Please do not upload this copyright pdf document to any other website. Breach of copyright may result in a criminal conviction.

This Acrobat document was generated by me, Colin Hinson, from a document held by me. I requested permission to publish this from Texas Instruments (twice) but received no reply. It is presented here (for free) and this pdf version of the document is my copyright in much the same way as a photograph would be. If you believe the document to be under other copyright, please contact me.

The document should have been downloaded from my website https://blunham.com/Radar, or any mirror site named on that site. If you downloaded it from elsewhere, please let me know (particularly if you were charged for it). You can contact me via my Genuki email page: https://www.genuki.org.uk/big/eng/YKS/various?recipient=colin

You may not copy the file for onward transmission of the data nor attempt to make monetary gain by the use of these files. If you want someone else to have a copy of the file, point them at the website. (https://blunham.com/Radar). **Please do not point them at the file itself as it may move or the site may be updated.**

It should be noted that most of the pages are identifiable as having been processed by me.

I put a lot of time into producing these files which is why you are met with this page when you open the file.

In order to generate this file, I need to scan the pages, split the double pages and remove any edge marks such as punch holes, clean up the pages, set the relevant pages to be all the same size and alignment. I then run Omnipage (OCR) to generate the searchable text and then generate the pdf file.

Hopefully after all that, I end up with a presentable file. If you find missing pages, pages in the wrong order, anything else wrong with the file or simply want to make a comment, please drop me a line (see above).

It is my hope that you find the file of use to you personally – I know that I would have liked to have found some of these files years ago – they would have saved me a lot of time !

Colin Hinson
In the village of Blunham, Bedfordshire.

Texas Instruments
TI-99/4 Home Computer

Diskette Software

Model PHD 5004

Programming Aids I

Five useful routines that extend the capability and flexibility of TI BASIC:

- **CATALOG — Catalogs diskette contents.**
- **I/O-SUBS — Gives you complete control of the display screen through three subroutines, DISPLAY-AT, ACCEPT-AT, and SCREEN-PRINT.**
- **LOWERCASE — Displays text in upper-case and lower-case letters.**
- **2ND-ASCII — Highlights specified text messages.**
- **CHARDEF — Makes character definition quick and easy.**

Designed for use with the TI-99/4 Home Computer and the TI Disk Memory System (TI Disk Drive Controller and TI Disk Memory Drive — sold separately).

As this manual was designed for the U.S. market, the warranty conditions described herein are not applicable in the U.K. The only valid Guarantee Conditions are those set forth in the "Users Reference Guide" accompanying the Home Computer.

CATALOG Description

Author: Texas Instruments

Language: TI-99/4 BASIC

Lines: 127

Hardware: TI-99/4 Home Computer
Disk Drive Controller and Disk Memory Drive
Solid State Thermal Printer (optional)

Media: Diskette

This program displays or prints a list of the programs and files on a disk. The advantage of this program is that the Disk Manager Command Module does not need to be plugged in. The display screens for this program are patterned after the Disk Manager module and duplicate its actions almost exactly.

STEP 1: Insert the program diskette into disk drive 1, select TI BASIC, and load the program by typing

OLD DSK1.CATALOG

When the cursor reappears, type RUN.

STEP 2: If you are using only one disk drive, remove the program diskette from drive 1, and insert the diskette you want to catalog. If you are using more than one disk drive, place the diskette you want to catalog in drive 2 or 3.

STEP 3: Select the appropriate disk drive number by pressing 1, 2, or 3. The diskette name is then displayed below this prompt.

STEP 4: Select the device on which you want the listing printed. If you want the choice displayed under the cursor, press ENTER. Choice 2 (Thermal Printer) and 3 (RS232 Interface) assume default parameters. If you select OTHER, you are prompted for a device-filename. See the appropriate user's manual for details.

STEP 5: If you wish to see a catalog for another disk, press any key and insert the disk. However, if you do not wish to see any additional catalogs, press SHIFT C (CLEAR) or SHIFT Q (QUIT) to stop the program.

