1/APRIL 1983

öS 70,--
DM 11,--
sfr 10,50

TI-99 JOURNAL

DIE UHR IM TI-99
ARBEITEN MIT DEM MINI-MEMORY-MODUL
ASSEMBLER UND BASIC KURS
MODUL-VERLOSUNG

NEUERSCHEINUNG !

TEXAS INSTRUMENTS

Bei uns fangen Home Computer erst bei 16 K RAM an.

TI 99/4A offers a number of features that will be hard to find in comparable systems.
The TI 99/4A has 16 KB of internal ROM, expandable to a combined

to use Solid State Software (R) Modules. Over 600 software-programs are already available worldwide.

We want to expand the already existing comprehensive software on

Wir machen es Ihnen leichter.

TEXAS INSTRUMENTS

TI EXTENDED BASIC

Deutsches Handbuch zum TI-Extended Basic-Modul
für TI 99/4A **öS 390,—**

NEU:
PARSEC-Weltraumspiel
öS 895,—

PROGRAMME FÜR:
BAUSTATIK
FINANZBUCHHALTUNG
LOHNVERRECHNUNG

EPSON-DRUCKER
ab öS 12.000,—

SEMINARE für TI 99-Anwender
(10% Preisvorteil für Abonnenten!)

ALLE MODULE sofort lieferbar!
(Maximal 14 Tage Lieferzeit)

KONSOLE:	öS 6.900,—
PERIPHERIE BOX:	öS 4.990,—
DISK-KONTROLLER:	öS 4.990,—
DISK-DRIVE	öS 8.990,—
V-24	öS 3.900,—
32 K RAM EXPANSIONS UNIT	öS 4.990,—

Ing. Herbert BLEIER
Elektrogroßhandel
Texas Instruments-Rechner
1030 Wien, Kölblgasse 12/31
Tel. **78 47 19**

Das Zählwerk direkt auf der Kassette

Das Prinzip beruht auf der Idee eine für Speicherzwecke verwende te Kompaktkassette direkt als In formationsträger für den Zählwerkstand zu verwenden.

Der Sinn dieser Sache ist es, einerseits Rekorder ohne Zählwerk einsetzen zu können, andererseits die ermittelten Positionen auch bei der Benützung eines anderen Rekorders verwenden zu können, was bei eingebauten Zählwerken ja kaum der Fall sein kann, da bei verschiedenen Rekordern auch das Zählwerk bei gleicher Bandlänge verschiedene Resulatate anzeigen würde.

Die Handhabung ist recht einfach:

Programm eintippen (Zeile 10-80)
Kassette auf Seite B bis zum Anfang spulen.
Programm starten.
Nun werden auf der Seite B in absteigender Folge Dateien mit dem Inhalt 300 bis 1 abgespeichert. (Siehe Abb. 1)
Diese "Datei" füllt ziemlich genau eine Seite einer C60-Kassette (daher dauert dieser Vorgang auch ca. 30 Min.).
Von der A Seite aus gesehen (auf die Programme oder Dateien gespeichert werden) befindet sich nun diese Markierung aufsteigend (siehe Abb.2).
Also jedem Programmstück ist eine Zählwerkszahl zugeordnet.

Nachdem nun ein Programm auf der A Seite mit SAVE CS1 aufgebracht wurde, muß der Zählwerkstand ermittelt werden.
Also:

Kassette umdrehen (Seite B)
Eintippen der Zeilen 100 - 150
Programm starten.
Nach dem Erscheinen der Zählwerkszahl auf dem Bildschirm Rekorder sofort manuel stoppen.
Zu diesem Wert 5 addieren und man hat den nächsten freien Speicherplatz ermittelt. (5 muß deshalb addiert werden, weil bis zum Anzeigen des Zählwerkstandes einige Plätze durch die Anlaufzeit des Rekorders nicht berücksichtigt werden.)

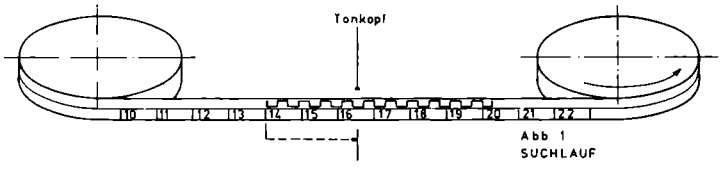


Abb 1
SUCHLAUF

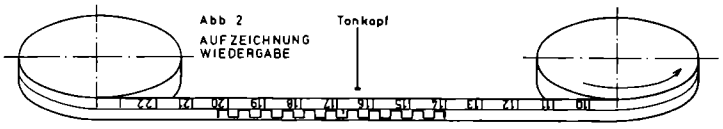


Abb 2
AUFZEICHNUNG
WIEDERGABE

```
10 OPEN #250:"CS1",INTERNAL,OUTPUT,FIXED
20 CALL CLEAR
30 FOR I=300 TO 1 STEP -1
40 A=I
50 PRINT #250:A
60 NEXT I
70 CLOSE #250
80 END
100 OPEN #250:"CS1",INTERNAL,INPUT ,FIXED
110 INPUT #250:A
120 PRINT A
130 GOTO 110
140 CLOSE #250
150 END
200 OPEN #250:"CS1",INTERNAL,INPUT ,FIXED
210 INPUT "NUMMER:":N
220 INPUT #250:A
225 PRINT A;
230 IF N=A THEN 250
240 GOTO 220
250 CLOSE #250
260 END
```

Bevor nun das nächste Programm mit SAVE CS1 auf die Kassette soll, ermittelt man mit dem Programm (Zeilen 200 - 260) auf der Seite B diese zuvor notierte freie Stelle durch Eingabe dieses Zählwerkstandes.
Das Band stoppt automatisch.
Kassette auf Seite A drehen und zu sicherndes Programm abspeichern.

Wenn sich diese Beschreibung auch etwas kompliziert anhört und die Vorgänge nicht auf Anhieb komplett einleuchten, so wird sich doch schon bald zeigen, daß die Handhabung nach kurzer Einübung sehr einfach und sicher ist.

Der große Vorteil, daß Bandstellen immer mit 100 %iger Treffsicherheit gefunden werden können, wird alleine schon ausreichen, selbst bei Rekordern mit Zählwerk auf dieses Verfahren um zusteigen.

Werbe Werkstatt Wiesmüller
1190 Wien, Krottenbachstr. 1 Tel. 37 41 58

ASSEMBLER

Leistungsfähig und rasch

TMS 9900 Maschinensprache

I. EINLEITUNG

Welcher Computerfreak empfindet nicht nach einiger Zeit bei seinen XBASIC-Spielen Bedauern, wenn Schüsse nicht treffen, weil Überdeckungen von Sprites durch den „langsamen" Interpreter nicht registriert werden?

Wer empfindet das Warten nicht als qualvoll, wenn sich der Computer bei der Lösung von Backtracking Problemen verabschiedet und lange Zeit Funkstille herrscht, bevor die Lösung am Bildschirm erscheint?

Was sagt der kommerzielle Anwender, wenn die Sortierung vieler Datensätze mehrere Minuten dauert?

Dem kann abgeholfen werden!

. . . die Lösung lautet TMS 9900

Assembler Maschinensprache.

Direkt vom BASIC oder XBASIC Programm aufgerufen.

Was bietet diese kraftvolle Sprache des TI 99/4A?

— superschnell auf Grund der 16-Bit Struktur

— 4 verschiedene Graphik Modes

— GRAPHIK Mode	24 x 32	Bildschirmposition
— MULTICOLOR Mode	48 x 64	Bildschirmposition
— EDITOR Mode	24 x 40	Bildschirmposition
— BIT MAP Mode	192 x 256	Bildschirmposition

— Vom weiteren Programmablauf unabhängige Ansteuerung von Sprites-, Sound- und Sprachkreisen.

— LINKING zu TI-BASIC, XBASIC und USCDp-Hochsprachen (Pascal).

TI stellt 3 Möglichkeiten zur Verfügung Maschinensprache auf dem TI-99/4A zu realisieren.

— EDITOR/ASSEMBLER	Modul und Disketten
— MINI MEMORY	Modul und Kassette
— USCD-SYSTEM	Erweiterungskarte und Disketten

Die kostengünstige Möglichkeit ist derzeit das MINI MEMORY Modul mit einem integrierten, batteriegepufferten 4 K RAM Speicher, welcher Maschinenprogramme aufnehmen kann.

Ein symbolischer LINE by LINE Assembler von Kassette geladen, ermöglicht die Eingabe von Maschinenprogrammen. Über dieses Modul wird in der nächsten Ausgabe ein ausführlicher Bericht gebracht werden.

Der USCDp-System Assembler ist derzeit in Österreich, ich glaube auch in Europa, noch nicht komplett verfügbar.

II. SOFTWARE und HARDWARE

Für Software-Entwicklungen ist das EDITOR/ASSEMBLER Paket die richtige Voraussetzung.

Es besteht aus einem Comand-Modul und 2 Disketten mit einer ca. 470 Seiten starken Dokumentation.

Die benötigte Mindestfiguration für Programmentwicklungen setzt sich zusammen aus:

— Konsole
— Erweiterungsbox
— 32 K Speicherkarte
— Disk-Kontroller
— 1—3 Disk Laufwerke

Die Programme können dann, mit geringfügigen Adressenmodifikationen, auch mit MINI MEMORY oder EXTENDED BASIC laufen.

Editor und Assembler können von Bildschirmen aus gewählt werden.

Die Comands sind leicht zu verstehen und einzuprogrammieren.

2.1 EDITOR

Der Editor wird automatisch von Diskette geladen und ist derzeit einer der besten am Markt erhältlichen.

Er besitzt einen EDIT- und einen COMAND-Mode und benützt ein 24 x 40 Zeichen Fenster, welches 3 Ausschnitte aus einem 80 Zeichen breitem Quellentext ermöglicht.

Mit dem EDITOR können auf komfortable Weise Quellencodes oder OBJECT CODES eingegeben, redigiert, Teile eingesetzt oder gelöscht, von Diskette geladen oder auf Diskette gespeichert werden.

Dabei stehen COMAND OPTIONEN wie ESCAPE, FIND, MOVE, INSERT, DELETE, COPY, SHOW, ADJUST und REPLACE zur Verfügung.

Jede Option wird durch Eintasten des ersten Buchstabens gewählt.

FIND — erlaubt das Auffinden eines beliebigen Strings innerhalb eines Textes, wobei [ANZAHL] (Startreihe, Endreihe) / TEXT / vorgegeben werden können.

REPLACE — diese Option ist ähnlich FIND, erlaubt aber zusätzlich den Austausch eines, mehrerer oder aller Strings gegen einen neuen.
z. B. /HELLO/ gegen /BYEBYE/ usw.

INSERT — lädt einen Textteil von einer Diskette und fügt ihn vor einer bestimmten Zeile ein.

DELETE — entfernt einen Zeilenblock.

MOVE — bewegt ein Programmsegment von einer Quelle zu einem Ziel, wobei die Quelle gelöscht wird.

COPY — wie MOVE, jedoch wird die Quelle nicht gelöscht.

SHOW — bewegt das Fenster so, daß die gewünschte Zeilennummer an der linken oberen Ecke des Fensters liegt.

ADJUST — fügt an den Quellentext links Zeilennummern an.

ESCAPE — damit wird der COMAND Mode verlassen und es kann wieder Text eingegeben werden.

Mit dem Editor, der auch Text im FIX 80 oder VAR 80 Format auf Diskette oder Drucker ausgibt, können auch 95% aller „word processing" Aufgaben gelöst werden.

2.2 ASSEMBLER

Der ASSEMBLER ist ein Programm, das mittels des EDITORS anstelle Maschinensprache Quellenprogramme in Maschinensprache Object Code übersetzt.

Der Object-Code wird auf Diskette und/oder Drucker geschrieben und kann später von der TI-99/4A CPU exekutiert werden.

2.3 PROGRAMMEXEKUTIONEN UND FEHLERSUCHE

Per Menü oder über Basic und XBasic kann ein einmal eingegebenes, editiertes und assembliertes Programm in die CPU geladen werden.

Falls es fehlerfrei ist, läuft es . . . hoffentlich!
Ansonsten verabschiedet sich der Computer und kann nur durch Ein- oder Ausschalten wieder zur Arbeit angeleitet werden.

Aber nicht verzweifeln.

Mit DEBUG stellt TI ein Debuggerprogramm zur Verfügung, das von Diskette geladen wird.

Es erlaubt das Setzen von Break Points, Inspektion des CPU-Memorys, der VDP-Register und vieles mehr.

2.4 BASICERWEITERUNGEN

Wird nach Einschub des EDITOR/ASSEMBLER Moduls in die Konsole von der Hauptwahlliste TI-BASIC gewählt. Es stehen folgende 7 zusätzliche BASIC Befehle zur Verfügung.

PEEK, PEEKV, CHARPAT, INIT, LOAD, LINK

CALL PEEK — liest Bytes aud dem CPU-RAM in Basic Variable.

CALL PEEKV — liest Bytes aus dem VDP-RAM in Basic Variable

CALL POKEV — schreibt Bytes in VDP-RAM.

CALL CHARPAT — schreibt Pattern Character Codes von ASC-II Zeichen in Basic Variable.

CALL INIT — initialisiert das CPU-MEMORY für Maschinensprachprogramme.

CALL LOAD — lädt Object Files oder Daten direkt in das CPU-MEMORY.

CALL LINK — stellt die Verbindung zwischen einem Basic und Maschinensprachprogramm her und übergibt bis zu 15 Parameter.

2.5 DOKUMENTATION

Die mitgelieferte Dokumentation (Englisch) ist sehr umfangreich.

Sie erläutert die Konsolenarchitektur ziemlich genau. Jeder Assemblerbefehl ist mit Beispielen exakt beschrieben.

Falls man schon eine Programmiersprache beherrscht, ist das Erlernen von TMS 9900 Assembler kinderleicht.
Gewisse nützliche Details, z. B. die Arbeit des Basic Interpreters, die genauen Pufferadressen für Basic Programme, Zeilen, Variable- und Stringspeicherstellen werden dezent verschwiegen.

Dies und einige, allerdings bei einer so umfangreichen Dokumentation, unvermeidliche Druckfehler, trüben manchmal die Freude an der Arbeit.

III. AUSBLICK

Diese Artikelserie wird sich in der Folge mit Assembler Programmierung des TI 99/4A beschäftigen. Ausgehend von einfachen Aufgaben (z. B. Uhr) bis zu komplizierten Graphikanwendungen wird an Hand von praktischen Beispielen Assemblerprogrammierung gezeigt.

Auf Konsolenarchitektur, Speicherorganisation wird dabei nicht eingegangen.

Darüber gibt das Assembler Manual und das 99'er JOURNAL erschöpfend Auskunft.

TI-99 JOURNAL

Editorial

Nachdem immer deutlicher wurde, daß fast alle TI-99/4(A) Besitzer isoliert ihrem Hobby nachgehen und mancher sogar durch Inserate in Fachmagazinen Kontakt zu Gleichgesinnten suchen mußte, entstand die Idee zur Gründung eines umfassenden Klubs.

Doch wie soll man immerhin tausende von TI-99 Benützer erreichen und verbinden, zumal durch die breite Streuung ein Klubbetrieb im herkömmlichen Sinn kaum in Frage kommt.
So wurde im Januar 1983 das TI-99 JOURNAL als Forum für Problemanalysen, Erfahrungsaustausch, Software-Vermittlung und beratendes Medium in allen Fragen der 99er-Schar, aus der Taufe gehoben.

Nachdem dem Initiator und Gründer, Thomas M. FIEDLER, von TI-ÖSTERREICH kooperative Unterstützung in Form von Informationen und Literatur zugesichert war, fand sich innerhalb kürzester Zeit ein sehr starker Mitarbeiterkreis.
Das Redaktionsteam versucht nicht nur Grundlagen, Tips und Tricks für die Arbeit am TI-99 zu bieten, sondern vor allem komplexes Spezialwissen in klarer, verständlicher Form in aufbauender Weise zu vermitteln.

So findet sich in der ersten Ausgabe (April 1983) ein BASIC-AUFBAUKURS (Dipl.Ing.Tatzl), die Einführung zur ASSEMBLER/MASCHINENSPRACHE-SERIE mit ausführlichen Programmbeispielen (Dipl. Ing.Dr. Peschetz), eine Berichtserie über die Hardware-Organisation (Rainer Bernert), graphisch veranschaulicht, sowie Modulbesprechungen, Kleinanzeigen, Grafikmöglichkeiten, Spielprogrammlistings, Lexikon, Kritiken und viele Tips und Hinweise.

Weiters besteht die Möglichkeit eigene Software zum Verkauf (Kleinanzeigen), Tausch oder Lizenz (50 % des Erlöses für den Autor) anzubieten (bzw. kleinere Programme und Kniffe zum Veröffentlichen zur Verfügung zu stellen), sowie aktiv durch Artikel, Besprechungen, Erfahrungsberichte und Kritiken am Inhalt mitzuarbeiten.

Klubmitglieder (das sind alle Abonnenten - es gibt KEINE Klubgebühr) erhalten außerdem eine Mitgliedskarte, mit der sie bei einigen Firmen EINKAUFSRABATTE erhalten.

Und außerdem besteht für jedes Klubmitglied die Möglichkeit für Problemlösungen die Mithilfe der TI-99 Redaktion in Anspruch zu nehmen.

Kontakte zu den deutschen und belgischen Klubs sind im entstehen. Ebenso wird an der Schaffung eines Klubtreffpunktes in Wien gearbeitet.

Auf dieser Seite des Herausgebers werden Sie auch in den nächsten Ausgaben auf Neuigkeiten (betreffend Mitarbeiter, Klubkontakte, usw.), ebenso wie auf Erweiterungen des TI-99 JOURNAL Service-Umfanges hingewiesen

Interessenten (Mitarbeit, TI-99 JOURNAL, Erfahrungsaustausch, usw.) schreiben an:

Thomas M. FIEDLER
TI-99 JOURNAL
Mollardgasse 32/3
1060 Wien

DAS PORTRÄT

Die rasche Entwicklung auf dem Gebiet der Mikroelektronik bringt auch bei Mikrocomputern jährlich eine Anzahl von Neuentwicklungen auf den Markt. Bei dieser Menge von Neuprodukten ist es selbst für Insider schwer Überblick zu bewahren. Auch versierten Verkaufskräften ist es oft nicht mehr möglich, Interessenten jene Informationen zu geben, die eine leichte Verkaufsentscheidung ermöglichen.

Hier ist guter Rat teuer.

Seit zwei Jahren gibt es nun im dritten Bezirk in WIEN, Kölblgasse 12, einen Fachhändler(Hr. Ing. BLEIER) für TEXAS INSTRUMENTS Produkte, der aufgrund seiner Spezialisierung auf diese Marke leichter imstande ist, Interessenten ausführlich zu informieren. Hier hat man ausgiebig Zeit für den Kunden. Empfehlenswert ist es aber, für ausführliche Demonstrationen einen Termin (Tel. 78 47 19) zu vereinbaren.

Auch findet man in diesem Kleinbetrieb lange und vergeblich gesuchtes Zubehör, sowohl für Rechner als auch Computer.

Ein MINI bei dem die Beratung und Betreuung MAXI sind.

INHALT

4 Zählwerk direkt auf der Kassette
5 Arbeiten in Assembler (mit Programml. in Extended BASIC + ASSEMBLER)
13 Mastermind - Das Spiel für Denker
15 V 24 - Schnittstelle
19 Grafik in BASIC + Extended BASIC
21 Minimemory - Modul: Testbericht
22 Bestellformular
23 Arbeiten mit dem Minimemory (Line by Line Assembler)
23 TI-99 als Heimorgel
25 BASIC-Grundkurs Teil 1:Einleitung
26 Die Uhr im TI-99 mit Minimemory-Modul
27 Tips für Umsteiger
29 Der kleine Riese (Der neue Compact-C.)
30 PARSEC - Der neue Hit
31 CPU des TI-99
34 Preisausschreiben
34 Impressum

IV. PROGRAMMBEISPIELE

4.1 GRUNDSÄTZLICHES

Wer Assembler programmiert, sollte sehr schnell seine BASIC Gewohnheiten vergessen.

Assembler ist eine compilierte Sprache und die voneinander getrennten Schritte sind

— Programmeingabe
— Compilierung

— Programmlauf

Die bei BASIC üblichen Fehlermeldungen gibt es nicht beim Editieren und nicht beim Ablauf. Bei Fehler in Maschinensprache „hängt" sich der Computer auf und kann nur mehr durch Ein- oder Ausschalten beruhigt werden.

Da hilft nur ein gutes Konzept und peinliche Genauigkeit bei der Programmerstellung.

Günstig ist ein von Pascal her bekannter, strukturierter Programmaufbau.

DEF Programmname (u)
REF Externe Referenzen

EQUATES (assembly Time Constants)
KONSTANTENDEKLARATIONEN
VARIABLENDEKLARATIONEN

Hauptprogrammkörper

Unterprogramme

END

Dieser Programmaufbau bietet den Vorteil, daß bei Übertragungen von Programmen von einen auf den anderen Modul nur der Programmkopf mit den EQUATES geändert werden muß (siehe XSORT$).

Das Programm selbst sollte in kleine, übersichtliche Prozeduren (max. 1 DIN A4 Seite) geglieder werden.

Diese können für sich selbst getestet werden, was die Fehlersuche ungemein erleichtert.

Diese Module werden nach Austestung aneinander gereiht, zusammen assembliert und das Programm läuft (. . . vielleicht?).

