nach Eingabe dieses Datums ausgegeben. Ist dieses Datum nicht eingetragen, erfolgt die Fehlermeldung "kein Eintrag". Dasselbe ist beim Löschen der Fall. Ein gefundenes Datum wird gelöscht. Mit "Termine" wird eine Übersicht über alle eingegebenen Termine ausgegeben. Das Programm wird mit B beendet.

```
010 DIM A$(50,50)
020 DIM D$(1,8)
030 DIM E$(1,41)
040 LET N=1
100 CLS
110 PRINT AT 3,3;"TERMIN"; AT 3,18;"E)INGEBEN"
120 PRINT AT 4,18;"A)USGEBEN"
130 PRINT AT 5,18;"L)OESCHEN)"
135 PRINT AT 6,18;"T)ERMINE"
140 PRINT AT 7,18;"B)EENDEN"
150 IF INKEY$="" THEN GOTO 150
160 IF INKEY$="E" THEN GOTO 500
170 IF INKEY$="A" THEN GOTO 700
180 IF INKEY$="L" THEN GOTO 900
190 IF INKEY$="B" THEN GOTO 998
195 IF INKEY$="T" THEN GOTO 800
199 GOTO 100
200 PRINT AT 10,1;"DATUM:TT.MM.JJ";
210 INPUT D$(1)
220 PRINT AT 10,7;D$(1)
230 RETURN
500 CLS
505 GOSUB 200
510 PRINT AT 12,1;"TEXT:";
520 INPUT E$(1)
```

```

530 PRINT E$(1)
560 GOSUB 1200
570 PRINT AT 20,6;"WEITERE EINTRAEGE (J/N) "
580 IF INKEY$="" THEN GOTO 580
590 IF INKEY$="J" THEN GOTO 500
599 GOTO 100
700 CLS
710 GOSUB 200
720 GOSUB 1500
730 IF B=1 THEN GOTO 100
740 PRINT AT 12,1;"TEXT:";A$(J,9 TO 41)
750 PAUSE 4E4
760 GOTO 100
800 CLS
810 FOR J=1 TO N
820 PRINT A$(J,1 TO 8)
830 NEXT J
840 PAUSE 4E4
850 GOTO 100
900 CLS
910 GOSUB 200
920 GOSUB 1500
930 IF B=1 THEN GOTO 100
940 GOSUB 1600
950 GOTO 100
998 CLS
999 PRINT AT 10,10;"ENDE"
1000 STOP
1200 REM EINSORTIEREN
1205 LET J=1
1210 IF N=1 THEN GOTO 1370
1220 IF D$(1)<A$(1,1 TO 8) THEN GOTO 1330
1230 LET J1=1
1235 LET J2=N
1240 LET J= INT ((J1+J2)/2)
1250 LET C$=A$(J,1 TO 8)
1260 IF C$=D$(1) THEN GOTO 1390
1270 IF D$(1)<C$ THEN LET J2=J
1280 IF D$(1)>C$ THEN LET J1=J
1290 IF J <> INT ((J1+J2)/2) THEN GOTO 1240
1300 LET J=J+1
1330 LET J1=N+1
1340 IF J1<J+1 THEN GOTO 1370

```

```

1350 LET A$(J1)=A$(J1-1)
1360 LET J1=J1-1
1365 GOTO 1340
1370 LET A$(J)=D$(1)+E$(1)
1380 LET N=N+1
1390 RETURN
1500 REM SUCHEN
1510 LET B=0
1525 LET J=1
1530 IF N <> 1 THEN GOTO 1550
1535 PRINT "LISTE LEER"
1540 PAUSE 4E4
1545 GOTO 100
1550 LET J2=N
1555 LET J1=1
1560 LET J= INT ((J1+J2)/2)
1565 LET C$=A$(J,1 TO 8)
1570 IF C$=D$(1) THEN GOTO 1599
1575 IF D$(1)<C$ THEN GOTO 1585
1580 LET J1=J
1582 GOTO 1590
1585 LET J2=J
1590 IF J <> INT ((J1+J2)/2) THEN GOTO 1560
1595 PRINT "KEIN EINTRAG"
1596 LET B=1
1597 PAUSE 4E4
1599 RETURN
1600 REM LOESCHEN
1610 LET J1=J
1615 IF J1=N THEN GOTO 1650
1620 LET A$(J1)=A$(J1+1)
1630 LET J1=J1+1
1640 GOTO 1615
1650 LET N=N-1
1660 RETURN

```

5.4 Terminkalender

Der Aufbau und der Programmablauf entspricht dem in Abbildung 5.2 gezeigten Programm LISTE. Es kann leicht in andere Programme mit 1 Schlüsselbegriff umgewandelt werden. Beispiele sind Inventurprogramme mit beliebiger Bestellnummer, einfache Kataloge, Namenskartei usw.

5.4 Schallplattenverzeichnis

Als Beispiel für das Arbeiten mit mehr als einem Schlüsselwort wird die Aufstellung eines einfachen Schallplattenverzeichnisses dienen. Das Programm zeigt Abbildung 5.5. In einer Zeile sollen folgende Angaben gespeichert werden.

15 Zeichen für das Orchester, Musikgruppe
15 Zeichen für den Interpret
15 Zeichen für den Titel
10 Zeichen für Anmerkungen.

Das Verzeichnis soll 100 Einträge umfassen, wobei entweder nach Orchester, Interpret oder Titel gesucht werden kann.

Über das Menue

E)INGABE
S)ORTIEREN
A)USSUCHEN
B)EENDEN
D)RUCKEN

können die einzelnen Programmteile angewählt werden. Bei der Eingabe werden dann die

MUSIKGRUPPE:
INTERPRET:
TITEL:
BEMERKUNGEN:

eingegeben. Diese Eingaben werden zusammengefasst und in dem Feld A\$(N,56) gespeichert. Beim Aufruf des Sortierprogramms kann dieses Feld entweder nach Gruppe, Interpret oder Titel sortiert werden. Das Sortierverfahren entspricht dem im Programm in Abbildung 5.3 beschriebenen Verfahren. Diese sortierte Liste kann mit D auf den Bildschirm ausgegeben werden. Dabei wird nach jedem 3. Ausdruck angehalten und nach Drücken einer Taste weitergemacht.

Das Suchen eines Titels, Interpret oder einer Gruppe erfolgt mit A. Das verwendete Suchverfahren setzt eine sortierte Liste voraus. Wenn

also ein Interpret gesucht ist und die Liste aber nach Titel sortiert ist, wird automatisch vom Programm neu nach Interpreten sortiert. Erst dann wird nach diesem gesucht.

```
010 DIM A$(100,56)
020 DIM O$(1,15)
030 DIM I$(1,15)
040 DIM T$(1,15)
050 DIM B$(1,10)
060 LET N=1
070 GOTO 500
100 CLS
110 PRINT AT 1,3;"EINGABE:"
120 PRINT AT 10,1;"MUSIKGRUPPE:";
130 INPUT O$(1)
135 LET C$=O$(1)
140 PRINT O$(1)
145 RETURN
150 PRINT AT 11,1;" INTERPRET:";
160 INPUT I$(1)
165 LET C$=I$(1)
170 PRINT I$(1)
175 RETURN
180 PRINT AT 12,1;" TITEL:";
190 INPUT T$(1)
195 LET C$=T$(1)
200 PRINT T$(1)
205 RETURN
210 PRINT AT 13,1;"BEMERKUNGEN:";
220 INPUT B$(1)
230 PRINT B$(1)
240 RETURN
250 GOSUB 100
252 GOSUB 150
254 GOSUB 180
256 GOSUB 210
260 LET A$(N)=O$(1)+I$(1)+T$(1)+B$(1)
270 LET N=N+1
280 PRINT AT 20,5;"WEITER J/N"
290 IF INKEY$="" THEN GOTO 290
300 IF INKEY$="N" THEN GOTO 500
310 GOTO 250
```

```

500 CLS
510 PRINT AT 1,5;"SCHALLPLATTENVERZEICHNISS"
520 PRINT AT 4,18;"E) INGABE"
530 PRINT AT 5,18;"S) ORTIEREN"
540 PRINT AT 6,18;"A) USSUCHEN"
545 PRINT AT 7,18;"B) EENDEN"
547 PRINT AT 8,18;"D) RUCKEN"
550 IF INKEY$="" THEN GOTO 550
560 IF INKEY$="E" THEN GOTO 250
570 IF INKEY$="S" THEN GOTO 600
580 IF INKEY$="A" THEN GOTO 1200
590 IF INKEY$="B" THEN GOTO 997
595 IF INKEY$="D" THEN GOTO 2000
599 GOTO 500
600 CLS
605 LET B=0
610 PRINT AT 1,1;"SORTIEREN NACH"
620 PRINT AT 3,1;"G) RUPPE I) NTERPRET T) ITEL"
630 IF INKEY$="" THEN GOTO 630
640 IF INKEY$="G" THEN GOSUB 700
650 IF INKEY$="I" THEN GOSUB 750
660 IF INKEY$="T" THEN GOSUB 800
670 IF S <> 0 AND S <> 1 AND S <> 2 THEN GOTO 600
675 IF B=1 THEN GOTO 1250
680 GOTO 850
700 LET S=0
710 LET U=1
720 LET V=15
740 RETURN
750 LET S=1
760 LET U=16
770 LET V=30
780 RETURN
800 LET S=2
810 LET U=31
820 LET V=45
830 RETURN
850 CLS
860 PRINT AT 10,6;"ICH SORTIERE"
870 GOSUB 1000
875 IF B=1 THEN GOTO 1300
880 GOTO 500
997 CLS

```

```

998 PRINT AT 10,10;"ENDE"
999 STOP
1000 REM SHELL-METZNER SORT
1010 LET M=N
1020 LET M= INT (M/2)
1030 IF M=0 THEN RETURN
1040 LET J=1
1045 LET K=N-M
1050 LET I=J
1060 LET L=I+M
1070 IF A$(I,U TO V) <= A$(L,U TO V) THEN
GOTO 1120
1080 LET H$=A$(I)
1083 LET A$(I)=A$(L)
1086 LET A$(L)=H$
1090 LET I=I-M
1100 IF I<1 THEN GOTO 1120
1110 GOTO 1060
1120 LET J=J+1
1130 IF J>K THEN GOTO 1020
1140 GOTO 1050
1200 CLS
1210 LET B=1
1220 PRINT AT 1,1;"SUCHEN NACH:"
1230 LET Z=S
1240 GOTO 620
1250 IF Z=S THEN GOTO 1300
1260 GOTO 850
1300 CLS
1310 IF S=0 THEN GOSUB 100
1320 IF S=1 THEN GOSUB 150
1330 IF S=2 THEN GOSUB 180
1340 CLS
1350 PRINT AT 10,6;"ICH SUCHE"
1360 GOSUB 1500
1370 IF K=1 THEN GOTO 500
1380 CLS
1390 PRINT AT 8,5;A$(J,1 TO 15)
1400 PRINT AT 9,5;A$(J,16 TO 30)
1410 PRINT AT 10,5;A$(J,31 TO 45)
1420 PRINT AT 11,5;A$(J,46 TO 56)
1430 PRINT AT 20,5;"WEITER J/N"
1440 IF INKEY$="" THEN GOTO 1440

```

```

1450 IF INKEY$="N" THEN GOTO 500
1460 GOTO 1200
1500 REM SUCHEN
1510 LET K=0
1525 LET J=1
1530 IF N <> 1 THEN GOTO 1550
1535 PRINT "LISTE LEER"
1540 GOTO 1597
1550 LET J1=1
1555 LET J2=N
1560 LET J= INT ((J1+J2)/2)
1565 LET V$=A$(J,U TO V)
1570 IF V$=C$ THEN GOTO 1599
1575 IF V$>C$ THEN GOTO 1585
1580 LET J1=J
1582 GOTO 1590
1585 LET J2=J
1590 IF J <> INT ((J1+J2)/2) THEN GOTO 1560
1595 PRINT "EINTRAG NICHT GEFUNDEN"
1597 PAUSE 4E4
1598 LET K=1
1599 RETURN
2000 CLS
2002 LET J=0
2005 FOR I=1 TO N
2010 PRINT A$(I,1 TO 15)
2020 PRINT A$(I,16 TO 30)
2030 PRINT A$(I,31 TO 45)
2040 PRINT A$(I,46 TO 56)
2045 PRINT
2050 LET J=J+1
2060 IF J=3 THEN GOTO 2080
2070 NEXT I
2080 LET J=0
2090 IF INKEY$="" THEN GOTO 2090
2095 IF I>N THEN GOTO 500
2097 CLS
2100 GOTO 2070

```

5.5 Schallplattenverzeichnis

Programmbeschreibung

Das Vorprogramm besteht aus den Zeilen 10 bis 70. Hier wird mit A\$(100,56) das Schallplattenverzeichnis eröffnet. Die folgenden Zeichenketten O\$, I\$, T\$ und B\$ sind die einzelnen Eingaben. Das Programm wird zum erstenmal mit RUN gestartet, dann nur noch mit GOTO 500.

Die Zeilen 100 bis 240 sind die Eingabeunterprogramme. Sie sind hier als Unterprogramme programmiert, da sie an zwei Stellen, bei der Eingabe und beim Aussuchen verwendet werden. Die eigentliche Eingabe findet in den Zeilen 250 bis 310 statt.

Das Menüprogramm beginnt in Zeile 500. Dies ist auch der Einsprung für den Programmstart. Ab Zeile 600 beginnt das Sortieren. Es kann entweder nach G)RUPPE, I)NTERPRET, T)ITEL sortiert werden. Je nach Eingabe wird in den Unterprogramme 700, 750 und 800 der Sortierparameter S und der Anfang (U) und Ende (V) des Schlüsselwortes festgelegt. Die Variable B zeigt an, ob man über den Sortieraufruf oder über das Aufsuchen in dieses Programmteil gesprungen ist. Das Sortieren beginnt in Zeile 850, mit der schon beschriebenen Sortieroutine ab Zeile 1000.

Das Aussuchen beginnt in Zeile 1200. Wenn Sortierparameter vor und nach der Sucheingabe gleich ist, kann sofort gesucht werden (siehe auch Programm LISTE in Abbildung 5.2). Sonst wird vorher noch sortiert. Wird ein Eintrag im Suchprogramm in den Zeilen 1500 – 1599 gefunden, so wird er in den Zeilen 1390 bis 1420 ausgegeben, andernfalls erfolgt eine Fehlermeldung. Nach Drücken einer Taste kehrt das Programm nach 500 zurück.

Die Ausgabe der Einträge erfolgt in den Zeilen 2000 – 2100.

Einige allgemeine Bemerkungen zu den Programmen in diesem Kapitel. Der Rechner nimmt es bei Zeichenketten ganz genau. Für ihn sind H. MEIER und H MEIER zwei verschiedene Namen. Man muß also genau wissen, was man eingegeben hat.

Alle Programme sind als Vorlagen für eigene Programme gedacht. Die einzelnen Programm-Module können ausgewechselt und zu anderen Programmen zusammengesetzt werden.

Sortier- und Suchverfahren benötigen Zeit. Es ist beim ZX81 deshalb oft sinnvoller kürzere Listen auf verschiedenen Kassetten zu halten.

2 Dinge sollte man nie vergessen.

1. Programm immer nur beim ersten Mal mit RUN, sonst mit GOTO starten.
2. Immer ein SAVE "PROGRAMM" nach jeder neuen Eingabe oder Änderung.

Programmieren in Maschinencode

6. Programmieren in Maschinencode

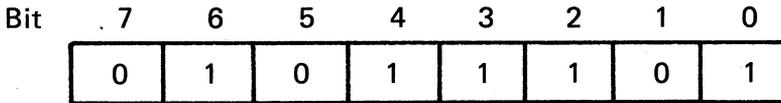
Das Programmieren in Maschinensprache ist wesentlich schwieriger, als das Programmieren in BASIC. Es setzt eine erhebliche Kenntnis des Prozessors und dessen Befehlsvorrat voraus. Dieses Kapitel soll keine Anleitung zum Lernen dieser Programmierung sein, sondern nur zeigen, wie man kleine Maschinenprogramme mit dem ZX81 ausführen kann. Diejenigen, die sich mehr mit der Programmierung in Maschinencode befassen wollen, seien auf das Literaturverzeichnis im Anhang hingewiesen. Hier nur eine ganz kurze Einführung.

6.1 Einführung in die Programmierung mit Maschinencode

Die kleinste Recheneinheit eines Computers ist ein Bit, eine Größe, die zwei Werte annehmen kann, 0 oder 1. 8 Bit werden zu einem Byte zusammengefasst. Mit einem Byte lassen sich die Zahlen 0 bis 255 ($2^N - 1$, mit $N = 8$) darstellen.

Diese Zahl kann nun im Rechner verschiedenes bedeuten. Es kann eine echte Zahl sein. Es kann ein Zeichen sein. Die Zahl 38 stellt im ZX81 den Buchstaben A dar (siehe Handbuch Anhang A), oder es ist ein Befehl für den Prozessor. Auch die BASIC-Befehle werden durch eine solche Zahl dargestellt. Der BASIC-Befehl THEN entspricht der Zahl 222.

Für die Schreibweise eines Bytes gibt es zwei Formen. Einmal die Angabe durch eine Dezimalzahl, zum zweiten als sogenannte Hexadezimalzahl. Diese Bezeichnung hat sich aus dem angelsächsischen Sprachgebrauch eingebürgert. In dieser Schreibweise werden die 8 bit eines Bytes in zwei 4 bit Gruppen aufgeteilt (Abbildung 6.1).



6.1 Ein Byte

Mit diesen 4 Bit lassen sich die Zahlen 0 bis 15 darstellen, wobei die Zahlen 0 bis 9 normal bezeichnet werden, die Zahlen 10 bis 15 durch die Buchstaben A – F (Abbildung 6.2).

0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

6.2 Hexadezimalzahlen

Die Zahl in dem oben angegebenen Byte entspricht einer Dezimalzahl 93 oder einer Hexadezimalzahl \$5D. Das vorangestellte \$-Zeichen soll zur Unterscheidung zwischen einer Dezimalzahl und einer Hexadezimal dienen. Oft wird auch ein H hinter der Zahl zur Kennzeichnung verwendet. 61 ist also eine Dezimalzahl, \$61 oder 61H eine Hexadezimalzahl, mit dem Dezimalwert 97. In seltenen Fällen wird eine Binärzahl als Bitmuster angegeben. Dies soll durch ein vorrangestelltes %-Zeichen

gekennzeichnet werden.

$$97 = \% 01100001 = \$61$$

Nun wollen wir uns ein vereinfachtes Programmiermodell eines Rechners mit einer Z80 CPU anschauen (Abbildung 6.3). Im oberen Bildteil ist der, dem Programmierer, zugängliche Teil der Z80 CPU gezeigt. Dieser besteht aus dem Akkumulator, dem Statusregister F und den Registern B, C, D, E, H und L. Alle diese Register sind 8-bit Register, wobei allerdings die Register BC, DE und HL zu 16-bit Registern zusammengefasst werden.

Die Verwendung dieser Register werden wir gleich an einigen Beispielen zeigen.

Es folgt noch ein spezieller Registersatz, mit den Indexregistern, dem Stapelzeiger, dem Befehlsfolgezähler, Interrupt und MMR Register.

Im Rahmen der Möglichkeiten, mit der wir uns hier mit dem CPU beschäftigen wollen, sind von diesen Registern nur der Stapelzeiger und der Befehlsfolgezähler interessant. Der Befehlsfolgezähler enthält immer die Adresse der Speicherzelle, deren Inhalt als nächster Befehl ausgeführt werden soll. Der Stapelzeiger zeigt auf einen besonderen Speicherbereich, dem Stapel. Hier werden von der CPU Daten, die augenblicklich nicht gebraucht werden, zwischengespeichert. Im Blockschaltbild folgt noch der Speicher. Die Speicherzellen sind 8-bit Speicher. Die Adressen (Hausnummern) gehen von 0 bis 65535 (0 bis \$FFFF). Dies gilt für einen voll ausgebauten Speicher. Beim ZX81 sind aber nur die Adressen 0 bis 15407 (0 bis \$43FF) bei der 1k-Version und 0 bis 32767 (0 bis \$7FFF) bei der 16k-Version vorhanden.

Dabei belegt das Betriebssystem und BASIC den Bereich 0 bis 8191 (0 bis \$1FFF). Aus diesen Speicherzellen kann nur gelesen, aber nichts hineingeschrieben werden (ROM - Read Only Memory). Ab 16384 (\$4000) beginnt der Arbeitsspeicher. In diesem kann geschrieben und wieder ausgelesen werden (RAM - Random Access Memory, Speicher mit wahlfreiem Zugriff).

Was für Befehle kann nun eine CPU ausführen.

Dies sind einmal Transportbefehle. Mit diesen Befehlen können Daten von Speicher zu den Registern, von den Registern zum Speicher und zwischen den Registern transportiert werden.

Beispiele:

LD A, (NN) Lade den Akkumulator mit dem Inhalt der Speicherzelle, mit der Adresse NN. Die Angabe von NN in runden Klammern zeigt an, daß der Inhalt der Zelle NN gemeint ist. Diese Bezeichnungsweise ist aber bei der Angabe des Befehlssatzes nicht konsequent eingehalten worden.

LD C, N Lade das Register C mit der Zahl N.

LD B, A Lade das Register B mit dem Inhalt des Registers A.

LD BC, NN Lade die Register B und C mit der 16 bit Zahl NN.

Desweiteren muß eine CPU Rechenoperationen ausführen können. Das eigentliche Rechenregister ist der Akkumulator. Das Ergebnis einer Rechenoperation wird im Akkumulator gespeichert.

Beispiele:

ADD A, B Addiere zum Inhalt des Akkumulators den Inhalt des B-Registers. Das Ergebnis ist in A gespeichert.

ADD A, A Verdopple den Inhalt des Akkumulators.

Es gibt noch eine Vielzahl von Befehlsarten. Dies sind z. B. Vergleichsbefehle, mit denen die Inhalte von Registern oder Speicherzellen verglichen werden können. Bitbefehle, mit denen einzelne Bit gesetzt oder abgefragt werden können. Sprungbefehle, mit denen ein Programm an einer anderen Stelle weitergeführt werden kann. Blockbefehle, mit denen durch einen einzigen Befehl ganze Datenblöcke im Speicher verschoben werden können, usw. Auf eine ausführliche Befehlsdarstellung sei auf die angegebene Literatur und auf die Befehlsliste im Anhang verwiesen.

Wir wollen hier ein kleines Beispiel betrachten. Wir wollen in Ma-

schinencode "Eins und Eins" zusammenzählen. Dieses Maschinenprogramm soll in ein BASIC-Programm eingefügt und von dort aus aufgerufen werden. Für die Ausgabe des Ergebnisses wollen wir eine Eigenheit des ZX81 ausnutzen. Beim Rücksprung aus dem Maschinenprogramm in das BASIC-Programm wird der Inhalt des BC-Registers als Dezimalzahl auf den Bildschirm ausgegeben, wenn das Programm mit PRINT USR gestartet wurde. Nun das Programm:

3E 01	LD	A, 1	Lade den Akkumulator mit 1
C6 01	ADD	A, 1	Addiere eine Eins dazu
06 00	LD	B, D	Null ins B-Register
4F	LD	C, A	Ergebnis ins C-Register
C9	RET		RETURN, Rücksprung nach BASIC

Die links angegebenen Bitmuster können aus der im Anhang angegebenen Befehlsliste entnommen werden. Wie binden wir nun dieses Maschinenprogramm in ein BASIC-Programm ein. Dafür gibt es 2 Möglichkeiten. Wir schaffen vor dem Beginn des BASIC-Programms Platz für den Maschinencode, oder wir setzen es ans Ende des Speichers.

Die erste Möglichkeit wird mit folgenden Programm in Abbildung 6.4 erreicht.

```
01 REM BBBBBBBB
10 LET X = 16514
20 POKE X,62
30 POKE X+1,01
40 POKE X+2,198
50 POKE X+3,01
60 POKE X+4,06
70 POKE X+5,00
80 POKE X+7,201
90 POKE X+7,201
100 PRINT USR X
```

6.4 Maschinencode in BASIC

In der ersten Zeile dieses Programms machen wir mit der REM-Anweisung Platz für 8 Byte. So lang ist unser Maschinenprogramm. Dann

setzen wir die Anfangsadresse A ! 16514. Diese Adresse wird folgendermaßen bestimmt: Alle BASIC-Programme beginnen bei Adresse 16509. Die ersten beiden Byte bilden die Zeilennummer. In den Zellen 16511 und 16512 ist die Länge der Zeile gespeichert. Die Verschlüsselung für den BASIC-Befehl REM steht in Zelle 16513. Das erste B belegt also die Speicherzelle 16514. In diese und die folgenden 7 Zellen wird das Maschinenprogramm durch POKE-Befehle gespeichert. Dazu müssen die Hexzahlen von Hand in Dezimalzahlen umgewandelt werden. Es ist

\$3E = 62
 \$C6 = 198
 \$4F = 79
 \$C9 = 201

Nachdem wir das Programm gestartet haben, erscheint auf den Bildschirm eine 2. Durch Ändern der Zeilen 30 und 60 können andere Zahlen addiert werden.

Eine andere Möglichkeit, ein Maschinenprogramm an die Stelle einer REM-Anweisung ist folgende. Dabei brauchen die Hexdezimalzahlen nicht in Dezimalzahlen von Hand umgewandelt werden, dies übernimmt der Rechner.

```

10 REM BBBB BBBB
20 LET A = 16514
30 LET M$="3E01C60106004FC9"
40 FOR I=1 TO LEN M$-1 STEP 2
50 POKE A + INT ((I-1)/2),(CODE M$(I)-28)*16+CODE M$
  (I+1)-28
60 NEXT I
70 PRINT USR A
  
```

6.5 Hexeingabe

Mit POKE 16515,N können wir den Wert dieser Speicherzelle ändern. Dann brauchen wir nur mit GOTO 70 unser Maschinenprogramm aufrufen, und erhalten ein neues Ergebnis. Im Prinzip können nun die

Zeilen 20 bis 60 gelöscht werden, ohne daß das Maschinenprogramm zerstört wird. Wenn wir allerdings LIST eingeben, dann sind in Zeile 10 alle B's durch andere Zeichen ersetzt worden.

Sie sieht jetzt so aus:

```
10 Y■ LEN ■■■ ? TAN
```

Der Basic Interpreter übersetzt die durch POKE gespeicherten Byte in seine Darstellung.

```
10 Y■ LEN ■■■ ? TAN           A = 16514.
```

Die zweite Möglichkeit, ein Maschinenprogramm abzuspeichern, besteht darin, es in die letzten Speicherzellen des Arbeitsspeichers zu schreiben. Die Zellen 16388 (\$4004) und 16389 (\$4005) enthalten die höchste verfügbare Speicheradresse. Diese Adresse ist mit RAMTOP bezeichnet (siehe Handbuch S. 178 und S. 171). Erniedrigt man diese Adresse um einige Byte, so kann man Platz für ein Maschinenprogramm schaffen.

Dazu benötigt man zuerst ein kleines Verschiebeprogramm (Abbildung 6.6). Das Programm fragt an, um wieviel Byte RAMTOP geändert werden soll. Nach dieser Eingabe wird der augenblickliche Wert geholt, die Zahl der Byte abgezogen und wieder zurückgeschrieben. Nach der Anweisung NEW ist das Verschiebeprogramm gelöscht, der freigemachte Speicher aber geschützt.

```
10 LET RT=16388
20 LET A=PEEK RT+256*PEEK(RT+1)
30 PRINT "WIEVIELE BYTE ?";
40 INPUT N
50 PRINT N
60 LET A=A-N
70 POKE A-INT (A/256)*256, RT
80 POKE INT(A/256),RT+1
90 NEW
```

6.6 RAMTOP-Verschiebung

In unseren Programm in Abbildung 6.5 müssen wir zur Eingabe die Zeile 20 in

```
20 LET RT = 16388
25 LET A = PEEK RT + PEEK (RT + 1) * 256
```

ändern.

Noch ein Hinweis. Beim Z80 sind die Adressen immer zuerst mit dem niederen Adressbyte, dann mit dem höheren Adressbyte gespeichert. Ist RAMTOP zum Beispiel \$437F, so ist \$7F in Speicherzelle 16388 und \$43 in Zelle 16389 gespeichert.