Author: Texas Instruments

Language: TI-99/4 BASIC

Lines: 104

Hardware: TI-99/4 Computer
Disk Controller and Drive or Cassette Tape Recorder
Thermal Printer (Optional)
RS232 Interface and compatible printer (Optional)

Media: Cassette and Diskette

The I/O SUBROUTINES are three routines designed to give the BASIC programmer complete control of the display screen. Each subroutine is independent from the others and can be separately added to your program.

DISPLAY-AT allows your program to display strings at specified screen locations. This subroutine emulates the DISPLAY-AT subroutine of Extended BASIC and eliminates the scrolling action of the PRINT statement.

ACCEPT-AT allows your program to accept data from a specified location on the display screen. This routine emulates the ACCEPT-AT statement of Extended BASIC and eliminates the scrolling action of the INPUT statement. The location on the screen, size of the input field, verification of numeric data, and optional blanking of the input field are featured. The routine allows you to move the cursor in the input fields with the right and left arrow keys. The ERASE, DELETE, REDO, BACK, PROC'D and BEGIN functions are also active.

SCREEN-PRINT outputs a copy of the display, including user-defined characters, to the TI Thermal Printer.

There are two ways to use these subroutines. The first method requires that you plan your program before typing it in. If your program requires these subroutines, load them into the computer before typing in the remainder of your program. This method is easiest. The second method is used when you decide these routines are required after you have entered your program. In this case, you must manually type in the required routine. A program listing is provided for your convenience.

Before calling one of the I/O SUBROUTINES, your main program must assign values to specific variables. In the case, ACCEPT-AT, the subroutine returns values in predefined variables. Throughout the I/O SUBROUTINES, all variables used internally by the subroutines begin with the letter 'Z' followed by a digit. It is recommended that your main program avoid naming variables in the range Z1 through Z9 and Z0(32).

DISPLAY-AT (Lines 10060 through 10150)

Before calling the DISPLAY-AT subroutine, your main program must assign values to these variables:

MSG$: The message string to be displayed.

ROW: MSG$ is displayed on this row. Legal values range from 1 through 24.

COL: The first character of MSG$ is displayed on this column. Legal values range from 1 through 32.

Once the variables are defined, the message string, MSG$, is displayed by executing the statement, GOSUB 10060. The remark statements at the beginning of the routine can be deleted.

ACCEPT-AT (lines 10230 through 10840)

Before calling the ACCEPT-AT subroutine, your main program must assign values to these variables:

ROW: Refers to the row where the input data will be located. Acceptable values for ROW range from 1 through 24.

COL: Refers to the first column where the input data will be located. COL ranges from 1 through 32.

CKNUM: Signals the subroutine to check for numeric data. If CKNUM=1, then only numeric data is accepted. (Exponential notation is not permitted by this routine.) If CKNUM is not equal to 1, then alphanumeric data is accepted.

FLEN: Refers to the size of the input field. The absolute value of FLEN gives the number of characters that can be accepted in the input field. If FLEN is positive, then the entire input field is blanked prior to accepting data. If FLEN is negative, then the input field is not changed prior to accepting data. The input field must not extend beyond the righthand screen boundary.

The ACCEPT-AT subroutine returns values in the following variables:

CNTR: Indicates the last key pressed by the user. The table below lists the five ways to terminate ACCEPT-AT and the corresponding values assigned to CNTR:

Value of CNTR	Key Pressed	Function
1	ENTER	
2	SHIFT R	REDO
3	SHIFT Z	BACK
4	SHIFT V	PROC'D
5	SHIFT W	BEGIN

RTN$: Contains the data string accepted.

RTN: Contains the numeric value of RTN$ if CKNUM=1. If CKNUM is not equal to 1, then RTN=0.

