Schönschrift und Textverarbeitung für SHARP-Computer

Fischel GmbH W.Meyer

ISBN: 3-924327-37-8

Herausgeber
Fischel GmbH
Kaiser-Friedrich-Str. 54a
1000 Berlin 12

Berlin 1987

I N H A L T S V E R Z E I C H N I S

Dipl.-Volksw. Winfried Meyer

SCHÖNSCHRIFT UND TEXT-VERARBEITUNG FÜR SHARP-COMPUTER

EINLEITUNG

Dieses Buch behandelt zwei verwandte Themen. Zum einen wird gezeigt, daß auch auf Pocket-Computern eine ernsthafte (Ihnen hoffentlich viel Spaß bereitende) Textverarbeitung möglich ist. Der Verfasser selbst hat die Texte des Buches auf einem PC 1600-System erfaßt und bearbeitet. Ein ähnliches Textverarbeitungssystem für den PC 1500(A), das auch auf dem PC 1600 laufen kann, wird hier in allen Einzelheiten besprochen, so daß Anpassungen an weitere Computer leicht möglich sind. Eine Hilfe dabei wird das gut gegliederte Programmlisting sein, das viel von Sprungmarken (Labels) gebraucht macht.

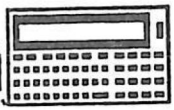
In Kapitel XI finden Sie auch ein komplettes Programm für den PC 1350/2500. Beide Programme zeichnen sich durch ein anwenderfreundliches Menüsystem aus und wirklich umfangreichen Bearbeitungs- (Editier-) funktionen, wie sie für Pocket-Computer bisher wohl kaum dargeboten wurden. Einige zusätzliche Maschinensprache-Programme sind das Salz in der Suppe.

Zum anderen widmen wir uns hier der Erzeugung, der Darstellung und dem Gebrauch der im Text verwendeten Zeichen, sozusagen den 'Atomen'

des Textes. Für diesen Zweck wird ein komfortabler 'Zeichengenerator' zur Verfügung gestellt, der alles automatisch macht. Trotzdem kommt die theoretische Erörterung nicht zu kurz. Viele Zeichensätze (u.a. IBM-Zeichensatz) bekommen Sie schon fertig geliefert.

Diese Zeichen nun können von allen möglichen Plottern (zeichnenden Druckern), gesteuert von unterschiedlichen Computern, verwendet und schließlich zu ausdrucksstarken Schriftgrafiken geformt werden. Dem liegt eine universelle Methode zugrunde, die im Schlußkapitel erläutert wird. Auch die Rechner MZ 700/800 können sich diese, mit einem geeigneten Plotter, zunutze machen, wie gezeigt werden wird. Verzichten mußten wir aus Platzgründen auf Übungsaufgaben und Spiele, aber der Leser wird trotzdem viel Spaß haben!

durch Information vorn



Kap. I: DER UMGANG MIT ZEICHENKETTEN

1. Variable, Listen und Felder

Die **Textverarbeitung**, wie wir sie in diesem Buch abhandeln, soll für möglichst viele Computer verwendbar sein. Nun wissen Sie, daß jede Maschine so ihre Eigenheiten hat. Das gilt nicht nur für die Maschine selbst (ihrer Hardware), sondern meist auch für ihre Arbeitsmittel (Software), d.h. das **Betriebssystem** und die verwendete **Programmiersprache**. Insbesondere findet man sehr unterschiedliche **BASIC-Dialekte** vor.

Ebenso ist es bei **SHARP-Computern**, obwohl ein gewisses Bemühen festgestellt werden kann, die verschiedenen Modelle im Hinblick auf BASIC miteinander zu vereinbaren (kompatibel zu machen). So sind beispielsweise der **PC 1500 mit dem 1500A** und der **PC 1350 mit dem PC 2500** nahezu **BASIC-kompatibel**. Andererseits gibt es etwa im Vergleich vom PC 1500 mit dem PC 1350 einige wichtige Unterschiede, die man im Auge behalten muß, wenn man Programme von dem einen Computer übernimmt, um sie auf den anderen zu übertragen. Darauf wird an verschiedener Stelle hingewiesen.

Bei allen Unterschieden ist dem **SHARP-BASIC** doch gemein die Verwendung von zwei Variablentypen, den **numerischen** und den **nicht-numerischen** Variablen. Letztere werden auch **Zeichenvariable** (String) genannt. Die Zeichenvariable sind grundlegend für die **Textverarbeitung in BASIC**, denn sie nehmen, wie ihr Name schon sagt, Zeichen bzw. Text auf und lassen eine Reihe von Manipulationen zu. Nehmen wir als Bezugswort: 'SHARP-COMPUTER'.

Diese **Information** stecken wir jetzt in eine

für den Rechner geeignete Schublade, eben der Zeichenvariablen, und geben dieser einen Namen. Bei den von uns betrachteten SHARP-Computern besteht der Name aus 1 oder 2 (signifikanten) Zeichen mit angehängtem $-Symbol. Wohl darf der Name bei vielen Rechnertypen auch länger sein, aber das SHARP-BASIC (d.h. der BASIC-Interpreter) wertet **nur 2 Zeichen** aus, von denen das 1. übrigens immer ein Buchstabe sein muß, während bei den weiteren auch Ziffern und Sonderzeichen zugelassen sind. So kommen wir zu der folgenden, in Form einer Gleichung geschriebenen, **Zuweisung** :

```
S$="SHARP-Computer"
```

Die Information in einer Zeichenvariablen, die **Zeichenkette** oder der String (engl.), muß zur deutlichen Unterscheidung von numerischen Variablen immer in Hochkomma ("") gesetzt werden. S1$="4711" wird vom Rechner sowohl anders intern abgelegt, als auch in anderer Weise verarbeitet als S1=4711. Rechnen kann der Comter nur mit der numerischen Variablen S1.

Anders als beim MZ 700/800 gibt es bei SHARP-Pocketcomputern einen besonderen Variablentyp, die **Standard-Variablen.** Diese können sowohl numerisch als auch nicht-numerisch, also Zeichenvariable sein. Zur Bezeichnung ist nur 1 Buchstabe zugelassen, wobei bei Zeichenvariablen noch das $-Symbol angehängt wird. Unser Beispiel S$ ist in diesem Sinne eine **Standard-Zeichenvariable.** Standardvariable haben bei den SHARP-Pocketcomputern (SHARP-PC) einen festen Platz (d.h. eine **nicht-verschiebbare Adresse** im Benutzer-RAM), was einige Vorteile bietet. Hier genügt die Feststellung, daß die Länge des Strings bei den Rechnern PC 1500(A)/ PC 1600 in Standard-Zeichenvariablen auf **16 Zeichen** begrenzt ist. Bei Rechnern wie dem PC 1350 oder der 1400er Serie ist die String-

länge aber auf **7 Zeichen** begrenzt. Versucht man, Zeichenvariablen mehr Zeichen als erlaubt zuzuweisen, werden die überzähligen einfach abgeschnitten, und das gilt auch für Standard-Zeichenvariablen. Als Zeichen gilt auch immer das **Leerzeichen** (SPACE), sogar wenn es am Ende eines Strings steht.

Während **Standardvariable** einen festen Platz haben, müssen die Speicherplätze für die übrigen Variablen erst eingerichtet werden. Die **Länge der Strings** in allgemeinen Zeichenvariablen entspricht genau der Länge von Standard-Zeichenvariablen.

Benötigt man **längere Zeichenketten**, muß man vorher eine oder mehrere Zeichenvariable **dimensionieren** mit Hilfe der DIM-Anweisung. Stellt man dabei alle **DIM-Anweisungen** an den Anfang des Programms, wird nicht erst bei Auftreten der Variablen im Programmablauf, sondern schon gleich nach Programmstart der für diese Variablen benötigte Speicherplatz reserviert.

Bei den Rechnern **MZ 700/ MZ 800** können Strings bis zu 256 Zeichen lang sein. Bei SHARP-PCs ist die Länge auf 80 Zeichen (79+CR) begrenzt, wobei die Länge in der DIM-Anweisung vermerkt sein muß, z.B.:

```
(1)   DIM A1$(256)*80                oder
(2)    DIM  A2$(99,  71)*11
```

Im Beispiel (1) liegt eine **Liste** vor, gebildet von den 256 Elementen der Zeichenvariablen A1$. Jedes Element kann bis zu 80 Zeichen aufnehmen. Im Beispiel (2) sprechen wir von einem **2-dimensionalen Feld,** bestehend aus in eine **Matrix** eingefügten Elementen. Die Elemente sind daher in **Zeilen** und **Spalten** angeordnet. Dabei repräsentiert die 1. Zahl in der Klammer

die Anzahl der Zeilen und die 2. Zahl die Anzahl der Spalten. Da man immer von 0 zählt, gibt es im Beispiel 100 Zeilen und 72 Spalten. Das erste **Element** heißt somit A2$(0,0) und das letzte A2$(99,71). In diesem Beispiel kann jedes Element 11 Zeichen aufnehmen.

Ganz analog zu Feldern mit String-Elementen gibt es natürlich auch **Felder mit numerischen Elementen.** Solche Felder enthalten also Zahlen, mit denen man rechnen kann, wobei das ganze Feld auch als Matrix aufgefaßt werden kann, auf die die Regeln der Matrizenrechnung anwendbar wären. Das entsprechende Feld zum Beispiel (2) würde durch DIM A2(99, 71) dimensioniert werden.

Listen wie im Beispiel (1) sind eigentlich nur Spezialfälle von Feldern, nämlich **1-dimensionale Felder,** und daher werden wir immer nur von Feldern sprechen. Beim **MZ 700/ MZ 800** gibt es auch **3-dimensionale Felder,** z.B.:

(3) DIM A3$(99,71, 1)

Die Elemente dieses Feldes kann man sich angeordnet denken in 100 Zeilen, 72 Spalten und 2 Schichten, also in einem **räumlichen Gebilde.** Es sind natürlich 2 Schichten, weil von 0 gezählt wird. Für weitere Einzelheiten über die Variablen sollte der Leser das Handbuch seines Rechners zu Rate ziehen.

2. Manipulationen an Zeichenketten.

Die Zeichenvariablen dienen uns als **Textspeicher**, sie sollen aber auch Veränderungen des Textes zulassen. Dazu stehen in BASIC eine Reihe von Funktionen zur Verfügung. Zunächst können wir Strings **"addieren"**, d.h. es ist erlaubt, die Zeichen aneinanderketten wie im folgenden Beispiel:

```
(4)    S$=S1$+"-"+S2$      wobei
     S1$="SHARP"  und  S2$="Computer"
```

somit S$="SHARP-Computer". In (4) steht zwischen S1$ und S2$ die **Zeichenkonstante** "-". Ihr Inhalt bleibt auf jeden Fall unverändert, im Gegensatz zu den Zeichenvariablen, die man auch mit anderem Inhalt füllen könnte. Daher benötigen wir für die Zeichenkonstante nicht unbedingt einen Namen, obwohl auch nichts dagegen spricht, sie beispielsweise durch S3="-" zu kennzeichnen.

Weiter ist es möglich, eine **Rangordnung** von Strings aufzustellen.

```
(5)    S2$<S1$
```

bedeutet, daß S2$="Computer" in der **lexikographischen Ordnung** vor S1$="SHARP" steht. Dabei sind die Zeichen selbst meist nach dem **ASCII-Standard** geordnet, bei dem jedes verfügbare Zeichen eine Nummer bekommt. In BASIC schreibt man dafür **ASC (Zeichen)**. So ist ASC "C"= 67. Man kann auch bilden **ASC (Zeichenkette)**. So ist ASC "Computer" ebenfalls = 67. Von dem String wird nämlich immer nur das 1. Zeichen ausgewertet. Die Großbuchstaben rangieren bei dem ASCII-Standard sämtlich vor den Kleinbuchstaben. Deshalb ist die ASC-Funktion nur eingeschränkt für **Sortierroutinen** (Routine = Hilfs{unter}programm) tauglich.

Verwechseln Sie bitte nicht ASC "C" mit ASC (C$). Im ersten Fall erhalten Sie die Nummer für die **Zeichenkonstante** "C", im zweiten Fall liegt eine **Zeichenvariable** vor mit unbestimmtem Inhalt. Wenn C$ beispielsweise="Esel" ist, wird ASC (C$)= 69. Überprüfen Sie dies! Im Gegensatz zu den Rechnern MZ 700/800 kann man bei den SHARP-PCs die **Klammern vor dem Funktionsargument** weglassen, also ASC C$ schreiben. Solche Abweichungen in der BASIC-Schreibweise kommen mehrfach vor, ohne daß wir sie besonders erwähnen. Schauen Sie bitte in Ihrem Handbuch nach! Wenn nichts anderes erwähnt ist, beziehen wir uns hier auf das BASIC des PC 1500(A).

Wenn man nun den **Nummern-Code** eines Zeichens kennt, liegt es auf der Hand, daraus wieder das Zeichen selbst zu erzeugen. Dies geschieht mit der CHR$-Funktion:

```
(6)    CHR$ 67=  "C"
```

Hiermit ist es möglich, beispielsweise unsere Zeichenkette S1$ aus Beisp. (1) durch **Zeichenaddition** zu erzeugen:

```
(7)    S1$  =  CHR$83+CHR$72+CHR$65+CHR$82
       +CHR$80
```

Um Zeichenketten wieder zu **zerlegen**, haben wir 3 Funktionen zur Verfügung.

```
(8)    T1$  =  LEFT$(S$,J)
       T2$  =  RIGHT$(S$,J)
       T3$  =  MID$(S$,I,J)
```

Hierbei ist S$ der String, der zerlegt werden soll, J die Anzahl der Zeichen, die entnommen werden soll und I die Position, wo das 1. Zeichen herkommen soll. Bei **LEFT$** werden

die Zeichen von links genommen und ihre Anzahl **von links** gezählt. Bei **RIGHT$** werden die Zeichen vom rechten Ende des Strings genommen und ihre Anzahl **von rechts** gezählt. Bei **MID$** verfährt man wie bei LEFT$, nur daß das 1. zu entnehmende Zeichen, von links gesehen, erst an I. Stelle steht. Bei der Verwendung dieser Funktionen in Programmen muß man darauf achten, daß I nie <1 wird, denn die **Zählung der Position** eines Zeichens beginnt mit 1, nicht mit 0. Dagegen darf J=0 sein, mit dem Effekt, daß der entstehende **Teilstring** leer ist, d.h. keine Zeichen enthält. Man schreibt das so:

```
(9)     T$  =  MID$(S$,I,0)
        T$  =  ""
```

und ebenso bei LEFT$ und RIGHT$. **Negative Funktionsargumente** sind nicht zugelassen, dagegen darf J größer sein als die Anzahl der überhaupt vorhandenen Zeichen im vorliegenden String. So ergibt sich z.B., wenn S1$="SHARP" und T$=MID$(S1$, 4, 66), als Resultat T$="RP".

Die **Zerlegung** unseres Eingangsbeispiels S$ = SHARP-Computer in die Komponenten S1$, S2$ und S3$ aus Beisp. (4) erfolgt dann so:

```
(10)    S1$  =  LEFT$(S$,5)
        S2$  =  RIGHT$(S$,8)
        S3$  =  MID$(S$,6,1)
```

,also S1$="SHARP", S2$="Computer" und S3$="-".

Sehr wichtig ist auch, die Anzahl der Zeichen einer Zeichenkette festzustellen oder, wie man ebenfalls sagt, die **Länge des Strings** zu bestimmen mit Hilfe der nachfolgenden Funktion

```
(11)    L  =  LEN  S$
```

Für S$="SHARP-Computer" ergibt sich L=14. Dies

paßt noch in eine Standardvariable.

Manchmal stehen in einem String nur Ziffern und keine sonstigen Zeichen, abgesehen vielleicht vom Dezimalpunkt. Wenn man mit den in der Zeichenvariablen enthaltenen Ziffern rechnen will, muß man erstere zuerst in eine numerische Variable umwandeln mit Hilfe der **VAL-Funktion**. Umgekehrt kann man auch eine Zahl in einen String umwandeln mit der **STR$-Funktion**. Für das Beispiel '4711' sieht das so aus:

```
(12)    X  =  VAL'4711'    oder    X = VALX$
        X  =  4711

        X$  =  STR$  X
        X$  ='4711'
```

Es kommt oft vor, daß Zahlen zunächst in eine Zeichenkette verpackt werden, um sie erst bei Bedarf zu einer rechenbaren Zahl zu machen. Ein Vorteil, von einer Zahl einen String zu bilden, ist, daß **Strings linksbündig** ausgedruckt werden (auch auf dem Display) und die 1. Ziffer so an genauer Position erscheint.

Speziellen Programmierkomfort bietet die Funktion **INSTR beim PC 1600**. Sie sucht in einem String nach der Position, wo ein bestimmtes Zeichen (von links gesehen) zum ersten Mal auftaucht. Damit könnte man den Teilstring S3$ aus Beisp. (10) wie folgt finden:

```
(13)    I     =  INSTR(S$,'-')
        S3$   =  MID$(S$,I,1)
```

3. Zahlen und Codes

Wir haben jetzt einen überblick über die für die Textverarbeitung so grundlegenden Zeichenketten gewonnen. Nun wollen wir uns etwas mit der **Verschlüsselung** der Zeichen befassen.

Es muß daran erinnert werden, daß der Computer **zur Darstellung** von Zahlen und Zeichen - ohne auf die dahinterstehenden elektrischen Vorgänge einzugehen - im Grunde **nur 2 Ziffern**, nämlich **0 und 1**, zur Verfügung hat. Nun ist die 'Vorratskammer für Informationen' des Rechners, sein Speicher, gleich, ob **ROM** (Festwertspeicher) oder **RAM** (Speicher mit beliebigem, sprich wahlfreiem Zugriff), **in folgender Weise organisiert:** Jede Speicherzelle ist ein kleines "Kommödchen" mit 8 Schubladen. Das Kommödchen heißt **Byte**, eine Schublade heißt **Bit**.

Wenn Sie nun, um im Bilde zu bleiben, in eine beliebige Schublade einen Zettel legen, gilt das Bit als **gesetzt = 1** (die Schublade ist belegt). Kommt kein Zettel hinein, gilt das entsprechende Bit als **zurückgesetzt = 0** (man sagt auch gelöscht). Wenn Sie nun ausprobieren, wieviele verschiedene Beladungen es für ein "Kommödchen" gibt, kommen Sie auf die **Anzahl 2 hoch 8 = 256**. Soviele Zeichen kann der Computer im Prinzip darstellen. Bleibt die letzte Schublade unbenutzt, sind es nur die Hälfte, nämlich 128.

Im ersten Fall sprechen wir von einer **8-Bit-Verschlüsselung (8-Bit-Code)**, im zweiten Fall liegt ein **7-Bit-Code** vor. Nun gibt es einen gebräuchlichen 7-Bit-Standard, den **ASCII-Code**, den die meisten Computer benutzen. Das gilt auch für **SHARP-Computer**, wobei der Code bei verschiedenen Geräten **zu einem 8-Bit-Code erweitert** worden ist. So finden sich in der Erweiterung beim **MZ 700/800** auch die deutschen Umlaute und beim **PC 1350** etwa japanische Zeichen. Beim **PC 1500(A)** kann man den Zeichensatz selbst erweitern (sog. 2. Zeichensatz), und beim **PC 1600** findet man den 8-Bit-Code von IBM, der selbst ein Standard ist. Aber auch der IBM-Code ist ein erweiterter ASCII-Code.

Mit jeder Schublade, die Sie zusätzlich zur Verfügung haben, können Sie **genau doppelt so viele Zahlen** darstellen wie vorher. So kommen Sie zur **Binär-Darstellung** einer Zahl. Z.B.:

(14) 00100101 (binär) = 37 (dezimal)

Die Bits (Schubladen") sind **numeriert.** Im obersten Bit, dem Bit 7, steht hier eine 0, im untersten Bit, dem Bit 0, steht hier eine 1. Nun reicht ein byte zur Darstellung größerer Zahlen nicht aus, daher benötigt man mindestens ein zweites. Wenn man von diesem 2. Byte nur das Bit 0 benutzt, kann man schon Zahlen bis 511 = 2*256-1 darstellen. Nun benutzt man üblicherweise das 2. Byte bis zum Bit 6. Bit 7 des 2. Bytes dient zur Aufnahme des **Vorzeichens.** Steht hier eine 0, ist die Zahl **positiv,** findet sich dort eine 1, ist die Zahl **negativ.** Auf diese Weise lassen sich **Zahlen von -32768 bis +32767** bzw. von -(2 hoch 15) bis +(2 hoch 15)-1 darstellen.

Kommen wir nun zu den **Hex(adezimal)-Zahlen** und betrachten wir wieder Beisp. (14). Um es vorwegzunehmen: ein **Hexadezimalsystem** ist ein Zahlensystem, das aus **16 Ziffern** aufgebaut ist, nämlich 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F. Da wir normalerweise zuwenig Ziffern haben, nehmen wir einfach die **Buchstaben A bis F** dazu. Wozu wird dies nun benötigt? - Die Antwort ist etwa so: um auf Bytes bezogene Binärdarstellungen von Zahlen **kürzer und übersichtlicher** wiederzugeben. Dazu teilen wir unser Byte in ein **oberes Halbbyte** (die 4 oberen Schubladen) und ein **unteres Halbbyte** (die 4 unteren Schubladen). Für Beisp. (14) sieht das so aus:

(14a) 0010 (binär) = 2 (hexadezimal)
0101 (binär) = 5 (hexadezimal)

also ist 00100101 (binär) = 25 (hexadezimal)

Hier kommen wir mit den üblichen Ziffern von 0 bis 9 aus, im Gegensatz zum nächsten Beispiel:

(15) 10101111 (binär) = 175 (dezimal)

Da uns jetzt die **Hex-Darstellung** interessiert, teilen wir das Byte wieder in 2 Halbbytes.

(15a) 1010 (binär) = A (hexadezimal)
1111 (binär) = F (hexadezimal)

also ist 10101111 (binär) = AF (hexadezimal). Dies ist nichts anderes als in Beisp. (14a), nur daß wir jetzt Buchstaben heranziehen mußten, um die 10 (dezimal) als Hex-Zahl, nämlich A (hexadezimal), desgleichen 15 (dez) als F (hex), darstellen zu können. **Hex-Zahlen** kennzeichnen wir von nun an durch **vorangestelltes &**. Die beiden Hexzahlen aus (14) und (15) heißen also: &25 und &AF.

Wir stellen hier noch einmal die **Halbbyte-Binärdarstellungen** den Hex-Ziffern gegenüber:

(16)

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

Natürlich kann man auch **größere Hex-Zahlen** bilden, und zwar einfach durch **Zusammenfassung aufeinanderfolgender Bytes**. So entspricht die Zahl 65535 (dezimal) = &FFFF.

Hex-Zahlen haben eine große Bedeutung für die **Adreßrechnung bei Maschinensprache/Assembler**,

aber jeder, der sich (auch in BASIC) darum kümmert, wo seine Daten und Texte eigentlich abgeblieben sind im Speicher, sollte damit gut vertraut sein. Übrigens kann man den (ASCII-) **Code jedes beliebigen Zeichens durch 2 Hexziffern** angeben. So entspricht etwa ASC "A" =&41 ASC "*" = &2A und ASC "1" = &31.

Wir betrachten nun noch die Art und Weise, wie eine Zahl in einer **numerischen Variablen** abgespeichert ist: als **Binär Codierte Dezimalzahl (BCD-Zahl)** . Wir untersuchen dies beim 1500(A). Bei anderen Rechnern sind die Verhältnisse ähnlich.

Jede BCD-Zahl benötigt 8 Bytes. Die Darstellung erfolgt in **wissenschaftlichem Format:** 1 Byte für den **Exponenten,** 1 Byte für das **Vorzeichen** der Mantisse, 5 Bytes für die **Mantisse,** und am Ende folgt ein Byte mit &00. Z.B.:

(17) 10*PI= (10 hoch 1)*(+3.141592654)

Der Exponent im 1. Byte ist als Binärzahl dargestellt (zugelassen für -99 bis +99). Im 2. Byte steht &00 für positiv und &80 für negativ. Dann folgt der **Zifernteil** (die Mantisse) der Zahl ohne Berücksichtigung des **Dezimalpunktes** mit 10-stelliger Genauigkeit. D.h. jedes Byte für die Mantisse nimmt 2 Ziffern auf. Die binäre Darstellung der Ziffern in jedem Halbbyte erfolgt wie in (16), aber eben nur für die Ziffern 0 bis 9, da es sich, wie gesagt, um eine **Dezimaldarstellung** handelt. Die Stellung des Dezimalpunktes (oder -kommas) drückt sich im Exponenten aus: je größer er ist, desto weiter rückt das Komma nach rechts.

4. Befehle zur Texteingabe

Zur Texteingabe steht, wie allgemein bekannt, in BASIC die **INPUT-Funktion** zur Verfügung. Wir

benutzen sie in Verbindung mit einer Zeichenvariablen, z.B. INPUT A$(0). Bei dieser Form erscheint an der Stelle, wo die Texteingabe beginnt, ein **Fragezeichen**. Dieses wird bei den Formen INPUT "";A$(0) und INPUT "Text: ";A$(0) **unterdrückt**. Besser noch ist die Form:

```
(18)  PRINT "Text:": CURSOR C: INPUT"";A$(0)
```

bei beliebigem C [0<=C<26 beim PC 1500(A)]. Während Texteingaben bei INPUT durch <ENTER> abgeschlossen werden müssen und unmittelbar angezeigt werden, ist dies bei **'INKEY$-Schleifen'** beides nicht der Fall. Nach der Zuweisung K$= INKEY$ wird bei jedem Ansteuern dieser Programmzeile der ASC-Code der gerade gedrückten Taste in K$ gespeichert. Erfolgt **kein Tastendruck** , wird K$ der Wert 0 zugewiesen. So verharrt das Programm in einer Zeile , bis ein Tastendruck erfolgt, z.B.:

```
(19)   10  K$=  INKEY$: IF ASC K$= 0 THEN 10
oder   10  K$=  INKEY$:  IF  K$=  "" THEN 10
```

5. **Befehle zur Textausgabe**

Zur Textausgabe IN BASIC auf dem Display dient der Befehl **PRINT**. Bei verschiedenen Computern gibt es unterschiedliche Möglichkeiten, diesen Befehl formatiert, d.h. mit dem **USING-Zusatz** zu gebrauchen - hier muß auf die Handbücher verwiesen werden. Zur Positionierung dient der **CURSOR-Befehl** mit 1 oder 2 angehängten Argumenten (CURSOR Spalte,Zeile). Wenn man nicht **WAIT n** eingegeben hat, stoppt der Computer (aber nicht der PC 1600) bei jedem PRINT. Der Befehl **PAUSE** setzt das Programm nach kurzer Anzeige fort. Zur Textausgabe auf Papier dient **LPRINT**, das mit dem Zusatz **TAB** positioniert werden kann. Für besondere Ein- und Ausgabemethoden (Maschinensprache) wird auf andere einschlägige Kapitel des Buches verwiesen.

Kap.II: ERZEUGUNG UND DARSTELLUNG DER ZEICHEN

1. Der '2. Zeichensatz' für den PC 1500(A)

Beim PC 155(A) haben wir die Möglichkeit, bis zu **128 neue Zeichen** zu kreieren, sowohl für das Display als auch für den Plotter CE-150. Für andere Rechner gibt es diese Möglichkeit so nicht, aber wie wir noch sehen werden, ist die Methode auch für andere SHARP-Computer und -Plotter nützlich.

Bleiben wir vorerst beim PC 1500(A). Die neuen Zeichen erhalten die Code-Nummern (ASC-Nummern) 128 bis 255 bzw. &80 bis &FF. Man sollte aber vielleicht **nicht alle Codes benutzen**. Schon R. v. Schlichtegroll hat darauf hingewiesen, daß eine **Neubelegung** der Tasten nur mit den Codes von &80 bis &DF erfolgen darf, da sonst **Fehlfunktionen** im BASIC-Interpreter auftreten können. Dennoch können alle Codes von &80 - &FF verwendet werden, denn man kann jedes Zeichen ja auch indirekt durch PRINT bzw. LPRINT CHR$ (Code-Nummer) erzeugen.

Der '2. Zeichensatz' wird **initialisiert** durch Aktivierung der **Speicherzellen &764E, &785D, &785E.** Damit wird der **Standard-Zeichensatz** (Codes bis &7F) ausgeblendet, und der Rechner greift nun auf **4 Tabellen im RAM** zu, die Sie vorher dort angelegt haben und die in bestimmter Weise aufeinander Bezug nehmen.

Dazu ein kleiner **Exkurs über Adressen**. Der Computer braucht, um eine Speicherzelle eindeutig ansprechen zu können, eine Adresse, die als HEX-Zahl 2 Byte einnimmt, z.B. &1000. Das 1. Byte (&10) nennen wir **High-Byte** und das 2. Byte (&00) heißt das **Low-byte** der Adresse.

Legen Sie nun eine **Tabelle** für die **Display-**

Zeichen (LCD-Tabelle) an, deren Anfangsadresse im Low-byte immer &00 sein muß, wie es z.B. bei &1000 ist. Das High-byte POKEn Sie später zur **Initialisierung** in das richtige **Systembyte** durch POKE &765E,&.., hier also POKE &765E,&10

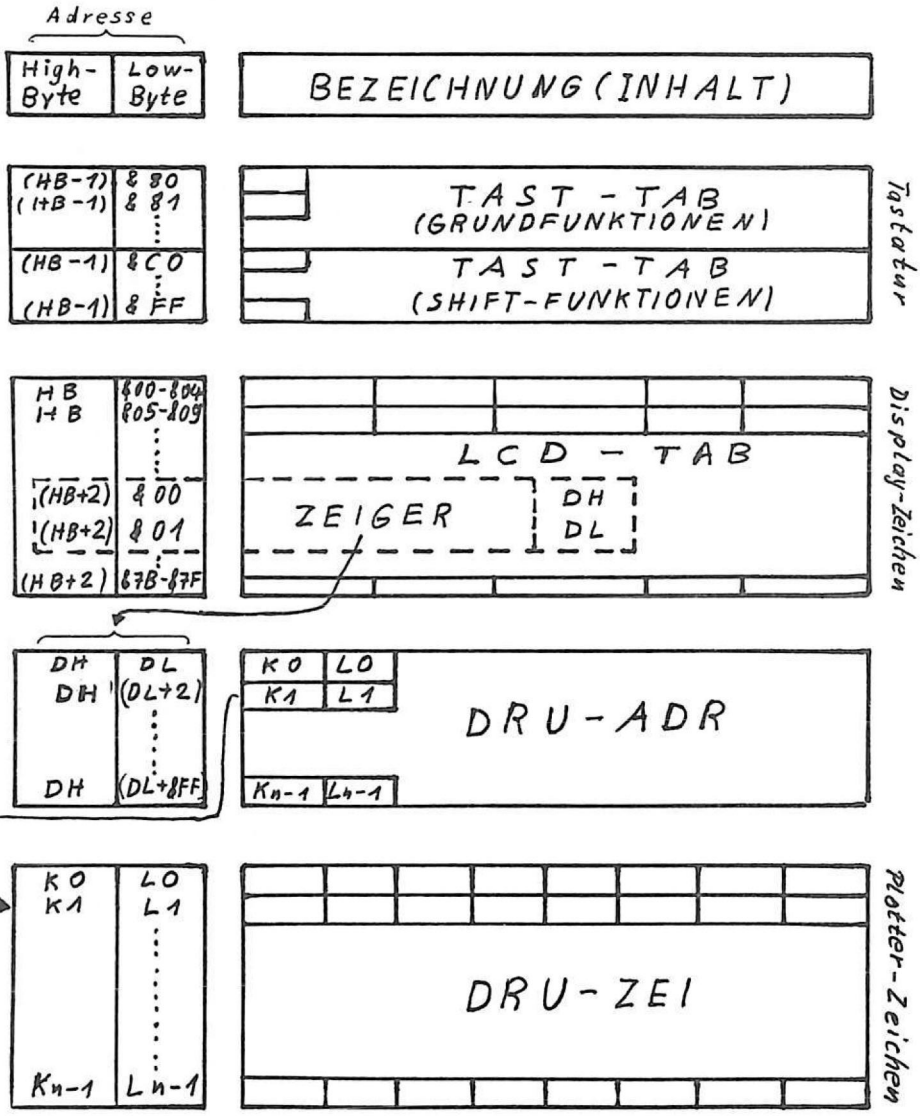
Für jedes Zeichen benötigen Sie 5 bytes in der LCD-Tabelle, entsprechend den 5 Spalten (GCURSOR-Positionen) eines Zeichens. Die **Zeichenmuster**, die Sie dort ablegen, entsprechen völlig der **GPRINT-darstellung** in BASIC für ein Zeichen, z.B. für den Buchstaben A:

```
(20)    GPRINT "7C1211127C"
        POKE &1000,&7C,&12,&11,&12,&7C
```

Sie werden natürlich nicht gerade dieses Muster entwerfen, da es sich ja schon im **Standardzeichensatz** befindet. Dieses 1. Zeichen bekommt den Code &80, das nächste, mit der Adresse &1005 den Code &81 usw., immer 5 Bytes weiter in der LCD-Tabelle. Diese strenge **Verknüpfung der Tabellenadresse** mit dem dazugehörigen Code ist erforderlich. Für den Code &FF gilt die Adresse (hier): &127B

Um diese Zeichen auch auf dem Plotter ausdrukken zu können, benötigen Sie noch 2 Tabellen: eine **Druckzeichen-Tabelle** (DRU-ZEI-Tabelle) und eine **Drucker-Adressen-Tabelle** (DRU-ADR-Tabelle). Dabei gibt DRU-ADR lediglich an, wo in DRU-ZEI irgendein Druckzeichen zu finden ist. DRU-ADR braucht daher für jede Adresse (wie üblich) 2 Bytes. Wo aber befindet sich DRU-ADR im RAM? - Dafür müssen Sie einen **Zeiger setzen**, und dieser Zeiger hat hier die Adressen &1200 und &1201. Diese beiden Bytes enthalten den Beginn der DRU-ADR-Tabelle. **Allgemein gilt** also: High-byte des Beginns der LCD-Tabelle + &02, diese Zahl gibt immer das hgih-byte der Position des Zeigers an.

TABELLEN (2. ZEICHENSATZ)



Wie die **Tabellen miteinander verzahnt** sind, sehen Sie auf der vorstehenden Skizze. Eine **weitere Tabelle**, die die **Belegung der Tasten** mit den Codes für die neuen Zeichen aufnimmt, beginnt 128 bytes vor der LCD-Tabelle. Die **Standardbelegung** der Tasten finden Sie in einer übersicht auf der nächsten Seite. Man kann diese mit Hilfe eines Maschinenprogramms (was wir später tun) einfach **ins RAM kopieren**, wenn man nur einzelne Tasten neu belegen will.

Bevor der 2. Zeichensatz benutzt werden kann, muß das **Systembyte &764E aktiviert** werden, indem man hier das bit 2 setzt:

(21) POKE &764E,(PEEK &764E)+&04

Bei der Verwendung des 2. Zeichensatzes darf man **nicht die SML-Taste benutzen**, da man hiermit dieses bit wieder zurücksetzt. **Erlaubt** ist die Betätigung der **SHIFT-Taste.**

Ebenso muß eine **weitere Systemzelle aktiviert** werden:

(22) POKE &785D,&00

Wenn Sie **nur neue Displayzeichen**, aber keine Druckerzeichen initialisieren wollen, geben Sie stattdessen POKE &785D,&80 ein. Beim **Ausschalten des Rechners**, sofern dies nicht durch AUTO POWER OFF geschieht, wird in die Zelle &785D der Wert &FF geschrieben. Folglich müssen bei Wiedereinschalten die Werte 0 oder &80 wieder in dieses Systembyte 'gePOKEt' werden. Unverändert beim Ausschalten bleibt die **dritte Systemzelle &785E**, in die durch

(23) POKE &785E, HB

das high-byte des Starts der LCD-Tabelle kommt.

TASTATUR - STANDARDBELEGUNG

(vorangestelltes '-' = SHIFT

Sp.1=Taste Sp.2=Code Sp.3=relat. Adr. in TAST-TAB)

SHI	01	83	F3	13	AB	- F2	22	E3		1	31	97
-SHI	01	C3	F4	14	BB	- F3	23	EB	-	1	31	D7
SML	02	B0	F5	15	93	- F4	24	FB		2	32	87
-SML	02	F0	F6	16	9B	- F5	25	D3	-	2	32	C7
◄	08	AE	CL	18	B5	- F6	26	DB		3	33	BF
◆	09	88	RCL	19	A0	(	28	99	-	3	33	FF
- ◆	09	C8	-RCL	19	E0	)	29	9F		4	34	96
↓	0A	B8	- CL	1A	F5	*	2A	A6	-	4	34	D6
↑	0B	80	DEF	1B	B3	+	2B	A7		5	35	86
►	0C	B7	-DEF	1B	F3	- -	2C	CE	-	5	35	C6
ENT	0D	98	- ►	1C	F7	-	2D	8E		6	36	BE
-ENT	0D	D8	- ◄	1D	EE	.	2E	8F	-	6	36	FE
OFF	0F	8D	-MOD	1E	F6	- .	2E	CF		7	37	95
-OFF	0F	CD	MOD	1F	B6	/	2F	A5	-	7	37	D5
F1	11	8B	SPA	20	A8	0	30	90		8	38	85
F2	12	A3	- F1	21	CB	- 0	30	D0	-	8	38	C5

9	39	BD	H	48	84	X	58	89	k	6B	DC
- 9	39	FD	I	49	9A	Y	59	82	l	6C	DE
- *	3A	E6	J	4A	94	Z	5A	B1	m	6D	D1
- +	3B	E7	K	4B	9C	- ↑	5B	C0	n	6E	C1
- (	3C	D9	L	4C	9E	- ↓	5D	F8	o	6F	DD
=	3D	AF	M	4D	91	-SPA	5E	E8	p	70	ED
-)	3E	DF	N	4E	81	a	61	F4	q	71	F2
- /	3F	E5	O	4F	9D	b	62	F9	r	72	EA
- =	40	EF	P	50	AD	c	63	E1	s	73	CC
A	41	B4	Q	51	B2	d	64	E4	t	74	FA
B	42	B9	R	52	AA	e	65	E2	u	75	D2
C	43	A1	S	53	8C	f	66	EC	v	76	E9
D	44	A4	T	54	BA	g	67	FC	w	77	CA
E	45	A2	U	55	92	h	68	C4	x	78	C9
F	46	AC	V	56	A9	i	69	DA	y	79	C2
G	47	BC	W	57	8A	j	6A	D4	z	7A	F1

Wir müssen nun noch sagen, **welche Informationen in die Druckzeichen-Tabelle** (DRU-ZEI) kommen. Jedes byte nimmt hier 4 Informationen auf: a) Länge eines Striches (bit 0 - 2), b) Richtung des Striches (bit 3 - 5), c) Stift oben/unten (bit 6), d) Zeichenende? (bit 7). Jeder Strich muß mindestens eine und darf höchstens 7 Einheiten lang sein, da 111(bin) = 7 die höchste in 3 bit darstellbare Zahl ist.

Als Richtungen sind die **8 Richtungen** der Windrose zugelassen:

(24)	OST	000 (bin)	WEST	100 (bin)
	NO	001 (bin)	SW	101 (bin)
	NORD	010 (bin)	SÜD	110 (bin)
	NW	011 (bin)	SO	111 (bin)

Bit 6 erhält für Stift unten eine 1, sonst 0. Bit 7 bekommt nur dann eine 1, wenn das Zeichen mit dem betreffenden Strich beendet ist.

Alle Informationen eines Bytes faßt man nun üblicherweise zu einer **zweistelligen Hex-Zahl** zusammen. Beispiel:

(25)	Länge (bit 0-2)=5		101 (bin)
	Richtung (bit 3-5)=NW		011 (bin)
	Stift unten		1 (bin)
	Ende?	nein	0 (bin)

Das ergibt 01011101 (bin) = &5D

Wir wollen noch erwähnen, daß für **Zeichen, die nicht ausgedruckt** werden sollen oder können, in der Tabelle DRU-ADR &A000 (in den 2 dafür vorgesehenen Bytes) stehen sollte. Damit werden dann jeweils **Leerzeichen (SPACE)** ausgedruckt. Der letzte Stich eines Zeichens muß unbedingt in ein 6 * 4 großes **Rasterfeld** (siehe Punkt II.2) zurückführen. War der Stift oben, wird noch ein Byte mit &CO angehängt.

2. Ein Zeichengenerator

Sie sehen also, was man alles beachten muß, wenn man einen 2. Zeichensatz für den PC 1500(A) entwerfen will. Die Zeichen, die man etwa auf **Millimeterpapier** vorzeichnen kann, sind Punkt für Punkt und Strich für Strich **umzusetzen in Informationen** für die 4 Tabellen. Der kleinste Fehler kann dabei zum Systermabsturz oder **irregulärem Verhalten des Plotters** CE-150 führen. Sollte Ihnen letzteres widerfahren, hilft oft nur Abziehen des Rechners vom Plotter.

Nach solchen Erfahrungen reifte beim Verfasser der Entschluß, die üblichen **Berechnungen zu automatisieren** (wozu ist ein Computer sonst da?), woraus der vorliegende **Zeichengenerator** entstand. Es können sowohl **Display-** als auch **Plotterzeichen** entworfen werden allein durch Drücken entsprechender Tasten. Die neuen Zeichen entstehen **vor Ihren Augen** auf dem Display und/oder dem Plotter und werden gleich richtig in den zuständigen Tabellen verankert.

Das **Programm 'ZEI-GEN'** wird gestartet mit RUN oder <DEF><Z>. Sodann ist das **High-byte der LCD-Tabelle** anzugeben. Wie Sie schon gemerkt haben, ist dieses immer der **Bezugspunkt** für alle anderen Tabellen. Setzen Sie aber vor Laden des Programms 'ZEI-GEN' den BASIC-Pointer durch NEW &.... ausreichend hoch, damit Ihre Tabellen vor überschreiben durch das BASIC-Programm geschützt sind. Je komplizierter Ihre Plotterzeichen ausfallen, desto mehr Speicherplatz werden Sie dafür benötigen. Es ist anzuraten, hier nicht zu knausern und den BASIC-Pointer lieber nach vollendeter Installation des 2. Zeichensatzes etwas zurückzusetzen. (Dabei würde das Programm ZEI-GEN, das dann ja nicht mehr gebraucht wird, gelöscht.)

