FOREWORD

We at PBJ would like to take this opportunity to thank you for your purchase and at the same time solicit your assistance. We would like to ask that you take a few minutes and send us your comments and suggestions, not only on the products, but specifically on the documentation. If there are areas that were not clear to you, chances are it was not clear to someone else. If you think an example or a picture might help clarify a point let us know. Remember, the only guide we have is your input. If it's negative we can change it, if it's positive at least we know we are on the right track.

The Radio Shack Color Computer is not only a fun computer, it is also a very powerful computer. Our aim is to provide high quality products that will enhance the computer's capability and the user's enjoyment.

THANK YOU
CONTENTS

Introduction ......................................................... 2
Installation and Use ........................................... 2

Video Driver ......................................................... 2
  Loading and Executing ........................................ 3
  Program Location and RAM Options ...................... 3
  Control Codes .................................................. 4
  Compatibility .................................................. 5
  Keyboard Functions .......................................... 6
  Screen Editing ................................................ 6
  Software Video Switch ....................................... 8
  Word-Pak Memory Map ....................................... 8

System Overview .................................................. 8
  Display Memory Access ...................................... 9
  Format Parameter Calculation ............................ 10
  Cursor Options ............................................... 11

Custom Character Sets ......................................... 12

Attachments:
  Character Generator Program
  Simple Basic Driver Listing
  Schematic Dwg.
  Component Layout Dwg.
INTRODUCTION

The WORD-PAK is a video board in a cartridge intended for use with the Radio Shack Color Computer. The cartridge features a high quality 80 column video display output, with full programmability of video format and cursor.

The WORD-PAK is fully compatible with any size system and can be implemented on cassette or disk based system.

The software supplied is a Video Driver program which integrates the WORD-PAK into the system and adds a set of powerful screen editing commands comparable to those available on the more expensive terminals.

INSTALLATION AND USE

To use the WORD-PAK on a cassette based system, simply insert the cartridge into the ROM expansion port (make sure the computer is off), connect the video output to a monitor (see Fig.1), and load the Video Driver program (see Loading and Executing).

A disk based system requires the use of a Y-connector to allow both the disk controller and the WORD-PAK to be connected to the computer at the same time. These Y-connectors are available from several sources and in different configurations. A "Y" ribbon cable is available from PBJ for use with the WORD-PAK. An alternate method is to use one of the compatible expansion busses available, such as the C-C BUS.

Connect the Y-connector to the computer through the ROM expansion port and then connect the WORD-PAK and the disk controller to the two available connectors. Connect the video output from the WORD-PAK to a monitor (see Fig.1), and load the Video Driver program (see Loading and Executing).

VIDEO DRIVER

The Video Driver program provided with the WORD-PAK is fully integrated into the existing Basic ROM software so that it can be used with standard Basic programs as well as other programs. The format of the display when the program is first executed defaults to 24 lines of 80 characters. This can be changed to different screen configurations through the use of control codes. The Driver also includes other control codes which allow various screen functions to be performed. These include: Erase to End of Line, Erase to End of Screen, Clear Screen, Home Cursor, and more. All of these features are controlled through the use of control code characters sent via the CHR$(n) Basic statement, machine language routines, or from the keyboard.

Another feature available through the Video Driver is full screen editing of Basic programs. Editing is accomplished simply by moving the cursor to the desired position on the screen and provides for inserting, deleting, or overtyping characters.
LOADING & EXECUTING

The Basic Driver for the WORD-PAK can be supplied in several different forms to suit the specific system configuration. For cassette based systems, the Basic Driver can be supplied in an EPROM which is installed on the WORD-PAK (Extended Basic required). This eases the installation, since it is done automatically on power-up. The Basic Driver can also be supplied on cassette if required.

For disk based systems, two options are available. The Basic Driver can be supplied either in an EPROM or on disk. Normally, the Basic Driver will be supplied on a disk for disk based systems. However, if the user has a 64K system and is using either the C-C BUS or Radio Shack’s Multipak then he/she can elect to have the Basic Driver supplied on an EPROM.

NOTE: The EPROM option is not available on the Word-PakII. However, this presents no disadvantage since the 32 column screen is available on power up.

For disk based systems where the Basic Driver was supplied on an EPROM, the following must be observed:

1. Install the WORD-PAK in slot 'O' on the C-C BUS or in slot '4' on the Multipak.

2. Install the Disk Controller in slot '1' on the C-C BUS or in slot '3' on the Multipak.

3. Connect video cable to WORD-PAK and turn power 'on'.

Two versions of the Video Driver program are provided on the disk: one is for 16/32K systems, the other is for 64K systems. Additionally, two loader programs are provided to ease the loading and execution of the program since video is not available on power up. The following are the names of the programs supplied on the disk:

<table>
<thead>
<tr>
<th>Program</th>
<th>Extension</th>
</tr>
</thead>
<tbody>
<tr>
<td>WP</td>
<td>.BIN</td>
</tr>
<tr>
<td>WP64</td>
<td>.BIN</td>
</tr>
<tr>
<td>V</td>
<td>.DAS</td>
</tr>
<tr>
<td>V1</td>
<td>.DAS</td>
</tr>
</tbody>
</table>

To get the WORD-PAK operational, simply type 'RUN "V"' for 16/32K systems or 'RUN "V1"' for 64K systems. Either WP or WP64 will be loaded and executed. (NOTE: The source listing for the Basic Driver is also provided on the disk.)