The ACCEPT-AT subroutine has built-in editing features. The left and right arrow keys (SHIFT S and SHIFT D) move the cursor back and forth in the input field. Also, DELETE and ERASE functions (SHIFT F and SHIFT T) blank the input field. The remark statements at the beginning of the routine can be deleted.

SCREEN-PRINT (lines 10920 through 11040)

Two entry options are available in the SCREEN-PRINT subroutine. If your main program calls the SCREEN-PRINT routine by executing the line GOSUB 10920, the display is printed by the TI Thermal Printer. User-defined characters are also printed. If your main program executes the SCREEN-PRINT routine with the line GOSUB 10940, a copy of the display is output to the device specified in the string OUTDEVICE$. In this case, you must assign the appropriate device-name to OUTDEVICE$ before calling SCREEN-PRINT. (See the appropriate user's manual for details.) No user-defined characters are printed.

Note that the file number used in this routine is 255. Do not use this file number in your main program.

STEP 1: Plan your application program.

STEP 2: Select TI BASIC. Load the subroutines from the diskette by typing

```
OLD DSK1.I/O-SUBS
```

Load the subroutines from cassette tape by typing

```
OLD CS1
```

STEP 3: Type in your main program.
Before calling DISPLAY-AT, assign values to these variables: ROW, COL, MSG$

Before calling ACCEPT-AT, assign values to these variables: ROW, COL, CKNUM, FLEN.

If printing the screen to a device other than the TI Thermal Printer, assign a value to OUTDEVICE$. Otherwise, no asssignment needs to be made.

STEP 4: To execute DISPLAY-AT, type GOSUB 10060.
To execute ACCEPT-AT, type GOSUB 10230.
To execute SCREEN-PRINT, type GOSUB 10920 or GOSUB 10940. (See the Background Section for the difference in these cases.)

The ACCEPT-AT routine returns values in these variables: CNTR, RTN$, RTN.
The other routines do not return any variables.

STEP 5: When your program is finished, delete any unused routines. Remark statements may also be deleted without any effect on the routines.