4.2 BEISPIEL 1 — CLOCK TEST

Auf CPU-Adresse >8379 liegt ein Byte, das bei jedem VDP-Interrupt (1/60 Sekunde) inkrementiert wird. Sind 255 Impulse erreicht, beginnt es wieder bei 0.

Dies kann man sich zunutze machen, um eine SOFTWARE-UHR zu programmieren.

Das Prinzip ist einfach:
— Timer rücksetzen (>00)
— nach 60 Impulsen einen Sekundenzähler inkrementieren
— nach 60 Sekunden einen Minutenzähler inkrementieren
— usw.

Und das Schöne an der Sache: Der Timer arbeitet weitgehend unabhängig vom anderen Programmablauf. Denn bevor 60 Impulse inkrementiert sind, lassen sich zig Befehle ausarbeiten.

Das Programm zeigt nur den Teil, wie Sekunden ermittel und auf dem Bildschirm angezeigt werden.

Der Rest ist reine Übungssache!

4.3 BEISPIEL 2 — STRINGSORT

Dieses Beispiel erfüllt den Wunsch, eine schnelle Sortierroutine ins BASIC oder XBASIC Programm einzubauen.

Es sortiert Datensätze im SUPER SHELLSORT ALLGORYTHMUS, wie ihn das Basic Beispiel 4.2 1 angibt.

Der Quellencode für den SHELLSORT ist im Listening 4.2.2 wiedergegeben.

Vergleicht man die Länge eines Textes mit dem 10 Zeilen Basic Programm, so fragt man sich unwillkürlich . . . welch ein Aufwand.

Und warum dies?

Allein der Vergleich 2-er Strings belegt mehr Zeilen Assemblerprogramm.

Die Antwort lautet Schnelligkeit.

Der Quellencode ist selbstdokumentierend.
Für auf- und absteigende Sortierung wurde die gesamte Programmsequenz wiederholt.

Das verkürzt die Rechenzeit, da zusätzliche Abfragen in den Programmschleifen vermieden werden.

Das Programm benützt für die Adressierung von Variablen im Speicher, für Fehlermeldungen und Integerfunktionen modulresidente Unterprogramme.

X-BASIC Modul und MINI-MEMORY Modul erlauben keine externen Programmreferenzen.

Hier wird auf eine absolute CPU-Adresse verwiesen, unter der Arbeitsregisterzeiger und Programmeintrittspunkt abgelegt ist.

Das UP findet sich in einem der ROMS im Modul.

Damit das Programm mit MINI MEMORY läuft müssen nur die EQUATES zu Beginn des Programmkopfes geändert werden.

Nach dem Assemblieren wird das Maschinenprogramm über

CALL INIT

CALL LOAD („DSK1", XSORT$)

geladen.

```
100 REM  VDP-INTERRUPTTIMER
110 REM  ******************
120 REM   .
130 REM  CLOCK
140 REM
150 REM  BY  §APESOFT
160 REM
170 REM
180 IMAGE ##::##::##
190 REM
200 TIMER=-31879
210 REM
220 CALL INIT
230 DISPLAY AT(8,5)ERASE ALL:"VDP-INTERRUPTTIMER"
240 DISPLAY AT(14,8):USING 180:STUNDE,MINUTE,SEKUNDE
250 DISPLAY AT(20,12):"BY": :"    § A P E S O F T"
260 CALL LOAD(TIMER,NULL)
270 CALL PEEK(TIMER,TIME)
280 IF TIME<48 THEN 270 ELSE CALL LOAD(TIMER,NULL)
290 SEKUNDE=SEKUNDE+1
300 IF SEKUNDE=60 THEN SEKUNDE=0 :: MINUTE=MINUTE+1
310 IF MINUTE=60 THEN MINUTE=0 :: STUNDE=STUNDE+1
320 IF STUNDE=24 THEN STUNDE=0
330 DISPLAY AT(14,8):USING 180:STUNDE,MINUTE,SEKUNDE
340 GOTO 270
```

```
100 REM   SORTIEREN EINES STRINGARRAYS  (ASSEMBLER)
110 REM   ****************************************
120 REM
130 REM   STRINGSORT BY §APESOFT
140 REM
150 OPTION BASE 1
160 DIM A$(100)
170 CALL CLEAR
180 PRINT "SORTIEREN EINES $ARRAYS":"======================": :"ASSEMBLERTEST":
 : :
190 READ MAXN
200 PRINT : :
210 INPUT "ANZAHL DER ELEMENTE (<=100)?":N
220 N=INT(N)
230 IF N>MAXN THEN 210
240 RESTORE 730
250 FOR I=1 TO N
260 READ A$(I)
270 NEXT I
280 T$="UNSORTIERTER ARRAY:"
290 GOSUB 620
300 INPUT "BASIC=1  ASSEMBLER=2? ":X
310 IF (X<1)+(X>2)THEN 300
320 ON X GOSUB 420,560
330 T$="SORTIERTER ARRAY:"
340 GOSUB 620
350 INPUT "WEITER (J/N)? ":M$
360 IF M$="J" THEN 200
370 STOP
380 REM
390 REM SHELLSORT BASIC
400 REM ***************
410 REM
420 M=N-1
430 M=INT(M*.3+.5)
440 FOR K=1 TO M :: FOR I=K+M TO N STEP M
450 X$=A$(I)
460 FOR J=I-M TO 1 STEP -M
470 IF X$>=A$(J)THEN 490 ELSE A$(J+M)=A$(J)
480 NEXT J
490 A$(J+M)=X$
500 NEXT I :: NEXT K
510 IF M>1 THEN 430 ELSE RETURN
520 REM
530 REM  SHELLSORT ASSEMBLER
540 REM  ********************
550 REM
560 CALL LINK("SORT$",A$(),N,3)
570 RETURN
580 REM
590 REM ARRAY AUSDRUCKEN
600 REM ****************
610 REM
620 PRINT : : :T$:SEG$("--------------------",1,LEN(T$)): :
630 FOR I=1 TO N
640 PRINT A$(I)
650 NEXT I
660 PRINT : :
670 RETURN
680 REM
690 REM DATEN
700 REM *****
710 REM
720 DATA 100
730 DATA MICHAELA,HANNES,ALMA,OLLIVER,ALEXANDER,KIM,GINGER,OMAMA,OPAPA,HERBERT
740 DATA THEOBALD,HEIM,RAIMUND,ULRICH,ANTON,MECHTHILD,WILLIBALD,EUGEN,GOTTFRIED,
ERICH
750 DATA GOTTFRIED,OLGA,NOAB,HEINRICH,KAMILLUS,EGON,MARIA,IRMGARD,FRIEDRICH,BERN
ULF
760 DATA ALEXANDER,MARGARETA,JULIA,MAGDALENE,MAGDALENA,BRIGITTA,CHRISTINA,JAKOB,
ANNA,RUDOLF
770 DATA NAZARIUS,BEATRIX,INGEBORG,IGNAZ,ALFONS,STEFAN,HANNES,LYDIA,MICHAELA,RAI
NER
780 DATA OLIVER,OSWALD,ALBERT,GUSTAV,JOHANN,ROMAN,LAURENTIUS,SUSANNA,HILARIA,RAD
EGUNG
790 DATA ALMA,AIDA,MEINHARD,STEFAN,MARIA,HYAZINTH,HELENE,SEBALD,BERHARD,MAXIMILI
AN
800 DATA PHILLIP,MARIA,BARTHOLOMAEUS,LUDWIG,MARGARETA,MONIKA,AUGUSTIN,SABINE,FEL
IX,KIM
810 DATA RAIMUND,RUTH,INGRID,GINGER,ROSALIA,GREGOR,ALBERT,BEATE,PETRUS,EBERHARD
820 DATA ROMAN,ATILA,BRUNO,HORST,THERESIA,MARIA-MAGDALENA,GUENTHER,CORDULA,JOHAN
NES,LEO
830 END
```

```
*       ******************
*       ******************
*       **              **
*       **  SHELLSORT   **
*       **              **
*       ******************
*       ******************
*
*
*
*       Programmname  §XSORT$S/§XSORT$     Version 1.0
*
*       TMS 9900 ASSEMBLER
*
*       By  §APESOFT
*
*       1983-03-01
*
*
*
*       ********************************
*       * CALL LINK("XSORT$",A$(),N,W) *
*       ********************************
*
*
*       Ubernimmt einen Stringarray A$() von XBASIC mit N Elementen und
*       ihn nach den Kriterien des SUPERSHELLSORTS  aufsteigend (W<>0) oder
*       absteigend (W=0). Dabei bleibt der Array im VDP RAM. Die CPU wird
*       nur als Buffer fuer den Vergleich benuetzt.
*       Dies kostet zwar Zeit, weil bei jedem Vergleichen und
*       Vertauschen UTILIES benuetzt werden muessen, aber
*       die Speicherplaetze werden optimal eingesetzt.
*
*       Die Strings koennen maximal 255 Bytes lange sein.
*       Die Groesse des Arrays wird vom Programm nicht begrenzt.
*
*
        IDT  'XSORT$'
*
*       ******************
*       * DEKLARATIONEN *
*       ******************

        DEF  SORT$
*
*
*       XBASIC EQUATES     ->   MINI MEMORY EQAUATES
*
NUMREF EQU  >200C         ->   >6044      Numerische Parameter Uebernahme
STRREF EQU  >2014         ->   >604C      String Parameter Uebernahme
STRASG EQU  >2010         ->   >6048      String Parameter Uebergabe
XMLLNK EQU  >201B         ->   >601C      Linking zu ROM Routinen
ERR    EQU  >2034         ->   >6050      Fehler Report
CFI    EQU  >12B8         ->   >1200      FPOINT -> INTERGER
ERRBA  EQU  >1C00         ->   >1600      Bad Argument
ERRIAL EQU  >1F00         ->   >1000      Incorrect Argument List
*
*
GPLWS  EQU  >83E0         GPL Worksspace Register
STATUS EQU  >837C         GPL Status Byte
ARGNMR EQU  >8312         Argumenteanzahl im LINK Statement
FAC    EQU  >834A         Floating Point Akkumulator
*
*       VARIABLE, KONSTANTE
*
MYWS   BSS  32            Eigene Arbeitsregister
A1$    BSS  256           Buffer fuer A1$
A2$    BSS  256           Buffer fuer A2$
W      BSS  2             Flag fuer auf- oder abgsteigende Sortierung
TWO    DATA 2
THREE  DATA 3             Wort fuer DIV
*
       EVEN
*
*
*       ERLAUTERUNGEN ZUR REGISTERBELEGUNG
*       **********************************
*
*       R0   Elementnummer des Arrays
*       R1   Parameternummer des LINK Statements
*            oder Indexzaehler fuer Vergleich (A1$)
*       R2   Zeiger zum Bufferanfang
*            oder Indexzaehler fuer Vergleich (A2$)
*       R3   Laenge von String A1$
*       R4   Laenge von String A2$
*       R5   Anzahl N der Elemente des Stringarrays
*       R6   Schleifenvariable I
*       R7   Schleifenvaribale J
*       R8   Schleifenvariable K
*       R9   Schleifenvariable M
*       R10  Enthaelt Rest der Integerdivision
*
*
*       ************
*       * PROGRAMM *
*       ************
*
*
SORT$  LWPI MYWS          Lade eigene Arbeitsregister
*
       CB   §ARGNMR,§THREE+1  Parameteranzahlkontrolle
       JEQ  CNTINE            3 Argumente sind notwendig
       LI   R0,ERRIAL         'INCORRECT ARGUMENT LIST'
       JMP  ERROR
*
CNTINE CLR  R0            Uebernahme von W
       LI   R1,3          3 LINK Parameter fuer SORT Wahl
       BLWP §NUMREF
       MOV  §FAC,§W
*
       LI   R1,2          2 Parameter im LINK Statement
       BLWP §NUMREF       Anzahl N der zu sortierenden Arrayelemente
       BLWP §XMLLNK       in FAC laden und in Integer konvertieren
       DATA CFI
```

```
*
        MOV  §FAC,R5       Anzahl N -> R5
        C    R5,§TWO       Fehler falls N<3
        JGT  DOSORT
*
        LI   R0,ERRBA      'BAD ARGUMENT'
ERROR   BLWP §ERR          Fehlermeldung
*
DOSORT  MOV  §FAC,R9       Startwert fuer Schleifenparameter M
        DEC  R9
        MOV  §W,§W         W<>0?  Ja...... aufsteigend
        JEQ  LOOPM2               Nein.... absteigend
*
*
*       AUFSTEIGENDE SORTIERUNG (W<>0)
*       ******************************
*
LOOPM1  MOV  R9,R10        Fuehrt die Operation
        AI   R10,2         M=INT((M+2)/3) aus
        CLR  R9
        DIV  §THREE,R9

*
        CLR  R8            Start fuer Schleife  K
LOOPK1  INC  R8            K=K+1
        C    R8,R9         K>M?
        JGT  NEXTM1        Ja -> Neuen M-Wert bestimmen
*
        MOV  R8,R6
LOOPI1  A    R9,R6         Start fuer Schleife  I  (I=K+M) Step +M
        C    R6,R5         I>N?
        JGT  LOOPK1        Ja -> Neuen K-Wert bestimmen
*
        MOV  R6,R0         Arrayparameter (I)
        LI   R1,1          LINK Parameter  1
        LI   R2,A1$        Lade Zeiger fuer String A1$
        SETO *R2           Maximale Stringlaenge 255 Bytes
        BLWP §STRREF       Hole String A1$ aus VDP Ram
*
        MOV  R6,R7
LOOPJ1  S    R9,R7         Start fuer Schleife  J  (J=I-M) Step -M
        CI   R7,>1         J<1?
        JLT  ASG11         Ja -> Zuweisung A$(J+M)=A$(J)
*
*       VERGLEICHE STRING A1$ MIT A2$
*
        MOVB §A1$,R3
        SRL  R3,8          Laenge des Strings A1$ -> R3
*
        MOV  R7,R0         Arrayparameter (J)
        LI   R1,1          LINK Parameter  1
        LI   R2,A2$        Lade Zeiger fuer String A2$
        SETO *R2           Maximale Stringlaenge 255 Bytes
        BLWP §STRREF       Hole String A2$ aus VDP Ram
*
        MOVB §A2$,R4
        SRL  R4,8          Laenge des Strings A2$ -> R4
```

```
*
        LI   R1,A1$+1      Zeiger zum Beginn von A1$
        LI   R2,A2$+1      Zeiger zum Beginn von A2$
COMP1   CB   *R1+,*R2+     Byteweiser Vergleich von String A1$ und A2$
        JEQ  NEXT11
        JH   ASG11         A1$>=A2$ -> Tauschoperation ASG2
        JMP  ASG21         A1$< A2$ -> Tauschoperation ASG1
*
NEXT11  DEC  R3            Bytezaehler fuer A1$ reduzieren
        JNE  NEXT21        R3<>0? Ja... Sprung zu String A2$
        CB   §A1$,§A2$     Nein... Strings gleich lange?
        JEQ  ASG11         A1$= A2$ -> Tauschoperation ASG1
        JMP  ASG21         A1$< A2$ -> Tauschoperation ASG2
*
NEXT21  DEC  R4            Bytezaehler fuer A2$ reduzieren
        JNE  COMP1         R4<>0? Ja... Bytes vergleichen
        CB   §A1$,§A2$     Nein... Strings gleich lange?
        JEQ  ASG11
*
ASG21   MOV  R7,R0         Arrayparameter
        A    R9,R0         (J+M)
        LI   R1,1          LINK Parameter   1
        LI   R2,A2$        Lade Zeiger fuer String A2$
        BLWP §STRASG       Weise String A2$ A$(J+M) zu
*
NEXTJ1  JMP  LOOPJ1        Erhoehe Index J
*
ASG11   MOV  R7,R0         Arrayparameter (J+M)
        A    R9,R0         (J+M)
        LI   R1,1          LINK Para,eter
        LI   R2,A1$        Lade Zeiger fuer String A1$
        BLWP §STRASG       Weise String A1$ A$(J+M) zu
*
NEXTI1  JMP  LOOPI1        Erhoehe Index I
NEXTK1  JMP  LOOPK1        Erhoehe Index K
*
NEXTM1  CI   R9,>1         M>1?
        JGT  LOOPM1        Berechne naechstes M
        JMP  BSCRTN
*
*
*       ABSTEIGENDE SORTIERUNG  (W=0)
*       *****************************
*
LOOPM2  MOV  R9,R10        Fuehrt die Operation
        AI   R10,2         M=INT((M+2)/3) aus
        CLR  R9
        DIV  §THREE,R9
*
        CLR  R8            Start fuer Schleife  K
LOOPK2  INC  R8            K=K+1
        C    R8,R9         K>M?
        JGT  NEXTM2        Ja -> Neuen M-Wert bestimmen
*
        MOV  R8,R6
LOOPI2  A    R9,R6         Start fuer Schleife  I  (I=K+M) Step +M
        C    R6,R5         I>N?
        JGT  LOOPK2        Ja -> Neuen K-Wert bestimmen
```

CALL LINK („SORT$,", AS() N, 1) verweist dann vom XBASIC ins Maschinenprogramm.
Nach Rückkehr zu BASIC ist der Stringarray A$() sortiert.

Anmerkung:
In Kürze wird bei CAPESOFT eine Diskette mit UTILIES für den kommerziellen Anwender mit von Basic oder X-Basic aufrufbaren Routinen vorliegen. Sorts, Matrixoperationen, Differential und Integralrechnung und Zeichendefinitionen.

V. SCHLUSS

Mit dem EDITOR/ASSEMBLER Paket stellt TI ein ausgewähltes Programmwerkzeug für Software Entwickler und fortgeschrittene Hobbyanwender zur Verfügung.

Es ermöglicht auf einfache und elegante Weise den Zugriff zu TMS 9900 Assembler und damit die Entwicklung superschneller Programme für Hobby und Beruf.

Der nächste Artikel wird sich ebenfalls mit einem Modul für Assemblersprache beschäftigen.

Dem sogenannten MINI MEMORY und als Programmbeispiel eine Uhr mit Melodiewecker und Stoppfunktionen.

Dabei werden Graphikprozessor und Soundprozessor erläutert.

TIPS BY CAPESOFT

Jeder Hobbyprogrammierer weiß, daß er über CALL SOUND Befehle Zeitnehmung in seine Programme einbauen kann.

z. B.

```
100  FOR I = 10 TO 0 STEP —1
110  CALL SOUND (1000, 22000, 20)
120  PRINT I
130  NEXT I
140  CALL SOUND (1, 22000, 20)
150  PRINT „START"
```

Die Sache hat nur einen Haken, eine von der Zeitermittlung unabhängiger Programmablauf ist nur schwer möglich.

Mit XBASIC und Speichererweiterung oder MINI MEMORY schafft das folgende Programmbeispiel Abhilfe.

Auf CPU-Adresse —31879 sitzt ein Interruptimer, der jede 1/60 Sekunde inkrementiert wird und das unabhängig vom anderen Programmablauf.

Da der BASIC Interpreter nicht immer Interrupts zuläßt, muß durch Stoppen festgestellt werden, wie viele Impulse einer Sekunde entsprechen.

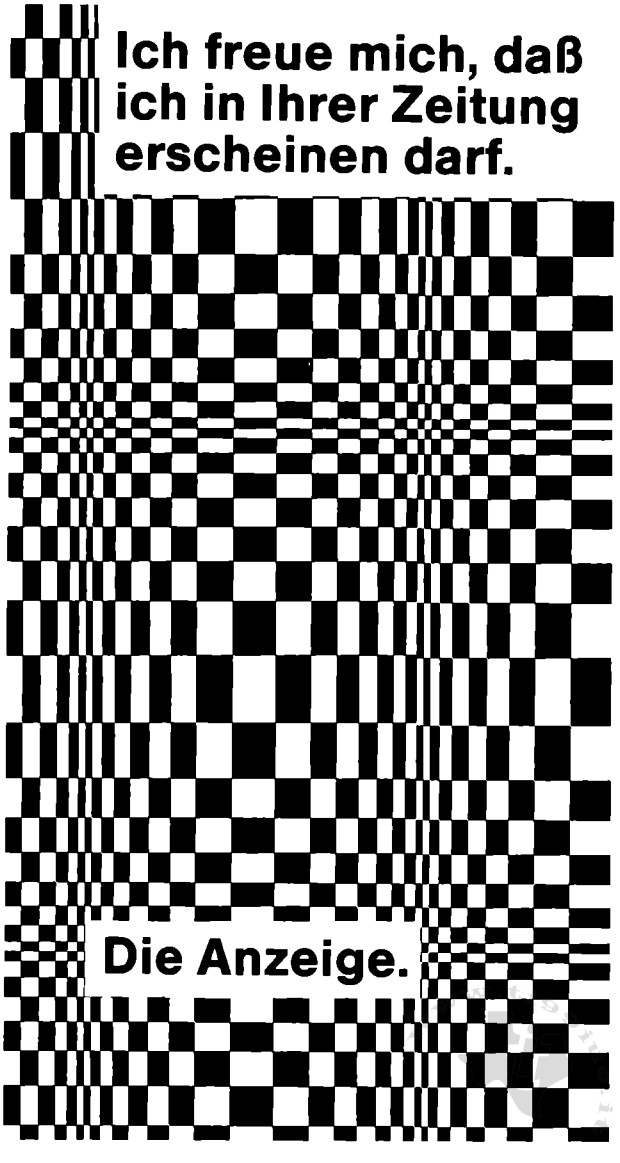
So läßt sich einfach ein BASIC Programm für eine Uhr schreiben.