PROGRAM LOCATION & RAM OPTIONS

As mentioned above, the Video Driver program is provided in two configurations. This allows users with 64K of memory the ability to store the program at the top of the memory map. In the first configuration, the Video Driver program is located at the top of the 16K or 32K of memory. The program determines if 16 or 32K is available, and relocates itself automatically.
The second configuration is for users having 64K of RAM available not used by
other programs. In this mode, the program automatically copies the Basic ROMs
into RAM and sets up a RESET vector to re-initialize to 64K when the RESET
button is pressed. It uses RAM starting at HFOOO, regardless of the system
configuration. This is compatible with Disk systems and Tape alike. No
noticeable difference in system performance should be noticed.

CONTROL CODES

Most of the time, screen control functions will be used with the Basic
statement "PRINT CHR$(n)". When using this method, values listed under the
Dec. column (see below) are used to implement the function. For example, to
clear the screen you would used the Basic statement "PRINT CHR$(12)" or you
can use CTRL "L" (depressing the 'CLEAR' key and the 'L' key).

Most of the control functions are completed with a single character code and
can be easily implemented from the keyboard (see KEYBOARD FUNCTIONS below).
Two of the functions provided, however, require more than one code and they
are; the x-y cursor positioning, and changing the characters per line.

The following is a list of the Control Codes recognized by the Video Driver
program and the function that each performs.

Control Codes

<table>
<thead>
<tr>
<th>Hex</th>
<th>Dec.</th>
<th>Ctrl.</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>7</td>
<td>G</td>
<td>Sound Bell tone</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>H</td>
<td>Backspace Cursor on character position.</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>I</td>
<td>Advance Cursor one character position.</td>
</tr>
<tr>
<td>A</td>
<td>10</td>
<td>J</td>
<td>Move Cursor down one line (scroll if at bottom).</td>
</tr>
<tr>
<td>B</td>
<td>11</td>
<td>K</td>
<td>Initiate x-y Cursor positioning.</td>
</tr>
<tr>
<td>C</td>
<td>12</td>
<td>L</td>
<td>Clear screen.</td>
</tr>
<tr>
<td>D</td>
<td>13</td>
<td>M</td>
<td>Move Cursor to the start of the next line.</td>
</tr>
<tr>
<td>10</td>
<td>16</td>
<td>P</td>
<td>Home Cursor (top left of screen).</td>
</tr>
<tr>
<td>11</td>
<td>17</td>
<td>Q</td>
<td>Turn destructive cursor ON.</td>
</tr>
<tr>
<td>12</td>
<td>18</td>
<td>R</td>
<td>Turn destructive cursor OFF.</td>
</tr>
<tr>
<td>15</td>
<td>21</td>
<td>U</td>
<td>Erase from cursor to end of line.</td>
</tr>
<tr>
<td>16</td>
<td>22</td>
<td>V</td>
<td>Erase from cursor to end of screen.</td>
</tr>
<tr>
<td>17</td>
<td>23</td>
<td>W</td>
<td>Turn reverse character mode on.</td>
</tr>
<tr>
<td>18</td>
<td>24</td>
<td>X</td>
<td>Turn reverse character mode off.</td>
</tr>
<tr>
<td>1B</td>
<td>27</td>
<td></td>
<td>Change number of displayed characters per line.</td>
</tr>
<tr>
<td>*1D</td>
<td>29</td>
<td></td>
<td>Enable/Disable smooth scrolling.</td>
</tr>
<tr>
<td>*1F</td>
<td>31</td>
<td></td>
<td>Change scroll rate.</td>
</tr>
</tbody>
</table>

* These codes will only work on the Word-PakII since the original Word-Pak
  does not have the capability to smooth scroll text on the screen.

The scroll rate can be varied by passing a value to the Basic Driver from 0
(for very fast smooth scrolling) to 255 (for very slow smooth scrolling). The
following statement would set the scrolling speed to "20":

PRINT CHR$(31);CHR$(20)
The x-y cursor positioning function allows the cursor to be positioned at any location on the screen with a minimum of effort. This can be useful for screen mapping and information updating. This is similar to the Basic PRINT @ function, but instead of using a single number for the location, a column position and a line number are used. These values must immediately follow the x-y positioning control code. For example to position the cursor in the middle of the screen (for an 80 character screen) and print the word "HELP", you would use the following statement:

```
PRINT CHR$(11);CHR$(38);CHR$(12);"HELP"
```

This will print the word HELP on line 12 starting at column 38. Notice that a ";" must be used between each character for the command to work properly.

Another function that requires more than one character is the 'Change characters per line' function. This is implemented by first sending an "Escape" character ($1B), decimal 27, and then following with code number for the desired characters per line.

The following are the available screen sizes and the corresponding code values.

<table>
<thead>
<tr>
<th>Chars/line</th>
<th>Code Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>0</td>
</tr>
<tr>
<td>64</td>
<td>2</td>
</tr>
<tr>
<td>32</td>
<td>7</td>
</tr>
</tbody>
</table>

The following Basic statement would be used to set the screen size to 32 characters per line:

```
PRINT CHR$(27);"7"
```

The Destructive cursor function allows you to tell the program whether or not to erase the character at the current cursor position. Some screen editing programs require it to be off to function correctly, while others require it to be on so that characters are erased during backspace operations. For this reason we have allowed it to be changed.

