BANKER USER MANUAL

<C> 1985 by

J&R ELECTRONICS
P.O. Box 2572
Columbia, Md. 21045

PREFACE

Welcome to the world of J&R Electronics. We would like to thank you for choosing our product. The following is a brief look at J&R and some of our policies.

J&R products are designed, assembled and tested by Jesse Jackson and Raymond Rowe, who are and have been dedicated CoCo nuts since day one when the CoCo was a 4K machine selling for approximately $350.00. Both are presently or have been co-authors of the Dr. ASCII column as well as having published articles of their own.

It is our policy to fully support our products in the future with product updates, new software availability, and new products. All this will be directed to our "Banker" customers first; by special mailing.

DISCLAIMER

J & R Electronics assumes no responsibility for any damages caused by the buyer during assembly, installation or use of the Banker. J & R Electronics will not be liable for damages, direct or consequential, general or special, nominal or exemplary, resulting from use of any products or documentations supplied by J & R Electronics.

This manual, the Banker and any documentation pertaining to the Banker is the property of J & R Electronics. Reproduction by any means, electrical or otherwise, is strictly prohibited except by prior written permission from J & R Electronics.
# INDEX

<table>
<thead>
<tr>
<th>SECTION</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>INTRODUCTION</td>
<td>3</td>
</tr>
<tr>
<td>2.0</td>
<td>SOFTWARE DESCRIPTION</td>
<td>5</td>
</tr>
<tr>
<td>3.0</td>
<td>CUSTOMIZATION EXAMPLES</td>
<td>12</td>
</tr>
<tr>
<td>4.0</td>
<td>PROGRAMMING INFORMATION</td>
<td>17</td>
</tr>
<tr>
<td>5.0</td>
<td>ADVANCED USER INFORMATION</td>
<td>18</td>
</tr>
</tbody>
</table>

# APPENDIX

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A.</td>
<td>BANK SWITCH ADDRESSES</td>
<td>24</td>
</tr>
<tr>
<td>B.</td>
<td>BANKER SOFTWARE MEMORY MAP</td>
<td>25</td>
</tr>
<tr>
<td>C.</td>
<td>WARRANTY</td>
<td>26</td>
</tr>
<tr>
<td>D.</td>
<td>OWNER REGISTRATION</td>
<td>27</td>
</tr>
</tbody>
</table>
1.0 INTRODUCTION

1.1 The "BANKER"

The "BANKER" is a memory expansion board for the Radio Shack COLOR COMPUTER. When your computer has the "BANKER" installed, it will appear to be a 64K computer until it is programmed for expanded memory operation. "BANKER II" is a 512K memory expansion board that can support 256K or 512K of expansion memory. Its predecessor, "BANKER I" supports only 256K. All versions of the software support both units at the 256K level. The two will hereafter be referred to simply as the "BANKER".

The "BANKER" works with COLOR BASIC, EXTENDED BASIC, DISK  EXTENDED BASIC, JDOS, ADOS, and WORDPAK, CASSETTE or DISK systems. The board was designed to be compatible with all hardware such as WORDPAK, PC pak, and MULTI-PAK.

A simple single-pole single throw switch can be added to force the "BANKER" to be absolutely compatible with any 64K software or hardware configuration. When installed properly, this switch may be toggled to either position with computer power on without harming the "BANKER" or the computer.

1.2 The "BANKER" and RESET condition

The "BANKER" has a power-on reset circuit that causes the bank latches to be programmed for CPU and VDG bank 0 when the computer is turned on. When the computer is RESET manually or by software, the "BANKER" will remain in the CPU and VDG bank it was programmed for before RESET. Closing the 64K compatibility switch (optional equipment), will reset the "BANKER" to CPU and VDG bank 0.

*** NOTICE ***

The 256K memory chips have a long retention of data after power is off. When powering down, wait 30 seconds or more before powering back up. A recommended software power down and back up is:

POKE &H71,0 ' cold start BYTE 11h, 0
POKE &H72,0 ' cold start LOC 11h, 0
Press the RESET button ......

1.3 Convention

Several conventions are used throughout the documentation. A BANK is any part of a contiguous memory segment of 64K bytes. A PAGE is a contiguous memory segment of 32K bytes in length. There are two PAGES per BANK of the expansion memory. The maximum number of banks available is found by dividing the total memory by 64K. For example, in a 256K system, there are four banks (256k / 64k = 4). The maximum number of pages is twice the number of banks. Banks and pages are numbered starting with zero.

The prefix symbol "&H" denotes a hexadecimal number and is synonymous with the BASIC prefix "&H". They will be used throughout the manual interchangeably.
The bank prefix symbol ">" will denote an upper 32K page of a bank with address range from $8000 to $FEFF. The bank prefix symbol "<" will denote the lower 32K page of that bank with address range $0000 to $7FFF. The bank prefix symbol "#" will denote the whole bank with address range $0000 to $FEFF. For example, <0 references the lower 32K of bank 0.

1.4 Cautions

Many of the programs supplied make extensive use of the expansion memory available. Most of these programs may be customized by you. Care should be taken when using the BANKRDSK, BANKRSPL, and PCOPYMOR programs simultaneously, if they are not configured properly, their storage area may overlap and yield unpredictable results. Consult the ADVANCED USER INFORMATION section for details on memory use.

Many 64K programs blindly copy ROM to RAM, this would inactivate the RAM DISK, PCOPYMOR, and any other 64K programs until you LOADM and EXEC them again.

Programs that use the assembly language instruction, CLR $FFD0, instead of STA $FFD0, re-program the "BANKER". This can happen when programming the SAM display offset or MAP type, and is usually easy to find and fix (see the CLRFIX utility program in the SOFTWARE section). The 64K compatibility switch will overcome this problem, but must be switched OFF to use the extra 64K banks provided by the memory expansion.

1.5 Upgrade policy

We value your patronage, we intend to offer generous upgrade allowances to our customers. To qualify, you MUST complete the registration form enclosed in this manual and return it to us within thirty days from the purchase date. Send us a self-addressed stamped envelope and we'll notify you of upgrades. MOST upgrades require only that you pay a nominal fee plus $3.00 shipping & handling, you must include the original SOFTWARE PAC diskette/tape when you upgrade.