6.2 Der Maschinencode-Editor MONI (16k)

Wenn man mit Maschinencode arbeitet, so ist die unterste Stufe der Programmierhilfe ein Programm, mit dem man den Inhalt von Speicherzellen ausgeben und ändern kann, und mit dem man Maschinenprogramme starten kann. Dazu dient der Maschinencode-Editor MONI (Abbildung 6.7).

```
100 LET BYTE=900
110 LET ADRESSE=2000
120 LET MENUE=1000
200 GOTO MENUE
300 PRINT " ";
305 LET B=N
310 IF D=1 THEN GOSUB 950
320 IF D=0 THEN PRINT N
330 RETURN
900 REM HEX BYTE
910 LET B= PEEK L
920 PRINT CHR$ ( INT (B/16)+28);
930 PRINT CHR$ (B-( INT (B/16)*16)+28);
940 RETURN
950 REM HEX-ADRESSE
952 LET H=4096
955 PRINT CHR$ ( INT (B/H)+28);
960 LET B=B- INT (B/H)*H
965 LET H=H/16
```

```

970 IF H >= 1 THEN GOTO 955
975 PRINT "H"; TAB 7;
980 RETURN
990 CLS
992 LET I=0
999 RETURN
1000 REM MENUE
1010 PRINT TAB 7;"MINI-MONITOR"
1020 PRINT
1030 PRINT TAB 7;"A) ENDERN"
1040 PRINT TAB 7;"I) NHALT"
1050 PRINT TAB 7;"S) TARTEN"
1070 PRINT TAB 7;"B) EENDEN"
1080 IF INKEY$="" THEN GOTO 1080
1090 IF INKEY$="A" THEN GOSUB 2500
1100 IF INKEY$="I" THEN GOSUB 4600
1110 IF INKEY$="B" THEN GOTO 3900
1115 IF INKEY$="S" THEN GOSUB 3000
1120 CLS
1130 GOTO MENUE
2000 REM ADRESSE
2005 LET D=1
2010 CLS
2020 PRINT "ADRESSE: ";
2030 INPUT A$
2040 PRINT A$;
2045 IF A$( LEN A$)="H" THEN LET D=0
2050 IF A$( LEN A$)="H" THEN GOTO 2100
2060 LET N= VAL A$
2070 RETURN
2100 LET N=0
2110 FOR X=1 TO LEN A$-1
2120 LET N=N*16+ CODE A$(X)-28
2130 NEXT X
2140 RETURN
2500 REM AENDRN
2505 LET I=0
2510 CLS
2520 GOSUB ADRESSE
2521 GOSUB 300
2522 LET L=N
2525 PRINT AT 21,1;"X) BEENDEN NL) VOR R) ZURUECK"
2527 PRINT AT 2,0;

```

```

2530 IF D=1 THEN PRINT L; TAB 7;
2532 LET B=L
2535 IF D=0 THEN GOSUB 950
2540 GOSUB BYTE
2550 PRINT " --> ";
2560 INPUT A$
2570 IF A$ <> "" THEN GOTO 2600
2580 LET L=L+1
2585 PRINT
2587 LET I=I+1
2590 IF I>18 THEN GOSUB 990
2595 GOTO 2530
2600 IF LEN A$>3 THEN GOTO 2560
2610 IF A$="R" THEN GOTO 2700
2620 IF A$="X" THEN RETURN
2630 GOSUB 2050
2635 LET B=N
2640 GOSUB 920
2650 POKE L,N
2660 GOTO 2580
2700 LET L=L-1
2710 GOTO 2585
3000 REM STARTEN
3010 GOSUB ADRESSE
3015 GOSUB 300
3020 PRINT USR N
3030 PAUSE 4E4
3040 RETURN
3900 CLS
3910 STOP
4600 REM INHALT
4610 GOSUB ADRESSE
4612 GOSUB 300
4615 PRINT AT 20,0;"W) EITER M) ENUE"
4618 PRINT AT 2,0
4620 FOR S=0 TO 15
4630 IF D=1 THEN PRINT N+S*8; TAB 7;
4632 LET B=N+S*8
4635 IF D=0 THEN GOSUB 950
4640 FOR X=0 TO 7
4650 LET L=N+S*8+X
4660 GOSUB BYTE
4670 PRINT " ";

```

```

4680 NEXT X
4690 PRINT
4692 IF INKEY$="" THEN GOTO 4692
4695 IF INKEY$="M" THEN GOTO 4760
4700 NEXT S
4710 PRINT
4740 LET N=L+1
4745 CLS
4750 GOTO 4615
4760 RETURN
5000 LET B=20480
5010 GOSUB 950

```

6.7 Maschinencode-Editor MONI

Dieser Monitor ist in BASIC geschrieben und belegt die Speicherzellen bis 497E H. Nach dem Starten des Programms mit GOTO 100 erscheint des Menü mit

```

A)ENDERN
I)NHALT
S)TARTEN
B)EENDEN

```

Ändern von Speicherinhalten

Durch Eingabe von A wird das Programm zum Ändern von Speicherinhalten aufgerufen. Als nächstes muß die Adresse eingegeben werden. Diese Eingabe kann sowohl in dezimaler, wie in hexadezimaler Schreibweise erfolgen. Bei hexadezimaler Eingabe muß nach der Adresse ein H folgen. Also 5000 H oder 20480. Auf dem Bildschirm wird die Adresse in dezimaler Form und der Inhalt als Hexadezimalzahl angezeigt. Also zum Beispiel

20480 21 →

An dieser Stelle kann nun eine neue Zahl in diese Zelle geschrieben werden, und es sind auch hier beide Schreibweisen erlaubt. Geben wir die Dezimalzahl 155 ein, so wird diese Zahl in die Zelle 20480 ge-

geschrieben und auf dem Bildschirm in hexadezimaler Form ausgegeben.

Dort erscheint dann folgende Ausgabe

```
20480 21 - -> 9B
20481 00 - ->
```

Die Zahl 9B hätte auch direkt mit 9BH eingegeben werden können. Am unteren Ende des Bildschirms steht die Zeile

```
X)BEENDEN      NL)VOR      R)ZURÜCK
```

Wird jetzt X (NL) eingegeben, so ist der Programmteil Ändern beendet und es erfolgt ein Rücksprung in das Menü. Wird nur NL eingegeben, so wird der Inhalt der angezeigten Zelle nicht verändert und die Adresse und der Inhalt der nächsten Zelle angezeigt.

```
20480 21 - -> 9B
20481 00 - ->
20482 00 - ->
```

Wird R (NL) eingegeben, so wird die vorangegangene Zelle angezeigt. Auf diese Weise können Programme in Maschinencode in die Zellen des ZX81 eingegeben werden. Vom Menü aus kann das Programm mit S gestartet werden. Nach der Eingabe von S erfolgt die Eingabe der Startadresse wie im Programmteil "Ändern".

In BASIC wird das Programm mit

```
PRINT USR N
```

gestartet, wobei N die eingegebene Adresse ist. Ist das Maschinenprogramm mit RET (C9) abgeschlossen, so wird nach der Ausführung des Programms nach BASIC zurückgesprungen, wobei durch den PRINT-Befehl der Inhalt des BC-Registers ausgedruckt wird. Der Rechner befindet sich nun in einer unendlichen Pausenschleife, die durch Betätigen einer Taste abgebrochen und somit in das Menüprogramm zurückgesprungen wird.

Mit I wird der Inhalt von Zellen angezeigt. Nach der Eingabe der Adresse wird der Inhalt dieser und der folgenden 7 Zellen ausgegeben.

Also zum Beispiel:

ADRESSE 1000H

4096 9A 0D D6 C4 38 F9 01 EC

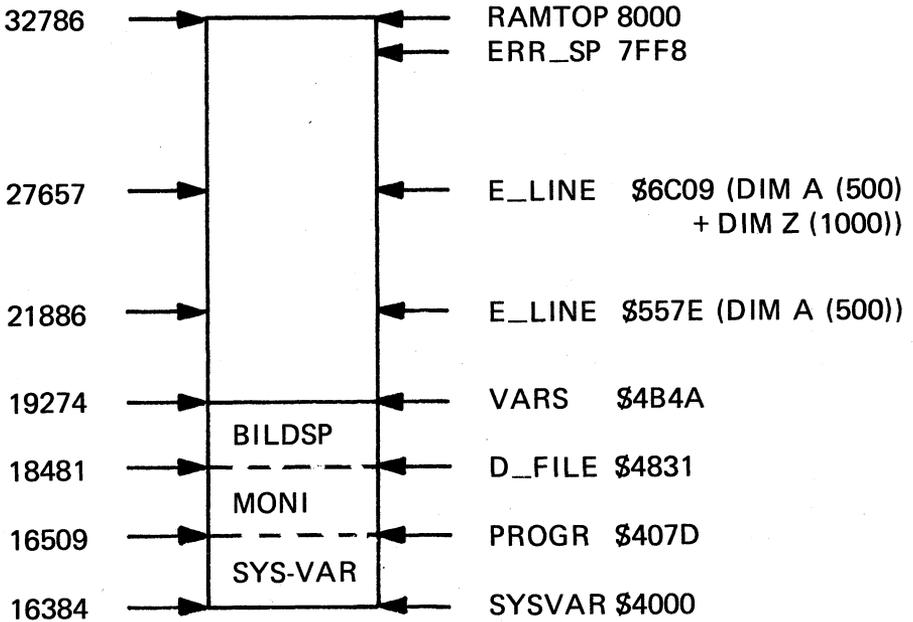
Nun kann mit W der Inhalt der folgenden 8 Zellen ausgegeben oder mit M in das Menü zurückgesprungen werden. Nach der Ausgabe von 16 Zeilen wird der Bildschirm gelöscht.

Speichern von Programmen in Maschinencode auf Kassette. Mit SAVE "(NAME)" werden immer die Systemvariablen, das BASIC-Programm und die vom Programm erzeugten Variablen gespeichert. Das Ende der Aufzeichnung ist der Inhalt der Adresse E_LINE (16404 = LSB, 16405 = MSB).

Im Programm MONI wird durch eine Dimensionsangabe für eine nicht verwendete Variable DIM A (500) diese Adresse auf \$557E = 21886, gesetzt, so daß alle Änderungen zwischen \$4B4A = 19274 und \$557E = 21886 auf Kassette geschrieben werden.

Soll dieser Bereich erweitert werden, so muß vor dem Start eine weitere Dimensionsangabe z. B. DIM Z (1000) gemacht werden. Das Programm muß dann mit GOTO 100 gestartet werden. Durch Anzeigen des Inhalts von E_LINE kann man feststellen, wie weit bei einem SAVE der Inhalt des Speichers aufgezeichnet wird. Eine nachträgliche Änderung des Speicherbereichs ist nicht möglich. Bei einem DIM-Befehl wird der reservierte Speicherplatz mit 00 vollgeschrieben und damit vorherige Einträge gelöscht. Ein direktes Ändern der Zelle E_LINE bringt das System zum Absturz.

In der Abbildung 6.8 ist die Speicherverteilung des Programms MONI mit DIM A (500) und deinem zusätzlichen, vor dem RUN eingegebenen DIM Z (1000) dargestellt.



6.8 Speicherverteilung MONI

Im folgenden soll noch ein Beispiel für das Arbeiten mit dem Monitor gegeben werden. Es wird ein Programm in Z80-Code eingegeben, das den freien Speicherplatz zwischen STKEND und dem Stapelzeiger der CPU angibt (siehe Handbuch S. 171).

Z80 Maschinenprogramm

```
LD HL, 0          21 00 00    ; (HL) = 0
ADD HL, SP        39          ; (SP) → HL
LD DE, (STEND)   ED 5B 1C 40 ; (401C) → DE
SBC HL, DE       ED 52          ; (HL) - (DE) → HL
LD B, H          44          ; (H) → B
LD C, L          4D          ; (L) → C
RET              C9          ; zurück nach BASIC
```

Diese Folge von Hexadezimalzahlen geben wir mit dem Programmteil "Ändern" ab Adresse 5000H ein. Nach Beenden der Eingabe wird mit I 5000H der Inhalt überprüft.

Mit S wird das Programm bei 5000H gestartet. Als Ausgabe erscheint die Zahl 10888. Nach Betätigen einer Taste wird ins Menü zurückgesprungen.

Wird nun mit SAVE "(NAME)" das Programm auf Kassette aufgezeichnet, so werden alle Speicherzellen bis \$557E auf Band geschrieben, somit auch das kleine Maschinenprogramm bei Adresse 5000H – 500CH. Mit dem LOAD "(NAME)" Befehl wird es wieder eingelesen.

Wird bei der Eingabe ein Fehler gemacht, so daß der Rechner aus dem Programm zu einer Fehlermeldung herausspringt, so sind die Daten nicht verloren, wenn man mit GOTO 100 neu beginnt.

Bei der Eingabe längerer Programme ist es lästig, immer wieder bei Hexeingabe H zu drücken. In Programm MONI2 werden alle Byte, die Änderung eingegeben werden als Hexadezimalzahlen interpretiert. Nur bei Adresseingaben wird bei Hexadezimalzahlen ein H verlangt. Die Programmänderungen zeigt Abbildung 6.9.

```
2600 IF LEN AS>2 THEN GOTO 2560
2630 LET N=0
2632 FOR X=1 TO LEN A$
2633 LET N=N*16+CODE A$(X)-28
2634 NEXT X
```

6.9 Änderungen zu MONI 2

Zum Schluß noch zwei Maschinenprogramme, welche die Verwendung von Blockbefehlen zeigen.

Das Programm in Abbildung 6.10 durchsucht den Speicherbereich von 0 bis \$2000 nach einem Byte dessen Wert (\$E2) im Akkumulator gespeichert ist.

5000	21 00 00	LD HL, 0000	Anfangsadresse → HL
5003	3E E2	LD A, N	N (Suchbyte) → A
5005	01 00 20	LD BC, 2000	Endadresse → BC
5008	ED B1	CPIR	Vergleiche A mit (HL), HL = HL + 1, BC = BC - 1
500A	44	LD B, H	(H) → B Für Ausgabe in Basic
500B	4D	LD C, L	(L) → C
500C	C9	RET	Rückkehr nach Basic

6.10 Suchen nach einem Byte

Das Programm in Abbildung 6.11 verschiebt 100 Byte von \$5000 nach \$6000.

Beide Programme können mit dem Programm MONI eingegeben und ausgeführt werden.

5200	21 00 50	LD HL, 5000	Anfangsadresse → HL
5203	11 00 60	LD DE, 6000	Zieladresse → DE
5206	01 00 01	LD BC, 100	Zahl der Byte → BC
5209	ED B0	LDIR	Blocktransfer
520B	C9	RET	Rückkehr nach Basic

6.11 Blocktransfer von 100 Byte

NOTIZEN

Anschluß einer PIO an den ZX81

7. Anschluß einer PIO an den ZX81

Bis jetzt haben wir den ZX81 nur als Computer für Datenverwaltung oder zum Lösen von Rechenaufgaben benutzt. Nun soll er aber auch für Steuerungsaufgaben verwendet werden. Der ZX81 soll also Daten von der Außenwelt aufnehmen, sie verarbeiten und Signale wieder abgeben können. An der Rückseite des Rechners ist eine Anschlußleiste mit Signalen von der CPU. Diese können wir nicht direkt verwenden, sondern wir benötigen noch einen Interface-Baustein.

Dazu verwenden wir einen zur Z80A CPU passenden Baustein, nämlich den Z80A-PIO (Parallel Input Output).

Dieser Baustein besitzt zwei Tore, TORA und TORB mit jeweils 8 Leitungen, die entweder als Ausgänge oder als Eingänge verwendet werden können. Auf weitere Einzelheiten wird im Laufe dieses Kapitels noch eingegangen werden.

Zuvor noch eine allgemeine Bemerkung. Bis jetzt haben wir uns nicht darum gekümmert, wie ein Rechner eine 1 oder 0 intern darstellt. Wenn wir aber z. B. die Binärzahl 01011101 über das Tor eingeben wollen, so müssen wir wissen, welchen Spannungswert einer 1 und welcher Spannungswert einer 0 entspricht. Denn es sind ja Spannungen, die an die Leitungen angelegt werden. Bei der Kennzeichnung dieser Spannungswerte hat sich die Bezeichnung TTL-kompatibel durchgesetzt. TTL ist die Transistor-Transistor Logik, mit der einer umfangreichsten Familie von integrierten Schaltungen aufgebaut wurde. Durch sie wurden folgende Pegel festgelegt.

1. Für die Ausgabe von Daten:

Eine 1 entspricht mindestens 2,4 Volt,
eine 0 entspricht maximal 0,4 Volt.

2. Für die Eingabe von Daten:

Eine 1 entspricht mindestens 2,0 Volt,
eine 0 entspricht maximal 0,8 Volt.

In beiden Fällen darf eine Spannung von 5,0 V nicht überschritten werden.

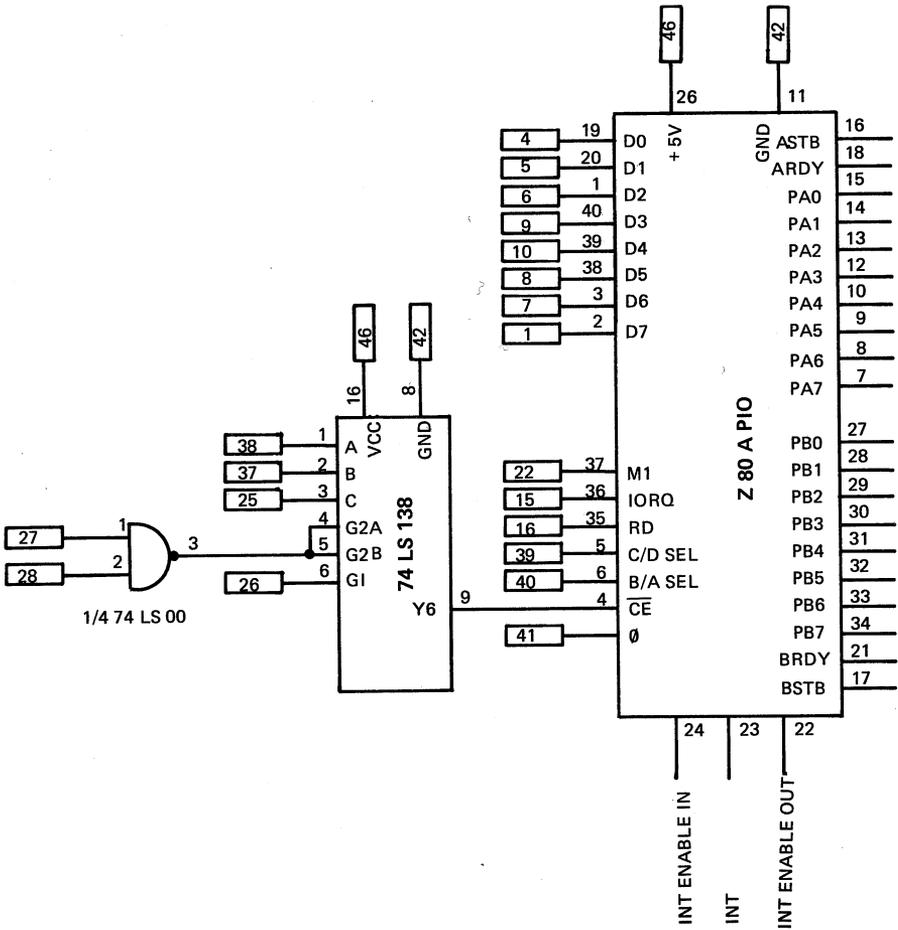
Wird in einem Datenblatt angegeben, daß die Ein- oder Ausgänge TTL-kompatibel sind, so werden obige Spannungspegel eingehalten. Das gilt dann auch für die Belastbarkeit. Ein Ausgang kann durch mindestens einen TTL-Eingang belastet werden. Dies sind aber Ströme von ca. 1 mA, so daß wir für das Schalten größerer Lasten Verstärker zwischen Tor und Last einschalten müssen.

7.1 Anschluß der PIO an den ZX81

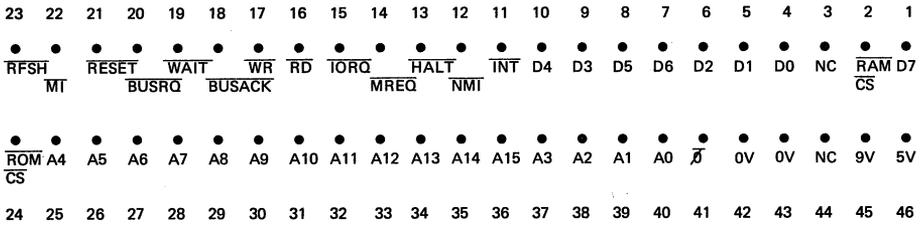
Zum Anschluß der PIO an den ZX81 benötigen wir die Adapterplatine EL81 mit den beiden Steckern und die Experimentierplatine EX81. Auf dieser wird die Schaltung nach Abbildung 7.1 aufgebaut. Abbildung 7.2 zeigt die Steckerbelegung mit Sicht auf die Anschlüsse.

Im Blockschaltbild sind auf der linken Seite der PIO die Anschlüsse, die zum ZX81 führen. Direkt verbunden sind die Datenleitungen D0 bis D7. (Die Ziffern in den Kästchen beziehen sich auf die Anschlußnummerierung in Abbildung 7.2.) Ferner sind direkt verbunden die Leitungen \overline{MI} , \overline{RD} , $\overline{\emptyset}$ und \overline{IORQ} . Letztere Leitung ist der I/O Request. Diese Leitung geht auf Null, wenn vom Prozessor ein IN oder OUT-Befehl ausgeführt wird.

Die Leitung C/\overline{D} SEL bestimmt, ob vom Prozessor ein Steuerwort (Control) oder Daten an die PIO übergeben werden. Ist diese Leitung 0, so sind es Daten, ist sie 1, so ist es ein Steuerwort. Diese Leitung ist mit der Adressleitung A1 verbunden.



7.1 Zusammenschaltung Z80 A PIO mit ZX81



7.2 Steckanschlüsse auf der Rückseite der Adapterplatine EL 81

Der Eingang $\overline{B/A}$ SEL bestimmt, ob Tor A oder TOR B angesprochen wird. Eine 0 an diesem Eingang wählt Tor A, eine 1 wählt Tor B aus. Dieser Eingang ist mit der Adressleitung A0 verbunden. Warum dies so gemacht wird, wird im nächsten Abschnitt, bei der Programmierung der PIO, deutlich werden.

Der Eingang \overline{CE} ist der Chip Enable Eingang. Eine 0 an diesem Eingang lässt den Baustein "aktiv" werden. Liegt aber eine 1 an, so ist es so, als ob dieser Baustein nicht vorhanden wäre. Dieses Signal wird aus den Adressen A2 bis A7 über einen 74 LS 00 (NAND-Gatter) und den 3 Bit Binärdekoder 74 LS 138. Das \overline{CE} Signal wird Null, wenn die Bitkombination

A7	A6	A5	A4	A3	A2	A1	A0
1	1	1	1	1	0	X	X

anliegt (siehe Anhang Wahrheitstabelle 74 LS 138).

Nicht angeschlossen sind die Interrupt-Leitungen, da wir im Rahmen dieses Buches nicht auf die Behandlung von Interrupts eingehen wollen. Die Bausteine auf der Platine werden an die 5-Volt Versorgung des Rechners angeschlossen. Sollten jedoch Erweiterungen auf der Platine gemacht werden, so ist es sinnvoller, diese aus einem getrennten Netzgerät zu versorgen.

Auf der rechten Seite der PIO sind im Blockschaltbild die Leitungen angegeben, die nach außen zu externen Geräten geführt werden können. Dies sind die Leitungen PA0 bis PA7 vor Tor A mit den Handshake-Leitungen ARDY und ASTB, sowie die Leitungen PB0 bis PB7, BSTB und BRDY.

Die Leitung ARDY ist ein Ausgang mit 1 als aktiven Pegel. Ist das Tor A als Ausgang geschaltet, so wird ARDY = 1, wenn Daten an das Tor ausgegeben wurden. Damit wird dem externen Gerät angezeigt, daß diese von Tor abgeholt werden können. Ist das Tor A als Eingang geschaltet, so zeigt ARDY = 1 an, daß das Tor leer ist, der Prozessor hat die Daten gelesen. Das externe Gerät kann neue Daten an das Tor weitergeben. Ist das Tor A für bi-direktionalen Betrieb geschaltet, so zeigt ARDY = 1 daß Daten an das externe Gerät abgegeben werden können. Diese werden jedoch erst dann auf die Ausgänge durchgeschaltet, wenn ein ASTB Signal empfangen wurde. In der 3. Betriebsart ist ARDY intern abgeschaltet.

Die Leitung ASTB ist ein Eingang, mit 0 als aktivem Pegel. Wenn das Tor A als Ausgang geschaltet ist, gibt eine positive Flanke von ASTB an, daß das externe Gerät die Daten empfangen hat. Ist das Tor A ein Eingang, so speichert ein negativer Impuls an ASTB Daten in das Eingangsregister. Bei bidirektionalen Betrieb wird durch ASTB, Daten auf die Ausgangsleitungen geschaltet. Die Leitungen BSTB und BRDY verhalten sich gleich, bis auf die Betriebsart: Tor A als bidirektionales Tor. Hier übergibt ein negativer Impuls Daten an das Eingangsregister von Tor A während BRDY = 1 anzeigt, daß Tor A leer ist, und Daten aufnehmen kann.

7.2 Programmierung der PIO

Die Programmierung der Tore erfolgt über ein Steuerwort. Das Format dieses Steuerwortes zeigt Abbildung 7.3. Mit dem Bit D6 und D7 wird die Betriebsart festgelegt. D5 und D4 sind unbedeutend und mit Bit D3 – D0 wird der PIO angezeigt, daß ein Steuerwort vorliegt.

D7	D6	D5	D4	D3	D2	D1	D0
M1	M0	X	X	1	1	1	1

7.3 Steuerwort zur Programmierung der PIO

Die Betriebsart ist folgendermaßen festgelegt:

M1	M0	Betriebsart	
0	0	0	Tor als Ausgang
0	1	1	Tor als Eingang
1	0	2	Bidirektional
1	1	3	Bit-control

Die Steuerworte lauten somit:

\$0F	Ausgang
\$4F	Eingang
\$8F	Bidirektional
\$CF	Bit-control

Dieses Steuerwort wird über die Datenleitungen an die PIO übergeben. Dabei muß Eingang C/D SEL = 1 sein. Diesen Eingang hatten wir mit A1 verbunden. Für welches Tor dieses Steuerwort gilt, wird über B/A SEL angewählt. Mit B/A SEL = 0 ist es Tor A. Dieser Eingang ist mit A0 verbunden. Mit der dekodierten Adresse für CE = \$F8 erhalten wir für die PIO folgende Adressen:

% 1 1 1 1 1 0 0 0	= \$F8	Tor A, Daten
% 1 1 1 1 1 0 0 1	= \$F9	Tor B, Daten
% 1 1 1 1 1 0 1 0	= \$FA	Tor A, Steuerwort
% 1 1 1 1 1 0 1 1	= \$FB	Tor B, Steuerwort

Mit dem Programm in Z80 Maschinencode wird das Tor A zu einem Ausgang (Abbildung 7.4).

LD A, 0F	3E 0F	0F → A, Ausgang
OUT FA, A	D3 FA	(A) → Tor A, Steuerwort
LD A, AA	3E AA	AA → A, Bitmuster
OUT F8, A	D3 F8	(A) → Tor A, Daten
RET C9		zurück nach Basic

7.4 Maschinenprogramm um Tor A als Ausgang zu schalten

Dieses Programm wollen wir nun in BASIC einbinden. Wir verwenden

dazu die in Kapitel 6 beschriebene Methode.

Es belegt 9 Byte für die wir durch REM AAAAAAAAAA in der ersten Zeile Platz schaffen. Das Programm beginnt wie üblich in Speicherzelle 16509. Die folgenden 2 Byte sind für die Zeilennummer, die nächsten 2 Byte sind für die Zeilenlänge und dann folgt noch 1 Byte für die Verschlüsselung der Anweisung REM. Das erste A steht also in Speicherzelle 16514. Dort lassen wir unser Maschinenprogramm beginnen.

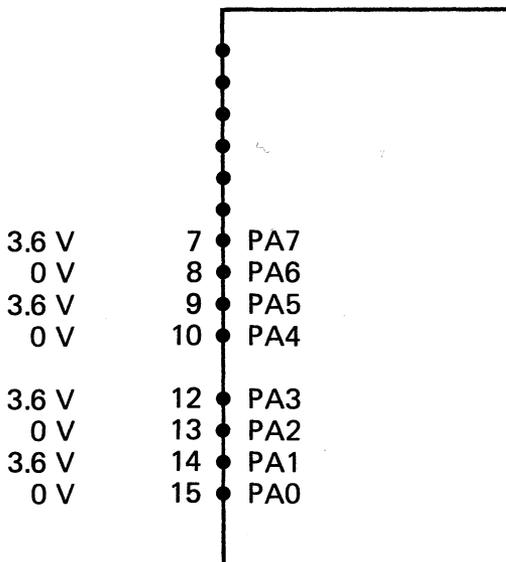
Da wir das Maschinenprogramm durch Poke-Befehle in den Speicher schreiben wollen, so müssen wir erst die Hexadezimalzahlen in Dezimalzahlen umwandeln. Es ist

\$3E	=	62
\$0F	=	15
\$D3	=	211
\$FA	=	250
\$AA	=	170
\$F8	=	248
\$C9	=	201

```
001  REM AAAAAAAAAA
010  LET X=16514
020  POKE X,62
030  POKE X+1,15
040  POKE X+2,211
050  POKE X+3,250
060  POKE X+4,62
080  POKE X+6,211
090  POKE X+7,248
110  PRINT "N=";
120  INPUT N
125  IF N<0 THEN STOP
130  PRINT N
140  POKE X+5,N
150  LET A=USR X
160  GOTO 110
```

7.5 BASIC-Programm zur Ausgabe von Zahlen an Tor A

Das Programm in Abbildung 7.5 speichert die Befehle ab 16514, mit Ausnahme von Byte 6, das erst nach Eingabe von N gespeichert wird. Das Maschinenprogramm wird in Zeile 150 aufgerufen. Mit einem Voltmeter im 5V Bereich kann man nun die Spannungen an den einzelnen Ausgängen nachmessen. Wenn wir für N = 170 eingegeben haben müssen die in Abbildung 7.6 angegebenen Spannungen zu messen sein. Die Beendigung des Programms geschieht durch Eingabe einer negativen Zahl. Man kann natürlich die in dem nächsten Kapitel angegebene Leuchtdiodenanzeige verwenden. Dann ist das Ergebnis gleich sichtbar.



7.6 Spannungen an Tor A, nach Ausgabe von N = 170

Wir wollen noch ein zweites Programm für die Ausgabe von Zahlen verwenden, bei dem die Zahlen als Hexadezimalzahlen eingegeben werden können.