This is a simple program that uses the I/O SUBROUTINES.

```
100 REM -TEST I/OSUBS
110 ROW=5
120 COL=17
130 INPUT "NUMERIC (1=YES 2=NO)?":CKNUM
140 FLEN=10
150 MSG$="INPUT HERE: "
160 REM -CALL DISPLAY AT
170 GOSUB 10060
180 REM -CALL ACCEPT AT
190 GOSUB 10230
200 CALL CLEAR
210 GOTO 130
 .
 .
 .
10000 REM  I/O-SUBS
 .
 .
 .
```

```
10000 REM  ******************
10010 REM  *  D I S P L A Y *
10020 REM  *                *
10030 REM  *       A T      *
10040 REM  ******************
10050 REM
10060 Z2=ROW
10070 Z3=COL-1
10080 FOR Z1=1 TO LEN(MSG$)
10090 Z3=Z3+1
10100 IF Z3<=32 THEN 10130
10110 Z2=Z2+1
10120 Z3=1
10130 CALL HCHAR(Z2,Z3,ASC(SEG$(MSG$,Z1,1)))
10140 NEXT Z1
10150 RETURN
10160 REM
10170 REM  ******************
10180 REM  *   A C C E P T  *
10190 REM  *                *
10200 REM  *       A T      *
10210 REM  ******************
10220 REM
10230 DIM Z0(32)
10240 Z0$=CHR$(13)&CHR$(6)&CHR$(15)&CHR$(12)&CHR$(14)&CHR$
      (8)&CHR$(9)&CHR$(3)&CHR$(7)
10250 IF FLEN<0 THEN 10270
10260 CALL HCHAR(ROW,COL,ASC("_"),ABS(FLEN))
10270 FOR Z1=1 TO ABS(FLEN)+1
10280 CALL GCHAR(ROW,COL+Z1-1,Z0(Z1))
10290 NEXT Z1
10300 Z7=0
10310 Z1=1
10320 CALL HCHAR(ROW,COL+Z1-1,30)
10330 CALL KEY(0,Z3,Z4)
10340 IF Z4=0 THEN 10330
10350 CNTR=POS(Z0$,CHR$(Z3),1)
10360 ON CNTR+1 GOTO 10500,10690,10690,10690,10690,10690,
      10390,10390,10460,10460
10370 REM
10380 REM  (* ARROW KEYS *)
10390 CALL HCHAR(ROW,COL+Z1-1,Z0(Z1))
10400 IF Z3=9 THEN 10630
10410 IF Z1=1 THEN 10320
```

```
10420 Z1=Z1-1
10430 GOTO 10320
10440 REM
10450 REM  (* ERASE *)
10460 CALL HCHAR(ROW,COL+Z1-1,Z0(Z1))
10470 GOTO 10260
10480 REM
10490 REM  (* NUMBER? *)
10500 IF CKNUM <> 1 THEN 10580
10510 IF (Z3=45)+(Z1=1)=-2 THEN 10580
10520 IF (Z7=0)+(Z3=46)+(ABS(FLEN)>1)<>-3 THEN 10550
10530 Z7=1
10540 GOTO 10580
10550 IF (Z3<48)+(Z3>57)<=-1 THEN 10330
10560 REM
10570 REM  (* LEGAL NOFUNC *)
10580 IF Z0(Z1)<>46 THEN 10600
10590 Z7=0
10600 IF Z1=ABS(FLEN)+1 THEN 10630
10610 Z0(Z1)=Z3
10620 CALL HCHAR(ROW,COL+Z1-1,Z3)
10630 Z1=Z1+1
10640 IF Z1<=ABS(FLEN)+1 THEN 10320
10650 Z1=Z1-1
10660 GOTO 10320
10670 REM
10680 REM  (* CONTROL KEYS *)
10690 CALL HCHAR(ROW,COL+Z1-1,Z0(Z1))
10700 RTN$=""
10710 RTN=0
10720 FOR Z2=1 TO ABS(FLEN)
10730 IF Z0(Z2)<>95 THEN 10760
10740 NEXT Z2
10750 RTN$="0"
10760 FOR Z3=Z2 TO ABS(FLEN)
10770 IF Z0(Z3)=95 THEN 10800
10780 RTN$=RTN$&CHR$(Z0(Z3))
10790 NEXT Z3
10800 RTNL=LEN(RTN$)
10810 IF (RTN$="")+(CKNUM=1)=-2 THEN 10260
10820 IF CKNUM<>1 THEN 10850
10830 IF (RTN$=".")+(RTN$="-.")<=-1 THEN 10320
10840 RTN=VAL(RTN$)
10850 RETURN
```

```
10860 REM
10870 REM  ******************
10880 REM  *   S C R E E N  *
10890 REM  *               *
10900 REM  *    P R I N T  *
10910 REM  ******************
10920 REM
10930 OPEN #OUTFILE:"TP.U.S",OUTPUT
10940 GOTO 10960
10950 OPEN #OUTFILE:OUTDEVICE$,OUTPUT
10960 FOR Z1=1 TO 24
10970 Z1$=""
10980 FOR Z2=1 TO 32
10990 CALL GCHAR(Z1,Z2,Z3)
11000 Z1$=Z1$&CHR$(Z3)
11010 NEXT Z2
11020 PRINT #OUTFILE:Z1$
11030 NEXT Z1
11040 CLOSE #OUTFILE
11050 RETURN
```

Author: Texas Instruments

Language: TI-99/4 BASIC

Lines: 68

Hardware: TI-99/4 Computer
Disk Controller and Drive or Cassette Recorder

Media: Cassette and Diskette

The two LOWERCASE LETTER subroutines let you display text in both uppercase and lowercase characters. The first subroutine defines a set of lowercase letters. The second displays character strings at specified screen locations using both uppercase and lowercase letters. All letters are displayed in lowercase unless preceded by a circumflex (^). The preceding circumflex is not printed.