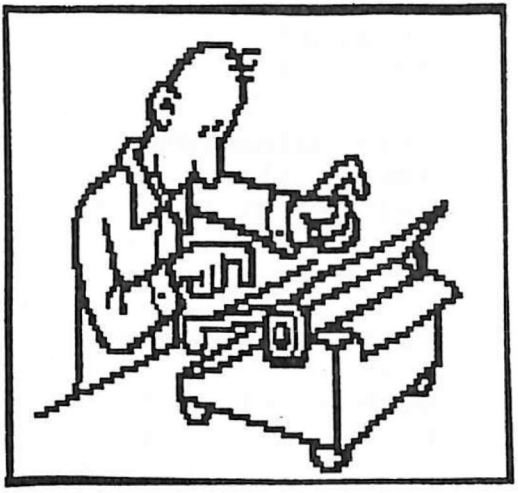
```
  100 REM ************************
  102 REM *   ZEI-GEN   V.1.0    *
  104 REM ************************
  110 REM (C) 1986 by W.MEYER,HASLOH
  115 REM --------------------
  120 "Z"CLEAR :CLS :P=20:Q=20:S=9
  121 DIM G(4,6),P(4),AD(64),K(64),L(64)
,H(47)
  122 INPUT "High-byte/LCD (HB)= ";HB
  123 IF HB<2BEEP 3:CLS :GOTO 122
  124 POKE &785D,0,HB:POKE &764E,&44:GOS
UB "TAST":GOSUB "LL"
  125 REM --------------------
  126 "C"WAIT 0:C=160:INPUT "CHR$ (160-2
23)...(0=ENDE)";C:IF C<32BEEP 1,99,99:EN
D
  127 Z=0:INPUT "DISPLAY(1)/PLOT(2)";Z
  128 IF Z=1GOSUB "D":CLS :GOTO 125
  131 TEXT :LF 8
  132 TB=&600:INPUT "TAB-ADRESSE...";TB:
REM * TB=(HB+4)*256+&40 <min.&600>
  133 IF TB<257BEEP 3:CLS :GOTO 132
  134 AD=TB
  135 REM --------------------
  136 "NEXT"GRAPH :CSIZE 2:GLCURSOR (50,
0):SORGN :X=0:Y=0
  137 GX=0:GY=0:COLOR 3
  140 FOR H=0TO 5
  142 FOR J=0TO 3:GLCURSOR (GX,GY):LPRIN
T ".":GX=GX+20
  144 NEXT J:GX=0:GY=GY+20:NEXT H:COLOR
2
  146 GLCURSOR (0,0)
  150 CLS :PAUSE "=> ZIFFERN (5=PEN UP/D
OWN)"
  155 IF TB=8BEEP 3:CLS :GOTO 150
  160 CLS :CURSOR 10:PRINT "Ziffer! (E=E
NDE)"
  189 REM --------------------
  190 "MO"R=0:Z=0:ZA=0:ZB=0:M=0:M$="":I=
0:ST=0:S=9
  194 REM --------------------
  195 "INK"K$=INKEY$ :IF K$=""THEN "INK"
  197 IF K$="E"GOSUB "LL":TEXT :BEEP 3:E
ND
  200 IF M=0AND K$="5"AND S=9LET S=0:CLS
 :PAUSE "PEN DOWN":ST=1:GOTO "INK"
  201 IF M=0AND K$="5"AND S=0LET S=9:CLS
 :PAUSE "PEN UP":ST=0:GOTO "INK"
  202 IF K$="5"AND S=9LET S=0:CLS :PAUSE
 "PEN DOWN":GOSUB "STW":ST=1:GOTO "INK"
  203 IF K$="5"AND S=0LET S=9:PAUSE "PEN
 UP":GOSUB "STW":ST=0:GOTO "INK"
  204 IF K$=" "AND L=1RLINE -(0,150),9:L
=0:GOTO "INK"
  205 IF K$=" "AND L=0RLINE -(0,-150),9:
L=1:GOTO "INK"
  206 IF K$<>M$OR M$="5"GOTO "GRW"
  207 REM --------------------
  208 "REPEAT"ZA=ZA+1:M$=K$:IF ZA=8BEEP
1,100,100:ZA=7:GOTO "INK"
  210 IF K$="6"RLINE -(P,0),S:X=X+1:GOSU
B 810:GOTO "INK"
  220 IF K$="9"RLINE -(P,Q),S:X=X+1:Y=Y+
1:GOSUB 820:GOTO "INK"
  230 IF K$="8"RLINE -(0,Q),S:Y=Y+1:GOSU
B 830:GOTO "INK"
  240 IF K$="7"RLINE -(-P,Q),S:X=X-1:Y=Y
+1:GOSUB 840:GOTO "INK"
  250 IF K$="4"RLINE -(-P,0),S:X=X-1:GOS
UB 850:GOTO "INK"
  260 IF K$="1"RLINE -(-P,-Q),S:X=X-1:Y=
Y-1:GOSUB 860:GOTO "INK"
  270 IF K$="2"RLINE -(0,-Q),S:Y=Y-1:GOS
UB 870:GOTO "INK"
  280 IF K$="3"RLINE -(P,-Q),S:X=X+1:Y=Y
-1:GOSUB 880:GOTO "INK"
  290 GOTO "INK"
  299 REM --------------------
  300 "GRW"BEEP 1:IF M=0LET M=1:GOTO "RE
PEAT"
  310 IF ST=1LET Z=Z+64
  315 IF K$=CHR$ 13GOTO "ENDZEI"
  320 H(I)=Z:PRINT H(I):I=I+1:R=0:Z=0:ZA
```

```
=0:ZB=0
  340 GOTO "REPEAT"
  449 REM --------------------
  450 "ENDZEI"IF X>30R Y>5BEEP 3:GOTO "N
EXT"
  460 IF ST=0LET H(I)=Z:I=I+1:Z=&C0:H(I)
=Z:GOTO "PROBE"
  470 Z=Z+128:H(I)=Z:GOTO "PROBE"
  499 REM --------------------
  500 "PROBE"IX=I:TEXT :LF 8:FOR I=0TO I
X:LPRINT H(I);:NEXT I
  505 ZW=AD:AD=&77D0
  510 GOSUB "TAB":CSIZE 4:POKE &764E,&44
:LPRINT :LPRINT " ";CHR$ 224;:CSIZE 2:LP
RINT "    CHR$ ";C
  520 POKE &764E,&40:AD=ZW:GOTO "CHARAC"
  549 REM --------------------
  550 "TAB"FOR I=0TO IX
  560 IF H(I)=0RETURN
  570 POKE AD,H(I):AD=AD+1
  580 NEXT I:RETURN
  599 REM --------------------
  600 "CHARAC":INPUT "ZEICHEN OK(J/N)?";
J$
  605 IF J$<>"J"BEEP 3:END
  610 PRINT "CHR$";C:CURSOR 11:INPUT C
  620 IF C<160BEEP 1:CLS :GOTO 610
  630 REM AD=TB+(C-160)*&20
  635 AD(C-160)=AD:LPRINT "ADR.(";C;")=
";AD
  640 GOSUB "TAB":C=C+1:CLS :LF 8:GOTO "
NEXT"
  799 REM --------------------
  810 IF R=0LET ZB=0:R=1
  812 Z=ZA+ZB:RETURN
  820 IF R=0LET ZB=8:R=1
  822 Z=ZA+ZB:RETURN
  830 IF R=0LET ZB=16:R=1
  832 Z=ZA+ZB:RETURN
  840 IF R=0LET ZB=24:R=1
  842 Z=ZA+ZB:RETURN
  850 IF R=0LET ZB=32:R=1
  852 Z=ZA+ZB:RETURN
  860 IF R=0LET ZB=40:R=1
  862 Z=ZA+ZB:RETURN
  870 IF R=0LET ZB=48:R=1
  872 Z=ZA+ZB:RETURN
  880 IF R=0LET ZB=56:R=1
  882 Z=ZA+ZB:RETURN
  899 REM --------------------
  900 "STW"IF ST=1LET Z=Z+64
  910 H(I)=Z:PRINT H(I):I=I+1:R=0:Z=0:ZA
=0:ZB=0:M$=K$:M=0:RETURN
 1999 REM **********************
 2000 "D"WAIT 0
 2010 GOSUB "TAST":POKE &764E,&48
 2029 REM --------------------
 2030 "USW."CLS :GOSUB "NULL":I=3:J=5
 2040 PRINT "CHR$ =";C;"      (0=ENDE)":
CURSOR 11:INPUT C
 2045 IF C<32BEEP 1,99,99:END
 2050 CLS :PAUSE :PAUSE
 2055 GCURSOR 0:GPRINT "7F7F":GCURSOR 9:
GPRINT "7F7F"
 2057 GCURSOR 42:PRINT "<== ZEICHEN (R=R
ET)";
 2059 REM --------------------
 2060 "DISP"GOSUB "BLINK"
 2065 K$=INKEY$ :IF INKEY$ =""GCURSOR J:
GPRINT B0:FOR N=1TO 5:NEXT N:GCURSOR J:G
PRINT B1:FOR N=1TO 5:NEXT N:GOTO 2065
 2067 GCURSOR J:GPRINT A0
 2070 IF K$=CHR$ &0AAND I<6LET I=I+1:BEE
P 1:GOTO "DISP"
 2075 IF K$="N"LET I=6:BEEP 1:GOTO "DISP
"
 2080 IF K$=CHR$ &0BAND I>0LET I=I-1:BEE
P 1:GOTO "DISP"
 2085 IF K$="M"LET I=0:BEEP 1:GOTO "DISP
"
 2090 IF K$=CHR$ &0CAND J<7LET J=J+1:BEE
P 1:GOTO "DISP"
 2092 IF K$="+"LET J=7:BEEP 1:GOTO "DISP
"
```

```
 2095 IF K$=CHR$ &08AND J>3LET J=J-1:BEE
P 1:GOTO "DISP"
 2097 IF K$="-"LET J=3:BEEP 1:GOTO "DISP
"
 2100 IF K$="/"GCURSOR J:GPRINT "7F":S=1
:FOR I=0TO 6:GOSUB "BIT":GOSUB "MERK":NE
XT I:I=3:BEEP 1,99,99:GOTO "DISP"
 2110 IF K$="*"GCURSOR J:GPRINT "00":S=0
:FOR I=0TO 6:GOSUB "BIT":GOSUB "MERK":NE
XT I:I=3:BEEP 1,99,99:GOTO "DISP"
 2150 IF K$="0"LET S=0:BEEP 1,99,99:GOSU
B "BIT":GOSUB "MERK":GOTO "DISP"
 2160 IF K$="."LET S=1:BEEP 1,99,99:GOSU
B "BIT":GOSUB "MERK":GOTO "DISP"
 2170 IF K$=CHR$ 13GOSUB "BUFF":GOSUB "F
ERTIG":C=C+1:GOTO "USW."
 2180 IF K$=CHR$ &18GOSUB "NULL":GOTO "U
SW."
 2190 IF K$="R"RETURN
 2200 BEEP 1,50,50:GOTO "DISP"
 2399 REM --------------------
 2400 "BLINK"GCURSOR J:A0=POINT J:A=A0
 2410 FOR I0=0TO I
 2420 B=A/2:C0=A-2*INT B:A=INT B
 2430 NEXT I0
 2440 IF C0=0LET B0=A0:B1=A0+2^I
 2450 IF C0=1LET B1=A0:B0=A0-2^I
 2460 RETURN
 2599 REM --------------------
 2600 "MERK"L=J-3:IF S=0LET G(L,I)=0:RET
URN
 2610 IF S=1LET G(L,I)=2^I:RETURN
 2699 REM --------------------
 2700 "FERTIG"FOR L=0TO 4:P(L)=0:FOR I=0
TO 6
 2710 P(L)=P(L)+G(L,I)
 2720 NEXT I:NEXT L
 2730 CLS
 2740 FOR L=0TO 4:GPRINT P(L);:LPRINT P(
L);:NEXT L
 2745 TEXT :LF 2
 2750 POKE 256*HB+(C-128)*5,P(0),P(1),P(
2),P(3),P(4)
 2755 GOSUB 1420+10*C
 2760 RETURN
 2799 REM --------------------
 2800 "BIT"GCURSOR J:A0=POINT J:A=A0
 2810 FOR I0=0TO I
 2820 B=A/2:C0=A-2*INT B:A=INT B
 2830 NEXT I0
 2840 IF S=0THEN 2870
 2850 IF C0=1LET AX=A0:GOTO 2890
 2860 IF C0=0LET AX=A0+2^I:GOTO 2890
 2870 IF C0=0LET AX=A0:GOTO 2890
 2880 IF C0=1LET AX=A0-2^I
 2890 GPRINT AX:RETURN
 2899 REM --------------------
 2900 "NULL"FOR L=0TO 4:P(L)=0
 2910 FOR K=0TO 6:G(L,K)=0
 2920 NEXT K:NEXT L
 2930 CLS :RETURN
 2999 REM ********************
 3000 "TAST"H=HB*256:G=H-256:U=H+3*256:V
=H+4*256
 3010 POKE G+&72,&48,&FE,&4A,&80,&58,HB-
1,&5A,&80,&6A,&7F,&F5,&88,&03,&9A
 3015 CALL G+&72
 3017 RETURN
 3020 POKE G+&C8,&A0:RETURN
 3030 POKE G+&E0,&A1:RETURN
 3040 POKE G+&D8,&A2:RETURN
 3050 POKE G+&CD,&A3:RETURN
 3060 POKE G+&D5,&A4:RETURN
 3070 POKE G+&C5,&A5:RETURN
 3080 POKE G+&FD,&A6:RETURN
 3090 POKE G+&D6,&A7:RETURN
 3100 POKE G+&C6,&A8:RETURN
 3110 POKE G+&FE,&A9:RETURN
 3120 POKE G+&D7,&AA:RETURN
 3130 POKE G+&C7,&AB:RETURN
 3140 POKE G+&F4,&AC:RETURN
 3150 POKE G+&F9,&AD:RETURN
 3160 POKE G+&E1,&AE:RETURN
 3170 POKE G+&E4,&AF:RETURN
```

```
3180 POKE G+&E2,&B0:RETURN
3190 POKE G+&EC,&B1:RETURN
3200 POKE G+&FC,&B2:RETURN
3210 POKE G+&C4,&B3:RETURN
3220 POKE G+&DA,&B4:RETURN
3230 POKE G+&D4,&B5:RETURN
3240 POKE G+&DC,&B6:RETURN
3250 POKE G+&DE,&B7:RETURN
3260 POKE G+&D1,&B8:RETURN
3270 POKE G+&C1,&B9:RETURN
3280 POKE G+&DD,&BA:RETURN
3290 POKE G+&ED,&BB:RETURN
3300 POKE G+&F2,&BC:RETURN
3310 POKE G+&EA,&BD:RETURN
3320 POKE G+&CC,&BE:RETURN
3330 POKE G+&FA,&BF:RETURN
3340 POKE G+&D2,&C0:RETURN
3350 POKE G+&E9,&C1:RETURN
3360 POKE G+&CA,&C2:RETURN
3370 POKE G+&C9,&C3:RETURN
3380 POKE G+&C2,&C4:RETURN
3390 POKE G+&F1,&C5:RETURN
3400 POKE G+&B4,&C6:RETURN
3410 POKE G+&B9,&C7:RETURN
3420 POKE G+&A1,&C8:RETURN
3430 POKE G+&A4,&C9:RETURN
3440 POKE G+&A2,&CA:RETURN
3450 POKE G+&AC,&CB:RETURN
3460 POKE G+&BC,&CC:RETURN
3470 POKE G+&84,&CD:RETURN
3480 POKE G+&9A,&CE:RETURN
3490 POKE G+&94,&CF:RETURN
3500 POKE G+&9C,&D0:RETURN
3510 POKE G+&9E,&D1:RETURN
3520 POKE G+&91,&D2:RETURN
3530 POKE G+&81,&D3:RETURN
3540 POKE G+&9D,&D4:RETURN
3550 POKE G+&AD,&D5:RETURN
3560 POKE G+&B2,&D6:RETURN
3570 POKE G+&AA,&D7:RETURN
3580 POKE G+&8C,&D8:RETURN
3590 POKE G+&BA,&D9:RETURN
 3600 POKE G+&92,&DA:RETURN
 3610 POKE G+&A9,&DB:RETURN
 3620 POKE G+&8A,&DC:RETURN
 3630 POKE G+&89,&DD:RETURN
 3640 POKE G+&82,&DE:RETURN
 3650 POKE G+&B1,&DF:RETURN
 3660 RETURN
 4999 REM **********************
 5000 "LL"GOSUB "NEUADR"
 5010 H=HB*256:G=H-256:U=H+3*256:V=H+4*2
56
 5050 REM ** ADR. FUER DR.-TAB **
 5060 POKE ((HB+2)*256),HB+3,0
 5100 REM ** DR.-ADR. **
 5110 FOR I=UTO U+&7ESTEP 2
 5115 POKE I,HB+4:NEXT I
 5120 K=0:FOR I=U+&01TO U+&7FSTEP 2
 5125 POKE I,K:K=K+2:NEXT I
 5130 POKE U+&40,K(0),L(0),K(1),L(1),K(2
),L(2),K(3),L(3)
 5135 POKE U+&48,K(4),L(4),K(5),L(5),K(6
),L(6),K(7),L(7)
 5140 POKE U+&50,K(8),L(8),K(9),L(9),K(1
0),L(10),K(11),L(11)
 5145 POKE U+&58,K(12),L(12),K(13),L(13)
,K(14),L(14),K(15),L(15)
 5150 POKE U+&60,K(16),L(16),K(17),L(17)
,K(18),L(18),K(19),L(19)
 5155 POKE U+&68,K(20),L(20),K(21),L(21)
,K(22),L(22),K(23),L(23)
 5160 POKE U+&70,K(24),L(24),K(25),L(25)
,K(26),L(26),K(27),L(27)
 5165 POKE U+&78,K(28),L(28),K(29),L(29)
,K(30),L(30),K(31),L(31)
 5170 POKE U+&80,K(32),L(32),K(33),L(33)
,K(34),L(34),K(35),L(35)
 5175 POKE U+&88,K(36),L(36),K(37),L(37)
,K(38),L(38),K(39),L(39)
 5180 POKE U+&90,K(40),L(40),K(41),L(41)
,K(42),L(42),K(43),L(43)
 5185 POKE U+&98,K(44),L(44),K(45),L(45)
,K(46),L(46),K(47),L(47)
```

```
 5190 POKE U+&A0,K(48),L(48),K(49),L(49)
,K(50),L(50),K(51),L(51)
 5195 POKE U+&A8,K(52),L(52),K(53),L(53)
,K(54),L(54),K(55),L(55)
 5200 POKE U+&B0,K(56),L(56),K(57),L(57)
,K(58),L(58),K(59),L(59)
 5205 POKE U+&B8,K(60),L(60),K(61),L(61)
,K(62),L(62),K(63),L(63),&77,&D0
 5210 REM ** DR.-DARST. **
 5220 FOR I=VTO V+&3ESTEP 2
 5230 POKE I,&A0,0:NEXT I
 5250 RETURN
 5500 "NEUADR"
 5510 FOR I=0TO 63
 5520 U=AD(I):K(I)=INT (U/256):L(I)=U-25
6*K(I)
 5530 NEXT I:RETURN
 6000 "BUFF":X$=INKEY$ :IF X$<>""THEN 60
00
 6010 RETURN
 7000 "L"GOSUB "LL":BEEP 3:END
 9999 REM --------------------
10000 "LISTADR"WAIT 0:FOR I=0TO 63
10010 GOSUB 11000
10020 LPRINT "CHR$";160+I;"  ";HH;"  ";H
L
10030 NEXT I
10040 END
11000 H=AD(I):HH=INT (H/256):HL=H-256*HH
11010 PRINT "CHR$";160+I;"  ";HH;"  ";HL
11020 RETURN
```



Als **Beispiel** wählen wir (für einen Rechner mit 16-KB-Erweiterung) **NEW &1000** und geben als High-byte/LCD = 2 (bzw. &02) ein. Nach kurzer Dauer werden Sie aufgefordert, die **Code-Nummer** [CHR$ (160-223)...] einzugeben, die Ihr **erstes Zeichen** tragen soll. Einfaches <ENTER> übernimmt die voreingestellte Nummer 160= &A0, während 0<ENTER> aus dem Programm führt. Nun erscheint:

DISPLAY(1)/PLOT(2)

auf der Anzeige, d.h. Sie **können wählen**, ob Sie Display- oder Plotterzeichen kreieren wollen.

Nach 1<ENTER> können Sie die nun auf dem Display eingeblendete Code-Nummer durch einfaches <ENTER> entweder bestätigen oder eine andere Nummer angeben. Endlich taucht **links ein blinkender Punkt** auf, eingerahmt von zwei dicken Balken. Hier wird das **Displayzeichen entworfen.**

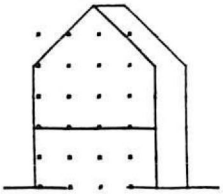
Der blinkende Punkt ist mit den **Cursortasten** (zwischen <SPACE> und <ENTER> und unterhalb der <MODE>-Taste) innerhalb einer **5*7-Zeichenmatrix** steuerbar, wobei jedesmal ein BEEP ertönt. Befinden Sie sich an einem **Rand**, ist stattdessen ein **kürzerer Ton** zu hören. Zu den Rändern gelangen Sie schneller mit den über bzw. links neben den Cursortasten liegenden Tasten <N>, <M>, <-> und <+>. Wollen Sie nun einen Punkt für Ihr zu entwerfendes Zeichen festlegen, drücken Sie bitte den Punkt <.>.

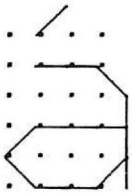
Soll stattdessen ein Punkt entfernt werden, drückt man Null <0>. Darüber hinaus besteht noch die Möglichkeit, durch </> gleich einen ganzen (vertikalen) Stich zu setzen. Mit <*> kann ein solcher entfernt werden.

Auf 3 Arten kann man den **Programmlauf** nun **fortsetzen.** <R> führt zur Programmmarke 'C' zurück. Mit <CL> löschen Sie das gegenwärtige Zeichen, soweit Sie es schon erstellt haben. Dann können Sie es gleich noch einmal versuchen. Gefällt Ihnen das Zeichen, drücken Sie nun <ENTER> und das Zeichen wird in der **Routine 'FERTIG'** in der LCD-Tabelle übernommen, um gleich zurückzukehren, wobei sich die **Code-Nummer** (für das nächste Zeichen) **um 1 erhöht.**

Anhand der **Marken** (Labels) im Programm ist gut zu verfolgen, welche Aufgaben die einzelnen Programmteile erledigen. Der **Display-Teil** beginnt bei der **Marke 'D'**, wonach gleich zur **Routine 'Tast'** verzweigt wird, wo in einem Maschinen-Pgm der Standardzeichensatz ins RAM (TAST-TAB) kopiert wird. Bei der **Marke 'USW.'**, zu der es bei jedem neuen Zeichen zurückgeht, erfolgt ein Sprung nach **'NULL'**. Dort wird das Feld P(L), das die 5 Spalten des Zeichens (wie (GPRINT) aufnimmt, vorab gelöscht. Bei der **Marke 'DISP'** erfolgt ein Sprung nach **'BLINK'**, wo die Werte für den blinkenden Punkt errechnet werden, in der Weise, daß das Zeichenmuster immer wieder hergestellt wird. In ähnlicher Weise arbeitet die **Routine 'BIT'**, die bei Setzen oder Löschen eines Punktes im Zeichenmuster angesteuert wird, wobei der Punkt in **'MERK'** festehalten wird. In der **Routine 'FERTIG'** werden schließlich die GPRINT-Muster erstellt und in der LCD-Tabelle abgelegt. Hier wird auch eine entsprechende Taste für das neuen Zeichen belegt (mit Hilfe einer berechneten Sprunganweisung).

Hat man anfangs (2) gewählt, kann man jetzt **Plotterzeichen entwerfen.** Dazu muß zunächst eine **Startadresse** (TAB-ADRESSE...) für die **Tabelle DRU-ZEI** angegeben werden, wo die Plotterzeichen abgelegt werden sollen. Einfaches <ENTER> übernimmt die **vorbelegte Adresse &600,** welches die früheste Adresse (bei HB=2) ist, die ohne Störungen möglich ist. Nun kommen wir zur **Marke 'NEXT',** zu der bei jedem neuen Zeichen zurückgeführt wird. Sofort wird nun ein **6*4-Raster** ausgedruckt, das ,wie im letzten Abschnitt erläutert, den Bereich darstellt, zu der der Plotterstift am Ende des Zeichens zurückgeführt werden muß, um Unregelmäßigkeiten des Plotters zu vermeiden.





```
 3 9 105 98 89 19
10 105 49 66 121 1
15 82 99 233

 á      CHR$  160
ADR.( 160)=  2014
```

Die Plotterzeichen werden nun **in starker Vergrößerung** entworfen, indem man je nach Richtung eine der **Zifferntasten** drückt, um sich jeweils 1 Schritt in einer der **8 Richtungen** zu bewegen. Dabei wird in der Variablen Z der Byte-Wert (für einen Strich) ständig erhöht (kumuliert). Die Taste <5> dient zum **Umschalten des Stiftes** in schreibender Position (PEN

DOWN) oder aufgehobener Position (PEN UP). Diese ganze Steuerung erfolgt in der **INKEY$-Schleife 'INK'.**

Bei **Richtungswechsel**, d.h. wenn Sie nicht die gleiche Taste drücken wie zuvor, wird die **Marke 'GRW'** angesprungen, was mit einem BEEP quittiert wird. Das geschieht, wenn Ungleichheit der INKEY-Variablen K$ mit der gemerkten Variablen M$ festgestellt wird. Man spricht in einem solchen Fall üblicherweise von **Gruppenwechsel.** Dieser wird auch nach <5> immer ausgelöst. Wesentlich bei der Routine 'GRW' ist nun, daß hier der kumulierte Byte-Wert Z von der Variablen H(I) übernommen und festgehalten wird. Hierin ist jetzt die ganze Information über Richtung, Länge usw. eines Striches (als Teil des Plotzeichens) enthalten. Im Normalfall springt die Programmsteuerung darauf zur **Marke 'REPEAT'**, um dort einen neuen Strich zu beginnen.

Wollen Sie zwischendurch **prüfen**, wie das Zeichen bislang aussieht, **drücken Sie <SPACE>** und dann zur Fortführung der Arbeit erneut <SPACE>. Ist das **Zeichen fertig** (und befinden Sie sich wieder im Raster), so **drücken Sie <ENTER>.** Darauf werden zunächst die **Byte-Werte,** die in der Tabelle DRU-ZEI abgelegt werden sollen, **ausgedruckt**, sowie das **'Probezeichen' geplottet** (CHR$ 223), das in der Variablen X$ abgelegt wird. Für diese Prozeduren benutzt das Programm die **Routinen 'ENDZEI', 'PROBE' und 'TAB'.** Wenn Sie nun die Code-Nummer bestätigen und bei 'ZEICHEN OK (J/N)?' J<ENTER> eingeben, kann das **neue Zeichen im Speicher** aufgenommen werden. Zum Beenden des Programmlaufs ist statt einer Ziffer <E> zu drücken.

3. Die Deutschen Sonderzeichen für den PC 1500(A)

Obwohl Sie nun in der Lage sind, alle gewünschten Zeichen selbst zu generieren, finden Sie hier noch ein **BASIC-Programm**, das Ihnen die **wichtigsten Sonderzeichen** wie Umlaute, ß, usw. auf den Reserveebenen I bis III zur Verfügung stellt. Es kann z.B **als Vorprogramm für unser Textprogramm** dienen und wird nach Erzeugung der **Maschinencode-Tabellen** für die Sonderzeichen gelöscht. Diese Tabellen sind nur 642 byte lang, was durch (scheinbare) **Überlappung der Einzeltabellen** ermöglicht wurde.

Das Programm fragt nach dem **Start-high-byte** der anzulegenden Tabellen und schreibt nach dessen Eingabe das Low-byte der Startadresse dazu. Sie geben daher nur zwei Ziffern ein. Tippen Sie also z.B. ein: 01<ENTER>, so erscheint auf dem Display als Startadresse für die Tabellen: &0180. Bei dieser Adresse beginnt an sich die **Tabelle mit den Tastaturcodes.** Folglich heißt das High-byte für die Startadresse der LCD-Tabelle &02 bzw. 2(dez), und diese Zahl wird einerseits der Variablen HB zugewiesen (hier: HB=2), andererseits der Systemzelle &785E (hier: POKE &785E, &02). Die beiden (in diesem Beispiel) letzten Adressen &400 und &401 zeigen auf den Start der Tabelle DRU-ADR. Daher müssen Sie **vor Laden des Programms** NEW &402 eingeben.

Sie können diese **Sonderzeichen** leicht **woanders plazieren,** beispielsweise in den Speicher ab &7C01 beim PC 1500A. Dazu geben Sie beim Start 7C ein. **Zwei Reserve-Tasten** sind mit den POKEs **für die Initialisierung** belegt, wie sie unter II.1 beschrieben wurde. Man darf hier auch die SML-Taste (!) benutzen, da die Zeichen auch dann weiter von Reserve aus abrufbar bleiben.

```
   10 "DEUZEI/R":REM V.2.2 *COPYRIGHT 19
85 Winfried Meyer, Hasloh, Am Barkenkamp
 18
   11 TEXT :CSIZE 1
   12 LPRINT "Starteingabe:H-BYTE(Start)
 ALS HEXA-DEZIMALZAHL ZWEISTELLIG EINGEB
EN."
   13 LPRINT "DAS PRORAMM BERECHNET DARA
US DAS  H-BYTE FUER DEN BEGINN DER LCD-T
ABELLE"
   14 LF 6
   15 "A"ST$="7C":CLS :WAIT 0:PRINT "STA
RT-ADR & ??";
   16 CURSOR 11:INPUT "";ST$:CURSOR 13:P
RINT "80":PAUSE
   17 IF ASC LEFT$ (ST$,1)>70OR ASC RIGH
T$ (ST$,1)>70OR LEN ST$<>2THEN 15
   18 VA=ASC LEFT$ (ST$,1):V0=VA:GOSUB 1
000:V1=V0*16
   19 VB=ASC RIGHT$ (ST$,1):V0=VB:GOSUB
1000:V2=V0:HB=V1+V2+1
   20 H=HB*256:G=(HB-1)*256:P=(HB+1)*256
   24 CSIZE 2:LPRINT "Startadresse=&";ST
$;"80"
   25 LPRINT "Endadr.=&(";ST$;"+3),01"
   26 LPRINT "(dezimal: ";(HB+2)*256+1;"
)"
   28 LF 1:LPRINT "HB=Starteingabe+1=";H
B;" (dezimal)":LF 3
   29 IF HB=125GOTO 50
   30 N$="N":CLS :WAIT 0:PRINT "mind. NE
W";(HB+2)*256+2;"eing.-J/N ":CURSOR 24:I
NPUT N$
   35 IF N$="J"OR N$="JA"GOTO 50
   40 CLS :PAUSE "NEW";(HB+2)*256+2;" ei
ngeben!":PAUSE :PAUSE
   45 END
   50 CLS :PRINT "EINEN MOMENT BITTE !"
   55 REM *TAB.1: TASTATUR
   60 POKE H+&80,&48,&FE,&4A,&80,&58,HB-
1,&5A,&80,&6A,&7F,&F5,&88,&03,&9A
   70 CALL H+&80
   80 REM *
   85 POKE G+&C8,&A0
   90 POKE G+&E0,&A1
   95 POKE G+&D8,&A2
  100 POKE G+&CD,&A3
  105 POKE G+&D5,&A4
  110 POKE G+&C5,&A5
  115 POKE G+&FD,&A6
  120 POKE G+&D6,&A7
  125 POKE G+&C6,&A8
  130 POKE G+&FE,&A9
  135 POKE G+&D7,&AA
  140 POKE G+&C7,&AB
  145 POKE G+&FF,&AC
  150 POKE G+&D0,&AD
  155 POKE G+&CF,&AE
  180 REM *TAB.2: LCD
  185 POKE H+&A0,&7F,&41,&5D,&41,&7F
  190 POKE H+&A5,&7F,&77,&7B,&41,&7F
  195 POKE H+&AA,&7F,&59,&4D,&51,&7F
  200 POKE H+&AF,&7F,&5D,&55,&41,&7F
  205 POKE H+&B4,&79,&14,&12,&14,&79
  210 POKE H+&B9,&3E,&41,&4B,&41,&3E
  215 POKE H+&BE,&3F,&40,&45,&40,&3F
  220 POKE H+&C3,&38,&45,&44,&3D,&40
  225 POKE H+&C8,&38,&45,&44,&45,&38
  230 POKE H+&CD,&3C,&41,&40,&21,&7C
  235 POKE H+&D2,&7E,&25,&25,&25,&1A
  240 POKE H+&D7,&60,&60,&60,&60,&60
  245 POKE H+&DC,&0A,&55,&55,&55,&28
  250 POKE H+&E1,0,0,&02,&05,0
  255 POKE H+&E6,&02,&05,0,&02,&05
  300 REM * ADRESSE FUER DRUCKERTAB.
  310 POKE ((HB+2)*256),HB,&00
  400 REM *TAB.3: DRUCKER (ADRESSEN)
  410 FOR I=HTO H+&7ESTEP 2
  415 POKE I,HB+1:NEXT I
  420 K=0:FOR I=H+&01TO H+&7FSTEP 2
  425 POKE I,K:K=K+2:NEXT I
  428 H1=HB+1
  430 POKE H+&40,&A0,0,&A0,0,&A0,0,&A0,0
,H1,&40,H1,&4F,H1,&62,H1,&71
```

```
  435 POKE H+&50,H1,&84,H1,&97,H1,&A7,H1
,&CB,H1,&BA,H1,&B2,H1,&CF
  500 REM *TAB.4: DRUCKER (DARSTELLUNG)
  502 FOR I=PTO P+&3ESTEP 2
  504 POKE I,&A0,0:NEXT I
  510 POKE P+&40,&54,&4A,&19,&51,&41,&69
,&02,&51,&41,&69,&29,&7A,&74,&12,&E4
  515 POKE P+&4F,&11,&54,&49,&42,&11,&51
,&41,&69,&22,&51,&41,&69,&31,&02,&79,&74
  516 POKE P+&5F,&69,&62,&D9
  520 POKE P+&62,&03,&62,&59,&55,&09,&49
,&61,&71,&02,&49,&61,&71,&39,&75,&E9
  525 POKE P+&71,&04,&59,&53,&62,&11,&51
,&41,&69,&02,&51,&41,&69,&21,&29,&69,&72
  526 POKE P+&81,&79,&41,&C9
  530 POKE P+&84,&11,&52,&49,&42,&11,&51
,&41,&69,&22,&51,&41,&69,&31,&02,&79,&72
  531 POKE P+&94,&69,&62,&D9
  535 POKE P+&97,&11,&53,&09,&49,&61,&71
,&02,&49,&61,&71,&39,&74,&12,&6A,&61,&D9
  540 POKE P+&A7,&55,&49,&42,&79,&69,&62
,&42,&79,&71,&69,&E2
  545 POKE P+&B2,&11,&0B,&61,&59,&49,&41
,&69,&F9
  550 POKE P+&BA,&11,&0C,&59,&62,&69,&79
,&42,&79,&69,&1A,&69,&79,&42,&79,&69,&62
,&D9
  555 POKE P+&CB,&29,&46,&1A,&C0
  560 POKE P+&CF,&14,&59,&49,&41,&69,&79
,&61,&03,&59,&49,&41,&69,&79,&E1
  570 POKE H+&90,&28,&43,&29,&57,&2E,&4D
,&65,&79,&65,&72
  600 POKE &764E,&44:POKE &785D,&00,HB
  900 REM * RESERVE-BELEGUNG
  901 POKE (256*(PEEK &7863)+&08),&20,&2
0,&A0,&20,&20,&20,&A1,&20,&20,&20,&A2,&2
0,&20
  902 POKE (256*(PEEK &7863)+&15),32,&A3
,32,32,32,&5D,&20,&20,&48,&42,&20,&20,&2
0
  910 POKE (256*(PEEK &7863)+&22),&20,&2
0,&A7,&20,&20,&20,&A8,&20,&20,&20,&A9,&2
0,&20
  920 POKE (256*(PEEK &7863)+&2F),&20,&A
B,&20,&20,&20,&7F,&20,&20,&20,&AA,&20,&2
0,&20
  930 POKE (256*(PEEK &7863)+&3C),&20,&2
0,&AD,&20,&20,&20,&AE,&20,&20,&20,&AC,&2
0
  940 POKE (256*(PEEK &7863)+&48),&20,&2
0,&A4,&20,&20,&20,&A5,&20,&20,&20,&A6,&2
0,&20,32
  943 POKE (256*(PEEK &7863)+&56),&01,&A
0,&02,&A1,&03,&A2,&04,&A3
  944 POKE (256*(PEEK &7863)+&5E),&05,&5
0,&4F,&4B,&45,&26,&37,&36,&34,&45
  945 POKE (256*(PEEK &7863)+&68),&2C,&2
6,&34,&34
  946 POKE (256*(PEEK &7863)+&6C),&06,&5
0,&4F,&4B,&45,&26,&37,&38,&35,&44,&2C,&3
0
  948 POKE (256*(PEEK &7863)+&78),&2C,&5
F
  950 POKE (256*(PEEK &7863)+&7A),&11,&A
7,&12,&A8,&13,&A9,&14,&AB,&15,&7F,&16,&A
A
  960 POKE (256*(PEEK &7863)+&86),&09,&A
D,&0A,&AE,&0B,&AC,&0C,&A4,&0D,&A5,&0E,&A
6,0
  998 WAIT :PRINT "Pgrm DEUZEI loeschen!
(NEW)"
  999 END
 1000 IF V0>47AND V0<58LET V0=V0-48:GOTO
 1030
 1010 IF V0>64AND V0<71LET V0=V0-55:GOTO
 1030
 1020 BEEP 1:GOTO 16
 1030 RETURN
```

4. Eine neue Schrift: SCHREIBSCHRIFT

Daß man mit einem Plotter nicht nur **Buchstaben** drucken, sondern **auch schreiben** kann, beweist das folgende Programm. Es wird hier als **Maschinencode-Listing** für den Speicherbereich &501 bis &DFF zunächst für den PC 1500(A) vorgestellt (doch siehe unter II.8). Unter Abänderung der die Tabellen aufrufenden Adressen innerhalb des Programms kann man diesen Maschinencode **in Schritten von 256 byte verschieben,** doch bereiten die Anpassungen mehr Mühe als bei dem BASIC-Programm im vorigen Abschnitt II.3. Das entsprechende BASIC-Programm ist mit 10 Kbyte zu lang zum Abdruck.

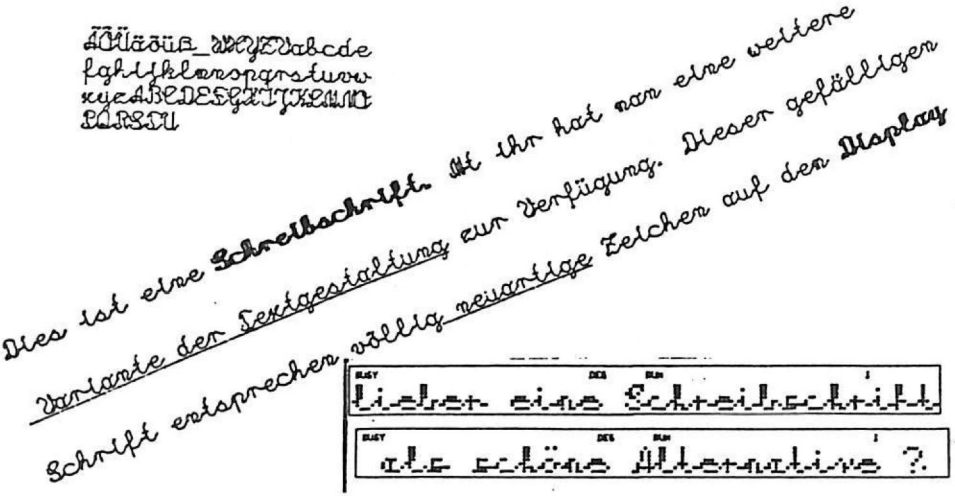
Man muß also NEW &E00 eingeben, wenn man sich vor **Überschreibungen** durch BASIC-Programme schützen will. Die Reserveebenen sind ähnlich wie bei dem vorigen Programm belegt, d.h. auch die Initialisierungen durch POKE &785D,0,6 und POKE &764E,&44 sind von Reserve aus abrufbar, wobei noch <ENTER> zu drücken ist. Aber die SML-Taste darf hier nicht gedrückt werden, da damit auf **Normal-Zeichensatz umgeschaltet** würde. Auf der Ebene III findet man neben den Zeichen M und D, die **für den Einsatz im Textprogramm** bestimmt sind, noch LPRINT. Das ist erforderlich, weil die das Wort 'LPRINT' bildenden Zeichen sonst nicht erreichbar sind.

Das Programm wird durch CLOAD M geladen und ist nach Initialisierung über Reserve sofort benutzbar. Mit den **Tasten-Grundfunktionen** erhält man **Kleinbuchstaben, Großbuchstaben** erreicht man über **<SHIFT><Taste>**. Die geplotteten Zeichen passen genau aneinander, so daß ein **geschlossenes Schriftbild** entsteht. Ein geschlossenes Schriftbild ist auch bei den **Display-Zeichen** möglich. Dazu ist lediglich der Zwischenraum zwischen 2 Zeichen mit GPRINT '20' zu überschreiben.

```
0008  20 20 A0 20 20 20 A1 20 20 20 A2 20   20 20 A3 20 20 20 5D 20 20 48 42 00
0020  00 00 20 20 A7 20 20 20 A8 20 20 20   A9 20 20 20 AB 20 20 20 7F 20 20 20
0038  AA 00 00 00 20 4C 50 2E 20 20 4D 20   20 20 4C 20 20 20 A4 20 20 20 A5 20
0050  20 20 A6 20 20 00 01 A0 02 A1 03 A2   04 A3 05 50 4F 4B 45 26 37 36 34 45
0068  2C 26 34 34 06 50 4F 4B 45 26 37 3B   35 44 2C 30 2C 5F 5F 11 A7 12 AB 13
0080  A9 14 AB 15 7F 16 AA 09 4C 50 2E 0A   4D 0B 4C 0C A4 0D A5 0E A6 00 00 00
0098  00 00 00 00 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00 00 00 00 00
00B0  00 00 00 00 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00 00 00

0500  06 00 06 0E 06 1A 06 29 06 3B 06 4B   06 56 06 69 06 77 06 86 06 97 06 A6
0518  06 B1 06 B9 06 C1 06 CC 06 DB 06 E4   06 F3 06 FC 07 09 07 19 07 25 07 32
0530  07 3E 07 4D 07 5C 07 67 07 72 07 80   07 89 07 95 07 9E 07 AD 07 B2 07 BF
0548  07 C9 07 D3 07 DC 07 EA 07 F5 08 02   08 06 08 0A 08 15 08 1F 08 2A 08 30
0560  08 35 08 58 08 7B 08 A8 08 AC 08 B2   08 B9 08 C5 08 CC 08 D1 08 DC 08 E4
0578  08 EC 08 F4 08 FA 09 02 0B BE C9 01   B8 38 35 32 09 C8 C7 11 C3 0F 2D 2E
0590  30 BD C5 15 BA 37 34 31 0D 28 B9 16   BB BF BC 29 19 B3 B5 12 B4 2F 2A 2B
05A8  20 C6 C2 13 B6 C0 08 3D 02 CA C1 1B   B1 18 1F 0C 0A B2 C4 14 B7 39 36 33

05C0  5B D8 AE 01 D2 A5 A8 AB A0 AD AC 21   DD A3 2C 2E 30 D7 DF 25 D4 A4 A7 AA
05D8  A2 3C D3 26 D5 D9 D6 3E A1 CD CF 22   CE 3F 3A 3B 5E B0 DC 23 D0 DA 1D 40
05F0  02 AF DB 1B CB 1A 1E 1C 5D CC DE 24   D1 A6 A9 33 31 79 41 49 59 41 49 14
0608  59 62 69 74 79 C1 01 41 4A 72 54 12   61 22 61 32 73 F9 03 09 69 62 59 52
0620  49 11 49 69 31 42 79 71 E4 03 62 59   49 43 1B 49 79 29 21 42 79 73 51 E9
0638  03 62 59 49 43 1C 41 02 41 2A 21 42   79 73 51 E9 03 62 59 49 43 1B 11 79
0650  29 42 79 73 51 E9 03 62 59 49 43 1B   11 49 41 79 69 61 59 32 42 79 73 51
0668  E9 31 79 41 49 59 41 49 12 59 62 69   72 79 C1 03 09 69 62 59 52 49 12 49

0680  48 FE 4A 80 58 05 5A 80 6A 7F F5 88   03 9A 02 41 28 43 29 57 2E 4D 65 79
0698  65 72 62 59 52 49 13 79 7F 41 5D 41   7F 7F 77 7B 41 7F 7F 59 4D 51 7F 7F
06B0  5D 55 41 7F 40 70 2D 24 7D 38 45 4C   7D 48 04 3D 40 3D 40 30 49 78 49 40
06C8  30 49 48 31 10 20 3A 40 3A 40 40 7E   05 27 1B 60 60 60 60 60 19 26 70 41
06E0  7F 41 22 1C 28 46 09 16 50 71 3F 41   69 5A 4E 23 19 26 40 21 1F 30 48 78
06F8  48 40 3F 4F 40 70 10 30 48 48 40 40   30 48 7F 40 40 30 58 58 58 50 7F 17
0710  10 20 20 10 58 58 50 28 10 7F 2F 60   40 10 3A 40 40 20 10 50 3A 20 20 10
0728  7F 1F 28 40 10 3F 4F 40 40 78 10 78   08 70 30 78 08 70 40 30 48 48 30 10

0740  20 78 18 18 20 18 78 10 10 10 10 78   10 10 20 40 78 68 68 20 10 3F 50 40
0758  20 20 38 40 38 40 10 08 70 20 18 38   40 20 48 30 10 50 20 30 58 40 58 30
0770  10 18 10 48 68 58 48 40 70 2E 21 7E   40 7F 45 45 3A 04 3E 49 45 43 41 3D
0788  41 42 3C 20 56 49 41 41 49 3D 11 12   12 04 1E 29 75 23 71 3F 08 7F 4F 41
07A0  41 42 22 1F 01 41 62 7F 21 41 61 1E   74 43 44 3E 29 45 63 5F 22 0E 01 7F
07B8  5F 22 0C 10 7F 3E 41 47 7E 44 41 3D   11 12 1C 3E 41 71 3E 10 41 7D 29 6A
07D0  44 24 4E 49 55 23 41 61 5D 42 22 01   3F 40 3F 40 11 09 69 79 41 49 59 54
07E8  29 51 49 42 79 71 69 61 79 72 29 51   49 23 C0 49 51 1A 12 71 31 71 61 69
```

```
0800    09 00 0B 01 64 F2 03 09 52 E4 11 4C   19 22 73 39 49 79 6A 02 E2 11 4C 19
0818    15 71 79 41 4B 59 62 69 71 7B 71 69   0A 01 6A 63 59 C9 04 5B 4B 21 6B FB
0830    4B 5B 01 7B EB 31 29 41 03 41 09 11   69 79 41 49 51 59 53 29 51 49 42 79
0848    41 34 6A E2 15 51 49 79 69 74 79 41   49 16 77 41 51 49 23 C0 15 51 49 7A
0860    71 6A 71 79 41 4A 54 59 72 79 41 32   6A E1 15 52 7A 71 6A 71 79 41 52 49
0878    52 75 41 51 49 54 59 72 79 41 32 6A   E1 15 52 7A 73 6A 52 4A 42 4A 51 61
0890    6A 74 79 4A 23 C0 15 51 49 7A 71 6A   71 79 41 4A 55 74 6A 79 72 69 61 51
08A8    4C 23 C0 15 51 49 79 41 49 42 69 61   6B 11 42 79 23 71 69 71 41 49 79 41

08C0    4A 23 C0 15 F5 39 31 55 62 44 35 D5   3A 56 63 32 12 4A 12 61 71 49 39 51
08D8    41 69 29 41 71 79 42 12 23 79 72 69   61 59 D2 16 71 49 61 02 71 49 61 2A
08F0    73 79 41 49 53 74 41 4A 23 C0 0A 01   21 63 12 43 0A 00 0A 02 0A 04 0A 06
0908    0A 08 0A 0A 0A 0C 0A 0E 0A 10 0A 12   0A 14 0A 16 0A 18 0A 1A 0A 1C 0A 1E
0920    0A 20 0A 22 0A 24 0A 26 0A 28 0A 2A   0A 2C 0A 2E 0A 30 0A 32 0A 34 0A 36
0938    0A 38 0A 3A 0A 3C 0A 3E A0 00 A0 00   A0 00 A0 00 0A 40 0A 59 0A 73 0A D9
0950    0B D0 0B E7 0A C2 0A FC 0B 6E 0B 85   0B 96 0B AB 0B 5C 0B 00 0B 0E 0B 1E
0968    0B 28 0B 39 0B 45 0B 53 0B 67 0B 75   0B 83 0B 96 0B B0 0B C0 0B D1 0B E0

09B0    0B F0 0C 00 0C 12 0C 20 0C 30 0C 40   0C 50 0C 5F 0C 70 0C B0 0C 91 0C A0
0998    0C B0 0C C4 0C D2 0C E3 0C F4 0D 20   0D 38 0D 0F 0D 4A 0D 5C 0D 77 0D 9B
09B0    0D B1 0D C1 0D D3 0D E8 07 E0 08 18   08 3A 08 4C 3A 56 54 2B 31 46 32 63
09C8    23 C3 12 19 43 15 F5 3A 55 03 E3 31   29 46 55 55 66 75 32 12 75 01 55 55
09E0    01 75 75 01 55 55 02 75 75 21 55 15   F5 31 29 46 55 66 75 01 55 01 75 01
09F8    55 02 75 21 D5 31 29 43 A0 00 A0 00   A0 00 A0 00 A0 00 A0 00 A0 00 A0 00
0A10    A0 00 A0 00 A0 00 A0 00 A0 00 A0 00   A0 00 A0 00 A0 00 A0 00 A0 00 A0 00
0A28    A0 00 A0 00 A0 00 A0 00 A0 00 A0 00   A0 00 A0 00 A0 00 A0 00 A0 00 A0 00

0A40    12 71 79 49 54 4A 12 61 71 49 39 51   41 69 29 77 12 59 62 69 79 44 49 23
0A58    C0 11 54 4A 12 61 71 49 39 51 41 69   29 41 72 79 41 49 33 6A 61 17 79 75
0A70    69 62 D9 15 51 49 12 71 49 61 02 71   49 61 2A 79 69 74 79 41 49 16 77 41
0A88    51 49 23 C0 51 14 51 64 7B EB 01 59   52 49 43 62 79 72 69 E1 31 29 49 55
0AA0    73 79 41 49 D3 13 49 43 62 F4 09 59   52 49 42 79 72 69 62 15 42 61 76 61
0AB8    C2 02 59 54 49 79 72 62 42 72 55 49   42 79 69 62 42 79 71 69 42 49 22 29
0AD0    E2 11 0B 61 59 49 41 69 F9 03 41 4A   1A 69 59 61 69 72 79 41 49 52 12 51
0AE8    41 69 22 51 41 69 3A F3 79 71 69 61   59 51 49 C1 03 61 59 54 29 46 1A C0

0B00    03 41 4A 1A 69 59 61 69 72 79 41 49   52 F3 12 4A 53 61 69 75 79 41 49 52
0B18    61 7A 41 49 23 C0 11 52 49 42 79 31   02 6A 63 D9 11 52 49 41 79 2A 21 79
0B30    41 49 55 49 33 3A 6A 61 D3 12 43 49   51 62 69 72 79 42 4A 23 C0 12 41 4A
0B48    53 61 69 75 74 14 79 42 4A 23 C0 11   52 49 41 79 49 29 72 69 61 59 03 79
0B60    72 69 61 51 4C 25 C0 12 4A 53 61 69   76 51 4A 41 73 41 4A 23 C0 12 41 49
0B78    12 49 61 71 31 73 79 41 4A 23 C0 12   41 49 12 49 61 71 31 73 79 72 69 61
0B90    51 4B 41 49 23 C0 56 49 41 73 6A 71   4B 79 69 61 79 71 21 42 4A 23 C0 2B
0BA8    20 DB D7 13 CB D5 08 3D 12 41 4A 53   61 69 75 79 42 4A 23 C0 CC 39 36 33
```

```
0BC0   12 51 49 74 53 49 74 53 49 79 71 69   71 41 4A 23 C0 12 51 49 79 73 53 49
0BD8   79 71 69 71 41 4A 23 C0 12 4A 41 71   79 42 12 23 79 72 69 61 59 D2 1D 40
0BF0   12 51 49 77 56 49 41 79 71 69 71 41   4A 23 C0 33 0C 69 72 69 61 59 52 49
0C08   41 79 76 21 12 49 41 4A 23 C0 12 51   49 41 74 53 49 7A 41 23 C0 C1 46 A0
0C20   12 4A 71 79 71 69 61 59 06 11 6A E2   34 88 20 28 12 41 49 19 42 13 69 75
0C38   79 41 4A 23 C0 08 80 32 14 73 79 41   49 53 74 41 4A 23 C0 00 C4 04 51 81
0C50   12 51 49 41 72 69 79 41 49 53 61 7A   41 23 C0 12 51 49 72 69 79 49 51 72
0C68   41 49 53 61 7A 41 23 C0 12 51 49 74   61 4C 61 69 72 79 41 4A 23 C0 0E 04

0C80   12 51 49 72 69 79 41 49 79 72 69 61   51 4C 1B 31 F3 12 51 49 43 69 61 6A
0C98   71 41 49 79 41 4A 23 C0 12 71 79 49   54 4A 77 12 59 62 69 79 44 49 23 C0
0CB0   12 71 79 49 55 29 51 49 42 79 71 69   61 41 79 72 49 6A D9 0C 15 71 79 41
0CC8   4B 59 62 69 75 79 42 4A 23 C0 41 49   55 29 51 49 42 79 74 69 71 4A 6A 61
0CE0   5A 69 F1 12 0A 42 09 51 59 62 69 71   79 6A 71 79 43 4A 23 C0 02 49 51 59
0CF8   53 33 02 59 62 11 51 49 42 79 41 34   6A 63 59 C9 30 30 49 4A 48 30 38 15
0D10   51 49 79 41 49 41 69 75 69 61 59 51   05 6A E1 44 15 71 79 41 4B 59 62 69
0D28   75 79 41 49 52 71 69 79 72 69 61 51   4C 23 C0 09 15 51 49 79 49 75 6A 59

0D40   4D 51 61 75 69 79 4A 23 C0 1A 15 51   49 79 41 49 41 6A 74 79 72 69 61 51
0D58   4C 23 C0 44 15 51 49 79 49 41 69 74   6A 59 51 31 0E 69 71 69 61 79 41 71
0D70   69 71 4A 23 C0 61 5E 01 15 71 79 41   4B 59 62 69 75 69 61 51 49 7A 41 4A
0D88   23 C0 41 7C 02 02 02 7C 2A 2A 2A 2A   2A 44 44 5F 44 44 40 12 71 79 49 59
0DA0   53 4A 77 51 49 53 49 51 75 69 71 41   51 49 23 C0 2A 12 71 79 49 54 4A 77
0DB8   4A 15 71 69 73 69 29 C0 00 11 54 4A   41 72 79 41 49 33 6A 61 17 79 75 69
0DD0   62 D9 19 16 02 74 79 69 61 59 49 13   51 49 42 79 71 69 61 3A 01 6A E2 88
0DE8   01 59 54 4A 41 79 75 69 62 09 61 51   41 7A 4A 23 C0 50 48 40 85 00 01 E0
```



5. 39 Zeichen in einer Zeile: MICRO-SCHRIFT

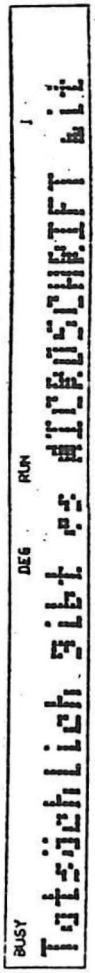
Benutzer des PC 1500(A) können auf dem Display immerhin 26 Zeichen darstellen, bei vielen anderen 'Pockets' sind es weniger. Noch etwas mehr Überblick bekommt man durch **Erweiterung der Anzeige um 50% auf 39 Zeichen**. Dies geht freilich nur softwaremäßig durch **schmalere Display-Zeichen**, die statt in einer 5*7-Matrix in einer 3*6-Matrix Platz finden. Erstaunlich ist die **gute Lesbarkeit** unserer MICRO-Zeichen.