CAPEGRAFIC

High Resolution Graphik für den TI-99/4A.
Superschnelles TMS 9900 Assembler Programm, von XBASIC und RAM Erweiterung oder BASIC und MINI-MEMORY ansprechbar.
Graphiken auf dem gesamten Bildschirm in allen Farben. Jedes Pixel einzeln sehbar.
Graphik und Text mischbar.
Leistungsfähige Basicbefehle wie
SETTO, RESET, IFSET, TURN, TURNTO, MOVE, REMOVE, MOVETA, REMOVETO, CIRCLE, CLCIRCLE, ELLIPS, CLELLIPS, RECT, CLRECT, WRITE, VALUES, GSAVE, GLOAD, GRAFIC, WINDOW und BYEBE.
Besprechung im nächsten Heft.

Bezug auf Diskette oder Kassette über den Fachhandel oder den TI99-JOURNAL-KLUB.

●



MASTER MIND

Das Spiel für Denker

Der Computer sucht sich eine Zahl deren Anzahl an Stellen (1 bis 10) wählbar ist. Ebenso wählbar ist die Anzahl der Variationen, d.h. wieviele verschiedene Ziffern (von 0 bis 9) vorkommen können. 10 bedeutet das alle Variationen möglich sind.

Nun wird der erste Versuch eingegeben (bei Anzahl 4 und Variationen 10 zum Beispiel: 2375). Der Computer bewertet diese Eingabe mit zwei Zahlenwerten: Der erste mit RIGHT (richtig) bezeichnete gibt an, wieviele Ziffern der geratenen Zahl auch in der Geheimzahl vorkommen. Der zweite mit PLACE (Platz) bezeichnete gibt Aufschluß darüber, wieviele Ziffern der geratenen Zahl auch an der selben Stelle wie in der Geheimzahl vorkommen.

Wenn man nicht mehr weiter weiß, so kann durch Eingabe von X statt der nächsten Versuchszahl die Lösung angezeigt werden. Es sollte aber nach einiger Übung nicht besonders schwierig sein nicht mehr als sechs Versuche zu benötigen.
Also viel Spaß.

```
*
        MOV  R6,R0        Arrayparameter (I)
        LI   R1,1         LINK Parameter  1
        LI   R2,A1$       Lade Zeiger fuer String A1$
        SETO *R2          Maximale Stringlaenge 255 Bytes
        BLWP §STRREF      Hole String A1$ aus VDP Ram
*
        MOV  R6,R7
LOOPJ2  S    R9,R7        Start fuer Schleife  J  (J=I-M) Step -M
        CI   R7,>1        J<1?
        JLT  ASG12        Ja -> Zuweisung A$(J+M)=A$(J)
*
*       VERGLEICHE STRING A1$ MIT A2$
*
        MOVB §A1$,R3
        SRL  R3,8         Laenge des Strings A1$ -> R3
*
        MOV  R7,R0        Arrayparameter (J)
        LI   R1,1         LINK Parameter  1
        LI   R2,A2$       Lade Zeiger fuer String A2$
        SETO *R2          Maximale Stringlaenge 255 Bytes
        BLWP §STRREF      Hole String A2$ aus VDP Ram
*
        MOVB §A2$,R4
        SRL  R4,8         Laenge des Strings A2$ -> R4
*
        LI   R1,A1$+1     Zeiger zum Beginn von A1$
        LI   R2,A2$+1     Zeiger zum Beginn von A2$
COMP2   CB   *R1+,*R2+    Byteweiser Vergleich von String A1$ und A2$
        JEQ  NEXT12
        JH   ASG22        A1$>A2$ -> Tauschoperation ASG22
        JMP  ASG12        A1$<A2$ -> Tauschoperation ASG12
*
NEXT12  DEC  R3           Bytezaehler fuer A1$ reduzieren
        JNE  NEXT22       R3<>0? Ja... Sprung zu String A2$
        CB   §A1$,§A2$    Nein... Strings gleich lange?
        JEQ  ASG22        A1$=A2$ -> Tauschoperation ASG22
        JMP  ASG12        A1$<A2$ -> Tauschoperation ASG12
*
NEXT22  DEC  R4           Bytezaehler fuer A2$ reduzieren
        JNE  COMP2        R4<>0? Ja... Bytes vergleichen
        CB   §A1$,§A2$    Nein... Strings gleich lange?
        JEQ  ASG12
*
ASG22   MOV  R7,R0        Arrayparameter
        A    R9,R0        (J+M)
        LI   R1,1         LINK Parameter   1
        LI   R2,A2$       Lade Zeiger fuer String A2$
        BLWP §STRASG      Weise String A2$ A$(J+M) zu
*
NEXTJ2  JMP  LOOPJ2       Erhoehe Index J


*
ASG12   MOV  R7,R0        Arrayparameter (J+M)
        A    R9,R0        (J+M)
        LI   R1,1         LINK Para,eter
        LI   R2,A1$       Lade Zeiger fuer String A1$
        BLWP §STRASG      Weise String A1$ A$(J+M) zu
*
NEXTI2  JMP  LOOPI2       Erhoehe Index I
NEXTK2  JMP  LOOPK2       Erhoehe Index K
*
NEXTM2  CI   R9,>1        M>1?
        JGT  LOOPM2       Berechne naechstes M
*
*
*       RUECKKEHR NACH BASIC
*
BSCRTN  CLR  R0
        MOVB R0,§STATUS   GPL STATUS Byte loeschen
        LWPI GPLWS        GPL Arbeitsregister laden
        B    §>0070       retour zu BASIC
*
        END
```

PROGRAMMABLAUF:

90-100 Farbe des Bildschirmes und der Zeichen von Satz 0-8
110 Datenfelddimensionierung
130-190 Anweisung fuer Auswahl und Abfrage
190-240 Erstellen der Geheimzahl
260-270 Display des Arbeitsfeldes
280 Eingabe des Versuches
290 Abfrage ob Loesung gewuenscht
300-320 Zerteilen des Versuches in Einzelziffern
330-400 Erechnen wie viele Stellen richtig und wieviele
        am richtigen Platz. Anzeigen des Ergebnisses.
410 Abfrage ob Platzierung und richtige Stellen gleich sind.
420-460 Abfrage nach neuem Spiel. Bei NEIN: STOP.
470 -500 Anschreiben der Loesung.
510-550 Abfrage nach neuem Spiel. Bei NEIN: END.

```
90 CALL SCREEN(2)
91 CALL COLOR(1,4,1)
92 CALL COLOR(2,4,1)
93 CALL COLOR(3,4,1)
94 CALL COLOR(4,4,1)
95 CALL COLOR(5,4,1)
96 CALL COLOR(6,4,1)
97 CALL COLOR(7,4,1)
98 CALL COLOR(8,4,1)
99 CALL COLOR(0,1,4)
100 REM CODEBREAKER XBASIC
110 DIM CODES$(8),GUESS$(8)
120 RANDOMIZE
130 DISPLAY AT(11,9)BEEP ERASE ALL:"MASTERMIND"
140 DISPLAY AT(19,1)BEEP:"ANZAHL D STELLEN?(1-8) 4"
150 DISPLAY AT(21,6)BEEP:"VARIATION? (1-10) 10"
160 ACCEPT AT(19,24)SIZE(-1)VALIDATE(DIGIT):CODES
165 IF CODES(1 OR CODES)8 THEN 160
170 ACCEPT AT(21,24)SIZE(-2)VALIDATE(DIGIT):DIGITS
175 IF DIGITS(1 OR DIGITS)10 THEN 170
180 IF CODES)DIGITS THEN DISPLAY AT(24,2)BEEP:"MEHR STELLEN ALS
ZIFFERN!" :: GOT
O 160
190 FOR A=1 TO CODES !CHOOSE CODES
200 CODE$(A)=STR$(INT(RND*DIGITS))
210 FOR B=0 TO A-1 !CHECK FOR DUPLICATES
220 IF CODE$(A)=CODE$(B)THEN 200
230 NEXT B
240 NEXT A
250 ROW=2
260 DISPLAY AT(1,1)ERASE ALL:"GUESS        RIGHT    PLACE"
270 DISPLAY AT(24,3):"LOESUNG: EINGABE VON X"
280 ACCEPT AT(ROW,1):C$
290 IF C$="X" THEN 470 !GIVE UP OR RESET
300 FOR D=1 TO CODES !BREAK UP GUESS
310 GUESS$(D)=SEG$(C$,D,1)
320 NEXT D
330 RIGHT,PLACE=0
340 FOR E=1 TO CODES !CHECK GUESS FOR CORRECTNESS
350 FOR F=1 TO CODES
360 IF CODE$(E)=GUESS$(F)THEN RIGHT=RIGHT+1 :: IF E=F THEN PLACE
=PLACE+1
370 NEXT F
380 NEXT E
390 DISPLAY AT(ROW,14):RIGHT
400 DISPLAY AT(ROW,22):PLACE
410 IF PLACE()CODES THEN ROW=ROW+1 :: IF ROW)21 THEN 470 ELSE 28
0
420 DISPLAY AT(23,1)BEEP:"GEWONNEN MIT";ROW-1;"VERSUCHEN."
430 DISPLAY AT(24,1)BEEP:"NEUES SPIEL?(J/N) J"
440 ACCEPT AT(24,19)SIZE(-1)BEEP VALIDATE("JN"):X$
450 IF X$="J" THEN 465
455 CALL CLEAR
460 STOP
465 CALL CLEAR :: GOTO 140
470 DISPLAY AT(23,1)BEEP:"LOESUNG IST" !LOSE,GIVE UP,OR RESET
480 FOR G=1 TO CODES
490 DISPLAY AT(23,12+G):CODE$(G)
500 NEXT G
510 DISPLAY AT(24,1)BEEP:"NEUES SPIEL?(J/N) J"
520 ACCEPT AT(24,19)SIZE(-1)BEEP VALIDATE("JN"):X$
530 IF X$="J" THEN CALL CLEAR :: GOTO 140
540 CALL CLEAR
550 END
```

V24 (RS232) Schnittstelle

Begriffserklärung

— Codierung[1])

Seit jeher verwendete man bei der Übermittlung von Nachrichten — mit Ausnahme der Sprechverbindungen — andere, dem entsprechenden System des Informationsaustauschens angepaßte Zeichen. Gleichviel, ob es sich dabei um Lichtsignale, Trommelschläge oder Morsezeichen handelt, ein auf beiden Seiten (Sender, Empfänger) vereinbarter Code (Schlüssel) muß hiebei festgelegt sein.
Beim Morsezeichen wird jedes zu übertragende Zeichen in ein elektrisches Signal umgesetzt, die bekannten Striche und Punkte, was bedeutet: langer Stromimpuls bzw. kurzer Stromimpuls.
Später hatte man jedem Zeichen (Buchstaben, Ziffern etc.) eine für alle Zeichen gleiche Anzahl an Stromschritten zugeordnet (Abb. 1). Die Anzahl der Stromimpulse, die ein Zeichen ergeben, richtet sich nach dem verwendeten Code.
Die Symbolik verbirgt sich nun dahinter, welches Muster diese gleich langen Schritte darstellen. jeder dieser Schritte gleicher zeitlicher Länge kann ein Strom-Schritt oder ein Kein-Strom-Schritt sein.
In der Digitaltechnik[2]) werden diese Zustände als logisch 1 bzw. logisch 0 bezeichnet (oder umgekehrt, je nach Vereinbarung). Dadurch ergibt sich eine Variationsanzahl von 2^n, wobei 2 für die beiden möglichen physikalischen Zustände steht und n für die Anzahl der Schritte (bits)[3]), die im entsprechenden Code ein Zeichen bilden.
Eine Datenmenge von 8 Bits nennt man Byte oder Oktett. Die Aneinanderreihung von z. B. 128 Oktetts (1024 bits) stellt eine Datenmenge dar, die als Einheit behandelt und, mit Steuerzeichen versehen, Datenpaket genannt wird.

— Datenrate

Je nach dem, wie schnell diese Bits über eine Leitung übertragen werden, ergibt sich eine auf eine Sekunde bezogene Anzahl von Bits: die Bit- oder Datenrate.
Beim gebräuchlichen Fernschreiber werden in den meisten Fällen 50 Bits/s gesendet. Die Datenrate kann auch in den niedrigeren Geschwindigkeiten mit Baud[4]) angegeben werden.
Moderne Datenübertragungseinrichtungen arbeiten mit 2400, 4800, 9600 Bits/s und mehr.

— Synchron-/Asynchronbetrieb

Das asynchrone Verfahren sei m. H. des Fernschreibers erklärt: Der Fernschreibapparat — man denke an einen alten, mechanischen — ordnet jedem Tastendruck eine bestimmte Stellung seiner Sendekontakte zu. Das Fernschreibalphabet (Int. Telegraphenalphabet Nr. 2) setzt ein Zeichen (Buchstabe, Ziffer etc.) aus 5 Schritten zusammen, die zu ihrem definierten Zeitpunkt Strom oder keinen Strom aufweisen. Diese wird durch die Kontakte des Apparates bestimmt. Beim Aussenden werden diese Kontakte abgetastet und das entsprechende Strombild an die Leitung angelegt.
Der Empfänger erhält diese Stromschritte und ordnet sie elektromechanisch über Magnete und Hebel dem betreffenden Zeichen (wieder Buchstabe, etc.) zu.
Damit aber dem Empfänger mitgeteilt werden kann, daß ein Zeichen an ihn herankommt und daß er seinen Antrieb (Motor) für die gesamte Mechanik einschalten muß, wird vom Sender ein Startschritt vor das eigentliche Zeichen gesetzt. Da man dem Empfänger auch noch den Schluß eines Zeichens mitteilen will, setzt man ans Ende des Zeichens — je nach System —, einen, eineinhalb oder zwei Stoppschritte.
Start- und Stoppschritt sind die Kennzeichen asynchroner Verfahren (Abb. 2).

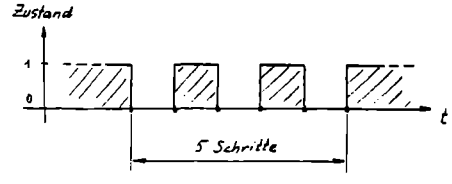


Abb. 1: Zeichen R im Int. Telegraphenalphabet Nr. 2

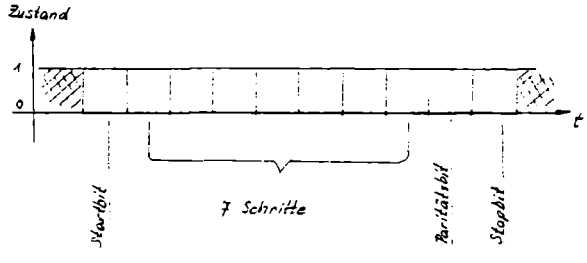


Abb. 2: 7-Schritt-Code mit Start-, Stop- und Paritätsbit

Bei der synchronen Datenübertragung legt ein für Sender und Empfänger gemeinsamer Takt, von z. B. einem Oszillator stammend, die Zeitpunkte fest, zu denen ein Schritt ablaufen muß, ungeachtet ob Strom oder kein Strom. Die Zeitpausen zwischen den einzelnen Zeichen werden durch Leerzeichen (Synchronzeichen) ausgefüllt, die von den Datenstationen selbsttätig ausgesandt werden. Sobald ein Zeichen übertragen wird, fällt es gezwungenermaßen in den vorgegebenen Taktraster. Die Datenübertragungsgeschwindigkeit wird durch die Taktfrequenz bestimmt. Geschwindigkeiten bis zu etwa 300 bit/s lassen sich noch durch asynchrone Verfahren erreichen; höhere Übertragungsraten verwenden ausschließlich synchrone. Mechanische Drucker sind in ihrer Druckgeschwindigkeit ebenfalls begrenzt und werden häufig über Pufferspeicher betrieben.

[1]) codicillus (lat.) = (geheimer) Brief
[2]) digital = ziffernmäßig von digitus (lat.) = Finger (mit dem man zählt)
[3]) das Bit, Kunstwort aus binary digit (engl.) = Binärziffer, Binärzeichen
[4]) Nach dem Konstrukteur eines Typendrucktelegraphen Baudot.

Grundsätzlich kann man eine Datenstation in Datenendeinrichtung (DEE) und Datenübertragungseinrichtung (DÜE) gliedern. Die Verbindung zwischen DEE und DÜE wird Schnittstelle genannt. Diese Schnittstelle besteht aus mehreren Leitungen, physikalischen Verbindungen (Drähten) also, auf denen der Datenfluß stattfindet, Steuerzeichen für die DÜE, für die DEE und verschiedene andere Informationen übertragen werden (Abb. 3).
Die DÜE bringt die von der DEE angelieferten Datensignale in eine für die Übertragung auf der angeschlossenen Übertragungsleitung geeignete Form oder formt die von der Übertragungsleitung empfangenen Signale in die Form um, wie sie für die DEE verständlich ist.
Den Eigenschaften der Übertragungswege entsprechend gibt es verschiedene Datenübertragungsverfahren. Die einen sind solche, die die Signale im Basisband — d. h. ohne Modulationsvorgang — durchführen, andere wieder benutzen einen modulierten

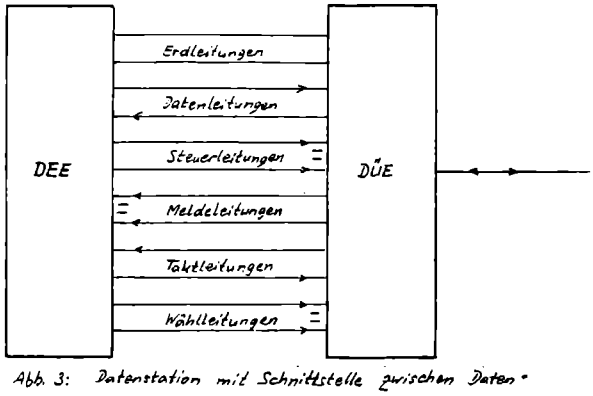


Abb. 3: Datenstation mit Schnittstelle zwischen Datenendeinrichtung und Datenübertragungseinrichtung.

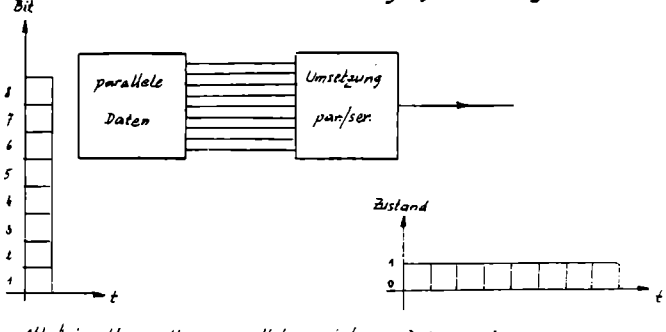


Abb. 4: Umwandlung parallel angebotener Daten auf seriellen Datenfluß

Träger, wie man dies vom Rundfunk her kennt. Entsprechend der den zu übertragenden Binärzeichen[5]) wird hiebei die Amplitude, die Frequenz oder die Phase des Trägers — der Trägerschwingung — verändert. Das Kunstwort MODEM (Modulation/DEModulation) weist auf diese Funktionen der DÜE hin.
Die Schnittstellenleitungen zwischen DEE und DÜE sowie deren Pegelwerte (Ströme, Spannungen) sind genormt, sodaß mit Hilfe einheitlicher Steckverbindungen auch andere als Datenübertragungsgeräte angeschlossen werden können.
Verbindlicher Normenausschuß ist in diesem Falle die CCITT (Comité Consultatif International Téléphonique et Télégraphique), die ihre Empfehlungen mit Buchstaben und Zahlen benennt.
Die Empfehlung V. 24 heißt: „Liste der Definitionen für Schnittstellenleitungen zwischen Dateneinrichtungen und Datenübertragungseinrichtungen". Sie enthält alle Benennungen der Leitungen und deren Steckerbeschaltung am 25-poligen Stecker.
Die entsprechende deutsche Norm heißt DIN 66020, Bl. 1. Nachdem nun DEE und DÜE zusammenarbeiten, übernimmt die DÜE, wie schon erwähnt, als Modem die wesentliche Aufgabe, die von der DEE abgegebenen binären Datensignale in bandbegrenzte Signale umzuwandeln und dann durch Modulation in den für die Datenübertragung benutzbaren Teil des Sprachfrequenzbandes umzusetzen (300—3400 Hz) und umgekehrt. Da die in verschiedenen nationalen Fernmeldenetzen eingesetzten Modems auch über internationale Leitungen und Verbindungen zusammenarbeiten sollen, müssen ihre Eigenschaften ebenfalls durch CCITT-Empfehlungen vereinbart werden.

— Serielle/Parallele Datenübertragung

Die Übertragung der Daten erfolgt bei den meisten Modems serielle, d. h. die Bits eines Zeichens werden zeitlich nacheinander übertragen. Daneben gibt es die Möglichkeit, alle Bits eines Zeichens parallel, d. h. gleichzeitig zu bearbeiten.
Um bei paralleler Übertragung eine der Anzahl der Codierungsschritte pro Zeichen entsprechende Leitungszahl zu vermeiden (bei 7 Bits/Zeichen wären dies 7 Leitungen), tastet man die Schritte gleichzeitig ab und moduliert, dem gegenwärtigen Zustand nach, unterschiedliche Frequenzen, die ebenfalls gleichzeitig, aber über eine einzige Leitung übertragen werden.
Bei der seriellen Übertragung werden die Zeichen mittels eines Parallel-Seriell-Umsetzers in die serielle Form gebracht (Abb. 4).