**COMPATIBILITY**

This package was designed to be as compatible and convenient to use as possible. Since some Basic programs use commands that affect the screen, we have tried to make them as compatible as possible with the new screen format. One of the most commonly used screen command is the 'PRINT @' statement. Under normal system operation, the location value cannot exceed 511 or else an error is generated. When using the Video Driver program, any value is allowed and is adjusted according to the current screen format (characters per line). To maintain compatibility with existing programs (without changing the PRINT @ values) just reprogram the characters per line to 32. This is accomplished with the Basic statement:

```
PRINT CHR$(27);"7"
```
When in this mode, all screen formatting should be identical to the original display. A benefit you now have with programmable line lengths is that you can now run software written for other computers. For example, by changing the line length to 64, you will be able to run many of the programs written for the Model I. Of course, only Basic programs will run, and some incompatibility exists in the commands available, but it will open a new source of software not available previously.

**KEYBOARD FUNCTIONS**

In order to implement all of the features that have been incorporated into the Video Driver, some of the keys on the keyboard have been reprogrammed to perform new functions. We have used the keys that are used the least to reduce the amount of re-learning by the user. The first key that has taken a new function is the 'CLEAR' key, it functions as the 'CONTROL' key under the Video Driver program. The only other key that has a new function is the 'DOWN ARROW' key. It is used to enter the screen editing mode. All the other keys still function as before.

**Down Arrow**

Enter editing mode.

**Control-Right Arrow/Control-Left Arrow**

Pressing the Control (CLEAR) key with either the Right or Left arrow keys initiates a speed mode which will allow you to manipulate the cursor more quickly while editing or reading basic lines. Otherwise operation is identical to Right or Left arrow.

**Control-">"**

This combination turns on the Auto-Insert mode. Any printable characters entered from the keyboard will be inserted at the cursor position and the remainder of the line will be shifted to the right. Characters pushed off the right side of the display are lost.

**Control-"<"**

If Auto-Insert is in effect, this combination will terminate it. Otherwise it deletes the character under the cursor, pulling the rest of the line to the left.

**SCREEN EDITING**

One of the more powerful features of the Basic Driver is the capability to edit Basic programs simply by moving the cursor to any position on the screen and make changes. To enable the editing mode simply press the 'DOWN ARROW' key. The cursor will start blinking and move to the beginning of the line that it's on. You will now be able to move the cursor up and down on the screen using the 'UP' and 'DOWN' arrow keys respectively. Once you have located the cursor on the line to be edited, press the space bar. The cursor will stop flashing and return to a steady cursor. Use the 'RIGHT' arrow key to move the cursor to the position where a change is required. You will now be able to, 1) overtype a character over an existing one, 2) insert new characters into the line, or 3) delete characters from the line. To enter the insert mode, depress the 'CLEAR' key and the '->' key.
To delete characters, depress the 'CLEAR' key and the '<' key. To demonstrate the editing features, type in the following Basic program:

```
10 FOR X=1 TO 5
20 PRINT "A"
30 NEXT
```

Run the program to see the results. Now, let's change the program so that it prints out "THIS IS A TEST" instead of just "A". Depress the 'DOWN' arrow key and then use the 'UP' or 'DOWN' arrow to move the cursor to line 20. Press the space bar and then move the cursor to the position where we are going to insert the new text. Enter the Insert mode by depressing the 'CLEAR' and '>' keys. Type in the new text. If you want to delete the 'A' that was there originally, use the 'CLEAR' and '<' keys. Now use the 'RIGHT' arrow key to move the cursor past the last character on the line and press 'ENTER'.

NOTE: Moving the cursor over text on the screen appears to the computer as if you had actually typed it in. Always make sure that the cursor is to the right of the text you want entered on the line before pressing 'ENTER'. Once you press 'ENTER' anything to the right of the cursor will be lost.

Now LIST or RUN the program to verify the change you've made. There are several things you should note when in the editing mode. First, the keys used in editing automatically repeat for as long as you keep them depressed. Second, editing can only be performed on a single line at a time. If a line is longer than 80 characters you won't be able to insert and delete properly. This should pose no problem since Basic lines are rarely that long. You will find more applications than those described above as you use the Basic Driver. For example you can also make use of the editing functions in the immediate mode, since moving the cursor over anything on the screen will seem as if you had actually typed it in.

It is important to recognize the difference between the normal control code functions and those implemented to facilitate editing of Basic lines. Although many of the control codes are available from the keyboard, they are primarily intended for use in your programs, and they have no effect on the Basic line buffer. This does not imply that they cannot be useful while editing a Basic program but rather that the user must understand their effect to properly utilize them.

As an example, suppose that you have finally produced a "finished and debugged" program and would now like to do some line packing to produce a more efficient "working version". List the range of lines you would like to work with. Use the Down arrow, Up arrow, and Spacebar to position the cursor at the beginning of the appropriate line. Read to the end of the line with the Right arrow key. Add a colon to separate statements. Now use Control-"m" to move the cursor to the beginning of the next line. Remember that although the cursor is moved, there is no update to Basic's line buffer. At this point you could use Control-"<" to delete the line number, but it might be wiser to use Control-"I" (Destructive cursor off) to move the cursor over the next character to be read. This way, you'll have a reminder to delete the extraneous lines after packing. You will not be allowed to enter more than 249 characters into the line buffer, so if the cursor stops you'll know it's time to start packing another line.
If you're unsure about changes you've made, list the line on another area of the screen. If things aren't quite what you expected, you can still start over on the original.

