Brückmann · Gerits · Wiens

Das große

# Drucker-Buch

Drucker-Anwendung
mit COMMODORE Computern
für Anfänger, Fortgeschrittene
und Profis

EIN DATA BECKER BUCH

Brückmann · Gerits · Wiens

Das große

# Drucker-Buch

Drucker-Anwendung
mit COMMODORE Computern
für Anfänger, Fortgeschrittene
und Profis

EIN DATA BECKER BUCH

ISBN 3-89011-020-7

Copyright (C) 1984 DATA BECKER GmbH
Merowingerstr. 30
4000 Düsseldorf

Alle Rechte vorbehalten. Kein Teil dieses Buches darf in irgendeiner Form (Druck, Fotokopie oder einem anderen Verfahren) ohne schriftliche Genehmigung der DATA BECKER GmbH reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

Wichtiger Hinweis!

Die in diesem Buch wiedergegebenen Schaltungen, Verfahren und Programme werden ohne Rücksicht auf die Patentlage mitgeteilt. Sie sind ausschließlich für Amateur- und Lehrzwecke bestimmt und dürfen nicht gewerblich genutzt werden.

Alle Schaltungen, technische Angaben und Programme in diesem Buch wurden von den Autoren mit größter Sorgfalt erarbeitet bzw. zusammengestellt und unter Einschaltung wirksamer Kontrollmaßnahmen reproduziert. Trotzdem sind Fehler nicht ganz auszuschließen. DATA BECKER sieht sich deshalb gezwungen, darauf hinzuweisen, daß weder eine Garantie noch die juristische Verantwortung oder irgendeine Haftung für Folgen, die auf fehlerhafte Angaben zurückgehen, übernommen werden kann. Für die Mitteilung eventueller Fehler sind die Autoren jederzeit dankbar.

*INHALT*


| Kapitel | | Seite |
|---|---|---|
| 1.0 | EINFÜHRUNG | 1 |
| 2.0 | SCHNITTSTELLEN | 3 |
| 2.1 | Der IEC-Bus des C64 | 3 |
| 2.1.1 | Grundsätzliches | 4 |
| 2.1.2 | Adressierung | 5 |
| 2.1.3 | Datentransfer | 7 |
| 2.2 | Die CENTRONICS-Schnittstelle | 11 |
| 2.3 | Die V24-Schnittstelle | 18 |
| 3.0 | DIE SEKUNDÄRADRESSEN | 25 |
| 3.1 | VC-1525 / GP-100VC / MPS-801 | 27 |
| 3.2 | VC-1526 | 29 |
| 3.3 | EPSON mit DATA BECKER-Interface | 33 |
| 4.0 | SCHREIBMASCHINE AM USER-PORT | 36 |
| 5.0 | PROGRAMMSAMMLUNG FÜR VC-20 UND C64 | 43 |
| 5.1 | Kleine Hilfsprogramme für alle Tage | 44 |
| 5.1.1 | Device Not Present? | 44 |
| 5.1.2 | Formatierung numerischer Daten und Alphastrings | 46 |
| 5.1.3 | Formatierte Programmlistings | 56 |
| 5.2 | Textverarbeitung | 69 |
| 5.2.1 | MINITEXT - ein einfacher Texteditor | 69 |
| 5.2.2 | Geht's auch größer? | 86 |
| 5.3 | Hardcopy | 96 |
| 5.3.1 | Texthardcopy | 97 |
| 5.3.2 | Grafik-Hardcopy | 103 |
| 5.4 | Grafik mit und ohne Einzelnadelsteuerung | 121 |
| 5.4.1 | Grafik - auch ohne Einzelnadelansteuerung | 122 |
| 5.4.2 | Grafik - mit Einzelnadelansteuerung | 129 |
| 5.4.3 | Darstellung dreidimensionaler HIRES-Grafik | 142 |
| 6.0 | DRUCKPRINZIPIEN | 152 |
| 6.1 | Der Typenraddrucker | 153 |

Kapitel Seite


6.2 Der parallele Matrixdrucker 155
6.3 Der serielle Matrixdrucker 157
7.0 DAS BETRIEBSSYSTEM DES MPS-801 160
7.1 Der Prozessor 8039 161
7.2 Blockschaltbild des MPS-801 165
7.3 Das Rom-Listing zum MPS-801 169
8.0 DER PRINTER-PLOTTER VC-1520 251
8.1 Arbeitsweise und Nutzungsmöglichkeiten 251
8.2 Die Sekundäradressen des VC-1520 260
8.3 Konstruktion von Figuren aller Art 289
8.3.1 Quadrat 289
8.3.2 Rechteck 290
8.3.3 Parallelogramm 299
8.3.4 Dreiecke 1 301
8.3.5 Kreisbögen 311
8.3.6 Dreiecke 2 320
8.4 Darstellung von Funktionen 322
8.5 Kleine Besonderheiten mit dem VC-1520 338
8.6 Statistik mit dem VC-1520 347
8.7 Hilfsprogramm für den VC-1520 358

## *1.0 EINFÜHRUNG*

Als Sie das erste Mal von diesem Buchtitel hörten, haben Sie sicher gedacht: "Wie um alles in der Welt kann man nur über ein so an sich problemloses Gerät wie einen Drucker ein ganzes Buch schreiben?!"
Nun halten Sie es in der Hand, und wie Sie sehen, ist es ganz schön umfangreich geworden. Was steht nun alles darin?

Gestatten Sie uns eine Frage: Würde es Ihnen in den Sinn kommen, ein Buch mit dem Titel 'Das große Handmixerbuch' zu schreiben? Sie werden das Ansinnen entrüstet zurückweisen. Warum eigentlich? Sie glauben ja gar nicht, was es darüber zu berichten gibt. Sicher, einfache Anwendungen wie Sahne schlagen brauchen nicht erwähnt werden, aber was haben Sie falsch gemacht, wenn Ihnen beim Rühren eines Hefeteigs mittels der Knethaken der Teig aus der Schüssel hüpft. Auf die Lösung werden Sie nicht so ohne weiteres kommen, weil Sie, solange es gut ging, das Prinzip nicht ergründet haben. Wir wollen Ihnen die Lösung verraten: Sie haben die beiden Knethaken vertauscht. Statt gegensinnig abwärts drehen sie nun gegensinnig aufwärts. Und was es erst alles über Zubehör zu schreiben gäbe.

Sehen Sie unser Buch ruhig unter diesem Aspekt. Es ist die ganze Bandbreite von der Steckdose bis zum Betriebssystem enthalten. Dazwischen liegt eine ganze Welt für alle Interessengruppen.
Äußerst interessant dürfte für die meisten Leser die umfangreiche Programmsammlung sein. Aber auch, wer z.B. die eckigen Klammern gegen Umlaute tauschen will, findet leicht die Stelle, wo er es machen muß.

Wir haben uns bemüht, sowohl bei den Beschreibungen als auch

bei den Programmen, die ganze Palette der für den C64-Bereich angebotenen Drucker zu berücksichtigen, so daß auch für Ihren Drucker einiges dabei sein dürfte.

Das Kapitel 8 ist ganz dem Printer-Plotter VC-1520 gewidmet. Hätte dieses Gerät in die übrigen Abschnitte Eingang finden wollen, hätten diese allzusehr aufgebläht werden müssen. Wir wollten ihn Ihnen aber nicht vorenthalten.

Bleibt uns jetzt nur noch übrig, Ihnen bei der Lektüre viel Freude und fette Beute zu wünschen.

Ihre Autoren

## *2.0 SCHNITTSTELLEN*

Es ist kaum ein Thema in der 'Homecomputerei' für einen Techniker so reizvoll wie das der Schnittstellen. Hier kann er sich austoben und eine Form in die andere überführen.
Warum dazu überhaupt die Notwendigkeit besteht, liegt auf der Hand: Der Drucker mit gerade den Fähigkeiten, die Sie wünschen, ist als Originalgerät nicht zu haben. Ebensoweit wie die Wunschvorstellungen vom idealen Gerät, weichen auch die Drucker selbst voneinander ab. Sind Sie der Meinung, Sie haben den richtigen gefunden, paßt er womöglich nicht zu Ihrem Rechner.
"Ist doch klar", wird Ihnen der Fachhändler sagen, "Ihr Rechner hat die Schnittstelle X und der Drucker den Anschluß Y. Das weiß doch jeder, daß das nicht funktionieren kann."
Ist das wirklich so klar?
In diesem Kapitel haben wir uns die Aufgabe gestellt, Ihnen die überhaupt in Frage kommenden Schnittstellen einmal ausführlich zu erläutern. Am Ende können Sie sich Ihr eigenes, fundiertes Urteil bilden, wenn es um die Verträglichkeit oder Anpaßbarkeit der verschiedenen Interfaces geht.

Natürlich beginnen wir, wie könnte es auch anders sein, wenn es um den C64 geht, mit dem IEC-Bus.

### *2.1 Der IEC-Bus des C64*

Der serielle IEC-Bus des C64 ist nicht identisch mit der Einrichtung gleichen Namens, über die 1974 in den USA etliche Hersteller eine Einigung erzielten, und die unter der Bezeichnung IEC-625 1 Eingang in die Deutsche Industrie-Norm fand.

Allerdings wird hier im Kleinen in etwa die Logik des 'großen' IEC-Bus nachvollzogen, was bei der geringen Anzahl der benutzten Leitungen sicher nicht einfach ist.
Aber sehen Sie selbst.

### *2.1.1 Grundsätzliches*

Der serielle IEC-Bus hat sechs Leitungen (gegenüber 24 beim parallelen). Diese sind im einzelnen:

1 SRQ Service ReQuest. Hat ein Gerät irgendeine Aufgabe erledigt und braucht nun neue Daten oder hat welche abzugeben oder verlangt sonst irgendeine Aktion, so kann es das dem Rechner über diese Leitung mitteilen.
Bei Commodore wird diese Funktion nicht verwendet.

2 GND

3 ATN ATeNtion. Immer, wenn der Rechner einen Befehl an ein Gerät übermitteln will, aktiviert er diese Leitung. Dadurch soll erreicht werden, daß alle am Bus angeschlossenen Geräte diesen Befehl mitbekommen, da ja von vornherein noch nicht feststeht, welches Gerät gemeint ist. Das stellt sich erst bei der Übermittlung der Geräteadresse heraus, weshalb diese auch immer zuerst geschickt wird, damit sich die anderen Geräte wieder vom Bus 'abhängen' können.

4 CLK CLocK. Da die Daten nicht byteweise, also 8 Bit auf einmal, sondern bitseriell übertragen werden, gibt der Sender jedem Bit einen Taktimpuls auf der Leitung CLK mit auf den Weg, womit die Gültigkeit der Datenleitung angezeigt wird.

5 DATA ist die Datenleitung, über die ein Datenbyte mit dem niedrigstwertigen Bit voran seriell geschoben wird. Darüberhinaus hat sie aber noch weitere Aufgaben zu erfüllen, wie Sie später sehen werden.

6 RES RESet. Ein Low-Pegel auf dieser Leitung veranlaßt alle angeschlossenen Geräte, in die Ruhestellung zu gehen, also denselben Zustand anzunehmen, als ob sie gerade eingeschaltet worden wären.

Die folgenden Ausführungen sollen nur der Beleuchtung des Prinzips dienen. Zur Demonstration nehmen wir die folgenden Basic-Zeilen:

```
10 OPEN 1,4
20 PRINT#1,"A"
```

Es soll also einfach der Buchstabe A auf einem Drucker mit der Geräteadresse 4 ausgegeben werden. Auslöser für die Aktionen auf dem Bus ist nur die Zeile 20. Von dem OPEN merkt der Drucker noch nichts, da, anders als bei der Floppy, hiermit kein Dateiname verbunden ist, der übermittelt werden müßte.

Die im folgenden Text den Abschnitten vorangestellten Zahlen entsprechen denen in den Timing-Diagrammen über der X-Achse eingetragenen.
Es geht los mit der

### *2.1.2 Adressierung*

1
ATN wird Low zum Zeichen, daß ein Befehl folgt.

2

Als Reaktion hierauf muß nun das Gerät innerhalb 1ms die Datenleitung nach Low ziehen. Der Rechner signalisiert darauf mit CLK=High, daß ein Byte bereitsteht.

3

Solange das Gerät noch nicht bereit ist, Daten aufzunehmen, hält es die Leitung DATA Low. Ist es soweit, wird DATA=High.

3-4

Nun kommt die Geräteadresse über die Leitung DATA, und zwar mit jedem High-Impuls auf der Leitung CLK ein Bit.
Dieses gesendete Byte hat den Wert 36 ($24). Hierin ist sowohl die Geräteadresse, nämlich 4, als auch die verlangte Aktion, nämlich LISTEN (d.h. auf Empfang schalten) enthalten.

5

Innerhalb von wiederum 1ms muß der Drucker jetzt die Leitung DATA nach Low bringen. Das dient zum einen dazu, den Empfang zu quittieren, zum anderen soll das Senden weiterer Daten solange verhindert werden, bis das Gerät zum erneuten Empfang in der Lage ist.

Sie sehen hier schon, daß DATA drei Aufgaben hat, nämlich
1. Übermittlung der Daten
2. Aufhalten der Daten
3. Quittieren der Daten

### *2.1.3 Datentransfer*

6

Der Datentransfer beginnt damit, daß der Sender CLK hochsetzt. Damit zeigt er an, daß ein Byte zum Transfer bereitsteht.

7

Seine Bereitschaft zeigt der Empfänger durch Hochsetzen von DATA an.

8-9

Mit jedem CLK-Impuls klappert nun ein Bit über die Leitung DATA.

9

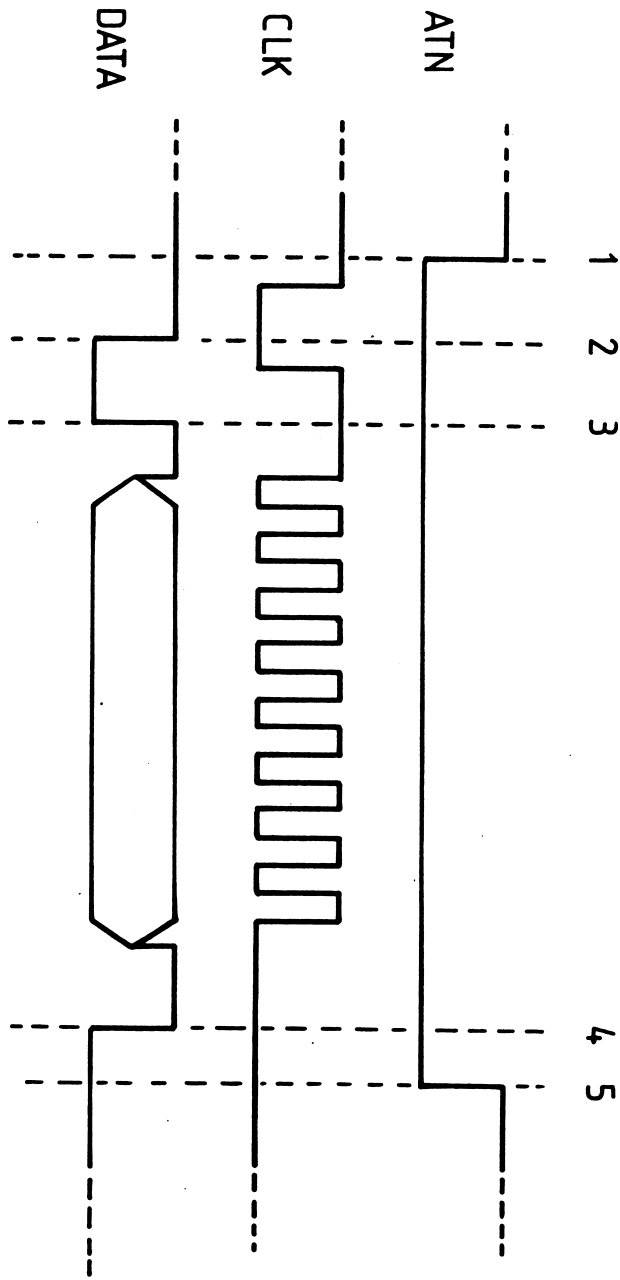
Das komplette Byte wird vom Empfänger mit DATA=Low quittiert. Übrigens war das gerade unser A aus Zeile 20.

11-12

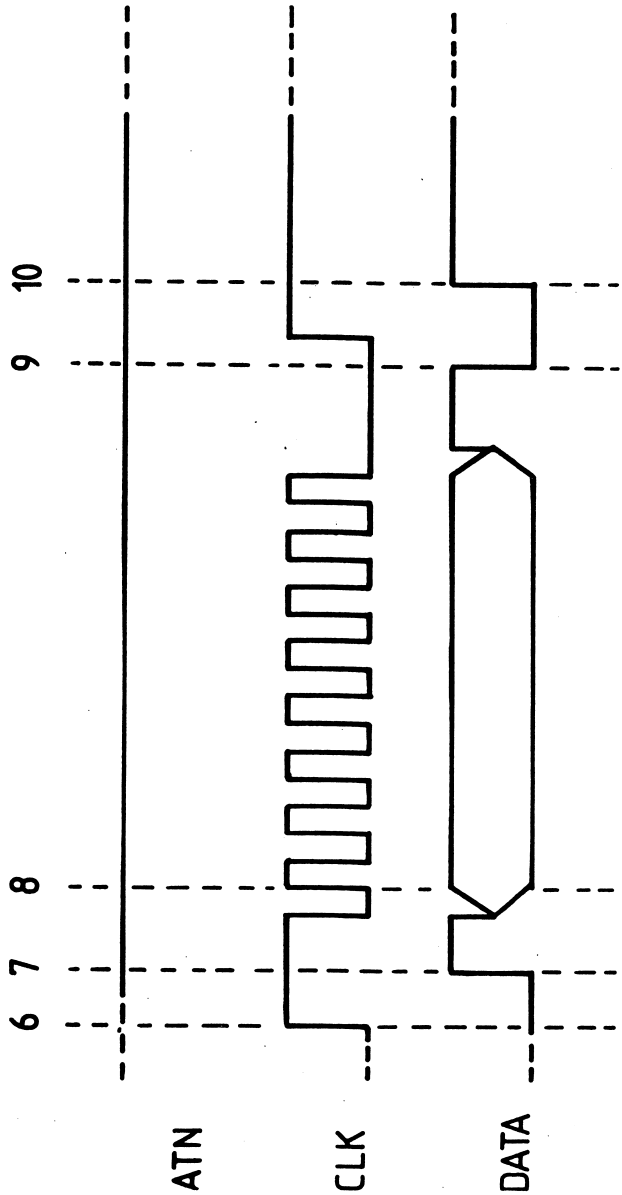
Auf die gleiche Weise folgt nun der abschließende chr$(13).

Nun bekommt die Leitung CLK eine weitere Aufgabe:
Üblicherweise wird das letzte Byte eines Datentransfers besonders gekennzeichnet, damit der Empfänger schon einmal mit der Verarbeitung beginnen kann. Mit dieser Funktion wird CLK betraut.

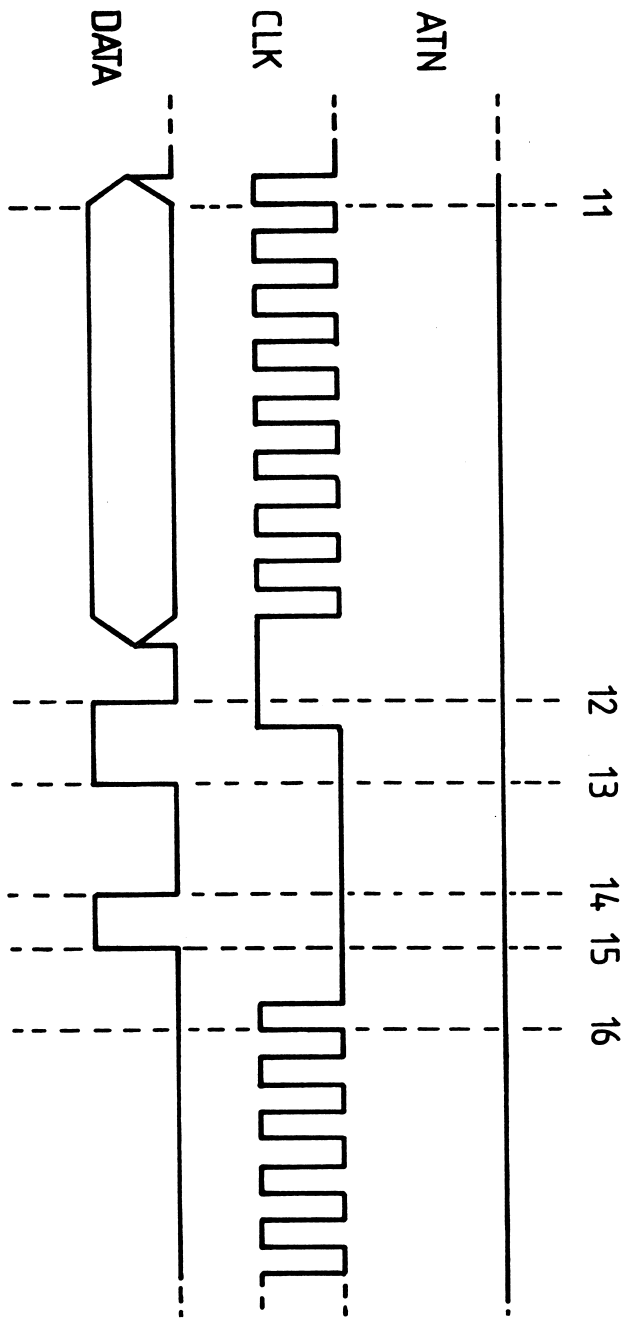
## Adressierung



## Datentransfer



**Letztes Datenbyte**



13-14

Die Vereinbarung sah bis jetzt so aus, daß nach Hochsetzen der Leitung DATA durch den Empfänger der Sender innerhalb von 0.2ms durch CLK=Low den Datentransfer einleitete. Nun bleibt aber CLK länger High, was schlicht bedeutet, daß das zuvor übertragene Byte das letzte war.
Auch dieser Umstand muß vom Empfänger mit DATA=Low quittiert werden.

15

Sobald DATA wieder High wird, überträgt der Sender ein Leerbyte (chr$(0) ab Punkt 16). Danach geht der Bus in den Ruhezustand.

Die Deadressierung erfolgt analog Punkt 1-5, wobei das Datenbyte den Wert 63 ($3F) aufweist. Das ist das sogenannte UNLISTEN-Kommando, was bedeutet, daß alle Geräte, die bisher als Datenempfänger arbeiteten, sich wieder von Bus abhängen können, da keine weiteren Daten mehr folgen.

Wie Sie sehen, funktioniert es wunderbar. Den gravierenden Nachteil, den ein serieller Bus systembedingt gegenüber dem parallelen hat, nämlich die wesentlich geringere Geschwindigkeit, wird durch drastisch verkürzte Wartezeiten (1ms gegenüber 64ms) teilweise aufgefangen.

## *2.2 Die CENTRONICS-Schnittstelle*

Die wohl bei Druckern weitestverbreitete Anschlußart ist die nach einem Vorschlag von Centronics. Diese sehr sichere Methode der Datenübertragung wurde von nahezu allen anderen Druckerherstellern aufgegriffen und mehr oder weniger vollständig in die eigenen Produkte implementiert.

Für Sie als Anwender ist es natürlich von besonderem Vorteil, daß Drucker mit dieser Anschlußart in der Regel auch die preiswerteste Variante eines bestimmten Modells darstellen. Das Problem besteht eigentlich nur darin, wie Sie ein solches Gerät an Ihren C64 anschließen können.

Die mit Abstand preiswerteste Methode ist in dem Buch '64 Tips & Tricks' ausführlich beschrieben. Sie benötigen dazu nur ein Verbindungskabel, um den Drucker am User-Port anzuschließen, und ein wenig Treibersoftware. Anhand der folgenden Beschreibung der Schnittstelle dürfte es allerdings für Sie nicht allzu schwer sein, einen derartigen Anschluß selbst zu realisieren, wenn Sie sich in der Handhabung des User-Port ein wenig auskennen.

Etwas kostspieliger, aber deutlich bequemer, ist ein Interface, welches den IEC-Bus des C64 in eine dem Drucker genehme Form umwandelt. Dieses ist in der Regel ein handliches Kästchen, das einfach hüben und drüben eingestöpselt wird, und schon ist die Sache perfekt. Diese Lösung hat allerdings einen Haken: Sie ist nur geeignet, wenn Sie sich ausschließlich auf die Wiedergabe von 'normalen' alphanumerischen Zeichen beschränken. Die zahlreichen Sonder- und Steuerzeichen, die Ihr C64 kennt, werden Sie mit einer derartigen Konfiguration nicht reproduzieren können, wenn der verwendete Drucker diese Zeichen nicht von Haus aus in seinem Zeichenvorrat enthält. Ansonsten ist die Sache aber recht brauchbar, zumal Sie keinerlei Treibersoftware benötigen. Vom Basic her wird der Drucker wie ein gewöhnliches Commodore-Gerät angesprochen. Was eine möglicherweise notwendige Umwandlung der Buchstabencodes betrifft, lesen Sie bitte im Abschnitt 3.3 die Erläuterungen zur Sekundäradresse 1.

Eine recht runde Lösung, was sowohl die Handhabung als auch die Preiswürdigkeit betrifft, ist ein Einbauinterface für die EPSON-Drucker. Mit einem derartigen Interface, wie auch unter 3.3 beschrieben, haben Sie gleich mehrere Fliegen mit einer Klappe geschlagen. Erstens wird es, wie auch ein Commodore-Drucker, einfach am IEC-Bus des C64 angeschlossen, das heißt leichte Handhabung von Basic. Zweitens ist es im Drucker eingebaut, was ein extern herumbaumelndes Kästchen überflüssig macht. Drittens, und das ist das Beste, kann der Drucker mit Hilfe dieses Interfaces die Sonder- und Steuerzeichen samt und sonders wiedergeben. Darüberhinaus ist das Gerät weitestgehend mit dem Drucker VC-1525 kompatibel, so daß auch auf jenen Drucker angelegte Standardsoftware ohne größere Probleme laufen dürfte.

Nun aber, damit Sie sich Ihre eigenen Gedanken machen können, zum Kern der Sache, nämlich der Centronics-Schnittstelle.
Sie ist in der Regel als 36-polige Cinch-Steckverbindung ausgeführt, von der aber nur gut die Hälfte aller Leitungen wirklich Signale führen. Beschränkt man sich auf die zur Datenübertragung absolut notwendigen, kann man auch mit elf Leitungen auskommen. Die anderen dienen zur Signalisierung gewisser Zustände im Drucker und sind auch nicht bei allen Herstellern am Stecker verfügbar.

Wir beginnen mit der Pinbelegung der Centronics-Buchse.

1 STROBE. Im Ruhezustand ist diese Leitung High. Ein Low-Impuls von wenigstens 1 Microsekunde signalisiert dem Drucker, daß ein gültiges Datenbyte anliegt.

2-9 DATA 0-7. Über diese Leitungen wird dem Drucker ein Byte angeboten.

10 ACKNLG. Mit einem Low-Impuls von ca. 12 Microsekunden quittiert der Drucker den Empfang des Datenbytes. Der Sender kann nun die Datenleitungen mit einem neuen Byte beaufschlagen (aber noch keinen STROBE hinterherschicken!).
Diese Leitung ist durchaus verzichtbar, da sie zur fehlerfreien Steuerung des Datenflusses nicht unbedingt notwendig ist. Die sog. reine BUSY-Steuerung reicht vollkommen, d.h. das unter 11 beschriebene Signal ist wesentlich wichtiger, weshalb auch die meisten Interfaces von ACKNLG keinen Gebrauch machen.

11 BUSY. Diese Leitung hält der Drucker High, wenn er nicht in der Lage ist, ein neues Datenbyte aufzunehmen. Die Gründe hierfür sind unterschiedlich. So kann der Drucker z.B. mit der Ausgabe einer zuvor gepufferten Zeile beschäftigt sein oder das Papier ist zu Ende oder der Drucker ist OFFLINE usw.
Der Datensender sollte tunlichst erst dann Daten mit STROBE an den Drucker übergeben, wenn BUSY=Low ist.

12 PE. Mit einem High auf dieser Leitung signalisiert der Drucker das Papierende.

16 GND. Dies ist der Bezugspegel für alle Signale.

18 +5V. Hiermit können externe Interfaces mit Spannung versorgt werden. Diese Stromversorgung steht aber nicht bei allen Druckern zur Verfügung. Sie sollten daher im Bedarfsfalle im Handbuch nachschauen.

31 INIT. Ein Low-Impuls von wenigstens 50 Microsekunden Länge auf dieser Leitung bringt den Drucker in die Grundstellung, als ob Sie ihn gerade eingeschaltet

hätten.

32 ERROR. Mit einem Low-Pegel auf dieser Leitung teilt der Drucker mit, daß er entweder kein Papier hat oder OFFLINE ist oder ein sonstiger Fehler auftrat.
Aus der Kombination der Signale PE und ERROR können Sie die Ursache für BUSY herleiten und programmtechnisch auswerten, wenn Sie die Möglichkeit haben, diese Signale in Ihren Rechner zu bekommen. Allerdings wird davon nur selten Gebrauch gemacht.

36 SLCT IN. Ein Low-Pegel auf dieser Leitung ermöglicht es dem Drucker erst, zu arbeiten. Gewöhnlich kann dieser Pegel auch im Innern des Gerätes mit Hilfe eines Schalters erzeugt werden.
Sollte sich Ihr Drucker einmal überhaupt nicht rühren wollen, so schauen Sie doch einmal danach. Möglicherweise haben Sie beim Verstellen irgendeines anderen Schalters diesen aus Versehen mit umgelegt.

Die nicht aufgeführten Leitungen haben entweder keine Funktion oder liegen auf Masse oder erfüllen andere, gerätespezifische Aufgaben. Vergewissern Sie sich hierüber im zum jeweiligen Drucker gehörigen Handbuch.

Es ist ohne weiteres möglich, das Gerät nur mit den Leitungen 1-9, 11 und 16 zu betreiben. Die übrigen müssen ohnehin nicht bei allen Fabrikaten im gleichen Maße vertreten sein.

Das nachfolgende Timing-Diagramm soll die Abläufe auf der Schnittstelle noch einmal veranschaulichen.

1
Wenn BUSY=Low ist, kann ein Byte auf die Datenleitungen gelegt werden.
Frühestens 0.5 Microsekunden danach darf erst der

2
STROBE ausgelöst werden, der eine Dauer von mindestens 1 Microsekunde haben sollte.
Zum Zeichen, daß das Byte zumindest angekommen ist, wird BUSY=High.

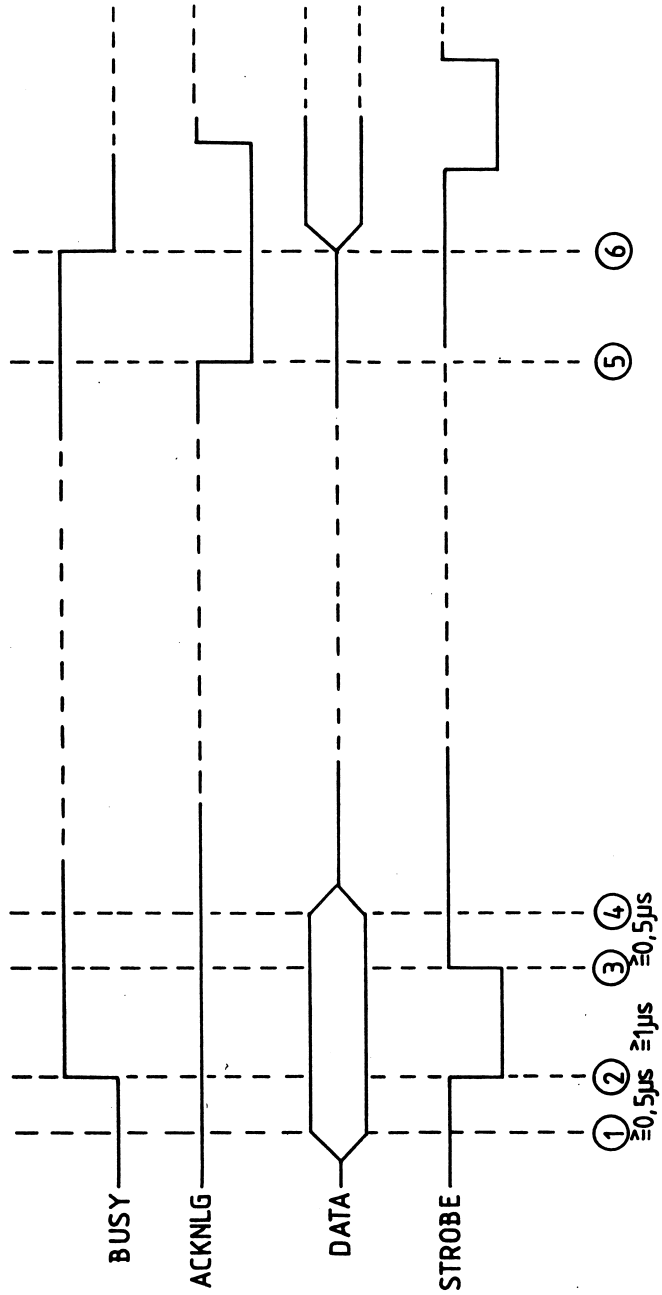
3
Nach der Wegnahme von STROBE müssen bis zum Zeitpunkt

4
noch wenigstens 0.5 Microsekunden verstreichen.
Erst jetzt dürfen die Datenleitungen ihren Zustand ändern.

5
Die Zeit, die bis hierhin benötigt wird, kann stark schwanken.
Sie hängt davon ab, was der Drucker inzwischen zu tun hat, d.h. ob er nur das empfangene Zeichen zu puffern braucht oder ob es das letzte Zeichen einer Zeile war, woraufhin er die gesamte Zeile ausdrucken muß.

6
Jetzt können schon neue Daten auf die Leitungen gelegt werden.
Wenn BUSY=Low ist, kann auch der STROBE ausgelöst werden, und die gleiche Sequenz läuft wieder ab.

**Timing-Diagramm einer Centronics-Schnittstelle**



## *2.3 Die V24-Schnittstelle*

Hierbei handelt es sich um eine ebenfalls sehr gebräuchliche Verbindungslogik zweier Geräte miteinander. Eine derartige Schnittstelle ist recht universell und wird in der Datenfernübertragung vorzugsweise eingesetzt, da sie sowohl von ihrer physikalischen Anlage als auch von ihrer Logik her als sehr sicher gilt.
Gewöhnlich ist eine V24-Strecke bidirektional, d.h. die Daten können in beide Richtungen fließen. Im Falle eines Druckers ist natürlich nur die Übertragung in eine Richtung notwendig, wenngleich auch bestimmte Steuerleitungen, ähnlich wie bei der zuvor beschriebenen Centronics-Schnittstelle, auch in die Gegenrichtung gehen.

Der Grund, weshalb wir uns hier ausführlich mit der V24-Schnittstelle (auch unter RS-232C bekannt) befassen, ist der, daß, wie bereits erwähnt, zum einen dieses Interface sehr verbreitet ist, zum anderen, weil die Bedienung bei Verwendung der RS-232-Cartridge von Commodore, da vom Betriebssystem des C64 unterstützt, eigentlich problemlos wäre, wenn nicht, und das ist der dritte Grund, weshalb wir dieser Schnittstelle einen eigenen Abschnitt widmen, wenn nämlich nicht die Parametrisierung und der Anschluß für den technischen Laien so fürchterlich undurchsichtig wäre.

Das Problem besteht in der Regel nicht nur darin, daß die richtigen Übertragungsparameter gewählt werden (was das ist, sehen Sie später), sondern auch in der Tatsache, daß zwei Geräte mit dieser Schnittstelle wohl manchmal mit einer einfachen eins-zu-eins-Verbindung betrieben werden können, oft aber auch nicht.
Auch wie dieses Problem zu lösen ist, werden wir Ihnen zeigen, damit Sie nicht, was oft vorkommt, vorzeitig

entmutigt werden, da Sie nicht wissen, ob der Fehler im Kabel oder in der Logik liegt, wenn es nicht klappt.

In den folgenden Ausführungen beziehen wir uns auf die RS-232-Cartridge VC-1011 von Commodore, da Sie wohl nur mit dieser zu tun haben werden.
Der Anschluß erfolgt über eine 25-polige Steckbuchse vom Typ D-Subminiatur. Das Gegenstück dazu erhalten Sie, falls Sie sich selbst ein Verbindungskabel herstellen wollen, bei jedem Elektronik-Fachhändler oder Computerladen.

Die nachfolgende Pinbeschreibung dieser Verbindung führen Sie sich bitte genau zu Gemüte. Insbesondere ist es wichtig, daß Sie sich die Datenrichtung merken, in der die jeweiligen Signale fließen, damit Sie beim Anschluß eines Druckers, nachdem Sie auch dessen Pinbelegung studiert haben, nicht etwa zwei Signalausgänge miteinander verbinden. Das führt zwar in der Regel keinen Schaden herbei, aber Sie suchen sich dämlich, wenn die Verbindung nicht funktioniert.
Aber keine Angst, auch dazu finden Sie Hinweise. Beginnen wir also mit der Pinbelegung der RS-232-Cartridge. Es werden hier nur die relevanten Signale aufgeführt. Die nicht beschriebenen Pins lassen Sie bei der Verdrahtung unbedingt frei.

2 TXD Transmitted Data. Signalausgang
Über diese Leitung werden die Nutzdaten bitseriell hinausgeschoben.

3 RXD Received Data. Signaleingang
Die vom angeschlossenen Gerät gesendeten Daten werden hier hereingeholt.
Bei Anschluß eines Druckers wird diese Leitung in der Regel nicht belegt.

4 RTS Request To Send. Signalausgang
Diese Leitung wird aktiviert (>3V), wenn eine Eingabe über die Schnittstelle erwartet wird.
Das ist bei Druckbetrieb in der Regel nicht der Fall.
Im 3-Draht-Betrieb ist dieses Signal immer aktiv.

5 CTS Clear To Send. Signaleingang
Nur wenn diese Leitung vom angeschlossenen Gerät aktiviert wird (>3V), kann der C64 Daten senden.
Im Druckerbetrieb muß diese Leitung vom angeschlossenen Gerät unbeding versorgt werden, um den Datenfluß zu steuern, da sonst Zeichen verloren gehen könnten.
Eine Ausnahme bildet der 3-Draht-Betrieb (s.u.) bei extrem niedrigen Übertragungsgeschwindigkeiten.

6 DSR Data Set Ready. Signaleingang
Diese Leitung wird vom Betriebssytem des C64 sozusagen als Anwesenheitsnachweis interpretiert, d.h. ob ein Gerät überhaupt an der Schnittstelle angeschlossen und eingeschaltet ist.
Ein Betrieb, gleichgültig ob Senden oder Empfangen, ist nur möglich, wenn dieses Signal einen Pegel von >3V aufweist.
Eine Ausnahme bildet auch hier wieder der 3-Draht-Betrieb.

7 GND Betriebsmasse. Bezugspegel für alle Signale.

8 DCD Data Carrier Detected. Signaleingang
Dieses Signal hat annähernd die gleiche Bedeutung wie 6. Sie verbinden es am besten im Stecker mit Pin 20.

20 DTR Data Terminal Ready. Signalausgang
Diese Leitung ist das Gegenstück zu 6. Beim ersten

OPEN-Befehl mit der Geräteadresse 2 wird dieses Signal aktiviert.

Die Übertragungsparameter werden rechnerseitig im OPEN-Befehl festgelegt, druckerseitig in der Regel über Schalter. Es ist also als erstes darauf zu achten, daß diese Parameter hüben wie drüben übereinstimmen.

Als Parameter kommen in Frage:

1. Die Baudrate.
   Diese bestimmt die Übertragungsgeschwindigkeit in Bit/Sekunde und ist die wohl wichtigste Festlegung, denn wenn diese nicht auf beiden Seiten übereinstimmt, läuft gar nichts.

2. Die Anzahl der Datenbits.
   Dieser Parameter legt fest, aus wievielen Bits ein Byte besteht. Beim Anschluß eines Druckers dürften es entweder 7 oder 8 sein.

3. Anzahl der Stopbits.
   In der Regel kommen hier nur 1 oder 2 in Frage. Diesen Parameter können Sie nach Geschmack gestalten, nur achten Sie auf Übereinstimmung.

4. Parität.
   Hier haben Sie die Auswahl zwischen keiner, gerader oder ungerader Parität.
   Wählen Sie ruhig keine Parität, denn das erspart Probleme bei 8 Datenbits, und Übertragungsfehler, die erkannt werden müßten, sind ohnehin nicht zu erwarten.
   Falls Sie nicht wissen, was Parität bedeutet:
   Ein Datenbyte besteht bekanntlich aus 0-Bits und 1-Bits.

Zur Prüfung, ob ein Byte unverstümmelt übertragen wurde, fügt man noch ein weiteres Bit hinzu, nämlich das sog. Paritätsbit. Dieses soll die Anzahl der 1-Bits auf eine gerade oder ungerade Zahl ergänzen, je nachdem, wie der Parameter gesetzt wurde.
Auf der Empfangsseite werden die 1-Bits gezählt und mit der Parametereinstellung verglichen. Stimmen sie nicht überein, handelt es sich um einen Übertragungsfehler, den der Drucker gewöhnlich durch Ignorieren des Zeichens straft.
Sollte es also bei Ihnen vorkommen, daß Sie alles richtig eingestellt und miteinander verbunden haben, und es rührt sich dennoch nichts, könnte Ihnen besagte Parität einen Streich spielen. Schauen Sie dort noch einmal nach und schalten Sie die Paritätsprüfung einfach aus.

Um Ihnen die Parametereistellung auf der Rechnerseite zu vereinfachen, haben wir die gebräuchlichsten Einstellungen einigen Variablen zugeordnet und alles Überflüssige weggelassen. Wenn Sie tiefergehend interessiert sind, können Sie im Buch '64 intern' mehr erfahren.

1. Baudrate: Variable br=

| | |
|---|---|
| 50 | 1 |
| 110 | 3 |
| 150 | 5 |
| 300 | 6 |
| 600 | 7 |
| 1200 | 8 |
| 2400 | 10 |

2. Datenbits: Variable db=

| | |
|---|---|
| 7 | 32 |
| 8 | 0 |

3. Stopbits: Variable sb=

| | |
|---|---|
| 1 | 0 |
| 2 | 128 |

4. Handshake: Variable hs=

| | |
|---|---|
| nein | 0 |
| ja | 1 |

Auf die Variable hs werden wir noch zu sprechen kommen. Nachdem Sie aus der Aufstellung das Passende ausgewählt haben, sieht der OPEN-Befehl so aus:

```
OPEN 1,2,0,chr$(br+db+sb)+chr$(hs)
```

Zur Herstellung des Kabels gehen Sie so vor:
Verbinden Sie den Pin 7 beider Stecker (Drucker- und Rechnerseite) miteinander.
Verbinden Sie, auf jeder Seite für sich, die Pins 4-5 und 6-8-20 miteinander. Damit überlisten Sie zunächst das Handshake und gaukeln beiden Seiten Betriebsbereitschaft vor.
Finden Sie nun gemäß Druckerhandbuch heraus, welcher Pin der Dateneingang ist (2 oder 3). Diesen verbinden Sie mit Pin 2 auf der Rechnerseite.
Falls jetzt die übrigen Parameter stimmen (denken Sie daran: keine Paritätsprüfung, in englischen Handbüchern oft auch als Parity disable bezeichnet), kann es schon losgehen. Nach dem OPEN und einem anschließenden PRINT muß das Gerät drucken.

Beim fortlaufenden Drucken werden Sie nun feststellen, daß jeweils am Beginn jeder Zeile einige Zeichen fehlen. Das liegt daran, daß wir dem Drucker noch nicht die Möglichkeit gegeben haben, den Datenfluß zu stoppen, wenn er

vorübergehend keine Zeichen mehr aufnehmen kann.
Sie müssen nun herausfinden, mit Hilfe welcher Leitung der Drucker diesen Zustand signalisiert. Setzen Sie die Variable hs für den OPEN-Befehl =1. Öffnen Sie die Brücke 4-5 auf der Rechnerseite und verbinden Sie Pin 5 der Rechnerseite auf der Druckerseite einmal mit der Brücke 4-5, ein andermal mit der Brücke 6-8-20. In einem von beiden Fällen müssen die Daten in jeder Zeile vollständig sein.
Ist der Parameter hs =0, werden die Handshakeleitungen nicht ausgewertet und der Rechner sendet munter weiter. Dieses Verfahren ist nur bei extrem niedrigen Baudraten möglich, bei denen der Drucker beim Ausdrucken der Zeichen noch mit der Eingabe schritthalten kann. Es muß beispielsweise bei Geräten ohne Eingangspuffer angewendet werden oder bei solchen, die nicht über die entsprechenden Signale verfügen, um den Datenfluß zu bremsen.

Das einzige, was Ihnen nun noch passieren kann, ist, daß Sie das, was Sie als Kleinschrift herausgegeben haben, auf dem Drucker als Großschrift wiederfinden. Dagegen können Sie nur programmtechnisch vorgehen. Lesen Sie dazu unter 3.3 die Erläuterungen zu Sekundäradresse 1.

Wie Sie sehen, ist es so schwer nicht, wenn man weiß, wie es geht.
Nur ist die Unzahl von Parametern und Leitungen recht verwirrend, so daß der Einstieg schwer zu finden ist.

## *3.0 DIE SEKUNDÄRADRESSEN*

Wohl kaum ein Thema ist so ergiebig wie das der Sekundäradressen. Diese 'grauen Eminenzen' können einem das Leben ganz schön schwer machen, wenn es um die Portabilität von Software geht, die sich eines Druckers bedient.
Geben Sie sich nicht der Illusion hin, daß schon der Prefix "sekundär" auf so etwas wie eine Zweitrangigkeit der Bedeutung hinweist. Diese Bezeichnung ist nur aus der Terminologie des IEC-Bus zu verstehen, wo es eine Primäradresse, nämlich die Geräteadresse gibt und eben die Sekundäradresse, die deshalb so heißt, weil sie als zweite dem Gerät übergeben wird.
Mit ihrer Hilfe werden in der Regel gerätespezifische Funktionen eingestellt, z.B. wie die in der Folge übermittelten Daten zu interpretieren sind.
Selbst bei einem so 'gewöhnlichen' Gerät wie einem Drucker sind der Interpretationsmöglichkeiten eine ganze Menge, wovon so mancher leidgeprüfte Programmierer ein Lied zu singen weiß und wie auch Sie am Ende dieses Kapitels erkennen werden.

Glauben Sie nicht, daß nun bei allen Druckern eine Sekundäradresse auch dieselben Funktionen auslöst, zumal sie von einem Hersteller stammen. Weit gefehlt. Es ist nicht einmal bei ein und demselben Drucker der Fall, wie Sie gleich sehen werden.
Der gravierendste Unterschied besteht darin, daß bei einem Gerät beispielsweise die Sekundäradresse 7 nur als Schalter dient, über die jedoch keine Druckdaten übertragen werden dürfen, bei einem anderen Gerät jedoch das genaue Gegenteil der Fall ist. Dieser Umstand verurteilt in allen Fällen die Anwendung von Software auf einen anderen Drucker als den, für den sie erstellt wurde, zum Scheitern.

Wer macht sich bei der Programmerstellung schon darüber Gedanken. Die PRINT-Statements werden dort ins Programm eingefügt, wo sie gerade gebraucht werden.
An dieser Stelle ein Tip, den Sie beherzigen sollten:
Schreiben Sie Ihre Druckerausgabe stets als Unterprogramm, auch wenn dieses nur aus einer einzigen PRINT-Anweisung besteht. Wenn uns noch weitere Drucker beschert werden, brauchen Sie nur an dieser einzigen oder an wenigen Stellen zu ändern. Ebenso verfahren Sie auch mit dem zugehörigen OPEN-Befehl. In der Regel braucht auch nur dieser geändert werden.

Sollte aus programmtechnischen Gründen ein derartiges Vorgehen nicht geraten erscheinen und Sie doch etliche PRINT's brauchen, gehen Sie wie folgt vor:
Definieren Sie log. Filenummer, Geräteadresse und Sekundäradresse am Programmanfang und beziehen sich beim späteren OPEN und PRINT auf diese. Damit haben Sie zwei Fliegen mit einer Klappe geschlagen, denn erstens haben Sie so die Sekundäradresse für spätere Änderungen im Griff, zweitens auch die log. Filenummer, was dann sehr nützlich ist, wenn Sie einmal Drucker mit Auto-Line-Feed benutzen und ein andermal solche ohne, denn wie Sie vielleicht wissen, fügt das Betriebssystem des C64 bei log. Filenummern >127 an die Druckdaten noch einen extra Zeilenvorschub an, bei log. Filenummern <128 jedoch nicht.

Das Ganze sieht dann so aus:

```
10 LF=1:DA=4:SA=7
:
100 OPEN LF,DA,SA
:
1000 PRINT#LF,.....
```

Nun können Sie die Werte in Zeile 10 bei Bedarf variieren, ohne das ganze Programm nach OPEN und PRINT absuchen zu müssen.

Wir wollen Ihnen nun die Sekundäradressfunktionen der gebräuchlichsten Drucker vorstellen. Bei Funktionen, die erfahrungsgemäß besonders fehlerträchtig sind, werden wir besonders darauf hinweisen.

### *3.1 VC-1525 / GP-100VC / MPS-801*

Diese Drucker kennen nur zwei Sekundäradressen, mit denen bestimmt wird, aus welchem Zeichenvorrat die Druckzeichen genommen werden sollen.
Genau wie der C64 haben auch die Drucker zwei Darstellungsarten.

Sekundäradresse 0 oder keine Sekundäradresse

```
10 OPEN 1,4,0  oder
10 OPEN 1,4
20 PRINT#1,"abcABC"
```

In dieser Betriebsart wird der Drucker die Kleinbuchstaben in Großschrift wiedergeben, und die Großbuchstaben als Grafikzeichen.
Diesen Modus können Sie für die Länge einer Zeile vorübergehend ändern, indem Sie das Steuerzeichen für 'Cursor down' voranschicken:

```
30 PRINT#1,CHR$(17)"abcABC"
```

Nur in dieser Zeile wird nun Groß- und Kleinschrift richtig

gedruckt.

Sekundäradresse 7

```
10 OPEN 1,4,7
20 PRINT#1,"abcABC"
```

Groß- und Kleinschrift werden richtig wiedergegeben.
Auch diese Betriebsart läßt sich vorübergehend ändern, indem das Steuerzeichen für 'Cursor up' vorangestellt wird:

```
30 PRINT#1,CHR$(145)"abcABC"
```

Hier erscheint nun die Kleinschrift in Großbuchstaben und die Großschrift in Grafikzeichen.
Bitte beachten Sie, daß eine Änderung der Betriebsart durch CHR$(17) oder CHR$(145) immer nur für das jeweilige PRINT-Statement gilt.

Natürlich lassen sich auch beide Betriebsarten innerhalb einer Zeile mischen. Entweder so:

```
10 OPEN 1,4
20 PRINT#1,"abc"chr$(17)"ABC"
```

oder so:

```
10 OPEN 1,4:OPEN 2,4,7
20PRINT#1,"abc";:PRINT#2,"ABC"
```

Beide Versionen zeigen das gleiche Ergebnis. Aus Kompatibilitätsgründen ist jedoch das erste Beispiel vorzuziehen, da es bei allen bis jetzt bekannten Commodore-Druckern wirkt.

*3.2 VC-1526*

Dieser Drucker hat es in jeder Beziehung in sich. Wohl kaum ein Gerät hat in der Vergangenheit soviel Verwirrung gestiftet. Um es kurz zu machen:
Die Geräte, die ab ca. 12/83 ausgeliefert wurden, entsprechen in ihrer Bedienung nicht dem Handbuch, sondern dem VC-1525. Bedauerlicherweise mit der Ausnahme, daß eine Einzelpunktgrafik mit vorangesetztem CHR$(8) nicht möglich ist.
Durch folgende Maßnahme läßt sich jedoch der 1526-Modus einstellen:
Nach Öffnen des Gerätes ist der Pin 16 des IC's auf Einbauplatz U4D auf Masse zu legen. Nun findet auch das beigefügte Handbuch wieder Anwendung.
Ob sich Ihr Gerät dazu eignet, finden Sie heraus, indem Sie den Selbsttest laufen lassen. In der ersten Zeile des Ausdrucks finden Sie die Versionsnummer des im Drucker enthaltenen Betriebssystems. Diese muß wenigstens 7 sein.
Für die 1526-Betriebsart nun die zahlreichen Sekundäradressen:

Sekundäradresse 0 oder keine Sekundäradresse

Die Druckdaten werden so wiedergegeben, wie über Sekundäradresse 7 oder 8 (siehe dort) eingestellt. Vorübergehende Änderung der Betriebsart ist durch Voransetzen von CHR$(17) für Klein- Großschrift, bzw. CHR$(145) für Groß- Grafik möglich.

Sekundäradresse 1

Hierüber werden zu formatierende Daten ausgegeben entsprechend der Formatanweisung, die zuvor über Sekundäradresse 2 übermittelt wurde.
Die Auswahl des Zeichenvorrats erfolgt so, wie zuvor über

die Sekundäradressen 7 oder 8 festgelegt. Auch hier läßt sich die Auswahl für eine Zeile ändern.
Denken Sie daran, daß die zu einem Feld gehörenden Daten durch CHR$(29) vom nächsten Feld getrennt werden müssen. Beispiele hierzu finden Sie im Druckerhandbuch.

Sekundäradresse 2

Über diese Sekundäradresse wird die Formatanweisung übertragen. Diese besagt, wo die in der Folge über Sekundäradresse 1 geschickten Daten auf dem Papier ihren Platz finden sollen. Besonders praktisch ist hierbei die Fähigkeit, auch numerische Werte formatiert auszugeben, d.h. Dezimalpunkte werden untereinandergesetzt. Mehr darüber finden Sie im Handbuch.

Sekundäradresse 3

Hierüber können Sie angeben, wieviele Zeilen pro Seite zu drucken sind, bevor ein sechszeiliger Leervorschub durchgeführt wird. Das ist dann sinnvoll, wenn Sie Endlospapier nicht über die Perforation hinweg bedrucken wollen. Auf diese Weise haben Sie dann am Kopf und am Fuß des Blattes je drei Leerzeilen.
Zu beachten ist hierbei, daß der übermittelte Wert der maximalen Zeilenkapazität des Blattes minus 6 entsprechen muß. Konkret heißt das, wenn Sie, wie in Deutschland üblich, Endlospapier mit einer Länge von 12" verwenden und Sie dieses mit einer Dichte von 6 Zeilen/Zoll bedrucken, passen 72 Zeilen auf das Blatt. Davon ist der obere und untere Rand von je 3 Zeilen abzuziehen. Das zugehörige Kommando sieht so aus:

```
10 OPEN 1,4,3
20 PRINT#1,CHR$(66)
30 CLOSE 1
```

Bitte beachten Sie, daß diese Maßnahme alleine noch keine Wirkung zeigt. Die Seitenunterteilung muß erst durch:

```
10 OPEN 1,4
:
500 PRINT#1,CHR$(147)CHR$(141);
```

aktiviert werden.

Sekundäradresse 4

Dies ist nicht mehr als eine nette Spielerei, denn hierüber wird die Diagnoseinrichtung des Druckers eingeschaltet. Das geht so:

```
10 OPEN 1,4,4:PRINT#1:CLOSE 1
```

In der Folge erhalten Sie nun bei den Drucker betreffenden Programmierfehlern einen Ausdruck über die Fehlerart.

Sekundäradresse 5

Über diese Sekundäradresse können Sie ein eigenes Zeichen definieren, welches in einer 8x8-Matrix Platz finden muß. Abrufen können Sie dieses Zeichen durch CHR$(254).
Auf diese Weise ist, wenn auch mühsam, sogar eine echte Einzelpunktgrafik zu realisieren, wie Sie im Abschnitt 5.3.2 sehen können.

Sekundäradresse 6

Hiermit können Sie den Abstand zwischen zwei Zeilen in Schritten von 1/216" bestimmen. Der gültige Wertebereich beträgt 1...127.
Beachten Sie, daß ein Punktabstand drei Schritten entspricht. Um also die Zeilen lückenlos aneinanderzureihen, ist folgendes Kommando notwendig:

```
10 OPEN 1,4,6:PRINT#1,CHR$(24):CLOSE 1
```

Sekundäradresse 7

Dies ist ein Schalter für Klein- Großschrift. Um diese Betriebsart einzuschalten, benutzen Sie folgenden Befehl:

```
10 OPEN 1,4,7:PRINT#1:CLOSE 1
```

Zu beachten ist, daß keine Daten hierüber geschickt werden. Diese gehen ausschließlich über die Sekundäradresse 0 oder 1.

Sekundäradresse 8

Diese stellt das Gegenstück zu Sekundäradresse 7 dar.

```
10 OPEN 1,4,8:PRINT#1:CLOSE 1
```

schaltet den Drucker in den Groß- Grafikmodus.

Sekundäradresse 9

```
10 OPEN 1,4,9:PRINT#1:CLOSE 1
```

schaltet den mit Sekundäradresse 4 aktivierten Diagnosemodus wieder aus.

Sekundäradresse 10

```
10 OPEN 1,4,10:PRINT#1:CLOSE 1
```

setzt den Drucker in die Grundstellung zurück, ebenso, als ob er gerade eingeschaltet worden wäre.
Von dieser Möglichkeit können Sie Gebrauch machen, wenn Sie im Verlauf eines Programmes mehrere Einstellungen verändert

haben und diese nicht auf dem gleichen Wege umständlich wieder rückgängig machen wollen.

### *3.3 EPSON-Drucker mit DATA BECKER-Interface*

Aufgrund der recht weiten Verbreitung dieser Konfiguration auch hierzu ein paar Worte.
Prinzipiell entspricht die Anwendung der Sekundäradressen der unter VC-1525 beschriebenen. Zusätzlich gibt es noch drei weitere Sekundäradressen.

Sekundäradresse 1

Hierzu müssen wir etwas weiter ausholen:
Alle handelsüblichen Drucker, mit Ausnahme der von Commodore, besitzen eine Zeichenverteilung nach ASCII. In dieser international anerkannten Vorschrift ist die Lage der Buchstaben und Sonderzeichen festgelegt. So liegt zum Beispiel bei CHR$(65) das "A" und bei CHR$(97) das "a". Bei Commodore jedoch (im Klein- Großmodus) liegt bei CHR$(65) das "a" und bei CHR$(193) das "A".
Die sich hieraus offensichtlich ergebenden Konflikte wurden durch Einführen der Sekundäradresse 2 (siehe dort) gelöst.
Die hier zu besprechende Sekundäradresse 1 jedoch läßt alle Zeichen vom Rechner ungefiltert an den Drucker durch. Sie wären deshalb nie in der Lage, mit normalem Text Kleinschrift auf dem Drucker hervorzubringen.
Die Sekundäradresse 1 ist aber auch nicht zur Übermittlung von Text gedacht, sondern zur Einstellung der vielfältigen Möglichkeiten der EPSON-Drucker, die jene laut Handbuch besitzen. Benutzen Sie deshalb nie die in den Steuersequenzen angegebenen Buchstaben, sondern holen Sie sich aus dem Anhang des Handbuches den entsprechenden CHR$.

Beispiel:
Sie wollen den linken Rand auf die Position 10 setzen. Im Handbuch finden Sie, daß hierzu der Kennbuchstabe "l" erforderlich ist. Aus der Character-Code-Tabelle im Anhang finden Sie den zugehörigen Wert 108. So sollte Ihr Kommando aussehen:

```
10 OPEN 1,4,1
20 PRINT#1,CHR$(27)CHR$(108)CHR$(10);
30 CLOSE 1
```

Geben Sie statt CHR$(108) "l" ein, landen Sie im Grafikmodus, da "l" bei Commodore im Drucker zu "L" nach ASCII wird.
Die Einzelnadelgrafik sollten Sie ebenfalls über diese Sekundäradresse abwickeln.
Beachten Sie, daß eine log. Filenummer >127 zu verwenden ist, wenn Sie doch normalen Zeilendruck über diese Sekundäradresse geben wollen, da hier der automatische Zeilenvorschub unterdrückt wird.

Sekundäradresse 2

Das oben angedeutete Dilemma ist mit dieser Sekundäradresse gelöst. Alle Daten, die hierüber ankommen, werden mit Hilfe einer Tabelle automatisch so umgewandelt, daß einem "a" auf dem Bildschirm auch ein "a" auf dem Drucker entspricht.
Sicher ist das auch über die Sekundäradresse 7 der Fall, jedoch wird dort die Original Commodore-Schrift erzeugt, die nicht jedermanns Geschmack ist. Bei Sekundäradresse 2 jedoch wird die schönere EPSON-Schrift verwendet, die auch die deutschen Umlaute ermöglicht. Einschränkend ist zu sagen, daß hierüber kein Ausdruck der Commodore Grafik- und Steuerzeichen möglich ist.
Beachten Sie, daß Sie über diese Sekundäradresse nicht mit

der Einzelnadelgrafik arbeiten sollten, da auch die Punktemuster eine Umwandlung über die Tabelle erfahren und das Ergebnis deshalb nicht so sein wird, wie Sie es erwarten.

Sekundäradresse 3

Diese gibt es nach Angaben des Herstellers erst seit 6/84. Hiermit wird die zuletzt aktive Sekundäradresse fixiert und bleibt bis zum Ausschalten des Gerätes erhalten, gleichgültig, über welche anderen Sekundäradressen die Daten auch kommen mögen. Der Verriegelung geschieht mit folgendem Kommando:

```
10 OPEN 1,4,3:PRINT#1:CLOSE 1
```

Der Sinn liegt darin, die üblichen Textverarbeitungsprogramme zu überlisten. Diese senden nämlich gewöhnlich die Daten über die Sekundäradresse 7, was einen Ausdruck in der Commodore-Schrift zur Folge hat. Da man aber nun schon einmal einen EPSON besitzt, möchte man auch dessen Schrift nutzen. Also führen Sie vor dem Laden der Textverarbeitung folgende Sequenz aus:

```
10 OPEN 1,4,2:OPEN 2,4,3
20 PRINT#1:PRINT#2
30 CLOSE 1:CLOSE 2
```

Nun ist für die gesamte Einschaltdauer des Druckers die Betriebsart der Sekundäradresse 2 fixiert.

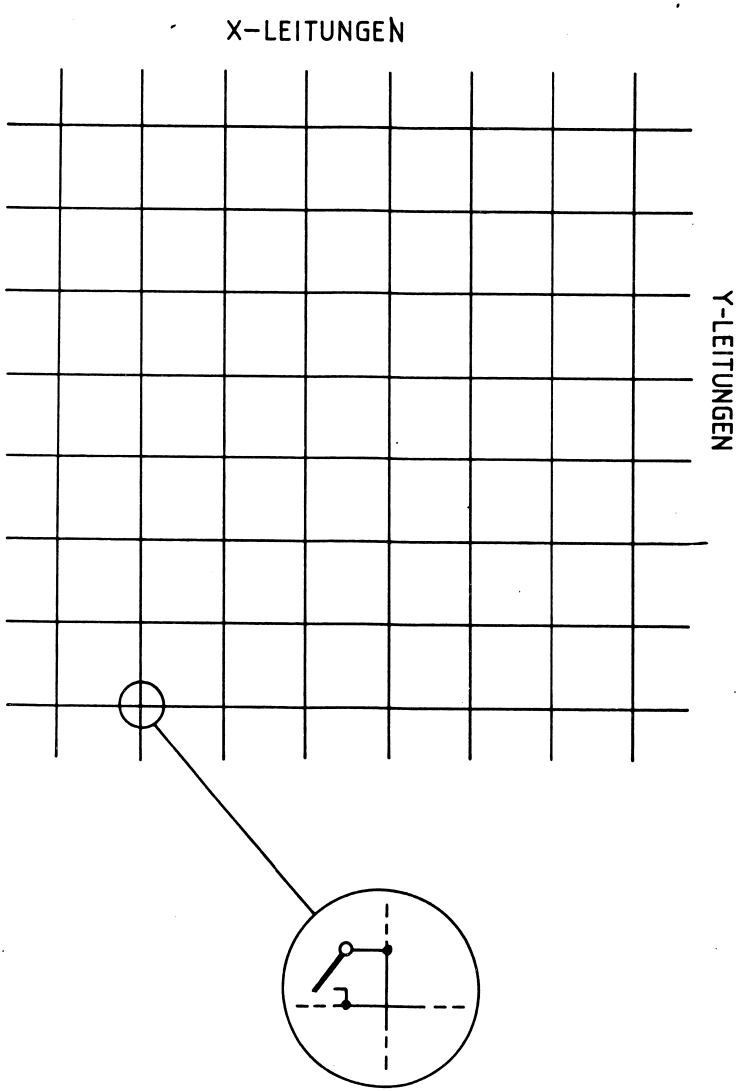
## *4.0 SCHREIBMASCHINE AM USER-PORT*

Um es gleich klarzustellen: Mit Schreibmaschine meinen wir eine moderne elektrische Schreibmaschine.
Um die hier vorgestellte Schaltung nachzubauen, brauchen Sie nicht unbedingt viel Geschick. Weitaus schwieriger ist es, die Maschine anzuzapfen und mit dem Interface zu verbinden. Wir möchten Ihnen dennoch diese reizvolle Möglichkeit nicht vorenthalten, denn vielleicht besitzen Sie so ein Gerät und es juckte Sie schon lange in den Fingern, es Ihrer 'Computerei' nutzbar zu machen.
Als erstes müssen Sie feststellen, ob Ihr gutes Stück sich überhaupt für die Aktion eignet, denn so einfach und preiswert (der Materialwert dürfte kaum über DM 20 liegen) die Schaltung auch ist, umso wählerischer ist sie bei den Schreibmaschinen. Wenn Sie aus eigener Kenntnis oder aus der Anleitung wissen, daß Ihre Maschine einen Microprozessor enthält, haben Sie schon gute Karten. Keine Angst, Sie brauchen denselben nicht umzuprogrammieren, aber das Vorhandensein eines solchen läßt darauf schließen, daß die Tastatur in Form einer X-Y-Matrix verdrahtet ist, und genau das ist die Voraussetzung für den Anschluß.

Wie finden Sie nun heraus, ob das bei Ihrer Maschine der Fall ist? Eigentlich ganz einfach. Sie benötigen nur ein Ohmmeter. Dann öffnen Sie das Gerät vorsichtig, so daß Sie an die Tastaturkontakte herankommen.
Bevor Sie jetzt weitermachen, wollen wir Ihnen das Prinzip einer Tastaturmatrix erklären. Danach ist alles ganz einfach.

Eine Tastaturmatrix ist im Prinzip so aufgebaut, wie das Bild auf der nächsten Seite zeigt. Zwar mögen es mal mehr und mal weniger Leitungen sein, aber das spielt keine Rolle.

**Allgemeine Tastaturmatrix**

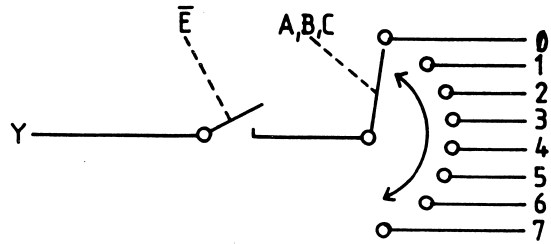


Wie Sie sehen, besteht jeder Kreuzungspunkt auf der Matrix aus einem Schalter, eben den Schreibmaschinentasten. Die Logik oder der Prozessor in der Maschine legt nun der Reihe nach an jede X-Leitung eine Spannung. Ist keine Taste gedrückt, passiert gar nichts. Ist jedoch eine gedrückt, wird diese Spannung auf der entsprechenden Y-Leitung erscheinen. Das bekommt natürlich der Prozessor mit und weiß nun genau, welche Taste gedrückt wurde, denn sie muß im Schnittpunkt der X-Leitung, auf die die Spannung gelegt wurde, mit der Y-Leitung, an der die Spannung erscheint, liegen.
Sie sollten sich vergewissern, daß diese Spannung den Betrag von 5V nicht überschreitet, da unser Interface nicht für eine höhere Spannung ausgelegt ist, aber das ist bei einer prozessorgesteuerten Maschine in der Regel nicht zu erwarten.
Unsere Aufgabe besteht nun darin, die Schalter mit Hilfe unseres Interface zu simulieren. Die Matrixanordnung eignet sich ideal dazu, da sie eine Menge Leitungen, von der übrigen Hardware ganz zu schweigen, erspart. Stellen Sie sich vor, Sie müßten zu jeder der ca. 50 Tasten je zwei Leitungen legen!
So brauchen nur die X- und die Y-Leitungen angezapft zu werden. Das sind bedeutend weniger.
Jetzt beginnt eine Fleißarbeit. Sie müssen nämlich herausfinden, wo die X- und Y-Leitungen liegen. Wie Sie auf dem Bild erkennen können, liegen jeweils einige Kontakte mit einer Seite zusammen an einer Leitung. Diese Zusammenhänge müssen Sie jetzt mit dem Ohmmeter herausmessen. Das bedeutet, daß Sie jeden Kontakt gegen jeden anderen messen müssen, denn es ist keineswegs zu erwarten, daß alle Tasten einer Leitung auch tatsächlich mechanisch nebeneinander liegen.
Sie werden jetzt etwa zwei Stunden gut zu tun haben. Danach

sollten Sie eine Anordnung ähnlich der auf dem Bild aufgefunden haben. Vergessen Sie nicht, eine Aufzeichnung der aufgefundenen Zusammenhänge zu machen, denn das erleichtert die spätere Programmierung erheblich.
Es kann sein, daß einige Tasten nicht mit weiteren an einer Leitung zusammenhängen. Das ist normal, denn es sind sicherlich die Shift- oder andere Steuertasten, die nicht über die Matrix abgefragt werden.

Nun zu unserer Schaltung:
Das Herz besteht aus zwei sogenannten Analogmultiplexern. Auf dem Bild unten sehen Sie, wie Sie sich das Innenleben vorzustellen haben.

ANALOGMULTIPLEXER
FUNKTIONSPRINZIP



Nehmen Sie das Bild in sich auf und schauen sich die vollständige Schaltung am Ende des Kapitels an. Sie sehen jetzt sofort, was wir vorhaben. Wir können nämlich eine von acht X-Leitungen mit einer von acht Y-Leitungen verbinden. Welche, das wird über den User-Port gesteuert. Damit gaukeln wir der Logik in der Schreibmaschine vor, daß eine Taste gedrückt wurde, nämlich die, die im Schnittpunkt der beiden Leitungen liegt.
Damit nicht ständig irgendwelche Leitungen verbunden sind, benutzen wir die Enable-Eingänge der Multiplexer, um sie nur für die Dauer eines tatsächlichen Tastendruckes durchzuschalten.

Das Ganze funktionert nun so, daß drei Bit des User-Port für die Auswahl der X-Leitung hergenommen werden, weitere drei Bit für die Auswahl der Y-Leitung und die beiden übrigen, um die Analogschalter für die Shift-Taste und eine weitere Taste Ihrer Wahl bedienen zu können.
Nachdem Sie alles aufgebaut und angeschlossen haben, nehmen Sie das folgend Programm zum Ausprobieren:

```
10 PA=56576:PB=56577
20 DA=56578:DB=56579
30 POKE DA,PEEK (DA) OR 4:REM DATENRICHTUNG SETZEN
40 POKE DB,255
50 POKE PA,PEEK (PA) OR 4
100 INPUT "X-LEITUNG";X
110 INPUT "Y-LEITUNG";Y
120 INPUT "SHIFT 0=NEIN, 1=JA";SH
130 CH=X+8*Y+64*SH:PRINT CH
140 GOSUB 200:GOTO 100
200 POKE PB,CH
210 POKE PA,PEEK (PA) AND 251
220 FOR DL=0 TO 100:NEXT DL
```

```
230 POKE PA,PEEK (PA) OR 4
240 RETURN
```

Schreiben Sie sich die gefundenen Zusammenhänge zwischen gedrucktem Buchstaben und X- Y-Leitung und Shift-Taste auf, indem Sie einfach neben jeden gefundenen Buchstaben die zum gleichen Zeitpunkt auf dem Bildschirm erscheinende Zahl notieren.
Damit und mit Hilfe des folgenden Programmes bauen Sie nun eine Übersetzungstabelle auf, die es Ihnen ermöglicht, Ihre Texte komfortabel zu drucken.

```
10 DIM CH(255)
20 GET A$:IF A$="" THEN 20
30 PRINT "ZU "A$" GEHOERENDE ZAHL";
40 INPUT CH
50 CH(ASC(A$))=CH
60 GOTO 20
```

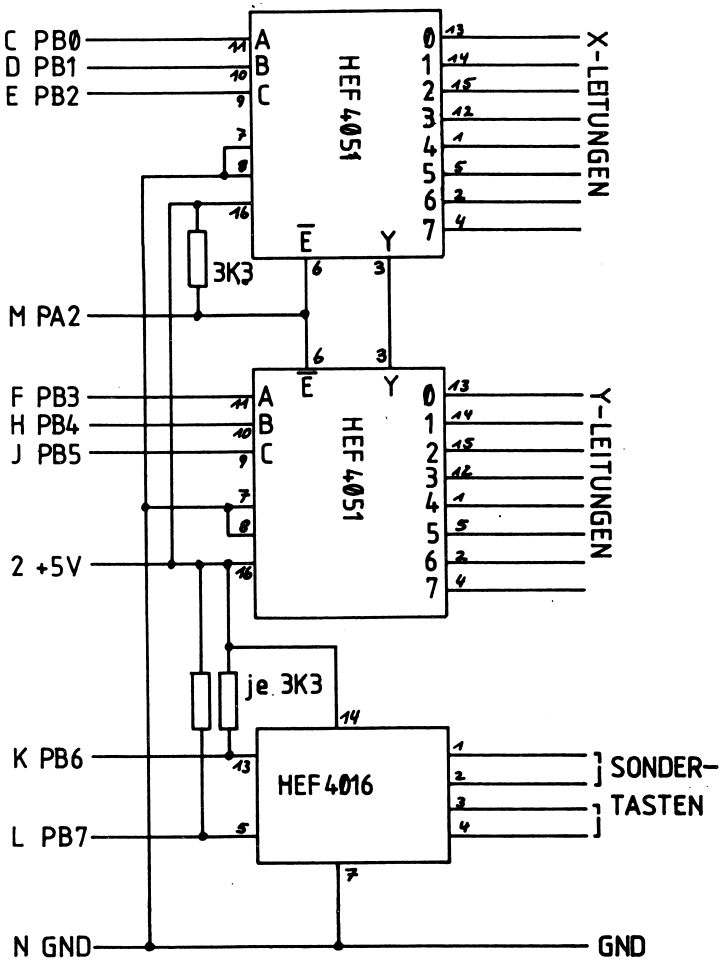
Wenn Sie fertig sind, halten Sie das Programm mit STOP an und speichern die Tabelle CH(0-255) auf Datasette oder Diskette ab, damit Sie sie bei späterem Gebrauch nur zu laden brauchen.
So einfach ist dann der Ausdruck von Texten: Ersetzen Sie die Zeilen 100 bis 140 aus dem ersten Programm gegen die folgende Routine.

```
100 FOR L=1 TO LEN(CH$)
110 CH=CH(ASC(MID$(CH$,L,1)))
120 GOSUB 200:NEXT L:RETURN
```

Einzige Voraussetzung: Ihr Text muß in CH$ enthalten sein und Sie springen die Routine einfach mit GOSUB 100 an. Die Zeilen 10 bis 50 müssen einmal zu Beginn Ihres Programmes

Geschwindigkeit trimmen.

So, das war's. Wir wünschen Ihnen viel Spaß beim Nachbau. Dem Schaltplan ist seiner Einfachheit wegen wohl nichts mehr hinzuzufügen.



## *5.0 PROGRAMMSAMMLUNG FÜR VC-20 UND C64*

In diesem Kapitel wollen wir Ihnen einige Programme präsentieren, die sicher interessante Anregungen für die Drucker-Programmierung beinhalten.
Dabei werden viele Gebiete möglicher Anwendungen behandelt.
Leider ist es bei der Fülle der verschiedenen lieferbaren Drucker unmöglich, jedes Programm für jeden Drucker zu bearbeiten und hier abzudrucken. Dann hätte dies Buch schnell den Umfang des Telefonbuchs einer Großstadt bekommen.
Die abgedruckten Programme zur Einzelnadelgrafik sind naturgemäß für die Besitzer von Typenraddruckern nicht sehr ergiebig. Aber auch an diese Leser haben wir gedacht und einige kleine Grafik-Programme in das Buch aufgenommen, die auf Typenraddruckern durchaus überraschende Ergebnisse erzielen.
An allen wichtigen Stellen geben wir aber Tips, wie die Programme an andere Drucker anzupassen sind. Wenn Sie also einen in den einzelnen Kapiteln nicht erwähnten Drucker besitzen, der andere als die verwendeten Druckersteuerzeichen benutzt, so finden Sie möglicherweise im Druckerhandbuch einen entsprechenden Steuerbefehl, der die gewünschten Aktionen bewirkt.
Unabhängig davon, ob nun alle Programme mit Ihrem Drucker laufen oder nicht, werden Sie aber sicher den Programmen Anregungen für eigene Programmierobjekte entnehmen können.

## *5.1 Kleine Hilfsprogramme für alle Tage*

Im Kapitel 5.1 werden Ihnen drei Hilfsprogramme vorgestellt, die sicher in der vorliegenden Form für alle Druckerbesitzer interessant sind.
Zuerst wollen wir eine kurze Routine vorstellen, die einen Programmabbruch bei nicht eingeschaltetem Drucker verhindert. Dieses kurze Programm kann in jedes Basic-Programm eingefügt oder angehängt werden und verhindert den lästigen DEVICE NOT PRESENT-Fehler.
Die zweite Routine ist schon etwas umfangreicher. Sie stellt ein sehr komfortables Unterprogramm zur Formatierung numerischer und alphanumerischer Daten dar.
Auch das letzte Hilfsprogramm ist eine Art formatierter Druckausgabe. Ausgegeben werden aber nicht Zahlen oder Texte, sondern Programmlistings. Diese Listings werden sehr übersichtlich ausgegeben. Jeweils zwischen den Befehlsworten werden Leerzeichen eingefügt, FOR-NEXT-Schleifen werden entsprechend eingerückt und IF-THEN-Strukturen aufgeschlüsselt. Damit lassen sich auch wirklich unübersichtliche Listings lesbar machen.