1.6 Future

We're dedicated to expanding the COCO in every direction possible. We would appreciate hearing what products you would like J & R to offer. We intend to offer reasonably priced software for the "BANKER", and products for the COCO in general.
2.0 SOFTWARE DESCRIPTION

2.1 BANKRDSK – (a.k.a the "ramdisk")

NOTE: BANKRDSK and RAMDISK will be use interchangeably throughout the manual in reference to this program.

The 35/40 TRACK RAM DISK program! It patches into your disk system to emulate a FAST disk drive. All DISK EXTENDED BASIC commands are functional on the ramdisk except DSKINI. If you BACKUP from BASIC or LOAD DRIVE from the menu, there is no need to INITIALIZE from the menu. If you want a "freshly formatted ramdisk, then use the INITIALIZE option in the menu. INITIALIZE is the equivalent of DSKINI for the ramdisk.

RAMDISK A, RAMDISK B, or both may be used by 512k versions of the "BANKER". Only RAMDISK A may be used with 256k versions. Ramdisk A uses banks 0-3, Ramdisk B uses bank 0, 4-7.

2.1.0 Menu Options

(Use the <BREAK> key to return to the menu.)

2.1.0.0 TOGGLE DRIVE (512K version)
512K versions are capable of having two RAM DISKS, DRIVE A and DRIVE B. This option switches to the opposite drive each time it is selected.

2.1.0.1 SET DRIVE
This assigns the RAM DISK a drive number between 0 and 3 (0 and 2 for JDOS). If you have drives 0 and 1, you may want to assign the RAM DISK as drive 2 or 3. If you have a program that uses drive 0 a lot, you may want to assign it as drive 0 to speed up the program. SET DRIVE will not modify the contents of the RAM DISK, it's like swapping a disk between drives.

2.1.0.2 SET TRACKS
This configures the RAM DISK for 35 or 40 tracks. RADIO SHACK DISK EXTENDED BASIC users should use 35 tracks, JDOS or ADOS users may need to set for 40 track mode.

2.1.0.3 INITIALIZE
This is the DSKINI equivalent for the RAM DISK, it fills the RAM DISK with &HFF's, like a freshly formatted disk. See the CUSTOMIZING section for details on the fast initialization option.

2.1.0.4 LOAD DRIVE
This option permits you to transfer a disk from a real drive into the RAM DISK. You may load from any drive into the RAM DISK, (i.e. load drive = 0, RAM DISK = 0 is valid option).

2.1.0.5 REMOVE
This option does not modify the contents of the RAM DISK, only removes it from the system, gracefully. Use this to free up the memory occupied by the RAMDISK without powering down the computer.
2.1.0.6 QUIT

Exits the RAMDISK menu and returns to BASIC.

2.1.1 As supplied

This program as supplied on your SOFTWARE PAC is configured for:

PDEST = $FD00, MAXBNK = 3 (256K).

See the CUSTOMIZING section for information on customizing this program for your particular system.

2.2 S. XXX

This is a BASIC program that replaces a file by the same name on the TELEWRITER-64 disk, speeding up the DISK I/O and/or providing the convenience of an additional drive. This program requires the program "BANKRDSK.BIN" to be on the TELEWRITER-64 disk.

You must modify the variable, P6, in the file named "U/BAS" on the TELEWRITER disk. Line number 12 should contain P6, change it from 56320 to 55808. Note that programs S/BIN and S/ASC have a variable in line # 2 called MX. This sets the Max drive number you can use, you may want to change this to assign the RAM DRIVE an unused drive number.

IMPORTANT: Be careful about answering "Yes" to the prompt "LOAD RAM DISK?". "BANKRDSK/BIN" loads at &H6FF0 for preparation, and transfers to permanent memory at &HFD00 after set-up. If you have been editing text, LOADing "BANKRDSK/BIN" may overwrite it!

It is best to LOAD the RAM DISK when you first come up in TELEWRITER, otherwise, you should SAVE your work, then LOAD the RAM DISK so that no text is lost if the buffer is overwritten.

2.3 PCOPYMOR

This program patches RADIO SHACK EXTENDED BASIC to allow extra pages of graphics from BASIC using the PCOPY command!

For instance, you could enter the line

PCOPY 1 TO 125

to copy a graphics page from page 1 to "BANKER" page 125. The table below shows how much extra graphics you can get with the "BANKER".

<table>
<thead>
<tr>
<th>EXTENDED COLOR BASIC</th>
<th>BANKER SCREENS (see note *)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMODE</td>
<td>PAGES</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

* NOTE - these are max added pages, "stock" program may differ.*
2.3.1 As supplied

This program as supplied on your SOFTWARE PAC is configured for:
PDEST = $F800, MAXBNK = 3 (256K), P6PBNK = 40 (128 PAGES).
See the CUSTOMIZING section for information on customizing this program for your particular system.

2.4 PCOPYDEM

A BASIC program which loads ten PMODE 4 pictures (6144 bytes per picture, total 61,440 bytes) into "BANKER'S" memory. These pictures are then PCOPY'd down one at a time into a PMODE 4,1 screen for display.

This demonstrates how an educational or adventure program written in BASIC could use the extra pages for faster animation. Once the "BANKER'S" expansion memory is loaded with extra pages of graphics, you can use PCOPY commands to move them back and forth in the expansion memory quickly, without accessing the disk or using slow draw commands again!

2.5 BANKRSPL - (a.k.a the "spooler")

This is a machine language SPOOLER program that stores printer data into a sizeable buffer, using interrupts to print buffered data while you're using the computer for other tasks! BANKRSPL/BIN may be customized to utilize available contiguous segments of memory in each bank.

NOTE: BANKRSPL and SPOOLER will be use interchangeably throughout the manual in reference to this program.

The spooler will work with parallel printer drivers and other software that use the standard output routines, provided their driver is installed prior to EXEC'ing BANKRSPL. Programs that modify the interrupt vectors or condition code of IRQ may disable the spooler.

The "stock" version supplied on the SOFTWARE PAC is configured to give over 30K of printer buffer when used with the RAM DISK (BANKRDSK) in 35 track mode.