SOFTWARE VIDEO SWITCH

The Word-PakII contains a software switch on board that allows selection of the displayed video to be either from the Word-PakII or from the computer.

NOTE: In order to use this feature, your computer must have been modified to generate a monochrome composite video signal.

A single bit of a latch located at HFF9C controls the source of the video signal. The following statement would be used to switch between the two signals:

POKE &HFF9C,0  (selects computer video output)
POKE &HFF9C,64  (selects word-PakII output)

WORD-PAK MEMORY MAP

49152-57343  (HCO00-FEFF)  Basic Driver ROM (Word-Pak only)
65432-65633  (HFF98-FF99)  CRTC Registers
65435  (HFF9B)  Display Memory Latch
65436  (HFF9C)  Video Latch (Word-PakII)

SYSTEM OVERVIEW

The heart of the WORD-PAK is a fully programmable LSI CRT Controller which handles all of the display parameters. For normal operation, the CRTC must be programmed for the desired display configuration. The initialization of the WORD-PAK is performed automatically by the Video Driver program and consists of storing the appropriate display format parameter values in the CRTC's internal registers. For an 80X24 display (60 Hz operation), the values in the table on the next page would be stored in each register. Refer to the sample calculations on the following pages to see how these values are derived.

NOTE: For a 50Hz system change R0 to 114, R2 to 95, and R3 to 53 in Table 1. For the Word-PakII, these values are 113, 94, and 19 respectively).
Table 1 - Format Parameter Values

<table>
<thead>
<tr>
<th>REGISTER NO.</th>
<th>VALUE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Word-Pak</td>
<td>Word-PakII</td>
</tr>
<tr>
<td></td>
<td>DEC</td>
<td>HEX</td>
</tr>
<tr>
<td>R0</td>
<td>111</td>
<td>5F</td>
</tr>
<tr>
<td>R1</td>
<td>80</td>
<td>50</td>
</tr>
<tr>
<td>R2</td>
<td>87</td>
<td>57</td>
</tr>
<tr>
<td>R3</td>
<td>60</td>
<td>3C</td>
</tr>
<tr>
<td>R4</td>
<td>28</td>
<td>1C</td>
</tr>
<tr>
<td>R5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>R6</td>
<td>25</td>
<td>19</td>
</tr>
<tr>
<td>R7</td>
<td>26</td>
<td>1A</td>
</tr>
<tr>
<td>R8</td>
<td>120</td>
<td>78</td>
</tr>
<tr>
<td>R9</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>R10</td>
<td>96</td>
<td>60</td>
</tr>
<tr>
<td>R11</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>R12</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R13</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R14</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R15</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R16</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R17</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R18</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R19</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R31</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

* The light pen capability is not implemented on the Word-Pak or WordPakII.

The CRTC's formatting registers are accessed indirectly, that is, the number of the register that is to be accessed is placed at location 65432 (HFP98) and then the value that is to be stored in that register is placed at location 65433 (HFP99). All formatting registers (R0 - R13) are write-only registers. Registers R14 and R15 are read/write registers.

DISPLAY MEMORY ACCESS

In order to provide transparent access of the display memory by the CPU (thus eliminating glitches and perturbations of the display when the CPU accesses the video RAM), a scheme has been implemented on the WORD-PAK that treats the display RAM as a single port (location) as far as the CPU is concerned. All display access, both for refresh and updating, is handled by the CRT Controller.

A dummy register (R31) in the CRTC controls access to the display memory by the CPU. Whenever the CPU is to read or write to the display memory, it must first store the address of the screen location that is to be accessed in registers 18 and 19. Next, the CPU must select Register 31 in the CRTC. This tells the CRTC that the CPU intends to perform an update of the display memory. The CPU must then check bit #7 of the Status Register. If it is set, then the CPU can read or write to the display memory latch.
The CRT Controller takes advantage of the horizontal and vertical retrace periods to update the display memory. Thus providing complete transparent access with no glitches or tearing of the display. Because of this, whenever the CPU writes to the display memory, data is not immediately transferred to the display but instead is held in a latch until the CRTC enters a retrace period. During the retrace, the CRTC transfers the data in the latch to the display memory. If the CRTC is in a retrace period when the CPU stores the data in the latch, then the data is transferred immediately.

**FORMAT PARAMETER CALCULATION**

The following calculations are provided only for reference and may be used as a guide for those who wish to format the display to a different configuration than that provided. The values generated here apply to the original Word-Pak.

**American Monitor Characteristics:**

- Vertical Scan Period (VSP) = 16.66mS
- Horizontal Scan Period (HP) = 63.5uS
- Displayed Portion of Horizontal Scan (DHS) = 51.24uS

Dot Clock Frequency (DC) = 14.318MHz

**Desired Format:**

- 80 Characters/line (Horizontal Displayed) HD
- 24 Lines (Vertical Displayed) VD
- 8 Dots/cell
- 9 Scan lines/character

DHS (actual) = \(\text{HD} \times \text{Dots/cell})/\text{DC}
= (80 \times 8)/14.318 = 44.69 \text{ uS}

This provides approx. 1/2" margin on each side of the display.

Character Clock (CC) = DC/Dots/character
= 14.318/8 = 1.789MHz

Horizontal Total (HT) = (CC*HP)-1
= (1.789*63.5)-1
= 112.6 (lower integer is 112)

This value is placed in CRT Controller Register RO (H70).