In dem Programm in Abbildung 7.7 wird wieder in Zeile 1 durch eine REM-Anweisung Platz für das Maschinenprogramm reserviert. Dieses Maschinenprogramm ist in einer Zeichenkette M\$ als Folge von Hexadezimalzahlen gespeichert. 2 Zeichen dieser Kette werden in eine Dezimalzahl gewandelt und an der entsprechenden Stelle durch einen

POKE-Befehl gespeichert. Ein neuer Zahlenwert wird durch eine 2-stellige Zeichenkette eingegeben, vom Programm umgewandelt und an der richtigen Stelle im Maschinenprogramm gespeichert. Durch den Aufruf des Maschinenprogramms in Zeile 110 wird der Wert an das Tor A ausgegeben.

```

001  REM AAAAAAAAAA
010  LET X=16514
020  LET M$="3E0FD3FA3E00D3F8C9"
030  FOR I=1 TO LEN M$-1 STEP 2
040  POKE X+ INT ((I-1)/2) ,( CODE M$(I)-28)*16+
      CODE M$(I+1)-28
050  NEXT I
060  PRINT "N=";
070  INPUT N$
075  IF N$="" THEN STOP
080  PRINT N$
100  POKE X+5,( CODE N$(1)-28)*16+ CODE N$(2)-28
110  LET A= USR X
120  GOTO 60

```

7.7 Programm zur Hex-Eingabe und Ausgabe nach Tor A

Nachdem wir nun Zahlenwerte ausgeben können, wollen wir Tor A als Eingang betreiben und Zahlenwerte einlesen. Wir verwenden dazu nicht Betriebsart 1, sondern die Betriebsart 3. In der Betriebsart 1 müßten wir für die Eingabe der Daten in das Eingangsregister, einen negativen Impuls auf der ASTB-Leitung erzeugen. Dies ist in Betriebsart 3 nicht notwendig. Außerdem können in dieser Betriebsart einzelne Leitungen als Eingang oder als Ausgang geschaltet sein.

D7	D6	D5	D4	D3	D2	D1	D0
1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0

7.8 Steuerwort für Betriebsart 3

Dazu wird nach Wahl der Betriebsart 3 ein zweites Steuerwort an das Steuerregister übergeben. Das Format dieses Steuerwortes zeigt Abbildung 7.8. Wird in diesem Steuerwort ein Bit auf 1 gesetzt, so ist diese Leitung ein Eingang, andernfalls ein Ausgang. Damit erhalten wir folgendes Maschinenprogramm (Abbildung 7.9).

LD A, CF	CF → A, Betriebsart 3
OUT FA, A	(A) → Tor A, Steuerwort
LD A, FF	FF → A, Lie Ltg. Eingänge
OUT FA, A	(A) → Tor A, Steuerwort
IN F8, A	(TORA) → A, Daten einlesen
LD B, 0	0 → B
RET	zurück in BASIC

7.9 Maschinenprogramm für Dateneingabe

In diesem Programm machen wir davon Gebrauch, daß, wenn wir das Maschinenprogramm mit PRINT USR X aufrufen, der Inhalt des BC-Registers ausgedruckt wird.

Legen wir nun eine Eingangsleitung, z. B. PA1 auf 1 (Spannung zwischen 3.0 und 5.0 V) so wird von ZX81 eine 2 auf den Bildschirm ausgegeben.

Die in diesem Beispiel für die Eingabe benutzte Betriebsart 3 hätte man auch für die Ausgabe verwenden können. Im Programm in Abbildung 7.9 würde dann die Zeile 3 LD A, 00 lauten. In dieser Betriebsart kann man auch Ein- und Ausgänge gleichzeitig betreiben. Ändert man Zeile 3 in LD A, F0 so sind die Leitungen PA0 bis PA3 Ausgänge, PA4 bis PA7 Eingänge.

Anstelle des Tores A kann man auch Tor B verwenden. Die Adresse für das Steuerwort ist dann \$FB und die Adresse für Daten \$F9.

Die Programme in diesem Kapitel sind so ausgelegt, daß sie auch mit der 1k Version des ZX81 arbeiten. Mit der 16k Version kann man auch den Maschinencode Monitor MONI verwenden und die Programme direkt eingeben und ausführen lassen.

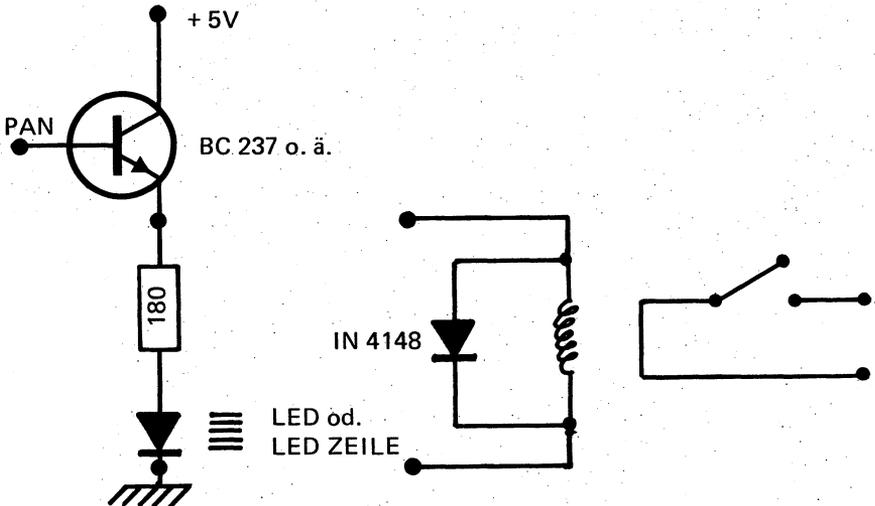
Steuerungsprogramme

8. Steuerungsprogramme

In diesem Kapitel sollen nun einige Anwendungen der Z80 A PIO in Verbindung mit dem ZX81 gezeigt werden.

Für diese Programme ist eine, wenn auch geringe, Kenntnis in Z80 Maschinenprogrammierung erforderlich. Die Sprache BASIC allein ist zu langsam, und gleichzeitig kennt das ZX81 BASIC keine IN- oder OUT-Befehle. Diese müssen in Maschinensprache formuliert werden und durch die USR Funktion aufgerufen werden, wie dies schon im letzten Kapitel geschehen ist.

8.1 Ansteuerung einer Leuchtdiodenzeile



8.1 Anschluß einer Leuchtdiode oder eines Relais an eine Ausgangsleitung der PIO

Zur Sichtbarmachung der Pegel an den Toren der PIO wird an jeden Anschluß eine Leuchtdiode über einen Transistor als Trennverstärker angeschlossen (Abbildung 8.1). Statt der Leuchtdiode und des Widerstandes kann auch ein 5V-Relais mit Schutzdiode angeschlossen werden. Die Verwendung eines Relais oder eines Opto-Kopplers ist immer dann sinnvoll, wenn Rechner und externe Geräte potentialmäßig voneinander getrennt werden müssen. Auf der Experimentierplatine ZX81 ist für die Transistoren und die Widerstände, sowie für Leuchtdiodenzeilen mit 8 Elementen Platz vorgesehen. Wenn größere Ströme geschaltet werden, so sollten die Transistoren aus einem externen Netzgerät versorgt werden.

8.2 Lauflicht

Wir wollen unsere Leuchtdioden der Reihe nach Aufleuchten und wieder Verlöschen lassen. Das soll sich solange wiederholen, bis das Programm durch BREAK unterbrochen wird. Wir benutzen für dieses Programm in Abbildung 8.2 den Anfang von Programm 7.7 bis Zeile 50.

Dann setzen wir $N = 0$ und bilden $M = 2^N$. Mit $N = 0$ ist $M = 1$, mit $N = 1$ ist $M = 2$, mit $N = 3$ ist $M = 8$ usw. Diese Zahlen speichern wir im Maschinenprogramm, das wir in Zeile 110 aufrufen. N wird jedesmal um 1 erhöht, wenn es größer als 7 ist, wird es wieder auf Null gesetzt.

```

001  REM AAAAAAAAAA
010  LET X=16514
020  LET M$="3E0FD3FA3E00D3F8C9"
030  FOR I=1 TO LEN M$-1 STEP 2
040  POKE X+ INT ((I-1)/2) , ( CODE M$(I)-28) *16+
      CODE M$(I+1)-28
050  NEXT I
060  LET N=0
070  LET M=2 ** N
080  POKE X+5 ,M
090  LET N=N+1
100  IF N>7 THEN LET N=0
110  LET A= USR X
120  GOTO 70

```

8.2 Lauflicht

8.3 Lichtbalken

Wir verändern das obige Programm so, daß die aufleuchtenden Leuchtdioden nicht mehr verlöschen, bis alle 8 brennen, dann werden sie der Reihe nach wieder ausgeschaltet und der Zyklus beginnt von Neuem. Das Programm zeigt Abbildung 8.3. Zuerst werden durch Zeile 75 $LET M = M + 2 ** N$ alle Leuchtdioden angeschaltet, solange $N < 8$ ist. Dann werden sie durch Zeile 120 $LET M = M - 2 ** N$ wieder ausgeschaltet, solange $N > 0$ ist. Dann beginnt das Programm von Neuem in Zeile 60.

```
001  REM AAAAAAAAAA
010  LET X=16514
020  LET M$="3E0FD3FA3E00D3F8C9"
030  FOR I=1 TO LEN M$-1 STEP 2
040  POKE X+ INT ((I-1)/2) ,( CODE M$(I)-28) *16+
      CODE M$(I+1)-28
050  NEXT I
060  LET N=0
070  LET M=0
075  LET M=M+2 ** N
077  POKE X+5,M
080  LET A= USR X
090  LET N=N+1
100  IF N<8 THEN GOTO 75
110  LET N=N-1
120  LET M=M-2 ** N
125  POKE X+5,M
130  LET A= USR X
140  IF N>0 THEN GOTO 110
150  GOTO 60
```

8.3 Lichtbalken

8.4 Blinklicht

Wir ändern nun unser Programm ab Zeile 60 folgendermaßen ab:

```
001  REM AAAAAAAAAA
010  LET X=16514
020  LET M$="3E0FD3FA3E00D3F8C9"
```

```

030 FOR I=1 TO LEN M$-1 STEP 2
040 POKE X+ INT ((I-1)/2), ( CODE M$(I)-28)*16+
    CODE M$(I+1)-28
050 NEXT I
060 LET N=0
070 POKE X+5,1
080 LET A=USR X
085 FOR I=1 TO 10
086 NEXT I
090 POKE X+5,0
100 LET A=USR X
105 FOR I=1 TO 10
106 NEXT I
110 GOTO 60

```

8.4 Blinklicht

Wir schalten also Leuchtdiode 1 ein, und gleich wieder aus. Wenn wir das Programm starten, so flackert LED1 mit einer Frequenz von ungefähr 16 Hz. Dies ist für ein Blinklicht viel zu hoch, so daß wir 2 Warteschleifen einführen müssen. Das vollständige Programm zeigt nun Abbildung 8.4. Durch Ändern der Endparameter in der FOR-NEXT Schleife läßt sich die Frequenz des Blinkens ändern.

Will man nun aber Geräusche und Töne erzeugen, so sind diese Frequenzen viel zu niedrig. Man muß also die Erzeugung höherer Frequenzen in Maschinensprache programmieren.

Anhang

9. Anhang

9.1 Vergleich ZX81 BASIC-Befehle mit anderen Rechnern

In Abbildung 9.1 sind die wichtigsten BASIC-Anweisungen für einige Rechner zusammengestellt.

	ZX81	Apple II	ATARI 400/800	PET	TRS-80 Lev. II
Systembefehle					
RUN	X	X	X	X	X
NEW	X	X	X	X	X
SAVE	X	X	X	X	X
LOAD	X	X	X	X	X
LIST	X	X	X	X	X
LLIST	X				X
COPY	X				
DELETE		X			X
Anweisungen					
LET	X	X	X	X	X
GOTO	X	X	X	X	X
IF-THEN	X	X	X	X	X
FOR-NEXT-STEP	X	X	X	X	X
GOSUB-RETURN	X	X	X	X	X
PRINT	X	X	X	X	X
PRINT AT	X				
INPUT	X	X	X	X	X
INKEY\$	X	GET\$		GET\$	X
STOP	X	X	X	X	X
READ-DATA		X	X	X	X
RESTORE		X	X	X	X
ON .. GOTO		X	X	X	X
DIM	X	X	X	X	X
REM	X	X	X	X	X
TAB	X	X		X	X
AND-OR-NOT	X	X		X	X

	ZX81	Apple II	ATARI 400/800	PET	TRS-80 Lev. II
Anweisungen f. Zeichenketten					
LEN	X	X	X	X	X
CHR\$	X	X	X	X	X
STR\$	X	X	X	X	X
ASC	CODE	X	X	X	X
LEFT\$		X		X	X
MID\$	(N TO M)	X	(N,M)	X	X
RIGHT\$		X		X	X
VAL	X	X	X	X	X
Funktionen					
INT	X	X	X		X
ABS	X	X	X	X	X
COS	X	X	X	X	X
SIN	X	X	X	X	X
ATN	X	X	X	X	X
SGN	X	X	X	X	X
SQR	X	X	X	X	X
TAN	X	X		X	X
PEEK	X	X	X	X	X
POKE	X	X	X	X	X
USR	X	CALL	X	X	X

9.1 Vergleich ZX81 BASIC-Befehle mit anderen Rechnern



Der leistungsfähige Mikrocomputer Sinclair ZX81. Als Peripherie ist der Sinclair ZX - Drucker erhältlich, der volle Alphanumerik in 32 Anschlägen Breite sowie hochentwickelte Grafiken bietet.

Im folgenden sollen nun einige Unterschiede zwischen dem im ZX81 verwendeten BASIC und anderen BASIC Dialekten besprochen werden. Eine der wesentlichsten Unterschiede ist folgender: Allen Variablen die im Programm verwendet werden, muß ein Wert zugewiesen werden, sonst erfolgt Fehlermeldung 2: "Nicht definierte Variable verwendet". Andere BASIC-Dialekte kennen diesen Fehler nicht, sie weisen jeder Variablen den Wert Null zu. Dies kann zu verfälschten Rechenergebnissen führen, wenn man z. B. bei der Berechnung des Ausdrucks

```
100 LET A = B * C + 4.5 * K
```

vergessen hat, der Variablen K vorher den richtigen Wert zuzuweisen. ZX81 merkt diesen Fehler.

Andererseits muß man aber in ZX81 BASIC darauf achten, daß tatsächlich allen Variablen ein Wert zugewiesen wird. Besonders dann, wenn man mit bedingten Anweisungen arbeitet.

Betrachten wir dazu das folgende kleine Beispiel:

```
100 INPUT A$
110 IF A$ = "J" THEN LET C = 1
120 PRINT C
```

Wird die Bedingung in Zeile 110 nicht erfüllt, so druckt jeder BASIC Interpreter in Zeile 120 den Wert 0 aus. Das ZX81 Basic liefert die Fehlermeldung 2/120. Für den ZX81 lautet das Programm richtig:

```
100 INPUT A$
105 LET C = 0
110 IF A$ = "J" THEN LET C = 1
120 PRINT C
```

Im ZX81 BASIC ist in einer Zeile immer nur eine Anweisung enthalten. Sollen bei einer bedingten Anweisung mehrere Befehle ausgeführt werden, so muß dies durch eine GOSUB oder durch eine GOTO-Anweisung ausgeführt werden. In anderen BASIC-Dialekten ist folgende Schreibweise erlaubt:

```

100 IF A > 0 THEN B = 0 : C = 199 : A$ = "JA" : GOTO 120
110 B = 1 : C = -199 : A$ = "NEIN"
120 PRINT "HIER GEHTS WEITER"

```

Wenn die Bedingung erfüllt ist, wird der Rest der Zeile abgearbeitet, ansonsten die Zeile 110.

Beim ZX81 wird man diese Abfrage folgendermaßen programmieren:

```

100 IF A > 0 THEN GOTO 200
110 LET B = 1
120 LET C = -199
130 LET A$ = "NEIN"
140 PRINT "HIER GEHTS WEITER"

199 STOP
200 LET B = 0
210 LET C = 199
220 LET A$ = "JA"
230 GOTO 140

```

oder aber

```

100 IF A > 0 THEN GOTO 150
110 LET B = 1
120 LET C = -199
130 LET A$ = "NEIN"
140 GOTO 180
150 LET B = 0
160 LET C = 199
170 LET A$ = "JA"
180 PRINT "HIER GEHTS WEITER"

```

Ein weiteres Beispiel ist im Programm DEZ-HEX Wandlung (Abbildung 4.13) enthalten.

Das ZX81 BASIC kann ein Unterprogramm durch einen Namen aufrufen. Diese Möglichkeit sollte oft angewendet werden, da sie erheblich zur Lesbarkeit des Programms beiträgt. Ein Programm ist viel leichter zu verstehen, wenn ein Unterprogrammaufruf in der folgenden Weise

geschieht:

```
50 LET RECHNE = 1000
```

```
.  
. .  
. . .
```

```
100 GOSUB RECHNE
```

Das ZX81 BASIC kennt keine Anweisung der Form

```
100 INPUT "BESTELLNUMMER: "; N
```

Den gleichen Ausdruck auf dem Bildschirm wie bei der obigen Anweisung erhält man bei ZX 81 mit folgenden Programmzeilen:

```
100 PRINT "BESTELLNUMMER: ";
```

```
110 INPUT N
```

```
120 PRINT N
```

Einige Unterschiede gibt es auch bei der Behandlung von Zeichenketten. In Microsoftbasic gibt der Befehl

```
50 A$ = "ABCDEF"
```

```
100 PRINT LEFT$(A$,4)
```

die Buchstabenfolge ABCD aus.

In ZX81 BASIC lautet der Befehl

```
50 LET A$ = "ABCDEF"
```

```
100 PRINT A$(1 TO 4)
```

Der Microsoft Befehl

```
100 PRINT RIGHT$(A$,3)
```

bewirkt die Ausgabe von DEF, und der Befehl

```
100 PRINT MID$(A$,2,4)
```

die Ausgabe von BCDE

Diese beiden Befehle in ZX81 BASIC lauten dann

PRINT RIGHT\$(A\$, N)	entspricht	LET L = LEN A\$ PRINT A\$(L-N+1 TO L)
PRINT MID\$(AS, N, M)	entspricht	PRINT A\$(N TO N+M-1)
und nochmals		
PRINT LEFT\$(A\$, N)	entspricht	PRINT A\$(1 TO N)

Bei der Verwendung der CHR\$ Funktion wird nicht der ASCII-Wert, sondern der intern verwendete Code ausgegeben. Dieser Code ist im Handbuch, Anhang A Seite 181 bis 187 angegeben.

Weitere Unterschiede zu anderen BASIC-Dialekten.

Der Befehl für die Potenzierung lautet beim ZX81

$a^b = a ** b$ und nicht $a \uparrow b$

Das Argument einer Funktion wird nicht geklammert, die Eingabe von Klammern bewirkt aber keine Fehlerberechnung.

PRINT SIN 45 entspricht PRINT SIN (45)

Wie bei Taschenrechnern, ist die Zahl π programmiert.

Wie die nicht vorhandenen Befehle ON GOTO und READ, DATA, RESTORE umgangen werden, ist im Handbuch beschrieben.

Einen Unterschied zu anderen BASIC-Rechnern gibt es auch bei der Ausführung des GOTO-Befehls. Beim ZX81 erscheint keine Fehlermeldung, wenn ein GOTO 230 programmiert ist, aber keine Zeile 230 vorhanden ist. Der Rechner sucht nach der nächsthöheren Zeilennummer und führt dann diese aus.

Die Eingabe von Zahlen kann auch in der Exponentenschreibweise erfolgen. Die Zahl 100 kann auch als 1E2 eingegeben werden. Dies entspricht der Eingabe 1×10^2 .

9.2 HEX-DEZ ZAHLEN

4		3		2		1	
HEX	DEZ	HEX	DEZ	HEX	DEZ	HEX	DEZ
0	0	0	0	0	0	0	0
1	4,096	1	256	1	16	1	1
2	8,192	2	512	2	32	2	2
3	12,288	3	768	3	48	3	3
4	16,384	4	1,024	4	64	4	4
5	20,480	5	1,280	5	80	5	5
6	24,576	6	1,536	6	96	6	6
7	28,672	7	1,792	7	112	7	7
8	32,768	8	2,048	8	128	8	8
9	36,864	9	2,304	9	144	9	9
A	40,960	A	2,560	A	160	A	10
B	45,056	B	2,816	B	176	B	11
C	49,152	C	3,072	C	192	C	12
D	53,248	D	3,328	D	208	D	13
E	57,344	E	3,584	E	224	E	14
F	61,440	F	3,840	F	240	F	15

9.1 Umwandlung von Hexadezimalzahlen in Dezimalzahlen

9.3 Tabelle für Vorwärts- und Rückwärtsverzweigungen

LSD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
MSD	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127

9.2 Vorwärts-Verzweigungen

LSD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
MSD																
8	128	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113
9	112	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97
A	96	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81
B	80	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65
C	64	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49
D	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33
E	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
F	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

9.21 Rückwärts-Verzweigungen

9.4 Befehlssatz für Z80 A CPU

OBJ CODE	SOURCE STATEMENT	OPERATION
8E DD8E05 FD8E05	ADC ADC ADC	A,(HL) A,(IX+d) A,(IY+d)
8F 88 89 8A 8B 8C 8D CE20	ADC ADC ADC ADC ADC ADC ADC ADC	A,A A,B A,C A,D A,E A,H A,L A,n
ED4A ED5A ED6A ED7A	ADC ADC ADC ADC	HL,BC HL,DE HL,HL HL,SP
86 DD8605 FD8605	ADD ADD ADD	A,(HL) A,(IX+d) A,(IY+d)
87 80 81 82 83 84 85 C620	ADD ADD ADD ADD ADD ADD ADD ADD	A,A A,B A,C A,D A,E A,H A,L A,n
09 19 29 39	ADD ADD ADD ADD	HL,BC HL,DE HL,HL HL,SP
DD09 DD19 DD29 DD39	ADD ADD ADD ADD	IX,BC IX,DE IX,IX IX,SP
FD09 FD19 FD29 FD39	ADD ADD ADD ADD	IY,BC IY,DE IY,IY IY,SP
A6 DDA605 FDA605	AND AND AND	(HL) (IX+d) (IY+d)
A7 A0 A1 A2 A3 A4 A5 E620	AND AND AND AND AND AND AND AND	A B C D E H L n
CB46 DDCB0546 FDCB0546	BIT BIT BIT	0,(HL) 0,(IX+d) 0,(IY+d)
CB47 CB40 CB41 CB42 CB43 CB44	BIT BIT BIT BIT BIT BIT	0,A 0,B 0,C 0,D 0,E 0,H

CODE	STATEMENT	OPERATION
CB45	BIT 0,L	Test Bit b of Location or Reg.
CB4E	BIT 1,(HL)	
DDCB054E	BIT 1,(IX+d)	
FDCB054E	BIT 1,(IY+d)	
CB4F	BIT 1,A	
CB48	BIT 1,B	
CB49	BIT 1,C	
CB4A	BIT 1,D	
CB4B	BIT 1,E	
CB4C	BIT 1,H	
CB4D	BIT 1,L	
CB56	BIT 2,(HL)	
DDCB0556	BIT 2,(IX+d)	
FDCB0556	BIT 2,(IY+d)	
CB57	BIT 2,A	
CB50	BIT 2,B	
CB51	BIT 2,C	
CB52	BIT 2,D	
CB53	BIT 2,E	
CB54	BIT 2,H	
CB55	BIT 2,L	
CB5E	BIT 3,(HL)	
DDCB055E	BIT 3,(IX+d)	
FDCB055E	BIT 3,(IY+d)	
CB5F	BIT 3,A	
CB58	BIT 3,B	
CB59	BIT 3,C	
CB5A	BIT 3,D	
CB5B	BIT 3,E	
CB5C	BIT 3,H	
CB5D	BIT 3,L	
CB66	BIT 4,(HL)	
DDCB0566	BIT 4,(IX+d)	
FDCB0566	BIT 4,(IY+d)	
CB67	BIT 4,A	
CB60	BIT 4,B	
CB61	BIT 4,C	
CB62	BIT 4,D	
CB63	BIT 4,E	
CB64	BIT 4,H	
CB65	BIT 4,L	
CB6E	BIT 5,(HL)	
DDCB056E	BIT 5,(IX+d)	
FDCB056E	BIT 5,(IY+d)	
CB6F	BIT 5,A	
CB68	BIT 5,B	
CB69	BIT 5,C	
CB6A	BIT 5,D	
CB6B	BIT 5,E	
CB6C	BIT 5,H	
CB6D	BIT 5,L	
CB76	BIT 6,(HL)	
DDCB0576	BIT 6,(IX+d)	
FDCB0576	BIT 6,(IY+d)	
CB77	BIT 6,A	
CB70	BIT 6,B	
CB71	BIT 6,C	
CB72	BIT 6,D	
CB73	BIT 6,E	
CB74	BIT 6,H	
CB75	BIT 6,L	
CB7E	BIT 7,(HL)	
DDCB057E	BIT 7,(IX+d)	

OBJ CODE	SOURCE STATEMENT	OPERATION	
FDCB057E CB7F CB78 CB79 CB7A CB7B CB7C CB7D	BIT BIT BIT BIT BIT BIT BIT BIT	7,(IY+d) 7,A 7,B 7,C 7,D 7,E 7,H 7,L	Test Bit b Location or Reg.
DC8405 FC8405 D48405 C48405 F48405 EC8405 E48405 CC8405	CALL CALL CALL CALL CALL CALL CALL CALL	C,nn M,nn NC,nn NZ,nn P,nn PE,nn PO,nn Z,nn	Call Subroutine at Location nn if Condi- tion True
CDB405	CALL	nn	Unconditional Call to Subroutine at nn
3F	CCF		Complement Carry Flag
BE DDBE05 FDBE05 BF B8 B9 BA BB BC BD FE20	CP CP CP CP CP CP CP CP CP CP CP	(HL) (IX+d) (IY+d) A B C D E H L n	Compare Operand with Acc.
EDA9	CPD		Compare Location (HL) and Acc. Decrement HL and BC
EDB9	CPDR		Compare Location (HL) and Acc. Decre- ment HL and BC, Repeat until BC = 0
EDA1	CPI		Compare Location (HL) and Acc., Incre- ment HL and Decre- ment BC
EDB1	CPIR		Compare Location (HL) and Acc. Incre- ment HL, Decrement BC, Repeat until BC = 0
2F	CPL		Complement Acc. (1's Comp)
27	DAA		Decimal Adjust Acc.
35 DD3505 FD3505 3D 05 0B 0D 15 1B	DEC DEC DEC DEC DEC DEC DEC DEC DEC	(HL) (IX+d) (IY+d) A B BC C D DE	Decrement Operand

OBJ CODE	SOURCE STATEMENT	OPERATION
1D 25 2B DD2B FD2B 2D 3B	DEC E DEC H DEC HL DEC IX DEC IY DEC L DEC SP	Decrement Operand
F3	DI	Disable Interrupts
102E	DJNZ e	Decrement B and Jump Relative if B = 0
FB	EI	Enable Interrupts
E3 DDE3 FDE3	EX (SP),HL EX (SP),IX EX (SP),IY	Exchange Location and (SP)
08	EX AF,AF'	Exchange the Con- tents of AF and AF'
EB	EX DE,HL	Exchange the Con- tents of DE and HL
D9	EXX	Exchange the Con- tents of BC,DE,HL with Contents of BC',DE',HL' Respec- tively
76	HALT	HALT (Wait for Inter- rupt or Reset)
ED46 ED56 ED5E	IM 0 IM 1 IM 2	Set Interrupt Mode
ED78 ED40 ED48 ED50 ED58 ED60 ED68	IN A,(C) IN B,(C) IN C,(C) IN D,(C) IN E,(C) IN H,(C) IN L,(C)	Load Reg. with Input from Device (C)
34 DD3405 FD3405 3C 04 03 0C 14 13 1C 24 23 DD23 FD23 2C 33	INC (HL) INC (IX+d) INC (IY+d) INC A INC B INC BC INC C INC D INC DE INC E INC H INC HL INC IX INC IY INC L INC SP	Increment Operand
DB20	IN A,(n)	Load Acc. with Input from Device n
EDAA	IND	Load Location (HL) with Input from Port (C), Decrement HL and B

OBJ CODE	SOURCE STATEMENT	OPERATION
ED538405	LD (nn),DE	Load Source to Des- tination
228405	LD (nn),HL	
DD228405	LD (nn),IX	
FD228405	LD (nn),IY	
ED738405	LD (nn),SP	
0A	LD A,(BC)	
1A	LD A,(DE)	
7E	LD A,(HL)	
DD7E05	LD A,(IX+d)	
FD7E05	LD A,(IY+d)	
3A8405	LD A,(nn)	
7F	LD A,A	
78	LD A,B	
79	LD A,C	
7A	LD A,D	
7B	LD A,E	
7C	LD A,H	
ED57	LD A,I	
7D	LD A,L	
3E20	LD A,n	
ED5F	LD A,R	
46	LD B,(HL)	
DD4605	LD B,(IX+d)	
FD4605	LD B,(IY+d)	
47	LD B,A	
40	LD B,B	
41	LD B,C	
42	LD B,D	
43	LD B,E	
44	LD B,H	
45	LD B,L	
0620	LD B,n	
ED488405	LD BC,(nn)	
018405	LD BC,nn	
4E	LD C,(HL)	
DD4E05	LD C,(IX+d)	
FD4E05	LD C,(IY+d)	
4F	LD C,A	
48	LD C,B	
49	LD C,C	
4A	LD C,D	
4B	LD C,E	
4C	LD C,H	
4D	LD C,L	
0E20	LD C,n	
56	LD D,(HL)	
DD5605	LD D,(IX+d)	
FD5605	LD D,(IY+d)	
57	LD D,A	
50	LD D,B	
51	LD D,C	
52	LD D,D	
53	LD D,E	
54	LD D,H	
55	LD D,L	
1620	LD D,n	
ED588405	LD DE,(nn)	
118405	LD DE,nn	
5E	LD E,(HL)	
DD5E05	LD E,(IX+d)	
FD5E05	LD E,(IY+d)	
5F	LD E,A	
58	LD E,B	
59	LD E,C	