Die **MICRO-Zeichen** werden erneut durch ein BASIC-Programm als '2. Zeichensatz' generiert, sind daher an (fast) beliebiger Adresse zu installieren, wobei man wie in II.3 vorgeht. Die **Initialisierung** geschieht wieder mit POKE &785D,0, HB und POKE &764E, &44 <ENTER>. Dies kann **hier nicht von Reserve** abgerufen werden, wie ebenfalls nicht die Umlaute. Die **Umlaute** Ä,ö, ü erhält man als **SHIFT-Funktion der Tasten 7,8,9**, dann die kleinen Umlaute als SHIFT-Funktionen der **Tasten 4,5, 6** und das ß als SHIFT-Funktion der Taste mit der **Ziffer 1**. Die Grundfunktionen der alphabetischen Tasten liefern **Großbuchstaben**, die SHIFT-Funktionen erzeugen **Kleinbuchstaben**. Die SML-Taste schaltet auf Normal-Zeichensatz um.

Der **Druck auf dem Display** muß nun so organisiert werden, daß der GCURSOR, der mit PEEK &7875 abfragbar ist, immer nur um 4 Positionen weitergesetzt wird nach Ausgabe von 1 Zeichen. Dies besorgt eine kleine **Maschinenroutine** in einem **speziellen Editor** für die Eingabe der MICRO-Zeichen. Diesen Editor finden Sie im Kapitel über die Texteingabe unter Punkt 4.

Die 3 letzten Zeichensätze wurden ohne den Zeichengenerator entwickelt, im Gegensatz zum folgenden IBM-Zeichensatz. In Kapitel XII wollen wir noch **Schönschriftzeichen** kreieren.

```
   10 "MICRO-SCHRIFT":REM *COPYRIGHT Win
fried Meyer, Hasloh
   11 REM *JAHR:1985*
   12 CLEAR :WAIT 0
   15 INPUT "LCD-HIGH-BYTE (&xx): ";HB:L
PRINT "LCD-HB(SCHREIB)=";HB
   20 H=HB*256:G=H-256:P=H+256:U=(HB+3)*
256:V=U+256:W=V+256:X=W+256:Y=X+256:Z=Y+
256
   30 N$="":PRINT "Bereits NEW)=";(HB+8)
*256;" eing.?":CURSOR 24:INPUT N$
   40 IF N$<>"J"END
   53 CLS :PRINT "EINEN MOMENT BITTE !"
   55 REM *TAST
   60 POKE H+&80,&48,&FE,&4A,&80,&58,HB-
1,&5A,&80,&6A,&7F,&F5,&88,&03,&9A
   70 CALL H+&80
  105 POKE G+&D5,&A0
  110 POKE G+&C5,&A1
  115 POKE G+&FD,&A2
  135 POKE G+&D6,&A3
  140 POKE G+&C6,&A4
  145 POKE G+&FE,&A5
  150 POKE G+&D7,&A6
  155 POKE G+&EF,&AA
  161 POKE G+&F4,&CB
  162 POKE G+&F9,&CC
  163 POKE G+&E1,&CD
  164 POKE G+&E4,&CE
  165 POKE G+&E2,&CF
  166 POKE G+&EC,&D0
  167 POKE G+&FC,&D1
  168 POKE G+&C4,&D2
  169 POKE G+&DA,&D3
  170 POKE G+&D4,&D4
  171 POKE G+&DC,&D5
  172 POKE G+&DE,&D6
  173 POKE G+&D1,&D7
  174 POKE G+&C1,&D8
  175 POKE G+&DD,&D9
  176 POKE G+&ED,&DA
  177 POKE G+&F2,&DB
  178 POKE G+&EA,&DC
  179 POKE G+&CC,&DD
  180 POKE G+&FA,&DE
  181 POKE G+&D2,&DF
  182 POKE G+&E9,&AB
  183 POKE G+&CA,&AC
  184 POKE G+&C9,&AD
  185 POKE G+&C2,&AE
  186 POKE G+&F1,&AF
  187 POKE G+&B4,&B1
  188 POKE G+&B9,&B2
  189 POKE G+&A1,&B3
  190 POKE G+&A4,&B4
  191 POKE G+&A2,&B5
  192 POKE G+&AC,&B6
  193 POKE G+&BC,&B7
  194 POKE G+&84,&B8
  195 POKE G+&9A,&B9
  196 POKE G+&94,&BA
  197 POKE G+&9C,&BB
  198 POKE G+&9E,&BC
  199 POKE G+&91,&BD
  200 POKE G+&81,&BE
  201 POKE G+&9D,&BF
  202 POKE G+&AD,&C0
  203 POKE G+&B2,&C1
  204 POKE G+&AA,&C2
  205 POKE G+&8C,&C3
  206 POKE G+&BA,&C4
  207 POKE G+&92,&C5
  208 POKE G+&A9,&C6
  209 POKE G+&8A,&C7
  210 POKE G+&89,&C8
  211 POKE G+&82,&C9
  212 POKE G+&B1,&CA
  216 REM *LCD
  217 POKE H+&A0,&7D,&14,&7D,0,0
  218 POKE H+&A5,&7D,&44,&7D,0,0
  219 POKE H+&AA,&7D,&40,&7D,0,0
  235 POKE H+&AF,&32,&48,&7A,0,0
  240 POKE H+&B4,&32,&48,&32,0,0
  245 POKE H+&B9,&7A,&40,&7A,0,0
  250 POKE H+&BE,&7C,&2A,&16,0,0
  260 POKE H+&D2,&63,&14,&63,&14,&63
  270 REM
  271 POKE H+&F5,&7C,&12,&7C,0,0
  272 POKE H+&FA,&7E,&5A,&6C,0,0
  273 POKE H+&FF,&3C,&42,&46,0,0
  274 POKE P+&04,&7E,&62,&7C,0,0
  275 POKE P+&09,&7E,&4A,&42,0,0
  276 POKE P+&0E,&7E,&0A,&02,0,0
  277 POKE P+&13,&7C,&52,&76,0,0
  278 POKE P+&18,&7E,&10,&7E,0,0
  279 POKE P+&1D,&42,&7E,&42,&0,0
  280 POKE P+&22,&62,&42,&3E,0,0
  281 POKE P+&27,&7E,&18,&66,0,0
  282 POKE P+&2C,&7E,&40,&40,&0,0
  283 POKE P+&31,&78,&3C,&7E,0,0
  284 POKE P+&36,&7C,&08,&3E,0,0
  285 POKE P+&3B,&7C,&42,&3E,0,0
  286 POKE P+&40,&7E,&12,&1C,0,0
  287 POKE P+&45,&3C,&32,&7E,0,0
  288 POKE P+&4A,&7E,&3A,&4C,0,0
  289 POKE P+&4F,&4C,&4A,&32,0,0
  290 POKE P+&54,&02,&7E,&02,&0,0
  291 POKE P+&59,&7E,&40,&7E,&0,0
  292 POKE P+&5E,&3E,&40,&3E,0,0
```



```
DEMO-PROGRAMM: PGM
-EINGABE MIT MICRO
-SCHRIFT (vorher i
nstall.!) IM RUN-M ODE
10:REM A*****1234
   56789@12345678
   90123456789*12
   34567890123456
   789*1234567890
   12345678
11:REM A*****1234
   56789@12345678
   90123456789*12
   34567890123456
   789*1234567890
   12345678
12:REM A*****1234
   56789@12345678
   90123456789*12
   34567890123456
   789*1234567890
   12345678
13:REM A*****1234
   56789@12345678
   90123456789*12
   34567890123456
   789*1234567890
   12345678
14:REM A*****1234
   56789@12345678
   90123456789*12
   34567890123456
   789*1234567890
   12345678
15:REM A*****1234
   56789@12345678
   90123456789*12
   34567890123456
   789*1234567890
   12345678
16:REM A*****1234
   56789@12345678
   90123456789*12
   34567890123456
   789*1234567890
   12345678
```

```
60000:"B"H=0:ZN=0:
      S=STATUS 2-
      STATUS 1
60001:WAIT 0
60002:H=H+1:IF H=1
      1END
60005:ZN=ZN+10:E$=
      "":CURSOR 0:
      PRINT ZN;
60010:INPUT E$:CLS
60012:IF E$=""BEEP
      5:END
60015:POKE &7BB0,0
      ,ZN
60020:FOR I=2TO 15
60025:POKE (&7BB0+
      I),(PEEK (&7
      650+I-2))
60030:NEXT I
60040:CALL &F957
60050:FOR J=2TO 15
60055:Z=PEEK (&7BB
      0+J)
60060:IF Z=0THEN 6
      0080
60070:NEXT J
60080:POKE S+2,J-1
      :POKE (S+J+1
      ),13,0,(ZN+1
      ),(75-J),241
      ,171
60085:POKE (S+79),
      13
60090:POKE S,0,ZN
60100:FOR I=3TO J
60110:POKE S+I,
      PEEK (&7BB0+
      I-1)
60120:NEXT I
60200:REM FOR I=0
      TO 15
60210:REM LPRINT
      PEEK (S+I);
60220:REM NEXT I
60230:LLIST ZN:LF
      -4
60240:S=S+80:GOTO
      60002
```

6. IBM-ZEICHENSATZ

Wer einen **umfangreichen Zeichensatz** sucht mit zusätzlichen **Grafiksymbolen,** stößt schnell auf den vom IBM-PC in Verbindung mit seinem Grafik-Drucker (Blockgrafik) verwendeten **internationalen Zeichensatz.** Der PC 1600 hat diesen Zeichensatz eingebaut und ist somit **IBM-datenkompatibel,** jedenfalls soweit es die benutzten Zeichen betrifft. Die dem Verfasser bekannten anderen SHARP-Pocketcomputer haben einen solchen Satz nicht, jedoch kann man sich die Plot-Zeichen mit unserem Zeichengenerator selbst schaffen, beim PC 1500(A) auch die Display-Zeichen.

In diesem Abschnitt findet der Leser den **kompletten IBM-Zeichensatz** ab Code-Nr. &80 für den PC 1500(A). Er wurde **erstellt mit dem Zeichengenerator,** der wegen der großen Anzahl der Zeichen allerdings leicht abgewandelt werden mußte. Die Plot-Zeichen sind auch für andere Rechner nutzbar, wie in Kap. II.8 und besonders in Kap. XII gezeigt wird. Die **Zeichen des 1. Zeichensatzes** (mit Code-Nr. kleiner &80) beim PC 1500(A) und anderen Rechnern sind auf dem **7-Bit-ASCII-Code** aufgebaut und weichen von den entsprechenden IBM-Zeichen nur bei wenigen Codes ab. Es soll dem Leser überlassen bleiben, wie er dieses Problemchen lösen will, sollten ihm die Zeichen wichtig sein.

Ein **anderes Problem** wurde auf die folgende Art gelöst: Zwar kann man beim PC 1500(A) bis zu **128 neue Zeichen** kreieren, aber es dürfen nur die Zeichen mit den Code-Nummern &A0 bis &DF über die Tastatur angesprochen werden, weil sonst der BASIC-Interpreter dieses Gerätes **Fehlbelegungen** produziert.

In der hier präsentierten Maschinencode-Gesamttabelle sind daher **2 verschiedene Tasta-**

```
0180    0B D3 DE 01 CD 38 35 32 09 DD DC 11   D8 0F 2D 2E 30 D2 DA 15 CF 37 34 31
0198    0D 28 CE 16 D0 D4 D1 29 19 C8 CA 12   C9 2F 2A 2B 20 DB D7 13 CB D5 08 3D
01B0    02 DF D6 1B C6 18 1F 0C 0A C7 D9 14   CC 39 36 33 5B B9 C4 01 B3 A5 A8 AB
01C8    A0 C3 C2 21 BE A3 2C 2E 30 B8 C0 25   B5 A4 A7 AA A2 3C B4 26 B6 BA B7 3E
01E0    A1 AE B0 22 AF 3F 3A 3B 5E C1 BD 23   B1 BB 1D 40 02 C5 BC 1B AC 1A 1E 1C
01F8    5D AD BF 24 B2 A6 A9 33 0E 51 31 11   0A 38 41 40 21 78 38 54 56 55 18 30
0210    4A 49 3A 40 30 49 48 39 40 30 49 4A   38 40 30 48 49 38 40 0C 52 32 12 04
0228    38 56 55 56 18 38 55 54 55 18 38 55   56 54 18 00 01 78 01 00 00 02 79 02

0240    00 00 01 7A 00 00 78 15 14 15 78 78   14 15 14 78 7C 54 56 55 44 32 4A 3C
0258    52 5C 7C 12 7F 49 49 30 4A 49 4A 30   30 49 48 49 30 30 49 4A 48 30 38 42
0270    41 22 78 38 41 42 20 78 0C 51 50 51   3C 38 45 44 45 3B 3C 41 40 41 3C 1C
0288    22 7F 22 24 48 7E 49 41 42 15 16 7C   16 15 7F 09 06 7F 44 20 48 3E 09 02
02A0    30 48 4A 39 40 00 00 7A 01 00 30 48   4A 49 30 38 40 42 21 78 7A 11 09 0A
02B8    71 7E 09 11 22 7D 26 29 29 27 28 26   29 29 29 26 30 48 45 40 20 38 08 08
02D0    08 08 08 08 08 08 38 17 08 04 6A 59   17 08 34 2A 7D 00 20 5D 20 00 08 14
02E8    22 49 14 14 49 22 14 08 AA 00 55 AA   00 AA 55 AA 55 AA 55 FF AA 55 FF 00

0300    00 00 FF 00 10 10 10 FF 00 14 14 14   FF 00 10 10 FF 00 FF 10 10 F0 10 F0
0318    14 14 14 FC 00 14 14 F7 00 FF 00 00   FF 00 FF 14 14 F4 04 FC 14 14 17 10
0330    1F 04 04 07 04 07 14 14 14 1F 00 10   10 10 F0 00 00 00 00 1F 10 10 10 10
0348    1F 10 10 10 10 F0 10 00 00 00 FF 10   10 10 10 10 10 10 10 10 FF 10 00 00
0360    00 FF 14 00 00 FF 00 FF 00 00 1F 10   17 00 00 FC 04 F4 14 14 17 10 17 14
0378    14 F4 04 F4 00 00 FF 00 F7 14 14 14   14 14 14 14 F7 00 F7 14 14 14 17 14
0390    10 10 1F 10 1F 14 14 14 F4 14 10 10   F0 10 F0 00 00 1F 10 1F 00 00 00 1F
03A8    14 00 00 00 FC 14 00 00 F0 10 F0 00   00 FF 00 FF 14 14 14 FF 14 10 10 10

03C0    1F 00 00 00 00 F0 10 FF FF FF FF FF   F0 F0 F0 F0 F0 FF FF FF 00 00 00 00
03D8    00 FF FF 0F 0F 0F 0F 0F 38 44 44 38   44 7E 11 25 25 1A 7F 01 01 01 03 44
03F0    3C 04 3C 44 63 55 49 41 63 38 44 44   3C 04 7C 20 05 00 20 08 04 7C 04 02
0408    08 55 77 55 08 3E 49 49 49 3E 5E 61   01 61 5E 30 4A 4D 49 30 18 24 18 24
0420    18 5C 32 2A 26 1D 1C 2A 49 49 41 7C   02 02 02 7C 2A 2A 2A 2A 2A 44 44 5F
0438    44 44 40 51 4A 44 40 40 44 4A 51 40   00 00 7E 01 02 20 40 3F 00 00 08 08
0450    2A 08 08 24 12 24 48 24 00 02 05 02   00 00 0C 0C 00 00 00 08 08 00 00 10
0468    20 7F 01 01 0F 02 01 0E 00 12 19 15   12 00 00 1C 1C 1C 00 1F 00 00 04 00
```

```
0500  06 00 06 0E 06 1A 06 29 06 3B 06 48  06 56 06 69 06 77 06 86 06 97 06 A6
0518  06 B1 06 B9 06 C1 06 CC 06 DB 06 E4  06 F3 06 FC 07 09 07 19 07 25 07 32
0530  07 3E 07 4D 07 5C 07 67 07 72 07 80  07 89 07 95 07 9E 07 AD 07 B2 07 BF
0548  07 C9 07 D3 07 DC 07 EA 07 F5 08 02  08 06 08 0A 08 15 08 1F 08 2A 08 30
0560  08 35 08 5B 08 7B 08 A8 08 AC 08 B2  08 B9 08 C5 08 CC 08 D1 08 DC 08 E4
0578  08 EC 08 F4 08 FA 09 02 09 06 09 0C  09 12 09 17 09 1D 09 21 09 28 09 2F
0590  09 39 09 41 09 49 09 53 09 5D 09 68  09 6F 09 7D 09 85 09 BC 09 93 09 9B
05A8  09 A2 09 A8 09 AD 09 B4 09 C0 09 CA  09 CF 09 D3 09 ED 09 FD 0A 0C 0A 1A

05C0  0A 6B 0A 72 0A 7F 0A B4 0A 8B 0A 92  0A 9C 0A A5 0A AA 0A B9 0A C3 0A D2
05D8  0A E1 0A EC 0A F8 0B 00 0B 05 0B 0B  0B 10 0B 15 0B 1B 0B 20 0B 25 0B 2F
05F0  0B 3B 0B 43 0B 4C 0B 4F 0B 56 0B 5D  0B 63 A0 00 31 79 41 49 59 41 49 14
0608  59 62 69 74 79 C1 01 41 4A 72 54 12  61 22 61 32 73 F9 03 09 69 62 59 52
0620  49 11 49 69 31 42 79 71 E4 03 62 59  49 43 1B 49 79 29 21 42 79 73 51 E9
0638  03 62 59 49 43 1C 41 02 41 2A 21 42  79 73 51 E9 03 62 59 49 43 1B 11 79
0650  29 42 79 73 51 E9 03 62 59 49 43 1B  11 49 41 79 69 61 59 32 42 79 73 51
0668  E9 31 79 41 49 59 41 49 12 59 62 69  72 79 C1 03 09 69 62 59 52 49 12 49

0680  79 2A 42 79 71 E4 03 09 69 62 59 52  13 41 02 41 2B 21 49 42 79 71 E4 03
0698  09 69 62 59 52 49 13 79 29 31 42 79  71 E4 01 42 09 15 61 22 61 21 3A 41
06B0  F4 01 42 16 59 69 32 41 F4 01 42 16  21 59 33 41 F4 54 4A 1A 41 02 41 2A
06C8  7A 74 52 E4 54 4A 19 59 49 41 79 69  61 29 02 7A 74 52 E4 04 64 56 44 1A
06E0  69 3A 32 E3 14 41 79 72 69 59 51 44  51 59 69 72 79 41 E1 54 4A 42 62 76
06F8  42 1B 21 C3 01 59 52 49 12 49 79 2A  42 79 72 69 E2 01 59 52 49 11 19 41
0710  02 41 2A 21 42 79 72 69 E2 01 59 52  49 42 1A 79 39 79 72 69 E2 02 61 59
0728  53 11 09 49 79 39 31 74 52 EA 02 61  59 53 11 09 79 39 01 74 52 EA 02 01

0740  62 59 53 12 41 02 41 32 75 69 62 0B  E9 01 59 54 49 19 41 03 61 29 21 42
0758  79 74 69 E2 03 62 59 55 11 41 03 61  39 75 E9 02 0A 69 62 59 52 49 42 79
0770  1A F6 03 09 69 61 59 61 71 41 55 49  79 2A 21 C2 12 44 12 64 12 7A 4A 6A
0788  F4 56 42 79 71 69 62 3B 01 61 53 39  E2 11 79 49 54 49 79 31 29 E2 03 09
07A0  69 62 59 13 0A 69 31 42 79 73 52 63  E9 13 0B 69 31 F4 03 49 52 59 62 11
07B8  09 69 31 69 72 79 C2 02 61 59 53 09  49 3A 74 52 EA 54 09 49 79 49 36 53
07D0  59 61 EA 56 09 49 79 49 32 76 51 DC  14 49 43 71 69 62 59 09 11 42 79 72
07E8  31 E4 0C 69 62 59 51 49 42 79 71 32  E4 01 42 49 51 1A 12 71 31 71 61 69

0800  71 F9 0B 01 64 F2 03 09 52 E4 11 4C  19 22 73 39 49 79 6A 02 E2 11 4C 19
0818  22 73 3A 01 63 4A F3 03 53 59 11 49  62 79 31 69 73 C2 04 5B 4B 21 6B FB
0830  4B 5B 01 7B EB 31 29 41 03 41 12 61  23 61 12 41 03 41 12 61 23 61 12 41
0848  03 41 12 61 23 61 3B 61 12 41 34 61  34 41 12 E1 31 29 42 12 62 12 42 12
0860  62 12 42 1A 42 31 42 11 42 32 62 32  42 32 62 32 42 32 62 11 62 12 42 12
0878  62 12 C2 31 29 43 01 42 19 63 21 61  11 43 01 42 19 63 21 61 11 43 01 42
0890  19 63 21 61 11 43 01 42 14 62 21 63  31 41 01 43 39 62 21 63 31 41 04 E3
08A8  3A 55 15 F5 3A 55 63 08 12 F5 3A 54  63 12 43 14 F6 39 31 55 62 13 0A 75
```

```
08C0   02 35 55 15 F5 39 31 55 62 44 35 D5   3A 56 63 32 C3 39 31 54 62 12 42 54
08D8   02 75 35 D5 39 31 55 55 02 75 35 D5   39 31 54 62 12 44 36 D6 0A 01 64 12
08F0   42 54 02 F6 0B 55 22 75 22 C4 0A 01   21 63 12 43 14 F6 3A 55 23 C3 0A 11
0908   55 3B 32 E3 0B 02 66 0B 12 F5 3A 55   63 06 E3 3A 55 55 3B 32 E3 0B 42 26
0920   C4 3A 55 23 46 1B 12 F5 3A 54 43 12   63 14 F6 39 31 55 55 02 75 75 0A 13
0938   E2 09 11 56 02 74 42 32 E4 39 31 56   44 32 62 34 D4 11 19 46 12 62 54 22
0950   74 22 C2 39 31 54 62 12 46 32 62 34   D4 39 31 55 55 02 74 42 32 62 34 D4
0968   11 19 46 12 62 24 C4 39 31 54 62 12   42 54 02 74 42 32 62 34 D4 11 19 46

0980   12 66 0B 11 F4 12 19 46 24 55 02 F5   3A 54 63 12 46 32 E3 31 39 55 22 46
0998   2A 33 D5 12 09 55 02 75 02 E4 0A 43   12 63 14 F6 3A 56 43 32 E3 39 31 55
09B0   44 2A 33 D5 39 31 55 15 75 22 46 2A   33 55 15 F5 3A 56 54 2B 31 46 32 63
09C8   23 C3 12 19 43 15 F5 3A 55 03 E3 31   29 46 55 55 66 75 32 12 75 01 55 55
09E0   01 75 75 01 55 55 02 75 75 21 55 15   F5 31 29 46 55 66 75 01 55 01 75 01
09F8   55 02 75 21 D5 31 29 43 55 55 63 75   75 01 55 55 01 75 35 D5 3A 43 55 55
0A10   63 75 75 02 55 55 21 75 35 D5 12 19   46 55 66 75 01 15 75 01 55 01 75 02
0A28   55 21 F5 01 59 52 49 7B 12 EB 54 4A   41 79 71 69 62 42 79 71 69 61 D9 0C

0A40   11 51 64 F6 49 53 61 44 34 59 D3 44   51 14 51 64 7B EB 01 59 52 49 43 62
0A58   79 72 69 E1 31 29 49 55 73 79 41 49   D3 13 49 43 62 F4 09 01 59 52 49 7B
0A70   12 EB 54 4A 41 79 71 69 62 42 79 71   69 61 D9 0C 11 51 64 F6 49 53 61 44
0A88   34 59 D3 44 51 14 51 64 7B EB 01 59   52 49 43 62 79 72 69 E1 31 29 49 55
0AA0   73 79 41 49 D3 13 49 43 62 F4 09 59   52 49 42 79 72 69 62 15 42 61 76 61
0AB8   C2 02 59 54 49 79 72 62 42 72 E9 11   71 41 52 59 52 49 42 79 72 69 72 09
0AD0   71 E1 01 59 52 49 41 09 59 61 69 79   41 79 72 69 E1 13 49 41 7A 41 49 59
0AE8   61 6A 61 D9 21 4D 29 21 79 71 69 61   59 51 49 C1 03 61 59 54 49 41 33 E2

0B00   55 49 41 79 F5 11 43 12 63 12 C3 44   1C 44 1A F4 03 63 09 4A DA 02 15 6A
0B18   7A 39 E3 31 39 56 49 F9 09 79 49 56   F6 09 01 14 61 29
```

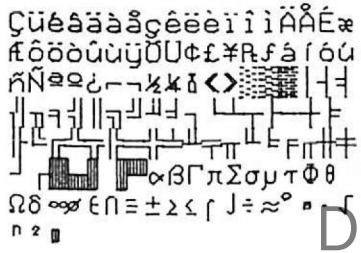
```
 10 REM -- 1. TEIL
 20 INPUT "H1B= ";H1B
 30 X=0:FOR I=0TO 19STEP 2:GOSUB 90:NEXT I
 40 X=1:FOR I=20TO 40STEP 2:GOSUB 90:NEXT I
 50 X=2:FOR I=41TO 62STEP 2:GOSUB 90:NEXT I
 60 X=3:FOR I=63TO 93STEP 2:GOSUB 90:NEXT I
 70 X=4:FOR I=94TO 110STEP 2:GOSUB 90:NEXT I
 80 X=5:FOR I=111TO 126STEP 2:GOSUB 90:NEXT
I
 85 GOTO 120
 90 POKE (H1B+3)*256+2*I,H1B+4+X:RETURN
100 REM ----------
110 REM -- 2. TEIL
120 H2B=H1B+10
130 X=0:FOR I=32TO 51STEP 2:GOSUB 190:NEXT I
140 X=1:FOR I=52TO 63STEP 2:GOSUB 190:NEXT I
150 X=4:FOR I=64TO 78STEP 2:GOSUB 190:NEXT I
160 X=5:FOR I=79TO 94STEP 2:GOSUB 190:NEXT I
170 GOTO 210
190 POKE (H2B+3)*256+2*I,H2B+4+X:RETURN
200 REM -- INITILISIERUNG
210 POKE &785D,0,H1B:POKE &764E,&44
220 END
300 FOR J=&80TO &FF:LPRINT CHR$ J;:NEXT J
400 FOR K=0TO &FF:LPRINT PEEK (&F00+K);" ";:
    NEXT K
```

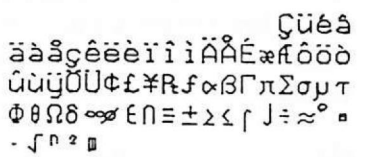
```
0BB0    0B D3 DE 01 CD 38 35 32 09 DD DC 11   DB 0F 2D 2E 30 D2 DA 15 CF 37 34 31
0B98    0D 28 CE 16 D0 D4 D1 29 19 C8 CA 12   C9 2F 2A 2B 20 DB D7 13 CB D5 08 3D
0BB0    02 DF D6 1B C6 18 1F 0C 0A C7 D9 14   CC 39 36 33 5B B9 C4 01 B3 A5 A8 AB
0BC8    A0 C3 C2 21 BE A3 2C 2E 30 B8 C0 25   B5 A4 A7 AA A2 3C B4 26 B6 BA B7 3E
0BE0    A1 AE B0 22 AF 3F 3A 3B 5E C1 BD 23   B1 BB 1D 40 02 C5 BC 1B AC 1A 1E 1C
0BF8    5D AD BF 24 B2 A6 A9 33

0CA0    0E 51 31 11 0A 38 41 40 21 78 38 54   56 55 18 30 4A 49 3A 40 30 49 48 39
0CB8    40 30 49 4A 38 40 30 48 49 38 40 0C   52 32 12 04 38 56 55 56 18 38 55 54
0CD0    55 18 38 55 56 54 18 00 01 78 01 00   00 02 79 02 00 00 01 7A 00 00 78 15
0CE8    14 15 78 78 14 15 14 78 7C 54 56 55   44 32 4A 3C 52 5C 7C 12 7F 49 49 30
0D00    4A 49 4A 30 30 49 48 49 30 30 49 4A   48 30 38 42 41 22 78 38 41 42 20 78
0D18    0C 51 50 51 3C 38 45 44 45 38 3C 41   40 41 3C 1C 22 7F 22 24 48 7E 49 41
0D30    42 15 16 7C 16 15 7F 09 06 7F 44 20   48 3E 09 02 38 44 44 38 44 7E 11 25
0D48    25 1A 7F 01 01 01 03 44 3C 04 3C 44   63 55 49 41 63 38 44 44 3C 04 7C 20

0D60    20 1C 20 08 04 7C 04 02 08 55 77 55   08 3E 49 49 49 3E 5E 61 01 61 5E 30
0D78    4A 4D 49 30 18 24 18 24 18 5C 32 2A   26 1D 1C 2A 49 49 41 7C 02 02 02 7C
0D90    2A 2A 2A 2A 2A 44 44 5F 44 44 40 51   4A 44 40 40 44 4A 51 40 00 00 7E 01
0DA8    02 20 40 3F 00 00 08 08 2A 08 08 24   12 24 48 24 00 02 05 02 00 00 0C 0C
0DC0    00 00 00 0B 08 00 00 10 20 7F 01 01   0F 02 01 0E 00 12 19 15 12 00 00 1C
0DD8    1C 1C 00 1F 00 0E 71 01

0E00    0F 00

0F00    A0 00 A0 00 A0 00 A0 00 A0 00 A0 00   A0 00 A0 00 A0 00 A0 00 A0 00 A0 00
0F18    A0 00 A0 00 A0 00 A0 00 A0 00 A0 00   A0 00 A0 00 A0 00 A0 00 A0 00 A0 00
0F30    A0 00 A0 00 A0 00 A0 00 A0 00 A0 00   A0 00 A0 00 06 00 06 0E 06 1A 06 29
0F48    06 38 06 48 06 56 06 69 06 77 06 86   06 97 06 A6 06 B1 06 B9 06 C1 06 CC
0F60    06 DB 06 E4 06 F3 06 FC 07 09 07 19   07 25 07 32 07 3E 07 4D 07 5C 07 67
0F78    07 72 07 80 07 89 07 95 0A 6B 0A 72   0A 7F 0A 84 0A 8B 0A 92 0A 9C 0A A5
0F90    0A AA 0A B9 0A C3 0A D2 0A E1 0A EC   0A F8 0B 00 0B 05 0B 0B 0B 10 0B 15
0FA8    0B 1B 0B 20 0B 25 0B 2F 0B 3B 0B 43   0B 4C 0B 4F 0B 56 0B 5D 0B 63 A0 00

0FC0    A0 00 A0 00 A0 00 A0 00 A0 00 A0 00   A0 00 A0 00 A0 00 A0 00 A0 00 A0 00
0FD8    A0 00 A0 00 A0 00 A0 00 A0 00 A0 00   A0 00 A0 00 A0 00 A0 00 A0 00 A0 00
0FF0    A0 00 A0 00 A0 00 A0 00 A0 00 A0 00   A0 00 A0 00
```

turbelegungen integriert. **Die 1.** gilt für die Codes &A0 bis &DF, wie beim Original-IBM-Satz, und wird mit **POKE &785D,0,2** <ENTER> initiiert. Die **2. Tastaturbelegung** (nicht zu verwechseln mit dem '2. Zeichensatz') wird mit **POKE &785D,0,12** <ENTER> initiiert. Sie gilt für die IBM-Codes &80 bis &9F und &E0 bis &FF, die hier aber ebenfalls auf die Codes &A0 bis &DF gelegt wird. **Sofern man** bei der 1. Belegung darauf **verzichtet**, die Zeichen über die Tastatur zu erreichen, kann man hier sogar alle Zeichen hervorbringen. So geht z.B. sowohl PRINT CHR$ &80 als auch LPRINT CHR$ &80.

Die **Gesamttabelle** geht von &180 bis &FFF. Vor dem Einladen ist also **NEW &1000** einzugeben. Sie ist verschiebbar in Schritten von &FF=256 Bytes. Für den Fall, daß ein anderer Speicherbereich gewählt werden soll oder muß, kommt das beigefügte **Verschiebeprogramm** zur Anwendung. Zu merken ist, daß das High-byte des Starts der LCD-Tabelle der **2. Belegung immer um 10 höher** als dasjenige der 1. Belegung gewählt werden muß, d.h. POKE &785E,H2B = POKE &785E,(H1B + 10).

Nicht vergessen werden darf zur Initialisierung der Tastatur **POKE &764E,&44**. Damit wird freilich der **1. Zeichensatz ausgeblendet**, und Sie können z.B. kein BASIC-Programm schreiben. Was tun? - Eine Möglichkeit wäre, wie wir es beim Programm DEUZEI (Kap. II.3) taten, die wichtigsten **Sonderzeichen auf die Reservetasten** zu legen. Man kann dann <SML> drücken und hat diese Zeichen jetzt trotzdem direkt auf Tastendruck zur Verfügung.





7. Display-Sonderzeichen für den PC 1350/2500

Eine dem PC 1500(A) entsprechende Möglichkeit, Sonderzeichen als '2. Zeichensatz' auf dem **Display des PC 1350/2500** darzustellen, ist dem Verfasser nicht bekannt. Man kann allerdings den Zeichen entsprechende **Bitbilder im RAM** speichern und diese dann auf eine gewünschte Stelle in die Anzeige bringen, doch leider sind die so erzeugten Bitmuster nicht den Zeichencodes direkt zuzuordnen. Beim PC 1350 bzw. 2500 liefern die Codes ab &80 **japanische Zeichen**, die der Leser wohl kaum benötigt.

In unserem **Textprogramm für den PC 1350/2500** in Kap. XI.2 erzeugen wir **Sonderzeichen** mittels der **GPRINT-Funktion**. In einer speziellen Leseroutine werden bestimmte **Tastenkombinationen mit dem Doppelpunkt**, die bei der Texteingabe für die Umlaute benutzt werden, **zu echten Umlauten umgewandelt**, aus 'a:' wird also z.B 'ä'. Die Kombination '@:' dient als Platzhalter für das 'β', das dann ebenso auf dem Display erscheint.

8. Die neuen Zeichen auf diversen Plottern

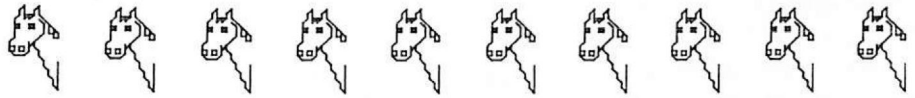
Die **Anpassung des PLot-Teils** unseres Zeichengenerators **an den PC 1350/2500** und einem zugehörigen Plotter ist lohnenswert, kann aber aus Platzgründen hier nicht vorgenommen werden. Als Muster können aber die entsprechenden **Versionen von 'ZPLOT'** aus dem Schlußkapitel (auch für den **MZ 700/800**) dienen, die für diverse Plotter ganz analog ausgeführt sind. Neben der **SCHREIBSCHRIFT** ist auch noch die **ANTIQA-Schrift** aus diesem Kapitel für diese und andere Rechner/Plotterkombinationen geeignet.

Es wird nämlich gezeigt werden, daß Plot-

Zeichen, die eigentlich als '2. Zeichensatz' für den PC 1500(A) gedacht waren, von allen übrigen Plottern ebenfalls 'lesbar' sind. Mehr noch: Man die **Zeichen beliebig drehen** und wenden unter x-beliebigem Winkel und so regelrechte **Kunstschriften** zustande bringen.

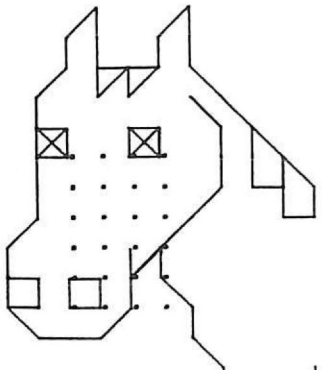
9. Spezialeffekte für Display und Plotter

Sie haben jetzt auf jeden Fall eine **Menge neuer Zeichen** zur Verfügung, schon wenn Sie nur auf die vollständigen in diesem Buch abgedruckten Zeichensätze zurückgreifen. Die Zeichen selbst können zusätzlich manipuliert werden, so daß Spezialeffekte entstehen. Dazu gehört auch die **Negativ- (Invers-) Darstellung** von Schrift, wie wir dies in unserem im nächsten Kapitel beginnenden Textverarbeitungs-Programm benutzen. Hier nun eine **Display-Kursivschrift** (eine Plot-Kursivschrift findet man in Kap. XII.1). Als BASIC-Schrift ist sie ziemlich langsam, aber **sehr effektvoll**! Die Routine ab Zeile 500 ist der von 'ZPLOT' (Kap. XII.2) ähnlich (dort ab Zeile 100) und könnte durch ein Maschinenprogramm ersetzt werden.



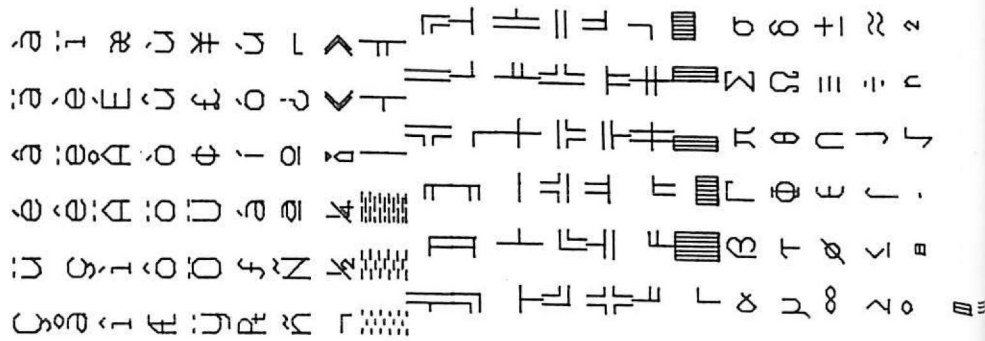
Als einen **speziellen Nebeneffekt** der Zeichengenerierung mit dem Plotter erhält man die Möglichkeit, auch kleine **Zeichnungen** zu erstellen, Figuren o.ä. Diese sind dann gleich in einer 'Bibliothek' gespeichert und beliebig abrufbar und vergrößerbar. Das **kleine Pferdchen** hat meine Tochter entworfen - viel mehr aus Tierliebe denn aus Computerbegeisterung!

```
 10 REM -- KURSIVSCHRIFT/DISPLAY
 15 REM (C) 1987 by Winfried Meyer, Hasloh
 20 "A"CLEAR :DIM T$(0)*24,A(6,151)
 30 INPUT "):";T$(0):L=LEN T$(0)
 40 WAIT 0:PRINT T$(0)
 50 FOR K=0TO L*6
 60 J=K+7:GOSUB 500
 70 NEXT K:CLS
200 FOR J=7TO L*J+8
210 G=A(6,J)+A(5,J-1)+A(4,J-2)+A(3,J-3)+A(
2,J-4)+A(1,J-5)+A(0,J-6):GCURSOR J-7:GPRINT
G
220 NEXT J
300 K$=INKEY$ :IF K$=""GOTO 300
310 GOTO 20
500 A=POINT K:B=A/2:C=A-2*INT B:A(0,J)=C
510 A=INT B:B=A/2:C=A-2*INT B:A(1,J)=C*2
520 A=INT B:B=A/2:C=A-2*INT B:A(2,J)=C*4
530 A=INT B:B=A/2:C=A-2*INT B:A(3,J)=C*8
540 A=INT B:B=A/2:C=A-2*INT B:A(4,J)=C*16
550 A=INT B:B=A/2:C=A-2*INT B:A(5,J)=C*32
560 A=INT B:B=A/2:C=A-2*INT B:A(6,J)=C*64
570 RETURN
```



```
 34 121 66 73 82 4
1 49 81 97 113 65
35 81 65 113 97 82
 73 83 65 113 97 7
3 2 65 113 97 81 1
21 81 105 34 89 81
 73 81 73 115 73 1
13 73 98 66 81 73
114 124 113 97 82
113 97 82 33 89 12
1 114 107 73 113 1
21 113 121 113 121
 113 121 113 121 8
6 28 33 233
```

ABCDEFGHIJKLMNOPQ
RSTUVWXYZ



```
BUSY                    DEG    RUN                    ]
neunhunddreißig Zeichen auf der Display.
```

Kap. III: PLANUNG UND DOKUMENTATION

1. Formales: Modul und Struktur

Um ein **Programm übersichtlich** erscheinen zu lassen, gibt es verschiedene Möglichkeiten. Zunächst kann man **Kommentare** bzw. **REM-Zeilen** einstreuen, was vor allem dem Programmlisting zugute kommt. Sodann soll das Programm möglichst **inhaltlich strukturiert** sein. Anhänger der **Strukturierten Programmierung** haben es mit Pocket-Computern allerdings schwer, da das **SHARP-BASIC** diese Methode kaum unterstützt. Beim **PC 1600** fiindet man wenigstens ein **IF... THEN...ELSE** vor. Beide Arten der Programmgestaltung verlangen zudem viel Speicherplatz, mit dem man bei den kleinen PC's bekanntlich haushalten muß. (Sie können bei der Eingabe der Programme die REM-Zeilen einfach weglassen)

Eine weitere Möglichkeit besteht darin, das Programm in möglichst unabhängige Teile zu zergliedern, **Module** genannt. Hiervon wird im vorliegenden Fall reichlich Gebrauch gemacht. Unterstützt wird dies auf der **Benutzeroberfläche,** d.h. bei dem was auf dem Display sichtbar wird, durch ein **Menü,** das die **Programmsteuerung** erleichtert (siehe nächstes Kapitel). Auch hier weichen wir von der strengen Form ab, indem auch Querverbindungen zwischen den Moduln zugelassen werden.