A SPEED feature has been added to improve computer response at baud rates less than 1200 baud. A SPEED of zero is the fastest, the SPOOLER will check the printer busy line every interrupt period (16.67ms), and output a character from the buffer if the printer is ready. A value of 255 is the slowest, checking the printer busy line only once in 255 interrupt periods (255 * 16.67ms), and outputting a character if the printer is ready.
2.5.1 SPOOLER CONTROL CODES

The spooler has several features that are controlled by codes passed via the keyboard or BASIC PRINT statements. The format is:

PRINT CHR$( CODE ) ' SPOOLER CONTROL CODE

; where CODE is one of the values listed below.

******************************************************************************
CODE      ACTION
******************************************************************************
1         RESET - RESETs the spooler to initial state. Any data in
            the buffer will be lost.
2         ON - re-activates the spooler, printed data will be
            stored in the buffer immediately after the last
            data received before an "OFF" code.
3         OFF - de-activates the spooler, printed data will not
            be stored in the buffer. Data in the buffer is
            preserved intact.
4         KILL - removes the spooler from the system, gracefully.
5         COPY - COPY the buffer contents by the value of NUMCPY.
            If more than one copy is desired, NUMCPY should
            be POKE'd before this command. After
            completion, NUMCPY will be reset to one.
            *** Use the COPY code after the text to be
            copied is printed in full.
6         SILENT- This code is a "toggle". If output is being sent
            to the printer, it will cease to be sent until
            the next SILENT toggle code is received. Input
            data to be printed will be stored in the
            spooler's buffer, regardless of the state of
            the SILENT flag.

2.5.1.1 COPY & SILENCE COMMAND EXAMPLE

LOAD "BANKR SPL":EXEC
PRINT CHR$(4) ' SILENCE PRINTER
PRINT #-2,"PRINT TO THE BUFFER"
PRINT CHR$(6) ' UNSILENCE PRINTER
...... WAIT FOR PRINTER TO FINISH ......
POKE &HFA02,3' SET NUMCPY TO 3
PRINT CHR$(5); ' SEND COPY COMMAND

Three copies of the buffer will be made.

See the CUSTOMIZING section for information on customizing the SPOOLER for your particular application.
2.6 BANKCOPY/BIN

A machine language utility program that copies ROM to RAM in all upper 32K banks. The "OK" prompt is modified to reflect which upper bank is active.

LOAD "BANKCOPY"
EXEC

Your basic prompt should now be >0", this means the upper bank is bank #0.

ZZ = PEEK(&HFFC9)

Your basic prompt should now be >1", this means the upper bank is bank #1. This demonstrates switching upper 32K banks while keeping the lower 32K bank always bank <0. Switching whole banks such as:

ZZ = PEEK(&HFFC1)

would "crash" the computer because this switches to >0 and <0, the latter has not been initialized for BASIC to use.

You could use this program to store data or machine language
programs in unused areas of the upper 32K in EACH BANK. For instance, a screen dump program in BANK 0 at $HE000, a monitor/debugger in BANK 1 at $E000, etc.

BANKCOPY lets you switch upper banks of 32K with the following commands:

ZZ = PEEK(&HFFC8) ' BANK >0
ZZ = PEEK(&HFFC9) ' BANK >1
ZZ = PEEK(&HFFCA) ' BANK >2
ZZ = PEEK(&HFFCB) ' BANK >3

2.7 BANKCOPY/TXT

This is the source code for BANKCOPY, to demonstrate how the "BANKER" is programmed in assembly language.

2.8 BANKERBAK

A single swap, multiple drive, multiple copy, fast backup utility program. This program copies an entire 35/40 track diskette into the "BANKER" memory. Multiple copies may then be backed up from memory to multiple drives without re-inserting the source diskette!

**** WRITE PROTECT YOUR BACKedUP SOURCE DISKETTE
**** DON'T USE the FBAK option
**** if you DON'T HAVE RSDOS 1.0 or 1.1!

CLEAR 200,&H6EF ' MUST USE WITH "FBAK" VERSION
LOAD "BANKERBAK"
EXEC

You will be prompted for inserting source/destination diskettes. Use the <BREAK> key to bypass selections.

2.8.1 As supplied

This program as supplied on your SOFTWARE PAC is configured for:
PDEST = $FD00, MAXBNK = 3 (256K), MAXTRK = 34 (35 tracks),
LDRIV = 0 (LOAD DRIVE) and COPY DRIVES 0 and 1.

A customizing option will enable use with 40 track drives and copies to be made on single or multiple drives. IF you have Radio Shack DISK EXTENDED BASIC Version 1.0 or 1.1, unformatted diskettes will be DSKINI’d if the FBAK option is on.
See the CUSTOMIZING section for information on customizing this program for your particular system.

2.9 BANKRPAG

A BASIC program which programs the SAM and the "BANKER" for viewing any area of the expansion memory in any GRAPHICS or TEXT display. Run this program after using BANKRBK or BANKRDSK to see what was on the disk, or after using the spooler to view text in the buffer!

2.10 CLRFINX

This is a utility program that changes machine language CLEAR op-codes, having operands in the range $FFC0 to $FFDF, to STA op-codes. The program fixes the CLEAR incompatibility, searching by filename (fastest method) or by track/sector range. Read the comments in the remarks of the program for information on specific program fixes.

The program also has a feature to disable the ROM to RAM routines used in 64K programs. Use this on 64K programs that you can't intervene to load the RAMDISK or SPPOOLER. Run the CLRFINX program on a BACKUP copy of the 64K program. Load and activate the RAMDISK or SPPOOLER, then load and activate the fixed 64K program.

An extra feature of this program is the ability to find and display machine language file addresses for you.

2.11 RAMDSKUT

This is a utility sub-routine that allows the user to change the Ramdisk's drive number, number of tracks, or load the Ramdisk from any drive. This routine should be used only when the Ramdisk has been activated. It uses less memory than the alternative of LOADM'ing BANKRDSK again to make the changes.

2.12 PAGER/BIN

This program configures the Banker's ram for multiple pages of 32K, effectively multiple 32K Color Computers. Any programs in memory when this program is EXEC'd will be copied to all pages. A command, PAGE N, is added to BASIC for switching pages in direct mode or from within a program. Data can be transferred between banks or pages of ram with the PPOKE and PEEK commands.