OBJ CODE	SOURCE STATEMENT	OPERATION
5A	LD E,D	Load Source to Destination
5B	LD E,E	
5C	LD E,H	
5D	LD E,L	
1E20	LD E,n	
66	LD H,(HL)	
DD6605	LD H,(IX+d)	
FD6605	LD H,(IY+d)	
67	LD H,A	
60	LD H,B	
61	LD H,C	
62	LD H,D	
63	LD H,E	
64	LD H,H	
65	LD H,L	
2620	LD H,n	
2A8405	LD HL,(nn)	
218405	LD HL,nn	
ED47	LD I,A	
DD2A8405	LD IX,(nn)	
DD218405	LD IX,nn	
FD2A8405	LD IY,(nn)	
FD218405	LD IY,nn	
6E	LD L,(HL)	
DD6E05	LD L,(IX+d)	
FD6E05	LD L,(IY+d)	
6F	LD L,A	
68	LD L,B	
69	LD L,C	
6A	LD L,D	
6B	LD L,E	
6C	LD L,H	
6D	LD L,L	
2E20	LD L,n	
ED4F	LD R,A	
ED7B8405	LD SP,(nn)	
F9	LD SP,HL	
DDF9	LD SP,IX	
fdf9	LD SP,IY	
318405	LD SP,nn	
EDA8	LDD	Load Location (DE) with Location (HL), Decrement DE,HL and BC
EDB8	LDDR	Load Location (DE) with Location (HL), Repeat until BC = 0
EDA0	LDI	Load Location (DE) with Location (HL), Increment DE,HL, Decrement BC
EDB0	LDIR	Load Location (DE) with Location (HL), Increment DE,HL, Decrement BC and Repeat until BC = 0
ED44	NEG	Negate Acc. (2's Complement)
00	NOP	No Operation
B6	OR (HL)	Logical "OR" of Operand and Acc.
DDB605	OR (IX+d)	

OBJ CODE	SOURCE STATEMENT	OPERATION
FDB605 B7 B0 B1 B2 B3 B4 B5 F620	OR (IY+d) OR A OR B OR C OR D OR E OR H OR L OR n	Logical "OR" of Operand and Acc.
ED8B	OTDR	Load Output Port (C) with Location (HL) Decrement HL and B, Repeat until B = 0
EDB3	OTIR	Load Output Port (C) with Location (HL), Increment HL, Decre- ment B, Repeat until B = 0
ED79 ED41 ED49 ED51 ED59 ED61 ED69	OUT (C),A OUT (C),B OUT (C),C OUT (C),D OUT (C),E OUT (C),H OUT (C),L	Load Output Port (C) with Reg.
D320	OUT (n),A	Load Output Port (n) with Acc.
EDAB	OUTD	Load Output Port (C) with Location (HL), Decrement HL and B
EDA3	OUTI	Load Output Port (C) with Location (HL), Increment HL and Decrement B
F1 C1 D1 E1 DDE1 FDE1	POP AF POP BC POP DE POP HL POP IX POP IY	Load Destination with Top of Stack
F5 C5 D5 E5 DDE5 FDE5	PUSH AF PUSH BC PUSH DE PUSH HL PUSH IX PUSH IY	Load Source to Stack
CB86 DDC80586 FDC80586 CB87 CB80 CB81 CB82 CB83 CB84 CB85 CB8E DDC8058E FDC8058E CB8F	RES 0,(HL) RES 0,(IX+d) RES 0,(IY+d) RES 0,A RES 0,B RES 0,C RES 0,D RES 0,E RES 0,H RES 0,L RES 1,(HL) RES 1,(IX+d) RES 1,(IY+d) RES 1,A	Reset Bit b of Operand

OBJ CODE	SOURCE STATEMENT	OPERATION
CB88	RES 1,B	Reset Bit b of Operand
CB89	RES 1,C	
CB8A	RES 1,D	
CB8B	RES 1,E	
CB8C	RES 1,H	
CB8D	RES 1,L	
CB96	RES 2,(HL)	
DDCB0596	RES 2,(IX+d)	
FDCB0596	RES 2,(IY+d)	
CB97	RES 2,A	
CB90	RES 2,B	
CB91	RES 2,C	
CB92	RES 2,D	
CB93	RES 2,E	
CB94	RES 2,H	
CB95	RES 2,L	
CB9E	RES 3,(HL)	
DDCB059E	RES 3,(IX+d)	
FDCB059E	RES 3,(IY+d)	
CB9F	RES 3,A	
CB98	RES 3,B	
CB99	RES 3,C	
CB9A	RES 3,D	
CB9B	RES 3,E	
CB9C	RES 3,H	
CB9D	RES 3,L	
CBA6	RES 4,(HL)	
DDCB05A6	RES 4,(IX+d)	
FDCB05A6	RES 4,(IY+d)	
CBA7	RES 4,A	
CBA0	RES 4,B	
CBA1	RES 4,C	
CBA2	RES 4,D	
DBA3	RES 4,E	
CBA4	RES 4,H	
CBA5	RES 4,L	
CBAE	RES 5,(HL)	
DDCB05AE	RES 5,(IX+d)	
FDCB05AE	RES 5,(IY+d)	
CBAF	RES 5,A	
CBA8	RES 5,B	
CBA9	RES 5,C	
CBAA	RES 5,D	
CBAB	RES 5,E	
CBAC	RES 5,H	
CBAD	RES 5,L	
CB86	RES 6,(HL)	
DDCB05B6	RES 6,(IX+d)	
FDCB05B6	RES 6,(IY+d)	
CB87	RES 6,A	
CB80	RES 6,B	
CB81	RES 6,C	
CB82	RES 6,D	
CB83	RES 6,E	
CB84	RES 6,H	
CB85	RES 6,L	
CB8E	RES 7,(HL)	
DDCB05BE	RES 7,(IX+d)	
FDCB05BE	RES 7,(IY+d)	
CB8F	RES 7,A	
CB88	RES 7,B	
CB89	RES 7,C	
CB8A	RES 7,D	

OBJ CODE	SOURCE STATEMENT		OPERATION	
CBBB	RES	7,E	Reset Bit b of Operand	
CBBC	RES	7,H		
CBBD	RES	7,L		
C9	RET		Return from Subroutine	
D8	RET	C	Return from Subroutine if Condi- tion True	
F8	RET	M		
D0	RET	NC		
C0	RET	NZ		
F0	RET	P		
E8	RET	PE		
E0	RET	P0		
C8	RET	Z		
ED4D	RETI		Return from Interrupt	
ED45	RETN		Return from Non- Maskable Interrupt	
CB16	RL	(HL)	Rotate Left Through Carry	
DDCB0516	RL	(IX+d)		
FDCB0516	RL	(IY+d)		
CB17	RL	A		
CB10	RL	B		
CB11	RL	C		
CB12	RL	D		
CB13	RL	E		
CB14	RL	H		
CB15	RL	L		
17	RLA			Rotate Left Acc. Through Carry
CB06	RLC	(HL)		Rotate Left Circular
DDCB0506	RLC	(IX+d)		
FDCB0506	RLC	(IY+d)		
CB07	RLC	A		
CB00	RLC	B		
CB01	RLC	C		
CB02	RLC	D		
CB03	RLC	E		
CB04	RLC	H		
CB05	RLC	L		
07	RLCA		Rotate Left Circular Acc.	
ED6F	RLD		Rotate Digit Left and Right between Acc. and and Location (HL)	
CB1E	RR	(HL)	Rotate Right Through Carry	
DDCB051E	RR	(IX+d)		
FDCB051E	RR	(IY+d)		
CB1F	RR	A		
CB18	RR	B		
CB19	RR	C		
CB1A	RR	D		
CB1B	RR	E		
CB1C	RR	H		
CB1D	RR	L		
1F	RRA			Rotate Right Acc. Through Carry
CB0E	RRC	(HL)	Rotate Right Circular	
DDCB050E	RRC	(IX+d)		
FDCB050E	RRC	(IY+d)		
CB0F	RRC	A		

OBJ CODE	SOURCE STATEMENT		OPERATION
CB08	RRC	B	Rotate Right Circular
CB09	RRC	C	
CB0A	RRC	D	
CB0B	RRC	E	
CB0C	RRC	H	
CB0D	RRC	L	
OF	RRCA		Rotate Right Circular Acc.
ED67	RRD		Rotate Digit Right and Left Between Acc. and Location (HL)
C7	RST	00H	Restart to Location
CF	RST	08H	
D7	RST	10H	
DF	RST	18H	
E7	RST	20H	
EF	RST	28H	
F7	RST	30H	
FF	RST	38H	
DE20	SBC	A,n	
9E	SBC	A,(HL)	
DD9E05	SBC	A,(IX+d)	
FD9E05	SBC	A,(IY+d)	
9F	SBC	A,A	
98	SBC	A,B	
99	SBC	A,C	
9A	SBC	A,D	
9B	SBC	A,E	
9C	SBC	A,H	
9D	SBC	A,L	
ED42	SBC	HL,BC	
ED52	SBC	HL,DE	
ED62	SBC	HL,HL	
ED72	SBC	HL,SP	
37	SCF		Set Carry Flag (C = 1)
CBC6	SET	0,(HL)	Set Bit b of Location
DDCB05C6	SET	0,(IX+d)	
FDCB05C6	SET	0,(IY+d)	
CBC7	SET	0,A	
CBC0	SET	0,B	
CBC1	SET	0,C	
CBC2	SET	0,D	
CBC3	SET	0,E	
CBC4	SET	0,H	
CBC5	SET	0,L	
CBCE	SET	1,(HL)	
DDCB05CE	SET	1,(IX+d)	
FDCB05CE	SET	1,(IY+d)	
CBCF	SET	1,A	
CBC8	SET	1,B	
CBC9	SET	1,C	
CBCA	SET	1,D	
CBCB	SET	1,E	
CBCC	SET	1,H	
CBCD	SET	1,L	
CBD6	SET	2,(HL)	
DDCB05D6	SET	2,(IX+d)	
FDCB05D6	SET	2,(IY+d)	
CBD7	SET	2,A	
CBD0	SET	2,B	
CBD1	SET	2,C	
CBD2	SET	2,D	

OBJ CODE	SOURCE STATEMENT	OPERATION
CBD3	SET 2,E	Set Bit b of Location
CBD4	SET 2,H	
CBD5	SET 2,L	
CBD8	SET 3,B	
CBDE	SET 3,(HL)	
DDCB05DE	SET 3,(IX+d)	
FDCB05DE	SET 3,(IY+d)	
CBDF	SET 3,A	
CBD9	SET 3,C	
CBDA	SET 3,D	
CBDB	SET 3,E	
CBDC	SET 3,H	
CBDD	SET 3,L	
CBE6	SET 4,(HL)	
DDCB05E6	SET 4,(IX+d)	
FDCB05E6	SET 4,(IY+d)	
CBE7	SET 4,A	
CBE0	SET 4,B	
CBE1	SET 4,C	
CBE2	SET 4,D	
CBE3	SET 4,E	
CBE4	SET 4,H	
CBE5	SET 4,L	
CBEE	SET 5,(HL)	
DDCB05EE	SET 5,(IX+d)	
FDCB05EE	SET 5,(IY+d)	
CBEF	SET 5,A	
CBEB	SET 5,B	
CBE9	SET 5,C	
CBEA	SET 5,D	
CBEB	SET 5,E	
CBEC	SET 5,H	
CBED	SET 5,L	
CBF6	SET 6,(HL)	
DDCB05F6	SET 6,(IX+d)	
FDCB05F6	SET 6,(IY+d)	
CBF7	SET 6,A	
CBF0	SET 6,B	
CBF1	SET 6,C	
CBF2	SET 6,D	
CBF3	SET 6,E	
CBF4	SET 6,H	
CBF5	SET 6,L	
CBFE	SET 7,(HL)	
DDCB05FE	SET 7,(IX+d)	
FDCB05FE	SET 7,(IY+d)	
CBFF	SET 7,A	
CBF8	SET 7,B	
CBF9	SET 7,C	
CBFA	SET 7,D	
CBFB	SET 7,E	
CBFC	SET 7,H	
CBFD	SET 7,L	
CB26	SLA (HL)	Shift Operand Left Arithmetic
DDCB0526	SLA (IX+d)	
FDCB0526	SLA (IY+d)	
CB27	SLA A	
CB20	SLA B	
CB21	SLA C	
CB22	SLA D	
CB23	SLA E	
CB24	SLA H	
CB25	SLA L	

OBJ CODE	SOURCE STATEMENT	OPERATION
CB2E	SRA (HL)	Shift Operand Right
DDCB052E	SRA (1X+d)	Arithmetic
FDCB052E	SRA (1Y+d)	
CB2F	SRA A	
CB28	SRA B	
CB29	SRA C	
CB2A	SRA D	
CB2B	SRA E	
CB2C	SRA H	
CB2D	SRA L	
CB3E	SRL (HL)	Shift Operand Right
DDCB053E	SRL (1X+d)	Logical
FDCB053E	SRL (1Y+d)	
CB3F	SRL A	
CB38	SRL B	
CB39	SRL C	
CB3A	SRL D	
CB3B	SRL E	
CB3C	SRL H	
CB3D	SRL L	
96	SUB (HL)	Subtract Operand
DD9605	SUB (1X+d)	from Acc.
FD9605	SUB (1Y+d)	
97	SUB A	
90	SUB B	
91	SUB C	
92	SUB D	
93	SUB E	
94	SUB H	
95	SUB L	
D620	SUB n	
AE	XOR (HL)	Exclusive "OR"
DDAE05	XOR (1X+d)	Operand and Acc.
FDAE05	XOR (1Y+d)	
AF	XOR A	
A8	XOR B	
A9	XOR C	
AA	XOR D	
AB	XOR E	
AC	XOR H	
AD	XOR L	
EE20	XOR n	

Example Values

nn EQU 584H
d EQU 5
n EQU 20H
e EQU 2EH

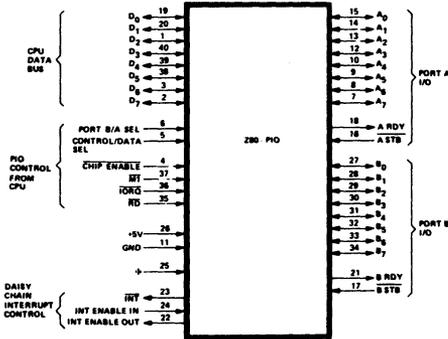
9.5 Z80 PIO - Pinbelegung



Z-80[®] PIO Z-80A PIO

COMPONENTS

Z-80 PIO Pin Description



- D₇-D₀ Z80-CPU Data Bus (bidirectional, tristate)
- B/A Sel Port B or A Select (input, active high)
- C/D Sel Control or Data Select (input, active high)
- CE Chip Enable (input, active low)
- Φ System Clock (input)

- M1 Machine Cycle One Signal from CPU (input, active low)
- IORQ Input/Output Request from Z80-CPU (input, active low)
- RD Read Cycle Status from the Z80-CPU (input, active low)
- IEI Interrupt Enable In (input, active high)
- IEO Interrupt Enable Out (output, active high). IEI and IEO form a daisy chain connection for priority interrupt control.
- INT Interrupt Request (output, open drain, active low)
- A₀-A₇ Port A Bus (bidirectional, tristate)
- A STB Port A Strobe Pulse from Peripheral Device (input, active low)
- A RDY Register A Ready (output, active high)
- B₀-B₇ Port B Bus (bidirectional, tristate)
- B STB Port B Strobe Pulse from Peripheral Device (input, active low)
- B RDY Register B Ready (output, active high)

9.6 Datenblatt 74 LS 138

54S/74S138 54LS/74LS138 1-OF-8 DECODER/DEMULTIPLEXER

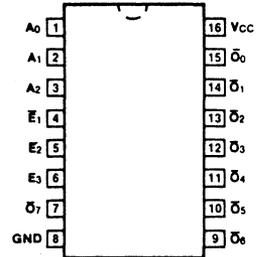
DESCRIPTION — The '138 is a high speed 1-of-8 decoder/demultiplexer. This device is ideally suited for high speed bipolar memory chip select address decoding. The multiple input enables allow parallel expansion to a 1-of-24 decoder using just three '138 devices or to a 1-of-32 decoder using four '138 devices and one inverter. The '138 is fabricated with the Schottky barrier diode process for high speed.

- SCHOTTKY PROCESS FOR HIGH SPEED
- DEMULTIPLEXING CAPABILITY
- MULTIPLE INPUT ENABLE FOR EASY EXPANSION
- ACTIVE LOW MUTUALLY EXCLUSIVE OUTPUTS

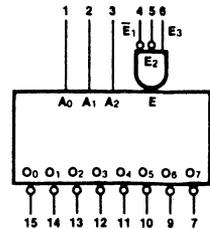
ORDERING CODE: See Section 9

PKGS	PIN OUT	COMMERCIAL GRADE	MILITARY GRADE	PKG TYPE
		V _{CC} = +5.0 V ±5%, T _A = 0°C to +70°C	V _{CC} = +5.0 V ±10%, T _A = -55°C to +125°C	
Plastic DIP (P)	A	74S138PC, 74LS138PC		9B
Ceramic DIP (D)	A	74S138DC, 74LS138DC	54S138DM, 54LS138DM	6B
Flatpak (F)	A	74S138FC, 74LS138FC	54S138FM, 54LS138FM	4L

**CONNECTION DIAGRAM
PINOUT A**



LOGIC SYMBOL



V_{CC} = Pin 16
GND = Pin 8

INPUT LOADING/FAN-OUT: See Section 3 for U.L. definitions

PIN NAMES	DESCRIPTION	54/74S (U.L.) HIGH/LOW	54/74LS (U.L.) HIGH/LOW
A ₀ — A ₂	Address Inputs	1.25/1.25	0.5/0.25
E ₁ , E ₂	Enable Inputs (Active LOW)	1.25/1.25	0.5/0.25
E ₃	Enable Input (Active HIGH)	1.25/1.25	0.5/0.25
O ₀ — O ₇	Outputs (Active LOW)	25/12.5	10/5.0 (2.5)

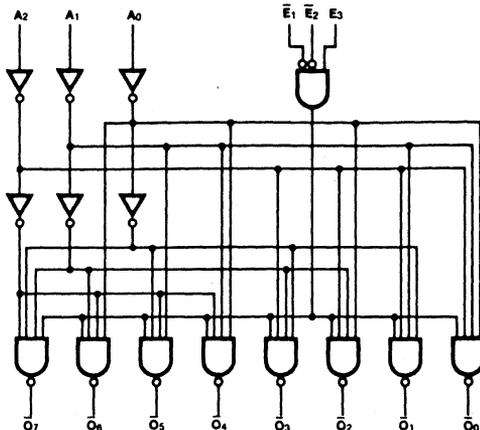
FUNCTIONAL DESCRIPTION — The '138 is a high speed 1-of-8 decoder/demultiplexer fabricated with the low power Schottky barrier diode process. The decoder accepts three binary weighted inputs (A_0 , A_1 , A_2) and when enabled provides eight mutually exclusive active LOW outputs (\bar{O}_0 — \bar{O}_7). The '138 features three Enable inputs, two active LOW (\bar{E}_1 , \bar{E}_2) and one active HIGH (E_3). All outputs will be HIGH unless \bar{E}_1 and \bar{E}_2 are LOW and E_3 is HIGH. This multiple enable function allows easy parallel expansion of the device to a 1-of-32 (5 lines to 32 lines) decoder with just four '138 devices and one inverter. (See Figure a.) The '138 can be used as an 8-output demultiplexer by using one of the active LOW Enable inputs as the data input and the other Enable inputs as strobes. The Enable inputs which are not used must be permanently tied to their appropriate active HIGH or active LOW state.

TRUTH TABLE

INPUTS						OUTPUTS							
\bar{E}_1	\bar{E}_2	E_3	A_0	A_1	A_2	\bar{O}_0	\bar{O}_1	\bar{O}_2	\bar{O}_3	\bar{O}_4	\bar{O}_5	\bar{O}_6	\bar{O}_7
H	X	X	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	L	X	X	X	H	H	H	H	H	H	H	H
L	L	H	L	L	L	L	H	H	H	H	H	H	H
L	L	H	H	L	L	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	L	H	H	H	H	H
L	L	H	H	H	L	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	L	H	H
L	L	H	L	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	L
L	L	H	H	H	H	H	H	H	H	H	H	H	L

H = HIGH Voltage Level
 L = LOW Voltage Level
 X = Immaterial

LOGIC DIAGRAM



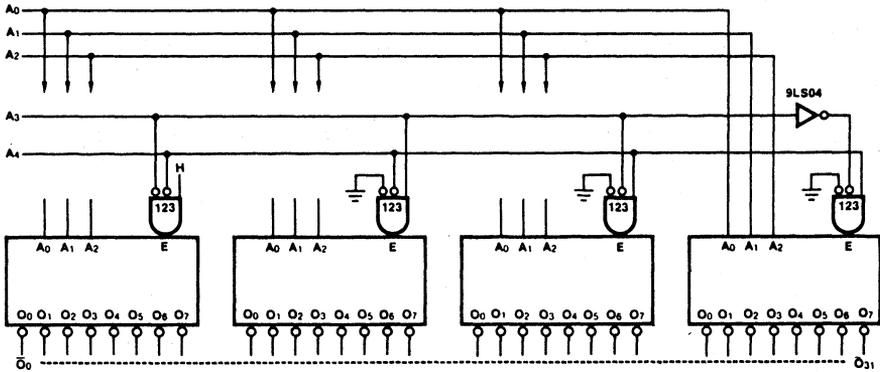


Fig. a

DC CHARACTERISTICS OVER OPERATING TEMPERATURE RANGE (unless otherwise specified)

SYMBOL	PARAMETER	54/74S		54/74LS		UNITS	CONDITIONS
		Min	Max	Min	Max		
I_{CC}	Power Supply Current	74		10		mA	$V_{CC} = \text{Max}$

AC CHARACTERISTICS: $V_{CC} = +5.0 \text{ V}$, $T_A = +25^\circ \text{C}$ (See Section 3 for waveforms and load configurations)

SYMBOL	PARAMETER	54/74S		54/74LS		UNITS	CONDITIONS
		$C_L = 15 \text{ pF}$ $R_L = 280 \Omega$		$C_L = 15 \text{ pF}$ $R_L = 280 \Omega$			
		Min	Max	Min	Max		
t_{PLH} t_{PHL}	Propagation Delay A_n to \bar{O}_n	12 12		18 27		ns	Figs. 3-1, 3-4, 3-5
t_{PLH} t_{PHL}	Propagation Delay \bar{E}_1 or \bar{E}_2 to \bar{O}_n	8.0 11		15 24		ns	Figs. 3-1, 3-5
t_{PLH} t_{PHL}	Propagation Delay E_3 to \bar{O}_n	11 11		18 28		ns	Figs. 3-1, 3-4

Programme und Tips für den ZX81-Anwender

9.7 Programme und Tips für den ZX81-Anwender

Diese Seite soll allen ZX81 Benutzern Anregungen, Tips für eigene Programme und für Anwendungen des kleinen Rechners geben. Das Wort "klein" bezieht sich hier auf die Abmessungen nicht aber auf die Möglichkeiten, die dieser Rechner bietet. Zu einem Preis, für den man nicht einmal einen Z80-Einplatinencomputer bekommt, bietet der Sinclair ZX81 8k Basic, Tastatur, Kassetteninterface, FS-Anschluß und Zugang zu den Daten- und Adressleitungen der CPU. Damit sind auch viele andere Anwendungsmöglichkeiten gegeben, wie zum Beispiel der Einsatz für Motorsteuerungen, Temperatur- und Füllstandmessungen, Zeitmessungen und vieles andere mehr.

1. Grafikprogramm MUSTER

Zum Anfang ein kleines Programm, das die Grafikzeichen mit dem Code 0 bis 10 (siehe Handbuch Seite 78) in einer quadratischen Anordnung, zufällig ausgewählt, auf den Bildschirm ausgibt. Die Zeichenausgabe beginnt in der Bildschirmmitte und wird von dort nach außen weitergezeichnet. In Abbildung 1 sind die Variablen für die Eckpunkte und die Ausgabe angegeben.

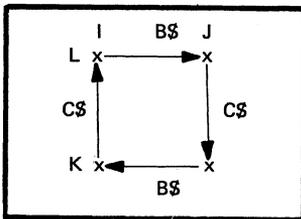


Abb. 1:

Variablenamen für Eckpunkt und Ausgabe

Das Programm ist in Abbildung 2 gezeigt. Durch Ändern der Zeilen 180 und 190 lassen sich

andere Muster erzeugen. Spielen Sie mit diesen Variablen und schauen Sie sich diese Ergebnisse auf den Bildschirm an.

```

110 LET I=10
120 LET J=10
130 LET K=10
140 LET L=10
150 LET B$="OCHR# (INT (RND*11))"
160 LET C$="OCHR# (INT (RND*11))"
170 NEXT I TO J
180 PRINT "AT M,N; B$"
190 NEXT J
200 LET I=J
210 LET J=J
220 PRINT "AT M,N; C$"
230 NEXT I TO L
240 LET I=L
250 LET J=L
260 PRINT "AT M,N; C$"
270 NEXT I TO I STEP -1
280 PRINT "AT M,N; B$"
290 NEXT J
300 LET I=L
310 LET J=L
320 PRINT "AT M,N; C$"
330 NEXT I TO K STEP -1
340 PRINT "AT M,N; C$"
350 LET I=H
360 LET J=I-1
370 LET K=J+1
380 LET L=K-1
390 IF K=L THEN STOP
400 LET L=L+1
410 GOTO 180

```

Abb. 2:

Grafikprogramm MUSTER

2. Ersatz für READ, DATA und RESTORE-Befehle

Die meisten BASIC-Interpreter bieten die Möglichkeit, Daten, die in DATA-Statements festgelegt sind, über die READ-Anweisung Variablen zuzuweisen (siehe auch Handbuch Seite 146).

Der ZX81 bietet hier eine viel bessere Möglichkeit. Beim Speichern des Programms auf Kassette wird nicht nur das Programm selbst, sondern auch alle Werte der Variablen beim augenblicklichen Programmzustand mit aufgezeichnet. Wird das Programm nach dem Laden von

Kassette nicht mit RUN, sondern mit GOTO gestartet, so bleiben diese Werte der Variablen erhalten.

Dazu ein Beispiel:

Die Vornamen von Familienmitgliedern sollen in einem Textfeld gespeichert werden. Zuerst die Lösung, wie sie mit anderen BASIC-Rechnern erlaubt ist, nicht aber beim ZX81.

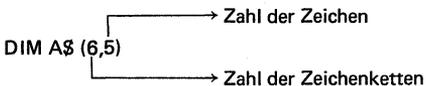
```
10 DIM A$(6)
20 FOR I = 1 TO 6
30 READ A$(I)
40 NEXT I
50 DATA HANS, INGE, KURT, ILSE, HEINZ, JUTTA
```

Zweite Möglichkeit, die sowohl bei anderen Rechnern, wie mit dem ZX81 funktioniert. Unterschiede gibt es nur bei der DIM-Anweisung.

Beispiel für ZX81

```
10 DIM A$(6,5)
20 A$(1) = "HANS"
30 A$(2) = "INGE"
40 A$(3) = "KURT"
50 A$(4) = "ILSE"
60 A$(5) = "HEINZ"
70 A$(6) = "JUTTA"
```

Die Zeile 10 hat bei den meisten anderen Basic-Rechnern nur eine einzige Angabe. DIM A\$(6). Dabei setzen diese Interpreter voraus, daß A\$ eine einzige Zeichenkette mit bis zu 255 Zeichen darstellt. A\$(6) ist dann ein Feld von 6 solchen Zeichenketten. Beim ZX81 ist dies anders. A\$ ist ebenfalls eine Zeichenkette. Die Angabe A\$(4) bedeutet aber das 4te Zeichen dieser Zeichenkette. Deshalb bei ZX81 2 Angaben.



In Worten ausgedrückt: 6 Zeichenketten mit jeweils 5 Zeichen. Frage: Welches Zeichen wird mit PRINT A\$(4,2) ausgedrückt, wenn A\$ unser obiges Beispiel ist ?

Dritte Lösung, nur mit ZX81 möglich:

```
10 DIM A$(6,5)
20 FOR I = 1 TO 6
30 INPUT A$(I)
40 NEXT I
```

Starten Sie dieses Programm und geben Sie die Namen ein. Danach setzen Sie das Programm mit

```
100 FOR I = 1 TO 6
110 PRINT A$(I)
120 NEXT I
```

fort. Mit GOTO 100 erhalten Sie einen Ausdruck der eingegebenen Namen.