2. Die Leistungsmerkmale des Textsystems

Durch die **modulare Struktur** unseres Textsystems können wir je nach vorhandenem Speicherplatz das Programm ausbauen. Grundlegend sind **Hauptmenü, Texteingabe** und **Textausgabe** (Druck). Sodann können **Editieren** (nachträgliches Verändern des Textes)mit unterschied-

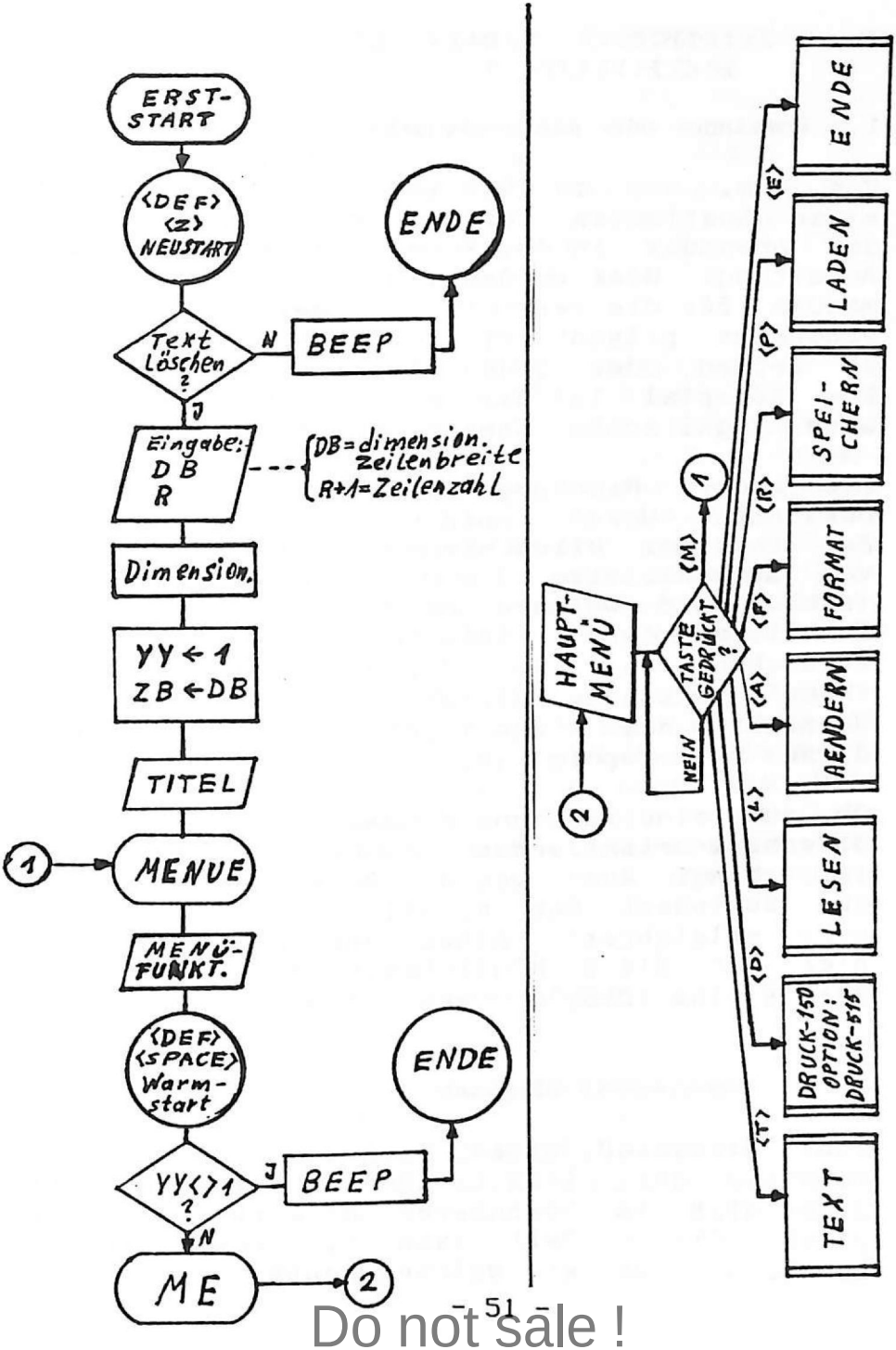
lichem Komfort und **Formatieren** zugefügt werden.

Das Formatieren stellt ein entscheidendes Hilfsmittel dar, einen bearbeiteten Text wieder in eine **kompakte Form** zu bringen. Schließlich sollte man noch eine Möglichkeit vorsehen, seine Texte programmgesteuert auf ein **externes Speichermedium** (Kassette u.a.) abzuspeichern bzw. es von dort zu laden. Fehlt der Speicherplatz, kann dies auch manuell geschehen. Als besonderen Komfort werden hier sowohl eine **Laufschrift** als auch eine **Fließschrift** auf dem Display vorgestellt.

Auch das **Menüsystem,** ob für 1-Zeilen- oder 4-Zeilendisplay, das wir im nächsten Kapitel behandeln, wird man als komfortabel bezeichnen können. Im weiteren Verlauf werden wir das **komplette Listing eines Textsystems für den PC 1500(A)** modul-weise wiedergeben und kommentieren. Es wird fast unverändert auch auf dem **PC 1600 im MODE 1** laufen und ist richtungsweisend für **Anpassungen an andere SHARP-Rechner.** Auf einige Besonderheiten dieser Geräte, z.B. **Druckeranpassungen** an einige DIN-A4-Plotter, gehen wir gesondert ein. Weiter hinten findet sich dann noch ein **komplettes Listing** für den Rechner **PC 1350 (2500)** nebst Kommentierung.

3. Programmablaufplan

Nachstehend folgt für das Hauptprogramm ein **Programmablaufplan** (Grobplan), der unter Verzicht auf allzu genaue Details dem besseren Überblick dienen soll. Dieser stellt den zeitlichen Verlauf der vom Programm abgearbeiteten Instruktionen dar, unter Berücksichtigung der Einwirkungsmöglichkeiten des Anwenders. Die (zeitlich gemeinte) Hauptrichtung geht von oben nach unten.


ERST-START
<DEF> <Z> NEUSTART
ENDE
Text Löschen ?
N
BEEP
J
Eingabe: DB R
DB=dimension. zeilenbreite
R+1=Zeilenzahl
Dimension.
YY ← 1
ZB ← DB
TITEL
1
MENUE
MENÜ-FUNKT.
<DEF> <SPACE> Warm-start
ENDE
YY<>1 ?
J
BEEP
N
ME
2
2
HAUPT-MENÜ
TASTE GEDRÜCKT ?
NEIN
<M>
1
<E>
ENDE
<P>
LADEN
<R>
SPEI-CHERN
<F>
FORMAT
<A>
AENDERN
<L>
LESEN
<D>
DRUCK-150 OPTION: DRUCK-515
<T>
TEXT

Kap. IV: MENÜ UND BENUTZER-FÜHRUNG

1. Kommando- oder menügesteuert?

Von dem, was zu tun ist, um ein Programm in einen bestimmten Verlauf zu zwingen, erfährt der Anwender in der Regel aus der beigefügten Anleitung. Hier werden ihm eine Reihe von Kommandos für die verschiedenen **Betriebsarten** des Programms präsentiert, die entweder auswendig zu lernen oder jedesmal nachzuschlagen sind. Ein Beispiel ist das berühmte **'Wordstar'**, das eine regelrechte **Kommandosprache** bietet.

Die **andere Richtung** wird von dem fast ebenso bekannten **'Word'** repräsentiert. Dabei werden dem Anwender **bildschirm-orientiert** eine Reihe von Auswahllisten, die **Menüs**, angeboten, wobei (evtl. über weitere Menüs) die gewünschten Funktionen durch einfache Tastendrucke zu erreichen sind bzw. durch Einsatz einer **'Maus'**. Optische Hilfen wie z.B. **Fenster** und Verzicht auf Steuerzeichen im Text runden diese Philosophie ab.

Ob nun eine **logische Kommandosprache** oder ein **bidschirm-orientiertes Menüsystem** effektiver ist, hängt auch von der Person des Anwenders ab. Zumindest der Einstieg wird durch Menüs sehr erleichtert. Daher entscheiden wir uns hier für die 2. Möglichkeit trotz aller durch das kleine Display gegebenen Beschränkungen.

2. Menütechnik allgemein

Vom **Geamtmenü** eines Programms bekommt man meistens nur einen kleinen Teil zu sehen, und dies gilt im besonderen Maße für Pocketcomputer. Dieser Teil kann als **'Menübild'** bezeichnet werden. Welches Menübild man gerade

vor sich hat, hängt von der **Menüstruktur** ab.

Eine einfache Möglichkeit ist die **Ringstruktur**. Dabei erscheinen nacheinander alle Menübilder, etwa nach Drücken von <ENTER> (bzw. <CR> beim MZ 700/800). Eine besondere Möglichkeit hat man bei SHARP-Pocketcomputern durch die **'definable keys'**, mit denen man beliebig durch <DEF><Taste> zu einem Menübild springen kann.

Eine **hierarchische Ordnung** verkörpert die **Baumstruktur**, bei denen die Menübilder auf unterschiedlichem Niveau liegen. Will man von einem Bild zu einem beliebigen anderen gelangen, muß man sich bei der reinen Form zuerst zu der gemeinsamen 'Astgabel' begeben. Bei einer **tiefen Gliederung** kann dieses Verfahren sehr ermüdend sein, daher führt man oft 'Seitensprünge' ein, allerdings wird man dabei, wie so oft, leicht die Orientierung verlieren.

Neben dem **Hautmenü** mit den von ihm abgeleiteten **Untermenüs** findet man bisweilen noch ein **Hilfsmenü**, das dazu dient, dem Nutzer die Programmfunktionen zu erläutern oder ihm Ratschläge zu geben, wie er mit dem Programm weiter kommt, wenn er sich festgefahren hat. Bei Pocketcomputern können dazu z.B. die **Reservetasten** dienen.

3. Display-Fenster und andere spezielle Techniken

Die moderne Computer-Software kennt eine Reihe von **Hilfsmitteln** zur effektvollen Menü-Gestaltung wie Kopfleisten, Fußleisten, verschiedene Schriftarten, Inversschrift, Windows, Pull-down-Menüs etc. Auch bei den Pockets braucht man nicht auf solche Techniken zu verzichten, selbst wenn man nur 1 Display-Zeile zur Verfü-

gung hat (siehe nächster Punkt). Hier zeigen wir an einem **Beispiel für das 4-Zeilen-Display** (PC 1600/1300/2500) eine Kombination einiger Techniken.

Nach Start des Programms durch <DEF><A> erscheint eine **Kopfzeile in Inversschrift** mit folgender Aufschrift:

HELP HAUPTM FORMAT ENDE

Danach erscheint ein Fragezeichen, und nun kann man das Display mit einem Text füllen. Bei einfachem <ENTER> wird ein vorbereiteter Text eingeblendet. **Alle Funktionen** werden jetzt ausschließlich **mit den Cursortasten** sowie mit <SPACE> und <ENTER> ausgelöst. Die Steuerung ist also fast ähnlich wie mit dem bekannten **Eingabegerät 'Maus'**, das ja nur 2 Tasten besitzt. Ähnlich fungieren hier <SPACE> und <ENTER>.

Mit den Cursortasten unter <MODE> wird jetzt ein **feiner Strich** bewegt, der jeweils einen Buchstaben der Kopfzeile unterstreicht. (Schauen Sie genau hin!) Je nachdem, unter welchem Wort sich der Strich-Cursor gerade befindet, bewirkt nun <ENTER> das zugehörige **Pulldown-Menü**, mit dem der Text dort überdeckt wird. Befanden Sie sich z.B. unter dem Buchstaben 'U', lesen Sie in dem **Fenster** untereinanderstehend jetzt :

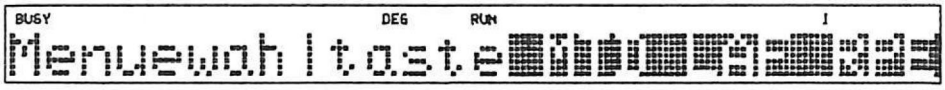
TEXT
LESEN
DRUCK
AEND.

Da aber nicht alle Funktionen des Hauptmenüs (HAUPTM) in dieses Fenster passen, kann man seinen **Inhalt skrollen** mit der (zwischen <SPACE> und <ENTER> liegenden) **Cursortaste 'Pfeil nach unten'.** Dann werden weitere Funktionen engeblendet. Dabei bleibt Ihr Text bis auf die Fen-

```
100:REM  PULLDOWN-MENUE "PULLDN-2"
110:"A"CLEAR :CLS :DIM A$(8)*79,W$(9)*5,SP$(8)*20
120:ST=1:W$(9)="-----":SP$(8)="                    "
130:ON ERROR GOTO 140
140:CLS :PRINT " HELP  HAUPTM  FORMAT ENDE":LINE (8,8)-(155,6),X,,BF
150:A$(8)="DER NEUE SHARP PC 1600 HAT SEHR GUTE MOEGLICHKEITEN ZUM EDITIEREN
    *****"
160:IF ST=1CURSOR 8,1:INPUT A$(8):ST=8:GOSUB 520
170:CURSOR 8,1:PRINT A$(8)
180:GOSUB 520
190:GOTO 200
200:CURSOR K,8:PRINT CHR$ &7F;W$(I);" ";CHR$ &7F
210:CURSOR K,1:PRINT CHR$ &7F;W$(I+1);" ";CHR$ &7F
220:CURSOR K,2:PRINT CHR$ &7F;W$(I+2);" ";CHR$ &7F
230:CURSOR K,3:PRINT CHR$ &7F;W$(I+3);" ";CHR$ &7F
240:IF I=8LET I=-1:BEEP 1,20,50
250:REM ------------
260:K$=INKEY$ :IF K$=""GOTO 260
270:IF S=1GOTO 340
280:IF K$=CHR$ &8CAND C<25LET C=C+1:FOR J=C*6TO (C+1)*6:PRESET (J-6,7):PSET
    (J,7):NEXT J:GOTO 260
290:IF K$=CHR$ &88AND C>8LET C=C-1:FOR J=C*6TO (C+1)*6:PRESET (J+6,7):PSET (
    J,7):NEXT J:GOTO 260
300:IF K$=CHR$ 13AND C>21CLS :BEEP 2,99,99:CURSOR 8,1:PRINT "E N D E":END
310:IF K$=CHR$ 13AND C>14LET UX=2:GOSUB 700:RESTORE 650:GOSUB 660:I=8:GOSUB
    520:K=13:S=1:GOTO 200
320:IF K$=CHR$ 13AND C>6LET UX=1:GOSUB 700:RESTORE 640:GOSUB 660:I=8:GOSUB 5
    20:K=5:S=1:GOTO 200
330:IF K$=CHR$ 13LET UX=4:GOSUB 700:RESTORE 630:GOSUB 660:I=8:GOSUB 520:K=8:
    S=1:GOTO 200
340:IF K$=CHR$ &8AAND UX>8GOSUB 520:I=I+1:GOTO 200
350:IF K$=" "CLS :UX=8:C=8:ST=8:S=8:GOTO 140
360:IF C>13AND C<20GOTO 450
370:IF C<6OR C>18GOTO 200
380:IF K$=CHR$ 13AND I=8GOSUB 520:GOSUB 540
390:IF K$=CHR$ 13AND I=1GOSUB 520:GOSUB 550
400:IF K$=CHR$ 13AND I=2GOSUB 520:GOSUB 560
410:IF K$=CHR$ 13AND I=3GOSUB 520:GOSUB 570
420:IF K$=CHR$ 13AND I=4GOSUB 520:GOSUB 580
430:IF K$=CHR$ 13AND I=5GOSUB 520:GOSUB 590
440:GOTO 200
450:IF K$=CHR$ 13AND I=8GOSUB 520:GOSUB 600
460:IF K$=CHR$ 13AND I=1GOSUB 520:GOSUB 610
470:IF K$=CHR$ 13AND I=2GOSUB 520:GOSUB 620
480:S=8:GOTO 200
490:REM ------------
500:J$=INKEY$ :IF J$=""GOTO 500
510:RETURN
520:J$=INKEY$ :IF J$<>""GOTO 520
530:RETURN
540:F=1:GOSUB 690:GOSUB 500:S=8:RETURN
550:F=2:GOSUB 690:GOSUB 500:S=8:RETURN
560:F=3:GOSUB 690:GOSUB 500:S=8:RETURN
570:F=4:GOSUB 690:GOSUB 500:S=8:RETURN
580:F=5:GOSUB 690:GOSUB 500:S=8:RETURN
590:F=6:GOSUB 690:GOSUB 500:S=8:RETURN
600:F=7:GOSUB 690:GOSUB 500:S=8:RETURN
610:F=8:GOSUB 690:GOSUB 500:S=8:RETURN
620:F=9:GOSUB 690:GOSUB 500:S=8:RETURN
630:DATA "OBER ","STE  ","FUNKT","ION  ","GILT  ","NACH ","ENTER","-1"
640:DATA "TEXT ","LESEN","DRUCK","AEND.","SPEI.","LADEN","-1"
650:DATA "FARBE","GROES","DRUCK","-1"
660:READ V$:IF V$="-1"RETURN
670:W$(I)=V$:I=I+1:GOTO 660
680:"CLS":CURSOR 8,1:PRINT SP$(8):CURSOR 8,2:PRINT SP$(8):CURSOR 8,3:PRINT S
    P$(8):RETURN
690:GOSUB "CLS":CURSOR 2,2:PRINT "HIER IST FUNKTION -";F;"-":GOSUB 500:
    RETURN
700:FOR I=8TO 8:W$(I)="      ":NEXT I:I=8:RETURN
```

sterinformationen erhalten. Was Sie nun tun müssen, steht in dem Pulldown-Menü, das Sie erhalten hätten, wenn Sie <ENTER> gedrückt hätten, als der Strich-Cursor unter **'HELP'** stand: OBERSTE FUNKTION GILT NACH ENTER. Also mit Skrollen die gewünschte Funktion nach oben bringen und dann <ENTER> drücken.

Damit gelangen Sie zur **eigentlichen Funktion.** In diesem **Demo-Programm** steht nur: HIER IST FUNKTION -n- , mit bestimmter Ziffer für n. Aber Sie können in der betreffenden Zeile selbst eine **nützliche Routine** einbauen. Zurück zum Pulldown-Menü führt jedenfalls, wie Sie sicher erraten, wieder <ENTER>. Wollen Sie nun Ihren **Text wieder vollständig** herstellen, drücken Sie bitte **<SPACE>** ! Das Skrollen geht bei allen 3 Pulldown-Menüs. Steht der Strich-Cursor unter **'ENDE'**, führt <ENTER> aus dem Programm. Das Programm läuft so auf dem PC 1600. Beim **PC 1350/2500** müssen Sie in der INKEY$-Schleife statt <SPACE>, <ENTER> und den Cursortasten **andere Funktionstasten** wählen, etwa <+>, <->, <(>, <)> und <*>. BEEP ändern!



4. Unser Hauptmenü

In diesem Abschnitt finden Sie den 1. Teil des **Programmes für den PC 1500(A)** in Kommentierung und als Listing. Wie schon erwähnt, sind viele Programmteile leicht auf andere SHARP-Rechner zu übertragen, sofern man die Struktur erkannt hat. Dazu trägt das aufbereitete Listing bei.

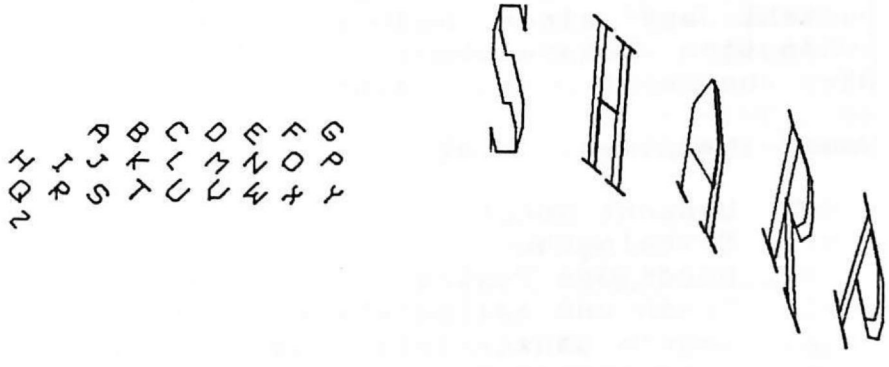
Um einen eindeutigen Bezug zu haben, geben wir dem Programm einen Namen:FormSTAR.Das Programm läuft mit oder ohne Sonderzeichen (die z.B. durch DEUZEI erzeugt sein können). Jedenfalls muß das Programm später immer mit CLOAD geladen werden **(nicht mit MERGE)**, d.h. es muß das 1. Programm im BASIC-Speicher sein. Dieses liegt am Maschinenprogramm, das nach Programmstart in die **Zeile 1** transportiert wird, wobei auf den Anfang (STATUS 2 - STATUS 1) des Programmspeichers Bezug genommen wird. Daher muß die (REM-) Zeile 1 eingeben werden und **darf später nicht gelöscht werden.** Das Mapro in Zeile 1 bewirkt die **Negativdarstellung** auf dem Display.

Gestartet wird das Programm mit **RUN oder <DEF><Z>.** Für ein klares Verständnis treffen wir jetzt die folgende Regelung:Wenn eine Taste mit bestimmter Aufschrift einfach gedrückt werden soll, schreiben wir die Aufschrift der Taste in spitzen Klammern, hier also <DEF> und <Z>. Dies geschieht **zur Unterscheidung von Zeichenfolgen,** die auf dem Display beim Eintippen erscheinen, die aber erst noch durch <ENTER> bestätigt werden müssen.

Beim **Erststart** werden alle Variablen (und damit evtl. Text) durch CLEAR gelöscht. Sodann wird die **Variable YY auf 1** gesetzt. Bei späterem **Neustart** des Textprogramms fragt formSTAR die Variable YY vor dem CLEAR ab, wonach auf dem Display die Frage erscheint: Text wirklich

```
   1 REM SYS:ZEILE EINGEBEN! ***
  10 REM *   FormSTAR-87
  20 REM *   COPYRIGHT 1987 by W.MEYER, 2087 Hasloh
  22 REM
  23 REM ******************************
  24 REM *   Start MIT Textloeschen   *
  25 REM ******************************
  26 REM
  27 "Z"CLS :WAIT 0:J$="J":IF YY=1PRINT "Text wirkl
. loesch.(J/N)":CURSOR 24:INPUT J$
  28 IF J$<>"J"BEEP 2:END
  29 CLEAR :CLS :PAUSE "  *** INITALISIERUNG ***":P
AUSE
  30 INPUT "Zei.breite? (36-79)";K$:DB=VAL K$
  32 IF DB<36OR DB>79BEEP 1:CLS :GOTO 30
  35 Q=INT ((MEM -1400)/DB)
  36 REM OPTION CE-515/6P * Q=INT ((MEM -1670)/DB)
  40 PRINT "Zeilenzahl(Max=";Q;")";:INPUT K
  42 R=K-1:IF R>Q-1OR R<0BEEP 1:CLS :GOTO 40
  50 DIM A$(R)*DB,B$(3)*80,C$(3)*80,D$(0)*80,E(5,1)
  60 YY=1:ZB=DB
  74 REM ------------------------------
  75 REM !   TITEL                     !
  76 REM ------------------------------
  80 CLS :PAUSE ">>FormSTAR-87<<";:GOSUB "ADR":PAUS
E :CURSOR 16:PRINT "Textsystem":PAUSE :PAUSE
  90 PAUSE "*** Menue-gesteuertes ***"
 100 PAUSE "*Textverarbeitungs-System*":PAUSE
 200 CLS :GOTO "ME"
 210 DATA " MENUE "," TEXT "," DRUCK "," LESEN ","
AENDERN "," FORMAT "
 215 DATA " SPEICHERN "," LADEN "," ENDE "
 218 REM * HAUPTMENUE
 220 "MENUE"Q$="MTDLAFRPE"
 230 RESTORE
 240 FOR I=1TO 9:CLS :CURSOR 4:PRINT MID$ (Q$,I,1);
:GOSUB "INV":READ R$:CURSOR 7:PAUSE R$:PAUSE :NEXT I
 248 REM
 249 REM ******************************
 250 REM *   Start OHNE Textloeschen  *
 251 REM ******************************
 252 REM
```

```
  265 " "ON ERROR GOTO "ME"
  267 IF YY<>1BEEP 5:END
  270 "ME"LOCK :CLS :BEEP 2:WAIT 0
  280 CURSOR 14:PRINT " MTD LAF RPE";:GOSUB "INV":CU
RSOR 0:PRINT "Menuewahltaste"
  299 REM ------------------
  300 K$=INKEY$ :IF ASC K$=0THEN 300
  301 REM K$=INKEY$ :IF K$=""THEN 300
  310 IF K$="M"THEN "MENUE"
  320 IF K$="T"THEN "TEXT"
  330 IF K$="D"THEN "DRUCK-150"
  335 REM * OPTION * IF K$="D"THEN "DRUCK-515"
  340 IF K$="A"THEN "AENDERN"
  350 IF K$="F"THEN "FORMAT"
  360 IF K$="L"THEN "LESEN"
  370 IF K$="R"THEN "SPEICHERN"
  380 IF K$="P"THEN "LADEN"
  390 IF K$="E"BEEP 2,99,99:UNLOCK :GOSUB 500:PRINT
C$(0):GOSUB "INV":FOR I=1TO 300:NEXT I:END
  400 BEEP 1:GOTO 300
  499 REM ------------------
  500 C$(0)=" ENDE des PROGRAMMLAUFS !":RETURN
  995 REM
```



Textverarbeitung

loeschen (J/N)? Mit dieser Sicherung soll **versehentliches Textlöschen** verhindert werden.

Das Programm wird **initialisiert**, nachdem Sie die Anzahl der **Textzeilen** und die **Zeilenbreite** festgelegt haben. Mit diesen Werten wird ein **Textfeld** A$(R)*DB dimensioniert. Die festgelegte Zeilenbreite DB wird anschließend durch ZB = DB übernommen. Für gewisse Zwecke (wie Formatieren, Drucken) kann man mit Hilfe der Variablen ZB die **Zeilenbreite später ändern**, d.h. kürzen. Zum Abspeichern des Textes auf Band benötigt man wieder die in DB festgelegte Breite (vgl. Kap. X.1).

Nach Einblendung des **Programmtitels** erscheint auf dem Display endlich unser **Hauptmenü**. Dieses ist eingebettet in ein **verzweigtes Menüsystem**, das der modularen Struktur des Programms angepaßt ist. D.h. wenn Sie vom Hauptmenü aus z.B. die Funktion LESEN ansteuern, geschieht alles, was Sie dann mit dem Programm machen, innerhalb des MODULS (Programmteils) LESEN, bis Sie es wieder verlassen. Das Menüsystem besteht aus einem Hauptmenü und den davon abhängigen **Untermenüs** in den einzelnen Moduln. Hier betrachten wir zunächst das Hauptmenü.

Unser **Hauptmenü** bietet **folgende Funktionen:**

<M> benennt Hauptmenü-Funktionen
<T> Texteingabe
<D> Druck des Textes
<L> Lesen und zeileninternes Korrigieren
<A> Ändern ganzer Zeilen/Zeilenblöcke
<F> Formatieren des Textes
<R> Speichern von Text auf Band (RECORD)
<P> Laden des Textes von Band (PLAY)
<E> Ende des Programmlaufs

In dem auf dem Display erscheinenden **invers-beschrifteten Fenster** sehen Sie die links ste-

henden **Buchstaben (Wortanfänge)** dieser Aufzählung. Durch Druck der entsprechenden Taste gelangen Sie in das zuständige Modul.

Realisiert wird diese **Steuerung durch Abfrage** der Variablen K$ (K$ = INKEY$). Nach <M>, also Drücken des Buchstabens M, werden die im Hauptmenü verfügbaren Funktionen als Wort nacheinander auf dem Display angezeigt. Außer nach <M> und <E> wird man durch **Druck einer zulässigen Taste** zu einem entsprechenden Modul geführt. Nach <E> wird der **Programmlauf beendet,** wobei aber der eingegebene Text im Textspeicher erhalten bleibt.

Natürlich darf bei Pocket-PC's der Computer nun **ohne Schaden für den Text** ausgeschaltet werden. Nur muß man bei **Wiedereinschalten,** soweit man im Text Sonderzeichen (DEUZEI) benutzt, **POKE &785D,0** <ENTER> eingeben, wenn man korrekte Zeichen darzustellen wünscht (siehe Kap. II.3). Sie können nun durch **<DEF> <SPACE>** wieder **ins Programm** gelangen, ohne den Text zu löschen, wobei Sie sofort im Hauptmenü landen. Das Programm ist ja bereits initialisiert. Nur beim **Erststart** können Sie nicht durch <DEF><SPACE> ins Programm kommen (stattdessen werden Sie durch BEEP 5 gewarnt), da die Variable YY = 0 ist, bevor das Programm nitialisiert wurde.

Nach Erreichen der **Programmarke** 'ME' wird die MODE-Taste des Rechners durch LOCK blockiert und erst wieder freigesetzt (UNLOCK) bei Verlassen des Programms durch <E>. Man sollte daher das Programm **nicht durch <BREAK>** stoppen, wenn dies vermeidbar ist. Notfalls muß man UNLOCK <ENTER> manuell eingeben.

Hat man ein Modul etwa irrtümlich angewählt, so genügt meist **einfaches Drücken von ENTER** , um zum Hauptmenü zurückzukommen.

Kap. V: TEXTEINGABE

1. Grundsätzliche Problematik

Bei der Texteingabe gilt es, zumal **bei Pocket-Computern, einige Beschränkungen** - so gut wie möglich - zu überwinden. Bei Pocket-Computern müssen wir uns, wenn wir nicht gerade ein Video-Interface zur Verfügung haben, mit der **geringen Anzeigekapazität** des Sichtfensters (Displays) zufrieden geben. Beim **4-Zeilen-Display** kann man wenigstens immer 1 Textzeile, wie sie nachher ausgedruckt wird, überblicken. Beim **1-Zeilen-Display** müssen Sie sich anders behelfen. Wir bieten dafür in den nächsten Abschnitten Lösungen an, wobei jede bestimmte Vorteile, aber auch Nachteile hat.

Sofern wir BASIC benutzen, haben wir bei den Taschen-Computern den beschränkten **Eingabepuffer** von 79 (+CR) Zeichen. Dies ist hinderlich, wenn ein Wort das Zeilenende überschreitet. Unsere **'Patentlösung'** für das 1-Zeilendisplay ist die **'segmentweise' Eingabe.** Nützlich ist auch ein **verbesserter Editor,** wie er für den PC 1500(A) vorgestellt wird.

Wir können die Zeichen natürlich einfach durch eine **INKEY$-Schleife** (bzw. GET-Schleife beim MZ 700/800) abrufen lassen als Endlos-Texteingabe. Dabei sind die **Kleinbuchstaben** aber nicht direkt eingebbar, auch müssen wir eine Art Editor entwickeln für **Korrekturen** während der Texteingabe. Das Problem liegt darin, daß die **Schleife leicht zu träge** wird und Buchstaben verschluckt. Ist die 'ultima ratio' also wieder ein Maschinensprache-Programm?

Diejenigen, die eine BROTHER EP44 oder ein ähnliches Gerät zur Verfügung haben, können dieses für ihren 'Pocket' auch als **grö@ere Tastatur** für die Texteingabe nutzen.

2. BASIC-Lösung(1): mittels INKEY$ (GET)

Bequem ist es natürlich, den **Text fortlaufend eingeben** zu können, ohne sich um Zeilenenden kümmern zu müssen. Dies ist besonders für Pocket-Computer ebenso wünschenswert wie **problematisch**, weil der Eingabepuffer, wie er bei der Funktion INPUT benutzt wird, zumeist auf 79 Zeichen begrenzt ist. Abhilfe scheint eine **INKEY$-Schleife** zu bieten. Hier ein Beispiel:

(25)
```
10  CLS: WAIT 0: D$(0)=""
20  C$=INKEY$: IF C$="" THEN 20
30  PRINT C$;
40  D$(0)=D$(0)+C$: L=LEN D$(0)
50  IF L>78 GOTO "MARKE"
60  W$=INKEY$: IF W$<>"" THEN 60
70  GOTO 20
```

Die Routine verharrt zunächst in Zeile 20, bis eine Taste gedrückt wird. In Zeile 30 wird das Zeichen ausgedruckt und in Zeile 40 in einer (vorher dimensionierten) Variablen gesammelt. Ist der Text lang genug für eine Textzeile, kann man eine Verarbeitungsroutine "MARKE" anansprungen und anschließend nach Zeile 20 zurückkehren. Zeile 60 ist ein **einfacher Trick**, um wiederholte Eingabe des gleichen Zeichens (Autorepeat) zu verhindern.

Diese INKEY-Schleife ist noch sehr schnell, d.h. nach Eingabe eines Zeichens kommen wir sofort nach 20 zurück, um das nächste Zeichen aufzunehmen. Wenn wir aber etliche Zeilen einfügen, um die Eingabe komfortabler zu machen, besteht die Gefahr, daß **bei schnellem Tippen Zeichen verschluckt** werden. Für komfortable INKEY-SCHLEIFEN brauchen wir deshalb einen schnellen Micro-Prozessor oder Interpreter, wie wir dies beim PC 1600 haben. Die **folgende Routine** stellt anders als (25) auch **Kleinbuchstaben** zur Verfügung:

```
(26)     5  DIM  D$(0)*80
         ................
        22 IF C$='*'BEEP 1,99,99:G=1:GOTO20
        24  IF  C$='/'BEEP  1: G=0: GOTO 20
        26  IF  G=1  GOTO  30
        27 IF ASC C$>&40 AND ASC C$<&B5 LET
           A=ASC  C$+32:  C$=CHR$  A
        28  IF  C$='-'LET  C$=','
        ................
        80 'MARKE' BEEP 3: REM Verarbeitung
        90  GOTO  10
```

Diese Zeilen sind zum Beisp. (25) zu ergänzen. Die Routine **läuft auf dem PC 1500(A)/1600.** Beim **PC 1350** können wir Zeile 28 weglassen, da das Komma hier nicht als SHIFT-, sondern als Grundfunktion der Tastatur vorhanden ist. In Zeile 22 muß außerdem BEEP 2 stehen (statt BEEP 1,99, 99). Beim **MZ 700/800** muß statt z.B. C$=INKEY$ stehen GET C$. Und natürlich läuft dort kein BEEP.

3. BASIC-Lösung(2): 'segmentweise' Eingabe

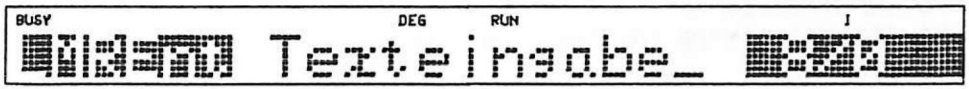
Das vorige Beispiel wäre noch auszubauen, um falsche Zeichen schon bei der Eingabe wieder tilgen zu können, was aber zu zunehmender Trägheit der INKEY-Schleife führen würde. Wir wollen daher beim **eingebauten, schellen Editor** unseres Rechners (SHARP-Editor) bleiben und das **Zeilenende-Problem** dadurch lösen, daß es gar nicht erst auftaucht.

Dieses Vorgehen, verbunden mit der INPUT-Funktion, bietet sich beim **Einzeilen-Display** an. Die Methode beruht darauf, daß der Text (ungefähr) dann mit <ENTER> weggedrückt wird, wenn das Display gefüllt ist. Diese **'Textsegmente'** werden zu einer **Textzeile** zusammengefügt. Überschreitet man dabei die zulässige Zeilenlänge, wird das letzte Segment in die nächste

Zeile übernommen.

Wir erläutern nun das **Texteingabe-Modul** für den **PC 1500(A)**. Bei Ansteuerung der Texteingabe werden Sie zunächst aufgefordert, die **Startzeile** anzugeben. Rechts sehen Sie ein inverses Fenster mit 'Sml' und 'Ent'. Das soll darauf hinweisen, die <SML>-Taste zu drücken, wenn Sie (überwiegend) Kleinbuchstaben eingeben wollen. Außerdem sind sämtliche Eingaben im Texteingabe-Modul **durch <ENTER> abzuschließen**, auch solche für Steuerungszwecke.

Auf der Anzeige erscheint darauf erneut ein Menübild mit 2 Fenstern. Das **linke bleibt während der Texteingabe bestehen**, wogegen das **rechte inverse Fenster überschrieben** werden darf und auch soll, um den verfügbaren Platz auf dem Display auszunutzen. 'L' im linken Fenster steht für Lesen, 'M' für Menü (Hauptmenü) und 'R' für Rückholen des letzten Segments. Ebenso ist zur **Ansteuerung** dieser Module bzw. Funktionen auch die Verwendung von Kleinbuchstaben (also 'l', 'm', 'r') möglich. 'Ent' verlangt den Abschluß durch <ENTER>.



Für die **Texteingabe** gilt **folgendes Verfahren:** Der Text einer Zeile wird in bis zu **4 'Häppchen' (Segmenten)** eingetippt, jeweils durch <ENTER> abgeschlossen. Also <ENTER> drücken, bevor eine Zeile ganz fertig eingegeben ist. Die Textzeile wird durch 'Stringaddition' **vom Programm zusammengesetzt.** Man darf auch durchaus noch weiterschreiben, wenn das Display voll ist. Erreicht man das 4. Segment, ertönt ein **Zeilenende-Signal,** ähnlich wie bei einer

```
   996 REM ********************************
   997 REM *   TEXTEINGABE                  *
   998 REM ********************************
   999 REM
  1000 "TEXT":CLS :D$(0)="":D=0
  1010 CLS :CURSOR 18:PRINT "Sml Ent";:GOSUB "INV":CU
RSOR 0
  1020 K$="":PRINT "ab Zeile :?      ";:CURSOR 10:INP
UT K$
  1023 IF K$=""GOTO "ME"
  1024 Z=VAL K$
  1025 IF K$=" "LET Z=ZM+1:REM * ZURUECK VOM LESEN (F
ORTSETZUNG TEXTEINGABE)
  1027 IF Z>RBEEP 1:GOTO 1010
  1030 GOTO 1040
  1034 REM ------------------
  1035 Z=Z+1
  1037 IF Z=R+1CLS :BEEP 2,99,99:PAUSE "ENDE DES TEXT
ES !":PAUSE :GOTO "ME"
  1040 FOR I=0TO 3
  1041 IF U=1LET I=1:B$(0)=C$(0):D$(0)=C$(0):D=LEN D$
(0):U=0
  1050 CLS :GPRINT "00";:PRINT "LMREnt";:CURSOR 21:PR
INT STR$ Z;"/";STR$ D:GOSUB "INV"
  1065 IF I=3BEEP 1,99,99
  1070 CURSOR 6:PRINT "                ";:CURSOR 7:REM
14*<SPACE>
  1075 INPUT B$(I)
  1080 IF B$(I)="M"OR B$(I)="m"LET A$(Z)=D$(0):GOSUB
1950:GOTO "ME"
  1083 IF B$(I)="MM"OR B$(I)="mm"GOSUB 1950:GOTO "ME"
  1090 IF B$(I)="L"OR B$(I)="l"LET B$(I)="":A$(Z)=D$(
0):ZM=Z:ZA=ZM:GOTO 3040
  1091 IF B$(I)="R"OR B$(I)="r"LET B$(I)="":GOSUB 150
0:GOTO 1070
  1092 REM * OPTION MARKE F.LEERZ. *IF B$(I)=""AND I=
0LET A$(Z)=CHR$ &7F:GOTO 1035
  1093 IF B$(I)=""GOTO 1150
  1095 BL=LEN B$(I):IF BL>ZBLET NN=3:GOSUB 1990:C$(3)
=B$(I):GOSUB 1970:GOTO 1150
  1100 C$(0)=D$(0):D=LEN D$(0)+BL
  1110 IF D>ZBLET NN=2:GOSUB 1990:D$(0)=C$(0):C$(0)=B
```

```
$(I):U=1:GOTO 1150
 1120 D$(0)=D$(0)+B$(I)
 1130 IF D>ZB-6IF D<=ZBGOTO 1150
 1140 NEXT I
 1149 REM ------------------
 1150 A$(Z)=D$(0)
 1180 GOSUB 1950
 1190 D$(0)="":D=0:Q=0:GOTO 1035
 1499 REM ------------------
 1500 IF Z=0AND I=0BEEP 1:RETURN
 1502 CURSOR 6:PRINT ": ":CURSOR 7
 1505 IF I=0PRINT RIGHT$ (A$(Z-1),13):GOTO 1520
 1511 CURSOR 7:PRINT RIGHT$ (D$(0),13)
 1520 K$=INKEY$ :BEEP 1,50,10:IF K$=""THEN 1520
 1530 RETURN
 1949 REM ------------------
 1950 FOR I=0TO 3:B$(I)="":NEXT I:D=0:RETURN
 1969 REM ------------------
 1970 FOR J=ZBTO 1STEP -1:C1$=MID$ (C$(3),J,1)
 1974 IF C1$=" "OR C1$=CHR$ 161OR C1$=CHR$ 162OR C1$
=CHR$ 163THEN 1980
 1976 NEXT J
 1979 REM ------------------
 1980 A$(Z)=D$(0):D$(0)=LEFT$ (C$(3),J):IF C1$=" "LE
T D$(0)=LEFT$ (C$(3),J-1)
 1982 C$(0)=RIGHT$ (C$(3),BL-J):U=1:Z=Z+1:RETURN
 1989 REM ------------------
 1990 FOR Q=RTO Z+NNSTEP -1:A$(Q)=A$(Q-NN):NEXT Q
 1992 A$(Z+NN-1)="":RETURN
 1995 REM
```

SHARP

SHARP

Schreibmaschine. Durch Drücken von <ENTER> ohne jede Texteingabe, d.h. durch Erzeugen eines **leeren Segments**, springt man **zur nächsten Zeile.** Ebenso gelangt man nach Abschluß des 4. Segments zur nächsten Zeile. **Leerzeilen** kann man erzeugen durch bloße Eingabe von **<ENTER> im 1. Segment** einer Zeile. Das Proramm springt dann sofort zur nächsten Zeile.

Um eine **Kontrolle** zu haben, in welcher Zeile und an welcher (ungefähren) **Textposition** man sich gerade befindet, wird bei jedem Segment das rechts stehende, überschreibbare inverse **Fenster neu aufgebaut.** Dabei steht vor dem **Schrägstrich** die Nummer der aktuellen Textzeile und hinter dem Schrägstrich die Anzahl der bereits eingegebenen Zeichen einer Zeile.

Die **Texterfassung** geschieht im wesentlichen durch die **folgenden Programmzeilen:**

```
(   )   'TE'  FOR  I=Q TO 3: REM meist Q=0
        ......................
        INPUT  B$(I)
        ......................
        D$(0)=D$(0)+B$(I)
        NEXT  I
        A$(Z)=D$(0)
        Z=Z+1:   GOTO  'TE'
```

Dazwischen liegen Anweisungen zum Menü-Aufbau und Steueranweisungen, um das Modul zu verlassen. In einer **inneren Schleife** werden die **Textsegmente gesammelt** und in einer **äußeren** der eigentlichen **Textvariablen A$(Z) übergeben,** um dann zur nächsten Zeile überzugehen.

Wenn man die **Texteingabe beenden** will, gibt man im aktuellen (noch leeren) Segment M bzw. m<ENTER> ein. Dabei wird dann der bisher in der aktuellen Zeile gesammelte Text A$(Z) übergeben. Nun kann es sein, daß man eine

bereits bestehende **Textzeile überschreiben** will. Gibt man dabei im 1. Segment nur M bzw. m<ENTER> ein, wird die betreffende Zeile gelöscht. Wichtig ist nun eine Funktion, die es erlaubt, eine irrtümlich angesteuerte **Zeile zu verlassen, ohne sie zu löschen**. Dies erreichen wir durch Eingabe von MM bzw. mm<ENTER>. Mit R bzw. r<ENTER> können wir das **letzte Segment erneut lesen** - weiter im Text durch <ENTER>.

```
60000 REM *MICRO-EDIT3*
60001 REM *COPYRIGHT WINFRIED MEYER,2087
 HASLOH*JAHR:1985
60004 CLEAR :INPUT "Anz. d. Zeilen= ";Y:
Y=Y-1:DIM A$(Y)*38
60005 "S"
60010 INPUT "!ACHTUNG! START= ";START
60020 POKE START,&58,&77,&5A,&F0,&BE,&E2
,&43,&51
60030 POKE START+8,&BE,&ED,&4D,&BE,&E4,&
2C,&B7,&00
60040 POKE START+&10,&99,&07,&B5,&00,&1E
,&F9,&9A
60500 "V"SLE=START+&17
60510 POKE SLE,&FD,&88,&FD,&98,&FD,&A8,&
A5,&76,&00,&F1,&B9,&0F,&0A,&A5,&76,&01
60520 POKE SLE+&10,&F1,&B9,&0F,&08,&68,&
78,&FD,&62,&6A,&4D,&66,&65,&1A,&25,&18,&
84
60530 POKE SLE+&20,&63,&04,&2E,&FD,&18,&
88,&0D,&6C,&77,&93,&15,&04,&F1,&AE,&77,&
4E
60540 POKE SLE+&30,&84,&F1,&AE,&77,&4F,&
FD,&2A,&FD,&1A,&FD,&0A,&F9,&9A
61000 "M":TEXT :INPUT "Start ab Zeile ?"
;Z
61010 WAIT 0:CLS :GCURSOR 0
61015 POKE &764E,&44:POKE &785D,&80,PEEK
 &785E
61020 I=0:A$(Z)=""
61030 GCURSOR I:GPRINT 0;:I=I+4
61040 CALL START:IF Z$=""THEN 61030
61045 IF ASC Z$)&20AND ASC Z$<&80LET I=I
+2
61050 IF Z$=CHR$ 13CLS :GCURSOR 0:Z=Z+1:
GOTO 61010
61055 IF Z$=CHR$ &18CLS :GCURSOR 0:TEXT
:END
61060 IF Z$=CHR$ &2ACLS :GCURSOR 0:GOSUB
 62000:Z=Z+1:GOTO 61010
61070 IF Z$=CHR$ &08AND I>7LET I=I-8:GOS
UB 61200:GOTO 61030
61080 IF I>152GOSUB 63000
61090 IF I<4LET I=4
61100 A$(Z)=A$(Z)+Z$
61110 GOTO 61030
61200 L=LEN A$(Z):A$(Z)=LEFT$ (A$(Z),L-1
):RETURN
62000 L=LEN A$(Z)
62010 FOR J=1TO L
62020 S$=MID$ (A$(Z),J,1):A=ASC S$:GOSUB
 62500:IF A<32LET A=32
62025 S$=CHR$ A
62030 LPRINT S$;
62040 NEXT J:RETURN
62500 IF A)&CALET A=A-&6A:RETURN
62510 IF A)&B0LET A=A-&70:RETURN
62520 IF A)&AALET A=A-&35:RETURN
62530 RETURN
63000 CALL SLE:CALL SLE:CALL SLE:CALL SL
E:I=152:RETURN
```

4. Maschinensprache-Editor für den PC 1500(A)

Bei aller Unterstützung durch das Menüsystem bietet unsere **BASIC-'Patentlösung'** doch keine echte **Endloseingabe,** bei der man völlig auf die Benutzung der <ENTER>-Taste verzichten kann. Dazu benötigen wir die Unterstützung durch die Maschinensprache. Hier folgt nun der in Kap. II.5 in Aussicht gestellte **Spezialeditor für die MICRO-Schrift** der die Methode der Endloseingabe mit der **39-Zeichen-Darstellung** auf dem Display verbindet.

Der Editor besteht aus einem BASIC-Programm mit **integriertem Erzeuger-Programm** für eine Maschinen-Routine, die die Darstellung der MICRO-Zeichen im **richtigen Abstand** übernimmt. Nachdem Sie die MICRO-Schrift in einem vor BASIC geschützten Bereich installiert haben, laden Sie bitte diesen Editor mit CLOAD (bzw. tippen ihn ein). Der **Erststart** muß mit RUN erfolgen, um die erwähnte Mapro-Routine zu erzeugen. **Wiederholte Starts** können dann mit <DEF><M> vorgenommen werden.

Nach Beantwortung der Startfragen können Sie nun den **Text endlos eingeben,** aber bitte nicht gar zu schnell, da sonst Buchstaben verschluckt werden. Sobald das Display gefüllt ist, rückt der Displayinhalt immer um **eine Stelle nach links,** so daß wieder Platz zum Weiterschreiben entsteht. Erlaubt, aber eben nicht vorgeschrieben, ist die Benutzung der <ENTER>-Taste zum Abschluß einer Zeile. Der Text wird nämlich in 39 Zeichen langen Zeilen gesammelt. Wollen Sie **breitere Textzeilen** haben, müssen Sie A$(Z) breiter dimensionieren und später umformatieren. Die Formatierung folgt in Kap. VIII.

Jeweils das letzte Zeichen kann zurückschreitend durch den **Rückwärts-Cursor** (DEL) gelöscht werden. Aus dem Programm führt <CL>.

5. Texte erfassen über CE-158 (z.B. mit EP44)

Ein merkwürdig Ding ist es doch mit Taschen-Computern: da hat man sich die (gar nicht so billigen) Zwerge besorgt, begeistert über soviel Kraft unter solch kleiner Schale, da verschwendet man schon Gedanken an Luxus-Erweiterungen wie Video-Interface, **externer Tastatur** usw. Machbar ist dieses mittlerweile alles. In diesem Buch wird indessen aufgezeigt, wie man - auch für die Textverarbeitung - ganz gut ohne solchen Aufwand zurechtkommt.

Mit einer Ausnahme. Demjenigen, der die vorzügliche Schnittstelle **CE-158** für seinen **PC 1500(A)** besitzt und dazu noch die weitverbreitete elektronische Schreibmaschine **BROTHER EP44**, soll gezeigt werden, wie er letztere auch zur **Texterfassung** in Verbindung mit FormSTAR benutzen kann. Ganz ähnlich dürfte die Sache mit anderen Geräten, die mit einer echten **RS-232C-Schnittstelle** ausgerüstet sind, funktionieren.

Hier folgt deshalb eine Lösung, bei der der **Text eigenständig auf der EP44** im NORMAL-Modus **erfaßt** wird, um dann im TERMINAL- Modus (der EP44) en bloc über das CE-158 in den PC1500(A) **transferiert** zu werden, bzw. in das Textprogramm FormSTAR. Der Verfasser muß davon ausgehen, daß der Besitzer einer EP44 weiß, wie er diese Maschine ohne Gefahr einer Beschädigung mit dem Rechner verbindetund daß er ein **geeignetes Kabel** dafür hat. So können wir uns hier auf die Beschreibung der Funtion der Routine beschränken.

Schreiben Sie nun einen Text in die EP44, nachdem Sie den **Rand** (LEFT MARGIN) auf eine Breite von genau **5 Zeichen** eingestellt haben, und vergessen Sie nicht, den Text in der EP44 zu speichern. Schalten Sie nach Eingabeschluß

```
60000 "K"REM * TRANS/T *
60005 REM  (C) 1985 by Winfried Meyer, H
ASLOH
60006 REM -- ACHTUNG!! --In Zeile 60110
heisst es tatsaechlich "INPUT %"
60007 "TRANS/T":CLS :WAIT 0:PRINT "Trans
f(1)   Druck(2)"
60010 K$=INKEY$ :IF K$=""THEN 60010
60011 K=VAL K$
60012 ON KGOTO 60015,60500
60013 GOTO 60010
60015 PAUSE "PARA. EP44:1200,8,N,CR,8,Y"
:PAUSE :PAUSE
60020 SETDEV :CLS
60030 IF YY=1THEN 60055
60040 CLS :INPUT "max. ZEILENZAHL?";K$:K
=VAL K$
60045 IF K=0BEEP 1:GOTO 60040
60047 R=K-1
60050 IF UY=0DIM A$(R)*80,Y$(0)*80,C$(1)
*80:UY=1:UX=1
60055 IF UX=0DIM Y$(0)*80:UX=1
60058 CLS :INPUT "wieviel ZEILEN transf.
?";K$:K=VAL K$
60060 CLS :RV=K
60061 REM zu uebertrag. Zeilenzahl um mi
nd. 2 kleiner als maximale Zeilenzahl
60062 IF K>=RBEEP 3:GOTO 60058
60070 PRINT "TEXT druecken (EP44)"
60080 SETCOM 1200,8,N,1:SETDEV KI:OUTSTA
T 0
60100 FOR I=0TO K
60110 INPUT %Y$(*)
60120 A$(I)=Y$(0)
60130 NEXT I:SETDEV :BEEP 1,100,100
60200 CLS :PRINT "A$(Z)  <== verschieb.T
EXT"
60210 FOR Z=0TO K-1:A$(Z)=A$(Z+1):NEXT Z
:A$(K)="":BEEP 3:END
60220 REM RETURN
60230 REM -- Der Teil TRANS/D folgt in K
ap. IX.4
```

die EP44 auf TERMINAL und verbinden Sie (OFF LINE) spätestens jetzt diese Maschine mit dem PC 1500(A) via CE-158. Geben Sie die **Übertragungsparameter** (sogenanntes Protokoll) der EP44, wie im Listing vermerkt, ein. Ganz zum Schluß öffnen Sie den Schalter des CE-158. Wenn der Rechner irgendwann **ERROR 150** meldet, ist vermutlich dieser Schalter auf OFF oder das CE-158 ohne **ausreichende Stromversorgung.** Das **Transferprogramm** TRANS/T können Sie nun an FormSTAR anbinden als Teil des Textsystems. Dazu müssen Sie nur in das Hauptmenü eine Zeile einfügen:

```
IF K$="K" GOTO "TRANS/T"
```

Am Schluß des Transferprogramms muß dann stehen:

```
GOTO "ME"
```

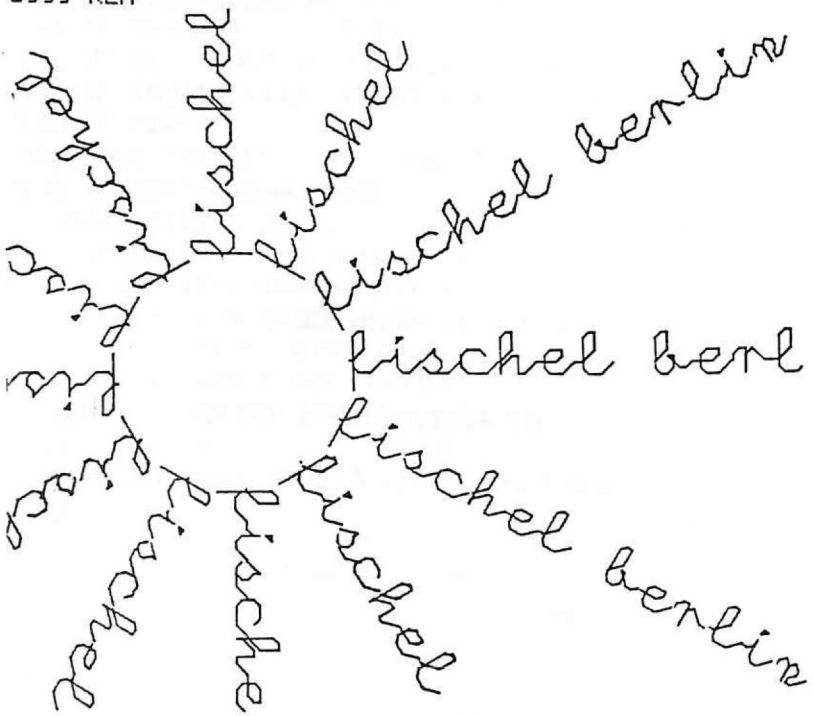
TRANS/T kann aber auch eigenständig verwendet werden oder mit MERGE zugeladen werden (hinter FormSTAR!). **Starten** Sie nun das Programm mit <DEF><K> und folgen Sie den Display-Anweisungen. Schließlich werden Sie aufgefordert, mit der EP44 ON LINE zu gehen und die **<TEXT>-Taste der EP44** zu drücken.

Die **Übertragung** bei vollem Textspeicher dauert ca. 30 Sekunden. Der Text befindet sich anschließend im Feld A$(R)*ZB (R= max. Zeilenzahl, ZB= Zeilenbreite). Wenn im Display der EP44 jetzt TEXT END erscheint, müssen Sie evtl. mehrmals die **Wagenrücklauftaste** der EP44 drücken (die mit dem gebogenen Pfeil). Schließlich erscheint auf dem Display des Computers: 'TRANSFER STOP'. Nun gehen Sie OFF LINE. Wenn Sie einen **PC 1600** verwenden, können Sie alle **Sonderzeichen** auf dessen Display richtig wiedergeben. Für den **PC 1500** müssen Sie einen 2. Zeichensatz (IBM-kompatibel) verwenden.

```
 2996 REM ******************************
 2997 REM *   LESEN /KORRIGIEREN       *
 2998 REM ******************************
 2999 REM
 3000 "LESEN"
 3010 CLS :S=0
 3020 CURSOR 23:PRINT "Ent";:GOSUB "INV":CURSOR 0
 3030 K$="":PRINT " Lesen ab Zeile:?";:CURSOR 16:INP
UT K$:ZA=VAL K$
 3035 IF K$=""GOTO "ME"
 3040 GOSUB "MEN-L"
 3046 PRINT " ";"<= LESESTEUER.":GOSUB "END-ZL"
 3050 Z=ZA:C=LEN A$(Z)+1
 3060 CURSOR 8:PRINT "                          ":GOSUB "TXT
-L":CURSOR 0:PRINT STR$ Z:REM 18*<SPACE>
 3098 REM ------------------
 3099 "INK-L"
 3100 M$=INKEY$
 3122 IF M$=""IF S=1GOTO 3600
 3124 IF M$=""IF S=2GOTO 3640
 3132 IF M$=CHR$ &0CLET S=0:GOTO 3600
 3134 IF M$=CHR$ &08LET S=0:GOTO 3640
 3142 IF M$=")"LET S=1:GOTO 3100
 3144 IF M$="("LET S=2:GOTO 3100
 3190 IF M$="+"GOSUB "EINF":GOSUB "MEN-L":GOSUB 3990
:S=0:GOSUB "TXT-L":GOTO "INK-L"
 3191 IF M$="-"GOSUB "LOE":GOSUB "MEN-L":GOSUB 3990:
S=0:GOSUB "TXT-L":GOTO "INK-L"
 3200 IF M$="*"GOSUB "UEB":GOSUB "MEN-L":GOSUB 3990:
S=0:GOSUB "TXT-L":GOTO "INK-L"
 3215 IF M$=CHR$ &0AIF Z<RLET Z=Z+1:CURSOR 0:PRINT "
  ":CURSOR 0:PRINT STR$ Z:C=LEN A$(Z):GOTO "D-LOE"
 3225 IF M$=CHR$ &0BIF Z>0LET Z=Z-1:CURSOR 0:PRINT "
  ":CURSOR 0:PRINT STR$ Z:C=LEN A$(Z):GOTO "D-LOE"
 3230 IF M$="M"WAIT :GOTO "ME"
 3235 IF M$<>""BEEP 1
 3240 S=0:GOTO "INK-L"
 3299 REM ----------
 3300 "D-LOE"CURSOR 8:PRINT A$(Z);"
  "
 3310 S=0:GOTO "INK-L"
 3599 REM ----------
```

```
 3600 C=C+1
 3605 IF C>LEN A$(Z)+1IF Z>OLET Z=Z-1:C=0:CURSOR 8:P
RINT "                    ":REM 18*<SPACE>
 3610 IF C>64IF Z=0LET S=0:GOTO 3680
 3617 GOSUB 3700:GOTO 3100
 3639 REM ----------
 3640 C=C-1
 3645 IF C<1IF Z<RLET Z=Z+1:C=LEN A$(Z)
 3647 IF C<19IF Z=RVLET S=0:GOSUB 3700:GOTO 3100
 3650 IF C<1IF Z=RLET S=0:GOTO 3680
 3657 GOSUB 3700:GOTO 3100
 3680 GOSUB 3994:GOTO 3100
 3699 REM ------------------
 3700 "TXT-L"CURSOR 1:PRINT " ":CURSOR 0:PRINT STR$
Z
 3701 CURSOR 7:PRINT " ";RIGHT$ (A$(Z),C);" ":RETURN

 3797 REM ------------------------------
 3798 REM !   Ueberschreiben           !
 3799 REM ------------------------------
 3800 "UEB"LC=LEN A$(Z)-C:C$(0)=LEFT$ (A$(Z),LC)
 3805 C$(3)="":CURSOR 7:INPUT C$(3)
 3810 LD=LEN C$(3):IF LC+LD>ZBBEEP 3:RETURN
 3815 IF C>LDLET C$(1)=C$(3)+RIGHT$ (A$(Z),C-LD)
 3820 IF C<=LDLET C$(1)=C$(3)
 3825 A$(Z)=C$(0)+C$(1):RETURN
 3837 REM ------------------------------
 3838 REM !   Loeschen                 !
 3839 REM ------------------------------
 3840 "LOE"CURSOR 0:PRINT "->??<<";
 3845 CURSOR 2:INPUT H$:H=VAL H$:PAUSE
 3850 L=LEN A$(Z):M=L-C
 3852 IF H>CLET H=C
 3855 REM IF C=00R C>LLET A$(Z)=RIGHT$ (A$(Z),L-H):R
ETURN
 3860 C$(0)=LEFT$ (A$(Z),M):C$(1)=RIGHT$ (A$(Z),C-H)
 3870 A$(Z)=C$(0)+C$(1)
 3880 RETURN
 3897 REM ------------------------------
 3898 REM !   Einfuegen                !
 3899 REM ------------------------------
 3900 "EINF"C$(2)=RIGHT$ (A$(Z),C):LC=LEN C$(2)
```

```
 3902 CURSOR 0:PRINT "<<<<<+";:CURSOR 0:D$(0)="":INP
UT D$(0)
 3905 D=LEN D$(0)
 3910 M=LEN A$(Z)-C:IF M<0LET M=0
 3920 C$(0)=LEFT$ (A$(Z),M):LM=LEN C$(0)
 3930 IF LM+D+LC<=ZBLET A$(Z)=C$(0)+D$(0)+C$(2):RETU
RN
 3940 GOSUB "1-FREI":ZM=ZM+1
 3950 IF LM+D<=ZBLET A$(Z)=C$(0)+D$(0):A$(Z+1)=C$(2)
:RETURN
 3960 A$(Z)=C$(0):A$(Z+1)=D$(0)
 3970 Z=Z+2:GOSUB "1-FREI":A$(Z)=C$(2):Z=Z-2
 3980 RETURN
 3989 REM ----------
 3990 IF C<=LEN A$(Z)-8LET C=C+8
 3991 RETURN
 3993 REM ----------
 3994 "MEN-L"PAUSE :CLS :CURSOR 2:GOSUB "KREUZ":PRIN
T "M*+-";
 3996 GOSUB "INV":CURSOR 7:RETURN
 3999 REM
```



rückwärtsgerichtet mit der Taste **'Klammer zu'**. Die Laufschrift setzt sich aber erst nach Loslassen dieser über <ENTER> gelegenen Tasten in Bewegung.

Die **Laufschrift** kann wieder **gestoppt** werden durch Drücken einer **beliebigen Taste** (außer <BREAK>), die nicht mit einer speziellen Funktion belegt ist. Die **Zeilennummer** der aktuellen Zeile wird immer ganz links angezeigt. So können Sie sich immer bestens im Text orientieren. Damit die selbsttätige Laufschrift bei der **letzten eingegebenen Textzeile** anhält, erfragt das Programm in der Routine 'END-ZL' (siehe Modul ROUTINEN) die Nummer dieser letzten benutzten Textzeile und speichert sie in der Variablen RV. Die Routine 'END-ZL' wird von mehreren Programmteilen aufgerufen.

Die bei der Marke 'INK-L'beginnende INKEY$-Schleife benutzt zur Steuerung der Laufschrift einen **Schalter** (Variable S). Durch Drücken einer **Klammertaste** <(> oder <)> wird der Schalter S=1 oder S=2 gesetzt. In diesem Falle verzweigt die Steuerung bei jedem Abfragen der INKEY$-Variablen sofort in die **verarbeitenden Routinen,** die mit C=C+1 und C-1 beginnen. Die Variable C gibt die Anzahl der momentan höchstens zulässigen Zeichen eines Strings an, die auf dem Display dargestellt werden sollen. Beim nächsten Durchlauf wird diese Anzahl entweder um 1 vermindert (Vorwärtsrichtung) oder um 1 erhöht (Rückwärtsrichtung).

Durch dauernde Ansteuerung der Routine 'TXT-L' mit veränderten C-Werten entsteht der Effekt einer Laufschrift. Bei Drücken einer Cursortaste wird der Schalter S=0, so daß nur bei erneutem Betätigen dieser Tasten in die verarbeitenden Routinen verzweigt wird.

3. Fließschrift beim PC 1500(A) in Maschinensprache

Da die **Laufschrift** zwar gut lesbar ist, aber etaws unruhig wirkt, entwickelte der Verfasser eine andere, die **Fließschrift** genannt werden soll, bei der die Schrift immer nur um **eine GCURSOR-Position** zur Zeit wandert. In BASIC wäre eine solche Schrift zu langsam, also mußte ein **Maschinen-Programm** her.

Das folgende Programm enthält einen **Vorspann zur Texteingabe**, der aber nicht sehr komfortabel ist. Lassen Sie diesen einfach weg, wenn Sie den Text mit FormSTAR eingeben. Laden Sie in dem Fall dieses Programm mit **MERGE** dazu und starten Sie seine Leseroutine mit <DEF><S>, sonst aber mit RUN.

Nach <DEF><S> läuft sofort die Fließschrift in **Vorwärtsrichtung** (auf die entsprechende Routine in **Rückwärtsrichtung** müssen wir aus Platzgründen verzichten). Mit den **Cursor-Tasten** zwischen <SPACE> und <ENTER> können verschiedene **Textzeilen angesteuert** werden. Mit <ENTER> verläßt man die Routine. Durch Druck einer beliebigen sonstigen Taste (außer BREAK!) bleibt die Schrift am Zeilenanfang stehen, **solange die Taste festgehalten** wird. Und nun löschen Sie einmal eine Textzeile, Z.B. A$(0) durch A$(0)="" und gehen anschließend durch <DEF><S> wieder ins Programm. **Nicht zu glauben**, die Zeile ist bis auf den ersten Buchstaben nach da! Das kommt daher, daß beim **normalen Löschen** einer Zeile im Variablenspeicher nur das 1. Zeichen mit &00 gelöscht wird.

Als nette Zugabe jetzt ein **'Röntgen-Monitor'** mit Fließschrift. Nach Start durch <DEF><L> und Eingabe einer Speicheradresse können Sie sich beispielsweise ansehen, wie Ihr BASIC-**Programm intern abgespeichert** wird unter

```
40D0    FD CB 84 28 04 2A FD 8A 0A FD 88 FD  A8 5A 06 14 FD 2A FD A8 AE 78 75 B5
40E8    1A CD 92 68 00 6A 00 CD AC CD A6 89  24 FD 98 BE E4 2C FD 1A B7 00 89 19
4100    56 14 B7 00 99 26 FD 2A 64 FD 0A 46  FD 88 4E 00 99 37 FD 0A AE 77 D0 F9
4118    9A FD 2A 9E 0B 10 60 00
```

```
  5 REM FLIESS-SCHRIFT/SPEZIAL-MONITOR
  6 REM (C)1986 by W.MEYER, HASLOH
 10 CLEAR :DIM A$(9)*79
 15 INPUT "Anzahl der Zeilen: ";K
 16 WAIT 0
 20 FOR I=0TO K:CLS
 30 PRINT "TEXT: ";:INPUT A$(I)
 40 IF A$(0)="*"THEN END
 50 NEXT I
 60 BEEP 2:END
 70 "S"POKE &77D0,0,0:CURSOR 0
 80 FOR I=0TO K
 90 X$="":CURSOR 0
100 CALL &40D0,A$(I)
110 IF X$=CHR$ 13BEEP 3:END
120 IF X$=CHR$ &0AAND I<KGOTO 150
130 IF X$=CHR$ &0BAND I>0LET I=I-2:GOTO 150
140 IF X$<>""LET I=I-1
150 NEXT I
160 BEEP 2:END
```

```
 10:REM  SKROLLEN (VIRTUELLES DISPLAY)
 20:"A"CLEAR :CLS :CURSOR 0,0:INPUT "ANZAHL DER ZEIL
    EN: ";R:R=R-1
 30:DIM A$(R)*79:CLS
 40:FOR I=0TO R
 50:CLS :INPUT A$(I):NEXT I
 60:"L"I=0:J=1:LINE (0,0)-(155,0),,,BF
 70:SP$="                ":REM 13*<SPACE>
 80:GOSUB 160
 90:K$=INKEY$ :IF K$=""THEN 90
100:IF K$="4"AND J<54LET J=J+1:GOSUB 160
110:IF K$="8"AND J>1LET J=J-1:GOSUB 160
120:IF K$="6"AND I>0LET I=I-1:GOSUB 150:GOSUB 160
130:IF K$="2"AND I<R-3LET I=I+1:GOSUB 150:GOSUB 160
140:GOTO 90
150:CURSOR 0,1:FOR K=1TO 3:PRINT SP$;SP$:NEXT K:
    RETURN
160:USING "###":CURSOR 0,0:PRINT J:CURSOR 9,0:PRINT
    " -";I;" - ":CURSOR 23,0:PRINT J+25:USING
170:CURSOR 0,1:PRINT MID$ (A$(I),J,25);" "
180:CURSOR 0,2:PRINT MID$ (A$(I+1),J,25);" "
190:CURSOR 0,3:PRINT MID$ (A$(I+2),J,25);" ":RETURN
```

```
 170 "L"CLS :WAIT 0
 180 POKE &77D0,0,0
 190 CLS :INPUT "STARTADR.(High-byte):
";P
 200 CLS :INPUT "STARTADR.(Low-byte): "
;Q
 210 POKE &40CA,&B5,&20,&48,P,&4A,Q
 220 R=256*PEEK &40CD+PEEK &40CF
 230 X$="":CALL &40CA
 240 IF X$=CHR$ 13BEEP 3:END
 250 IF X$=CHR$ 32GOSUB 300:GOTO 290
 251 REM --- <SPACE> zeigt die momentan
e Startadresse an
 260 IF X$=CHR$ &0BGOSUB 400:GOTO 290
 270 IF X$=CHR$ &0AGOSUB 500:GOTO 290
 280 IF X$<>""GOTO 230
 290 R=256*PEEK &40CD+PEEK &40CF:GOTO 2
10
 300 PRINT "ADRESSE = ";R
 310 K$=INKEY$ :IF K$<>""GOTO 310
 320 RETURN
 400 IF Q<=0LET Q=224:P=P-1:RETURN
 410 Q=Q-32:RETURN
 500 Q=Q+32
 510 IF Q>255LET Q=0:P=P+1:RETURN
 520 RETURN
 600 REM * BEI ENTSPRECHENDER AENDERUNG
 DER EINSPRUNGADRESSEN IST DAS PGM AUCH
BEI
 610 REM * ANDERER LADEADRESSE DES (REL
OKATIBLEN) MASCHINENPGMS LAUFFAEHIG
```

Berücksichtigung der Token für die Befehle. Zur **Steuerung** in diesmal 32-byte-Schritten benutzen Sie wieder besagte Cursor-Tasten und gehen mit <ENTER> aus dem Programm. Die Routine ist im Speicher **verschiebbar.**

4. Horizontales und vertikales Skrollen

Ein **horizontales Skrollen** beim Einzeilen-Display fanden wir bei der Laufschrift bzw. Fließschrift vor. Eingearbeitet in das Textsystem hatten wir auch ein **vertikales Skrollen** zur Verfügung. Skrollen bezeichnet also die Möglichkeit, bisher unsichtbare, nachfolgende Textausschnitte einzublenden.

Für das **4-Zeilen-Display** folgt jetzt eine Ausführung mit **Kopfzeile in Negativ-Darstellung.** In dieser werden links und rechts permanent die begrenzenden Spalten angezeigt, während in der Mitte die aktuelle Zeilennummer steht. In diesem mit <DEF><A> zu startenden **Demo-Programm** sind zunächst einige Textzeilen einzugeben. Der Text kann dann mit den **Zifferntasten** <4>, <6>, <8> und <2> geskrollt werden. Die (3 unteren) Zeilen der Anzeige wirken dabei wie ein Fenster, das beliebig über eine Textseite bzw. den ganzen Text bewegt werden kann (virtuelle Darstellung). Soll das Programm auf dem **PC 1600** laufen, bekommt die Variable AB (Anzeigenbreite) den Wert 26 zugewiesen, beim **PC 1350/2500** den Wert 24.

Kap. VII:

BEARBEITEN DES TEXTES: GRUNDFUNKTIONEN

1. Zeichenorientierte Veränderungen

a) beim Einzeilen-Display durch BASIC

Eine Textverarbeitung ohne die Möglichkeit zum **Editieren**, d.h. des Veränderns von Text in einem speziellen Arbeitsgang, bleibt unvollständig. Die primitivste Möglichkeit wäre, einzelne Zeilen völlig neu zu schreiben. Wenn aber beispielsweise in der betreffenden Zeile nur 1 Buchstabe falsch ist, wäre das gewiß sehr aufwendig.

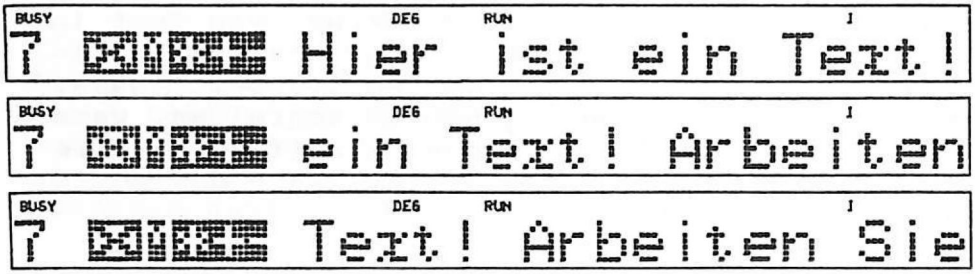
Deshalb ist bei **FormSTAR** in den Lesemodus eine **Korrekturfunktion** eingebunden. Genauer gesagt sind es 3 Korekturfunktionen: **Löschen**, **Einfügen** und **Überschreiben**, wobei die Veränderung jeweils an der Zeile vorgenommen wird, deren Nummer links auf dem Display sichtbar ist.

Betrachten Sie jetzt bitte dazu noch einmal das in VI.2 abgedruckte Listing des Moduls LESEN/KORRIGIEREN. Wenn M$ die Werte '*', '-' oder '+' zugewiesen bekommt, also nach Tastendruck von <*>, <-> oder <+>, verzweigt das Programm in die Routinen 'UEB', 'LOE' oder 'EINF'. Dies bedeutet überschreiben, Löschen und Einfügen. Die zugehörigen **Symbole** sehen Sie im **Negativ-Fenster** auf dem Display.

Um nun Zeichen zu **löschen**, müssen wir die eine Textzeile enthaltende Zeichenvariable durch Stringmanipulationen um H Zeichen kürzen. Der verbleibende Rest wird zur neuen Textzeile zusammengesetzt, wie dies in der **Löschroutine** 'LOE' geschieht. Als **Stellgröße** dient die Variable C, die wir bekanntlich bei der Lauf-

schrift beeinflussen können. Daraus folgt das praktische Vorgehen beim Löschvorgang:

Fahren Sie den Text einer zu bearbeitenden Zeile mit Hilfe der Cursortasten an den rechten Rand des (linksstehenden) Fensters heran. Das, was dann noch zu sehen ist, wird von **links her bearbeitet.** Dieses Verfahren gilt



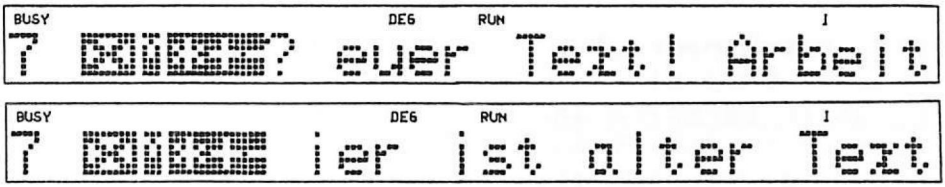
ebenso für die Funktionen überschreiben und Einfügen. Zum **Löschen** drücken Sie jetzt <-> und geben anschließend die **Anzahl der zu löschenden Zeichen** nebst <ENTER> ein. Ein Ausschnitt der so verkürzten Zeile erscheint schließlich als **Standschrift** auf dem Display. Nun können Sie die **Laufschrift** wieder in Gang setzen, entweder mit den Cursor- oder selbsttätig mit den Klammertasten.

Genauso verfahren Sie beim **Einfügen,** nur daß Sie jetzt <+> drücken. Der einzufügende Text

gelangt in der Folge genau an die **Bruchstelle** zwischen dem **'hinter dem Fenster'** verschwundenen Text und dem **noch sichtbaren** Text. Geben Sie den Zwischentext unbekümmert ein, auch wenn Sie den auf dem Display sichbaren Text (scheinbar) überschreiben. Nach Abschluß Ihrer Eingabe durch <ENTER> erscheint diesmal ein Ausschnitt der **verlängerten Zeile** auf dem Display.

Für den Fall, daß die **neue Textzeile länger** als die zulässige (dimensionierte) Länge ist, ist vorgesorgt. In diesem Fall wird der ganze der aktuellen Zeile folgende Text um **eine Zeile nach hinten**, d.h. zu den oberen Zeilennummern hin, **verschoben**, und der überschüssige Teil **(ohne Wort-Bruch)** auf die nun freie nächste Zeile geschrieben. Bei **längeren Einschüben** werden unter Umständen sogar 2 neue Zeilen eingerichtet, so daß ungleiche Zeilenlängen entstehen. Das schadet aber nichts, weil diese Ungleichheiten mit der in Kap. VIII zu besprechenden **Formatierungsfunktion** wieder beseitigt werden können.

Als **3. Möglichkeit** können Sie nach <*> Text auch tatsächlich und auf dem Display sichtbar **überschreiben**. Dauerhaft überschrieben ist der Text aber erst nach abschließendem <ENTER>.



Man hat daher die Mölichkeit, durch <CL> nebst <ENTER> die Prozedur ohne Textveränderung abzubrechen. Die Texteingabe beginnt an der Stelle des Fragezeichens. Übrigens kann man

beim überschreiben von Text ebenso wie beim Einfügen die unter <MODE> stehenden **Cursortasten mit normaler Wirkung** benutzen, da man sich solange im normalen SHARP-Editor (BASIC-Befehl INPUT) befindet, bis <ENTER> eingegeben wurde.

Keinesfalls darf man aber die Cursortaste 'Pfeil nach unten' benutzen, da der Rechner dann in den **Einzelschritt-Modus** verfällt. **Verboten** sind auch die **normalen Anführungszeichen** aus den gleichen Gründen wie bei der Texteingabe (siehe V.3). **Zulässig** ist ein überschreiben über die bestehende Länge der aktuellen Textzeile hinaus bis zur dimensionierten Länge. Nützlich erweist sich das überschreiben beim **nachträglichen 'Einfügen' von Farbsteuerzeichen** für die Druckroutine. Diese nämlich treten an die Stelle von normalen Wortzwischenräumen (SPACES) - siehe dazu Kap. IX.1c.

b) <u>beim PC 1500 (A) mit Maschinensprache</u>

Der Vorteil der Editorlösung des vorigen Abschnitts ist seine vollständige **Einbindung** in das **Menüsystem,** speziell das Untermenü LESEN. Hier folgt noch eine **unhabhängige Lösung** mit einem in Assembler bzw. Maschinensprache geschriebenen Kern.

Die **Maschinenroutine** stellt eine Texteingabe-Methode dar, die in der <u>Wirkungsweise zwischen en BASIC-Befehlen INPUT und INKEY$</u> steht. Sie wird mit CALL REKEY,V aufgerufen, wobei REKEY für die **Startadresse** im Speicher steht und V die Anzahl der Zeichen bedeutet, die auf dem Display dargestellt werden können (bis zu 26), bevor der Display-Inhalt ohne <ENTER> verschwindet. Hierbei wird die Routine wieder verlassen. Allerdings bewirkt auch vorzeitiges <ENTER> ein Hinausgehen aus der Routine.

```
1700    42 04 2A 58 77 5A B0 15 AE 77 A0 FD   A8 BE E2 43 FD C8 B7 0C 8B 42 FD 8A
1718    B7 08 89 6A A5 78 75 BE 17 E3 B1 06   AE 78 75 56 15 FD C8 B5 7B BE ED 57
1730    FD C8 A5 78 75 B3 06 AE 78 75 FD 8A   BE E4 2C B7 00 99 07 A5 78 75 B1 06
1748    AE 78 75 FD 8A BE ED 57 FD 2A 60 60   88 4F 8E 89 FD 8A 55 FD C8 B5 7D BE
1760    ED 57 BE E4 2C B7 00 99 07 A5 78 75   BE 17 F6 FD 8A BE ED 57 FD C8 A5 78
1778    75 B3 06 AE 78 75 FD 8A FD 2A 88 7D   8E 5B 51 FD C8 BE ED 57 FD 8A B7 20
1790    89 0E FD C8 A5 78 75 B3 05 AE 78 75   FD 8A 8E 29 FD C8 A5 78 75 B3 04 AE
17A8    78 75 FD 8A B5 20 FD C8 FD 88 FD A8   BE ED EF FD 2A FD 0A FD 8A FD C8 A5

17C0    78 75 B3 01 AE 78 75 FD 8A BE E4 2C   B7 0D 8B 0A B7 00 99 0B FD 2A 88 D1
17D8    8E 07 FD 2A 56 A5 77 A0 51 F9 9A FD   C8 B7 00 89 0A FD 8A B5 06 AE 78 75
17F0    54 8E 02 FD 8A 9A FD C8 B7 96 89 06   FD 8A B1 06 8E 02 FD 8A 9A
```

```
3998 REM EDITOR/SCHREIB
3999 REM (C) 1986 by Winfried Meyer, Ha
sloh
4000 "F"CLEAR :CLS :INPUT "Zei.zahl:";R
4005 WAIT 0:PRINT "Startzeile: ";:INPUT
Z
4010 CLS :PAUSE "...<< hier":PAUSE
4015 DIM A$(R)*78,B$(2)*26
4020 FOR K=0TO 2
4030 CLS :V=26:CALL &1700,V
4035 BEEP 3,10,50
4040 B$(K)=V$+W$
4050 FOR I=0TO 26:POKE &77B0+I,0:NEXT I
4060 L=LEN B$(K)
4070 IF RIGHT$ (B$(K),1)=CHR$ &18LET B$
(K)=LEFT$ (B$(K),L-1):GOTO 4200
4080 A$(Z)=A$(Z)+B$(K)
4090 NEXT K
4100 Z=Z+1:IF Z>RBEEP 5:END
4110 GOTO 4020
4200 A$(Z)=A$(Z)+B$(K)
4210 FOR I=0TO 26:POKE &77B0+I,0:NEXT I
4220 BEEP 1,99,99:END
```

REKEY = &1700

```
7000 "L"CURSOR 0:WAIT 0:PRINT "Startzei
le: ";:INPUT Z
7005 CLS
7010 FOR K=0TO 2
7030 B$(K)=MID$ (A$(Z),26*K+1,26)
7032 NEXT K
7040 FOR K=0TO 2
7041 V$=LEFT$ (B$(K),16):W$=MID$ (B$(K)
,17,10)
7042 PRINT B$(K)
7050 V=26:CALL &1700,V
7055 CLS
7060 B$(K)=V$+W$
7062 FOR I=0TO 26:POKE &77B0+I,0:NEXT I
7065 A$(Z)=B$(0)+B$(1)+B$(2)
7066 GOSUB 7300
7070 NEXT K
7100 Z=Z+1:IF Z>RBEEP 5:END
7110 GOTO 7010
7300 PRINT "0"
7305 C$=INKEY$ :IF C$=""THEN 7305
7310 IF C$="0"RETURN
7320 IF C$=CHR$ &18BEEP 1,99,99:END
7330 GOTO 7305
8000 "D"CSIZE 1:FOR Z=0TO R
8010 LPRINT A$(Z):NEXT Z:END
```

Der Text wird dabei in den BASIC-Standardvariablen V$ (16 Zeichen) und W$ (10 Zeichen) über die Maschinenroutine erfaßt und kann von BASIC weiterverarbeitet werden. **Interessant ist folgendes:** Man kann auch V$ und W$ über BASIC vorbelegen und hat dann bei Aufruf von REKEY,V schon einen Text, der sich innerhalb der Routine bearbeiten, d.h. überschreiben läßt. Dafür stehen 2 (!) **Cursor zur Verfügung** in Form von geschweiften Klammern. Verwendet werden die Cussortasten unter <MODE>. Die 'Klammer-Cursor' **verschwinden wieder bei Loslassen** dieser Tasten, reagieren also nur auf direkten Druck.

Beachten Sie bitte die unterschiedliche Anwendung der Cursor. Beim **Rückwärts-Cursor** liegt die Schreibposition auf der Cursor-Position, beim **Vorwärts-Cursor** dagegen eine Stelle vor der Cursor-Position.

Wir können die Routine - eingebunden in ein BASIC-Programm - also sowohl zur Texteingabe als auch zum **Editieren** benutzen. Beide Teile werden hier im Zusammenhang wiedergegeben, obwohl der 1. Teil eigentlich in Kap.V gehört. Dieses Modul dient an sich dazu, die **SCHREIBSCHRIFT** zu ver/bearbeiten, kann aber auch mit **Normalschrift** verwendet werden.

Beim **Texteingabe-Teil** wird der Text in 3 * 26 Zeichen langen Teilen für die 78 Zeichen lange Zeile gesammelt. Verlassen kann man diesen Programmteil (den Sie mit DEF F starten können) mit <CL>.
Der **eigentliche Editor** wird mit <DEF><L> gestartet. Mittels abwechselnd <ENTER> und <0> können Sie sich den nächsten Textteil zeigen lassen. Programmende durch <ENTER> nebst <CL>.

2. Zeilenorientierte Veränderungen

a) Zeilen löschen und einfügen

Eine weitere Möglichkeit, den Text zu verändern, besteht darin, **ganze Zeilen** bzw. **Zeilenblöcke** zu manipulieren. Dafür bietet FormSTAR ein eigenes Modul an, dessen Listing in diesem Abschnitt wiedergegeben wird.

Nach <A> im Hauptmenü kommen wir zum **'Änderungsmodus'**, der sich uns mit einem Untermenü präsentiert. **Folgende Funktionen** stehen zur Verfügung:

<L>esen <+> =Einfügen <-> =Löschen
<C>opy <M>enü

Die beiden letzten Funktionen sind **Rückkehrmöglichkeiten** zum Lesemodus und zum Hauptmenü. Übrigens kommen wir auch hier wie in den anderen Modulen durch einfaches <ENTER> zurück zum Hauptmenü.

Nach <+> erfolgt eine Verzweigung in die **Routine 'PLUS'**, bei der nach Eingabe der Zeilennummer K, bei der die Operation beginnen soll, und Angabe einer gewünschten Anzahl V Leerzeilen in den Text eingefügt werden. Dabei werden die Textzeilen in einem 1. Schritt auf die Zeilen mit um V höheren Nummern kopiert. Im 2. Schritt werden die V Zeilen, die der K-1. Zeile folgen, so gelöscht, daß V Leerzeilen entstehen, d.h. A$(K)="",...,A$(K+V-1)="".

Nach <-> erfolgt eine Verzweigung in die **Routine 'Minus'**, bei der nach Angabe der Zeilennummer K, bei der die Operation beginnen soll,

und Angabe einer gewünschten Anzahl V Zeilen gelöscht werden. Dabei werden die Textzeilen, von der Zeile K beginnend, durch Textzeilen mit um V höherer Zeilennummer überschrieben. In einem 2. Schritt werden die letzten V Zeilen mit Leerzeilen überschrieben (gelöscht).

b) Zeilen kopieren

Das **Kopieren von Zeilen** geschieht im Prinzip genau wie das Einfügen von Leerzeilen, nur daß dabei im 2. Schritt die betreffenden Zeilen nicht gelöscht, sondern durch die zu kopierenden Zeilen überschrieben werden.

Nach <C> verzweigt das Programm zur **Routine 'COPY'.** Nach Eingabe einer Startzeile T1 und einer einschließlichen Schlußzeile T2 sind die T zu kopierenden Zeilen festgelegt, die ab Zeile T3 in den Text eingefügt werden. Man kann sowohl Zeilen mit niedriger Nummer nach oben kopieren als auch Zeilen mit hoher Nummer nach unten. Nur dürfen sich **nicht beide Bereiche überschneiden.** Die Routine verweigert daher Eingaben, bei denen T3 (wo der zu kopierende Text eingesetzt wird) zwischen T1 und T2+1 liegt. Ebenso wird ein Hinausschieben von Text aus dem Textspeicher vermieden, indem die Routine auf eine andere Routine namens 'END-ZL' zugreift, welche die letzte beschriebene Textzeile liefert.

c) Zeilenblöcke verschieben

Durch das Kopieren steht ein Block von Textzeilen natürlich zweimal im Speicher. Durch anschließendes **Löschen des Textes an der ursprünglichen Stelle** mit Hilfe der über <-> erreichbaren Löschroutine kann man einen **Textblocktransfer** bewirken. Leerzeilen entstehen nicht, da der Text beim Löschen im Effekt zusammengeschoben wird. Man könnte leicht

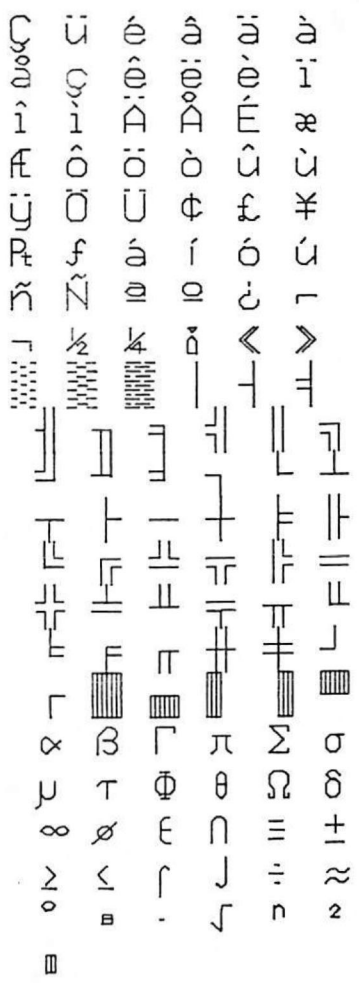
beide Schritte in einer Routine vereinen, indem man beide Vorgänge einfach automatisch hintereinander ausführen läßt.

3. Textstellen suchen und Text ersetzen

Die folgende Routine ist in FormSTAR noch nicht integriert, aber sie wäre leicht dort einzubauen. Dazu braucht man nur die **Marke 'S'** anzuspringen und das END der Zeilen 10090 sowie 11020 durch ein RETURN ersetzen. Als **DEMO-Programm** startet man es mit RUN, wonach in einem **Vorspann** zunächst einige Textzeilen eingegeben werden. Anschließend geht man mit <DEF><S> in die **Such-Routine.**

Hier schreibt man hinter 'SUCHE:' eine im Text vermutete **Zeichenkombination** bzw. 1 oder mehrere Wörter bis zur Gesamtlänge von 16 Zeichen. Die Routine vergleicht nun diese Kombination Stück für Stück mit dem Text. Bei Textgleichheit erfolgt ein Sprung zum **Unterprogramm 'Ersetzen'**, wo nun ein **Ersatzwort** eingegeben werden kann. Dabei wird links immer die Zeilennummer sowie die Textspalte angegeben, wo das gefundene Wort steht. Drückt man, statt hinter 'ERSATZ:' ein neues Wort einzugeben, einfach <ENTER>, bleibt das gesuchte Wort n der betreffenden Stelle unverändert erhalten, und die Routine sucht nun nach der **nächsten Textstelle** mit der gleichen Zeichenkomination. In dieser relativ einfachen Routine darf die Gesamtzeile die definierte Zeilenbreite nicht überschreiten, da sonst das Ersatzwort nicht angenommen wird.

```
10000 REM SUCH-ROUTINE
10010 INPUT "ANZ. D. ZEILEN: ";R:R=R-1:I
NPUT "ZEILENBREITE: ";ZB
10020 DIM A$(R)*ZB,C$(2)*ZB
10030 FOR Z=0TO R:INPUT A$(Z):NEXT Z:BEE
P 2:END
10040 "S"CLS :WAIT 0:INPUT "SUCHE: ";D$:
DS=LEN D$
10050 FOR Z=0TO R:L=LEN A$(Z)
10060 FOR J=1TO L-DS+1
10070 E$=MID$ (A$(Z),J,DS)
10080 IF E$=D$GOSUB 11000
10090 NEXT J:NEXT Z:BEEP 2,99,99:END
10100 REM --------
11000 REM --- ERSETZEN
11010 CLS :PRINT "Z:";STR$ Z;" SP:";STR$
 J;:C$(1)=D$:INPUT " ERSATZ: ";C$(1)
11020 IF C$(1)="***"BEEP 2,99,99:END
11030 L1=LEN C$(1)
11040 C$(0)=LEFT$ (A$(Z),J-1):C$(2)=RIGH
T$ (A$(Z),L-J-DS+1)
11050 L0=LEN C$(0):L2=LEN C$(2)
11060 IF L0+L1+L2<=ZBLET A$(Z)=C$(0)+C$(
1)+C$(2):RETURN
11070 CLS :BEEP 3:PAUSE "ZU LANG":GOTO 1
1010
```



Kap. VIII:

BEARBEITEN DES TEXTES: ERWEITERTE FUNKTIONEN

1. Formatieren

a) BASIC-Lösungen

Im letzten Kapitel war gefragt, **was** für ein Text letztendlich im Speicher stehen soll. In diesem geht es darum festzulegen, **wie** er dort, vorbereitet für den Ausdruck auf Papier, vorhanden sein soll. In dieser Abteilung finden Sie ein Listing des **Moduls FORMATIEREN** für FormSTAR. Seine Aufgabe ist es, wesentlich **ungleiche Zeilenlängen**, wie sie etwa bei der Korrektur entstehen, so auszugleichen, daß immer ein **kompakter Text** entsteht.

Sie erreichen dieses Modul nach <F> im Hauptmenü. Nun werden Anfangs- und Schlußzeile des Teils vom bestehenden Text eingegeben, den es zu formatieren gilt. Weiter wird die **Zeilenbreite** vorgegeben. Dann beginnt die Routine ihre Arbeit nach einem **einfachen Prinzip:** Es werden immer 2 Zeilen miteinander verglichen, und zwar die **aktuelle Zeile** mit der **Folgezeile.** Von der Folgezeile werden immer noch soviele ganze Wörter in die aktuelle Zeile übernommen, wie hier hineinpassen. Dabei wird die Nummer der aktuell bearbeiteten Zeile auf dem Display eingeblendet.

Im einzelnen **geschieht folgendes:** Zunächst wird die Routine 'END-ZL' angesprungen, wo die letzte beschriebene Zeile festgestellt wird. Sie wird u.a. gebraucht, um Fehleingaben auszuschließen. Dann geht es in die **äußere Schleife** 'NEXT', zu der immer bei Aufnahme einer neuen aktuellen Zeile zurückgeführt wird. Ist entweder die aktuelle oder die Folgezeile leer, wird eine Unterroutine '1-LOE' ange-

```
 4999 REM ******************************
 5000 REM *   FORMATIEREN                *
 5001 REM ******************************
 5002 REM
 5005 "FORMAT"
 5010 CLS :CURSOR 23:PRINT "Ent";:GOSUB "INV":CURSOR 0
 5012 ZB=DB:PRINT "Zei.breite? (36-";STR$ DB;")  ":CUR
SOR 20:INPUT ZB
 5014 IF ZB<36OR ZB>DBBEEP 1:CLS :GOTO 5010
 5016 CLS :CURSOR 23:PRINT "Ent";:GOSUB "INV":CURSOR 0
 5020 PRINT "VonZeile:   bisZeile:";
 5030 CURSOR 9:A$="":INPUT A$:ZA=VAL A$
 5035 CURSOR 21:B$="":INPUT B$:ZZ=VAL B$
 5037 IF A$=""OR B$=""GOTO "ME"
 5038 IF ZA>ZZBEEP 1:GOTO 5010
 5040 GOSUB 9500:GOSUB "END-ZL"
 5041 IF ZA>RV-1BEEP 1:GOTO 5010
 5042 IF ZZ>RV-1LET ZZ=RV-1
 5045 Z=ZA
 5049 REM ------------------
 5050 "NEXT"CURSOR 24:PRINT STR$ Z
 5052 IF Z>RV-1GOTO "ME"
 5053 IF A$(Z+1)=""LET Z=Z+1:GOSUB "1-LOE":Z=Z-1:ZZ=ZZ
-1:GOTO 5053
 5055 IF A$(Z)=""GOSUB "1-LOE":ZZ=ZZ-1:GOTO "NEXT"
 5057 IF A$(Z+1)=CHR$ &7FLET Z=Z+2:GOTO "NEXT"
 5058 IF LEFT$ (A$(Z+1),1)=CHR$ &7FLET Z=Z+1:GOTO "NEX
T"
 5069 REM ------------------
 5070 LA=LEN A$(Z)
 5072 IF LA>ZBGOSUB 5700:Z=Z+1:GOTO "NEXT"
 5075 L1=LEN A$(Z+1)
 5077 IF L1>ZBGOSUB 5750:GOTO "NEXT"
 5079 IF L1<ZB-LATHEN "KOMPR"
 5080 B$(0)=LEFT$ (A$(Z+1),ZB-LA)
 5090 W=LEN B$(0)-1:IF W<=0THEN 5250
 5092 U$=RIGHT$ (A$(Z),1)
 5094 IF U$=" "OR (ASC U$>159AND ASC U$<164)LET W=W+1
 5105 J=W
 5109 REM ------------------
 5110 "INN"B$=MID$ (B$(0),J,1)
 5120 IF B$=" "OR (ASC B$>159AND ASC B$<164)LET W=J:GO
```

```
TO "EX"
 5130 J=J-1:IF J<=1GOTO 5250
 5140 GOTO "INN"
 5199 REM ------------------
 5200 "EX"C$(0)=LEFT$ (A$(Z+1),W)
 5210 V$=RIGHT$ (A$(Z),1):W$=LEFT$ (B$(0),1)
 5212 IF ASC V$>159AND ASC W$<164THEN 5222
 5214 IF ASC W$>159AND ASC W$<164THEN 5222
 5220 A$(Z)=A$(Z)+" "+C$(0):GOTO 5224
 5222 A$(Z)=A$(Z)+C$(0)
 5224 LA=LEN A$(Z):IF RIGHT$ (A$(Z),1)=" "LET A$(Z)=LE
FT$ (A$(Z),LA-1)
 5230 A$(Z+1)=RIGHT$ (A$(Z+1),L1-W)
 5250 Z=Z+1:IF Z>ZZGOTO "ME"
 5270 GOTO "NEXT"
 5299 REM ------------------
 5300 "KOMPR"W$=RIGHT$ (A$(Z),1)
 5310 IF ASC W$>159AND ASC W$<164LET D$(0)=A$(Z):GOTO
5330
 5320 D$(0)=A$(Z)+" "
 5330 A$(Z)=D$(0)+A$(Z+1):A$(Z+1)="":GOTO "NEXT"
 5599 REM ------------------
 5600 "CRACK"M=LEN D$(0):M1=M
 5610 U$=MID$ (D$(0),M,1)
 5620 IF U$=" "OR (ASC U$>159AND ASC U$<164)LET M1=M:G
OTO 5650
 5630 M=M-1:IF M=0GOTO 5650
 5640 GOTO 5610
 5650 D$(0)=LEFT$ (D$(0),M1)
 5660 D=LEN D$(0):IF RIGHT$ (D$(0),1)=" "LET D$(0)=LEF
T$ (D$(0),D-1)
 5670 RETURN
 5699 REM ------------------
 5700 Z=Z+1:GOSUB "1-FREI":ZZ=ZZ+1:Z=Z-1:D$(0)=LEFT$ (
A$(Z),ZB):GOSUB "CRACK"
 5710 A$(Z+1)=RIGHT$ (A$(Z),LA-M1):A$(Z)=D$(0):RETURN
 5750 Z=Z+2:GOSUB "1-FREI":ZZ=ZZ+1:Z=Z-2:D$(0)=LEFT$ (
A$(Z+1),ZB):GOSUB "CRACK"
 5760 A$(Z+2)=RIGHT$ (A$(Z+1),L1-M1):A$(Z+1)=D$(0):RET
URN
 5995 REM
```

sprungen, wo der Text um eine Zeile aufrückt.

Nun fragt die Routine ab, ob am Anfang einer Zeile 1 oder mehrere **'inverse Blöcke'** (CHR$&7F) stehen. Dieses Sonderzeichen wird von dem Programm DEUZEI zur Verfügung gestellt (Kap.II.3). Auf dem Papier wird dieses Zeichen als Leerzeichen wiedergegeben. Wir benutzen es daher zum **Einrücken** einer Zeile um 2 oder mehrere Stellen. Steht in einer Zeile aber **nur 1** 'inverser Block', so markiert dieses Zeichen eine Leerzeile, die beim Formatieren erhalten, d.h. übersprungen, werden soll. Auf diese Weise werden **Absätze** berücksichtigt. Denken Sie daran bei der Texteingabe!

Dann werden die **Länge LA der aktuellen Zeile** (A-Zeile) und die **Länge L1 der Folgezeile** (F-Zeile) festgestellt und mit der **gewählten** Zeilenbreite (ZB), die kleiner oder gleich der **dimensionierten** Zeilenbreite (DB) sein darf, verglichen. Ist die ZB hier nun größer als die Zeilenlänge der A-Zeile (LA) , werden in der **Routine 'CRACK'** zunächst 2 Zeilen daraus gemacht. Sind LA + L1 kleiner als ZB, wird in der **Routine 'KOMPR'** alles auf 1 Zeile gebracht.

Der **wesentliche Schritt** folgt jetzt: Von der (um 1 verminderten) Anzahl der Zeichen der F-Zeile, die noch in die A-Zeile passen würden, wird, von rechts kommend, das 1. Leerzeichen gesucht. Das geschieht in der **inneren Schleife** 'INN'. An dieser Stelle ist natürlich gerade ein Wort beendet. Ist das Leerzeichen gefunden, erfolgt ein Sprung nach 'EX'. Hier 'weiß' die Routine, wieviel Zeichen (W) der F-Zeile noch in die A-Zeile passen ohne 'Wort-Bruch'; d.h. es werden **nur ganze Wörter** übernommen, der Rest verbleibt in der F-Zeile.

Das Format-Modul berücksichtigt auch **'Farb-**

steuerzeichen' (siehe nächtes Kap.), was sie etwas kompliziert erscheinen läßt. Diese Steuerzeichen werden in diesem Modul im Prinzip **wie normale Wortzwischenräume** behandelt. Z.B. erfolgt der Sprung von 'INN' nach 'EX' ebenfalls nach Auftreten eines Steuerzeichens, da auch sie Worte voneinander trennen.

b) Maschinensprache-Lösung für PC 1500(A)

Eigentlich sollte der Leser und Anwender von FormSTAR für eine solch phantastische Sache wie der Formatierung **etwas Geduld** mitbringen, aber: der Mensch neigt wie beim Auto so auch beim Computer zum Geschwindigkeitsrausch. Also muß wieder ein Mapro her, das den am stärksten **zeitkritischen** Teil des Moduls beschleunigen kann. Dieser kritische Teil wird von Manipulationen der A-Zeile mit der F-Zeile und insbesondere der Schleife 'INN' gebildet.

```
41C0   2A AE 77 D0 84 AE 77 D1 04 AE 77 D2   62 6E FF 8B 57 FD A8 FD 88 62 62 62
41D8   A5 77 D0 FD CA B5 20 07 8B 09 44 88   08 B5 20 41 B5 00 0E FD 0A FD 2A B5
41F0   00 07 8B 05 44 88 08 8E 2F A5 77 D0   DF 20 AE 77 D3 28 A5 77 D0 A0 2A AE
4208   77 D4 A5 77 D1 08 A5 77 D2 0A A5 77   D0 FD CA 62 62 6E FF 8B 0B 24 FD CA
4220   B5 20 07 8B 05 46 88 08 F9 9A FD A8   A5 77 D1 08 A5 77 D2 0A A5 77 D3 FD
4238   CA B5 20 41 FD 6A A4 AE 77 D5 24 AE   77 D6 A5 77 D1 08 A5 77 D2 0A A5 77
4250   D0 FD CA 84 AE 77 D8 04 AE 77 D9 A4   18 24 1A 94 AE 77 DA 14 AE 77 DB FD
4268   2A FD A8 62 6E FF 8B 03 F5 88 03 FD   2A FD A8 A5 77 D4 DF 20 2A AE 77 D7

4280   62 6E FF 8B 05 B5 00 51 88 05 A5 77   D8 08 A5 77 D9 0A FD 5A FD 2A FD A8
4298   24 DD FD CA A5 77 D0 20 2A 62 6E FF   8B 03 F5 88 03 FD 2A B5 00 51 88 05
42B0   F9 9A 03 01 82 28 A0 00 00 10 00 40   14 44 48 00
```

```
15000:REM * EINFACHE BASIC-ROUTINE Z
      UR ANWENDUNG DES FORMAT-MAPROS
15010:FM=&41C0:REM * DAS MAPRO IST F
      REI VERSCHIEBBAR
15020:CLS :PRINT "FMT v.Zei.:   b.Ze
      i.:"
15025:CURSOR 11:INPUT A$:ZA=VAL A$
15030:CURSOR 21:ZZ=0:INPUT Z$:ZZ=VAL
      Z$
15035:IF ZZ=0THEN "ME"
15040:Z=ZA
15050:"NAECHST"CURSOR 24:PRINT STR$
      Z
15055:IF A$(Z)=""GOSUB 16000:GOTO "N
      AECHST"
15060:LA=LEN A$(Z)
15065:L1=LEN A$(Z+1):D=LA+L1
15070:IF D<ZBLET A$(Z)=A$(Z)+" "+A$(
      Z+1):A$(Z+1)=""
15075:IF A$(Z+1)=""LET Z=Z+1:GOSUB 9
      800:Z=Z-1:GOTO "NAECHST"
15080:CALL FM,A$(Z)
15085:Z=Z+1
15090:IF Z=ZZ+1GOTO "ME"
15095:GOTO "NAECHST"
15500:REM ----------
16000:IF Z>R-1RETURN
16010:FOR Q=ZTO R-1
16020:A$(Q)=A$(Q+1)
16030:NEXT Q:A$(R)="":RETURN
```

Die abgedruckte **Maschinen-Routine** berücksichtigt leider (noch nicht) die Farb-Steuerzeichen, kann also nur für Text genutzt werden, der nicht farbig ausgedruckt werden soll. Sie ist eingebettet in einem BASIC-Programm, das darum auch einfacher als dasjenige im letzten Abschnitt gehalten ist. Mit den erwähnten Einschränkungen ist jenes durch das vorliegende ersetzbar.

c) Umbruch (Zeitungsspalten)

Von unserem **komfortablen** Formatierungs-Modul zum **mehrspaltigen Druck**, wie er etwa **in Zeitungen** anzutreffen ist, ist es nur ein kleiner Schritt. Die Textzeilen müssen dazu nur so kurz umformatiert werden, daß **2 oder mehrere** davon in eine Reihe passen. Notwendig ist noch eine **Blocksatz-Routine**, die im nächsten Kapitel, Absatz 1b, behandelt wird. Zumindest **längere Wörter** müssen in Zeitungsspalten aber getrennt werden, dazu mehr im nächsten Abschnitt. Beim Thema Drucken wird auch der **mehrspaltige Druck** behandelt werden. Doch zunächst müssen sie hier passend **kurze Zeilen** formatieren unter Berücksichtigeng der Zwischenräume bei den Spaltenblöcken.

Ein **netter Scherz** ist es, die Zeilenlängen beim Formatieren variabel, d.h. abhängig von einer eine Kurve definierenden Funktion, zu gestalten. Dazu muß nur für jede A-Zeile die Zeilenbreite ZB neu errechnet werden.

2. **Silbentrennung**

Die gewohnte Weise, annähernd gleiche Zeilenlängen zu bekommen, ist die Silbentrennung am Zeilenende. Diese stellt für Computer, die auf **einige Megabyte RAM** zugreifen können, kein

entscheidendes Hindernis dar, da passende Programme auf ein **Lexikon** zurückgreifen können, in der **alle Trennmöglichkeiten** seiner Wörter vorhanden sind. Damit wird eine fast 100-prozentige **automatische Silbentrennung** möglich.

Bei **beschränktem Speicherplatz** nimmt eine Routine zunächst eine **Wortanalyse** vor, als deren Ergebnis das Programm nach bestimmten Regeln und Berücksichtigung diverser Ausnahmen **Trennvorschläge** macht. Diese stimmen oft mit einer erstaunlich hohen **Trefferwahrscheinlichkeit** mit den richtigen Trennungen überein. Eine Methode besteht darin, von den Wörtern **Vokal-Konsonannten-Masken** anzufertigen, d.h. es interessiert hier nur noch, ob ein Buchstabe Vokal oder Konsonannt ist, innerhalb einer **Gruppe von 3 Buchstaben**. Z.B. bedeutet '1 0 0' eine Folge von einem Vokal und 2 folgenden Konsonannten. Ein Wort, in dem diese Gruppe vorkommt, würde (i.d. Regel) vor dem 2. Konsonanten getrennt werden.

Beispiel:	KIN-DER
Maske:	010 010

Bei **Pocketcomputern** kommt noch das Problem dazu, eine geeignete Darstellungsweise des Trennprozesses auf dem **Display** zu finden, was zu einem erheblichen Programmieraufwand führt und leider den Rahmen des Buches sprengt. (In 2 Versionen seines **Textprogrammes 'MeTRO'** hat der Verfasser eine Silbentrennung realisiert.)

Eine **einfache, manuelle Silbentrennung** für das 1-Zeilen-Display arbeitet nach folgendem Prinzip: Zu sehen ist jeweils das Ende der **aktuellen Zeile (A-Zeile)** auf der linken Seite. Durch einen Strich abgetrennt sieht man auf der rechten Displayhälfte die **Folgezeile (F-Zeile).** Mittels der Cursortasten können nun die **mittelständigen Buchstaben** der A-Zeile und

der F-Zeile **überwechseln**. Auch kann ein Binde-Strich zugefügt werden. Während diese Operationen auf dem **Display nur optisch** stattfinden, werden schließlich durch <ENTER> die entsprechenden Stringmanipulationen wirklich ausgeführt. Sodann wird die F-Zeile zur A-Zeile, so daß wir von ihr nun wieder das Zeilenende sehen und so folgt.

3. Weitere Möglichkeiten der Textverarbeitung

Wer einmal angefangen hat, ein Textverarbeitungsprogramm zu schreiben, kann davon für lange Zeit gefangen genommen werden: immer gibt es Verbesserungen, neue Anwendungen, erweiterte Funktionen, schnellere und elegantere Wege. Stets stellt dies bei Pocketcomputern aber auch einen Kampf mit dem knapp werdenden Speicherplatz dar.

Vieles könnte noch eingebaut werden:

- eine Adressenverwaltung für Serienbriefe
- Textbausteine mit Standardformulierungen
- eine Fußnotenverwaltung
- ein Taschenrechner für Kalkulationen
- Benutzer-Schnittstellen zu anderen Pgmn
- u.s.w.

In der Konsequenz erhält man ein **'Integriertes Pgrogramm'** mit Text, Graphik, Kalkulation und Datenverwaltung. Neuerdings wandeln sich Textprogramme zu **'Desk-Top-Publishing'-Systemen** aus, wovon der Leser des Schlußkapitels auch eine Prise mitbekommt. Jedoch verfolgt der Anwender eines auf seinem Pocket laufenden Textprogrammes meist harmlosere Ziele, wie etwa die **mobile Datenerfassung,** und dafür reichen die in diesem Buch behandelten Programme und Routinen allemal.

Kap. IX:

TEXTGESTALTUNG UND DRUCK

1. Vor Ausdruck

a) Rechtsbündig abschließen und zentrieren

Zu guter Letzt will der Leser das Ergebnis seiner Mühen, den fertigerstellten Text, schwarz auf weiß oder in unserem Fall auch bunt, nach Hause tragen. Jedoch den letzten Schliff bekommt das **Textbild** erst jetzt während des Druckens. Die in diesem Modul vorgenommenen **Gestaltungen** einer Textzeile verändern Text im Speicher nicht, sondern **wirken nur auf die aktuelle Druckzeile**. Ein Vorteil dieser Methode ist, daß dadurch der auf dem Display lesbare Text nicht unnötig mit **Steuerzeichen** belastet wird.

Eingearbeitet in FormSTAR sind u.a. **rechtsbündiger Abschluß** und **Zentrieren**. Ein normaler Text (im sog. **Flattersatz**) ist **linksbündig** abgeschlossen, hat also am linken Rand eine glatte Begrenzung. Es ist nun sehr einfach, den glatten Rand an die andere Seite zu verlegen. Dazu muß man nur, wie in der **Routine 'RECHTS'**, um soviele Stellen einrücken, wie die Differenz aus gewählter Zeilenlänge ZB und Länge der aktuellen Zeile LA beträgt. Auch die Benutzung von LPRINT TAB wäre möglich.

Ein interessanter Effekt läßt sich mit dem Zentrieren erreichen. Hierbei sind der **linke und der rechte Rand gleich groß**. Genau ist das aber nur dann der Fall, wenn die Zeichenanzahl des Gesamtrandes eine gerade Zahl ist.

b) Blocksatz

Eine gute Möglichkeit, ein **ruhiges Schriftbild** zu bekommen, stellt der Blocksatz dar. Dabei werden soviele Leerstellen in die Textzeile eingestreut, daß **gleichlange Zeilen** entstehen. Die hierbei angesteuerte **Routine 'BLSTZ'** berücksichtigt auch **Farb-Steuerzeichen** (siehe nächster Punkt). Ihr **Arbeitsprinzip** besteht darin, daß sie, von links kommend, ein Leerzeichen (bzw. Farb-Steuerzeichen) sucht und dann ein weiteres Leerzeichen einfügt, sofern damit die Zeilenlänge nicht größer als die gewählte Zeilenlänge wird. Hat die Zeile B$(1) die Maximallänge ZB erreicht, erfolgt der Rücksprung in die Druckroutine, wo mit D$(0)=B$(1) die **Zeile als Druckzeile** zur Verfügung gestellt wird. Bei relativ kurzen Zeilen erfolgt ein **weiterer Durchlauf** durch die Routine 'BLSTZ'. Zeilen, die kürzer als 4/5*ZB sind, bleiben sinvollerweise unbearbeitet.

Ein (fast) völlig ebenmäßiges Schriftbild wird durch **Diagolnalschrift**, die viele Matrixdrucker heute schon eingebaut haben, erreicht. Auf einem Plotter können wir sie hervorbringen durch eine **Feinsteuerung der Druckpostionen** (z.B. GLCURSOR-Positionen beim PC 1500(A)).

c) Verwendung von Steuerzeichen für Farbe etc.

Eine hervorragende Möglichkeit, Textteile herauszustellen, besteht im **Buntdruck**. Bei FormSTAR kann die Farbe **wortweise gewechselt** werden. Für diesen Zweck werden in die Wortzwischenräume anstelle der Leerzeichen Steuerzeichen gesetzt, entweder schon bei der Texteingabe oder durch überschreiben im LESE-Modus. Sie gelten jeweils für das folgende Wort und müssen ohne Zwischenraum (SPACE) direkt

vor diesem stehen. FormSTAR benutzt für diesen Zweck die frei definierten Zeichen mit den Code-Nummern 160 bis 163. Sie erscheinen auf dem Display als **inverse Ziffern**, die das Schriftbild nicht besonders stören, und werden auf dem Papier als Leerzeichen wiedergegeben. Ein weiteres Steuerzeichen, der **'inverse Block'** Code-Nummer 127), wurde schon beim Formatieren (VII.1a) abgehandelt.

2. DIN-A4-Simulation für CE-150

War es schon beim Display gelungen, die **hardwaremäßige Begrenztheit** des PC 1500(A)-Systems zu überwinden - das Mittel dazu waren Fenster, Lauf- und Fließschrift sowie ein vertikales Skrollen - so ist solches auch beim CE-150 zu erreichen. Dazu verwendet FormSTAR **folgende Methode:** Der Druck erfolgt in Papierlaufrich-

```
 1996 REM ********************************
 1997 REM *   DRUCKEN mit CE-150          *
 1998 REM ********************************
 1999 REM
 2000 "DRUCK-150"
 2002 C$="":INPUT "Grundfarbe(0-3): ";C$:C=VAL C$
 2005 IF C$=""THEN "ME"
 2008 IF C>3CLS :GOTO 2002
 2010 PRINT "Nor-0/Blo-1/Ztr-2/Rch-3:":BS=0:CURSOR 2
4:INPUT BS
 2012 IF BS>3CLS :GOTO 2010
 2013 CSIZE 2:S=0:GOSUB 9500:GOSUB "END-ZL"
 2014 REM ------------------
 2015 "STREI"FOR J=0TO 5
 2020 BEEP 2,99,99:T=TIME
 2025 U$=INKEY$ :IF U$=CHR$ 13TEXT :GOTO "ME"
 2030 IF TIME >=T+0.0004THEN 2040
 2035 GOTO 2025
 2040 TEXT :LF 4
 2045 GRAPH :ROTATE 1:GOSUB "VRAND":H=-1
 2050 GLCURSOR (210,0):COLOR C:LPRINT "-"
 2054 REM ------------------
```

```
 2055 FOR Z=S+6*J+ATO S+6*J+5-B
 2060 IF Z>RV+1THEN 2090
 2065 H=H+1:COLOR C:F=0
 2070 IF Z>1LET O$=RIGHT$ (A$(Z-2),1):GOSUB 2250
 2071 IF BS=1LET B$(1)=A$(Z-1):GOSUB "BLSTZ":D$(O)=B$(
1):GOTO 2075
 2072 IF BS=2LET B$(1)=A$(Z-1):GOSUB "ZENTR":D$(O)=B$(
1):GOTO 2075
 2073 IF BS=3LET B$(1)=A$(Z-1):GOSUB "RECHTS":D$(O)=B$
(1):GOTO 2075
 2074 D$(O)=A$(Z-1)
 2075 GLCURSOR (G,O):GOSUB "LI-DRU"
 2080 E(H,O)=F:E(H,1)=P:G=G-39
 2085 NEXT Z
 2090 GOSUB 2290:H=-1
 2095 REM ------------------
 2100 FOR Z=S+6*J+ATO S+6*J+5-B
 2105 IF Z>RV+1THEN 2130
 2110 H=H+1:COLOR C
 2115 GLCURSOR (G,-E(H,1)*12):F=E(H,O):COLOR F
 2116 IF BS=1LET B$(1)=A$(Z-1):GOSUB "BLSTZ":D$(O)=B$(
1):GOTO 2120
 2117 IF BS=2LET B$(1)=A$(Z-1):GOSUB "ZENTR":D$(O)=B$(
1):GOTO 2120
 2118 IF BS=3LET B$(1)=A$(Z-1):GOSUB "RECHTS":D$(O)=B$
(1):GOTO 2120
 2119 D$(O)=A$(Z-1)
 2120 GOSUB "RE-DRU":G=G-39
 2125 NEXT Z
 2129 REM ------------------
 2130 GOSUB "TRENN"
 2135 GLCURSOR (0,-900):SORGN
 2140 IF Z>RV+1TEXT :GOTO "ME"
 2145 NEXT J
 2150 S=S+33:GOTO "STREI"
 2159 REM ------------------
 2160 "LI-DRU"E$=LEFT$ (D$(O),1):P=36
 2162 IF E$=" "COLOR C:P=35:GOTO 2175
 2164 IF E$=CHR$ 160COLOR O:P=35:GOTO 2175
 2166 IF E$=CHR$ 161COLOR 1:P=35:GOTO 2175
 2168 IF E$=CHR$ 162COLOR 2:P=35:GOTO 2175
 2170 IF E$=CHR$ 163COLOR 3:P=35:GOTO 2175
```

```
2172 LPRINT E$
2175 IF D$(0)=""RETURN
2180 FOR E=2TO 36:E$=MID$ (D$(0),E,1)
2182 IF E$=" "LET F=C:COLOR C:GOTO 2195
2184 IF E$=CHR$ 160LET F=0:COLOR 0:GOTO 2195
2186 IF E$=CHR$ 161LET F=1:COLOR 1:GOTO 2195
2188 IF E$=CHR$ 162LET F=2:COLOR 2:GOTO 2195
2190 IF E$=CHR$ 163LET F=3:COLOR 3:GOTO 2195
2195 LPRINT E$
2197 NEXT E
2198 RETURN
2199 REM ------------------
2200 "RE-DRU"FOR E=37TO LEN D$(0):E$=MID$ (D$(0),E,1)
2202 IF E$=" "COLOR C:GOTO 2215
2204 IF E$=CHR$ 160COLOR 0:GOTO 2215
2206 IF E$=CHR$ 161COLOR 1:GOTO 2215
2208 IF E$=CHR$ 162COLOR 2:GOTO 2215
2210 IF E$=CHR$ 163COLOR 3
2215 LPRINT E$
2217 NEXT E
2218 RETURN
2250 IF O$=CHR$ 160COLOR 0:RETURN
2252 IF O$=CHR$ 161COLOR 1:RETURN
2254 IF O$=CHR$ 162COLOR 2:RETURN
2256 IF O$=CHR$ 163COLOR 3:RETURN
2258 RETURN
2279 REM ------------------
2280 "TRENN"G=196:COLOR 0
2282 FOR I=0TO 5
2284 GLCURSOR (G,-840):LPRINT "!":G=G-39
2286 NEXT I
2288 RETURN
2289 REM ------------------
2290 "VRAND"G=196:A=0:B=0
2291 IF J=0LET A=1:G=157
2292 IF J=5LET B=2
2294 RETURN
2295 REM
```

```
2910 REM ------------------
2912 "BLSTZ"
2914 LA=LEN B$(1):IF LA<ZB-INT (ZB/6)RETURN
2916 IF ZB-LA<1RETURN
2918 REM ----------
2920 "FIND-SPC"FOR I=1TO ZB
2922 AG$=MID$ (B$(1),I,1)
2924 IF AG$=""GOTO 2932
2926 IF AG$=" "OR (ASC AG$>159AND ASC AG$<164)GOSUB "
DEHN"
2928 IF ZB-LA<1LET I=ZB
2930 NEXT I
2932 IF LA<ZBGOTO "FIND-SPC"
2934 RETURN
2936 REM ----------
2938 "DEHN"C$(0)=LEFT$ (B$(1),I-1)
2940 C$(1)=" "+RIGHT$ (B$(1),LA-I+1)
2942 B$(1)=C$(0)+C$(1):LA=LEN B$(1)
2944 BG$=MID$ (B$(1),I+1,1)
2946 IF BG$=" "OR (ASC BG$>159AND ASC BG$<164)LET I=I
+1:GOTO 2944
2948 RETURN
2960 REM ------------------
2962 "ZENTR"LA=LEN B$(1):RA=ZB-LA:LR=INT (RA/2):RR=RA
-LR
2964 LR$="":FOR I=1TO LR:LR$=LR$+" ":NEXT I
2966 RR$="":FOR I=1TO RR:RR$=RR$+" ":NEXT I
2968 B$(1)=LR$+B$(1)+RR$:RETURN
2970 REM ------------------
2972 "RECHTS"LA=LEN B$(1):LR=ZB-LA:LR$=""
2974 FOR I=1TO LR:LR$=LR$+" ":NEXT I
2976 B$(1)=LR$+B$(1):RETURN
2990 REM ------------------
2995 REM
```

tung (ROTATE 1) auf 6 **Papierstreifen**, die so lang sind wie ein DIN-A4-Blatt breit. Diese werden **überlappend** solcherart **zusammengesetzt** (geklebt), daß die Länge eines DIN-A4-Blattes erreicht wird. Dazu erscheint auf jedem Streifen zunächst ein kurzer Strich, bis zu dem die Überlappung erfolgen soll. Damit erreichen wir genau die Größe einer DIN-A4-Seite.

Da das Papier **nur bis zu 10 Zentimeter** zurückzieht (diese Sperre ist durch Eingriffe ins Betriebssystem zwar zu überwinden, aber die Positionierung erfolgt dann nicht mehr mit hinreichender Genauigkeit), werden auf jedem Streifen zunächst **36 Zeichen lange Halbzeilen** ausgedruckt. Dann erst werden die Zeilen vervollständigt.

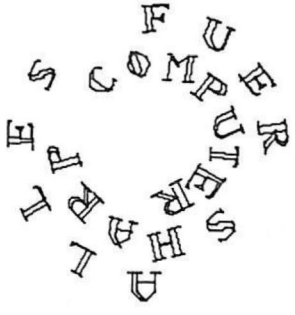
Wenn man den Druck durch **<BREAK>** stoppt, befindet sich der Rechner noch im **GRAPH-Modus.** Ebenfalls ist die **<MODE>-Taste** durch LOCK **gesperrt.** Um das zu vermeiden, ist die Möglichkeit vorgesehen, jeweils nach Ausdruck eines Streifens **regulär** aus dem Programm zu gehen. Dann nämlich ertönt ein **Signal**, wonach man für etwa **4 Sekunden** die Gelegenheit hat, den Druck durch <ENTER> zu beenden. Nach Ablauf dieser Frist beginnt andernfalls die Beschriftung eines neuen Streifens.

3. Textdruck mit größeren Plottern

Einen größeren Plotter haben die Rechner **MZ 700/800** zur Verfügung. Im Kleinstdruck lassen sich damit immerhin 80 Zeichen im Normaldruck auf eine Reihe bringen. Man kann hier wohl auf die obige Methode der DIN-A4-Simulation verzichten, die auch hier durchaus möglich wäre (siehe Kap. XI.3).

```
2296 REM ******************************
2297 REM *   DRUCKEN mit CE-515/6P    *
2298 REM ******************************
2299 REM
2300 "DRUCK-515"
2304 CLS :PRINT "VonZeile:   bisZeile:"
2306 CURSOR 9:K$="":INPUT K$
2307 K=VAL K$:IF K$=""GOTO "ME"
2308 IF K>RBEEP 1:GOTO 2304
2309 CURSOR 21:INPUT G$
2310 G=VAL G$:IF G<KBEEP 1:GOTO 2304
2311 IF G>RBEEP 1:GOTO 2304
2312 CLS :S$="                ":RA=0:INPUT
"Rand= ";RA
2313 REM ------------------
2314 GOSUB 9500
2315 OPN "LPRT":CONSOLE 0,0
2316 REM SYS SETCOM 1200,7,N,2:SETDEV PO:
OUTSTAT 0:CONSOLE 79
2317 CLS :SG=2:INPUT "SCHRIFTGROESSE :";S
G
2318 IF SG<1OR SG>15OR SG*ZB>158-RATHEN 2
312
2320 CLS :SF=0:INPUT "SCHRIFTFARBE :";SF
2322 IF SF>3THEN 2320
2323 CLS :AB=1:INPUT "Zei.abstand(1-9)= :
";AB
2324 IF AB<=0OR AB>96OTO 2323
2326 LPRINT CHR$ 27+"?"+CHR$ (&60+SG);
2327 LPRINT CHR$ 27+CHR$ (&30+SF);
2330 P=0:RZ=INT (112/(AB+SG-1))
2339 REM ------------------
2340 FOR Z=P*RZ+KTO (P+1)*RZ-1
2350 IF Z>GLPRINT CHR$ 27+"a":SETDEV :GOT
O "ME"
2355 LPRINT CHR$ 27+CHR$ (&30+SF);:IF Z>0
LET RI$=RIGHT$ (A$(Z-1),1):GOSUB 2900
2360 GOSUB 2500
2365 LPRINT "":LPRINT CHR$ 27+"a"
2367 IF AB>1LPRINT CHR$ 27+"?"+"a";:FOR I
=1TO AB-1:LPRINT CHR$ &0A;:NEXT I
2368 IF AB>1LPRINT CHR$ 27+"?"+CHR$ (&60+
SG);
2370 NEXT Z
2400 P=P+1:CLS :INPUT "SEITENWECHSEL AUSG
.?(J/N)";J$
2410 IF J$<>"J"SETDEV :GOTO "ME"
2420 GOTO 2340
2499 REM ------------------
2500 LPRINT CHR$ 27+"b"
2560 LPRINT "P";0$;
2580 FOR F=1TO LEN A$(Z):E$=MID$ (A$(Z),F
,1)
2600 IF E$=" "LPRINT "":LPRINT CHR$ 27;ST
R$ SF:LPRINT "P ";:GOTO 2690
2610 IF E$=CHR$ 161LPRINT "":LPRINT CHR$
27;"1":GOSUB 2800:GOTO 2690
2620 IF E$=CHR$ 162LPRINT "":LPRINT CHR$
27;"2":GOSUB 2800:GOTO 2690
2630 IF E$=CHR$ 163LPRINT "":LPRINT CHR$
27;"3":GOSUB 2800:GOTO 2690
2640 REM ----------
2650 IF E$=CHR$ 164GOSUB 2700:LPRINT "PA"
;:GOTO 2690
2655 IF E$=CHR$ 165GOSUB 2700:LPRINT "PO"
;:GOTO 2690
2660 IF E$=CHR$ 166GOSUB 2700:LPRINT "PU"
;:GOTO 2690
2665 IF E$=CHR$ 167GOSUB 2710:LPRINT "Pa"
;:GOTO 2690
2670 IF E$=CHR$ 168GOSUB 2710:LPRINT "Po"
;:GOTO 2690
2675 IF E$=CHR$ 169GOSUB 2710:LPRINT "Pu"
;:GOTO 2690
2680 IF E$=CHR$ 170GOSUB 2760:LPRINT "P "
;:GOTO 2690
2682 IF E$=CHR$ 127LPRINT "":LPRINT "P ";
:GOTO 2690
2685 LPRINT E$;
2690 NEXT F
2695 RETURN
2699 REM ------------------
2700 V=SG:W=SG*7
2701 LPRINT "":LPRINT "I";:LPRINT "R";V;"
```

```
,";W
 2702 LPRINT "J1,0,0,1,-1,0,0,-1"
 2703 VV=2*V
 2704 LPRINT "R";VV;",0"
 2706 LPRINT "J1,0,0,1,-1,0,0,-1"
 2708 LPRINT "H":RETURN
 2710 V=SG:W=SG*5
 2711 LPRINT "":LPRINT "I";:LPRINT "R";V;"
,";W
 2712 LPRINT "J1,0,0,1,-1,0,0,-1"
 2714 LPRINT "R";VV;",0"
 2716 LPRINT "J1,0,0,1,-1,0,0,-1"
 2718 LPRINT "H":RETURN
 2760 LPRINT "":LPRINT "I";
 2762 S1=0:T1=5*SG:S2=SG:T2=SG:S3=2*SG:T3=
0:S4=SG:T4=-SG:S5=0:T5=-SG
 2764 S6=-SG:T6=-SG:S7=SG:T7=-SG:S8=0:T8=-
SG:S9=-2*SG:T9=0
 2766 LPRINT "J";S1;",";T1;",";S2;",";T2;"
,";S3;",";T3;",";S4;",";T4;",";S5;",";T5;
 2768 LPRINT ",";S6;",";T6;",";S7;",";T7;"
,";S8;",";T8;",";S9;",";T9;
 2770 LPRINT "H":RETURN
 2799 REM ------------------
 2800 IF F=1LPRINT "P";:RETURN
 2810 LPRINT "P ";:RETURN
 2899 REM ------------------
 2900 IF RI$=CHR$ 161LPRINT CHR$ 27+"1";:R
ETURN
 2902 IF RI$=CHR$ 162LPRINT CHR$ 27+"2";:R
ETURN
 2904 IF RI$=CHR$ 164LPRINT CHR$ 27+"4";:R
ETURN
 2906 RETURN
```



Einen **echten DIN-A4-Plotter** besitzt das PC 1600-System, den **CE-1600P.** Dieser ist hervorragend ansteuerbar, abgesehen von seinem ruhigen Betrieb und seinem präzisen Arbeiten (siehe Kap. XI.1). Universell ist der **CE-516P,** Nachfolger des **515P,** der sich von seinem Vorgänger neben einem zusätzlichem DIP-Schalter vor allem durch die eingebaute Software unterscheidet. Diese bietet zusätzliche Graphikmöglichkeiten und Zeichensätze für den Betrieb an einem MZ 800 ebenso wie an einem IBM-Rechner.

Als Option für FormStar finden Sie in diesem Abschnitt eine **Druckroutine mit dem CE-515P.** Sie arbeitet völlig im **GRAPH-Mode,** da leider Deutsche Sonderzeichen im TEXT-Mode hier nicht existieren. Ebensogut kann aber auch ein CE-516P (im GRAPH-Mode) verwendet werden. Eine **analoge Druckroutine** findet der Besitzer eines **PC 1350/2500** in Kap. XI.2.

Unsere Routine arbeitet grundsätzlich wie beim CE-150, nur daß die Zeilen gleich vollständig ausgedruckt werden. **Wählbar** sind hier **Rand, Schriftgröße, Schriftgrundfarbe** (natürlich werden auch Farb-Steuerzeichen verarbeitet) und **Zeilenabstand.** Frei wählbar sind diese Parameter aber nur, soweit sie mit der vorgewählen Zeilenbreite (die beim Formatieren eingestellt wird) vereinbar sind. Am **Seitenende** stoppt der Druck, damit Sie ein neues Blatt einspannen können.

4. Textausgabe mit EP44 von BROTHER

Da viele Benutzer von Pocket-Computern auch die EP44 ihr eigen nennen, die neben einem hervorstechenden Schriftbild noch über eine **bidirektionale Schnittstelle (RS-232C)** verfügt, folgt dafür unten die Druckroutine als Option für FormSTAR.

Der linke Rand ist (im zulässigen Rahmen) einstellbar. Diese Routine ergänzt die Texteingabe-Routine aus Kap.V.5 und kann mit dieser zusammen auch als eigenständiges Programm verwendet werden.

```
60500 "L"REM -- Hier 2. Teil des Program
ms von Kap. V.5
60502 "TRANS/D"CLS :WAIT 0
60505 PAUSE "** AUSDRUCK PC ==>EP44 **":
WAIT
60510 PAUSE "PARA. EP44:1200,8,N,CR,8,Y"
:PAUSE :PAUSE
60520 INPUT "RAND= ";K$:RA=VAL K$
60530 SETCOM 1200,8,N,1:SETDEV PO:OUTSTA
T 0
60534 CLS :K$="":INPUT "Start ab Zeile ?
";K$
60536 ZA=VAL K$:IF ZA>RCLS :BEEP 1:GOTO
60534
60540 CLS :K$="":INPUT "Druck bis Zeile
?";K$:K=VAL K$
60545 IF K$=""BEEP 3:END
60547 CLS :K=K-1
60550 PRINT "TEXTAUSDRUCK !!"
60560 FOR Z=ZATO K
60567 FOR E=1TO 500:NEXT E
60590 LPRINT TAB RA;A$(Z)
60595 NEXT Z:BEEP 3:END
60600 REM RETURN
```

5. Verwendung von Thermo- und Matrixdruckern)

Statt der EP44 sind auch andere (Thermo-) Drucker mit einem RS-232C-Interface in ähnlicher Weise zu verwenden und ansteuerbar. Für den PC 1350 gibt es noch die Option des kleinen Thermodruckers **CE-126P** als einfache Lösung. Besitzer eines **PC 2500** sollten versuchen, die Routine in Kap.IX.3 abzuwandeln. Dazu müssen dort u.a. die Befehle, die mit der RS-232C zu tun haben, getilgt werden, da der **eingebaute Plotter direkt angesteuert** wird. Dagegen dürfte ein **CE-140P** (Sieben-Farb-Drucker) die obige Routine ziemlich direkt verwenden können, im Rahmen der gegebenen Druckbreite und natürlich ohne weitere Steuerzeichen mit nur 4 Farben. Alle sinnvollen Optionen sind ebenfalls mit dem **PC 1450** verwendbar, sofern man das Textprogramm für diesen Rechner angepaßt hat.

Kap. X: KOMMUNIKATION

1. Texte speichern und laden

Das reguläre Medium für die **Massenspeicherung** (von Daten, Programmen und Texten) ist bei den meisten Nutzern von Pocketcomputern immer noch der **Kassettenrecorder**. Hier folgen deshalb die die Module SPEICHERN und LADEN für FormSTAR. Bei ihrer Ansteuerung werden Sie zunächst aufgefordert, den Recorder herzurichten. Dann bitte <ENTER> drücken! Nun soll ein **Dateiname** eingegeben werden. Wird hier einfach <ENTER> gedrückt, kommt man unmittelbar zum Hauptmenü zurück. Bei Eingabe eines Dateinamens wird im Modul SPEICHERN Ihr Text aufs Band geschrieben, nachdem vorher auch R (Zeilenzahl-1) und DB (definierte Zeilenbreite) aufgezeichnet wurden. Verwenden Sie nur gutes Bandmaterial!

Im Modul LADEN wird anfänglich genauso verfahren. Dann liest das Programm zuerst R und DB. Sie müssen sich nun vergewissern, daß der einzulesende Text mit genau **denselben Parametern** abgespeichert wurde, wie das Programm jetzt beim Textladen **dimensioniert** ist (im Hinblick auf R und DB) - andernfalls funktioniert das Laden nicht. Dazu erscheint auf dem Display eine **Meldung.** Stimmen die Werte nicht mit denen des Programms überein, muß das Programm abgebrochen werden (drücken Sie <N>) und dann nach <DEF><Z> mit den übereinstimmenden Werten **neu dimensioniert** werden. Spulen Sie nun das Band zurück und beginnen den Ladevorgang neu! Am besten vermerkt man sich die Werte auf dem beigegebenen Kassettenetikett.

2. Daten- und Textaustausch über RS-232C

Einige SHARP-Computer haben eine eingebaute **serielle (RS-232C-) Schnittstelle,** über die prinzipiell Daten mit allen Geräten ausge-

tauscht werden können, die eine Schnittstelle nach der gleichen (vollständigen) **Norm** haben. leider haben die Götter hier vor den Erfolg den Schweiß gesetzt. Vonnöten ist nicht nur der richtige **Pegel**, sondern auch das geeignete **Kabel**, die passenden **Stecker** und ein korrekt getimetes **'Handshaking'** (wer was wann Senden oder Empfangen kann). Die ausführliche Behandlung dieser Themen sprengt den Rahmen dieses Buches, aber manchmal erzielt man mit seinen Bordmitteln (sprich: Kabeln) überraschende Ergebnisse.

Der **Textaustausch** zwischen **PC 1500(A) und EP44** via CE-158 stellte auch schon eine solche **Kommunikation** dar. Hier folgt eine Routine, mit der Text vom PC 1350 über das CE-158 zum PC 1500(A) transferiert werden kann. Dafür müssen Sie, wenn Sie die Möglichkeit haben, in das Programm in Kap. XI.2 (für den PC 1350/2500) an geeignter Stelle folgende **Zeilen einfügen:**

```
"C"CLOSE:  OPEN  "300,N,8,1,A,C,&1A"
 PRINT  1A$(*)
```

Auf der **Empfängerseite** [PC 1500(A)] benutzen Sie das folgende Programm:

```
16000  "X"CLEAR:  SETCOM  300,8,N,1
16010  SETDEV: INPUT "R= ";R,"ZB= ";ZB:CLS:
       WAIT  0:  PRINT  "ICH  WARTE..."
16020  SETDEV  KI:  OUTSTAT  0
16030  DIM  A$(R)*ZB,  Y$(0)
16040  FOR  Z=0  TO  R
16050  INPUT%  Y$(*):  A$(Z)=  Y$(0)
16060  NEXT  Z:  END
```

In Zeile 16050 benutzen wir tatsächlich INPUT% (siehe dazu Handbuch zum CE-158). Starten Sie zunächst die Empfängerseite mit <DEF><X>, dann erst den **Sender** mit <DEF><C>! Andere Computer brauchen eine andere spezifische Anpassung.

```
 5996 REM ******************************
 5997 REM *   TEXT SPEICHERN           *
 5998 REM ******************************
 5999 REM
 6000 "SPEICHERN"
 6010 CLS :CURSOR 22:PRINT " Ent";:GOSUB "INV":CURSO
R 0
 6020 WAIT :PRINT "Recorder auf Aufnahme":WAIT 0
 6030 N$="":INPUT "Name d. Textdatei:";N$
 6035 IF N$=""THEN "ME"
 6040 GOSUB 9500
 6050 PRINT #N$;R,DB:PRINT #A$(*)
 6060 GOTO "ME"
 6995 REM
 6996 REM ******************************
 6997 REM *   TEXT LADEN               *
 6998 REM ******************************
 6999 REM
 7000 "LADEN"
 7010 CLS :CURSOR 22:PRINT " Ent";:GOSUB "INV":CURSO
R 0
 7020 WAIT :PRINT "Record. auf Wiedergabe":WAIT 0
 7030 N$="":INPUT "Name d. Textdatei:";N$
 7035 IF N$=""THEN "ME"
 7040 GOSUB 9500
 7050 INPUT #N$;R,DB
 7052 PRINT "DIM=";R;"/Zei.br=";DB;"/OK?(J/N)"
 7054 K$=INKEY$ :IF K$=""THEN 7054
 7056 IF K$<>"J"THEN "ME"
 7058 INPUT #A$(*)
 7060 GOTO "ME"
 7995 REM
 7996 REM ******************************
 7997 REM *   ROUTINEN                 *
 7998 REM ******************************
 7999 REM
 8000 "END-ZL"FOR Q=RTO OSTEP -1
 8020 IF A$(Q)<>""LET RV=Q:RETURN
 8030 NEXT Q:RETURN
 8035 REM ----------
 9000 REM *  MAPRO fuer Zeile 1
 9005 "ADR"AD=STATUS 2-STATUS 1+10
```

```
 9006 POKE AD,&68,&78,&6A,&4D,&FD,&62,&25,&BD,&FF,&2
E,&88,&06,&6C,&77,&93,&0E,&9A
 9010 "INV"CALL AD:RETURN
 9499 REM ----------
 9500 CLS :CURSOR 5:PRINT ">> formSTAR-87 <<":RETURN

 9699 REM ----------
 9700 "1-FREI"IF RV=RBEEP 3:RETURN
 9705 FOR Q=RVTO ZSTEP -1
 9710 A$(Q+1)=A$(Q)
 9720 NEXT Q
 9730 A$(Z)="":RV=RV+1:RETURN
 9799 REM ----------
 9800 "1-LOE"IF Z>RV-1GOTO "ME"
 9810 FOR Q=ZTO RV-1
 9820 A$(Q)=A$(Q+1)
 9830 NEXT Q
 9840 A$(RV)="":RV=RV-1:RETURN
 9999 REM ----------
10000 "KREUZ"CURSOR 2:GPRINT "001C08223622081C00";:R
ETURN
```

Kap. XI: TEXTSYSTEME FÜR WEITERE RECHNER

1. Besonderheiten beim PC 1600

Wenn man BASIC-Programme des PC 1500(A) auf dem PC 1600 laufen lassen will (natürlich im MODE 1), muß man mit einigen Überraschungen rechnen, wie im folgenden: So meldet der PC 1600 oft **ERROR 15** bei komplizierten Programmzeilen (jedenfalls bei dem Exemplar des Verfassers), wo der PC 1500(A) die Zeile noch verarbeitet. Das gleiche Problem kann übrigens auch beim PC 1350 auftreten. Zur Abhilfe muß man die Programmzeilen entkoppeln, d.h. einfacher machen. Z.B. sind bei FormSTAR die durch die **Logik-Funktion OR** verbundenen bedingten Sprunganweisungen (etwa Zeile 1080) durch 2 Programmzeilen ohne OR-Logik zu ersetzen:

```
IF  B$(I)="M"  LET  .......
IF  B$(I)="m"  LET  .......
```

Weiter bewirkt ein Tastendruck bei einer **INKEY$-Abfrage** die übernahme des Zeichens von einer nachfolgenden INPUT-Anweisung, was doch sehr stört. Um hier gegenzusteuern, kann man eine 2. INKEY$-Abfrage (mit Warte-Funktion) einbauen. Fügen Sie bei FormSTAR bitte ein:

```
305  L$=INKEY$:  IF L$<>"" THEN 305
```

So verfahren Sie allgemein bei INKEY$-Schleifen bzw. -abfragen. Schließlich funktioniert auch die **Laufschrift** bei FormSTAR nicht ohne Veränderungen. Ersetzen Sie hier Zeile 3701 durch die Zeilen 3702 bis 3704:

```
3702 CURSOR 7: PRINT " ";RIGHT$ (A$(Z),C);
3703  IF  C<18  PRINT  "  "
3704  RETURN
```

2. Ein komplettes Programm für den PC1350/2500

a) Installation des Systems

Für den Rechner PC 1350 finden sie hier ein vollständiges Programm, da eine direkte übernahme des 1500-Programmes nicht möglich ist. Dafür weisen die beiden Rechnertypen sowohl hardwaremäßig (z.B. Display) als auch softwaremäßig (z.B. BASIC) zu große Unterschiede auf. Lediglich das Modul Ändern ließe sich annähernd direkt übertragen. Für den Fall, daß man Maschinen-Routinen einbauen will, hat man es obendrein mit einem ganz andersartigem Prozessor zu tun als beim PC 1500(A).

Dennoch ist die Grundstruktur beider Programme sehr ähnlich, so daß der geneigte Leser an den entsprechenden Stellen auch die frühere Programmbeschreibung zu Rate ziehen kann, denn hier bleibt nur Raum für eine knappere Darstellung. Das Programm läuft natürlich auch auf dem PC 2500, wobei die Anpassung an den eingebauten Plotter (soweit gewünscht) allerdings dem Leser überlassen bleibt. Ohne die Möglichkeiten des PC 1600 ganz auszuschöpfen, läuft das vorliegende Programm mit einigen Abänderungen auch auf diesem Rechner. Beachten Sie dazu bitte die Ausführungen unter Punkt 1.

Zum Betrieb benötigen Sie mindestens eine 8-KB-Speichererweiterung (RAM CARD 8), wobei die Programmlänge von ca. 8,4 KB die Kapazität dieser Karte knapp überschreitet. Wollen Sie daher die CARD gelegentlich entfernen, ohne das Programm zu zerstören, müssen Sie einen für Sie weniger wichtigen Programmteil entfernen. Außerdem muß vorher MEM "C" eingegeben werden.

Auch dieses Programm wird wieder mit RUN <ENTER> oder DEF <Z> gestartet bei Erst- oder Neustart. Die Frage 'TEXT WIRKLICH LöSCHEN (J/J)' muß, auch beim ersten Lauf, mit J<ENTER> beantwortet werden, sonst kommen Sie nicht ins Textsystem. Dies ist ein Schutz vor versehentlichem Löschen eines schon erstellten Textes. Soll ein solcher Text erhalten bleiben, geht man mit <DEF><SPACE> ins Programm.

Nach jedem Neu- oder Erststart wird der Textspeicher des Programms neu dimensioniert. Zugelassen sind 16-78 Zeichen pro Zeile. Schließlich erscheint das Hauuptmenü.

b) Das Menüsytem

Durch das Programm geleitet ein verzweigtes Menüsystem, bestehend aus einem Hauptmenü und mehreren Untermenüs, deren Titel jeweils in einer

invers-beschrifteten Kopfzeile angezeigt ist. Das Hauptmenü hat folgendes Aussehen:

```
   ### HAUPTMENUE ###
<T>ext  <F>ormat  <R>ec.
<D>ruck <K>orrig. <P>lay
<L>esen <A>endern <E>nde
```

Durch Drücken des jeweiligen Anfangsbuchstabens einer Funktion gelangen Sie in das entsprechende Programmmodul. Man kann nun aus diesen Modulen sofort zurückkehren, ohne daß die angewählte Funktion ausgeführt wird. Dies geschieht durch einfaches <ENTER> mit Ausnahme des Moduls <A>endern. Hier führt Drücken von <M> zurück. Bei Rec. (Speichern) und Play (Laden) wird auf die Frage nach dem Dateinamen einfach <ENTER> eingegeben, wenn unmittelbare Rückkehr gewünscht wird. Auf diese Weise kann irrtümliches Anwählen einer Funktion korrigiert werden.

Aus dem Programm führt <E>, wobei der eingegebene Text erhalten bleibt, solange Sie weder CLEAR eingeben, noch die Variablen A$(Z) benutzen.

c) Die Texteingabe

Je nachdem, wie breit Sie die Zeilen gewählt haben, erscheinen an unterschiedlicher Stelle auf dem Display zwei schwarze Kästchen sowie oben links ein Fragezeichen. Sie dürfen Ihren Zeilentext nun höchstens so weit schreiben, bis das erste Kästchen verschwindet, weil dann die Zeile voll ist. Jetzt oder früher wird <ENTER> gedrückt, wonach Sie zur nächsten Zeile gelangen. Die Zeilennummer kann unten rechts in runden Klammern gelesen werden. Einfaches <ENTER> ohne Texteingabe führt hier nur dann zu einer Leerzeile, wenn die Zeile bislang unbeschrieben ist, andernfalls bleibt der Zeilentext erhalten.

Das Programm benutzt zur Textaufnahme eine INPUT-Funktion. Man darf daher keine (normalen) Anführungsstriche im Text benutzen, um den Rechner nicht zu irritieren. Sollte Ihnen dieses dennoch passieren, gehen Sie nach Löschen der Fehlermeldung einfach wieder mit <DEF><SPACE> ins Textsystem. Hinter den beiden Kästchen sehen Sie folgende Information:

```
<M m MM>
```

Sie bezeichnen die Möglichkeiten, zum Menü zurückzukommen. Nach M<ENTER> oder m<ENTER> als 1. und einzige Eingabe in einer Zeile kehren Sie zurück, ohne

daß der Buchstabe M bzw. m in den Text aufgenommen würde.

Die Deutschen Sonderzeichen werden auf dem Display im Modul Lesen und beim Druck mit den Plottern CE-515P/516P dargestellt. Hier bei der Texteingabe steht nur eine Hilfsmethode zur Verfügung: Statt eines Umlautes setzt man einen Doppelpunkt hinter einen zutreffenden Vokal. Man schreibt also 'scho:n' statt 'schön' und 'U:lzen' statt 'ülzen'. Für das 'ß' haben Sie die Wahl, entweder 's:' oder '@:' zu schreiben. Bequemer ist das erstere, doch prägnanter das zweite - Geschmackssache.

Wollen Sie den Doppelpunkt in normaler Bedeutung verwenden, muß vor ihm ein Leerzeichen (SPC) stehen. Eine weitere Bedeutung hat der Doppelpunkt beim Formatieren (siehe dort). Um ein Wort später farbig ausdrucken zu können, muß in dem Zwischenraum vor dem betreffenden Wort ein Farb-Steuerzeichen stehen. Die Farb- Steuerzeichen sind '<' für blau, '>' für grün und ' ' für rot. Der Buntdruck erfolgt im GRAPH-Mode der Plotter CE-515P/516P. Für eine normale Verwendung der fraglichen Zeichen ist nur der TEXT-Mode dieser Plotter geeignet (siehe unter Drucken).

d) Lesen und Korrigieren

Anders als beim Programm für den PC 1500(A) ist die Korrektur hier nicht in dem Lese-Modul integriert. Der nach <L> erreichte Lesemodus, der etwas gemächlich arbeitet, dient nur der Darstellung des Textes mit korrekten Deutschen Sonderzeichen. Die Information <M + -> hinter dem Kartenzeichen 'KREUZ' gibt die Einflußmöglichkeiten an. Zur Ansteuerung neuer Zeilen werden hier nicht die Cursortasten benutzt, sondern die Tasten <+> und <->. Drücken von <M> führt zum Hauptmenü.

Eine ähnliche Information über die Einflußmöglichkeiten findet man im Korrekturmodus, der nach <K> im Hauptmenü erreicht wird, diesmal hinter dem Kartenzeichen 'Pik-Herz':

<LEUM + ->

Die 3 neuen Funktionen hier sind <L>öschen, <E>infügen und <U>eberschreiben. Die Darstellung des Textes entspricht hier genau der Texteingabe. Eingebaut ist ein Editor mit Dauerfunktion (Autorepeat). Dieser BASIC-Editor (!) verwendet einen 'Cursor-Block', bestehend aus folgenden Tasten:

( )
/ :

Die beiden Klammertasten bewegen den Cursor vorwärts und rückwärts. Mit den beiden darunterliegenden Tasten läßt sich Cursor vertikal bewegen, falls der Text mehr als eine Reihe auf der Anzeige einnimmt. Mit Drücken von <*> setzt man an die Stelle des Cursors eine Marke.

Zum Löschen eines Textteils werden 2 Marken gesetzt. Nach Setzen der 2. Marke und Drücken von <L> wird der Text zwischen den Marken gelöscht, und anschließend erscheint der verkürzte Text auf dem Display.

Zum Einfügen wird nur 1 Marke gesetzt. Nach folgendem <E> geben Sie einen Zwischentext ein, der mit <ENTER> abzuschließen ist, und anschließend erscheint der verlängerte Text auf dem Display. Texteinfügen ist auch am Ende der Zeile möglich. Überschreitet der Gesamttext die dimensionierte Zeilenlänge, richtet das Programm selbsttätig 1 oder sogar 2 neue Zwischenzeilen ein, so daß kein Text verlorengeht.

Für das überschreiben von Text wird überhaupt keine Marke benötigt. Man fährt nur mit dem Cursor zur entsprechenden Textstelle und drückt dann <U>. An der Stelle des Fragezeichens beginnt dann das überschreiben, welches durch <ENTER> beendet wird. Der überschriebene Text darf nicht über den letzten Buchstaben des bestehenden Textes hinausführen, da die Aktion sonst wirkungslos bleibt.

e) Ändern und Formatieren

Nach <A> im Hauptmenü erscheint folgendes Untermenü:

```
      *** AENDERN ***

  <L>OESCHEN   <E>INFUEGEN
  <C>OPIEREN   <M>ENUE
```

In ganz ähnlicher Weise wie vorher beim PC 1500(A) handelt es sich hier um Manipulationen ganzer Zeilen bzw. Zeilenblöcke. Die Copy-Funktion indessen arbeitet etwas anders: Sofern n Zeilen kopiert werden sollen, werden die letzten n Zeilen quasi als 'Umschlagplatz' benutzt - hier darf also kein Text stehen! Auch hier ist Blocktransfer von Textzeilen möglich, wenn man nach dem Kopieren die Zeilen an der ursprünglichen Stelle löscht.

Beim Formatieren, zu erreichen nach <F> im Hauptmenü, handelt es sich darum, die Zeilen so weit als möglich mit Text aufzufüllen. Dies kann besonders nach

Korrekturen nötig sein. Sollen beim Formatieren Absätze und Leerzeilen erhalten bleiben, muß in einer betreffenden Zeile ein Doppelpunkt (:) stehen und sonst nichts. Beim Druck auf den Plottern 515P/516P im GRAPH-Mode wird ein an 1. Stelle stehender Doppelpunkt nicht mit ausgedruckt, so daß dort eine Leerzeile entsteht.

f) Drucken

Nach <D> im Hauptmenü sehen Sie folgendes Untermenü:

```
      *** DRUCKEN ***
  WELCHER DRUCKER ?
  GRAPH(1)  TEXT(2)  126P(3)
          :
```

Die Ziffern (1) und (2) beziehen sich auf die DIN-A4-Plotter CE-515P und CE-516P. Für den GRAPH-Mode(1) ist es gleich, welchen der beiden Plotter Sie benutzen. Die Deutschen Sonderzeichen (Ä,ö,Ü,ä,ö,ü,ß) werden in jedem Fall in der richtigen Schriftgröße geplottet.Die Farb-Steuerzeichen (<,>,) erscheinen im GRAPH-Mode nicht im gedruckten Text. Dagen wird in diesem Modus das einem Farb-Steuerzeichen direkt folgende Wort entsprechend farbig ausgedruckt. Ein Rand ist vorgebbar.

Im TEXT-Mode sind die beiden Plotter ebenfalls zu betreiben mit einheitlich wählbarer Druckfarbe. Die beim CE-516P möglichen Umlaute werden in diesem Programm im TEXT-Mode nicht genutzt. Der Ausdruck erfolgt also so, wie der Text bei der Eingabe erfaßt wurde. Daher werden auch die Zeichen '<')' ' hier mit ausgedruckt, also nicht als Steuerzeichen verwendet. Gleiches gilt für den Doppelpunkt.

Zum CE-140P ist zu sagen, daß dieser Printer/Potter ganz analog von den vorgenannten Rechnern anzusteuern ist, da er ebenfalls das serielle Interface benutzt. Eine Anpassung an die Routinen (1) und (2) sollte dem geneigten Leser leicht gelingen. Mit Drücken von (3) wird der kleine Matrixdrucker CE-126P angewählt. Der Druck erfolgt wie unter (2) ohne Umlaute. Sollte dieser Drucker gelegentlich nicht ansprechen, gehen Sie bitte aus dem Programm und geben CLOSE <ENTER> ein. Gehen Sie nun mit <DEF><SPC> ins Textsystem und wählen Sie die Druckart (3) erneut an! Eine Anpassung des eingebauten Plotters des PC 2500 an eine der 3 Druckroutinen sollte leicht möglich sein, wenn man alle schnittstellen-spezifischen Befehle wie OPEN und CLOSE usw. fortläßt. Über die RS-232C kann natürlich auch ein Din-A4-Plotter angesteuert werden.

```
 10:REM * MaTRO/ 1950(2500) V.2.1
 20:REM * COPYRIGHT: 1986 by WINFRIED MEYER: 208
    7 HASLOH
 30:"Z": CLS : WAIT 0: PRINT "": PRINT "  M a T
    R O / 1 3 5 0": PAUSE : PAUSE
 35:WAIT 100
 40:LINE (3,3)-(148,20),X,BF
 42:CLS : WAIT 0: PRINT "TEXT": INPUT "WIRKLICH
    LOESCHEN (J/N)";J$
 45:IF J$<>"J" BEEP 2: CLS : END
 50:CLEAR : CLS : WAIT 0
 60:PRINT "": PRINT "ZEILENBREITE :": INPUT K$
 62:ZB= VAL K$
 65:IF ZB<10 OR ZB>78 CLS : GOTO 60
 70:Q= INT (( MEM -1000)/ZB)
 75:IF Q>255 LET Q=255
 80:PRINT "ZEILENZAHL(Max=";Q;")";: INPUT X$
 85:R= VAL X$-1: CLS
 90:IF R<9 OR R>Q-1 THEN 80
100:DIM A$(R)*ZB,B$(8)*80,C$(8)*80,D$(8)*80
110:YY=1
200:" " IF YY<>1 BEEP 2: END
300:REM * M
305:CLS : WAIT 0: PRINT "   *** HAUPTMENUE ***":
    GOSUB 6700
310:PRINT "<T>ext  <F>ormat  <R>ec.": PRINT "<D>
    ruck <K>onfig. <P>laU"
320:PRINT "<L>esen <A>endern <E>nde"
330:K$= INKEY$ : IF K$="" THEN 330
340:IF K$="T" THEN 1000
350:IF K$="D" THEN 2000
360:IF K$="L" THEN 3000
370:IF K$="F" THEN 3500
380:IF K$="A" THEN 4500
390:IF K$="K" THEN 6000
400:IF K$="R" THEN 5000
405:IF K$="P" THEN 5500
410:IF K$="E" CLS : PRINT : PRINT "ENDE des PROG
    RAMMLAUFS": END
420:GOTO 330
1000:REM * T
1005:CLS : PRINT "*** TEXTEINGABE ***": GOSUB 6
    700: PRINT ""
1010:K$="": PRINT "TEXTEING. von Zeile";:
    CURSOR 68: INPUT K$
1015:IF K$="" THEN 300
1016:J= VAL K$
1018:IF J>R THEN 1005
1020:FOR Z=J TO R
1025:CLS : CURSOR (ZB): PRINT CHR$ 249; CHR$ 24
    9;"<M m MM>"
1028:USING "###"
1030:CURSOR 18,3: PRINT "(";Z;")": USING :
    GOSUB 1500
1032:IF A$(Z)="M" LET A$(Z)="": GOTO 300
1033:IF A$(Z)="m" LET A$(Z)="": GOTO 300
1035:IF D$(8)="MM" GOTO 300
1040:NEXT Z: CLS : PAUSE "TEXTENDE": GOTO 300
1500:D$(8)="": CURSOR 8: INPUT D$(8)
1505:IF D$(8)="" RETURN
1507:IF D$(8)="MM" RETURN
1510:A$(Z)=D$(8): RETURN
1999:REM * D
2000:CLS : PRINT "    *** DRUCKEN ***": GOSUB 6
    700: PRINT ""
2001:CURSOR 24: PRINT "WELCHER DRUCKER?": PRINT
    "GRAPH(1) TEXT(2) 126P(3)"
2002:CURSOR 9,3: INPUT ":";DR: ON DR GOTO 2004,
    2940,2930
2003:GOTO 300
2004:GOSUB 5850
2005:CLS : PRINT "VonZeile:   bisZeile:"
2006:K$="": CURSOR 9: INPUT K$
2007:IF K$="" CLOSE : GOTO 300
2008:K= VAL K$
2009:IF K>R GOTO 2025
2010:G$="": CURSOR 21: INPUT G$
2011:IF G$="" CLOSE : GOTO 300
2012:G= VAL G$: IF G>R THEN 2010
2013:CLS :S$=                  ": INPUT "RAND= "
    ;RA
2014:G$= LEFT$ (S$,RA)
2015:CLS :SG=2: INPUT "SCHRIFTGROESSE : ";SG
2016:IF G=0 CLOSE : GOTO 300
2017:IF SG<1 OR SG>15 OR SG*ZB>158-RA THEN 2013
2018:LPRINT CHR$ 27+"?"+ CHR$ (&60+SG);
2020:CLS :FB=0: INPUT "GRUNDFARBE:";FB
2022:IF FB<0 OR FB>3 THEN 2020
2024:LPRINT CHR$ 27+ CHR$ (&30+FB);
2030:P=0:RZ= INT (102/SG)
2040:FOR Z=P*RZ+K TO (P+1)*RZ-1
2050:IF Z>G LPRINT "": CLOSE : GOTO 300
2055:LPRINT CHR$ 27+ CHR$ (&30+FB); IF Z>0 LET
    RI$= RIGHT$ (A$(Z-1),1): GOSUB 2900
2060:D$(0)=A$(Z): GOSUB 7000: GOSUB 2500
2065:LPRINT "": LPRINT CHR$ 27+"a";
2070:NEXT Z
2200:P=P+1: CLS : INPUT "SEITENWECHSEL AUSGEFUE
    HRT ? (J/N)";J$
2210:IF J$<>"J" CLOSE : GOTO 300
2220:GOTO 2040
2490:REM *
2500:LPRINT CHR$ 27+"b";
2510:LPRINT "P";
2520:LPRINT D$;
2530:FOR F=1 TO LEN D$(0)-1:E$= MID$ (D$(0),F,1
    ):EE$= MID$ (D$(0),F+1,1)
2600:IF E$=" " LPRINT "": LPRINT CHR$ 27;FB:
    LPRINT "P ";: GOSUB 2850: GOTO 2690
2610:IF E$="<" LPRINT "": LPRINT CHR$ 27;"1":
    GOSUB 2800: GOTO 2690
2620:IF E$=">" LPRINT "": LPRINT CHR$ 27;"2":
    GOSUB 2800: GOTO 2690
2630:IF E$="^" LPRINT "": LPRINT CHR$ 27;"3":
    GOSUB 2800: GOTO 2690
2635:IF E$=":" GOTO 2690
2640:IF EE$<>":" GOTO 2685
2650:IF E$="A" GOSUB 2700: LPRINT "PA";: GOTO 2
    690
2655:IF E$="O" GOSUB 2700: LPRINT "PO";: GOTO 2
    690
2660:IF E$="U" GOSUB 2700: LPRINT "PU";: GOTO 2
    690
2665:IF E$="a" GOSUB 2710: LPRINT "Pa";: GOTO 2
    690
2670:IF E$="o" GOSUB 2710: LPRINT "Po";: GOTO 2
    690
2675:IF E$="u" GOSUB 2710: LPRINT "Pu";: GOTO 2
    690
2680:IF E$="s" OR E$="ß" GOSUB 2700: LPRINT "P
    ";: GOTO 2690
2685:LPRINT E$;
2690:NEXT F
2692:IF EE$<>":" AND EE$<>"<" AND EE$<>">" AND
    EE$<>"^" LPRINT EE$
2695:RETURN
2700:V=SG:W=SG*7
2701:LPRINT "": LPRINT "I";: LPRINT "R";U;",";W
2702:LPRINT "J1,0,0,1,-1,0,0,-1"
2703:UX=2*U
2704:LPRINT "R";UX;",0"
2700:LPRINT "J1,0,0,1,-1,0,0,-1"
2708:LPRINT "H": RETURN
2710:V=SG:W=SG*5
2711:LPRINT "": LPRINT "I";: LPRINT "R";U;",";W
2712:LPRINT "J1,0,0,1,-1,0,0,-1"
2713:UX=2*U
2714:LPRINT "R";UX;",0"
```

```
2710:LPRINT "J1,0,0,1,-1,0,0,-1"
2718:LPRINT "H": RETURN
2700:LPRINT "": LPRINT "I";
2760:S1=0:T1=5*SG:S2=SG:T2=SG:S3=2*SG:T3=0:S4=S
     G:T4=-SG:S5=0:T5=-SG
2767:S6=-SG:T6=-SG:S7=SG:T7=-SG:S8=0:T8=-SG:S9=
     -2*SG:T9=0
2768:LPRINT "J";S1;",";T1;",";S2;",";T2;",";S3;
     ",";T3;",";S4;",";T4;",";S5;",";T5;
2769:LPRINT ",";S6;",";T6;",";S7;",";T7;",";S8;
     ",";T8;",";S9;",";T9
2770:LPRINT "H": RETURN
2800:IF F=1 LPRINT "P";: RETURN
2810:LPRINT "P ";: RETURN
2850:IF EE$=":" LPRINT EE$;: RETURN
2900:IF RI$="<" LPRINT CHR$ 27+"1";: RETURN
2902:IF RI$=">" LPRINT CHR$ 27+"2";: RETURN
2904:IF RI$="^" LPRINT CHR$ 27+"3";: RETURN
2905:RETURN
2930:CLOSE : GOTO 2950
2940:GOSUB 5850
2942:LPRINT CHR$ 27+"a";
2943:LPRINT CHR$ 27+"?b";
2944:CLS :FB=0: INPUT "SCHRIFTFARBE : ";FB
2940:IF FB<0 OR FB>3 THEN 2944
2948:LPRINT CHR$ 27+ CHR$ (&30+FB);
2950:J$="":K$="": CLS : INPUT "von Zeile:";J$:
     INPUT "bis Zeile:";K$
2952:U= VAL J$:B= VAL K$
2954:IF B>R THEN 2950
2958:IF J$="" OR K$="" THEN 300
2960:FOR Z=U TO B: LPRINT A$(Z): NEXT Z: GOTO 3
     00
3000:REM * L
3005:CLS : PRINT "   *** LESEN ***": GOSUB 6700
     : PRINT ""
3010:K$="": PRINT "ab Zeile": CURSOR 57: INPUT
     K$
3012:IF K$="" THEN 300
3013:Z= VAL K$: IF Z>R THEN 3005
3015:CLS :U1=0:U2=0:J=0: GOSUB 3300
3020:K$= INKEY$ : IF K$="" THEN 3020
3025:RT$= RIGHT$ (A$(Z),1)
3030:IF K$="+" AND Z<R LET Z=Z+1:U1=0:J=0: CLS
     : GOSUB 3300
3040:IF K$="-" AND Z>0 LET Z=Z-1:U1=0:J=0: CLS
     : GOSUB 3300
3080:IF K$="M" THEN 300
3090:GOTO 3020
3300:D$(0)=A$(Z)+ CHR$ 248:LD= LEN D$(0)
3310:FOR I=2 TO LD
3320:MD$= MID$ (D$(0),I,1)
3340:IF MD$=":" LET UM$= MID$ (D$(0),I-1,2):
     GOSUB 3400: GOSUB 3500: GOSUB 3500
3350:NEXT I
3355:U2=LD
3356:PRINT MID$ (D$(0),U1+1,U2-U1-1)
3360:CURSOR 13,3: PRINT CHR$ 248; STR$ Z;" <M +
     ->": RETURN
3400:U2=I
3410:PRINT MID$ (D$(0),U1+1,U2-U1-2);" ";
3420:U1=U2
3490:RETURN
3500:IF UM$="a:" GCURSOR (CX,CY): GPRINT "38454
     43D40": GOTO 3505
3510:IF UM$="o:" GCURSOR (CX,CY): GPRINT "38454
     44538": GOTO 3505
3520:IF UM$="u:" GCURSOR (CX,CY): GPRINT "3C414
     0217C": GOTO 3505
3530:IF UM$="s:" OR UM$="ß:" GCURSOR (CX,CY):
     GPRINT "7E2525251A": GOTO 3505
3540:IF UM$="A:" GCURSOR (CX,CY): GPRINT "79141
     21479": GOTO 3505
3550:IF UM$="O:" GCURSOR (CX,CY): GPRINT "3E414
     B413E": GOTO 3505
3560:IF UM$="U:" GCURSOR (CX,CY): GPRINT "3F484
     5483F"
3505:J=J+1
3570:RETURN
3580:CX=(I+23)*8:CY=-1
3583:CX=CX-8*24:CY=CY+8: IF CX<=8*24 GOTO 3588
3586:GOTO 3583
3588:CX=CX-J*8
3590:GCURSOR (CX,CY): RETURN
3600:REM * F
3650:K$="*": CLS : PRINT "  *** FORMATIEREN ***
     ": GOSUB 6700: PRINT ""
3660:PRINT "VonZeile:   bisZeile:"
3670:K$="": CURSOR 57: INPUT K$
3680:IF K$="" THEN 300
3681:K= VAL K$
3690:IF K>(R-3) THEN 3660
3700:G$="": CURSOR 69: INPUT G$
3710:IF G$="" THEN 300
3711:G= VAL G$
3720:IF G>R-2 BEEP 1: GOTO 3650
3725:IF G<Z LET X=Z:Z=G:G=X
3730:CLS : PRINT "G E D U L D !"
3740:Z=K
3750:CURSOR 24: PRINT STR$ Z
3760:IF A$(Z)="" GOSUB 4110:G=G-1: GOTO 4090
3765:IF A$(Z+1)=":" LET Z=Z+2: GOTO 4090
3770:IF A$(Z+1)="" LET Z=Z+1: GOSUB 4110:Z=Z-1:
     G=G-1: GOTO 4090
3775:REM *
3780:L= LEN A$(Z)
3810:L1= LEN A$(Z+1)
3840:B$(0)= LEFT$ (A$(Z+1),ZB-L)
3850:W= LEN B$(0)-1: IF W<=0 THEN 4000
3860:U$= RIGHT$ (A$(Z),1)
3870:IF U$="<" OR U$=">" OR U$="^" LET W=W+1
3880:IF L1<ZB-L THEN 4240
3885:REM *
3890:ML=W
3900:U$= MID$ (B$(0),ML,1)
3910:IF U$=" " OR U$="<" OR U$=">" OR U$="^"
     LET M=ML: GOTO 3940
3920:ML=ML-1: IF ML<=1 THEN 4000
3930:GOTO 3900
3935:REM *
3940:C$(0)= LEFT$ (A$(Z+1),M)
3950:U$= RIGHT$ (A$(Z),1):U1$= LEFT$ (B$(0),1)
3970:IF U$="<" OR U$=">" OR U$="^" THEN 3990
3975:IF U1$="<" OR U1$=">" OR U1$="^" THEN 3990
3980:A$(Z)=A$(Z)+" "+C$(0): GOTO 4000
3990:A$(Z)=A$(Z)+C$(0)
4000:L= LEN A$(Z): IF RIGHT$ (A$(Z),1)=" " LET
     A$(Z)= LEFT$ (A$(Z),L-1)
4000:A$(Z+1)= RIGHT$ (A$(Z+1),L1-M)
4070:IF A$(Z+1)="" THEN 3770
4080:Z=Z+1
4090:IF Z=G+1 THEN 300
4100:GOTO 3750
4105:REM *
4110:IF Z>R RETURN
4120:FOR Q=Z TO R-1:A$(Q)=A$(Q+1): NEXT Q:A$(R)
     ="": RETURN
4230:REM *
4240:U$= RIGHT$ (A$(Z),1)
4250:IF U$="<" OR U$=">" OR U$="^" LET D$(0)=A$
     (Z): GOTO 4270
4260:D$(0)=A$(Z)+" "
4270:A$(Z)=D$(0)+A$(Z+1):A$(Z+1)="": GOTO 3770
4500:REM * A
4510:CLS : PRINT "    *** AENDERN  ***": GOSUB
     6700: PRINT ""
```

```
4515:PRINT "<L>OESCHEN  <E>INFUEGEN": PRINT "<C
    >OPIEREN  <M>ENUE"
4520:K$= INKEY$ : IF K$="" THEN 4520
4530:IF K$="L" THEN 4700
4540:IF K$="E" THEN 4800
4550:IF K$="C" THEN 4900
4560:IF K$="M" THEN 300
4570:GOTO 4520
4700:REM *
4720:CLS :K=0: PRINT "Loeschen ab Zeile: ?";:
    CURSOR 18: INPUT K$
4725:K= VAL K$+1: IF K>R THEN 4720
4730:CLS :U=0: PRINT "wieviel Zeil. loe: ?";:
    CURSOR 18: INPUT U$
4735:U= VAL U$
4740:IF U=0 CLS : GOTO 4500
4750:FOR Q=K-1 TO R-U
4760:A$(Q)=A$(Q+U): NEXT Q
4770:FOR Q=R-U TO R
4780:A$(Q)="": NEXT Q
4790:CLS : GOTO 4500
4800:REM *
4820:CLS : PRINT "Verschieb. ab Zeile: ?";:
    CURSOR 20: INPUT W$
4825:W= VAL W$: IF W>R THEN 4820
4830:CLS :L=0: PRINT "um wieviel Zeilen: ?";:
    CURSOR 18: INPUT L$:L= VAL L$
4835:IF L=0 CLS : GOTO 4500
4840:FOR Q=R TO W+L-1 STEP -1
4850:A$(Q)=A$(Q-L)
4860:NEXT Q
4870:FOR Q=W+L-1 TO W STEP -1
4880:A$(Q)="": NEXT Q
4890:CLS : GOTO 4500
4900:REM *
4905:CLS : PRINT "COPY:"
4910:CURSOR 8,1: PRINT "von Zeile": CURSOR 13,1
    : INPUT X$
4911:U= VAL X$: IF U>R THEN 4905
4912:CURSOR 8,2: PRINT "nach Zeile": CURSOR 13,
    2: INPUT X$
4913:N= VAL X$: IF N>R THEN 4905
4915:CURSOR 8,3: PRINT "Anzahl Zei.": CURSOR 13
    ,3: INPUT X$:W= VAL X$
4920:FOR Q=0 TO W-1:A$(N+Q)=A$(U+Q): NEXT Q:
    GOTO 4510
5000:REM * S
5010:CLS : PRINT "*** TEXT SPEICHERN ***":
    GOSUB 6700
5020:WAIT 1: PRINT "RECORDER AUF AUFNAHME  <ENT
    >": WAIT 0
5030:GOSUB 5900
5035:IF N$="" THEN 300
5050:PRINT #N$,R,ZB: FOR J=1 TO 500: NEXT J:
    PRINT #"OK.....":A$(X)
5060:GOTO 300
5070:PRINT #1A$(X)
5490:PAUSE : GOTO 300
5500:REM * L
5510:CLS : PRINT "  *** TEXT LADEN ***": GOSUB
    6700
5520:WAIT 1: PRINT "RECORDER AUF WIEDERGABE <ENT
    >": WAIT 0
5530:GOSUB 5900
5535:IF N$="" THEN 300
5550:INPUT #N$,R,ZB: PRINT "DIM = ";R;"   ZB ="
    ;ZB;"  NAME = ";N$: INPUT "(J/N)?";J$
5552:IF J$<>"J" THEN 5500
5560:GOTO 300
5570:INPUT #1A$(X): GOTO 300
5850:CLOSE : OPEN "1200,N,8,1,A,C,&1A": CONSOLE
    79: RETURN
5900:N$="": INPUT "NAME DER TEXTDATEI: ";N$
5910:RETURN
6000:REM * K
6010:CLS : PRINT "* AENDERN IN DER ZEILE":
    GOSUB 6700: PRINT ""
6020:K$="": PRINT "KORRIGIEREN der Zeile":
    CURSOR 20: INPUT K$
6025:Z= VAL K$: IF Z>R THEN 6010
6030:IF K$="" THEN 300
6080:CLS : PRINT A$(Z);: CURSOR 18,3: PRINT
    CHR$ 245; STR$ (Z);" <LEUM + ->"
6090:C=0:L= LEN A$(Z):D$(0)=A$(Z)
6100:K$= INKEY$ : IF K$="" THEN 6100
6110:IF K$=")" AND C<L CURSOR 0:C=C+1: PRINT
    LEFT$ (D$(0),C); CHR$ 249; RIGHT$ (D$(0),L
    -C): GOTO 6100
6120:IF K$="(" AND C>0 CURSOR 0:C=C-1: PRINT
    LEFT$ (D$(0),C); CHR$ 249; RIGHT$ (D$(0),L
    -C): GOTO 6100
6130:IF K$="+" AND Z<R LET Z=Z+1: CLS : PRINT A
    $(Z);: CURSOR 18,3: PRINT CHR$ 245; STR$ (
    Z);" <LEUM + ->": GOTO 6090
6140:IF K$="-" AND Z>0 LET Z=Z-1: CLS : PRINT A
    $(Z);: CURSOR 18,3: PRINT CHR$ 245; STR$ (
    Z);" <LEUM + ->": GOTO 6090
6145:IF K$=";" AND C<=L-24 LET C=C+23:K$=")":
    GOTO 6110
6147:IF K$="/" AND C>23 LET C=C-23:K$="(": GOTO
    6120
6150:IF K$="X" GOSUB 6300:D$(0)=B$(0)+ CHR$ 247
    +C$(0):L=L+1: CLS : PRINT D$(0): GOTO 6100
6160:IF K$="L" GOSUB 6500: BEEP 1: GOTO 6000
6170:IF K$="E" GOSUB 6400: BEEP 1: GOTO 6000
6175:IF K$="U" GOSUB 6000: BEEP 1: GOTO 6000
6180:IF K$="M" GOTO 300
6200:GOTO 6100
6300:B$(0)= LEFT$ (D$(0),C)
6310:C$(0)= RIGHT$ (D$(0),L-C): RETURN
6400:M1=0:LD=0
6405:FOR I=1 TO L
6410:MD$= MID$ (D$(0),I,1)
6420:IF MD$= CHR$ 247 LET M1=I: GOTO 6428
6425:NEXT I: RETURN
6428:I=L: NEXT I
6430:B$(0)= LEFT$ (D$(0),M1-1):L0= LEN B$(0):C$
    (0)= RIGHT$ (D$(0),L-M1):L1= LEN C$(0)
6435:CLS : PRINT B$(0): PRINT "<TEXT>: ";: BEEP
    1: INPUT D$(0)
6437:LD= LEN D$(0)
6440:IF L0+LD+L1<=ZB LET A$(Z)=B$(0)+D$(0)+C$(0
    ): CLS : RETURN
6445:GOSUB 6480
6450:IF L0+LD<=ZB LET A$(Z)=B$(0)+D$(0):A$(Z+1)
    =C$(0): CLS : RETURN
6455:IF LD+L1<=ZB LET A$(Z)=B$(0):A$(Z+1)=D$(0)
    +C$(0): CLS
6460:RETURN
6480:FOR Q=R TO Z+2 STEP -1:A$(Q)=A$(Q-1): NEXT
    Q: RETURN
6500:M1=0:M2=0
6505:FOR I=1 TO L
6510:MD$= MID$ (D$(0),I,1)
6520:IF M1<>0 AND MD$= CHR$ 247 LET M2=I:I=L:
    GOTO 6540
6530:IF MD$= CHR$ 247 LET M1=I
6540:NEXT I
6550:IF M1>0 LET B$(0)= LEFT$ (D$(0),M1-1)
6555:C$(0)= RIGHT$ (D$(0),L-M2):D$(0)=B$(0)+C$(
    0)
6560:IF M1>0 AND M2>0 CLS :A$(Z)=D$(0):L= LEN D
    $(0)
6570:RETURN
6600:B$(0)= LEFT$ (D$(0),C):L0= LEN B$(0):
    CURSOR C
6605:INPUT U$
6606:LU= LEN U$
```

g) Speichern und Laden des Textes

Zur Abspeicherung Ihrer Texte auf Kassette dient das durch <R> erreichbare Modul. Stellen Sie den Recorder auf Aufnahme und geben Sie einen Dateinamen ein. Bloßes <ENTER> ohne Dateinamen führt zum Hauptmenü zurück. Verwenden Sie nur gutes Bandmaterial! Vermerken Sie sich Dateinamen, Zeilenzahl und Zeilenbreite.

Das Laden auf Band gespeicherter Texte von Kassette erfolgt nach <P> im Hauptmenü und nachdem Sie den Recorder auf Wiedergabe eingestellt haben. Bloßes <ENTER> ohne Dateinamen bringt Sie wieder zum Hauptmenü. Für den Fall, daß Sie Zeilenzahl und -breite vergessen haben sollten, überprüfen Sie diese Angaben beim Einladen des Textes, die bald als Meldung auf dem Display erscheinen. Sollten diese Angaben nicht mit der augenblicklichen Dimensionierung des Textspeichers übereinstimmen (ZB=Zeilenbreite, R=Zeilenzahl-1), so müssen Sie die Prozedur abbrechen (<BRK>), da das Laden so nicht funktioniert. Starten Sie sodann das Programm neu mit <DEF><Z> und dimensionieren Sie es entsprechend, bevor Sie wieder <P> drücken!

```
6000:IF L-LØ-LU<Ø CLS : PAUSE "xxxxxxx": RETURN
6010:C$(Ø)= RIGHT$ (D$(Ø),L-LØ-LU)
6020:A$(Z)=B$(Ø)+U$+C$(Ø): CLS : RETURN
6700:LINE (3,Ø)-(140,6),X,BF: RETURN
7000:REM * BLSTZ
7010:L= LEN D$(Ø): IF L<ZB- INT (ZB/5) RETURN
7020:IF ZB-L<1 RETURN
7030:FOR I=1 TO ZB
7040:A$= MID$ (D$(Ø),I,1)
7050:IF A$="" THEN 7090
7060:IF A$=" " OR A$="<" OR A$=">" OR A$="^"
     GOSUB 7100
7070:IF ZB-L<1 LET I=ZB
7080:NEXT I
7090:IF L-ZB THEN 7030
7099:RETURN
7100:C$(Ø)= LEFT$ (D$(Ø),I-1)
7110:B$(Ø)=" "+ RIGHT$ (D$(Ø),L-I+1)
7120:D$(Ø)=C$(Ø)+B$(Ø):L= LEN D$(Ø)
7130:B$= MID$ (D$(Ø),I+1,1)
7140:IF B$=" " OR B$="<" OR B$=">" OR B$="^"
     LET I=I+1: GOTO 7030
7150:RETURN
```

Kap. XII:

PLAKATMALEREI MIT VERSCHIEDENEN PLOTTERN

1. Diagonaldruck

Wir nehmen in diesem Schlußkapitel den Faden von Kap. II.8 wieder auf und sehen uns die Möglichkeiten an, wie **beliebige Zeichensätze** auf den **verschiedensten Plottern** kunstvoll zu verwenden sind. Hier von **Plakatmalerei** zu sprechen, scheint nicht übertrieben, denn der Kreativität des Anwenders wird breiter Raum gegeben.

Bekanntlich schreiben die SHARP-Plotter nur in 4 Richtungen, obwohl sie doch in 8 Richtungen Striche zeichnen können. Die 4 **Hauptrichtungen** werden etwa beim PC 1500/CE-150 durch **ROTATE 0 bis 3** eingestellt. Die **4 Nebenrichtungen** könnte man sinnentsprechend mit **ROTATE 0.5, 1.5, 2.5 und 3.5** bezeichnen, aber natürlich sprechen die Geräte darauf nicht an. Auf verhältnismäßig einfache Weise ist es aber doch möglich, den **gewünschten Effekt** zu erzielen, und zwar folgendermaßen:

a) Wir benutzen (im GRAPH-Modus) eine Druckroutine, die die Zeichen diagonal versetzt darstellt in den 4 Nebenrichtungen.

b) Wir stellen einen 2. Zeichensatz mit gedrehten Zeichen zur Verfügung.

Wohl schreibt der eingesetzte Plotter hier weiter in einer der 4 Hauptrichtungen, aber es entsteht der Eindruck, daß er in einer Nebenrichtung arbeite (Simulation). Wenn wir in dem Verfahren nur Teil b) benutzen, haben wir auch eine **Kursivschrift** zur Verfügung, was ebenfalls eine angenehme Sache ist. Da wir beim

```
0600   61 42 61 54 4A 41 7A 74 41 63 41 54   5A 7A 72 E4 6B 7B 41 63 41 DA 71 E1
0618   49 61 49 59 69 79 51 49 56 41 76 62   45 49 51 59 49 51 59 65 44 79 71 69
0630   62 42 79 71 E9 41 71 E1 10 07 19 00   34 24 4B 02 05 51 69 62 5A 52 4A 42
0648   79 51 22 69 74 F9 41 12 51 61 71 41   12 51 59 79 61 71 41 01 51 41 71 E1
0660   61 44 4A 52 5A 64 41 76 41 56 41 79   74 E9 61 59 69 79 49 61 59 69 79 49
0678   61 69 79 49 61 69 79 49 61 46 51 69   63 53 42 79 52 69 62 53 44 71 59 65
0690   41 F6 79 49 59 71 69 79 49 D9 91 05   00 21 01 01 61 43 61 53 42 79 52 69
06A8   62 53 44 71 59 65 41 F6 41 F6 03 23   81 01 13 01 81 01 09 01 93 01 0D 93

06C0   02 5A 52 4A 42 79 51 22 69 74 79 61   41 49 79 53 41 63 41 72 E9 40 58 03
06D8   00 94 08 00 80 10 00 00 61 43 62 56   61 43 61 76 53 43 53 61 43 61 76 41
06F0   63 41 53 E3 E9 04 21 D5 04 18 0B 10   21 88 64 01 04 63 41 56 61 43 61 F6
0708   53 43 53 61 43 61 76 41 63 41 53 E3   E9 31 02 20 00 02 00 2D 00 41 38 01
0720   11 79 41 4A 54 41 63 41 F5 43 53 61   43 61 76 41 63 41 53 E3 E9 28 A0 00
0738   00 10 00 40 14 44 48 00 61 43 62 56   61 43 61 76 52 4C 41 62 41 6C 11 7B
0750   61 43 61 DB 01 04 00 8C 10 04 00 40   80 01 08 10 61 46 51 69 63 56 41 63
0768   41 F6 41 62 41 6C 11 7B 61 43 61 DB   80 28 08 01 08 00 84 02 80 03 40 0C

0780   61 42 61 56 61 41 7A 71 5A 7A 51 4A   42 61 76 41 62 56 76 E1 01 01 01 81
0798   81 01 11 01 81 01 01 21 61 42 61 56   7D 55 41 62 41 76 DD 4A 42 61 76 41
07B0   62 56 76 E1 01 03 05 01 09 81 01 21   81 03 11 21 12 52 4A 41 7A 72 6A 61
07C8   5A 79 54 49 79 74 E9 41 62 56 76 E1   72 02 02 22 06 02 42 02 22 02 02 02
07E0   61 43 61 56 42 79 71 69 62 43 49 51   59 65 41 F6 62 56 76 E1 03 82 02 82
07F8   02 06 02 03 42 82 02 02 12 52 4A 41   7A 72 6A 72 41 61 59 51 49 54 59 69
0810   74 59 FA E1 20 00 04 40 02 00 02 02   00 00 40 01 61 43 62 56 61 45 79 71
0828   69 63 7B 61 43 61 5B 41 49 51 59 62   F6 04 8C 02 04 00 01 00 08 A0 01 12

0840   51 79 43 49 51 5A 61 59 51 43 79 51   24 69 71 7A 41 79 71 E3 F6 00 00 01
0858   80 10 08 10 00 00 00 00 01 43 61 56   42 71 59 64 71 49 41 F6 24 69 71 7A
0870   41 79 71 E3 F6 20 00 00 00 60 01 80   00 20 09 48 12 54 61 43 61 74 7A 41
0888   49 55 41 62 41 75 69 62 DA 79 71 E3   F6 00 00 01 00 41 10 00 40 20 08 20
08A0   03 4A 54 41 62 22 63 41 71 79 71 79   71 79 52 59 51 59 51 F1 F6 08 20 00
08B8   00 00 80 A8 30 20 04 00 02 5A 54 61   43 61 74 79 71 53 42 61 73 61 41 4A
08D0   54 41 62 41 74 EA 02 02 02 8A 02 02   03 02 02 02 61 42 61 4A 52 5A 61 42
08E8   61 42 61 7A 4A 41 62 41 6B 7B 41 63   41 DA 0A 02 02 02 22 4A 12 C6 02 02

0900   01 43 62 54 5A 61 43 61 7A 4A 41 62   41 6A F4 41 6B 7B 41 63 41 DA 04 08
0918   10 00 20 00 00 01 00 00 45 51 69 63   4A 52 4A 65 71 49 43 6A 72 EA F4 41
0930   6B 7B 41 63 41 DA 00 18 00 00 01 00   80 20 00 00
```

PC 1500(A) mit einem **echten '2. Zeichensatz'**

arbeiten, kann man die Zeichen ebenfalls vergrößern usw.

Der Leser findet nun nun ein **Initialprogramm** vor **für beliebige Zeichensätze**, unter anderem auch für die **DIAGONAL-Schrift**. Die Plot-Zeichen müssen zuvor in einem vor BASIC geschützten Maschinensprache-Bereich abgelegt werden. Die hier tabellierten Diagonal- **Plot-Zeichen**, die im Speicher **frei verschiebbar** (relokatibel) sind, können z.B. auf Adresse &600 geladen werden. Um auch **Display-Zeichen** zur Verfügung zu haben, sind hier die **MICRO-Zeichen aus Kap. II.5** eingearbeitet - der Leser kann mit Hilfe des Zeichengenerators ja andere Display-Zeichen schaffen, die seinem speziellen Zeichensatz angemessen sind.

Eine **Warntafel** muß aber aufgestellt werden: beim Experimentieren mit 2. Zeichensätzen wird es dem Leser bestimmt einmal passieren, daß der Plotter CE-150 ein irreguläres Verhalten zeigt, insofern, daß er wilde Zeichen druckt und **endlosen Papiervorschub** oder auch -rückzug produziert. Dann müssen Sie meistens den **Rechner vom Plotter abziehen**, wonach sich auch ein RESET des Rechners empfiehlt. Das alles ist unangenehm, aber ein Beinbruch ist es nicht.

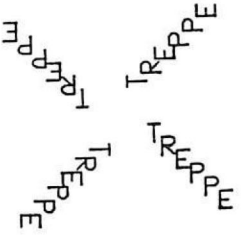
Der Plotter des Verfassers jedenfalls ist durch diverse solcher Vorfälle nicht beschädigt worden, **ohne** daß er hier eine **Garantie** für die Geräte des Lesers geben könnte. Ausgelöst wird das **irreguläre Plotterverhalten**, wenn das Programm in einem Speicherbereich nach Plot-Zeichen sucht, wo keine vorhanden sind. Das kann schon dann der Fall sein, wenn nach Ausschalten des Rechners in der **Speichersystemzelle &785D keine 0 (Null)** steht.

```
  100 "J"REM Initialpgm fuer DIAGONALSCH
RIFT/ANTIQUA-SCHRIFT
  110 HB=&02:REM LCD-High-Byte
  120 GOSUB 600
  130 POKE (HB+2)*256,&05,0:REM Zeiger f
uer DRU-ADR
  140 REM ----- DRU-ADR
  150 FOR I=0TO &40:POKE &500+2*I,&A0,0:
NEXT I:REM  SPACES
  160 FOR I=1TO &1A
  162 K=&500+2*(I+&40):L=&600+&20*(I-1)
  164 LH=INT (L/256):LL=L-256*LH
  166 POKE K,LH,LL
  168 NEXT I
  170 FOR I=&5BTO &7F:POKE &500+2*I,&A0,
0:NEXT I:REM  SPACES
  180 REM ----- DRU-ZEI
  190 REM DRU-ZEI-Tabelle (mit ZEI-GEN)
von &600-&93F
  200 REM ----- TAST-TAB (Kopie Standard
belegung)
  210 POKE &172,&48,&FE,&4A,&80,&58,HB-1
,&5A,&80,&6A,&7F,&F5,&88,&03,&9A
  220 CALL &172
  230 GOSUB 300
  240 REM  INITIALISIERUNG
  250 CLS :WAIT 0:PRINT "ALLES OK? (J/N)
"
  260 C$=INKEY$ :IF C$=""THEN 260
  270 IF C$<>"J"BEEP 3:END
  280 POKE &785D,0,2:POKE &764E,&44
  290 BEEP 1:END
  300 REM Spezialbelegung
  310 G=(HB-1)*256
  320 POKE G+&F4,&C1
  330 POKE G+&F9,&C2
  340 POKE G+&E1,&C3
  350 POKE G+&E4,&C4
  360 POKE G+&E2,&C5
  370 POKE G+&EC,&C6
  380 POKE G+&FC,&C7
  390 POKE G+&C4,&C8
  400 POKE G+&DA,&C9
  410 POKE G+&D4,&CA
  420 POKE G+&DC,&CB
  430 POKE G+&DE,&CC
  440 POKE G+&D1,&CD
  450 POKE G+&C1,&CE
  460 POKE G+&DD,&CF
  470 POKE G+&ED,&D0
  480 POKE G+&F2,&D1
  490 POKE G+&EA,&D2
  500 POKE G+&CC,&D3
  510 POKE G+&FA,&D4
  520 POKE G+&D2,&D5
  530 POKE G+&E9,&D6
  540 POKE G+&CA,&D7
  550 POKE G+&C9,&D8
  560 POKE G+&C2,&D9
  570 POKE G+&F1,&DA
  580 RETURN
  600 REM LCD-TAB
  610 H=HB*256
  620 POKE H+&145,&78,&24,&7E,0,0
  630 POKE H+&14A,&7E,&5A,&6C,0,0
  640 POKE H+&14F,&3C,&42,&46,0,0
  650 POKE H+&154,&7E,&62,&7C,0,0
  660 POKE H+&159,&7E,&4A,&42,0,0
  670 POKE H+&15E,&7E,&0A,&02,0,0
  680 POKE H+&163,&7C,&52,&66,0,0
  690 POKE H+&168,&7E,&10,&7E,0,0
  700 POKE H+&16D,&42,&7E,&42,0,0
  710 POKE H+&172,&62,&42,&3E,0,0
  720 POKE H+&177,&7E,&18,&66,0,0
  730 POKE H+&17C,&7E,&40,&40,0,0
  740 POKE H+&181,&78,&3C,&7E,0,0
  750 POKE H+&186,&7C,&08,&1E,0,0
  760 POKE H+&18B,&7C,&42,&7E,0,0
  770 POKE H+&190,&7E,&12,&1C,0,0
  780 POKE H+&195,&3C,&32,&7E,0,0
  790 POKE H+&19A,&7E,&3A,&4C,0,0
  800 POKE H+&19F,&4C,&4A,&32,0,0
  810 POKE H+&1A4,&02,&7E,&02,0,0
  820 POKE H+&1A9,&7E,&40,&7E,0,0
```

Die neuen Zeichen werden vom Initialprogramm mit den **Code-Nummern &C1 - &5A** versehen, überlagern damit also die **Kleinbuchstaben**. Daher sind sie auch mit <SHIFT><Taste> auf das Display zu holen, das 'A' also z.B. mit <SHIFT> <A>. Sollen mehr als die 26 Zeichen installiert werden, muß das Initialprogramm entsprechend erweitert werden.

Wie dieser Zeichensatz auch mit anderen **Rechner/Plotterkombinationen** zu nutzen ist, wird im folgenden Abschnitt klar werden.

```
 830 POKE H+&1AE,&3E,&60,&3E,0,0
 840 POKE H+&1B3,&1E,&38,&7E,0,0
 850 POKE H+&1B8,&6E,&10,&6E,0,0
 860 POKE H+&1BD,&0E,&68,&7E,0,0
 870 POKE H+&1C2,&66,&52,&4E,0,0
 880 RETURN
1000 REM DRUCK-ROUT. FUER DIAGONALSCHR.
1005 "X"CLEAR :GRAPH :GLCURSOR (110,0):
SORGN
1010 DIM D$(0)*80
1015 INPUT "ROTATE:";RO:RO=INT RO:ROTAT
E RO
1020 INPUT "TEXT:";D$(0)
1030 FOR I=1TO LEN D$(0)
1040 M$=MID$ (D$(0),I,1)
1045 IF RO=0GOSUB 1130
1050 ON ROGOSUB 1100,1110,1120:GOTO 106
0
1060 GLCURSOR (X,Y):LPRINT M$
1070 NEXT I:GLCURSOR (0,0):TEXT :END
1100 X=X-9:Y=Y-9:RETURN
1110 X=X-9:Y=Y+9:RETURN
1120 X=X+9:Y=Y+9:RETURN
1130 X=X+9:Y=Y-9:RETURN
```



NORMALSCHRIFT

DIAGONAL

2. Zirkular-, Spiral- und Sterndruck

In diesem Abschnitt finden Sie einen weiteren Zeichensatz von etwas **schöneren Zeichen**, der ebenfalls mit dem Initialprogramm des letzten Abschnitts installiert werden kann. Diese Schriftart wird auch sonst oft verwendet bei vielen Druckerzeugnissen und nennt sich **'Klassizistische Antiqua'**. Wir wollen diesen Zeichensatz nun **auf unterschiedlichen Plottern** benutzen mit Hilfe des **Programmes 'ZPLOT'**, das geladen werden soll, nachdem die Zeichen dieses (oder eines anderen) Zeichensatzes in einem vor BASIC geschützten Bereich eingebracht worden sind.

ZPLOT läuft auf verschieden Rechnern in gleicher oder ähnlicher Form. Z.B. ist die Version für den **PC 1600** identisch mit der des PC **1500(A)**, gleich ob der 1600 im Mode 0 oder 1 arbeitet, wobei einmal der Plotter **CE-150** benutzt wird, das andere Mal der **CE-1600P**. In einer anderen Version benutzen die beiden Rechner die Schnittstelle CE-158, um mit dem Plottern **CE-515P oder CE-516P** zusammenarbeiten zu können. Diese Programmversion ist der ursprünglichen wesensgleich, nimmt aber auf die **unterschiedliche Plottersyntax** Rücksicht.

Auch der **PC 1350** kann mit den Plottern CE-515P und CE-516P zusammenarbeiten, und zwar ist die Programmversion für diesen Rechner der Version für den PC 1500/1600 im Zusammenhang mit diesen Plottern **absolut gleich** bis auf Zeile 505, welche das Interface RS-232C initiiert, und dem einfacheren BEEP. Es genügt also, **ein einziges Programm** zu besprechen und dann jeweils nur die **speziellen Abänderungen** für bestimmte Rechner/Plotterkombinationen anzugeben. Wer noch andere Kombinationen sein eigen nennt, wird die Sache bestimmt in den Griff kriegen, da das Programm prinzipiell das gleiche ist.

```
0600  13 4B 43 73 69 5B 7B EA EA F1 4B 63  4B F2 50 43 2F 45 50 20 2B 31 2F 32
0618  29 22 3B 3A F0 84 32 30 03 11 61 5A  4C 7A 71 69 61 5A 7A 71 E9 D1 72 69
0630  79 E1 22 F1 92 22 4C 22 0D 07 D5 31  F0 BB 3A F0 02 0C 51 5A 61 6B 71 7A
0648  C1 72 6A 71 41 D1 20 30 2F 31 2F 32  2F 33 29 20 3F 22 3A F0 84 32 32 3A
0660  13 7A 41 4B 51 5A EC F3 F1 62 4B 24  0D 07 D6 11 F1 96 43 4F 3E 33 F1 B2
0678  31 3A F1 92 32 30 30 35 03 5B 4C 7B  21 31 22 DA 72 E2 32 3A F0 B5 43 4F
0690  3A F1 94 39 35 30 30 0D 07 E4 09 F1  A5 4A 3D 30 13 4C 7B 23 31 DA 22 DA
06A8  5A C2 62 4A 5A 82 99 61 69 F2 3A 54  3D F1 5B 0D 07 E6 19 55 24 3D F1 5C

06C0  02 0C 51 5A 61 6B 71 7A 41 49 DA 4A  62 49 81 B3 E1 F2 22 0D 07 E7 15 F1
06D8  96 F1 5B 3E 3D 54 2B 30 13 4C 3B 6A  5B 7B EA 32 30 32 35 0D 07 E8 07 F1
06F0  92 32 30 32 32 0D 07 E9 07 E6 86 3A  F0 B6 34 0D 01 0D 59 7A 59 6C 59 FA
0708  31 3A F1 94 39 34 30 30 0D 07 F8 10  E6 82 28 32 31 30 2C 30 29 3A F0 B9
0720  14 71 79 41 0B 59 7A 59 EB 5A 3D 50  2A 33 33 2B 36 2A 4A 2B 41 F1 B1 50
0738  2A 33 33 2B 36 2A 4A 2B 13 4C 3A 31  65 41 F3 E6 82 28 47 2C 30 29 3A F1
0750  94 32 35 30 30 3A 47 3D 47 2D 33 39  0D 0B 16 04 13 0C 6C FA 65 41 F3 E6
0768  82 28 30 2C 2D 36 2A 5A 42 29 3A E6  84 0D 08 2A 07 F1 94 39 34 30 30 0D

0780  13 4C 73 43 EC 5A 3D 50 2A 33 33 2B  36 2A 4A 2B 41 F1 B1 50 2A 33 33 2B
0798  36 2A 4A 2B 35 2D 42 0D 13 4C 69 76  4C EC 47 2C 30 29 3A F1 94 32 36 30
07B0  30 3A 47 3D 47 2D 33 39 0D 08 7A 04  F1 9A 5A 0D 13 51 4B 41 7A 71 6B 61
07C8  DA 0D 08 8E 15 E6 82 28 30 2C 2D 28  39 36 30 2D 36 2A 5A 42 29 29 3A E6
07E0  13 4C 7A 71 69 61 DA 5A 3E 3D 52 4C  F1 92 22 4D 45 22 0D 08 98 0A F1 9A
07F8  4A 3A 50 3D 50 2B 31 0D 03 51 61 5A  51 4B 41 7A 71 6B D2 C4 2C 44 24 28
0810  30 29 3D F1 7A 28 41 24 28 5A 2D 31  29 2C 5A 42 13 4C 7A 71 69 61 5A F4
0828  71 6B D2 30 29 3A F0 B9 44 24 28 30  29 3A F1 99 0D 0A 28 35 44 24 28 30

0840  13 71 7A 41 49 51 59 03 51 5A 61 69  71 F9 5A 42 2F 32 29 2B 31 2C 5A 42
0858  2F 32 29 3A 44 3D F1 64 12 01 4C 5A  7C 5A EC B9 44 24 28 30 29 3A F1 99
0870  0D 0A F0 07 F1 94 39 39 30 30 0D 0A  F5 36 F0 97 13 7A 41 4C 1B 6C F1 76
0888  2E 5A 65 69 2E 22 3A F0 84 31 32 3A  F0 91 56 5A 3A F0 84 31 34 3A F0 97
08A0  12 01 53 4B 3B 6B E3 2E 22 3A F0 84  32 31 3A F0 91 4E 5A 0D 0A FA 29 E6
08B8  80 31 3A F1 A5 4A 3D 56 13 43 49 69  73 4C 1B EC 4A 3B 22 3A 20 22 3B 41
08D0  24 28 4A 29 3A F1 9A 4A 3A E6 B0 32  3A F1 99 0D 12 49 44 49 1A 69 74 E9
08E8  4E 22 F0 88 3A F0 97 22 3E 3E 4C 45  53 45 4E 3C 3C 22 0D 0B BA 0D F1 A5

0900  09 4A 52 49 3A 69 E2 E9 2D 31 0D 0B  BC 19 F1 96 41 24 28 51 29 3C 3E 22
0918  22 F1 98 52 4C 3D 51 3A 13 0C 7A 69  64 69 FA E9 51 5A 61 69 71 F9 07 D2
0930  36 44 52 24 3D 22 22 3A F1 94 39 39  30 30 3A F0
```

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

SHARP

SHARP

SHARP

WINFRIED MEYER
WINFRIED MEYER HASLO

Extra eingegangen wird hier noch auf die Kombination **MZ-800/MZ-1P16** - die auch den **MZ700/ eingeb. Plotter** simulieren kann - weil hier nicht nur plotterseitig, sondern auch beim Rechner eine **andere BASIC-Syntax** vorliegt.

ZPLOT benutzt vom '2. Zeichensatz' **nur die Plot-Zeichen** (DRU-ZEI), die als Tabelle vorliegen müssen. Aus dieser werden die Zeichen **Byte für Byte ausgelesen** und in kleine Striche umgesetzt, was schließlich einen Buchstaben oder Sonderzeichen ergibt usw. Dazu wird ein vorliegendes Byte (in dem Unterprogramm Zeile 80 - 390) so **gesplittet**, daß es wieder die **4 Informationen** liefert, die zur Bewegung des Plotterstiftes vonnöten sind: Länge, Richtung, Sift oben/unten und (nicht) letzter Strich eines Zeichens.

Gestartet wird das Programm mit RUN oder <DEF> <D>, wonach zunächst die Startadresse der Tabelle DRU-ZEI angegeben wird. Der Wert &600 ist vorgegeben, ein einfaches <ENTER> übernimmt diesen Wert. Dann können noch **Schriftgröße und -farbe** eingestellt werden, der **Nullpunkt** der Zeichenebene und ein **Dehnungsfaktor** in Ordinaten- wie Abszissenrichtung. Hiermit können Zeichen in gewünschter Weise **verzerrt** werden. Vorgegeben ist (bei einfachem <ENTER>) hier der Wert 1, bei dem keine Verzerrung stattfindet. Schließlich bedeutet **Radius** der Abstand aller oder (bei Sternschrift) des letzten Zeichens vom Nullpunkt der Zeichenebene.

Endlich gibt man einen **zu plottenden Text** ein, der auch von einem Textprogramm zur Verfügung gestellt werden kann, wenn man ZPLOT an dieses ankoppelt. Durch wenige Änderungen ab Zeile 740 kann man jetzt eine der 3 Ausprägungen erzeugen: **Sternschrift, Zirkularschrift oder Spiralschrift.** Entsprechende Änderungen wären

noch in den Zeilen 1500,1510 vorzunehmen, wo die jeweilige **Druckposition** festgelegt wird. Nun wird der Text in der Schleife von Zeile 1010 bis 1170 buchstabenweise abgearbeitet.

Die einzelnen Schritte laufen folgendermaßen ab: Zunächst wird der **ASC-Code eines Zeichens** der Zeichenkette bestimmt (Zeile 1040). Dann berechnet das Programm aufgrund dieses Codes die **Adresse,** wo ein entsprechendes neues Zeichen **in DRU-ZEI** zu finden ist, und übergibt diese der **Variablen AD.** Die Zeichen unserer 'Antiqua' liegen alle im Abstand von &20. Die Abstände beispielsweise der SCHREIBSCHRIFT-Zeichen sind ungleich. Sie könnten als Data-Zeilen an ZPLOT angehängt werden und in Zeile 1060 mit einer READ-Anweisung eingelesen werden.

Als nächstes werden in einem Unterprogramm ab Zeile 100 - wie schon beschrieben - Informationen ausgelesen und den **Variablen L,R,P und Q zugewiesen,** die den weiteren Ablauf steuern. **Je nach Richtung** wird jetzt in eine der auf Zeilennummer 1990 folgenden Zeilen verzweigt, wo die **Koordinaten** des gerade aus 1 Byte ausgelesenen Striches (bzw. Druckstiftbewegung) **umgewandelt** und somit für den Druck festgelegt werden. Der **Druck** mit auf- oder abgesetztem Stift findet schließlich in den Zeilen 1130 und 1140 statt. Bevor ein neuer Buchstabe angefangen wird, wird nach Zeile 1500 verzweigt, wo die **Ausgangsposition** eingestellt wird. Diese Position ist sehr abhängig von den vorzugebenden **Parametern** für das ganze Druckbild, als da sind Winkel, Radius, Schrittweite (SW) usw. Kleine Veränderungen hier zeigen große Wirkungen im ganzen Druckbild.

Noch einmal zurück zu den **Koordinatentransformationen** ab Zeile 2000, die ein Schlüssel zum

Verständnis der ganzen Methode sind. Eingearbeitet sind hier die Parameter G (Schriftgröße) und O (Winkel). Jeder aus der Tabelle DRU-ZEI geholte Strich (mit bestimmter Länge L und Richtung R) wird hier also noch einmal unter einem bestimmten Winkel gesehen, wobei er eine neue Richtung bekommt. Von diesem neuen Strich wird jetzt **trigonometrisch** die X- und die Y- Komponente bestimmt. Diebei sind die Berechnungen für die **Nebenrichtungen** (Zeilen 2001, 2003, 2005, 2007) komplizierter, da diese Richtungen schon von Haus aus (ohne den Parameter O) sowohl eine X- als auch eine Y- Komponente haben.

Das ganze gilt natürlich auch für 'Striche' mit aufgehobenem Stift. Die **Komponenten X und Y,** die noch abhängig sind von L und G und letztlich auch vom Dehnungsfaktor DX bzw. DY, bestimmen schließlich die **Lage des Striches** in den Zeilen 1130 und 1140.

Bei der **Sternschrift** bleibt der einmal eingegebene **Winkel gleich.** Die einzelnen Zeichen kommen unter einem bestimmten Winkel **auf dem Radiusstrahl** zu liegen. Die Schrittweite ist abhängig von der Zahl der Zeichen. Wechselt man den Winkel bei einem neuen Text (oder noch einmal dem gleichen Text) unter Beibehaltung des Nullpunkts der Zeichenebene, kann man ein **sternartiges Gebilde** erzeugen.

Bei der **Zirkularschrift** ist der **Winkel O veränderlich** im Laufe des Druckvorgangs, und die Position bei Start eines neuen Zeichens wird so eingestellt (Zeile 1500), daß die Zeichen **auf dem Kreisumfang** zu liegen kommen. Hier sind **viele Variationen** denkbar, beispielsweise ließe sich für die Druckposition auch eine **Ellipse** zugrunde legen. Hier kommt noch eine **Spiralschrift.** Dabei lassen wir den Winkel 2 oder mehrere Male volle 360 Grad durchlaufen,

```
 10:REM *** ZPLOT
    ***
 20:GOTO 500
 80:REM --- BYTE S
    PLITTEN
 90:REM ---LAENGE
    (BIT 0-2)
100:A=PEEK AD
110:B=A/2:C=A-2*
    INT B
120:L0=C
130:A=INT B
140:B=A/2:C=A-2*
    INT B
150:L1=C*2
160:A=INT B
170:B=A/2:C=A-2*
    INT B
180:L2=C*2*2
190:L=L0+L1+L2
200:REM ---RICHTUN
    G (BIT 3-5)
210:A=INT B
220:B=A/2:C=A-2*
    INT B
230:R0=C
240:A=INT B
250:B=A/2:C=A-2*
    INT B
260:R1=C*2
270:A=INT B
280:B=A/2:C=A-2*
    INT B
290:R2=C*2*2
300:R=R0+R1+R2
310:REM --- PEN UP
    /DOWN
320:A=INT B
330:B=A/2:C=A-2*
    INT B
340:P=C
350:REM ---ZEICHEN
    ENDE?
360:A=INT B
370:B=A/2:C=A-2*
    INT B
380:Q=C
390:RETURN
400:REM ----------
500:"D"REM --- DRU
    CKROUTINE F. C
    E-1600P MIT NE
    UEN ZEICHEN
510:CLS :CLEAR
520:ST=&600:INPUT
    "STARTADR. F.
    DRU-ZEI: ";ST
530:INPUT "SCHRIFT
    GROESSE: ";G
540:CSIZE G
550:INPUT "SCHRIFT
    FARBE: ";SF
560:COLOR SF
570:INPUT "URSPRUN
    G -X:";UX
580:INPUT "URSPRUN
    G -Y:";UY
590:DX=1:INPUT "DE
    HN.FAFT-X (1-4
    )";DX
600:DY=1:INPUT "DE
    HN.FAFT-Y (1-4
    )";DY
610:INPUT "RADIUS
    (30- 450) : ";
    RA
620:GRAPH
700:REM ----------
710:DIM A$(0)*79
720:"J"INPUT "TEXT
    : ";A$(0)
730:N=LEN A$(0):SW
    =RA/N
740:REM ---SPEZIEL
    L: STERNSCHRIF
    T
750:INPUT "WINKEL:
     ";O
760:REM ---
770:GLCURSOR (UX,U
    Y):SORGN
780:GOSUB 1500
 1000:REM --------
1010:REM --- DRUC
     K EINER ZEIL
     E A$(Z)
1020:FOR F=1TO N
1030:REM --- ASC-
     CODE FUER ZU
      DRUCKENDES
     ZEICHEN
1040:E$=MID$ (A$(
     Z),F,1):U=
     ASC E$:IF E$
     =" "THEN 116
     0
1050:REM --- VERZ
     WEIGUNG ZU D
     RU-ZEI
1060:AD=ST+(U-65)
     *&20
1070:REM ---
1080:"STRICH"
     GOSUB 100
1090:ON RGOSUB 20
     01,2002,2003
     ,2004,2005,2
     006,2007
1100:IF R=0GOSUB
     2000
1110:X=DX*X:Y=DY*
     Y
1120:REM --- AUSD
     RUCK DES STR
     ICHES
1130:IF P=0RLINE
     -(X,Y),9
1140:IF P=1RLINE
     -(X,Y),0
1150:IF Q=0LET AD
     =AD+1:GOTO "
     STRICH"
1160:GOSUB 1500
1170:NEXT F
1180:REM ---
1190:GLCURSOR (0,
     0):GLCURSOR
     (-UX,-UY)
1200:BEEP 3,99,99
     :TEXT :END
```

wobei der Radius ständig kürzer wird.

Wünschenswert ist natürlich eine Steigerung der Plotgeschwindigkeit. Nur wenig bringt das Weglassen der REM-Zeilen. Eine Steigerung etwa um den Faktor 2 bewirkt beim PC 1500(A) ein kleines Maschinenprogramm (ab Zeile 5000), das die BASIC-Routine ab Zeile 100 ersetzt. In Zeile 1080 verzweigen wir also zu dieser (verschiebbaren) Routine.

```
1490:REM --- DRUC
     KPOSITION
1500:W=W+SW:PY=W*
     SIN O:PX=W*
     COS O
1510:GLCURSOR (PX
     ,PY):RETURN
1990:REM --- KOOR
     DINATEN D. S
     TRICHES
2000:X=L*G*COS O:
     Y=L*G*SIN O:
     RETURN
2001:X=L*G*(COS O
     +COS (O+90))
     :Y=L*G*(SIN
     O+SIN (O+90)
     ):RETURN
2002:X=L*G*COS (O
     +90):Y=L*G*
     SIN (O+90):
     RETURN
2003:X=L*G*(COS (
     O+90)+COS (O
     +180)):Y=L*G
     *(SIN (O+90)
     +SIN (O+180)
     ):RETURN
2004:X=L*G*COS (O
     +180):Y=L*G*
     SIN (O+180):
     RETURN
2005:X=L*G*(COS (
     O+180)+COS (
     O+270)):Y=L*
     G*(SIN (O+18
     0)+SIN (O+27
     0)):RETURN
2006:X=L*G*COS (O
     +270):Y=L*G*
     SIN (O+270):
     RETURN
2007:X=L*G*(COS (
     O+270)+COS O
     ):Y=L*G*(SIN
     (O+270)+SIN
     O):RETURN
5000:QX=&F00
5010:POKE QX, &04,
     &FD, &C8, &B9,
     &07, &F1, &AE,
     &79, &5A, &FD,
     &8A, &FD, &C8,
     &B9, &38, &D9
5020:POKE (QX+&10
     ), &AE, &79, &8
     A, &FD, &8A, &F
     D, &C8, &B9, &4
     0, &D5, &D5, &A
     E, &79, &7A, &F
     D, &8A
5030:POKE (QX+&20
     ), &FD, &C8, &B
     9, &80, &F1, &D
     9, &AE, &79, &8
     2, &FD, &8A, &F
     9, &9A
5040:RETURN
```

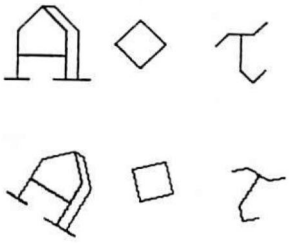
```
  10 REM *** ZPLOT *** (515/6P)
  20 REM (C) 1987 by Winfried Meyer, Ha
sloh
  30 GOTO 500
  80 REM --- BYTE SPLITTEN
  90 REM ---LAENGE (BIT 0-2)
 100 A=PEEK AD
 110 B=A/2:C=A-2*INT B
 120 L0=C
 130 A=INT B
 140 B=A/2:C=A-2*INT B
 150 L1=C*2
 160 A=INT B
 170 B=A/2:C=A-2*INT B
 180 L2=C*2*2
 190 L=L0+L1+L2
 200 REM ---RICHTUNG (BIT 3-5)
 210 A=INT B
 220 B=A/2:C=A-2*INT B
 230 R0=C
 240 A=INT B
 250 B=A/2:C=A-2*INT B
 260 R1=C*2
 270 A=INT B
 280 B=A/2:C=A-2*INT B
 290 R2=C*2*2
 300 R=R0+R1+R2
 310 REM --- PEN UP/DOWN
 320 A=INT B
 330 B=A/2:C=A-2*INT B
 340 P=C
 350 REM ---ZEICHENENDE?
 360 A=INT B
 370 B=A/2:C=A-2*INT B
 380 Q=C
 390 RETURN
 400 REM ----------
 500 "D"REM --- DRUCKROUTINE F. CE-515P
P MIT NEUEN ZEICHEN
 505 OPN "LPRT":CONSOLE 0,0:SETCOM 1200
 510 CLS :CLEAR
 515 GOSUB 5000
 520 ST=&600:INPUT "STARTADR. F. DRU-ZE
I: ";ST
 530 INPUT "SCHRIFTGROESSE: ";G
 540 LPRINT CHR$ 27+"?"+CHR$ (&60+G);
 542 REM CSIZE G
 550 INPUT "SCHRIFTFARBE: ";SF
 560 LPRINT CHR$ 27+CHR$ (&30+SF);
 562 REM COLOR SF
 570 INPUT "URSPRUNG -X:";UX
 580 INPUT "URSPRUNG -Y:";UY
 590 DX=1:INPUT "DEHN.FAFT-X (1-4)";DX
 600 DY=1:INPUT "DEHN.FAFT-Y (1-4)";DY
 610 INPUT "RADIUS (30- 450) : ";RA
 620 LPRINT CHR$ 27+"b";
 622 REM GRAPH
 700 REM ----------
 710 DIM A$(0)*79
 720 "J"INPUT "TEXT: ";A$(0)
 730 N=LEN A$(0):SW=RA/N
 740 REM ---SPEZIELL: SPIRALSCHRIFT
 750 OO=360/N:O=90+2*OO
 760 REM ---
 770 LPRINT "M";UX;",";UY:LPRINT "I"
 771 REM LPRINT "M";UX;",";UY:PX=-RA:PY
=0:LPRINT "I"
 772 REM GLCURSOR (UX,UY):SORGN
 780 GOSUB 1500
1000 REM ----------
1010 REM --- DRUCK EINER ZEILE A$(Z)
1020 FOR F=1TO N
1030 REM --- ASC-CODE FUER ZU DRUCKENDE
S ZEICHEN
1040 E$=MID$ (A$(Z),F,1):U=ASC E$:IF E$
=" "THEN 1160
1050 REM --- VERZWEIGUNG ZU DRU-ZEI
1060 AD=ST+(U-65)*&20
1070 REM ---
1080 "STRICH"A=PEEK AD:CALL QX,A
1085 REM "STRICH"GOSUB 100
1090 ON RGOSUB 2001,2002,2003,2004,2005
,2006,2007
1100 IF R=0GOSUB 2000
```

```
1110 X=DX*X:Y=DY*Y
1112 REM --- KORREKTURFAKTOR
1114 XF=X-INT X:GOSUB 1400
1116 YF=Y-INT Y:GOSUB 1410
1120 REM --- AUSDRUCK DES STRICHES
1130 IF P=0LPRINT "R";X;",";Y
1132 REM IF P=0RLINE -(X,Y),9
1140 IF P=1LPRINT "J";X;",";Y
1142 REM IF P=1RLINE -(X,Y),0
1150 IF Q=0LPRINT ",";:AD=AD+1:GOTO "ST
RICH"
1152 REM IF Q=0LET AD=AD+1:GOTO "STRICH
"
1160 GOSUB 1500
1170 NEXT F
1180 REM ---
1190 LPRINT "H":LPRINT "M";-UX;",";-UY
1192 REM GLCURSOR (0,0):GLCURSOR (-UX,-
UY)
1200 BEEP 3,99,99:LPRINT CHR$ 27+"a":EN
D
1202 REM BEEP 3,99,99:TEXT :END
1400 IF XF<0.5LET X=INT X:RETURN
1405 X=1+INT X:RETURN
1410 IF YF<0.5LET Y=INT Y:RETURN
1415 Y=1+INT Y:RETURN
1490 REM --- DRUCKPOSITION
1500 O=O-2*OO:RA=RA-30/(N/2):PY=RA*SIN
(O+90):PX=RA*COS (O+90)
1510 LPRINT "M";PX;",";PY:RETURN
1512 REM GLCURSOR (PX,PY):RETURN
1990 REM --- KOORDINATEN D. STRICHES
2000 X=L*G*COS O:Y=L*G*SIN O:RETURN
2001 X=L*G*(COS O+COS (O+90)):Y=L*G*(SI
N O+SIN (O+90)):RETURN
2002 X=L*G*COS (O+90):Y=L*G*SIN (O+90):
RETURN
2003 X=L*G*(COS (O+90)+COS (O+180)):Y=L
*G*(SIN (O+90)+SIN (O+180)):RETURN
2004 X=L*G*COS (O+180):Y=L*G*SIN (O+180
):RETURN
2005 X=L*G*(COS (O+180)+COS (O+270)):Y=
L*G*(SIN (O+180)+SIN (O+270)):RETURN
2006 X=L*G*COS (O+270):Y=L*G*SIN (O+270
):RETURN
2007 X=L*G*(COS (O+270)+COS O):Y=L*G*(S
IN (O+270)+SIN O):RETURN
5000 QX=&F00
5010 POKE QX,&04,&FD,&C8,&B9,&07,&F1,&A
E,&79,&5A,&FD,&8A,&FD,&C8,&B9,&38,&D9
5020 POKE (QX+&10),&AE,&79,&8A,&FD,&8A,
&FD,&C8,&B9,&40,&D5,&D5,&AE,&79,&7A,&FD,
&8A
5030 POKE (QX+&20),&FD,&C8,&B9,&80,&F1,
&D9,&AE,&79,&82,&FD,&8A,&F9,&9A
5040 RETURN
```

```
740:REM ---SPEZIEL
    L: SPIRALSCHRI
    FT
750:OO=360/N:O=90+
    3*OO
1500:O=O-3*OO:RA=
     RA-40/(N/3):
     PY=RA*SIN (O
     +90):PX=RA*
     COS (O+90)
```

```
740:REM ---SPEZIEL
    L: ZIRKULARSCH
    RIFT
750:OO=360/N:O=90+
    OO
1500:O=O-OO:PY=RA
     *SIN (O+90):
     PX=RA*COS (O
     +90)

515:GOSUB 5000
1080:"STRICH"A=
     PEEK AD:CALL
     QX,A
```

```
10 REM --- ZPLOT/ MZ 800
20 REM (C) Winfied Meyer, Hasloh
30 GOTO800
100 REM --- BYTE SPLITTEN
110 REM --- LAENGE (BIT 0-2)
120 A=PEEK(AD)
130 B=A/2:C=A-2*INT(B)
140 L0=C
150 A=INT(B)
160 B=A/2:C=A-2*INT(B)
170 L1=C*2
180 A=INT(B)
190 B=A/2:C=A-2*INT(B)
200 L2=C*2*2
210 L=L0+L1+L2
220 REM --- RICHTUNG (BIT 3-5)
230 A=INT(B)
240 B=A/2:C=A-2*INT(B)
250 R0=C
260 A=INT(B)
270 B=A/2:C=A-2*INT(B)
280 R1=C*2
290 A=INT(B)
300 B=A/2:C=A-2*INT(B)
310 R2=C*2*2
320 R=R0+R1+R2
330 REM --- PEN UP/DOWN
340 A=INT(B)
350 B=A/2:C=A-2*INT(B)
360 P=C
370 REM --- ZEICHENENDE?
380 A=INT(B)
390 B=A/2:C=A-2*INT(B)
400 Q=C
410 RETURN
420 REM -----------------
790 REM --- DRUCKEN
800 CLS:CLR:PMODE GR
810 INPUT"STARTADR. FUER DRUCKZEI.-TAB: ";ST
820 INPUT"SCHRIFTGROESSE: ";G
840 INPUT"SCHRIFTFARBE: ";SF
850 PCOLOR SF
860 INPUT"WINKEL: ";O:W=O*X/180
900 DIM A$(0)
910 INPUT"TEXT: ";A$(0)
980 REM --- DRUCK EINER ZEILE
990 PMOVE50,-100:HSET
1000 FORF=1TOLEN(A$(0))
1010 REM --- CODE-NR F. ZU DRUCK. ZEICHEN
1020 E$=MID$(A$(0),F,1):U=ASC(E$):IFE$=" "HSET:GOTO1130
1030 REM --- ADR. IN DRUCKZEICHEN-TAB
1040 AD=ST+(U-65)*32
1045 HSET
1050 REM --- STRICH
1060 GOSUB100
1070 ONRGOSUB2001,2002,2003,2004,2005,2006,2007
1080 IFR=0GOSUB2000
1090 REM --- AUSDRUCK DES STRICHES
1100 IFP=0RMOVE X,Y
1110 IFP=1RLINE X,Y

1120 IFQ=0LETAD=AD+1:GOTO1060
1130 PHOME:PMOVE8*G,0
1140 NEXTF
1145 PMOVE-50,100:PMODE TN
1150 END
1999 REM --- KOORDINATEN-TRANSFORM.
2000 X=L*G*COS(W):Y=L*G*SIN(W):RETURN
2001 X=L*G*(COS(W)+COS(W+X/2)):Y=L*G*(SIN(W)+SIN(W+X/2)):RETURN
2002 X=L*G*COS(W+X/2):Y=L*G*SIN(W+X/2):RETURN
2003 X=L*G*(COS(W+X/2)+COS(W+X)):Y=L*G*(SIN(W+X/2)+SIN(W+X)):RETURN
2004 X=L*G*COS(W+X):Y=L*G*SIN(W+X):RETURN
2005 X=L*G*(COS(W+X)+COS(W-X/2)):Y=L*G*(SIN(W+X)+SIN(W-X/2)):RETURN
2006 X=L*G*COS(W-X/2):Y=L*G*SIN(W-X/2):RETURN
2007 X=L*G*(COS(W-X/2)+COS(W)):Y=L*G*(SIN(W-X/2)+SIN(W)):RETURN
```



```
9998 REM --- TEST (3 ZEICHEN)
9999 REM * ACHTUNG * STARTEN SIE DAS TESTPROGRAMM ZEITLICH VOR ZPLOT MIT GOTO 10
000
10000 POKE$D000,97,66,97,82,68,100,82,74,122,110,84,98,65,122,110,65,227
10010 POKE$D020,$13,$4A,$7A,$6A,$DA
10020 POKE$D040,$13,$42,$49,$34,$69,$59,$D3
10030 END
10040 REM GEBEN SIE BEI ZPLOT ALS TEXT "ABC" EIN!
```

Bestellschein

Alles für SHARP Computer

Bestellschein — Bitte vollständig und lesbar ausfüllen, unterschreiben und einsenden an Fischel GmbH, Kaiser-Friedrich-Str. 54a, D-1000 Berlin 12

☐ Ich abonniere die Zeitschrift "Alles für Sharp Computer"von der nächsten erreichbaren Ausgabe an (Preis pro Jahr 72 DM, Ausland 84 DM, Luftpostzuschlag 12 DM).

Das Abonnement verlängert sich um ein Jahr zu den dann jeweils gültigen Bedingungen, wenn es nicht 2 Monate vor Ablauf schriftlich gekündigt wird.

☐ Ich bestelle folgende schon erschienene Exemplare von "Alles für Sharp Computer" (Stückpreis 6 DM, Ausland 7 DM):
Heftnr.: ... , ... , ... , ... , ...

Alle Preise incl. 7 % Mwst.

Der Gesamtbetrag von ..... DM

☐ liegt bar bei

☐ liegt als Verrechnungsscheck bei (schnellste Erledigung)

☐ wurde am ..... auf das Postgirokonto der Fischel GmbH, Kontonr. 461533-103, BLZ 10010010, Postgiroamt Berlin überwiesen (Bearbeitung nach Zahlungseingang)

☐ liegt (nur bei kleineren Beträgen) in Briefmarken oder internationalen Antwortscheinen bei.

Name,Vorname .....

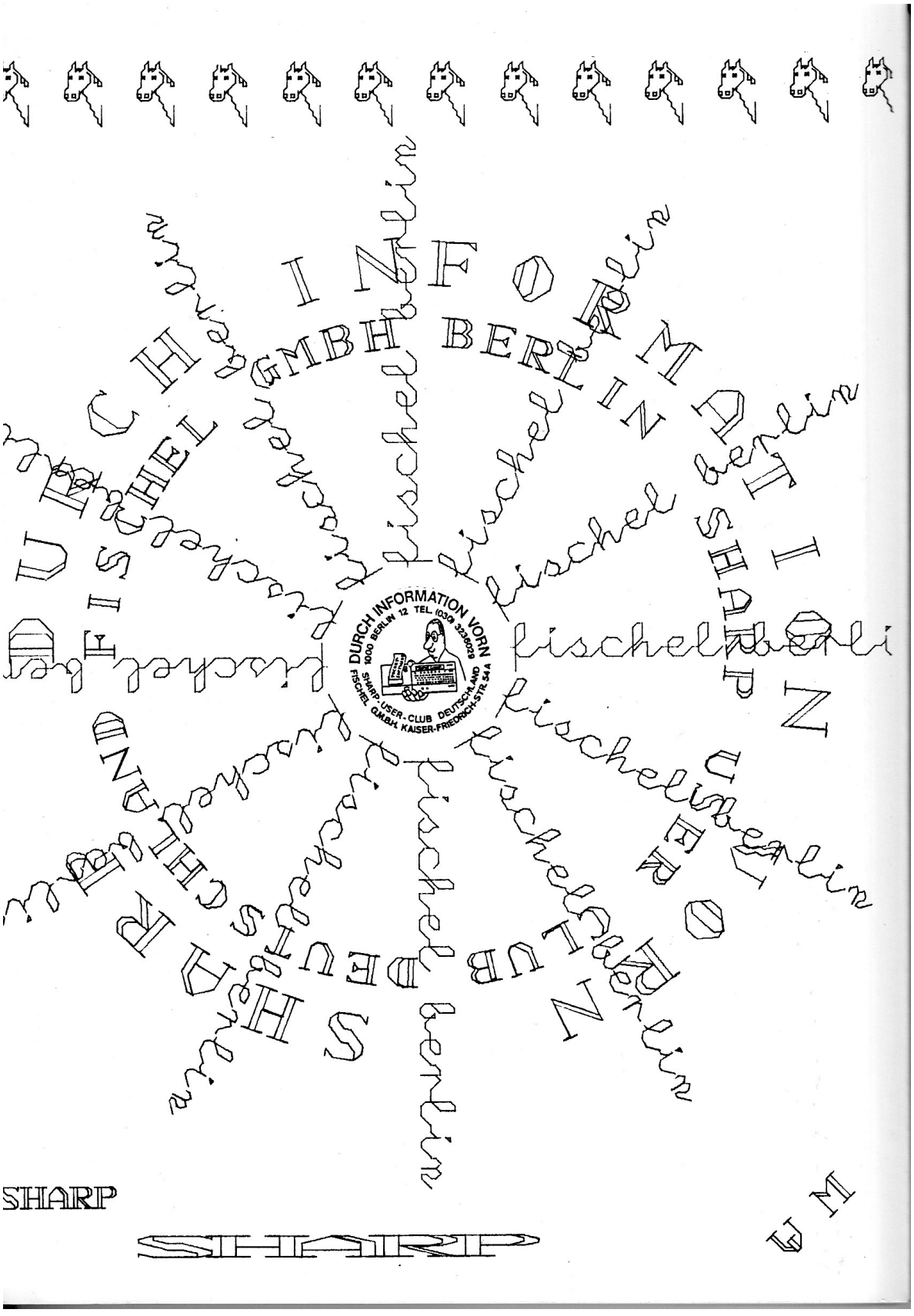
Straße .....

PLZ/Ort .....

Datum,Unterschrift .....

Mir ist bekannt, daß ich diese Bestellung innerhalb von 8 Tagen bei der Bestelladresse widerrufen kann. Zur Wahrung der Frist genügt die rechtzeitige Absendung. Ich bestätige dies durch meine zweite Unterschrift.

Datum,Unterschrift .....


DURCH INFORMATION VORN
1000 BERLIN 12 TEL (030) 3235029
SHARP-USER-CLUB DEUTSCHLAND
FISCHEL GMBH, KAISER-FRIEDRICH-STR. 54 A
SHARP
SHARP