HP (actual) = (HT+1)/CC
= (112+1)/1.789=63.16uS

Character Row Period(CRP) = Scan lines/row*HP(actual)
= 9*63.16 = 568.44uS

Vertical Total(VT) = (VSP/CRP)-1
= 16660/568.44 = 28.30 (smaller integer is 28)

This value is placed in CRT Controller register R4 (H1C).
Vertical Total Adjust (VTA) = \((VSP-(VT+1)\times CRP)/HP\)
\[ = \frac{(16660-(28+1)\times 568.44)}{63.16} = 2.77 \text{ (closest integer is 3)} \]

This value is placed in CRT Controller register R5 (H3).

Horizontal Sync Position (HSP) = \(HD+(HT-HD)/3\)
\[ = 80+(112-80)/3 = 90.66 \text{ (lower integer is 90)} \]

This value is placed in CRT Controller register R7 (H5A).

Vertical Sync Position (VSP) = \(VD+(VT-VD)/2\)
\[ = 25+(28-25)/2 = 26.5 \text{ (lower integer is 26)} \]

This value is placed in CRT Controller register R7 (H1A).

Horizontal and Vertical Sync Widths depend on the type of monitor used. However, most standard monitors require a horizontal sync pulse width of approximately 4.7uS +/-0.32uS, and a vertical sync pulse of about 190uS.

Horizontal Sync Width = Horizontal Sync Pulse/CC
\[ = 4.76/.558 = 8.53 \text{ (closest integer is 9)} \]

This value is placed in the lower nibble of CRT Controller register R3 (Hx9).

Vertical Sync Width = Vertical Sync Pulse/HP
\[ = 190/63.16 = 3 \]

This value is placed in the upper nibble of CRT Controller register R3 (H3x).

The above theoretical values are a first approximation. Actual values should be obtained by viewing the display and adjusting the values accordingly. However, the values listed in Table 1 should suit most standard American monitors.

CURSOR OPTIONS

The WORD-PAK cursor is fully programmable and it's features are controlled by registers R10 and R11 in the CRTC. The cursor's format is controlled as follows: Register R10 sets the start scan line and register R11 sets the end scan line. If we wanted an underline cursor, the start scan line would be the seventh scan line and the end scan line would be the eighth, since we only want one scan line (the last one) as the cursor.

POKE 65432,10:POKE 65433,7
POKE 65432,11:POKE 65433,8

The first poke in each line tells the CRTC which register we want to change, and the second stores the value. Note: the value for the cursor must be less than 8 (scan lines).
WORD-PAK User's Manual
PBJ, Inc. P.O. Box 813 N. Bergen New Jersey

In addition to the ability to program the cursor format, we can also program the blink rate or whether we want a cursor at all. Bits 5 and 6 of register R10 controls this feature as follows:

<table>
<thead>
<tr>
<th>b6 b5</th>
<th>Cursor Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>No blinking (steady cursor)</td>
</tr>
<tr>
<td>0 1</td>
<td>No cursor displayed</td>
</tr>
<tr>
<td>1 0</td>
<td>Blink @ 1/16 field rate</td>
</tr>
<tr>
<td>1 1</td>
<td>Blink @ 1/32 field rate</td>
</tr>
</tbody>
</table>

ADDITIONAL CRT C REGISTERS

MODE CONTROL REGISTER (R8)

Hardware dictates this register be set to 120 (H78).

STATUS REGISTER (read-only 65432, HFF98)

7 6 5 4 3 2 1 0 Bit number of Status Register
::: :::: :::::
::: :::: :::::
::: :::: :::::
::: :::: :::::
::: :::: :::::
::: :::: :::::
::: :::: :::::
::: :::: :::::

-------------- Vertical blanking
::: 0 = active display
::: 1 = vertical retrace

-------------- Light pen status register

-------------- Update status
0 = display memory latch read or written (not ready)
1 = data transferred from latch to memory (ready)

CUSTOM CHARACTER SETS

The WORD-PAK is supplied with a character generator which has a set of 128 upper case, lower case, and graphic characters. Each character is formed in an 8x8 matrix (8x10 for the Word-PakII). The character generator is a 2716 type EPROM which allows it to be replaced with custom programmed character sets. (Note: The upper 1K of the ROM is used to store the inverted video character set for the Word-Pak. The Word-PakII's inverse video is generated in hardware).

To define a character in ROM, use the upper 6 address bits as the character code and the lower 3 address bits to define the eight bytes of bit pattern information. For example, the ASCII code for a blank space is 32 (H20). To define that character in ROM, we would store eight bytes of zeroes at addresses H100 thru H107.

- 12 -
To define the graphic character shown below, with a character code of 0 (the first 31 ASCII codes are control codes). We would store the data indicated at the addresses H000-H007.