Nun werden die Zeilen 10 bis 40 gelöscht und das übrig gebliebene Programm mit SAVE "DEMO" auf Kassette geschrieben. Vor dem Wiedereinlesen des Programms wird der Rechner kurz ausgeschaltet oder mit NEW das Programm gelöscht. Nun wird das Programm mit LOAD "DEMO" wieder eingelesen und *ja nicht mit RUN*, sondern mit GOTO 100 gestartet. Die eingegebenen Namen sind erhalten geblieben und werden ausgegeben. Nur RUN oder eine neue Dimensionierungsangabe für diese Zeichenkette löscht den Inhalt.

Das BASIC des ZX 81 nimmt es mit der Interpretation der Sprache manchmal genauer, als andere BASIC-Dialekte. Dies ist auch bei Vergleichen von Zeichenketten der Fall. Betrachten wir folgenden Fall. Wir haben für uns für die Einträge in ein Computernotizbuch durch

```
DIM A$(50,20)
```

Platz reserviert für 50 Einträge zu je 20 Zeichen. (Dies sind 1000 Zeichen = 1000 Byte, also nur mit dem 16K Modul möglich. Diejenigen, die das folgende ohne dieses Modul ausprobieren möchten, geben nur DIM A\$(5,20) ein.

Als Beispiel wollen wir in unser Notizbuch nur Namen eintragen, also

```
A$(1) = "HANS"
A$(2) = "GEORG"
A$(3) = "HILDE"
```

Dies mag vorerst genügen. Wenn wir nun in dieser Liste einen Namen suchen wollen, so könnte dies durch folgendes Programm geschehen.

```
10 PRINT "NAME";
20 INPUT B$
30 LET N=3 (Zahl der Einträge)
40 FOR I = 1 TO N
50 IF A$(I) = B$ THEN 100
60 NEXT I
70 PRINT "NICHT GEFUNDEN"
```

```

80 STOP
100 PRINT B$;"IST DER ";I;"TE EINTRAG"

```

Wenn wir nun für B\$ = "GEORG" eingeben, so müßte das Programm diesen Namen finden. Das tut es aber nicht. Warum?

Wir haben unsere Einträge mit DIM (50,20) angegeben. Alle Einträge sind 20 Zeichen lang, auch wenn der eigentliche Eintrag kürzer ist. Der eingegebene Name ist aber nur 5 Zeichen lang. Deshalb wird keine Übereinstimmung gefunden. Man kann dies leicht feststellen, wenn man nach dem Durchlauf des Programms PRINT LEN A\$(2), LEN B\$ eingibt.

Man kann dies beheben, indem man B\$ auch dimensioniert. Man fügt also ein:

```

5 DIM B$(1,20)

```

und ersetzt in den Zeilen 20, 50 und 100 für B\$ gleich B\$(1) ein. Dann funktioniert das Aufsuchen.

In anderen BASIC-Dialekten gibt es die folgenden Anweisungen für Zeichenketten: LEFT\$(A\$, N) und RIGHT\$(A\$, N). Mit der ersten Anweisung wurden die ersten N-Zeichen, beginnend vom linken Rand der Zeichenkette, mit der zweiten die N am weitesten rechts stehenden Zeichen.

Das ZX-81 BASIC kennt diese Befehle nicht. Man kann sie aber leicht programmieren.

```

PRINT A$(1 TO 5)

```

druckt die ersten 5 Zeichen aus und PRINT A\$(LEN A\$-5 TO LEN A\$) die 5 am weitesten rechts stehenden Zeichen.

Dazu ein kleines Programm:

```

10 LET A$ = "ELCOMP"
20 LET L = LEN A$
30 FOR I = 1 TO L
40 PRINT A$(1 TO I)
50 NEXT I

```

Nun noch ein Programm für Schüler. Es bestimmt den größten, gemeinsamen Teiler (GGT) zweier Zahlen'

```

100 PRINT "GROESSTER GEMEINSAMER
TEILER"
110 PRINT "A = ";
120 INPUT A
130 PRINT A
140 PRINT "B = ";
150 INPUT B
160 PRINT B
170 LET Q = INT (A/B)
180 LET R = A-Q*B
190 LET A = B
200 LET B = R
210 IF R > 0 THEN GOTO 170
220 PRINT "GGT = "; A

```

9.8 Literaturhinweis

- The ZX81 Pocket Book, T. Toms, Phillips Associates, Epsom
- BASIC für Fortgeschrittene, C. Lorenz, Hofacker Verlag
- BASIC Programmierhandbuch, C. Lorenz, Hofacker Verlag
- Programmieren in Maschinensprache (Z80), C. Lorenz, Hofacker Verlag
- Z80 Assembler Handbuch, Hofacker Verlag
- Zilog Z-80 CPU, Programming Reference Card

NOTIZEN

NOTIZEN

Hofacker-Bücher

Deutsch

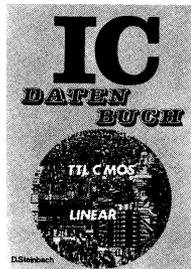


TBB-Transistor-, Berechnungs- und Bauleitungs-HB, Band 1, Hofacker
Völlig neu überarbeitete Auflage. Das Buch soll bei der täglichen Arbeit im Labor, in der Werkstatt oder am Elektronik-Hobbytisch Ihnen ein guter Begleiter sein. Berechnungsgrundlagen, Berechnungsbeispiele, Tabellen, Vergleichslisten, Digitaltechnik, Netzgeräte, BASIC-Programme zur Berechnung spezieller Schaltungen usw. sind in übersichtlicher Form dargestellt. Ca. 300 Seiten.

Best.-Nr. 1 **29,80 DM**

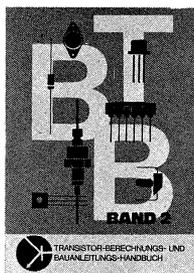


Electronic im Auto, H. Gebauer
Mit Handbuch für Polizeiradar. Ein Buch für jeden technisch interessierten Autofahrer. Es zeigt Ihnen die vielen Möglichkeiten zur Verbesserung von Sicherheit, Leistung und Fahrkomfort in Ihrem Auto. Thyristorzündung, Beschleunigungsmesser, Drehzahlmesser, Batterieladegerät, Alarmanlagen u. v. a.
Best.-Nr. 3 **9,80 DM**



IC-Datenbuch, D. Steinbach
Daten- und Auswahllisten der gebräuchlichsten integrierten Schaltkreise. Digital und analog. Gerade bei ICs ist es wichtig die Anschlussfolgen genau zu kennen. Die wichtigsten TTL-Schaltkreise, NF-Verstärker, C-Mos Serie, lineare Schaltungen wie Operationsverstärker, Komparatoren, Spannungsregler, Trigger-Schaltungen, u. v. a. Das IC-Datenbuch wird auch Ihnen ein unentbehrlicher Begleiter bei allen Arbeiten mit integrierten Schaltungen sein.

Best.-Nr. 5 **9,80 DM**



TBB-Handbuch, Band 2, Hofacker
Dieses Buch ist die Fortsetzung des erfolgreichen Handbuches, Band 1. Ein Buch, das sich in der Hand des Praktikers bestens bewährt hat. Weitere neueste Schaltbeispiele und Berechnungsgrundl., Experimentier- und Versuchsbeschreibungen. Integrierte Spannungsregler, Wärmeableitung, Operationsverstärker Einführung, RC-Zeitglieder, Transistortester u. v. a.

Best.-Nr. 2 **19,80 DM**



IC-Handbuch, C. Lorenz
Ein Handbuch für digitale und lineare integrierte Schaltungen. Daten- und Auswahl-, Vergleichslisten, Gehäuseformen, Grundlagen, viele Schaltbeispiele, Printvorlagen, u. v. a. Alles über TTL-Technik, C MOS, MOS-Schaltungen, integrierte NF-Verstärker u. v. a.
Best.-Nr. 4 **19,80 DM**



IC-Schaltungen, D. Steinbach
Hier finden Sie eine gelungene Zusammenstellung der wichtigsten Anwendungsbeispiele aus dem Bereich der integrierten Schaltungen. TTL, C MOS, Linear. Alle Schaltungen sind übersichtlich und klar dargestellt und mit einer kurzen, jedoch sehr genauen Beschreibung versehen. Tastenprellung, Zähler, Impulsgeber, Codierer, Dekodierer, Datenübertragung, Serien-Parallelwandler, Digitalvoltmeter u. v. a.
Best.-Nr. 6 **19,80 DM**



Elektronik Schaltungen, Hofacker
Die ideale Schaltungssammlung zum Basteln u. Experimentieren, Schaltungen mit Operationsverstärkern, Spannungsreglern, TTL, C-MOS-Schaltkreisen. MOS Uhr mit Wecker-elektronischer Würfel u. v. a.
Best.-Nr. 7 9,80 DM



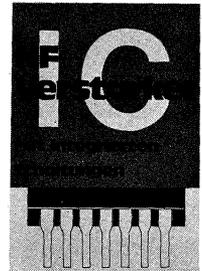
IC Bauanleitungen Handbuch -IC-KIT, C. Lorenz
Ein Bauanleitungsbuch mit vielen hochinteressanten Bauanleitungen aus dem Bereich der LSI Schaltungstechnik. Schaltbeispiele mit Printvorlagen zum Selbsterstellen der Leiterplatten mit genauesten Beschreibungen. Hochaktuell und brandneu: Funktionsgenerator XR 2206, MOS-Uhr mit Wecker, programmierbarem Wecktongenerator, Schlummerautomatik, IC-Netzteil, Experimentieranleitung und Grundkurs über Flip Flops, u. v. a. Zu allen Schaltungen finden Sie Platinenvorlagen oder Sie können die Experimentierschaltungen auf der Experimentierplatine WH-1 g durchführen. Über 100 Seiten.
Best.-Nr. 8 19,80 DM



Feldeffekttransistoren, C. Lorenz
Der Feldeffekttransistor (FET) gehört heute zu den interessantesten Bauteilen überhaupt. Wie man damit experimentiert, wie man seine Funktion versteht und wie man damit brauchbare u. hochinteressante Schaltungen aufbauen kann, zeigt Ihnen dieses Buch. Grundlagen, Kennlinienfelder, Tabellen, Rechenbeispiele, Anschlußbilder und eine Vergleichsliste für Feldeffekttransistoren bilden den Kern dieser umfangreichen Darstellung. Alles in allem finden Sie hier eine praxisnahe und komplette Arbeitsunterlage, mit der Sie im Beruf und auch im Hobby erfolgreich arbeiten können.
Best.-Nr. 9 9,80 DM



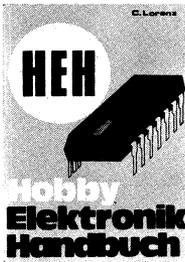
Elektronik und Radio, C. Lorenz
4. Auflage. Völlig neu bearbeitet und stark erweitert. Eine Schritt für Schritt Einführung in die Radiotechnik mit vielen Bildern. Vom einfachen Diodenempfänger (Detektor) bis zu interessanten Sender- und Empfängerschaltungen (Minispione). IC-Radio, IC-Sender, Antennen, Berechnungsgrundlagen, Tabellen u. v. a. Über 150 Seiten.
Best.-Nr. 10 19,80 DM



NF-Verstärker, C. Lorenz
Grundlagen der integrierten NF-Verstärker, Berechnung von kompletten IC-NF-Verstärkerstufen. Anwendungsbeispiele mit den interessantesten und gebräuchlichsten Standard IC-NF-Verstärkern wie TBA 800, TBA 830, usw. Printvorlagen, Auswahltabellen, Experimentieranleitungen und Anschlußbilder machen dieses Buch zu einem unentbehrlichen Begleiter für alle, die sich m. NF-Verstärkern beschäftigen wollen.
Best.-Nr. 11 9,80 DM



BIS, Beispiele integrierter Schaltungen, H. Bernstein
Auf über 130 Seiten Anwendungsbeispiele mit integrierten Schaltkreisen, Zeitgeber 555, Funktionsgenerator ICL 8038, Opto Elektronik, Operationsverstärker, Festwertspeicher (ROM), u. v. a.
Best.-Nr. 12 19,80 DM



HEH, Hobby Elektronik Handbuch
C. Lorenz
 Das Schaltungsbuch f. jeden Hobbyelektroniker, Schaltbeispiele und Bauanleitungen aus dem gesamten Hobbybereich. Lichtorgeln, Alarmanlagen, Eiswarngerät fürs Auto, PLL-Schaltungen u. v. a.
Best.-Nr. 13 9,80 DM



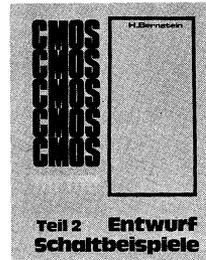
IC-Vergleichsliste, C. Lorenz
 Neben integrierten Schaltungen sind auch noch Transistoren, Dioden, optoelektronische Bauteile, Sicherungen und Röhren in dieses praktische HB aufgenommen worden. Das ideale Nachschlagewerk, wenn es dann gilt das richtige Bauelement für die gewünschte Anwendung zu finden. Dieses Buch wird Ihnen in der Praxis viele Dienste erweisen können. Viele Auswahl- und Vergleichslisten mit Datenangaben. Ca. 800 Seiten.
Best.-Nr. 14 29,80 DM



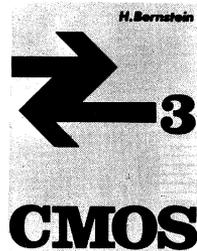
Opto-Handbuch, C. Lorenz
 Das Handbuch für die gesamte Optoelektronik. Eine Einführung und ein ideales Nachschlagewerk. Grundlagen, Definitionen aller Kenngrößen, Opto-Lexikon, Berechnungsgrundlagen, Lichtsender, Lichtempfänger, Anzeigen, Infrarot Detektoren, Optokoppler, Opto-Vergleichsliste, u. v. a. 106 Seiten.
Best.-Nr. 15 19,80 DM



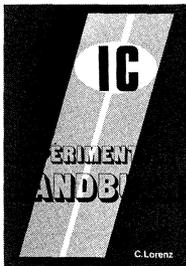
C MOS Einführung, Entwurf, Schaltbeispiele, Teil 1 H. Bernstein
 Vom C MOS Gatterbaustein über Schieberegister und Zähler bis hin zum C MOS Schreib- Lesespeicher. Insgesamt werden neunzehn interessante und bekannte C MOS Schaltkreise beschrieben. Zu jedem Bauelement sind genaue Daten, Schaltbild und Anwendungsbeispiele angegeben. Im großen Applikationsteil finden Sie: C MOS-Kippstufen, Addierwerke u. Rechenschaltungen, Digital Analog Wandler, Schieberegister für analoge Spannungen, Multiplexsysteme f. analoge Signale u. v. a. Eine komplette Einführung u. gut geeignet für das Selbststudium der C MOS Technik. 140 Seiten.
Best.-Nr. 16 19,80 DM



C MOS Entwurf u. Schaltbeispiele, Teil 2, H. Bernstein
 Fortsetzung von Best.-Nr. 16. Anwendungsbeispiele mit genauen Schaltungsbeschreibungen und Bauelementunterlagen. Daten, Anschlußbelegungen weiterer wichtiger hochintegrierter C MOS Elemente. Ein komplettes Arbeits- u. Experimentierbuch. C-MOS Uhrenschaltungen, Schieberegisterschaltungen, Parallel-Serien Umsetzung, statische u. dynamische Speicherschaltungen, Zählschaltungen, Digital Analog-Wandler, Analog Digital Wandler. Digital Voltmeter, I/O Registerschaltungen. RAM und ROM Anwendungen. Über 140 Seiten.
Best.-Nr. 17 19,80 DM



C MOS Entwurf u. Schaltbeispiele, Teil 3, H. Bernstein
 Fortsetzung von Best.-Nr. 17. Eine sehr umfangreiche Applikationsammlung mit hochintegrierten C MOS Elementen. Speicher- und Steuerschaltungen, Multiplex- und Datenbussysteme, Liquid Cristal Anzeigen, Uhrenschaltungen, PLL-Schaltungen, Optoelektronik in Verbindung mit C MOS. Aufbau und Wirkungsweise der Prozeß-rechentechnik, Arithmetische Logische Einheiten (ALU) u. andere wichtige Funktionen aus der Prozeß-rechentechnik. RAMs, ROMs, und FIFO-Speicherschaltungen.
Best.-Nr. 18 19,80 DM

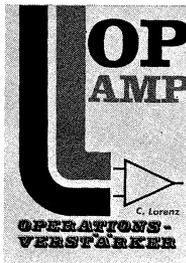


IC Experimentier Handbuch -IC, -EX, C. Lorenz

Eine sehr umfangreiche Schaltungs- und Bauanleitungssammlung mit neuen, jedoch meist beim Fachhandel erhältliche Standard ICs. Rechnerschaltungen, Mikroprozessoren, I/O-Schaltungen, Stoppuhren, druckende und anzeigende Rechner, Digitalvoltmeter, Hilfschaltungen für den Elektronik Experimentierer, A/D-Wandler, Frequenzzähler u. v. a. Viele Schaltungen können auf der IC KIT Experimentierplatine WH-1g aufgebaut werden.

Best.-Nr. 19

19,80 DM

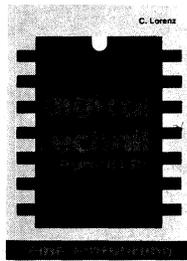


Operationsverstärker, C. Lorenz

Dieses Buch umfaßt das gesamte Gebiet der linearen Schaltungstechnik und stellt ein in dieser Preislage bisher noch nie dagewesenes Nachschlagwerk und Einführungshandbuch dar. Bestens geeignet für das Selbststudium. Nach einer pädagogisch geschickt gemachten Einführung folgen theoretische Arbeitsunterlagen und die zugehörigen Schaltbeispiele mit Daten und Gehäuseanschlüssen. Dieses wertvolle Buch dürfte seinen Platz auch bei Ihren Arbeitsunterlagen finden, und wird dann immer von Nutzen sein, wenn es um die Lösung von nicht routinemäßigen Aufgaben geht. Über 150 Seiten.

Best.-Nr. 20

19,80 DM

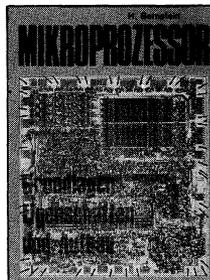


Digitaltechnik Grundkurs, C. Lorenz

Ein Einführungskurs in die Digitaltechnik für Anfänger und Fortgeschrittene. Ein Fachbuch für den programmierten Selbstunterricht. Der ideale Kurzlehrgang für das Selbststudium. Der Kurs vermittelt Ihnen alle wichtigen Grundkenntnisse vom TTL-Gatter bis zum Mikroprozessor und Lösung von Schaltungsaufgaben durch Software. Viele Versuchsaufbauten u. Experimente aus diesem Kurs können auf der IC-KIT Platine WH-1g durchgeführt werden. Grundlagen, Gatter, Zähler, programmierbare Zähler, IC-Tester, Schieberegister, Speicher, Mikroprozessoren u. v. a.

Best.-Nr. 21

19,80 DM



Mikroprozessoren, Eigenschaften u. Aufbau, Teil 1, H. Bernstein

Grundlagen, Eigenschaften u. Aufbau von Mikroprozessoren. Organisation von Recheneinheiten und Mikroprogr. Programmierung und Klassifizierung v. Mikroprozessoren. Ablaufdiagramm, Flußdiagramm. Ein Cip-Technik und Multi Chip-Technik, Transfer- und Sprungfunktionen. Speichertechnik: RAMs ROMs, FIFO, FILO. Programmierbare logische Arrays (PLA). Anwendungsbeispiele u. Anwendungsbereiche. Über 120 Seiten.

Best.-Nr. 22

19,80 DM

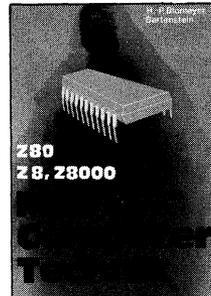


Elektronik Grundkurs, C. Lorenz

Eine leichtverständliche und pädagogisch geschickt gemachte Einführung in die Technik der elektronischen Schaltungen. Ein Kurzlehrgang und Schnellkurs zugleich. Aber auch ein recht brauchbares Nachschlagwerk für den fortgeschrittenen Elektroniker. Mit wenig Mühe können Sie sich hier die Grundkenntnisse der elektronischen Schaltungspraxis aneignen. Das Buch schafft die Voraussetzungen für ein erfolgreiches und sicheres Arbeiten mit interessanten Schaltkreisen modernster Technologien. Unentbehrlich f. das Experimentieren mit den heutigen modernen hochintegrierten Schaltkreisen.

Best.-Nr. 23

9,80 DM



Mikrocomputer Technik, Hans Peter Blomeyer-Bartenstein

Völlig neue Auflage: Herbst 1979. In diesem Buche finden Sie eine umfassende, einführende u. weiterführende Hilfe zum Einstieg in die Mikrocomputertechnik mit vielen Schalt- und Programmierbeispielen. Als praktische Betrachtungsgrundlage dient das supermoderne Mikrocomputerkonzept Z80A von ZILOG. Das Buch geht auf alle wichtigen Zusammenhänge ein und erklärt diese dem Leser so ausführlich, daß kaum noch Fragen offen bleiben. über 240 Seiten.

Best.-Nr. 24

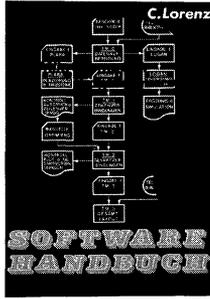
29,80 DM



Hobby Computer Handbuch, C. Lorenz

Eine leicht verständliche Einführung in die Mikrocomputertechnik. Diese sehr umfangreiche Einführung in die Microcomputertechnik dürfte zu diesem Preis einmalig sein. Auf über 450 Seiten finden Sie – Grundlagen der Computer- und Microcomputer-Technik, was ist ein Microcomputer? Microcomputer KITS, Einplatinencomputer, CAT, OSI, POLY 88 u. v. a. Das ideale allumfassende Buch für den Microcomputertechniker. Für Industrieanwendung ebenso geeignet wie für den Hobby-Computer-Fan.

Best.-Nr. 25 29,80 DM



Mikroprozessor Software Handbuch C. Lorenz

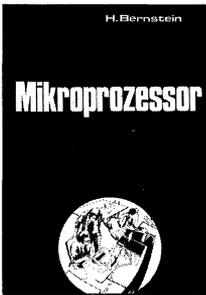
Grundlagen und Einführung in die Mikrocomputerprogrammierung und Programmiersprachen (BASIC, FORTRAN, Assembler-Sprachen). Zusammenstellung der wichtigsten Befehlslisten: 8080, Z80, M6800, 6502 etc. Ein Software Handbuch für jeden, der mit Mikroprozessoren oder Mikrocomputern zu tun hat.

Best.-Nr. 27 29,80 DM

ohne Abbildung
Erscheint ca. Ende 1981

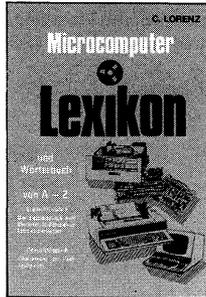
Mikrocomputer Datenbuch, Lorenz
Eine übersichtliche u. sehr informative Zusammenstellung der wichtigsten Mikroprozessorbausteine auf dem Markt. 8080A, 8085, 8048, Z80, Z8, 6500, 6800, 2650, 1802, F8, 3870, 6809, PACE u. v. a. Daten, Anschlußbilder, wichtige technische und elektrische Daten, Architektur, grundlegende Eigenschaften. Zu jedem Mikroprozessor werden dann auch noch die peripheren Bausteine sowie RAM und ROM Elemente behandelt. Das ideale Handbuch für jeden modernen Elektroniker. Ideal für den Computer-Servic. (Englisch)

Best.-Nr. 29 49,80 DM



Mikroprozessor, Teil 2, H. Bernstein
Die Fortsetzung von Best.-Nr. 22. Technologie von Mikroprozessor und Speicherbausteinen. Festwertspeicher, PROM, EPROM, FIFO, Schieberegister, MPR-, ARL- und SAR-Register. Aufbau eines Mikroprozessorsystems mit 8080, RAM- und ROM-Schnittstellen. Befehlsatz 8080. Über 120 Seiten.

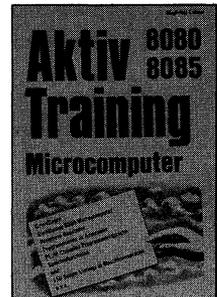
Best.-Nr. 26 19,80 DM



Microcomputer Lexikon u. Wörterbuch von A – Z, C. Lorenz

Einglich/Deutsch – Der Fachausdruck wird übersetzt, ausführlich erklärt und erläutert. Deutsch/Englisch – Übersetzung des Fachausdrucks. Ein Hilfs- und Arbeitsbuch für jeden, der sich heute mit der modernsten Elektronik beschäftigt. Viele engl. Ausdrücke werden heute in der Elektronik, Computer- und Mikroprozessortechnik verwendet und oft fehlt uns eine genaueste und präzise Erläuterung. Ein Lexikon und Wörterbuch in einem einzigen Buch vereint.

Best.-Nr. 28 29,80 DM



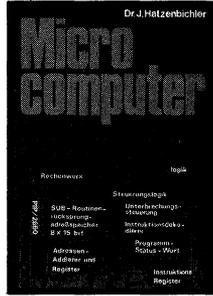
Aktiv Training Microcomputer 8080 8085, S. Lehrer

Dieses Werk mit über 360 Seiten beschäftigt sich ausschließlich mit den Microcomputerbausteinen und Peripherieelementen der 8080A und 8085 Mikroprozessoren. Grundlagen Einführung 8080 Microcomputer, Programmieren in Assembler, PL/M Cimpiler, Speicherbausteine u. v. a. Ideal für jeden, der ein System mit 8080, 8085 CPU besetzt. Auch der Z-80 Systembesitzer kann von diesem Buch viel profitieren. Am Schluß des Buches finden Sie noch ca. 150 Programm listings in Maschinensprache (Nützliche Utilities und Spielprogramme).

Best.-Nr. 30 49,80 DM



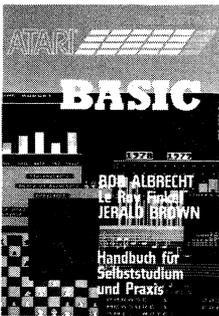
57 Programme in BASIC, Lorenz
 Ein Buch mit techn.-wissenschaftlichen Programmen u. einer großen Anzahl von Spielprogrammen in BASIC (Games). Ein Buch für jeden, der sich mit dem faszinierenden Hobby der Mikrocomputertechnik befassen will. Alle Listings sind in BASIC und können auf den meisten Personal Computer Systemen gefahren werden.
 Best.-Nr. 31 39,00 DM



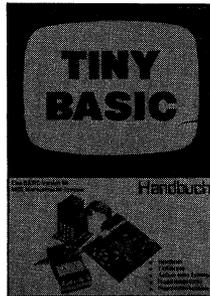
Mikrocomputer Programmierbeispiele für 2650, Dr. J. Hatzenbichler
 Eine Einführung in die Programmierung von Mikrocomputern anhand des Prozessors 2650 von Signetics. Viele Programmierbeispiele in Maschinensprache, die Sie auf einem preiswerten Mikroprozessorsystem MIKIT 2650-P2 ausführen können. Zeitschleifenprogr., Blinkschaltung, Lauflicht, Stufenzähler, Stopuhr, Reaktionszeittester, u. a. Zu diesem Buch ist auch ein komplett aufgebautes und getestetes Mikrocomputersystem erhältlich, auf dem Sie alle beschriebenen Programme selbst ausführen können. Über 120 Seiten. (Nur solange Vorrat reicht, wird nicht mehr nachgedruckt.)
 Best.-Nr. 33 19,80 DM



Der freundliche Computer – Was können Sie mit einem Personal Computer anfangen? T. Munnecke
 Das Buch soll Ihnen auf die im Titel gestellte Frage eine ausführliche Antwort geben. Es eignet sich für alle, die bisher viel über Mikros gehört haben und gerne ausführlicher Bescheid wissen möchten. Viele interessante Fakten – Welche Computersprache? Welche Anwendungen? Welches Gerät soll ich mir kaufen? 153 Seiten.
 Best.-Nr. 35 29,80 DM



ATARI BASIC Handbuch
 Das komplette Einführungs- und Programmierhandbuch für die neuen ATARI-Computer ATARI 400/800. Anhand von vielen Beispielen wird die Leistungsfähigkeit des ATARI-Computer-Systems gezeigt. Auch für den Anfänger ideal als Einstieg gedacht. Das Buch kann auch als Kurs (Lehrgang) für die allg. BASIC-Sprache verwendet werden.
 Best.-Nr. 32 29,80 DM



TINY BASIC Handbuch, Hermann
 Das erste deutschsprachige Handbuch über Tom Pittman's TINY BASIC. Eine Einführung in die TINY BASIC-Programmiersprache. Wie kann ich meinen Computer (KIM-1) erweitern und BASIC programmieren. Systemvorschläge. Viele Programmierbeispiele, Tricks und Kniffe.
 Best.-Nr. 34 19,80 DM

ohne Abbildung
Microcomputer und Roboter
 Ein Buch für diejenigen, die sich externe Schaltungen für Microcomputer bauen möchte, die roboterartige Funktionen ausführen können. Spracherkennung, Analog-/Digital-Wandler, Digital-/Analog-Wandler, Ultraschallsensoren, Lichtschranken, Tonerzeugung u. v. a. Erscheint Anfang 1982.
 Best.-Nr. 36 29,80 DM

ohne Abbildung
Oszillographen Handbuch
 Ein Buch für jeden, der seinen Oszillographen optimal nutzen will. Der Anfänger, der noch nie einen Oszillographen benutzt hat, wird auf einfache Weise mit der Technik und Handhabung vertraut gemacht. Auch die Anwendung in Zusammenhang mit den modernen Mikrocomputersystemen wird beschrieben. Oszillograph als alphanumeres Darstellgerät u. v. mehr. Erscheint ca. Anfang 1982.
 Best.-Nr. 103 19,80 DM

ohne Abbildung

1000 Elektronik Schaltungen

Ein ideales Handbuch für jeden, der öfter eine Schaltung zur Lösung eines bestimmten Problems sucht. Tausend Schaltungen aus fast allen Bereichen der Elektronik. Industrielle Steuerschaltungen, Microcomputer, Peripherie, Hobby-Elektronik-Schaltungen u. v. a. mehr. Ein Buch, daß bei keinem Elektroniker fehlen sollte. Erscheint ca. Mitte 1982.