COMMANDS ADDED:

PAGE <N> : Switch or identify page.
PPOKE PAGE,ADDRESS,DATA : Page poke DATA to ADDRESS in PAGE.
PPEEK PAGE,ADDRESS,DATA : Page peek DATA at ADDRESS in PAGE.

The command PAGE alone, will print the current page number. The command PAGE N, where N is a valid page number, will switch to the page given by N.
To get the idea of chaining programs, enter these lines:

```
10 PDEST = &H7800 ' PAGER DEST ADDRESS
20 MAXBNK = PEEK(PDEST+2) ' MAXIMUM BANK #
30 PRINT"THIS IS PAGE #:"; 
40 PAGID = PEEK(PDEST+6) ' PAGE #
50 PRINT PAGID
50 FOR I = 0 TO 500: NEXT
60 IF PAGID < MAXBNK THEN PAGID = PAGID +1 ELSE PAGID = 0
70 PRINT"SWITCHING TO PAGE #:";PAGID
80 FOR I = 0 TO 500: NEXT
90 PAGE P 'Leave a space after the P'
100 GOTO10
LOAD"PAGER":EXEC:RUN
```

The PAGER will work as multiple pages of 64K, this action is automatic if you run any 64K program while PAGER is activated (BANKCOPY, for example).

**CAUTION:** A 64K program may write over the PAGER program, causing it to deactivate or crash. Because PAGER adds BASIC commands, using available tokens, it may not be compatible with some enhancement DOS's or other programs that add tokenized BASIC commands.

2.13 PAGER/TXT

This is the SOURCE code for PAGER/BIN. It provides proven routines for programming the "BANKER" as banks of 64k or pages of 32k.

2.14 OS9BTFIX

A BASIC program that fixes the OS9 BOOT to cure an incompatibility with the "BANKER". You must use this program on your BACKedUP bootable OS9 diskettes to utilize OS9 RAMDISK.

2.15 OS9 RAMDISK

This is a separate OS9 format (35 track) diskette containing install files, documentation, the OS9 RAMDISK program, and SOURCE CODE. You must have "booted-up" into the OS9 operating system to use this diskette. Place this diskette in /D1 and use the INSTALL.35, INSTALL.40SB, or INSTALL.40DS procedure to copy the appropriate programs from the diskette to a BACKedUP copy of your SYSTEM DISKETTE in /DO. List the "readme" file on the diskette, using OS9's list command, for further information.
3.0 CUSTOMIZING THE SOFTWARE

The programs on the SOFTWARE PAC support TAPE OR DISK systems and 256K or 512k memory versions. Most are configured for 256K disk systems, and will need simple changes. These will be covered in the examples given below.

There is no difference in the TAPE and DISK versions of the SOFTWARE PAC. All programs on tape may be transferred to disk. Some BASIC programs have a variable named TD to identify Tape or Disk system. Read the remarks in the BASIC program and modify this variable according to whether you have a TAPE or DISK system.

Most of the machine language programs provided on your SOFTWARE PAC can be customized for your system. Section 5.0 ADVANCED USER INFORMATION, provides information on customizing locations in programs that may be customized.

The following examples of user customization will help you configure these programs for your system.

*** NOTE: USE A BACKedUP COPY OF THE PROGRAM WHEN CUSTOMIZING ***

3.1 BANKRDSK - RAMDISK CUSTOMIZATION

The following locations may be useful if you desire to customize a ramdisk. The program loads at &H6FF0, and relocates itself to PDEST (&HFD00) when EXEC’d.

LOAD &H6FF0, END &H7AFF, EXEC &H6FF0

3.1.1 RAMDISK B option (512k versions only)

LOADM"BANKRDSK"
POKE &H7008,7 ' MAXBNK = 7
SAVEM"BANKRDSK",&H6FF0,&H7AFF,&H6FF0

3.1.2 Fast Initialization option (clears directory track only)

LOADM"BANKRDSK"
POKE &H6FF3,255 ' FSTINI 0=off 255=on
SAVEM"BANKRDSK",&H6FF0,&H7AFF,&H6FF0

3.1.3 AUTO Initialization (INITIALIZE, bypass menu options)

LOADM"BANKRDSK"
POKE &H6FF4,255 ' AUTO mode enable
POKE &H6FF5,255 ' AUTO init on
SAVEM"BANKRDSK",&H6FF0,&H7AFF,&H6FF0

3.1.4 AUTO LOAD option (LOAD from drive, bypass menu options)

LOADM"BANKRDSK"
POKE &H6FF4,255 ' AUTO mode enable
POKE &H6FF4,0 ' LOAD RAMDISK A from drive 0
POKE &H6FF7,1 ' LOAD RAMDISK B from drive 1
POKE &H700F,0 ' RAMDISK A = DRIVE #0
POKE &H7010,34 ' RAMDISK A = 35 TRACK
POKE &H7014,1 ' RAMDISK B = DRIVE #1
POKE &H7015,39 ' RAMDISK A = 40 TRACK
SAVEM"BANKRDSK",&H6FF0,&H7AFF,&H6FF0
3.1.5 RELOCATE option

LOADM"BANKRDSK"
POKE &H7003,&HE0 " PDEST = &H7003 & 7004
POKE &H7004,&H00 " MOVE TO $E000-$E0FF
SAVEM"BANKRDSK",&H6FF0,&H7AFF,&H6FF0

3.2 BANKCOPY CUSTOMIZATION
The customization for this program is one poke for 256K or 512K versions. The source code is included on the SOFTWARE PAC.

3.2.1 Memory size option

LOADM"BANKCOPY"
POKE &H0E02,3 ' 3=256K 7=512K
SAVEM"BANKCOPY",&H0E00,&H0E5F,&H0E00

3.3 BANKRBBAK CUSTOMIZATION
The following locations may be useful if you desire to customize this program. The program loads at &H6FF0, and relocates itself to PDEST (&HFD00) when EXEC'd.