The following program illustrates how a set of custom characters can be generated on the computer's normal display. The data generated can be dumped to tape and later programmed into an EPROM.
10 'CHARACTER CREATING ROUTINE FOR
20 '80X24 DISPLAY BOARD
30 ' * VER 1.6
40 '*******************************
40 CLEAR 2000, &H5000; V=&H5000; A$=" "+"*" '***** SET VARIABLES *****
50 LINEINPUT "NEW OR OLD CHARACTER SET N/O? ":C$  
70 IF C$="N" THEN 120
80 LINEINPUT "ENTER LOAD FILE NAME: ":C$  
90 F=FILE(SUBSTR) '***** SET FILE LOAD ADDR TO X'50XX' *****
100 LOADMAC$ "/BIN" '***** GET FILE *****
110 GOTo 140
120 CLS:PRINT" INITIALIZING FILE"
130 FOR I=0 TO 255:POKE V+I,255:NEXTI '***** FOR 2K EPROM *****
140 CLS:LINEINPUT "CHAR. NUMBER, IN HEX: ":C$  
150 IF LEN(C$)=0 THEN 550 '***** EXIT *****
160 R=8+VAL("&H"+C$) '***** COMPUTE CHARACTER OFFSET *****
170 R=R+&H5000
180 GOSUB 310 '***** DISPLAY CHARACTER *****
190 LINEINPUT "CHANGE? Y/N: ":D$  
200 IF D$()"Y" THEN 140
210 FOR I=0 TO 7 '***** THIS MODIFIES THE CHARACTER *****
220 PRINT "ENTER BYTE: ":I  
230 LINE INPUT ":D$  
240 Q=VAL("&H"+D$)
250 GOSUB 470 '***** ROTATE BYTE FOR MIRROR IMAGE*****
260 POKER+1,Q
270 NEXTI
280 GOSUB 310 '***** DISPLAY CHARACTER *****
290 FOR I=0 TO 255:NEXTI '***** SHORT PAUSE LOOP *****
300 GOTo 140
310 CLS '***** CHARACTER PRINT SUBROUTINE *****
320 PRINT "PRINT" CHARACTER # ":C$  
330 PRINT "8 4 2 1 8 4 2 1" '***** SETUP HEADER *****
340 FORC=0 TO 7
350 M=128
360 Q=PEEK(A+C) '***** GET BYTE *****
370 GOSUB 470 '***** ROTATE FROM MIRROR IMAGE *****
380 FOR I=0 TO 7 '***** THIS BUILDS THE DISPLAY LINE *****
390 B=M AND Q
400 W=I+ABS(B()*0)
410 B=B+" "+MID$(A$,1,W)
420 PRINT RIGHT$(B$,2);  
430 Q=RIGHT$("0"+HEX$(W,2))
440 M=M/2:NEXTI:PRINT" '***** PRINT THE LINE *****
450 NEXTC '***** GET NEXT LINE *****
460 RETURN
470 Z=0:L=1 '***** SUBROUTINE TO ROTATE THE BITS *****
480 FORX=0 TO 7
490 T=0 AND L
500 IF T=0 THEN 510
510 Z=Z+128/L
520 L=L+2
530 NEXTX
540 Q=Z:RETURN '***** END ROTATE SUBROUTINE *****
550 LINEINPUT "ENTER SAVE FILE NAME: ":C$  
560 SAVE C$ "/BIN", &H5000, &H57FF, &H1000
570 F=32:GOSUB 500:END '***** RESET LOAD ADDR TO X'20XX' AND STOP *****
580 OPEN"D", #1, C$ "/BIN", 1:FIELD#1.1ASZ%
590 LSETZ$=CHR$(F):PUT#1,4:CLOSE:RETURN
00100 ****************************
00110 0012 00160 UPD  EQU  $12  CRTC UPDATE REG #
00000 00170 CRS  EQU  $0E  CRTC UPDATE REG #
0000C 00180 DIS  EQU  $0C  CRTC DIS STRT REG

0000 00190 COUNT  RMB  1  CHARACTER COUNTER FOR PRINTER
7D00 C2000 ORG $7D00
7D00 30 8D 0B 0210 LEAX FORPAR.PCR

7D04 5F 00220 NA3  CLR#
7D05 A6 80 00230 LOOP1 LDA .X+ STORE FORMAT
7D07 F7 FF98 00240 STB $FF98 VALUES AT CRTC
7D0A B7 FF99 00250 STA $FF99
7D0D 5C 00260 IMCB ARE ALL VALUES STORED?
7D0E C1 00270 CMPB #$14 NO; KEEP DOING IT

7D10 25 F3 00280 BCS LOOP1
7D12 BE 0168 00290 POIT LDX #$168
7D15 BF 7D05 00300 STX RTN1+1
7D18 BF 01A1 00310 NA5 LDX #$01A1
7D1B BF 7E36 00320 STX RTN2+1
7D1E BF 017A 00330 NA6 LDX #$017A
7D21 BF 7E79 00340 STX RTN3+1
7D24 30 8D 0032 00350 NA7 LEAX ENTRY.PCR
7D28 B6 0167 00360 NA1 LDA #$0167 REDIRECT BASIC
7D2B B7 7D04 00370 STA RTN1
7D2E B6 01A0 00380 NA8 LDA #$01A0
7D31 B7 7E35 00390 STA RTN2
7D34 B6 0179 00400 NA9 LDA #$0179
7D37 B7 7E78 00410 STA RTN3
7D3A B6 7E 00420 NB1 LDA #$7E
7D3C B7 0167 00430 STA #$167 HOOKS FOR OUTPUT
7D3F B7 01A0 00440 STA #$01A0 PRINT, AND
7D42 B7 0179 00450 STA #$0179 CLEAR SCREEN
7D45 BF 0168 00460 STX #$0168
7D48 30 8D 00E3 00470 LEAX CLEAR.PCR
7D4C BF 01A1 00480 NA2 STX #$01A1
7D4F 30 8D 00E5 00490 LEAX PRINT.PCR
7D53 BF 017A 00500 NA4 STX #$017A
7D56 17 0122 00510 LBSR CLSCR GO AND CLEAR SCREEN