— Fehlersicherung

Zusätzlich zu den Bits der Zeichencodierung kann noch ein weiteres Bit zum Zwecke der Fehlererkennung hinzugefügt werden. Beim heute viel benutzten ASCII-Code[6]) wird den ein Zeichen bildenden 7 Bits ein achtes angeschlossen (s. Abb. 2).
Dieses Bit wird Paritätsbit genannt; es kann einen der beiden logischen Zustände 0 oder 1 einnehmen, je nach Parität. Dies ist das einfachste Verfahren.
Man faßt also 7 Datenbits zu einem Datenwort zusammen und vereinbart gerade (even) oder ungerade (odd) Parität.
Bei der geraden Parität setzt man das hinzugefügte Paritätsbit auf Null, wenn die Zahl der Einsen im Datenwort gerade ist. Man setzt auf Eins, wenn sie ungerade ist. Dadurch ist die Gesamtzahl der übertragenen Einsen in einem Datenwort einschließlich Datenwort immer gerade. Bei der ungeraden Parität ist dies umgekehrt.
Auf der Empfängerseite berechnet man das Paritätsbit auf dieselbe Weise aus dem Datenwort und vergleicht es mit dem übertragenen Paritätsbit. Ergibt sich ein Unterschied, liegt ein Übertragungsfehler vor. Auf diese Weise läßt sich jeder Einzelfehler erkennen. Eine Korrektur ist jedoch nicht möglich, da das fehlerhafte Bit nicht lokalisierbar ist (hiefür gibt es besondere Methoden, z. B. Blocksicherung). Sind mehrere Bits gestört, kann man eine ungerade Fehlerzahl erkennen, eine gerade hingegen nicht.

[5]) binär (lat.) = aus zwei Einheiten bestehend
[6]) American-Standard Code for Information Interchange

Kleinanzeigen

Unter dieser Rubrik veröffentlicht TI-99 JOURNAL Textanzeigen zu einem besonders günstigen Preis.

Nutzen Sie den Anzeigen Service, wenn Sie

- Hardware oder Software verkaufen oder kaufen wollen.
- Programme tauschen möchten
- Literatur suchen oder anbieten.

Je Zeile stehen 32 Zeichen (einschließlich Satzzeichen und Zwischenräumen) zur Verfügung.

Private Anzeigen:
öS 30,-/Zeile, inkl. Abgaben
Gewerbliche Anzeigen:
öS 46,-/Zeile, exkl. Abgaben
Chiffregebühr:
öS 30,- exkl. MWSt.

Endlich GRAFIK-BEFEHLE
Auf Cass/Disk.in EXT-BASIC
Punkt Gerade Kreis Bogen Sektor
Ellipse Vieleck Rechteck ZAHL
RAUM 1 BEFEHL! 256X184 PUNKTE
PREIS: öS 600,- Rainer BERNERT
1190 WIEN,KLABUNDG.5-7 36555+2

DISASSEMBLER DISASSEMBLER
von Rainer BERNERT
Übersetzt Maschinenkode
zurück in Assembler!
Ermöglicht Ausdrucke von
Assemblerprogrammen!
Unersätzlich beim Ändern, Erweitern,Ausdrucken in Assembl
PREIS: öS 520,- (Klub: - 10 %!)
BEZUG: TI-99 JOURNAL

H P 4 1 C
Verkaufe originalverpackt neuen HP 41C (Geschenk). GARANTIE: 6 Monate. Inkl. Handbücher, Accu Tasche, Programme, usw.
STATT öS 4.700,- NUR 4.000,-
Anfragen an: TI-99 JOURNAL

SUPER NEBENVERDIENST

TEXAS INSTRUMENTS sucht laufend junge TI-99 Benutzer für Gerätedemonstrationen (Messen, Handel, Schulen). Interessenten melden sich direkt bei:
TI 2345 Brunn,Industriestr.B/16
Hr.WANDL, Tel.02236/84621-40

VERKAUFE: TI-99A THERMODRUCKER
1 Disketten-Antrieb+Steuergerät
5 MODULE:Extended-BASIC deutsch+
Disk-Manager+Personalrecordkeep+
Schach+Videospiele II,Tel.677354

TI99-JOURNAL TIPS

EXTENDED BASIC: HANDBUCH IN DEUTSCH

Zum Extended BASIC-MODUL gibt es jetzt auch in Österreich eine deutsche Version des Handbuches.

Durch die besonders verständliche Form (ähnlicher Aufbau wie die Originalausgabe) ist dieses Handbuch ein unentbehrliches Werkzeug für die Arbeit mit Extended BASIC.
Bezug:
TI-99 JOURNAL-KLUB öS 390,-
(Klubmitglieder: öS 351,-)

KOPIERSCHUTZ BEI DISKETTEN

Mit Extendend BASIC besteht die Möglichkeit Programme auf Diskette oder Kassette "PROTECTED" abzuspeichern. Das heißt, daß das Programm nur verarbeitet, aber nicht gelistet oder mit SAVE abgespeichert werden kann. Mit dem Disk-Manager-MODUL ist es jedoch möglich das Programm zu kopieren

Es besteht jedoch die Möglichkeit die Diskette beim Initialisieren so so präparieren, daß ein Kopieren mit dem Disk-Manager auch nicht mehr möglich ist.

Dieser Kniff wird jedoch von TEXAS nicht "verraten", und so besteht nur im Einzelfall die Möglichkeit sich diesen Schutz direkt bei TEXAS auf der Diskette anbringen zu lassen.

TEXTFEHLER IM ROM

1) DATENVERWALTUNG-MODUL

Im Modul "Datenverwaltung und -analyse" wird beim Speichern bzw. Einlesen einer bestehenden Datei auf Diskette nach dem Dateinamen gefragt.

Korrekt muß jedoch die Gerätebezeichnung UND der Dateiname angegeben werden.
z.B.: DSK1.KUNDEN

2)DISKMANAGER-MODUL

Im Diskmanager-MODUL gibt es in der deutschen Version Textfehler im ROM. So erscheint z.B. beim Einlegen einer nicht initialisierten oder fehlerhaften Diskette " SKETTE NICHT INITIALISIERT" bzw. " EHLERHAFTE DISKETTE".

Weiters erscheint immer am unteren Bildschirmrand statt PROC'D, REDO, BACK oder BEGINN folgende falsche Auswahlliste: PROC'D, REDO, REDO oder BEGINN.

Diese Fehler beinflussen aber nicht den korrekten Ablauf der Programme.

BILDABSCHALTUNG - WARUM?

Sicherlich ist Ihnen schon aufgefallen, daß sich der Bildschirm automatisch nach einiger Zeit, wenn keine Taste gedruckt wird "abschaltet".

Dieses Löschen des Bildschirmes dient dazu, ein "Ausbrennen" durch "Projezieren" eines immer gleichgestalteten Bildes zu vermeiden.

QUIT - NICHT SEHR BENUTZERFREUNDLICH

Besondere Aufmerksamkeit sollte man der QUIT-Taste (FCTN + "=") widmen.

Beim Erarbeiten eines Programmes, in dem natürlich auch das "+"-Zeichen Verwendung findet, kann es sehr leicht durch ein wenig Unaufmerksamkeit zu einem folgenschweren Irrtum kommen. Man möchte ein "+" durch SHIFT "=" eingeben und "rutscht" aber dabei von SHIFT auf FCTN. In diesem Augenblick erscheint das Titelbild und alles bisher Eingegebene ist verloren.

Abhilfe kann bei diesem Problem nicht geschaffen werden. Man sollte sich daher von Anfang an angewöhnen, bereits Teile des Programmes auf Diskette oder Kassette zu sichern und sich so die Möglichkeit schaffen, bei irrtümlichen Drücken von QUIT oder einem STROMAUSFALL nicht alles zu verlieren, sondern nur die letzten Programmschritte neu tippen zu müssen.

TI-99 JOURNAL

IM JUNI

Telefonmodem
Testbericht: CC-40
Sprachsynthesizer
Modulverlosung
Hardware: GRAFIK (VDP, Bildschirm)
BASIC Grundkurs: 2. Teil: Rechner- und Programmstruktur
Tips für Umsteiger: 2. Teil
Hochauflösende Grafik - superschnell im TMS 9900 Assembler.
Programmieren in Assembler
Raumschiffkampf - (Programmlisting)
TEXTVERARBEITUNG mit dem TI-99
8K-BYTE Programmlisting

DIE SCHNITTSTELLE V. 24

— Allgemeines

Zum Zwecke der Vereinheitlichung besteht wie zwischen DÜE und DEE auch beim Zusammenschluß von Zusatzgeräten mit der DEE oder der DÜE eine einheitliche Schnittstelle.
Diese Schnittstelle setzt sich aus mehreren physikalischen Verbindungen zusammen (Schnittstellenleitungen), über denen neben dem Informationsfluß (Daten) auch noch Steuerzeichen, z. B. Zeichen zur Steuerung eines Druckers, übertragen werden.
Die CCITT-Empfehlung V.24 bezieht sich auf diese Leitungen zur Übertragung binärer Daten, Steuer- und Schrittaktsignalen und erstreckt sich auch auf Zwischen- und Zusatzeinrichtungen, die mit DEE oder DÜE verbunden sind.
Bei jedem für den jeweiligen Einsatz bestimmten Gerät wird aus der Skala der in der V.24 definierten Schnittstellenleitungen eine entsprechende Wahl getroffen (diese Auswahl ist in weiteren Empfehlungen, z. B. V.10, V.11 etc. wiederum festgehalten).
Diese können u. a. angewendet werden auf:

a) synchrone oder asynchrone Datenübertragung
b) Datenübertragung auf festgeschalteten Leitungen
c) Datenübertragung über das Wählnetz (Telephon- oder Datenwählnetz)
d) Verhältnisse, bei denen nur kurze Verbindungskabel zwischen DEE und DÜE verwendet werden (geschirmt max. 10—15 Meter).

— Die wichtigsten Schnittstellenleitungen

Gemäß der CCITT-Empfehlung V.24 sind die Schnittstellenleitungen wie folgt definiert:

Leitung 101 — Schutzerde (Stift 1)
Leitung 102 — Erdleitung (Stift 7)
Diese Leitung bildet den gemeinsamen Rückleiter für die unsymmetrischen Schnittstellenleitungen mit den elektrischen Eigenschaften entsprechend der Empfehlung V.28 und liefert das Gleichstrom-Bezugspotential für symmetrische Schnittstellenleitungen entsprechend den Empfehlungen V.11 und V.35.

Leitung 103 — Sendedaten (Stift 2)

Richtung: Zur Datenübertragungseinrichtung.
Die von der DEE erzeugten und über den Datenkanal an eine oder mehrere ferne Datenstationen zu übertragenden Datensignale werden über diese Leitung an die DÜE geleitet.

Leitung 104 — Empfangsdaten (Stift 3)

Richtung: Von der DÜE
Die Datensignale, die als Folge der von einer fernen Datenstation empfangenen Datenkanal-Leitungssignale von der DÜE erzeugt werden, gelangen über diese Leitung zur DEE.

Leitung 105 — Sendeteil einschalten (Stift 4)

Richtung: Zur DÜE.
Die auf dieser Leitung übertragenen Signale steuern die Sendefunktionen der DÜE.
Der EIN-Zustand bewirkt, daß die DÜE im Datenkanal den Betriebszustand „Senden" annimmt.
Der AUS-Zustand bewirkt, daß die DÜE im Datenkanal den Betriebszustand „nicht Senden" annimmt, wenn alle auf die Leitung 103 (Sendedaten) gegebenen Daten übertragen worden sind.

Leitung 106 — Sendebereitschaft (Stift 5)

Richtung: Von der DÜE
Die Signale auf dieser Leitung zeigen an, ob die DÜE darauf vorbereitet ist, Daten in den Datenkanal zu senden. Der EIN-Zustand zeigt an, daß dies der Fall ist; der AUS-Zustand, daß die DÜE nicht darauf vorbereitet ist.

Leitung 107 — Betriebsbereitschaft (Stift 6)

Richtung: Von der DÜE
Die Signale auf dieser Leitung zeigen an, ob die DÜE betriebsbereit ist, oder nicht.
Der EIN-Zustand zeigt an, daß der Signalwandler oder eine ähnliche Einrichtung an die Leitung angeschaltet ist und daß die DÜE bereit ist, weitere Steuerzeichen mit der DEE auszutauschen, um den Austausch von Daten einzuleiten.
Der AUS-Zustand zeigt an, daß die DÜE nicht betriebsbereit ist.

Leitung 108 — DEE betriebsbereit (Stift 20)

Diese Signale steuern das An- oder Abschalten des Signalwandlers oder ähnlicher Einrichtungen an die bzw. von der Übertragungsleitung.
Der EIN-Zustand, welcher anzeigt, daß sich die DEE in betriebsbereitem Zustand befindet, bereitet die DÜE vor, damit diese die genannte Einrichtung anschaltet und erhält die Verbindung aufrecht, nachdem sie durch äußere Mittel hergestellt worden ist.
Die DEE darf immer dann den EIN-Zustand auf Leitung 108 anbieten, wenn sie bereit ist, Daten zu senden oder zu empfangen.
Der AUS-Zustand bewirkt, daß die DÜE den Signalwandler oder eine ähnliche Einrichtung von der Übertragungsleitung abtrennt, wenn die Übertragung sämtlicher zuvor auf die Leitung 103 und/oder auf die Leitung 118 (Hilfskanal-Sendedaten) gegebener Daten abgeschlossen ist.

Leitung 109 — Empfangssignalpegel (Stift 8)

Richtung: Von der DÜE.
Die Signale auf dieser Leitung zeigen an, ob das empfangene Datenkanal-Leitungssignal innerhalb der jeweiligen Grenzen liegt (EIN-Zustand), die in der entsprechenden Empfehlung für DÜE angegeben sind oder nicht (AUS-Zustand).

— Datenübertragung an entfernte DEE's

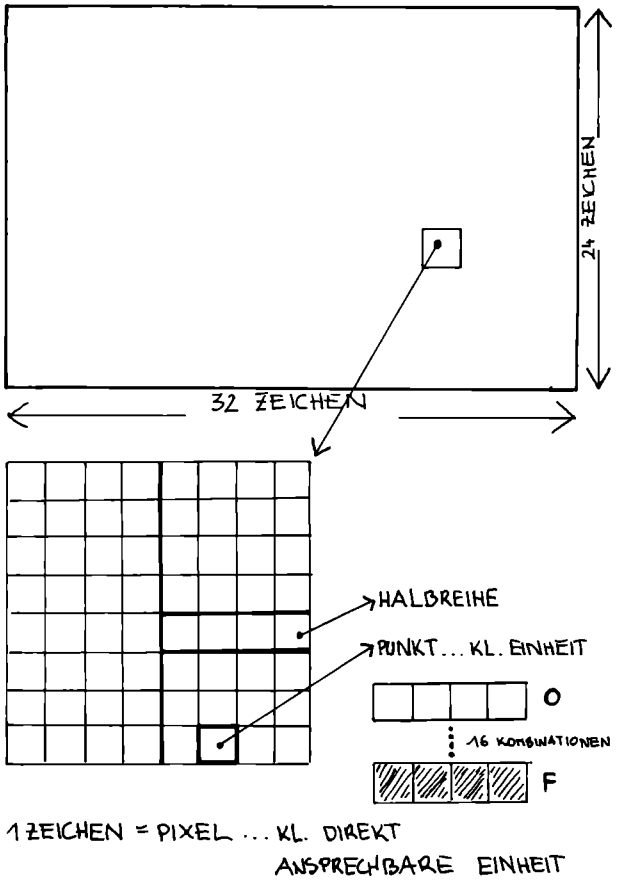
Für die Datenübertragung über das Leitungsnetz der Post gibt es mehrere Möglichkeiten. Fast allen gemeinsam ist jedoch die Anschalteerlaubnis als Folge des Zulassungsverfahrens für das betreffende Gerät (Modem und/oder DEE + Zusatzeinrichtungen).

Friedrich F. Brezina ● Wird fortgesetzt.

GRAFIK

Grafik in TI-BASIC und TI-EXT.BASIC

Der Ti besitzt leider kaum Grafikbefehle. Man muß also jeden Punkt einer Geraden, eines Kreises, . . . berechnen. Der Bildschirm ist wie folgt organisiert:



Es gibt nun die Möglichkeit der Pixelgrafik. Kleinste Einheit ist ein Pixel. Die Auflösung dieser Grafik ist sehr schlecht. Ein kleines Programm zeigt die begrenzten Möglichkeiten.

```
10 X = X + 1
20 PRINT TAB(INT(10*SIN(X/2)+14.5));,,*"
30 GOTO 10
```

Mit den Befehlen HCHAR, VCHAR können Geraden sehr schnell gezeichnet werden.

Eine weit bessere Auflösung bietet ein kleines Programm, das jedes Pixel in die einzelnen 64 Punkte zerlegt und so in einer Matrix von 256 x 192 Punkte jeden Punkt zugänglich macht.

Der Weg eine Grafik zu erzeugen ist nun folgender:

1. Errechnen des zu setzenden Punktes
2. Errechnen der Pixelposition
3. Errechnen der Position im Pixel
4. aus 3 den HEXADEZIMAL-Code bestimmen
5. mit CALL CHAR das neue Zeichen definieren
6. mit CALL HCHAR (VCHAR) setzt man das neue Zeichen auf die Pixelposition.

Zuerst möchte ich auf einige grundsätzliche Vorgänge eingehen: Jedes Zeichen besteht aus 64 Punkten, die gesetzt oder gelöscht werden können. Das Zeichen oder Pixel wird in 16 Halbreihen unterteilt. Jede Halbreihe besteht aus 4 Stellen (Bits) und kann in 2↑4 =16 Kombinationen auftreten. Um 16 Kombinationen benennen zu können wird das Hexadezimalsystem eingeführt: O — F.

8 HEX-Ziffern beschreiben ein Zeichen und bilden einen HEX-Code. Der HEX-Code eines Pixels wird in einem Speicherbereich abgelegt, der dem Bildschirm zur Verfügung steht. Dieser Prozeß wird von der CPU (Central Processor Unit) dem VDP (Video Display Processor) übergeben, der die gesamte Bildschirmverwaltung übernimmt.

Der TI hat den Standard-ASCII Zeichensatz (Code 32-127) und darüber hinaus

128—159 Zeichen in TI-BASIC
128—143 Zeichen in TI-EXTENDED BASIC

Die Form der Zeichen des ASCII-Codes werden beim Einschalten des Gerätes von einem ROM (Read Only Memory) ins VDP-RAM (Random Access Memory) übertragen.

Die Kleinbuchstaben werden durch Weglassen von Zeilen generiert. Im RAM können die Zeichen neu definiert werden.

Aufgabe des Programmierers ist es, ein Programm zu entwerfen, das die benötigten Punkte einschaltet.

Hier ist ein Programmteil der als Eingabe

Y, X Koordination am BS	X = 1—256
	Y = 1—192

verlangt und die Ausgabe

YG, XG, Pixelkoordinaten	XG = 1—32
	YG = 1—24
YK, XK Koordinaten im Pixel	XK = 1—8
	YK = 1—8

erzeugt.

```
.
YG =INT (Y/8+.9)
XG =INT (X/8+.9)
YK =8—YG 8+Y
XK =8—XG 8+X
.
.
.
.
```

Es muß dann nur mehr der HEX-Code ausgerechnet werden (z. B. mit Binärzahlen 2↑Stelle in der Halbreihe). Dann mit CHAR das Zeichen definiert und zum Schluß mit HCHAR (VCHAR) an die Stelle der Pixelkoordinaten gebracht werden.

Nun zu den Farben:
Die verfügbaren Zeichen sind in 8er Blöcke aufgeteilt, wobei jedem Block eine Vordergrundfarbe (eingeschalteter Punkt) und eine Hintergrundfarbe (ausgeschalteter Punkt) zugewiesen werden kann.
Es gibt 16 Farben von O—F im HEX-Code.
Mit CALL SCREEN kann die Bildschirmfarbe verändert werden. Die Standard Bildschirmfarbe ist Hellgrün (Code 4).
Mit CALL COLOR (Zeichenblock, Vordergrundfarbe, Hintergrundfarbe) kann die Farbe verändert werden.
Mehr ist über die Farben und deren Programmierung nicht zu sagen, da in den Manuals des TI dieser Punkt genau beschrieben wird.

IN EXTENDED BASIC gibt es SPRITES.
SPRITES sind Grafiken, die am Bildschirm bewegt werden können. Sie sind entweder 8 x 8 oder 16 x 16 Punkte groß.

Vorteile:

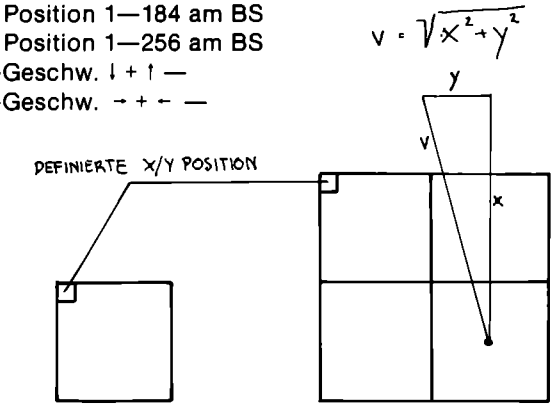
- bewegte Grafiken „rucken" nicht, da die Auflösung bei der Bewegung größer ist (Abstand von Position 1 zu Position 2 < als 1 Punkt.
- schnelles Handhaben der Grafiken durch eigene Befehle
- Spiele können nun programmiert werden.

Nachteile:

- „Nachladen" der Grafiken ist langsam (CALL CHAR)
- geringe Maximalgröße; mehrere Sprites sind schlecht zusammenziehbar.
- Nur 4 SPRITES in einer Zeile.