### *5.1.1 Device Not Present?*

Um Programme 'Wasserdicht' zu machen, um also einen unfreiwilligen Absturz zu verhindern, stellt das Basic der Commodore-Geräte keine Möglichkeiten wie z.B. einen ON-ERROR-Befehl zur Vefügung.
Nun lassen sich syntaktische Fehler oder Fehler wie das Teilen durch Null durch einen guten Programmierstil sicher umgehen. Anders sieht es aber bei Programmen aus, die auf Drucker oder Floppy zugreifen. Hier ist die Verhinderung von Fehlern, speziell des DEVICE NOT PRESENT, in Basic nur mit

Tricks zu erreichen.

Daß diese Tricks noch nicht allgemein bekannt sind, kann man sehr schön an einigen Programmen sehen, bei denen sich die Software-Entwickler alle Mühe gemacht haben, das Programm zu schützen. Greift man aber bei diesen Programmen auf einen nicht eingeschalteten Drucker zu, dann befindet man sich blitzartig im Direktmodus und kann das Programm nach belieben listen oder modifizieren.

Ein kurzes Basic-Unterprogramm von wenigen Bytes macht hiermit für alle Zeiten Schluß.

```
100 REM ************************
110 REM **                    **
120 REM **      DRUCKER       **
130 REM **                    **
140 REM ************************
150 POKE768,61
160 OPEN1,4
170 PRINT#1,"";
180 CLOSE1
190 POKE768,139
200 IFST<>-128THEN220
210 PRINT"DRUCKER NICHT EINGESCHALTET"
220 RETURN
230 REM ************************
240 REM **                    **
250 REM **      FLOPPY        **
260 REM **                    **
270 REM ************************
280 POKE768,61
290 OPEN1,8,15,"I"
300 CLOSE1
```

```
310 POKE768,139
320 IFST<>-128THEN340
330 PRINT"FLOPPY NICHT EINGESCHALTET"
340 RETURN
```

Da sich das Programm für Drucker und Floppy nicht sehr unterscheidet, haben wir gleich beide Versionen aufgeführt. Beide Routinen können mit GOSUB aufgerufen werden. Natürlich können Sie die ausgegebenen Meldungen entsprechend anpassen.

Allerdings sollten Sie bei der Verwendung dieser Routinen die BREAK-Taste ausschalten, da der in den Zeilen 150 und 280 verbogene Vektor auf die Eingabe-Routine im Direktmodus zeigt. Durch den POKE in den beiden Zeilen wird dieser Vektor, der gleichzeitig für die Ausgabe der Fehlermeldungen zuständig ist, auf einen Return-Befehl umgelegt. Dadurch werden zum einen die Ausgabe von Fehlermeldungen unterdrückt. Des weiteren ist aber nach diesem POKE kein Direktmodus mehr möglich. Wird also das Programm nach dem ersten POKE unterbrochen, sei es durch BREAK oder RUN/STOP-RESTORE, so verabschiedet sich der Rechner.
Nur der 'kalte Reset', also das Aus- und wieder Einschalten kann dann noch etwas helfen.

### *5.1.2 Formatierung numerischer und alphanumerischer Daten*

Jeder, der schon einmal vor der Aufgabe stand, eine Druckerausgabe zu programmieren, in der sowohl Texte als auch Zahlen schön geordnet untereinander ausgedruckt werden sollten, der wird Haare raufend den manchmal etwas 'mageren' Basic-Dialekt der Commodore-Computer verdammen.
Wenn Benutzer anderer Computer mit PRINT USING oder ähnlich komfortablen Befehlen strotzen, dann stellt sich bei

Commodore-Benutzern mindestens ein intensives Neidgefühl ein. Da wird dann das Handbuch von vorn nach hinten und zurück geblättert, aber es bleibt dabei: So einfach ist die Sache nicht zu lösen.

Als dann eines guten Tages die ersten Drucker 1526 auf dem Markt erschienen, da war eine große Hoffnung und Freude in der großen Commodore-Gemeinde. Hatte dieser Drucker doch unter anderem beeindruckende und wahrlich einfach zu handhabende Möglichkeiten der formatierten Druckausgabe. Daß er dabei nicht grafikfähig war, konnte man als Wermutstropfen dabei wohl schlucken.

Aber diese Freude währte nicht lang. Zusätzlich zu den im Handbuch aufgeführten Eigenschaften neigte er in sporadischer aber geradezu boshafter Weise zum völligen Blokkieren des seriellen Busses, sehr unpraktisch, wenn gerade dann Daten auf die Floppy mußten.

Dieser Umstand blieb natürlich auch dem Hersteller nicht lange verborgen, der sofort reagierte, und ein Austausch-ROM ankündigte.
Danach allerdings verlangsamte sich die Reaktionsgeschwindigkeit drastisch. Wer jetzt an das Sprichwort 'Gut Ding will Weile haben' dachte, der sah sich enttäuscht.
Um den Drucker für den reibungslosen Einsatz fit zu machen, benötigten die Software-Entwickler des Drucker-Betriebssystems einiges an Speicherplatz, der aber in der existierenden Fassung so ohne weiteres wohl nicht verfügbar war. Also wurde zur 'Axt' gegriffen und ganz rabiat Platz geschaffen. Ausgerechnet die formatierte Ausgabe mußte dran glauben.
Womit wir wieder beim Anfang sind.

Jetzt haben Sie die Alternative zwischen den folgenden fünf Punkten. Sie können:

1. einen neuen Rechner mit PRINT USING kaufen oder
2. eine Befehlserweiterung wie z.B. MASTER für Ihren Rechner kaufen oder
3. eine geeignete Routine selber schreiben oder
4. die abgedruckte Routine verwenden oder
5. auf den formatierten Ausdruck ganz einfach verzichten.

Die preiswerteste Methode, zu einem guten Ergebnis zu gelangen, ist zweifellos die im Punkt 4 genannte. Bevor wir jedoch zu dieser Routine kommen, wollen wir uns einmal anschauen, welche Möglichkeiten zur Ausgabensteuerung das Commodore-Basic und die Drucker uns zur Verfügung stellen.

Was jeder Drucker kann, ist, die Zeichen so zu wiederzugeben, wie sie kommen.
Die Ausgabe eines Briefes ist, vom Randausgleich einmal abgesehen, die unproblematischste Form der Ausgabe.
Anders sieht es bereits beim Drucken von Rechnungen aus. Da müssen Textteil und Zahlenteil ausgerichtet werden, nämlich in der Regel der Text linksbündig, die Zahlen rechtsbündig.
Eine Möglichkeit, das Erscheinungsbild zu beeinflussen, bietet Ihnen schon Ihr Rechner von Haus aus: Die Funktion TAB.
Leider verhält sich diese Funktion bei Ausgabe auf den Bildschirm anders als beim Drucker.
Möglicherweise ist es Ihnen schon aufgefallen, daß beispielsweise die Zeile

```
print"hallo"tab(10)"wie geht es"
```

auf dem Bildschirm ganz ordentlich

```
hallo    wie geht es
```

produziert, auf dem Drucker jedoch

```
hallo          wie geht es
```

Wie ist das möglich? Ganz einfach, das Betriebssystem wandelt den tab(x) bei der Ausgabe auf ein anderes Gerät als den Bildschirm in ein spc(x) um.
Tab bedeutet ja eine absolute Position vom linken Rand, während spc eine Distanz vom augenblicklichen Stand des Cursor, bzw. Druckkopfes bestimmt.
Abhängig von der Länge des ersten Textes wird sich auch der Anfang des zweiten Textes verschieben.
Was also ist zu tun?
Eine Möglichkeit gibt es, die jedoch nur auf einigen Commodore-Druckern anwendbar ist:
Die Bestimmung der absoluten Position vom linken Rand mittels Wagenrücklauf ohne Zeilenvorschub und anschließendes Drucken des zweiten Zeilenteiles.
Einen Rücklauf des Druckkopfes ohne Papiertransport können Sie bei manchen Commodore-Druckern durch Senden von chr$(141) erreichen. Am konkreten Beispiel sieht das für den Druck einer Rechnungszeile so aus:

```
10 OPEN1,4
20 PRINT#1,"VERLÄNGERUNGSSCHNUR"CHR$(141)SPC(40)"DM 6.45"
```

Der Text wird nun linksbündig gedruckt, und die Preisangabe ab der Spalte 40.
Das verdanken wir dem einfachen Trick, daß wir erst den Text gedruckt, dann einen Wagenrücklauf ohne Zeilenvorschub verursacht, anschließend ein Vorrücken um 40 Zeichen

befohlen und dann den Preis gedruckt haben.
Sie sehen also, daß Sie bei der Ausgabe auf Drucker die Wirkung eines echten TAB durch Voranstellen eines chr$(141) erreichen können.
Drucker, die diese Möglichkeit nicht bieten, dies ist z.B. der 1525, haben zur Positionierung des Druckkopfes einen eigenen Steuercode. Es ist dies der Code CHR$(16).
Durch diesen Steuercode haben Sie die Möglichkeit, den Druckkopf auf jede der 80 vorhandenen Schreibstellen zu positionieren. Die Angabe der Druckposition muß dabei als Ascii-Zeichen an den Drucker geschickt werden.
Auch hierfür wollen wir wieder ein konkretes Beispiel zeigen:

```
10 OPEN 1,4
20 PRINT#1,"ERDBEERKONFITÜRE"CHR$(16)"40";"DM 3.12"
```

Auch bei diesem Beispiel wird der Preis beginnend in der Spalte 40 ausgedruckt. Die Druckposition wird durch die 40 vor den Zeichen DM bestimmt.

Sie haben aber damit erst erreicht, daß die gewünschten Texte an den durch spc(x) oder die im Text festgelegten Stellen beginnen, was jedoch nicht in allen Fällen auch zweckmäßig ist.
Um bei der Rechnungszeile aus dem vorigen Beispiel zu bleiben:
Angenommen, eine nach dem gleichen Schema übergebene Zeile enthielte als Betrag den String "DM 66.45". Der Betrag würde zwar wieder an der gleichen Position beginnen, jedoch würde der Dezimalpunkt gegenüber der darüberliegenden Zeile um eine Stelle nach rechts rücken. Stellen Sie sich vor, so würde eine Rechnung mit Beträgen unterschiedlicher Vorkommastellenzahl gedruckt.

Das Ergebnis wäre als Exponat einer Kuriositätensammlung geeignet, nicht jedoch zur Dokumentation Ihrer finanziellen Forderungen.
Wie fast alles, ist natürlich auch dieses Problem jeweils programmtechnisch zu lösen, indem Sie zunächst aus der Zahl einen String machen, diesen dann auf die Lage des Dezimalpunktes untersuchen und den spc-Parameter oder die direkte Druckposition davon abhängig machen, wobei natürlich die unterschiedlichen Positionen einer Zeile auch unterschiedlich behandelt werden müssen.

Bei der Rechnung kann nun eine Zeile etliche Felder haben. Das beginnt mit der Positionsnummer, dann kommt die Menge, dann die Artikelbezeichnung, dann der Einzelpreis und abschließend der Gesamtpreis.
Wenn Sie das nach obigem Schema realisieren wollten, hätten Sie gut zu tun.

Wie einfach wäre es, wenn man das Format einer Zeile unterschieden nach alphanumerischen und rein numerischen Feldern einmalig aufbauen könnte und die Ausgabe unter allen Umständen diesem Format entspräche.

Diesen nicht unerheblichen Aufwand haben wir Ihnen mit dem jetzt folgenden Unterprogramm abgenommen und zwar auf die für Sie allerbequemste Weise.

```
48000 REM
48001 REM******************** FORMATIEREN *************
48002 REM
48020 IFFZ%=0THENFZ%=1:FL%=LEN(FO$):FP$=""
48040 FI%=LEN(FI$):FB$="":FC%=1
48050 IFMID$(FO$,FZ%,1)=" "THENFZ%=FZ%+1:FB$=FB$+" ":GOTO48050
48060 FM$=MID$(FO$,FZ%,1)
```

```
48080 FV%=0:FH%=0:FA%=0:FB%=0:FF%=0
48100 IFMID$(FO$,FZ%,1)="."THENFZ%=FZ%+1:FF%=2:GOTO48180
48120 IFFZ%>FL%THENFA%=FA%+1:FZ%=FZ%+1:GOTO48240
48140 IFMID$(FO$,FZ%,1)=" "THEN48240
48160 FA%=FA%+1:FZ%=FZ%+1:GOTO48100
48180 IFMID$(FO$,FZ%,1)=" "ORFZ%>FL%THEN48240
48200 FB%=FB%+1:FZ%=FZ%+1
48220 GOTO48180
48240 IFMID$(FI$,FC%,1)="."THENFC%=FC%+1:FF%=FF%+4:GOTO48320
48260 FV%=FV%+1:FC%=FC%+1
48280 IFFC%>FI%THENGOTO48380
48300 GOTO 48240
48320 IFFC%>FI%THEN48380
48340 FH%=FH%+1:FC%=FC%+1
48360 GOTO48320
48380 IFFM$="A"THEN48780
48400 L2=FA%-FV%:IFL2<0THENFF%=FF%+1:GOTO48720
48420 IFL2=0THEN48460
48440 FORL1=1TOL2:FB$=FB$+" ":NEXTL1
48460 IFFV%=0THEN48520
48480 L2=FV%:FORL1=1TOL2
48500 FB$=FB$+MID$(FI$,L1,1):NEXTL1
48520 IFFF%=0THEN48920
48540 FF%=0:FB$=FB$+".":L2=FV%+2
48560 IFFB%=0THEN48920
48580 IFFB%=FH%THENL3=FB%:GOTO48620
48600 L3=-FH%*(FB%>FH%)-FB%*(FH%>FB%)-1
48620 FORL1=L2TOL2+L3
48640 FB$=FB$+MID$(FI$,L1,1):NEXTL1
48660 IFFH%>=FB%THEN48910
48680 L2=FB%-FH%:FORL1=1TOL2
48700 FB$=FB$+"0":NEXTL1:GOTO48910
48720 L2=FA%+FB%:IF(FF%AND2)THENFF%=1:L2=L2+1
48740 FORL1=1TOL2:FB$=FB$+"*"
```

```
48760 NEXTL1:GOTO48910
48780 IF(FF%AND4)THENFV%=FV%+FH%+1
48800 L2=FV%:FORL1=1TOL2
48820 IFMID$(FI$,L1,1)<>" "THENL3=L1:L1=L2
48840 NEXTL1
48860 L2=-FV%*(FA%>FV%)-FA%*(FV%>FA%)
48870 IFL2=0THENL2=FV%
48880 FORL1=L3TOL3+L2
48900 FB$=FB$+MID$(FI$,L1,1):NEXTL1
48902 IFFV%>=FA%THEN48910
48904 L3=FA%-L2:FORL1=1TOL3
48906 FB$=FB$+" ":NEXTL1
48910 IFFZ%>FL%THENFZ%=0:GOTO48960
48920 L2=FZ%:L3=FL%:FORL1=L2TOL3
48930 IFMID$(FO$,L1,1)<>" "THENFZ%=L1:L1=L3:GOTO48950
48940 FB$=FB$+" "
48950 NEXTL1
48960 FP$=FP$+FB$:FF%=(FF%AND1)
48970 RETURN
```

Da das Programm viele interne Variablen enthält, wollen wir Ihnen nur diejenigen vorstellen, die den Ablauf der Routine wesentlich beeinflussen.

fo$ muß das von Ihnen gewünschte Zeilenformat enthalten.
fi$ müssen Sie mit den Daten versorgen, die Sie gerne ausgerichtet hätten.
fb$ enthält den gerade aufbereiteten Teilstring, während
fp$ den vollständigen Druckstring enthält.
fz% ist der Zeiger auf den gerade bearbeiteten Terminus in fo$.
ff% signalisiert einen Überlauf der Vorkommastellen (ff%<>0).
fa% enthält die Anzahl der Vorkommastellen des gerade

bearbeiteten Feldes in fo$.
fb% wie oben, jedoch Nachkommastellen.
fv% Anzahl der Vorkommastellen in fi$.
fh% Anzahl der Nachkommastellen in fi$.

Der Kommentar:

48020-
48080 Abhängig vom erstmaligen Einsprung (fz%=0) werden einige Variable rückgesetzt.
48100-
48160 Die Vorkommastellen des gerade bearbeiteten Feldes von fo$ nach fa%.
48180-
48220 Wie oben, jedoch Nachkommastellen nach fb%.
48240-
48300 Anzahl der Vorkommastellen in fi$ nach fv%.
48320-
48360 Anzahl der Nachkommastellen in fi$ nach fh%.
48400-
48440 Hier beginnt die Ausrichtung numerischer Felder. Falls fi$ weniger Vorkommastellen als fo$ enthält, wird mit Leerzeichen aufgefüllt.
48460-
48500 Die gültigen Vorkommastellen kommen nach fb$.
48560-
48640 Die gültigen Nachkommastellen kommen nach fb$.
48660-
48700 Falls fi$ weniger Nachkommastellen als fo$ enthält, wird mit Nullen aufgefüllt.
48720-
48760 Hatte fi$ mehr Vorkommastellen als fo$, wird das gesamte Feld mit '*' gefüllt, damit es auf dem Ausdruck sofort ins Auge fällt. Sie haben in diesem

Falle das Feld zu knapp bemessen. Dem Programm wird dieser Zustand durch ff%<>0 signalisiert.

48780-

48840 Hier beginnt die Aufbereitung für Alphafelder. Zunächst werden führende Leerstellen aus fi$ unterdrückt, damit die sichtbaren Zeichen auch tatsächlich am Feldanfang beginnen.

48860-

48900 Aus fi$ werden Zeichen gemäß fo$ nach fb$ übertragen. Übrige Zeichen aus fi$ werden abgeschnitten, übrige Zeichen aus fo$ mit Leerstellen aufgefüllt.

48910-

48950 Gemäß der Distanz zum nächsten Feld in fo$ wird fb$ mit Leerstellen aufgefüllt.

48960-

48970 fb$ wird auf fp$ addiert und das Aufräumen des Speichers angestoßen.

Die Bedienung gestaltet sich recht einfach:
Stellen Sie fest, wieviele Felder es in Ihrer Zeile gibt, und ob diese Alphadaten oder Zahlen enthalten sollen.
Anschließend halten Sie dieses Format in fo$ fest. Das sähe für die oben beschriebene Rechnung so aus:
fo$="99 999 aaaaaaaaaa 999.99 9999.99"
Sie sehen schon, daß numerische Felder mit '9' gekennzeichnet werden, Alphafelder mit 'a'. Die unterschiedliche Behandlung besteht darin, daß die Daten in Alphafeldern linksbündig wiedergegeben werden, numerische jedoch rechtsbündig in Übereinstimmung des Dezimalpunktes mit seiner Lage in fo$.
Nun brauchen Sie nur noch fi$ mit Ihren Daten zu versorgen und das Programm sooft anzuspringen, wie Sie Felder in fo$ definiert haben, natürlich mit immer anderen Daten in fi$.
Am Schluß der Prozedur haben Sie in fp$ die vollständige

Druckzeile.
Das könnte in Ihrem Programm dann so aussehen:

```
10 OPEN1,4
20 FO$="99  999  AAAAAAAAAA   999.99   9999.99"
30 FI$=STR$(PO):GOSUB48000:REM POSITIONSNUMMER
40 FI$=STR$(AN):GOSUB48000:REM ANZAHL
50 FI$=AR$:GOSUB48000:REM ARTIKELBEZEICHNUNG
60 FI$=STR$(EP):GOSUB48000:REM EINZELPREIS
70 FI$=STR$(AN*EP):GOSUB48000:REM GES.-PREIS
80 PRINT#1,FP$
```

Sie sehen, daß die Routine fünfmal angesprungen wird, nämlich sooft, wie Felder in fo$ definiert sind.
Die Definition in fo$ brauchen Sie natürlich nur einmal übergeben. Sie bleibt solange erhalten, bis Sie ein anderes Format in fo$ hinterlegen.
Ein kleiner Tip:
Übergeben Sie ausnahmsweise einmal für eine Zeile weniger Felder, als gemäß fo$ der Fall sein sollte, müssen Sie für die nächste Zeile fz%=0 setzen, damit fo$ wieder von vorne abgetastet wird.

### *5.1.3 Formatierte Programmlistings*

Wer kennt das nicht: da wird im Eifer des Gefechts bei der Programmierung mit jedem Byte gespart, so daß zum guten Schluß das Programm zwar läuft, im Programm aber weder Blanks (Leerzeichen) noch REMs enthalten sind. Soll jetzt an dem Programm nach einigen Wochen eine Änderung vorgenommen werden, dann beginnt das große Rätselraten.
Auch wenn jetzt ein Listing zur Hand ist, so ist es doch einigermaßen mühsam, die Struktur des Programms zu durch-

schauen. BASIC ist eben keine besonders strukturierte Programmiersprache.

Nun haben wir nicht die endgültige Lösung für dies Problem. Besser als jedes nachträgliche Mittel ist sicher gutes Planen und Vorarbeiten, um das Programm gleich im Entstehen in eine durchschaubare Form zu bekommen. Aber mit Hilfe des abgedruckten Basic-Programms sind Sie immerhin in der Lage, von auf Diskette oder Cassette abgespeicherten Basic-Programmen ein gut lesbares Listing zu erzeugen, welches zwischen allen Befehlsworten Leerzeichen einfügt und durch Einrückungen die Verschachtelung von FOR-NEXT-Schleifen zeigt. Außerdem wird nach jedem Doppelpunkt eine neue Druckzeile begonnen.

Um die Funktionsweise des Programms zu verstehen, sind allerdings etwas detailliertere Kenntnisse des Basic-Interpreters von VC-20 und C-64 und der Art der Programmspeicherung nötig.

Zuerst wollen wir einmal betrachten, wie ein Basic-Programm überhaupt im Speicher steht. Wenn Sie einen Maschinensprache-Monitor haben, können Sie die folgenden Schritte zum besseren Verständnis nachvollziehen.

Zuerst wird einmal der Monitor geladen. Danach geben wir die Programmzeile

```
10 PRINT"HALLO"
```

ein. Jetzt suchen wir im Speicher unser Programm.
Besitzer eines C 64 finden an den Speicherstellen $0800 bis $080F die Werte:

```
0800  00 0E 08 0A 00 99 22 48
0808  41 4C 4C 4F 22 00 00 00
```

Allerdings verschieben sich diese Adressen bei einem VC-20. Je nach eingesteckter Speichererweiterung ergeben sich die Adressen $0400, $1000 oder $1200. Beachten Sie bitte als Besitzer eines VC-20 diese geänderten Adressbereiche, da wir im folgenden von der Adresslage im C 64 ausgehen.

Was hat diese Zeichenfolge nun zu bedeuten?
Nun, die Speicherzelle $0800 muß immer Null sein; sie kennzeichnet den Beginn des Basic-Programmspeichers. Jeder andere Wert in dieser Speicherzelle ergibt bei RUN einen SYNTAX ERROR.
Der Inhalt der Adressen $0801 und $0802 stellen den Bezug zur nächsten möglichen Programmzeile dar. In diesem konkreten Fall bedeuten die Werte, daß die nächste Basic-Zeile im Speicher an der Adresse $080E beginnen wird. Die in diesen Speicherplätzen enthaltene Information wird für GOTOs, GOSUBs, aber auch für FOR-NEXT-Schleifen benutzt. Man bezeichnet diese Adressen auch als Link-Adressen.
Die folgenden zwei Speicherplätze enthalten die aktuelle Zeilennummer, auch wieder im Format Lowbyte-Highbyte. In unserem Beispiel erhielten wir $00A0. Als Dezimalzahl ausgedrückt ergibt das 10.
Jetzt kommen wir zu dem eigentlichen Befehl der Zeile, der Print-Anweisung. Wie man es aber dreht und wendet, die Buchstaben für den Befehl sind in diesem Speicherausschnitt nicht zu entdecken. Statt dessen findet sich ein Byte mit dem Wert 99. Um herauszubekommen, was es mit dieser 99 auf sich hat, können wir ja diesen Wert mit dem Monitor einmal verändern, und anschließend prüfen, ob sich irgendwelche Auswirkungen auf unser Basic-Programm ergeben. Ändern wir doch einmal die 99 einmal in eine 97, verlassen den Monitor

und listen das Basic-Programm.
Jetzt haben wir plötzlich eine ganz andere Zeile im Speicher stehen. Die Print-Anweisung ist verschwunden, statt dessen lautet die Zeile jetzt :

```
10 POKE"HALLO"
```

Wie man sehen kann, werden die Anweisungen in einem Programm in einen Ein-Byte-Code übersetzt. Dabei hat jede Anweisung ihren speziellen und einmaligen Code. Durch probieren kann jetzt leicht überprüft werden, welchem Befehl welcher Code zugewiesen ist. Sie werden schnell feststellen, daß die Codes, auch Tokens genannt, immer größer als $7F sind.
Das bedeutet, daß bei allen Tokens das höchstwertige Bit gesetzt ist. Dadurch ist das Auffinden der Tokens in einem Programm sehr einfach, da in Basic-Programmen Zeichen mit gesetztem höchstwertigem Bit sonst nur in Anführungszeichen, also in Strings vorkommen. Steht also ein solches Zeichen mit gesetztem höchstwertigem Bit im Programm und ist die Anzahl der Anführungszeichen in der Zeile bis zu diesem Punkt Null oder eine grade Zahl, dann muß es sich um ein Token handeln.

Zusätzlich zu den Tokens existiert im ROM des Rechners eine Tabelle mit allen Befehlen des Basic. Diese Tabelle benutzt der Computer bei der Eingabe von Programmen, um die Syntax der eingegebenen Befehle zu prüfen. Bei der Eingabe einer Zeile werden die Zeichen direkt überprüft, ob in diesem Text ein Befehl enthalten ist. Ist das der Fall, dann wird dieser Befehl sofort in das entsprechende Token gewandelt. Diese Tabelle werden wir später in unserem Programm ebenfalls benutzen.

Doch kommen wir nach so langer Vorrede nun zu dem Programm.

Das abgedruckte Listing ist in der vorliegenden Form unmittelbar auf dem C 64 lauffähig, für den VC-20 (mit mindestens 3K-Erweiterung!) ist die in der Zeile 470 enthaltene Änderung an dem Programm vorzunehmen.

```
100 DIMBT$(75)
110 LM$="      ":NF$="  " :BL$=".   "
120 FORI=1TO6:HZ$=HZ$+LM$:NEXT
130 HZ$=HZ$+"SEITE "
140 F1$=CHR$(145)+LM$
150 SN=1
160 PRINTCHR$(147)
170 PRINT
180 PRINT
190 PRINT"      LIST - PROGRAMM"
200 PRINT
210 PRINT
220 INPUT"    NAME DES PROGRAMMS";NA$
230 NP$=NA$+",P,R"+CHR$(34)
240 PRINT
250 PRINT"       AUSGABE AUF "
260 PRINT"   "CHR$(18)"D"CHR$(146)"RUCKER ODER "CHR$(18)"F"CHR$(146)"LOPPY";
270 INPUT GA$
280 IFGA$=""THENGA=3:GOTO360
290 IFGA$="F"THENGA=8:NL$=NA$+",S,W"+CHR$(34):GOTO360
300 IFGA$="D"THENGA=4
310 PRINT:INPUT"    ZEILEN PRO SEITE ";ZN:
320 ZN=(ZN=0)*-66+ZN
330 PRINT:INPUT"    UEBERSCHRIFT     ";UE$
340 HZ$=RIGHT$(HZ$,LEN(HZ$)-LEN(UE$))
350 UE$=UE$+HZ$
360 PRINT
370 INPUT"    LINKER RAND      ";L
380 PRINT
```

```
390 PRINT
400 FORI=0TOL
410 LM$=LM$+" "
420 NEXT
430 OPEN15,8,15
440 OPEN1,8,4,NP$
450 GOSUB1150
460 I=0
470 FORA=41118TO41372 REM VC-20 FORA=49310TO49564
480 X=PEEK(A)
490 IFX>127THEN520
500 BT$(I)=BT$(I)+CHR$(X)
510 GOTO560
520 X=XAND127
530 BT$(I)=BT$(I)+CHR$(X)
540 PRINTF1$;BT$(I);LM$
550 I=I+1
560 NEXT
570 PRINT
580 PRINT
590 BT$(11)=BT$(11)+"   "
600 BT$(39)=BT$(39)+" "
610 REM *********  PROGRAMM LESEN   ***********
620 PRINTCHR$(147)
630 IFGA<8THENOPEN4,GA :GOTO650
640 OPEN4,GA,3,NA$+".LST,S,W" .
650 IFGA<>3THENPRINT"AUSGABE ERFOLGT AUF GERAET #";GA
660 PRINT#4
670 GOSUB1280
680 GOSUB1180
690 GOSUB1180
700 REM PRG-START SKIP
710 REM *********    HAUPTSCHLEIFE  ***********
720 GOSUB1180
```

```
730 GOSUB1180    : REM SKIP LINK ADRESSE
740 REM
750 GOSUB1180
760 LB=A
770 REM ZEILE  LOW - BYTE
780 GOSUB1180
790 HB=A
800 REM ZEILE HIGH - BYTE
810 ZN$=STR$(HB*256+LB)
820 LN$=LEFT$(LM$,LEN(LM$)-LEN(ZN$))+ZN$+NF$
830 NN=0
840 REM *********   ZEILENSCHLEIFE  ***********
850 IFLEN(LN$)<70THEN910
860 PRINT#4,LN$;:IFNOT QFTHENPRINT#4
870 IFQFTHENPRINT#4,CHR$(34)
880 LN$=LM$+NF$:IFQFTHENLN$=LN$+CHR$(34)
890 NN=0
900 GOSUB1220
910 GOSUB1180
920 REM EIN BYTE HOLEN
930 IFA=0THENPRINT#4, LN$:LN$="":QF=0:RF=0:GOSUB1220:GOTO720
940 IF NOT RF THEN 970    : REM BEI REM REST SO LESEN
950 LN$=LN$+CHR$(A)
960 GOTO850
970 IFA=32ANDNOTQFTHEN 850
980 IF A<>44 THEN1010
990 IF A=44 AND NN=0THEN1010
1000 IFNF$<>"  "THENNF$=LEFT$(NF$,LEN(NF$)-4)
1010 IFA=34THENQF=NOTQF
1020 IFA=58ANDNOTQFTHENPRINT#4,LN$;":":LN$=LM$+NF$:NN=0:GOSUB1220:GOTO850
1030 IFA>127ANDNOTQFTHENGOSUB1060:GOTO850
1040 LN$=LN$+CHR$(A)
1050 GOTO850
1060 IFRIGHT$(LN$,1)<>" "THENLN$=LN$+" "
```

```
1070 IFA=128+39THENPRINT#4,LN$:LN$=LM$+NF$ :GOSUB1220  : REM THEN
1080 IFA=129THENNF$=NF$+BL$ : REM BL$= 4 SPACES FUER FOR NEXT
1090 IFA=130THENNN=1: GOSUB1130
1100 IFA=143THENRF=NOTRF    : REM REMFLAG ON
1110 LN$=LN$+BT$(A-128)+" "
1120 RETURN
1130 IFNF$<>"  "THENNF$=LEFT$(NF$,LEN(NF$)-4):LN$=LEFT$(LN$,LEN(LN$)-4)
1140 RETURN
1150 INPUT#15,A$,B$,C$,D$
1160 IFA$<>"00"THENPRINTB$:CLOSE1:CLOSE4:CLOSE15:END
1170 RETURN
1180 GET#1,A$
1190 A=ASC(A$+CHR$(0))
1200 IF ST <>0  THENCLOSE1:CLOSE4:CLOSE15:END
1210 RETURN
1220 ZZ=ZZ+1
1230 IFZZ=ZN THEN1250
1240 RETURN
1250 FORI=1TO72-ZN
1260 PRINT#4
1270 NEXT
1280 PRINT#4,LM$;"  ";UE$;SN
1290 PRINT#4
1300 PRINT#4
1310 ZZ=3
1320 SN=SN+1
1330 RETURN
```

Um die Funktion des Programms zu verstehen, sind einige Kenntnisse des Aufbaus eines Basic-Programms nötig. Darum soll es etwas ausführlicher als üblich besprochen werden. Auch wenn Sie das Programm erweitern wollen, ist die Beschreibung sicher hilfreich. Eine mögliche Erweiterung wäre z.B. die Umsetzung aller reverser Zeichen in die Form

CHR$(x). Damit wäre das Programm dann auch auf Druckern einsetzbar, die diese Zeichen nicht darstellen können.
In den Zeilen 100 - 150 werden einige wichtige Variablen initialisiert. Das Array BT$ enthält später die einzelnen Basic-Befehlsworte, LM$ definiert einen linken Rand.
Nach dem Löschen des Bildschirms wird nach dem Namen des zu listenden Programms und nach dem Ausgabegerät gefragt. Als Ausgabegerät stehen natürlich der Drucker, aber auch die Floppy zur Verfügung. Letztere Ausgabe wurde gewählt, damit die Möglichkeit besteht, ein mit dem Programm aufbereitetes Listing in eine Textverarbeitung zu übernehmen, und hier dann Kommentare in das Listing zu bringen. Als Filename wird in diesem Fall der zuvor eingegebene Programmname genommen.
Wird die Frage nach dem Ausgabegerät nur mit RETURN beantwortet, so ist als Ausgabegerät der Bildschirm angesprochen. Dadurch ergibt sich die Möglichkeit, sich vorab über das Erscheinungsbild des Listings zu informieren.
Bei der Wahl Bildschirm oder Floppy werden die Zeilen 310 bis 350 übersprungen. In diesen Zeilen werden bei Druckerausgabe die Parameter für Zeilen pro Seite und ein Überschriftstext abgefragt. Der Überschriftstext erscheint dann beim Ausdruck zusammen mit der Seitennummer auf jeder ausgedruckten Seite.
Wird die Abfrage der Zeilen pro Seite nur mit RETURN beantwortet, so ist eine Anzahl von 66 Zeilen eingestellt.
In der Zeile 370 wird ein gewünschter linker Rand abgefragt. Hier ist ein Wert von 6 bis 10 empfehlenswert, damit das erzeugte Listing nicht durch eventuelle Abheftlöcher 'verhunzt' wird. Auch diese Eingabe kann aber nur mit RETURN beantwortet werden. In diesem Fall wird kein zusätzlicher Platz für Heftlöcher o.Ä. vorgesehen.
Nach dem Eröffnen des Fehlerkanals und der Programmdatei wird zur Zeile 1150 verzweigt, wo eine eventuelle Fehlermeldung abgefragt und angezeigt wird. Häufigste Fehlerur-

sache ist wohl die Meldung 'FILE NOT FOUND'.
Nachdem auf diese Weise festgestellt wurde, daß die gewünschte Datei auch vorhanden ist, wird in der FOR-NEXT-Schleife in den Zeilen 470-560 die schon erwähnte Liste der Befehlsworte aus dem Basic-ROM gePEEKd. Diese Liste hat allerdings eine kleine Besonderheit. Alle Befehlsworte stehen direkt hintereinander. Um jetzt die einzelnen Befehle voneinander trennen zu können, ist jeder letzte Buchstabe eines Befehlswortes mit gesetztem achtem Datenbit in dieser Tabelle gespeichert. Dieser Fall wird mit der Zeile 490 erkannt und entsprechend zur Zeile 520 verzweigt. Hier wird dies Bit gelöscht und der Befehl wird auf dem Bildschirm angezeigt.
Anschließend wird der Variablenindex erhöht und der nächste Befehl wird aus dem Speicher gePEEKd.
Zu guter letzt werden die Befehle IF und THEN insgesamt 5 Zeichen lang gemacht, d.h. mit Leerzeichen aufgefüllt.
Und jetzt gehts los, wir holen die Bytes von der Diskette.
Diese Aufgabe erledigt das Unterprogramm in den Zeilen 1180 - 1210. Mit jedem Aufruf dieses Unterprogramms wird ein Zeichen von der Diskette geholt und in der Variable A zur Verfügung gestellt.
Zuerst wird diese Routine zweimal aufgerufen, der Wert in A wird aber nicht weiter verwendet. Das hat seine Ursache in dem Umstand, das die ersten beiden Bytes eines jeden Programmfiles die Ladeadresse angeben. Diese Werte sind für uns aber nicht interessant.
Jetzt kommt das Programm in die bei Zeile 710 beginnende Hauptschleife. Auch hier werden zuerst zwei Bytes geholt und nicht weiter verwendet. Es handelt sich bei diesen Werten um die schon beschriebene Link-Adresse. Die nächsten beiden Bytes allerdings sind da schon wichtiger, sie enthalten ja die Zeilennummer. Diese wird aus den geholten Werten in den Zeilen 810, 820 berechnet und formatiert.

In der Zeile 840 beginnt nun die Zeilenschleife, in der alle zu dieser Zeile gehörenden Bytes geholt und behandelt werden.
Zuerst wird einmal geprüft, ob der bisher zusammengesetzte String LN$ noch kleiner als 70 Zeichen ist. Ist diese Anzahl überschritten, dann wird der bisherige Anteil der Zeile ausgegeben und das Unterprogramm in der Zeile 1220 aufgerufen. Hier wird geprüft, ob die Anzahl der gedruckten Zeilen das Maximum schon erreicht hat und bei Bedarf ein Seitenvorschub ausgegeben.
Nach dieser Überprüfung wird nun in der Zeile 910 ein Byte geholt und jetzt beginnt die eigentliche Arbeit. Je nach Wert der Variable A müssen jetzt verschiedene Aktionen eingeleitet werden. Wir wollen der Reihe nach sehen, was im Einzelnen geschieht.
Zuerst wird geprüft, ob A gleich 0 ist. Das ist der Fall beim Erreichen des Zeilenendes. In diesem Fall muß LN$ ausgegeben und der Zeilenzähler für die Druckzeilen erhöht werden.
Ist A gleich 32, dann wurde der Wert eines Leerzeichens übergeben. Wenn sich dies Leerzeichen im Programmtext befindet, dann wird es übersprungen, es wird das nächste Byte geholt. Befindet sich dieses Blank dagegen in Anführungszeichen, dann wird es später dem Zeilenstring LN$ zugefügt. Die Unterscheidung wird mit der Variable QF vorgenommen, die weiter unten erläutert wird.
Danach folgt eine Prüfung auf einen Wert von 44 entsprechend einem Komma. Diese Prüfung ist wichtig im Zusammenhang mit dem NEXT-Befehl mit mehreren durch Komma getrennten Variablen.
In der Zeile 1010 wird A auf den Wert 34 gleich Anführungszeichen geprüft. Ist A gleich 34, dann wird mit der NOT-Operation der Variablen QF der jeweils entsprechende Wert Logisch wahr = -1 oder Logisch nicht wahr = 0

zugewiesen. Mit jedem Anführungszeichen wird dieser Wert also getauscht. Auf diese Art kann ohne großen Programmieraufwand entschieden werden, ob ein Zeichen im Programmtext oder in Anführungszeichen steht.
Anschließend erfolgt eine Prüfung auf den Doppelpunkt gleich 58. Auch hier muß wieder an Hand von QF festgestellt werden, ob der Doppelpunkt im Programmtext oder in einem Literal, also einem Textstring, steht. Steht der Doppelpunkt im Programm, dann wird die Zeile LN$ mit dem Doppelpunkt zum Ende gedruckt und eine neue Druckzeile begonnen. Auf diese Weise werden Zeilen mit mehreren durch Doppelpunkt getrennten Statements auf mehrere Druckzeilen verteilt und damit Übersichtlicher.
Ist aber das empfangene Zeichen größer als 127 und steht es im Programmtext, dann handelt es sich um ein Token. Steht das Zeichen jedoch in Anführungsstrichen, dann wird es dem LN$ zugefügt.
Der Wert 167 (Zeile 1070) steht für das Token THEN. Stellt das Programm ein THEN-Token fest, dann wird die bisherige Zeile gedruckt, um die IF-THEN-Struktur sichtbarer zu machen.
In der Zeile 1080 wird der Interpretercode für FOR gefunden. Beim Finden dieses Tokens wird der String NF$ um die in BL$ definierte Zeichenfolge erweitert. Auf diese Weise werden FOR-NEXT-Schleifen eingerückt und dadurch übersichtlicher.
In der Zeile 1090 wird auf das Vorhandensein eines NEXT-Befehls geschaut. Wird dieser Befehl registriert, dann werden die beim FOR-Befehl zugefügten Zeichen im NF$ in der Zeile 1130 wieder entfernt. Um auch die Besonderheit des NEXT mit mehreren durch Komma getrennten Variablen zu berücksichtigen, wird zusätzlich das NN-Flag auf 1 gesetzt.
Die Zeile 1100 schließlich prüft auf das Vorhandensein eines REM-Statements. In diesem Fall wird das RF (Remflag) gesetzt, um nach dem REM folgende Zeichen ohne weitere

Überprüfung und evtl. Wandlung auszudrucken.
Nach dem Durchlaufen der Zeilen 1060 bis 1100 wird dem Zeilenstring LN$ die Zeichenkette für den entsprechenden Interpretercode zugeführt. Dazu wird das entsprechende Arrayelement direkt über das Token-128 addressiert.
Zusätzlich wird ein Leerzeichen angehängt, um die Lesbarkeit zu erhöhen. Danach wird in die Zeilenschleife zurückgesprungen.

Auf den ersten Blick sieht das Programm sehr aufwendig aus. Im Grunde werden aber nur die Funktionen in Basic nachvollzogen, die der Computer bei jedem LIST ausführt, mit dem Unterschied, daß diese 'LIST'-Funktion um einige Zusatzfunktionen erweitert wurde.

## *5.2 Textverarbeitung*

Textverarbeitung gehört neben der Datei-Verwaltung mit Sicherheit zu den Haupteinsatzgebieten der Computer. Da bei Textverarbeitung auch immer ein Drucker im Spiel ist, haben wir uns des Themas einmal angenommen, und es ist das kleine Programm 'Minitext' entstanden. Sicher kann dies Programm nicht mit richtigen Textverarbeitungsprogrammen wie z.B. TEXTOMAT konkurrieren. Aber zum schnellen Tippen eines Briefs oder einer Notiz ist das Programm sicher einsetzbar. Da es in Maschinensprache geschrieben ist, können Sie sicher auch Anregungen finden, wie die Druckersteuerung von Maschinensprache aus zu bewerkstelligen ist.
Das zweite Programm ist eine Textverarbeitung ganz besonderer Art. Mit diesem Programm lassen sich kleine Texte wahrhaft groß in Form bringen. Die Buchstaben werden in einem vielfachen der sonstigen Größe ausgegeben, sicher recht interessant für das Erstellen von Plakaten.

### *5.2.1 MINITEXT - ein einfacher Texteditor*

Wo ein Drucker und ein Computer nebeneinander stehen, da darf auch eine Textverarbeitung nicht fehlen. Nun haben Textverarbeitungsprogramme von der Leistungsfähigkeit z.B. des TEXTOMAT einen Umfang, daß allein mit einem Assemblerlisting leicht 300 und mehr Seiten gefüllt werden könnten. Obwohl wir es nicht ausprobiert haben, schätzen wir ein entsprechendes Programm in Form von Data-Zeilen auf einen Umfang von mindestens 40 bis 50 Seiten. Vermutlich würde auch hartgesotten 'Abtippern' bei einem solchen Umfang die Lust vergehen, wenn sich solche Programme im verfügbaren Speicher überhaupt realisieren lassen.

Also haben wir nach einem Weg gesucht, ein passables Programm mit weniger Aufwand zu schreiben. Der oft beschrittene Weg, eine Textverarbeitung in Basic zu schreiben, wurde von vorn herein ausgeschlossen. Bei einem so intensiven Gebrauch der String-Funktionen wie gerade bei solchen Programmen tritt zu häufig die sogenannte Garbagge Collection in Kraft.
Bei der Garbagge Collection werden die nicht mehr benötigten Strings im Speicher entfernt. Dieser Vorgang kann je nach Programm und Anzahl der verwendeten Variablen zwischen wenigen Sekunden bis zu über 30 Minuten (!) dauern.
Während dieser Zeit nimmt der Computer keine Eingaben von der Tastatur an. Er scheint abgestürzt und sicher wird der eine oder andere Leser schon kopfschüttelnd vor seinem Gerät gesessen und sich über die 'stotternde' Abarbeitung seines Programms gewundert haben.

Als zweiter Punkt, der gegen einen Basic-Texteditor spricht, ist die relative Größe eines solchen Programms. Allein um die wichtigsten Funktionen ausführen zu können, muß ein solches Programm einen gewissen Umfang haben.

Es gibt einen weiteren, sehr mühsamen Weg, der sicher von Ihnen schon versucht worden ist. Bei dieser Methode wird der Text wie ein normales Basic-Programm in Print#-Befehle geschrieben. Diese Methode ist allerdings sehr unbefriedigend, da Sie bei jeder Zeile die Zeilennummer und die Befehlsfolge 'PRINT# LF, "' schreiben müssen. Auch müssen Sie bei der Eingabe stets darauf achten, daß die eingegebene Zeile nicht zu lang wird. Das ist aber beim flüssigen Schreiben sehr hinderlich.
Andererseits steht einem mit dieser zuletzt beschriebenen Methode der sehr komfortable Bildschirmeditor und die Befehle zum Speichern und Laden von Texten zur Verfügung.

Unsere Überlegung war nun, den vorhandenen Bildschirm-Editor zu nutzen, und auf die zuvor geschilderten Nachteile weitestgehend zu verzichten.
Entstanden aus diesen Überlegungenen ist ein kleines Maschinenprogramm von etwas mehr als 500 Bytes (genau 508).
Wenn man sich einmal ansieht, wie Texte in Print-Anweisungen stehen, so stellt man fest, daß vor dem Text immer Anführungsstriche stehen müssen. Auch ohne Anführungsstriche lassen sich hinter eine Zeilenzahl Texte eingeben, dann allerdings geschehen erstaunliche Dinge, wenn im Groß-Kleinschrift-Modus Großbuchstaben, also geshiftete Zeichen eingegeben werden. Diese Zeichen verschwinden einfach, oder es stehen plötzlich Befehlsworte des Basic im Text.
Gibt man nun eine Zeilennummer und direkt darauf folgend Anführungsstriche ein, dann kann man Text so eingeben, daß Groß- und Kleinbuchstaben beim anschließenden Listen erhalten bleiben. Der Computer befindet sich im sogenannten Quote-Modus.
Allerdings gibt es nun ein weiteres Problem. Die Cursor-Steuertasten, die Tasten RUN/STOP und CLR/HOME sowie die Funktionstasten erzeugen reverse Grafikzeichen oder Buchstaben. Ein Editieren der Programmzeile mit Hilfe der Cursortasten ist damit nicht mehr möglich, es sei denn, der Quote-Modus wird durch weitere Anführungszeichen wieder aufgehoben.
Alle diese Punkte werden vom ersten Teil des Maschinensprache-Programms abgearbeitet. Die Zeilennummern werden selbstständig in der Art des Autobefehls einiger Befehlserweiterungen von Computer erzeugt. Auch die Anführungszeichen brauchen Sie nicht einzugeben, diese Arbeit wird ebenfalls vom Programm übernommen.
Der eingeschaltete Quote-Modus wird durch eine spezielle Speicherzelle im Ram, genauer in der Zero-Page, des Rechners signalisiert. Nachdem die Anführungszeichen auf dem

Bildschirm erscheinen, wird diese Speicherzelle so geladen, daß dem Computer ein ausgeschalteter Quote-Modus vorgegaukelt wird.
Jetzt bleibt nur noch die Überprüfung der Zeilenlänge. Man kann häufig beobachten, daß Anwender des berühmten 'Zwei-Finger-Such-Systems' mit den Augen auf der Tastatur kleben. Vor allem bei fließender Texteingabe wird nur selten ein Blick auf den Bildschirm geworfen. Darum eignet sich zur Überprüfung der Zeilenlänge nur ein akustisches Signal.
In der vorhin schon erwähnten Zero-Page (so werden die ersten 256 Bytes des Ram bezeichet) befindet sich eine Speicherzelle, die angibt, wie viele Zeichen in der Basic-Zeile stehen, die gerade eingegeben wird. Unser kleines Programm schaut nun regelmäßig (im Interrupt, also 60 mal in der Sekunde) in dieser Speicherzelle nach, ob ein bestimmter Wert erreicht ist. Sobald dies der Fall ist, wird über den Fernsehlautsprecher ein akustisches Signal erzeugt. Jetzt sollten Sie spätestens mit dem Ende des nächsten Worts die Return-Taste betätigen. Bei sehr langen Worten ist es evtl. sogar nötig, das Wort zu trennen, und den Rest in die nächste Zeile zu schreiben.
Sobald die Return-Taste gedrückt ist, wird sofort eine neue Zeilennummer mit Anführungszeichen geschrieben.

Der zweite Teil des Programms ist für den Ausdruck des so erstellten Textes verantwortlich. Natürlich wäre auch eine Ausgabe mittels LIST möglich, aber wer möchte schon gern einen Brief schreiben, bei dem jede Zeile mit einer Zeilennummer beginnt. Dieser Programmteil druckt die störenden Zeilennummern und die Anführungsstriche natürlich nicht mit. Statt dessen wird ein linker Rand erzeugt, so daß die Buchstaben nicht unmittelbar an der Abreißkante beginnen.
Um die Handhabung des Programms zu erleichtern, ist es in

den Basic-Interpreter eingebunden. Nach dem Aufruf des Programms mit SYS stehen zwei neue Befehle zur Verfügung. Es sind dies der !GET-Befehl um Zeilen einzugeben sowie der !PRINT-Befehl zum Ausdrucken des Textes. Beim letzteren ist auch die Abkürzung !? möglich, da das Fragezeichen ja als Abkürzung des Print-Befehls zugelassen ist.
Aber jetzt soll erst einmal das Programm kommen. Haben Sie schon ein wenig Übung in der Assembler-Programmierung, dann sollten Sie kaum Schwierigkeiten haben, das Programm nach ihren Bedürfnissen zu modifizieren oder zu erweitern.

```
c000                *=   $c000
c000        beepflg =    2
c000        formfeed =   6
c000        lm      =    10
c000        cr      =    $0d
c000        lo      =    $14
c000        hi      =    $15
c000        txtan   =    $2b
c000        txtend  =    $2d
c000        fac     =    $62
c000        text    =    $71
c000        chrget  =    $73
c000        chrgot  =    $79
c000        prtcode =    $99
c000        getcode =    $a1
c000        filnm   =    $b7
c000        lgfnr   =    $b8
c000        secad   =    $b9
c000        devad   =    $ba
c000        line    =    $fb
c000        incr    =    $fd
c000        zeilen  =    $fe
```

```
c000            bufferl  =    $101
c000            buffer2  =    $200
c000            vector   =    $0302
c000            vectorl  =    $0308
c000            irqvec   =    $0314
c000            basloop  =    $a474
c000            oldvec   =    $a483
c000            mainloop =    $a486
c000            goon     =    $a569
c000            weiter   =    $a576
c000            basstrt  =    $a68e
c000            execold  =    $a7e7
c000            chkcom   =    $aefd
c000            syntax   =    $af08
c000            illquant =    $b248
c000            getpar   =    $b7eb
c000            intfloat =    $bc49
c000            floatasc =    $bddd
c000            sid      =    $d400
c000            clrscrn  =    $e544
c000            irqold   =    $ea31
c000            open     =    $ffc0
c000            close    =    $ffc3
c000            ckout    =    $ffc9
c000            clrch    =    $ffcc
c000            input    =    $ffcf
c000            bsout    =    $ffd2
c000            stop     =    $ffe1
                 ;******* hier gehts los *********
                 ;
c000 a9 41              lda  #<testin
c002 a0 c0              ldy  #>testin
c004 8d 08 03           sta  vectorl
c007 8c 09 03           sty  vectorl+1
```

```
c00a a9 80              lda  #128          ; shft-commodore disable
c00c 8d 91 02           sta  $0291
c00f a9 17              lda  #$17          ; gross-klein einschalten
c011 8d 18 d0           sta  $d018
c014 a9 00              lda  #0
c016 a8                 tay
c017 91 2b              sta  ($2b),y       ; basic-programm löschen
c019 c8                 iny
c01a 91 2b              sta  ($2b),y
c01c a5 2b              lda  $2b           ; und vectoren setzen
c01e 18                 clc
c01f 69 02              adc  #$02
c021 85 2d              sta  $2d
c023 a5 2c              lda  $2c
c025 69 00              adc  #0
c027 85 2e              sta  $2e
c029 20 8e a6           jsr  basstrt
c02c 20 44 e5           jsr  clrscrn       ; clear screen
c02f a9 80              lda  #$80
c031 8d 8a 02           sta  650           ; repeat alle tasten
c034 a9 64              lda  #100          ; erste zeilennummer
c036 85 fb              sta  line
c038 a9 00              lda  # 0
c03a 85 fc              sta  line+1
c03c a9 0a              lda  #10           ; zeilenabstand
c03e 85 fd              sta  incr
c040 60                 rts                ; und zurueck
                        ; prüfen auf !
c041 20 73 00 testin    jsr  chrget
c044 c9 21              cmp  #"!"
c046 f0 06              beq  test2         ; wenn ! dann weiter testen
c048 20 79 00           jsr  chrgot
c04b 4c e7 a7           jmp  execold       ; sonst normaler befehl
c04e 20 73 00 test2     jsr  chrget        ; nächstes zeichen holen
```

```
c051 c9 a1            cmp  #getcode     ; ist es !get ?
c053 f0 0a            beq  get
c055 c9 99            cmp  #prtcode     ; ist es !print ?
c057 f0 03            beq  prt
c059 4c 08 af         jmp  syntax       ; weder noch dann syntax error
c05c 4c 41 c1 prt     jmp  prt1
c05f 20 73 00 get     jsr  chrget       ; war !get
c062 f0 0d            beq  10           ; weitere parameter?
c064 20 eb b7         jsr  getpar       ; dann holen
c067 86 fd            stx  incr         ; und merken
c069 a5 14            lda  lo
c06b 85 fb            sta  line
c06d a5 15            lda  hi
c06f 85 fc            sta  line +1
c071 a9 80    10      lda  #128
c073 8d 91 02         sta  $0291
c076 a9 17            lda  #$17
c078 8d 18 d0         sta  $d018
c07b a9 85            lda  #<auto       ; vektor für eingabe eine
c07d 8d 02 03         sta  vector       ; zeile auf auto-routine
c080 a9 c0            lda  #>auto       ; verbiegen
c082 8d 03 03         sta  vector+1
c085 20 8b c0 auto    jsr  autonum
c088 4c 86 a4         jmp  mainloop
c08b ad 14 03 autonum lda  irqvec       ; beep eingeschaltet?
c08e c9 03            cmp  #<beepon
c090 f0 0c            beq  w0
c092 78               sei
c093 a9 03            lda  #<beepon     ; interrupt vector
c095 8d 14 03         sta  irqvec       ; auf beep-routine
c098 a9 c1            lda  #>beepon     ; verbiegen
c09a 8d 15 03         sta  irqvec+1
c09d 58               cli
c09e a5 fb    w0      lda  line         ; zeilennummer holen
```

```
c0a0 a6 fc          ldx  line+1       ; und in ascii-format
c0a2 85 63          sta  fac+1        ; wandeln
c0a4 86 62          stx  fac
c0a6 a2 90          ldx  #$90
c0a8 38             sec
c0a9 20 49 bc       jsr  intfloat
c0ac 20 dd bd       jsr  floatasc
c0af a2 00          ldx  #0
c0b1 bd 01 01 11    lda  buffer1,x
c0b4 f0 09          beq  12
c0b6 9d 00 02       sta  buffer2,x    ; zeilennummer in eingabebuffer
c0b9 20 d2 ff       jsr  bsout        ; und auf bildschirm
c0bc e8             inx
c0bd d0 f2          bne  11
c0bf a5 fb    12    lda  line         ; zeilennummer erhöhen
c0c1 18             clc
c0c2 65 fd          adc  incr
c0c4 85 fb          sta  line
c0c6 90 02          bcc  13
c0c8 e6 fc          inc  line +1
c0ca a9 20    13    lda  #" "         ; leerzeichen
c0cc 9d 00 02       sta  buffer2,x    ; in eingabebuffer
c0cf 20 d2 ff       jsr  bsout        ; und auf bildschirm
c0d2 e8             inx
c0d3 a9 22          lda  #34          ; anführungsstriche
c0d5 9d 00 02       sta  buffer2,x    ; in egb
c0d8 e8             inx
c0d9 20 d2 ff       jsr  bsout        ; und auf bs
c0dc a9 00          lda  #0
c0de 85 d4          sta  $d4          ; quote-flag loeschen
c0e0 20 cf ff       jsr  input
c0e3 c9 0d          cmp  #cr          ; nur return gedrückt?
c0e5 f0 03          beq  14
c0e7 4c 69 a5       jmp  goon         ; wenn nicht, dann weiter
```

```
c0ea a5 fb     14       lda  line        ; sonst zeilennummer wieder
c0ec e5 fd              sbc  incr        ; erniedrigen;
c0ee 85 fb              sta  line
c0f0 b0 02              bcs  15
c0f2 c6 fc              dec  line+1
c0f4 a9 83     15       lda  #<oldvec    ; und eingabe vektor
c0f6 a0 a4              ldy  #>oldvec    ; rücksetzen
c0f8 8d 02 03           sta  vector
c0fb 8c 03 03           sty  vector+1
c0fe a2 00              ldx  #0          ; eingabebuffer löschen
c100 4c 76 a5           jmp  weiter
c103 a5 d3     beepon   lda  $d3          ; cursor-spalte
c105 c9 40              cmp  #64         ; 64 zeichen ?
c107 f0 07              beq  beep        ; dann beepen
c109 a9 00              lda  #0          ; sonst flag löschen
c10b 85 02              sta  beepflg     ; darf wieder beepen
c10d 4c 31 ea           jmp  irqold      ; alte irq-routine
c110 a5 02     beep     lda  beepflg     ; darf beepen?
c112 d0 2a              bne  end         ; wenn nicht dann weiter
c114 a9 08              lda  # 8
c116 8d 18 d4           sta  sid+24      ; lautstaerke
c119 a9 08              lda  # 8
c11b 8d 03 d4           sta  sid+3       ; tastverh
c11e a9 00              lda  # 0
c120 8d 00 d4           sta  sid         ; lowbyte freq
c123 a9 78              lda  # 120
c125 8d 01 d4           sta  sid+1       ; highbyte freq
c128 a9 ff              lda  # 255
c12a 8d 05 d4           sta  sid+5       ; attack/decay
c12d a9 f9              lda  # 249
c12f 8d 06 d4           sta  sid+6       ; sust/release
c132 a9 41              lda  # 65
c134 8d 04 d4           sta  sid+4       ; steuerreg
c137 a9 40              lda  # 64
```

```
c139 8d 04 d4          sta  sid+4        ; steuerreg
c13c 85 02             sta  beepflg      ; einmal und nicht wieder
c13e 4c 31 ea end      jmp  irqold       ; irq exit
            ;*****  print   routine  *****
c141 a9 04    prtl     lda  #4           ; geraetenr
c143 85 ba             sta  devad
c145 a9 7e             lda  #126         ; filenr
c147 85 b8             sta  lgfnr
c149 a9 00             lda  #0           ; filename
c14b 85 b7             sta  filnm
c14d a9 00             lda  #0           ; sekundearadresse
c14f 85 b9             sta  secad
c151 20 c0 ff          jsr  open         ; file eröffnen
c154 a6 b8             ldx  lgfnr
c156 20 c9 ff          jsr  ckout        ; ausgabe auf drucker
c159 a5 2d             lda  txtan+2      ;  = variablenstart
c15b 38                sec
c15c e9 02             sbc  #2           ; minus 2
c15e c5 2b             cmp  txtan        ; wie basic-start?
c160 d0 0b             bne  druckl
c162 a5 2e             lda  txtan+3
c164 e9 00             sbc  #0
c166 c5 2c             cmp  txtan+1
c168 d0 03             bne  druckl
c16a 4c fa c1          jmp  endl         ; kein text im speicher
c16d a5 2b    druckl   lda  txtan        ; skip link,
c16f 18                clc               ; zeilennummer und
c170 69 05             adc  #5           ; anführungszeichen
c172 85 71             sta  text         ; position im text merken
c174 a5 2c             lda  txtan+1      ; highbyte auch merken
c176 85 72             sta  text+1
c178 a9 42             lda  #66          ; zeilen pro  seite
c17a 85 fe             sta  zeilen
c17c a0 0a             ldy  #lm
```

```
c17e a9 20          lda  #" "        ; linken rand drucken
c180 20 d2 ff lmloopl jsr  bsout
c183 88             dey
c184 d0 fa          bne  lmloopl
c186 a9 11          lda  #17         ; cursor down
c188 20 d2 ff       jsr  bsout       ; für groß-kleinschrift
c18b a0 00          ldy  #0          ; ********** loop ******
c18d b1 71    loop  lda  (text),y
c18f d0 03          bne  w1          ; wenn null dann basiczeile ende
c191 4c a4 c1       jmp  null
c194 20 d2 ff w1    jsr  bsout       ; zeichen schicken
c197 20 e1 ff       jsr  stop        ; stop gedrueckt ?
c19a f0 51          beq  ende        ; dann ende drucken
c19c e6 71          inc  text        ; zeiger auf text erhöhen
c19e d0 02          bne  w2
cla0 e6 72          inc  text+1
cla2 d0 e9    w2    bne  loop
cla4 a9 0d    null  lda  #cr
cla6 20 d2 ff       jsr  bsout       ; linefeed schicken
cla9 c6 fe          dec  zeilen      ; zeilencounter -1
clab d0 0c          bne  w5          ; 66 zeilen gedruckt ?
clad a0 06          ldy  #formfeed   ; dann seitenvorschub
claf 20 d2 ff we    jsr  bsout
clb2 88             dey
clb3 d0 fa          bne  we
clb5 a9 42          lda  #66         ; zeilencounter wieder auf 66
clb7 85 fe          sta  zeilen
clb9 a0 0a    w5    ldy  #lm         ; linken rand für nächste zeile
clbb a9 20          lda  #" "
clbd 20 d2 ff lmloop jsr bsout
clc0 88             dey
clc1 d0 fa          bne  lmloop
clc3 a9 11          lda  #17         ; cursor down
clc5 20 d2 ff       jsr  bsout
```

```
clc8 a0 03          ldy  #3
clca e6 71    16    inc  text        ; skip null und link-adress
clcc d0 02          bne  w3
clce e6 72          inc  text+1
cld0 88       w3    dey
cld1 d0 f7          bne  16
cld3 a5 71          lda  text
cld5 c5 2d          cmp  txtend      ; ende basic-pgm ?
cld7 d0 06          bne  w4
cld9 a5 72          lda  text+1
cldb c5 2e          cmp  txtend+1
cldd f0 0e          beq  ende
cldf a0 03    w4    ldy  #3
cle1 e6 71    17    inc  text        ; skip zeilennummer
cle3 d0 02          bne  w6          ; und anf-zeichen
cle5 e6 72          inc  text+1
cle7 88       w6    dey
cle8 d0 f7          bne  17
clea 4c 8d cl       jmp  loop        ; nächste zeile drucken
cled a9 0d    ende  lda  #cr
clef 20 d2 ff       jsr  bsout       ; lezten linefeed ausgeben
clf2 20 cc ff       jsr  clrch       ; ausgabe auf bildschirm
clf5 a5 b8          lda  lgfnr
clf7 20 c3 ff       jsr  close       ; file schließen
clfa 4c 74 a4 endl  jmp  basloop     ; zurück in basic
```

Da das Laden solcher Programme mit einem Basic-Lader recht praktisch ist und keine Assembler- oder Monitorprogramme erfordert, ist nachfolgend noch das entsprechende Programm abgedruckt.
Wenn Sie dies Programm in den Rechner eingegeben haben, dann sollten Sie es unbedingt vor dem Starten auf Diskette oder Kassette speichern, da es sich nach dem RUN selber löscht!

```
100 for i = 49152 to 49660
110 read x : poke i,x : s=s+x : next
120 data 169, 65,160,192,141,  8,  3,140,  9,  3,169,128
130 data 141,145,  2,169, 23,141, 24,208,169,  0,168,145
140 data  43,200,145, 43,165, 43, 24,105,  2,133, 45,165
150 data  44,105,  0,133, 46, 32,142,166, 32, 68,229,169
160 data 128,141,138,  2,169,100,133,251,169,  0,133,252
170 data 169, 10,133,253, 96, 32,115,  0,201, 33,240,  6
180 data  32,121,  0, 76,231,167, 32,115,  0,201,161,240
190 data  10,201,153,240,  3, 76,  8,175, 76, 65,193, 32
200 data 115,  0,240, 13, 32,235,183,134,253,165, 20,133
210 data 251,165, 21,133,252,169,128,141,145,  2,169, 23
220 data 141, 24,208,169,133,141,  2,  3,169,192,141,  3
230 data   3, 32,139,192, 76,134,164,173, 20,  3,201,  3
240 data 240, 12,120,169,  3,141, 20,  3,169,193,141, 21
250 data   3, 88,165,251,166,252,133, 99,134, 98,162,144
260 data  56, 32, 73,188, 32,221,189,162,  0,189,  1,  1
270 data 240,  9,157,  0,  2, 32,210,255,232,208,242,165
280 data 251, 24,101,253,133,251,144,  2,230,252,169, 32
290 data 157,  0,  2, 32,210,255,232,169, 34,157,  0,  2
300 data 232, 32,210,255,169,  0,133,212, 32,207,255,201
310 data  13,240,  3, 76,105,165,165,251,229,253,133,251
320 data 176,  2,198,252,169,131,160,164,141,  2,  3,140
330 data   3,  3,162,  0, 76,118,165,165,211,201, 64,240
340 data   7,169,  0,133,  2, 76, 49,234,165,  2,208, 42
350 data 169,  8,141, 24,212,169,  8,141,  3,212,169,  0
360 data 141,  0,212,169,120,141,  1,212,169,255,141,  5
370 data 212,169,249,141,  6,212,169, 65,141,  4,212,169
380 data  64,141,  4,212,133,  2, 76, 49,234,169,  4,133
390 data 186,169,126,133,184,169,  0,133,183,169,  0,133
400 data 185, 32,192,255,166,184, 32,201,255,165, 45, 56
410 data 233,  2,197, 43,208, 11,165, 46,233,  0,197, 44
420 data 208,  3, 76,250,193,165, 43, 24,105,  5,133,113
430 data 165, 44,133,114,169, 66,133,254,169, 17, 32,210
```

```
440 data 255,160, 10,169, 32, 32,210,255,136,208,250,160
450 data   0,177,113,208,  3, 76,164,193, 32,210,255, 32
460 data 225,255,240, 81,230,113,208,  2,230,114,208,233
470 data 169, 13, 32,210,255,198,254,208, 12,160,  6, 32
480 data 210,255,136,208,250,169, 66,133,254,169, 17, 32
490 data 210,255,160, 10,169, 32, 32,210,255,136,208,250
500 data 160,  3,230,113,208,  2,230,114,136,208,247,165
510 data 113,197, 45,208,  6,165,114,197, 46,240, 14,160
520 data   3,230,113,208,  2,230,114,136,208,247, 76,141
530 data 193,169, 13, 32,210,255, 32,204,255,165,184, 32
540 data 195,255, 76,116,164
550 if s <> 64452 then print "fehler in datas !!" : end
560 print "ok"
570 sys 12*4096
```

Doch kommen wir nun zu einer etwas detailierten Gebrauchsanleitung.
Das Programm ist in der vorliegenden Fassung direkt auf dem C64 lauffähig und kann mit allen Commodore-Druckern über den seriellen IEC-Bus betrieben werden.
Nach dem Starten des Programms mit dem SYS-Befehl werden verschiedene Speicherbereiche initialisiert und somit quasi ein NEW-Befehl ausgeführt. Damit ist auch das Lade-Programm gelöscht und die Texteingabe kann ohne weiteres beginnen. Um in den Eingabe-Modus zu gelangen, geben Sie einfach ohne Zeilennummer den Befehl !GET ein. Sofort erscheinen auf dem Bildschirm die Zeilennummer 100 und Anführungsstriche. Jetzt können Sie munter beginnen, Ihren Brief oder Ihr Manuskript einzugeben.
Der schon erwähnte Beep ertönt, wenn die Zeile länger als 64 Zeichen wird. Beachten Sie aber dabei, daß die ganze Zeile einschließlich Zeilennummer gezählt wird. Nach dem Beep sollten Sie bei nächster Gelegenheit die RETURN-Taste drücken, da sonst die Druckerausgabe etwas durcheinander

gerät. Da vor jede Zeile 10 Leerzeichen ausgegeben werden, passen in eine Druckzeile nur noch 70 Zeichen. Wird diese Anzahl überschritten, dann macht der Drucker selbsttätig einen Zeilenvorschub und druckt die überstehenden Zeichen ohne linken Rand auf das Papier.

Den Eingabemodus können Sie verlassen, indem Sie direkt nach Erscheinen der Zeilennummer die Taste RETURN drücken. Dadurch kommen Sie in den normalen Eingabe-Modus des Rechners. Jetzt können Sie alle Befehle eingeben, die der Computer sonst auch kann. Im besonderen werden es der List-Befehl sowie Load und Save sein. Aber auch andere Befehle können Sie durchaus eingeben.
Haben Sie den Eingabe-Modus verlassen, so können Sie also den schon erstellten Text listen und wie ein normales Basic-Programm editieren. Geben Sie nun wieder !GET ein, so wird mit der zuletzt verwendeten und nur mit RETURN 'belegten' Zeilennummer fortgefahren. Selbstverständlich können Sie Zeilen auch 'per Hand' eingeben. Dabei schreiben Sie die Zeilen genau so wie sie im Eingabe-Modus auch auf dem Bildschirm erscheinen.
Zusätzlich hat der !GET-Befehl noch die Möglichkeit, zwei Parameter anzugeben. Die Syntax lautet dann !GETzeilennummer,zeilenabstand. !GET 200,5 also startet bei der Zeile 200, die nächste Zeile ist dann 205. Diese Möglichkeit ist vor allem dann nützlich, wenn Sie an einem schon begonnenen Text weiterarbeiten wollen. Dann geben Sie eine Zeilennummer an, die hinter der letzten, im Text verwendeten liegt.
Der !PRINT-Befehl kennt keine weiteren Parameter. Sobald er eingegeben wird, eröffnet das Programm einen Kanal mit der logischen File-Nummer 126 und schickt den Text in Groß-Kleinschrift auf den Drucker. Beim Ausdrucken wird die Stop-Taste abgefragt, so daß Sie den Ausdruck jederzeit stoppen können.

Wollen Sie im Text von den verschiedenen Möglichkeiten des Druckers Gebrauch machen, so geht auch dies. Nehmen wir als Beispiel einmal den Text:

```
XY-Disketten in der Familienpackung
    2000 stueck nur 15.34
```

Jetzt soll die untere Zeile des Textes in Breitschrift ausgedruckt werden. Das Einschalten der Breitschrift geschieht bei den meißten Druckern mit dem Steuercode CHR$(14). Das Problem ist, dem Drucker diesen Steuercode mitzuteilen. Allerdings gibt es am C64 ja die CONTROL-Taste. Wird diese Taste zusammen mit einer Buchstabentaste gedrückt, dann erzeugt der C64 ein Zeichen mit dem Wert 1 bei CONTROL-A bis zum Wert 26 bei CONTROL-Z. Den Wert 14 erhalten wir bei Verwendung von CONTROL-N. Dieses Zeichen können wir aber nur im Quote-Modus als Zeichen in einen Text eingeben. Darum ist das Vorgehen etwas kompliziert, da der Quote-Modus ja in der Eingabe ausgeschaltet ist. Es geht aber doch.
Zuerst schreiben Sie den Text wie gewohnt. Dann verlassen Sie den Eingabe-Modus durch drücken von RETURN nach der nächsten Zeilennummer. Jetzt fahren Sie mit dem Cursor auf die 2 in dem eingegebenen Text. Durch Drücken von SHIFT-INST/DEL machen Sie Platz für ein weiteres Zeichen und drücken nun CONTROL-N. Auf dem Bildschirm erscheint an dieser Stelle ein reverses 'n', daß beim späteren Ausdruck die Umschaltung auf Breitschrift bewirkt. Genau so können Sie alle Codes von 1 bis 26 in den Text einfügen. Den Code 27 erhalten Sie durch Drücken von CONTROL-:. Es erscheint eine reverse eckige Klammer auf, dem Code 29 durch CONTROL-; wobei eine reverse eckige Klammer zu erscheint.
Auf jeden Fall vermeiden sollten Sie die Kombination CONTROL-Klammeraffe. Diese Tastenkombination erzeugt den

Wert 0, der aber als Markierung für das Ende der Basic-Zeile verwendet wird. Alles, was in der Zeile auf die Null folgt, wird dann beim Ausdruck unterschlagen.

Jeder, der seinen Computer öfter als Textsystem verwenden möchte, wird über kurz oder lang nicht um den Kauf eines richtigen Textverarbeitungsprogrmms herumkommen. Mit diesem Minitextprogramm sind solche Dinge wie Blöcke verschieben, das Verketten von Texten zu einem langen, insgesamt auszudruckenden Text oder das Schreiben von Rundschreiben nicht oder nur schwer möglich. Wer jedoch nur ab und zu einmal einen kurzen Brief oder eine Notiz zu tippen hat, dem kann dies Programm sicher eine Hilfe sein.

### *5.2.2 Geht's auch größer?*

Zum Ausdrucken von Briefen, Manuskripten oder Listings ist die Zeichengröße der Matrixdrucker ja ganz prima. Was aber, wenn die Zeichen einmal größer dargestellt werden sollen?
Nun, wenn Sie Besitzer eines 7-Nadel-Matrixdruckers sind (dazu zählen MPS 801, VC1515, VC1525, aber auch GP-700 oder EPSON mit entsprechenden Interfaces), dann können Sie die folgenden Programme zum Ausdruck großer Zeichen ohne weiteres verwenden.
Die Schwierigkeit beim Drucken eigener Zeichen ist immer die Erstellung des Zeichensatzes. Die Menge einzugebener Daten läßt leicht Frust aufkommen, und ein vielversprechendes Programmierobjekt wird oftmals leider nur zu schnell in der Schublade verschwinden, da es einfach nicht jedermanns Sache ist, mehrere KByte Daten einzutippen.
Warum aber sollten wir uns die Mühe machen, einen Zeichengenerator zu entwerfen, wenn wir im Computer schon einen zur Verfügung haben?

Sowohl beim VC-20 wie auch beim C 64 besteht ja die Möglichkeit, den eingebauten Zeichengenerator auszulesen. Damit haben wir schon mal eine Grundlage, auf der wir dann weitermachen können.

Halt! werden jetzt einige Besitzer des C 64 rufen. Der Zeichengenerator im 64 ist ja in Basic gar nicht auszulesen, ohne daß sich der Computer 'aufhängt'. Diese Ansicht ist nur bedingt richtig, es gibt einen einfachen Trick, das Auslesen doch zu ermöglichen.

Durch die teilweise mehrfache Belegung des Adressbereichs des C 64 ergibt sich die Situation, daß der Zeichengenerator und die IO-ICs, das sind der VIC, die CIAs und der SID, im selben Adressraum liegen. Ob der Prozessor auf den Zeichengenerator oder auf die Peripherie-ICs zugreift, wird mit dem Bit 2 des Prozessorports mit der Bezeichnung -CHAREN entschieden.

Ist dies Bit auf Low, dann ist der Zeichengenerator eingeschaltet. Dieses Bit kann von Basic aus mit einem POKE auf die Adresse 1 beeinflusst werden.

Ohne weitere Maßnahmen wird der C 64 aber nach einem POKE 1,51 (-CHAREN auf Low) nach der READY-Meldung keine weitere Aktivität mehr zeigen. Das liegt daran, daß wir durch den POKE auch die CIA1 dem Zugriff des Prozessors entzogen haben. Dies IC erzeugt aber im C 64 periodisch (60 mal in der Sekunde) einen Interrupt. Dieser Interrupt kann nur bearbeitet werden, wenn der Prozessor die Möglichkeit hat, auf die CIA zuzugreifen.

Soll also der Zeichengenerator ausgelesen werden, so muß zusätzlich dieser Interrupt abgeschaltet werden.

In Maschinensprache gibt es zum Abschalten des Interrupt einen Befehl, den SEI.