Use GOSUB 13000 to assign lowercase character codes 96 through 122 to the corresponding ASCII codes. Execute this statement before attempting to display lowercase characters. Before calling the DISPLAY-AT routine, assign values to MSG$, ROW, and COL. MSG$ is the string or message to be displayed. If it is longer than one line, output is automatically continued to the next line. Any alphabetic character preceded by a circumflex (^) is capitalized. All other alphabetic characters appear in lowercase. ROW refers to the line on which MSG$ is printed. COL is the column in which the first character of MSG$ is placed.

All temporary variables used in LOWERCASE LETTERS begin with the letter Z and end with a single digit. Avoid using these variables in your main program. Remark statements can be deleted without affecting performance. DATA statements can also be combined to save program space if necessary.

STEP 1: Select TI BASIC. Load the subroutines from the diskette by typing
OLD DSK1.LOWERCASE
Or load the subroutines from the cassette by typing
OLD CS1

STEP 2: Type in your program.

STEP 3: In the beginning of your program use GOSUB 13100 to define the lowercase character set.

STEP 4: Later within your program, whenever you wish to display a character string, assign values to the variables MSG$, ROW, and COL. Precede any letter that you wish to capitalize with a circumflex (^).
Example: MSG$="^DEAR^JOHN,"

STEP 5: Use the line GOSUB 13500 to display your message.

This is a simple program that uses the LOWERCASE LETTERS subroutine.

```
110 REM-SET UP LOWERCASE CHARACTERS
120 GOSUB 13100
130 CALL CLEAR
140 REM-DISPLAY MESSAGE
150 ROW=3
160 COL=2
170 MSG$="^DEAR ^JOHN,"
180 GOSUB 13500
190 STOP
13000 REM LOWERCASE ROUTINES
  .
  .
  .
  .
13660 RETURN
```

OUTPUT: Dear John,

```
13000 REM
13010 REM
13020 REM
13030 REM
13040 REM  *****************
13050 REM  *  LOWER CASE   *
13060 REM  *    ROUTINE    *
13070 REM  *****************
13080 REM
13090 REM  *CHARACTER DEFINITIONS*
13100 DATA 0000003848483400
13110 DATA 0040407048483000
13120 DATA 0000003840403800
13130 DATA 0008083848483400
13140 DATA 000018243C201800
13150 DATA 0018282038202000
13160 DATA 0000384848380830
13170 DATA 0020203824242400
13180 DATA 0010003010103800
13190 DATA 0010003010105020
13200 DATA 0040405060504800
13210 DATA 0030101010103800
13220 DATA 0000006C54545400
13230 DATA 0000007848484800
13240 DATA 0000003048483000
13250 DATA 0000705848704040
13260 DATA 0000384848380808
13270 DATA 0000002830202000
13280 DATA 0000182038083000
13290 DATA 0010103810101800
13300 DATA 0000004848483400
13310 DATA 0000002424281000
13320 DATA 0000004454542800
13330 DATA 0000002810282800
13340 DATA 0000004848301020
13350 DATA 0000003810203800
13360 REM   **READ DATA AND DEFINE CHARACTERS**
13370 RESTORE 13100
13380 FOR Z0=97 TO 122
13390 READ Z$
13400 CALL CHAR(Z0,Z$)
13410 NEXT Z0
13420 RETURN
13430 REM
```

```
13440 REM  *****************
13450 REM  *  DISPLAY-AT   *
13460 REM  *  UPPER AND    *
13470 REM  *  LOWER CASE   *
13480 REM  *****************
13490 REM
13500 Z2=COL-1
13510 Z4=ROW
13520 FOR Z1=1 TO LEN(MSG$)
13530 Z2=Z2+1
13540 Z3=ASC(SEG$(MSG$,Z1,1))
13550 IF Z3 < >94 THEN 13600
13560 Z1=Z1+1
13570 IF Z1 > LEN(MSG$)THEN 13670
13580 Z3=ASC(SEG$(MSG$,Z1,1))
13590 GOTO 13620
13600 IF (Z3 < 65)+(Z3<90) < =-1 THEN 13620
13610 Z3=Z3+32
13620 CALL HCHAR(Z4,Z2,Z3)
13630 IF Z2 < 32 THEN 13660
13640 Z2=0
13650 Z4=Z4+1
13660 NEXT Z1
13670 RETURN
```