Best.-Nr. 104 49,00 DM

ohne Abbildung

Praktische Antennentechnik

Ein Buch für jeden Funkamateuer oder Hobbyfunker. Grundlagen, Einführung, praktische Beispiele, Berechnungsgrundlagen u. v. a. Erscheint ca. Anfang 1982.

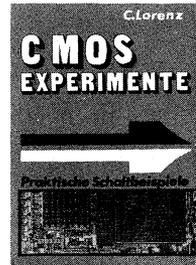
Best.-Nr. 107 19,80 DM



TTL-Experimentierbuch

Eine kleine Einführung in die Digitaltechnik. Grundlagen der Digitaltechnik kurz erklärt. Bauanleitung für einen praktischen Logik-Tester. Viele Experimente u. Anwendungsschaltungen mit dem TTL Gatterbaustein 7400. Das ideale Einführungsbuch in die Digitaltechnik.

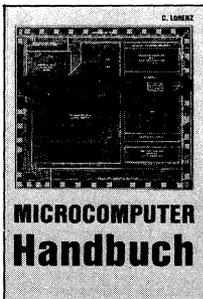
Best.-Nr. 105 1 5,00 DM



CMOS-Experimentierbuch

Eine kleine Einführung in die CMOS Schaltungstechnik. CMOS Grundlagen, Behandlungshinweise für CMOS Bausteine, Zusammenst. der wichtigsten CMOS Bausteine und deren Anschlußbilder. Viele praktische CMOS Schaltbeispiele. Ideal für jeden Elektroniker.

Best.-Nr. 106 1 5,00 DM



Handbuch für SC/MP

Ein echtes Handbuch für SC/MP (INS8060) Microcomputerbesitzer und solche die es werden wollen. Komplette Einführung mit vielen Schaltbeispielen, Schaltbildern und Programmlistings in Maschinensprache und TINY BASIC. Komplettes ELBUG-Listing, CPU-Karte, RAM-Karte, ROM-Karte, ROM-Programmierer, Cassetten-Interface, Fernsehinterface, Netzteil, Hex Ein-/Ausgabe, SC/MP Einkartenmicrocomputer u. v. a. 330 Seiten.

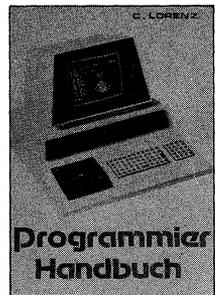
Best.-Nr. 108 29,80 DM



6502 Microcomputer Programmierung, P. Heuer

Eine deutschsprachige Einführung in die Maschinensprachenprogrammierung anhand des 6502 Microcomputers. Ein echtes Anleitungsbuch zum Einstieg in die Microcomputertechnik mit Hilfe des KIM-1. Viele Programmierbeispiele, die von einem Pädagogen speziell für Anfänger entwickelt wurden. Auch PET, AIM, SYM und ATARI-Besitzer brauchen dieses Buch.

Best.-Nr. 109 29,80 DM



Programmierhandbuch für PET

Ein für den PET produziertes Buch. Es beginnt da, wo Ihr mitgeliefertes Handbuch aufhört. Einführung Maschinensprache, Assembler, Ein-/Ausgabeprogrammierung, Programmiertricks, Analog / Digital-Wandler, Graphik, Spracherkennung u. v. a. mehr. Viele Listings, die Sie selbst eintippen können. Viele Informationen eignen sich auch für VC-20-Besitzer. 324 Seiten.

Best.-Nr. 110 29,80 DM



Programmieren mit TRS-80, Stübs
 Das erste in einem deutschen Verlag produzierte Buch über den erfolgreichen Personal Computer von TANDY. Ein Buch für jeden, der einen TRS-80 bereits besitzt oder vor der Entscheidung steht, welchen Computer er sich anschaffen soll. Einführung, Programmiertricks, Erweiterungen, Maschinenprogrammierung und viele Programme (Listing mit Beschreibung). 202 Seiten.
 Best.-Nr. 111 29,80 DM



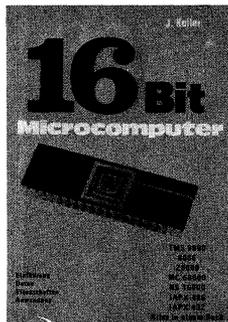
BASIC Programmierhandbuch
 Einführung und Nachschlagewerk. Speziell für die BASIC-Versionen der modernen Microcomputersysteme. Jeder Befehl wird ausführlich beschrieben und ein Beispielprogramm gezeigt. Sehr übersichtlich und praktisch. Am Schluß finden Sie ein komplettes BASIC-Programm, das Ihnen über einen Computer BASIC lehrt.
 Best.-Nr. 113 19,80 DM



Der Microcomputer im Kleinbetrieb
 Das Buch für jeden Geschäftsmann. Auf über 170 Seiten erfahren Sie, was Sie als Gewerbetreibender oder freiberuflich Tätige über Microcomputer und die Anwendung wissen sollten. Geschichtlicher Hintergrund Geräteauswahl, Beispiele aus der Praxis, Programmbeispiele wie z. B. Textverarbeitung, Reisebüro, Ladenkasse, Adressverwaltung, u. v. a. Betriebswirtschaftliche Auswertung, Finanzbuchhaltung, Erfolgsanalyse mit dem Microcomputer, Liquiditätsrechnung, kurzfristige Erfolgsrechnung, Microcomputer für Freiberufler, Grundlagen der Finanzbuchhaltung für Microcomputeranwender. Dieses Buch kann Ihnen als Geschäftsmann für die Zukunft tausende einsparen.
 Best.-Nr. 114 39,80 DM



PASCAL Handbuch, E. Flögel
 Von BASIC zu PASCAL. Ein Einführungs- Lehr und Arbeitsbuch für jeden der sich mit PASCAL beschäftigen will oder muß. Viele Programmbeispiele, viele Tricks wie PEEK und POKE, Einbinden von Maschinenprogrammen u. v. a.
 Best.-Nr. 112 29,80 DM



16 Bit Microcomputer, J. Koller
 Einführung, Daten, Eigenschaften, Anwendungen. Dieses Werk ist eine echte Sensation! Alle 16 Bit Prozessoren werden beschrieben und erläutert. Applikationsbeispiele, Programmierhinweise. TMS 9900, 8086, Z8000, MC 68000, NS 16000, IAPX 486, IAPX 432. Über 370 Seiten.
 Best.-Nr. 116 29,80 DM

ohne Abbildung

FORTRAN für Heimcomputer
 Einführung in die FORTRAN-Programmiersprache mit vielen Beispielen. Grundsätzliches über die verschiedenen Microcomputersysteme, die bereits mit FORTRAN-Compiler lieferbar sind. Allgemeine Übersicht, Tips und Hinweise. Erscheint ca. Ende 1982.
 Best.-Nr. 117 19,80 DM



Programmieren in Maschinsprache mit 6502, E. Flügel, W. Hofacker
Das deutschsprachige Werk über 6502 Maschinenprogrammierung. Einführung, Grundlagen, Eigenschaften, Adressierungsarten, Befehlsarten. Wie entwickelt man ein 6502 Maschinenprogramm? Handassemblierung, viele Programmbeispiele mit genauen Angaben direkt zum Eingeben in den Apple II mit Adressangabe (keine blutleeren Beispiele ohne Adresse), Verwendung von Assemblern. AIM-Assembler, Disassembler, Relocator, 6522 VIA, 6520, Interrupt, Fehlersuche in Maschinenprogrammen, Maschinsprache, Programmiertricks. Spezielle Abschnitte für Maschinsprachenprogrammierung über PET, CBM 3000, CBM 4000 u. VC-20, ATARI 400/800, Apple II, AIM sowie Ohio Scientific Challenger. Dieses Buch sollte jeder 6502 Systemanwender besitzen. Ca. 240 Seiten.

Best.-Nr. 118 49,00 DM

Anwenderprogramme für TRS-80 von Martin Stübs

Ein Buch, voll mit interessanten Anwenderprogrammen für TRS-80 Level II 16K und Video Genie (teilweise Diskette u./od. Cassette). Hauptsächlich Programme für den Manager, Geschäftsmann, Klein- und Mittelbetrieb. Auch einige interessante Spiele sind enthalten. Terminkalender, Reservierungsprogramm für Omnibusunternehmen und Hotels, Textverarbeitung, usw.

Best.-Nr. 120 29,80 DM



BASIC für Fortgeschrittene

Endlich ein BASIC-Buch für den fortgeschrittenen Programmierer. Alle wichtigen Befehle aus der Stringmanipulation, Disk-Befehle, WAIT, INSTR, WHILE WHEND usw. werden an Beispielen besprochen. Alle Befehle sind übersichtlich wie in einem Nachschlagewerk mit großen Überschriften angeordnet. Dann folgt ein umfangreicher BASIC Kurs für Fortgeschrittene mit vielen Beispielen (Inventur, Rechnungen schreiben, Adressverwaltung usw.). Am Schluß finden Sie dann noch einen Vergleich der wichtigsten Sortiermethoden sowie ein Programm zur Vorhersage von Ereignissen.

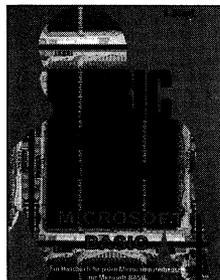
Best.-Nr. 122 39,00 DM



Programmieren in Maschinsprache (Z80), C. Lorenz

Eine sehr ausführliche Einführung in die Z80 Maschinsprache mit vielen Beispielen. Die Beispiele können mit Hilfe des TRS-80 Level II sowie dem T-BUG von TANDY und den T-BUG-Erweiterungen (IN LOCO, T-STEP, T-LEGS) ausgeführt werden. Ein unentbehrliches Buch für jeden, dem die BASIC-Programmiersprache von der Geschwindigkeit her zur Lösung seiner Aufgaben nicht mehr ausreicht.

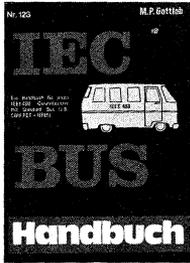
Best.-Nr. 119 49,00 DM



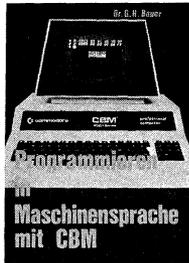
Microsoft BASIC-Handbuch

Die deutsche Übersetzung des erfolgreichen Microsoft BASIC-Handbooks. Leicht verständliche Einführung mit vielen interessanten Programmbeispielen. Das kompetente Werk von Microsoft selbst. Ideal als Zusatzliteratur zu jedem BASIC-Buch.

Best.-Nr. 121 29,80 DM



IEC Bus-Handbuch, M. P. Gottlob
 Ein Handbuch und Nachschlagewerk für alle Besitzer von Computern mit IEC (IEEE 488 Bus). Dazu gehört auch der PET sowie alle CBM-Computer. Grundlagen, das BUS-System, Meßdatenübertragung, Adressierung eines Instruments, kl. IEC-BUS-Lexikon u.v.a.
Best.-Nr. 123 19,80 DM



Programmieren in Maschinensprache mit CBM

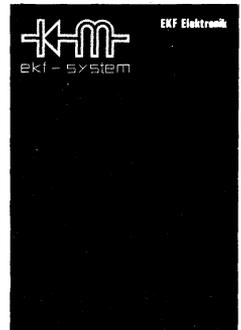
An Hand eines praktischen Beispiels (Sortieroutine) wird der Unterschied zwischen BASIC und Maschinensprache gezeigt. Das Maschinenprogramm kann mit dem leistungsfähigen MONJANA/1 Monitor in ROM erstellt werden. Am Schluß finden Sie weitere wichtige Informationen wie Dez/Hex-Umrechnungstabelle, Befehlslisten, ASCII-Tabelle sowie eine ROM-Vergleichsliste zwischen 8k PET und den neuen CBM-Maschinen.
Best.-Nr. 124 19,80 DM
MONJANA Monitor im ROM
Best.-Nr. 1241 79,00 DM

ohne Abbildung

ELCOMP, Fachzeitschrift für Microcomputertechnik

Die kompetente Fachzeitschrift für das moderne Gebiet der Microcomputertechnik. Erscheint 10 x pro Jahr. Jahrespreis * **59,00 DM** incl. MwSt., Porto und Verpackung. Wer die neuesten Informationen aus diesem Gebiet für sich nützen möchte, muß ELCOMP lesen. Software, Technische Tips, Programmiertricks, Bauanleitungen, Systembeschreibungen, u. v. a. Jeden Monat brandneu. ISSN-Nr. 0171-0958
Best.-Nr. 125 je Heft * 4,50 DM
ELCOMP-Doppelheft
 Von den 10 Heften erscheint im Juli/August und November/Dezember ein Doppelheft.
Best.-Nr. 126 * 9,00 DM

* ab 1.1.1982 gelten die neuen Preise:
 Abonnement 69,00 DM, Einzelheft 5,00 DM
 Doppelheft 10,00 DM



Einführung in die Microcomputer Programmierung mit 6800

Eine sehr gute Einführung in die Microcomputertechnik mit Hilfe des Mikroprozessors 6800. Ausführliche Erklärungen mit vielen Beispielen und Anleitungen. Theoretische Grundlagen. CPU-Architektur, Befehlssatz, Systemaufbau, Hilfsmittel der Programmierung, Trainingsprogramme, Systemkomponenten, FIRMWARE. Ein komplettes Monitorprogramm (Betriebssystem) ist als Listing enthalten.
Best.-Nr. 127 49,00 DM

ohne Abbildung

Programmieren mit dem CBM

Ein Hand- und Programmierbuch für alle CBM-Besitzer der 3000, 4000 sowie der 8000er Serie. Viele Tricks und Programmierbeispiele, Anleitungen. Erscheint Ende 1981.
Best.-Nr. 128 29,80 DM

ohne Abbildung

ELCOMP Leser Programmierhandb.

Hier fassen wir die besten Programme unserer ELCOMP-Leser zusammen. Programme für PET, CBM, TRS-80, AIM, Superboard, C4P, Exidy, Sharp, MZ80K, Apple II, Nascom I und II und TI 99/4 werden als Listing mit kurzer Beschreibung allen Lesern zugänglich gemacht. Erscheint Anfang 1982.
Best.-Nr. 129 69,00 DM



Programmierbeispiele für CBM
 Ein Buch mit vielen BASIC-Programmen für CBM und PET. Spiele, Geschäftsbereich, Erziehung und Wissenschaft, Utilities, Hilfen für Maschinensprachenprogrammierung, trickreiche Programme. Viele Programme für wenig Geld.
Best.-Nr. 130 19,80 DM

ohne Abbildung

Cobol für Anfänger

Eine Einführung in die Cobol-Programmierung für den Microcomputer-Besitzer. Erscheint ca. Mitte 1982.

Best.-Nr. 131 19,80 DM

ohne Abbildung

CP/M Handbuch

Grundlagen, Einführung, Hilfs- und Handbuch für jeden der mit dem "Software-Bus" arbeiten möchte. Ideal auch für Anfänger. Praktisches Handbuch für den Profi. Erscheint ca. Ende 1981.

Best.-Nr. 132 19,80 DM

ohne Abbildung

Welches Betriebssystem brauche ich?

Ein Leitfaden und Handbuch für den Anwender der gehobenen Microcomputerklasse. UNIX, CP/M, OASIS, NORTHSTAR DOS, USCD' Apple DOS, ATARI DOS, CP/A, usw. Erscheint Mitte 1982.

Best.-Nr. 133 19,80 DM

ohne Abbildung

Microcomputer im Unterrichtsfeld M. Penzkofer

Wie werden Microcomputer im Schulbetrieb eingesetzt? Grundlagen Wissenswertes, Tips, Erfahrungen und viele Beispiele. Für alle die die moderne Mikroelektronik für Schule Ausbildung und Erziehung nutzen wollen. Erscheint ca. Anfang 1982.

Best.-Nr. 136 29,80 DM

ohne Abbildung

FORTH Handbuch und Einführung von E. Flögel, W. Hofacker

FORTH ist nicht nur eine sehr leistungsfähige Programmiersprache – es ist schon fast eine "Religion". FORTH eignet sich bestens für industrielle Steuerungen, Grafik etc. Grundlagen und viele Programmbeispiele (Apple II, Ohio, ATARI usw.). Erscheint Mitte 1982.

Best.-Nr. 137 39,00 DM

ohne Abbildung

ADA - Handbuch, E. Flögel

Vorankündigung für Mitte 1982. Mit einer praktischen und leicht verständlichen Einführung soll hier dem Leser diese neue Programmiersprache nähergebracht werden. APA wurde, wie Cobol vom US-Verteidigungsministerium entwickelt und soll die steigenden Kosten bei der Softwareentwicklung eindämmen. ADA ähnelt in gewisser Weise d. Programmiersprache PASCAL und dürfte bald zum neuen Standard werden. Zur Zeit (Mitte 1981) gibt es jedoch noch keinen Microcomputer auf denen ADA implementiert ist.

Best.-Nr. 138 29,80 DM

ohne Abbildung

BASIC für blutige Laien von W. Hofacker

Endlich ein Buch für den Anfänger und Laien. Ziel des Buches ist es, dem "blutigen Laien" die Grundlage der Programmiersprache BASIC zu vermitteln. Lernen wird nun zum echten Vergnügen und Freizeitspaß. Auch der Preis macht Spaß. Erscheint ca. Ende 1981.

Best.-Nr. 139 nur 19,80 DM



Redysoft News

Das Magazin für den Microcomputer-einsatz im kommerziellen Bereich. Redysoft-News erscheint in zwangsloser Folge. Bisher sind drei Hefte erschienen. Aus dem Inhalt:

Heft 1: Microcomputer für freiberuflich Tätige. Die Anwendung von Microcomputern im Geschäftsbereich. Microcomputer-Markt. Ein Microcomputer zum Arbeiten. Die Microschule. MICROS werden voll geschäftsfähig. Challenger Superboard, der ideale Einstieg.

Heft 2: ACCEL - ein Compiler f. TRS-80 Level II BASIC. Maschinenprogramme von Southern Software für TRS-80 Level II

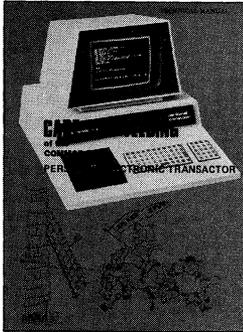
Laden von Maschinenprogrammen ACCEL, DLOAD, RENUM, HDUMP, FGRAF, XREF, ZBUG, SDUMP, TSAVE, USRN, SRCH, LIFE1 + LIFE2. Auch Siemens auf VLSI-Kurs. Dateiverwaltung. Struktur von FIBU-80.

Heft 3: APPLE Utility Disk. Bilanzbuchhaltung u. Finanzbuchhaltung f. den Microcomputer TRS-80 Model II. ACCEL u. ACCEL2 ? - BASIC-Compiler für den TRS-80 Model I. Eine Spielcassette für den TRS-80. Adressenverwaltung. Disassembler. TAPECHECK. SAVE. Terminkalender. Lagerverwaltung. ADEKART. Computerspiele für höchste Ansprüche.

Best.-Nr. 135 je Heft 2,00 DM

ELCOMP-Bücher

Englisch

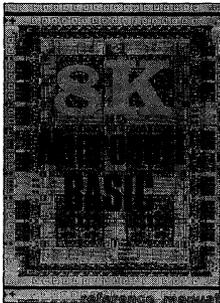


Care and Feeding of the Commodore PET

Das ideale Buch für den Hardware-Bastler. Viele Tricks, Schaltbilder, Hinweise und Erläuterungen für den, der gerne selbst Erweiterungen bauen möchte. Memory Map für 8k PET und CBM, Bauanleitung für eine serielle Schnittstelle u. v. a.
Best.-Nr. 150 19,80 DM

Complex Sound Generation with SN 76477

Ein Applikationsheft für einen der interessantesten integrierten Bausteine unserer Zeit. Ein LSI-Baustein zur Tonerzeugung. Je nach äußerer Beschaltung können Sie mit diesem Baustein die verrücktesten Töne erzeugen. Dampfisenbahngläus mit Dampfpeife, Vogelgezwitscher, Hundegebell, elektronische Orgel, Schuß mit Explosion u. v. a. mehr.
Best.-Nr. 154 19,80 DM

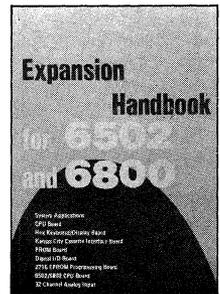


Microsoft 8k BASIC Reference Manual

Eine sehr gute BASIC-Einführung. Auch als Handbuch zum Nachschlagen bestens geeignet. Ideal für jeden PET, CBM, TRS-80, KIM-BASIC, SYM-BASIC, AIM- und APPLE-Besitzer. 73 Seiten DIN A4 mit vielen Beispielen.
Best.-Nr. 151 19,80 DM

Expansion Handbook for 6502 and 6800

Das ideale Handbuch für alle KIM, SYM, AIM, PET und Challenger Computer-Freunde. Das Buch beschäftigt sich ausschließlich mit dem S-44-Bus. Dies ist exakt der Bus von SYM, AIM und KIM. Sehr viele Schaltbilder: CPU-Platine, Hex-Tastatur Eingabe, Knsas City Interface, RAM u. ROM-Karte, Analog-Eingabe Board u. v. a. Das Buch ist für jeden 6502 Systembesitzer unentbehrlich. Ca. 150 Seiten.
Best.-Nr. 152 19,80 DM



intel[®]

Application Notes

8080
8085



8255
8251

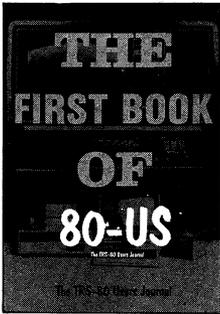


Intel Application Notes (8080, 8085, 8251)

Dieses Buch braucht jeder, der mit 8080, 8085 oder Z-80 Mikroprozessoren arbeitet.

Wir haben die interessantesten Applikationsberichte in diesem Buch zusammengefasst. Aus dem Inhalt: Designing with Intel's Static RAM's 2102, Memory Design with the Intel 2107B. 8255 Programmable Peripheral Interface Applications, Using the 8202 Dynamic RAM Controller u. v. a.

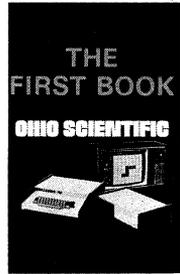
Best.-Nr. 153 29,80 DM



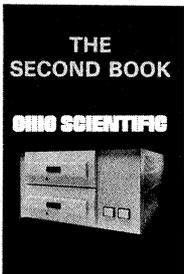
The First Book of 80-US
 Für den TRS-80 Freund eine echte Preissensation. Die ersten fünf Hefte aus 80-US Journals in einem Sammelband zusammengefaßt. Voll mit vielen sehr interessanten Hard- und Softwareideen, Tricks. Viele komplette Programmbeispiele (Listings) in BASIC u. Z-80 Maschinsprache. Über 250 Seiten DIN A4. Farbiger Umschlag. Dieses Buch sollte jeder TRS-80 Besitzer oder der es werden will im Schrank haben.
 Best.-Nr. 155 29,80 DM



Small Business Programs, S. Roberts
 Ein Buch für diejenigen, der die modernen Microcomputer (speziell TRS-80, Apple, PET, North Star, Challenger) zur Rationalisierung in seinem Klein- oder Mittelbetrieb einsetzen möchte. Viele nützliche Tips, Hinweise und Programmierbeispiele. Dieses Buch sollte jeder Geschäftsmann u. Microcomputerfreund besitzen.
 Best.-Nr. 156 29,80 DM



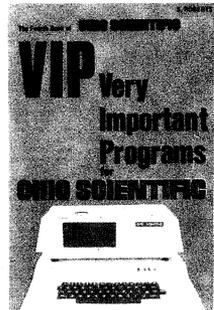
The first Book of Ohio Scientific
 Das erste weltweit produzierte Buch für die erfolgreiche Ohio Scientific Challenger Computerserie. Grundlagen, viele Programmiertricks, Hardwaretips, Umbauanleitungen, Programmierbeispiele u. v. a. Glanzumschlag, 186 Seiten.
 Best.-Nr. 157 19,80 DM



The second Book of Ohio Scientific
 Eingehende Beschreibungen über praktische und geschäftsorientierte Software. Speicher Test Programm, Tricks und Tips für Disketten-Anwender. Mini-Floppy-Expansion u. v. a. 159 Seiten.
 Best.-Nr. 158 19,80 DM

ohne Abbildung
 Erscheint ca. Ende 1981

The third book of Ohio
 Wie erweitere ich mein Challenger System ? Universelle I/O-Karte, EPROM-Burner für 2716, EPROM, RAM-Karte, 6522 VIA-Karte. Wo notwendig mit kompletter Software. Dieses Buch braucht jeder Ohio-Benutzer. Komplette Schaltbilder und Aufbauhinweise.
 Best.-Nr. 159 19,80 DM



The fourth Book of Ohio Scientific
 Ein Buch voll mit Programmen für das Superboard, C4P, C4PMF und C28P. Die Softwarequelle für jeden Challenger-Fan. Alle Programme sind getestet und auch auf Cassette verfügbar. 170 Seiten Listings und Beschreibungen.
 Best.-Nr. 160 29,80 DM
 Best.-Nr. 8324 Cassette 29,80 DM

ohne Abbildung

The fifth Book of Ohio Scientific Fortsetzung der beliebten Reihe über die Ohio Scientific Computer. Hier liegt der Schwerpunkt wieder bei der Software. Viel interessante Software. Eine komplette Adressverwaltung mit Label und Listenausgabe. Mehrere Fakturierprogramme, viele interessante Spiele wie Autorennen, Tankwar, Tonerzeugung, Joystickprogrammierung usw. Erscheint ca. Anfang 1982.

Best.-Nr. 161 19,80 DM

ohne Abbildung

25 Programs for the ATARI 25 sehr interessante Programme, Beschreibungen, Tips und Tricks für die ATARI 400 und 800 Computer. Wesentliche Unterscheidungsmerkmale zwischen ATARI - BASIC, MICROSOFT - BASIC und BASIC unter CP/A werden erläutert. Verwendung des ATARI - Editor / Assemblers zur Erstellung einfacher Maschinenprogramme. Erscheint ca. Anfang 1982.

Best.-Nr. 162 19,80 DM

ohne Abbildung

The peripheral Handbook Ein Handbuch über Drucker, Floppys, Hard-Disks, und alles was Sie um den Microcomputer herum benötigen. Triks und Kniffe. Technische Beschreibung. Erscheint ca. Mitte 1982.

Best.-Nr. 163 29,80 DM

ohne Abbildung

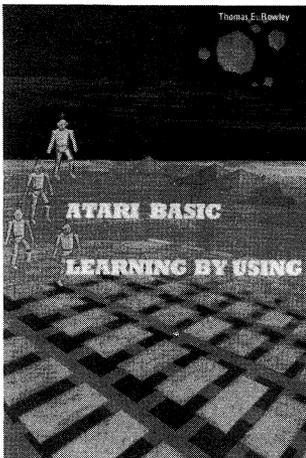
6502 Primer Erscheint ca. Anfang 1982.

Best.-Nr. 165 29,80 DM

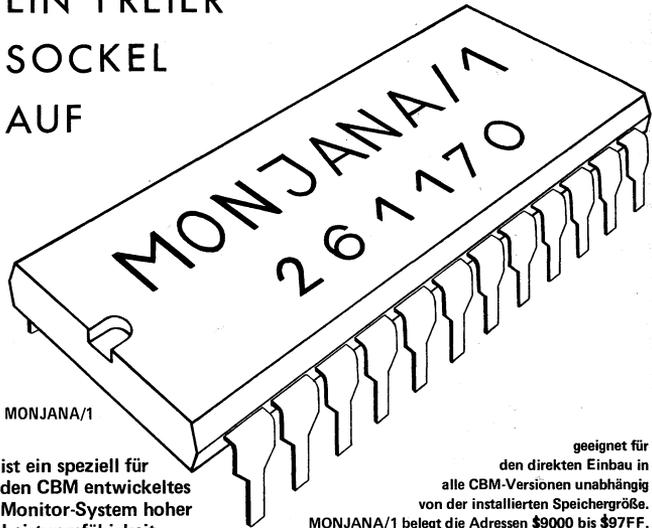
ATARI-BASIC - Learning by Using von Tom Rowley

Ein sehr interessantes Lehr- und Programmierbuch für jeden ATARI-Besitzer. Kurze Programme und Lernbeispiele machen das Programmieren zum Kinderspiel. Viele Programme in diesem Buch verwenden weiterführende raffinierte Programmier-techniken andere wiederum sind einfach und ideal für den Erstkontakt mit dem ATARI-Mikrocomputer. Tonerzeugung, Grafik, Joystickprogrammierung, Player Missile Graphic, Video-Kunst u. v. a. Ca. 75 Seiten.