In Basic können wir diesen Befehl nicht anwenden, dafür können wir aber auf andere Weise das selbe erreichen. Wir beseitigen einfach die Quelle des Interrupts. Das ist

möglich durch Löschen des sogenannten ICR, Interrupt Control Register, in der CIA 1.
Eine detaillierte Beschreibung der Vorgänge würde hier den Rahmen sprengen, wichtig ist aber, daß durch POKE 56333,127 die Interrupts abgeschaltet und durch POKE 56333,129 wieder eingeschaltet werden.
Beim VC-20 ist die Situation natürlich viel einfacher. Hier liegt der Zeichengenerator immer im Zugriffsbereich des Prozessors, und kann ohne besondere Vorkehrungen ausgelesen werden.
Nachdem jetzt klar ist, wie wir an den Zeichengenerator kommen, wollen wir überlegen, wie die Daten genutzt werden können.
Dazu wollen wir uns zuerst einmal ansehen, wie ein Zeichen aus dem Zeichengeneraor aufgebaut ist.
Genau wie beim Matrix-Drucker werden die einzelnen Zeichen auf dem Bildschirm in Form von Punkten dargestellt. Dabei werden die Zeichen in einer 8*8-Matrix erzeugt. Anders als beim Drucker werden die Zeichen aber nicht aus untereinander gedruckten Nadelreihen zusammengesetzt. Bedingt durch das Funktionsprinzip von Fernseher und Monitor werden die Zeichen durch horizontale Punktreihen erzeugt.
Die acht Punkte einer solchen horizontalen Punktreihe werden Zeichengenerator in 8 Bits = einem Byte gespeichert. Ist ein Bit gesetzt, dann erscheint auf dem Bildschirm ein Punkt, ist das Zeichen dagegen gelöscht, so erscheint an dieser Stelle die Hintergrundfarbe. Ein ganzes Zeichen setzt sich aus 8 solcher Bytes zusammen, die im Zeichengenerator nacheinander abgespeichert sind.
Als Besonderheit ist noch zu vermerken, daß die Zeichen in der Reihenfolge des Bildschirmcodes abgespeichert sind, nicht nach ihren ASC-Werten. Die ersten 8 Bytes des Zeichengenerator enthalten also die Bitmuster für den Klammeraffen, die folgenden 8 Bytes bestimmen das Aussehen

des großen 'A'.
Wenn wir aus den Daten ein großes Zeichen auf dem Drucker erzeugen wollen, so bietet sich für diesen Zweck die Verwendung des Einzelnadelmodus an. Wir wollen jedes gesetzte Bit durch ein schwarzes Feld mit der Größe von sieben Punkten vertikal und sechs Punkten horizontal darstellen. Da in hochauflösender Grafik 480 Punkte in einer Zeile gedruckt werden können, ergibt sich durch diese Wahl der 'Punktgröße' eine Zeichenmatrix von 64 Punkten vertikal und 48 Punkten horizontal. Somit bekommen wir genau 10 Zeichen in eine Zeile.
Doch kommen wir jetzt zu dem Programm. Da das Erstellen des Zeichengenerators relativ viel Zeit in Anspruch nimmt, haben wir das Programm geteilt. Das erste Programm erstellt aus den ersten 64 Zeichen des Zeichengenerators die Zeichenfolgen für die Ansteuerung des Druckers. Die erzeugten Daten werden für die spätere Verwendung auf die Floppy in eine sequentielle Datei geschrieben.
Das zweite Programm liest dann die Daten aus der Datei in ein Array und fragt nach dem zu druckenden Text. Dieser Text wird dann untersucht und anschließend ausgedruckt.
Die Begrenzung der darstellbaren Zeichen ist natürlich auch auf den gesamten Zeichenvorrat erweiterbar, allerdings werden Sie wohl kaum die Grafikzeichen oder Cursorsteuerzeichen in dieser Form ausdrucken wollen. Auch wird die Datei auf der Diskette gleich vier mal so groß (in der jetzigen Form 67 Blocks) und der Einleseprozeß wird im gleichen Maß verlängert.
Leser ohne Diskettenlaufwerk können recht einfach die beiden Programme zu einem Programm verbinden, allerdings müssen Sie dann jedes mal die Zeit zum Erstellen der Daten für den Drucker abwarten.

Zuerst das Listing des Programms zum Erstellen der Druckdaten.

```
100 DIM A%( 63,7)
110 POKE56320+13,127     : REM ENTFÄLLT BEI VC-20
120 POKE 1,51            : REM ENTFÄLLT BEI VC-20
130 A=13*4096            : REM 8*4096 BEI VC-20
140 FORK=0TO 63
150 PRINTK,
160 FORI=0TO7
170 A%(K,I)=PEEK(A+I)
180 NEXTI
190 A=A+8
200 NEXTK
210 POKE1,55             : REM ENTFÄLLT BEI VC-20
220 POKE56320+13,129     : REM ENTFÄLLT BEI VC-20
230 OPEN15,8,15
240 OPEN1,8,4,"NORM CHR DATA,S,W"
250 GOSUB460
260 PRINTCHR$(147);
270 FORK=0TO 63
280 PRINTCHR$(K+((K<32)*-64));
290 FORJ=0TO7
300 A=A%(K,J)
310 FORI=7TO0STEP-1
320 IF(2↑I=(AAND2↑I))THENGOSUB420:GOTO340
330 GOSUB440
340 NEXTI
350 PRINT#1,X$
360 X$=""
370 NEXTJ
380 NEXTK
390 CLOSE1
400 CLOSE15
```

```
410 END
420 X$=X$+CHR$(8)+CHR$(26)+CHR$(5)+CHR$(255)
430 RETURN
440 X$=X$+CHR$(8)+CHR$(26)+CHR$(5)+CHR$(128)
450 RETURN
460 INPUT#15,EN,ET$,ET,ES
470 IFEN=0THENRETURN
480 PRINTEN;ET$;ET,ES
490 CLOSE15:CLOSE1
```

Und gleich hinterher das Programm zum Drucken der großen Zeichen.

```
100 DIM A$( 63,7)
110 PRINTCHR$(147)
120 OPEN15,8,15
130 OPEN4,8,4,"NORM CHR DATA,S,R"
140 GOSUB380
150 FORK=0TO 63
160 PRINTK,
170 FORI=0TO7
180 INPUT#4,A$(K,I)
190 NEXTI,K
200 CLOSE4
210 CLOSE15
220 PRINT
230 A$=""
240 INPUT"TEXT (MAX.10 ZEICHEN)";A$
250 IFA$<>""THEN290
260 PRINT#1,CHR$(15)
270 CLOSE1
280 END
290 PRINTCHR$(147);A$
300 OPEN1,4
```

```
310 FORI=0TO7
320 FORL=1TOLEN(A$)
330 K=ASC(MID$(A$,L,1))
340 K=K+((K>63)*64)
350 PRINT#1,A$(K,I);
360 NEXTL:PRINT#1:NEXTI
370 GOTO220
380 INPUT#15,EN,ET$,ES,ET
390 IFEN=0THENRETURN
400 PRINTEN;ET$;ES;ET
410 CLOSE15
420 CLOSE4
```

Nun ist aber eine Begrenzung auf die 10 Zeichen in einer Zeile nicht unbedingt optimal. Sollen längere Texte ausgedruckt werden, so gibt es mit dem vorgestellten Programm nur die Möglichkeit, die Texte nacheinander auszudrucken, die Textstreifen auszuschneiden und aneinander zu kleben.
Auch ist vielleicht die Zeichengröße nicht für alle Fälle ausreichend. Mit den beiden nun vorgestellten Programmen werden in beiden Fällen Abhilfe geschaffen.
Um längere Texte auszugeben, gibt es den Ausweg, den Text um 90 Grad zu drehen, das Papier also quer zu bedrucken. Jetzt ist die Textlänge nur noch durch die Länge des Papiers begrenzt (immerhin gute 600 Meter mit einem Karton zu 2000 Blatt). Durch die Drehung können die Zeichen auch größer dargestellt werden. Das Programm fragt bei der Erstellung der Druckdaten nach der gewünschten Größe. Diese kann in Stufen zwischen 1 und 5 gewählt werden. Während in der Größe 1 die Zeichen etwa das selbe Format wie in dem vorigen Programm aufweisen, erstreckt sich in der Größe 5 ein Zeichen über die gesamte Papierbreite. Das ergibt Zeichen, die auch aus einigen 10 Metern Entfernung noch gut lesbar

sind. Dadurch ergeben sich sicher Einsatzmöglichkeiten für z.B. Werbezwecke oder Sportveranstaltungen.
Wenn allerdings längere Texte ausgedruckt werden sollen, dann sollten Sie schon ein wenig Geduld haben. Die Druckgeschwindigkeit läßt sich nicht mehr in Zeichen pro Sekunde, sondern nur noch in Zeichen pro Minute ausdrücken. Da die Zeichen als schwarze Balken gedruckt werden, empfiehlt es sich, dem Drucker nach etwa 10 Zeichen (in der Größe 5) eine kleine Ruhepause zu gönnen, da sonst der Druckkopf sehr heiß werden könnte.

```
100 DIM A%( 63,7)
110 PRINTCHR$(147)
120 NA$="ZEICHENDATA"
130 INPUT "GROESSE (1-5)";V
140 IFV<1ORV>5THEN110
150 PRINTCHR$(147);
160 NA$=NA$+STR$(V)+",S,W"
170 POKE56320+13,127 : REM INTERRUPT VERBIETEN NUR C64
180 POKE 1,51        : REM CHAREN LOW          NUR C64
190 REM CHARGENERATOR NACH ARRAY A%(K,I)
200 A=13*4096        : REM A=8*4096 BEI VC-20
210 FORK=0TO 63:PRINTK,:FORI=0TO7
220 A%(K,I)=PEEK(A+I)
230 NEXT:A=A+8:NEXT
240 POKE1,55:POKE56320+13,129 : REM INTERRUPT ZULASSEN.
250 REM ZEILE 135 ENTFÄLLT FÜR VC-20
260 OPEN15,8,15
270 OPEN1,8,4,NA$
280 GOSUB440
290 PRINTCHR$(147);
300 FORK=0TO 63:PRINTCHR$(K+((K<32)*-64));
310 FORI=7TO0STEP-1
320 FORJ=7TO0STEP-1
```

```
330 A=A%(K,J)
340 IF(2↑I=(AAND2↑I))THENGOSUB400:GOTO360
350 GOSUB420
360 NEXTJ:PRINT#1,X$:X$=""
370 NEXTI,K
380 CLOSE1:CLOSE15
390 END
400 X$=X$+CHR$(8)+CHR$(26)+CHR$(12*V)+CHR$(255)
410 RETURN
420 X$=X$+CHR$(8)+CHR$(26)+CHR$(12*V)+CHR$(128)
430 RETURN
440 INPUT#1,EN,ET$,ET,ES
450 IFEN=0THENRETURN
460 PRINTEN;ET$;ET;ES
470 CLOSE1:CLOSE15
```

Und gleich hinterher wieder das Druckprogramm:

```
100 DIM A%( 63,7)
110 PRINTCHR$(147)
120 NA$="ZEICHENDATA"
130 INPUT "GROESSE (1-5)";V
140 IFV<1ORV>5THEN110
150 PRINTCHR$(147);
160 NA$=NA$+STR$(V)+",S,W"
170 POKE56320+13,127 : REM ENTFÄLLT BEI VC-20
180 POKE 1,51        : REM ENTFÄLLT BEI VC-20
190 REM CHARGENERATOR NACH ARRAY A%(K,I)
200 A=13*4096        : REM A=8*4096 BEI VC-20
210 FORK=0TO 63:PRINTK,:FORI=0TO7
220 A%(K,I)=PEEK(A+I)
230 NEXT:A=A+8:NEXT
240 POKE1,55         : REM ENTFÄLLT BEI VC-20
250 POKE56320+13,129 : REM ENTFÄLLT BEI VC-20
```

```
260 OPEN15,8,15
270 OPEN1,8,4,NA$
280 GOSUB450
290 PRINTCHR$(147);
300 FORK=0TO 63
310 PRINTCHR$(K+((K<32)*-64));
320 FORI=7TO0STEP-1
330 FORJ=7TO0STEP-1
340 A=A%(K,J)
350 IF(2↑I=(AAND2↑I))THENGOSUB410:GOTO370
360 GOSUB430
370 NEXTJ:PRINT#1,X$:X$=""
380 NEXTI,K
390 CLOSE1:CLOSE15
400 END
410 X$=X$+CHR$(8)+CHR$(26)+CHR$(12*V)+CHR$(255)
420 RETURN
430 X$=X$+CHR$(8)+CHR$(26)+CHR$(12*V)+CHR$(128)
440 RETURN
450 INPUT#1,EN,ET$,ES,ET
460 IFEN=0THENRETURN
470 PRINTEN;ET$;ES;ET
480 CLOSE1:CLOSE15
```

## *5.3 Hardcopy*

Noch vor wenigen Jahren waren Drucker für den Hobbybereich geradezu unerschwinglich. Selbst einfache Matrixdrucker ohne besondere Fähigkeiten kosteten astronomische Summen, und das zu Zeiten, in denen die Brieftaschen und Scheckhefte der begeisterten Computerbesitzer ohnehin durch ungleich höhere Hardwarekosten als heute strapaziert wurden. Da wurde dann zum Ausdrucken alles verwendet, was irgendwie geeignet schien.
Besonders ausgemusterte Postfernschreiber wurden trotz der damit verbundenen Schwierigkeiten gerne und fast zu jedem Preis (500 DM war preiswert) gekauft. Die Freude war riesig, wenn nach dem Bau spezieller Interfaces und der Programmierung geeigneter Treiberprogramme mit viel Getöse und Gerumpel die ersten Programmlistings ausgedruckt werden konnten. Man nahm sogar in Kauf, daß der zur Verfügung stehende Zeichenvorrat dieser monströsen Apparaturen natürlich überhaupt nicht den Anforderungen gerecht wurde.

Zu diesen Zeiten konnte man in manchen Fachzeitschriften Programmlistings ganz besonderer Art bewundern. Hier waren pfiffige Leute auf die Idee gekommen, Hardcopy spezieller Machart vom Bildschirminhalt anzufertigen, und mit mehreren, hintereinander abgedruckten Exemplaren dieser Hardcopy das ganze Listing auszugeben. Gemeint sind Bildschirmfotos!

Nun haben sich in der Zwischenzeit die Verhältnisse glücklicherweise dramatisch verbessert. Die bei den Fotos verwendete Idee der Hardcopy ist aber heute so aktuell wie vor einigen Jahren. Nur muß man nicht mehr gleich ein Fotolabor neben seinem Computer haben, um eine Grafik oder einen Text vom Bildschirm aufs Papier zu bekommen.
Um den Bildschirminhalt aufs Papier zu bannen, ist neben dem

Drucker nur noch ein kleines Programm erforderlich. Den Drucker haben Sie ja wohl, die Programme werden Sie gleich kennenlernen.

### *5.3.1 Texthardcopy*

Die auf dem Bildschirm erscheinenden Zeichen sind nur eine spezielle Wiedergabe eines bestimmten Speicherbereichs in Ihrem Computer. Da der Speicher (jedenfalls das RAM) ja frei zu lesen und zu beschreiben ist, kann nicht nur mit POKE ein Zeichen auf dem Bildschirm erzeugt werden, genau so lassen sich mit der umgekehrten Funktion PEEK die Zeichen aus dem Bildschirmspeicher lesen.
Steht z.B. in der linken oberen Ecke des Bildschirms der Buchstabe A, dann erhalten wir durch PEEK(adresse) den Wert 1. Für den Wert 'adresse' muß natürlich die Adresse des ersten Bildschirmspeicherplatzes angegeben werden.
Die erste Adresse des Bildschirmspeichers liegt beim C 64 normalerweise bei 1024 (hex $0400). Beim VC-20 liegt die Startadresse je nach Speichererweiterung bei 7680 ($1E00) oder bei 4096 ($1000). Ab dieser Adresse liegt dann die Information, die auf dem Bildschirm angezeigt wird.
Leider hat die Firma Commodore einen etwas ungewöhnlichen Weg beschritten, was die Codierung der angezeigten Zeichen angeht. Der im Bildschirm-RAM benötigte Wert für ein bestimmtes Zeichen ist nur bei wenigen Zeichen mit den Werten gleich, die beim CHR$-String angegeben werden müssen. Dadurch werden Umrechnungen erforderlich, die die im Video-RAM abgespeicherten Werte in die entsprechenden Ascii-Werte wandeln.

Eine möglichst universelle Hardcopy-Routine sollte sowohl im Programm aufgerufen werden, als auch auf Tastendruck einen

Bildschirmabdruck erzeugen können. Beide Möglichkeiten sind im nachfolgenden Programm gegeben. Zum einen kann innerhalb eines Programms ohne Zutun des Bedieners mit einem SYS-Befehl die HC-Routine aufgerufen werden. Aber auch durch gleichzeitiges Drücken der Tasten CONTROL und F1 kann eine HC erzeugt werden. Das abgedruckte Assemblerlisting ist für den C 64 zugeschnitten, die anschließend abgedrucketen Basic-Loader sind jeweils für C 64 und VC-20 angepaßt. VC-20-Benutzer, die das Assemblerlisting verwenden wollen, benutzen am besten zur Anpassung eine Vergleichstabelle der Betriebssystem-Routinen der beiden Computer.

```
033C                *=   $033C
033C        IRQ      =   $314
033C        TAST     =   $EA87
033C        CTRL     =   $028D
033C        INTER    =   $EA31
033C        IRQFLAG  =   $02
033C        TASTE    =   $C5
033C        FA       =   $BA
033C        LF       =   $B8
033C        TEMP     =   $71
033C        FNLEN    =   $B7
033C        ANZAHL   =   $C6
033C        SA       =   $B9
033C        OPEN     =   $FFC0
033C        CKOUT    =   $FFC9
033C        BSOUT    =   $FFD2
033C        STOP     =   $FFE1
033C        STORE    =   $67
033C        CLRCH    =   $FFCC
033C        CLOSE    =   $FFC3
033C 78             SEI          ; INTERRUPT VERBIETEN
033D A9 49          LDA  #<NEU   ; INTERRUPTVECTOR AUF
```

```
033F A0 03          LDY  #>NEU    ; HARDCOPY UMBIEGEN
0341 8D 14 03       STA  IRQ
0344 8C 15 03       STY  IRQ+1
0347 58             CLI
0348 60             RTS           ; UND ZURÜCK
            ; NEUE INTERRUPT-ROUTINE
0349 20 87 EA NEU   JSR  TAST     ; TASTATUR ABFRAGEN
034C AD 8D 02       LDA  CTRL     ; CONTROL-TASTE GEDRÜCKT?
034F C9 04          CMP  #4       ; DANN BIT 2 GESETZT
0351 D0 79          BNE  EXIT1    ; WENN NICHT, DANN KEINE HC
0353 A5 C5          LDA  TASTE
0355 C9 04          CMP  #4       ; F1 GEDRÜCKT?
0357 D0 73          BNE  EXIT1    ; WENN NICHT, DANN KEINE HC
0359 A9 00          LDA  #0
035B 85 C6          STA  ANZAHL
035D F0 04          BEQ  HCOPY    ; UNBEDINGTER SPRUNG
035F A9 FF  SYSEIN  LDA  #$FF     ; EINSPRUNG FÜR SYS
0361 85 02          STA  IRQFLAG  ; MERKEN, DAS NICHT ÜBER IRQ
0363 A9 04  HCOPY   LDA  #4       ; HARDCOPY
0365 85 BA          STA  FA       ; GERÄTEADRESSE
0367 A9 7E          LDA  #126
0369 85 B8          STA  LF       ; LOG.FILENUMMER
036B A9 00          LDA  #0
036D A0 04          LDY  #4       ; VIDEORAM BEI $0400
036F 85 71          STA  TEMP
0371 84 72          STY  TEMP+1
0373 85 B7          STA  FNLEN    ; KEIN FILENAME
0375 A9 00          LDA  #0
0377 85 B9          STA  SA       ; SEKUNDÄRADRESSE
0379 20 C0 FF       JSR  OPEN
037C A6 B8          LDX  LF
037E 20 C9 FF       JSR  CKOUT
0381 A2 19          LDX  #25      ; 25 BILDSCHIRMZEILEN
0383 A9 0D  LOOP    LDA  #13      ; LINEFEED
```

```
0385 20 D2 FF          JSR  BSOUT     ; AUSGEBEN
0388 20 E1 FF          JSR  STOP      ; STOPTASTE GEDRÜCKT?
038B F0 2E             BEQ  EXIT      ; DANN SCHLUSS
038D A0 00             LDY  #0
038F B1 71     LOOP2   LDA  (TEMP),Y ; ZEICHEN AUS VIDEORAM HOLEN
0391 85 67             STA  STORE     ; UND MERKEN
0393 29 3F             AND  #$3F
0395 06 67             ASL  STORE
0397 24 67             BIT  STORE     ; BILDSCHIRMCODE
0399 10 02             BPL  * + 4     ; IN ASCII WANDELN
039B 09 80             ORA  #$80
039D 70 02             BVS  * + 4
039F 09 40             ORA  #$40
03A1 20 D2 FF          JSR  BSOUT     ; UND AUSGEBEN
03A4 C8                INY
03A5 C0 28             CPY  #40       ; 40 ZEICHEN GEDRUCKT?
03A7 D0 E6             BNE  LOOP2
03A9 98                TYA
03AA 18                CLC
03AB 65 71             ADC  TEMP      ; ZEIGER AUF VIDEORAM UM
03AD 85 71             STA  TEMP      ; 40 ERHÖHEN
03AF 90 02             BCC  * + 4
03B1 E6 72             INC  TEMP+1
03B3 CA                DEX
03B4 D0 CD             BNE  LOOP      ; SCHON ALLE ZEILEN GEDRUCKT?
03B6 A9 0D             LDA  #13       ; NEUE ZEILE AUSGEBEN
03B8 20 D2 FF          JSR  BSOUT
03BB 20 CC FF EXIT     JSR  CLRCH     ; AUSGABE WIEDER AUF BILDSCHIRM
03BE A9 7E             LDA  #126
03C0 20 C3 FF          JSR  CLOSE     ; FILE SCHLIESSEN
03C3 A5 02             LDA  IRQFLAG   ; ÜBER CONTROL-F1 HC AUSGELÖST?
03C5 F0 05             BEQ  EXIT1     ; DANN NACH EXIT1
03C7 A9 00             LDA  #$00      ; FLAG LÖSCHEN
03C9 85 02             STA  IRQFLAG
```

```
03CB 60                RTS          ; RETURN FÜR AUFRUF ÜBER SYS
03CC 4C 31 EA EXIT1    JMP  INTER   ; ALTE INTERRUPTROUTINE AUFRUFEN
```

Wie man sehen kann, liegt das Programm im Cassettenpuffer ab der Adresse 828. Diese Adresse ist auch beim SYS-Befehl zur Aktivierung der Routine anzugeben. Als logische Filenummer wurde 126 verwendet. Wenn Sie in ihren Programmen ebenfalls diese Filenummer benutzen, so sollten Sie nach dem Gebrauch der HC dieses File wieder öffnen, da es natürlich zum Ende geschlossen wird. Um aber eventuellen Fehlern vorzubeugen, sollten Sie diese Filenummer für die Hardcopy reservieren und nicht in Ihren Programmen verwenden.
Um eine HC zu bekommen, genügt es, wie erwähnt die Tasten CONTROL und F1 gleichzeitig zu drücken. Wollen Sie dagegen im Programm die HC auslösen, so können Sie dies durch SYS 863 erreichen. In beiden Fällen kann der Ausdruck gestoppt werden, wenn Sie die STOP-Taste gedrückt halten. Damit ist natürlich auch ein eventuell laufendes Programm unterbrochen.

Doch jetzt die Basic-Loader für C 64 und VC-20. Zuerst das C 64-Programm.

```
100 FOR I = 828 TO 974
110 READ X : POKE I,X : S=S+X : NEXT
120 DATA 120,169, 73,160,  3,141, 20,  3,140, 21,  3, 88
130 DATA  96, 32,135,234,173,141,  2,201,  4,208,121,165
140 DATA 197,201,  4,208,115,169,  0,133,198,240,  4,169
150 DATA 255,133,  2,169,  4,133,186,169,126,133,184,169
160 DATA   0,160,  4,133,113,132,114,133,183,169,  0,133
170 DATA 185, 32,192,255,166,184, 32,201,255,162, 25,169
180 DATA  13, 32,210,255, 32,225,255,240, 46,160,  0,177
```

```
190 DATA 113,133,103, 41, 63,  6,103, 36,103, 16,  2,  9
200 DATA 128,112,  2,  9, 64, 32,210,255,200,192, 40,208
210 DATA 230,152, 24,101,113,133,113,144,  2,230,114,202
220 DATA 208,205,169, 13, 32,210,255, 32,204,255,169,126
230 DATA  32,195,255,165,  2,240,  5,169,  0,133,  2, 96
240 DATA  76, 49,234
250 IF S <> 17800 THEN PRINT "FEHLER IN DATAS !!" : END
260 PRINT "OK"
270 SYS 828
```

Im folgenden Listing für den VC-20 ist das Highbyte der Startadresse des Video-Rams unterstrichen. Eingesetzt ist der Wert für einen VC-20 im Grundausbau. Haben Sie eine Speichererweiterung von mindestens 8 K eingesteckt, so setzen Sie an Stelle des Wertes 30 den Wert 16.

```
100 FOR I = 828 TO 974
110 READ X : POKE I,X : S=S+X : NEXT
120 DATA 120,169, 73,160,  3,141, 20,  3,140, 21,  3, 88
130 DATA  96, 32, 30,235,173,141,  2,201,  4,208,121,165
140 DATA 197,201,  4,208,115,169,  0,133,198,240,  4,169
150 DATA 255,133,  2,169,  4,133,186,169,126,133,184,169
160 DATA   0,160, 30,133,113,132,114,133,183,169,  0,133
170 DATA 185, 32,192,255,166,184, 32,201,255,162, 23,169
180 DATA  13, 32,210,255, 32,225,255,240, 46,160,  0,177
190 DATA 113,133,103, 41, 63,  6,103, 36,103, 16,  2,  9
200 DATA 128,112,  2,  9, 64, 32,210,255,200,192, 22,208
210 DATA 230,152, 24,101,113,133,113,144,  2,230,114,202
220 DATA 208,205,169, 13, 32,210,255, 32,204,255,169,126
230 DATA  32,195,255,165,  2,240,  5,169,  0,133,  2, 96
240 DATA  76,191,234
250 IF S <> 17844 THEN PRINT "FEHLER IN DATAS !!" : END
260 PRINT "OK"
270 SYS 828
```

### *5.3.2 Grafik-Hardcopy*

Nachdem wir jetzt von Text-Bildschirmen einen Abdruck erzeugen können, wollen wir doch mal sehen, ob sich nicht auch Grafik in dieser Form ausgeben läßt.
Gute Grafik-Erweiterungen, wie z.B. 'SUPERGRAFIK 64' enthalten bereits die entsprechenden Routinen und Befehle, um vom Grafik-Bildschirm einen Ausdruck auch auf verschiedenen Druckern zu erzeugen.
Da aber die Grafik des C64 auch von Basic mit PEEK und POKE zu handhaben ist, fehlen oft geeignete Programme, um die Grafik dauerhaft aufs Papier zu bannen. Das aber ist ja besonders reizvoll, da man auf die in Basic geschriebenen Programme recht lange warten muß, bis das Bild fertig ist.

Wir wollen im folgenden zwei Grafik-Hardcopy-Routinen für den C64 vorstellen, die uns freundlicher Weise von Herrn Plenge, dem Autor des Programms 'SUPERGRAFIK 64' zur Verfügung gestellt wurden.
Die erste Routine ist für den Betrieb der 7-Nadel-Drucker gedacht. Die zweite Routine ist ein besonderer Leckerbissen, sie ermöglicht den Ausdruck des Grafik-Bidschirms auf einem 1526, und das, obwohl dieser Drucker ja eigentlich nicht grafik-fähig ist.
Beide Routinen wurden im Adressbereich so angepasst, daß ein Betrieb mit der Grafik-Hilfe aus dem Buch €4 INTERN möglich ist.
Zuerst wollen wir Ihnen wie gewohnt die Assemblerlistings vorstellen, im Anschluß daran folgen die Basic-Loader.
Kommen wir zunächst zur Grafik-Hardcopy für die 7-Nadel-Drucker.

```
C200                    *=   $C200
C200          A         =    $AC
C200          B         =    A+1
C200          USE       =    $FD
C200          XK        =    $14
C200          FLG       =    $97
C200          OFFX      =    $63
C200          GRMEM     =    $2000
C200 20 F1 B7           JSR  $B7F1
C203 86 67              STX  $67
C205 20 0F F3           JSR  $F30F   ;SUCHT LOG. FILENR.
C208 20 1F F3           JSR  $F31F   ;SETZT FILEPARAM.
C20B A6 67              LDX  $67
C20D 20 C9 FF           JSR  $FFC9   ;KANAL OEFFNEN
C210 A9 FF              LDA  #$FF
C212 85 61              STA  $61     ;MASKE
C214 A9 07              LDA  #7
C216 85 FD              STA  USE     ;PACKENGROESSE
C218 A9 1C              LDA  #28
C21A 85 97              STA  FLG     ;ZEILENZAEHLER
C21C A9 00              LDA  #0
C21E 8D 0D C3           STA  ZWIS    ;YK-MERKER
C221 A9 28    SEV1      LDA  #40
C223 8D 0F C3           STA  FLG2    ;PACKENZAEHLER
C226 A2 04              LDX  #4
C228 BD B1 C2 SEV11     LDA  SEVTAB,X  ;MITTENZENTRIERT
C22B 20 D2 FF           JSR  $FFD2
C22E CA                 DEX
C22F 10 F7              BPL  SEV11
C231 A9 00              LDA  #0
C233 85 63              STA  $63
C235 85 64              STA  $64     ;XK=0
C237 AD 0D C3 SEV2      LDA  ZWIS
C23A 85 65              STA  $65     ;YK
```

```
C23C A9 00           LDA  #0
C23E 85 FE           STA  USE+1
C240 A5 63    SEV3   LDA  $63     ;XK-L
C242 A6 64           LDX  $64     ;XK-H
C244 A4 65           LDY  $65     ;YK
C246 20 B6 C2        JSR  HPOSN   ;POSITION BERECHNEN
C249 A0 00           LDY  #0
C24B B1 AC           LDA  (A),Y
C24D A6 FE           LDX  USE+1
C24F 9D 11 C3        STA  BUFFER,X  ;IN BUFFER
C252 E6 65           INC  $65     ;YK
C254 E8              INX
C255 86 FE           STX  USE+1
C257 E4 FD           CPX  USE
C259 D0 E5           BNE  SEV3
C25B A9 00           LDA  #0
C25D A0 07           LDY  #7
C25F A6 FD    SEV4   LDX  USE
C261 1E 11 C3 SEV5   ASL  BUFFER,X
C264 2A              ROL  A
C265 CA              DEX
C266 10 F9           BPL  SEV5
C268 25 61           AND  $61     ;BEI LETZTER ZEILE=#$0F
C26A 09 80           ORA  #$80    ;HIGH-BYTE
C26C 20 D2 FF        JSR  $FFD2   ;SENDEN
C26F 88              DEY
C270 10 ED           BPL  SEV4
C272 A5 63           LDA  $63     ;XK-L
C274 18              CLC
C275 69 08           ADC  #8
C277 85 63           STA  $63     ;XK+8
C279 90 02           BCC  SEV6
C27B E6 64           INC  $64     ;XK-H
C27D CE 0F C3 SEV6   DEC  FLG2
```

```
C280 D0 B5          BNE  SEV2
C282 A9 0D          LDA  #$D
C284 20 D2 FF       JSR  $FFD2
C287 AD 0D C3       LDA  ZWIS
C28A 18             CLC
C28B 69 07          ADC  #7
C28D 8D 0D C3       STA  ZWIS     ;MERKER+7
C290 C6 97          DEC  FLG
C292 F0 03          BEQ  SEV81
C294 4C 21 C2 SEV8  JMP  SEV1
C297 A9 04    SEV81 LDA  #4
C299 C5 FD          CMP  USE
C29B F0 0C          BEQ  SEV7
C29D 85 FD          STA  USE      ;LETZTE ZEILE
C29F A9 01          LDA  #1
C2A1 85 97          STA  FLG
C2A3 A9 0F          LDA  #$F
C2A5 85 61          STA  $61      ;MASKE
C2A7 D0 EB          BNE  SEV8
C2A9 A9 0F    SEV7  LDA  #15      ;NORMAL MODUS
C2AB 20 D2 FF       JSR  $FFD2
C2AE 4C CC FF       JMP  $FFCC    ;SCHLIESST KANAL
C2B1 50 00 10 SEVTAB .BYT 80,0,16,27,8
C2B6                              ;
C2B6 85 14    HPOSN STA  XK
C2B8 86 15          STX  XK+1
C2BA 98             TYA
C2BB 4A             LSR  A
C2BC 4A             LSR  A
C2BD 4A             LSR  A
C2BE AA             TAX
C2BF BD EF C2       LDA  MULH,X
C2C2 85 AD          STA  B
C2C4 8A             TXA
```

```
C2C5 29 03           AND  #3
C2C7 AA              TAX
C2C8 BD 09 C3        LDA  MULL,X
C2CB 85 AC           STA  A
C2CD 98              TYA
C2CE 29 07           AND  #7
C2D0 18              CLC
C2D1 65 AC           ADC  A
C2D3 85 AC           STA  A
C2D5 A5 14           LDA  XK
C2D7 29 F8           AND  #$F8
C2D9 85 63           STA  OFFX
C2DB A9 20           LDA  #>GRMEM  ;LAUFENDE GRAPHIKSEITE
C2DD 05 AD           ORA  B
C2DF 85 AD           STA  B
C2E1 18              CLC
C2E2 A5 AC           LDA  A
C2E4 65 63           ADC  OFFX
C2E6 85 AC           STA  A
C2E8 A5 AD           LDA  B
C2EA 65 15           ADC  XK+1
C2EC 85 AD           STA  B
C2EE 60              RTS
C2EF 00 01 02 MULH   .BYT 0,1,2,3,5,6,7,8,10,11,12,13,15,16
C2FD 11 12 14        .BYT 17,18,20,21,22,23,25,26,27,28,30,31
C309 00 40 80 MULL   .BYT 0,$40,$80,$C0
C30D 00 00    ZWIS   .WOR 0000
C30F 00 00    FLG2   .WOR 0000
C311 00       BUFFER .BYT 00
```

Gleich anschließend das Listing für die Benutzer des Druckers VC 1526.

```
C200                    .OPT P1
C200                    *=   $C200
C200           ZF       =    $68
C200           Z3       =    $69      ; ZWISCHENSPEICHER FUER SPLITTING
C200           A        =    $AC
C200           B        =    A+1
C200           GRMEM    =    $2000
C200 20 F1 B7           JSR  $B7F1
C203 86 67              STX  $67
C205 20 0F F3           JSR  $F30F    ;SUCHT LOG. FILENR.
C208 20 1F F3           JSR  $F31F    ;SETZT FILEPARAM.
C20B A9 05     EIGHT    LDA  #5
C20D 20 2F C3           JSR  OPENCH   ; OPEN FUER SA=5
C210 A9 06              LDA  #6
C212 20 2F C3           JSR  OPENCH   ; OPEN FUER SA=6
C215 A2 16              LDX  #$16
C217 20 C9 FF           JSR  $FFC9
C21A A9 14              LDA  #20      ; ZEILENABSTAND FUER GRAFIK
C21C 20 D2 FF           JSR  $FFD2
C21F A9 0D              LDA  #13      ; ENDEZEICHEN
C221 20 D2 FF           JSR  $FFD2
C224 20 CC FF           JSR  $FFCC    ; KANAL LOESCHEN
C227 A0 19              LDY  #25      ; ANZAHL ZEILEN
C229 84 64              STY  $64      ; SETZEN
C22B A9 02     HA10     LDA  #2       ; DOPPELT HOCH
C22D 85 68              STA  ZF       ; WIEDERHOLZAEHLER
C22F A9 00     HA10A    LDA  #0       ; ZAEHLER FUER FUEHRENDE BLANKS
C231 85 65              STA  $65      ; SETZEN
C233 A9 28              LDA  #40      ; ANZAHL SPALTEN
C235 85 63              STA  $63      ; SETZEN
C237 A9 19              LDA  #25
C239 38                 SEC
C23A E5 64              SBC  $64      ;ZEILEN NOCH ZU DRUCKEN
```

```
C23C AA                 TAX
C23D BD 38 C3           LDA  MULH,X
C240 09 20              ORA  #>GRMEM
C242 85 AD              STA  B
C244 8A                 TXA
C245 29 03              AND  #3
C247 AA                 TAX
C248 BD 52 C3           LDA  MULL,X
C24B 85 AC              STA  A
C24D A2 00    HA11      LDX  #0       ; GRAFIKDATEN HOLEN
C24F A0 08              LDY  #8
C251 A1 AC    HA1L2     LDA  (A,X)
C253 99 56 C3           STA  BUFFER,Y
C256 E6 AC              INC  A
C258 D0 02              BNE  HA1L01
C25A E6 AD              INC  B
C25C 88       HA1L01    DEY
C25D D0 F2              BNE  HA1L2
C25F A0 08              LDY  #8
C261 A2 08    HA1L4     LDX  #8
C263 1E 56 C3 HA1L3     ASL  BUFFER,X
C266 2A                 ROL  A
C267 CA                 DEX
C268 D0 F9              BNE  HA1L3
C26A 20 0E C3           JSR  BSPLIT
C26D 99 5E C3           STA  BUFFER+8,Y
C270 88                 DEY
C271 D0 EE              BNE  HA1L4
C273 A0 08    HA1L02    LDY  #8       ; GRAFIK DRUCKEN
C275 20 A6 C2           JSR  PRDFCH
C278 A0 04              LDY  #4
C27A 20 A6 C2           JSR  PRDFCH
C27D C6 63              DEC  $63
C27F D0 CC              BNE  HA11
```

```
C281 C6 68          DEC  ZF
C283 D0 AA          BNE  HA10A
C285 C6 64   HA11L50 DEC  $64     ; ZEILENZAEHLER ERNIEDRIGEN
C287 D0 A2          BNE  HA10    ; NOCH NICHT FERTIG
C289 A2 16          LDX  #$16
C28B 20 C9 FF       JSR  $FFC9   ; SA=6
C28E A9 24          LDA  #36
C290 20 D2 FF       JSR  $FFD2   ; ZEILENABSTAND TEXT
C293 A9 0D          LDA  #13
C295 20 D2 FF       JSR  $FFD2   ; ENDEZEICHEN
C298 20 CC FF       JSR  $FFCC
C29B A9 16          LDA  #$16    ; SA=6
C29D 20 C3 FF       JSR  $FFC3
C2A0 A9 19          LDA  #25
C2A2 20 C3 FF       JSR  $FFC3   ; CLOSE FUER SONDERZEICHENKANAL
C2A5 60             RTS          ; FERTIG
C2A6 A9 00   PRDFCH LDA  #0
C2A8 85 66          STA  $66
C2AA A2 15          LDX  #$15
C2AC 20 C9 FF       JSR  $FFC9   ; KANAL FUER ZEICHENDEFINITION
C2AF 98             TYA
C2B0 48             PHA
C2B1 A2 04          LDX  #4
C2B3 B9 5E C3 HA62  LDA  BUFFER+8,Y
C2B6 20 D2 FF       JSR  $FFD2
C2B9 20 D2 FF       JSR  $FFD2
C2BC 05 66          ORA  $66
C2BE 85 66          STA  $66
C2C0 88             DEY
C2C1 CA             DEX
C2C2 D0 EF          BNE  HA62
C2C4 20 CC FF       JSR  $FFCC   ; KANAL SCHLIESSEN
C2C7 A5 66          LDA  $66
C2C9 F0 31          BEQ  HA61
```

```
C2CB A6 67          LDX  $67
C2CD 20 C9 FF       JSR  $FFC9    ; DATENKANAL
C2D0 A5 65          LDA  $65      ; ANZAHL BLANKS
C2D2 F0 09          BEQ  HA63     ; KEINE
C2D4 AA             TAX
C2D5 A9 20          LDA  #$20
C2D7 20 D2 FF HA64  JSR  $FFD2
C2DA CA             DEX
C2DB D0 FA          BNE  HA64
C2DD A9 FE    HA63  LDA  #254     ; SONDERZEICHEN
C2DF 20 D2 FF       JSR  $FFD2
C2E2 68       HA61A PLA
C2E3 C9 04          CMP  #4
C2E5 D0 0A          BNE  HA75A
C2E7 A5 63          LDA  $63
C2E9 C9 01          CMP  #1
C2EB D0 04          BNE  HA75A
C2ED A9 0D          LDA  #13
C2EF D0 02          BNE  HA75
C2F1 A9 8D    HA75A LDA  #141
C2F3 20 D2 FF HA75  JSR  $FFD2
C2F6 20 CC FF       JSR  $FFCC    ; KANAL SCHLIESSEN
C2F9 E6 65          INC  $65
C2FB 60             RTS
C2FC A5 63    HA61  LDA  $63
C2FE C9 01          CMP  #1
C300 D0 08          BNE  HA61B
C302 A6 67          LDX  $67
C304 20 C9 FF       JSR  $FFC9
C307 4C E2 C2       JMP  HA61A
C30A 68       HA61B PLA
C30B E6 65          INC  $65
C30D 60             RTS
C30E A6 68    BSPLIT LDX ZF       ; ZEILENFLAG
```

```
C310 E0 01            CPX  #1       ; DRUCK DER ZWEITEN ZEILE ?
C312 F0 04            BEQ  HA90     ; JA
C314 4A               LSR  A
C315 4A               LSR  A
C316 4A               LSR  A
C317 4A               LSR  A        ; OBERES HALBBYTE NACH UNTEN
C318 29 0F    HA90    AND  #$0F     ; TABELLENINDEX BILDEN
C31A AA               TAX
C31B BD 1F C3         LDA  HATAB1,X ; SPLITWERT HOLEN ä
C31E 60               RTS
C31F 00       HATAB1  .BYT %00000000
C320 03               .BYT %00000011
C321 0C               .BYT %00001100
C322 0F               .BYT %00001111
C323 30               .BYT %00110000
C324 33               .BYT %00110011
C325 3C               .BYT %00111100
C326 3F               .BYT %00111111
C327 C0               .BYT %11000000
C328 C3               .BYT %11000011
C329 CC               .BYT %11001100
C32A CF               .BYT %11001111
C32B F0               .BYT %11110000
C32C F3               .BYT %11110011
C32D FC               .BYT %11111100
C32E FF               .BYT %11111111
C32F 85 B9    OPENCH  STA  $B9
C331 09 10            ORA  #$10
C333 85 B8            STA  $B8
C335 4C C0 FF         JMP  $FFC0
C338 00 01 02 MULH    .BYT 0,1,2,3,5,6,7,8,10,11,12,13,15,16
C346 11 12 14         .BYT 17,18,20,21,22,23,25,26,27,28,30,31
C352 00 40 80 MULL    .BYT 0,$40,$80,$C0
C356 00       BUFFER  .BYT 00
```

Beide Programme gehen von einer Grafik-Seite ab $2000 aus. In diesem Bereich sind die meisten Grafiken angesiedelt. Auch das schon erwähnte Grafik-Hilfe-Programm aus dem Buch 64 INTERN legt die Grafik in diesem Bereich ab. Sollte sich eine auszudruckende Grafik in einem anderen Adressbereich befinden, so kann ohne weiteres durch Ändern von GRMEM in der Variablen-Deklaration eine Grafik aus einem anderen Adressbereich ausgedruckt werden.
Die Grafik im Speicher ist in der Art organisiert, daß das Byte an der Adresse $2000 (=8192 dez) für die acht horizontal nebeneinander liegenden Grafik-Punkte in der oberen linken Ecke zuständig ist. Wird in diesem Byte ein Bit gesetzt, so erscheint an der entsprechenden Stelle ein Punkt, ist das Bit dagegen Low, so wird die Hintergrungfarbe ausgegeben. Das nächste Byte, also $2001 ist für die darunter liegenden 8 Punkte zuständig.
Bis einschließlich des Bytes an der Adresse $2007 werden jeweils die zugehörigen Bytes untereinander auf dem Bildschirm als horizontale Punkterreihen dargestellt. Das Byte mit der Adresse $2008 jedoch erzeugt bei gesetzten Bits Punkte in der obersten Punktereihe neben denen der Adresse $2000.
Eine kleine Grafik zeigt diese Belegung noch einmal deutlich.

```
$2000 . . . . . $2008 . . . . . .
$2001 . . . . . $2009 . . . . . .
$2002 . . . . . $2010 . . . . . .
$2003 . . . . . .
$2004 . . . . . .
$2005 . . . . . .       u.s.w
$2006 . . . . . .
$2007 . . . . . .
```

Jeweils 8 Bytes a 8 Bits bilden somit eine 64-Bit-Block.
Da der Drucker aber nur vertikale Punkte-Reihen ausdrucken kann, muß ein HC-Programm die gleichwertigen Bits von auf diese Weise zusammenhängenden Bytes isolieren und zu einem Byte zusammensetzen.
Das Problem bei den 7-Nadeldruckern ist nun, daß für eine Punkte-Reihe nur 7 Bytes berücksichtigt werden können. Das achte Bit muß dann beim Druck der nächsten Nadelreihe berücksichtigt werden.
Eine weitere Schwierigkeit ist die Tatsache, daß die vorhandenen 25 Zeilen = 200 Punktereihen sich nicht ohne Rest durch 7 teilen lassen. Wird die Grafik ausgedruckt, dann bleiben am Schluß 4 Punktereihen übrig, die noch einmal gesondert behandelt werden müssen.

Diese Schwierigkeiten hat ein 8-Nadel-Drucker wie der 1526 natürlich nicht. Hier sind die 200 Punktereihen in 25 Druckzeilen gut auszugeben, wenn der Drucker Grafik überhaupt ausdrucken kann.....
Nun ist am 1526 eine Tatsache relativ unbekannt. Er verfügt über die Möglichkeit, ein Zeichen beliebig in einer 8*8-Matrix auszugeben. Allerdings wird im Handbuch nur ein sehr uninteressantes Beispiel gegeben, der Drucker produziert 10 mal das Commodore-Zeichen. Das aber ist keine Anwendung, die man unbedingt jeden Tag benötigt.
Erschwert wird das Arbeiten mit dem frei definierbaren Zeichen noch durch die Tatsache, daß nur ein solches Zeichen pro Zeile definiert werden darf.
Aber es gibt Hilfe!
Der 1526 kennt ein spezielles Zeichen, der eine Zeile beendet, aber keinen Zeilenvorschub bewirkt. Es ist dies das Zeichen CHR$(141). Es handelt sich hierbei um den Wert 13 + 128. CHR$(13) ist der Wert, der sonst eine Zeile mit Zeilenvorschub beendet, CHR$(141) allerdings ist ein

geshiftetes CHR$(13), d.h. mit gesetztem höchstem Bit übertragen.
Nach diesem Zeichen kann nun das definierbare Zeichen neu definiert und ausgedruckt werden, allerdings muß das entsprechende Programm einen Zähler mitführen, auf welcher Druckposition dies neue Zeichen gedruckt werden soll, denn der Druckkopf steht wieder auf der 1. Schreibstelle (auch wenn er mechanisch nicht dahin gefahren ist). Um das Zeichen an die richtige Stelle zu bekommen, muß eine entsprechende Anzahl an Leerstellen (Blanks) vorweggeschickt werden.
Diese Ausgabe von Grafik-Daten ist natürlich recht zeitintensiv, da die Daten nicht einfach hintereinander ausgegeben werden können. Auch die zum Zeilenende immer mehr werdenden Leerzeichen, die mit übertragen werden müssen, machen sich deutlich bemerkbar. Aber in Maschinensprache ist eine ganz annehmbare Geschwindigkeit zu realisieren. In Basic allerdings würde eine Hardcopy enorm viel Zeit verbrauchen.
Alle die beschriebenen Vorgänge werden in eleganter Weise vom Programm übernommen.
Zusätzlich macht das Programm für den 1526 noch etwas. Es vergrößert die Grafik um den Faktor 2.
Wenn man einmal ein wenig nachrechnet, so kommt man bei einer Zeichenmatrix von 8*8 Punkten auf maximal 640 Punkte (8 Punkte*80 Zeichen) pro Druckzeile. Der C64 stellt nur 40 Zeichen dar, hat aber ebenfalls die 8*8-Matrix. Damit hat der Drucker die doppelte Punktezahl in horizontaler Richtung zur Verfügung.
Würde nun eine exakte Copy erstellt, dann wäre das Blatt nur zur Hälfte genutzt. Durch die Verdopplung der Punkte in horizontaler wie vertikaler Richtung läßt sich aber das Papier gut nutzen.

Doch kommen wir nun zu den Basic-Loadern.
Zuerst wieder das Programm für die 7-Nadel-Drucker, daran

anschließend das Programm zum Betrieb des 1526.

```
100 FOR I = 49664 TO 49933
110 READ X : POKE I,X : S=S+X : NEXT
120 DATA  32,241,183,134,103, 32, 15,243, 32, 31,243,166
130 DATA 103, 32,201,255,169,255,133, 97,169,  7,133,253
140 DATA 169, 28,133,151,169,  0,141, 13,195,169, 40,141
150 DATA  15,195,162,  4,189,177,194, 32,210,255,202, 16
160 DATA 247,169,  0,133, 99,133,100,173, 13,195,133,101
170 DATA 169,  0,133,254,165, 99,166,100,164,101, 32,182
180 DATA 194,160,  0,177,172,166,254,157, 17,195,230,101
190 DATA 232,134,254,228,253,208,229,169,  0,160,  7,166
200 DATA 253, 30, 17,195, 42,202, 16,249, 37, 97,  9,128
210 DATA  32,210,255,136, 16,237,165, 99, 24,105,  8,133
220 DATA  99,144,  2,230,100,206, 15,195,208,181,169, 13
230 DATA  32,210,255,173, 13,195, 24,105,  7,141, 13,195
240 DATA 198,151,240,  3, 76, 33,194,169,  4,197,253,240
250 DATA  12,133,253,169,  1,133,151,169, 15,133, 97,208
260 DATA 235,169, 15, 32,210,255, 76,204,255, 80,  0, 16
270 DATA  27,  8,133, 20,134, 21,152, 74, 74, 74,170,189
280 DATA 239,194,133,173,138, 41,  3,170,189,  9,195,133
290 DATA 172,152, 41,  7, 24,101,172,133,172,165, 20, 41
300 DATA 248,133, 99,169, 32,  5,173,133,173, 24,165,172
310 DATA 101, 99,133,172,165,173,101, 21,133,173, 96,  0
320 DATA   1,  2,  3,  5,  6,  7,  8, 10, 11, 12, 13, 15
330 DATA  16, 17, 18, 20, 21, 22, 23, 25, 26, 27, 28, 30
340 DATA  31,  0, 64,128,192,203
350 IF S <> 31355 THEN PRINT "FEHLER IN DATAS !!" : END
360 PRINT "OK"
```

Eine ganze Menge Datas sind es ja, die da eingetippt werden müssen, aber mit der Prüfsumme werden Fehler bei der Eingabe sicher erkannt.
Wir werden ab und zu gefragt, ob die Prüfsumme von diesem

oder jenem Programm nicht stimme, jedesmal ergäben sich beim Programmlauf Fehler. Dazu ist zu sagen, daß die Prüfsumme nicht von uns mit Bleistift und Papier, sondern ganz standesgemäß mit dem C64 errechnet wird. Auch das Umsetzen der Maschinensprache-Programme geschieht mit einem kleinen Programm. Ändern Sie also nicht einfach die Prüfsumme, wenn ein Fehler in den Datas angezeigt wird. Mit Sicherheit liegt dann noch ein Fehler in Ihrem abgetippten Programm vor.

Doch jetzt das noch etwas längere Listing für den Drucker 1526.

```
100 FOR I = 49664 TO 50006
110 READ X : POKE I,X : S=S+X : NEXT
120 DATA  32,241,183,134,103, 32, 15,243, 32, 31,243,169
130 DATA   5, 32, 47,195,169,  6, 32, 47,195,162, 22, 32
140 DATA 201,255,169, 20, 32,210,255,169, 13, 32,210,255
150 DATA  32,204,255,160, 25,132,100,169,  2,133,104,169
160 DATA   0,133,101,169, 40,133, 99,169, 25, 56,229,100
170 DATA 170,189, 56,195,  9, 32,133,173,138, 41,  3,170
180 DATA 189, 82,195,133,172,162,  0,160,  8,161,172,153
190 DATA  86,195,230,172,208,  2,230,173,136,208,242,160
200 DATA   8,162,  8, 30, 86,195, 42,202,208,249, 32, 14
210 DATA 195,153, 94,195,136,208,238,160,  8, 32,166,194
220 DATA 160,  4, 32,166,194,198, 99,208,204,198,104,208
230 DATA 170,198,100,208,162,162, 22, 32,201,255,169, 36
240 DATA  32,210,255,169, 13, 32,210,255, 32,204,255,169
250 DATA  22, 32,195,255,169, 25, 32,195,255, 96,169,  0
260 DATA 133,102,162, 21, 32,201,255,152, 72,162,  4,185
270 DATA  94,195, 32,210,255, 32,210,255,  5,102,133,102
280 DATA 136,202,208,239, 32,204,255,165,102,240, 49,166
290 DATA 103, 32,201,255,165,101,240,  9,170,169, 32, 32
300 DATA 210,255,202,208,250,169,254, 32,210,255,104,201
310 DATA   4,208, 10,165, 99,201,  1,208,  4,169, 13,208
```

```
320 DATA   2,169,141, 32,210,255, 32,204,255,230,101, 96
330 DATA 165, 99,201,  1,208,  8,166,103, 32,201,255, 76
340 DATA 226,194,104,230,101, 96,166,104,224,  1,240,  4
350 DATA  74, 74, 74, 74, 41, 15,170,189, 31,195, 96,  0
360 DATA   3, 12, 15, 48, 51, 60, 63,192,195,204,207,240
370 DATA 243,252,255,133,185,  9, 16,133,184, 76,192,255
380 DATA   0,  1,  2,  3,  5,  6,  7,  8, 10, 11, 12, 13
390 DATA  15, 16, 17, 18, 20, 21, 22, 23, 25, 26, 27, 28
400 DATA  30, 31,  0, 64,128,192,  0
410 IF S <> 41908 THEN PRINT "FEHLER IN DATAS !!" : END
420 PRINT "OK"
```

Zum Abschluß sollen noch einige Hinweise zur Handhabung der Programme sowie zum Drucker 1526 folgen.
Nach dem Laden und Starten des entsprechend Ihres Druckers ausgewählten Programms können Sie jedes beliebige Programm starten. Wollen Sie dann nach Fertigstellung der Grafik auf dem Bildschirm eine HC, dann können Sie die Routine mit

```
SYS 12*4096+2*256,1f
```

aufrufen. Die als zusätzlicher Parameter geforderte logische Filenummer muß vorher im Programm in einer entsprechenden OPEN-Anweisung dem Drucker unter der Geräteadresse 4 zugeteilt worden sein.
Ein entsprechender Programmausschnitt könnte in etwa lauten:

```
10
20
30 irgendein Grafik-Programm
40 GOSUB 1000 : REM HARDCOPY
50 hier gehts weiter
:
:
```

```
:
999 END
1000 OPEN 1,4
1010 SYS 49664,1
1020 CLOSE1
1030 RETURN
```

Wenn Sie als Besitzer des 1526 das entsprechende Programm eingetippt haben, die Hardcopy-Funktion aber nicht ausgeführt wird, dann kann es daran liegen, daß Ihr Drucker mit dem Austausch-ROM ausgerüstet worden ist. Dies Austausch-Rom hat zusätzlich zu den im Handbuch beschriebenen Möglichkeiten eine neue Betriebsart. Wird bei diesem Gerät eine kleine Hardware-Änderung vorgenommen, dann verhält sich der Drucker wie ein 1525. Entsprechende Steuerzeichen und Sekundäradressen des 1526 werden dann ignoriert. Aber auch im 1525-Modus ist ein Ausdrucken von Grafiken nicht möglich.
Wie kann man nun feststellen, welches Betriebssystem im Drucker eingesetzt ist?
Da gibt es zwei Möglichkeiten. Zum einen kann der Drucker geöffnet und nach Entfernen eines Blechdeckels das Betriebssystem-Eprom (ein 28-poliges IC mit einem Aufkleber) lokalisiert werden. Trägt dies IC die Aufschrift '1526 /C', dann ist es mit Sicherheit die alte Betriebssystemversion, die ihre Macken hat, die Hardcopy mit dem vorgestellten Programm aber verarbeitet. Trägt dies IC aber die Aufschrift '325341-06 H', dann ist eine neue Version des Betriebssystems eingesetzt.
Ist diese neue Version eingesetzt, dann sollten Sie einmal nachsehen, ob das IC U4D, ein 6532 mit allen Beinchen im Sockel sitzt. Die Bezeichnung U4D finden Sie auf der Leiterplatte an der Schmalseite des ICs. Ist das Beinchen mit der Nummer 16 (von links oben gezählt) nicht im Sockel,

sondern mit einem Stückchen Draht mit der Leiterplatte verbunden, dann befindet sich der Drucker in der 1525-Betriebsart. In diesem Fall funktioniert das vorgestellte Programm nicht.
Aber auch ohne Öffnen des Druckers kann festgestellt werden, welche Version und welche Betriebsart im Drucker eingestellt ist.
Führen Sie zuerst einen Selbsttest durch. Dazu schalten Sie das Gerät mit festgehaltenem Papiervorschubtaster ein. In der ersten Zeile, die dann gedruckt wird, erscheint unter anderem die Bezeichnung 'REV.' und eine folgende Nummer. Erscheint hier eine 1.0, dann ist wieder das alte Betriebssystem eingesetzt. Erscheint dagegen 7.0, so handelt es sich um die neue Version. Ob in diesem Fall der Drucker im 1526- oder im 1525-Modus arbeitet, kann durch einen einfachen Trick festgestellt werden. Sie schicken dem Drucker einfach über die Sekundäradresse 7 einen zu druckenden Text. Reagiert der Drucker und gibt den Text aus, dann ist die 1525-Betriebsart durch die zuvor beschriebene Brücke am IC U4D eingestellt. Druckt der Drucker dagegen nicht, dann arbeitet der Drucker in der 1526-Betriebsart und unsere Hardcopy funktioniert.

Natürlich können noch verschiedene andere Betriebssystem-Versionen vorhanden sein, wir haben uns hier aber auf die beiden am häufigsten eingesetzten Versionen beschränkt.

Damit wären wir am Ende des Kapitels über die Hardcopys angekommen. Aber bleiben wir noch ein wenig beim Thema Grafik. Im nächsten Kapitel sollen alle Leser auf ihre Kosten kommen, die sich für Grafik interessieren.

## *5.4 Grafik mit und ohne Einzelnadelsteuerung*

Grafik gehört sicher zu den interessantesten, aber auch schwierigsten Gebieten der Programmierung. Bestes Beispiel ist die Programmierung der hochauflösenden Grafik des C-64. Hier lassen sich ganz großartige Dinge auf den Bildschirm des C-64 zaubern, vorausgesetzt man weiß wie.
Ganz ähnlich sieht es mit der Grafikprogrammierung der Drucker aus. Eine sauber ausgedruckte Grafik ist als Ergebnis von mehreren Programmierstunden immer ein besonderes Erlebnis.

Wir wollen in den nächsten Kapiteln versuchen, Sie Schritt für Schritt mit den Möglichkeiten der Grafikprogrammierung vertraut machen. Sollte Ihr Drucker über keine besonderen Grafik-Möglichkeiten verfügen, dann lesen Sie ruhig weiter, Sie werden Staunen, was mit einfachen Mitteln möglich ist.
Zuerst werden wir versuchen, mit dem normalen oder Textmodus einige nette Grafiken zu erzeugen. Dabei werden zunächst zweidimensionale Bilder entstehen, später dann aber auch dreidimensionale.
Danach wenden wir uns dann der hochauflösenden Grafik zu. Hier werden dann recht ansehnliche Grafiken entstehen.

Doch kommen wir zunächst zu den einfachen Möglichkeiten. Die folgenden Programme laufen auf Druckern ohne besondere Möglichkeiten, die Darstellung zu verändern. Diese Programme laufen auf allen Druckern, die mindestens 80 Zeichen pro Zeile darstellen können. Damit haben auch Benutzer von Typenraddruckern Gelegenheit, die Programme zu probieren und damit zu experimentieren.

Allerdings gehen wir in den Programmen auch auf einige bei Druckern häufiger zu findende Besonderheiten ein, um die

Ergebnisse zu verbessern.
Schauen Sie also im Einzelfall nach, ob die verwendeten Steuerzeichen in der aufgeführten oder ähnlichen Form auch auf Ihrem Drucker verfügbar sind. Häufig sind bei unterschiedlichen Druckern identische Betriebsarten durch unterschiedliche Steuerzeichen zu erreichen.

### *5.4.1 Grafik - auch ohne Einzelnadelansteuerung*

Das erste Programm, das hier vorgestellt wird, ist wohl die einfachste denkbare Form. Es erzeugt eine Sinuskurve aus Sternchen, die vertikal über das Papier läuft.

```
10 PI = 3.14159 : OPEN 4,4
20 FOR A =0 TO 2*PI STEP.25
30 B = SIN(A)
40 C = B*39+40
50 FOR I =0 TO C:PRINT#4,CHR$(32);:NEXT
60 PRINT#4,"*"
70 NEXT A
80 CLOSE4
```

Wenn Sie dies Beispiel ausprobieren, werden Sie wohl vom Ergebnis etwas enttäuscht. Die erzeugte Kurve ist nur mit etwas Phantasie als Sinuskurve zu erkennen. Wenn Sie aber einmal etwas mit den Parametern experimentieren, werden Sie sicher die eine oder andere Kurve produzieren können, die ein halbwegs gutes Aussehen hat. Besonders eine Verringerung der Step-Weite in der Zeile 20 verbessert das Aussehen, wobei die Kurve allerdings in die Länge gezogen wird.
Diesen Effekt kann man aber bei vielen Druckern vermindern,

wenn der Zeilenabstand verringert wird. Eine Änderung des Zeilenabstandes bei den 7-Nadel-Druckern erfolgt durch Ändern der Zeile 60:

```
60 PRINT#4,"*";CHR$(8):PRINT$4,CHR$(15);
```

Diese Drucker kennen nur zwei verschiedene Zeilenabstände, einmal den im Textmodus üblichen Abstand von 6 Zeilen pro Zoll, zum anderen den im Grafik-Modus benötigten Abstand von 9 Zeilen pro Zoll.
Wir nutzen in unserem Beispiel die Zeilenschaltung des Grafikmodus. Dazu schalten wir den Grafikmodus am Ende der Zeile ein, geben dem Drucker die Anweisung, einen Zeilenvorschub vorzunehmen und schalten dann sofort in den Textmodus zurück.

Der 1526 bietet die Möglichkeit, den Zeilenabstand in Stufen von .1176 mm zu verändern. Der Zeilenabstand wird durch einen Wert spezifiziert, der über die Sekundäradresse 6 dem Drucker mitgeteilt wird. Wollen Sie den Zeilenabstand verringern, so schicken Sie dem Drucker einen Wert kleiner 36. Auf diesen Wert ist der Drucker nach dem Einschalten eingestellt. Um z.B. die Druckzeilen direkt aneinander zu drucken, muß der Zeilenabstand mit

```
5 OPEN 6,4,6 : PRINT$6,CHR$(24): CLOSE6
```

geändert werden.
Leider verwenden die Druckerhersteller die unterschiedlichsten Steuerzeichen, um den Zeilenabstand einzustellen. Sollten Sie einen hier nicht erwähnten Drucker besitzen, so schauen Sie doch einmal im Handbuch nach. Irgendeine Möglichkeit wird es sicher geben.

Ein weiterer Schritt zur Verbesserung besteht, wenn der Drucker die Möglichkeit bietet, die Druckposition nicht nur in ganzen Zeichen, sondern in Punktepositionen zu bestimmen. Diese Möglichkeit ist naturgegeben nur an Matrixdruckern zu finden, allerdings bieten Typenraddrucker mit Proportionalschrift manchmal vergleichbare Möglichkeiten.
Die Commodoredrucker 1515, 1525 u.s.w, also die 7-Nadel-Typen bieten diese selten genutzte Möglichkeit ebenfalls. Da diese Drucker 80 Zeichen in einer 7*6-Matrix darstellen, ergibt sich eine Auflösung von 6*80 = 480 Positionen.
An dieser Stelle tritt allerdings ein kleines Problem auf. Da wir dem Drucker nur Werte von 0 bis 255 in der Form CHR$(x) übermitteln können, wären wir gar nicht in der Lage, alle Druckpositionen zu spezifizieren. Darum muß die Positionsangabe in zwei Bytes erfolgen.
Diese beiden Bytes werden im Format CHR$(HIGHBYTE); CHR$(LOWBYTE) dem Drucker mitgeteilt, nachdem dieser durch ein CHR$(27);CHR$(16) auf den Empfang der Druckpositionscodes vorbereitet wurde.
Um also auf der Punkteposition 200 ein '*' zu drucken, muß nach einer entsprechenden OPEN-Anweisung die Zeile

```
PRINT#1,CHR$(27);CHR$(16);CHR$(0);CHR$(200);"*"
```

an den Drucker geschickt werden.
Für eine Punkteposition von 300 würde die Zeile lauten:

```
PRINT#1,CHR$(27);CHR$(16);CHR$(1);CHR$(44);"*"
```

Bei der letzten Kombination wurde das dritte Zeichen zu eins, das vierte Zeichen errechnete sich aus 300 minus 256 = 44.
Auf diese Weise können nun alle 480 Druckpositionen

angewählt werden.
In dem folgenden Programm, einer Erweiterung des Funktionenplotters, haben wir von dieser Möglichkeit Gebrauch gemacht. Zusätzlich sind in diesem Programm noch eine Skala eingebaut, sowie eine Skalierung. Auf diese Weise lassen sich auch Funktionen darstellen, deren Werte außerhalb der Grenzen von -1 bis 1 liegen, ohne vorher zu wissen, in welchem Bereich die Ergebnisse wohl liegen werden.
Mit diesem Programm lassen sich schon recht ansehnliche Graphen ausdrucken.

```
100 REM ************************************
110 REM ***                              ***
120 REM ***     FUNKTIONSPLOTTER         ***
130 REM ***                              ***
140 REM ***     MIT SKALIERUNG           ***
150 REM ***                              ***
160 REM ***     FUER 1515,1525 ETC.      ***
170 REM ***                              ***
180 REM ************************************
190 PI=3.1415926
200 GOSUB450  : REM GRENZEN ERMITTELN
210 OPEN4,4
220 FORI=1TO80
230 PRINT#4,CHR$(249);
240 NEXT
250 PRINT#4,CHR$(8)
260 FORA=0TO2*PISTEP.1
270 GOSUB530
280 C=B*MY+240
290 HB=INT(C/256)
300 LB=C AND255
310 IFC>=235ANDC<=245THEN390
```

```
320 IFC<=240THEN360
330 PRINT#4,CHR$(15)CHR$(27)CHR$(16)CHR$(0)CHR$(240)CHR$(245);
340 PRINT#4,CHR$(27)CHR$(16)CHR$(HB)CHR$(LB)"*"CHR$(8)
350 GOTO400
360 PRINT#4,CHR$(15)CHR$(27)CHR$(16)CHR$(HB)CHR$(LB)"*";
370 PRINT#4,CHR$(27)CHR$(16)CHR$(0)CHR$(240)CHR$(245)CHR$(8)
380 GOTO400
390 PRINT#4,CHR$(15)CHR$(27)CHR$(16)CHR$(HB)CHR$(LB)"*"CHR$(8)
400 NEXTA
410 PRINT#1,CHR$(15)
420 CLOSE1
430 END
440 REM SCALIERUNG
450 FORA=0TO2*PISTEP.1
460 GOSUB530
470 IFB>MATHENMA=B
480 IFB<MITHENMI=B
490 NEXT
500 IFABS(MI)>MATHENMY=233/ABS(MI):RETURN
510 MY=233/MA:RETURN
520 REM FUNKTION IN ZEILE 530 EINSETZEN
530 B=SIN(A)
540 RETURN
```

Das Zeichen der Null-Linie ist etwas aufwendig. Je nach errechneter Punkteposition muß entschieden werden, ob der zu druckende Punkt rechts oder links der Null-Linie liegt, oder sich mit der Null-Linie überschneidet. Entsprechend werden das Sternchen und die Null-Linie in der benötigten Reihenfolge gedruckt. Überdecken sich jedoch Null-Linie und Sternchen, dann wird die Ausgabe der Null-Linie unterdrückt. In der Zeit vor dem ersten Drucken werden alle Werte der Funktion schon einmal ausgerechnet. Die Ergebnisse werden benötigt, um den Maßstab der Darstellung festzulegen. Diese

Zeit kann für umfangreichere Funktionen nicht unerheblich sein. Etwas Geduld ist aber bei allen rechenintensiven Basic-Programmen nötig.

Wer mit diesen Ausdrucken immer noch nicht zufrieden ist, findet im Kapitel über die hochauflösende Grafik einen Funktionenplotter, der die Möglichkeiten dieses Grafikbetriebes nutzt und sehr gute Ergebnisse bietet.

Zum Schluß ein letzter Tip für die Darstellung von Grafiken mit dieser niedrigen Auflösungsstufe.
Viele Drucker sind in der Lage, Zeichen im sogenannten 'Compressed Mode' auszudrucken. In dieser Betriebsart werden statt 80 meist 132 Zeichen pro Zeile ausgegeben. Es hat sich gezeigt, daß das Aussehen einer Grafik besser wird, je mehr Punkte auf einer Fläche dargestellt werden können. Damit gibt es noch einen Weg, zu einem besseren Ausdruck zu gelangen. In den vorigen Beispielen müssen Sie, wenn Sie diese Möglichkeit benutzen können, die entsprechenden Werte mit dem Faktor 132/80 = 1.65 multiplizieren. Dieser Tip gilt entsprechend auch für das folgende Programm, mit dem wir versuchen wollen, eine dreidimensionale Grafik darzustellen.

Wenn Sie das erste Programm dieses Kapitels probiert haben, werden Sie möglicherweise denken, daß die Darstellung dreidimensionaler Grafik mit Sternchen schlechthin unmöglich ist.
Lassen Sie sich überraschen. Wie wir meinen, läßt sich das Ergebnis durchaus sehen.

Doch zuerst einmal das Programm.

```
100 DIMA$(80)
110 OPEN4,4
120 FORX=1TO72
130 H=0:L=120
140 FORI=1TO80:A$(I)=" ":NEXT
150 FORY=1TO48
160 XT=(X-Y)*2
170 IFXT<1THEN290
180 IFXT>48THEN280
190 C=SQR((XT-24)*(XT-24)+(Y-24)*(Y-24))+1
200 W=XT+EXP(-.01*C*C)*45+3
210 IFW>HTHEN240
220 IFW<LTHEN260
230 GOTO280
240 H=W
250 IFW>LTHEN270
260 L=W
270 A$(W)="*"
280 NEXTY
290 FORI=1TO80:PRINT#4,A$(I);:NEXTI
300 NEXTX
```

So kurz dieses Programm auch ist, einfach ist es bei Leibe nicht. Wir wollen an dieser Stelle nicht das ganze Programm Zeile für Zeile auseinander nehmen, sondern Ihnen nur wieder einige Anregungen für mögliche Modifikationen geben.
Zunächst gilt natürlich das bei den bisherigen Programmen gesagte.
Eine Anpassung an eine größere Zeilenlänge (z.B. 132 Zeichen/Zeile) erfordert ein Multiplizieren aller Werte mit dem schon genannten Faktor 1.65. Besonders gilt dies für die DIM-Anweisung zu Beginn des Programms. Ausnahmen bilden nur

der Wert 2 in der Zeile 160 sowie der Wert 0.01 in der Zeile 200.
Auch die beschriebenen Maßnahmen zur Reduzierung des Zeilenabstands sollten Sie bei diesem Programm einmal ausprobieren.
Um eine andere Oberfläche darzustellen, kann die Funktion in der Zeile 200 geändert werden. Probieren Sie doch einmal:

```
200 W=XT-EXP(-.01*C*C)*45+25
```

Natürlich sind die auf diese Weise erzeugten Grafiken den in High-Res, also Einzelnadel-Steuerung nicht ebenbürtig. Dazu ist die Menge der darstellbaren Punkte auf der Fläche einfach zu gering. Allerdings reduziert sich natürlich durch diese geringere Anzahl der Punkte auch die Anzahl der benötigten Berechnungen. Das beschleunigt den Programmablauf enorm. Richtig 'große' in Einzelnadel-Grafik erzeugte Bilder können da zum vollständigen Berechnen aller Punkte einige Stunden benötigen.

### *5.4.2 Grafik - mit Einzelnadelansteuerung*

Im letzten Kapitel haben wir versucht, die Drucker ohne besondere Grafikfähigkeiten zu überlisten, uns doch einige Grafiken zu erzeugen. In diesem Kapitel nun werden die Drucker an die Reihe kommen, die über einen solchen Grafikmodus verfügen.
Durch die drastische Steigerung der Anzahl der ansteuerbaren Punkte verbessert sich das Erscheinungsbild ganz enorm. Sie haben sicher schon einmal solche Grafiken gesehen. Da gibt es den sogenannten Hut, ein klassisches Beispiel.
Wie aber werden solche Grafiken programmiert? Nun, nach dem Durcharbeiten dieses Kapitels werden Sie in der Lage sein,

auch solche komplexen dreidimensionalen Darstellungen zu erzeugen, vorausgesetzt natürlich, Ihr Drucker ist zu einem Einzelnadeldruck in der Lage.

Die Beispiele werden die Drucker 1515, 1525, GP 100 VC, MPS 801 sowie EPSON mit Interface berücksichtigen. Die Besitzer von anderen Grafik-Druckern wie z.B. ITOH sollten aber nicht sofort das Handtuch werfen, mit Hilfe des Druckerhandbuchs und den Erläuterungen im Text werden Anpassungen gar nicht so schwer.

Doch wenden wir uns zunächst einmal den Grundlagen der Einzelnadelansteuerung zu.
Leider ist die Einzelnadelgrafik in den Druckerhandbüchern fast immer nur sehr oberflächlich behandelt, wobei mehr Fragen aufgeworfen als geklärt werden.
Dabei ist es oft diese Eigenschaft des Druckers, die die Kaufentscheidung beeinflußt.

Um einen Drucker im Grafikmodus zu betreiben, ist es nötig, einige elementare Dinge über den Computer zu wissen.
Der VC-20 und der C64 sind sogenannte 8-Bit-Computer. Wie Sie sicher wissen, ist das Bit die kleinste darstellbare Informationseinheit. Ein Bit kennt nämlich zwei Zustände, gesetzt oder gelöscht. Diese Zustände werden auch mit Wahr und Unwahr oder Ja und Nein bezeichnet. Techniker benutzen gern die Bezeichnungen High und Low, entsprechend den Spannungspegeln, die an einer ein Bit entsprechenden Leitung zu messen sind. Zu guter Letzt gibt es noch die Bezeichnungen 1 für High und 0 für Low.
Diese letzte Darstellungsweise hat besonders dann ihren Vorteil, wenn es darum geht, mit den Bit-Werten zu rechen. Genau dies wollen wir jetzt tun, um die tieferen Hintergründe der Grafik-Programmierung zu durchschauen.

Die kleinste Informationseinheit ist also ein Bit mit zwei möglichen Zuständen. Sie mögen jetzt fragen, wieso zwei Zustände notig sind. Ohne jetzt zu sehr abzuschweifen kann man sagen, daß mit nur einem Zustand nur ein 'vielleicht' darstellbar ist und das ja keine konkrete Information im Gegensatz zu einem bestimmten Ja oder Nein darstellt.
Nehmen wir einmal zwei Bits. Damit können wir schon vier verschiedene Kombinationen darstellen. Es sind dies:

| Bit 1 | Bit 2 |
|---|---|
| NEIN | NEIN |
| NEIN | JA |
| JA | NEIN |
| JA | JA |

Bei nur einem Bit mehr ergeben sich bereits 8 Möglichkeiten, bei vier Bit sind es bereits 16, bei 5 Bits schon 32 mögliche Kombinationen. Wir wollen diese nicht alle durchspielen, versuchen Sie es einmal selbst.

Dies schnelle Steigen der Anahl der möglichen Kombinationen ist Ihnen vielleicht aus der Erzählung bekannt, in der ein Schachspieler von seinem König 'nur' Weizen wünscht, und zwar auf dem ersten Feld ein Korn, dann zwei Körner, auf dem nächsten Feld 4 Körner u.s.w. Auch hierbei ergibt sich mit jedem Feld (bei uns Bit) eine Verdopplung der Anzahl.

Wenn man sich die Zahlen einmal genauer betrachtet, dann stellt man fest, daß die Anzahl der Kombinationen sich sehr einfach errechnen läßt. Man muß nur die Zahl 2 mit sich selbst malnehmen, und zwar für jedes zur Verfügung stehende Bit einmal. Anders ausgedrückt, bei zwei Bits rechnet man 2 * 2, bei vier Bits 2 * 2 * 2 * 2.
Nun werden Sie sicher wissen, daß man einen solchen Ausdruck

auch anders schreiben kann. Die dafür gebräuchliche Darstellung lautet 2 hoch 2 oder 2 hoch 4. Wir verwenden, genau wie Ihr Computer, den Pfeil nach oben ↑ also Zeichen für die 'Hoch'-Funktion.

Da nun Ihr Computer mit 8-Bit-Werten arbeitet, können wir 2↑8 = 256 verschiedene Kombinationen aus gesetzten und gelöschten Bits erzeugen. Es sind dies die Werte 0 bis 255. Die Null wird also als Wert voll mitgezählt. Diese Null erhalten wir, wenn keins der 8 Bits gesetzt ist.
Genau wie in der kleinen Tabelle mit den zwei Bits müssen wir jetzt unsere 8 Bits nummerieren. Dabei wählen wir die gebräuchliche Art, die Bits mit den Ziffern von 0 bis 7 zu bezeichen. Diese Bezeichnung ist zwar ein klein wenig verwirrend, aber durchaus nicht willkürlich. Nehmen wir zum besseren Verständnis doch noch einmal unsere kleine Tabelle von vorhin und ändern die Bezeichnung der Bits entsprechend.

| Bit 1 | Bit 0 | |
|---|---|---|
| NEIN | NEIN | = 0 |
| NEIN | JA | = 1 |
| JA | NEIN | = 2 |
| JA | JA | = 3 |

Wichtig ist jetzt, daß jedes gesetzte Bit einen bestimmten Wert bedeutet. So hat ein gesetztes Bit 0 den Wert 1, ein gesetztes Bit 1 den Wert 2. Sind beide Bits gesetzt, dann addieren sich die beiden Werte.
Erweiterte man unsere kleine Tabelle um ein Bit, dann hätte dies Bit den Wert 4. Jetzt ist die selbe Regel wie vorhin erkennbar: der Wert eines Bits errechnet sich aus der Zahl 2 hoch seiner Bezeichnung. Ein gesetztes Bit 2 bedeutet also 2↑2 = 4. Genau so errechnet sich der Wert eines gesetzten siebten Bits aus 2↑7 = 128.