Author: Texas Instruments

Language: TI-99/4 BASIC

Lines: 126

Hardware: TI-99/4 Computer
Disk Controller and Drive or Cassette Recorder

Media: Diskette and Cassette

The two 2ND ASCII SET subroutines are designed to display text strings in more than one combination of colors. These routines are useful for highlighting special messages on the screen. The first routine sets up a second set of characters starting at character code 96 and assigns a foreground (light blue) and background (white) color. The second routine displays a message using either the regular ASCII set or the second set. Message strings preceded by an asterisk (*) are displayed in light blue text against a white background.

2ND ASCII SET — Background

The second ASCII set is assigned to character codes 96 through 159. If the output is longer than one line, it is automatically continued in column one of the following line.

Any message preceded by an asterisk (*) appears in the alternate colors. This initial asterisk is not displayed. All other asterisks within the message, however, are displayed.

The colors of the second ASCII set can be changed by changing the DATA statement in line 10760. The first number listed represents the foreground color, while the second indicates the background color.

Before calling the DISPLAY-AT subroutine in line 10960, values must be assigned to the following variables:

MSG$: The string to be displayed is stored here.

ROW : MSG$ is displayed in this row. Legal values range from 1 through 24.

COL : The first character of MSG$ is displayed here. Legal values range from 1 through 32.

STEP 1: Select TI BASIC. Load the subroutines from the diskette by typing

```
OLD DSK1.2ND-ASCII
```

Or load the subroutines from the cassette by typing

```
OLD CS1
```

STEP 2: Type in your program.

STEP 3: In the beginning of your program use GOSUB 10110 to set up the second character set.

STEP 4: Assign values to the variables MSG$, ROW, and COL. Precede any message that you wish to appear in different colors by an asterisk. Example: MSG$="*DEAR JOHN"

STEP 5: Use the line GOSUB 10960 to display your message.

This is a simple program that uses the 2ND ASCII SET subroutines:

```
100 REM -INITIALIZE 2ND ASCII SET
110 GOSUB 10110
120 CALL CLEAR
130 ROW=3
140 COL=2
150 MSG$="*DEAR JOHN"
160 GOSUB 10960
170 STOP
10000 REM-DISPLAY 2ND ASCII SET -
  .
  .
  .
11250 RETURN
```