LOAD &H6FF0,END &H76FF,EXEC &H6FF0

3.3.1 40 TRACK option

LOADM"BANKRBBAK"
POKE &H7006,39 ' 40 TRACKS
SAVEM"BANKRBBAK",&H6FF0,&H76FF,&H6FF0

3.3.2 LOAD (SOURCE) DRIVE option

LOADM"BANKRBBAK"
POKE &H7008,1 ' LOAD DRIVE = 1
SAVEM"BANKRBBAK",&H6FF0,&H76FF,&H6FF0

3.3.3 Multiple drive copy option

LOADM"BANKRBBAK"
POKE &H7009,255 ' DRIVE 0 COPY = ON
POKE &H700A,255 ' DRIVE 1 COPY = ON
POKE &H700B,255 ' DRIVE 2 COPY = ON
POKE &H700C,0 ' DRIVE 3 COPY = OFF
SAVEM"BANKRBBAK",&H6FF0,&H76FF,&H6FF0

3.3.4 FORMAT with BACKUP option (USE WITH RSDOS 1.0/1.1 ONLY!)

LOADM"BANKRBBAK"
POKE &H6FF5,255 ' FBAK 0=OFF 255=ON
SAVEM"BANKRBBAK",&H6FF0,&H76FF,&H6FF0

***** WITH THIS OPTION, USE THE FOLLOWING PROCEDURE

CLEAR 200,&H6FEF ' MOVE STACK UNDER BANKRBBAK
LOADM"BANKRBBAK"
EXEC
3.4 BANKTEST CUSTOMIZATION

The BANKTEST program may be customized for testing smaller areas of memory, or different patterns, such as rotating bit patterns. An example of a BASIC program to perform a rotating bit test on addresses $0E00 to $0EFF in bank 6 is given below.

```
10 LOADM"BANKTEST"
20 POKE &H7008,6" BANK TO TEST
30 POKE &H7B13,0" TEST LOW MEMORY
40 POKE &H7B14,&H0E:POKE &H7B15,&H00" BEGL ADDR
50 POKE &H7B16,&H0E:POKE &H7B17,&HFF" ENDL ADDR
60 P = 1" FIRST BIT
70 FOR B = 0 TO 7 " BITS TO CHECK
80 POKE &H7B1C,P " PUT PATTERN
90 EXEC &H7B00 " RUN TEST
100 IF PEEK (&H7B0C) = 0 THEN GOSUB 150" ERROR?
110 P = P * 2" POWERS OF TWO, NEXT BIT
120 NEXT B " CHECK BITS 0-7
130 PRINT" TEST COMPLETE 
140 END ****************************
150 BAD = PEEK(&H7B0D)*256 + PEEK(&H7B0E) " FAIL ADDR
160 SAVDAT = PEEK(&H7B1D) " ORIGINAL DATA READ
170 BDATA = PEEK(&H7B1E) " BAD READ DATA
180 PRINT "ERROR AT ADDRESS ";HEX$(BAD)="/"BAD
190 PRINT " DATA ";HEX$(SAVDAT)="/"SAVDAT
200 PRINT " WRITE ";HEX$(P)="/"P
210 PRINT " READ ";HEX$(BAD)="/"BAD
220 RETURN ****************************
```

3.5 BANKRSPL - SPOOLER CUSTOMIZATION

The following locations may be useful if you desire to customize a spooler. The program loads at &H6FF0, and relocates itself to PDEST (&HFA00) through &F0FF when EXEC'd.

LOAD &H6FF0 ,END &H76FF ,EXEC &H6FF0

3.5.1 200K buffer in DISK SYSTEM (DOS 1.0):

We won't be using the RAM DISK, so the spooler can be moved to the end of ram in bank 0. Changing PDEST from &HFA00 to &HFC00 will re-locate the program to &HFC00 when it EXEC's. Note that this affects the COPY POKE, NUMCPY will now be &HFC02 (PDEST+2).
DISK BASIC 1.0 ends at &H7FF, so we can use from &H800 to &HFBFF in BANK 0 for our buffer. All of BANKS 1,2, and 3 may be used, except the area that the spooler driver program occupies ( &FC00 to &FEFF ), the buffer addresses are &H0000 to &FBFF. Here are the steps:

LOADM"BANKRSPL"
POKE &H7006, &HFC: POKE &H7007, 00" PDEST CHANGE
POKE &H7026, &HF: POKE &H702D, &HF" END0 ADDRESS
POKE &H702E, &HDB: POKE &H702F, &H00" BEG0 ADDRESS
POKE &H7030, &HFB: POKE &H7031, &HF" END1 ADDRESS
POKE &H7032, &H00: POKE &H7033, &H00" BEG1 ADDRESS
POKE &H7034, &HFB: POKE &H7035, &HF" END2 ADDRESS
POKE &H7036, &H00: POKE &H7037, &H00" BEG2 ADDRESS
POKE &H7038, &HFB: POKE &H7039, &HF" END3 ADDRESS
POKE &H703A, &H00: POKE &H703B, &H00" BEG3 ADDRESS
SAVEM"SPL200K", &H6FF0, &H76FF, &H6FF0" CUSTOM FILE

3.5.2 SPEED option

If you're using less than 1200 BAUD for your printer, you may find the SPoolER doesn't appear to be working. That's because the software "bit-banger" is spending nearly 16 milliseconds sending the character to the printer. One interrupt interval is 16.67 milliseconds, that's how often the spooler tries to send a character. As you can see, little time is left to do anything other than send data to the printer. The speed option multiplies the interrupt interval that the spooler checks the printer, making time available for other processing.

LOADM"BANKRSPL"
POKE &H7021, 10" 10 * 16.67ms = 167ms intervals
SAVEM"BANKRSPL", &H6FF0, &H76FF, &H6FF0

3.6 PAGER CUSTOMIZATION

The customization for this program is a poke for 256K or 512k versions or a poke for relocation of the driver in the lower 32K. The source code is included on the SOFTWARE PAC.

3.6.1 Memory size option

LOADM"PAGER"
POKE &H1247, 7" 3=256K 7=512K
SAVEM"PAGER", &H0E00, &H13BF, &H0E00

3.6.2 Relocation option

This is an example of re-locating the PAGER so it resides at &OE00-&OFFF. Note that it had to be offset loaded and saved because the installation code is normally loaded from &OE00 to &13BF.