7D59 39 00520 RTS
7D5A 34 37 00530 ENTRY PSHS X,Y,A,B,CC SAVE ALL REGS
7D5C BE 7ECE 00540 LDX DPS

00550 ****************************
00560 #This portion of the program will allow #
00570 #Printer without auto-linefeed to be #
00580 #used with the Word-Pak. Also it will #
00590 #Print the line length that is read in #
00600 #Address 135 Dec. #
00610 #Delete lines 550-700 if your printer #
00620 #Has auto-linefeed. #
00630 ****************************

7D5F 06 6F 00640 LDR $6F PRINT TO WHAT DEVICE?
7D61 C1 FE 00650 CMPB $FE IS IT TO PRINTER?
7D63 26 1A 00660 BNE NB2 NO; CONTINUE AS NORMAL
7D65 06 9B 00670 LDR $9B CHECK LINE LENGTH WE SPECIFY
7D67 0C 00 00680 INC COUNT INCREMENT COUNTER
7D69 01 00 00690 CMPB COUNT IS IT THE END OF THE LINE?
7E6B 27 84 00790 BEQ LFEED YES: GO DO A LINE FEED
7E6D 81 80 00710 CMPA $80 IS IT A C/RETURN?
7E6F 26 61 00720 BNE DONE NO: CONTINUE LIKE NORMAL
7E71 86 8A 00730 LFEED A LINE FEED
7E73 0F 80 00740 CLR COUNT A RETURN
7E75 BD A285 00750 JSR $A285 SEND TO PRINTER
7E78 86 80 00760 LDA $80 A C/RETURN
7E7A BD A285 00770 JSR $A285
7E7D 20 53 00780 BRA DONE
7E7F 91 80 00790 NB2 CMPA $8 IS IT A BACKSPACE?
7E81 26 1C 00800 BNE CHK1 NO: GOTO NEXT CHECK
7E83 BC 7ECC 00810 CMPX LST IS CURSOR AT START OF
7E86 22 88 00820 NB3 BHI BAKSP LINE?
7E88 86 20 00830 LDA $20 PUT BLANK THERE
7E8A 80 4B 00840 BSR PUT
7E8C 8D 57 00850 BSR MOVE TELL CRTC THE LOCATION
7E8E 20 42 00860 BRA DONE
7E90 30 1F 00870 BAKSP LEAX -1.X MOVE CURSOR LEFT
7E92 BF 7ECE 00880 STX DPS
7E95 80 46 00890 NB4 BSR MOVE TELL CRTC THE LOCATION
7E97 86 2B 00900 LDA $29 PUT A BLANK THERE
7E99 BD 80 3C 00910 BSR PUT
7E9B 8D 48 00920 BSR MOVE
7E9D 20 33 00930 BRA DONE
7E9F 81 80 00940 CHK1 CMPA $80 IS IT A C/RETURN?
7DA1 26 12 00950 BNE CHK2 NO: GOTO NEXT CHECK
7DAB 83 80 00960 LDX LST
7DAD 30 88 50 00970 NB5 LEAX $50.X ADD 88($50) TO LINE
7DAD BF 7ECE 00980 STX DPS
7DAF BF 7ECC 00990 NB6 STX LST
7DB0 BD 54 01000 NB7 BSR SCROLL SCROLL SCREEN
7DB1 BD 80 32 01010 BSR MOVE TELL CRTC NEW LOCATION
7DB3 20 1D 01020 BRA DONE
7DB5 81 20 01030 CHK2 CMPA $20 IS IT A PRINTABLE
7DB7 25 19 01040 BCS DONE ASCII CODE<=$32?.
7DB9 8D 1C 01050 BSR PUT PUT IT ON SCREEN
7DBB 80 81 01060 LEAX 1.X ADVANCE CURSOR
7DBD BF 7ECE 01070 STX DPS
7DBF BE 7ECC 01080 NB8 LDX LST IS CURSOR AT START OF
7DC3 30 88 50 01090 NB9 LEAX $50.X NEW LINE? THEN ADD 88 TO LINE
7DC5 BD 7ECE 010A0 CMPX DPS
7DC9 22 05 01100 NC1 BHI FIN
7DCA BF 7ECC 01110 STX LST
7DCE 30 35 01130 NC2 RSR SCROLL SCROLL SCREEN
7DD0 80 13 01140 FIN BSR MOVE TELL CRTC
7DD2 35 37 01150 DONE PULS X,Y,A,B,CC REGAIN REGS AND
7DD4 00 00 01160 FCH 00 CONTINUE
7DD5 0000 0117D EBD 00 BASIC HOOK
7DD7 06 1F 01180 PUT LDB $1F TELL CRTC THAT CHAR
7DD7 07 FF 01190 STB $FF98 IS COMING
7DD7 06 FF 99 01200 WAIT LDB $FF98 IS CRT READY?
7DD7 29 FB 01210 BPL WAIT NO: KEEP WAITING
7DE1 BF 7F 01220 STA $FF98 PUT CHAR THERE
7DE4 39 01230 RTS
7DE5 BE 7ECE 01240 MOVE LDX DPS TELL CRTC TO MOVE
7DE8 86 12 01250 NC3 LDA $UPD CURSOR AND UPDATE
7DEA 8D 85 01260 BSR UPRDREG DISPLAY POSITION
7DEC 86 8E 01270 LDA $CRS
7DEE 8D 81 01280 BSR UPRDREG
7DF0 39 01290 RTS
70F1 34 02 01300 UPDREG PSHS A UPDATE CRTC REG
70F3 07 FF98 01310 STA $FF98 ACCUM A CONTAINS REG #