Die folgende Tabelle listet einmal alle möglichen Werte auf:

```
 Bit 7  6  5  4  3  2  1  0
     :  :  :  :  :  :  :  :
     :  :  :  :  :  :  :  ...  2↑0 =   1
     :  :  :  :  :  :  :  .....  2↑1 =   2
     :  :  :  :  :  :  .........  2↑2 =   4.
     :  :  :  :  :  ...........  2↑3 =   8
     :  :  :  :  .............  2↑4 =  16
     :  :  :  ...............  2↑5 =  32
     :  :  .................  2↑6 =  64
     :  ...................  2↑7 = 128
                                    ---
                                    255
```

So weit, so gut. Was hat das ganze aber mit der Einzelnadelansteuerung zu tun, mögen Sie vielleicht fragen. Die Antwort darauf ist denkbar einfach. Genau wie die dezimalen Werte der einzelnen Bits in unserer Tabelle untereinander stehen, genau so werden die Punktreihen ausgedruckt. Dabei werden den einzelnen Nadeln oder Punkte genau so bewertet, wie in unserem Beispiel die Bits.
Allerdings muß an dieser Stelle gleich eine kleine Einschränkung gemacht werden. Die gezeigte Reihenfolge von oben nach unten hat nur für die Drucker 1515 und aufwärts Gültigkeit. Bei Epson-Druckern und einigen anderen Druckern muß die Reihenfolge umgedreht werden, d.h. die unterste Nadel hat die Wertigkeit 1, die oberste Nadel die Wertigkeit 128.
Da jetzt die Bewertung der einzelnen Punkte klar ist, wollen wir einmal versuchen, mit der Grafik etwas auszugeben. Um uns für den Anfang die Arbeit etwas zu vereinfachen, gehen wir von einem hypothetischen Drucker aus. Dieser nur in unserer Vorstellung existierende Drucker hat Möglichkeiten, die kein auf dem Markt befindlicher Drucker jemals haben

wird. Er verfügt über alle Befehle und Möglichkeiten, die wir uns nur vorstellen können.
Versuchen wir doch einmal auf unserem gedachten Drucker die Darstellung eines Zeichens, das eine gewisse Ähnlichkeit mit einem kleinen Kreis aufweist.
Dabei wollen wir im Moment einmal davon ausgehen, daß die oberste Punktreihe die Wertigkeit 1 hat.
Unser Kreis soll vergrößert etwa dies Aussehen haben:

```
  ***        1
 *   *       2
*     *      4
*     *      8
*     *     16
 *   *      32
  ***       64
```

Um uns die Aufgabe ein wenig zu vereinfachen, stehen die Wertigkeiten der Punkte gleich in den entsprechendenen Zeilen. Wir müssen jetzt nur die Werte jeder Spalte addieren. Da das Zeichen so schön symmetrisch ist, brauchen wir nur die Werte für die ersten drei Spalten zu berechnen, sie wiederholen sich dann.
Für die erste Spalte bekommen wir das Ergebnis 4+8+16 = 28. In der zweiten Spalte erhalten wir 2+32 = 34 und in der dritten Spalte ist das Ergebnis 1+64 = 65. Die 65 wiederholt sich in den beiden folgenden Spalten, dann kommt wieder eine 34 und zum Schluß die 28.
Nachden wir diese Werte jetzt bestimmt haben, stellt sich die Frage, was damit geschehen soll. Da diese Werte als Grafik ausgedruckt werden sollen, müssen wir den Drucker in den Grafik-Modus schalten.
Bei unserem gedachten Drucker sei einmal der Wert 128 der Ein/Ausschalter der Grafik.

Zum Einschalten des Grafik-Modus benutzen wir die Anweisung:

```
PRINT#1,CHR$(128)
```

Jetzt können wir nacheinander die zuvor bestimmten Werte mit PRINT-Anweisungen an den Drucker schicken. Dazu werden die Werte für die Grafik-Bytes günstigerweise in DATA-Zeilen abgelegt.
Ein vollständiges Programm für unseren hypothetischen Drucker könnte also aussehen:

```
10 OPEN 4,4
20 PRINT#4,CHR$(128) : REM SOLL GRAFIK-MODUS EINSCHALTEN
30 FOR I = 1 TO 7    : REM ANZAHL ZU ÜBERTRAGENDER DATEN
40 READ A            : REM HOLT EINZELNES BYTE
50 PRINT#4,CHR$(A);  : REM UND SCHICKT ES ZUM DRUCKER
60 NEXT I
70 PRINT#4,CHR$(128) : REM SOLL GRAFIK-MODUS AUSSCHALTEN
80 CLOSE4
90 DATA 28,34,65,65,65,34,28
```

Das hier beschriebene Programm kann nur funktionieren, wenn unser gedachter Drucker nur 7 Nadeln im Grafik-Modus ansteuern kann. Würde eine achte Nadel hinzu kommen, wäre ein Ein- und Ausschalten der Grafik mit dem Code 128 nicht möglich, da mit diesem Wert ja die achte Nadel angesteuert würde.
Wäre unser Drucker ein 8-Nadel-Drucker, so müssen wir also einen anderen Weg beschreiten. Als Ausweg können wir aber das folgende Verfahren benutzen. Auch weiterhin wollen wir einmal den Wert 128 zum Einschalten des Grafik-Modus verwenden. Nach dem Einschalten des Grafikmodus gibt nun das erste an den Drucker geschickte Zeichen die Anzahl der folgenden Grafik-Bytes an. Ist diese Anzahl erreicht, dann

befindet sich der Drucker ohne weitere Maßnahmen im Text-Modus.
In unserem Beispielprogramm würde dadurch die Zeile 70 ganz entfallen. Die Zeile 20 würde lauten:

```
20 PRINT#4,CHR$(128);CHR$(7);
```

Wie wir sehen, gibt es also zwischen den Grafik-Modi der 7- und 8-Nadel-Drucker einen sehr gewichtigen Unterschied, der vor allem für Besitzer von 8-Nadeldruckern wie EPSON etc. interessant ist, wenn ein Grafik-Programm auf diesen Drucker angepaßt werden soll.

Aber wenden wir uns jetzt den Grafik-Modi konkreter Drucker zu.
Betrachten wir zunächst die 7-Nadel-Drucker. Bei diesen Druckern gibt es zur Grafik-Steuerung zwei Steuerzeichen. Mit dem Code 8 wird die Grafik eingeschaltet, der Code 15 schaltet die Grafik wieder aus.
Damit ergibt sich aber ein kleines Problem mit dem Code zum Abschalten der Grafik. Bei unserem vorigen Modell waren ja die Grafik-Codes alle Zeichen von 0 bis 127. Damit würde ja auch der Code 15 als Grafik-Byte interpretiert und als Punktmuster ausgedruckt. Nun haben sich die Entwickler der Drucker einen besonderen Trick einfallen lassen. Zu jedem Grafikbyte muß einfach 128 addiert werden. Damit liegen die Grafik-Bytes im Bereich von 128 bis 255. Aus diesem Grund kann mit dem Code 15 in den Text-Modus zurückgeschaltet werden.
Der Code 8 bewirkt aber noch eine Funktion des Druckers. Er vergrößert die Anzahl der zu druckenden Zeilen von 6 Zeilen pro Zoll im Text-Modus auf 9 Zeilen pro Zoll. Durch diese Maßnahme sind wir in der Lage, die Grafik lückenlos an einander zu drucken.

Wenn Sie einen EPSON-Drucker am VC-20 oder C64 mit oder ohne spezielles Interface betreiben, dann stehen Ihnen sehr leistungsstarke Möglichkeiten zur Ansteuerung der Einzelnadel-Grafik zur Verfügung. Da diese Drucker insgesamt sogar über 9 Nadeln verfügen, muß man hier nicht die Beschränkungen der 7-Nadel-Drucker berücksichtigen. Jedes Grafikbyte wird mit acht Bit übertragen und auch ausgedruckt. Eine Nadel wird hierbei ignoriert.
An dieser Stelle eine kleine Bemerkung: der FX-80 kennt einen sogenannten 9-Nadel-Grafik-Modus, bei dem immer zwei Grafikbytes übertragen werden müssen. Das 'Flagschiff' der EPSON-Drucker, der LQ 1500 ist sogar mit einem Druckkopf mit 24 Nadeln ausgerüstet. Diese 24 Nadeln sind in speziellen Grafik-Modi alle ansteuerbar. Damit sind Grafiken möglich, die fast das Aussehen von Fotos haben.

Die Wertigkeit der Nadeln ist beim Epson-Drucker genau umgekehrt zu den Commodore-Druckern, d.h. die Nadel mit dem Wert 1 ist die unterste, die mit dem Wert 128 die oberste Nadel.
Auch muß die Anzahl der zu übertragenden Grafikbytes bekannt sein, was aber normalerweise kein Problem darstellt, da die Grafikdaten ja errechnet werden, und die Anzahl der Ergebnisse im Programm klar ist.

Doch wenden wir uns nun dem ersten kleinen Programm zu, einem Funktionenplotter ähnlich dem Programm aus dem vorigen Kapitel. Diese Funktionenplotter haben einen Großen Vorteil, Sie sind schnell programmiert und die Ergebnisse lassen sich sehen. Wir stellen Ihnen das Programm in zwei Versionen vor, einmal für die Commodore-Drucker und eine zweite Version für EPSON-Drucker. Da beide ausführlich kommentiert werden, sollten sie sich leicht auch auf andere Grafik-Drucker wie z.B. Itoh oder Seikosha umschreiben lassen.

Der jetzt folgende Funktionenplotter besticht sicher nicht durch besonders trickreiche Programmierung oder eine gesonders gelungene Benutzerführung. Wir haben aber absichtlich das Programm so einfach wie möglich gestaltet, um auf die wesentlichen Punkte, nämlich die Ansteuerung der Grafik besseres Augenmerk legen zu können.

```
10 REM
20 REM   FUNKTIONSPLOTTER
30 REM
40 REM   IN EINZELNADELANSTEUERUNG
50 REM
60 REM   FUER COMMODORE DRUCKER MIT
70 REM
80 REM   7 NADEL-GRAFIK
90 PI = 3.14159
100 DEFFNA(X)=SIN(X)
110 INPUT"HORIZONTAL AUFLOESUNG";X%
120 IFX%<10RX%>479THEN110
130 INPUT"VERTIKALE  AUFLOESUNG";Y%
140 IFY%<1THEN130
150 INPUT"DEFINITIONSBEREICH   ";A,B
160 A=A*PI:B=B*PI:IFA>=BTHEN150
170 INPUT"WERTEBEREICH         ";C,D
180 IFC>=DTHEN170
190 DIMY%(X%)
200 VF=Y%/(D-C)
210 HF=(B-A)/X%
220 H=1
230 FORI=6TO0STEP-1
240   Y$(I)=CHR$(H+128)
250   H=H+H
260 NEXT
270 FORI=1TOX%
```

```
280    Y=FNA(A+HF*I)
290    Y%(I)=(Y-C)*VF+.5
300 NEXT
310 OPEN1,4
320 PRINT#1,CHR$(8);
330 A$=CHR$(128)
340 FORH=Y%-6TO-6STEP-7
350 REM
360    FORI=1TOX%
370       Y=Y%(I)-H
380       IFY<7ANDY>=0THENPRINT#1,Y$(Y);:NEXT:GOTO410
390       PRINT#1,A$;
400    NEXT
410    PRINT#1
420 NEXT
430 PRINT#1,CHR$(15):CLOSE1
```

Nachfolgend die Zeilen, die bei Verwendung eines EPSON-Druckers getauscht werden müssen. In unserem Beispiel sind wir von der Verwendung eines EPSON-Druckers mit DATA-BECKER VC-Einbau-Interface ausgegangen. Betreiben Sie den Drucker über den User-Port als Centronics-Drucker oder verwenden Sie ein anderes Interface, dann müssen Sie gegebenenfalls die OPEN-Anweisung entsprechend den Erfordernissen anpassen.

```
 10 REM
 20 REM FUNKTIONSPLOTTER FÜR MX/RX/FX
 30 REM
120 IFX%<1ORX%>960THEN110
230 FORI=0TO7
310 OPEN1,4,1
320 PRINT#1,CHR$(27)CHR$(65)CHR$(8);
330 A$=CHR$(0)
```

```
340 FORH=Y%-7TO-7STEP-8
350   PRINT#1,CHR$(27)CHR(76)CHR$(X%AND255)CHR$(X%/256);
380     IFY<8ANDY>=0THENPRINT#1,Y$(I);:NEXT:GOTO410
430 PRINT$1:CLOSE1
```

In der 100 wird die Funktion definiert, die grafisch ausgegeben wird. Diese Funktion können Sie nach eigenen Wünschen definieren, probieren Sie einmal verschiedene aus. Danach werden die wichtigsten Parameter abgefragt. Die Horizontalauflösung bestimmt die Breite der Grafik auf dem Papier. Zum schnellen ausprobieren können Sie hier kleine Werte (etwa 100) eingeben. Damit erhalten Sie eine schmale Grafik, die aber recht flott ausgegeben wird. Das Maximum beträgt natürlich 479, die Anzahl der möglichen Grafik-Punkte.
Die vertikale Auflösung ist nicht begrenzt, Sie sollten aber keine Werte eingeben, die wesentlich größer als die Horizontal-Auflösung sind, sonst wird die Grafik in vertikaler Richtung verzerrt.
Der Definitionsbereich bestimmt die Werte, für die die Funktion berechnet wird, während der Wertebereich angibt, welcher Bereich in vertikaler Richtung angezeigt wird. Sind bei einem Ausdruck die Spitzen z.B. einer Sinuskurve nicht mit ausgedruckt, so wählen Sie den Wertebereich einfach etwas größer, dann sollten alle Punkte dargestellt werden.

Nach der Eingabe dieser Werte werden die Variablen HF und VF errechnet. Diese Variablen werden zur Bestimmung der zu plottenden Punkte benötigt.
In den Zeilen 230 bis 260 werden jetzt die möglichen Bitmuster errechnet und in dem Array Y$(I) gespeichert. Dabei wird für die 7-Nadel-Drucker gleich das höchste Bit mitgesetzt (+128), da ja Grafikdaten bei diesen Druckern im Bereich 128 bis 255 liegen. Bei den EPSON-Druckern beginnt

hier der erste große Unterschied. Zum einen sind die Nadeln von unten nach oben nummeriert und zudem haben wir ja 8 Nadeln statt 7. Aus diesem Grund wird die Schleife in den Zeilen 230 bis 260 in der Commodore-Version nur 6 mal, in der EPSON-Version dagegen 7 mal durchlaufen. Da auch die Reihenfolge und Anzahl der Nadeln bei den beiden Druckergruppen unterschiedlich ist, wird einmal die Schleife von 6 bis 0, zum anderen von 0 bis 7 durchlaufen.
Soll dies Programm z.B. an einen Itoh angepaßt werden, so ist die Nadelreihenfolge wie bei den Commodore-Druckern. Die oberste Nadel hat bei diesen Druckern die Wertigkeit 1, die untere Nadel die Wertigkeit 128. Die Schleife muß also ebenfalls 'rückwärts', also von 7 nach 0 durchlaufen werden.
Jetzt wird die Funktion für alle Werte durchgerechnet und im Array Y% abgespeichert. Dies geschieht in den Zeilen 270 bis 300.
Danach beginnt der Ausdruck. Zuerst wird der Druckerkanal eröffnet. Die Angaben im OPEN-Befehl müssen natürlich entsprechend den Gegebenheiten angepaßt werden.
Der Ausdruck geschieht in zwei Schleifen. Die erste Schleife ist die Zeilenschleife. Sie läuft vom Wert der vertikalen Auflösung in Y% -6 in Steps von 7 entsprechend den 7 Nadeln nach -6. Für einen 8-Nadeldrucker ändern sich die Werte entsprechend in der Zeile 340.
Die zweite Schleife läuft von 1 bis zum angegebenen Wert der horizontalen Auflösung. Die Laufvariable I dieser zweiten Schleife wird nun benutzt, um den zuvor berechneten Wert der Funktion für den entsprechenden Punkt auf der X-Achse zu bekommen. Von diesem Funktionswert wird der Wert der Zeilenschleife abgezogen. Liegt das Ergebnis im Bereich von 0 bis 6 (0-7 bei EPSON), dann wird dieser Wert benutzt, um das Zeichen zu drucken, das den Punkt an die richtige Stelle setzt. Diese Punktemuster sind ja in der richtigen Reihenfolge zuvor im Array Y$ definiert worden. In allen

anderen Fällen wird nur ein Wert von 128 an den Drucker geschickt, der eine leere Grafik-Punktereihe bedeutet.
Für den EPSON-Drucker wird an dieser Stelle das Zeichen CHR$(0) geschickt.

So, nachdem Sie die ersten Versuche in Einzelnadelgrafik hoffentlich gut verdaut haben, wollen wir doch einmal sehen, ob nicht noch umfangreichere Grafiken möglich sind. Besonders reizvoll ist die Darstellung von dreidimensionalen Grafiken, deren Programmierung zwar schwieriger ist, die aber durch die große Anzahl an zur Verfügung stehenden Punkten ein sehr gutes Aussehen bekommen.
Mit Hilfe der Angaben aus diesem Kapitel sollte es Ihnen auch gelingen, die folgenden Programme für andere Drucker anzupassen, da im weiteren nur noch gezielt auf die Commodore-Drucker eingegangen wird.

### *5.4.3 Darstellung dreidimensionaler HIRES-Grafik*

Das Darstellen von dreidimensionalen Flächen und Körpern ist sicher eine echte Herausforderung an jeden Programmierer. Hat man dazu noch die Möglichkeit, die Ausgabe auf den Drucker zu bekommen, dann sollte man sich eigentlich sofort an die Arbeit begeben und entsprechende Grafiken programmieren. Allerdings sind für ein erfolgreiches Programmieren doch einige mathematische Grundfertigkeiten nötig.
Das Problem bei der Darstellung 3-dimensionaler Gegenstände ist ja, daß nur zwei Dimensionen auf dem Papier zur Verfügung stehen. Die dritte Dimension, die Tiefe, kann nur durch verschiedene Schwarz- und Grautöne dargestellt werden. Die dafür erforderliche Auflösung ist überzeugend nur im Hires-Modus, also mit Einzelnadelsteuerung möglich.

Wir haben zwei Programme entwickelt, die ganz nette Bilder

auf dem Drucker ausgeben. Da ist zuerst ein Programm, mit dem die Abbildung einer Kugel erzeugt wird. Das Ergebnis ist erstaunlich. Betrachtet man das Bild aus einiger Entfernung, so ist der Eindruck der 3-Dimensionalität verblüffend.
Das zweite Programm ist vergleichbar einem dreidimensionalen Funktionsplotter. Mit diesem Programm sind Flächen darstellbar, die durch Wellen verworfen sind und einen sehr plastischen Eindruck hervorrufen. Mit diesem Programm sind Bilder möglich, die dem berühmten Hut sehr ähnlich sind, aber auch ganz andere Oberflächen lassen sich darstellen.

Aber kommen wir zur Kugel. Mit dem nachfolgenden Programm können Kugeln, durch Ändern von zwei Programmzeilen aber auch andere Gegenstände, ausgedruckt werden.

```
100 REM ****************************
110 REM **                        **
120 REM **         KUGEL          **
130 REM **                        **
140 REM ****************************
150 PRINTCHR$(147)
160 PRINT "KUGEL"
170 INPUT "DURCHMESSER IN PUNKTEN (480 MAX.)";D
180 HD=D/2:FF=D/10
190 INPUT "DRUCKER ODER FLOPPY    ( D / F ) ";D$
200 IFD$="D"THENOPEN1,4:PRINT#1:PRINT#1,CHR$(8);
210 IFD$="F"THENOPEN1,8,4,"0:KUGELDATEN,S,W"
220 SK=1/10000:R=1
230 FORI=0TO6
240 MK(I)=R
250 R=R+R
260 NEXT
270 R=0
280 DIMWE%(D)
```

```
290 FORZ=-1TO1STEP1/HD
300 ZZ=ZZ+1
310 PRINTZZ;
320 R=R+1
330 MY=SQR(1-Z*Z)-SK
340 Y=-MY
350 X=SQR(ABS(1-Y*Y-Z*Z))+SK
360 A=SQR(ABS((1+Y*Y/X/X)*(1+Z*Z/X/X)))
370 S=1/A/FF
380 IFS<0THENS=0
390 Y=Y+S
400 IFY>MYTHENWE%(HD+MY*HD)=WE%(HD+MY*HD)ORMK(R-1):GOTO430
410 WE%(HD+Y*HD)=WE%(HD+Y*HD)ORMK(R-1)
420 GOTO350
430 IFR>6THENGOSUB490
440 NEXTZ
450 IFR>0THENGOSUB490
460 PRINT#1,CHR$(15)
470 CLOSE1
480 END
490 FORQ=1TOD
500 PRINT#1,CHR$(WE%(Q)+128);
510 WE%(Q)=0
520 NEXT:PRINT#1:R=0
530 RETURN
```

Nach dem Starten des Programms werden Sie gefragt, welchen Durchmesser die Kugel haben soll. Das Maximum liegt bei 480, der Anzahl der Punkte pro Zeile. Je kleiner der Durchmesser gewählt wird, desdo schneller ist das Bild fertig.
Apropos 'schneller fertig'. Wollen Sie eine Kugel mit maximalem Durchmesser ausdrucken, dann sollten Sie sicher sein, daß Sie den Rechner in den nächsten Stunden nicht benötigen. Am besten sollten Sie den Rechner abends

einschalten, dann haben Sie am nächsten Morgen ein schönes Ergebnis.
Da es nun nicht jedermanns Sache ist, nachts durch lautes Druckgeräusch gestört zu werden, besteht die Möglichkeit, die errechneten Daten auf einer Diskette zu speichern. Die auf der Diskette gespeicherten Daten können dann in verhältnismäßig kurzer Zeit auf dem Drucker ausgegeben werden. Auch können so mehrere identische Kugeln gedruckt werden, ohne jedes mal die lange Rechenzeit in Kauf zu nehmen.
Zum Ausdrucken der Daten ist das folgende kleine Programm gedacht.

```
100 OPEN4,4
110 PRINT#4,CHR$(8)
120 OPEN1,8,4,"KUGELDATEN,S,R"
130 GET#1,A$:IFST=64THEN150
140 PRINT#4,A$;:GOTO130
150 PRINT#4,CHR$(15)
160 CLOSE4
170 CLOSE1
```

Doch betrachten wir noch kurz das eigentliche Programm. Nach der Eingabe des Durchmessers und des Ausgabegerätes werden aus dem Durchmesser einige benötigte Werte errechnet. Zuerst ist dies der Radius, der für die Bestimmung des Punktes zum Mittelpunkt benötigt wird, danach die Variable FF (für Füllfaktor), die für die Anzahl der Punkte in der Grafik benötigt wird. Diese letzte Variable sollten Sie ruhig einmal etwas verändern und sehen, wie sich eine Änderung bemerkbar macht.
Es folgt das Öffnen des Drucker- oder Floppy-Kanals und die Initialisierung der Variablen SK (für sehr klein) und R. SK ist ein fast verschwindend kleiner Wert, der verhindert, daß

ein DIVISION BY ZERO-ERROR in der Zeile 360 auftreten kann. Die Variable R steht für Nadelreihe und wird zum Beginn benötigt, um die Bitmuster in der schon bekannten Weise in das Array MK zu speichern.
Die eigentliche Berechnung wird in der FOR-NEXT-Schleife in den Zeilen 290 bis 440 vorgenommen. Dabei sind die Zeilen 350 und 360 für die Form des Bildes verantwortlich. Auch hier wird die Variable R mitgeführt und bestimmt wieder die momentane Nadelreihe. Sind alle Punkte für die 7 Nadeln berechnet, so werden sie auf das Ausgabegerät geschickt. Danach werden die nächsten Punkte berechnet.

Wichtig zur Anpassung an andere Drucker ist nur eine Variable. Es ist die Variable HD. Sie besimmt in der FOR-NEXT-Schleife die Anzahl der Punktereihen.
Da nun nicht alle Drucker aus einen Feld von 100 mal 100 Punkten ein exaktes Quadrat machen, wird bei diesen Geräten eine Kugel zu einem Ei verformt. Je nach Verhältnis von vertikalen zu horizontalen Punkten kann das Ei auf der Spitze oder auf der Seite liegt. Ist das bei Ihrem Drucker der Fall, dann sollten Sie HD entsprechend größer oder kleiner machen. Je kleiner HD wird, desdo 'platter' wird die Kugel, wird HD dagegen größer, dann steht das Ei auf der Spitze.
Alle anderen Anpassungen beziehen sich nur in der bekannten Weise auf die Steuerzeichen zum Einschalten der Grafik. Müssen bei einem 8-Nadel-Drucker Angaben über die Anzahl der Grafik-Daten errechnet werden, so haben Sie die benötigte Information in der Variable D, dem Durchmesser. In dieser Variable ist ja die Anzahl der Punkte in horizontaler Richtung von Ihnen eingegeben worden.

Da aber immer nur Kugeln irgendwann auch ihren Reiz verlieren, soll jetzt das Programm zum Erzeugen von

beliebigen Flächen folgen.

Das folgende Programm ist Abschluß und Höhepunkt unserer kleinen Programmsammlung für Druckerbesitzer.

Mit diesem Programm lassen sich die komplexesten Flächen erzeugen. Nicht nur ein Punkt, von dem aus sich Wellen über die Fläche ausbreiten, läßt sich eingeben, nein, bis zu vier solcher Punkte können eingegeben werden, wobei jeder Punkt noch unterschiedliche Wellenlängen und Wellenhöhen haben kann. Die Anzahl der möglichen Grafiken ist schier unendlich.

Wie alle Basic-Programme mit einer Menge von sich immer wiederholenden Berechnungen ist auch dies Programm relativ langsam. Alle diese Programme sind dankbares Objekt guter Compiler. Die Rechenzeit kann durch kompilieren um einiges verkürzt werden.
Übrigens dauert eine volle Fläche mit einer Spitze etwas länger als 2 Stunden, wenn die Daten auf Diskette geschrieben werden.
Wie auch das vorher besprochene Programm läuft dieses zusammen mit allen 7-Nadel-Druckern. Allerdings bereitet das Anpassen an andere Druckertypen keine große Schwierigkeit.

```
100 PI=3.14159
110 DIML%(480),NS(6),HM(4),XP(4),YP(4),WX(4),WY(4)
120 X=1
130 FORI=0TO6
140 NS(I)=X
150 X=X+X
160 NEXT
170 PRINTCHR$(147)
```

```
180 PRINT:PRINT"          3D-FLAECHENPLOTTER"
190 FORI=0TO480
200 L%(I)=128
210 NEXT
220 I=0
230 PRINT:PRINT"   GUENSTIGE WERTE ODER BEREICHE"
240 PRINT"     WERDEN IN KLAMMEN ANGEGEBEN"
250 PRINT:PRINT"          WUENSCHEN SIE DIE"
260 PRINT:PRINT"  AUSGABE AUF FLOPPY ODER DRUCKER"
270 INPUT"                ";A$
280 IFA$="F"THENOPEN1,8,4,"3D-DATEN,S,W" :GOTO310
290 IFA$="D"THENOPEN 1,4:GOTO310
300 GOTO 270
310 PRINT#1:PRINT#1,CHR$(8);
320 PRINT:PRINT"      BITTE PARAMETER EINGEBEN"
330 PRINT:PRINT
340 INPUT"DOTLINIEN VERTIKAL       ";A$:LN%=VAL(A$)
350 MF%=(LN%*2/3):IFLN%<1ORLN%>360THEN340
360 PRINT:INPUT"ANZAHL DER SPITZEN (1-4)";A$:NP%=VAL(A$)
370 IF NP%<1ORNP%>4THEN360
380 PRINT:INPUT"LINKER RAND         (150)";A$:LM%=VAL(A$)
390 PRINTCHR$(147):GOTO420
400 FOR I=1TONP%
410 PRINT CHR$(19)
420 PRINT:PRINT"   SPITZE NR."I:PRINT"   X POSITION (0 BIS"MF%")";
430 IFFF=1THEN INPUT A$:XP(I)=VAL(A$)
440 IFFF=0THENPRINT
450 PRINT:PRINT"   SPITZE NR."I:PRINT"   Y POSITION (0 BIS"MF%")";
460 IFFF=1THEN INPUT A$:YP(I)=VAL(A$):M%=MF%/7
470 IFFF=0THENPRINT
480 PRINT:PRINT"   SPITZE NR."I:PRINT"   X WELLENLÄNGE     ("M%")";
490 IFFF=1THEN INPUT A$:WX(I)=2*PI/VAL(A$)
500 IFFF=0THENPRINT
510 PRINT:PRINT"   SPITZE NR."I:PRINT"   Y WELLENLÄNGE     ("M%")";
```

```
520 IFFF=1THEN INPUT A$:WY(I)=2*PI/VAL(A$):M%=LM%+XP(I)-20
530 IFFF=0THENPRINT
540 PRINT:PRINT"   SPITZE NR."I:PRINT"   HOEHE      (20 BIS"M%")";
550 IFFF=1THEN INPUTA$:HM(I)=VAL(A$)
560 IFFF=0THENFF=1:GOTO400
570 NEXTI:PRINT
580 GN=1/10000:K=0:MM%=480
590 FORI=1TOLN%*1.25:PRINTI;
600 IL=2000:IH=-IL
610 FORJ=1TOMF%
620 IFJ>ITHENJ=MF%:GOTO740
630 IX=(I-J)*2:IFIX>MF%THEN740
640 X=IX:Y=J:Z=IX+LM%
650 FORN=1TONP
660 X1=(X-XP(N))*WX(N)+GN
670 Y1=(Y-YP(N))*WY(N)+GN
680 W=SQR(X1*X1+Y1*Y1)+GN:Z=Z-SIN( W)/W*HM(N)
690 NEXTN
700 IZ=Z
710 IFIZ<0ORIZ>MM%GOTO740
720 IFIZ>IHTHENIH=IZ:L%(IZ)=L%(IZ)ORNS(K)
730 IFIZ<ILTHENIL=IZ:L%(IZ)=L%(IZ)ORNS(K)
740 NEXTJ
750 K=K+1
760 IFK>6THENGOSUB820
770 NEXTI
780 IFK>0THEN  GOSUB820
790 GOSUB820
800 PRINT#1,CHR$(15):CLOSE1
810 END
820 FORJ=1TOMM%
830 PRINT#1,CHR$(L%(J));:L%(J)=128
840 NEXTJ:K=0:PRINT:PRINT#1
850 RETURN
```

Auch bei diesem Programm kann die Ausgabe wahlweise direkt auf den Drucker oder zuerst auf die Floppy gehen. Werden die Daten auf der Floppy gespeichert, dann können sie mit dem kurzen Lade-Programm, das Ihnen schon von der Kugel her bekannt ist, ausgedruckt werden. Vergessen Sie aber nicht, den richtigen File-Namen einzusetzen. Sonst heißt es 'File Not Found'.

Bei diesem Programm sind einige Parameter einzugeben. Die jeweils günstigsten Werte oder Bereiche werden Ihnen als Vorschlag angezeigt.
Zuerst werden Sie nach der vertikalen Anzahl der Druckpunkte gefragt. Wählen Sie zum Probieren hier kleine Werte, etwa 50 oder 80 Punkte reichen, um das spätere Aussehen zu beurteilen. Gefällt Ihnen dann das Bild, so können Sie es mit mehr Punkten ausdrucken.
Die nächste Eingabe fragt die Anzahl der Spitzen ab. Eine Spitze ist der Mittelpnkt von Wellenlinien, vergleichbar der Stelle auf der Wasseroberfläche, an der ein geworfener Stein eingetaucht ist.
Nach der Eingabe dieser drei Werte erscheint auf dem Bildschirm eine Maske, in die die Parameter für die Positionen der Spitzen, die Wellenlängen sowie die Höhen der Spitzen eingegeben werden können.
Verblüffen mag vielleicht die Frage nach Wellenlänge in X- und Y-Richtung, aber es lassen sich tatsächlich nicht nur kreisrunde sondern auch ovale Wellenformen ausgeben.

Sind alle Parameter eingegeben, geht das große Rechnen los. Der Faktor 1.25 in der Zeile 590 kann von Drucker zu Drucker unterschiedlich sein, je nach Verhältnis von horizontalen zu vertikalen Druckpositionen. Im übelsten Fall werden die Wellen dann trotz gleicher Wellenlängen in X- und Y-Richtung oval ausgedruckt. Experimentieren Sie in diesem Fall ein

wenig mit anderen Werten, Sie finden sicher schnell einen geeigneten Wert.
In den Zeilen 620 bis 670 werden die Punkte berechnet, in den Zeilen 690 bis 720 wird entschieden, ob es sich um einen sichtbaren Punkt handelt, oder ob der Punkt verdeckt und damit unsichtbar ist. Entsprechend wird das richtige Bitmuster aus dem Array NS auf den eventuell schon vorhandenen Wert aufgeODERt.
Der Ausdruck der Grafik in den Zeilen 810 bis 830 ist ganz ähnlich der Routine, die schon bei der Kugel verwendet wurde.

Damit ist das Kapitel Grafik beendet. Mit den gezeigten Methoden und ein wenig Fantasie können Sie sicher ähnliche Grafiken programmieren. Nur Mut, auch wenn es nicht auf Anhieb klappt, ein hämisches 'SYNTAX TERROR' auf dem Bildschirm ist nichts Ehrenrühriges.

## *6.0 DRUCKPRINZIPIEN*

Von der Typenhebelschreibmaschine bis zum Hochleistungszeilendrucker war es ein ganz schön langer Weg. Nachdem man nun nach allgemeiner Einführung der EDV nun die Möglichkeit hatte, Berechnungen blitzschnell durchzuführen, kam man zu der Erkenntnis, daß das Ganze wenig nützt, wenn die Ergebnisse nicht ebenso blitzartig zu Papier gebracht werden können. Sie werden es kaum glauben, aber Papier ist immer noch der wichtigste Datenträger. Ist ja auch klar: Versuchen Sie doch einmal, sich auf einer Diskette handschriftliche Notizen zu machen.

Was gab es damals eigentlich an Geräten, die sich überhaupt zum ferngesteuerten Druck eigneten? Sicher, den guten alten Fernschreiber, und der wurde in der Tat gründlich hergenommen. Viele Computerfreaks machen das aus Kostengründen auch heute noch. Aber mit einer Druckgeschwindigkeit von sechs Zeichen/Sekunde ist nun mal kein Blumentopf zu gewinnen.

In der sog. Groß-EDV erfand man dann auch schnell neue Druckverfahren. Platz war ja genug vorhanden, und so waren (und sind) denn auch die Drucker recht schrankähnlich in ihren Ausmaßen. Aber sie hatten es in sich. Sie schafften lockere 600 Zeilen! pro Minute. Der Drucker stand an zentraler Stelle, und jeder Benutzer mußte sich dort seinen Ausdruck abholen.

Die Entwicklung, von der Sie heute als Homecomputer-Anwender profitieren, setzte erst ein, als man in der gehobenen EDV durch den Anschluß von mehreren Bildschirmen an einen Großrechner quasi den Computer an den Arbeitsplatz holte. Wo aber ein Terminal auf dem Schreibtisch steht, darf auch ein Drucker nicht fehlen, wenn man die Effizienz nicht in Frage stellen will. Also mußten kleine, aber gleichwohl leistungsfähige Drucker her.

Mittlerweile hielt der Computer auch Einzug in die guten Stuben. Also mußte die Entwicklung auch in Richtung Kostensenkung getrieben werden. Was dabei herauskam, kann sich durchaus sehen lassen, wie wir meinen.

Es haben sich, was den rein mechanischen Vorgang des Druckens betrifft, drei Verfahren herausgebildet, von denen jedes seinen berechtigten Platz auf der Preis-Leistungsskala einnimmt.
Diese drei Verfahren möchten wir Ihnen nachfolgend einmal vorstellen.

## *6.1 Der Typenraddrucker*

Das Druckprinzip eines solchen Gerätes wurde aus der allseits bekannten Kugelkopfmaschine hergeleitet.
Voraussetzung für die Verarbeitung von Endlospapier ist eine horizontal unbewegliche Papiertransporteinrichtung. Nun ist jedoch ein Kugelkopfmechanismus so kompliziert, daß er meistens stationär eingebaut wird, und das Papier mittels eines Druckwagens an ihm vorbeigeführt wird. In Ausnahmefällen ist der Kugelkopf auch verschiebbar, jedoch erfordert das eine aufwendige und damit teure und schwere Horizontalführung.
Darüberhinaus ist der Kugelkopf als solcher recht kostspielig, was dann zu Buche schlägt, wenn man zwischen mehreren Schriftarten wechseln will.

Deshalb hat man sich Gedanken gemacht, wie ein derartiger Mechanismus durch ein leichtes und preiswertes Pendant zu ersetzen ist, welches leicht auf einem Transportschlitten vor dem Papier hin- und hergefahren werden kann.
Das Typenrad war geboren. Das Prinzip ist genial einfach:

Man nehme eine Scheibe, auf deren Umfang alle notwendigen Buchstaben, Zahlen und Sonderzeichen angebracht sind, montiere sie auf die Achse eines kleinen Motors und schleudere sie gegen das Papier, wenn das gewünschte Zeichen gerade in der richtigen Position steht. Übrigens, Kinderschreibmaschinen arbeiten nach dem gleichen Prinzip, nur daß dort die Scheibe von Hand gedreht werden muß.
Nun wird natürlich nicht die ganze Scheibe gegen das Papiergedrückt, sondern nur das Kreissegment, welches das Zeichen trägt. Zu diesem Zweck sind die Typen an federnden Zungen strahlenförmig auf der Achse befestigt. In Höhe der Druckposition befindet sich ein kleiner, elektromagnetisch angetriebener Hammer, der die gerade vor ihm befindliche Type mitsamt dem Farbband gegen das Papier schleudert.
Die Zeichenauswahl geschieht über einen Schrittmotor, auf dessen Achse das Typenrad befestigt ist.
Um das Wesen eines Schrittmotors kurz zu erklären: Ein gewöhnlicher Elektromotor läuft los, sobald er Spannung erhält. Wird der Strom abgeschaltet, läuft er mehr oder weniger lange nach und kommt dann zum Stillstand. Ein solcher Motor wäre für unseren Zweck denkbar ungeeignet, denn es kommt ja darauf an, das zu druckende Zeichen möglichst schnell und exakt vor den Hammer zu plazieren.
Ein Schrittmotor nun, ohne jetzt tief in die Materie einzusteigen, bewegt sich bei jedem Stromimpuls um einen genau definierten Winkel. Deshalb läßt sich bei bekannter Verteilung der Zeichen auf dem Typenrad durch einfaches Abzählen der Impulse das gewünschte Zeichen genau vor den Hammer positionieren.

Die gesamte Anordnung ist auf einem Schlitten montiert, der ebenfalls von einem Schrittmotor bewegt wird, um so auch die Zeichen horizontal möglichst genau an ihren Platz zu setzen.

Die Schriftqualität eines solchen Druckers ist erstklassig. Die Schriftart läßt sich durch einfaches Auswechseln des Typenrades bequem ändern.
Natürlich dauert die Bewegung des Typenrades seine Zeit. Deshalb zählt ein Typenraddrucker systembedingt nicht zu den Schnellsten. Sie können mit einer Druckgeschwindigkeit von ca. 15 bis 80 Zeichen/Sekunde rechnen. Man kann sagen, daß der Preis proportional mit der Druckgeschwindigkeit steigt. Er liegt ca. zwischen DM 1400 und DM 8000.

Bevorzugtes Einsatzgebiet eines Typenraddruckers ist natürlich, wegen des hervorragenden Schriftbildes, Korrespondenz jeder Art.

### *6.2 Der parallele Matrixdrucker*

In dem Bemühen, die Druckgeschwindigkeit bei geringen Kosten zu erhöhen, mußten neue Wege beschritten werden. Das Typenprinzip, wie auch immer geartet, steht wegen seiner Massenträgheit einer wesentlichen Heraufsetzung der Geschwindigkeit entgegen. Es mußte also ein total anderes Prinzip her.
Dieses sollte möglichst viele Zeichen mit möglichst wenig Bewegung reproduzieren können. Was also lag näher, als die Zeichen aus nur wenigen Segmenten zusammenzusetzen. Man ging noch weiter.
Da der Druckkopf sich ja ohnehin über das Papier bewegt, braucht man man nur genügend Segmente für einen senkrechten Ausschnitt des Zeichens. Der nächste Ausschnitt kann ja bei fortschreitender Bewegung des Kopfes von denselben Segmenten gedruckt werden.
Diese Überlegung reduziert das Segment zum Punkt, denn der dichteste senkrechte Ausschnitt ist der Strich, und der kann

aus Punkten zusammengesetzt werden. Folglich ist jeder weniger dichte Ausschnitt gleichermaßen reproduzierbar.
Ein Punkt kann durch das Drücken einer Nadel gegen Farbband und Papier erzeugt werden. Eine Nadel ist klein und leicht, hat demzufolge eine geringe Massenträgheit und kann deshalb sehr schnell bewegt werden.

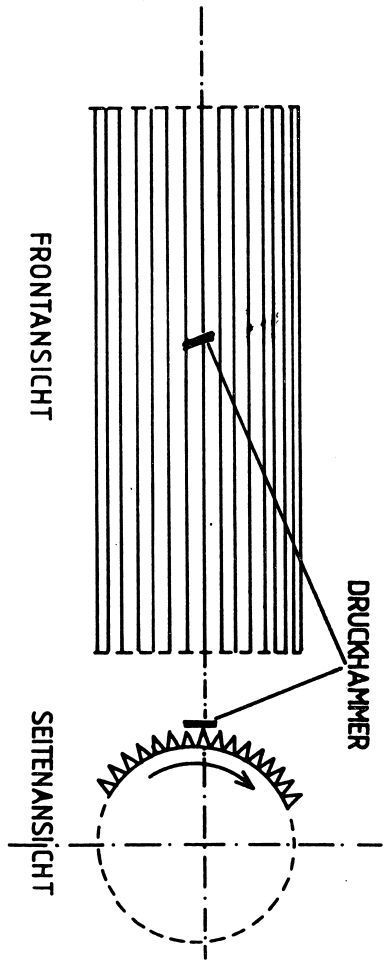
Der Vorteil, der sich aus diesem Druckprinzip ergibt, liegt auf der Hand. Man ist nämlich nicht unbedingt auf einen festen Zeichenvorrat fixiert, sondern kann auf diese Weise jedes beliebige Zeichen erzeugen, wie man an der Einzelnadelgrafikfähigkeit, die mittlerweile alle derartigen Drucker besitzen, sehen kann.
Der Zeichenvorrat ist im Betriebssystem des Gerätes verankert. Wie so etwas aussieht, können Sie im Abschnitt 7.3 nachschauen. Der Zeichensatz wird einfach mit dem Wert des ankommenden Zeichens adressiert, das Punktemuster herausgeholt und damit die Nadeln des Druckkopfes angetrieben. Auf dem Papier findet sich dann das Bitmuster des Zeichenspeichers wieder.
Die Anzahl der Drucknadeln bewegt sich gewöhnlich zwischen sieben und neun. Je mehr Nadeln vorhanden sind, desto feiner ist die Schrift.
Die Druckgeschwindigkeit bewegt sich zwischen 100 und 400 Zeichen/Sekunde. Im sogenannten Schönschriftmodus kann sie bis auf 40 Zeichen absinken, da hierbei der Druckkopf zweimal über das Papier fährt, wobei das Papier vor dem zweiten Durchlauf um Bruchteile eines Millimeters verschoben wird. Dadurch wird das Bild dichter. Bei einigen Geräten hat man es bis zur Perfektion getrieben, so daß das Schriftbild von dem eines Typenraddruckers kaum mehr zu unterscheiden ist.
Die Preise bewegen sich zwischen ca. DM 1000 und DM 4000.
Ein solches Gerät ist hervorragend für den 'Vieldrucker'

geeignet, der auch ab und zu mal seine Korrespondenz damit erledigt.

### *6.3 Der serielle Matrixdrucker*

Dieser Terminus war bis vor kurzem noch absolut unbekannt. Er entstand, als die Homecomputer deutlich unter die DM 1000-Grenze rutschten. Erfahrungsgemäß findet Computerzubehör (dazu zählen wir auch die Drucker) nur dann regen Zuspruch, wenn es nicht teurer als der Rechner selbst ist.
Findigen Köpfen gelang es tatsächlich, die Druckmechanik auf ein derartiges Minimum zu reduzieren, daß ein Preis von ca. DM 600 bis DM 900 herauskam.
Wie ist das möglich? Ganz einfach; man braucht nur den Gedanken aus Abschnitt 6.2 zur Reduktion der Anzahl der notwendigen Segmente wieder aufgreifen.
Hat man nicht den Ehrgeiz, eine ganze Spalte auf einmal zu drucken, kommt man selbstverständlich auch mit einer Nadel aus.
Sie haben sicher schon gemerkt, wo hier der Hase im Pfeffer liegt. Der spaltenweise Druck war ja nur möglich, weil der Kopf kontinuierlich am Papier vorbeigeführt wurde. Jetzt aber, bei nur einer Nadel, müßte zum Druck einer Spalte entweder Kopf oder Papier auch noch vertikal bewegt werden. Es kann Ihnen sicher niemand erzählen, daß die Mechanik dadurch billiger würde.
Es ist auch nicht an dem. Weder Kopf noch Papier machen eine derartige Bewegung.
?????
Die Lösung ist genial einfach. Da sich das Prinzip nur verbal nicht veranschaulichen läßt, schauen Sie sich zu der folgenden Erläuterung einmal die Schemazeichnung auf der nächsten Seite an.

Schema des seriellen Matrixdruckers



Bisher war es üblicherweise so, daß das Papier vor einer festen Anschlagleiste oder einer Gummiwalze geführt wurde. Hier haben wir es jetzt mit einer sogenannten Stachelwalze zu tun.

Diese Stachelwalze ist fest mit dem Kopfbewegungsmechanismus verkoppelt. Bewegt sich der Kopf auf seinem Schlitten, dreht sich auch die Walze. Der Druckhammer ist nicht mehr als Nadel ausgebildet, sondern als Balken. Dafür, daß er beim Auftreffen auf das Papier dennoch nur einen Punkt hinterläßt, sorgt eben die Stachelwalze, denn ein Abdruck erfolgt nur dort, wo der Hammer und ein erhabener Teil der Walze aufeinandertreffen.

Wenn der Kopf von links nach rechts fährt, dreht sich vor ihm die Walze von unten nach oben. Was jetzt passiert, ist völlig klar: Der Hammer muß zum Druck einer Spalte wiederholt auf das Papier schlagen. Bei jedem weiteren Schlag ist die Walze aber ein Stückchen höher gewandert. Demzufolge sitzt auch der auf dem Papier erzeugte Punkt ein Stück höher als der vorangegangene. Da aber der Kopf auch weiterrutscht, ist der Druckhammer etwas schräg angeordnet, um die Bewegung auszugleichen und die Spalte senkrecht erscheinen zu lassen.

Damit der steuernde Prozessor weiß, wann er den Hammer auslösen darf, wird von der Walze eine sogenannte Taktscheibe angetrieben, deren Schlitze sich in einer Lichtschranke bewegen. Bei jedem Lichtimpuls darf der Hammer einmal gefeuert werden, da sich dann die Walze um eine Punktedicke fortbewegt hat.

Die Druckgeschwindigkeit ist natürlich nicht berauschend, da der Druckhammer ja viel öfter bewegt werden muß als eine Nadel des Paralleldruckers. Sie können mit ca. 50 bis 80 Zeichen/Sekunde rechnen.

Für gelegentliche Druckarbeiten sicher ein Gerät mit einem vorzüglichen Preis- Leistungsverhältnis.

## *7.0 DAS BETRIEBSSYSTEM DES MPS 801*

Es ist sicher schon viel über Betriebssysteme aller Art geschrieben worden, aber ein Betriebssystem für Drucker war unseres Wissens bisher noch nicht dabei.
Wir sind der Meinung, daß man dieses Buch schwerlich als abgerundet betrachten darf, wenn es nicht auch etwas für den Profi oder engagierten Laien enthält.
Die Bezeichnung "Betriebssystem" mag wohl für einen Drucker etwas hochtrabend klingen, aber wie sie sehen werden, kann man es durchaus als solches bezeichnen, denn je einfacher der Aufbau eines Gerätes ist, desto umfangreicher muß die Betriebssoftware sein.
Der MPS 801 ist ein Paradebeispiel für minimalen Aufwand im Aufbau. Er arbeitet nach dem Prinzip des in 6.3 beschriebenen seriellen Matrixdruckers.
Unsere Wahl fiel auch deswegen auf das Gerät, weil es noch ziemlich neu am Markt ist und deshalb auch noch einige Zeit angeboten werden wird, so daß sich auch aus dieser Sicht der Aufwand lohnt.

Es war natürlich gar nicht so einfach, das Betriebssystem auch nur halbwegs zu kommentieren. Es fing damit an, daß wir überhaupt erst einmal einen Disassembler für den Prozessor 8039 schreiben mußten. Dann gab es Interpretationsschwierigkeiten bei den Befehlen, da wir bisher nur den 6502 oder den Z80 gewohnt waren.
Entsprechend holperig sind auch die Kommentare, wofür wir vorsorglich um Entschuldigung bitten.
Nun gut, wir haben es überstanden. Damit Sie sich als 6510-Spezi etwas leichter tun, werden wir Ihnen zunächst die Besonderheiten des Prozessors 8039 etwas näherbringen und Ihnen auch einige Befehle vorstellen, die aus der Kenntnis der 6502-Maschinensprache nicht so ohne weiteres abgeleitet

werden können.
Danach werden wir ein wenig die Hardware des Druckers anhand des Blockschaltbildes erklären und dabei die zu ihrer Handhabung notwendigen Befehle anreißen.

## *7.1 Der Prozessor 8039*

Der 8039 zählt zur Klasse der Single-Chip-Prozessoren, d.h. er enthält praktisch alles, was zum Betrieb einer einfachen umgebenden Hardware notwendig ist, ohne daß unterstützende Bausteine wie Ram oder I/O-Ports notwendig wären. Alles, was er braucht, ist der externe Programmspeicher und, wie in unserem Falle, Treiberbausteine, um Motoren und dergleichen antreiben zu können. Das ist aber wirklich schon alles.
Kurz ein paar Leistungsdaten:

* 128 Bytes Ram
* Zähler/Zeitgeber
* 16 I/O-Leitungen
* 2 separate Testeingänge

Das befindet sich wohlgemerkt alles auf dem IC.
Nun hat das Ding, bei aller Kompaktheit, so seine Tücken. Eine davon ist, daß nur 2K Programmspeicher durchgängig adressiert werden können. Für einen weiteren 2K-Bereich muß umgeschaltet werden. 4K ist das Maximum. Eine andere Einschränkung ist, daß der Stack nur acht Ebenen hat. Unterprogrammschachtelung ist daher nur in begrenztem Umfang möglich. Davon wird auch recht wenig Gebrauch gemacht, wie Sie sehen werden.
Nun einige Befehle, die Sie sicher nicht kennen.

Sprungbefehle:
jt0 oder jnt0: springen, falls Leitung T0 high oder low ist

jtl oder jntl: wie oben, jedoch für Leitung Tl
jni : springen, falls Leitung INT=low
jf0 : springen, falls Flag0 gesetzt
jfl : wie oben, jedoch für Flagl

Die Flags werden als Merker im Programm verwendet. Sie werden gelöscht mit clr f0, bzw. clr fl. Gesetzt werden sie mit anschließendem cpl f0, bzw. cpl fl.

Alle Sprungbefehle, mit Ausnahme von jmp und call, können nur innerhalb einer Page springen. Deshalb sind deren Operanden auch nur ein Byte lang. Angenommen, Sie haben folgenden Befehl:

0ABF jnz 00

Der Sprung wird, falls der Akkuinhalt ungleich Null ist, nicht etwa nach 0000 führen, sondern nach 0A00. Die Page 0A in unserem Beispiel kann mit derartigen Sprüngen nicht verlassen werden.

Anders ist es bei jmp und call. Diese können sich innerhalb eines 2K-Bereiches bewegen. Soll der Bereich verlassen werden, muß zuvor der Speicher umgeschaltet werden. Das geschieht so:

0000 sel mbl
0001 jmp 000

Dieser Sprung führt nicht etwa nach 0000, sondern nach 0800, da zuvor der Speicher umgeschaltet wurde. Die Umschaltung erfolgt mit sel mb0 für den unteren 2K-Bereich, und mit sel mbl für den oberen. Sie wird erst beim jeweils nächsten Befehl wirksam, da ja sonst der jmp in unserem Beispiel nicht mehr ausgeführt werden würde.

Damit Sie bei der Lektüre des Rom-Listings nicht bei jedem derartigen Befehl schauen müssen, ob nicht vielleicht ein sel mb0 oder sel mbl davorstand, haben wir von uns aus die Zieladressen bereits entsprechend korrigiert.

Transferbefehle:

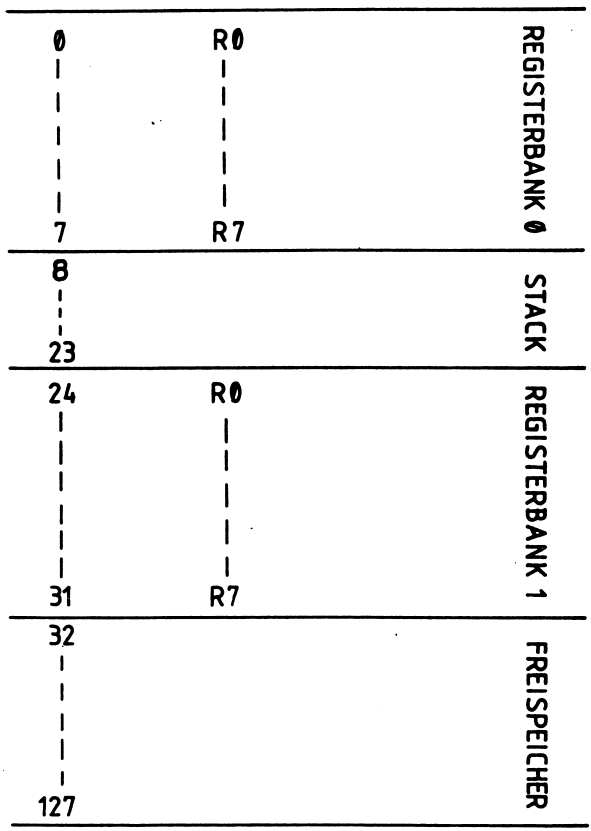
Transferiert werden kann alles vom Akkumulator und zum Akkumulator. Immediate-Operanden (das sind die mit vorangestelltem #) können überall hin transferiert werden. Ist einem Operanden ein 'Klammeraffe' vorangestellt, ist dieser nicht Ziel oder Quelle der Operation, sondern enthält die Adresse des Operanden. Dieses Verfahren ist bei Zugriffen auf den Ram erforderlich, soweit der Zugriff nicht im Bereich der Register liegt.

Die Speicherorganisation:
Wie Sie an der Memory-Map auf der nächsten Seite sehen können, ist der Speicher in zwei Registerbanks, Stack und übrigen Speicher aufgeteilt. Er kann auf allen Positionen indirekt adressiert werden. Im Bereich der Register ist auch ein direkter Zugriff möglich.
Beispiel:

```
0000  mov rl,#03
0002  sel rbl
0003  mov rl,#04
```

Mit dem ersten Befehl wird Speicherzelle 1 geladen, mit dem dritten Befehl jedoch Speicherzelle 25, da zuvor auf die Registerbank 1 umgeschaltet wurde. Zurück geht es wieder mit sel rb0.

## Memory-Map des 8039




Der Bereich 34....123 wird im MPS-801 als Zeilenpuffer benutzt.

Sicher vermissen Sie Vergleichsbefehle. Solche gibt es im Befehlsvorrat des 8039 nicht. Verglichen werden kann nur indirekt, indem auf den Vergleichsoperanden z.B. das Einerkomplement aufaddiert und das Ergebnis auf Null abgefragt wird.
Angenommen, sie haben im Akku einen Wert stehen, von dem Sie gerne wüßten, ob er =3 ist, gehen Sie so vor:

```
0123  add a,#FD
0125  jz  40
```

Falls der Akku 3 enthalten hatte, wird der Sprung durchgeführt. Ein wenig umständlich, aber es geht.

Wir wollen es dabei bewenden lassen. Beispiele und Erläuterungen zur I/O-Programmierung finden Sie im nachsten Abschnitt.

## *7.2 Blockschaltbild MPS 801*

Was sie auf der Abbildung sehen, ist tatsächlich fast alles. Wir haben der Übersicht wegen nur den Programmspeicher und die Leistungstreiber für die Motoren usw. weggelassen. Wir wollen Ihnen nun kurz die Signalnamen erläutern.

Die Signale auf der linken Seite gehören zum IEC-Bus, der Ihnen ja aus 2.1 bekannt ist. Die Signale auf der rechten Seite steuern die Innereien des Druckers.

TEST Dieses Signal stammt vom Schalter auf der Rückseite des Druckers. Steht dieser in Stellung TEST, ist diese Leitung=low.

4/5 Auch dies ist ein Schaltersignal und ist low, wenn der Schalter auf Geräteadresse 5 steht, ansonsten high.

TAKTRAD Dies ist das Signal von der in 6.3 angesprochenen Taktscheibenlichtschranke.

GRUNDSTELLUNG Steht der Druckwagen am linken Anschlag, unterbricht er dort eine Lichtschranke. Ist das der Fall, ist dieses Signal=high.

LF-TASTE Wenn die LF-Taste auf der Geräteoberseite gedrückt wird, ist diese Leitung=low.

KOPFMOTOR EIN Bei low an diesem Ausgang wird der Transportmechanismus für den Druckkopf in Gang gesetzt. Gleichzeitig drehen sich dann auch Stachelwalze und Taktscheibe.

WAGENRÜCKLAUF EIN Low an dieser Leitung löst eine Kupplung aus, die den Druckkopf vom Antrieb trennt. Nun wird der Kopf durch Federkraft in die Grundstellung zurückgezogen.

ZEILENVORSCHUB EIN Das ist die Hauptstromversorgung für den Zeilentransport-Schrittmotor. Tatsächlich bewegen wird sich der Motor nur durch ein umlaufendes Signal an den Leitungen LF1, LF2, LF3 und LF4.

DRUCKHAMMER EIN Wie der Name sagt, wird mit low auf dieser Leitung der Druckhammer gefeuert.

Programmierung der Portleitungen:
Die Signalausgänge werden in der Regel direkt gesteuert. Ein Low wird durch ein logisches UND erzeugt, ein High durch logisches ODER, z.B.

```
0011  anl p1,#FE
0013  orl p1,#01
```

Durch diese Befehlsfolge würde der Druckhammer kurz

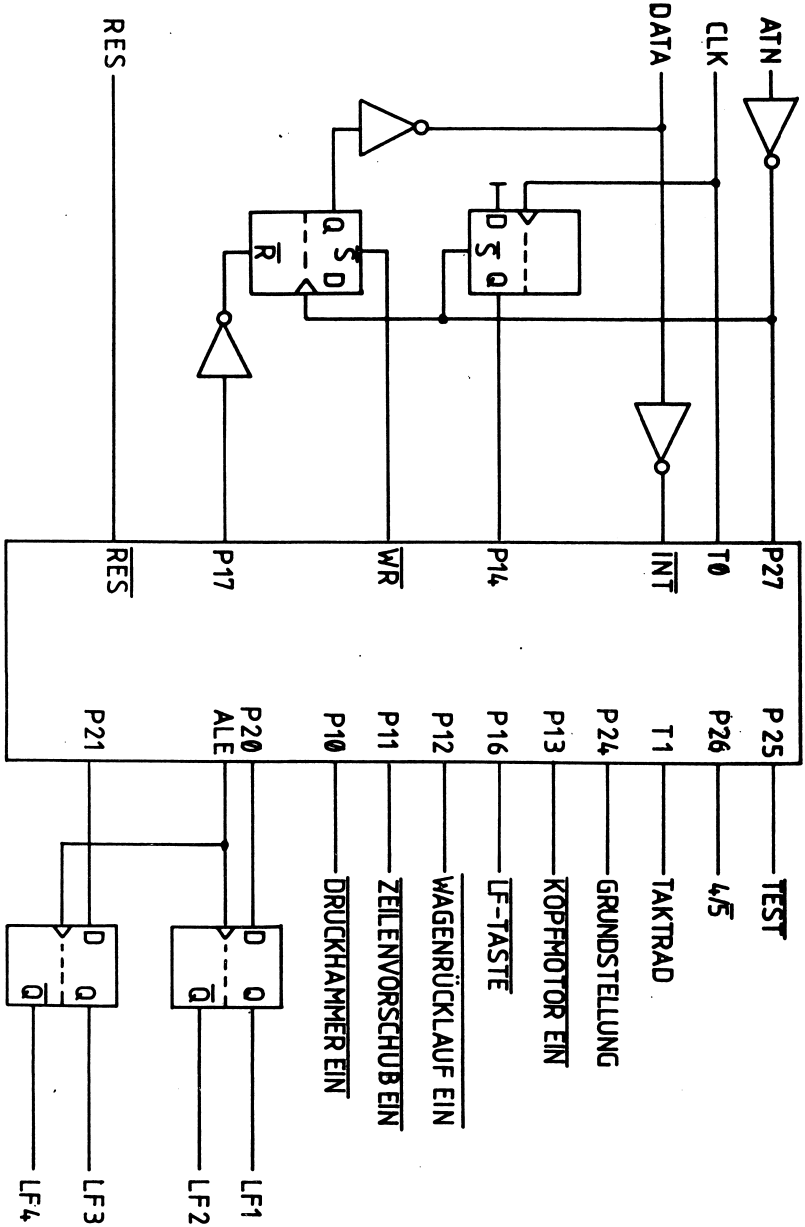
ausgelöst werden.
Der Zustand der Portleitungen kann durch Hereinholen in den Akku und anschließendes Testen des Gewünschten Bits festgestellt werden.

```
0025  mov a,p2
0026  jb6 47
```

Mit dieser Sequenz würde ein Sprung nach 0047 durchgeführt, wenn der Adreßschalter auf 4 steht.
Die Leitungen T0 und T1 werden mit den schon beschriebenen Sprüngen jt0, jt1, jnt0 und jnt1 abgefragt.

So, nun kann es ins Betriebssystem gehen. Sollte Ihnen der eine oder andere Befehl unklar bleiben, besorgen Sie sich eine Befehlsliste beim Elektronik-Fachhändler. Das dürfte nicht allzu schwer sein, da der 8039 von recht vielen Herstellern angeboten wird.