Best.-Nr. 164 19,80 DM



IN IHREM CBM WARTET EIN FREIER SOCKEL AUF



MONJANA/1

ist ein speziell für den CBM entwickeltes Monitor-System hoher Leistungsfähigkeit

geeignet für den direkten Einbau in alle CBM-Versionen unabhängig von der installierten Speichergröße. MONJANA/1 belegt die Adressen \$9000 bis \$97FF.

Ing. W. Hofacker GmbH, Tegernseerstr. 18, 8150 Holzkirchen

BESTELLUNG

... Stück Monitor-System MONJANA/1 (2K-Byte-EPROM, incl. ausführlicher Systembeschreibung

zum Stückpreis von DM 79,- + Porto Lieferung per Nachnahme

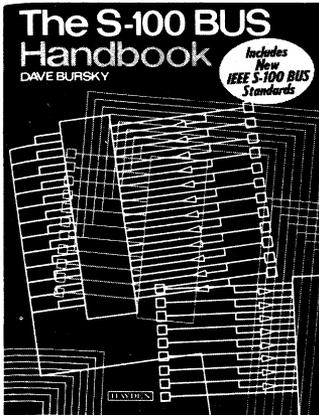
Monjana-1 Der Maschinensprachen-Monitor, der auch Ihnen gute Dienste leisten wird. Ideal für jeden, der die Leistungsfähigkeit seines CBMs voll ausschöpfen will.

Heute noch bestellen !

Name/Vorname _____
Straße / Hs-Nr. _____
PLZ _____ Ort _____
Datum..... Unterschrift.....



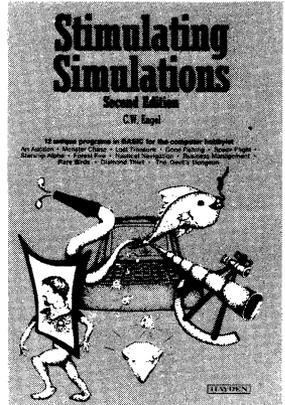
Hayden-Bücher



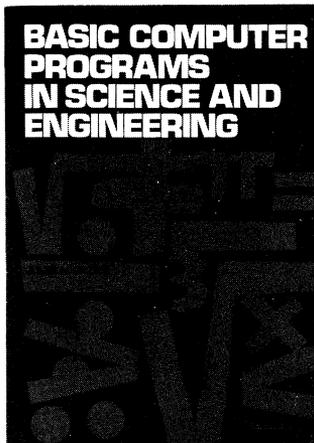
The S-100 Bus Handbook
(enthält den Neuen S-100 Standard). Alles was Sie über den S-100 Bus wissen sollten. Einführung, Schaltungen, Schaltbilder der oft verwendeten S-100 Platinen. Service u. Fehlersuche in S-100-Systemen, Interface-beispiele u. v. a. 256 Seiten Großformat.
Best.-Nr.: 254 49,00 DM



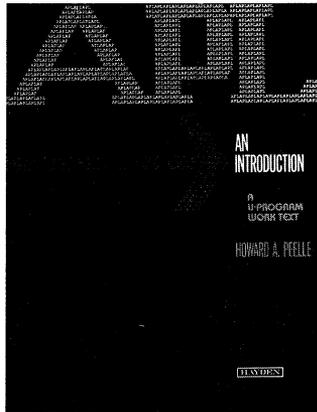
BASIC BASIC
Eine Einf. in die BASIC-Programmierung v. Microcomp. Eines der besten BASIC-Bücher weltweit. Viele Beispiele aus dem professionellen u. wissenschaftl. Bereich.
Best.-Nr.: 255 39,00 DM



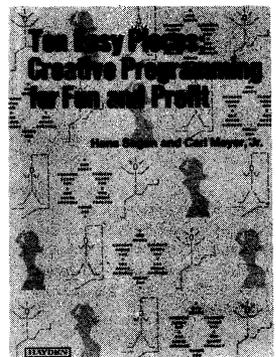
Simulating Simulations, v. C. W. Engel
12 einzigartige Programme in BASIC f. d. Comp.-Hobbyisten. (Kunstaktion, Monster Chase, Lost Treasure, Gone Fishing, Space Flight, Starship, Forest Fire, Navigation, Business Management, usw.)
Best.-Nr.: 256 19,80 DM



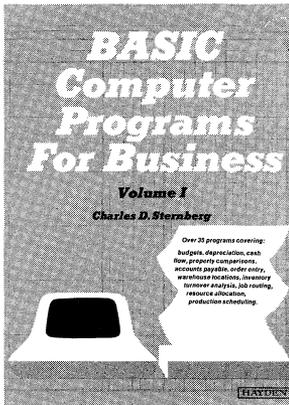
BASIC Computer Programs in Science and Engineering. (250 Seiten Großformat) Programme aus den verschiedensten Bereichen m. genauen Beschreibungen. Komplexe Zahlen, Engineering, Mathematik, Matrizen, Analyse von Daten, Elektrotechnik, Electronic, Filterberechnung usw.
Best.-Nr.: 257 39,00 DM



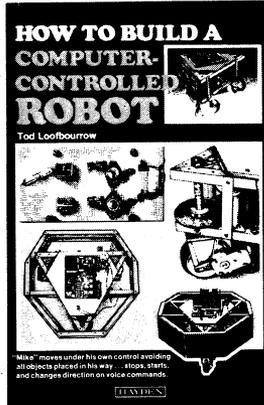
APL - An Introduction
Ein Programmier- und Arbeitsbuch. Ein Einführungsbuch zum Selbststudium von APL. Viele Beisp., Übungsbeisp. m. Lösungen.
245 Seiten Großformat.
Best.-Nr. 258 39,00 DM



Ten Easy Pieces:
Creative Programming for Firm and Profi
10 Beispielhaft beschriebene BASIC-Programme. Schritt f. Schritt wird jeder Befehl u. Zusammenhang erläutert. Schwerpunkt: Simulationsprogramme. Sehr gut geeignet f. Ausbildung, Selbststudium und als Nachschlagewerk.
Best.-Nr. 259 29,80 DM



BASIC Computer Programs für Business, Vol. 1 Charles D. Sternberg
 Über 35 Programme aus dem Gebiet : Budget, Abschreibung, Cash-Flow, Grundstückswertvergleich, Accounts Payable, Auftragseingang, Inventur-Umschlagsanalyse, Produkt-Terminplanung u. v. a. Alle Programme sind ausführlich beschrieben. Ein sehr wertvolles Buch. 250 Seiten, Großformat
 Best.-Nr. 260 39,-

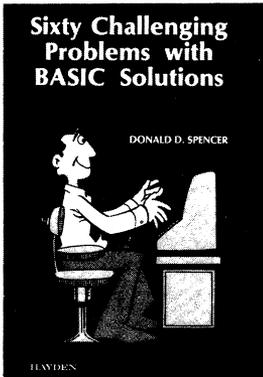


How to build a Computer controlled Robot, v. Tod Loofbourrow
 Komplette Bauanleitung für einen mit KIM-1 gesteuerten Roboter. Viele Schaltungen A/D- D/A-Wandler mit Software. (Ultraschall-Detektion, Spracherkennung u.v.a)
 Best.-Nr.: 253 35,00 DM

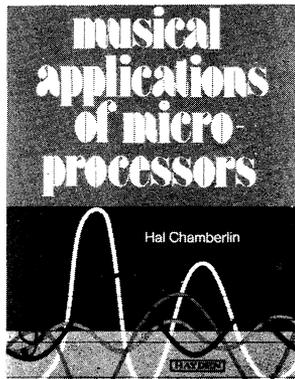


Home Computers Can Make Your Rich
 Ein Experte zeigt Ihnen wie man mit Personal-Computern Geld verdienen kann. Was ist zu beachten? Viele Ideen.
 Best.-Nr. 262 19,80 DM

ohne Abbildung
BASIC Computer Programs For Business
 Best.-Nr. 261 39,00 DM
The complete 1802 Cookbook
 Best.-Nr. 264 19,80 DM



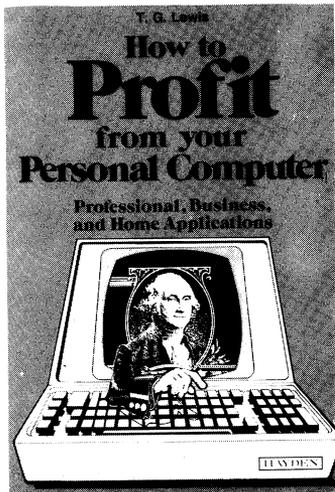
Sixty Challenging Problems with BASIC Solutions. 60 Spiele, Puzzles, mathematische und wissenschaftliche Problembeschreibungen, die der Programmierer in BASIC lösen soll. Am Ende des Buches sind zur Kontrolle alle Lösungen (60 BASIC Programme) aufgeführt.
 Best.-Nr.: 263 19,80 DM



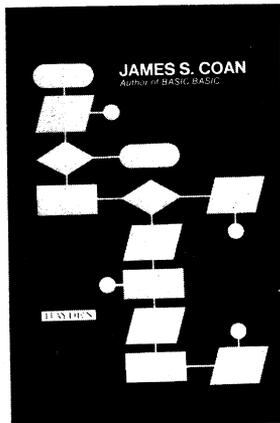
Musical Applications of Microprocessors von Hal Chamberlin.
 Hal Chamberlin kann als Papst für Computermusik bezeichnet werden. In diesen 660seitigen Buch hat er sein Wissen und seine Erfahrungen der Öffentlichkeit zugänglich gemacht. Grundlagen, Einführung, Schaltungsbeisp., Programmbeisp., digitale Tonzeugung, digitale Filter u. v. a.
 Best.-Nr.: 265 79,00 DM



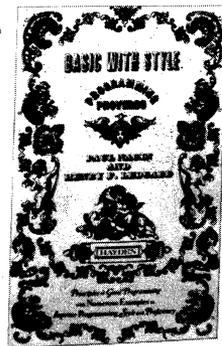
Advanced BASIC, Applications a. Problems
 Kann als Fortsetzung von Nr. 255 angesehen werden. Tiefgehende Informationen über Files, Strings, Geometrie, Reihen u. Folgen, Matrizen, Statistik u. Simulationen.
 Best.-Nr. 266 39,00 DM



How to Profit from your Personal Computer
 Dieses Buch ist für den Geschäftsmann und Computer-Hobbyisten geschrieben. Es zeigt Ihnen wie Sie den Computer für sich arbeiten lassen können und dabei noch Geld verdienen. (Viele Tips und Programmbeispiele in BASIC.)
 Best.-Nr. 267 **39,00 DM**



Basic FORTRAN
 Vom Autor des Buches "BASIC BASIC". Eine ausgezeichnete Einführung in die Programmiersprache FORTRAN. Schritt für Schritt mit vielen Beispielen (über 80) lernen Sie diese leistungsfähige Sprache.
 Best.-Nr. 271 (engl.) **45,00 DM**



BASIC with Style
COBOL with Style
PASCAL with Style
 Diese Buchserie ist eine Art "Hohe-Schule" für den ernsthaften Programmierer. Viele Tips Beispiele, nützliche Unter- und Hilfsprogramme für die Programmiererstellung am Arbeitsplatz oder in der Freizeit. Top-Down Programming, Program Standard Odds and Ends sind die Hauptthemen in diesen drei sehr wertvollen Büchern. Jedes Buch ca. 145 - 200 Seiten.

PASCAL with Style
 Best.-Nr. 268 **39,00 DM**

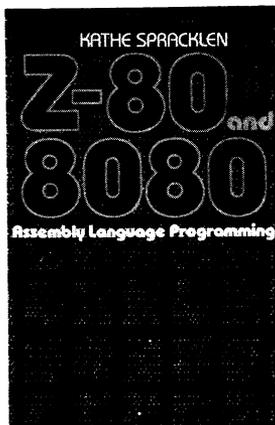
COBOL with Style
 Best.-Nr. 269 **39,00 DM**

BASIC with Style
 Best.-Nr. 270 **39,00 DM**

Endlich ein Buch über den 1802 Microprozessor

Den leistungsfähigen 1802 Microcomputer in CMOS Technologie, der heute vorwiegend in der Industrie und KFZ-Technik eingesetzt wird und in riesigen Stückzahlen produziert wird, wurde bis jetzt wenig allgemeines Interesse entgegengebracht. Ganz zu unrecht! Den dieser Prozessor ist für das oben genannte Einsatzgebiet hervorragend geeignet. Das Buch eignet sich für den Einsteiger genau so wie für den Experten. Einführung, Grundl. der Assembler Programmierung, Befehlssatz mit Erläuterungen. Ein kompletter 1802 Assembler als Listing, Übungsbeispiele m. Antworten.

Best.-Nr. 264 **19,80 DM**

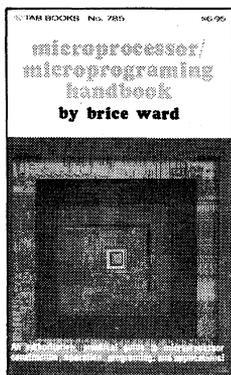


Z-80 and 8080 Assembly Language Programming
 Ein Einführungsbuch in die Maschinensprache für 8080 und Z-80 Systeme. Es bringt Ihnen fast alles, was Sie zur optimalen Nutzung Ihres Computers brauchen. Ideal für Anfänger.
 Best.-Nr. 272 (engl.) **39,00 DM**



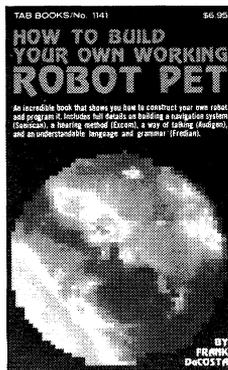
Beat the ODDS
 Microcomputer Simulationen von Casino-Spielen, Trente-et-quarante, Roulette, Blackjack, Craps, Chemin-de-fer. Eine extrem nützliche Programmierhilfe für realistische Simulationen der wichtigsten Casino Spiele.
 Best. Nr. 273 **39,00 DM**

TAB-Books



Microprocessor Programming for the Computer Hobbyist

Dieses buch ist speziell für den Computer-Hobbyisten geschrieben, der sich bereits mit weiterführenden Programmieretechniken und Datenstrukturen beschäftigen möchte. Inhalt: Höhere Programmiersprachen (PL/M, PL/1), arithmetische Funktionen, Suchprogramme werden im Zusammenhang mit dem Schachspielproblem behandelt. Über 380 Seiten in englischer Sprache
Best.-Nr. 952 DM 39,-



How to build your own working ROBOT PET

An incredible book that shows you how to construct your own robot and program it. Includes full details on building a navigation system (Soniscan), a hearing method (Excom), a way of talking (Audigen) and an understandable language and grammar (Fredian).

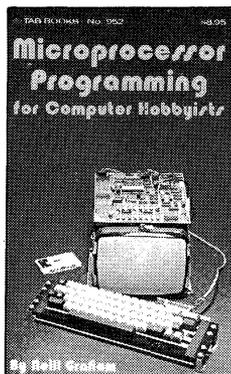
Eine wertvolle Hilfe für alle, die sich einen Roboter bauen wollen. (8085) CPU). Viele wertvolle Schaltungen, die Sie im Zusammenhang mit TRS-80 oder 8085-Computern verwenden können. 238 Seiten.

Best.-Nr. 1141 DM 29,80

Microprocessor/Microprogramming Handbook, Brice Ward

Ein praktisches Handbuch für jeden, der sich mit Mikroprozessoren beschäftigen möchte. Auf über 290 Seiten finden Sie Grundlagen, alles über Programmierung und Anwendungsbeispiele der interessantesten integrierten Schaltungen. Inhalt: Einführung in die Mikroprozessortechnik. Der Mikroprozessor von innen. MCS4, MCS40, MCS80, Speichersysteme, Mikroprogrammierung in Maschinen- und Assemblersprache.

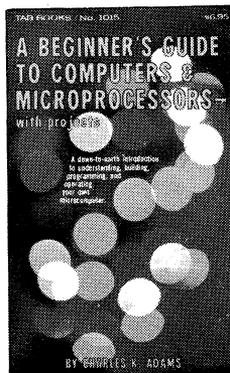
Best.-Nr. 785 DM 35,-

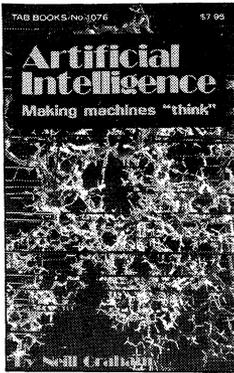


Beginner's Guide to Computer Programming, Brice Ward

Eine ideale Einführung in die Programmierung von Computern (Mikroprozessoren). Das Buch beginnt mit der Entwicklung einer einfachen Programmiersprache für den eigenen Bereich und zum Selbststudium, sodann wird auf andere Sprachen übergegangen. Inhalt: Grundlegende Programmkonzepte, I/O-Schaltungen, Flußdiagramme, Programmtest, Schleifen, Indexregister, versch. Programmiersprachen, Compiler, Cobol u. v. a., 480 Seiten, 364 Bilder, in englischer Sprache.

Best.-Nr. 1015 DM 29,80





Artificial Intelligence, Neil Graham
 Hier ist endlich ein Buch speziell über künstliche Intelligenz. Der Autor benutzt Computerspiele und Robotertechnik für die illustrative Behandlung dieses brandheißen Themas. Was ist der Unterschied zwischen einer Lösung einer komplexen Programmieraufgabe und der Programmerstellung für intelligente Computerentscheidungen.

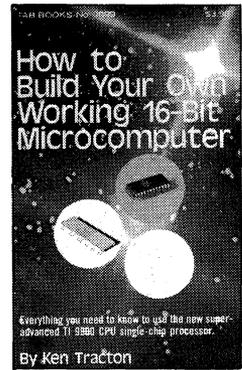
Best.-Nr. 1076 DM 29,80

Microprocessor Cookbook, von Michael Hordeski

A chip-by-chip-comparison of today's modern microprocessors.

Die wichtigsten Prozessortypen werden genau beschrieben (Schaltung, Blockdiagramm, Befehlslisten, Programmbeispiele) und verglichen: 8080, 6800, F8, Z80, TMS 9900, SC/MP, Bit Slices, R6500, 264 S.

Best.-Nr. 1053 DM 24,80



How to Build your Own Working 16-Bit-Microcomputer von Ken Tracton

Alles, was Sie über den neuen Supermicroprozessor TI9900 wissen müssen. Beschreibung des Schaltkreises, Peripherie-Bausteine für TMS9900, Tricks, Systemaufbau etc. Sehr interessant, da der TMS9900 das Herz des neuen Texas Instruments Personal Computers ist.

Best.-Nr. 1099 DM 14,80



Digital Interfacing with an Analog World, Joseph J. Carr

Wie entwirft und baut man Interfaceschaltungen für Microcomputer? Der Schwerpunkt liegt bei der Verbindung mit der analogen Welt. Viele Tabellen, Schaltungen, Berechnungen und Hinweise. Über 400 Seiten.

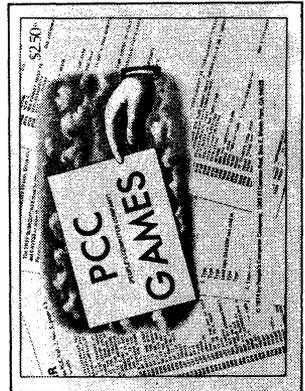
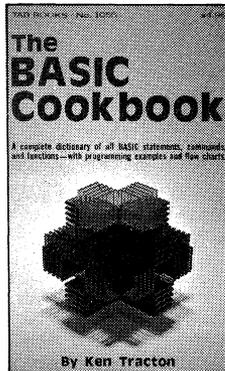
Best.-Nr. 1070 DM 39,-

The BASIC Cookbook, von Ken Tracton

Ein komplettes Dictionary mit BASIC-Befehlen. Alle Befehle sind in alphabetischer Reihenfolge geordnet und jeder Befehl genau beschrieben. Zu jedem Befehl ist ein Demonstrationsbeispiel beigelegt.

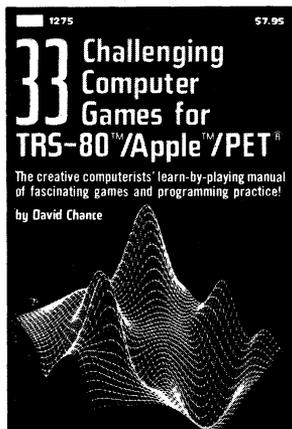
Dieses Buch braucht jeder BASIC-Programmierer!

Best.-Nr. 1055 DM 24,80



25 Games in BASIC (Listings). Die Programme laufen mit kleinen Änderungen auf den meisten Microcomputern. Inhalt: Number, Letter, Stars, Trap, Bagels, Mugwump, Hürkle, Snark, Reverse, Button Chomp, Taxman u. v. a.

Best.-Nr. 8057 DM 9,80



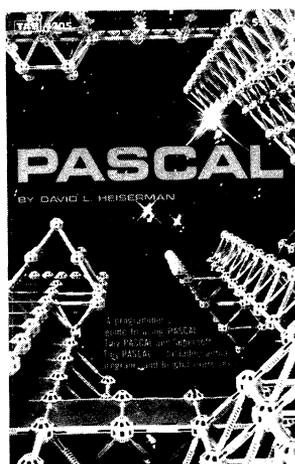
33 Challenging Computer Games for TRS-80, APPLE, PET

Das Handbuch für den Computerist, der durch Spielen lernen will. Über faszinierende Spiele zur Programmierpraxis. Ein Buch f. jeden Personal Computer-Bes.
Best.-Nr.: 1275 29,80 DM



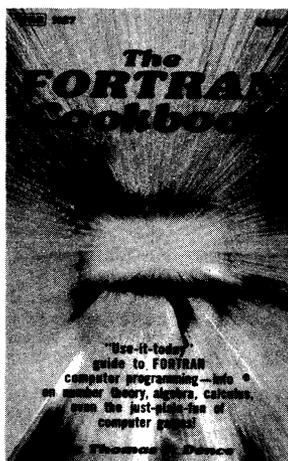
Handbook of Microprocessor Applications (6800, 6802)

Wie werden Microcomputer in der Praxis eingesetzt. Interfaetechniken u. Anwendung v. Maschinensprache, Schaltungsbeispiele mit Software. Beschäftigt sich in erster Linie mit 6800, 6802 und den zugehörigen Peripheriebausteinen.
Best.-Nr.: 1203 29,80 DM



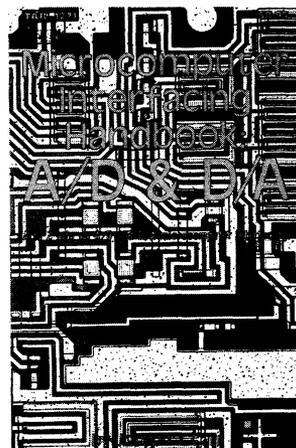
PASCAL

Eine Programmieranleitung f. d. PASCAL-Freund. Wie verwendet man TINY PASCAL und Supersoft-PASCAL. Passt ideal zu diesen Paketen und TRS-80. Viele Programmierbeispiele. (350 Seiten)
Best.-Nr. 1205 35,00 DM



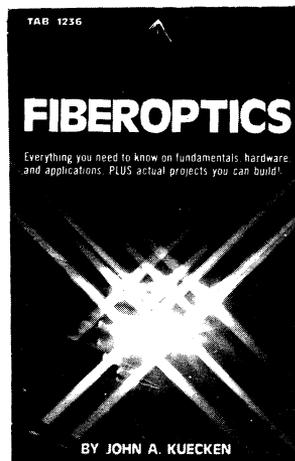
The FORTRAN Cookbook

Heute noch verwenden! eine Einführung in die FORTRAN-Programmierspr., Grundlagen, Beispiele, Übungen und Beispiele aus dem Microcomputerbereich. (Mathem. Probleme bei Computerspielen u. v. a.)
Best.-Nr.: 1187 29,80 DM



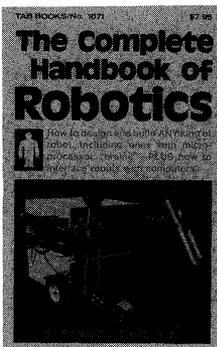
Microcomputer Interfacing Handbook: A/D- und D/A-Wandler

Ein Hand- u. Nachschlagebuch f. jeden der Analog/Digital od. Digital/Analog-Wandler an seinen Microcomputer anschließen will. Unentbehrlich f. die Anwendung des Microcomputers im industriellen Einsatz.
Best.-Nr.: 1271 35,00 DM



FIBEROPTICS

Alles was Sie über d. Grundlagen, Hardware und Applikationen dieser faszinierenden neuen Technik wissen sollten. Mit Projekten, die Sie selbst bauen können. (Übertragungsraten bis zu 1.354×10^9 Bildpunkte pro Sekunde, das bringt die Fiberoptic!)
Best.-Nr.: 1236 29,80 DM

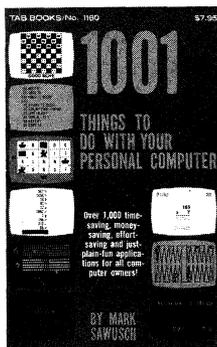


The Complete Handbook of Robotics
E. Safford, jr.

Wie entwickle und baue ich einen Roboter. Es werden auch microcomputer-gesteuerte Roboter beschrieben, und wie man Roboter mit Computern zusammenschaltet. Terminologie, Entwicklungsstand, Sensoren, Grundlagen, das Roboter "Gehirn", Servomechanismus, Tips, Interfacing u. v. a.

Best.-Nr. 1071

DM 29,80

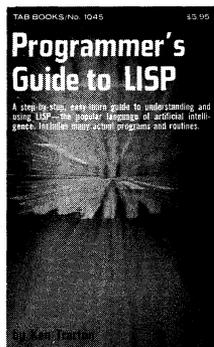


1001 Things to do with your Personal Computer, Mark Sawusch

Computeranwendungen für jedermann, Geschäft und Finanzprogramme, mathematische Applikationen, technisch/wissenschaftliche Anwendung von Personal Computern, Anwendungen in Erziehung und Ausbildung, Hobby Computer, Spiele, Control- und periphere Applikationen, künstliche Intelligenz. Viele Programme in BASIC, viele Tips. 330 Seiten

Best.-Nr. 1160

DM 29,80

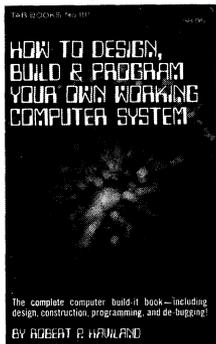


Programmer's Guide to LISP, K. Tracton

Eine Einführung in die Computersprache LISP. LISP ist eine spezielle Interpretation, die sich gut für die Programmierung im Bereich künstliche Intelligenz eignet. Das Buch zeigt Ihnen Wege und Lösungen. Viele Programmbeispiele. 210 Seiten.

Best.-Nr. 1045

DM 24,80



How to Design, Build and Program your own Working Computer System

R. Haviland

Dieses Buch zeigt Ihnen Schritt für Schritt, wie Sie einen eigenen Computer aufbauen können. Es lehnt sich dabei an den erfolgreichen SC/MP von National an. Ein Buch, das jeder SC/MP-Besitzer haben muß.

Best.-Nr. 1111

DM 29,80

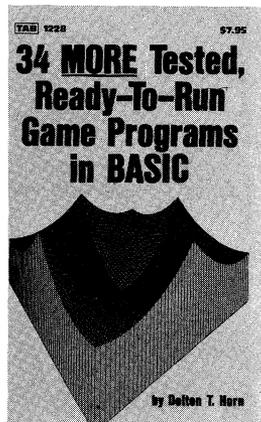


24 Tested Ready to RUN Game Programs in BASIC, Ken Tracton

Spaß- und Spielprogramme in BASIC. Viele Programme enthalten spezielle Anpassungshinweise an die Homecomputertypen TRS-80 und PET. Viele Graphik- und Zeichenprogramme. Ideal für jeden BASIC-Computer-Besitzer.

Best.-Nr. 1085

DM 24,80



34 MORE Tested Ready-To-Run Game Programs in BASIC

34 Spielprogramme für Video Genie und TRS-80, aber auch für andere BASIC Personalcomputer. Ein Buch, welches Ihnen viel Freude mit Ihrem Computer bereiten wird.

Best.-Nr. 1228

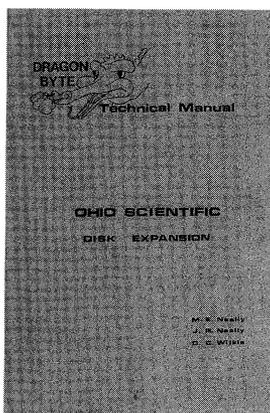
35,00 DM



67 Ready to Run Programs in BASIC

Ein Buch mit vielen Programmbeispielen für alle die einen BASIC-Computer haben (ab 4K RAM). Viele Tips, Anregungen und Hilfen bringen auch dem Erstanwender großen Nutzen.

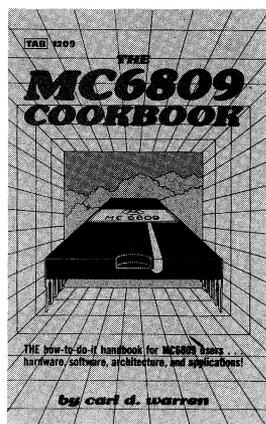
Best.-Nr. 1195 29,80 DM



Dragon Byte Disk Expansion Book

Ein Büchlein für den, der seinen Ohio Scientific Computer zum Diskettensystem selbst ausbauen will. Sehr gute technische Informationen, die Sie sonst nur schwer finden.