OUTPUT: DEAR JOHN

The letters appear in blue on a white background.

```
10000 REM
10010 REM
10020 REM
10030 REM
10040 REM  ***************
10050 REM  *  2ND ASCII   *
10060 REM  *   ROUTINE    *
10070 REM  ***************
10080 REM
10090 REM  **CHARACTER DEFINITIONS**
10100 REM
10110 DATA 0000000000000000
10120 DATA 0010101010001000
10130 DATA 0028280000000000
10140 DATA 00287C28287C2800
10150 DATA 0038543018543800
10160 DATA 00444C1830644400
10170 DATA 0020502054483400
10180 DATA 0008102000000000
10190 DATA 0008101010100800
10200 DATA 0020101010102000
10210 DATA 0044287C28440000
10220 DATA 0010107C10100000
10230 DATA 0000000030102000
10240 DATA 0000007C00000000
10250 DATA 0000000000303000
10260 DATA 0004081020408000
10270 DATA 0038444444443800
10280 DATA 0010301010103800
10290 DATA 0038440810207C00
10300 DATA 0038441804443800
10310 DATA 00081828487C0800
10320 DATA 0078407804443800
10330 DATA 0038407844443800
10340 DATA 007C040810202000
10350 DATA 0038443844443800
10360 DATA 003844443C047800
10370 DATA 0030300030300000
10380 DATA 0030300030102000
10390 DATA 0000040810080400
10400 DATA 0000007C007C0000
10410 DATA 0000402010204000
10420 DATA 0038440810001000
10430 DATA 0038445458403C00
```

```
10440 DATA 003844447C444400
10450 DATA 0078447844447800
10460 DATA 0038444040443800
10470 DATA 0078444444447800
10480 DATA 007C407840407C00
10490 DATA 007C407840404000
10500 DATA 003844404C443800
10510 DATA 0044447C44444400
10520 DATA 0038101010103800
10530 DATA 0004040404443800
10540 DATA 0048506050484400
10550 DATA 0040404040407C00
10560 DATA 00446C5444444400
10570 DATA 0044645454C44400
10580 DATA 007C444444447C00
10590 DATA 0078444478404000
10600 DATA 00384444544C3C00
10610 DATA 0078444478484400
10620 DATA 0038443008443800
10630 DATA 007C101010101000
10640 DATA 0044444444443800
10650 DATA 0044444444281000
10660 DATA 0044444454542800
10670 DATA 0044281010284400
10680 DATA 0044442810101000
10690 DATA 007C081020407C00
10700 DATA 001C101010101C00
10710 DATA 0080402010080400
10720 DATA 0070101010107000
10730 DATA 0010284400000000
10740 DATA 000000000000FC00
10750 REM  **FOREGROUND, BACKGROUND**
10760 DATA 6,16
10770 REM
10780 REM  **SET UP CHARACTER  CODES**
10790 RESTORE 10000
10800 FOR Z0=96 TO 159
10810 READ Z$
10820 CALL CHAR(Z0,Z$)
10830 NEXT Z0
10840 REM
10850 REM  **SET UP COLOR**
10860 READ Z1,Z2
10870 FOR Z0=9 TO 16
10880 CALL COLOR(Z0,Z1,Z2)
```

```
10890 NEXT Z0
10900 RETURN
10910 REM
10920 REM  ***************
10930 REM  *  DISPLAY-AT *
10940 REM  ***************
10950 REM
10960 Z2=COL-1
10970 Z4-ROW
10980 REM
10990 REM  **FIRST OR SECOND SET?**
11000 IF MSG$=""THEN 11250
11010 IF ASC(SEG$(MSG$,1,1))=42 THEN 11170 ELSE 11050
11020 REM
11030 REM  **FIRST SET**
11040 REM
11050 FOR Z1=1 TO LEN(MSG$)
11060 Z2=Z2+1
11070 Z3=ASC(SEG$(MSG$,Z1,1))
11080 CALL HCHAR(Z4,Z2,Z3)
11090 IF Z2 < 32 THEN 11120
11100 Z2=0
11110 Z4=Z4+1
11120 NEXT Z1
11130 RETURN
11140 REM
11150 REM  **SECOND SET**
11160 REM
11170 FOR Z1=2 TO LEN(MSG$)
11180 Z2=Z2+1
11190 Z3=ASC(SEG$(MSG$,Z1,1))+64
11200 CALL HCHAR(Z4,Z2,Z3)
11210 IF Z2 < 32 THEN 11240
11220 Z2=0
11230 Z4=Z4+1
11240 NEXT Z1
```

Author: Texas Instruments

Language: TI-99/4 BASIC

Lines: 339

Hardware: TI-99/4 Computer
Disk Controller and Drive or Cassette Recorder
TI Thermal Printer (Optional)

Media: Diskette and Cassette

CHARACTER DEFINITION helps you design user-defined characters quickly and easily. On an 8x8 character grid, representing the 8 rows of 8 dots in a character, you control a cursor that turns on or off the dot below it. This 8x8 grid can then be translated into an actual character. The hexadecimal pattern identifier required by the CALL CHAR subprogram is also displayed.