## Blockschaltbild des MPS 801



```
0000  0410  jmp  010
0002  00    nop
0003  00    nop
0004  00    nop
0005  00    nop
0006  00    nop
****************************** Fehlereinsprung
0007  65    stop tcnt
0008  35    dis  tcnti
0009  15    dis  i
000A  89FF  orl  p1,#FF  (255)
000C  8AFC  orl  p2,#FC  (252)
000E  040E  jmp  00E           Dead-Lock
0010  89FF  orl  p1,#FF  (255)
0012  8AFC  orl  p2,#FC  (252)
0014  15    dis  i
0015  E5    sel  mb0
0016  C5    sel  rb0
0017  65    stop tcnt
0018  35    dis  tcnti
0019  99FD  anl  p1,#FD  (253)
001B  F5    sel  mb1
001C  D437  call 637           Delay
001E  E5    sel  mb0
001F  89FF  orl  p1,#FF  (255)
******************************
0021  27    clr  a             Registerbank 0
0022  B802  mov  r0,#02  (2)   und
0024  B916  mov  r1,#16  (22)  Stack
0026  A5    clr  f1            loeschen
0027  A0    mov  @r0,a
0028  18    inc  r0
```

```
0029  E927  djnz rl,27
******************************
002B  0A    in   a,p2          ATN ?
002C  F22B  jb7  2B            ja, warten
002E  997F  anl  pl,#7F  (127) DATA freigeben
0030  BC01  mov  r4,#01  (1)
0032  5400  call 200           Kopftransport und Taktrad testen
0034  B811  mov  r0,#11  (17)
0036  F0    mov  a,@r0
0037  53FC  anl  a,#FC   (252)
0039  A0    mov  @r0,a
003A  5402  call 202           Kopf in
003C  99FB  anl  pl,#FB  (251) Grundstellung
003E  B4CD  call 5CD           bringen
0040  B810  mov  r0,#10  (16)
0042  B018  mov  @r0,#18 (24)
******************************
0044  27    clr  a             Registerbank 1
0045  B816  mov  r0,#16  (22)
0047  A0    mov  @r0,a         und
0048  18    inc  r0
0049  A0    mov  @r0,a         Speicher
004A  B81A  mov  r0,#1A  (26)
004C  B966  mov  rl,#66  (102) loeschen
004E  A0    mov  @r0,a
004F  18    inc  r0
0050  E94E  djnz rl,4E
******************************
0052  85    clr  f0
0053  AA    mov  r2,a          Registerbank 0
0054  AB    mov  r3,a
0055  AD    mov  r5,a          ruecksetzen
0056  AE    mov  r6,a
0057  B821  mov  r0,#21  (33)
```

```
0059  B002  mov  @r0,#02 (2)
005B  18    inc  r0
005C  FC    mov  a,r4
005D  37    cpl  a
005E  B008  mov  @r0,#08 (8)
0060  1268  jb0  68
0062  B00F  mov  @r0,#0F (15)
0064  F268  jb7  68
0066  B00E  mov  @r0,#0E (14)
0068  18    inc  r0
0069  B918  mov  r1,#18  (24)
006B  B100  mov  @r1,#00 (0)
006D  2379  mov  a,#79   (121)
006F  5C    anl  a,r4
0070  AC    mov  r4,a
0071  F8    mov  a,r0
0072  AF    mov  r7,a
0073  E400  jmp  700            ENDE RESET
***************************** Einsprung aus LISTEN mit
                                Datenbyte in R2
0075  FE    mov  a,r6
0076  C67A  jz   7A
0078  4427  jmp  227
007A  FC    mov  a,r4
007B  128E  jb0  8E
007D  FA    mov  a,r2
007E  F282  jb7  82           Datenbyte > 127 ? ja -->
0080  2400  jmp  100
0082  B821  mov  r0,#21  (33)
0084  B000  mov  @r0,#00 (0)
0086  BB01  mov  r3,#01  (1)
0088  B918  mov  r1,#18  (24)
008A  B100  mov  @r1,#00 (0)
008C  04B5  jmp  0B5
```

```
*******************************
008E  FA     mov  a,r2
008F  5360   anl  a,#60   (96)  abdruckbares Zeichen ?
0091  9695   jnz  95             ja -->
0093  2400   jmp  100
0095  FA     mov  a,r2
0096  03BF   add  a,#BF   (191) >64 ?
0098  F6A4   jc   A4             ja -->
009A  031F   add  a,#1F   (31)  Anfuehrungszeichen ?
009C  96A2   jnz  A2             nein -->
009E  FC     mov  a,r4           Anfuehrungszeichen-
009F  D302   xrl  a,#02   (2)    Flag
00A1  AC     mov  r4,a           flippen
00A2  0482   jmp  082
00A4  FC     mov  a,r4
00A5  72AE   jb3  AE
00A7  FA     mov  a,r2
00A8  F5     sel  mb1
00A9  9400   call C00
00AB  E5     sel  mb0
00AC  0482   jmp  082
00AE  FA     mov  a,r2
00AF  F5     sel  mb1
00B0  34D8   call 9D8
00B2  E5     sel  mb0
00B3  0482   jmp  082
******************************* Byte puffern
00B5  FF     mov  a,r7
00B6  A8     mov  r0,a
00B7  FA     mov  a,r2
00B8  A0     mov  @r0,a
00B9  1F     inc  r7             Pufferzeiger +1
00BA  FE     mov  a,r6
00BB  96CC   jnz  CC
```

```
00BD  FB    mov  a,r3
00BE  F2CE  jb7  CE
00C0  2384  mov  a,#84   (132)
00C2  6F    add  a,r7           Pufferzeiger >123 ?
00C3  F6D4  jc   D4             ja -->
00C5  D5    sel  rb1
00C6  F8    mov  a,r0
00C7  C5    sel  rb0
00C8  03FB  add  a,#FB   (251)
00CA  F6D4  jc   D4
00CC  E400  jmp  700            zurueck zur Warteschleife
00CE  B820  mov  r0,#20  (32)
00D0  FB    mov  a,r3
00D1  A0    mov  @r0,a
00D2  04E9  jmp  0E9
****************************** Puffer voll
00D4  D5    sel  rb1
00D5  F8    mov  a,r0
00D6  C5    sel  rb0
00D7  E7    rl   a
00D8  E7    rl   a
00D9  E7    rl   a
00DA  E7    rl   a
00DB  5370  anl  a,#70   (112)
00DD  A8    mov  r0,a
00DE  FC    mov  a,r4
00DF  538F  anl  a,#8F   (143)
00E1  48B8  jnt1 B8
00E3  20    xch  a,@r0
00E4  A0    mov  @r0,a
00E5  FF    mov  a,r7           Linefeed
00E6  A8    mov  r0,a           an Puffer
00E7  B00A  mov  @r0,#0A (10)   anhaengen
00E9  2378  mov  a,#78   (120)
```

```
00EB  5C    anl  a,r4
00EC  4301  orl  a,#01   (1)
00EE  AC    mov  r4,a
00EF  BF22  mov  r7,#22  (34)  Pufferzeiger ruecksetzen
00F1  44AD  jmp  2AD           Puffer ausdrucken

00F3  00 00 00 00 00 00 00 00 00 00 00 00 00

****************************** Auswertung der Steuerzeichen
0100  FC    mov  a,r4
0101  37    cpl  a             Anfuehrungszeichenmodus ?
0102  3207  jbl  07            nein -->
0104  F5    sel  mbl
0105  04C3  jmp  8C3
0107  23F2  mov  a,#F2   (242) Breitschrift ein ?
0109  6A    add  a,r2
010A  C638  jz   38            ja -->
010C  07    dec  a             Breitschrift aus ?
010D  C638  jz   38            ja -->
010F  07    dec  a             Druckpositionsbestimmung ?
0110  C669  jz   69            ja -->
0112  07    dec  a             Textmodus ein ?
0113  C673  jz   73            ja -->
0115  07    dec  a             Reverse ein ?
0116  C682  jz   82            ja -->
0118  23F8  mov  a,#F8   (248)
011A  6A    add  a,r2          Bit-Image-Mode ?
011B  C65F  jz   5F            ja -->
011D  23E5  mov  a,#E5   (229)
011F  6A    add  a,r2          Druckstartposition (27) ?
0120  C69C  jz   9C            ja -->
0122  17    dec  a
0123  C6A1  jz   A1
0125  FA    mov  a,r2
```

```
0126  036F  add  a,#6F   (111) Grafikmodus ?
0128  C6A8  jz   A8             ja -->
012A  07    dec  a              Reverse aus ?
012B  C6B2  jz   B2             ja -->
012D  23F6  mov  a,#F6   (246)
012F  6A    add  a,r2           Linefeed ?
0130  C6BE  jz   BE             ja -->
0132  03FD  add  a,#FD   (253) CR ?
0134  C6BE  jz   BE             ja -->
0136  E400  jmp  700            zurueck zur Warteschleife
****************************** Steuerzeichen 14/15
0138  B810  mov  r0,#10  (16)
013A  B018  mov  @r0,#18 (24)
013C  2301  mov  a,#01   (1)
013E  4C    orl  a,r4
013F  AC    mov  r4,a
0140  B921  mov  r1,#21  (33)
0142  F1    mov  a,@r1
0143  325A  jb1  5A
0145  124C  jb0  4C
0147  D5    sel  rb1
0148  18    inc  r0
0149  C5    sel  rb0
014A  245B  jmp  15B
014C  B4F0  call 5F0            Die letzten drei
014E  B4F0  call 5F0            Bytes des Puffers
0150  B4F0  call 5F0            ausnullen
0152  F1    mov  a,@r1
0153  F257  jb7  57
0155  245B  jmp  15B
0157  D5    sel  rb1
0158  C8    dec  r0
0159  C5    sel  rb0
015A  CF    dec  r7
```

```
015B  B102  mov  @r1,#02 (2)
015D  04B5  jmp  0B5
****************************** Bit-Image-Mode ein
015F  B810  mov  r0,#10  (16)
0161  B010  mov  @r0,#10 (16)
0163  FC    mov  a,r4
0164  537C  anl  a,#7C   (124)
0166  AC    mov  r4,a
0167  2440  jmp  140
****************************** Steuerzeichen 16
0169  BD02  mov  r5,#02  (2)
016B  BB01  mov  r3,#01  (1)
016D  D5    sel  rb1
016E  18    inc  r0
016F  C5    sel  rb0
0170  1E    inc  r6
0171  04B5  jmp  0B5
****************************** Textmodus ein
0173  FC    mov  a,r4
0174  1278  jb0  78
0176  E400  jmp  700
0178  B814  mov  r0,#14  (20)
017A  2301  mov  a,#01   (1)
017C  40    orl  a,@r0
017D  53FD  anl  a,#FD   (253)
017F  A0    mov  @r0,a
0180  E400  jmp  700
****************************** Reverse ein
0182  FC    mov  a,r4
0183  1287  jb0  87
0185  2476  jmp  176
0187  5276  jb2  76
0189  5378  anl  a,#78   (120)
018B  4305  orl  a,#05   (5)
```

```
018D  AC    mov  r4,a
018E  B921  mov  r1,#21  (33)
0190  F1    mov  a,@r1
0191  CF    dec  r7
0192  5298  jb2  98
0194  D5    sel  rb1
0195  18    inc  r0
0196  C5    sel  rb0
0197  1F    inc  r7
0198  B104  mov  @r1,#04 (4)
019A  04B5  jmp  0B5
****************************** Druckstart 27
019C  BD01  mov  r5,#01  (1)
019E  1E    inc  r6
019F  E400  jmp  700
****************************** Steuerzeichen 28
01A1  FC    mov  a,r4
01A2  129F  jb0  9F
01A4  BD04  mov  r5,#04  (4)
01A6  246D  jmp  16D
****************************** Grafikmodus ein
01A8  B814  mov  r0,#14  (20)
01AA  2302  mov  a,#02   (2)
01AC  40    orl  a,@r0
01AD  53FE  anl  a,#FE   (254)
01AF  A0    mov  @r0,a
01B0  E400  jmp  700
****************************** Reverse aus
01B2  FC    mov  a,r4
01B3  52B7  jb2  B7
01B5  E400  jmp  700
01B7  5378  anl  a,#78   (120)
01B9  4301  orl  a,#01   (1)
01BB  AC    mov  r4,a
```

```
01BC  248E  jmp  18E
****************************** Linefeed oder CR
01BE  FC    mov  a,r4
01BF  53FD  anl  a,#FD   (253)
01C1  AC    mov  r4,a
01C2  BA0A  mov  r2,#0A  (10)
01C4  FB    mov  a,r3
01C5  C6CB  jz   CB
01C7  BB81  mov  r3,#81  (129)
01C9  04B5  jmp  0B5
01CB  BB80  mov  r3,#80  (128)
01CD  04B5  jmp  0B5

01CF  00
01D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0200  99F3  anl  pl,#F3  (243) Wagenruecklauf + Kopfmotor ein
0202  99FB  anl  pl,#FB  (251) Wagenruecklauf ein
0204  B4CD  call 5CD           Kopf in Grundstellung bringen
0206  FC    mov  a,r4
0207  538F  anl  a,#8F   (143)
0209  AC    mov  r4,a
020A  D46D  call 66D           Kopfbewegung und Taktrad testen
020C  F9    mov  a,r1
020D  03E7  add  a,#E7   (231)
020F  F618  jc   18
0211  B001  mov  @r0,#01 (1)
0213  030C  add  a,#0C   (12)
0215  F618  jc   18
0217  10    inc  @r0
0218  B810  mov  r0,#10  (16)
021A  27    clr  a
```

```
021B  62    mov  t,a
021C  55    strt t
021D  1621  jtf  21
021F  441D  jmp  21D
0221  E81D  djnz r0,1D
0223  65    stop tcnt
0224  8908  orl  p1,#08  (8)   Kopfmotor aus
0226  83    ret
*******************************
0227  FD    mov  a,r5
0228  1247  jb0  47
022A  3282  jb1  82
022C  FE    mov  a,r6
022D  07    dec  a
022E  9632  jnz  32
0230  246D  jmp  16D
0232  FA    mov  a,r2
0233  F243  jb7  43
0235  B4F0  call 5F0
0237  B4F0  call 5F0
0239  D5    sel  rb1
023A  C8    dec  r0
023B  C5    sel  rb0
023C  B4C9  call 5C9
023E  B917  mov  r1,#17  (23)
0240  A1    mov  @r1,a
0241  E400  jmp  700
0243  B4C9  call 5C9
0245  0482  jmp  082
0247  FE    mov  a,r6
0248  07    dec  a
0249  C672  jz   72
024B  07    dec  a
024C  C679  jz   79
```

```
024E  B917  mov  r1,#17  (23)
0250  F1    mov  a,@r1
0251  A9    mov  r1,a
0252  C65C  jz   5C
0254  07    dec  a
0255  9637  jnz  37
0257  FA    mov  a,r2
0258  0320  add  a,#20   (32)
025A  F637  jc   37
025C  B821  mov  r0,#21  (33)
025E  F0    mov  a,@r0
025F  12A7  jb0  A7
0261  B081  mov  @r0,#81 (129)
0263  3267  jb1  67
0265  B001  mov  @r0,#01 (1)
0267  FF    mov  a,r7
0268  29    xch  a,r1
0269  A1    mov  @r1,a
026A  1F    inc  r7
026B  B4C9  call 5C9
026D  B917  mov  r1,#17  (23)
026F  A1    mov  @r1,a
0270  04B5  jmp  0B5
0272  23F0  mov  a,#F0   (240)
0274  6A    add  a,r2
0275  963C  jnz  3C
0277  246D  jmp  16D
0279  FA    mov  a,r2
027A  5301  anl  a,#01   (1)
027C  B917  mov  r1,#17  (23)
027E  A1    mov  @r1,a
027F  1E    inc  r6
0280  E400  jmp  700
0282  FE    mov  a,r6
```

```
0283  07    dec  a
0284  9689  jnz  89
0286  FA    mov  a,r2
0287  447C  jmp  27C
0289  B917  mov  r1,#17  (23)
028B  F1    mov  a,@r1
028C  530F  anl  a,#0F   (15)
028E  E7    rl   a
028F  A8    mov  r0,a
0290  B904  mov  r1,#04  (4)
0292  68    add  a,r0
0293  E992  djnz r1,92
0295  A8    mov  r0,a
0296  FA    mov  a,r2
0297  530F  anl  a,#0F   (15)
0299  68    add  a,r0
029A  A8    mov  r0,a
029B  BE05  mov  r6,#05  (5)
029D  68    add  a,r0
029E  E6A1  jnc  A1
02A0  19    inc  r1
02A1  EE9D  djnz r6,9D
02A3  AA    mov  r2,a
02A4  F9    mov  a,r1
02A5  4452  jmp  252
02A7  B4F0  call 5F0
02A9  CF    dec  r7
02AA  CF    dec  r7
02AB  4467  jmp  267
****************************** Puffer ausdrucken
02AD  27    clr  a
02AE  B815  mov  r0,#15  (21)
02B0  B003  mov  @r0,#03 (3)
02B2  85    clr  f0
```

```
02B3  AB    mov  r3,a
02B4  BE18  mov  r6,#18  (24)
02B6  B81E  mov  r0,#1E  (30)  Zeichenzaehler ruecksetzen
02B8  A0    mov  @r0,a
02B9  18    inc  r0
02BA  A0    mov  @r0,a
02BB  BD80  mov  r5,#80  (128)
02BD  6401  jmp  301           Ausdruck starten
******************************
02BF  FD    mov  a,r5
02C0  F2C4  jb7  C4
02C2  8400  jmp  400           Zeichen drucken
02C4  65    stop tcnt
02C5  99FB  anl  p1,#FB  (251) Wagenrueckl. ein, das uebrige aus
02C7  FC    mov  a,r4
02C8  538F  anl  a,#8F   (143)
02CA  AC    mov  r4,a
02CB  27    clr  a
02CC  AD    mov  r5,a
02CD  A8    mov  r0,a
02CE  0A    in   a,p2          Kopf in Grundstellung ?
02CF  92D7  jb4  D7            ja -->
02D1  D410  call 610           Zeitueberwachung
02D3  E8CE  djnz r0,CE
02D5  0407  jmp  007           Dead Lock
02D7  D46D  call 66D           warten, bis Kopf die Grund-
02D9  F0    mov  a,@r0         stellung verlassen hat
02DA  C6E0  jz   E0
02DC  32E9  jb1  E9
02DE  8400  jmp  400           Zeichen drucken
02E0  F9    mov  a,r1
02E1  03F3  add  a,#F3   (243)
02E3  F6DE  jc   DE
02E5  D4A1  call 6A1           Takt abwarten
```

```
02E7  8400  jmp  400           Zeichen drucken
02E9  F9    mov  a,r1
02EA  03E7  add  a,#E7   (231)
02EC  F6DE  jc   DE
02EE  44E5  jmp  2E5

02F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

****************************** Puffer ausdrucken
0300  AC    mov  r4,a
0301  FF    mov  a,r7
0302  A8    mov  r0,a
0303  F0    mov  a,@r0        Byte holen
0304  1F    inc  r7           Pufferzeiger +1
0305  AA    mov  r2,a         Byte -> R2
0306  03F0  add  a,#F0   (240) Druckposition bestimmen ?
0308  961A  jnz  1A          nein -->
030A  85    clr  f0
030B  95    cpl  f0
030C  FF    mov  a,r7
030D  A8    mov  r0,a
030E  F0    mov  a,@r0
030F  1F    inc  r7
0310  B917  mov  r1,#17  (23)
0312  A1    mov  @r1,a
0313  C9    dec  r1
0314  18    inc  r0
0315  F0    mov  a,@r0
0316  A1    mov  @r1,a
0317  1F    inc  r7
0318  8472  jmp  472         Startposition berechnen
******************************
031A  17    dec  a           Textmodus ein ?
031B  9623  jnz  23          nein -->
```

```
031D  FC     mov  a,r4
031E  537C   anl  a,#7C   (124)
0320  17     dec  a
0321  6400   jmp  300
*******************************
0323  17     dec  a              Reverse ein ?
0324  962D   jnz  2D             nein -->
0326  FC     mov  a,r4
0327  537C   anl  a,#7C   (124)
0329  4381   orl  a,#81   (129)
032B  6400   jmp  300
*******************************
032D  03FC   add  a,#FC   (252) =22 ?
032F  9636   jnz  36             nein -->
0331  FC     mov  a,r4
0332  4304   orl  a,#04   (4)
0334  6400   jmp  300
*******************************
0336  0380   add  a,#80   (128) =150 ?
0338  9643   jnz  43             nein
033A  FC     mov  a,r4
033B  123F   jb0  3F
033D  44BF   jmp  2BF
033F  53F9   anl  a,#F9   (249)
0341  6400   jmp  300
*******************************
0343  038A   add  a,#8A   (138) =12 ?
0345  964D   jnz  4D             nein -->
0347  85     clr  f0
0348  FC     mov  a,r4
0349  537C   anl  a,#7C   (124) Anfuehrungsflag aus
034B  6400   jmp  300
*******************************
034D  03EE   add  a,#EE   (238) =30 ?
```

```
034F  9660  jnz  60            nein -->
0351  FF    mov  a,r7
0352  A8    mov  r0,a
0353  F0    mov  a,@r0
0354  AB    mov  r3,a
0355  1F    inc  r7
0356  18    inc  r0
0357  F0    mov  a,@r0
0358  AA    mov  r2,a
0359  1F    inc  r7
035A  FC    mov  a,r4
035B  4302  orl  a,#02   (2)
035D  AC    mov  r4,a
035E  44BF  jmp  2BF
******************************
0360  0310  add  a,#10   (16)  Breitschrift ein ?
0362  9666  jnz  66            nein -->
0364  84FE  jmp  4FE
******************************
0366  FC    mov  a,r4
0367  126B  jb0  6B
0369  44BF  jmp  2BF
****************************** Zeichen zum Druck aufbereiten
                               und spaltenweise ab Adresse
                               24 hinterlegen
036B  B81F  mov  r0,#1F  (31)
036D  F0    mov  a,@r0
036E  C684  jz   84
0370  C8    dec  r0
0371  FC    mov  a,r4
0372  F27B  jb7  7B
0374  F0    mov  a,@r0
0375  0325  add  a,#25   (37)
0377  F682  jc   82
```

```
0379  6484  jmp  384
037B  F0    mov  a,@r0
037C  032B  add  a,#2B   (43)
037E  F682  jc   82
0380  6484  jmp  384
0382  849A  jmp  49A           Linefeed
0384  FA    mov  a,r2
0385  0340  add  a,#40   (64)  Zeichen >191 ?
0387  E696  jnc  96            nein -->
0389  D461  call 661           Posit. im Chargen. berechnen
038B  F8    mov  a,r0
038C  F5    sel  mb1
038D  74D8  call BD8           Matrixspalte holen
038F  E5    sel  mb0
0390  18    inc  r0
0391  19    inc  r1
0392  EB8B  djnz r3,8B         wiederholen, bis 6 Spalten fertig
0394  64D6  jmp  3D6           ausdrucken
0396  0320  add  a,#20   (32)  Zeichen >159 ?
0398  E6A7  jnc  A7            nein -->
039A  D461  call 661           Pos. im Chargen. berechnen
039C  F8    mov  a,r0
039D  F5    sel  mb1
039E  54C0  call AC0           Matrixspalte holen
03A0  E5    sel  mb0
03A1  18    inc  r0
03A2  19    inc  r1
03A3  EB9C  djnz r3,9C         wiederholen, bis 6 Spalten fertig
03A5  64D6  jmp  3D6           ausdrucken
03A7  0340  add  a,#40   (64)  Zeichen >95 ?
03A9  E6B8  jnc  B8            nein -->
03AB  D461  call 661           Pos. im Chargen. berechnen
03AD  F8    mov  a,r0
03AE  F5    sel  mb1
```

```
03AF  B4C0  call DC0          Matrixspalte holen
03B1  E5    sel  mb0
03B2  18    inc  r0
03B3  19    inc  r1
03B4  EBAD  djnz r3,AD        wiederholen, bis 6 Spalten fertig
03B6  64D6  jmp  3D6          ausdrucken
03B8  0320  add  a,#20   (32) Zeichen >63 ?
03BA  E6C9  jnc  C9           nein -->
03BC  D461  call 661          Pos. im Chargen. berechnen
03BE  F8    mov  a,r0
03BF  F5    sel  mb1
03C0  34C0  call 9C0          Matrixspalte holen
03C2  E5    sel  mb0
03C3  18    inc  r0
03C4  19    inc  r1
03C5  EBBE  djnz r3,BE        wiederholen, bis 6 Spalten fertig
03C7  64D6  jmp  3D6          ausdrucken
03C9  0320  add  a,#20   (32)
03CB  D461  call 661          Pos. im Chargen. berechnen
03CD  F8    mov  a,r0
03CE  F5    sel  mb1
03CF  14C0  call 8C0          Matrixspalte holen
03D1  E5    sel  mb0
03D2  18    inc  r0
03D3  19    inc  r1
03D4  EBCD  djnz r3,CD        wiederholen, bis 6 Spalten fertig
03D6  FE    mov  a,r6
03D7  A9    mov  r1,a
03D8  F1    mov  a,@r1
03D9  AA    mov  r2,a
03DA  FC    mov  a,r4
03DB  52DF  jb2  DF
03DD  44BF  jmp  2BF          ausdrucken
03DF  FA    mov  a,r2
```

```
03E0  37     cpl  a
03E1  AA     mov  r2,a
03E2  44BF   jmp  2BF            ausdrucken

03E4  00 00 00 00 00 00 00 00 00 00 00 00
03F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

****************************** Zeichen drucken
0400  D4E8   call 6E8            Zeilenende erreicht ?
0402  F690   jc   90             ja -->
0404  FA     mov  a,r2
0405  537F   anl  a,#7F   (127)
0407  A9     mov  r1,a
0408  B815   mov  r0,#15  (21)
040A  10     inc  @r0
040B  F0     mov  a,@r0
040C  D304   xrl  a,#04   (4)
040E  C61B   jz   1B
0410  B8F0   mov  r0,#F0  (240)
0412  42     mov  a,t
0413  D302   xrl  a,#02   (2)
0415  C61E   jz   1E
0417  E812   djnz r0,12
0419  0407   jmp  007            Dead Lock
041B  A0     mov  @r0,a
041C  8420   jmp  420
041E  461E   jntl 1E             Taktluecke abwarten
0420  65     stop tcnt
0421  23C0   mov  a,#C0   (192)
0423  62     mov  t,a            Taktueberwachung
0424  55     strt t              einschalten
0425  25     en   tcnti
0426  F9     mov  a,r1           naechsten Punkt der
0427  77     rr   a              Matrixspalte holen
```

```
0428  A9     mov  rl,a
0429  37     cpl  a
042A  562A   jtl  2A             Takt abwarten
042C  F23D   jb7  3D             Punkt drucken ? nein -->
042E  99FE   anl  pl,#FE  (254)  Hammer feuern
0430  27     clr  a
0431  B842   mov  r0,#42  (66)
0433  F9     mov  a,rl
0434  D239   jb6  39
0436  00     nop
0437  B84E   mov  r0,#4E  (78)
0439  E839   djnz r0,39
043B  8901   orl  pl,#01  (1)    Hammer aus
043D  1D     inc  r5             Punktzaehler +1
043E  463E   jntl 3E             Taktluecke abwarten
0440  65     stop tcnt           Ueberwachung aus
0441  35     dis  tcnti
0442  23F9   mov  a,#F9   (249) 7 Punkte gedruckt ?
0444  6D     add  a,r5
0445  E621   jnc  21             nein --> naechster Punkt
0447  C651   jz   51             ja -->
0449  03F9   add  a,#F9   (249) Breitschrift fertig ?
044B  9621   jnz  21             nein -->
044D  F4CD   call 7CD            Spaltenzaehl.+1 & Taktzaehl. strt
044F  8464   jmp  464
0451  F4CD   call 7CD            -------- " ------------ " ------
0453  B672   jf0  72
0455  FC     mov  a,r4
0456  1262   jb0  62
0458  37     cpl  a
0459  325D   jbl  5D
045B  EB00   djnz r3,00
045D  4302   orl  a,#02   (2)
045F  37     cpl  a
```

```
0460  6400  jmp  300
0462  F26E  jb7  6E
0464  1E    inc  r6
0465  FE    mov  a,r6
0466  03E2  add  a,#E2   (226)
0468  9670  jnz  70
046A  BE18  mov  r6,#18  (24)
046C  6401  jmp  301           naechstes Byte holen
046E  BD07  mov  r5,#07  (7)
0470  64D6  jmp  3D6
****************************** Druckstartposition berechnen
0472  B81F  mov  r0,#1F  (31)
0474  B917  mov  r1,#17  (23)
0476  F1    mov  a,@r1
0477  AA    mov  r2,a
0478  F0    mov  a,@r0
0479  37    cpl  a
047A  6A    add  a,r2
047B  F68C  jc   8C
047D  37    cpl  a
047E  9689  jnz  89
0480  C8    dec  r0
0481  C9    dec  r1
0482  F1    mov  a,@r1
0483  AA    mov  r2,a
0484  F0    mov  a,@r0
0485  37    cpl  a
0486  6A    add  a,r2
0487  F68C  jc   8C
0489  85    clr  f0
048A  6401  jmp  301
048C  27    clr  a
048D  AA    mov  r2,a
048E  44BF  jmp  2BF
```

```
*****************************
0490  FC    mov  a,r4
0491  37    cpl  a
0492  329A  jbl  9A
0494  CF    dec  r7
0495  CF    dec  r7
0496  FF    mov  a,r7
0497  A8    mov  r0,a
0498  FB    mov  a,r3
0499  A0    mov  @r0,a
049A  CF    dec  r7
049B  65    stop tcnt
049C  B4FC  call 5FC           Linefeed
049E  B820  mov  r0,#20  (32)
04A0  F0    mov  a,@r0
04A1  37    cpl  a
04A2  F2AA  jb7  AA
04A4  FC    mov  a,r4
04A5  538F  anl  a,#8F   (143)
04A7  AC    mov  r4,a
04A8  44AD  jmp  2AD           Puffer ausdrucken
04AA  B822  mov  r0,#22  (34)
04AC  FC    mov  a,r4
04AD  37    cpl  a
04AE  52B3  jb2  B3
04B0  B012  mov  @r0,#12 (18)
04B2  18    inc  r0
04B3  B008  mov  @r0,#08 (8)
04B5  12BD  jb0  BD
04B7  B00F  mov  @r0,#0F (15)
04B9  F2BD  jb7  BD
04BB  B00E  mov  @r0,#0E (14)
04BD  FF    mov  a,r7
04BE  A9    mov  r1,a
```

```
04BF  F1    mov  a,@r1
04C0  AA    mov  r2,a
04C1  18    inc  r0
04C2  F9    mov  a,r1
04C3  0381  add  a,#81   (129)
04C5  F6CC  jc   CC
04C7  19    inc  r1
04C8  FA    mov  a,r2
04C9  A0    mov  @r0,a
04CA  84BF  jmp  4BF
04CC  C8    dec  r0
04CD  F0    mov  a,@r0
04CE  A9    mov  r1,a
04CF  27    clr  a
04D0  A0    mov  @r0,a
04D1  F9    mov  a,r1
04D2  03F6  add  a,#F6   (246)
04D4  96CC  jnz  CC
04D6  F8    mov  a,r0
04D7  AF    mov  r7,a
04D8  27    clr  a
04D9  A0    mov  @r0,a
04DA  18    inc  r0
04DB  F8    mov  a,r0
04DC  0380  add  a,#80   (128)
04DE  E6D8  jnc  D8
04E0  B4C9  call 5C9
04E2  B820  mov  r0,#20  (32)
04E4  F0    mov  a,@r0
04E5  77    rr   a
04E6  77    rr   a
04E7  77    rr   a
04E8  77    rr   a
04E9  5307  anl  a,#07   (7)
```

```
04EB  D5    sel  rb1
04EC  A8    mov  r0,a
04ED  C5    sel  rb0
04EE  D305  xrl  a,#05   (5)
04F0  96F4  jnz  F4
04F2  04E5  jmp  0E5
04F4  F0    mov  a,@r0
04F5  4320  orl  a,#20   (32)
04F7  A44A  jmp  54A
04F9  00    nop
04FA  00    nop
04FB  00    nop
04FC  00    nop
04FD  00    nop
****************************** Breitschrift
04FE  B820  mov  r0,#20  (32)
0500  F0    mov  a,@r0
0501  F251  jb7  51
0503  8908  orl  p1,#08  (8)
0505  FD    mov  a,r5
0506  D380  xrl  a,#80    (128)
0508  C60F  jz   0F
050A  99FB  anl  p1,#FB  (251)
050C  65    stop tcnt
050D  B4CD  call 5CD
050F  B821  mov  r0,#21  (33)
0511  B000  mov  @r0,#00 (0)
0513  18    inc  r0
0514  B918  mov  r1,#18  (24)
0516  B100  mov  @r1,#00 (0)
0518  FC    mov  a,r4
0519  37    cpl  a
051A  521F  jb2  1F
051C  B012  mov  @r0,#12 (18)
```

```
051E  18    inc  r0
051F  B008  mov  @r0,#08 (8)
0521  1229  jb0  29
0523  B00F  mov  @r0,#0F (15)
0525  F229  jb7  29
0527  B00E  mov  @r0,#0E (14)
0529  18    inc  r0
052A  B010  mov  @r0,#10 (16)
052C  18    inc  r0
052D  B91F  mov  r1,#1F  (31)
052F  F1    mov  a,@r1
0530  A0    mov  @r0,a
0531  C9    dec  r1
0532  18    inc  r0
0533  F1    mov  a,@r1
0534  A0    mov  @r0,a
0535  18    inc  r0
0536  F8    mov  a,r0
0537  AF    mov  r7,a
0538  B4C9  call 5C9
053A  F8    mov  a,r0
053B  37    cpl  a
053C  0381  add  a,#81   (129)
053E  A9    mov  r1,a
053F  27    clr  a
0540  A0    mov  @r0,a
0541  18    inc  r0
0542  E940  djnz r1,40
0544  B820  mov  r0,#20  (32)
0546  F0    mov  a,@r0
0547  5302  anl  a,#02   (2)
0549  4C    orl  a,r4
054A  532F  anl  a,#2F   (47)
054C  AC    mov  r4,a
```

```
054D  BB01  mov  r3,#01  (1)
054F  E400  jmp  700
0551  B4FC  call 5FC
0553  B87F  mov  r0,#7F  (127)
0555  F0    mov  a,@r0
0556  C662  jz   62
0558  0A    in   a,p2
0559  37    cpl  a
055A  B264  jb5  64
055C  FC    mov  a,r4
055D  5371  anl  a,#71   (113)
055F  4301  orl  a,#01   (1)
0561  AC    mov  r4,a
0562  0444  jmp  044
0564  B87F  mov  r0,#7F  (127)
0566  F0    mov  a,@r0
0567  966F  jnz  6F
0569  B020  mov  @r0,#20 (32)
056B  23F7  mov  a,#F7   (247)
056D  5C    anl  a,r4
056E  AC    mov  r4,a
056F  B922  mov  r1,#22  (34)
0571  B10F  mov  @r1,#0F (15)
0573  19    inc  r1
0574  FC    mov  a,r4
0575  BC00  mov  r4,#00  (0)
0577  37    cpl  a
0578  727C  jb3  7C
057A  BC01  mov  r4,#01  (1)
057C  BB50  mov  r3,#50  (80)
057E  F0    mov  a,@r0
057F  AA    mov  r2,a
0580  03BF  add  a,#BF   (191)
0582  F6AD  jc   AD
```

```
0584  0341  add  a,#41   (65)
0586  A1    mov  @r1,a
0587  10    inc  @r0
0588  19    inc  r1
0589  F0    mov  a,@r0
058A  C6BF  jz   BF
058C  0380  add  a,#80   (128)
058E  9692  jnz  92
0590  B0A0  mov  @r0,#A0 (160)
0592  EB7E  djnz r3,7E
0594  B10A  mov  @r1,#0A (10)
0596  BB81  mov  r3,#81  (129)
0598  BF22  mov  r7,#22  (34)
059A  FC    mov  a,r4
059B  C6A1  jz   A1
059D  BC09  mov  r4,#09  (9)
059F  A4A3  jmp  5A3
05A1  BC01  mov  r4,#01  (1)
05A3  B820  mov  r0,#20  (32)
05A5  B081  mov  @r0,#81 (129)
05A7  B810  mov  r0,#10  (16)
05A9  B018  mov  @r0,#18 (24)
05AB  44AD  jmp  2AD
05AD  FC    mov  a,r4
05AE  C6B6  jz   B6
05B0  F5    sel  mb1
05B1  34D8  call 9D8
05B3  E5    sel  mb0
05B4  A4BA  jmp  5BA
05B6  F5    sel  mb1
05B7  9400  call C00
05B9  E5    sel  mb0
05BA  FA    mov  a,r2
05BB  B87F  mov  r0,#7F  (127)
```

```
05BD  A486  jmp  586
05BF  FC    mov  a,r4
05C0  BC01  mov  r4,#01  (1)
05C2  C6C5  jz   C5
05C4  CC    dec  r4
05C5  B020  mov  @r0,#20 (32)
05C7  A492  jmp  592
05C9  27    clr  a
05CA  AD    mov  r5,a
05CB  AE    mov  r6,a
05CC  83    ret
*******************************
05CD  FC    mov  a,r4
05CE  B2E6  jb5  E6
05D0  D425  call 625          Timer aufziehen
05D2  0A    in   a,p2
05D3  37    cpl  a            Kopf in Grundstellung ?
05D4  92E7  jb4  E7           nein -->
05D6  D425  call 625
05D8  4320  orl  a,#20   (32)
05DA  AC    mov  r4,a
05DB  FC    mov  a,r4
05DC  37    cpl  a
05DD  B2E6  jb5  E6
05DF  D430  call 630
05E1  0A    in   a,p2
05E2  92DB  jb4  DB
05E4  A4D2  jmp  5D2
05E6  83    ret
05E7  D430  call 630          Timer testen
05E9  A4D2  jmp  5D2
******************************* Linefeed
05EB  F5    sel  mb1
05EC  C400  jmp  E00
```

```
05EE  E5    sel  mb0
05EF  83    ret
*****************************
05F0  CF    dec  r7             letztes Byte
05F1  FF    mov  a,r7           des Puffers
05F2  A8    mov  r0,a           ausnullen
05F3  27    clr  a
05F4  A0    mov  @r0,a
05F5  83    ret
*****************************
05F6  D410  call 610
05F8  D410  call 610
05FA  C410  jmp  610
***************************** Linefeed
05FC  8908  orl  p1,#08  (8)   Kopfmotor aus
05FE  B820  mov  r0,#20  (32)
0600  F0    mov  a,@r0
0601  37    cpl  a
0602  F206  jb7  06
0604  120D  jb0  0D
0606  99FB  anl  p1,#FB  (251) Wagenruecklauf ein
0608  B4EB  call 5EB           Linefeed
060A  B4CD  call 5CD           Warten bis Kopf in Grundstellung
060C  83    ret
060D  B4EB  call 5EB           Linefeed
060F  83    ret
***************************** Zeitueberwachung
0610  234F  mov  a,#4F   (79)
0612  62    mov  t,a
0613  55    strt t
0614  1618  jtf  18
0616  C414  jmp  614
0618  161C  jtf  1C
061A  C418  jmp  618
```

```
061C  1620  jtf  20
061E  C41C  jmp  61C
0620  FC    mov  a,r4
0621  B233  jb5  33
0623  65    stop tcnt
0624  83    ret
*******************************
0625  27    clr  a
0626  D5    sel  rb1
0627  A9    mov  r1,a
0628  C5    sel  rb0
0629  62    mov  t,a
062A  55    strt t
062B  FC    mov  a,r4
062C  53AF  anl  a,#AF   (175)
062E  AC    mov  r4,a
062F  83    ret
*******************************
0630  1635  jtf  35
0632  83    ret
0633  D5    sel  rb1
0634  19    inc  r1
0635  D5    sel  rb1
0636  19    inc  r1
0637  F9    mov  a,r1
0638  C5    sel  rb0
0639  9648  jnz  48
063B  FC    mov  a,r4
063C  37    cpl  a
063D  9243  jb4  43
063F  D25C  jb6  5C
0641  0407  jmp  007
0643  37    cpl  a
0644  4310  orl  a,#10   (16)
```

```
0646  AC     mov  r4,a
0647  83     ret
0648  03F3   add  a,#F3   (243)
064A  FC     mov  a,r4
064B  37     cpl  a
064C  B232   jb5  32
064E  E632   jnc  32
0650  FC     mov  a,r4
0651  538F   anl  a,#8F   (143)
0653  AC     mov  r4,a
0654  8904   orl  p1,#04  (4)   Wagenruecklauf aus
0656  27     clr  a
0657  D5     sel  rb1
0658  A9     mov  r1,a
0659  C5     sel  rb0
065A  65     stop tcnt
065B  83     ret
065C  37     cpl  a
065D  4340   orl  a,#40   (64)
065F  AC     mov  r4,a
0660  83     ret
*******************************
0661  AA     mov  r2,a          Position
0662  97     clr  c             im
0663  F7     rlc  a             Character-
0664  AA     mov  r2,a          generator
0665  F7     rlc  a             berechnen
0666  6A     add  a,r2
0667  A8     mov  r0,a
0668  B918   mov  r1,#18  (24)
066A  BB06   mov  r3,#06  (6)
066C  83     ret
*******************************
066D  99F3   anl  p1,#F3  (243) Kopfmotor un Wagenruecklauf ein
```

```
066F  27    clr  a
0670  62    mov  t,a
0671  B80D  mov  r0,#0D  (13)
0673  55    strt t
0674  1678  jtf  78
0676  C474  jmp  674
0678  E874  djnz r0,74
067A  65    stop tcnt
067B  23C0  mov  a,#C0  (192)
067D  62    mov  t,a
067E  55    strt t
067F  25    en   tcnti        Dead Lock, wenn kein Takt kommt
0680  4684  jntl 84           zwei Takte
0682  5682  jtl  82           des
0684  4684  jntl 84           Taktrades
0686  5686  jtl  86           abwarten
0688  65    stop tcnt
0689  35    dis  tcnti
068A  8904  orl  p1,#04  (4)  Wagenruecklauf aus
068C  27    clr  a
068D  62    mov  t,a
068E  55    strt t
068F  B920  mov  r1,#20  (32)
0691  0A    in   a,p2         hat der Kopf
0692  92BD  jb4  BD           die Grundstellung
0694  0A    in   a,p2         verlassen ?
0695  92BD  jb4  BD           nein -->
0697  B900  mov  r1,#00  (0)
0699  65    stop tcnt
069A  169C  jtf  9C
069C  23C0  mov  a,#C0  (192)
069E  62    mov  t,a
069F  55    strt t
06A0  25    en   tcnti
```

```
****************************** Takt abwarten
06A1  56A1  jtl  A1
06A3  46A3  jntl A3
06A5  65    stop tcnt
06A6  35    dis  tcnti
06A7  19    inc  r1
06A8  27    clr  a
06A9  62    mov  t,a
06AA  B81E  mov  r0,#1E  (30)
06AC  A0    mov  @r0,a
06AD  18    inc  r0
06AE  A0    mov  @r0,a
06AF  55    strt t
06B0  46A3  jntl A3
06B2  42    mov  a,t
06B3  37    cpl  a
06B4  72B0  jb3  B0
06B6  12B0  jb0  B0
06B8  65    stop tcnt
06B9  B811  mov  r0,#11  (17)
06BB  97    clr  c
06BC  83    ret
******************************
06BD  16C1  jtf  C1            Timer abgelaufen ? ja -->
06BF  C491  jmp  691
06C1  E991  djnz r1,91         wiederholen, bis R1=0, dann
06C3  0407  jmp  007           Dead Lock
****************************** Linefeed
06C5  8908  orl  p1,#08  (8)   Kopfmotor aus
06C7  B810  mov  r0,#10  (16)
06C9  B018  mov  @r0,#18 (24)
06CB  0A    in   a,p2
06CC  92D2  jb4  D2
06CE  99FB  anl  p1,#FB  (251) Wagenruecklauf ein
```

```
06D0  B4CD  call 5CD
06D2  09    in   a,p1          Linefeed-Taste gedrueckt ?
06D3  D2DD  jb6  DD            nein -->
06D5  B4EB  call 5EB
06D7  B4F6  call 5F6
06D9  B4F6  call 5F6
06DB  C4D2  jmp  6D2
06DD  E400  jmp  700           Wait-Loop
*******************************
06DF  997F  anl  p1,#7F  (127) DATA-Flip-Flop freigeben
06E1  90    movx @r0,a         DATA low-setzen
06E2  83    ret
*******************************
06E3  8980  orl  p1,#80  (128) DATA high-setzen
06E5  997F  anl  p1,#7F  (127)
06E7  83    ret
******************************* Zeilenende pruefen
06E8  97    clr  c
06E9  B81F  mov  r0,#1F  (31)
06EB  F0    mov  a,@r0
06EC  C6F6  jz   F6
06EE  03FE  add  a,#FE   (254)
06F0  F6F6  jc   F6
06F2  C8    dec  r0
06F3  F0    mov  a,@r0
06F4  0320  add  a,#20   (32)
06F6  83    ret

06F7  00 00 00 00 00 00 00 00 00

******************************* Wait-Loop
0700  7613  jf1  13            als LISTENER adressiert ? ja -->
0702  0A    in   a,p2          ATN ?
0703  F21A  jb7  1A            ja -->
```

```
0705  D4E3  call 6E3           DATA high-setzen
0707  0A    in   a,p2          ATN ?
0708  F21A  jb7  1A            ja -->
070A  B20E  jb5  0E            Testmode ?
070C  A464  jmp  564           ja -->
070E  09    in   a,p1          Linefeed-Taste gedrueckt ?
070F  D200  jb6  00            nein -->
0711  C4C5  jmp  6C5           Zeilenvorschub ausloesen
0713  0A    in   a,p2          ATN ?
0714  F21A  jb7  1A            ja -->
0716  364C  jt0  4C            CLK high ? ja -->
0718  E400  jmp  700           warten
****************************** Einsprung bei ATN
071A  D4DF  call 6DF           DATA low-setzen
071C  B8DC  mov  r0,#DC  (220) Zaehler setzen
071E  D4DF  call 6DF           DATA low-setzen
0720  E81E  djnz r0,1E         warten
0722  09    in   a,p1          CLK low ?
0723  37    cpl  a
0724  9200  jb4  00            nein -->
0726  B814  mov  r0,#14  (20)
0728  23CF  mov  a,#CF  (207)
072A  50    anl  a,@r0
072B  A0    mov  @r0,a
072C  B814  mov  r0,#14  (20)  Zaehler f. warten auf CLK setzen
072E  B908  mov  r1,#08  (8)   Bitzaehler setzen
0730  8980  orl  p1,#80  (128) DATA f. Eingabe freigeben
0732  BA01  mov  r2,#01  (1)
0734  863C  jni  3C            DATA high ? ja -->
0736  0A    in   a,p2          ATN noch low ?
0737  37    cpl  a
0738  F200  jb7  00            nein, Fehlalarm -->
073A  E434  jmp  734
073C  2643  jnt0 43            CLK noch low ? ja -->
```

```
073E  0A    in   a,p2          ATN noch low ?
073F  F248  jb7  48            ja -->
0741  E400  jmp  700           Fehlalarm -->
0743  0A    in   a,p2          ATN noch low ?
0744  F27D  jb7  7D            ja -->
0746  E400  jmp  700           Fehlalarm -->
0748  E83C  djnz r0,3C         warten auf CLK=low
074A  E469  jmp  769
****************************** auf Byte warten
074C  B814  mov  r0,#14  (20)
074E  B908  mov  r1,#08  (8)   Bitzaehler setzen
0750  8980  orl  p1,#80  (128) DATA high-setzen
0752  BA14  mov  r2,#14  (20)
0754  0A    in   a,p2          ATN low ?
0755  F21A  jb7  1A            ja -->
0757  865B  jni  5B            DATA high ? ja -->
0759  E452  jmp  752           sonst warten
075B  2664  jnt0 64            CLK noch low ? ja -->
075D  0A    in   a,p2          ATN low ?
075E  F21A  jb7  1A            ja -->
0760  E85B  djnz r0,5B
0762  E469  jmp  769
0764  0A    in   a,p2          ATN low ?
0765  F21A  jb7  1A            ja -->
0767  E47D  jmp  77D
0769  D4DF  call 6DF           DATA low-setzen
076B  B821  mov  r0,#21  (33)
076D  E86D  djnz r0,6D         warten
076F  8980  orl  p1,#80  (128) DATA high-setzen
0771  0A    in   a,p2
0772  EA78  djnz r2,78
0774  F27A  jb7  7A
0776  E400  jmp  700
0778  F21A  jb7  1A
```

```
077A  1A     inc  r2
077B  3671   jt0  71
****************************** Byte holen
077D  267D   jnt0 7D            warten auf CLK=high
077F  97     clr  c             Bits
0780  8683   jni  83            nach
0782  A7     cpl  c             R0
0783  F8     mov  a,r0          schieben
0784  67     rrc  a
0785  A8     mov  r0,a
0786  E971   djnz r1,71         wiederholen, bis Bitzaehler=0
0788  0A     in   a,p2          ATN low ?
0789  EA8F   djnz r2,8F
078B  F291   jb7  91            ja -->
078D  E400   jmp  700           Wait-Loop
078F  F21A   jb7  1A            ATN low ? ja -->
0791  1A     inc  r2
0792  3688   jt0  88            warten, bis CLK=low
0794  F8     mov  a,r0          empfangenes Byte
0795  37     cpl  a             invertieren und
0796  AA     mov  r2,a          nach R2 bringen
0797  0A     in   a,p2          ATN low ?
0798  F2A7   jb7  A7            ja -->
079A  769E   jf1  9E            als LISTENER adressiert ? ja -->
079C  E400   jmp  700           Wait-Loop
079E  D4DF   call 6DF           DATA low-setzen (quittieren)
07A0  B814   mov  r0,#14  (20)
07A2  F0     mov  a,@r0
07A3  F200   jb7  00
07A5  0475   jmp  075
07A7  237F   mov  a,#7F   (127) waehrend ATN
07A9  5A     anl  a,r2          empfangene Adresse
07AA  AA     mov  r2,a          nach R2 bringen
07AB  B814   mov  r0,#14  (20)
```

```
07AD  23C1  mov  a,#C1   (193) UNLISTEN ?
07AF  6A    add  a,r2
07B0  96B7  jnz  B7             nein -->
07B2  A5    clr  f1             LISTEN-Flag ruecksetzen
07B3  B000  mov  @r0,#00 (0)
07B5  E4BA  jmp  7BA
07B7  F5    sel  mb1
07B8  C478  jmp  E78            Adresse pruefen
07BA  D4DF  call 6DF            DATA low-setzen
07BC  36C3  jt0  C3             CLK high ? ja -->
07BE  0A    in   a,p2           ATN low ?
07BF  F2BC  jb7  BC             ja -->
07C1  E400  jmp  700            Wait-Loop
07C3  0A    in   a,p2           ATN low ?
07C4  F22C  jb7  2C             ja -->
07C6  E4C1  jmp  7C1
****************************** warten bis Adressierungs-
                                zyklus beendet
07C8  0A    in   a,p2           ATN low ?
07C9  F2C8  jb7  C8             ja -->
07CB  E400  jmp  700            Wait-Loop
******************************
07CD  45    strt cnt            Taktzaehler
07CE  27    clr  a              starten
07CF  62    mov  t,a            &
07D0  AD    mov  r5,a           Zeichen-
07D1  B81E  mov  r0,#1E  (30)   zaehler
07D3  10    inc  @r0            +1
07D4  F0    mov  a,@r0
07D5  18    inc  r0
07D6  96D9  jnz  D9
07D8  10    inc  @r0
07D9  83    ret
```

```
07DA  00 00 00 00 00 00
07E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
07F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
0800    00 00 00 00 00 00
```

```
0806    00 00 5F 00 00 00
          *
          *
          *
          *
          *

          *
```

```
080C    00 07 00 07 00 00
         * *
         * *
         * *
```

```
0812    14 7F 14 7F 14 00
         * *
         * *
        *****
         * *
        *****
         * *
         * *
```

```
0818    24 2A 7F 2A 12 00
          *
         ****
        * *
         ***
          * *
        ****
          *
```

```
081E    63 13 08 64 63 00
        **  *
        **  *
            *
           *
          *
        *  **
        *  **
```

```
0824    30 4E 59 26 50 00
```

```
082A    00 04 02 01 00 00
```

```
0830    00 1C 22 41 00 00
```

```
0836    00 41 22 1C 00 00
```

```
083C    2A 1C 7F 1C 2A 00
```

```
0842    08 08 3E 08 08 00
```

```
0848    00 40 30 00 00 00
```



```
084E    08 08 08 08 08 08
```



```
0854    00 60 60 00 00 00
```



```
085A    60 10 08 04 03 00
```

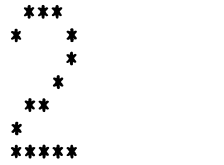


```
0860    3E 51 49 45 3E 00
```

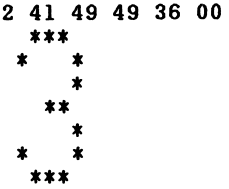


```
0866    00 42 7F 40 00 00
```

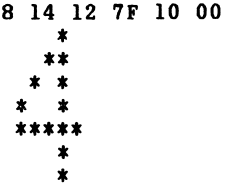
086C 62 51 51 49 46 00



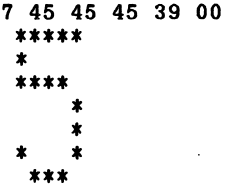
0872 22 41 49 49 36 00



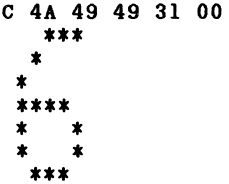
0878 18 14 12 7F 10 00



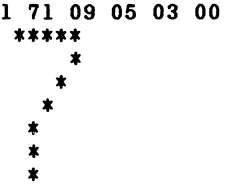
087E 27 45 45 45 39 00



0884 3C 4A 49 49 31 00

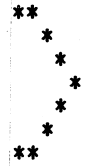


088A 01 71 09 05 03 00

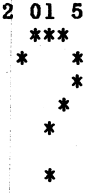


```
0890    36 49 49 49 36 00
          ***
         *   *
         *   *
          ***
         *   *
         *   *
          ***

0896    46 49 49 29 1E 00
          ***
         *   *
         *   *
          ****
             *
            *
         ***

089C    00 00 12 00 00 00
            *


            *


08A2    00 40 32 00 00 00
            *


            *
            *
           *

08A8    08 14 22 41 41 00
             **
            *
           *
          *
           *
            *
             **

08AE    14 14 14 14 14 00

          *****

          *****
```

08B4 41 41 22 14 08 00



08BA 02 01 51 09 06 00



```
*******************************
08C0  A3    movp a,@a          Matrixspalte
08C1  A1    mov  @r1,a         holen
08C2  83    ret
*******************************
08C3  FA    mov  a,r2
08C4  F2D7  jb7  D7
08C6  03E0  add  a,#E0   (224) Zeichen>31 ?
08C8  F6D1  jc   D1            ja -->
08CA  0313  add  a,#13   (19)  CR ?
08CC  C6D4  jz   D4            ja -->
08CE  FC    mov  a,r4
08CF  12D7  jb0  D7
08D1  E5    sel  mb0
08D2  E400  jmp  700           Wait-Loop
08D4  E5    sel  mb0
08D5  2407  jmp  107           Steuerzeichen auswerten
08D7  34C3  call 9C3           Reverse-ein einfuegen
08D9  FA    mov  a,r2          aus Steuerzeichen
08DA  0340  add  a,#40   (64)  druckbares Zeichen
08DC  AA    mov  r2,a          machen
08DD  C470  jmp  E70
08DF  FC    mov  a,r4
08E0  72E6  jb3  E6
08E2  9400  call C000
08E4  04E8  jmp  8E8
08E6  34D8  call 9D8
08E8  FF    mov  a,r7
08E9  A8    mov  r0,a
08EA  FA    mov  a,r2
08EB  A0    mov  @r0,a
08EC  1F    inc  r7
08ED  FC    mov  a,r4
08EE  53FB  anl  a,#FB   (251)
```

```
08F0  AC     mov  r4,a
08F1  B821   mov  r0,#21  (33)
08F3  B004   mov  @r0,#04 (4)
08F5  BB01   mov  r3,#01  (1)
08F7  B818   mov  r0,#18  (24)
08F9  B001   mov  @r0,#01 (1)
08FB  BA92   mov  r2,#92  (146)
08FD  E5     sel  mb0
08FE  04B5   jmp  0B5
```

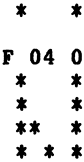
```
0900    3E 41 5D 55 5E 00
          ***
         *   *
         * ***
         * * *
         * ***
         *
          ****

0906    7E 09 09 09 7E 00
          ***
         *   *
         *   *
         *****
         *   *
         *   *
         *   *

090C    41 7F 49 49 36 00
         ****
          *  *
          *  *
          ***
          *  *
          *  *
         ****

0912    3E 41 41 41 22 00
          ***
         *   *
         *
         *
         *
         *   *
          ***

0918    41 7F 41 41 3E 00
         ****
          *  *
          *  *
          *  *
          *  *
          *  *
         ****

091E    7F 49 49 49 41 00
         *****
         *
         *
         ****
         *
         *
         *****
```

```
0924    7F 09 09 09 01 00
          *****
          *
          *
          ****
          *
          *
          *

092A    3E 41 41 49 3A 00
           ***
          *   *
          *
          *  **
          *   *
          *   *
           ***

0930    7F 08 08 08 7F 00
          *   *
          *   *
          *   *
          *****
          *   *
          *   *
          *   *

0936    00 41 7F 41 00 00
           ***
            *
            *
            *
            *
            *
           ***

093C    20 40 41 3F 01 00
            ***
              *
              *
              *
              *
          *   *
           **

0942    7F 08 14 22 41 00
          *   *
          *  *
          * *
          **
          * *
          *  *
          *   *
```

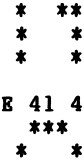
0948 7F 40 40 40 40 00

094E 7F 02 0C 02 7F 00

0954 7F 04 08 10 7F 00

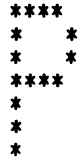


095A 3E 41 41 41 3E 00



0960 7F 09 09 09 06 00

0966 3E 41 51 21 5E 00



```
096C    7F 09 19 29 46 00
          ****
          *   *
          *   *
          ****
          * *
          *  *
          *   *

0972    26 49 49 49 32 00
           ***
          *   *
          *
           ***
              *
          *   *
           ***

0978    01 01 7F 01 01 00
          *****
            *
            *
            *
            *
            *
            *

097E    3F 40 40 40 3F 00
          *   *
          *   *
          *   *
          *   *
          *   *
          *   *
           ***

0984    07 18 60 18 07 00
          *   *
          *   *
          *   *
           * *
           * *
            *
            *

098A    7F 20 18 20 7F 00
          *   *
          *   *
          *   *
          * * *
          * * *
          ** **
          *   *
```

0990 63 14 08 14 63 00

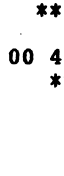
0996 07 08 78 08 07 00

099C 61 51 49 45 43 00

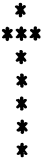
09A2 00 7F 41 41 00 00

09A8 48 7E 49 49 42 00

09AE 00 41 41 7F 00 00



09B4 00 04 02 7F 02 04



09BA 08 1C 2A 08 08 08

```
*******************************
09C0  A3    movp a,@a           Matrixspalte
09C1  A1    mov  @r1,a          holen
09C2  83    ret
*******************************
09C3  FC    mov  a,r4           Zeichen
09C4  52D2  jb2  D2
09C6  B821  mov  r0,#21  (33)   fuer
09C8  F0    mov  a,@r0
09C9  CF    dec  r7             Reverse
09CA  52D2  jb2  D2
09CC  1F    inc  r7             im
09CD  FF    mov  a,r7
09CE  A8    mov  r0,a           Puffer
09CF  B012  mov  @r0,#12 (18)
09D1  1F    inc  r7             einfuegen
09D2  83    ret
*******************************
09D3  FA    mov  a,r2           aus Steuercode
09D4  0340  add  a,#40   (64)   druckbares
09D6  AA    mov  r2,a           Zeichen
09D7  83    ret                 machen
******************************* Zeichen konvertieren
09D8  B814  mov  r0,#14  (20)
09DA  F0    mov  a,@r0
09DB  37    cpl  a
09DC  32E0  jb1  E0
09DE  8409  jmp  409
09E0  9409  call 409
09E2  FA    mov  a,r2           Zeichen
09E3  0325  add  a,#25   (37)   >219 ?
09E5  F6FE  jc   FE             ja -->
09E7  031B  add  a,#1B   (27)   =192 ?
09E9  C6FE  jz   FE             ja -->
```

```
09EB  E6F1  jnc  F1          <192 ? ja -->
09ED  0340  add  a,#40   (64)
09EF  24FD  jmp  1FD
09F1  0320  add  a,#20   (32)  >160 ?
09F3  F6FE  jc   FE            ja -->
09F5  FA    mov  a,r2
09F6  03A5  add  a,#A5   (165) >91 ?
09F8  F6FE  jc   FE            ja -->
09FA  FA    mov  a,r2
09FB  0380  add  a,#80   (128)
09FD  AA    mov  r2,a
09FE  C44B  jmp  64B
```

```
0A00    00 00 00 00 00 00
```

```
0A06    7F 7F 7F 00 00 00
          ***
          ***
          ***
          ***
          ***
          ***
          ***
0A0C    78 78 78 78 78 78
```

```
          ******
          ******
          ******
          ******
0A12    01 01 01 01 01 01
          ******
```

```
0A18    40 40 40 40 40 40
```

```
          ******
0A1E    7F 00 00 00 00 00
          *
          *
          *
          *
          *
          *
          *
```

```
0A24    55 2A 55 2A 55 2A
```

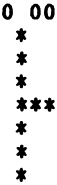
```
0A2A    00 00 00 00 00 7F
```

```
0A30    50 28 50 28 50 28
```

```
0A36    3F 1F 0F 07 03 01
```

```
0A3C    00 00 00 00 00 7F
```

```
0A42    00 00 00 7F 08 08
```



```
0A48    00 00 00 78 78 78


           ***
           ***
           ***
           ***

0A4E    00 00 00 0F 08 08
           *
           *
           *
           ***



0A54    08 08 08 78 00 00


         ****
            *
            *
            *

0A5A    60 60 60 60 60 60


         ******
         ******

0A60    00 00 00 78 08 08


            ***
            *
            *
            *

0A66    08 08 08 0F 08 08
            *
            *
            *
         ******
```

```
0A6C    08 08 08 78 08 08


          ******
             *
             *
             *

0A72    08 08 08 7F 00 00
             *
             *
             *
          ****
             *
             *
             *

0A78    7F 00 00 00 00 00
          *
          *
          *
          *
          *
          *
          *

0A7E    7F 7F 00 00 00 00
          **
          **
          **
          **
          **
          **
          **

0A84    00 00 00 00 7F 7F
                **
                **
                **
                **
                **
                **
                **

0A8A    01 01 01 01 01 01
          ******
```

```
0A90    03 03 03 03 03 03
          ******
          ******
```

```
0A96    70 70 70 70 70 70
```

```
0A9C    40 40 40 40 40 7F
```

```
0AA2    78 78 78 00 00 00
```

```
0AA8    00 00 00 07 07 07
```



```
0AAE    08 08 08 0F 00 00
```

```
0AB4    07 07 07 00 00 00
          ***
          ***
          ***
```

```
0ABA    07 07 07 78 78 78
          ***
          ***
          ***
              ***
              ***
              ***
              ***
```

```
*******************************
0AC0  A3      movp  a,@a        Matrixspalte
0AC1  A1      mov   @r1,a       holen
0AC2  83      ret
*******************************

0AC3  00 00 00 00 00 00 00 00 00 00 00 00 00
0AD0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0AE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0AF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
0B00    08 08 08 08 08 08

          ******


0B06    1C 4E 7F 4E 1C 00
            *
           ***
          *****
          *****
          * * *
            *
           ***

0B0C    00 00 7F 00 00 00
            *
            *
            *
            *
            *
            *
            *

0B12    04 04 04 04 04 04

          ******


0B18    02 02 02 02 02 02

          ******


0B1E    01 01 01 01 01 01
          ******
```

```
0B24    20 20 20 20 20 20
```



```
0B2A    00 7F 00 00 00 00
```



```
0B30    00 00 00 00 7F 00
```



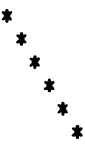
```
0B36    08 08 10 60 00 00
```

```
0B3C    00 00 00 03 04 08
```

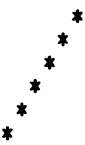
```
0B42    08 08 04 03 00 00
```

```
0B48    7F 40 40 40 40 40
```

```
0B4E    02 04 08 10 20 40
```



```
0B54    40 20 10 08 04 02
```



```
0B5A    7F 01 01 01 01 01
```

```
0B60    01 01 01 01 01 7F
```

```
0B66    3C 7E 7E 7E 3C 00
```

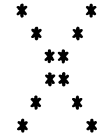
```
0B6C    20 20 20 20 20 20
```

```
0B72    1C 3E 7C 3E 1C 00
```

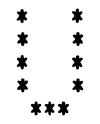
```
0B78    00 7F 00 00 00 00
```

```
0B7E    00 00 00 60 10 08
```

```
0B84    42 24 18 18 24 42
```



```
0B8A    3C 42 42 42 3C 00
```



0B90 1C 0A 7F 0A 1C 00

0B96 00 00 00 00 7F 00

0B9C 18 3C 7E 3C 18 00

0BA2 08 08 08 7F 08 08

0BA8 55 2A 55 00 00 00



0BAE 00 00 00 7F 00 00

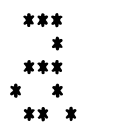
```
0BB4    08 7C 04 7C 04 00


          ****
         ** *
          * *
          * *
          * *

0BBA    01 03 07 0F 1F 3F
         ******
          *****
           ****
            ***
             **
              *


0BC0    00 00 00 7F 7F 7F
            ***
            ***
            ***
            ***
            ***
            ***
            ***

0BC6    10 10 10 10 10 10



         ******


0BCC    78 20 10 08 04 00


             *
          *  *
          * *
          **
          *

0BD2    12 49 24 12 49 24
            *  *
          *  *
            *  *
           *  *
          *  *
            *  *
           *  *
```

```
*******************************
0BD8  A3     movp a,@a          Matrixspalte
0BD9  A1     mov  @r1,a         holen
0BDA  83     ret
*******************************

0BDB  00 00 00 00 00
0BE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0BF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0C00  B814   mov  r0,#14  (20)
0C02  F0     mov  a,@r0
0C03  1207   jb0  07
0C05  8409   jmp  C09
0C07  24E0   jmp  9E0
******************************* Zeichen konvertieren
0C09  FA     mov  a,r2          Zeichen
0C0A  0360   add  a,#60   (96)  <160 ?
0C0C  E620   jnc  20            ja -->
0C0E  03C0   add  a,#C0   (192) <224 ?
0C10  E634   jnc  34            ja -->
0C12  03E1   add  a,#E1   (225) =255 ?
0C14  C61D   jz   1D            ja -->
0C16  0309   add  a,#09   (9)   =246 ?
0C18  9630   jnz  30            nein -->
0C1A  BAE0   mov  r2,#E0  (224)
0C1C  83     ret
0C1D  BADE   mov  r2,#DE  (222)
0C1F  83     ret
0C20  0340   add  a,#40   (64)  <96 ?
0C22  E634   jnc  34            ja -->
0C24  03FA   add  a,#FA   (250) =102 ?
0C26  962B   jnz  2B            nein -->
0C28  BAE1   mov  r2,#E1  (225)
```

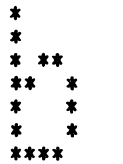
```
0C2A  83     ret
0C2B  FA     mov  a,r2
0C2C  0360   add  a,#60   (96)
0C2E  8433   jmp  C33
0C30  FA     mov  a,r2
0C31  03C0   add  a,#C0   (192)
0C33  AA     mov  r2,a
0C34  83     ret

0C35  00 00 00 00 00 00 00 00 00 00 00
0C40  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0C50  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0C60  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0C70  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0C80  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0C90  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0CA0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0CB0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0CC0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0CD0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0CE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0CF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
0D00    08 08 08 08 08 08
```

```
0D06    20 54 54 3C 40 00
```

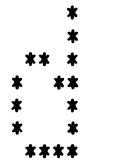


```
0D0C    7F 48 44 44 38 00
```



```
0D12    38 44 44 44 44 00
```

```
0D18    38 44 44 48 7F 00
```



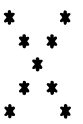
```
0D1E    38 54 54 54 08 00
```

```
0D24    08 7E 09 01 02 00
          **
         *  *
         *
        ***
         *
         *
         *

0D2A    0C 52 52 52 3C 00

         ***
        *   *
        *   *
         ****
            *
         ***

0D30    7F 08 04 04 78 00
        *
        *
        * **
        **  *
        *   *
        *   *
        *   *

0D36    00 44 7D 40 00 00
         *

        **
         *
         *
         *
        ***

0D3C    00 20 40 3D 00 00
           *

           *
           *
           *
         * *
          *

0D42    7F 10 28 44 00 00
        *
        *
        *  *
        * *
        **
        * *
        *  *
```

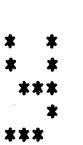
```
0D48    00 01 7F 40 00 00
          **
           *
           *
           *
           *
           *
           **

0D4E    7C 04 38 04 78 00

         ** *
         * * *
         * * *
         * * *
         *   *

0D54    04 78 04 04 78 00

         * **
          *  *
          *  *
          *  *
          *  *

0D5A    38 44 44 44 38 00

          ***
         *   *
         *   *
         *   *
          ***

0D60    7E 12 12 0C 00 00

         ***
         *  *
         *  *
         ***
         *
         *

0D66    0C 12 12 7E 00 00

          ***
         *  *
         *  *
          ***
             *
             *
```

```
0D6C    7C 08 04 04 08 00

          * **
          **  *
          *
          *
          *

0D72    48 54 54 54 20 00

           ***
          *
           ***
              *
          ****

0D78    04 3F 44 44 20 00
           *
           *
          ****
           *
           *
           *  *
            **

0D7E    3C 40 40 3C 40 00

          *  *
          *  *
          *  *
          *  *
           ** *

0D84    0C 30 40 30 0C 00

          *   *
          *   *
           * *
           * *
            *

0D8A    3C 40 30 40 3C 00

          *   *
          *   *
          * * *
          * * *
           * *
```

0D90 44 28 10 28 44 00

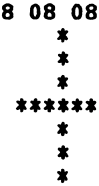


0D96 4C 50 50 3C 00 00



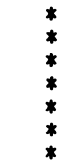
0D9C 44 64 54 4C 44 00

0DA2 08 08 08 7F 08 08



0DA8 55 2A 55 00 00 00

0DAE 00 00 00 7F 00 00



```
0DB4    73 73 0C 0C 73 73
          **  **
          **  **
            **
            **
          **  **
          **  **
          **  **

0DBA    24 49 12 24 49 12
           *  *
            *  *
          *  *
            *  *
              *  *
          *  *
            *  *
```

```
*******************************
0DC0  A3     movp a,@a          Matrixspalte
0DC1  A1     mov  @r1,a         holen
0DC2  83     ret
*******************************

0DC3  00 00 00 00 00 00 00 00 00 00 00 00 00
0DD0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0DE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0DF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

****************************** Linefeed, Schrittzahl
                               in Speicher 16
0E00  B910   mov  r1,#10  (16)
0E02  F1     mov  a,@r1
0E03  A9     mov  r1,a
0E04  99FD   anl  p1,#FD  (253) Linefeed-Motor ein
0E06  D437   call E37           Delay
0E08  D424   call E24           Motor ein Schritt vor
0E0A  E90E   djnz r1,0E
0E0C  C41D   jmp  E1D
0E0E  D43B   call E3B           Delay
0E10  C419   jmp  E19
0E12  D43F   call E3F           Delay
0E14  E918   djnz r1,18
0E16  D43F   call E3F           Delay
0E18  19     inc  r1
0E19  D424   call E24           Motor ein Schritt vor
0E1B  E912   djnz r1,12
0E1D  D437   call E37           Delay
0E1F  8902   orl  p1,#02  (2)   Linefeed-Motor aus
0E21  E5     sel  mb0
0E22  A4EE   jmp  5EE
****************************** Linefeed-Motor einen
```

```
                                        Schritt vorruecken
0E24  0A    in   a,p2
0E25  122C  jb0  2C
0E27  3234  jbl  34
0E29  8A02  orl  p2,#02  (2)
0E2B  83    ret
0E2C  3231  jbl  31
0E2E  9AFE  anl  p2,#FE  (254)
0E30  83    ret
0E31  9AFD  anl  p2,#FD  (253)
0E33  83    ret
0E34  8A01  orl  p2,#01  (1)
0E36  83    ret
****************************** Delay
0E37  2330  mov  a,#30   (48)
0E39  C441  jmp  E41
0E3B  2351  mov  a,#51   (81)
0E3D  C441  jmp  E41
0E3F  23A7  mov  a,#A7   (167)
0E41  62    mov  t,a
0E42  55    strt t
0E43  1645  jtf  45
0E45  1649  jtf  49
0E47  C445  jmp  E45
0E49  65    stop tcnt
0E4A  83    ret
****************************** Zeichenkonvertierung
0E4B  FA    mov  a,r2           Zeichen
0E4C  031F  add  a,#1F   (31)   =225 ?
0E4E  9653  jnz  53             nein -->
0E50  BA46  mov  r2,#46  (70)
0E52  83    ret
0E53  0301  add  a,#01   (1)    =224 ?
0E55  9658  jnz  58             nein -->
```

```
0E57  83    ret
0E58  0320  add  a,#20   (32)  =192 ?
0E5A  C657  jz   57             ja -->
0E5C  E662  jnc  62             <192 ? ja -->
0E5E  0360  add  a,#60   (96)
0E60  AA    mov  r2,a
0E61  83    ret
0E62  0306  add  a,#06   (6)   =186 ?
0E64  9669  jnz  69             nein -->
0E66  BAE2  mov  r2,#E2  (226)
0E68  83    ret
0E69  0311  add  a,#11   (17)  =169 ?
0E6B  9657  jnz  57             nein -->
0E6D  BAE3  mov  r2,#E3  (227)
0E6F  83    ret
******************************
0E70  03C0  add  a,#C0   (192)
0E72  C676  jz   76
0E74  04DF  jmp  8DF
0E76  04E8  jmp  8E8
****************************** Adresse pruefen
0E78  F0    mov  a,@r0          bereits adressiert ?
0E79  9295  jb4  95             ja --> Sekundaeradresse
0E7B  0A    in   a,p2           hat der Drucker Adresse 4 ?
0E7C  D282  jb6  82             ja -->
0E7E  23DB  mov  a,#DB   (219)
0E80  C484  jmp  E84
0E82  23DC  mov  a,#DC   (220)
0E84  6A    add  a,r2           richtige LISTENER-Adresse ?
0E85  C68A  jz   8A             ja -->
0E87  E5    sel  mb0            sonst
0E88  E4C8  jmp  7C8            zurueck
****************************** Drucker ist adressiert
0E8A  2310  mov  a,#10   (16)
```

```
0E8C  40     orl  a,@r0          LISTENER-Flags
0E8D  A0     mov  @r0,a
0E8E  A5     clr  f1             setzen
0E8F  B5     cpl  f1
0E90  C4AE   jmp  EAE
*******************************
0E92  E5     sel  mb0
0E93  E4BA   jmp  7BA
******************************* Sekundaeradresse pruefen
0E95  B292   jb5  92             bereits dagewesen ? ja -
0E97  2320   mov  a,#20   (32)
0E99  40     orl  a,@r0
0E9A  A0     mov  @r0,a
0E9B  FA     mov  a,r2
0E9C  0396   add  a,#96   (150)
0E9E  C6BE   jz   BE
0EA0  0302   add  a,#02   (2)
0EA2  C6AE   jz   AE
0EA4  17     dec  a
0EA5  C6B4   jz   B4
0EA7  17     dec  a
0EA8  C6BE   jz   BE
0EAA  0306   add  a,#06   (6)
0EAC  96BE   jnz  BE
0EAE  23F7   mov  a,#F7   (247)
0EB0  5C     anl  a,r4
0EB1  AC     mov  r4,a
0EB2  C4B8   jmp  EB8
0EB4  2308   mov  a,#08   (8)
0EB6  4C     orl  a,r4
0EB7  AC     mov  r4,a
0EB8  237C   mov  a,#7C   (124)
0EBA  50     anl  a,@r0
0EBB  A0     mov  @r0,a
```

```
0EBC  C4C2  jmp  EC2
0EBE  2380  mov  a,#80   (128)
0EC0  40    orl  a,@r0
0EC1  A0    mov  @r0,a
0EC2  E5    sel  mb0
0EC3  E4BA  jmp  7BA
****************************** E N D E **********************
```

Ab hier bis zum Speicherende bei $0FFF stehen nur noch $00.

## *8.0 DER PRINTER-PLOTTER VC-1520*

### *8.1 Arbeitweise und Nutzungsmöglichkeiten des Plotters*

Um die optimale Ausnutzung der Möglichkeiten des Plotters zu gewährleisten und die Funktionen zu verstehen ist es wichtig, zunächst einmal etwas über die allgemeine Arbeitsweise des Gerätes zu erfahren. Dazu soll ein Blick auf die wichtigsten technischen Daten dienen:

4 Kugelschreiber (schwarz, blau, grün, rot)
4 verschiedene Zeichengrößen
10/20/40 oder 80 Zeichen pro Plotterzeile
13 Schriftzeichen pro Sekunde bei kleinster Schriftart
96 verschiedene Zeichen
50 Zeichenschritte pro Zentimeter
480 Zeichenschritte in x-Richtung
+/- 999 Zeichenschritte in y-Richtung
264 Zeichenschritte pro Sekunde
5 Zentimeter Zeichenstrecke pro Sekunde (x- oder y-Richtung)
7 Zentimeter Zeichenstrecke pro Sekunde unter 45 Grad-Winkel
11.4 Zentimeter breites Papier (4,5 Zoll)

Diese Angaben sollen zunächst genügen, da man sich nun schon ein Bild von der Leistungsfähigkeit des Plotters machen kann. Das Gerät ist bei einer vollen Ausnutzung ein kleines Wunderwerk. Das erkennt man z.B. an der Auflösungsfähigkeit beim Zeichnen; auf einem Quadrat von 480 x 480 Schritten können nämlich 230.400 Punkte unabhängig voneinander angesteuert werden. Auch die Zeichengeschwindigkeit läßt nichts zu wünschen übrig, da 5 cm pro sec. durchaus die Vorstellung vom angenehmen Arbeiten decken.
Verwirrend mag es vielleicht auf den ersten Blick sein, daß man in x-Richtung oder y-Richtung jeweils nur die bekannten

5 cm zeichnen kann, während doch unter einem Winkel von 45 Grad eine Zeichengeschwindigkeit von 7 cm pro sec zu erkennen ist. Dieser scheinbare Widerspruch klärt sich aber mathematisch recht leicht auf:

Wir denken uns zur Vereinfachung der Erklärung einfach einmal, daß zuerst eine Sekunde in x-Richtung und anschließend eine Sekunde in y-Richtung gezeichnet wurde. Man hat also dann die beiden 5 cm langen Geraden sekrecht aufeinander stehen, unter einem Winkel von 90 Grad also. Hier gilt nun der Satz des Pythagoras. Wir haben nämlich einen rechten Winkel und zwei Seiten und können nun die Zeichenstrecke der fehlenden Seite berechnen.

Es gilt $x^2+y^2=z^2$
oder mit Zahlen:

$$5^2+5^2=z^2$$

Somit ist z die Wurzel aus 50, ungefähr sieben cm. Durch die Kombination beider Zeichenrichtungen bleibt die Zeichenzeit von einer Sekunde bestehen und es werden trotzdem mehr cm zurückgelegt. Diese Analyse wurde deshalb so ausführlich vorgenommen, da das Programmieren mit dem Plotter von Beginn an verstanden werden muß. um die Arbeit mit dem Gerät nicht in ein Abtippen von bestehenden Routinen ausarten zu lassen - schließlich will man sich für das Investierte auch selber kreativ betätigen können.

Als nächstes sollten Sie die Schutzhaube des Plotters ruhig einmal entfernen und sich die mechanischen Funktionsteile etwas genauer betrachten.

Zuerst wird die Kugelschreibertrommel ins Auge fallen, da mit ihr der Plotter sein Können zu Papier bringt. Aber auch die anderen "Helfer" verdienen einen kurzen Blick, da mit

ihrer Präzision die Funktionsgenauigkeit und Funktionsicherheit steht oder fällt. Links und rechts der Gummiwalze sieht man jeweils einen komplizierten Zahnradsatz, der den Walzentransport und den Transport übernimmt; denn die Trommel kann sich nur in x-Richtung bewegen und so muß das Papier die y-Richtung versorgen.
Auf der Gummiwalze befindet sich ein Nadel-Traktorantrieb (man kann die feinen Spitzen kaum sehen), der präzise arbeiten kann und trotzdem das Papier nicht stark in Mitleidenschaft zieht. Als letztes bleibt von den mechanischen Teilen noch der kleine grüne Block in der rechten Ecke des Getriebes. Dieses ist ein kleiner Elektromagnet, der den jeweiligen Stift auf das Papier setzt oder ihn abnimmt. Das geschieht durch eine Leiste, die den Druck der Trommelfelder entweder überwindet und den Stift damit setzt, oder zurückgezogen die Trommelfeder wirken läßt und der Stift somit vom Papier genommen wird.
Dies soll von der technischen Seite zunächst genug sein, um die Leistungsfähigkeit des Plotters 1520 zu begründen, doch vorab muß noch eine Verhaltensregel eingehalten werden, um an seinem Gerät Freude zu haben. Wenden Sie niemals Gewalt an. Das Gerät ist sehr robust konstruiert und ist sehr weit belastbar, aber die feinen Drahtseile und Zahnräder halten einen neugierigen Menschen nicht lage aus. Beachten Sie deshalb die folgenden Regeln im Umgang mit dem Plotter:

- Die Walze nur im Notfall von Hand drehen (etwa zum Einlegen einer neuen Papierrolle)
- Die Kugelschreibertrommel niemals von Hand drehen.
- Mechanische Teile, wie die Leiste des Elektromagneten oder den Zahnradsatz nicht unnötig von Hand bewegen, auch hier kann die Zeichengenauigkeit verschlechtert werden.
- Daß nicht ohne Papier gedruckt werden darf, die Stromversorgung während des Druckvorgangs nicht

unterbrochen werden sollte und nicht im Regen geplottet werden kann, dürfte ebenfalls selbstverständlich sein.

Zum Ende dieser technischen Einführung noch kurz etwas zur Wartung und Reinigungdes Gerätes:

- Setzen Sie das Gerät keiner zu großen Staubbelastung aus, die gut gefetteten Antriebsteile verkleben dann langsam aber sicher und es kommt zu Funktionsstörungen.
- Wird die Rolle aus Versehen mit Kugelschreiber bemalt, so öffnen Sie sofort die Abdeckhaube und wischen die Rolle mit einem leicht angefeuchteten Tuch ab.
- Sollte es zu einem (recht unwahrscheinlichen) Papierstau kommen, so ist ebenfalls die Haube zu öffnen und der Stau von Hand zu beseitigen. Drücken Sie nicht die Paper-Feed-Taste!

Betrachtet man sich nun den Plotter unter dem Aspekt der Einsatzmöglichkeiten und bezieht die vorhergehenden Ausführungen mit ein, so kann man erkennen, daß der Plotter von sich aus alle nur denkbaren Einsatzmöglichkeiten von mathematischen Figuren bis zur Statistik hin bietet. Sie müssen es nur aus ihm herausholen.
Das Ziel des "Herausholens", verbunden mit einer effektiven Nutzung, soll auch durch alle folgenden Beschreibungen erreicht werden. Lösungsmöglichkeiten von Problemen sind mit vielen Beispielen verknüpft und dienen nicht nur dem Abtippen mit anschließendem "Aha"-Effekt, sondern vielmehr zur Anregung, eigene Probleme schneller und einfacher zu lösen.
Die gewählten Beispiele können keinen Anspruch auf Vollständigkeit erheben. Diese würde den zur Verfügung stehenden Rahmen sprengen - aber die Beispiele werden ausführlich erklärt und der Plotter 1520 wird auch für Sie

zum Spezialisten, indem Sie ihn selber dazu machen.

Es gilt aber nach wie vor: der Plotter kann nur jenes leisten, was man ihm aufträgt, und dazu gehören die nötigen Programmroutinen. Um diese Routinen schreiben zu können folgt nun eine ausführliche Erklärung des Befehlssatzes.
Als Peripheriegerät zählt der Plotter 1520 zu den "intelligenten" Geräten. Das bedeutet, daß im Rechnerspeicher des Commodore 64 kein Speicherplatz abgezweigt werden braucht, um den Plotter zu steuern. Dieser besitzt nämlich eigene Speicherelemente und ein eigenes Betriebssystem mit einer eigenen Befehlssyntax.
Die Kommunikation erfolgt, wie es auch von anderen Geräten (Floppy 1541 oder Datsette) bekannt sein dürfte, über Befehle wie OPEN, PRINT# oder CLOSE. Diese Befehle werden noch mit Adressen verknüpft und über Sekundäradressen noch weiter aufgespalten und diversifiziert.
- OPEN
Dieser Befehl öffnet in Verbindung mit einer Adresskette, auf die gleich näher eingegangen wird, den Plotter 1520 für bestimmte Funktionen, die durch eben diese Adresskette ausgewählt werden. Erst nach dem OPEN - Befehl ist das Gerät in der Lage, Anweisungen zu bearbeiten, ganz gleich ob, im Direktmodus gearbeitet wird oder die Steuerung über Programm erfolgen soll.
Die Adresskette hinter dem OPEN - Befehl besteht in der Regel aus drei Zahlen, die durch ein Komma voneinander zu trennen sind. Allgemein sieht die Anweisung also folgendermaßen aus:

OPEN nr, ga, bf

Die Kürzel bedeuten folgendes:
nr ist die Nummer, unter der später Print# oder Close

Befehle angesprochen werden. Sie können Nummern von 1 bis 225 wählen, müssen dann aber das gerade Gesagte unbedingt beachten, da sonst ein "File not open" Error auf Sie wartet. Wir werden aber im Laufe der weiteren Ausführungen noch auf diesen Befehl zurückkommen und den Gebrauch an Beispielen dokumentieren.
Auch das ga, die Geräteadresse, schafft keine Probleme; sie ist durch die Konstruktion fest vorgegeben und kann ohne weiteres nicht geändert werden. Bei dem Plotter 1520 ist diese Geräteadresse die 6. Diese 6 wird bei allen Beispielen als zweite Nummer auftauchen und bedarf ansonsten keiner Erklärung.
Interessanter wird es da schon bei der noch verbleibenden, dritten Zahl bf, dem Befehl. Über diesen Befehl, auch Sekundäradresse genannt, wird der Prozessor im 1520 über unsere Absichten informiert. Diese Zahl ist die für die geplante Ausführung entscheidende Zahl. Mit ihr wählt man die Funktionen des Gerätes an und führt sie später aus.
Folgende Befehle oder Sekundäradressen sind für den Plotter 1520 zulässig, bzw. sinnvoll:

bf=0 Normaler Druckverkehr zum Schreiben oder Listen
bf=1 Zeichenroutinen erledigen
bf=2 Steuerung der Farbtrommel zur Farbwahl
bf=3 Anwahl der Zeichengröße
bf=4 Drehung der Zeichen um 90 Grad
bf=5 Auswahl der Linierart des 1520
bf=6 Umschalten des Plotters zwischen Groß/Klein/Grafik-Modus
bf=7 Rücksetzen des Plotters in den Urzustand, Löschung aller im Plotterrechenspeicher vorhandener Daten.

Somit können jetzt kurz ein paar erlaubte OPEN-Befehle gezeigt werden.

OPEN 34,6,2 öffnet das File Nummer 34 bezüglich des Plotters 1520 (6) und wählt die Steuerung der Farbtrommel an.

OPEN 1,6,1 öffnet das File Nummer 1 bezüglich des Plotters 1520 (6) und bereitet das Gerät auf die Abarbeitung einer Zeichenroutine vor.

OPEN 99,9,9 ist nicht sinnvoll, da der Plotter mit einer 9 anstatt einer 6 angesprochen wurde und auch eine 9 als Sekundäradresse nicht sinnvoll sein kann.

Sie erhalten einen "Device not present" Error.

Auf alle Sekundäradressen wird nach der Erklärung der Befehle mit Beispielen eingegangen, da das Handbuch des Plotters etwas stiefmütterlich über einige Befehle hinweggeht und für den Anfänger nicht immer ausführlich genug erklärt.

- PRINT#

Dieser Befehl führt weitere Anweisungen, die an den Plotter gegeben wurden, aus. Er ist verantwortlich für die Sichtbarmachung der berechneten Daten auf dem Papier, dreht die Farbtrommel zum gewünschten Kugelschreiber oder setzt den Plotter zurück.

Der Befehl lautet allgemein:

PRINT# nr, Anweisung

Sie erinnern sich an das vorher Gesagte? nr war die Nummer, mit der Sie das File geöffnet hatten. Sie kann zwischen 1 und 255 liegen. Wollen Sie sich nun mit dem PRINT#-Befehl auf ein mit OPEN geöffnetes File beziehen, so muß das nr des OPEN-Befehls gleich dem nr des PRINT#-Befehls sein. Im Klartext bedeutet das: Öffnen Sie ein File mit OPEN 1,6,1, so ist hier die erste 1 nur unsere Bezugsnummer.

Sollen nun Befehle in Bezug auf das geöffnete File getätigt

werden, so muß der PRINT#-Befehl lauten:
PRINT# 1, Anweisung
Auf die Anweisungen soll jetzt noch keine Rücksicht genommen werden. Diese werden bei den Sekundäradressen in den bereits erwähnten Beispielen folgen.
Zur Struktur des Wortes PRINT# nur noch eine kleine Anmerkung, die sich auch im Commodore 64 - Handbuch findet, aber trotzdem kurz erwähnt werden soll. Wollen Sie nämlich das Wort PRINT# abkürzen, so wird oft der Fehler begangen, es mit ?# zu versuchen. Dieses fuktioniert nicht und kostet Sie einen "Syntax" Error.
Versuchen Sie es dagegen mit P shift R, so gelingt die Abkürzung ohne Beanstandungen durch den Computer.

- CLOSE
Dieser Befehl schließt ein mit OPEN geöffnetes File. Er beendet damit die Benutzung des PRINT#-Befehls in Bezug auf die geschlossene Nummer nr. Die Eingabeform des CLOSE-Befehls ist
CLOSE nr
Sie schließen ein File, welches mit OPEN 2,6,2 geöffnet wurde, also mit CLOSE 2. Versuchen Sie nun ruhig einmal, ein PRINT# 2,2 einzugeben. Der Plotter kann diesen Befehl nicht mehr empfangen und es wird ein "File not open" Error ausgegeben.
- CMD
Der CMD-Befehl ist auch für den Plotter 1520 vorhanden und erfüllt auch bei ihm nützliche Aufgaben. Da der 1520 nämlich auch zum Auslisten kleinerer Programmroutinen durchaus geeignet ist, muß man dem Befehl Bedeutung beimessen.

Wie wendet man die Anweisung nun richtig an, um z.B. ein Programm zu listen? Hierbei gibt es zwei Möglichkeiten. Sie können zum einen, wenn das Programm geladen wurde, die

Befehle im Tischrechnerformat eingeben, d. h. direkt und ohne Programmzeilen.

Die Reihenfolge der Schritte ist dann wie folgt:

a) Das Programm muß im Speicher des VC 64 vorhanden sein.
b) Sie geben nun OPEN 4,6 ein, da der CMD-Befehl nur im Zusammenhang mit OPEN lauffähig ist.
c) Nun geben Sie CMD 4 ein, um alle weiteren Befehlsausführungen direkt auf das mit OPEN geöffnete Peripheriegerät zu senden.
d) Jetzt wird der Befehl erwartet und Sie können z.B. LIST eingeben. Das im Rechner befindliche Programm wird mit dem Plotter ausgelistet.
e) Auch nach dem Listvorgang ist der CMD-Befehl noch wirksam und somit muß der Kanal wieder freigeschaltet werden. Dazu gibt man nacheinander PRINT# nr und CLOSE nr ein.

Noch einmal zusammengefaßt gilt für Direkteingaben:

```
OPEN nr,6
CMD nr
LIST
PRINT# nr
CLOSE nr
```

Der CMD - Befehl hängt aber nicht von der Direkteingabe ab, die für das Listen aber angebrachter erscheint, da die Eingaben auf dem Listing nicht erscheinen. Sie können selbstverständlich das CMD auch im Ablauf einer Programmroutine verwenden. Dazu sehen Sie sich das nachfolgende Programm einmal an.

```
10 OPEN4,6
20 CMD4,"PLOTTER 1520"
30 LIST
```

In Zeile 10 wird der Plotter zum Schreiben geöffnet, Zeile 20 arbeitet nach dem Prinzip CMD 4, Befehl. Der Befehl ist bei diesem Beispiel eben ein Text, der in Anführungsstrichen stehen muß. Lassen Sie dieses Programm laufen, so ist nach dem Listen das File noch geöffnet und Sie müssen es von Hand mit PRINT# 4 und CLOSE 4 schließen. Bauen Sie nun probehalber einmal zwei weitere Zeilen in das Programm ein:

```
40 PRINT# 4
50 CLOSE 4
```

Lassen Sie es ablaufen. Nachdem es beendet ist, geben Sie CMD 4, "PLOTTER 1520" ein, doch Sie erhalten keine Fehlermeldung, obwohl doch ein File angesprochen wird, welches in Zeile 50 geschlossen wurde. Diese Tatsache ist nicht etwa ein Fehler in Ihrem Programm oder gar ein Plotterfehler, sondern erklärt sich mit dem Betriebssystem des Commodore 64. Dort wird nämlich das Programm nach einem gefundenen und ausgeführten LIST-Befehl abgebrochen. Deshalb sind die Zeilen 40/50 nutzlos und das File muß von Hand geschlossen werden.