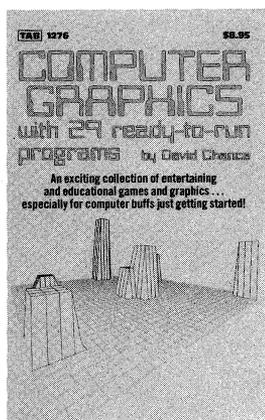
Best.-Nr. 311 29,80 DM



The MC 6809 Cookbook

Endlich ein ausführliches Handbuch für den extrem Leistungsfähigen 6809 Prozessor, Hardware, Software, Architektur. Wie macht mans? Von Carl Warren.

Best.-Nr. 1209 29,80 DM



Computer Graphics with 29 ready-to-run Programs

Eine Einführung in die Microcomputergrafik für Anfänger. Eine wahre Fundgrube für denjenigen der gute Programmbeispiele mit Grafik sucht. (Ideal für Video Genie und TRS-80 Level II BASIC.)

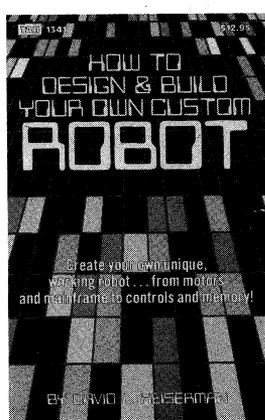
Best.-Nr. 1276 39,00 DM



How to build your own working Microcomputer

Dieses Buch zeigt Ihnen wie Sie sich selbst einen einfachen Microcomputer bauen können. Programmispiele, Funktionsdiagramme, Schaltbilder. Auch für Anfänger geeignet.

Best.-Nr. 1200 49,00 DM



How to Design and Build your own Custom Robot

Bauen Sie Ihren eigenen Roboter, der Ihre Arbeiten erledigen kann. Einführung, elektrische und mechanische Grundlagen, Microcomputer-Hardware (8085 und 780) u. v. a.

Best.-Nr. 1341 59,00 DM

PROGRAMM BIBLIOTHEK

BASIC



Application Software!

INHALTSVERZEICHNIS DER BASIC PROGRAMMBIBLIOTHEK VOL I - VOL V

VOLUME ONE

Part 1 - Business + Personal Bookkeeping Programs

Name	Description
Bond	Computers price and interest for bond purchases.
Building	Analyzes the cost of building design proposals.
Compound	Computes effective compound interest rates.
Cyclic	Determines seasonal coefficients for two cycles.
Decision 1	Makes a Lease/Buy decision for you.
Decision 2	Makes a decision on whether to buy a component or make it.
Depreciation	Calculates depreciation by 4 different methods.
Efficient	Cal. the most efficient assignment of resources and / or personnel.
Flow	Predicts your yearly cash flow.
Installment	Performs monthly installment accounting.
Interest	Computes interest accruals, monthly.
Investments	Computes annual rates of return on investments.
Mortgage	Makes a comparison of mortgage terms.
Optimize	Optimizes the layout for a plant, shop, office, etc.
Order	Determines your economic order quantity for inventory items.
Pert Tree	Performs an analysis of a pert network.
Rate	Computes true annual interest rates.
Return 1	Computes lessor's rate of return for uncertain assets.
Return 2	Computes a lessor's rate of return after taxes.
Schedule 1	Schedules N jobs in a shop with M machines.

Part 2 - Games + Pictures

Name	Description
Animals Four	Teach the computer all about animals.
Astronaut	Land your spaceship on another planet.
Bagel	Advanced number game, numbers may be algebraic, few clues.
Bio Cycle	Calculate your Bio-Life Cycle and plan your days.
Cannons	An advanced war game with big guns.
Checkers	Plays a regulation game of checkers.
Craps	A dice game with hard way odds.
Dogfight	Air fight w/missiles; between a phantom and a mig.
Golf	Plays any number of holes; inc. obstacle course.
Judy	Have a rap session with Judy via your computer.
Line Up	Simple number game, all you have to do is unscramble them.
Pony	Authentic horse race, any number of players.
Roulette	Gamblers delight, plays Las Vegas rules.
Sky Diver	Sky dive on another planet
Tank	A war game between two tanks.
Teach Me	Teach the computer to learn new things.
A Newman	He's absolutely MAD! MAD! MAD!
J. F. K.	Our 35th. president.
Linus	Loveable "Peanuts" character, w/blanket.
Ms. Santa	A modern miss to put a twinkle in your eye.
Nixon	Former "United States" president.
Noel Noel	Christmas or anytime this is a beautiful creation.
Nude	A true work of art for anyone's gallery.
Peace	A message for all seasons.

Name	Description
Policeman	True and blue, he's the law.
Santa's Sleigh	In banner form, perfect for decorating the mantle.
Snoopy	That paragon of Dogdom even plays football.
Virgin	A picture you can read as well as see.

VOLUME TWO

Part 3 - Math + Engineering Programs

Name	Description
Beam	Evaluates and selects steel beam sizes.
Conv.	Calculates convolutions.
Filter	Calculates low pass filter components.
Fit	Performs interpolations by spline fits.
Integration 1	Uses Gaussion Quadrature to do integration.
Integration 2	Integrates a function by spline fits.
Intensity	Calc. and plots RF or Acoustic intensities.
Lola	Calc. Long. and Lat. from interstellar fix or distance.
Macro	Simulates a language compiler.
Max. Min.	Calc. the max. + min. values of funct. over a special interval.
Navaid	Calc. position from altitude and azimuth of celestial bodies.
Optical	Calculates Blackbody energies, w/filter look-up tables.
Planet	Calculates Sun and Moon positions, hourly.
PSD	Calculates Power Spectral Densities and FFT's.
Rand 1	Generates random numbers between 0 and 1.
Rand 2	Generates random integers between (X) and (Y).
Solve	Solves polynominals by "Bairstows Methods".
Sphere Trian	Solves any spherical triangle.
Stars	Locates 50 stars (celestial).
Track	Calc. course and distance and incremental vectors.
Triangle	Solves for all parts of any triangle.
Variable	Finds all variables in BASIC programs.
Vector	Calc. final position; given start and motion vectors

Part 4 - Plotting + Statistics Programs

Binomial	Calculates binomial probability distributions.
Chi-Sq.	Applies the Chi-Square test to samples.
Coeff	Calc. coefficients of fourier series to apprx. a function.
Confidence 1	Calculates confidence limits on linear regressions.
Confidence 2	Calculates confidence limits for a sample mean.
Correlations	Performs auto and cross correlations with plots.
Curve	Fits 6 different curves by the least squares method.
Differences	Calculates difference of means in non-equal variances.
Dual Plot	Plots two functions on the same sheet.
Exp-Distri	Calculates exponential distributions for a sample.
Least Squares	Performs least squares fit by linear, exp., or power function.
Paired	Compares 2 groups of data using the rank test.
Plot	Plots 6 equations on the same sheet.
Plots	Plots data points on standard teletypes.
Polynomial Fit	Performs least squares polynomial fit.
Regression	Performs multiple linear fit with or without transformations.
Stat 1	Finds the mean, variance and standard deviations.

Name	Description
Stat 2	Computes various stat. measures for a variable.
T-Distribution	Calculates normal and T-distributions.
Unpaired	Compares 2 groups of unpaired data.
Variance 1	Performs one way analysis of variances.
Variance 2	Analyzes a variance table of one way random design.
XY	Plots functions of X and Y.

APPENDIX A - BASIC STATEMENT DEFINITIONS

VOLUME THREE

Part 5 - Advanced Business Programs

Billing	Performs posting and billing of accounts.
Inventory	Maintains data for inventory records.
Payroll	Computes payrolls with full set of deductions.
Risk	Performs a risk analysis on capital investments.
Schedule 2	Performs the most effi. scheduling of men or resources to loca.
Shipping	Solves the problem of scheduling and assignments.
Stocks	Computes the value of stocks.
Switch	Calculates the effects of a bond switch.

VOLUME FOUR

General Purpose Programs

Bingo	An age old favorite. "B9, C23, D4, E13, F21, BINGO!
Bonds	Computes the yields for a bond for different periods.
Bull	If you ever dreamed of being a Matador, here's your chance.
Enterprise	Take charge of the Enterprise while Capt. Kirk is on leave.
Football	Authentic NFL version of this well known sport.
Funds 1	Calculates long-term predictions of funds.
Funds 2	Plots the results of Funds 1.
Go-Moku	Ancient Chinese game of chance.
Jack	Plays Blackjack, Las Vegas style.
Life	Life is truly a battle for survival, a real challenger!
Loans	Calculates annuities, loans and mortgages.
Mazes	Generates unique maze puzzles for you to solve.
Poker	Five card draw - for up to 5 players.
Popul	Performs population projections for defined areas.
Profits	Determines the profitability of a firms various depts.
Qubic	3-Dimensional Tic-Tac-Toe.
Rates	Calc. the effective annual interest rate for stated interest.
Retire	Calculates your Civil Service Retirement benefits.
Savings	Computes savings plan profiles.
SBA	Calculates repayment schedules for SBA loans.
Tic-Tac-Toe	An all time favorite for young and old alike.

VOLUME FIVE

Experimenter's Programs

Andy Cap	Draws this famous cartoon character.
Baseball	Plays a full 9 innings of baseball.
Compare	Compares two groups of data.
Confid 10	Determines the confidence limits for a normal population.
Descrip	Provides a description of uni-variant data.
Differ	Computes the diff. of the means for data of equal variance.
Engine	Calculates the otto cycle of engines.
Fourier	This program evaluates fourier series.
Horse	Draws a picture of a horse.
Integers	Computes integers as the sum of other integers.

Name	Description
Logic	Determines conclusions from logic statements.
Playboy	Draws the playboy symbol.
Primes	Factors numbers into their primes.
Probal	Calc. Chi-Sq. and probabilities from 2X2 data sets.
Quadrac	Solves quadratic equations
Red Baron	Draws a picture of the infamous Red Baron.
Regression 2	Calculates linear regressions.
Road Runner	"Beep! Beep!" Draws a picture of the Road Runner.
Roulette	Computerized "Wheel of Fortune", plays roulette.
Santa	Old Saint Nick appears as jolly as ever.
Stat 10	Calculates quantities for two groups of paired data.
Stat 11	Computes sample statistics.
Steel	Calculates steel beam capacities.
Top	Computes cost for surfacing a road or driveway, etc.
Vary	Performs an analysis of a vari. table; one-way random design.
Xmas	Generates a "SINGING" Christmas card.

APPENDIX B - STATEMENT CONVERSION ALGORITHMS

VOLUME SIX

A Complete Business System

Ledger Maintains ALL Company accounts and generates ALL financial reports. Includes routines for: Pyrl, Inv. Depr. A/R, A/P, Balance Sheets and Profit + Loss statements, etc.

ACBS rev: 80 Users Manual - A Proprietary Package

VOLUME SEVEN

Professional Programs

Chess Designed to challenge the average player, fairly comprehensive. Great fun for all, offers a unique opportunity for beginners in need of an opponent.

Medbil For Doctors and Dentists alike, a complete patient billing system which also permits the maintaining of a patient history record.

Wdproc Wordprocessing for lawyers, publishers, writers etc. Write, store and change from rough draft to final copy in a variety of formats.

VOLUME SEVEN

Professional Programs

Utility

Licensing Agreement

BESTELLSCHEIN

Heute noch an Ihren Buch- oder Fachhändler absenden! Bitte senden Sie mir folgende Bücher:

- Stck. Vol. I , Best.Nr. 80/50, DM 99,--
- Stck. Vol. II , Best.Nr. 80/51, DM 99,--
- Stck. Vol. III, Best.Nr. 80/52, DM 149,--
- Stck. Vol. IV, Best.Nr. 80/53, DM 39,--
- Stck. Vol. V , Best.Nr. 80/54, DM 39,--
- Stck. Vol. VI, Best.Nr. 80/48, DM 199,--
- Stck. Vol. VII, Best.Nr. 80/49, DM 159,--
- Stck. Programm Bibliothek BASIC, (Volume I bis Vol. V) Best.Nr. 80/21, DM 425,--

Hofacker Verlag

GRUNDAUSSTATTUNG

Die Grundausrüstung der BASIC Programmbibliothek besteht aus fünf Büchern mit insgesamt 1100 Seiten Programm listings und genauen Beschreibungen. (Format DIN A 4)

Sie finden in diesem Werk 149 verschiedene BASIC Programme aus folgenden Bereichen:

- Buchhaltung
- Computerspiele
- Bildprogramme
- Mathematik und Ingenieurwissenschaften
- Statistik, technische Mechanik etc.
- Commerzielle Programme, Rechnungswesen, Fakturierung
- Inventurprogramme, Lohn- und Gehaltsbuchhaltung
- Experimentierprogramme u.v.a.

Jedes Programm ist genau beschrieben.

Die fünf Bände bilden eine komplette "Do It Yourself" Programmbibliothek, die sehr einfach zu handhaben ist. Es werden jedoch Grundkenntnisse in der Programmierung vorausgesetzt.

PREISSENSATION

Bei 149 Programmen in dieser umfangreichen Bibliothek bezahlen Sie nur
DM 2.85 pro BASIC Programm - Das ist einmalig!

BESCHREIBUNG DER PROGRAMMBIBLIOTHEK

Diese Bibliothek ist die umfangreichste und universalste ihrer Art auf dem Weltmarkt. Die Besonderheit dieser fünf Werke liegt darin, daß dem Anwender die Möglichkeit gegeben wird, in erster Linie sinnvolle und produktive Aufgaben wie Buchhaltung, Statistik, mathematische Probleme etc., zu lösen. Computerspiele sind jedoch auch enthalten.

Alle Programme wurden mehrmals auf verschiedenen Systemen ausgeführt und getestet. Jedes Programm ist mit einer genauen Beschreibung und einer Aufstellung der infrage kommenden Anwender versehen. Befehle und mögliche Grenzen der Programme sind auch aufgezeigt, wenn die Programme auf unterschiedlichen Systemen ausgeführt werden sollen. Auch der benötigte Speicherbereich ist für jedes Programm genau angegeben.

Jedes Programm ist in seiner vollständigen Größe aufgeführt und nicht reduziert. Es ist in sich selbst vollständig und kann im Ablauf genau verfolgt werden. Es gibt Aufschluß über alle wichtigen Daten. Bei den meisten Programmen folgt unmittelbar nach dem Programmlisting ein Probelauf. Die meisten Programme sind in kompatibelem BASIC geschrieben, welche auf den meisten 4 K BASIC-Versionen ausgeführt werden können.

Best.Nr. 80/21, 5 Bücher mit 1100 Seiten DIN A 4 (149 Programme) nur DM 425,--

Die Bände sind auch einzeln erhältlich:

BASIC Volume I

Programme über Buchhaltung, Computerspiele und Bilder (Computerbilder per Programm)

Best.Nr. 80/50 DM 99,--

BASIC Volume II

Programme über mathematische Probleme, technische Programme, Zeichnen, Plotting und Statistik. Grundlegende Statements.

Best.Nr. 80/51 DM 99,--

BASIC Volume III

Erweiterte Geschäftsprogramme für den kommerziellen Bereich. Fakturierung, Inventuren, Gehaltsbuchhaltung.

Best.Nr. 80/52 DM 149,--

BASIC Volume IV
BASIC Programme aus allen Bereichen. Universelle Anwendung.
Best.Nr. 80/53

DM 39,--

BASIC Volume V
Experimentierprogramme.
Best.Nr. 80/54

DM 39,--



BASIC Volume VI
Ein komplettes, interaktives Programmpaket für Floppy Disk-Einsatz im kommerziellen Bereich. Gehaltsabrechnung, Inventur, Gewinn und Verlustrechnung u. v. a.
Best.Nr. 80/48

DM 199,--

BASIC Volume VII
Das Schachprogramm, auf das Sie schon lange gewartet haben. Spielt zwei Farben. Leichter bis mittlerer Schwierigkeitsgrad. Kann einfach erweitert werden. Zugzeit ab 3 Minuten.
Best.Nr.

Medical Billing package (Patientengeschichte und Kostenübersicht für Ärzte)

Disk interaktives Textverarbeitungssystem.

Best.Nr. 80/49

DM 159,--

Genauere Beschreibung des BASIC Volume VII

Schachprogramm (Chess)

Diese Version eines Schachprogrammes in BASIC ist für die meisten 12 K BASIC-Versionen (mit oder ohne kleinen Änderungen) geeignet. Weiterhin sollten noch ca. 12 K im RAM-Bereich zur Verfügung stehen. Das Programm spielt Schach im Schwierigkeitsgrad für Anfänger, kann aber durch entsprechende Änderungen auch schwieriger gestaltet werden. Züge werden im Bereich von 3 Minuten oder mehr durchgeführt. Der erste Zug benötigt ca. 3 Minuten. Die Zugzeit steigt dann entsprechend dem Spielfortschritt entsprechend an. Das Programm kann wahlweise schwarz oder weiß spielen. Das Spielfeld wird auf dem Bildschirm angezeigt oder kann auf einem Drucker ausgegeben werden.

BASIC-Programm für die Arztpraxis (Medbil)

Dieses Programm wurde zur Erleichterung der so teuren und mühsamen Alltagsarbeit der Ärzte entwickelt. Dieses Programm ermöglicht die Speicherung und Durchsicht jeder Patientengeschichte auf dem Bildschirm, die vorher in einer Datenbank gespeichert wurde. Weiterhin besteht eine Möglichkeit der schnellen Überprüfung von aufgelaufenen Behandlungskosten pro Patient. Auf einer 250 K Byte Diskette können etwa 110 Patienten mit all ihren Daten aufgenommen werden. Mit einer Minifloppy (80 K Byte) reduziert sich die mögliche Patientenzahl auf ca. 25 pro Diskette.

Textverarbeitungsprogramm (Wrdpro)

Dieses Programm ermöglicht dem Besitzer von Microcomputern Texte zusammenzustellen und in beliebigen Formaten wieder auszudrucken. Das Programm läuft auf den meisten extended BASIC-Versionen mit mindestens 15 K freiem RAM-Bereich.

**W. Hofacker
Verlag**

Auch Sie brauchen ELCOMP!



Preis pro Heft 29,80 DM
incl. MwSt. und Versand
Zurückliegende Hefte sind zu
Originalpreisen noch verfügbar.



4. Jahrgang Volume IV iF7 9.00 F 9.00 US 70.00 Juli-August 1981 DM 9.00 Nr. 7/8
- ELCOMP**
FACHZEITSCHRIFT
FÜR MICROCOMPUTER
- 01 ATARI bald Nr. 3 im Personal Computermarkt?
 - 06 Aktuelle Microcomputer Forum 1981
 - 07 Lösung des Dreifarbproblems durch Differenzverfahren
 - 10 TRS-80 und Funktionsrechner
 - 26 Programmieren in Maschinensprache, Teil 2
 - 32 Ein Computer ohne Name
 - 32 Programm: SHAFEMAKER für APPLE II
 - 38 Datensicherung in Masterprozessoren
 - 40 Programme STARR
 - 45 Disk-BASIC in Level 2
 - 49 Der Interface-Baukasten PIA 6520
 - 54 Erlebnisbericht: Chung-Yuen 'boy' PASCAL auf dem TRS-80 Model I
 - 58 ATARI 400 ATARI 800
 - 61 Satellitenkommunikationssysteme
 - 67 APL 80 Interpreter für den TRS-80
 - 70 Was kennt man BASIC, Teil 29
 - 72 Neu der Model-Kiste der IBM-Rechner
 - 75 Computer und Kinder
 - 77 Aufruf zum Personal Computing Fest
 - 78 Ein Bericht bei Pickles & Trivet
 - 79 Applikations
 - 81 Microcomputer im Unterricht
 - 86 STRING User Information
 - 85 Hardware Nachrichten
 - 89 Software Nachrichten
 - 94 Messen und Auswertungen
 - 94 Neue Publikationen
 - 95 Kurz notiert
 - 98 Buchbesprechungen
 - 99 Microcomputer
 - 109 Computertrends und VIELNUTZ
 - 102 SIBAS 85, Teil 17
 - 107 PASCAL-Kurz
 - 118 Der SUPER-ELP P von ROXONIX

ELCOMP

HOFACKER-VERLAG

Ing. W. Hofacker GmbH
Tegernseer Straße 18

D-8150 Holzkirchen/Obb.

Die Fachzeitschrift für MICROCOMPUTER
Eine unentbehrliche Informationsquelle für alle Elektroniker

Microcomputer-Anwendungsbeispiele
Künstliche Intelligenz
Block-Strukturierte Programme
Datenverarbeitung im Kleinbetrieb
Club-Neuheiten
Computer und Kunst
Musik mit dem Computer
Monitore für 8080, 6800, 6502, Z 80,
SC/MP, 2650, 1802
Eigenbau-Computersysteme
Interface-Techniken
Microcomputer KITS

Neue Produkte
Betriebssysteme für Floppys
Programmiertechniken
Software-Quellen
Programmierbeispiele
Soziale Aspekte der Microcomputer-
technik
Technologische Neuheiten
Anwendungen in der Meß- und Regel-
technik
Anwendungen bei Funk-Amateuren

ELCOMP

Absender
Bitte deutlich ausfüllen

Vorname/Name

Beruf

Straße/Nr.

Plz Ort

ABSENDER:

.....
Name, Vorname

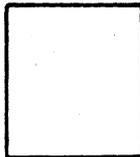
.....
Straße

()
.....

PLZ Ort

.....
Telefon

POSTKARTE

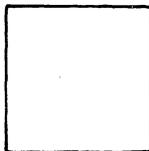


ELCOMP

MIKROCOMPUTER BOOK STORE

Tegernseerstr. 18

D-8150 Holzkirchen /Obb.



ELCOMP

Ing. W. Hofacker GmbH
Tegernseer Straße 18

D-8150 Holzkirchen

Das Abonnement wird automatisch verlängert
(Kündigung 8 Wochen z. Abonnement-Ablauf)

Leercassetten für

Microcomputer



C-10

Die ideale Cassettenlänge für Ihren Personalcomputer.
Praktisch – handlich und betriebssicher

Kassetten mit nur 10 Minuten Spieldauer (2 x 5 Minuten) haben sich zur Aufzeichnung von Daten im Mikrocomputerbereich bestens bewährt.

Vorteile der C-10 Computer Cassette vom HOFACKER Verlag:

- weniger Bandsalat
- kurze Rückspulzeiten
- schnelles Auffinden von Programmen
- bessere Gleichlauf Eigenschaften
- einfache Programmverwaltung

Die C-10 HOFACKER Datencassette bietet weiterhin:

- extrem hoch aussteuerbares Bandmaterial (Agfa)
- hochwertiges Cassettengehäuse, 5fach verschraubt
- Tefloneinlage für gute Laufruhe
- Staubsicheres Glasfenster

Die C-10 HOFACKER Datencassette wird seit 1978 speziell für Microcomputeranwender produziert. Die Cassetten bieten ein Höchstmaß an Betriebssicherheit bezüglich fehlerfreier Aufnahme und Wiedergabe.

Hier eine kurze Übersicht über die Anzahl der Bytes, die Sie auf eine C-10 Cassette abspeichern können:

Computer	Speichermöglichkeit	Computer	Speichermöglichkeit
ATARI 400/800	16K	APPLE	36K
Sharp MZ-80	32K	APPLE II	16K
AIM 65	16K	Heathkit	36K
Ohio Scientific	10K	Kansas City Std.	16K
TRS-80	16K	KIM-1	12K
TRS-80 Color Computer	24K	NASCOM	12K
Video Genie	16K	Exidy Sorcerer	12K
Sinclair ZX80/81	16K	SYM-1	12K

BESTELLSCHEIN

Menge	Beschreibung	Preis/DM	Gesamt
	1 Cassette	3,50	
	10 Cassetten	29,80	
	100 Cassetten	249,00	

Lieferanschrift

Name.....

Straße.....

Ort.....

Datum..... Unterschrift.....

HOFACKER

Ing. W. Hofacker GmbH
Tegernseerstr. 18
D-8150 Holzkirchen
Tel.: (0 80 24) 73 31

Weitere interessante Bücher von Hofacker:

Best.-Nr.	Titel	Preis/DM	Best.-Nr.	Titel	Preis/DM
Bücher in deutscher Sprache					
1	Transistor Berechnungs- u. Bauanleitungsbuch – 1	29,80	122	BASIC für Fortgeschrittene	39,00
2	Transistor Berechnungs- u. Bauanleitungsbuch – 2	19,80	123	IEC-Bus Handbuch	19,80
3	Elektronik im Auto	9,80	124	Programmieren in Maschinensprache mit CBM	19,80
4	IC-Handbuch, TTL, CMOS, Linear	19,80	127	Einführung i. die Microcomputer-Progr. mit 6800	49,00
5	IC-Datenbuch, TTL, CMOS, Linear	9,80	128	Programmieren mit dem CBM	29,80
6	IC-Schaltungen, TTL, CMOS, Linear	19,80	129	ELCOMP-Leser Programmierhandbuch	69,00
7	Elektronik Schaltungen	9,80	130	Programmierbeispiele für CBM	19,80
8	IC-Bauanleitungsbuch	19,80	131	COBOL für Anfänger	19,80
9	Feldeffekttransistoren	9,80	132	CP/M-Handbuch	19,80
10	Elektronik und Radio	19,80	133	Welches Betriebssystem brauche ich ?	19,80
11	IC-NF Verstärker	9,80	139	BASIC für blutige Laien	19,80
12	Beispiele Integrierter Schaltungen (BIS)	19,80	140	ZX81 Programmierhandbuch	29,80
13	HEH, Hobby Elektronik Handbuch	9,80	141	Programmierhandbuch für VC-20	29,80
14	IC-Vergleichsliste	29,80	Bücher in englischer Sprache		
15	Optoelektronik Handbuch	19,80	150	Care and Feeding of the Commodore PET	19,80
16	CMOS Teil 1, Einführung, Entwurf, Schaltbeispiele	19,80	151	8k Microsoft BASIC Reference Manual	19,80
17	CMOS Teil 2, Entwurf und Schaltbeispiele	19,80	152	Expansion Handbook for 6502 and 6800	19,80
18	CMOS Teil 3, Entwurf und Schaltbeispiele	19,80	153	Microcomputer Application Notes	29,80
19	IC-Experimentier Handbuch	19,80	154	Complex Sound Generation using the SN76477	19,80
20	Operationsverstärker	19,80	155	The First Book of 80-US (TRS-80)	19,80
21	Digitaltechnik Grundkurs	19,80	156	Small Business Programs	29,80
22	Mikroprozessoren, Eigenschaften und Aufbau	19,80	157	The First Book of Ohio Scientific	19,80
23	Elektronik Grundkurs, Kurzlehrgang Elektronik	9,80	158	The Second Book of Ohio Scientific	19,80
24	Microcomputer-Technik	29,80	159	The Third Book of Ohio Scientific	19,80
25	Hobby Computer Handbuch	29,80	160	The Fourth Book of Ohio Scientific	29,80
26	Mikroprozessor, Teil 2	19,80	161	The Fifth Book of Ohio Scientific	19,80
27	Microcomputer Software Handbuch	29,80	162	ATARI Games in BASIC	19,80
28	Lexikon + Wörterb. f. Elektr. u. Mikroprozessor LEM	29,80	163	The Peripheral Handbook	29,80
29	Mikrocomputer Datenbuch	49,80	164	ATARI-BASIC Learning by Using	19,80
30	Aktivtraining Mikrocomputer	49,80	1050	The Most Popular Subroutines in BASIC	24,80
31	57 Programme in BASIC	39,00	1053	Microprocessor Cookbook	24,80
32	ATARI BASIC Handbuch	29,80	1055	The BASIC Cookbook	24,80
33	Microcomputer Programmierbeispiele	19,80	1062	The A to Z Book of Computer Games	29,80
34	TINY-BASIC Handbuch	19,80	1070	Digital Interfacing with an Analog World	39,00
35	Der freundliche Computer	29,80	1071	The Complete Handbook of Robotics	29,80
103	Oszillographen-Handbuch	19,80	1076	Artificial Intelligence	29,80
104	1000 Elektronik Schaltungen	49,00	1085	24 Tested Ready to RUN Game Programs in BASIC	24,80
107	Praktische Antennentechnik	19,80	1088	Illustrated Dictionary of Microcomputer Terminology	35,00
108	SC/MP Mikrocomputer-Handbuch	29,80	1095	Programs in BASIC for Electronic Engineers	19,80
109	6502 Microcomputer Programmierung	29,80	1099	How to Build Your own Working 16-Bit Microc.	14,80
110	Programmierhandbuch für PET	29,80	1141	How to Build Your own Working ROBOT PET	29,80
111	Programmieren mit TRS-80	29,80	1160	1001 Things to do with Y.P.C.	29,80
112	PASCAL-Programmier-Handbuch	29,80	1169	The Giant Book of Computers	39,00
113	BASIC-Programmier-Handbuch	19,80	8029	Z-80 Assemblerhandbuch	29,80
114	Der Microcomputer im Kleinbetrieb	39,80	8042	6500 Software Manual	19,80
115	6809 Programmier Handbuch	49,00	8043	6500 Hardware Manual	19,80
116	Einführung 16-Bit Microcomputer	29,80	8048	BASIC Software Vol. VI	199,00
117	FORTRAN für Heimcomputer	19,80	8049	BASIC Software Vol. VII	159,00
118	Programmieren in Maschinensprache mit dem 6502	49,00	8050	BASIC Software Vol. I	99,00
119	Programmieren in Maschinensprache (Z80)	49,00	8051	BASIC Software Vol. II	99,00
120	Anwenderprogramme für TRS-80 u. Video Genie	29,80	8052	BASIC Software Vol. III	149,00
121	Microsoft BASIC-Handbuch	29,80	8053	BASIC Software Vol. IV	39,00
			8054	BASIC Software Vol. V	39,00

HOFACKER