You can define up to four unique characters, seeing the resulting graphics by themselves and clustered together. Any character can be placed in the work area for modification, and the character can be printed on the TI Thermal Printer.

STEP 1: Select TI BASIC. Load the program from the diskette by typing

```
OLD DSK1.CHARDEF
```

Or load the program from the cassette by typing

```
OLD CS1
```

When the cursor reappears, type RUN.

STEP 2: Each color is listed on the screen with a corresponding number. The prompts below the list ask for the foreground and the background colors to be used on the screen. If you do not have specific colors in mind, enter 2 for a black foreground and 4 for a light green background. (These are the default colors displayed when a BASIC program is running.)

STEP 3: After a short pause, a flashing cursor appears in the upper-left corner of the 8x8 grid, indicating that the program is ready to respond to your commands.

STEP 4: A list of available functions appears below the grid. To activate a function, press the appropriate letter or number listed on the screen. A description of available functions follows:

0 - Turns the character-dot covered by the cursor OFF. This corresponds to selecting the background color. If the MOVE function is on, the cursor automatically advances to the next character position. If held down, this function automatically repeats until the key is no longer pressed. At the edge of the work character the cursor moves to the next line. The cursor does not advance if the MOVE function is off.

1 - Turns ON the dot covered by the cursor. This corresponds to selecting the foreground color.

C - Erases all the character-dots in the work area grid by turning them off.

H - Returns the cursor to the upper left corner of the work character (the "home" position).

I - Inverts the character in the grid. The ON dots are turned off, and the OFF dots are turned on.

N - Selects the "normal" mode for this program, allowing definition of a character in an 8x8 dot matrix. (See the T function).

P - Prints the four pattern-identifier character strings and the four user-defined characters on the TI Thermal Printer.

T - Puts the program in a "TP MODE" and is used when the defined characters are to be printed on the TI Thermal Printer. All dots which cannot be printed are filled with asterisks (*). The cursor cannot be moved outside the printable region.

X (DOWN ARROW)- Moves the cursor down. If held down, it automatically repeats until the key is released. If the MOVE function is on, the cursor moves to the previous line when the left edge of the work character is reached. If the MOVE function is off, the cursor stops at the edge.

D (RIGHT ARROW)- Moves the cursor to the right.

E (UP ARROW)- Moves the cursor up.

K - Copies the work character to one of the four displayed characters so that you can see the character in its actual size.

M - Turns the MOVE function ON or OFF each time it is pressed. See O and S for an explanation of how to use this function.

R - Recalls one of the 4 displayed characters to the grid for modification.

S - (LEFT ARROW) Moves the cursor one position to the left.

The Character Definition display screen is represented below. In this example, only the first character has been defined. The character in the work area corresponds to the hexadecimal code in line 1. All commands are listed in the lower region of the screen.

```
CALL CHAR(X,"000028106C102800")

 TI-99/4 CHARACTER DEFINITION
 MOVE=OFF            MODE=NORMAL

 1 2 3 4 5 6 7 8    1=000028106C102800
1                 1  2=0000000000000000
2                 2  3=0000000000000000
3     0   0       3  4=0000000000000000
4       0         4
5  0 0    0 0     5    *                *
6       0         6
7     0   0       7   ***
8                 8   ***              12
 1 2 3 4 5 6 7 8    ***              34
     W O R K         1    2    3    4


 0=TURN DOT OFF          1=TURN DOT ON
 T=TP MODE               N=NORMAL MODE
 H=HOME CURSOR           I=INVERT
 C=CLEAR                 P=PRINT
 S=LEFT D=RIGHT          E=UP X=DOWN
 R=RECALL                K=KEEP
 M=MOVE ON/OFF
```

If you want to use the defined character in a program, include the hexadecimal code in a CALL CHAR statement. For example,

```
120 CALL CHAR(97,"000028106C102800")
```