Diese Darstellungen sollen als Einführung oder Wiederholung genügen.

### *8.2 Die Sekundäradressen des VC-1520*

Diese Angabe war beim OPEN-Befehl die letzte Ziffer in unserer Adresskette. Sie kann zwischen 0 und 7 angewählt werden und erfüllt dann bestimmte Funktionen, die mit dem PRINT# nr aufgerufen werden können. Sollten Sie die Bedeutungen der einzelnen Ziffern nicht mehr im Kopf haben, so blättern Sie ein Stück zurück zum OPEN-Befehl. Dort

finden Sie die entsprechende Liste der Zahlen mit ihrem Bedeutungen. Es werden nun alle Adressen durchgegangen und an kleinen Programmbeispielen dokumentiert.

a) bf = 1

Schon bei der ersten Ziffer, die den Befehlssatz zum Zeichnen im Plotter freigibt, wird man sich, besonders im Interesse der Anfänger, etwas länger aufhalten müssen, als es das Handbuch des 1520 tut, denn relative, absolute und sonstige Zeichenfunktionen müssen ja genau verstanden worden sein, um die Zeichenkunst des Gerätes sinnvoll und effektiv zu nutzen. Wer sich bereits auskennt, der kann zur Wiederholung vielleicht trotzdem weiterlesen.

Wie Sie wissen, ist zuerst die Sekundäradresse in Bezug auf den Plotter anzusprechen. Wir wählen OPEN 1,6,1.
Nun kann der Befehlssatz des Plotters angesprochen werden. Dies geht ja mit der bekannten PRINT# - Funktion. Die Form der Eingabe lautet:

```
PRINT# 1, "Anweisung"; x-Daten;y-Daten oder
PRINT# 1, "Anweisung", x-Daten,y-Daten oder
PRINT# 1, "Anweisung"  x-Daten,y-Daten
```

Die Trennung von Anweisung, x-Daten und y-Daten kann also durch ein Semikolon, ein Komma oder eine Leerstelle (Space) erfolgen. Es existieren insgesamt sechs Anweisungen, die der Plotter verstehen und somit ausführen kann.

1. H bringt die Kugelschreibertrommel in ihre Ausgangsposition, also auf die linke Seite der Walze. Diese Stellung ist der sogenannte absolute Nullpunkt. Er ist Ihre Ausgangsposition, die absolute Anfangsstellung x= 0 : y=0.

Auf diesen absoluten Nullpunkt beziehen sich zwei weitere Befehle, das "D" und das "M".

2. M steht für Bewegung der Trommel in Bezug auf den absoluten Nullpunkt auf der linken Seite des Plotters. Der Ausgangspunkt des Befehls bleibt also immer x=0:y=0! Der Stift ist vom Papier abgezogen, die Trommel bewegt sich zu den angegebenen Koordinaten (x=0-480:y=+/-999 es wird aber noch nicht gezeichnet!

3. D ist im Gebrauch dem Befehl M identisch, doch nun wird der Elektromagnet angeregt und die Leiste drückt den gewählten Kugelschreiber auf das Papier und der Plotter zeichnet zu den angegebenen Koordinaten. Auch der D - Befehl bezieht sich immer auf den absoluten Nullpunkt, also die Stellung (0/0) am linken Walzenrand.

Die Art der Eingaben wird nachher an Beispielen geübt und erläutert, doch nun zuerst zu den drei übrigen Zeichenbefehlen des 1520.

4. I Neben dem absoluten Nullpunkt (0/0) existiert noch ein weiterer, ein relativer Nullpunkt. Diesen können Sie frei anwählen und ändern. Sie fahren z.B. zuerst mit M an eine ganz bestimmte Position und geben dann PRINT# 1, "I" ein. Diese angefahrene Position ist dann, nachdem sie durch I abgespeichert wurde, Ihr derzeitiger, relativer Nullpunkt. Setzen Sie die Kugelschreibertrommel auf eine andere Position und geben dann wiederum PRINT# 1, "I" ein, so wird der alte, relative Nullpunkt durch den neuen ersetzt. Wie zuvor bei unserem H, beziehen sich auch auf den Befehl I zwei weitere Steuerungsanweisungen.

5. R entspricht in der Ausführung dem M-Befehl, d. h. auch

hier ist der Kugelschreiber vom Papier abgezogen und die Trommel bewegt sich an die angegebene x/y-Position. Es gilt für diesen R-Befehl nur ein anderes Bezugssystem, nämlich der relative Nullpunkt. Setzen Sie z.B. den relativen Nullpunkt auf 100/-100 und geben dann PRINT#1,"R";1;1 ein, so werden diese Schritte von 100/-18o getätigt und nicht etwa von 0/0 ! Um den gleichen Effekt vom absoluten Nullpunkt aus zu erhalten, müßte man also mit dem M-Befehl arbeiten und als Koordinaten 101/-99 eingeben.

6. J ist der Zeichenbefehl in Bezugnahme auf den relativen Nullpunkt und entspricht in der Koordinatenhandhabung dem D. Der Stift wird gesetzt und es wird eine Linie zu der gewünschten Position gezogen.

Nach dieser Erläuterung soll die Anwendung der Befehle an einigen kleinen Routinen gezeigt werden. Wir wollen versuchen, ein Dreieck zu konstruieren und damit die Unterschiede der absoluten und relativen Zeichenweise aufzeigen. Sehen Sie sich die folgende Routine einmal an:

```
10 OPEN1,6,1
20 PRINT#1,"M";100;0
30 PRINT#1,"D";300;0
40 PRINT#1,"D";200;100
50 PRINT#1,"D";100;0
60 PRINT#1,"H"
70 PRINT#1,"M";0;-100
80 CLOSE1
```

Haben Sie das Programm abgetippt, so starten Sie es. Man erhält dann obiges Bild. Sie erkennen deutlich, daß es sich hier um die absolute Zeichenweise handelt, da nur die Befehle H,M und D erscheinen. In Zeile 20 bewegt sich der Kopf mit dem Schreiber an die Anfangsposition 100/0. Die Zeilen 30 - 50 zeichnen das Dreieck, wobei alle Positionen

von 0/0 aus gerechnet werden müssen! Die Zeilen 60 und 70 schieben das Papier noch etwas heraus, so daß das Werk sichtbar wird. Man kann hier schon erkennen, daß es besonders bei komplizierteren Zeichnungen nicht angebracht erscheint, nur mit absoluten Nullpunkten zu arbeiten, da das kein ideales Bezugssystem darstellt. Es wäre schon bei diesem einfachen Dreieck aus Gründen der Übersicht einfacher, eine Mischform der relativen und absoluten Befehlssätze zu verwenden. Dies wird mit dem folgenden Programm realisiert.

```
10 OPEN1,6,1
20 PRINT#1,"M";100;0
30 PRINT#1,"I"
40 PRINT#1,"J";200;0
50 PRINT#1,"J";100;100
60 PRINT#1,"J";0;0
70 PRINT#1,"R";-100;-100
80 CLOSE1
```

Lassen Sie das Programm laufen, so kann man an dem Endresultat keine Veränderungen sehen, aber der Weg, auf dem das Ziel erreicht wurde, ist etwas anders. Identisch geblieben ist die Fixierung des Startpunktes auf 100/0 in Zeile 20. Doch nun wird nicht sofort und damit absolut weitergezeichnet, sondern die Position 100/0 wird mit PRINT # 1, "I" zum relativen Nullpunkt erklärt. Von Zeile 40 - 60 wird dann das Dreieck mit dem Befehl "J" gezeichnet. Die Berechnungen sind nun einfacher geworden, da man sich direkt auf die linke Ecke des Dreiecks beziehen kann.

Es empfehlen sich also in jedem Fall Mischformen der Befehle, wenn Sie den Rechenaufwand gering halten wollen. Denn versuchen wir nun, das Dreieck nur mit den Relativanweisungen zu gestalten, so sieht das folgendermaßen

aus:

```
10 OPEN1,6,1
20 PRINT#1,"R";100;0
30 PRINT#1;"I"
40 PRINT#1,"J";200;0
50 PRINT#1,"J";100;100
60 PRINT#1,"J";0;0
70 PRINT#1,"R";-100;-100
80 CLOSE1
```

In Zeile 20 wurde das M durch ein R ersetzt. Dieses war möglich, da vor dem Zeichenbeginn absoluter und relativer Nullpunkt identisch sind und somit keine große Programmänderung erforderlich war. Man sollte aber immer die bestmögliche Kombination der Befehle benutzen. Dies wird sich im Verlauf der weiteren Darstellungen noch zeigen, wenn die Routinen komplizierter werden oder mathematische Funktionen dargestellt werden sollen. Die Sekundäradresse muß also unbedingt verstanden worden sein, wenn man selber Zeichnungen entwerfen möchte. An ihr kann man nicht schnell vorbeilesen, sondern, wie im Beispiel, ruhig kurz eingeben und analysieren!

b) bf = 2

Mit dieser Sekundäradresse ist es möglich, die Trommel mit den Kugelschreibern von einem Programm aus zu drehen und somit die Farben zu wechseln. Diese Adresse ersetzt die COLOR CHANGE Taste an Ihrem Plotter. Sie müssen zunächst das File mit OPEN 2,6,2 oder OPEN nr,6,2 öffnen, wobei das unter OPEN Gesagte und Beschriebene zu beachten ist (nr zwischen 1 und 255). Über PRINT# 2, Farbe kann man dann einen der vier Kugelschreiber anwählen. Sinnvoll für die Farben sind folgende Zahlen:

Schwarzer Stift = 0,4,8,12,16 ..... (0+n*4<255)
Blauer Stift = 1,5,9,13,17 ..... (1+n*4<255)
Grüner Stift = 2,6,10,14,18..... (2+n*4<255)
Roter Stift = 3,7,11,15,19..... (3+n*4<255)

Wählen Sie für die Farbe eine Ziffer über 255, so wird automatisch Schwarz eingestellt. Sicherlich ist es Ihnen schon aufgefallen, daß Schwarz die vom Plotter gewählte Anfangsfarbe ist. Dies ist durch die Konstruktion vorgegeben. Man sieht bei geöffneter Schutzhaube an der Trommel einen Metallstreifen, mit dem der 1520 die Stellung der Trommel überprüfen kann.

Die Farbwahl, welche Sie getroffen haben, können Sie dem 1520 entweder als Zahl in Anführungsstrichen, Normalzahl oder Variable eingeben. Das folgende Beispiel zeigt die Kombinationen der Farben und ihre Reihenfolge an. Sie können den ausgeruckten Zettel als Merkblatt verwenden, wenn Sie die Reihenfolge der Zahlen vergessen sollten.

```
100 REM====1520 SETZEN=================
110 OPEN4,6
120 OPEN2,6,2
130 REM====NAMEN LESEN=================
140 FORF=0TO3
150 READF$(F)
160 NEXT
170 REM====FARBE WAEHLEN/DRUCKEN=======
180 FORF=0TO3
190 PRINT#2,F
200 PRINT#4,F;F$(F)
210 NEXT
220 REM====DATEN=======================
230 DATASCHWARZ,BLAU,GRUEN,ROT
240 REM====ENDE=======================
250 CLOSE2
260 CLOSE4
```

Im ersten Teil des Programms werden die Namen der vier Farben in einem Feld hinterlegt. Der zweite Teil (ab Zeile 180) bringt die Namen mit der jeweiligen Farbe in Zusammenhang und plottet das Ergebnis aus. Als zweites Programm zu dieser Sekundäradresse noch eine Rechenhilfe für die Farbziffern, die ja zwischen 0 und 255 liegen dürfen. Das Programm fragt nach der Ziffer und gibt die dazugehörige Farbe aus. Darunter steht eine Routine, die den Vorgang umkehrt und alle möglichen Zahlen für eine bestimmte Farbe angibt.

```
100 REM====DATEN LESEN==================
110 FORF=0TO3
120 READF$(F)
130 NEXT
140 REM====EINGABE/BERECHNUNG==========
150 INPUT"sKENNZAHL DER FARBE";K$
160 K=VAL(K$)
170 K1=VAL(K$)
180 IFK>255ORK<0THEN150
190 IFK>3THENK=K-4
200 IFK>=4THEN190
210 REM====AUSGABE======================
220 PRINT
230 PRINTK1"= "F$(K)
240 REM====NEUSTART====================
250 FORN=1TO1000
260 NEXT
270 RESTORE
280 GOTO100
290 REM====DATEN=======================
300 DATASCHWARZ,BLAU,GRUEN,ROT
```

```
100 REM====EINGABE====================
110 PRINT"sWELCHE FARBE?"
120 PRINT
130 PRINT"S    B    G    R"
140 PRINT
150 GETA$:WAIT203,63
160 REM====AUSWERTUNG=================
170 IFA$="S"THENK=0:GOTO230
180 IFA$="B"THENK=1:GOTO230
190 IFA$="G"THENK=2:GOTO230
200 IFA$="R"THENK=3:GOTO230
210 GOTO150
220 REM====BERECHNEN==================
230 FORF=0TO63
240 G=K+F*4
250 PRINTG,:
260 NEXT
270 REM====NEUBEGINN==================
280 PRINT
290 PRINT"<TASTE>"
300 WAIT203,63
310 GOTO100

READY.
```

c) bf = 3

Diese Sekundäradresse ist für den Druckbetrieb mit dem eingebauten Zeichensatz von großer Bedeutung. Mit ihr kann man die vier zur Verfügung stehenden Zeichengrößen des 1520 anwählen und somit auch die Zahl der Zeichen in einer Zeile festlegen. Sie öffnen zunächst mit OPEN nr,6,3 den betreffenden Kanal und legen dann mit PRINT# 3, Zahl die Schriftgröße fest. Bei der Zahl beim PRINT#-Befehl gilt übrigens genau dasselbe, was für die Zahlenauswahl bei der Sekundäradresse 2 gesagt wurde. Es dürfen Ziffern zwischen 0 und 255 gewählt werden. Sinnvoll sind jedoch auch hier besonders 0,1,2,3.

Die Zahl darf direkt, in Anführungsstrichen oder in einer Variablen dargestellt werden. Die folgende Tabelle gibt Aufschluß über die Wahl der Zeichengröße:

PRINT # 3,0 ergibt die kleinste Schrift mit 80 Zeichen pro Zeile.

PRINT # 3,1 ergibt die Normalschrift mit 40 Zeichen pro Zeile.

PRINT # 3,2 ergibt eine mittelgroße Schrift mit 20 Zeichen pro Zeile.

PRINT # 3,3 ergibt die größte Schrift mit 10 Zeichen pro Zeile.

Vielleicht haben Sie im Handbuch bei Farbeinstellung oder auch bei der Wahl der Zeichengröße den Begriff Default-Wert bemerkt. Diese Bezeichnung besagt eigentlich nur, wie der Plotter eingestellt ist, wenn Sie ihn einschalten. Das

Betriebssystem setzt ihn auf Normalschrift mit schwarzer Farbe, mehr steckt nicht dahinter. Wollen Sie sich die unterschiedlichen Schriftgrößen ansehen, so geben Sie die folgende, kleine Programmroutine einmal ein.

```
100 REM====1520 SETZEN=================
110 OPEN4,6
120 OPEN3,6,3
130 PRINT#3,0
140 REM====ZEICHENGROESSE==============
150 FORZ=0TO3
160 PRINT#3,Z
170 REM====DRUCKEN=====================
180 PRINT#4,"PLOTTER"
190 Y=X+2
200 IFX<YTHENX=X+1:PRINT#4:GOTO200
230 NEXTZ
240 REM====ENDE========================
250 PRINT#3,0
260 CLOSE3
270 CLOSE4
```

Nach dem Starten des Programms erscheint auf dem Papier dann folgendes Bild.

Da die Handhabung der Zeichenwahl wohl nun jedem klar sein

dürfte, muß noch etwas Theorie für die technisch Interessierten folgen. Haben Sie schon einmal darauf geachtet, wie ein Buchstabe, z.B. das Y, überhaupt gezeichnet wird? Diese Frage und einige andere schriftbildspezifische Probleme legt der folgende Abschnitt dar.

Zuerst zu der Frage, wie einzelne Buchstaben oder Symbole gezeichnet werden.

Das Y beginnt der 1520 von unten her zu zeichnen. Er malt dann nach rechts oben. Dann wird nicht etwa eine Linie doppelt gezogen, um nach links oben zu gelangen, sondern die wirtschaftliche Arbeitsweise wurde bei der Konstruktion besonders beachtet. Das Gerät geht jetzt nämlich den direkten Weg zur linken oberen Ecke, indem es den Kugelschreiber ganz kurz vom Papier abzieht, die entsprechende Position anfährt (der kürzeste Weg zwischen zwei Punkten ist ja die Gerade) uns dan mit gesetztem Stift das Y vervollständigt.

Andere Zeichen, die zwar kompliziert aussehen, aber von der Struktur her einsichtig sind, werden geplottet, ohne den Stift einmal vom Papier zu nehmen. Zu dieser Gattung zählt z.B., auch der Klammeraffe, der von innen nach außen abgearbeitet wird. Bei wieder anderen Zeichen, die sich nicht mit einem Strich auf das Papier bringen lassen, werden auch manche Teile zweifach überfahren, wenn es der kürzeste Weg so vorschreibt.
Ein Beispiel ist das #, von dem erst der linke Teilhaken, die obere Gerade und die rechte Senkrechte in einem Zug erledigt werden. Dann wird der Schreiber vom Papier genommen und die Trommel bewegt sich zum unteren Teil der linken Senkrechten. Von dort aus wird dann das # bis zum rechten

oberen Haken vollendet.
Diese Bewegungsabläufe wurden nur deshalb so genau geschildert, damit man sich als Besitzer eines Plotters anders mit seiner Plottergeschwindigkeit auseinandersetzt. Denn bei dieser Vielzahl von diversen Bewegungsabläufen ist die Geschwindigkeit von 13 Zeichen pro Sekunde bei kleinster Schrift eher verwunderlich als lästig langsam. Es ist hier ein gutes Mittelmaß gefunden worden, um Geschwindigkeit und damit einen gewissen Komfort und die Präzision und Sauberkeit des Schriftbildes gleichermaßen zu gewährleisten, denn eine noch höhere Geschwindigkeit würde das Bild der Schrift verzerrt wiedergeben.
Auch der Zwischenraum benachbart liegender Zeichen ist selbst bei kleinster Schrift mit 80 Zeichen pro Zeile noch so ausgelegt, daß man von Trennschärfe reden kann. Das Gesamtbild verschwimmt auch bei längerer Arbeit mit Kleinschrift nicht. Die Leserlichkeit läßt nichts zu wünschen übrig. Dieses Plotter - Schriftbild zählt zu den wirklichen Stärken des 1520. Dabei soll das Gerät nicht heiliggesprochen werden, da es auch einige Nachteile besitzt, aber man sollte sich vor der Beurteilung immer wieder kurz vor Augen führen, daß der Plotter weder ein Typenrad mit festgelegten Zeichen noch eine Punktmatrix besitzt! Alles wird ja mit dem einfachen Kugelschreiber erledigt, und deshalb muß man bestimmte Kompromisse bezüglich der Arbeitsgeschwindigkeit eingehen.
Soweit zu dem Aufbau des Zeichensatzes mit etwas Theorie. Nun soll es mit den Sekundäradressen weitergehen, um später in die Programmierung des 1520 erfolgreich einzusteigen.

d) bf = 4

Der Plotter 1520 besitzt einige nützliche Sonderfunktionen, von dene eine in der Sekundäradresse 4 enthalten ist. Diese

Funktionen sollen Erleichterungen für bestimmte Programmroutinen mit sich bringen. Über die Sekundäradresse 4 kamm man alle Zeichen des 1520 um 90 Grad verdreht ausgeben lassen. Dies mag auf den ersten Blick überflüssig erscheinen. Es gibt aber doch Probleme, die man mit Hilfe der Zeichendrehung eleganter lösen kann. Dazu zählt z.B. die Beschriftung eines Koordinatensystems, wenn längere Erklärungen zu einer bestimmten Achse nötig sind. Die x-Achse kann dann im normalen Schriftmodus behandelt werden, während man an der y-Achse sinnvollerweise schon eine Drehung der Zeichen vornehmen sollte.

Das Eingabeformat sieht so aus:

```
OPEN nr,6,4
PRINT# 4,Zahl
```

Die Zahl darf, wie üblich, zwischen 0 und 255 liegen, es bieten sich aber auch hier, nicht zuletzt aus Gründen der Übersicht, die Ziffern 0 und 1 an. Der programmierte Standardwert nach dem Einschalten des Gerätes ist die 0. Sie steht also für die Normalschrift. Gibt man hingegen nach dem Öffnen des Files PRINT# 4,1 ein, so hat man auf die Drehung um 90 Grad geschaltet. Alle nun folgenden ASCII-Zeichen werden automatisch um 90 Grad nach rechts gedreht. Wie zuvor bei Farbgebung oder Schriftanwahl bestehen auch bei der Zeichendrehung dieselben Möglichkeiten, was die Übergabe der Zahl an die Sekundäradresse betrifft. Die Zahl darf auch hier direkt in Anführungsstrichen oder in Form einer Variablen übergeben werden.

Die Drehung der ASCII-Zeichen ist übrigens nicht vom Default oder Standardwert der Schrift abhängig. Die Zeichen werden bei allen eingestellten Schriftgrößen gedreht und auch in

der gewählten Größe dargestellt. Die Möglichkeiten der Sekundäradresse 4 sollen an den nachstehenden Beispielen kurz angesprochen werden und zu eigenen Experimenten anregen:

```
100 REM====1520 SETZEN==================
110 OPEN5,6
120 OPEN4,6,4
130 PRINT#4,0
140 REM====DRUCKEN=====================
150 PRINT#5,"PLOTTER 1520"
160 PRINT#4,1
170 PRINT#5,"PLOTTER 1520"
180 REM====ENDE========================
190 PRINT#4,0
200 CLOSE4
210 CLOSE5
```

Dies ist die einfachste Methode, Zeichen an der x-Achse zu drehen, gleichzeitig aber auch die unsauberste. Zum einen liegen die Zeichen zu nahe aneinander, zum anderen muß man, nachdem man sich das Papier um 90 Grad gedreht hat, von unten nach oben lesen, um den "PLOTTER 1520" zu entziffern. Die gerade gezeigte Routine beinhaltet nur das, was allgemein über die Sekundäradresse gesagt wurde und braucht deshalb wohl kaum näher erläutert zu werden. Wir wollen nun vielmehr versuchen, den Vorgang von eben umzudrehen, d.h. ein Programm entwickeln, welches die Zeichen des Textes "PLOTTER 1520" weiterhin an der x-Achse dreht, aber den Text von oben nach unten lesbar machen soll. Sie erinnern sich, daß bei dem letzten Beispiel nach Drehung des Papiers ja von unten nach oben gelesen werden mußte. Die einfachste, aber bei der Erstellung umständlichste Methode ist es, das letzte Programm zu benutzen und den Text einfach verkehrtherum einzugeben, also anstatt "PLOTTER 1520" dann eben "0251 RETTOLP".
Es wird aber wohl jeder der gleichen Ansicht sein und zu dem

gleichen Entschluß über diese Möglichkeit kommen, denn die Nachteile sind unverkennbar.

- Großer Zeitaufwand beim Erstellen
- Viele Fehlermöglichkeiten
- Zeichen stehen zu eng beieinander

Man muß sich also schon etwas mehr Mühe geben, will man die genannten Nachteile wirkungsvoll umgehen. Das folgende kleine Programm zeigt eine der Möglichkeiten, das genannte Problem zu lösen.

```
100 REM====1520 SETZEN=================
110 OPEN5,6
120 OPEN4,6,4
130 OPEN1,6,1
140 PRINT#4,1
150 REM====ZEICHEN BERECHNEN===========
160 X$="PLOTTER 1520"
170 FORX=LEN(X$)TO1STEP-1
180 Y$=MID$(X$,X,1)
190 REM====ZEICHEN DRUCKEN=============
200 PRINT#1,"M",20*(LEN(X$)-X);0
210 PRINT#5,Y$;:
220 NEXT
230 REM====ENDE========================
240 PRINT#4,0
250 PRINT#1,"M";0;0
260 CLOSE1
270 CLOSE4
280 CLOSE5
```

Zum Verständnis des Programms eine kurze Analyse der entscheidenden Zeichen 150-220.

Zuerst wird der Text, in unserem Fall wieder "PLOTTER 1520" in der Variablen X$ hinterlegt (Zeile 160). In Zeile 170 beginnt eine FOR - NEXT - Schleife, die beim Ablauf des Programms alle Stellungen der Zeichen von hinten nach vorne durchgeht. Damit auch die Stellung des letzten Zeichens getroffen wird, muß, um die Textlänge nicht immer abzuzählen, mit LEN (X$) gearbeitet werden. Wir hätten für die Zeile 170 aber auch

FORX=12 TO 1 STEP -1

schreiben dürfen, wenn wir die Buchstaben vorher zählen.

In Y$ / (Zeile 180) wird mit dem MID$ - Befehl jeweils ein Zeichen aus X$ gelesen, und zwar von hinten nach vorne, da Zeile 170 ja heruntergezählt. Zeile 200 bringt die Trommel mit den Kugelschreibern auf die richtige Position; dies wird mit dem bereits bekannten M - Befehl erledigt. Die Stellung auf der x-Achsse läßt sich leicht berechnen. Sie sehen dort den Ausdruck 20-(LEN(X$)-X).
Vielleicht zuerst einen Blick in den Klammerausdruck; LEN(X$) verkörpert nichts weiter als die Zeichenanzahl von X$. Da wir von links nach rechts schreiben, muß der Abstand zum absoluten Nullpunkt 0/0 folglich größer werden.
Dies erreicht man in diesem Beispiel dadurch, daß man zunächst X von LEN(X$) subtrahiert und mit einer Konstanten (hier 20) multipliziert. Da X kleiner wird (siehe Zeile 170 ...STEP -1) wird die Differenz kleiner und der Faktor somit größer. Die Trommel rückt immer weiter von der rechten Kante der Walze ab. In Zeile 210 wird unser Buchstabe dann zu Papier gebracht. Hier ist zu beachten, daß das Semikolon nicht vergessen wird, da sonst die Schrift diagonal erscheint! Der NEXT-Aufruf beginnt die Schleife in Zeile 170 erneut.
Die Programmanalysen werden deshalb so detailliert gegeben, da gerade diese Ausführungen für die Plotter-Anfänger gedacht sind. Eine ähnliche Routine, die in etwa den gleichen Lösungsweg beschreitet, findet sich nämlich auch im Plotter 1520 Handbuch. Dort wird keine weitere Erklärung zu den Programmschritten gegeben, die besonders für den Neuling wichtig wären.
Damit ist die Zeichendrehung in Richtung der x-Achse erledigt und wir können uns der y-Achse zuwenden. Sollten Sie die letzte Routine abgetippt haben, so löschen Sie diese bitte noch nicht. Viele Zeilen können direkt übernommen werden.

```
100 REM====1520 SETZEN=================
110 OPEN5,6
120 OPEN4,6,4
130 OPEN1,6,1
140 PRINT#4,1
150 REM====ZEICHEN BERECHNEN===========
160 X$="PLOTTER 1520"
170 FORX=1TOLEN(X$)
180 Y$=MID$(X$,X,1)
190 REM====ZEICHEN DRUCKEN=============
200 PRINT#5,Y$
210 NEXT
220 REM====ENDE=======================
230 PRINT#4,0
240 PRINT#5
250 CLOSE1
260 CLOSE4
270 CLOSE5
```

Von Zeile 150 bis 210 steht nun der Programmteil, der die Buchstaben des Wortes "PLOTTER 1520" einzeln liest und druckt. Jetzt ist es einfach möglich, das Wort von vorne nach hinten auszulesen und die Buchstaben untereinander zu drucken. Eine nähere Analyse erübrigt sich. Vielleicht lesen Sie nochmals den vorherigen Abschnitt durch.
Sie werden aber später nochmals mit der Zeichendrehung konfrontiert werden, wenn es bei den Statistikfunktionen um das Säulendiagramm gehen wird. Die Möglichkeiten zur Nutzung dieser Funktion sind also durchaus gegeben, wenn auch die Effektivität der Sekundäradressen zunächst, bei ungenauem Hinsehen, nicht gegeben zu sein scheint.

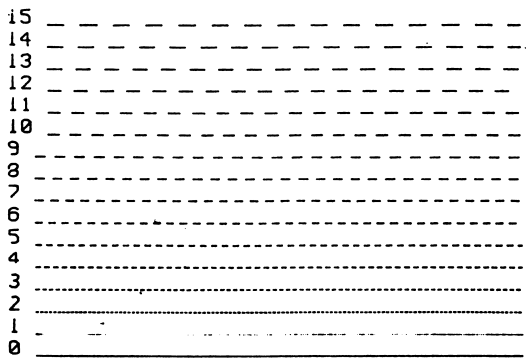
e) bf = 5

Eine der nützlichen Funktionen des Betriebssystems des 1520 ist die Linierfunktion. Sie läßt sich mit der Sekundäradresse 5 abrufen und beinhaltet eine Kapazität von insgesamt 16 Linienarten. Diese Linien reichen von der in einem Strich gezogenen Geraden bis hin zur groß gestrichelten Linie. Wie Sie es jetzt bereits von den

anderen Adressen kennen, öffenen Sie zunächst das File mit OPEN nr,6,5.

Dann folgt die PRINT#5, Zahl Anweisung, wobei die Zahl wieder im Bereich 0-255 liegen darf. Sie können auch hier die Kennzahl direkt, als Variablenausdruck oder als Ziffer in Anführungsstriche setzen. Sinnvoll sind hier besonders die Nummern 0-15.

Der Wert beim Einschalten des 1520 ist 0, die Standardlinie. Diese wird nicht gestrichelt, sondern als Gerade gezogen. Ersetzen Sie dann die 0 durch eine Zahl von 1-15, so wird der gewünschte Typ ausgedruckt. Sehen Sie sich nun erst einmal alle möglichen Linien an. Danach dann ein Programm, mit dem man einen Typ auswählen und drucken lassen kann.



```
100 REM====1520 SETZEN=================
110 OPEN4,6
120 OPEN5,6,5
130 OPEN1,6,1
140 PRINT#5,0
150 REM====EINGABE====================
160 INPUT"sWELCHE LINIENNUMMER";A$
170 A=VAL(A$)
180 IFA<0ORA>15THEN160
190 PRINT
200 INPUT"LAENGE IN CM (MAX. 8 CM";B$
210 B=VAL(B$)
220 IFB<0ORB>8THENPRINT"qq":GOTO200
230 REM====ZEICHNUNG==================
240 L=50*(B+1)
250 PRINT#4,A;
260 PRINT#5,A;
270 PRINT#1,"D";L;0
280 PRINT#5,0
290 PRINT#4
300 REM====ENDE=======================
310 CLOSE1
320 CLOSE5
330 CLOSE4
```

Die Zeilen 100 - 140 setzen den Plotter auf den Linien-Standardwert und öffnen die nötigen Files. Bei der INPUT-Eingabe in Zeile 160 taucht ein Zeichen auf, welches bisher noch neu ist. Es handelt sich um das revers dargestellte Herzchen, welches ja erscheint, wenn man SHIFT/HOME drückt. Daß der 1520 keine reversen Zeichen darstellen kann, (dazu auch später noch etwas), behilft er sich damit, die betreffenden Buchstaben zu unterstreichen. Das s steht also für SHIFT/HOME. Den gesamten Sonderzeichensatz finden Sie im Anschluß an die Sekundäradressen. Nachdem Sie bei dem Programm Ihre Eingaben gemacht haben, beginnt der Teil des Programms, der die Linie ausdruckt. (ZEILE 240-290). Dazu muß die eingegebene Zentimeterzahl erst in Plotter-Längeneinheiten umgewandelt werden; dies geschieht in Zeile 240. In Zeile 250 wird die Linienkennzahl vor der Linie gedruckt, Zeile 260 stellt die

Sekundäradresse 5 auf den richtigen Wert ein. Die eigentliche Linie entsteht mit dem bekannten D-Befehl und wird bis zur berechneten x-Koordinate L gezeichnet. Messen Sie die gewählten Linien einmal nach. der Fehler ist minimal und die Schrittweite von 0.2 mm wird somit sehr genau eingehalten. Die Linierfunktion hat unbegrenzte Einsatzmöglichkeiten, da auch komplette Figuren damit erstellt werden können.

```
100 REM====1520 SETZEN=================
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#5,5
140 REM====ZEICHNEN====================
150 PRINT#1,"M";100;0
160 PRINT#1,"I"
170 PRINT#1,"J";200;0
180 PRINT#1,"J";100;100
190 PRINT#1,"J";0;0
200 PRINT#1,"R";-100;-100
210 REM====ENDE========================
220 PRINT#5,0
230 CLOSE1
240 CLOSE5
```

Dazu greifen wir auf eine bereits konstruierte Figur zurück, das Dreieck nämlich, anhand dessen die absoluten und relativen Zeichenbefehle gezeigt wurden. Die Zeichenroutine ist in Bezug auf die Position und die Seitenlängen des Dreiecks identisch, nur wurde mit den Zeilen 120 und 130 die Sekundäradresse 5 angesprochen und mit PRINT# 5,5 eine Linienart gewählt. Das Dreieck wird nun gestrichelt dargestellt. Diese Art zu zeichnen ist insbesondere dann von Vorteil, wenn man in einer Statistik sehr viele Geraden (Messwerte) zu verarbeiten hat und die vier Farben als Unterscheidungsmerkmal nicht mehr ausreichen; Sie können im Notfall 64 Kurven oder Geraden zeichnen, die sich alle noch voneinander unterscheiden lassen, denn es stehen pro Farbe

jeweils 16 Linientypen zur Wahl. Diese Kapazität wird wohl auch dem größten Statistiker genügen?

f) bf = 6

Es ist Ihnen sicherlich schon aufgefallen, daß der Plotter, auch wenn der C64 auf Kleinschrift umgeschaltet ist, seine eigenen Gesetze bezüglich der Groß-/Kleinschrift zu haben scheint.
Dies ist notwendig, da der 1520 über keinen Grafiksatz verfügt. Sie können das sehen, wenn Sie versuchen, im Großschriftmodus des Computers das Wort PLOTTER 1520 geshiftet drucken zu lassen.

Es erscheinen nun nicht etwa die (vielleicht) erwarteten Grafikzeichen, die als zweite Funktionen auf den entsprechenden Tasten liegen, sondern die geshifteten Buchstaben werden in Kleinschrift ausgegeben, die Normalen in Großschrift. Daran wird sich auch nichts ändern, wenn man den Computer mit PRINTCHR$(14) auf Kleinschriftmodus schaltet. Auf dem Bildschirm erscheint nun zwar Kleinschrift und über SHIFT Großschrift, aber der Plotter macht weiterhin genau das Gegenteil. Dies liegt daran, daß der 1520 die Umschaltung nicht registiert und diese somit von seinem Operationssystem nicht ausgewertet werden kann. Die Sekundäradresse 6 macht es dem Programmierer nun möglich, dem Gerät seine Absicht mitzuteilen. Dabei wird nach dem gerade Gesagten unter 2 Zuständen differenziert.

1) Der Großschriftmodus mit Kleinschrift über SHIFT = 0

2) Der Kleinschriftmodus mit Großschrift über SHIFT = 1

Das Eingabeformat lautet wie folgt: Zuerst muß das File mit

OPEN nr,6,6 geöffnet werden, dann erfolgt die Eingabe PRINT# 6, Zahl, wobei die 0 und die 1 sinnvoll sind. Geben Sie z.B. PRINT# 6,1 ein, so sind Sie sofort im Kleinschriftmodus und können über SHIFT die Großschrift erreichen.

Man arbeitet so praktisch wie mit einer Schreibmaschine. Dabei ist es dem Plotter aber nun wiederum gleichgültig, ob der Computer im Groß- oder Kleinschriftmodus arbeitet, das geshiftete Zeichen wird, egal ob es auf dem Bildschirm als Grafiksymbol oder als Buchstabe erscheint (modusabhängig!), als ein Großbuchstabe zu Papier gebracht. Der programmierte Startardwert ist der, den man mit PRINT# 6,0 wieder erreichen kann, d.h. nach dem Einschalten werden geshiftete Buchstaben in kleiner Schrift gedruckt.

Um den Unterschied der beiden Zustände zu verdeutlichen, soll mit dem folgenden Programm der Zeichensatz ausgelistet werden. Es ist ratsam, vor dem eintippen der Routine den C64 auf Kleinschrift umzuschalten, da sonst geshiftete Buchstaben als Grafiksymbol nicht immer gleich erkannt werden. Auch das Listing wurde im Kleinschriftmodus (1) erstellt.

```
100 rem====1520 setzen==================
110 open4,6
120 open6,6,6
130 print#6,0
140 rem====text=========================
150 print#4,"6,0 = normalmodus"
160 gosub270
170 print#6,1
180 gosub270
190 print#4,"6,1 = Kleinschriftmodus"
200 gosub270
210 print#6,0
220 gosub270
230 rem====ende=========================
240 close6
250 close4:end
260 rem====plotten======================
270 forx=1to35
280 print#4,chr$(90-x);:
290 next
300 print#4
310 return
```

**Nachdem das Programm gestartet wurde, erhält man folgenden Ausdruck:**

```
6,0 = NORMALMODUS
YXWVUTSRQPONMLKJIHGFEDCBA@?>=<;:987
yxwvutsrqponmlkjihgfedcba—?>=<;:987
6,1 = Kleinschriftmodus
yxwvutsrqponmlkjihgfedcba—?>=<;:987
YXWVUTSRQPONMLKJIHGFEDCBA@?>=<;:987
```

**Das Ergebnis des Programms bestätigt nochmals die oben ausgeführten Zusammenhänge. Im Normalmodus wird zuerst die Großschrift und geshiftet dann die Kleinschrift ausgedruckt, während es sich in Kleinschriftmodus genau entgegengesetzt verhält. Wir benötigen etwas später, bei dem Schreibmaschinenprogramm im Direktmodus, diese Sekundäradresse.**

g) bf= 7

Wir sind mittlerweile schon bei der letzten Sekundäradresse, der Nummer 7 angelangt. Diese läßt sich sehr kurz behandeln, da man diese auch nur selten benötigt. Geben Sie OPEN 7,6,7 und anschließend PRINT# 7 ein. Sofort darauf wird der Selbsttest durchgeführt, den der 1520 auch direkt nach dem Einschalten automatisch vornimmt. Wird nun die Sekundärdresse 7 angesprochen, so wird das Gerät in den Zustand versetzt, in dem es sich direkt nach dem Anstellen befindet. Dies bedeutet

- Selbsttest

- Rücksetzung alle Sekundäradressen auf den Normalwert

- Löschung der im Arbeitsspeicher befindlichen Daten

Nach den Darlegungen der letzten Abschnitte kann nun bald in die Programmierung eingestiegen werden, da jetzt der Grundstock gelegt ist, so daß auch ein Neuling die Programme verstehen und analysieren kann, denn die gesamten Darstellungen können nur einen Anstoß geben, selber Routinen nach speziellen, eigenen Wünschen zu entwerfen. Es sei hier nochmals betont, daß ein Anspruch auf Vollständigkeit bei Programmbeispielen nicht bestehen kann. Der 1520 ist so extrem vielseitig einsetzbar, daß man allein ein Buch mit technischen Graphen füllen könnte. Dies ist aber nicht der Sinn der Sache, denn es soll für jedermann etwas dabeisein. Darum wird versucht, im zur Verfügung stehenden Rahmen möglichst viele Themenbereiche anzuschneiden, und Sie zu eigenen Experimenten anzuregen. Dies ist auch der Grund, aus dem manche geometrischen Figuren oder Routinen ausführlich analysiert werden; dies geschieht keinesfalls, um Sie zu langweilen, sondern vielmehr, um Verständnishilfen zu geben.

Den aufmerksamen Leser wird es gewundert haben, daß mit der Sekundäradresse 1 in der Besprechung begonnen wurde. Dies liegt nur darin begründet, daß die Adresse 0 für den Ausdruck von ASCII-Zeichen verantwortlich ist und somit im Zusammenhang mit dem Zeichensatz besprochen werden sollte, um Unterschiede zur Bildschirm oder Druckerausgabe aufzuzeigen. Betrachten wir zuerst den Normalmodus des 1520. Die beiden folgenden Tabellen demonstrieren einen Teil des Zeichensatzes, einmal normal und dann geshiftet.

| | | | |
|---|---|---|---|
| $ | % | & | ' |
| ( | ) | * | + |
| , | - | . | / |
| 0 | 1 | 2 | 3 |
| 4 | 5 | 6 | 7 |
| 8 | 9 | : | ; |
| < | = | > | ? |
| @ | A | B | C |
| D | E | F | G |
| H | I | J | K |
| L | M | N | O |
| P | Q | R | S |
| T | U | V | W |
| X | Y | Z | [ |
| £ | ] | ↑ | ← |

| | | | |
|---|---|---|---|
| $ | % | & | ' |
| ( | ) | * | + |
| , | - | . | / |
| 0 | 1 | 2 | 3 |
| 4 | 5 | 6 | 7 |
| 8 | 9 | : | ; |
| < | = | > | ? |
| — | a | b | c |
| d | e | f | g |
| h | i | j | k |
| l | m | n | o |
| p | q | r | s |
| t | u | v | w |
| x | y | z | ▌ |
| ▁ | ▵ | π | □ |

Man erkennt, daß von CHR$(33) bis CHR$(95) der Zeichensatz mit den Bildschirmausgaben übereinstimmt. Bei geshifteten Buchstaben, die auf dem Bildschirm ab CHR$(161) als Grafiksymbole erscheinen, wird im Plotter-Normalmodus Kleinschrift ausgegeben. Schaltet man mit PRINT# 6,1 in den Kleinschriftmodus um, so sieht das Plotterlisting genau andersherum aus. Hier gilt ab CHR$ (33) der untere Tabellenausdruck, der bei Großschrift noch vor CHR$(161) lag.
Soweit ist also in der Form der Ausgabe kein großer Unterschied festzustellen gewesen. Aber was ist mit den Steuerzeichen, die auf Bildschirm oder Drucker als reserve Zeichen erscheinen?
Der C64 besitzt ja sehr viele solcher Symbole, angefangen mit der Cursorsteuerung bis hin zur Farbgebung der Zeichen. Wir wissen nämlich, daß der 1520 aus Zeitgründen keine reversen Zeichen darstellen kann, denn das genaue Ausmalen eines Kästchens würde viel zu lange dauern. Was ist aber nun, wenn in einem Programm diese Sonderzeichen benutzt werden, anstatt alles mit CHR$(X)-Werten zu schreiben, da dieses wesentlich länger dauert?

Man hat sich bei der Konstruktion also etwas anderes einfallen lassen. Die reversen Zeichen werden als der Buchstabe gedruckt, der als Grundfunktion auf der betreffenden Taste steht (bei Cursor down das Q). Dann wird dieses Zeichen unterstrichen, damit man beim Lesen des Listings das Zeichen als reverses Zeichen identifizieren kann. Um sich an die Plotterschreibweise zu gewöhnen, folgt nun ein Teil mit Tabellen, die den Unterschied zwischen Drucker und Plotter 1520 bei der Ausgabe der reversen Zeichen darstellen. Dabei wird hinter jedem Symbol kurz die Tastenfolge beschrieben, mit der man die betreffenden Sonderzeichen erreicht. Dabei wird auch der

**Kleinschriftmodus beachtet, da dort die Anwahl der Zeichen etwas anders aussieht.**

| "sonderzeichen" | "SONDERZEICHEN" | "SONDERZEICHEN" |
|---|---|---|
| "E = funktionstaste | "E = FUNKTIONSTASTE | "■ = FUNKTIONSTASTE 1" |
| "I = funktionstaste | "I = FUNKTIONSTASTE | "■ = FUNKTIONSTASTE 2" |
| "F = funktionstaste | "F = FUNKTIONSTASTE | "■ = FUNKTIONSTASTE 3" |
| "J = funktionstaste | "J = FUNKTIONSTASTE | "■ = FUNKTIONSTASTE 4" |
| "G = funktionstaste | "G = FUNKTIONSTASTE | "■ = FUNKTIONSTASTE 5" |
| "K = funktionstaste | "K = FUNKTIONSTASTE | "■ = FUNKTIONSTASTE 6" |
| "H = funktionstaste | "H = FUNKTIONSTASTE | "■ = FUNKTIONSTASTE 7" |
| "L = funktionstaste | "L = FUNKTIONSTASTE | "■ = FUNKTIONSTASTE 8" |
| "T = delay" | "T = DELAY" | "■ = DELAY" |
| "S = home" | "S = HOME" | "■ = HOME" |
| "S = clear" | "S = CLEAR" | "■ = CLEAR" |
| "Q = cursor ab" | "Q = CURSOR AB" | "■ = CURSOR AB" |
| "Q = cursor auf" | "Q = CURSOR AUF" | "■ = CURSOR AUF" |
| "▲ = cursor rechts" | "] = CURSOR RECHTS" | "■ = CURSOR RECHTS" |
| "] = cursor links" | "▲ = CURSOR LINKS" | "■ = CURSOR LINKS" |
| "P = control/blk | "P = CONTROL/BLK | "■ = CONTROL/BLK |
| "P = control/wht | "F = CONTROL/WHT | "■ = CONTROL/WHT |
| "_ = control/red | "↓ = CONTROL/RED | "■ = CONTROL/RED |
| "← = control/cyn | "□ = CONTROL/CYN | "■ = CONTROL/CYN |
| "↓ = control/pur | "_ = CONTROL/PUR | "■ = CONTROL/PUR |
| "π = control/grn | "↑ = CONTROL/GRN | "■ = CONTROL/GRN |
| "□ = control/blu | "← = CONTROL/BLU | "■ = CONTROL/BLU |
| "↑ = control/yel | "π = CONTROL/YEL | "■ = CONTROL/YEL |
| "R = control/rvs on" | "R = CONTROL/RVS ON" | "■ = CONTROL/RVS ON" |
| "R = control/rvs off | "F = CONTROL/RVS OFF | "■ = CONTROL/RVS OFF" |
| "A = c=/blk | "A = C=/BLK | "■ = C=/BLK |
| "U = c=/wht | "U = C=/WHT | "■ = C=/WHT |
| "V = c=/red | "V = C=/RED | "■ = C=/RED |
| "W = c=/cyn | "W = C=/CYN | "■ = C=/CYN |
| "X = c=/pur | "X = C=/PUR | "■ = C=/PUR |
| "Y = c=/grn | "Y = C=/GRN | "■ = C=/GRN |
| "Z = c=/blu | "Z = C=/BLU | "■ = C=/BLU |
| "F = c=/yel | "↑ = C=/YEL | "■ = C=/YEL |

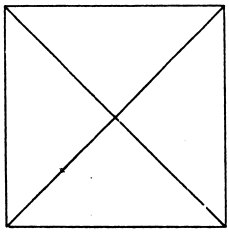
**Bitte behalten Sie einige der Symbole parat, da bei den nun folgenden Programmen teilweise mit ihnen gearbeitet wird.**

## *8.3 Konstruktion von Figuren aller Art*

Nachdem die Theorie des Plotters 1520 soweit wie nötig dargestellt wurde, ist es nun Zeit, sich an die Praxis zu begeben, um die Theorie sinnvoll anzuwenden. Im folgenden soll es zunächst um die Konstruktion von einfachen geometrischen Figuren gehen, beginnend mit dem Rechteck bis hin zu Kreis oder Ellipse. Dabei soll durchaus Wert auf mathematische Ansprüche gelegt werden, so werden Höhnen oder Radien eingezeichnet, die Einheiten werden so weit wie es möglich ist in Zentimetern gewählt. Bei den Programmen wird darauf geachtet werden daß eine kurze Erklärung erfolgt und nach dem Listing auch ein Bild des Werkes erscheint, damit man sich beim Nichtfunktionieren der eigenen Eingaben besser auf Fehlersuche begeben kann.

### *8.3.1 Quadrat*

```
100 REM====QUADRAT=====================
110 OPEN1,6,1
120 OPEN2,6,2
130 PRINT#2,1
140 REM====ZEICHNEN====================
150 PRINT#1,"M";100;-50
160 PRINT#1,"I"
170 S=200
180 PRINT#1,"J";S;0
190 PRINT#1,"J";S;S
200 PRINT#1,"J";0;S
210 PRINT#1,"J";0;0
220 REM====DIAGONALEN==================
230 PRINT#2,3
240 PRINT#1,"J";S;S
250 PRINT#1,"R";S;0
260 PRINT#1,"J";0;S
270 REM====ENDE========================
280 PRINT#1,"R";-100;-100
310 CLOSE1
320 CLOSE2
```

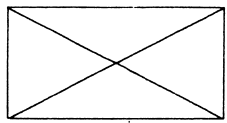


Wie Sie aus dem Listing ersehen, ist die Konstruktion eines Quadrates sehr einfach, zumal auch alle Seiten gleich lang sind. In Zeile 170 wird die Seitenlänge S festgelegt, von Zeile 180 bis 210 wird das eigentliche Quadrat gezeichnet. Dann folgen die beiden Diagonalen, die auch hier gleich lang sind. Dieses Beispiel wendet nur das zuvor Besprochene in der ursprünglichen Befehlsform an und dient nur der kurzen Wiederholung. Dieses soll auch das nun folgende Rechteck tun, bevor wir uns an etwas schwierigere Figuren wenden und damit auch eine andere Programmierweise einführen müssen. Dazu aber gleich mehr.

### *8.3.2 Rechteck*

Zuerst das Listing eines Rechtecks mit Diagonalen:

```
100 REM====RECHTECK====================
110 OPEN1,6,1
120 OPEN2,6,2
130 PRINT#2,1
140 REM====ZEICHNEN====================
150 PRINT#1,"M";100;-50
160 PRINT#1,"I"
170 S=200:H=100
180 PRINT#1,"J";S;0
190 PRINT#1,"J";S;H
200 PRINT#1,"J";0;H
210 PRINT#1,"J";0;0
220 REM====DIAGONALEN==================
230 PRINT#2,3
240 PRINT#1,"J";S;H
250 PRINT#1,"R";S;0
260 PRINT#1,"J";0;H
270 REM====ENDE========================
280 PRINT#1,"R";-100;-100
310 CLOSE1
320 CLOSE2

READY.
```



Wenn Sie das Rechteckprogramm mit dem des Quadrates vergleichen, so können Sie feststellen, daß kaum Unterschiede auszumachen sind; in Zeile 170 wurde lediglich eine zweite Variable für eine andere Kantenlänge eingeführt und die Zeichenroutine dementsprechend umgestellt. Diese Beispiele sind aber so einfach, daß hier eine Analyse des Programmablaufes in allen Details noch nicht sinnvoll erscheint.

Wir wollen jetzt versuchen, eine räumliche Figur zu konstruieren, beginnend mit der einfachsten, dem Quader. Bevor aber mit dem Zeichnen schwieriger Figuren begonnen

werden kann, muß die Art des Programmierens unbedingt abgeändert werden, um den Aufwand beim Eintippen zu verringern. Dabei muß trotzdem beachtet werden, daß man einen geeigneten Mittelweg findet, damit die Struktur und damit die Übersichtlichkeit eines Programms noch gegeben ist, aber die Arbeitszeit gesenkt wird.
Um diesen geeigneten Mittelweg ausfindig zu machen, wird am Beispiel des Quaders einmal ausführlich gezeigt, wie Sie das Programm kürzen können, ohne dabei die Übersicht zu verlieren. Als erstes ein Beispiel zur Lösung des Quaderproblems, wie man es nach Möglichkeit nicht tun sollte, wenn man Zeit sparen möchte und das Programmieren Freude machen soll.

```
100 REM====1520 SETZEN=================
110 OPEN1,6,1
120 OPEN2,6,2
130 OPEN5,6,5:PRINT#5,0
140 REM====ZEICHNEN====================
150 X=100:Y=-50:S=200
160 PRINT#1,"M";X;Y
170 PRINT#1,"I"
180 PRINT#1,"J";S;0
190 PRINT#1,"J";S;S
200 PRINT#1,"J";0;S
210 PRINT#1,"J";0;0
220 X=70:Y=70
230 PRINT#1,"R";S;0
240 PRINT#1,"I";
250 PRINT#1,"J";X;Y
260 PRINT#1,"J";X;Y+S
270 PRINT#1,"J";0;S
280 PRINT#1,"R";-S;S
290 PRINT#1,"J";-S+X;Y+S
300 PRINT#1,"J";X;Y+S
310 REM====STRICHELN===================
320 PRINT#5,7
330 PRINT#1,"R";X;Y
340 PRINT#1,"J";-S+X;Y
350 PRINT#1,"J";-S;0
360 PRINT#1,"R";-S+X;Y+S
370 PRINT#1,"J";-S+X;Y
380 REM====ENDE=======================
390 PRINT#1,"M";0;-100
400 PRINT#5,0
410 CLOSE1
420 CLOSE2
430 CLOSE5
```

Hier wurden in den Zeilen 150 und 220 die in Frage kommenden Kantenstücke der Figur in der Figur in den Variablen x,y und s hinterlegt. Dann wird der Quader mit jeweils einem PRINT# 1, "Befehl", Koordinatenvariablen gezeichnet bzw. gestrichelt. Dies ist durch die Variablen, die die Koordinaten darstellen, schon eine schlechte Eigenschaft des Programms, denn Sie müssen die Positionen erst im Kopf errechnen, wollen Sie die Routine nachvollziehen. Zum anderen stört der ungeheuere Aufwand mit den PRINT#-Anweisungen, und auch die unökonomische Fahrweise der

Kugelschreibertrommel wird manchem Leser komisch erscheinen. Das Programm dient aber, wie bereits gesagt, der Abschreckung, und es soll nun versucht werden, das Programm in zwei Schritten zu verkürzen, so daß eine kleine Routine verbleibt, die aber dennoch soweit einsichtig ist, daß man das Programm nachvollziehen kann.
Der erste Schritt zum Erfolg ist nun der, daß wir die PRINT # - Befehle reduzieren und die Fahrweise der Trommel dahingehend optimieren, daß der kürzeste Weg gegangen wird. Dazu wird der relative Nullpunkt mit "I" auf die linke obere Ecke der Vorderansicht (100/200) verlegt, da dann der äußere Teil der Figur in einem Zuge gezeichnet werden kann und man mit dem Stift keine Umwege gehen braucht.
Die Koordinaten werden bezüglich des relativen Nullpunktes direkt, also ohne Variablenausdruck, in DATA - Zeilen zusammengefaßt und dann mit READ eingelesen.
Auch die Befehle des 1520 können in Data - Zeilen vereint werden (M,D,J,H). Wenden wir diese Maßnahmen auf das vorige Programm an, so erhalten wir als zweiten Schritt folgendes Listing:

```
100 REM====1520 SETZEN=================
110 OPEN5,6,5
120 OPEN1,6,1
130 PRINT#5,0
140 PRINT#1,"M";100;200
150 PRINT#1,"I"
160 REM====DATEN LESEN/ZEICHNEN========
170 FORX=0TO7
180 READY,Z
190 PRINT#1,"J";Y;Z
200 NEXT
210 REM====DATEN LESEN/ZEICHNEN========
220 FORX=0TO7
230 READX$,Y,Z
240 IFX=2THENPRINT#5,7
250 PRINT#1,X$,Y,Z
260 NEXT
270 REM====ENDE========================
280 PRINT#5,0
290 CLOSE1
300 CLOSE5
310 REM====DATEN (ZEICHNEN)============
320 DATA0,-200,200,-200,270,-130
330 DATA270,70,70,70,0,0,200,0
340 DATA270,70
350 REM====DATEN (ZEICHNEN&STRICHELN)==
360 DATAR,200,0,J,200,-200,R,270,-130
370 DATAJ,70,-130,J,0,-200,R,70,-130
380 DATAJ,70,70,M,0,-100
```

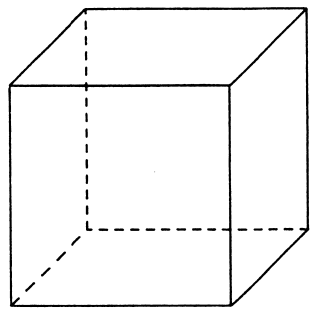
Haben Sie diese Version eingegeben, so starten Sie das Programm mit RUN. Sie werden genau dasselbe Bild erhalten, wie zuvor. Das Programm ist aber trotz einer ausführlichen REM-Dokumentation schon kürzer als das letzte. Auch die Zeichenzeit ist geringer, da der Zeichenweg verändert wurde. Von Zeile 170-260 sehen Sie nun noch zwei FOR NEXT - Schleifen, die beim Dateneinlesen noch zwischen Stricheln und Zeichnen differenzieren.

Dies ist, wie gesagt auch nur ein Zwischenschritt, der den Konstruktionsablauf einer Routine unterstreichen soll. Haben Sie den ersten Schritt verstanden, so können Sie auch den nächsten betrachten; ansonsten sollte zurückgeblättert

werden, da sehr viele Programme in den weiteren Ausführungen DATA-Zeilen enthalten.

Auf die kürzeste Form gebracht, mit der sich der Aspekt der Verständlichkeit und Übersichtlichkeit noch rechtfertigen läßt, kommt man, wenn man beide Leseschleifen zu einer einzigen zusammenfaßt und nach Erreichen eines bestimmten Wertes auf Linierung umschaltet. (Zeile 190). Die DATA-Befehle enthalten nun den gesamten Quader.

```
100 REM====1520 SETZEN=================
110 OPEN5,6,5
120 OPEN1,6,1
130 PRINT#5,0
140 PRINT#1,"M";100;200
150 PRINT#1,"I"
160 REM====DATEN LESEN/ZEICHNEN========
170 FORX=0TO15
180 READX$,Y,Z
190 IFX=10THENPRINT#5,7
200 PRINT#1,X$,Y,Z
210 NEXT
220 REM====ENDE========================
230 PRINT#5,0
240 CLOSE1
250 CLOSE5
260 REM====DATEN GESAMT================
270 DATAJ,,-200,J,200,-200,J,270,-130
280 DATAJ,270,70,J,70,70,J,,,J,200,0
290 DATAJ,270,70,R,200,,J,200,-200
300 DATAR,270,-130,J,70,-130,J,,-200
310 DATA R,70,-130,J,70,70,M,,-100

READY.
```



**Vergleichen Sie jetzt einmal dieses Listing mit dem ersten Durchgang; man kann hier sehen, daß zu einem Programm mehr als nur die Idee gehört. Man muß auch versuchen, eine wirtschaftliche Programmierweise zu entwickeln, um die Schnelligkeit des 1520 zu nutzen. Dabei kann man das letzte**

Listing wohl noch als verständlich gelten lassen, da die Koordinaten direkt in den DATA-Zeilen stehen.

Zusammenfassend kann man zur Schreibweise einer 1520 Routine die nachstehenden Punkte anführen:

- Genaue Planung der Figur (Zeichnung!)
- Auf dem Papier den kürzesten Arbeitsweg für den Plotter ermitteln
- Unnötig viele PRINT#-Anweisungen vermeiden (DATA-Zeilen!)
- Lesbarkeit bewahren (REM-Zeilen und eine gewisse Struktur)

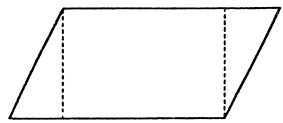
Beachten Sie diese wenigen Punkte bei der Entwicklung von Zeichenprogrammen, so werden Sie zum einen mehr Freude am Programmieren haben, und der Plotter kann seine Möglichkeiten voll zur Entfaltung bringen. Die drei letzten Programme können bei Problemen den Weg vom Entwurf zum Endprodukt dokumentieren.

Die folgenden Programme werden sich ausschließlich auf die letzten beiden Programmiermöglichkeiten stützen, also eine Unterteilung in zwei Einleseschleifen für verschiedene Funktionen, oder die Zusammenfassung zu einem DATA-Block.

Nach den Erklärungen zur weiteren Programmierung jetzt zwei Beispiele von Figuren, die einfach zu konstruieren sind und in der Geometrie einen festen Platz haben, das Parallelogramm und die Raute. Auch hierbei sollen wieder Sonderstrecken wie Diagonalen oder Höhen gestrichelt eingezeichnet werden.

### *8.3.3 Parallelogramm*

```
100 REM====PARALLELOGRAMM==============
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#5,0
140 REM====ANFANGSPOSITION=============
150 PRINT#1,"M";100;0
160 PRINT#1,"I"
170 REM====PARALLELOGRAMM ZEICHNEN=====
180 FORX=0TO3
190 READY,Z
200 PRINT#1,"J";Y;Z
210 NEXT
220 REM====HOEHEN ZEICHNEN=============
230 PRINT#5,3
240 FORX=0TO4
250 READX$,Y,Z
260 PRINT#1,X$,Y,Z
270 NEXT
280 REM====ENDE========================
290 PRINT#5,0
300 CLOSE1
310 CLOSE5
320 REM====DATEN PARALLELOGRAMM========
330 DATA200,0,250,100,50,100,0,0
340 REM====DATEN HOEHEN================
350 DATAR,50,0,J,50,100,R,200,100
360 DATAJ,200,0,M,0,-100

READY.
```

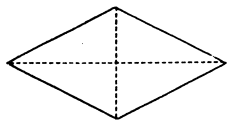


**Hier wurde nochmals die ausführliche Methode des Listings gewählt. Das Programm bietet keinerlei Schwierigkeiten,**

womit sich eine Dokumentation erübrigt

Auch eine Raute ist sehr einfach zu konstruieren.

```
100 REM====RAUTE=======================
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#5,0
140 REM====ANFANGSPOSITION=============
150 PRINT#1,"M";100;0
160 PRINT#1,"I"
170 REM====DATEN LESEN/ZEICHNEN========
180 FORX=0TO7
190 READX$,Y,Z
200 IFX=4THENPRINT#5,3
210 PRINT#1,X$,Y,Z
220 NEXT
230 REM====ENDE========================
240 PRINT#5,0
250 CLOSE1
260 CLOSE5
270 REM====DATEN GESAMT================
280 DATAJ,100,-50,J,200,,J,100,50,J,,
290 DATAJ,200,,R,100,50,J,100,-50
300 DATAM,,-150

READY.
```



Es werden zusätzlich zu der Figur noch die beiden Dialonalen gestrichelt eingezeichnet. Dieses Listing bleibt im übrigen auch übersichtlich, wenn wir die Methode benutzen, alle DATA's in einem Block zusammenzufassen. Bitte beachten Sie beim abtippen, daß vorkommende Nullen herausgekkürzt wurden,

also mehrere Kommata aufeinander folgen können. Der Rest des Programms ist in dieser Form jetzt schon sehr häufig vorgekommen und dürfte für jedermann leicht nachvollziehbar sein.

### *8.3.4 Dreiecke 1*

Bevor es mit räumlichen Figuren weitergeht, erfolgt ein kurzer Rückblick auf die Dreieckskonstruktionen. Dabei sollen drei Formen betrachtet werden, die in der Praxis der Geometrie sehr oft vertreten sind.

a) Gleichseitige Dreiecke
b) Gleichschenklige Dreiecke
c) Rechtwinkelige Dreiecke

Beginnen wir mit den gleichseitigen Dreiecken. Diese zeichnen sich wie der Name auch schon sagt, dadurch aus, daß alle drei Seiten gleich lang sein müssen und jeder Winkel 60 Grad groß ist. Um dieses gleich lang sein müssen und jeder Winkel 60 Grad groß ist. Um dieses Wissen in eine Zeichnung umzusetzen benötigt man erst einige Überlegungen. Die Grundseite läßt sich auch mit dem Plotter ohne weiteres zeichnen, da die Lage wahlfrei ist. Das Problem besteht in der Tatsache, daß wir zwar die Längen aller Seiten kennen, aber der Bezugspunkt in positiver y-Richtung nicht vorhanden ist.

Mit einer kurzen Rechnung kann man die Schwierigkeiten umgehen:

Bekannt sind die Längen aller drei Seiten; wenn man eine Höhe als Mittelsenkrechte auf die Grundseite einzeichnet, so

wird das gedachte Dreieck in zwei gleiche, rechtwinklige Dreiecke unterteilt. Jetzt kann man die Höhe berechnen und besitzt mit dem Endpunkt der Höhenstrecke den Bezugspunkt für die beiden fehlenden Seiten des Dreiecks.

Es existieren zur Berechnung der Höhe zwei Möglichkeiten. Zum einen kann der Satz des Pythagoras verwendet werden, aber mit einem Rechner ist auch der Sinus ein Lösungsweg.

Jetzt muß diese Theorie in allgemein gültige Formeln gesetzt werden, um mit dem Einsetzen der jeweiligen, beliebigen Kantenlängen sofort die Höhe ermitteln zu können und die Zeichnung anfertigen zu lassen. Die nachstehende Entwicklung einer Formel wurde mit dem Satz des Pythagoras vorgenommen.

Seitenlänge=l
Höhe=h

Allgemein gilt:
$h^2+(l/2)^2=l^2$
$h^2=l^2-(l/2)^2$
$h^2=3/4*l^2$

$\underline{h=SQR(3/4*l^2)}$

Diese Formel ist die allgemeine Höhenformel für ein gleichseitiges Dreieck. Nehmen wir nun als konkretes Beispiel 4 cm Seitenlänge. Es ergibt sich dann für h = SQR (12) oder 3,464. Nehmen Sie zur Kontrolle ruhig ein Stück Papier, zeichnen Sie die Grundseite mit 4 cm und die Höhe als Mittelsenkrechte mit 3,464 cm. Die verbleibenden Strecken von den Endpunkten der Grundseite zum zweiten Endpunkt der Höhe ergeben genau 4 cm.

Nachdem diese Formel nachgewiesen wurde, kann man sich mit den geschaffenen Grundlagen daran begeben, ein sinnvolles und funktionierendes Programm zu schreiben. Sie sehen an diesem noch recht einfachen Beispiel, wie wichtig die Vorarbeit schon bei kleinen Routinen ist. Nun das Listing:

```
100 REM====DREIECK====================
110 OPEN5,6,5
120 OPEN1,6,1
130 PRINT#1,"M";200;-50
140 PRINT#1,"I"
150 REM====EINGABE====================
160 PRINT"SL = 1 - 8 CM"
170 INPUT"QWELCHE SEITENLAENGE";A$
180 L=VAL(A$)
190 IFL<1ORL>8THEN160
200 L=L*50
210 REM====BERECHNUNGEN/ZEICHNUNG======
220 H=SQR(3/4*L↑2)
230 PRINT#1,"J";-L/2;0
240 PRINT#1,"J";0;H
250 PRINT#1,"J";L/2;0
260 PRINT#1,"J";0;0
270 REM====HOEHE ZEICHNEN==============
280 PRINT#5,5
290 PRINT#1,"J";0;H
300 PRINT#5,0
310 REM====ENDE=======================
320 PRINT#1,"M";0;-100
330 CLOSE1
340 CLOSE5
```

In Zeile 220 kommt genau die Formel vor, die allgemein entwickelt worden war. Geben Sie ein paar Werte ein und messen die Seiten in die Höhe nach. Die erhaltenen Angaben stimmen mit den Berechnungen überein und werden in Zentimetern gezeichnet, daher auch die Zeile 200 mit L = L * 50 (1 cm = 50 Programmschritte der Trommel). Die Beschreibung, wie man auf ein solches Programm wie das der Rechtwinkligen kommen kann, wurde nur deshalb so ausführlich dargelegt, damit man einsehen kann, daß ein Programm nur so gut werden kann, wie die Vorbereitungen dazu waren.

Da anhand einer Zeichnung zuerst die allgemeine Gesetzmäßigkeit für die Höhenformel hergeleitet wurde, kann man daraus ein universell einsetzbares Programm anfertigen und die Seitenlängen freier anwählen. Die Höhe läßt sich auch durch Probieren finden, und, wenn die Zeichnung genau genug ist, auch durch eine Konstruktion mit dem Zirkel.
Diese Methoden haben aber dann schon nicht mehr den gewünschten Effekt, wenn die Seitenwahl dem Benutzer offengehalten werden soll. Es geht, besonders bei der Erstellung von Plotterroutinen, nichts, ohne die nötigen Vorbereitungen getroffen zu haben. Wenden Sie deshalb besser etwas mehr Zeit zum Erstellen eines Konzeptes auf. Das Programm wird dadurch besser!

Jetzt dürfte die Konzipierung eines gleichschenkligen Dreiecks wenig Probleme bereiten. Denn die Höhenformel kann ohne weiteres beibehalten werden; die Berechnungen müssen zusammen mit der Eingaberoutine nur etwas erweitert werden, da nun zwei Längenangaben abgefragt werden müssen:

1) Länge der Grundseite
2) Länge der Seiten (gleichschenklig)

Sind diese Eingaben gemacht worden, so kann der Mittelpunkt der Grundseite berechnet werden. Dann läßt sich die Höhe als Mittelsenkrechte berechnen und die Seiten können eingezeichnet werden. Die Programmstruktur zeigt die folgende Liste:

Algorithmus

1) Plotter setzen
2) Grundseite/Schenkel abfragen
3) Höhe berechnen

4) Dreieck zeichnen

5) Höhe zeichnen (stricheln)

6) Ende

Setzen wir diesen Ablaufplan in ein Programm um, so erhalten wir

```
100 REM====1520 SETZEN=================
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#1,"M";200;-50
140 PRINT#1,"I"
150 REM====GRUNDSEITE/SCHENKEL=========
160 PRINT"SMAX. 8 CM FUER GRUNDSEITE"
170 PRINT"QMAX.15 CM FUER SCHENKEL"
180 INPUT"QQGRUNDSEITE";A$
190 INPUT"QQSCHENKEL  ";B$
200 A=VAL(A$):A=ABS(A)
210 B=VAL(B$):B=ABS(B)
220 IFA>8ORB>15ORA<1THEN160
230 REM====HOEHE BERECHNEN=============
240 A=A*50
250 B=B*50
260 H=SQR(B↑2-(A/2)↑2)
270 REM====DREIECK ZEICHNEN============
280 PRINT#1,"J";-A/2;0
290 PRINT#1,"J";0;H
300 PRINT#1,"J";A/2;0
310 PRINT#1,"J";0;0
320 REM====HOEHE ZEICHNEN==============
330 PRINT#5,5
340 PRINT#1,"J";0;H
350 REM====ENDE=======================
360 PRINT#5,0
370 PRINT#1,"M";0;-100
380 CLOSE1
390 CLOSE5
```

Zur Analyse des Programms lassen sich folgende Punkte bemerken:

Es wird Ihnen sicherlich schon aufgefallen sein, daß bei der INPUT-Abfrage stets mit Stringvariablen gearbeitet wird, wo doch nur Längenangaben und damit Zahlen eingelesen werden

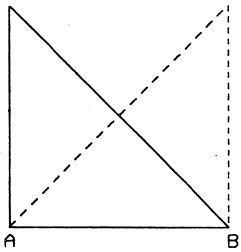
Dies hat nur den einen Grund, daß das Programm bei einer versehentlichen Eingabe eines Buchstaben anstatt einer Zahl nicht einen REDO FROM START Error präsentiert. Es werden zunächst alle Eingaben akzeptiert und in Zeile 220 wird dann aussortiert. Während bei einer 0 für die Grundseite das Programm an den Anfang der Leseroutine (Zeile 160) zurückspringt, werden Schenkel von 0 cm zugelassen. Sie sehen auf dem Papier dann nur die Grundseite.
Dem aufmerksamen Leser wird auch sicherlich nicht entgangen sein, daß etwas weiter oben gesagt wurde, die Höhenformel könne beibehalten werden, doch man sieht gegenüber $H=SQR(3/4*L^2)$ nun plötzlich $H=SQR(B^2-(A/2)^2)$

Das ist kein Widerspruch. Die erste Berechnung von H wurde, wie die zweite auch, mit dem Satz des Pythagoras erledigt, nur ist der erste Fall ein Spezialfall, da alle drei Seiten des Dreiecks genau gleich lang waren. Sie erinnern sich, daß in der Herleitung der Formel auch zuerst stand $H=SQR(L^2-(L/2)^2)$, bevor dann zu $H=SQR(3/4*L^2)$ umgeformt werden konnte?

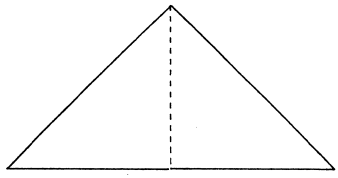
Der Rest der Routine besthet aus den hinreichend bekannten Befehlen für den 1520.

Es verbleibt noch das rechtwinklige Dreieck, doch hierbei werden wir uns zunächst auf die rechten Winkel auf der Grundseite (Punkte A und B eines Dreiecks) beschränken.
Diese Dreiecke sind zwar sehr einfach zu konstruieren, aber um an dieser Stelle schon ein Programm für beliebige Dreiecke mit rechtem Winkel zu besprechen, fehlen noch gewisse Grundlagen, um das Programm zu verstehen. Nachdem der Kreis besprochen worden ist, können wir ein Näherungsverfahren entwickeln, um rechtwinklige Dreiecke

beliebiger Art zu entwerfen (Thaleskreis).
Nun aber zu den drei Möglichkeiten, ein rechtwinkliges Dreieck zu zeichnen:
1) Man kann am linken Ende der Strecke der Grundseite (Punkt A) eines Dreiecks eine Senkrechte zeichnen, und das andere Ende der Strecke (Punkt C) mit dem rechten Ende der Grundseite (Punkt B) verbinden. Gezeichnetes Dreieck.
2) Dieser Vorgang kann auch auf Punkt B bezogen werden, die Senkrechte endet dann wieder in C, und A kann mit C verbunden werden.
Diese beide Methoden besitzen jedoch den Nachteil, daß die Grundseite nicht in x-Richtung liegt, sondern die Strecke BC bzw. AC ist. (Grundseite liegt dem rechten Winkel immer gegenüber).




3) Um die Grundseite nun an ihrem Platz zu lassen, können wir noch zu einer anderen Methode greifen.

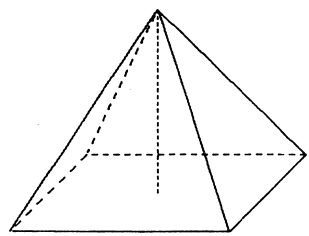


Die Grundseite bildet hier ordnungsgemäß die Strecke AB. Am Punkte C beträgt der Winkel genau 90 Grad. Dieses erreicht man dadurch, daß man die Winkel an den Punkten A und B mit jeweils 45 Grad anwählt, da die Winkelsumme im Dreieck bekanntlich 180 Grad beträgt und somit für Punkt C 90 Grad verbleiben. Diese Zeichnung läßt sich immer dann so anfertigen, wenn das Dreieck gleichschenklig ist und die Höhe gleich der halben Grundseite ist. Mit dem Thaleskreis hingegen wird die Konstruktion variabler, da dann auch Winkel angegeben werden können, aber dazu gleich.
Für die ersten beiden Dreiecksarten werden keine Programme angegeben, da sie keine Formeln beinhalten und sehr leicht zu schreiben sind. Für Beispiel 3 dient das folgende Programm. Mit ihm wird das Dreieck mit rechten Winkeln in C gezeichnet, Sie brauchen nur die Länge der Grundseite einzugeben.

```
100 REM====1520 SETZEN=================
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#5,0
140 PRINT#1,"M";240;-50
150 PRINT#1,"I"
160 REM====EINGABE=====================
170 PRINT"sGRUNDSEITE MAX. 8 CM"
180 INPUT"QQGRUNDSEITE";A$
190 A=VAL(A$):A=ABS(A)
200 IFA>8THEN170
210 B=A*25
220 REM====DREIECK ZEICHNEN============
230 PRINT#1,"J";-B;0
240 PRINT#1,"J";0;B
250 PRINT#1,"J";B;0
260 PRINT#1,"J";0;0
270 REM====HOEHE ZEICHNEN==============
280 PRINT#5,5
290 PRINT#1,"J";0;B
300 REM====ENDE========================
310 PRINT#5,0
320 PRINT#1,"M";0;-100
330 CLOSE1
340 CLOSE5
```

In den Zeilen 170 bis 210 wird die Länge der Grundseite in Zentimetern abgefragt. Es folgt eine Prüfung auf zu große oder zu kleine Werte und Zeile 210 setzt die Einheit auf Plotterformat um. Dann wird die übliche Zeichenmethode angewandt. Das Dreieck wird gezeichnet und die Höhe gestrichelt. Der obere Winkel beträgt nun genau 90 Grad.
Bevor wir uns an Kreise und Ellipsen wagen, noch ein letzter räumlicher Körper, die Pyramide. Sie besteht aus einer quadratischen Grundfläche. An den Kanten der Fläche liegt jeweils ein Dreieck in der Art, daß die Spitzen der vier Dreiecke auch die Spitze der Pyramide bilden. Das nachstehende Programm stellt die Pyramide perspektivisch dar:

```
100 REM====1520 SETZEN=================
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#1,"M";100;0
140 PRINT#1,"I"
150 REM====DATEN LESEN=================
160 FORX=0TO11
170 READX$,Y,Z
180 IFX=6THENPRINT#5,5
190 IFX=10THENPRINT#5,3
200 PRINT#1,X$,Y,Z
210 NEXT
220 REM====ENDE========================
230 PRINT#5,0
240 CLOSE1
250 CLOSE5
260 END
270 REM====DATEN GESAMT================
280 DATAJ,200,0,J,270,70,J,135,200
290 DATAJ,200,0,R,135,200,J,0,0
300 DATAJ,70,70,J,270,70,R,70,70
310 DATAJ,135,200,J,135,35,M,0,-100

READY.
```

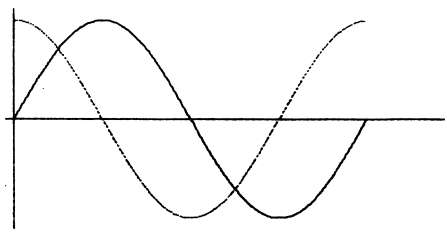


**Da bei einer räumlichen Zeichnung erfahrungsgemäß viele verschiedene Zeichen- und Bewegungsvorgänge vorhanden sind (siehe auch Würfel), ist es hier sinnvoll, die Programmierungshinweise mit DATA - Zeilen zu benutzen. Die Zeilen 280/290 sind für die Vordergrundzeichnung (durchgezogene Linien) verantwortlich , 300/310 stricheln die Höhe und den Hintergrund.**

Auch bei den Lesezeilen von 160-210 wurde die Tipparbeit auf ein Minimum reduziert, aber der Programmablauf trotzdem übersichtlich gestaltet. Die Zeilen 180/190 schalten den Plotter auf die jeweiligen Linienarten um.

### *8.3.5 Kreisbögen*

Nachdem bisher genügend gerade Linien gezogen wurden, können wir uns langsam daran begeben, beide Zeichenrichtungen des Plotters so zu kombinieren, daß ein Bogen entsteht.
Will man das jedoch erreichen, so kommt man mit den bisher kennengelernten Funktionen oder Formeln nicht mehr aus; Sinus und Cosinus werden unumgänglich.
Zur Erinnerung zunächst eine Zeichnung, in der beide Funktionen eingezeichnet sind. Die Sinuskurve ist durchgezeichnet, der Cosinus ist gestrichelt. Zur Erstellung solcher Funktionsgraphen werden wir im übrigen später noch kommen.



Gelingt es nun, diese beiden Zeichnungsweisen Sinus und Cosinus sinnvoll miteinander zu verbinden, so eröffnet diese Kombination zusammen mit dem 1520 ein neues Anwendungsgebiet, welches Kreise und Ellipsen etc. umfaßt.
Wir werden mit einem Kreis beginnen und als erstes ein

Listing präsentieren, welches die Grundlagen zur Erstellung eines Kreises beinhaltet. Dieses wird dann gleich ausführlich analysiert.

```
100 REM====1520 SETZEN=================
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#1,"M";240;0
140 PRINT#1,"I"
150 REM===KREIS ZEICHNEN==============
160 FORX=-1.6TO4.8STEP.04
170 Y=150*COS(X)
180 Z=150*SIN(X)
190 IFX=-1.6THENPRINT#1,"R";Y;Z
200 PRINT#1,"J";Y;Z
210 NEXT
220 REM====RADIUS======================
230 PRINT#5,5
240 PRINT#1,"J";0;0
250 REM====ENDE========================
260 PRINT#5,0
270 PRINT#1,"M";0;-200
280 CLOSE1
290 CLOSE5
```

Diese Erklärungen werden mit den Zeilen 160 bis 210 begonnen, da das Öffnen des 1520 kein Problem mehr sein dürfte.

160 Der Bereich der Zahlen, die in die Sinus und Cosinus Funktion eingesetzt werden sollen, wird zusammen mit der Schrittweite festgelegt. In unserem Beispiel finden Sie ... STEP .04, also 160 Stück. Mit der Schrittweite legt man die Zeichenauflösung fest. Zu große Werte sind nicht zu empfehlen, da das Gesamtbild zu sehr darunter leidet - bei zu kleinen Zahlen ist der Zeitaufwand sehr groß. Dadurch, daß bei -1.6 begonnen wurde, ist der Anfangspunkt der Zeichnung um ein Viertel des Kreises verlegt worden. Geben Sie FOR X=0 TO 6.4 ein, so erhalten Sie natürlich dasselbe Ergebnis, bloß der

Zeichenbeginn liegt anders.

170 Hier wird in der Variablen Y der COS des jeweiligen x-Wertes abgelegt. Der errechnete Cosinus wird noch mit einer Konstanten (im Beispiel 150) multipliziert. Diese Konstante fixiert den Radius für die x-Richtung. Ein Teil des Kreises wird mit dieser Funktion erstellt.

180 Die Variable Z bekommt hier den Sinus-Anteil des Kreises zugeordnet. Der errechnete Sinus-Wert wird mit einem konstanten Faktor multipliziert, der den Radius für die y-Richtung bestimmt. Da wir einen Kreis erhalten wollen, muß der Radius für beide Ausdehnungsrichtungen gleich sein.

190 In Zeile 130 war der Kreismittelpunkt auf 240/0 gelegt worden. Somit muß der Plotter, bevor er zu zeichnen beginnt, erst um die Strecke des Kreisradius nach außen fahren. Dies geschieht sofort, nachdem der erste Wert (-1.6) in die Funktionen eingesetzt wurde. Beginnen Sie in Zeile 160 bei 0, so steht entsprechend in Zeile 190 auch eine 0!

200 Diese Zeile zeichnet die Teilkurve bis zum errechneten Punkt und Zeile 210 beginnt den gesamten Vorgang erneut.

Diese Analyse ist besonders für einen Neuling auf dem Gebiet des 1520 interessant, da im Handbuch eine ähnliche Routine undokumentiert erscheint. So muß man erst lange ausprobieren, was passiert, wenn man diese oder jene Zahl ändert - aber wenn die Analyse verstanden worden ist, so erscheint das Programm des Kreises gleich in einem anderem Licht.

Mit dem nächsten Programm soll für den Anwender die Kreisfunktion etwas komfortabler aufbereitet werden:

```
100 REM====1520 SETZEN==========
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#1,"M";240;0
140 PRINT#1,"I"
150 REM====EINGABE==========
160 PRINT"SRADIUS MAX 4 CM"
170 PRINT"QAUFLOESUNG .01 - .1"
180 INPUT"QRADIUS";A$
190 INPUT"QAUFLOESUNG";B$
200 A=VAL(A$):A=ABS(A)
210 B=VAL(B$):B=ABS(B)
220 IFA>4THEN160
230 IFB>.1ORB<.01THEN160
240 A=A*50
250 REM====KREIS ZEICHNEN==========
260 FORX=-1.6TO4.8STEPB
270 Y=A*COS(X)
280 Z=A*SIN(X)
290 IFX=-1.6THENPRINT#1,"R";Y;Z
300 PRINT#1,"J";Y;Z
310 NEXT
320 REM====RADIUS==========
330 PRINT#5,5
340 PRINT#1,"J";0;0
350 REM====ENDE==========
360 PRINT#5,0
370 PRINT#1,"M";0;-200
380 CLOSE1
390 CLOSE5
```

**Sie können, wenn Sie das Programm gestartet haben, den Radius und die Auflösung der Zeichnung und damit auch die Zeichengeschwindigkeit festlegen. Die Zeichnung erfolgt mit gestrichelter Einzeichnung des Radius.**
**Die Analyse des Kreisproblems zeigte, daß für x und y-Richtung zwei verschiedene Radien zur Verfügung stehen, die man auch unterschiedlich anwählen kann. Mit diesem Wissen läßt sich damit auch eine Ellipse konstruieren. Diese Figur ist ein in die Länge gezogener Kreis, der durch die Ausdehnung zwei Radien besitzt.**

**Das letzte Listing kann durch ein paar zusätzlich eingefügte Zeilen schnell auf das Zeichnen von Ellipsen umgeschrieben**

werden; wünschen Sie einen Kreis, so ist dieses weiterhin möglich, wenn die Radien gleich angegeben werden.

```
100 REM====1520 SETZEN==================
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#1,"M";240;0
140 PRINT#1,"I"
150 REM====EINGABE=====================
160 PRINT"sRADIUS MAX 4 CM ←"
170 PRINT"QRADIUS MAX 8 CM ↑"
180 PRINT"QAUFLOESUNG .01 - .1"
190 INPUT"QRADIUS ←";A$
200 INPUT"QRADIUS ↑";A1$
210 INPUT"QAUFLOESUNG";B$
220 A=VAL(A$):A=ABS(A)
230 REM====TEST AUF EINGABEFEHLER======
240 A1=VAL(A1$):A1=ABS(A1)
250 B=VAL(B$):B=ABS(B)
260 IFA>4THEN160
270 IFA1>8THEN160
280 IFB>.1ORB<.01THEN160
290 A=A*50
300 A1=A1*50
310 REM====ELLIPSE ZEICHNEN============
320 FORX=-1.6TO4.8STEPB
330 Y=A*COS(X)
340 Z=A1*SIN(X)
350 IFX=-1.6THENPRINT#1,"R";Y;Z
360 PRINT#1,"J";Y;Z
370 NEXT
380 REM====RADIEN======================
390 PRINT#5,5
400 PRINT#1,"R";0;0
410 PRINT#1,"J";A;0
420 PRINT#1,"R";0;0
430 PRINT#1,"J";0;A1
440 REM====ENDE========================
450 PRINT#5,0
460 PRINT#1,"M";0;-200
470 CLOSE1
480 CLOSE5
```

Wenn mit diesem Programm Probleme auftauchen, so ist es empfehlenswert, die Theorie über die Konstruktion eines Kreises nochmals zu wiederholen.
Wir wollen nun die gewonnenen Erkenntnisse in die Praxis

umsetzen und zwei räumliche Körper konstruieren, die Kurven und Strecken kombinieren, den Zylinder und den Kegel. Dabei wird wiederum auf die perspektivische Ansicht Wert gelegt. Bei einem Zylinder kommen die Ellipsen zur Anwendung - es werden zwei Stück benötigt. Diese verbindet man dann mit zwei Strecken zur Gesamtkonstruktion. Der Startpunkt zum Zeichnen der unteren Ellipse muß jetzt aber verlegt werden, da nur eine Hälfte durchgezogen, die andere aber gestrichelt erscheinen soll, um Vordergrund und Hintergrund zu trennen. Um dieses zu erreichen, liegt die FOR - NEXT Schleife mit anderen Werten versehen im Programm. Jetzt wird der Bereich von - 3.2 bis 0 eingesetzt und so erscheint genau die untere Hälfte der Ellipse auf dem Papier. Den positiven Bereich von 0 bis 3.2 liest man ein, nachdem mit PRINT# 5,X die Linienart verändert worden ist.

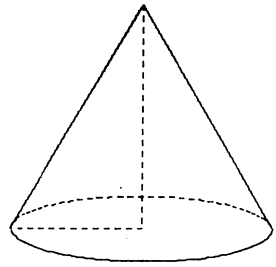
```
100 REM====1520 SETZEN=================
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#1,"M";240;0
140 PRINT#1,"I"
150 REM====ELLIPSEN ZEICHNEN===========
160 FORX=-3.2TO3.2STEP.04
170 Y=120*COS(X)
180 Z=30*SIN(X)
190 IFX=-3.2THENPRINT#1,"R",Y,Z
200 IFBN=1THENU=U+1
210 IFU=80THENPRINT#5,5
220 PRINT#1,"J";Y;Z
230 NEXT
240 IFBN=1THEN320
250 REM====ELLIPSE 2 VORBEREITEN=======
260 PRINT#1,"J";-120;-300
270 PRINT#1,"R";0;-300
280 PRINT#1,"I"
290 BN=1
300 GOTO160
310 REM====SEITE/HOEHE=================
320 PRINT#1,"R";120;0
330 PRINT#5,0
340 PRINT#1,"J";120;300
350 PRINT#1,"R";0;300
360 PRINT#5,5
370 PRINT#1,"J";0;0
380 REM====ENDE=======================
390 PRINT#5,0
400 PRINT#1,"R";-240;-100
410 CLOSE1
420 CLOSE5
```

Das Programm beginnt, den Zylinder mit der oberen Ellipse zu zeichnen. (ab Zeile 160). Der Faktor für den Radius wurde dabei für x-Richtung mit 120, für die y-Richtung hingegen mit 30 festgelegt. Bei Änderungen im Programm sollten Sie dies beachten, da die Zeichnung der Seitenstrecken sich auf diese beiden Werte bezieht und aus diesem Grunde dann auch geändert werden müßten.

Von Zeile 260 bis 300 wird die Ellipse 2 vorbereitet; die erste Seitengerade wird zu Papier gebracht und ein neuer relativer Nullpunkt definiert.Dann erfolgt ein Rücksprung in die Zeichenroutine der Ellipsen ab Zeile 150, wobei in Zeile

210 diesesmal auf stricheln umgeschaltet wird, wenn die untere Hälfte passiert wurde. Es fehlen noch die zweite Seite und die gestrichelten Höhen. Das erledigen die Zeilen 320 bis 370.
Bei einem Kegel wird die Angelegenheit wieder etwas einfacher, da nur eine Ellipse nötig ist, um eine räumliche Darstellung zu erzielen.

```
100 REM====1520 SETZEN=================
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#1,"M";240;0
140 PRINT#1,"I"
150 REM====ELLIPSEN ZEICHNEN===========
160 FORX=-3.2TO3.2STEP.04
170 Y=120*COS(X)
180 Z=30*SIN(X)
190 IFX=-3.2THENPRINT#1,"R",Y,Z
200 U=U+1
210 IFU=80THENPRINT#5,5
220 PRINT#1,"J";Y;Z
230 NEXT
240 REM====KEGEL ZEICHNEN==============
250 PRINT#1,"J";0;0
260 PRINT#1,"J";0;200
270 PRINT#5,0
280 PRINT#1,"J";-120;0
290 PRINT#1,"R";120;0
300 PRINT#1,"J";0;200
310 REM====ENDE========================
320 PRINT#1,"M";0;-100
330 CLOSE1
340 CLOSE5

READY.
```



**Das Teilprogramm für die Ellipse von Zeile 160 bis 230 braucht hier nicht mehr erklärt zu werden, da es fast genau aus der Routine für den Zylinder entnommen wurde. Auch die Vollendung des Kegels ab Zeile 250 ist einsichtig, da nur die Grundbewegungsarten vorliegen.**

### *8.3.6 Dreiecke 2*

Zum Abschluß der Darlegungen bezüglich geometrischer Figuren nun das rechtwinklige Dreieck (rechter Winkel gegenüber der Grundseite) mit verschiedener Schenkellänge. Dieses Problem wurde aus dem Grunde an das Ende gestellt, weil die Theorie der Konstruktion eines Kreises nötig war, und diese bei der Abhandlung der Dreiecke noch nicht bekannt war. Das Problem sollte mit dem Thaleskreis gelöst werden; zur Wiederholung des Thaleskreises kurz die wichtigsten Merkmale:
Schlägt man um den Mittelpunkt einer Strecke mit der Länge 1 einen Halbkreis mit dem Radius 1/2, so entsteht, wenn man das linke Ende der Strecke 1 (Punkt A) mit einem beliebigen Punkt auf dem Halbkreis verbindet und von diesem Punkt, eine Strecke zum rechten Ende der Seite 1 zieht, im Punkte C immer ein rechtwinkliges Dreieck.
Da diese Theorie etwas komplizierter klingt, wollen wir ein Programm konzipieren, mit dem man beliebige, rechtwinklige Dreiecke zeichnen kann und der Winkel gegenüber der Grundseite AB liegen soll. Um ein Programm zu erstellen, das präzise arbeiten kann, muß zunächst die Vorgehensweise geklärt werden.
Das Programm soll zusätzlich als Demonstrationsroutine dienen. Darum ist es nützlich, den Halbkreis mit einzuzeichnen, um die Richtigkeit des Satzes zu belegen.
Algorithmus

1) Absprungpunkt definieren
2) Halbkreis zeichnen und am Absprungpunkt Dreieck zeichnen
3) Halbkreis beenden
4) Grundseite einzeichnen

Dieses wurde im folgenden Programm umgesetzt

```
100 REM====1520 SETZEN==================
110 OPEN1,6,1
120 OPEN5,6,5
130 PRINT#1,"M";240;0
140 PRINT#1,"I":E=50
150 REM====THALESKREIS=================
160 FORX=0TO3.2STEP.04
170 Y=120*COS(X)
180 Z=120*SIN(X)
190 IFX=0THENPRINT#1,"R",Y,Z
200 V=V+1
210 IFV=ETHENPRINT#1,"J";120;0
220 IFV=ETHENPRINT#1,"R";Y-.04;Z-.04
230 IFV=ETHENPRINT#1,"J";-120;0
240 PRINT#1,"J";Y;Z
250 NEXT
260 REM====GRUNDSEITE==================
270 PRINT#1,"R";120;0
280 PRINT#1,"J";-120;0
290 REM====ENDE=======================
300 PRINT#1,"M";0;-100
310 PRINT#5,0
320 CLOSE5
330 CLOSE1
```

In Zeile 140 wird mit E=50 ein Punkt definiert, an dem die Zeichnung des Halbkreises unterbrochen und das Dreieck eingezeichnet werden soll. Diese Zahl scheint zunächst nichtssagend zu sein, wird aber verständlich, wenn man sich die FOR-NEXT-Schleife für den Halbkreis ansieht. Dort wird der Kreis in 80 einzelne Zeichenabschnitte (3,2/0.04=80) unterteilt. Zeile 200 erhöht die Variable V bei jedem Zeichenschritt um eins, sodaß bei V=E=50 das Dreieck gezeichnet werden kann. Ist die Absprungstelle erreicht (Zeilen 210-230), so werden die Eckpunkte der Grundseite AB angesteuert und in C, dem Absprungpunkt, entsteht zwangsläufig ein rechter Winkel.
Das Einzeichnen der Grundseite ist schließlich kein Problem mehr, und die Zeichnung ist vollendet. Versuchen Sie ruhig einmal, den Wert für E zwischen 1 und 89 zu variieren, es wird immer ein rechtwinkliges Dreieck herauskommen!

Die bisher behandelten geometrischen Figuren stehen nur als Beispiele da, um den Programmaufbau und die Problemlösung zu dokumentieren; eine Vollständigkeit ist unmöglich zu realisieren - es konnten nur die bekanntesten zur Srache kommen. Wenn Sie eine Figur vermissen, die Sie sehr interessiert hätte, so versuchen Sie die dargestellte Theorie in die Praxis umzusetzen und konstruieren Sie die Figur mit einem eigenen Programm. Zur Anregung noch ein paar Figuren:

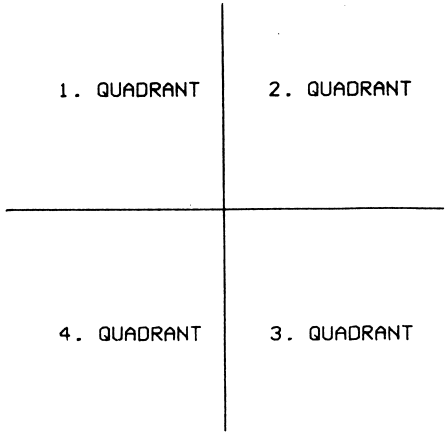
a) Prisma
b) Pyramide-/Kegelstumpf
c) Kugel (räumlich)
d) Ellipsoid

Die genannten Figuren kann man mit dem Wissen aus den bisherigen Erläuterungen ohne weiteres konstruieren; es gehört etwas Überlegung dazu, aber es ist plottertechnisch möglich.
Mit dem Wunsch, daß alle Konstruktionsversuche der obigen Vorschläge gelingen, gehen wir zu einem ebenfalls sehr nützlichen und interessanten Thema über.

## *8.4 Darstellung von Funktionen*

Mit dem Wort Funktionsgraph verbindet man größtenteils den Sinus, den Cosinus und die x-Funktionen. Da besonders letztere sehr verbreitet sind (Kurvendiskussion), sollen die x-Graphen und ihre Darstellung im Mittelpunkt dieses Teilbereiches stehen. Interessant ist hier der Bereich von x bis x, da sich (schul-)mathematische Problemstellungen oft in diesem Bereich ansiedeln, und auch die allseits

gebräuchliche Parabel x in dieses Gebiet fällt.
Damit es bei Beschreibungen keine Verständigungsprobleme gibt, kurz etwas zur Zeichenweise der Funktionen. Die Graphen werden, wie es bei Funktionen üblich ist, in einem Koordinatensystem eingezeichnet. Dieses besitzt eine negative und eine positive x- bzw. y-Achse. Man unterteilt das Koordinatenkreuz auch häufig in Quadranten (Teilbereiche). Von diesen Quadranten existieren vier Stück. Sie werden im Uhrzeigersinn nummeriert, beginnend mit dem Bereich über der negativen x-Achse.




Alle Beschreibungen nehmen Bezug auf diese Vereinbarung. Jetzt werden einige Arten von Koordinatensystemen vorgestellt, damit man für jede Zeichnung die passende Version auswählen kann. Denn das Koordinatenkreuz ist nicht immer nötig; will man z.B. eine Parabel zeichnen, so wird der Bereich unter der x-Achse solange nicht benötigt, wie es sich um eine Normalparabel oder auf der positiven y-Achse

verschobenen Parabel handelt.
Eine Sinus oder Cosinuskurve bewegt sich fast immer in Richtung der positiven x-Achse und zu beiden Seiten der y-Achse. Der negative Anteil der x-Achse kann somit gespart werden.

X = X

Diese Funktion ergibt in dieser Form immer eine Ursprungsgerade, die den Nullpunkt unter einem Winkel von 45 Grad schneidet. Bei Y = X liegt die Gerade im 2. Quadranten und geht nach dem Nulldurchgang in den 4. Quadranten über. Bei Y = X gilt der Geradenverlauf vom Ersten in den Dritten. Diese Tatsache muß von einem Programm ebenso verarbeitet werden können, wie die, daß man die Gerade auf der y-Achse verschieben und um den Ursprung drehen kann. Vereinigt man die genannten Forderungen so erhält man folgendes Programm:

```
100 REM====1520 SETZEN==================
110 OPEN4,6
120 OPEN1,6,1
130 OPEN2,6,2
140 REM====KOORDINATENSYSTEM============
150 FORA=1TO6
160 READX$,Y,Z
170 PRINT#1,X$;Y;Z
180 NEXT
190 DATAM,0,0,D,480,0,M,240,240
200 DATAD,240,-240,M,240,0,I,0,0
210 REM====DEFINITION DER FUNKTION=====
220 DEFFNA(X)=X
230 REM====ZEICHNEN====================
240 FORA=200TO-200STEP-1
250 IFA=200THENPRINT#1,"R";A;FNA(A)
260 PRINT#1,"J";A;FNA(A)
270 NEXT
280 PRINT#1,"M";0;0
290 REM====ENDE========================
300 CLOSE1
310 CLOSE2
320 CLOSE4
```

Eine Analyse der kleinen Routine erscheint äußerst sinnvoll, da auch $x^n$-Funktionen darauf aufgebaut werden.

150

-180 Die Daten für das Koordinatenkreuz werden eingelesen und gezeichnet.

190

-200 Daten für das Koordinatensystem

220 Der C64 besitzt die Möglichkeit, Funktionen zu definieren (DEF), um dem Programmierenden Arbeit zu ersparen. Man kann im Verlauf des weiteren Programms mit PRINT FNA (ZAHL) etc. Bezug auf die zuvor definierte Funktion nehmen. Die betreffende Zahl wird eingesetzt und man hat sofort das benötigte Ergebnis, ohne jedesmal die Hauptfunktion eingeben zu müssen.
Von dieser Technik macht auch diese Routine Gebrauch, da so auch die Übersicht erhalten bleibt. Mit dem Ausdruck DEFFNA (X) = X wird also dem Computer nur mitgeteilt, daß die zu bearbeitende Funktion das Aussehen Y = X besitzt. Die Ursprungsgerade ist so definiert.
Wollen Sie die später entstehende Gerade auf der Y-Achse verschieben, so addieren Sie einen bestimmten Wert für die Verschiebung nach oben und subtrahieren einen bestimmten Wert für die Verschiebung nach unten. (50 = 1 cm). Auf diese Art können beispielsweise parallele Geraden einfach gezeichnet werden.

240 Nun gilt es, für die jeweilige Funktion einen individuellen Bereich zu finden, in dessen Rahmen Zahlen in die Funktion eingesetzt werden können.
Beim 1520 ist das Maß aller Dinge die Länge der X-Achse, nämlich 480 Zeichenschritte. Für eine Ursprungsgerade bietet sich der Bereich von 200 bis 200 an, da der X-Wert dem Y-Wert entspricht, und deshalb

auch die X Werte zwischen 200 und -200 schwanken. Dieses liegt im erlaubten Bereich. Mit der Schrittweite von eins wird eine gute Zeichnung erzielt werden.

260 Die Werte werden hier gezeichnet, der Y-Wert wird berechnet, indem der X - Wert in die definierte Funktion eingesetzt wird, (Y = X).

Auch die letzte Forderung, eine variable Steigung zu erreichen, läßt sich mit einigen kleinen Änderungen erfüllen.
Versuchen Sie folgendes:

In Zeile 240 werden die 200 und -200 durch 100 und -100 ersetzt. Für Zeile 250 gilt dann dementsprechend FORA=100.... Das A in den Zeilen 250/260 wird nun mit dem Faktor 2 multipliziert. Sie erhalten also für 260:

```
260 PRINT#1, "J"; 2-A; FNA (A)
```

Für 250 gilt Entsprechendes. Starten Sie dieses Programm, so ist die Steigung der Geraden sichtbar geringer geworden. Mit dem gezeigten Programm lassen sich also schon alle Anforderungen der Geradengleichung Y=mx+b erfüllen. Das m ist die Steigung, die sich beliebig variieren läßt, b ist die Strecke vom Nullpunkt (0/0) bis zum Schnittpunkt der Geraden mit der Y-Achse. Auch diese Forderung der Verschiebung der Geraden an der Y-Achse wurde oben bearbeitet. Das Programm zur Erstellung einer Geraden dient in seiner Grundform als Ausgangspunkt für das Plotten der Funktionen von $x^n$ für n>1.
Sollten Sie das Geradenprogramm eingetippt haben, so kann es im Arbeitsspeicher stehengalassen werden, denn die Programmteile wie das Koordinatensystem sind fast identisch. Bevor das Programm nun auf die Zeichnung vom Graphen im

Bereich von $x^2$-$x^5$ umgeschrieben werden kann, noch etwas zur Vorgehensweise bei der Umstellung.
Die Anforderungen an die Zeichnung sollen mit Sicherheit bestehen bleiben, d.h. die Auflösung muß im Prinzip gleich bleiben. Um diesem Ziel möglichst nahe zu kommen, muß die Veränderung an einer anderen Stelle im Programm vorgenommen werden. Die eleganteste Möglichkeit ist die, den mit FNA (A) berechneten Wert mit einer Konstanten zu multiplizieren, die den erhöhten Exponenten der Funktion in der Weise vermindert, daß die Öffnungen der Zeichnungsanfangspunkte der Graphen fast identisch sind.

Damit Sie nun nicht durch Ausprobieren den Spaß an der Angelegenheit verlieren müssen, kurz eine theoretische Betrachtung; dazu vergleichen wir die Funktionen von $x - x^4$ miteinander. Sie werden das Prinzip sofort erkennen können. Haben wir die Funktion Y = X, so setzen wir in Gedanken einmal die Zahl 100 für X ein, das Ergebnis bleibt 100. Diese Tatsache nehmen wir jetzt als Ausgangspunkt für die weiteren Überlegungen. Werden in die obige Funktion Werte eingesetzt und als Maximum eben 100, so zeichnet der Plotter nur bis 100, wenn ein Schritt gleich 1 entspricht. Die Zechnung liegt bei passender Wahl des Nullpunktes (z.B. in der Mitte des Zettels) im erlaubten Bereich in x-Richtung (0-480). Doch wird der Exponent um eins erhöht, so erhält man die Funktion $Y=X^2$. Werden wieder Zahlen bis 100 eingesetzt, so liegt das Ergebnis für Y schon bei $100^2$ = 10.000; das Ergebnis ist bei Vermehrung des Exponenten um eins also 100mal größer.

Dies bestätigen auch die Funktionen

$Y=X^3$ und $Y=X^4$.

Damit auch für nicht ganzzahlige Exponenten der Proportionalitätsfaktor leichter zu ermitteln ist, gleich eine allgemeine Formel. Doch nun erst weiter auf dem Weg zum konstanten Faktor: Da der Wertbereich, der in die Funktion eingesetzt wird, als größte Zahl die 100 behalten soll, die errechnete Zahl für Y aber mit jeder Erhöhung des Exponenten von X steigt, so muß die mt FNA (A) berechnete Zahl in der Weise mit einem Faktor multipliziert werden, daß die Lösung für Y immer identisch ist, egal welchen Exponenten X besitzt.
Also wird bei der Parabel mit der Funktionsvorschrift $X^2$ die Lösung für Y immer mit .01 multipliziert. Dann ist der größtmögliche Wert wieder 100, da 100*100*0.01 100 ergibt. Für die $X^3$ liegt der Proportionalitätsfaktor bei 0,0001.

Damit Ihnen bei den vielen Nullen keine Fehler unterlaufen hier eine Berechnungsformel für die Konstante C. Sie wählen bei $X^n=Y$ für n eine beliebige Zahl.

Dabei müssen also die natürlichen Zahlen nicht alleine zur Anwendung kommen, auch rationale Zahlen etc. sind möglich. Haben Sie sich für einen Exponenten entschieden, so gilt für die Konstante, mit der man FNA (A) multipliziert folgender Ausdruck:
$C=1/100^{(n-1)}$

Beispiel zur Anwendung der Formel:

Sie möchten die Funktion $X^6$ graphisch darstellen lassen: der Exponent, das n ist 6. Diese Zahl wird in die gezeigte Formel eingesetzt und Sie erhalten

$C=1/100^{(6-1)}$

und ausgerechnet ergibt sich $10^{-10}$.
Schon haben Sie dann den Wert parat, mit dem die Funktion $Y=X^6$ in Verbindung zu bringen ist, um im Bereich bis 100 für den Wert x eine Zeichnung zu erhalten.

Die nächsten drei Routinen wenden diese Theorie praktisch an, es handelt sich um die Darstellung der Funktionen $X^2$, $X^3$, $X^4$.

Sehen Sie sich die Listings kurz durch, um die Konstruktion zu verstehen. Eine Dokumentation schließt sich direkt an die Programme an.

```
100 REM====1520 SETZEN=============
110 OPEN1,6,1
120 REM====KOORDINATENSYSTEM===========
130 FORA=1TO6
140 READX$,Y,Z
150 PRINT#1,X$;Y;Z
160 NEXT
170 DATAM,0,0,D,480,0,M,240,240
180 DATAD,240,-20,M,240,0,I,0,0
190 REM====DEFINITION DER FUNKTION=====
200 DEFFNA(X)=X↑2
210 C=.02
220 REM====ZEICHNEN====================
230 FORA=100TO-100STEP-1
240 IFA=100THENPRINT#1,"R";A;C*FNA(A)
250 PRINT#1,"J";A;C*FNA(A)
260 NEXT
270 PRINT#1,"M";0;0
280 REM====ENDE========================
290 CLOSE1
```

```
100 REM====1520 SETZEN==================
110 OPEN1,6,1
120 REM====KOORDINATENSYSTEM============
130 FORA=1TO6
140 READX$,Y,Z
150 PRINT#1,X$;Y;Z
160 NEXT
170 DATAM,0,0,D,480,0,M,240,240
180 DATAD,240,-240,M,240,0,I,0,0
190 REM====DEFINITION DER FUNKTION=====
200 DEFFNA(X)=X↑3
210 C=.0002
220 REM====ZEICHNEN====================
230 FORA=100TO-100STEP-1
240 IFA=100THENPRINT#1,"R";A;C*FNA(A)
250 PRINT#1,"J";A;C*FNA(A)
260 NEXT
270 PRINT#1,"M";0;0
280 REM====ENDE========================
290 CLOSE1
```

```
100 REM====1520 SETZEN==================
110 OPEN1,6,1
120 REM====KOORDINATENSYSTEM============
130 FORA=1TO6
140 READX$,Y,Z
150 PRINT#1,X$;Y;Z
160 NEXT
170 DATAM,0,0,D,480,0,M,240,240
180 DATAD,240,-20,M,240,0,I,0,0
190 REM====DEFINITION DER FUNKTION=====
200 DEFFNA(X)=X↑4
210 C=.000002
220 REM====ZEICHNEN====================
230 FORA=100TO-100STEP-1
240 IFA=100THENPRINT#1,"R";A;C*FNA(A)
250 PRINT#1,"J";A;C*FNA(A)
260 NEXT
270 PRINT#1,"M";0;0
280 REM====ENDE========================
290 CLOSE1
```

**Wie Sie bei der Betrachtng der drei oberen Programme sicherlich bemerkt haben, sind die Listings fast identisch. Unterschiede lassen sich jeweils nur in drei Zeilen feststellen. Zeile 180: In dieser DATA-Zeile werden Teile der Zeichenbefehle für das Koordinatensystem abgelegt. Bei**

der $X^2$-Funktion benötigen wir nur den oberen Teil des Koordinatenkreuzes, da $X^2$ die Normalparabel und somit nicht auf der Y-Achse verschoben ist. Sie sehen
180 DATAD, 240,-20,.....
Bei der $X^3$-Darstellung benötigt man wegen des ungeraden Exponenten Quadranten oberhalb und unterhalb der x-Achse, während $X^4$ wieder in den Quadranten 1 und 2 zu finden ist. Im Listing der $X^3$ Funktion wird die Zeile 180 deshalb so begonnen:
180 DATAD, 240, -240,.....

Die negative Zahl gibt in den Data Zeilen die Länge der negativen y-Achse an.

Zeile 200: Diese Zeile definiert die jeweilige Funktion, die bearbeitet werden soll und Zeile 210 fixiert den dazugehörenden Proportionalitätsfaktor, der mit der oben gezeigten Formel berechnet wurde und dann mit zwei multipliziert worden ist, da der Wertebereich nur noch von 100 bis -100 geht.

| Funktion | Faktor |
|---|---|
| $x^2$ | 0.02 |
| $x^3$ | 0.0002 |
| $x^4$ | 0.000002 |

Das Programm läßt sich also sehr schnell an andere Funktionen anpassen; Sie benötigen nur wenig Zeit dazu.
Um aber die entwickelte Formel für C im Programm anzuwenden, jetzt eine kleine Routine, die im Bereich der natürlichen Zahlen $N_0$ von 0 - 11 arbeitet und den von Ihnen angewählten Exponenten in die Formel einsetzt und den Graphen der

Funktion erstellt. Als erstes wird ein Koordinatenkreuz gezeichnet, dann wird nach dem Exponenten gefragt, C wird berechnet und der Graph kann zu Papier gebracht werden.

```
100 REM====1520 SETZEN==================
110 OPEN1,6,1
120 REM====KOORDINATENSYSTEM============
130 FORA=1TO6
140 READX$,Y,Z
150 PRINT#1,X$;Y;Z
160 NEXT
170 DATAM,0,0,D,480,0,M,240,240
180 DATAD,240,-240,M,240,0,I,0,0
190 REM====DEFINITION DER FUNKTION=====
200 INPUT"sWELCHER EXPONENT (N E N)";N$
210 N=VAL(N$):N=ABS(N)
220 IFN>11THEN200
230 FORP=NTO0STEP-1
240 IFP=0THEN270
250 NEXT
260 GOTO200
270 DEFFNA(X)=X↑N
280 C=2/100↑(N-1)
290 REM====ZEICHNEN====================
300 FORA=100TO-100STEP-1
310 IFA=100THENPRINT#1,"R";A;C*FNA(A)
320 PRINT#1,"J";A;C*FNA(A)
330 NEXT
340 PRINT#1,"M";0;0
350 REM====ENDE========================
360 CLOSE1
```

Die Eingaberoutine liegt in diesem Programm von Zeile 200 bis 250, Zeile 220 testet die Eingabe auf Überlauf aus, damit kein Overflow Error gemeldet wird. Die Zeilen von 230 bis 250 beschränken den Zahlenbereich zusätzlich auf $N_0$, den Bereich der natürlichen Zahlen inclusive der Null.
Sind die gemachten Eingaben korrekt, so wird mit $x^n$ in Zeile 270 die Funktion definiert. Die Formel für die Konstante kommt in Zeile 280 zur Anwendung, nur wurde gleich der Faktor mit eingebracht, da der Wertebereich der einzusetzenden Zahlen (Zeile 300) sich zwischen 100 und -100 bewegen. Schreiben Sie für die zwei in der Zeile 280 eine

eins, so wird die Funktion zu gedrungen und zu breit. Die Zeichenroutine kann aus den vorhergegangenen Programmen übernommen werden, es haben sich lediglich die Zeilennummern geändert.

Bevor wir zur Darstellung der Sinus und Cosinuskurve schreiten, noch eine Anregung für die geplagten Menschen, die mit der Kurvendiskussion zu tun haben. Es gibt nämlich Möglichkeiten, die Arbeit zu erleichtern, wenn Zeichnungen angefertigt werden müssen. Dabei sollte man von rein qualitativen Zeichnungen ausgehen, da die geringe Breite des Zeichenpapiers es nicht als besonders angebracht erscheinen läßt, mit der Einheit Zentimeter zu arbeiten.
Das Programm für eine komplette Kurvendiskussion hier zu zeigen, ist aus Platzgründen nicht möglich, da das Programm 61 Blocks belegt und das Listing dementsprechend lang ausfallen würde. Wir können den Teil der komplizierten Berechnungen aber streichen, und uns auf die Zeichnung konzentrieren. Was gilt es hierbei zu beachten?

Bei der Kurvendiskussion ist es üblich, um die Nullstellen, Minima, Maxima oder Wendestellen zu berechnen, zweimal die Grundfunktion abzuleiten.

Dieses soll mit einem Programm gezeigt werden, welches Sie dazu anregen soll, dieses vielleicht auszubauen oder es als Leitfaden für eigene Experimente zu nutzen.

Bei dem Beispielprogramm wollen wir uns auf Funktionen mit ganzen Exponenten beziehen und bis $Y=X^5$ zur Anwendung bringen.

Kurz zur Wiederholung die Regel der Ableitung:
Die Variable mit bestimmtem Exponenten (z. B. $X^4$) wird in

der Art abgeleitet, daß man die Variable mit dem Exponenten multipliziert und den alten Exponenten um eins vermindert ($4*X^3$).

Allgemein gilt also:
$X^n$ ergibt abgeleitet $n*X^{(n-1)}$.

Für die Zweite Ableitung gelten entsprechende Regeln.

Den gesamten Vorgang können Sie auch umkehren, wenn Sie beispielsweise eine abgeleitete Funktionsvorschrift haben, aber die Stammfunktion nicht kennen.

Beispiel: Die erste Ableitung einer Funktion ist mit $3*X^2$ bekannt. Sie können nun die zweite Ableitung bilden und haben die Zeichnungen der ersten und zweiten Ableitungen, aber das Wichtigste, die Grundfunktion fehlt.

Wir haben also $3*X^2$ als erste Ableitung:

Die Stammfunktion einer abgeleiteten Funktion erhält man, indem man den Exponenten um eins erhöht und mit dem Kehrwert des neuen Exponenten den Ausdruck multipliziert. Dies ist so oft durchzuführen, wie die Funktion abgeleitet wurde.

Wenden wir diese Theorie einmal auf die Vorschrift $3*X^2$ an.

1) Erhöhung des Exponenten um 1 ergibt $3*X^{(2+1)}$

2) Multiplizieren mit dem Kehrwert des Exponenten ergibt $(3*X^{(2+1)})/(2+1)$.

Zusammengefaßt erhält man $x^3$, da der Rest gekürzt werden kann. $X^3$ ist also die Stammfunktion zu der gegebenen

Ableitung $3*X^2$.
Prüfen Sie es nach, indem Sie $x^3$ ableiten. Sie erhalten das richtige Ergebnis. Ist Ihnen nur die dritte Ableitung bekannt, so ist der gesamte Vorgang dreimal durchzuführen. Nun aber zu unserem Programm, welches, wie bereits erwähnt, nur Anregungen geben soll und auch kann.

```
100 REM====EINGABE====================
110 INPUT"■DIE FUNKTION BITTE";X$
120 REM====FUNKTION AUSLESEN===========
130 FORX=1TOLEN(X$)
140 Y$=MID$(X$,X,1)
150 Y=VAL(Y$)
160 IFY$=""THENNEXT
170 IFY$="X"THENE=E+1:Z=1
180 IFY$="↑"THENNEXT
190 IFE=1ANDY>0THENA=Y
200 IFE<1ANDY>0THENB=Y
210 IFY$="-"THEN110
220 NEXT
230 REM====ABLEITEN====================
240 PRINT"Q1.)ABLEITUNG";A*B;"*X↑";A-1
250 C=A*B*(A-1)
260 PRINT"Q2.)ABLEITUNG";C;"*X↑";A-2
```

Wie Sie selber sehen, ist das kurze Programm nur als Beispiel gedacht. Es können nur kleine Funktionen eigeben werden, , wie etwa $9*X^7$ etc. Die erste und die zweite Ableitung werden dann ausgegeben.

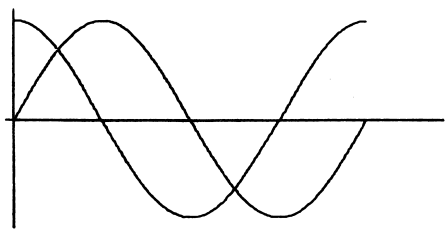
Diese Technik läßt sich aber beliebig ausweiten, so daß auch mit dem entsprechendem Programmieraufwand eine umfassendere Funktion zu bearbeiten ist. Ein solches Programm ist zum einen aber zu speziell und zudem viel zu lang; daher erscheint es nicht sinnvoll, es an dieser Stelle zu zeigen.

Zum Schluß der Darstellungen von mathematischen Funktionen noch die wichtigen trigonometrischen Funktionen Sinus und Cosinus. Da diese beiden Funktionsarten bereits im C 64

vordefiniert sind, ist die Erstellung eines kleinen Programms schnell zu bewerkstelligen.

```
100 REM====1520 SETZEN=================
110 OPEN1,6,1
120 OPEN2,6,2
130 PRINT#2,0
140 REM====KOORDINATENSYSTEM===========
150 PRINT#1,"D";400;0
160 PRINT#1,"M";10;100
170 PRINT#1,"D";10;-100
180 PRINT#1,"M";10;0
190 PRINT#1,"I"
200 REM====FUNKTION====================
210 DEFFNA(X)=SIN(X)
220 REM====PUNKT BERECHNEN/PLOTTEN=====
230 FORX=0TO320
240 PRINT#1,"J";X;FNA(X/51)*90
250 NEXT
260 REM====ENDE========================
270 PRINT#1,"R";0;0
280 CLOSE1
290 CLOSE2
```

In Zeile 210 wird die jeweilige Funktion, hier der Sinus, definiert. Sie können ohne sonstige Veränderungen auch den Cosinus einsetzen. Mit einem geeigneten Vergrößerungsfaktor (Zeile 240) kann man die Kurve nach oben und zur Seite hin ausdehnen. Setzen Sie Sinus und Cosinus nacheinander ein und lassen Sie das Programm laufen, so entsteht folgendes Bild:



Der Sinus ist gezeichnet vom Ursprung an, der Cosinus ist um

90 Grad phasenverschoben, er hat bei 0 sein Maximum erreicht und fällt dann zunächst ab. Wollen Sie die eingeschlossene Fläche der beiden Funktionen ausmalen oder ausschraffieren, so kann die folgende Routine helfen.

```
100 REM====1520 SETZEN==================
110 OPEN1,6,1
120 OPEN2,6,2
130 PRINT#2,0
140 REM====KOORDINATENSYSTEM============
150 PRINT#1,"D";400;0
160 PRINT#1,"M";10;100
170 PRINT#1,"D";10;-100
180 PRINT#1,"M";10;0
190 PRINT#1,"I"
200 REM====FUNKTION====================
210 DEFFNA(X)=SIN(X)
220 DEFFNB(X)=COS(X)
230 REM====PUNKT BERECHNEN/PLOTTEN=====
240 FORX=0TO320
250 PRINT#1,"J";X;FNA(X/51)*90
260 PRINT#1,"J";X;FNB(X/51)*90
270 NEXT
280 REM====ENDE========================
290 PRINT#1,"R";0;0
300 CLOSE1
310 CLOSE2
```

Hier werden in Zeile 210 und 220 die Funktionen Sinus und Cosinus einzeln in Definitionen abgelegt. Dann wird abwechselnd jeweils ein Punkt auf der Sinus und dann auf der Cosinuskurve angesprochen und die Punkte miteinander verbunden. Das Programm in dieser Form malt die Fläche völlig aus. Wollen Sie nur schraffieren, so schreiben Sie die Zeile 240 einfach etwas um und fügen ein STEP ein.
(z.B. FORX=0TO320STEP4)
Dann liegen die berechneten Punkte auf den Graphen weiter auseinander und es entsteht eine Art Schraffur. Dies Programm aber nur nebenbei für interessierte Leser.

Auch die Behandlung der Darstellung von Funktionen konnte

mit den erfolgten Beschreibungen nur vorgestellt, aber nicht vervollkommnet werden. Man kann alleine mit elektrotechnischen Funktionen zwar einen ganzen Band füllen, aber schließlich sollen diese Schilderungen vor allem Anregungen geben und nicht zu speziell werden.

### *8.5 Kleine Besonderheiten mit dem VC-1526*

Bevor mit Statistikfunktionen begonnen werden soll, eine kleine Einführung in ein paar Besonderheiten, die sich auch mit einem Plotter bewerkstelligen lassen. Das Printusing ist nämlich keinesfalls das Privileg eines Druckers oder des Bildschirms; auch der 1520 läßt sich mit einem kleinen Programm dazu bringen, z.B. Zahlen oder auch Text fein säuberlich in einer Spalte zu plotten.

Das Beispielprogramm schreibt jeweils zwei Zahlen in eine Reihe und richtet diese nach ihren Dezimalpunkten aus. Dann soll eine neue Zeile begonnen werden. Sehen Sie sich das neue Programm einmal an:

```
100 REM====1520 SETZEN=================
110 OPEN4,6
120 PRINT#4
130 REM====LAENGE DER ZAHL BESTIMMEN===
140 GZ=4
150 DZ=3
160 A=0
170 REM====EINGABE/VORBEREITUNG========
180 INPUT"ZAHL";NR
190 A=A+1
200 NR$=STR$(NR)
210 NN=LEN(NR$):W=0
220 FORX=1TONN
230 IFMID$(NR$,X,1)="."THEN250
240 W=W+1:NEXT
250 REM====DRUCKEN DER ZAHL============
260 PRINT#4,TAB(GZ+1-W);NR;:
270 PRINT#4,TAB((DZ+GZ)-(GZ-W+NN)+1);:
280 IFA<2THEN180
290 PRINT#4
300 A=0
310 GOTO180
```

140 In der Varialen GZ werden die ganzzahligen Stellen (Vorkommateil) festgelegt.

150 In der Variablen DZ wird die Anzahl der Dezimalstellen festgelegt. Bitte halten Sie sich bei der Eingabe an die Werte, die in Zeile 140/150 vordefiniert wurden!

200

-240 bereitet den Druckvorgang vor, indem die Lage des Dezimalpunktes lokalisiert wird.

260 druckt die Zahl aus und mit Zeile 270 wird der Restbereich bis zum Beginn der nächsten Zahl mit Leerstellen ausgefüllt. Wenn dann die nächste Zeile folgt, so wird zuerst der gesamte Zahlenkomplex der ersten Zahl nach dem Dezimalpunkt ausgerichtet, dann erfolgt der Ausdruck.

Wollen Sie dieses Programm beenden, so können Sie nur mit RUN/STOP-RESTORE aussteigen, da sofort nach einem beendeten Ausdruck eine neue Zahl abgefragt wird! Führen Sie

anschließend ein Carriage-Return durch, d.h. betätigen Sie einmal die Paperfeed-Taste, um die Trommel an Ihren Ausgangspunkt zurückzuholen. Das Printusing läßt sich in sehr vielen Variationsmöglichkeiten nutzen. Man kann beispielsweise noch einen Teilstring mit aufnehmen, um eine Bestandsliste samt Preisen zu realisieren. Dies bewerkstelligt man, indem eine gewisse Textlänge definiert wird; der Text darf z.B. eine Länge von 25 Stellen besitzen. Diese 25 dient dann vor dem nächsten Ausdruck in derselben Zeile als Richtmaß und bis zu dieser Markierung wird der Text mit Leerstellen aufgefüllt, wenn er weniger als 25 Stellen umfaßt.

Die Form der Eingabe könnte dann etwa lauten:

PRINT#4,"TEXT";(Definierte Textlänge=tatsächliche Textlänge)

Das Printusing spielt aber bei dem 1520 keine große Rolle, da er sich mehr den zeichnerischen Aufgaben widmen soll. Dazu werden wir gleich kommen, doch zuvor noch eine Möglichkeit, den 1520 im Direktmodus über die Tastatur als Schreibmaschine anzusprechen. Dies ist besonders dann angebracht, wenn man bei der Ausgabe auf Papier sich nicht schon vorher auf einen Text festlegen will, wie dieses bei der Beschriftung von Zeichnungen etc. der Fall sein wird. Auch bei diesem Problem kommen wir mit einem sehr kleinem Programm aus.

```
100 REM====1520 SETZEN=================
110 OPEN4,6
120 REM====TASTE ABFRAGEN==============
130 PRINT"s"
140 POKE198,0
150 WAIT198,1
160 GETX$:PRINTX$;:
170 IFX$="e"THEN220
180 REM====ZEICHEN DRUCKEN=============
190 PRINT#4,X$;:
200 GOTO140
210 REM====ENDE=======================
220 CLOSE4
```

Die Abwicklung der Eingabeabfrage läuft in diesem Fall über die Abfrage des Tastaturpuffers. Bei $00C6 oder dezimal 198 ist die Zahl der Zeichen im Tastaturpuffer abgelegt.

140 löscht zunächst den Pufferinhalt, indem man eine Null in die Speicherstelle schreibt.

150 hält das Programm genau solange an, bis in der Speicherstelle 198 genau ein Zeichen steht. Würde man z.B. WAIT 198,3 eingeben, so würde erst nach der Eingabe des dritten Zeichens weitergearbeitet werden. Das entsprechnde Zeichen wird dann mit GET gelesen und in Zeile

190 ausgegeben. Das Symbol in der Zeile

170 ist die Abkürzung für die Funktionstaste eins. Wenn Sie diese Taste betätigen, so wird das Programm beendet.

Bei der Statistik mit dem 1520 wird dieser Direktmodus noch benötigt, um ein Säulendiagramm zu beschriften. Nun aber zu dem Programm, mit dessen Hilfe man mit dem Plotter kreativ werden kann: Der "JOYDRAWER".

Dieser schreckliche Ausdruck beinhaltet eigentlich nur die Überlegung, daß man mit den Adressen 56320 oder 56321 (je nach Port) die Richtung bestimmt, in die der Joystick gerade

bewegt wird. Das sieht dann folgendermaßen aus:

```
100 REM====JOYSTICKABFRAGE==============
110 X=PEEK(56320)
120 IF(XAND1)=0THENGOTO *ZEILENNUMMER*
130 IF(XAND2)=0THENGOTO *ZEILENNUMMER*
140 IF(XAND4)=0THENGOTO *ZEILENNUMMER*
150 IF(XAND8)=0THENGOTO *ZEILENNUMMER*
160 IF(XAND16)=0THENGOTO*ZEILENNUMMER*
170 GOTO110
```

Nachdem nun eine der Richtungen festgestellt wurde, schickt man hier das Programm in eine bestimmte Zeile, um es dort ganz bestimmte Befehle ausführen zu lassen.

Dieses kann jedoch nicht das Ziel unserer Bemühungen sein, da sich mit dieser Abfrage ein "joysticktechnisches" Problem offenbart, daß einem die Freude am Zeichnen schon vorher verderben kann. Wie Sie wissen, ist der Joystick insgesamt mit Membrantastern ausgestattet, einem für den Feuerknopf, die anderen für je eine Bewegungsrichtung. Diese Taster werden über Plastikstifte betätigt, die mit dem Hebel in Verbindung stehen.
Wenn man den Hebel in eine diagonale Richtung zieht, so werden jeweils zwei Taster gleichzeitig betätigt, doch davon hat man mit dem obenstehenden Abfrageprogramm nicht das geringste. Denn es können so niemals beide Taster gleichzeitig abgefragt werden, da sich der Computer bei der Abarbeitung nicht aufteilen kann, um zwei GOTO-Befehle gleichzeitig auszuführen.
So wird also der Taster, der Bruchteile von Sekunden früher kommt, bedient, und das Ergebnis ist, daß man auf dem Papier keine Diagonale ziehen kann, sondern nur ein Gebilde, welches aus Ecken besteht und mehr wie eine Treppe aussieht. Wir müssen also zu einer anderen Methode greifen und die Abfrage ohne GOTO abhandeln. Gleichzeitig ist zu beachten,

daß das Abfrageprogramm und die folgende Ausführung so miteinander gekoppelt werden, daß eine einzige Schleife entsteht, die pro Sekunde oft genug durchlaufen wird, um auch zwei Taster abzufragen, damit als Zeichenergebnis eine Diagonale entstehen kann. Eine mögliche Problemlösung bietet das folgende Programm:

```
100 REM====1520 SETZEN=================
110 SW=3:C=2
120 OPEN4,6
130 OPEN1,6,1
140 OPEN2,6,2:PRINT#2,C
150 REM====JOYSTICKABFRAGE=============
160 J=PEEK(56320)
170 IF(JAND1)=0THENY=Y+SW
180 IF(JAND2)=0THENY=Y-SW
190 IF(JAND4)=0THENX=X-SW
200 IF(JAND8)=0THENX=X+SW
210 IF(JAND16)=0ANDKK=0THENKK=1:GOTO240
220 IF(JAND16)=0ANDKK=1THENKK=0
230 REM====UEBERLAUF TESTEN============
240 IFX<0THENX=0:GOTO160
250 IFY<-480THENY=-480:GOTO160
260 IFX>480THENX=480:GOTO160
270 IFY>480THENY=480:GOTO160
280 REM====ZEICHNEN====================
290 IFKK=1THENK$="D"
300 IFKK=0THENK$="M"
310 PRINT#1,K$;X;Y
320 GOTO160
```

110 In zwei Variablen (SW=Schrittweite, C=Farbe) können die Zeichenauflösung und damit auch die Zeichengeschwindigkeit und die Farbe des Kugelschreibers gewählt werden. Schrittweite 1 entspricht der höchsten Auflösung beim Zeichenvorgang.

160

-200 Die vier Grundrichtungen des Joysticks werden abgefragt und, dabei werden die Variablen Y und X je nach Bewegungsrichtung vergrößert oder auch verringert. Zur Orientierung wurden die Variablen entsprechend den

Achsenbezeichnungen gewählt, d.h. bewegen Sie den Hebel des Joysticks nach oben, so wird Y vergrößert (oben=positive Y-Achse).

210 Der Feuerknopf wird hier abgefragt und die Variable KK wird von 0 auf eins umgesetzt. Dies ist erforderlich, damit der Kugelschreiber gesetzt und vom Papier abgenommen werden kann. Ist KK nun eins, so wird Zeile 220 übersprungen. Es folgt ein direkter Einsprung in die Zeichenroutine.

220 Wird der Feuerknopf nun ein zweitesmal betätigt, so wird Zeile 210 übergangen, da die Bedingung für KK=0 nicht erfüllt ist. War der Stift mit KK=1 bis zu diesem Zeitpunkt gesetzt, so wird bei der nächsten Hebelbewegung der Stift vom Papier abgezogen, da KK wieder auf 0 zurückgesetzt wurde.

240

-270 Mit der Überlaufprüfung wird verhindert, daß die Variablen X und Y weiter vergrößert werden, wenn die Trommel eine Randposition erreicht.

290

-300 Der Zeichen- bzw. Bewegungsbefehl (D/M) wird in der Variablen K$ abgelegt, je nachdem wie KK gesetzt ist.

310 Der Zeichenbefehl wird zusammen mit den aktuellen Werten von X und Y ausgeführt, die Routine kann erneut durchlaufen werden.

Haben Sie das Programm abgetippt, so starten Sie es und die Vorteile werden bald klar. Die Zeichengeschwindigkeit läßt sich durchaus vertreten und die Diagonalen werden stufenfrei gezeichnet.

Zur Bedienung vielleicht noch ein paar Worte:
Wird der Feuerknopf gedrückt, so tut sich an der Trommel samt Stiften überhaupt nichts. Erst wenn der Hebel bewegt

wird, schaltet der Stift um. Dies hat den Vorteil, daß während der Zeichenbewegung gleichzeitig der Feuerknopf gedrückt werden kann und so (je nach Schrittweite) auch gestrichelt gezeichnet werden kann, da der Stift immer gsetzt und wieder abgezogen wird. Der Joystick wird im übrigen an Port 2 erwartet.

Alle Leser, die keinen Joystick besitzen, können das Programm auf Tastaturbedienung umstellen. Dazu wird in Zeile 160 die Adresse von 56320 auf 56321 geändert. nun ist auf Port 1 umgestellt worden und Port 1 ist mit den Steuersignalen auch über die Tastatur direkt zu erreichen.

| TASTE | FUNKTION |
|---|---|
| 2 | rechts |
| CTRL | links |
| 1 | oben |
| < | unten |
| SPACE | Feuerknopf |

Das gezeigte Programm soll vor allen Dingen der Dokumentation dienen und die Arbeitsweise der Joystickabfrage zeigen. Nachdem die theoretische Seite abgeklärt ist, kann man die Routine etwas erweitern, damit sie für den Benutzer etwas komfortabler wird. Erreicht wird dies durch die Einbeziehung der Funktionstasten.

```
100 REM===DATEN LESEN=================
110 PRINT"s"
120 FORF=0TO3
130 READB$(F)
140 NEXT
150 REM====1520 SETZEN=================
160 SW=3:C=0
170 OPEN4,6
180 OPEN1,6,1
190 OPEN2,6,2
200 PRINT#2,C
210 REM====JOYSTICKABFRAGE=============
220 GETA$:J=PEEK(56320)
230 IF(JAND1)=0THENY=Y+SW
240 IF(JAND2)=0THENY=Y-SW
250 IF(JAND4)=0THENX=X-SW
260 IF(JAND8)=0THENX=X+SW
270 IF(JAND16)=0ANDKK=0THENKK=1:GOTO350
280 IF(JAND16)=0ANDKK=1THENKK=0
290 IFA$=CHR$(133)THENCLOSE1:CLOSE2:END
300 IFA$=CHR$(134)THENSW=SW+1
310 IFA$=CHR$(138)ANDSW>1THENSW=SW-1
320 IFA$=CHR$(135)THENC=C+1
330 IFC=4THENC=0
340 REM====UEBERLAUF TESTEN============
350 IFX<0THENX=0:GOTO200
360 IFY<-480THENY=-480:GOTO200
370 IFX>480THENX=480:GOTO200
380 IFY>480THENY=480:GOTO200
390 REM====ZEICHNEN====================
400 IFKK=1THENK$="D"
410 IFKK=0THENK$="M"
420 PRINT#1,K$;X;Y
430 PRINT"STEP";SW,B$(C)"
440 PRINT"qq"
450 IFSW<10THENPOKE1070,32
460 GOTO200
470 DATASCHWARZ,BLAU,GRUEN,ROT
```

| TASTE | FUNKTION |
|---|---|
| F1 | Programmende |
| F2 | erhöht Schrittweite um 1 |
| F4 | vermindert Schrittweite um 1 |
| F5 | dreht die Farbtrommel um ein Viertel weiter. |

Somit können im Programm die Farben gewechselt werden und auch die Auflösung ist variabler geworden.

## *8.6 Statistik mit dem VC-1526*

Oft kommt man auch als ein Nichtstatistiker in die Situation, ein paar Zahlen graphisch darstellen zu müssen. Das können z.B. Meßwerte oder andere Angaben sein, die dann meistens in Prozent angegeben sind. Wir wollen im folgenden zwei Möglichkeiten der statistische Darstellung mit dem Plotter 1520 aufzeigen, die wohl am gebräuchlichsten sind und oft benötigt werden.
Das ist zum einen das Säulendiagramm, zum anderen der "Statistikkuchen". Beide Programme nehmen die Werte in Prozent auf und stellen sie auch dementsprechend dar. Beginnen wir mit dem Säulendiagramm:

```
100 REM====1520 SETZEN==================
110 OPEN4,6
120 OPEN1,6,1
130 OPEN2,6,2
140 REM====EINGABE=====================
150 INPUT"{s}WIEVIELE ANGABEN (MAX.7)";A$
160 A=VAL(A$)
170 IFLEN(A$)>1ORA<1ORA>7THEN150
180 FORS=1TOA
190 PRINT
200 PRINTS;". ANGABE (IN %)";:INPUTB(S)
210 IFB(S)>100THEN150
220 NEXT
230 REM====KOORDINATEN=================
240 PRINT#1,"D";450;0
250 PRINT#4,"X"
260 PRINT#1,"M";20;-20
270 PRINT#1,"D";20;520
280 PRINT#4,"Y"
290 PRINT#1,"M";20;-480
300 PRINT#1,"I"
310 REM====MASSTAB ANTRAGEN============
320 FORE=0TO500STEP50
330 PRINT#1,"R";0;E
340 PRINT#1,"J";-10;E
350 NEXT
360 PRINT#1,"H"
370 REM====SAEULEN=====================
380 FORS=1TOA
390 PRINT#1,"R";50;0:PRINT#1,"I"
400 Y(S)=B(S)*5
410 PRINT#1,"J";0;Y(S)
420 PRINT#1,"J";40;Y(S)
430 PRINT#1,"J";40;0
440 NEXT
450 PRINT#1,"R";-A*50;0
460 PRINT#1,"I"
470 PRINT#1,"R";50;-20
480 PRINT#1,"I"
490 REM====BESCHRIFTEN=================
500 POKE198,0:WAIT198,1
510 GETX$:IFX$<>"{e}"THENPRINT#4,X$;:
520 IFX$<>"{e}"THEN500
530 REM====ENDE========================
540 CLOSE1
550 CLOSE2
560 CLOSE4
```

**Sie können mit dem Programm bis zu sieben Säulen zeichnen, die bis zu 10 cm=100% hoch sein dürfen. An der Y-Achse wird**

jeden Zentimeter eine Markierung gesetzt, die also immer genau 10% entspricht. Die Beschriftung wurde hier sehr variabel gestaltet, da nach Beendigung des Zeichenvorgangs auf den Direktmodus (Schreibmaschine) umgeschaltet wird. Sie können dann über die Tastatur unter die Säulen Zahlen, Kennungen etc. schreiben. Das Programm soll aber nun ausführlich erklärt werden:

150 fragt nach der Anzahl der Säulen, die Sie darstellen wollen.

170 prüft nach, ob die Eingabe korrekt gewesen ist.

180

-220 durchläuft eine INPUT-Routine so oft, wie sie Säulen angegeben haben. Bitte achten Sie hier darauf, daß die Eingaben nur numerisch sind, das Programm prüft dies hier nicht nach. Die eingegebenen Werte werden in einem eindimensionalen Feld hinterlegt und in Zeile 210 wird getestet, ob die Zahlen kleiner oder gleich 100% sind. Ist die Eingabe fehlerfrei getätigt worden, so wird mit den Zeilen

240

-300 ein Koordinatensystem mit Achsenbeschriftungen konstruiert. Dieser Programmteil ist jedoch nicht so schwierig, so daß wir zum nächsten Abschnitt übergehen können.

320

-350 ist für die Unterteilung der Y-Achse in 1 cm-Schritte verantwortlich. Da 100% 10 cm entsprechen sollen, wurde in Zeile 320... STEP50 gewählt, da ein Zwischenschritt 0,2 mm umfaßt und 50*0,02 mm einen Zentimeter ergibt.Dies wird insgesamt 10 mal durchgeführt. Die Gesamtstrecke wird ab dann als 100% angesehen. Im Bereich von Zeile

380

-440 können nach geleisteter Vorarbeit jetzt die eigentichen Säulen gezeichnet werden.

Dieser Programmteil soll etwas genauer betrachtet werden, da er besonders wichtig ist, wenn Sie Änderungen etwa der Säulenbreite etc. vornehmen wollen.

380 Die Zeichenschleife wird mit dem ersten Wert begonnen und endet mit A, der Säulenanzahl.

390 Hier wird bei jedem Durchlauf ein neuer relativer Nullpunkt definiert, nachdem die Trommel ein Stück weiter nach rechts gezogen worden ist. Diese Methode wird angewandt, damit die Zeichenroutine gleich bleiben kann, und nur der Zeichenbeginn der einzelnen Säulen verlegt wird.

400 Die in der Input unter B (S) abgelegten Prozentzahlen werden hier in Längeneinheiten des Plotters umgerechnet; Dabei gilt 500 Zeichenschritte = 10 cm = 100%. Mit einem einfachen Dreisatz läßt sich der Faktor ermitteln, mit dem die %-Eingabe zu multiplizieren ist um auch die richtige Längeneinheit zu erhalten.

Es gilt:
500 Schritte = 100%
X Schritte = B(S)%
Somit ist nur X unbekannt, da B(S) ja zuvor eingegeben wurde.
Dreisatz:
500 Schritte 100% = X Schritte B(S)%
daraus folgt dann
B(S)% - 500 Schritte 100% = X Schritte
und so ergibt sich die Formel in Zeile 400.

410

-440 Die Säule wird nun zu Papier gebracht, die Breite ist mit 40 vorgegeben, die Höhe wird mit Y(S) individuell gezeichnet.

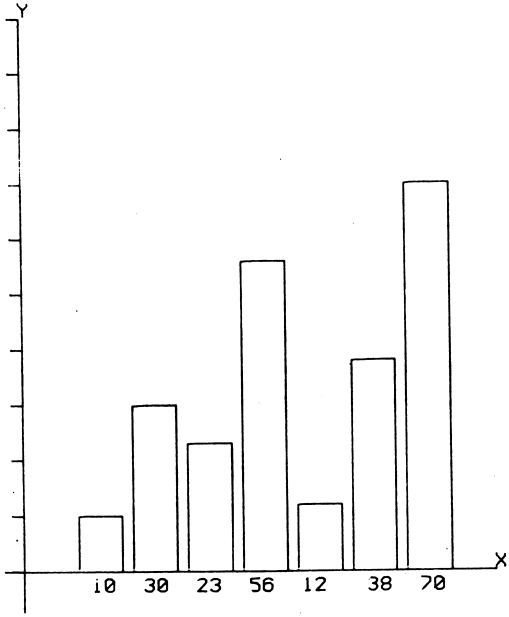
450

-480 Hier wird der Plotter auf den Direktmodus zum Beschriften vorbereitet. Die Kugelschreibertrommel wird unterhalb der ersten Säule positioniert, ein neuer relativer Nullpunkt wird an dieser Stelle festgelegt.

500

-520 Diese Routine erwartet eine Taste, die dann jeweils direkt auf das Papier gedruckt wird (Schreibmaschine). Haben Sie die Beschriftung beendet, so drücken Sie Funktionstaste 1 und das Programm ist beendet.

Zur Wiederholung noch kurz etwas zu Zeile 500. Dort wird der Tastaturpuffer oder besser gesagt die Zeichenanzahl in demselben auf Null gesetzt. Dies geschieht durch einen POKE-Befehl.
Bei 198 ist die Zeichenanzahl im Speicher abgelegt. WAIT 198,1 wartet solange, bis genau ein Zeichen im Puffer steht. Dann wird das empfangene Zeichen mit GET in eine Variable geholt und gedruckt. Die gewählte Programmroutine zeichnet sich durch genaue Zeichenauflösung und eine passable Geschwindigkeit aus. Man kann das Listing verändern, wenn mehr Säulen gewünscht werden, indem in Zeile 390 die 50 entweder erhöht oder vermindert wird und gleichzeitig in Zeile 420/430 die Säulenbreite als solche verändert wird. Unten sehen Sie ein Bild eines Probedurchlaufs mit der Routine.




Aber auch der "Statistikkuchen" ist bei der Darstellung von Zahlen mit Prozentangaben sehr beliebt. Dabei werden die Protentangaben in Teile des Kreisbogens umgesetzt und es entstehen mehr oder weniger große "Kuchenstücke". Diese Kreise sind sehr übersichtlich, da der Gesamtinhalt des Kreises 100% entspricht.
Mit etwas Aufwand ist es möglich, die Arbeit in die Hände des 1520 zu legen; man braucht dann selber nur noch die Prozentangaben einzugeben. Bis es soweit ist müssen jedoch erst einige Probleme ausgeräumt werden, um auch das gewünschte Resultat zu erzielen.
Damit das Programm übersichtlich gestaltet werden konnte, werden zuerst nur die Radien eingezeichnet, dann folgt am Ende der Kreis. Somit können Zeichenungenauigkeiten, die

durch Sinus und Cosinus unumgänglich sind, aufgefangen werden, aber dazu gleich bei der Programmbeschreibung noch mehr. Mit dem abgedruckten Programm lassen sind maximal 30 Prozentangaben auf einmal verarbeiten. Der Benutzer wird zu Anfang nach der Anzahl der Eingaben gefragt. Dann wird die Zeichnung angefertigt.
Dieser Vorgang nimmt etwas Zeit in Anspruch (bei 10 Angaben etwa eine Minute), ist dafür aber trotz eines Näherungsverfahrens sehr zuverlässig und genau!

```
100 REM====1520 SETZEN=================
110 OPEN1,6,1
120 OPEN4,6
130 PRINT#1,"M";240;0
140 PRINT#1,"I"
150 REM====EINGABE====================
160 DIMV(30)
170 INPUT"{SHIFT-Q}{SHIFT-Q}WIEVIELE TEILE <=30!";M$
180 M=VAL(M$):M=ABS(M)
190 FORN=1TOM
200 PRINT"TEIL";N;:
210 INPUTV(N)
220 NEXT
230 REM====TEST AUF UEBERLAUF==========
240 FORN=1TOM
250 P=P+V(N)
260 IFP>100THENCLR:GOTO170
270 PRINT"{SHIFT-S}PRUEFSUMME";TAB(20)P;"   "
280 NEXT:N=0
290 REM====UNTERTEILUNG================
300 N=N+1
310 K=128*V(N)/100
320 FORL=0TO6.4STEP.05:Z=Z+1
330 X=200*SIN(L)
340 Y=200*COS(L)
350 IFZ-K<1ANDZ-K>=0THEN400
360 IFL=0THENPRINT#1,"J";X;Y
370 PRINT#1,"R";X;Y
380 NEXT:GOTO470
390 REM====NAECHSTER WERT==============
400 Z=0
410 PRINT#1,"J";0;0
420 PRINT#1,"R";X;Y
430 N=N+1
440 K=128*V(N)/100
450 GOTO370
460 REM====KREIS ZEICHNEN==============
470 FORL=0TO6.4STEP.05
480 X=200*SIN(L)
490 Y=200*COS(L)
500 IFL=0THENPRINT#1,"R";X;Y
510 PRINT#1,"J";X;Y
520 NEXT
530 REM====BESCHRIFTEN=================
540 PRINT#1,"R";0;200
550 POKE198,0:WAIT198,1
560 GETA$:IFA$="{SHIFT-E}"THEN600
570 PRINT#4,A$;:
580 GOTO550
590 REM====ENDE=======================
600 CLOSE1
610 CLOSE4
620 POKE214,12
630 PRINT
```

Bei diesem etwas komplizierteren Programm lohnt sich eine Dokumentation, da man dann besser die Variationsmöglichkeiten im Programm erkennen kann.

160 Da die Prozentwerte, die der Benutzer eingibt, in einem eindimensionalen Feld hinterlegt werden, dieses aber bei der gezeigten Anwendung auch mehr als die normalerweise zulässigen 11 Eingaben enthalten könnte, muß es dimensioniert werden, damit man keinen Bad Subscript Error erhält. In unserem Programm sind dann durch DIMV(30) insgesamt 31 Eingaben möglich.

170

-220 Die Benutzereingaben werden abgefragt und in dem zuvor dimensionierten Feld abgelegt. Übrigens ist in Zeile 170 nicht etwa ein Fehler unterlaufen, weil in dem kommentierenden Text steht ...<=30, wo doch aber 31 Elemente zulässig sind. Dieses eine verbleibende Element ist eine Sicherheitsreserve, mehr nicht.

240

-280 Hier wird der Feldinhalt aufsummiert und es erfolgt eine Überlaufkontrolle. Dabei sind alle Eingaben in das Feld zugelassen, deren Summe <=100% ist. Nur wenn Sie mit den Prozenten über die 100% - Marke kommen, so würde ja, da der Kreis nach 360 Grad wieder von vorne beginnt in die erste Angabe hereingezeichnet werden. Liegen die Angaben hingegen unter 100%, so ist dies für die Zeichnung nicht weiter problematisch, da der Kreisbogen noch nicht wieder geschlossen worden ist.

310

-350 legt den Absprungwert K fest, bei dem der Kreisbogen verlassen und ein Radius eingezeichnet werden soll. Diese Formel ist jedoch nur in Zusammenhang mit der nächsten Zeile verständlich. Dort wird nämlich die Zeichnung des Kreises vorbereitet mit

FORL = 0T06.4STEP.05

Die Zeichnung besteht also nach 6.4/.05 aus insgesamt 128 Einzelschritten. Und eben diese 128 erscheint dann auch in Zeile 310, wo die Eingabe der Prozentzahl auf Plotterformat umgesetzt wird. So erhält man nach dem Dreisatz für K genau die Stelle auf dem Kreisbogen, die, bezogen auf den Gesamtkreis, der eingegebenen Prozentzahl entspricht. Gleichzeitig wird mit der Zeile 320 bei jedem Durchlauf (von 128) die Zwischensumme Z um eins erhöht.

Dies wird solange wiederholt, bis K ungefähr gleich der Zwischensumme ist, was mit Zeile 350 überprüft wird.

Die Zeilen 330 bis 350 sind dem aufmerksamen Leser sicherlich noch bekannt, sie gehören zur Kreisroutine, wobei die Konstante (hier 200) den Radius fixiert.

360 bereitet bei annähernder Gleichheit von K und Z den Absprung in eine Unterroutine vor, die ab Zeile 400 zu finden ist.

400

-450 Der nächste Eingabewert wird eingelesen, die Konstante neu berechnet und die Zwischensumme zurückgesetzt. Dann erfolgt ein Rücksprung in die Zeichenroutine

470

-520 braucht nicht mehr beschrieben werden. Hier wird nur nach altbewährter Methode der Kreis um die Mittelpunktstrahlen gezeichnet. Auch der Beschriftungsmodus von

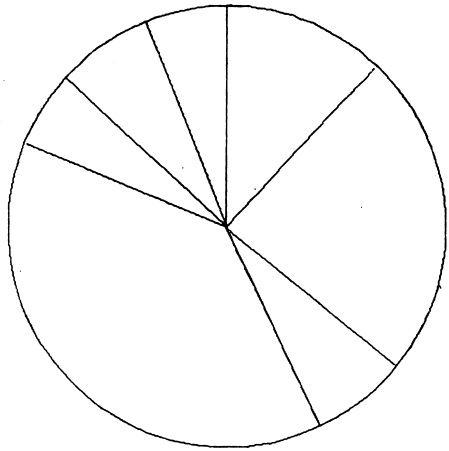
540

-580 ist schon öfter beschrieben worden. Nachdem die Beschriftung beendet ist, können Sie mit der Funktionstaste eins das Programm beenden.

Die Statistikfunktionen mit dem 1520 wurden auch theoretisch deshalb so ausführlich behandelt, da sich hier ein besonders

großes Anwendungsgebiet für einen Plotter auftut. Es müßte nun möglich sein, sich sein eigenes Programm zu entwickeln, daß den eigenen Vorstellungen auch dementsprechend gerecht wird.

Ein Beispiel zeigt noch das Bild unter diesen Ausführungen.



BEISPIEL STATISTIKKUCHEN (UNBESCHRIFTET)

Mit dem folgenden Hilfsprogramm in Basic wartet zwar etwas Tipparbeit auf Sie, aber die Mühe wird sich für jeden lohnen. Das Programm überstützt die Sekundäradressen des 1520 und bietet sogar vorgegebene Routinen wie den Kreis oder die Linie. Zudem besitzt das Programm den Vorteil, daß es nahezu beliebig durch den Benutzer nach eigenen Vorstellungen zu erweitern ist. Die Arbeitsgeschwindigkeit erscheint für BASIC-Verhältnisse noch recht passabel. Der Befehlssatz wurde dahingehend ausgelegt, daß sowohl die englische Eingabe als auch die deusche Version verstanden werden.

Im Anschluß an dieses Programm findet sich eine ausführliche Beschreibung des Programms und der Befehle.

Damit Ihnen die Entscheidung, sich an die Arbeit zu machen, leichter fällt, ein kleiner Auszug aus dem Befehlssatz:

CIRCLE,Mx,My,Rx,Ry,L zeichnet einen Kreis, in dem sowohl der Mittelpunkt, die Radien in X und Y-Richtung als auch die Linienart wählbar ist.

LINE,X1,X2,Y1,Y2,L zeichnet eine Linie von einem Pukt zu einem anderen, wobei die Linienart wählbar ist.

PAPER UP/DOWN,N verschiebt das Papier um n-Schritte nach oben oder unten.

TURN/TEXT dreht einen beliebigen Text um 90 Grad entweder an der X oder der Y-Achse.

Änderungen von Farbe, Zeichengröße etc. sind ebenfalls

selbstverständlich, doch diese Schilderungen folgen später bei der Befehlssatzbeschreibung mit einigen Beispielen.