BEFEHL: CALL SPRITES (= a, b, c, d, e, f, g)

a) Nummer 1—28; höhere Nummer überführt niedrigere
b) Zeichennummer bei 16 x 16 32 + I*4 I . . 0—27
8 x 8 32—143
c) Vordergrundfarbe 2—16/Hintergrund = transparent (1)
d) Y Position 1—184 am BS
e) X Position 1—256 am BS
f) y-Geschw. ↓ + ↑ —
g) x-Geschw. → + ← —



VERGLEICH im GRAFIK-TEST zwischen TI-BASIC und TI-EXTENDED BASIC

	TI-BASIC	TI-EXT. BASIC
Anzahl der frei zu definierenden Zeichen	128	112
Verarbeitungsgeschw. s/100 Durchläufe		
CHAR	11	8.6
HCHAR	4.9	2.0
VCHAR	4.9	2.0
Gchar	4.8	2.1
COLOR	4.8	1.8
SCREEN	3.5	1.0
CHARPAT	—	ergibt den HEX-Code eines Zeichens
CHARSET	—	setzt alle Zeichen wieder zurück auf ASCII-Zeichen
SPRITES:		Im folgenden wir mit Grafik „Sprite" bezeichnet
SPRITE	—	setzt eine Grafik auf beliebige Position und bewegt sie mit beliebiger Geschwindigkeit
MAGNIFY	—	bestimmt die Größe der Grafik
COLOR	—	bestimmt die Farbe der Grafik
LOCATE	—	setzt eine best. Sprite auf eine andere Position
PATTERN	—	ändert die CHARACTER Nummer
MOTION	—	bestimmt die Bewegung der Grafik
COINC	—	stellt fest, ob in definierter Entfernung eine andere Grafik oder eine best. BS-Position ist
DISTANCE	—	berechnet die Entfernung von 2 Grafiken oder 1 Grafik und 1 Pos.
POSITION	—	gibt die Position der Grafik an.

Abschließend noch ein Tip: Verwenden Sie bei Grafiken wenn möglich keine Kreisfunktionen. Diese sind langsam und ungenau. Für die meisten Anwendungen gibt es andere Formen. Auch Näherungsformeln lassen sich gut verwenden.

TI99-JOURNAL-TEST: Das MINIMEMORY-Modul

DAS ARBEITEN MIT DEM MINI MEMORY

Ein TESTBERICHT von BERNERT Rainer.

Vor einiger Zeit las ich in einer Softwarebroschüre über das Mini MEMORY. Obwohl mir nicht ganz klar war was dieses Modul leisten kann bestellte ich es sofort, denn in letzter Zeit war ich an die Grenzen von TI-BASIC gestoßen. Die Idee von eigenen Unterprogrammen in Assembler eingebunden in TI-BASIC faszinierte mich. Eine Woche später hatte ich es.
Zu Hause probierte ich es sofort aus. Als erstes sah ich mir das Demo Programm „LINES" an. Die Geschwindigkeit mit der da Linien gezogen wurden war überwältigend. Fassungslos sah ich minutenlang zu. Nach einiger Zeit war das aber zu langweilig und ich nahm mir das gelbe Heft (MINIMEMORY) und fing an zu lesen. Ich kam zum „SAMPLE PROGRAMM". Dieses Programm ist als Programmierbeispiel in ASSEMBLER aufgelistet und kann STRINGS an beliebiger Stelle am Bildschirm anzeigen.
Hier erfuhr ich zum ersten Mal den Unterschied zwischen höheren Programmiersprachen (BASIC, EXTENDED BASIC) und dem maschinensprache orientierten ASSEMBLER.
Wofür EXTENDED BASIC einen Befehl benötigt braucht der ASSEMBLER 80. Vor dem Eintippen mußte ich den Line by Line Assembler mit Kassettenrekorder ins Minimem laden. Dann tippte ich das Beispiel ein. Kleine Fehler wie A1 statt AI besserte ich sofort aus. (Solche Fehler passieren wenn Programmlistings in Heften abgetippt und nicht photographiert werden.)

Endlich war ich fertig. Ich beendete den ASSEMBLER und „ging" ins BASIC. Es passierte nach dem Aufruf lt. Buch nichts.
Ich schaltete ab und testete Zeile für Zeile mit EASY BUG durch. Nach einigen Ausbesserungen lief das Programm.
Nach diesem Teilerfolg ging ich mit System vor. Ich las beide Heftchen, die beim Modul mitgeliefert werden, durch.
Schon bald merkte ich, daß ich ohne EDITOR/ASSEMBLER — (der große Bruder vom LINE BY LINE ASSEMBLER) Handbuch nichts in ASSEMBLER programmieren kann. Es fehlt im Lieferumfang die Beschreibung der ASSEMBLER-Befehle. (Das EDITOR/ASSEMBLER Handbuch kann um 590,— bezogen werden. Ein Handbuch um ca. 300,— wird zur Zeit vom TI99-JOURNAL-KLUB ausgearbeitet.
In der Zeit in der ich das Handbuch noch nicht hatte, probierte ich die anderen Funktionen des MINI MEMORY's aus.
Ich peekte und pokte wild herum und zerstörte öfters den Line by line ASSEMBLER.
Mittels Kassettenrekorder lud ich ihn wieder ein.
Bald aber hatte ich das Handbuch (500 Seiten) das mir sehr genaue Informationen über Aufbau des Computers und den ASSEMBLER Befehlen gab. Mit diesem ging ich an die Programmierarbeit. Mein 1. Programm konnte ich auf Tastendruck den jeweiligen Buchstaben am Bildschirm zeigen.
Ich verbesserte es und hatte bald ein Text-Programm. Bald kannte ich alle Befehle und konnte auch damit umgehen.
Routienen in ASSEMBLER schreiben, die die Graphik, den Ton, usw. direkt ansprechen wurde möglich.

Was kann das MINI MEMORY?

1. Benützung als „Pseudo Disc".
   Diese Verwendungsart spielt sicher eine untergeordnete Rolle da der verfügbare Speicher (4 K) für Programme in TI-BASIC oder Daten sehr klein ist. Daten können direkt oder sequentiell abgelegt werden.

2. Beinhaltet einige TI-BASIC Unterprogramme.

CALL INIT	löscht alle Daten und Programme des MINI MEMORY's.
CALL LOAD	poket (verändert) ins CPU RAM.
CALL LINK	Verbindet BASIC mit ASSEMBLERUNTERPROGRAMMEN.
CALL PEEK	zeigt den Inhalt des CPU RAM's.
CALL PEEKV	zeigt den Inhalt des VDP RAM's.
CALL POKEV	poket den Inhalt des VDP RAM's.
CALL CHARPAT	stellt den HEX-Code eines Zeichens zur Verfügung.

3. **EASY BUG**

   M . . . Anschauen und Verändern des CPU RAM's.
   V . . . Anschauen und Verändern des VDP RAM's.
   G . . . Anschauen des GROM's. (Grafik Read Only Memory)
   E . . . Ausführen von ASSEMBLER Programmen.
   C . . . Ändern von CRU (Comonication Register Unit)-Bits
   S . . . Abspeichern von CPU RAM auf Kassette.
   L . . . Laden von Kassetten ins CPU RAM.

4. **MINI MEMORY**

1 . . . Load and Run	lädt ein Programm von Diskette und führt es aus.
2 . . . Run	führt ein Programm (ASSEMBLER) aus.
3 . . . Re-Initialize	löscht den CPU RAM Speicher.

Mit 2 und „NEW" kann man ASSEMBLER Programmieren.

Der **LINE BY LINE ASSEMBLER:**

Man gibt über die Tastatur in ASSEMBLER-Syntax die Befehle ein. Mit ENTER wird der ASSEMBLER Befehl **sofort** in Maschinencode umgewandelt. Bei falscher Syntax erscheint „ERROR".
Bei falschen Werten (Sprungbefehle) erscheint „RE-ERROR" (Reach-Error). Wegen dem geringen Platzangebot konnte kein EDITOR verwirklicht werden. So wird das Ausbessern zur Prozedur.
Hat man einen falschen Buchstaben eingetippt so muß man die ganze Zeile neu eingeben.
Die eingegebenen Zeilen werden in einem 9-Seiten Buffer abgelegt. Mit den Coursortasten (↑,↓)kann man diesen Buffer verschieben. Beendet man den ASSEMBLER und testet das Programm so wird beim neuerlichen Einstieg (zur Korrektur) der Buffer gelöscht.
Es ist nun sehr schwer den Fehler zu finden da man den Maschinencode nur langsam lesen kann. Abhilfe schafft hier ein Programm das das Zurückübersetzen von Maschinensprache in ASSEMBLER schafft =DISASSEMBLER (im Handel erhältlich). Auch können mit diesem **DISASSEMBLER** ASSEMBLERLISTEN ausgedruckt werden.
Auch vermißt man in der Testphase eine RESET Taste sehr, da Programme in ASSEMBLER leicht „abstürzen".
EASY BUG (E) hat sich in der Testphase sehr bewährt da das Starten eines Programmes mit Hilfe von EASY BUG schneller geht. Ein großer Vorteil ist die Batterie im MINI MEMORY. Man erspart sich das Laden von Programmen da diese erhalten bleiben.

Allgemeine Daten:

Das Modul MINI MEMORY beinhaltet:
4 K BYTES RAM
4 K BYTES ROM
6 K BYTES GROM

RAM . . . RANDOM ACCESS MEMORY = freier Speicher (Assembler)
ROM . . . READ ONLY MEMORY = Routinen
GROM . . . GRAFIK READ ONLY MEMORY = Grafik Routinen

Zusammenfassung:

Die Arbeit in ASSEMBLER ist etwas langwierig doch lohnt ein gelungenes Programm durch erstaunliche Geschwindigkeit die Mühe. Auch fasziniert es täglich neue Möglichkeiten der Programmierung in ASSEMBLER zu erkennen. Man stößt an keine softwaremäßigen Grenzen. Man setzt sie sich selber. Es ist keine Vorbildung in ASSEMBLER erforderlich.
Auch gewöhnt man sich rasch an die HEXADEZIMAL-Schreibweise. Der fehlende EDITOR macht ein Ausbessern oder Ändern zur mühevollen Kleinarbeit, ein DISASSEMBLER hilft bei der Suche von Fehlern.

PLUS:

- niedriger Preis
- Batterie
- Grafik, Ton, Ein/Ausgabeport direkt steuerbar
- vorprogrammierte Routinen im ROM und GROM

MINUS:

- ungenügende Beschreibung (EDITOR/ASSEMBLER Handbuch muß gekauft werden)
- kein DISASSEMBLER

BESTELLSCHEIN

Ich bestelle ein Jahres-ABO TI-99 Journal (6 Ausgaben) zum

Preis von öS 420,-- (incl. Porto und Verpackung)
DM 66,--
sfr 63,--

o ich habe einen Scheck beigefügt

o ich habe den Betrag am _____ auf das PSK-Konto WIEN 6.333.544 überwiesen.

Absender: Name _____

Anschrift _____

Bestellschein in Kuvert stecken und senden an:

TI-99 Journal, Thomas M. Fiedler
1060 Wien, Mollardgasse 32/3

SPEICHERPLATZ-ERWEITERUNG für Assembler-programme in MINIMEM

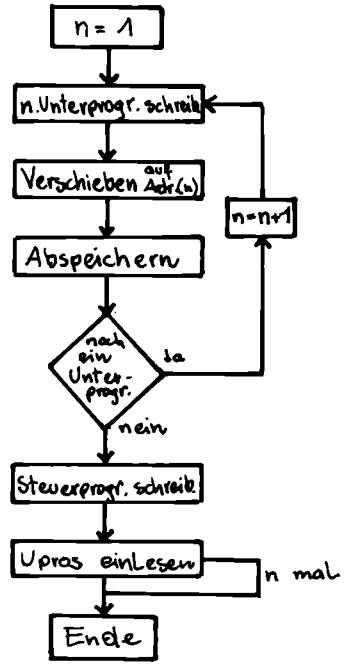
LINE BY LINE ASSEMBLER

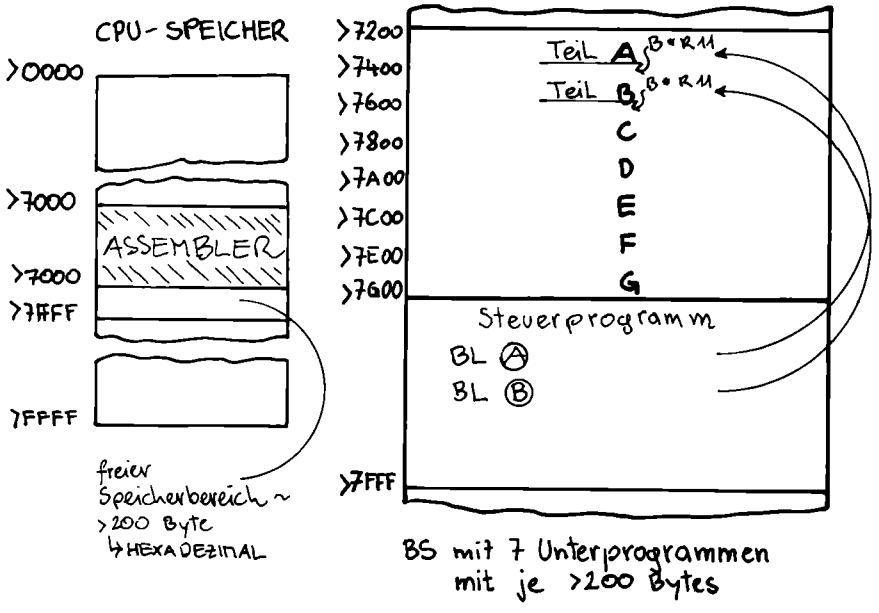
Da der Line by line Assembler 3300 Byte RAM benötigt bleiben für Programme (ohne Speicher - erweiterung) noch 700 Byte übrig

Dies ist für den Anfänger sicher mehr als genug, doch benötigen anspruchsvollere Programme mehr Platz.

K O P I E R P R O G R A M M

Es ist zweckmäßig das Kopierprogramm zu Beginn an das Ende des freien Speicherbereiches zu schreiben.





Mit einem winzigen Programm kann man den gesamten RAM Bereich des MINI MEMORY's benutzen. Die einzige Auflage ist, daß man modular (Unterprogrammtechnik) programmiert.

Man programmiert im freien Speicherbereich einen abgeschlossenen Programmteil. Mit einem Kopierprogramm kopiert man es an den Anfang des RAM's (7200). Jetzt lädt man es mit EASY BUG auf Kassette (S7200 TO 71FF). Nun lädt man den ASSEMBLER neu, da er durch das Kopieren zerstört worden ist.

Man schreibt nun das nächste Unterprogramm, verschiebt es nach 7400 und lädt es auf den Kassettenrekorder. Am Schluß schreibt man ein Steuerprogramm, das die einzelnen Unterprogramme aufruft.

```
7FE8  200  LI R0, Anfangsadresse alt
7FEA  Anfangsadr. alt
7FEC  201  LI R1, Endadresse alt
7FEE  Endadr. alt
7FF0  202  LI R2, Anfangsadresse neu
7FF2  Anfangsadr. neu
7FF4  C490 MOV +R0, +R2
7FF6  0580 INCT R0
7FF8  0582 INCT R2
7FFA  8040 C R0, R1
7FFC  12FB JLE  7FF4
7FFE  045B B +R11
```

R0.. Anfangsadr. des zu kopierenden Feldes
R1.. Endadr. des zu kopierenden Feldes
R2.. Anfangsadr. des neuen Feldes (Zielort)

TI99 als HEIMORGEL

Durch ein Belegen des jeweiligen Tastenkodes mit einer Frequenz ist es recht einfach möglich, die Tastatur als "Heimorgel" zu verwenden.

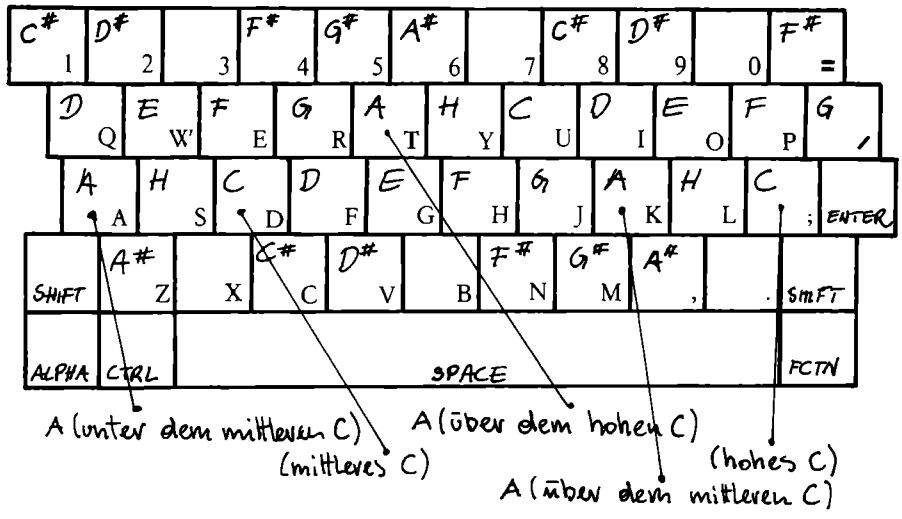
Die Belegung der Tasten in Form von Datenfelder (T) wurde der Überschaubarkeit für Anfänger wegen gewählt. Denn anhand dieser Datenfelder, T(ASCII-Kode)= Frequenz, ist es ganz einfach die Tasten in gewünschter Form umzubelegen.
Dem geübten Programmierer ist es unschwer möglich das Programm durch Verändern in DATA-Anweisungen enorm zu verkürzen.

Das eigentliche Musikprogramm mit Tastenabfrage befindet sich in den Zeilen 100 -210.

Also viel Spaß beim Musizieren!!

→

```
1 DIM T(100)
2 T(65)=220
3 T(83)=247
4 T(68)=262
5 T(70)=294
6 T(71)=330
7 T(72)=349
8 T(74)=392
9 T(75)=440
10 T(76)=494
11 T(59)=523
12 T(90)=233
13 T(67)=277
14 T(86)=311
15 T(78)=370
16 T(77)=415
17 T(44)=466
18 T(81)=587
19 T(87)=659
20 T(69)=698
21 T(82)=784
22 T(84)=880
23 T(89)=988
24 T(85)=1047
25 T(73)=1175
26 T(79)=1319
27 T(80)=1397
28 T(47)=1568
29 T(49)=554
30 T(50)=622
31 T(52)=740
32 T(53)=831
33 T(54)=932
34 T(55)=1109
35 T(57)=1245
36 T(61)=1480
37 T(88)=2000
38 T(66)=2000
39 T(46)=2000
40 T(51)=2000
41 T(55)=2000
42 T(48)=2000
90 CALL CLEAR →
95 PRINT TAB(10);"* PLAY *": : : : : : :"BEENDEN: ENTER DRUECKEN
"
100 CALL KEY(0,K,S)
110 ON S+2 GOTO 140,100,130
130 L=0
140 L=L+1
145 IF T(K)=2000 THEN 100
150 CALL SOUND(-70,T(K),L,T(K)*8,L+3,T(K)/2,L+3)
160 IF L=20 THEN 200
170 GOTO 100
200 L=L-1
210 GOTO 100
```



ALLGEMEINE MASCHINEN VERLEIH GES.MBH
Wien 6, Mariahilfer Straße 77 - 79
Telefon 96 19 51

AMV **BÜROMASCHINEN SUPERMARKT**

im 1. Stock
Rolltreppe
links

IM GENERALI - EINKAUFSCENTER

RIESENAUSWAHL - PROMTE LIEFERUNG . DISKONTPREISE

SAMSTAG VON 9 - 12 UHR GEÖFFNET, WOCHENTAGS DURCHGEHEND VON 9 - 18 UHR

GRÖSSTES LAGER AN ALLEM WAS VON TEXAS INSTRUMENTS KOMMT

EINSCHLIESSLICH HOME - COMPUTER TI-99 UND SOFTWARE

- Über 100 verschiedene Taschenrechner der Marken:
- TEXAS INSTRUMENTS - SHARP - CASIO - MBO u.v.a.
- Über 20 verschiedene Taschen- und Bürodiktiergeräte der Marken: PHILIPS - PEARLCORDER - GRUNDIG - SANYO SONY - MEMOCORD - ASSMANN - TOSHIBA uva.
- Kopiergeräte von SHARP - CANON - MINNESOTA "3M"
- Mehr als 20 Schreibmaschinenmodelle - MECHANISCHE - ELEKTRISCH - ELEKTRONISCH - MIT TYPENHEBEL - MIT KUGELKOPF - MIT TYPENRAD DER MARKEN SMITH CORONA - BROTHER - OLIVETTI u.a.
- ÜBER 20 VERSCHIEDENE TISCHRECHENER
- 6 MODELLE ANRUFBEANTWORTER

UND VIELES ANDERES !

NEUHEITEN - SONDERAKTIONEN - GELEGENHEITSKÄUFE

BASIC-Grundkurs Teil 1

Konzept für BASIC-Grundkurs

Teil 1: Einleitung:

Abgrenzung zu tastenprogrammierbaren Rechnern, Grundlagen der Rechentechnik in BASIC.

Teil 2: Der Rechner und die Programmstruktur:

Betriebszustände, Variablen, Ausführungstaste, arithmetische Operationen, Felder.

Teil 3: Elemente der Programmiersprache:

Beschreibung der wichtigsten Elemente, wie Eingabe, Ausgabe mit Format, Programmbeginn, Programmende, Übungen.

Teil 4: Eingabe, lineare Verarbeitung, Ausgabe:

Praktische Anwendung und Diskussion der Beispiele

Teil 5: Verzweigung:

bedingt, unbedingt, verschiedene Verzweigungsmöglichkeiten

Teil 6: Schleifen:

Einfache, verschachtelte Schleifen mit Übungsbeispielen

Teil 7: Unterprogramme:

Darstellung an Fallbeispielen

Teil 8: Indizierte Verarbeitung:

Eingaben, Sprünge

Teil 9: Speichertechnik:

Mehrfachverwendung von Datenspeichern

Teil 10: Programmierübungen:

Anwendung der Kenntnisse an einigen kleinen etwas größeren Beispielen, als bisher besprochen.

Teil 11: Ausführliches Fallbeispiel:

mit Kommentaren zu den einzelnen Anweisungen.

TEIL 1: Einleitung

Das neue Magazin, das heute vor Ihnen liegt, hat es sich u. a. zur Aufgabe gemacht, nicht nur den Insider der Computer- und vor allem der Softwaretechnik, sondern auch den anzusprechen, der die ersten Schritte in das Reich der Datenverarbeitung auf mehr oder minder spielerische Weise wagen will.
Mit dem Kurs wollen wir auch neue Wege der Wissensvermittlung gehen: Sie lernen fast vom Beginn an mit dem Rechner umzugehen und ihn zu programmieren. Auf diese Weise werden die einzelnen der Reihe nach zu erläuternden Grundanweisungen sofort geübt. Wir werden nicht verabsäumen, Ihnen nach Maßgabe des Lernfortschritts auch Aufgaben zu stellen, die Sie bis zum Erscheinen des jeweils nächsten 99-JOURNALS lösen sollten. Im darauffolgenden Kursabschnitt werden wir, bevor im Vortrag weiter gegangen wird, die Lösungen aus unserer Sicht diskutieren, sodaß Sie die Möglichkeit bekommen, Ihre Lösungen mit unseren Vorschlägen zu vergleichen.
Dem Grundkurs wird ein sogenanntes **Pilotbeispiel** unterlegt und dieses mit jedem Unterrichtsfortschritt erweitert und zu einer umfassenderen Lösung ausgebaut. Daneben werden Ihnen an zusätzlichen Beispielen die Wirkungsweise einzelner Programmfunktionen hautnah vorgeführt. Leser, die zuvor einen tastenprogrammierbaren Taschenrechner benutzt haben, werden in eigenen Spalten dieses Journals in jedem Heft Tips finden, die Ihnen zeigen, welche Punkte besonders zu beachten sind, wenn sie den liebgewohnten TI-59 zurücklegen und sich einem BASIC-Rechner zuwenden; ob dies nun der TI-99/4A oder der neue kleine Compakt-Rechner CC-40 ist, spielt dann keine besondere Rolle. In diesen Tips werden Umstände angesprochen, die in der Programmiersprache BASIC wesentlich anders ablaufen, als Sie das auf dem tastenprogrammierbaren Rechner gewohnt sind. Aber auch Benutzer von UPN-Rechnern finden fallweise Hinweise, die Ihnen den Einstieg in die BASIC-Programmierung erleichtern. Im Verlauf dieses Grundkurses auf derartige Unterschiede in der Programmiertechnik eingehen, hieße den Rahmen dieses Kurses sprengen.
Am Ende werden wir unsere Lernfortschritte an einem umfangreichen Beispiel abprüfen können.
Die Programmiersprache BASIC (abgeleitet von „**B**eginners **A**llpurpose **S**ymbolic **I**nstruction **C**ode") stellt an den Anfänger keine unüberwindlichen Ansprüche. Ursprünglich für Tischrechner und Anlagen der mittleren Datentechnik konzipiert, hat diese Sprache auch in den Bereich der Taschenrechner Eingang gefunden, was sich besonders an der Neuheit CC-40 zeigt. Die Art der Darstellung einzelner Anweisungen,die sowohl über das Tastenfeld, als auch im Wege einer Programm-Anweisung benutzt werden können, ist derart leicht leserlich,daß Sie vom Beginn dieses Kurses kaum mit Verständnisschwierigkeiten zu kämpfen haben werden. Dennoch sei Ihnen aber eine intensive Mitarbeit anempfohlen. Sie sollten die einzelnen dargestellten Beispiele so lange nachvollziehen, bis es ihnen gelungen ist, dieselben gedanklich vollständig zu durchdringen. Wenn Sie darüber hinaus die zeitliche Distanz bis zum Erscheinen des jeweils nächsten Heftes des 99-JOURNALS mit eigenen Programmierübungen ähnlicher Beispiele überbrücken, kann Ihnen nichts mehr passieren!
Der Kurs hat nicht die Absicht, der Bedienungsanleitung Ihres Rechners Konkurrenz zu machen, sondern will darüber hinausgehend die Dinge noch intensiver und praxisnäher darstellen und die Anleitung in dieser Hinsicht ergänzen. Zufolge der Zeitdauer des Kurses werden auch nicht alle **Statements** (Programm-Anweisungen) besprochen, sondern nur jene, die erfahrungsgemäß die größten Anwendungsprobleme bereiten. Nach Abschluß dieses Kurses haben Sie mit

Sicherheit denselben in diesem Punkt durch die Bedienungsanleitung des Rechners ergänzt und spezielle Rechnerfunktionen kennen gelernt, die wir hier aus Zeit- und Platzmangel gar nicht erörtern konnten.
Wir werden fallweise auch auf Unterschiede zu anderen BASIC-Varianten zu sprechen kommen, falls dies erforderlich scheint. Auf diese Weise lernen Sie nämlich auch aus anderen Fachzeitschriften langsam ein BASIC-Programm anderer Rechner lesen und können so versuchen, dieses für das von Ihnen benutzte Gerät, beispielsweise den TI-99/4A oder auch den kleinen CC-40 anwendbar umzuschreiben, falls dies für Sie notwendig ist.
Wir müssen nämlich auch wissen, daß es gerade die am weitesten verbreitete Programmiersprache BASIC ist, die — und da muß man sagen leider — auch die vielfältigsten, meist geräteabhängigen **Dialekte** umfaßt. Nun mag dieser Umstand verschiedene Gründe haben; wir müssen uns damit auseinandersetzen und zusehen, für unsere Rechner auch aus anderen Richtungen interessante Programmierhinweise aufzunehmen, was uns dann besser gelingt, wenn wir zumindest von einigen Unterschieden Kenntnis haben.
Um den Einstieg in die Materie leichter zu schaffen, meinen wir, daß wir Sie in diesem Kurs nicht mit einem Zuviel an **Grammatik** überfordern wollen und uns daher auf die wichtigsten **Sprachelemente** von BASIC beschränken sollten. Wir meinen auch, daß sich gerade an den genannten Geräten die Sprache BASIC sehr leicht wird erlernen lassen, weil das Grundkonzept von BASIC für den TI-99/4A ohnehin nicht allzu umfangreich ist.
Um die Einleitung nicht unnötig auszuweiten, verweisen wir auf die Tips für Anfänger in diesem Heft, in denen wir die Auseinandersetzung mit arithmetischen Operationen betreiben und in diesem Beitrag auch besonders auf die ersten Unterschiede eingehen, die denjenigen besonders auffallen werden, die zuvor einen tastenprogrammierbaren Rechner, beispielsweise einen TI-58 oder Ti-59, benutzt haben.
In diesem Sinne freuen wir uns auf Sie im nächsten Heft, in dem wir allgemeines zum Rechner und der Programmstruktur der Programmiersprache BASIC sagen wollen; der eigentliche Kursbeginn findet dann in Heft 3/83 des 99-JOURNALS statt, wenn wir die Auseinandersetzungen mit Fragen der Problemanalyse eröffnen.
Zum Abschluß dieser Einleitung dürfen wir noch einige Worte zu diesem Thema verlieren: Erfolg oder Mißerfolg hängen vielfach auch damit zusammen, wie weit es gelungen ist, ein Problem oder eine Aufgabe soweit aufzubereiten, daß es für eine rechnergesteuerte Verarbeitung reif ist. In diesem Beitrag werden nur die wesentlichsten Punkte angesprochen, die es Ihnen ermöglichen, die gestellte Aufgabe so zu sehen, daß sich daraus in letzter Konsequenz auch die einzelnen Programmanweisungen ableiten lassen. Wir halten diese für äußerst wichtig; mit diesen Kenntnissen vorbelastet, lassen sich dann die einzelnen Programme auch leichter schreiben. Nachdem gerade diese problemanalytischen Überlegungen kaum wo zu finden sind, mit Sicherheit in keiner Bedienungsanleitung (wo sie auch nicht hin gehören), glauben wir, diesem Thema einen eigenen Kursabschnitt widmen zu müssen, sodaß wir dann in Heft 4/83 mit den ersten Anweisungen der Programmiersprache BASIC beginnen können.
Mit dieser kurzen Kursübersicht verabschieden wir uns heute von Ihnen bis zum nächsten Mal!

●

DIE UHR IM TI99

Im Artikel "Assembler - leistungsfähig und rasch" in diesem Heft wurde unter anderem auch auf die Uhr im TI-99 hingewiesen und ein Programmlisting abgedruckt. Zur Ausführung dieses Programms ist jedoch die Modulbox, die Speichererweiterung, ein Disklaufwerk, usw. erforderlich. Nachfolgend ist ein Programmlisting für dessen Ausführung lediglich das MINIMEMORY-Modul notwendig ist.

Die Erhöhung des Speicherplatzes 31879 der CPU erfolgt sechzig Mal in der Minute um jeweils 1. Diese Erhöhung ist jedoch unabhängig vom laufenden Programm. Das bedeutet, daß man jeweils wo es im Programm angebracht erscheint die "ZEIT" einblenden kann.

Für nähere Informationen siehe Artikel "Arbeiten in Assembler" in diesem Heft.

```
100 REM  VDP-INTERRUPTTIMER
110 REM  ******************
130 REM  CLOCK
150 REM  BY  @APESOFT
151 CALL CLEAR
152 INPUT "START: STUNDE,MINUTE ":STUNDE,MINUTE
200 TIMER=-31879
220 CALL INIT
239 CALL CLEAR
240 PRINT STUNDE;":";MINUTE;":";SEKUNDE
250 CALL LOAD(TIMER,NULL)
270 CALL PEEK(TIMER,TIME)
280 IF TIME<47 THEN 270
284 CALL LOAD(TIMER,NULL)
290 SEKUNDE=SEKUNDE+1
300 IF SEKUNDE=60 THEN 304 ELSE 310
304 SEKUNDE=0
306 MINUTE=MINUTE+1
310 IF MINUTE=60 THEN 314 ELSE 320
314 MINUTE=0
316 STUNDE=STUNDE+1
320 IF STUNDE=24 THEN 324 ELSE 328
324 STUNDE=0
328 CALL CLEAR
330 PRINT STUNDE;":";MINUTE;":";SEKUNDE
340 GOTO 270
```

TIPS für Umsteiger

Tips für Umsteiger: Arithmetische Operationen

Wenn Sie bisher einen tastenprogrammierbaren Taschenrechner betrieben haben, zum Beispiel einen TI-58 oder TI-59, und nun beabsichtigen, auf einen BASIC-programmierbaren Rechner, etwa den neuen ab etwa Mai dieses Jahres erhältlichen CC-40 oder den TI-99/4A umzusteigen, dann sollten Sie sich als erstes mit den **arithmetischen Operationen** auseinandersetzen.
Nachdem die genannten tastenprogrammierbaren Taschenrechner wie die BASIC-programmierbaren zwar beide im Sinne der algebraischen Logik arbeiten, wird es wenig Probleme geben; Umsteiger aus den Bereichen der UPN-Logik werden sich da etwas schwerer tun. Trotz vorhandener Analogie sind doch einige Unterschiede vorhanden, auf die geachtet werden muß. In diesem Beitrag wollen wir Ihnen einige Hinweise dazu geben.

Ausführung einer arithmetischen Operation über das Tastenfeld

Wir beginnen mit einem sehr auffälligen Unterschied zwischen den beiden genannten Rechnersystemen und zwar mit dem **Abschluß einer Rechenoperation:** Tastenprogrammierbare Taschenrechner mit der algebraischen Rechenlogik schließen bekanntlich eine arithmetische Operation mit Betätigung der **Gleichheitstaste** (=) ab. Bei BASIC-Rechnern gibt es eine andere Taste als Ausführungstaste, die (ENTER)-Taste. Bei anderen BASIC-Rechnern kennt man als Operationsabschlußtasten auch solche mit den Bezeichnungen (END OF LINE) oder (NEW LINE).
Die BASIC-(ENTER)-Taste ist aber nicht zu verwechseln mit der(ENTER)-Taste bei den mit der umgekehrten polnischen Notation arbeitenden Taschenrechnern! In der Programmiersprache BASIC hat die auch vorhandende Gleichheitstaste eine ganz andere Funktion: Sie schließt keine Rechenoperation ab, sondern hat die Funktion einer **Wertzuweisung.** Das bedeutet, daß die beiden Operationen

$$A = (2 + 3)^5 \text{ und } A = 3125$$

gleichwertig sind. In beiden Fällen wird ein Zahlenwert 3125 dem Speicher mit der Bezeichnung A (Variablenspeicher zugewiesen, wenn nach dieser Anweisung die (ENTER)-Taste gedrückt wird. Bei Ausführung der analogen Operation auf einem TI-59 müssen folgende Tasten gedrückt werden, wenn das erste der beiden Beispiele über das Tastenfeld ausgeführt wird:

[() (2) (+) (3) ()] (yx) (5) (=)

In der Anzeige erhalten Sie den Wert 3125. Wenn nun diese Zahl für eine weitere Verwendung zwischengespeichert werden soll, muß sie mit einer **zusätzlichen Anweisung,** beispielsweise (ST0) (0) (1) in den Speicher mit der Adresse 01 befördert werden; das ist ein sehr wesentlicher Unterschied. In BASIC kennen wir die Funktion der Anzeige in anderer Weise: Wenn Sie die Potenzrechnung $(2 + 3)^5$ über das Tastenfeld ausführen wollen, müssen Sie folgende Tasten betätigen:

PRINT (() (2) (+) (3) ()) ( ) (5) (ENTER)

In der Anzeige erhalten Sie nun das Ergebnis mit 3125. Sie können nunmehr dasselbe nicht wie bei den tastenprogrammierbaren Rechnern einem Speicher zuweisen! Hätten Sie das beabsichtigt, müßten vor der letzten Tastenfolge die Tastenbetätigungen

(A) (=)

gesetzt werden, um in den Variablenspeicher A die Zahl 3125 hineinzubekommen.

Das Gleichheitszeichen hat also vorrangig die Funktion einer **Wertzuweisung** und wird daher in erster Linie in Programmen verwendet.

Dezimalteil einer Dezimalzahl

Während auf BASIC-Rechnern die Funktion INT (Abtrennung, bzw. Isolierung des ganzzahligen Anteils einer Dezimalzahl) vorhanden ist, fehlt in bisher allen Fällen die analoge Funktion zur Isolierung des Dezimalteils, die beispielsweise bei UPN-Rechnern mit der Funktion (FRC), bei den Rechnern TI-58/59 mit der Tastenfolge (INV) (2nd) (INT) ausgeführt werden kann. Um in BASIC diese Operation ausführen zu können und gleichzeitig mit dem Dezimalteil weiterzurechnen, ist es erforderlich, daß dieser Dezimalteil nach Abtrennung in einem Hilfsspeicher, beispielsweise B, abgestellt wird. Der Dezimalteil ist mit folgender Anweisung zu berechnen, wobei vorausgesetzt wird, daß die Dezimalzahl im Speicher A vorliegt:

B = A — INT A

Addition in einen Speicher

Bei tastenprogrammierbaren Taschenrechnern hatte für die algebraische Logik die Operation mit den Tasten Aufsummierung der Zahl 3125 in Speicher 01 mit der Betätigung der Tasten

(3) (1) (2) (5) (SUM) (0) (1)

ausgeführt werden müssen; für BASIC und den Variablenspeicher A sieht der Vorgang so aus:

A = A + 3125

Die übrigen auf den Grundrechnungsarten basierenden speichertechnischen Operationen werden in analoger Weise ausgeführt.

Erhöhung des Speicherinhalts um 1

Beim TI-58/59 gab es für die Speicher R00 bis R09 eine bequeme Form der Anhebung des Speicherinhalts um den Wert 1; im folgenden Beispiel verwenden wir dazu wieder den Speicher mit der Adresse 01:

(OP) (2) (1)

Der BASIC-Rechner schafft diese Operation mit der Anweisung

A = A + 1

Ermäßigung des Speicherinhaltes um 1

Analog dazu gilt, daß beiden genannten Taschenrechnern für die gleichen Register 00 bis 09 ebenfalls eine einfache Reduzierung des Speicherinhalts um 1 möglich ist; dazu sind die Tasten

(OP) (3) (1)

zu betätigen. Der BASIC-Rechner benötigt zur gleichen Vorgangsweise die Anweisung

A = A — 1

Nun gibt es doch einen gravierenden Unterschied: Während die Taschenrechnerfunktion **nur** bei den Registern 00 bis 09 funktioniert, zumindest auf diese einfache Weise, klappt die vorgenannte Operation bei **allen** BASIC-Speichern. Hätte man beispielsweise die Verminderung um die Zahl 1 am Speicherinhalt mit der Adresse 10 vornehmen wollen, hätte diese Operation mit den Tasten

1 (INV) (SUM) (1) (0)

ausgeführt werden müssen; analoges gilt auch für die Anhebung des Speicherinhalts eines beliebigen Speichers um 1 ab einschließlich der Adresse 10.

Kaufmännische Rundung

Der BASIC-Umsteiger wird sich auch mit der Rundung befassen müssen, wenn er beispielsweise von einem TI-58/59 auf einen BASIC-Rechner gehen will. Auf seinem „alten" Gerät war die Rundung kein Problem: Er stellt seinen Rechner auf eine Fix-Komma-Einstellung, beispielsweise mit (2nd) (Fix) (2) auf eine Anzeige mit zwei Dezimalstellen und rundet den augenblicklich in der Anzeige befindlichen Wert mit der Tastenfolge (EE) (INV) (EE). Eine analoge Funktion findet sich auf BASIC-Rechner nicht. Auch in diesen Fällen regiert die Gleitkommaanzeige, die mit einer Formatanweisung auf eine bestimmte Anzahl an Dezimalstellen eingestellt werden kann. Nur wird man in diesen Fällen damit rechnen müssen, daß mit der Formatfunktion keine automatische Rundung verbunden ist, wie dies beim Taschenrechner der Fall war. Wollen wir eine Rundung auf N Dezimalstellen vornehmen, müssen wir den im Variablenspeicher A befindlichen Wert wie folgt behandeln; die Anzahl der Dezimalstellen befindet sich im Variablenspeicher N:

A = INT (A · 10 ^ N + 0,5) / (10 ^ N)

Damit ist die Zahl in A auf N Dezimalstellen gerundet. Der Trick dieser Operation liegt darin, daß wir mit dem Operator 10 ^ N die bewußte Zahl einer Dezimalstellenverschiebung unterzogen, worauf die Dezimalzahl 0,5 addiert und mit der Funktion INT das Ergebnis zu einer ganzen Zahl gemacht wurde. Die nachfolgende Rückverschiebung des Dezimalpunktes an die alte Stelle gab der ursprünglichen Zahl ihre alte Größenordnung zurück; auf diese Weise kann bei einer beliebigen Anzahl an Dezimalstellen auch bei BASIC-Rechnern gerundet werden. Man wird im Bedarfsfall dafür allerdings ein Unterprogramm einrichten, muß aber für zwei Hilfsspeicher sorgen: Einen für die zu rundende Zahl und den zweiten für die Anzahl der gewünschten Dezimalstellen, auf die die betreffende Zahl zu runden ist.
Damit sind die wichtigsten Operationen im arithmetischen Bereich erwähnt. Wir meinen, daß diesen Hinweisen entsprechende Beachtung geschenkt werden sollte.
In dieser Ecke werden Sie auch in Hinkunft mit einer Reihe von Unterschieden konfrontiert sein, die zwischen den beiden Rechnerarten bestehen.

●

FACHLITERATUR

Diese Rubrik wird sich darum bemühen Ihnen von Neuerscheinungen zu berichten. Ebenso wird hier auf interessante Allgemeinliteratur hingewiesen werden.

ASSEMBLER-HANDBUCH IN DEUTSCH

von Rainer BERNERT
Alle Maschinen/Assemblerbefehle
Grundsätzliche Informationen
Für schnelles Programmieren in Assembler.
Für MINIMEMORY-MODUL und EDITOR/ASSEMBLER.
PREIS: öS 280,- (Für Klubmitglieder: - 10 %)
BEZUG: TI-99 JOURNAL 1060 Wien, Mollardgasse 32/3

EXTENDED BASIC IN DEUTSCH

Durch besonders verständliche Form (ähnlicher Aufbau wie die Originalausgabe) ist dieses Handbuch ein unentbehrliches Werkzeug für die Arbeit mit EXTENDED BASIC.
PREIS: öS 390,- (Klubmitglieder: - 10 %)
BEZUG: TI-99 JOURNAL 1060 Wien, Mollardgasse 32/3

DER KLEINE RIESE

COMPACT-COMPUTER CC-40

NEU

Der CC-40 ist der erste einer neuen Computerserie von Texas Instruments, der Compact Computer 40. Der CC-40 ist programmierbar in erweitertem BASIC und kann mit vorprogrammierten Software-Modulen eingesetzt werden, die entweder via einsteckbaren Solid-State-Modulen oder von kleinen Bandkassetten eingelesen werden, den sogenannten Wafertape TM - Kassetten. Das System wird von Batterien gespeist und findet in einer Aktentasche Platz. Es kann für Rechnungen vor Ort, für Datenmitteilungen verwendet werden, oder als kleiner Tischcomputer ohne Kabel für den privaten Gebrauch.