CLEAR 200, &H6000" MOVE STACK DOWN
LOADM"PAGER", &H6000" LOAD IT HIGHER
POKE &H6E02, &H0E: POKE &H6E03, &H00 " &OE00
SAVEM"PAGER/LOW", &H6E00, &H73BF, &H6E00
3.7 PCOPYMOR CUSTOMIZATION

This program may be customized for the number of pages per bank, re-location (upper 32K), and maximum bank number to use (1 thru 7).

3.7.1 Memory size option

LOADM"PCOPYMOR"
POKE &H7006,7 ' 3=256K 7=512K
SAVEM"PCOPYMOR",&H6FF0,&H74FF,&H6FF0

3.7.2 Maximizing PCOPY & Relocation option

This is an example of maximizing the number of pages per bank, which requires re-locating the PCOPYMOR so it resides at the end of memory.

The driver is just under 256 bytes, we can use $0000-$FDFF in banks 1-7. Pages are $0600 bytes in size, so 42 pages will fit, ending at $FBFF. All we need to poke is the number of pages per bank (PGPBnk) and the maximum bank number (MAXBnk). When we’re done you’ll have 302 pages of graphics in a 512K system!

*** CAUTION:
This customization leaves no room for the SPOOLER or the RAMDISK.

LOADM"PCOPYMOR"
POKE &H6FF5,42 ' PAGES PER BANK
SAVEM"PCOPYMAX",&H6FF0,&H74FF,&H6FF0

3.8 For the USER with 512K

You may have RAMDISK "B" in the upper 192K, the PCOPYMOR patch in the lower 256K, and a 60K SPOOLER in the last bank of 64K! Use the above examples as a guide.

The PCOPYMOR program should be configured for 3 banks (MAXBnk = 3), no more than 40 pages per bank (PGPBnk = 40), and residing at $FB00 (PDEST = &HFB).

The SPOOLER should be configured to use only the last bank of 64K, use addresses $0000-$F7FF, and reside at $FA00.

The RAMDISK should be configured for 512K (MAXBnk = 7), no drive options for RAMDISK "A" and reside at $FD00. After you've customized them individually, load them only in the order listed here. I've given these customized files extensions of "/512".

LOADM"BANDISK/512" ' INSTALL USES ALL 7 BANKS
EXEC ' THE RAMDISK *DON'T USE "A"*
LOADM"BANKRSPL/512" ' NOW THE SPOOLER
EXEC ' THE SPOOLER
LOADM"PCOPYMOR/512" ' LAST
EXEC ' THE PCOPYMOR PATCH
4.0 PROGRAMMING INFORMATION

The "BANKER" partitions the expansion memory into banks of 64K and pages of 32K. The "BANKER" can be used to obtain banks of 32K from $8000-$FEFF (MAP TYPE 0), banks of 64K from $0000-$FEFF (MAP TYPE 1), or pages of 32K from $0000-$7FFF (PAGE BIT in SAM).

Additionally, the VDG display bank is separately programmable from the CPU memory bank. The CPU memory bank is independently programmable from the VDG display bank. The "BANKER" is programmed by PEEKing (reading) the address corresponding to the bits to be programmed. The "BANKER" overlays the "SAM" addresses which are POKEd (written). This prevents hardware address conflicts with add-ons like WORD-PAK, MULTI-PAK, and RS-232 PAK.

The VDG bank may be switched from BASIC using PEEK commands with no preparation of memory first. The CPU bank, while also switchable by BASIC using PEEK commands, may require preparation of memory by machine language programs prior to issuing the PEEK commands. This is because BASIC expects certain memory locations to have already been prepared for its use, if you switch banks without preparation, BASIC may (probably will) crash.

Study the bank switching addresses (APPENDIX A), and the BANKCOPY or PAGER source code to understand how to program the "BANKER" for your application.
5.0 ADVANCED USER INFORMATION

This section provides advanced user information to enable you to customize the software for your specific needs. J&R provides these as a courtesy to you, and is not responsible for any erroneous entries or effects thereof. Furthermore, J&R does not guarantee future versions will be mapped identically, although it is the author's intent to maintain them unchanged when possible.

5.1 Common Variables

There are some variables that appear in most of the machine language programs on your SOFTWARE PAC. These are needed to maintain compatibility with memory size options of the "BANKER" and various versions of software and operating systems.

5.1.1 MAXBNK

MAXBNK is a variable used by the software to initialize the program's memory allocation. It is usually set to 3 for 256K systems, and 7 for 512K systems. It may be less, however, to prevent driver installation from over-running banks that it does not use. In a 512k system using 128 pages of graphics and RAMDISK B, for example, the variable MAXBNK in PCOPYMOR should be set to three, and it should be 7 in BANKRDSK.

5.1.2 RESVEC

This is the RESET vector address for the RESET protection routine. Most "BANKER" programs use $03F0 to store this small amount of code. There are conflicts, however, TELEWRITER-64 uses this area. If this variable is set to $FFFF, the program's RESET protect routine will be bypassed. You may locate it anywhere within $0000-$FEFF (CAREFULLY, usually <$8000 ).

5.1.3 PDEST

This is the variable that holds the ultimate execution address of the program after it is moved by the installation program. It should always be >$8000 for BANKRDSK, BANKRSPL, and PCOPYMOR. For PAGER V2.0, it should be <$8000.

The installation program is position independent so that you can move it if it interferes with the PDEST location at which you want the driver to reside.
5.2 Software Variable Locations

The code (U) preceding a variable indicates a variable that it was intended to be user customized. The code (R) denotes a returned variable to be read by the user. Unmarked variables are fixed or calculated by the program.