Doch sehen Sie zuerst das etwas umfangreichere Listing:

```
10 OPEN1,6,1
20 OPEN2,6,2
30 OPEN3,6,3
40 OPEN4;6,4
50 OPEN5,6,5
60 OPEN6,6,6
70 OPEN7,6,7
80 OPEN8,6:PRINT"s":DIMEN$(100):DIMW1(10
0):RESTORE
90 GOSUB9000:POKE53280,0:POKE53281,2:PRI
NT"E"
100 REM====EINGABE====================
110 E$="←":PO=56320
120 PRINT"< ";:
130 GETA$:IFA$=""THEN130
140 IFA$="Q"ORA$="q"ORA$="]"ORA$="↑"ORA$
=CHR$(34)ORA$="s"ORA$="S"THEN130
150 IFA$=CHR$(13)THENPRINT:PRINT"q<":Z=0
:S=S+1:GOTO240
160 IFA$=CHR$(20)THENZ=Z-1:IFZ<0THENZ=0:
GOTO130
170 IFA$=CHR$(20)THEN190
180 Z=Z+1:IFZ>20THENA$=CHR$(13):GOTO150
190 PRINTTAB(3)"↑"A$+E$;:
200 IFA$="x"THEN330
210 T$=T$+A$
220 GOTO130
230 IFA$=CHR$(20)THENTT$=LEFT$(TT$,LEN(T
T$)-2)
240 Y=LEN(T$):E=E+1
250 FORX=1TOY
260 S$=MID$(T$,X,1)
270 IFS$=" "ORS$="↑"ORS$="←"THENNEXT
280 EN$(E)=EN$(E)+S$
290 IFS$<>" "ANDS$<>"↑"ANDS$<>"←"THENNEX
T
300 T$="":A$="":S$=""
310 GOTO120
320 REM====ABARBEITEN=================
330 E=1
350 REM====BEFEHL LESEN================
360 FORX=1TOLEN(EN$(E))
370 BF$=MID$(EN$(E),X,1)
380 IFBF$<>","THENB1$=B1$+BF$:NEXT
390 REM====ZAHLEN LESEN================
400 FORX=LEN(B1$)+2TOLEN(EN$(E))
```

```
410 Z1$=MID$(EN$(E),X,1)
420 IFZ1$<>","THENN1$=N1$+Z1$:NEXT
430 REM====ZAHLEN LESEN================
440 FORX=LEN(N1$)+2+LEN(B1$)+1TOLEN(EN$(
E))
450 Z2$=MID$(EN$(E),X,1)
460 IFZ2$<>","THENN2$=N2$+Z2$:NEXT
470 REM====ZAHLEN LESEN================
480 FORX=LEN(N1$)+1+LEN(B1$)+1+LEN(N2$)+
2TOLEN(EN$(E))
490 Z3$=MID$(EN$(E),X,1)
500 IFZ3$<>","THENN3$=N3$+Z3$:NEXT
510 REM====ZAHLEN LESEN================
520 FORX=LEN(N1$)+1+LEN(B1$)+1+LEN(N2$)+
1+LEN(N3$)+2TOLEN(EN$(E))
530 Z4$=MID$(EN$(E),X,1)
540 IFZ4$<>","THENN4$=N4$+Z4$:NEXT
550 REM====ZAHLEN LESEN================
570 FORX=LEN(N1$)+1+LEN(B1$)+1+LEN(N2$)+
1+LEN(N3$)+1+LEN(N4$)+2TOLEN(EN$(E))
580 Z5$=MID$(EN$(E),X,1)
590 IFZ5$<>" "THENN5$=N5$+Z5$:NEXT
600 REM====STRINGS IN ZAHLEN===========
610 N1=VAL(N1$):N1=ABS(N1)
620 N2=VAL(N2$):N2=ABS(N2)
630 N3=VAL(N3$):N3=ABS(N3)
640 N4=VAL(N4$):N4=ABS(N4)
650 N5=VAL(N5$):N5=ABS(N5)
660 REM====AUSFUEHRUNG=================
670 REM====DER EINGEGEBENEN BEFEHLE====
680 IFB1$<>"COLOR"ANDB1$<>"FARBE"THENGOT
0710
690 IFN1>255THENPRINT#1,0:GOTO710
700 PRINT#2,N1
710 IFB1$<>"PEN"ANDB1$<>"STIFT"THEN740
720 IFN1>255THENN1=0
730 PRINT#2,N1:PRINT#1,"M";480,0
740 IFB1$<>"LINE"ANDB1$<>"LINIE"THEN780
750 IFN1>480ORN2>999ORN3>480ORN4>999ORN5
>15THEN780
760 PRINT#5,N5
770 PRINT#1,"M";N1;N2:PRINT#1,"D";N3;N4
780 IFB1$<>"SIGN"ANDB1$<>"ZEICHEN"THENGO
TO810
790 IFN1>255THENN1=1
800 PRINT#3,N1
810 IFB1$<>"TEST"THEN830
820 PRINT#8,"T E S T"
830 IFB1$<>"PAPERUP"ANDB1$<>"PAPIERAUF"T
HEN870
840 IFN1>999THENN1=998
850 PRINT#1,"M";0;N1
860 PRINT#8
870 IFB1$<>"PAPERDOWN"ANDB1$<>"PAPIERAB"
THEN910
```

```
880 IFN1>999THENN1=998
890 PRINT#1,"M";0;-N1
900 PRINT#8
910 IFB1$<>"HOME"ANDB1$<>"HEIM"THEN930
920 PRINT#1,"H"
930 IFB1$<>"CIRCLE"ANDB1$<>"KREIS"THENGO
TO1010
940 IFN1+N3>480ORN2+N4>999ORN5>15ORN1-N3
<0THENPRINT"WERTE ZU HOCH"
950 PRINT#1,"M";N1;N2:PRINT#1,"I":PRINT#
5,N5
960 FORX=-3.2TO3.2STEP.04
970 Y=N3*COS(X):Z=N4*SIN(X)
980 IFX=-3.2THENPRINT#1,"R";Y;Z
990 PRINT#1,"J";Y;Z:NEXT
1000 PRINT#1,"M";0;0:PRINT#1,"I"
1010 IFB1$<>"BASIC"THEN1030
1020 SYS64738
1030 IFLEFT$(B1$,5)<>"TURN-"ANDLEFT$(B1$
,5)<>"DREH-"THEN1090
1040 FORX=LEN(B1$)TO6STEP-1:PRINT#4,1
1050 Y$=MID$(B1$,X,1):
1060 PRINT#1,"M",N1*(LEN(B1$)-X)
1070 PRINT#8,Y$;:NEXT:PRINT#4,0
1080 PRINT#8
1090 IFLEFT$(B1$,5)<>"TURN/"ANDLEFT$(B1$
,5)<>"DREH/"THEN1130
1100 PRINT#4,1:FORX=6TOLEN(B1$)
1110 Y$=MID$(B1$,X,1)
1120 PRINT#8,Y$:NEXT:PRINT#8:PRINT#4,0
1130 IFB1$<>"SCREEN"ANDB1$<>"BILD"THEN11
60
1140 IFN1>15ORN2>15THENN1=0:N2=2
1150 POKE53280,N1:POKE53281,N2
1160 IFB1$<>"0-SET"THEN1190
1170 PRINT#2,0:PRINT#3,1:PRINT#4,0:PRINT
#5,0:PRINT#6,0
1180 POKE53280,0:POKE53281,3:PRINT"E"
1190 IFB1$<>"DIRECT"ANDB1$<>"DIREKT"THEN
1250
1200 IFN1>255ORN2>255THEN1230
1210 PRINT#6,N1:PRINT#3,N2
1230 POKE198,0:WAIT198,1
1240 GETA$:IFA$<>""THENPRINTA$;:PRINT#8
,A$;:GOTO1230
1250 IFB1$<>"PORT"THEN1290
1260 IFN1>2ORN1<1THENPO=56320
1270 IFN1=1THENPO=56321
1280 IFN1=2THENPO=56320
```

```
1290 IFB1$<>"JOY"THEN1680
1320 PRINT:RESTORE:FORF=0TO3
1330 READB$(F)
1340 NEXT
1350 REM====1520 SETZEN================
1360 SW=3:C=0
1400 PRINT#2,C
1410 REM====JOYSTICKABFRAGE============
1420 GETA$:J=PEEK(PO)
1430 IF(JAND1)=0THENY=Y+SW
1440 IF(JAND2)=0THENY=Y-SW
1450 IF(JAND4)=0THENX=X-SW
1460 IF(JAND8)=0THENX=X+SW
1470 IF(JAND16)=0ANDKK=0THENKK=1:GOTO155
0
1480 IF(JAND16)=0ANDKK=1THENKK=0
1490 IFA$=CHR$(133)THEN8000
1500 IFA$=CHR$(134)THENSW=SW+1
1510 IFA$=CHR$(138)ANDSW>1THENSW=SW-1
1520 IFA$=CHR$(135)THENC=C+1
1530 IFC=4THENC=0
1540 REM====UEBERLAUF TESTEN===========
1550 IFX<0THENX=0:GOTO1400
1560 IFY<-480THENY=-480:GOTO1400
1570 IFX>480THENX=480:GOTO1400
1580 IFY>480THENY=480:GOTO1400
1590 REM====ZEICHNEN==================
1600 IFKK=1THENK$="D"
1610 IFKK=0THENK$="M"
1620 PRINT#1,K$;X;Y
1630 PRINT"STEP";SW,B$(C)"
1640 PRINT"qq"
1650 IFSW<10THENPOKE1070,32
1660 GOTO1400
1670 DATASCHWARZ,BLAU,GRUEN,ROT
1680 IFB1$<>"PAUSE"THEN8000
1690 WAIT203,63
8000 REM====RUECKSETZEN===============
8010 N1$="":N2$="":N3$="":N4$="":N5$=""
8020 X$="":Y$="":N1=0:N2=0:N3=0:N4=0:N5=
0
8030 E=E+1:IFB1$<>""THENB1$="":GOTO350
8040 CLR:RUN
9000 REM====STATUSZEILE================
9010 PRINT"sE   xxx PLOTTER 1520 HILFSPR
OGRAMM xxx"
9020 RETURN
```

Nun, nachdem Sie die Abtipparbeit geleistet haben, kurz etwas Grundsätzliches zum Programm. Sie sehen von Zeile 10 bis Zeile 650 die Eingabe und Vorbereitungsroutine des Programms. Dann folgt bis Zeile 1690 der Befehlsvorrat samt den Ausführungen. Zeile 8000 bis 8040 setzen das Programm zurück, in 9000ff findet man die Statuszeile. Alle diese Teilprogramme werden nacher noch ausführlich erklärt, da sehr viele Stringoperationen vorkommen.
Das Programm ist in der Konzeption so geplant worden, daß der Anwender eigene Vorstellungen mit einbringen kann, das heißt, er ist in der Lage, eigene Befehle zu definieren. Dazu ist von Zeile 1700 bis Zeile 7990 Freiraum gelassen worden. Zur Verdeutlichung der Programmierweise sind schon einige Befehle eingebaut worden, damit das Programm erprobt werden kann. Der Befehlssatz ist das Thema der folgenden Ausführungen.

Das Programm akzeptiert pro Befehl jeweils ein Befehlswort und darauf folgend bis zu fünf Zahlenangaben. Diese Eingaben müssen durch ein Komma getrennt werden. (z.B. CIRCLE, 100,100,90,80,0). An Befehlen sind im Programm enthalten:

COLOR, x x=0=schwarz
x=1=blau
x=2=grün
x=3=rot

PEN,x fährt die Trommel mit der gewählten Stiftfarbe auf die Auswechselposition am rechten Walzenende. Für x gelten dieselben Angaben, wie bei COLOR.
LINE,x1,x1,x2,y2,L zeichnet eine Linie von den Anfangskoordinaten x1,y2 zu den Endkoordinaten x2,y2 mit der Linierart L. L ist aus den vorherigen Ausführungen bereits bekannt und darf zwischen 0 und 15 variiert werden.

SIGN,x stellt die Zeichengröße ein. Werte zwischen 0 (kleinste Schrift) und 3 (größte Schrift) sind sinnvoll.

TEST schreibt das Wort Test auf das Papier.

PAPER UP,x schiebt das Papier um x Plotterschritte, (0,2mm pro Schritt) nach oben.

PAPER DOWN,x schiebt das Papier um x Plotterschritte nach unten.

HOME fährt die Trommel an ihre Ausgangsposition zurück, und bezihet sich jeweils auf den absoluten oder relativen Nullpunkt.

CIRCLE,Mx,My,Rx,Ry,L zeichnet Kreis oder Ellipse auf. Dabei ist:
Mx = Mittelpunktskoordinate auf der x-Achse
My = Mittelpunktskoordinate auf der y-Achse
Rx = Radius in x-Richtung in Zeichenschritten
Ry = Radius in y-Richtung in Zeichenschritten
L = Linienart (0-15)

BASIC führt Systemreset aus, die können mit diesem Befehl das Programm beenden und löschen.

TURN-TEXT,x dreht das Wort Text oder eine beliebige andere Eingabe an der X-Achse um 90 Grad, wobei die Zeichen mit einem Abstand von x-Zeichenschritten gedruckt werden.

TURN/TEXT dreht das Wort oder jede andere Eingabe an der y-Achse um 90 Grad.

SCREEN,r,b löscht die Bildschirmfarbe und ändert sie mit den Farbwerten 0-15 für r und b um. Dabei ist r der Rahmen, b das Bild. Die Farben entsprechen folgenden Zahlen.

0 schwarz 8 orange
1 weiß 9 braun
2 rot 10 hellrot
3 türkis 11 grau (1.Stufe)
4 violett 12 grau (2.Stufe)
5 grün 13 hellgrün
6 blau 14 hellblau
7 gelb 15 grau (3. Stufe)
0-SET setzt alle Werte auf die Defaultwerte (Grundwerte) zurück.

DIRECT,x,y erlaubt es, den 1520 als Schreibmaschine zu benutzen;
x = 0 Grosschrift, über Shift Kleinschrift
x = 1 Kleinschrift über Shift Grosschrift
y kann zwischen 0 und 3 liegen und setzt die Zeichengröße fest.

PORT,x wählt den Joystickport, für x=1 oder 2. Eingestellt ist vom Programm aus der PORT 2.
JOY schaltet auf Joystickbenutzung um, wobei die Funktionstasten folgende Bedeutung haben.

F1 = weiter im Programm

F3 = Schrittweite um 1 erhöhen

F4 = Schrittweite um 1 vermindern

F5 = Farbwechsel

PAUSE stopt das Programm solange, bis eine Taste betätigt wird.
Soweit der Demonstrationsbefehlssatz, den Sie beliebig erweitern können, aber dazu später noch etwas.

Sie können einen Befehl eintippen. Dann drücken Sie die Return-Taste. Zur Ausführung des Befehls ist dann das x zu drücken: Der Befehl wird ausgeführt und das Programm ist für eine neue Eingabe parat. Drücken Sie nicht das x, so erwartet das Programm den nächsten Befehl, den es an den ersten anhängt. Auf diese Weise können Programme mit bis zu 100 Befehlen geschrieben werden.

Zum Programm selbst:

Zeile 10 - 90 bereiten das Programm vor, es werden alle möglichen Plotterspezifischen Files geöffnet, das Feld zur Programmspeicherung wird dimensioniert. Mit Einsprung nach Zeile 9000 wird die Statuszeile gesetzt, dann wird der Bildschirm gesetzt.

Die Eingaberoutine ab Zeile 100 ist eine Stingoperation, die Zeichen werden mit GET einzeln abgefragt. Zeile 140 blockiert die Annahme von Cursortasten und Bildschirmoperationen wie CLR.
In Zeile 210 wird in der Variablen T$ das gesamte Eingegebene abgelegt. Pro Befehl werden maximal 70 Zeichen zugelassen, dann erfolgt ein AUTO-RETURN.
Zeile 200 überprüft, ob der Befehl zur Abarbeitung gegeben wurde. Andernfalls ergibt sich automatisch ein Rücksprung zur Aufnahme des nächsten Befehls. Wird der Abarbeitungsbefehl erteilt, so erfolgt der Absprung nach 330.

Mit jedem RETURN erfolgt von der Zeile 150 aus ein Sprung nach Zeile 240ff, wo der Befehl in einem Feld (eindimensional) mit der Variablen EN$ (E) abgelegt wird. Dabei wird nun aber der Befehl aus T$ nicht einfach übergeben, sondern es werden schon Vorbereitungen für die Abarbeitung getroffen.
Zeile 270 siebt nämlich Leerstellen, den Pfeilcursor und den Hochpfeil aus, um den Befehl zu komprimieren. Geben Sie z.B. anstatt CIRCLE... etwa C IR C LE... ein, so werden alle Leerstellen entfernt. Unsere Eingaben ruhen jetzt als ein String, der ein Gemisch aus Zahlen und Befehlswort ist, im Feld EN$(E).

Wird jetzt nach Beendigung aller Eingaben das * betätigt, so wird nach Zeile 330 gesprungen, wo die Leseroutine steht. Diese hat die Aufgabe, den Befehl und die fünf Zahlenangaben voneinander zu trennen und sie abzuspeichern. Jetzt wird auch klar, warum alle Eingaben durch ein Komma zu trennen sind. Dieses dient als Indikator für Befehlsende-Neubeginn.

Von Zeile 350-380 wird zuerst das Befehlswort gelesen, da es zu Beginn des Strings liegt. Die gelesenen Variablen werden in der Variablen B1$ hinterlegt, bis Zeile 380 ein Komma feststellt. Dann wird die erste Zahl gelesen, die zunächst in der Variablen N1$ landet. Die übrigen folgen in N2$ - N5$.

Zeile 600-650 stellt nun die rein numerischen Variablen dar; die Strings werden umgewandelt, und die Zahlen liegen nun von N1 bis N5

Jetzt ist die Eingaberoutine zunächst beendet. Die eigenen Befehle können abgearbeitet werden, was bei uns ab Zeile 660 geschieht. Wollen Sie also das Programm erweitern, so

arbeiten Sie mit der eben gezeigten Variablenreihenfolge. Soll z.B. ein Befehl wie
CUBE,Seitenlänge,anfangspunkt,Farbe,Drehung,Endposition
definiert werden, so liegen die Zahlen in den folgenden Variablen.

```
CUBE          =B1$
Seitenlänge   =N1
Anfangspunkt  =N2
Farbe         =N3
Drehung       =N4
Endposition   =N5
```

Jetzt können die Variablen verarbeitet werden. Bei einem Engabefehler wird der Befehl übersprungen. Die Abarbeitung des nächsten beginnt.

Zeile 8000ff setzt das Programm mit fast allen Variablen zurück, der nächste Befehl wird aus dem Feld gelesen.

Natürlich werden die Befehle, die das Programm bis jetzt beinhaltet, auch in deutscher Sprache empfangen.

```
COLOR       = FARBE
PAPER DOWN  = PAPIER AB
PAPER UP    = PAPIER AUF
PEN         = STIFT
LINE        = LINIE
HOME        = HEIM
CIRCLE      = KREIS
SCREEN      = BILD
DIRECT      = DIREKT
SIGN        = ZEICHEN
```

Für Ihre Programmierung eigener Befehle soll hier noch viel

Erfolg gewünscht werden. Zum Abschluß noch ein paar kleine Beispiele mit dem Hilfsprogramm. Ansonsten bleibt zu hoffen, daß die Ausführungen über den Plotter 1520 Ihnen das Gerät nähergebracht haben. Vielleicht konnten nur Anregungen gegeben werden, aber die gezeigten Programme sind so dokumentiert worden, daß sie verstanden werden können - somit sollte man von einem Nur-Abtippen weggeführt werden, hin zu mehr Kreativität.

# DATA BECKER'S NEUE BÜCHER UND PROGRAMME FÜR COMMODORE

## Spickzettel ade.

Ein neues DATA BECKER BUCH, das den Einsatz des COMMODORE 64 in der Schule entscheidend mitprägen dürfte, wurde von Professor Voß geschrieben. Besonders für Schüler der Mittel- und Oberstufe geschrieben, enthält das Buch viele interessante Problemlösungs- und Lernprogramme, die besonders ausführlich und leicht verständlich beschrieben sind. Sie ermöglichen ein intensives und anregendes Lernen, unter anderem mit folgenden Themen: Satz des Pythagoras, quadratische Gleichungen, geometrische Reihen, Pendelbewegungen, mechanische Hebel, Molekülbildung, exponentielles Wachstum, Vokabeln lernen, unregelmäßige Verben, Zinseszinsrechnung. Ein kurzer Überblick über die Grundlagen der EDV, eine knappe Wiederholung der wichtigsten BASIC-Elemente und eine Einführung in die Grundzüge der Problemanalyse vervollständigen das Ganze. Mit diesem Buch machen die Hausaufgaben wieder Spaß!

DAS SCHULBUCH ZUM COMMODORE 64, 1984, über 300 Seiten, DM 49,–

## Macht Druck.

DAS GROSSE DRUCKERBUCH für Drucker-Anwender mit COMMODORE-Computern ist endlich da! Es enthält eine riesige Sammlung von Tips & Tricks, Programmlistings und Hardwareinformationen. Rolf Brückmann und Klaus Gerits beschäftigen sich mit Sekundäradressen, Anschluß einer Schreibmaschine am Userport, Druckerschnittstellen (Centronics, V 24, IEC-Bus), hochauflösender Grafik, Text- und Grafikhardcopy, Grafik mit Standardzeichensatz, formatierter Datenausgabe, Plakatschrift, Textverarbeitung und vieles mehr. Zusätzlich wird das Betriebssystem des MPS801 zerlegt, mit Prozessorbeschreibung (8035), Blockschaltbild und einem ausführlich kommentierten ROM-Listing. Thomas Wiens schrieb den Teil über die Programmierung des Plotters VC-1520: Handhabung des Plotters, Programmierung von Sonderzeichen, Funktionendarstellung, Kuchen und Säulendiagramme, Entwurf dreidimensionaler Gegenstände. Natürlich wieder viele interessante Listings. Unentbehrlich für jeden, der einen COMMODORE 64 oder VC-20 und einen Drucker besitzt.

DAS GROSSE DRUCKERBUCH, 1984, über 300 Seiten, DM 49,–

## Tempo!

MASCHINENSPRACHE FÜR FORTGESCHRITTENE ist bereits das zweite Buch von Lothar Englisch zum Thema Maschinenprogrammierung mit dem COMMODORE 64. Hier wird von der Problemanalyse bis zum Maschinensprachealgorithmus in die Grundlagen der professionellen Maschinenspracheprogrammierung eingeführt. In diesem Buch finden Sie unter anderem folgende Themen behandelt: Problemlösungen in Maschinensprache, Programmierung von Interruptroutinen, Interruptquellen beim COMMODORE 64, Interrupts durch CIA's und Videocontroller, Programmierung der Ein-Ausgabe-Bausteine, die CIA's des COMMODORE 64, Timer, Echtzeituhr, parallele und serielle Ein/Ausgabe, BASIC-Erweiterungen, Programmierung eigener BASIC-Befehle und -Funktionen, Möglichkeiten zur Einbindung ins Betriebssystem sowie viele weitere Tips & Tricks zur Maschinenprogrammierung. Dieses Buch sollte jeder haben, der wirklich intensiv mit der Maschinensprache des COMMODORE 64 arbeiten will.

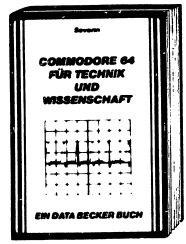
MASCHINENSPRACHE FÜR FORTGESCHRITTENE, 1984, ca. 200 Seiten, DM 39,–

## Tausendsassa.

Fast alles, was man mit dem COMMODORE 64 machen kann, ist in diesem Buch ausführlich beschrieben. Es ist nicht nur spannend zu lesen wie ein Roman, sondern enthält neben nützlichen Programmlistings vor allem viele, viele Anwendungsmöglichkeiten des C64. Dabei wurde besonderer Wert darauf gelegt, daß das Buch auch für Laien leicht verständlich ist. Eine Auswahl aus der Themenvielfalt: Gedichte vom Computer, Einladung zur Party, Diplomarbeit – professionell gestaltet, individuelle Werbebriefe, Autokosten im Griff, Baukostenberechnung, Taschenrechner, Rezeptkartei, Lagerliste, persönliches Gesundheitsarchiv, Diätplan elektronisch, intelligentes Wörterbuch, kleine Notenschule, CAD für Handarbeit, Routenoptimierung, Schaufensterwerbung, Strategiespiele. Teilweise sind Programmlistings fertig zum Eintippen enthalten, soweit sich die „Rezepte" auf 1–2 Seiten realisieren ließen. Wenn Sie bisher nicht immer wußten, was Sie mit Ihrem 64er alles anfangen sollten, nach dem Lesen des IDEENBUCHES wissen Sie's bestimmt!

DAS IDEENBUCH ZUM COMMODORE 64, 1984, über 200 Seiten, DM 29,–

# Prof. 64.



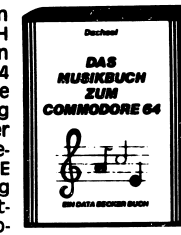

Ein faszinierendes Buch, um in die Welt der Wissenschaft einzusteigen, hat Rainer Severin geschrieben. Zunächst werden Variablentypen, Rechengenauigkeit und nützliche POKE-Adressen des COMMODORE 64 bezüglich den Anforderungen wissenschaftlicher Probleme analysiert. Verschiedene Sortieralgorithmen wie Bubble, Quick und Shell-Sort werden miteinander verglichen. Die Programmbeispiele aus der Mathematik nehmen dabei eine zentrale Stelle im Buch ein: Nullstellen nach Newton, numerische Ableitung mit dem Differenzenquotienten, lineare und nichtlineare Regression, Chi-Quadrat-Verteilung und Anpassungstest, Fourieranalyse und -synthese, Skalar-, Vektor- und Spatprodukt, ein Programmpaket zur Matrizenrechnung für Inversion, Eigenwerte und vieles weitere mehr. Programme aus der Chemie (Periodensystem), Physik, Biologie (Schadstoffe in Gewässern – Erfassung der Meßwerte), Astronomie (Planetenpositionen) und Technik (Berechnung komplexer Netzwerke, Platinenlayout am Bildschirm) und viele weitere Softwarelistings zeigen die riesigen Möglichkeiten auf, die der Computer in Wissenschaft und Technik hat.

COMMODORE 64 FÜR TECHNIK UND WISSENSCHAFT, 1984, über 200 Seiten, DM 49,–

# Grundkurs.

Das neue BASIC-Trainingsbuch zum C-64 ist eine ausführliche, didaktisch gut geschriebene Einführung in das CBM BASIC V2. Alle Befehle werden ausführlich erläutert. Dieses Buch geht aber über eine reine Befehlsbeschreibung hinaus, es wird eine fundierte Einführung in die Programmierung gegeben. Von der Problemanalyse bis zum fertigen Algorithmus lernt man das Entwerfen eines Programmes und den Entwurf von Datenflußplänen. ASCII-Code und verschiedene Zahlensysteme wie hexadezimal, binär und dezimal sind nach der Lektüre des Buches keine Fremdworte mehr. Die Programmierung von Schleifen, Sprüngen, bedingten Sprüngen lernt man leicht durch „learning by doing". So enthält das Trainingsbuch viele Aufgaben, Übungen und unzählige Beispiele. Den Schluß des Buches bildet eine Einführung ins professionelle Programmieren, in der es um mehrdimensionale Felder, Menuesteuerung und Unterprogrammtechnik geht. Endlich ein Buch, das Ihnen wirklich hilft, solide und sicher BASIC zu lernen.

BASIC TRAININGSBUCH ZUM COMMODORE 64, 1984, ca. 250 Seiten, DM 39,–

# Sang und Klang!

Der COMMODORE 64 ist ein Musikgenie. DAS MUSIKBUCH hilft Ihnen, die riesigen Klangmöglichkeiten des C64 zu nutzen. Die Themenbreite reicht von einer Einführung in die Computermusik über die Erklärung der Hardwaregrundlagen des COMMODORE 64 und die Programmierung in BASIC bis hin zur fortgeschrittenen Musikprogrammierung in Maschinensprache. Einiges aus dem Inhalt: Soundregister des COMMODORE 64, Gate-Signal, Programmierung der "ADSR"-Werte, Synchronisation und Ring-Modulation, Counterprinzip, lineare und nichtlineare Musikprogrammierung, Frequenzmodulation, Interrupts in der Musikprogrammierung und vieles mehr. Zahlreiche Beispielprogramme, komplette Songs und nützliche Routinen ergänzen den Text. Geschrieben wurde das Buch von Thomas Dachsel, dem Autor der weltbekannten Musikprogramme Synthimat und Synthesound. Erschließen Sie sich die Welt des Sounds und der Computermusik mit dem Musikbuch zum C-64!

DAS MUSIKBUCH ZUM COMMODORE 64, über 200 Seiten, DM 39,–

# Nützlich.

Das Trainingsbuch zu MULTIPLAN bietet eine gute Einführung in die Grundlagen der Tabellenkalkulation. Dabei wird großer Wert auf ein möglichst schnelles Einarbeiten in die wichtigsten Befehle gelegt, so daß man bald sicher mit MULTIPLAN arbeiten kann, ob nun auf dem COMMODORE 64 oder einem anderen Rechner. Am Ende wird man in der Lage sein, den umfangreichen Befehlssatz von MULTIPLAN auch kommerziell zu nutzen. Übungen am Ende jedes Kapitels sorgen dafür, daß man das Gelernte lange behält. Grundlage des Buches sind viele Seminare, die der Autor zu MULTIPLAN konzipiert und erfolgreich durchgeführt hat.

DAS TRAININGSBUCH ZU MULTIPLAN, 1984, ca. 250 Seiten, DM 49,–

# Für Tüftler.

Ein hochinteressantes Buch für Hobbyelektroniker hat Rolf Brückmann vorgelegt. Er ist ein engagierter Techniker, für den der Computer Hobby und Beruf zur gleichen Zeit ist. Vor allem aber kennt er den C-64 in- und auswendig. So werden einführend die Schnittstellen des COMMODORE 64 detailliert beschrieben und kurz die Funktionsweise der CIAs 6526 erläutert. Hauptteil des Buches sind die Beschreibungen der vielfältigen Einsatzmöglichkeiten des COMMODORE 64. Die vielen Schaltungen, von Rolf Brückmann alle selbst

entwickelt, sind jeweils umfangreich dokumentiert und leichtverständlich erklärt. Die Reihe der hier ausführlich behandelten Anwendungen mit dem COMMODORE 64 ist äußerst umfangreich: Motorsteuerung, Stoppuhr mit Lichtschranke, Lichtorgel, A/D-Wandler, Spannungsmessung, Temperaturmessung und vieles mehr. Dazu kommen noch eine Reihe kompletter Schaltungen zum Selberbauen, wie ein EPROM Programmiergerät für den C-64, eine EPROM-Karte, ein Frequenzzähler und Sprachein/ausgabe (!). Zusätzlich sind jeweils Schaltplan, Softwarelisting und zu einigen Schaltungen sogar zusätzlich Platinenlayouts vorhanden.

DER COMMODORE 64 UND DER REST DER WELT, 1984, ca. 220 Seiten, DM 49,–

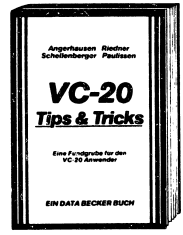
## Computerkünstler.

Das Grafikbuch zum COMMODORE 64 Buch aus der Bestseller-Serie von DATA BECKER stammt aus der Feder von Axel Plenge. Es geht weit über die reine Hardware-Beschreibung der Grafikeigenschaften des C-64 hinaus. Der Inhalt reicht von den Grundlagen der Grafikprogrammierung bis zum Computer Aided Design. Es ist ein Buch für alle, die mit ihrem C-64 kreativ tätig sein wollen. Themen sind z.B.: Zeichensatzprogrammierung, bewegte Sprites, High-Resolution, Multicolor-Grafik, Lightpenanwendungen, Betriebsarten des VIC, Verschieben der Bildschirmspeicher, IRQ-Handhabung, 3-Dimensionale Grafik, Projektionen, Kurven-, Balken- und Kuchendiagramme, Laufschriften, Animation, bewegte Bilder. Viele Programmlistings und Beispiele sind selbstverständlich. Das COMMODORE-BASIC V2 unterstützt die herausragenden Grafikeigenschaften des C-64 bekanntlich kaum. Hier helfen die vielen Beispielprogramme in diesem Buch weiter, die die faszinierende Welt der Computergrafik jedermann zugänglich machen. Kompetent ist der Autor dazu wie kaum ein anderer, schließlich hat er das äußerst leistungsfähige Programm SUPERGRAFIK geschrieben.

DAS GRAFIKBUCH ZUM COMMODORE 64, 1984, 295 Seiten, DM 39,–

## Vielfalt.

Auf dem neuesten Stand ist VC-20 TIPS & TRICKS von Dirk Paulissen gebracht worden, der über hundert Seiten hinzufügte. Bisher schon enthalten waren Informationen über Speicheraufbau des VC-20 und die Erweiterungsmöglichkeiten, ein Grafikkapitel über programmierbare Zeichen, Laufschrift und die Supererweiterung. Stark erweitert wurde der Abschnitt über POKEs und andere nützliche Routinen. Ob es um die Programmierung der Funktionstasten, Programme die sich selber starten, „Maus"-Simulation mit dem Joystick oder die Änderung von Speicherbereichen geht, man ist immer wieder über die Fülle der Möglichkeiten erstaunt. Der Clou dieses Buches sind aber die vielen Programmlistings. Die BASIC-Erweiterungen allein stellen schon ein erstklassiges Toolkit dar: APPEND (Anhängen von Programmen, AUTO (automatische Zeilennummerierung), BASIC-Befehle auf Tastendruck, PRINT POSITION, UNNEW, Strings größer als 88 Zeichen einlesen und vieles mehr. Die Bandbreite reicht von Spielen wie Goldgräber oder Starshooter bis zu nützlichen Programmen wie Cassetteninhaltsverzeichnis und -katalog mit automatischem Suchen nach Dateien und einem Terminkalender. Für den VC-20 Anwender ist dieser 324 Seiten-Wälzer eine wahre Fundgrube, in der es immer etwas neues zu entdecken gibt.




VC-20 TIPS & TRICKS, 3. erweiterte und überarbeitete Auflage, 1984, 324 Seiten, DM 49,–

## Interessant.

Einen guten Einstieg in PASCAL bietet dieses Trainingsbuch. Es gibt eine leichtverständliche Einführung, sowohl in UCSD-PASCAL wie auch in PASCAL64, wobei allerdings EDV-und BASIC-Grundkenntnisse vorausgesetzt werden. Der Autor, Ottmar Korbmacher, ist Student der Mathematik. Ihm gelingt es, in einem sprachlich aufgelockerten Stil mit vielen interessanten Beispielprogrammen, dem Leser Programmstrukturen, Ein/Ausgabe, Arithmetik und Funktionen, Prozeduren und Rekursionen, Sets, Files und Records näherzubringen. Die Übungsaufgaben am Ende jeden Kapitels helfen dabei, das Gelernte zu vertiefen. Ein Anhang mit allen PASCAL-Schlüsselworten, der ansich schon ein umfangreiches Lexikon darstellt, macht das Buch für jeden PASCAL-Anwender interessant.

DAS TRAININGSBUCH ZU PASCAL, 1984, ca. 250 Seiten, DM 39,–

## Bewährt.

Die bereits dritte Auflage von VC-20 INTERN ist wieder erheblich erweitert worden. Das Buch beschäftigt sich ausführlich mit der Technik und dem Betriebssystem des VC-20. Dazu gehört natürlich zuerst einmal ein ausführlich dokumentiertes ROM-Listing. Dazu gehört auch die Belegung der Zeropage, dem wichtigsten Speicherbereich für den 6502-Prozessor, eine übersichtliche Auflistung der Adressen aller Betriebssystemroutinen, ihrer Bedeutung und ihrer Übergabeparameter. Dies ermöglicht dem Programmierer endlich, den VC-20 von Maschinensprache aus sinnvoll einzusetzen. Denn warum Routinen, die bereits vorhanden sind, noch einmal schreiben? Weiterer Inhalt: Einführung in die Maschinensprache – Maschinensprachemonitor, Assembler, Disassembler – Verbindung von Maschinensprache- und BASIC-Programmen – Beschreibung der wichtigen IC's des VC-20 – Blockschaltbild – drei Original COMMODORE-Schaltpläne. Das Buch braucht jeder der sich intensiv mit der Maschinenspracheprogrammierung des VC-20 auseinandersetzen möchte.

VC-20 INTERN, 3. Auflage, 1984, ca. 230 Seiten, DM 49,–

## Starthilfe!

Das sollte Ihr erstes Buch zum COMMODORE 64 sein: 64 FÜR EINSTEIGER ist eine sehr leicht verständliche Einführung in Handhabung, Einsatz, Ausbaumöglichkeiten und Programmierung des COMMODORE 64, die keinerlei Vorkenntnisse voraussetzt. Sie reicht vom Anschluß des Geräts über die Erklärung der einzelnen Tasten und Funktionen sowie die Peripheriegeräte und ihre Bedienung bis zum ersten Befehl. Schritt für Schritt führt das Buch Sie in die Programmiersprache BASIC ein, wobei Sie nach und nach eine komplette Adressenverwaltung erstellen, die Sie anschließend nutzen können. Zahlreiche Abbildungen und Bildschirmfotos ergänzen den Text. Viele Anwendungsbeispiele geben nützliche Anregungen zum sinnvollen Einsatz des COMMODORE 64. Das Buch ist sowohl als Einführung als auch als Orientierung vor dem 64er Kauf gut geeignet.

64 FÜR EINSTEIGER, 1984, ca. 200 Seiten, DM 29,–

## Von A bis Z.

So etwas haben Sie gesucht: Umfassendes Nachschlagewerk zum COMMODORE 64 und seiner Programmierung. Allgemeines Computerlexikon mit Fachwissen von A–Z und Fachwörterbuch mit Übersetzungen wichtiger englischer Fachbegriffe – das DATA BECKER LEXIKON ZUM COMMODORE 64 stellt praktisch drei Bücher in einem dar. Es enthält eine unglaubliche Vielfalt an Informationen und dient so zugleich als kompetentes Nachschlagewerk und als unentbehrliches Arbeitsmittel. Viele Abbildungen und Beispiele ergänzen den Text. Ein Muß für jeden COMMODORE 64 Anwender!

DAS DATA BECKER LEXIKON ZUM COMMODORE 64, 1984, 354 Seiten, DM 49,–

## Fundgrube.

64 Tips & Tricks ist eine hochinteressante Sammlung von Anregungen zur fortgeschrittenen Programmierung des COMMODORE 64, POKE's und andere nützliche Routinen, interessanten Programmen sowie interessanten Programmiertips & -tricks. Aus dem Inhalt: 3D-Graphik in BASIC – Farbige Balkengraphik – Definition eines eigenen Zeichensatzes – Tastaturbelegung und ihre Änderung – Dateneingabe mit Komfort – Simulation der Maus mit einem Joystick – BASIC für Fortgeschrittene – C-64 spricht deutsch – CP/M auf dem COMMODORE 64 – Druckeranschluß über den USER-Port – Datenübertragung von und zu anderen Rechnern – Expansion-Port – Synthesizer in Stereo – Retten einer nicht ordnungsgemäß geschlossenen Datei – Erzeugen einer BASIC-Zeile in BASIC – Kassettenpuffer als Datenspeicher – Sortieren von Stringfelder – Multitasking auf dem COMMODORE 64 – POKE's und die Zeropage – GOTO, GOSUB und RESTORE mit berechneten Zeilennummern, INSTR und STRING-Funktion – Repeat-Funktion für alle Tasten – und vieles andere mehr. Alle Maschinenprogramme mit BASIC-Ladeprogrammen. 64 Tips & Tricks ist eine echte Fundgrube für jeden COMMODORE 64 Anwender. Schon über 65000mal verkauft!

64 TIPS & TRICKS, 1984, über 300 Seiten, DM 49,–

## Know-how!

350 Seiten dick ist die 4. erweiterte und überarbeitete Auflage von 64 INTERN geworden. Das bereits über 65000mal verkaufte Standardwerk bietet jetzt noch mehr Informationen. Hinzugekommen ist ein Kapitel über den IEC-Bus und viele, viele Ergänzungen, die sich im Laufe der Zeit angesammelt haben. Ebenfalls überarbeitet und noch ausführlicher ist jetzt die Dokumentation des ROM-Listings. Weitere Themen: genaue Beschreibung des Sound- und Video-Controllers mit vielen Hinweisen zur Programmierung von Sound und Grafik, der Ein/Ausgabesteuerung (CIAs), BASIC-Erweiterungen (RENEW, HARDCOPY, PRINTUSING), Hinweise zur Maschinenprogrammierung wie Nutzung der E/A-Routinen des Betriebssystems, Programmierung der Schnittstelle RS 232, ein Vergleich VC20 – C-64 – CBM zur Umsetzung von Programmen. Dies und viele weitere Informationen machen das umfangreiche Werk zu einem unentbehrlichen Arbeitsmittel für jeden, der sich ernsthaft mit Betriebssystem und Technik des C-64 auseinandersetzen will. Zum professionellen Gehalt des Buches tragen auch zwei Original-COMMODORE-Schaltpläne zum Ausklappen und zahlreiche ausführlich beschriebene und dokumentierte Fotos, Schaltbilder und Blockdiagramme bei.

64 INTERN, 4. überarbeitete und erweiterte Auflage, 1984, ca. 350 Seiten, DM 69,–

## Erfolgreich.

64 für Profis zeigt, wie man erfolgreich Anwendungsprobleme in BASIC löst und verrät die Erfolgsgeheimnisse der Programmierprofis. Vom Programmentwurf über Menüsteuerung, Maskenaufbau, Parametrisierung, Datenzugriff und Druckausgabe bis hin zur guten Dokumentation wird anschaulich mit vielen Beispielen dargestellt wie Profi-Programmierung vor sich geht. Besonders stolz sind wir auf die völlig neuartige Datenzugriffsmethode QUISAM, die in diesem Buch zum ersten Mal vorgestellt wird. QUISAM erlaubt eine beliebige Datensatzlänge, die dynamisch mit der Eingabe der Daten wächst. Eine lauffertige Literaturstellenverwaltung veranschaulicht die Arbeitsweise von QUISAM. Neben diesem Programm finden Sie noch weitere Programme zur Lager- und Adressenverwaltung, Textverarbeitung und einen Reportgenerator. Alle diese Programme sind mit Variablenliste versehen und ausführlich beschrieben. Damit sind diese für Ihre Erweiterungen offen und können von Ihnen an Ihre persönlichen Bedürfnisse angepaßt werden. Steigen Sie in die Welt der Programmierprofis ein.

64 FÜR PROFIS, 2. Auflage, 1984, ca. 300 Seiten, DM 49,–

## Rundum gut!

Endlich ein Buch, das Ihnen ausführlich und verständlich die Arbeit mit der Floppy VC-1541 erklärt. Das große Floppybuch ist für Anfänger, Fortgeschrittene und Profis gleichermaßen. interessant. Sein Inhalt reicht von der Programmspeicherung bis zum DOS-Zugriff, von der sequentiellen Datenspeicherung bis zum Direktzugriff, von der technischen Beschreibung bis zum ausführlich dokumentierten DOS-Listing, von den Systembefehlen bis zur detaillierten Beschreibung der Programme auf der Test-Demo-Diskette. Exakt beschriebene Beispiel- und Hilfsprogramme ergänzen dieses neue Superbuch. Aus dem Inhalt: Speichern von Programmen – Floppy-Systembefehle – Sequentielle Datenspeicherung – relative Datenspeicherung – Fehlermeldungen und ihre Ursachen – Direktzugriff – DOS-Listing der VC-1541 – BASIC-Erweiterungen und Programme – Overlaytechnik – Diskmonitor – IEC-Bus und serieller Bus – Vergleich mit den großen CBM-Floppies. Ein Muß für jeden Floppy-Anwender! Bereits über 45.000mal verkauft.

DAS GROSSE FLOPPY-BUCH, 2. überarbeitete Auflage, 1984, ca. 320 Seiten, DM 49,–

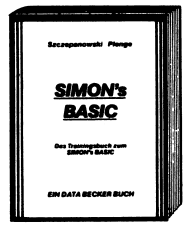
## Füttern erwünscht!

Diese beliebte umfangreiche Programmsammlung hat es in sich. Über 50 Spitzenprogramme für den COMMODORE 64 aus den unterschiedlichsten Bereichen, von attraktiven Superspielen (Senso, Pengo, Master Mind, Seeschlacht, Poisson Square, Memory) über Grafik- und Soundprogramme (Fourier 64, Akustograph, Funktionsplotter) und mathematische Programme (Kurvendiskussion, Dreieck) sowie Utilities (SORT, RENUMBER, DISK INIT, MENUE) bis hin zu kompletten Anwendungsprogrammen wie „Videothek", „File Manager" und einer komfortablen Haushaltsbuchführung, in der fast professionell gebucht wird. Der Hit zu jedem Programm sind aktuelle Programmiertips und Tricks der einzelnen Autoren zum Selbermachen. Also nicht nur abtippen, sondern auch dabei lernen und wichtige Anregungen für die eigene Programmierung sammeln.

DATA BECKER's GROSSE 64er PROGRAMMSAMMLUNG, 1984, 250 Seiten, DM 49,–

# Bestseller aus bester Hand

## BASIC-PLUS.



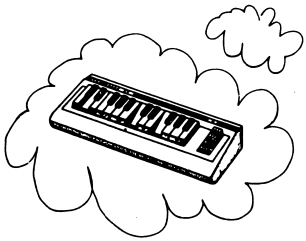

SIMON's BASIC ist ein Hit – wenn man es richtig nutzen kann. Auf über 300 Seiten erklärt Ihnen das DATA BECKER Trainingsbuch detailliert den Umgang mit den über 100 Befehlen des SIMON's BASIC. Alle Befehle werden ausführlich dargestellt, auch die, die nicht im Handbuch stehen! Natürlich zeigen wir auch die Macken des SIMON's BASIC und geben wichtige Hinweise wie man diese umgeht. Natürlich enthält das Buch viele Beispielprogramme und viele interessante Programmiertricks. Weiterer Inhalt: Einführung in das CBM-BASIC 2.0 – Programmierhilfen – Fehlerbehandlung – Programmschutz – Programmstruktur – Variablen – Zahlenbehandlung – Eingabekontrolle – Ein/Ausgabe Peripheriebefehle – Graphik – Zeichensatzerstellung – Sprites – Musik – SIMON's BASIC und die Verträglichkeit mit anderen Erweiterungen und Programmen. Dazu ein umfangreicher Anhang. Nach jedem Kapitel finden Sie Testaufgaben zum optimalen Selbststudium und zur Lernerfolgskontrolle.

DAS TRAININGSBUCH ZUM SIMON's BASIC, 2. überarbeitete Auflage, 1984, ca. 380 Seiten, DM 49,–

## Schrittmacher.

Eine leicht verständliche Einführung in die Maschinenspracheprogrammierung für alle, denen das C-64 BASIC nicht mehr ausreicht. Sie lernen Aufbau und Arbeitsweise des 6510-Mikroprozessors kennen und anwenden. Dabei werden die Analogien zu BASIC Ihnen beim Verständnis helfen. Ein weiteres Kapitel beschäftigt sich mit der Eingabe von Maschinenprogrammen. Dort erfahren Sie auch alles über Monitor-Programme sowie über Assembler. Zum einfachen und komfortablen Erstellen Ihrer eigenen Maschinensprache enthält das Buch einen kompletten ASSEMBLER, damit Sie gleich von Anfang an komfortabel und effektiv programmieren können. Weiterhin finden Sie dort einen DISASSEMBLER, mit dem Sie sich Ihre Maschinenprogramme oder die Routinen des BASIC-Interpreters und des BASIC-Betriebssystems ansehen können. Ein besonderer Clou ist ein in BASIC geschriebener Einzelschrittsimulator, mit dem Sie Ihre Programme schrittweise ausführen können. Dabei werden Sie nach jedem Schritt über Registerinhalte und Flags informiert und können den logischen Ablauf Ihres Programmes verfolgen. Eine unschätzbare Hilfe, besonders für den Anfänger. Als Beispielprogramm finden Sie ausführlich beschriebene Routinen zur Grafikprogrammierung und für BASIC-Erweiterungen. Natürlich sind alle Beispiele und Programme auf den C-64 zugeschnitten.

DAS MASCHINENSPRACHEBUCH ZUM COMMODORE 64, ca. 200 Seiten, DM 39,–



## SYNTHIMAT

SYNTHIMAT verwandelt Ihren COMMODORE 64 in einen professionellen, polyphonen, dreistimmigen Synthesizer, der in seinen unglaublich vielen Möglichkeiten großen Systemen kaum nachsteht.

**SYNTHIMAT in Stichworten:**
drei Oszillatoren (VCOs) mit 7 Fußlagen und 8 Wellenformen – drei Hüllkurvengeneratoren (ADSRs) – ein Filter (VCF) mit 8 Betriebsarten und Resonanzregulierung – VCF mit Eingang für externe Signalquelle – ein Verstärker (VCA) – Ringmodulation mit allen drei VCOs – 8 softwaremäßig realisierte Oszillatoren (LFOs) – kräftiger Klang durch polyphones Spielen – zwei Manuale (Solo und Begleitung) – speichern von bis zu 256 Klangregistern – schneller Registerwechsel – speichern von 9 Registerdateien auf Diskette – „Bandaufnahme" auf Diskette durch direktes Spielen – keine lästige Noteneingabe – speichern von bis zu 9 „Bandaufnahmen" je Diskette – integrierte 24 Stunden-Echtzeituhr – einstellbares PITCH-BENDING – farblich gekennzeichnete, übersichtlich angeordnete Module – umfangreiches Handbuch – läuft mit einem Diskettenlaufwerk – Diskettenprogramm.

DM 99,–

## STRUKTO 64

STRUKTO 64 ist eine fantastische neue Programmiersprache für strukturiertes Programmieren mit dem C-64 und für alle Programmierer geeignet, die den C-64 als Allround-Computer einsetzen und auf einfache Weise anspruchsvolle Programme erstellen wollen.

**STRUKTO 64 in Stichworten:**
Interpretersprache, die die Vorzüge von BASIC und PASCAL vereint – strukturiertes Programmieren – übersichtliche Programme – leichte Erlernbarkeit – einfache Bedienung – eingebautes Toolkit erleichtert das Eingeben und Verbessern von Programmen – leichteres Arbeiten mit der Floppy – Sprite-Editor ermöglicht das Einlesen der Sprite-Formen direkt vom Bildschirm – Graphikbedienung wird mit gut durchdachten Befehlen unterstützt – Abspielen von Musik ist unabhängig vom Programmablauf möglich – ca. 80 neue Befehle – lieferbar als Diskettenprogramm – ausführliches deutsches Handbuch.

DM 99,–

NEU Superbase 64

Für viele ein Traum, für die meisten bisher zu teuer: die Rede ist von einer echten Datenbank für den 64er. SUPERBASE 64 füllt eine Lücke. Nicht allein die Kapazität, die verwaltet werden kann, bewegt sich in professionellen Regionen, die ausgeprägten Fähigkeiten des SUPERBASE 64 im Rechnen und Kalkulieren lassen dieses Paket beinahe als Rund-Um-Software erscheinen.

**SUPERBASE 64 in Stichworten:**
maximale Datensatzlänge 1108 Zeichen, verteilt auf bis zu 4 Bildschirmseiten – bis zu 127 Felder pro Datensatz, wobei Textfelder bis zu 255 Zeichen lang sein können – insgesamt 15 Einzeldateien können zu einer SUPERBASE-Datenbank verknüpft werden – Speicherkapazität nur durch Diskette begrenzt – umfangreiche Auswertungsmöglichkeiten und komfortabler Report-Generator – Kalkulationsmöglichkeiten und Rechnen – Import- (Einlesen von externen Daten) und Export- (Ausgabe von SUPERBASE Dateien als sequentielle Datei) Funktionen ermöglichen Datenaustausch mit anderen Programmen – durch leistungsfähige, eigene Datenbanksprache auch als kompletter Anwendungsgenerator verwendbar.

DM 398,–

## MASTER 64

MASTER 64 ist ein professionelles Programmentwicklungssystem für den C-64, das es Ihnen ermöglicht, die Programmentwicklungszeit auf einen Bruchteil der sonst üblichen Zeit zu reduzieren. MASTER 64 bietet einen Programmkomfort, den Sie nutzen sollten.

**MASTER 64 in Stichworten:**
70 zusätzliche Befehle – Bildschirmmaskengenerator – definieren von Bildschirmzonen – Eingabe aus Zonen – formatierte Ausgabe – Abspeicherung von Bildschirminhalten – Arbeiten mit mehreren Bildschirmmasken – ISAM Dateiverwaltung, in der Datensätze über einen Zugriffschlüssel angesprochen werden können – Datensätze bis zu 254 Zeichen – Schlüssellänge bis zu 30 Zeichen – Dateigröße nur von Diskettenkapazität abhängig – Zugriff über Schlüssel und Auswahlmasken – Bildschirm- und Druckmaskengenerator – Erstellung beliebiger Formulare und Ausgabemasken – BASIC-Erweiterungen – Toolkitfunktionen – Mehrfachgenaue Arithmetik (Rechnen mit 22 Stellen Genauigkeit).

DM 198,–

## TEXTOMAT

Das Bearbeiten von Texten gehört zum wichtigsten Betätigungsfeld von Homecomputer-Anwendern. So ist es auch nicht verwunderlich, daß eine Unzahl verschiedener Textprogramme für den 64er angeboten wird. TEXTOMAT zeichnet sich dadurch aus, daß er auch vom Einsteiger sofort benutzt werden kann. Über eine Menuezeile können alle Funktionen angewählt werden. Selbstverständlich beherrscht TEXTOMAT deutsche Umlaute und Sonderzeichen.

### TEXTOMAT in Stichworten:

Diskettenprogramm – durchgehend menuegesteuert – deutscher Zeichensatz auch auf COMMODORE-Druckern Rechenfunktionen für alle Grundrechenarten – 24.000 Zeichen pro Text im Speicher – beliebig lange Texte durch Verknüpfung – horizontales Scrolling für 80 Zeichen pro Zeile – läuft mit 1 oder 2 Floppies – frei programmierbare Steuerzeichen – Formularsteuerung für Randeinstellung u.s.w. – komplette Bausteinverarbeitung – Blockoperationen, Suchen und Ersetzen – Serienbriefschreibung mit DATAMAT – formatierte Ausgabe auf Bildschirm – an fast jeden Drucker anpaßbar – ausführliches deutsches Handbuch mit Übungslektionen.

DM 99,–

## PAINT PIC

Malen (!) mit dem Computer, welch eine faszinierende Idee. Mit dem Malprogramm PAINT PIC für den COMMODORE 64 wird diese Idee Realität. Mit PAINT PIC ist es auch für den Einsteiger leicht, fantastische Computerbilder zu erstellen. Man kann die Bilder auf Diskette abspeichern und wieder laden und selbstverständlich steht auch weiterhin der COMMODORE-Zeichensatz zur Verfügung. Wichtig: PAINT PIC benötigt keine zusätzliche Hardware.

### PAINT PIC in Stichworten:

Programmsteuerung: Tastatur – Steuerung des Stifts: Cursortasten und eckige Klammer (diag.) (Joystick kann benutzt werden) – Routinen: Linien, Rechtecke, Dreiecke, Parallelogramme, Kreise, Kreisbögen, Ellipsen, Bestimmung von Mittelpunkt, und perspektivischer Linie, Kopieren und Drehen von Teilbildern, Verdoppeln, halbieren und spiegeln von Teilbildern – Modi: Malstiftmodus (schmale Linie) Pinselmodus (8 verschiedene Breiten) (Art der Linie selbst definierbar) – Textmodus (kompl. Zeichensatz COMMODORE) (Hoch-Tiefschrift) – Speichern: Teilbilder (Blöcke) oder ganze Bilder – Menue: 1 Hauptmenue mit 8 Untermenues – mit ausführlichem deutschen Handbuch – Diskettenprogramm – Bilder kann man auf Diskette abspeichern.

DM 99,–

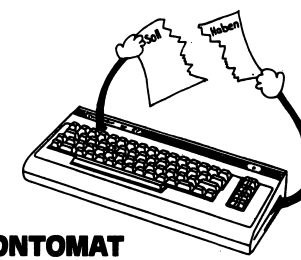
## PROFIMAT

Wer sich tiefer in die Innereien des Computers begeben will, kommt ohne besonderes Werkzeug nicht aus. Einerseits muß der volle Einblick in alle Speicherbereiche möglich sein, andererseits soll der Umgang mit Maschinenprogrammen so komfortabel wie möglich gestaltet sein. PROFIMAT hat Lösungen für beide Probleme: Der Maschinensprache-Monitor PROFI-MON bietet alle Hilfsmittel zum Umgang mit Maschinenprogrammen; PROFI-ASS ist ein Macro-Assembler, der das Schreiben von Maschinenprogrammen fast so einfach macht wie das Programmieren in BASIC.

### PROFIMAT in Stichworten:

Registerinhalte und Flags anzeigen – Speicherinhalte anzeigen – Maschinenprogramme laden, ausführen und speichern – Speicherbereiche durchsuchen, vergleichen, füllen und verschieben – echter Einzelschrittmodus – Setzen von Unterbrechungspunkten – schneller Trace-Modus – Rückkehr zu BASIC – formatfreie Eingabe – Verkettung beliebig vieler Quellprogramme – erzeugter Objektcode kann in Speicher oder auf Diskette gehen – formatiertes Assemblerlisting – ladbare Symboltabellen – redefinierbare Symbole – Operatoren – Unterstützung der Fließkommaarithmetik – bedingte Assemblierung – Assemblerschleifen – MACROS mit beliebigen Parametern.

DM 99,–

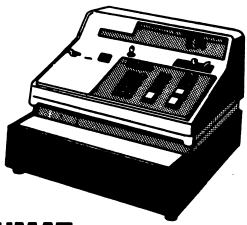



## KONTOMAT

KONTOMAT ist ein menuegesteuertes Einnahme-Überschußprogramm nach §4(3) EStG mit Kassenbuch, Bankkontenüberwachung, automatischer Steuerbuchung, AFA Tabellenerstellung, Kontenblättern, Ermittlung der UST.-Voranmeldungswerte und Monats- und Jahresabrechnung. Der neue KONTOMAT ist voll parameterisiert und läßt sich damit an Ihre Bedürfnisse anpassen. Für alle Gewerbetreibenden, die nicht laut HGB zur Buchführung verpflichtet sind. KONTOMAT ist für den gewerblichen Einsatz, aber auch als Lernprogramm oder zur Haushaltsbuchführung geeignet.

**KONTOMAT in Stichworten:**
Diskettenprogramm – maximal 120 Konten – Beträge mit bis zu 6 Vor- und 2 Nachkommastellen – 4 Mehrwert- und Vorsteuersätze – intervallmäßige Belegeingabe – 4 Buchungsarten (SOLL, HABEN, SOLL/HABEN und HABEN/SOLL) – Anzeige der Soll- und Habensumme bei mehrfachen Buchungssätzen – komfortable Belegeingabe mit Datum, Buchungstext, Steuerkennzeichen und Betrag – Druck des Journals während der Belegeingabe – Druck von umfangreichen Kontenblättern – Druck einer Summen- und Saldenliste mit Monats- und Jahresumsatzsummen – betriebswirtschaftliche Auswertung mit Druckausgabe – Ermittlung und Druckausgabe der Umsatzsteuerzahllast – Speicherung der Anlagegüter und automatische Abschreibung am Jahresende – übersichtliche AfA-Liste – arbeitet mit 1 oder 2 Laufwerken – umfangreiches deutsches Handbuch.

DM 148,–

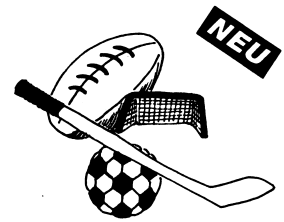


## FAKTUMAT

Mit FAKTUMAT ist das Schreiben von Rechnungen kein Alptraum mehr. Eine Sofortfakturierung mit integrierter Lagerbuchführung. Individuelle Anpassung von Steuersätzen, Maßeinheiten und Firmendaten. Kunden- und Artikelstamm voll pflegbar. Schneller Zugriff auf Kunden- und Artikeldaten, über freidefinierbaren, 6-stelligen Schlüssel. Automatische Fortschreibung von Artikel- und Kundendaten, individuell nutzbar. Alles in allem die Arbeits- und Zeitersparnis, die Sie sich schon immer gewünscht haben.

**FAKTUMAT in Stichworten:**
voll menuegesteuert – läuft mit einer oder zwei Floppies – Diskettenwechsel (eine Floppy) nur beim Wechsel vom Hauptmenue ins Unterprogramm und umgekehrt – mit Ausnahme des Ausschaltens der Floppy während der Verarbeitung werden alle Fehler abgefangen (z. B. Drucker nicht eingeschaltet – arbeitet mit 1525, 1526 (?), MPS 801, EPSON Drucker und DATA BECKER Interface – voll parameterisiert: Firmenkopf, MWSt. und Rabattsätze, Größe der Dateien beliebig wählbar – 5 Zeilen für Firmenkopf je 30 Zeichen (erste Zeile erscheint auf der Rechnung in Breitschrift – 4 Mehrwertsteuer-Sätze; während der Rechnungsschreibung können also Artikel mit unterschiedlichem Mehrwertsteuer-Satz verrechnet werden – 10 Rabattsätze (Rabattsatz 1 vorbelegt mit 0%), bei der Rechnungsschreibung kann jedem Artikel ein Rabattsatz zugewiesen werden – maximal 1900 Artikel bei 50 Kunden oder 950 Kunden bei 100 Artikel (max. Artikel = [1000-Kunden]*2; max. Kunden = [2000-Artikel]/2) – manuelle Eingabe von Artikeln und/oder Kunde während der Rechnungsschreibung – d. h. es können mehr Artikel verrechnet weden als überhaupt in die Datei passen (bei Verzicht auf Lagerbuchführung) bzw. es können Rechnungen an Kunden geschrieben werden, die nicht erfaßt wurden – integrierte Lagerbuchführung mit Ausgabe einer Inventurliste – Rechnungsbeträge und Datum werden in der Kundendatei festgehalten – Druck von: Rechnung (mit Abbuchen aus Lager), Rechnung (ohne Abbuchen aus Lager), Lieferschein – deutsches detailliertes Handbuch mit Übungs- und Anwendungsteil – deutsche Bedienerführung innerhalb des Programms (z. B. „Artikel nicht vorhanden" anstelle „RECORD NOT PRESENT").

DM 148,–

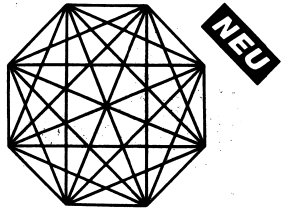



## UNI-TAB

Heute schon die Bundesliga-Tabelle von morgen kennen, das geht mit UNI-TAB. Alle Rechnereien, die man ohne dieses Programm nie machen würde, lassen sich in Sekundenschnelle durchführen. Wer will, kann mit simulierten Spielergebnissen den Weltmeister '86 vorausberechnen. Aber nicht nur Fußball-Ligen können tabellarisch erfaßt werden, fast alle Sportarten sind UNI-TAB-fähig. Gag am Rande: für viele Sportarten stehen die bekannten Piktogramme zur Verfügung.

**UNI-TAB in Stichworten:**
Menuesteuerung über die Funktionstasten mit leicht verständlichen Auswahlmöglichkeiten – Bedienerfreundlich (Mannschaften werden über Kennzahlen gesteuert) – Ligen mit 4 bis 20 Mannschaften können verwaltet werden (6 bis 38 Spieltage möglich) – unsinnige Ligen (z. B. 13 Mannschaften sollen 5 Spieltage absolvieren) sind ausgeschlossen – favorisierte Mannschaft kann während des Programmablaufs durch reverse Darstellung gekennzeichnet werden – Tabelle kann geändert werden (wichtig bei Spielanullierungen) – drei verschiedene Tabellenarten können abgespeichert und später eingelesen werden (die aktuelle Tabelle [unabhängig von der Vollständigkeit eines Spieltages], der komplette Spieltag [Vollständigkeit und Nummer des Spieltages werden automatisch errechnet], die simulierte Tabelle [der Anwender kann so selbst Schicksal spielen und seinen Tip später mit dem tatsächlichen Geschehen vergleichen]) – zwei verschiedene Arten der Saisonübersicht (die statistische Übersicht zeigt an, welchen Tabellenplatz das jeweilige Team bei welchem Punkte- und Torverhältnis an den einzelnen Spieltagen einnahm; die graphische Übersicht zeigt die Leistungskurve jeder Mannschaft) – alle Tabellen und Graphiken sind als Hardcopy auf einem Drucker darstellbar – bei Fehlbedienung (z. B. gewünschte Druckausgabe bei nicht eingeschaltetem Drucker) erscheinen leicht verständliche deutsche Fehlermeldungen.

DM 69,–




## SUPERGRAFIK 64

Entdecken Sie die faszinierende Welt der Computergraphik mit SUPERGRAFIK 64, der starken Befehlserweiterung mit den vielseitigen Möglichkeiten. Durch die neue verbesserte Version jetzt noch leistungsstärker.

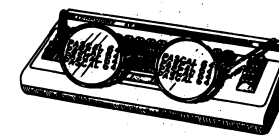
**SUPERGRAFIK 64 in Stichworten:**

2 unabhängige Graphikseiten (320 x 200 Punkte) – logische Verknüpfung der beiden Graphikseiten (AND, OR, EXOR) – 1 Standard Low-Graphik Seite (80 x 50 Punkte) – Normalfarben Graphik (300 x 200 Punkte) – Multicolor-Graphik (160 x 200 Punkte) – verdecktes Zeichnen (z. B. Text sichtbar, Graphikseite 2 wird erstellt) – Textfenster in der Graphik – 183 Befehle und Befehlskombinationen (1. Für jeden Befehl wählbare Zwischenmodi: Zeichnen, Löschen, Punktieren, Graphik-Cursor bewegen, Zeichnen mit/ohne Farbsetzung, Punkte zählen; 2. Durch einfache Befehle zu steuernde Graphikfiguren: Punkt, Linie, Linienschar, Linie vom Graphik-Cursor, Kreise, Kreisbögen, Ellipse, Ellipsenbögen, selbstdefinierbare Figuren, rotieren und vergrößern dieser Figuren, Rahmen, Feld, Text in Graphik; 3. Weitere Graphikbefehle: Graphikseiten- und Moduswechsel, Graphik löschen, Graphik invertieren, Scrolling von Text und Graphik, Wählen der Rahmen-, Hintergrund-, Zeichen- oder Punktfarbe) – Speichern, Laden von Graphik (auch verdeckt) – Kopieren des Textbildschirms in die Graphikseite – Hardcopies für EPSON, Seikosha GP100VC, Farb(!)drucker Seikosha GP700 und andere mit DATA BECKER Interface – 16! Sprites gleichzeitig auf dem Bildschirm – alle Sprite-Eigenschaften veränderbar – Positionieren und Bewegen (!) von 16 Sprites gleichzeitig und unabhängig voneinander, während das übrige Programm weiterläuft (IRQ) – Sprite-Kollisionsüberprüfung, Joystickunterstützung – automatische Unterbrechung des BASIC-Programms bei Kollisionen (Interrupt), Sprung in Unterbrechungsroutine, dann Weiterführung des Hauptprogramms – komfortable Soundprogrammierung mit Verstellung aller möglichen Sound-Parameter (Lautstärke, Klang, Filter, Tonhöhe, Tonlänge), ebenfalls unabhängig vom übrigen Programmlauf – zahlreichen Programmiertools (MERGE, RENUMBER usw.) – umfangreiche Anleitung – Diskettenprogramm.

DM 99,–

## PASCAL 64

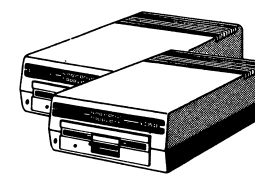
Beim Wort „Compiler" fällt dem Eingeweihten sicher der Begriff „Geschwindigkeit" ein. Ein PASCAL-Compiler sollte jedoch weitere Assoziationen wecken. Strukturiertes Programmieren heißt das Zauberwort. PASCAL wurde eigens zu didaktischen Zwecken entwickelt und erfüllt diese Aufgabe auch heute noch. Der PASCAL 64 Compiler bringt diese phantastische Programmiersprache auf den 64er.
Gerade die neue, verbesserte Version unterstützt die Möglichkeiten des C-64 in jeder Hinsicht und macht leistungsfähige Programme möglich.



**PASCAL 64 in Stichworten:**

besitzt einen sehr umfangreichen Befehlsvorrat – erlaubt Interruptprogrammierung und bietet Schnittstellen zu Monitor und Assembler – erzeugt sehr schnelle Programme in reinem Maschinencode – unterstützt relative Dateiverwaltung, Graphik und Sound – bietet die Datentypen REAL, INTEGER, CHAR und BOOLEAN sowie Aufzähltypen und POINTER, die zu Datenstrukturen RECORD, SET, ARRAY und PACKED ARRAY kombiniert werden können – erlaubt vorzeitigen Abschluß von Prozeduren mit EXIT, uneingeschränkte Rekursionen und komfortable Verarbeitung von Teilfeldern (Strings) – ist ein ausgereiftes, deutsches Produkt und wird mit ausführlichem Handbuch geliefert.

DM 99,–



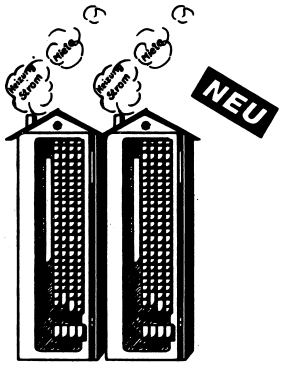
## DISKOMAT

Der Umgang mit Diskettenlaufwerken ist für viele noch immer mit Geheimnissen belastet. Andere stören sich an den wenig komfortablen Diskettenbefehlen des BASIC V2. DISKOMAT bringt Abhilfe; alle Diskettenbefehle des BASIC 4.0 stehen zur Verfügung. Außerdem können mit dem Programm SUPERTWIN zwei 1541-Laufwerke wie ein Doppellaufwerk verwaltet werden. Für Benutzer, die sich die Fähigkeiten der Floppy 1541 ganz erschließen wollen, steht der DISK-MONITOR bereit; er macht es endlich möglich, den direkten Zugriff auf einzelne Blocks einfach und bequem vorzunehmen.

**DISKOMAT in Stichworten:**

Diskettenprogramm – DISK BASIC unterstützt Diskettenbefehle des BASIC 4.0 (CONCAT, HEADER, APPEND, RENAME, OPEN, COLLECT, DSAVE, SCRATCH, DCLOSE, BACKUP, DLOAD, DIRECTORY, RECORD, COPY, CATALOG, DS & DS$) – SUPER TWIN behandelt 2 Laufwerke 1541 wie ein Doppellaufwerk – DISK-MONITOR ermöglicht direkte Analyse und Manipulation von Disketten (direktes Lesen und Schreiben einzelner Blöcke, ändern von Blöcken mittels Bildschirm-Editor, Anzeige des Diskettenstatus, direktes Absenden von Disketten-Befehlen) – ausführliches deutsches Handbuch beschreibt jeden einzelnen der 3 Programmteile.

DM 99,–




## HAUSVERWALTUNG

Jetzt können alle Hausbesitzer aufatmen: das Programm HAUSVERWALTUNG bietet Ihnen eine sehr komfortable Verwaltung der Mietwohnungen mit dem COMMODORE 64.
Alles, was Sie dazu brauchen, ist ein COMMODORE 64, ein Diskettenlaufwerk 1541, ein anschlußfähiger Drucker und das obengenannte Programm HAUSVERWALTUNG. Die nachfolgenden und viele weitere leistungsfähige Features ermöglichen eine äußerst rationelle Verwaltung Ihrer Mietwohnungen.

### HAUSVERWALTUNG in Stichworten:

Diketten programm – Verwaltung von 50 Einheiten pro Objekt möglich – Stammdatenverwaltung für Häuser und Mieter – Verbuchen der Miete, Nebenkosten und Garagenmieten – Mietkontoanzeige – Haus- und Mieteraufstellung – Mahnungen – Verbuchen der anfallenden Kosten – Kostengegenüberstellung – Jahresendabrechnung mit automatischem Jahresübertrag – umfangreiches deutsches Handbuch.

DM 198,–

## TRAININGSKURS zu ADA

Diese Programmiersprache der Zukunft, die das Pentagon in Auftrag gegeben hat, wird jetzt durch DATA BECKER auch dem C-64 Anwender zugänglich gemacht durch den TRAININGSKURS zu ADA, der eine sehr gute Einführung in diese Supersprache bietet. Der dazu gelieferte Compiler liefert ein umfangreiches Subset der Sprache.

### ADA in Stichworten:

blockstrukturierte Programme – modularer Aufbau der Programme – ermöglicht die Behandlung von Ausnahmezuständen – Fehlerüberprüfung beim Übersetzen und zur Laufzeit – ermöglicht das einfache Einbinden von Maschinenprogrammen – sehr leichtes Arbeiten mit Programmbibliotheken – Programmdiskette enthält Editor, Übersetzer, Assembler und Disassembler – umfangreiches deutsches Handbuch.

DM 198,–

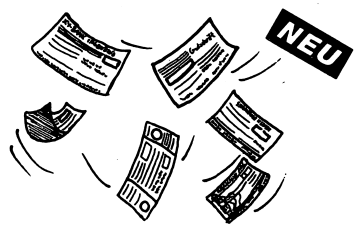
## DATAMAT

Daten verwalten kann ein schier endloses Hantieren mit Karteikästen und Aktenordnern bedeuten; kann aber auch C-64 plus DATAMAT heißen. Dann wird Suchen und Sortieren zum Spaß. Der DATAMAT bietet in seiner neuen Version einiges, was in dieser Preisklasse bisher unvorstellbar schien. Nicht nur Geschwindigkeit und Bedienungsfreundlichkeit wurden weiter verbessert, auch die Anpassung an die meisten Drucker ist inzwischen machbar.

### DATAMAT in Stichworten:

menuegesteuertes Diskettenprogramm, dadurch extrem einfach zu bedienen – für jede Art von Daten – völlig frei gestaltbare Eingabemaske – 50 Felder pro Datensatz – 253 Zeichen pro Datensatz – bis zu 2000 Datensätze pro Datei je nach Umfang – Schnittstelle zu TEXTOMAT – läuft mit 1 oder 2 Floppies – völlig in Maschinensprache – extrem schnell – deutscher Zeichensatz auch auf COMMODORE-Druckern – fast jeder Drucker anschließbar – ausdrucken über RS 232 – duplizieren der Datendiskette – verbesserte Benutzerführung – Hauptprogramm komplett im Speicher (kein Diskettenwechsel mehr) – integrierte Minitextverarbeitung – deutsches Handbuch mit Übungslektionen
Sie können:
jeden Datensatz in 2 – 3 Sekunden suchen – nach beliebigen Feldern selektieren – nach allen Feldern gleichzeitig sortieren – Listen in völlig freiem Format drucken – Etiketten drucken.

DM 99,–




## ZAHLUNGSVERKEHR

Umfangreicher Zahlungsverkehr kann zur Plage werden. Das Software-Paket ZAHLUNGSVERKEHR übernimmt den größten Teil dieser Arbeit. Außer den notwendigen Fähigkeiten für das Ausfüllen und Auflisten von Überweisungen und Schecks ist der ZAHLUNGSVERKEHR in der Lage, Sammellisten, Einzugslisten etc. selbständig zusammenzustellen.

### ZAHLUNGSVERKEHR in Stichworten:

Diskettenprogramm – max. 100 Zahlungsempfänger pro Diskette – drei definierbare Absenderbanken – 25 Zahlungsdateien – 14 frei definierbare Formulare – Kontrolldruck bei Belegeingabe möglich – Eingabe von Rechnungsdaten oder eines Verwendungszwecks – Ausdruck einer Sammel-Überweisungsliste – Korrekturmöglichkeit der einzelnen Zahlungsdateien – arbeitet mit einer oder zwei Floppies – umfangreiches deutsches Handbuch.

DM 148,–

# ACHTUNG AUTOREN

**SIE** haben ein gutes Programm oder ein Manuskript zu einem interessanten Buch geschrieben oder würden dies gerne tun

**SIE** würden das Ergebnis Ihrer Arbeit gerne in größeren Stückzahlen vermarktet sehen

**SIE** suchen dafür den leistungsfähigen Verlag und Vertriebspartner Ihres Vertrauens

**WIR** besitzen große Erfahrung in der professionellen Vermarktung von Software und Literatur, nicht nur in Deutschland, sondern auch weltweit

**WIR** suchen weitere Autoren, mit denen wir gemeinsame Erfolge erringen können

**WIR** haben in den letzten 12 Monaten mit dem Verkauf von 200.000 Büchern und 50.000 Programmen unsere Leistungsfähigkeit unter Beweis gestellt

**SIE** und **WIR** sollten zusammenarbeiten.

Bitte, senden Sie eine Beschreibung Ihres Programms oder Ihres Buchprojekts an Dr. Achim Becker c/o DATA BECKER, oder fordern Sie einfach unsere unverbindlichen „Informationen für Autoren" an.

***IHR GROSSER PARTNER FÜR KLEINE COMPUTER***

# DATA BECKER

Merowingerstraße 30 · 4000 Düsseldorf · Telefon (02 11) 31 00 10 · im Hause AUTO BECKER

## DAS STEHT DRIN:

Das GROSSE DRUCKERBUCH für Druckeranwender mit einem Commodore 64 oder VC-20 enthält eine riesige Sammlung von Tips & Tricks, Programmlistings und Hardwareinformationen. Ob Sie nun einen Typenrad- oder Matrixdrucker oder den Plotter VC-1520 besitzen, in jedem Fall ist das GROSSE DRUCKERBUCH eine unschätzbare Informationsquelle.

Aus dem Inhalt:

- Schnittstellen
- Sekundäradressen
- Schreibmaschine am Userport
- Formatierung
- Textverarbeitung
- Hardcopy
- High-Res-Graphik
- Druckprinzipien
- Betriebssystem des MPS-801
- Programmierung des Plotters VC 1520

## UND GESCHRIEBEN HABEN DIESES BUCH:

Klaus Gerits ist bekannter Fachautor (64 INTERN, 64 FÜR PROFIS) und Leiter der DATA BECKER Entwicklungsabteilung.
Rolf Brückmann ist engagierter Techniker in der DATA BECKER Entwicklungsabteilung und zuständig vor allem für Schnittstellen und Druckeranpassung. Er ist ebenfalls Fachautor (64 INTERN, 64 UND DER REST DER WELT).
Thomas Wiens ist erfahrener Commodore-Anwender. Aus dem Fehlen einer ausführlichen Anleitung zum Umgang mit dem Plotter VC-1520 machte er eine Tugend und schrieb mit über 100 Seiten ein leichtverständliches Anwenderhandbuch.

**ISBN 3-89011-020-7**