UNBEGRENZTE MÖGLICHKEITEN

Der CC-40 besitz einen eingebauten 34K-Byte Lesespeicher (ROM), der einen extended BASIC-Interpreter enthält, der Ihnen das Programmieren in erweitertem BASIC unmittelbar nach dem Einschalten gestattet.

Ein Anschluß für steckbare Module gestattet die Anwendung von Software bis 128 K-Byte ROM-Speicher. Er kann auch verwendet werden, um die RAM-Speicherkapazität bis 18 K-Bytes des CC-40 zu erweitern. Die Rückseite der Konsole enthält einen Interface Hexbus TM. Damit kann ein beliebiges Hexbus TM - Peripheriegerät an die Baureihe des CC-40 angeschlossen werden.

PREISGÜNSTIGE PERIPHERIEGERÄTE ERWEITERN DEN LEISTUNGSBEREICH

Der Drucker/Plotter HX-1000 gestattet graphische Darstellungen mit x- und y-Koordinaten in vier Farben, wie Kreisdiagramme, Regressionskurven. Auch Programmlisten und Ergebnisse druckt er aus... bis zu 36 Zeichen pro Zeile. Er verwendet Normalpapier in 57 mm breiten Rollen.

Der Watertape TM Digital-Bandantrieb HX-2000 stellt Ihnen bis 48 K-Byte Speicherkapazität pro Kassette zur Verfügung. Dies gestattet schnellen Zugriff zu Programmen und Daten und bietet echte Vorteile gegenüber der Speicherung auf Kassetten. Der vollständig von der Konsole des CC-40 aus gesteuerte Watertape TM Antrieb ist leicht zu benutzen und arbeitet sehr schnell (Datenübertragungsgeschwindigkeit 8.000 Bit pro Sekunde) und zuverlässig.

RS-232, Parallel Interface, HX-3000.

Für den Anschluß des CC-40 an einen Standard-Drucker, ein Modem, oder sogar an einen anderen Computer.

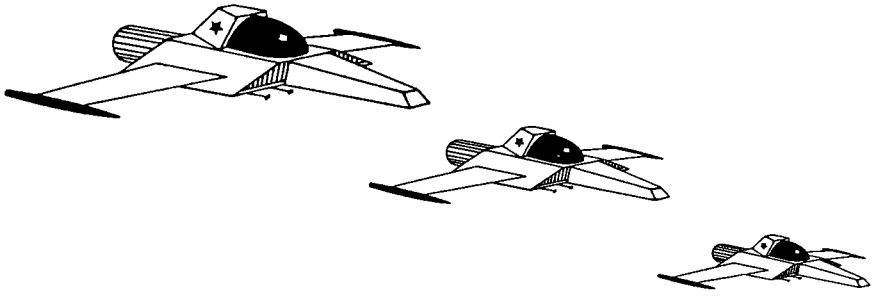
DIE SOFTWARE ERWEITERT SEINE KAPAZITÄT

Achtzehn Software-Pakete - vier einsteckbare Solid-State-Software - Module TM und 14 Wafertape TM - Kassetten sind jetzt bereit. Die Dialoge werden in der von Ihnen gewählten Sprache angezeigt: Englisch, Französisch oder Deutsch.

TECHNISCHE DATEN

Mikroprozessor	:	8 Bite - TMS 70C20 - CMOS CPU.
ROM-Speicher	:	Eingebaut: 34 K Byte Erweiterung: bis 128 K-Byte Solid State Software.
RAM-Speicher	:	Eingebaute Kapazität 6 K Erweiterungsmöglichkeit bis zu 18 K.
Max. RAM-Speicher	:	18 K-Byte Constant Memory TM.
Max. Gesamtspeicherkapazität (ROM+RAM)	:	168 K-Byte
Programmsprachen	:	BASIC und TMS 7000 Assembler.
Anzeige	:	31 Zeichen - Punktmatrix (5 x 8). 18 Indikatoren (davon 6 programmierbar) Anzeige - Kontrasteinstellung.
Tastatur	:	Tasten mit automatischer Wiederholung.
Summton	:	Durch Software gesteuert.
Automatische Stromsparschaltung TM	:	Kann durch die Software außer Kraft gesetzt werden.
Stromquelle	:	4 Mignon - Batterien (AA). Batterie-Lebensdauer 200 Stunden (Alkali Mangan Batterien) Netzadapter Zubehör.
Abmessung	:	240 mm x 145 mm.
Gewicht	:	600 Gramm.

PARSEC-Kampf im All



DAS NEUE SPIELMODUL VON TI

erforderliche Konfiguration: Konsole empfehlenswerter Zusatz: JOYSTICKS

KAMPF IM ALL

Als Commander des Raumschiffes PARSEC sieht man sich einer Vielzahl von "tötlichen" Gefahren gegenüber. Hier heißt es richtig und rasch gegen die Angriffe von sieben verschiedenen außerirdischen Kampfschiffen zu reagieren sich einen Weg durch einen Asteroiden-Gürtel zu schießen und in immer enger werdenden Tanktunnels Treibstoff nachzuladen.

Hochauflösende Grafik erzeugt ein "PLASTISCHES WELTALL".
Das Raumschiff ist horizontal, vertikal, vorwärts und rückwärts bewegbar und mit einer Laserkanone ausgestattet.
Bei angeschlossenem Sprachsynthesizer erfolgen alle wichtigen Warnungen und Hinweise auch noch akustisch!

Drei verschiedene Geschwindigkeitsstufen ermöglichen ein differenziertes Manövrieren des Raumschiffes, um das Steuern durch den Tanktunnel komfortabler zu gestalten.

Die Abbildung zeigt die verschiedenen "Feinde" und die Trefferhöhe der verschiedenen Schwierigkeitsstufen.

Zuerst erscheinen die SWOOPERS, unbewaffnet, aber auf Kollisionskurs, die, wenn sie nicht abgeschossen werden immer rascher fliegen.

Als Commander muß man nicht nur gegen die fliegenden Feinde kämpfen. Ein überhitzter Laser führt ebenso zur Zerstörung des eigenen Raumschiffes, wie das Auftreffen auf dem Boden.
Doch nun wird man das erste Mal mit bewaffneten Jägern konfrontiert, den mit Photonen-Kanonen ausgerüsteten URBITES.

Dann kommen die schmalen LTF's (Light Triangular Fighters), die den SWOOPERS nicht unähnlich sind.

Die DRAMITES sind auch kanonenbestückt, reagieren aber unglaublich rasch und auch die Schußfolge ist wesentlich höher als die der URBITES.

Als nächstes sieht man sich den SAUCERS gegenüber, die versuchen das Raumschiff von hinten zu kollidieren, dann ihren Kurs ändern und von vorne angreifen.
Das Angriffsschema ist zufällig!

Zum Schluß trifft man auf die BYNITES. Ähnlich den URBITES schießen sie jedoch ein ganzes Bündel von hochexplosiven Raketen ab.

Nun heißt es, falls man es bis hierher geschafft hat, noch durch den Asteroidengürtel einen Weg zu schießen, natürlich ohne den Laser zu überhitzen.

Die Planetenoberfläche färbt sich von gelb auf grün, um die 2. Schwierigkeitsstufe anzuzeigen.

Nun müssen alle Angreifer zweimal getroffen werden.
Ab dem dritten Durchgang müssen alle dreimal getroffen werden, außerdem wird der Luftkampf viel schneller, die Angreifer manövrieren rascher und der Abstand zum eigenen Schiff wird geringer

BEWERTUNG:

Was uns gefällt:

Einzigartige grafische Auflösung
Gut eingesetzter Ton
Enorme Steigerungsfähigkeit

Was uns weniger gefällt:

Für den Anfänger geht alles ein bißchen rasch und so bleibt kaum Zeit, um die Warnungen und Anleitung, die in die "Planetenoberfläche" eingeblendet werden, zu lesen.

●

PARSEC

SCORING SYSTEM CHART

LEVEL	SCREEN COLOR	SWOOPERS	URBITES	LTFs	DRAMITES	SAUCERS	BYNITES	ASTEROID BELT	KILLER SATELLITES
1	DARK YELLOW	100	100	200	200	300	300	1000	-
2	DARK GREEN	200	200	300	300	400	400	2000	-
3	DARK RED	300	300	400	400	500	500	3000	-
4	DARK BLUE	300	300	400	400	500	500	4000	400
5	MAGENTA	300	300	400	400	500	500	4500	400
6	LIGHT RED	300	300	400	400	500	500	5000	400
7	LIGHT GREEN	300	300	400	400	500	500	5500	400
8	CYAN	300	300	400	400	500	500	6000	400
9	DARK BLUE	300	300	400	400	500	500	6500	400
10	MEDIUM RED	300	300	400	400	500	500	7000	400
11	LIGHT BLUE	300	300	400	400	500	500	7500	400
12	MEDIUM GREEN	300	300	400	400	500	500	8000	400
13	GREY	300	300	400	400	500	500	8500	400
14	WHITE	300	300	400	400	500	500	9000	400
15	DARK YELLOW	300	300	400	400	500	500	9500	400
16	DARK YELLOW	300	300	400	400	500	500	10000	400

CPU des TI99

Ich möc e in dieser Spalte über das Innenleben des TI-99 schreiben.
Zuerst werde ich auf die einzelnen Bauteile eingehen und dann deren Zusammenwirken beschreiben

Ziel dieser Artikel ist es über die Hardware und deren Anwendung zu informieren.

Sollten Sie Fragen, die Hardware betreffend, haben so schreiben Sie bitte an unsere Redaktion, Kennwort: HARDWARE.

Allgemeines über die CPU

Oft wird der Name CPU (Central Processor Unit) genannt, doch die wenigsten wissen, daß diese aus vielen Bauteilen besteht und viele verschiedene Operationen ausführt.

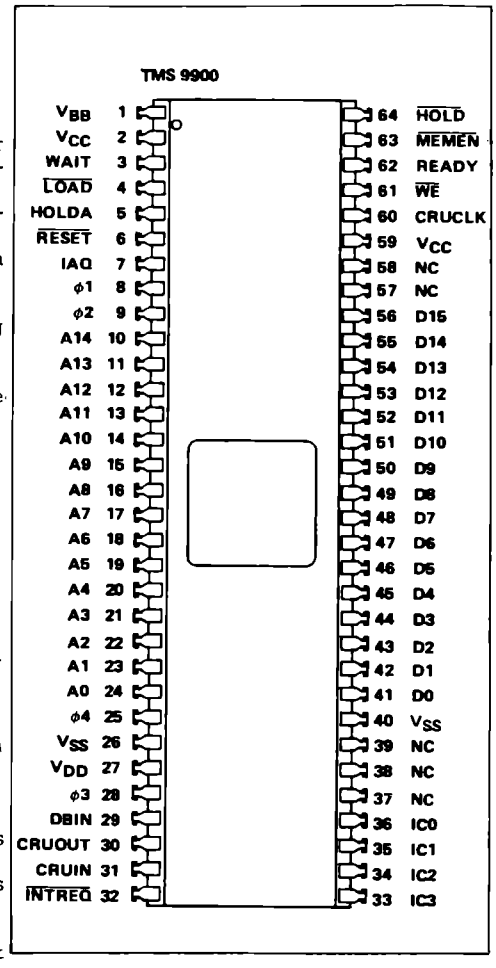
Der Zweckmäßigkeit halber werden diese Bauteile oft auf einem Chip vereinigt.

Um sich ein Bild von der Arbeitsweise der CPU machen zu können, verfolgen wir einen Befehlzyklus

Die Hardware Register der CPU (TMS 9900), WP und PC müssen mit Speicheradressen geladen sein.
Es wird ein Wort (Adresse des PC) in die CPU geladen.
Der Befehl wird im IR (Instruction Register) abgelegt und von der Instruction decode and controll unit ausgewertet.
Sie sagt auch ob ein weiteres Wort (z.B. Operand) in die CPU geholt wird. Eine Speicheradresse wird, wenn nötig, generiert (JUMP-Befehle).

Nun führt der Kern der CPU die ALU eine Summe von logischen und arithmetischen Operationen aus.
Die ALU ist in MIKRO programmiert. Diese "Programmiersprache" kennt nur UND, ODER, NICHT, SHIFT, .. und erzeugt durch feste Verdrahtung dieser Elemente die Maschinenbefehle.
Nachdem das Ergebnis vorliegt wird das Status Register beeinflußt und der PC erhöht.

Das Register S dient zum Abspeichern von Adressen.
Das Ergebnis wird laut Befehl dem Speicher übergeben.
Nun ist die CPU für einen neuen Befehlszyklus bereit.



Die CPU

Central Processor Unit

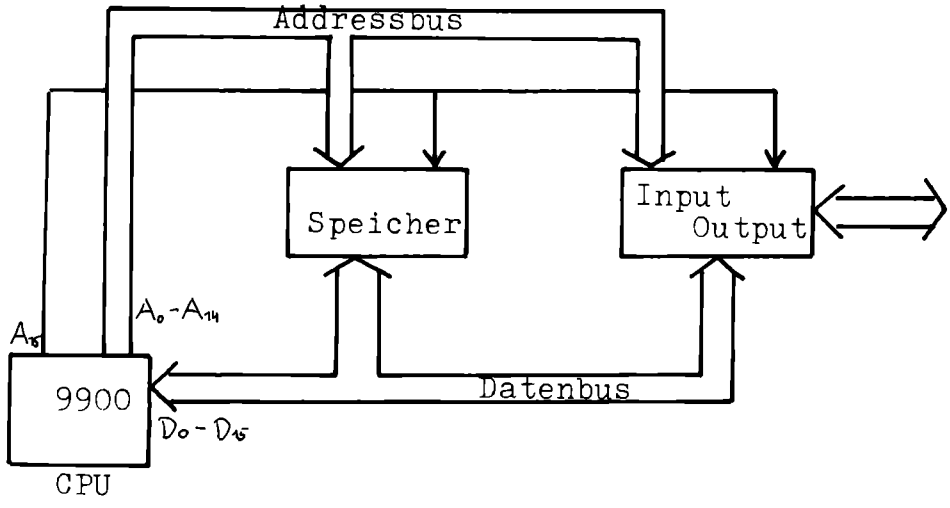
Im TI-99 ist eine TMS 9900 implementiert.
Der TMS 9900 Mikroprocessor ist eine ein-Chip 16 bit CPU. Er war einer der ersten 16-bit CPU's und wurde für Mini-Mikrocomputer entworfen.

Allgemeine Daten:

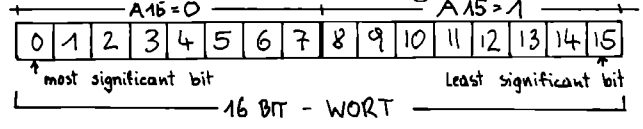
32 768 Wörter (65 K Byte) adressierbar
15 Addressleitungen (16. Leitung = Phantomadresse)
16 Datenleistungen
3 Betriebsspannungen (- 5 V, 5 V, 12 V)
3,3 MHz Taktfrequenz (4 Takte)
3 Register in der CPU (Hardwareregister)
Arbeitsbereichzeiger
Programmzeiger
Zustandsregister
16 Register im RAM (Softwareregister)
64 PIN's
69 Befehle

Die CPU besitz 3 Hardwareregister. Daneben stehen 16 Softwareregister für arithmetische und logische Verknüpfungen zur Verfügung.

Diese Softwareregister sind im RAM untergebracht. Es kann mit einem Befehl ein anderer RAM-Bereich als Arbeitsbereich definiert werden. Der Vorteil, daß Registerinhalte nicht verloren gehen liegt auf der Hand.



most significant Byte gerade Adresse — least significant Byte ungerade Adresse



Die meisten Operationen des 9900 sind Wort-Oper.

Ein Wort besteht aus 2 Byte mit je 8 Bit. Bit 0 ist das signifikante Bit. Ein Wort wird mittels "Phantomadresse" A 15 in 2 Byte gespalten. Dies gibt die Möglichkeit der 8 Bit Datenorganisation Dadurch spart man Leistungen. Im TI-99 werden die Daten wie folgt verarbeitet:

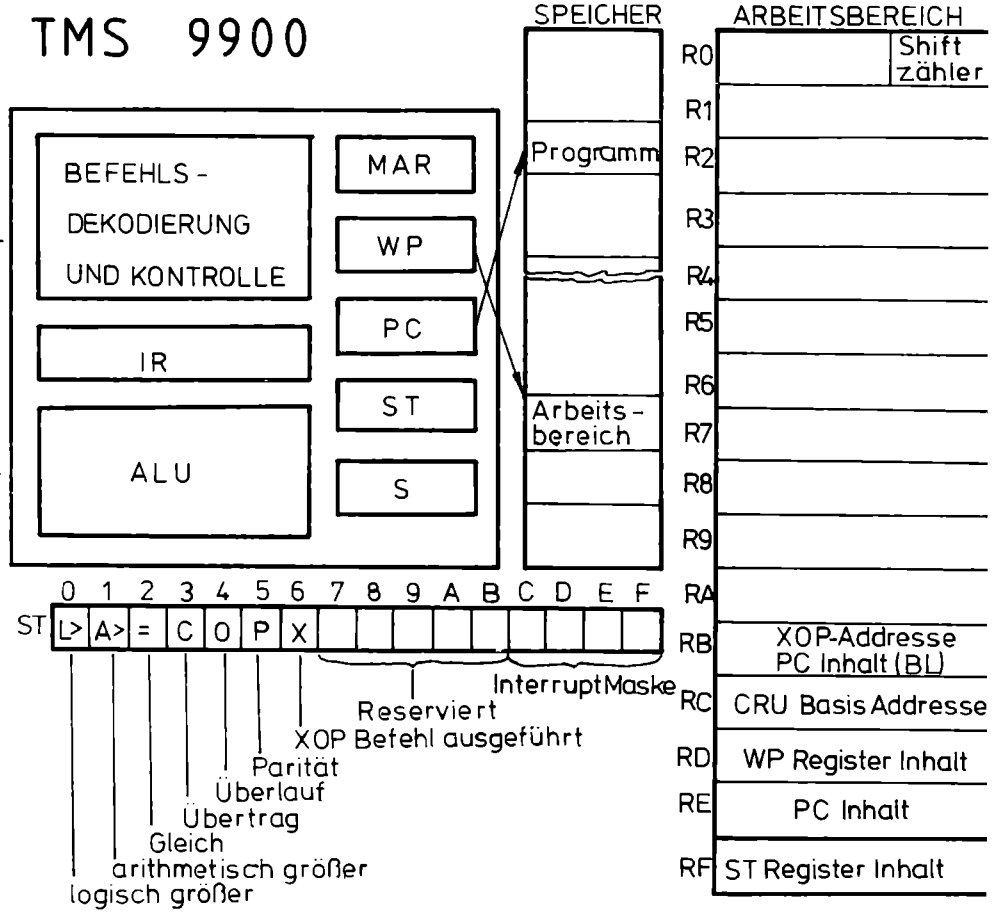
Auf die Adressleistungen A0 - A14 wird eine Adresse gelegt. A15 ist 0. Nun werden die ersten 8 Bit gelesen und abgelegt.
A15 wird 1. Die zweiten 8 Bit werden gelesen und mit den ersten 8 zu einem Wort verknüpft und an die CPU weitergegeben.

Der Schreibvorgang ist analog. Hier werden die Signale von einem 16 Bit zu 8 Bit Daten-Konverter aufgespalten.

Programmieren des 9900 CPU:

69 Befehle machen das Programmieren in Assembler relativ angenehm. Auch erkennt man bald den Vorteil der Softwareregister im RAM.

Obwohl die TMS-9900-CPU schon eine alte Dame ist kann sie heute durchaus noch Vergleichen standhalten.



IR.. Instructin Register (= Befehlsregister)
ALU. Arithmetic Logic Unit führt einen Maschinenbefehl als Summe vieler logischer und arithmetischer Operationen aus.
MAR. Memory Address Register (= Speicheradressenregister) speichert die externe Speicheradresse.
WP.. Workspace Pointer (=Arbeitsbereich Zeiger) zeigt (beinhaltet) auf die Adresse von R0 der Softwareregister. 15 folgende Wörter sind die Register R1 - RF.
PC.. Program Counter (=Programmzähler) zeigt (beinhaltet) auf die als nächstes auszuführende Adresse (Operation).
ST.. Status Register zeigt den Status des abgelaufenen Programms an.
XOP. EXtended OPeration Maschinenbefehl. Führt getrennt vom Hauptprogramm Anweisungen aus.
CRU. Communication Register Unit ein befehlsorientiertes Ein- Ausgabe bit-adressierbares Interface.

HARDWAREREGISTER:

Arbeitsbereichzeiger:

Dieses Register beinhaltet die Adresse von R0. 15 darauffolgende Speicherwörter sind R1 - R 15

Programmzählerregister:

Dieses Register beinhaltet die Adresse der nächsten Instruktion

Zustandsregister:

Es beinhaltet 16 Flags die den Zustand des abgelaufenen Programms kennzeichnen.

SOFTWAREREGISTER:

Register 11 - 15 haben beim Verzweigen in Programmen eine definierte Funktion, um den Zustand vor der Verzweigung wiederherstellen zu können.

Es stehen alle Register für Operationen zur Verfügung.

Pinbelegung des TMS 9900

Bezeichnung	Pin	In/Out
A0 (MSB)	24	O
A1	23	O
A2	22	O
A3	21	O
A4	20	O
A5	19	O
A6	18	O
A7	17	O
A8	16	O
A9	15	O
A10	14	O
A11	13	O
A12	12	O
A13	11	O
A14 (LSB)	10	O

Beschreibung

ADRESSBUS

A0 bis A14 stellen den Adressbus dar. Dieser 3-state Bus dient zum Transport von Adressen wenn MEMEN aktiv ist zu einem externen Speicher.