5.2.1 BANKRDISK Version 2.0 - (the RAMDISK(S))

0001 0E00 ORG $6FF0
0002 6FF0 1600F1 START LBRA INSTAL
0003 6FF3 00 (U) FSTINI FCB $00 FAST INIT FLAG IFNE,FAST
0004 6FF4 00 (U) AUTOFL FCB $00 AUTO FLAG - IFNE, AUTO ON
0005 6FF5 00 (U) AUTINI FCB $00 IFNE, AUTO INIT IS ON
0006 6FF6 00 (U) AUTLDA FCB $FF DRIVE # TO LOAD A FROM
0007 6FF7 00 (U) AUTLDB FCB $FF DRIVE # TO LOAD B FROM
0008 6FFB 00 FDB $0000 RSV'D
0009 6FFA 00 FDB $0000 RSV'D
0010 6FFC 0000 FDB $0000
0011 6FFE 0000 PADDD FDB $0000
0012 7000 16001A PGMBEG LBRA RAMDISK
0013 7003 FD00 (U) PDEST FDB $FD00 DESTINATION MOVE
0014 7005 03F0 (U) RESVEC FDB $03F0 RESET VECTOR
0015 7007 00 BANK FCB $00
0016 7008 03 (U) MAXBNK FCB $03 512K=7 256K=3
0017 7009 03 (U) MAXDRV FCB $03 HIGHEST DRIVE
0018 700A 0000 DISKRET FDB $0000 SBKCON REHOOK
0019 700C 00 RDFLG FCB $00 IFNE, READ DISK
0020 700D 00 WRFILG FCB $00 IFNE,WRITE DISK
0021 700E 00 DDKRVL FCB $00 DISK DRIVE TO LOAD
0022 700F 00 (U) DRVA FCB $00 DRIVE A ASSIGNMENT #
0023 7010 22 (U) MXTKA FCB $34 MAX TRACK DRIVE A
0024 7011 12 MXBNK FCB $18 MAX SECTOR DRIVE A
0025 7012 01 BEGA FCB $01 START BANK FOR A
0026 7013 03 ENDA FCB $03 END BANK FOR A
0027 7014 FF (U) DRVB FCB $FF DRIVE B ASSIGNMENT #
0028 7015 22 (U) MXTKB FCB $34 MAX TRACK DRIVE B
0029 7016 12 MXBEB FCB $18 MAX SECTOR DRIVE B
0030 7017 04 BEGB FCB $04 START BANK FOR B
0031 7018 07 ENDB FCB $07 END BANK FOR B
0032 0005 TABSIZ EQU DRVB-DRVA
0033 7019 00 DRVAFIL FCB $00 0=DRVA FF=DRVB
0034 701A 00000C BANKTB FCB 0,14,28 BANK TABLE
0035 7A11 ZEND EQU *-1
0036 7A12 END

5.2.2 BANKCOPY Version 2.0

0001 0E00 ORG BANKCOPY
0002 0E00 DRG $0E00
0003 0E00 2002 BRA START
0004 0E02 07 (U) MAXBNK FCB $07 512K (3 FOR 256K VERSIONS)
0041 0E56 END
5.2.3 BANKRBAK Version 2.0

0001 OE00 ORG $6FF0
0002 6FF0 160070 START LBRA BAK
0003 6FF3 03F0 (U) RESVEC FDB $03F0 RESET PROTECT
0004 6FF5 00 (U) FBAK FCB $00 IFNE, FORMAT+BAK
0005 6FF6 0000 FDB $0000 RSV'D
0006 6FF8 0000 FDB $0000 RSV'D
0007 6FFA 0000 FDB $0000 RSV'D
0008 6FFC 0000 FDB $0000 RSV'D
0009 6FFE 0000 FDB $0000 RSV'D

* ORG $7000
0010 7000 160015 PGMBEGIN LBRA RWDSK
0011 7003 FD00 (U) PDEST FDB $FD00 DESTINATION MOVE
0012 7005 03 (U) MAXBNK FCB $03 3=256K 7=512K
0013 7006 22 (U) MAXTRK FCB 34 MAX TRACKS
0014 7007 12 MAXSEC FCB 18 MAX SECTOR
0015 7008 00 (U) LDRIV FCB $00 LOAD DRIVE (INPUT SOURCE)
0016 7009 FF (U) DRIVE0 FCB $FF IFNE, ON (OUTPUT DESTN)
0017 700A FF (U) DRIVE1 FCB $FF IFNE, ON (OUTPUT DESTN)
0018 700B 00 (U) DRIVE2 FCB $00 IFNE, ON (OUTPUT DESTN)
0019 700C 00 (U) DRIVE3 FCB $00 IFNE, ON (OUTPUT DESTN)
0027 7FF5 ZEND EQU *-1
0028 7FF6 END

5.2.4 BANKTEST Version 2.0

0001 OE00 ORG $7800
0002 7800 160170 BNKTST LBRA PUTMSG
0003 7803 160046 PGMBEGIN LBRA BTL
0004 7806 0200 VRBN FDB $0200
0005 7808 160015 LBRA BTH
0006 780B 00 (U) BANK FCB $00 BANK TO TEST
0007 780C FF (R) NOERR FCB $FF IFNE, NO ERROR
0008 780D FFFF (R) BAD FDB $FFFF BAD ADDRESS
0009 780F 0000 BEGA FDB $0000 DEG ADDRESS TO TEST
0010 7811 7FFE ENDA FDB $7FFE END ADDRESS TO TEST
0011 7813 00 (U) UPLD FCB $00 IFEQ, LO ELSE HI 32K
0012 7814 0000 (U) BEGL FDB $0000 BEG LOW ADDRESS
0013 7816 7FFE ENDL FDB $7FFE END LOW
0014 7818 8000 (U) BEGH FDB $8000 BEG HIGH
0015 781A FEFE (U) ENDH FDB $FEFE END HIGH
0016 781C FF (U) PATTRN FCB $FF PATTERN
0017 781D 00 (R) SAVDAT FCB $00 SAVED ORIGINAL DATA
0018 781E 00 (R) BDATA FCB $00 BAD DATA
0019 781F 03 (U) MAXBNK FCB $03 512K=7 256K=3
0020 7D31 ZEND EQU *-1
0021 7D32 END BNKTST
"BANKER" USER MANUAL: <c> 1985 by J&R ELECTRONICS

5.2.5 BANKRBSPL Version 2.0- (the SPooler)

0001 0E00  ORG $6FF0
0002 6FF0 1602DB  START LBRA INSTAL
0003 6FF3 03F0 (U) RESVEC FDB $03F0 RESET VECTOR
0004 6FF5 00  FCB $00
0005 6FF6 0000  FDB $0000 RSV"D
0006 6FF8 0000  FDB $0000 RSV"D
0007 6FFA 0000  FDB $0000 RSV"D
0008 6FFC 0000  FDB $0000 RSV"D
0009 6FFE 0000  FDB $0000 RSV"D