70F6 1F 10 01320 TFR X,D AND X CONTAINS VALUE TO
70F8 07 FF99 01330 STA $FF99 BE STORED.
70F8 35 02 01340 PULS A
70FD 4C 01350 INCA
70FE 07 FF98 01360 STA $FF98
70E1 07 FF99 01370 STB $FF99
70E4 39 01380 RTS
70E5 0E 7ECC 0T390 SCROLL LDX LST
70E8 8C 0D0 01400 NC4 CMPX #$7D0 IS CURSOR BELOW
70E8 2B 21 01410 BMI RTRN SCREEN?
70E8 17 00C 01420 LBSR CLRLIN YES? THEN SCROLL
70E1 0E 7ECA 01430 LDX DST ONE LINE
70E1 30 08 00 01440 HC5 LEAX "$0,X
70E2 6C 0500 01450 CMPX #$5000 IS IT TIME TO
70E1 2B 0C 01460 BMI SAME START A NEW SCREEN?
70E2 08 0700 01470 LDX #$700 SET CURS AND DIS-
70E1 BF 7ECC 01480 STX LST PLAY POS TO BOTTOM
70E2 6F 7ECE 01490 NC6 STX DPS
70E2 8E 0000 01500 NC7 LDX @0 RESET DISP START
70E2 BF 7ECA 01510 SAME STX DST
70E2 56 0C 01520 NC8 LDA @DIS TELL CRTC TO
70E2 C0 0C 01530 BSR UPDREG START NEW DISPLAY
70E2 39 01540 RTRN RTS
70E2 F4 37 01550 CLEAR PSHS X,Y,A,B,CC
70E3 8D 48 01560 BSR CLSCR
70E3 35 37 01570 PULS X,Y,A,B,CC
70E3 00 01580 RTN2 FCB 00
70E3 00 0000 01590 FDB 00
70E3 34 37 01600 PRINT PSHS X,Y,A,B,CC
70E3 9A 06 01610 LDX "$A6
70E3 C6 08 01620 LUKAGH LDA ,X CHECK TO SEE IF
70E3 E1 07 01630 CMPA #$87 IT IS A PRINT @
70E4 27 34 01640 BEO NOMOVE
70E4 81 40 01650 CMPA #$40
70E4 26 F6 01660 BNE LUKAGH
70E4 25 50 01670 LDA #$50 IF IT, IS MOVE
70E4 6F 05 01680 CLRB CURSOR TO NEW
70E4 99 8E 01690 LDX "$8 LOCATION
70E4 8B 30 09 FC00 01700 LEAX #$400.X
70E4 FC 0C 01710 LOOP4 CMPX #$20
70E5 2B 06 01720 BMI NEWLOC
70E5 4C 01730 INC
70E5 30 0E 00 01740 LEAX -$20,X
70E5 2B 0F 01750 BRA LOOP4
70E5 A0 01760 NEWLOC MUL
70E5 C3 0200 01770 ADDD #$200
70E5 ED 7ECC 01780 STD LST
70E6 1C 7ECA 01790 HC9 LDD DST
70E6 73 7ECC 01800 HD1 ADDD LST
70E6 7C 7ECC 01810 HD2 STD LST
70E6 A3 10 01820 HD3 TFR X,D
70E6 BD 7ECC 01830 LDX LST
70E6 F9 08 01840 HD4 AGX
70E7 8F 7ECE 01850 STX DPS
70E7 33 1F 01860 HD5 LBSR MOVE
70E7 4E 37 01870 NOMOVE PULS X,Y,A,B,CC
70E7 00 01880 RTN3 FCB 00
70E7 00 0000 01890 FDB 00
Fig. 1
WORD-PAK User's Manual  
PBJ, Inc. P.O. Box 813 N. Bergen New Jersey

WARRANTY

All equipment manufactured by PBJ, Inc. is warranted to be free from defects in material and workmanship for a period of 90 days from date of sale. Defects not caused by user negligence, misuse, or abuse will be repaired free of charge, provided the equipment is returned, postpaid, to PBJ, Inc. within the warranty period. All equipment manufactured by PBJ, Inc. is fully tested prior to being shipped, therefore PBJ, Inc. reserves the right to determine which repairs are in warranty where shipping damage, misuse, or abuse is in question. This warranty is limited to the replacement of defective parts, no responsibility is assumed for damage to other equipment. All software sold with the equipment is supplied on an "AS-IS" basis, without warranty.

REPAIR POLICY

The minimum service charge for all repairs is $15.00. Repair costs will be calculated as parts cost plus $25.00 per hour of labor. Repaired equipment will be returned C.O.D. for shipping and repair costs.

WARRANTY REGISTRATION

Please fill out the information requested below and mail to:

PBJ, Inc.  
PO Box 813  
North Bergen, NJ 07047

Place purchased:

Purchaser's Name:
Street:
City:
State:
Zip:

Equipment Purchased:
Serial No.:
Date of Purchase: