FORTH CHARACTER CREATOR

(requires Monitor 2
or later version)

The programmable character RAM option permits the user to define and use an alternative character set to that stored in the character ROM. This FORTH program allows the user to examine the shapeof any programmable graphics character and to modify and store it.

When the machine is powered up, the graphics RAM will contain random bits. When in FORTH

PC - turns these graphics on
PCX - returns to ROM characters

There are 128 characters (ASCII numbers 0 to 127). Each CHARACTER CREATOR program will display a grid 8 wide, 16 high which you can fill with white blocks or blanks.

USING THE PROGRAM

1) The program listing uses HEX numbers. Type HEX and then enter the program from the listing and save it on cassette tape. If you enter it as text (with FORTH 1.2) the listing will occupy about 14 blocks beginning at B5ØØ. The compiled version should occupy memory from BØØØ to about B45Ø.

2) After the compiled version is complete enter an ASCII number (ØØ to 7F in hex)
(0 to 127 in decimal)
then space, then CHAR

Example - 41 CHAR (for A character if in Hex)

3) The screen will display the ASCII number, the ROM form of the character in inverse video and an 8 by 16 grid of [ ] brackets showing the current graphics shape of that number.

4) If you wish to erase the present shape, press CTRL and DELETE simultaneously, and the grid pattern will clear.

5) The cursor will appear at the top left of the grid. Use these keys to move the cursor

```
             E - up
S - left                D - right
             X - down
```

When the cursor is placed at a [ ] you wish to fill, press the > key. To clear the space, press the < key (do not use shift). By moving the cursor to each bracket block you want filled, you can create your own pattern.

6) When the character is defined, press RETURN. WAIT will briefly appear as the new character is stored into graphics RAM.

7) Repeat from Step 2 for further characters.

8) These new characters will remain in graphics RAM as long as power is on. To save your characters from later use, use the CHAR SAVE/LOAD program.

NOTES

Define 20 CHAR (in hex) or 32 CHAR (in decimal) as blank to correspond with a space in the character ROM. Define any other characters(s) to see how they look in normal size, type PC to change to graphics and then type the key for the new character. Return to ROM characters using PCX. To display characters from 00 to IF (hex) or 0 to 31 (decimal)

Define these words

```
  :  RW     1   BDD9   C!   ;
  :  RWX    0   BDD9   C!   ;
```

Then RW 3 EMIT RWX will display ascii character 3.

PROGRAM COMMENTS ON DEFINITIONS

GRID - uses BRKT 16 times to form an 8 by 16 pattern

The following words define keyboard functions when KEY is used in INP definition

DN - move down

UP - move up

[ ] - print [ ] and then backspace

SP - print inverse space then backspace

RHT - move right to next [ ] after checking for right border

LFT - move left to next [ ] after checking for left border

SH - shows the present character, in raw mode and inverse video

CH? - prints the character number, uses SH and positions cursor onto grid

CL - CTRL DELETE clears grid

INP - input routine from keyboard until a RETURN is pressed

RCH - reads single character from graphics RAM

WCH - writes single character to graphics RAM

FORTH CHARACTER CREATOR (all numbers in HEX)

```
: BRKT ." [][][][][][][][]" ;

: GRID 11 1 DO CR I 0A .R 3 SPACES BRKT LOOP ;

: DEMIT DUP EMIT EMIT ;

: DN DUP 58 = IF 18 EMIT THEN ;

: UP DUP 45 = IF  5 EMIT THEN ;

: [] DUP 2C = IF ." []" 13 DEMIT THEN ;

: SP DUP 2E = IF A0 DEMIT 13 DEMIT THEN ;

: RHT DUP 44 = IF BDF9 C@ 1B < IF 4 DEMIT THEN THEN ;

: LFT DUP 53 = IF BDF9 C@ 0D > IF 13 DEMIT THEN THEN ;

: SH IV SPACE 1 BDD9 C! EMIT 0  BDD9 C! SPACE IVX ;

  0 VARIABLE CH

: CH? CH C@ DUP 0 0 CPOS . ." CHAR" CR CR SH 0D 0 CPOS ;

  B480 VARIABLE M

: CL DUP 1F = IF CLS GRID CH? THEN ;

: INP BEGIN KEY DN UP LFT RHT [] SP CL 0D = IF 1 ELSE 0 THEN UNTIL ;

  0 VARIABLE B  80 VARIABLE W

: RCH 19 EMIT CH C@ EMIT 10 0 DO BDA0 I + C@ M @ I + C! LOOP ;

: WCH 1B EMIT CH C@ EMIT 10 0 DO M @ I + C@ EMIT LOOP ;

: W/ W C@ 2 / W C! ;

  0 VARIABLE Q

: B+ 20 = IF B C@ W C@ + B C! THEN W/ ;

: BB CPOS IV ."   " IVX ;

: WCONV 10 0 DO 0 B C! 80 W C! 10 0 DO BE0E I + J 20 * + C@ B+ 2 +LOOP B C@ M @ I + C! LOOP ;

: RCONV 10 0 DO 80 W C! BDA0 I + C@ Q C! 10 0 DO Q C@ W C@ - DUP -1 > IF 0D I + J BB Q C! ELSE DROP THEN W/ 2 +LOOP LOOP ;

: CHAR CH C! CLS       GRID RCH RCONV CH? INP 0 6 CPOS ." WAIT" 0 6 CPOS WCONV WCH ."     " ;
```

W/ - divides W by 2

B+ - adds W to current value of B if a grid position is filled

BB - prints a white block

WCONV - changes the screen grid pattern to 16 bytes

RCONV - changes the 16 bytes read from the graphics RAM into a screen pattern

CHAR - forms the grid, and current pattern, inputs changes and writes new pattern to RAM