Wenn MEMEN nicht aktiv ist, dann transportiert er I/O-bit Adressen und externe Befehlsadressen zum I/O-System.

DO (MSB)	41	I/O	DATENBUS DO bis D15 stellen den Datenbus dar. Dieser 3-state Bus transportiert Daten zum (schreiben) und vom (lesen) externen Speicher wenn MEMEN aktiv ist.
D1	42	I/O	
D2	43	I/O	
D3	44	I/O	
D4	45	I/O	
D5	46	I/O	
D6	47	I/O	
D7	48	I/O	
D8	49	I/O	
D9	50	I/O	
D10	51	I/O	
D11	52	I/O	
D12	53	I/O	
D13	54	I/O	
D14	55	I/O	
D15	56	I/O	
V_{BB}	1		STROMVERSORGUNG - 5 V NOM
V_{CC}	2,59		5 V NOM Pin 2 und Pin 59 parallel 12 V NOM
V_{DD}	27		
V_{SS}	26,40		Erdung Pin 26 und Pin 40 prallel
O1	8	In	TAKTE Die Takte werden vom TMS 9904 erzeugt.
O2	9	In	
O3	28	In	
O4	25	In	
DBIN	29	Out	KONTROLLBUS Data Bus IN. Wenn DBIN aktiv ist zeigt, daß der TMS 9900 seine Output Buffers unzugänglich gemacht hat um Speicherdaten über den Datenbus während MEMEN aktiv ist aufzunehmen.
$\overline{\text{MEMEN}}$	63	Out	MEMory ENable. (= Speicher zugänglich). MEMEN deutet darauf hin, daß der Adressbus eine Speicheradresse enthält.
$\overline{\text{WE}}$	61	Out	Write Enable (= schreiben möglich). Wenn aktiv dann hält der Prozessor Daten zum Schreiben bereit.
CRUCLK	60	Out	CRU CLocK. Zeigt, daß Daten einer externen Interface Logik an CRUOUT anliegen oder daß über A0 bis A2 Dekodierungsdaten anliegen
CRUIN	31	In	lädt Daten von einer externen Interface Logik in den Prozessor.
CRUOUT	3+	Out	Hier werden seriell Daten ausgegeben bei bestimmten Befehlen und wenn CRUCLK aktiv ist.
$\overline{\text{INTREQ}}$	32	In	INTerrupt REQuest (= Unterbrechungsaufforderung). Wenn aktiv, dann wird ein Interruptcode von IC0 bis IC3 geladen (Priorität der Unterbrechungen kann geändert werden).
IC0 (MSB)	36	In	Unterbrechungskodes.
IC1	35	In	
IC2	34	In	0001 höchste Priorität 1111 niedrigste Priorität
IC3 (LSB)	33	In	
$\overline{\text{HOLD}}$	64	In	SPEICHERKONTROLLE Veranlaßt den TMS 9900 in einen HOLD Status zu gehen damit z.B. eine externe Logik direkten Speicherzugriff hat.
HOLDA	5	Out	Zeigt, daß der Prozessor im HOLD Status ist.
READY	62	In	(= Bereit) Zeigt, daß der Speicher für Schreib- oder Leseoperationen bereit ist.
WAIT	3	Out	(= Warte) Zeigt, daß der Prozessor im WAIT Status ist.
IAQ	7	Out	TIMING Instruction AQuisition. Ist aktiv während eines Speicherzykluses wenn der Prozessor einen Befehl aus dem Speicher holt.
$\overline{\text{LOAD}}$	4	In	Lädt von einer definierten Speicheradresse WP und PC. Wenn LOAD zwischendurch angesprochen wird, dann wird zuerst ein RESET ausgeführt.
$\overline{\text{RESET}}$	6	In	Mit RESET wird ein definierter Anfangszustand erreicht. (WP, PC bekommen Werte zugewiesen)

Ein Strich über einer Bezeichnung heißt, daß der Pin LOW-AKTIV ist.
Das heißt, der Normalzustand ist 5 V.
Bei Aktivierung wird auf den Pin eine Spannung von 0 V gelegt.

TI99-JOURNAL-
PREISAUSSCHREIBEN

GEWINNEN SIE EINES VON 3 PARSEC-MODULEN

In diesem TI-99 JOURNAL PREISAUSSCHREIBEN geht es um Ihr Wissen aus der Welt der Computertechnik.
Die Redaktion hat für Sie sieben Fragen zusammengestellt.
Die jeweiligen Buchstaben der richtigen Einzelantworten ergeben einen wichtigen BASIC-Befehl.

Das richtige Lösungswort schreiben Sie bitte auf eine Postkarte und senden diese an:

Kennwort: PARSEC
Thomas M. FIEDLER
TI-99 JOURNAL
Mollardgasse 32/3
1060 Wien

Einsendeschluß ist der 30. April 1983 (Datum des Poststempels.)

Die Namen der Gewinner werden in der JUNI-Ausgabe des TI-99 JOURNALS veröffentlicht.

Die Gewinner werden unter Ausschluß des Rechtsweges ermittelt. Mitarbeiter der Zeitschrift und deren Angehörige sind von der Teilnahme ausgeschlossen.

1) Englischer Ausdruck für die Steuerknüppel des TI-99.
   a) Control Device — K
   b) Joy sticks — D
   c) Control sticks — L

2) Was bedeutet "PAL" beim Fernsehgerät?
   a) Phase Alternating Line — I
   b) Primary Acces Line — E
   c) Parallel Activities Line — O

3) Wo werden die Töne des TI-99 erzeugt?
   a) Im Programm — P
   b) Im RAM-Teil — R
   c) Im Sound-Generator — S

4) Was bedeutet der Ausdruck "ROM"?
   a) Read Only Memory — P
   b) Read Only Modify — M
   c) Real Operating Modul — T

5) Eine Schnittstelle ist
   a) Hardware/Software, die notwendig ist um ein Gerät oder System mit einem anderen zu verbinden — L
   b) Sprung in ein Unterprogramm — N
   c) Leitung die ausschließlich zur Übertragung von Daten vom Computer zum Drucker dient — F

6) Die CPU des TI-99 heißt:
   a) TPS 9090 — F
   b) TMS 9900 — A
   c) TMS 9009 — U

7) Mit dem BASIC-Befehl POKE kann man:
   a) Den Inhalt einer Speicherstelle lesen — Z
   b) Ein Assemblerprogramm laden — X
   c) Einen Wert in eine Speicherzeile laden — Y

Impressum

Medienverleger und -herausgeber:
Thomas M. Fiedler
Mollardgasse 32/3
1060 Wien

Layout und Produktion:
Werbe Werkstatt Wiesmüller

Hersteller:
Werbe Werkstatt Wiesmüller
Krottenbachstraße 1
1190 Wien 37 41 58

Redaktion:
Rainer Bernert
Fritz Brezina
Thomas Fiedler
Dipl. Ing. Dr. Peschetz
Dipl. Ing. Tatzl
Alle:
1060 Wien, Mollardgasse 32/3

Verlags- und Herstellungsort:
WIEN

Offenlegung § 25:
Medienverleger Thomas M.Fiedler, 1060 Wien, Mollardgasse 32/3, zu 100 %.
Die Richtung der Zeitung ergibt sich aus den Meinungen der Redakteure und ist sachlich informativ.

Anzeigenpreise:
Für die Anzeigenpreise gilt derzeit die Anzeigenpreisliste 1/83

Bei Kleinanzeigen gelten folden de Sätze:
Private: öS 30,-/Zeile inkl. aller Abgaben
Gewerbl. öS 46,-/Zeile exkl. aller Abgaben
Eine Zeile besteht aus 32 Zeichen (einschl. Satzzeichen und Zwischenräumen)
Chiffregebühr: öS 30,- exkl. MWSt

Erscheinungsweise:
zweimonatlich

Bezugspreis:

Jahresabonnement:
Inland: öS 420,- (inkl. Porto und Verpackung)
Ausland: BRD DM 66,-
Schweiz sfr 63,-
sonstige Länder auf Anfrage

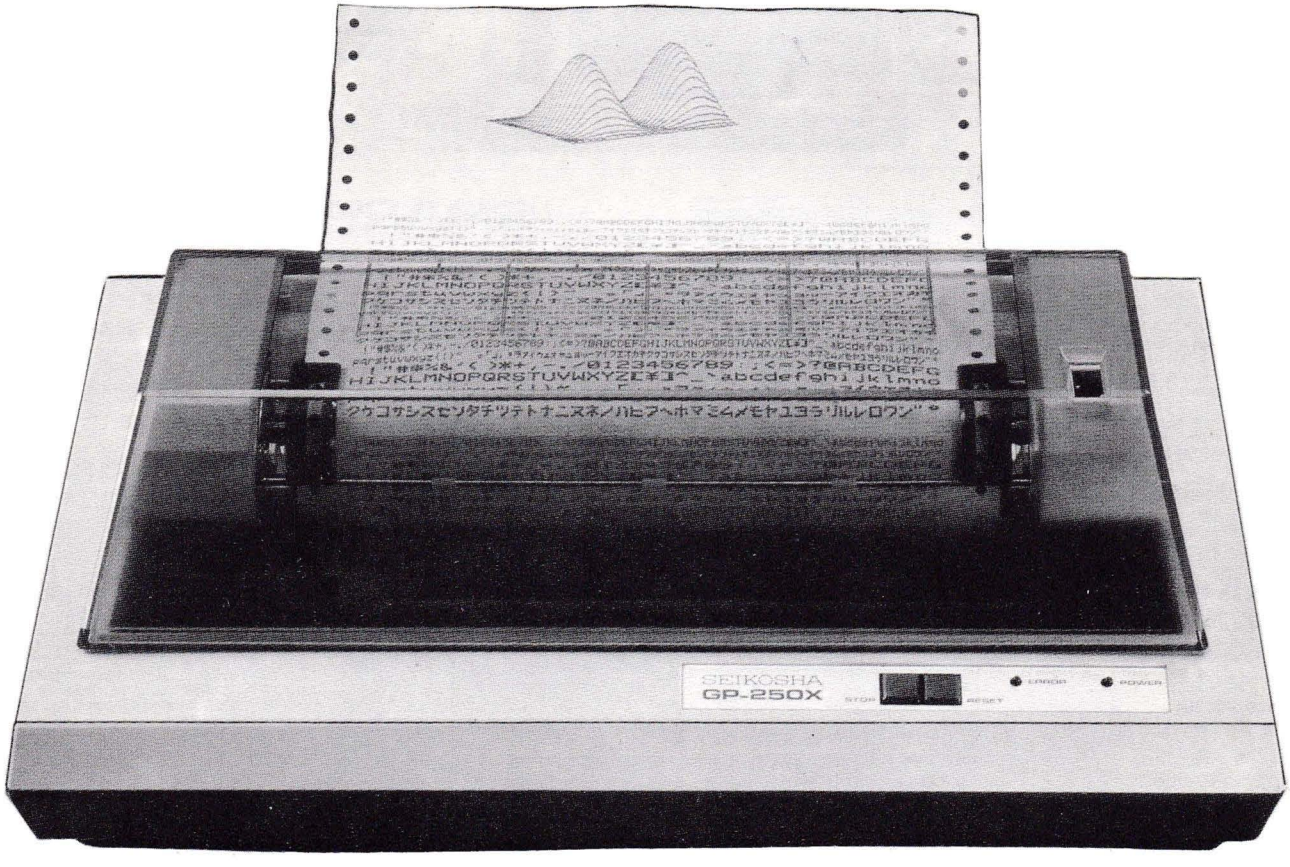
Bankverbindung:
PSK-Wien 6.333.544

AN ALLE TI-FANS!

(Und solche die es noch werden wollen)

Wir bieten Ihnen den Grafikprinter,
der zu Ihrem TI paßt und den Sie
immer schon gesucht haben — den

SEIKOSHA GP-250 X



50 ZEICHEN/SEC.
VOLLGRAFIKFÄHIG
DEUTSCHER ZEICHENSATZ
DOPPELT HOCH / BREITSCHRIFT
NORMALPAPIER
64 ZEICHEN ZUSÄTZLICH FREI DEFINIERBAR
V 24 + CENTRONICS INTERFACE STANDARD
DAS WICHTIGSTE IN ALLER KÜRZE:

PREIS: **öS 8.250,—** EXCL. MWST.

ERHÄLTLICH: BEI IHREM TI-HÄNDLER GENERALVERTRIEB:

burisch
Vertrieb elektrotechnischer Erzeugnisse

Burisch Ges.m.b.H. & Co. KG
A-1210 Wien, Scheydgasse 31 · Telefon (0 22 2) 38 76 38-0
Telex 01-13310

- Thanks to 99'er User: ***Alfredo Cevolini***, for the Scan of this document.

- Reworked by TI99 Italian User Club (info@ti99iuc.it) - 2014

Downloaded from www.ti99iuc.it

Herlango hat das größte Computer-Sortiment

preisgesenkt

commodore

Commodore VC 20
5 Kb RAM erweiterbar auf 32 Kb/20 Kb ROM, BASIC und Assembler, Video- und Antennenanschluß an jeden Fernseher, 16 Farben, 4 Tongeneratoren/eingebautes Cassetten- und serielles Interface (direkter Anschluß von Massenspeicher und Drucker)
statt ~~5.900,–~~ nur **4.450,–**

Commodore VC 64 NEUHEIT!
64 Kb RAM, BASIC und Assembler, CP/M-Option, Video- und Antennenanschluß an jeden Fernseher, 16 Farben, 4 Tongeneratoren, eingebauter Synthesizer, hochauflösende Graphik, eingebautes Cassetten- und serielles Interface
nur **9.990,–**

Drucker VC 1525
Matrixdrucker, 80 Zeichen pro Zeile, 30 Zeichen/s
nur **7.375,–**

Floppy Disk VC 1541
„intelligenter" Massenspeicher, direkter Zugriff, 170 Kb auf 5¼"-Diskette
statt ~~8.895,–~~ nur **7.375,–**

Norminterface RS 232/V 24 VC 1011
statt 1.290,– nur **945,–**

Spielmodule und Spielcassetten
viele verschiedene Steckmodule, z. B. Star Battle 590,– oder Schach „Sargon II" 845,–, Panzerschlacht 220,–

Fertige Programme auf Cassetten
NEUHEIT!
z. B. Haushaltskontoführung 990,–

Datasette
programm-steuerbare Compactcassettenstation, Stromversorgung direkt vom Computer
statt ~~1.680,–~~ nur **1.475,–**

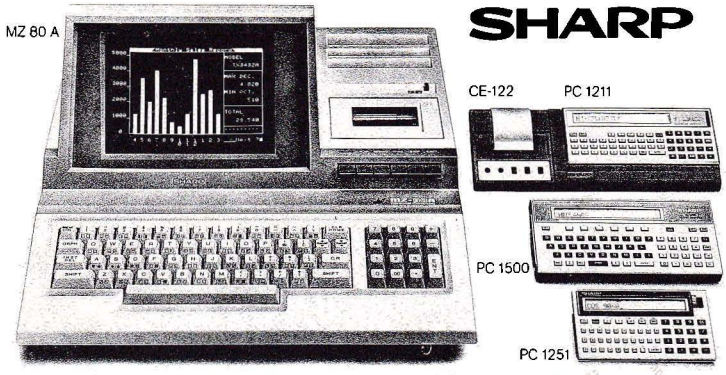
Joystick
Steuerknüppel für vielseitige Spielmöglichkeiten
statt ~~440,–~~ nur **335,–**

SHARP

MZ 80 A 19.990,–
der kompakte Personalcomputer mit den vielseitigen Anwendungsmöglichkeiten im kommerziellen und privaten Bereich. 4 Kb ROM auf maximal 48 Kb RAM erweiterbar (im Gerät). 23-cm-Bildschirm, eingebauter Cassettenrecorder. Option: Matrixdrucker, Floppy Disk.

PC 1251 NEUHEIT!
4,2 Kb RAM, 24 Kb ROM, völlig ortsungebundener Computer (mit CE-125-Drucker, Cassettenrecorder, auch mit Massenspeicher und Drucker), BASIC, 24stellige Punkt-Matrix-Anzeige, 80-Zeichen-Puffer, Permanentspeicher (Programme und Daten auch im abgeschalteten Zustand erhalten), 18 reservierbare Tasten, Programme gegen Auslisten schützbar.
Superpreis nur **2.990,–**

Drucker- und Mikro-Cassetten-Einheit
macht aus dem PC 1251 ein komplettes Computersystem, Netz- oder Batteriebetrieb
NEUHEIT! NUR 3.990,–



PC 1500 nur 5.690,–
3,5 Kb RAM erweiterbar auf 32 Kb, 16 Kb ROM, der Standard-Taschen-Computer mit zahlreicher Software, BASIC, 26stellige Punktmatrix-Anzeige (80-Zeichen-Puffer), Permanent-Speicher, 18 reservierbare Tasten, eingebaute Uhr mit Alarm, Schreibmaschinen-Tastatur mit 10er Block und Sonderzeichen.

CE-150 Printer/Plotter
ein echter Farbgraphik-Drucker und Plotter, nur 4.980,–

CE-159 8 Kb RAM-Speichererweiterung
durch eingebaute Batterie-Permanentspeicher nur 2.990,–

PC 1211
1,4 Kb RAM, BASIC, 24stellige Punktmatrix-Anzeige (90-Stellen-Puffer), Permanentspeicher, 18 reservierbare Tasten
nur **3.490,–**

CE-122 Cassetteninterface und Drucker für PC 1211 2.390,–

Sinclair ZX 81
1 Kb RAM erweiterbar auf 64 Kb, BASIC, Antennenanschluß an jeden Fernseher, direkter Anschluß von Cassettenrecorder und Thermodrucker, Folientastatur mit Mehrfachbelegung, Graphik.
statt ~~2.990,–~~ nur **2.490,–**

Thermodrucker nur 2.990,–
Doppelte Buchhaltung für kleinere Betriebe, das preisgünstigste Buchhaltungssystem in Österreich, KEIN COMPUTERWISSEN notwendig – das Programm führt den Benutzer.
NEUHEIT! nur 14.990,–

Spectrum 48 K NEUHEIT!
40 Kb RAM, 8 Kb ROM, BASIC, Antennenanschluß, direkter Anschluß an Thermodrucker und Cassettenrecorder, Drucktasten, Farbe, Tongenerator, hochauflösende Graphik, Ausgang für eine Minifloppy vorhanden (Minifloppy in Vorbereitung), alle ZX-81-Programme über Spezial-Conversionscassette (Missing Link) verwendbar
nur **6.990,–**

Spectrum 16 K NEUHEIT!
wie Spectrum 48 K, aber nur 8 Kb RAM + 8 Kb ROM 4.990,–

Missing Link Conversionscassette für ZX-81-Programme 990,–

HP 41 CV nur 6.650,–
der transportable, technische Computer mit der großten Software-Bibliothek und Peripherie, 2,2 Kb RAM, 23stellige alphanumerische Anzeige.

HP 41 C
wie HP 41 CV, aber nur 227 b RAM **4.690,–**

HP 75 C NEUHEIT!
16 Kb RAM erweiterbar auf 24 Kb/48 Kb ROM, eingebautes Zeilendisplay, 32 Zeichen (96 Zeichen im Puffer), Video- und Antennenanschluß an jeden Fernseher, folgende Merkmale sind eingebaut: Kartenleser (jede Karte 3,5 Kb), HP-IL-Interface (besonders interessant für Besitzer von Hp 41 und Peripherie) **24.190,–**

Vergleichspreise beziehen sich auf die Preise vor dem 1. 2. 1983

T.I. 99/4 A statt 7.990,– nur 6.990,–
16 Kb RAM erweiterbar auf 48 Kb/26 Kb ROM, BASIC, LOGO, UCSD-Pascal und Maschinensprache, 16-Bit-Processor, Video- und Antennenanschluß an jeden Fernseher, Farben, Ton, direkter Anschluß eines Cassettenrecorders, über Peripheriebox Benutzung von: Diskettenstationen, RS 232, Pascal, Drucker, Sprachsynthesizer, Steckmodul-Port.

Peripherie-Box statt ~~5.990,–~~ nur 4.990,–

32 Kb RAM Speichererweiterung
statt 7.990,– nur 4.990,–

Matrix-Drucker
80 Zeichen pro Zeile, 30 Zeichen/s, 154 Zeichen mit Graphic-Zeichen 14.990,–

Extended BASIC
mittels Steckmodul zusätzliche Befehle und hochauflösende Graphik

Spiele in Modulform ab 699,–

Casio PB-100
das „Einsteig-Gerät" für den Computer-Neuling, ca. 800 Byte RAM, 544 Schritte, 26 Speicher, Punkt-Matrix-Anzeige
nur **1.990,–**

Casio FX 702 P
der bewährte, preiswerte BASIC-Computer mit alphanumerischer Punktmatrix-Anzeige. 1.680 Schritte, 26 Speicher, im Lieferumfang Einführungskurs BASIC, ausbaufähig (Drucker, Cassettenstation)
nur **2.990,–**

Drucker FP-10 1.990,–
Interface FA 2 998,–

Herlango hat eine große Anzahl weiterer Geräte im Sortiment. Bei näheren Auskünften wenden Sie sich an einen Spezialisten in einer unserer 37 Filialen.

125 Jahre herlango an der Spitze

Ausführliche Demonstration des TI 99/4A im Herlango-Center, Mariahilfer Str. 51
Freitag, 8. 4. und 29. 4., 8 - 18 Uhr, Samstag, 9. 4. und 30. 4., 9 - 12 Uhr