0010 7000 16004C  PGMBEGIN LBRA INSTBUF
0011 7003 16004F  LBRA XTRBUF
0012 7006 FA00 (U) PDEST FDB $FA00 DESTINATION MOVE
0013 7008 01 (U) NUMCPY FCB $01 NUMBER COPIES
0014 700A 03  INSDKN FCB $03 INSERT BANK
0015 700C 7E00  INSAADD FDB $7E00 INSERT ADDR
0016 700C 03  XTRBNK FCB $03 EXTRACT BANK
0017 700D 7E00  XTRADD FDB $7E00 EXTRACT ADDR
0018 700F 03  BEGBNK FCB $03 COPY BEGIN BANK
0019 7010 7E00  BEGBADD FDB $7E00 COPY BEGIN ADDR
0020 7012 03  ENDBNK FCB $03 COPY END
0021 7013 7E00  ENDBADD FDB $7E00 COPY END ADDR
0022 7015 00  BANK FCB $00 BANK #
0023 7016 00  FCB $00 RSV"D
0024 7017 000000  OLDIRQ FCB $0,0,0
0025 701A 000000  OLDPKH FCB $0,0,0
0026 701D 03 (U) MAXBNK FCB $03 MAX BANK
0027 701E 00  SPLOFF FCB $00 IFNE, SPOOL OFF
0028 701F 00  CPYFGB FCB $00 IFNE, COPY ON
0029 7020 00  SILENC FCB $00 IFNE, PRINTER SILENCED
0030 7021 00 (U) SPEED FCB $00 SPEED COUNT 0=OFF 255=MAX
0031 7022 00  HKFLAG FCB $00 IFNE, HOOKED
0032 7023 00  EMPTY FCB $00 IFNE, BUF EMPTY
0034 7024 01 (U) CTLLTAB FCB $01 RESTART CODE
0035 7025 02 (U) FCB $02 ON CODE
0036 7026 03 (U) FCB $03 OFF CODE
0037 7027 04 (U) FCB $04 CANCEL CODE
0038 7028 05 (U) FCB $05 COPY
0039 7029 06 (U) FCB $06 SILENT TOGGLE
0040 702A 04  FCB $04 RSV"D
0041 702B 04  FCB $04 RSV"D

0042 702C 00000000 (U) BUFTAB FDB $0000, $0000 BANK 0 (END,BEG)
0043 7030 00000000 (U) FDB $0000, $0000 BANK 1
0044 7034 00000000 (U) FDB $0000, $0000 BANK 2
0045 7038 FF77EE00 (U) FDB $FF77, $EE00 BANK 3
0046 703C 00000000 (U) FDB $0000, $0000 BANK 4
0047 7040 00000000 (U) FDB $0000, $0000 BANK 5
0048 7044 00000000 (U) FDB $0000, $0000 BANK 6
0049 7048 00000000 (U) FDB $0000, $0000 BANK 7
0050 704C 00  INSTPOS FCB $00 INS POSITION
0051 704D 00  XTRPOS FCB $00 XTR POSITION
0052 704E 00  COUNT FCB $00 SPEED COUNTER
0054 766F  ZEND END

******************************************************************************
5.2.6 PAGER Version 2.0

* THIS IS TEMPORARY CODE FOR INITIALIZATION

0001 0E00           ORG $0E00
0002 0E00 2005      START BRA START1
0003 7B00           START1 EQU $7B00
0004 0E02 7B00      (U) DESTIN EQU $7B00
0005 0E04 00        (U) MAPTYP FCB $00
0006 0E05 0000      OFFSET FDB $0000 OFFSET FROM HERE TO DEST

* ORG [PDEST,PCR]

* THIS IS THE PERMANENT CODE THAT MUST REMAIN FOR THE PAGE COMMAND

0007 1245 200F      FNDCMD BRA BB1

* 0008 1245

TRANSFER

0009 1247 03        (U) MAXBNK FCB $03
0010 1248 07        MAXPAG FCB $07
0011 1249 04        PAGMSK FCB $04
0012 124A 00        PAGE FCB $00
0013 124B 00        (R) PAGID FCB $00
0014 124C 00        NEWCMD FCB $00
0015 124D 0000      NCMADD FDB $0000
0016 124F 00        LASTCM FCB $00
0017 1250 00        NEWFUN FCB $00
0018 1251 0000      NFNADD FDB $0000
0019 1253 00        LASTFN FCB $00
0020 1254 0000      RETSTK FDB $0000
0021 13AF           ZEND EQU $-1
0022 0166           ZSIZ EQU CODFIN-CODE SIZE OF ACTUAL PAGER
0023 13BO           END

*******************************************************************************
"BANKER" USER MANUAL : <c> 1985 by J&R ELECTRONICS

5.2.7 PCOPYMDR Version 2.0

0001 0E00  ORG $6FF0
0002 6FF0 160105  LBRA INSTAL
0003 6FF3 03F0 (U) RESVEC FDB $03F0  RESET VECTOR
0004 6FF5 28 (U) PGPNK FCB 40  PAGES PER BANK
0005 6FF6 00  FCB $00  RSV'D
0006 6FF7 00  FCB $00  RSV'D
0007 6FF8 0000  FDB $0000  RSV'D
0008 6FFA 0000  FDB $0000  RSV'D
0009 6FFC 0000  FDB $0000  RSV'D
0010 6FFE 0000  FDB $0000  RSV'D

0011 7000 160016  PGMBEG LBRA PCOPY
0012 7003 FB00 (U) PDEST FDB $FB00  DESTINATION MOVE
0013 7005 00  BANK FCB $00  FOR MOVUP
0014 7006 03 (U) MAXBNK FCB $03  256K=3 512K=7
0015 7007 0000  MAXPAG FDB $0000  MAX PAGE#
0016 7009 00  SRBPNK FCB $00
0017 700A 0000  SRCLAT FDB $0000
0018 700C 0000  SRGBEG FDB $0000
0019 700E 0000  SRCEND FDB $0000
0020 7010 00  DSBPNK FCB $00
0021 7011 0000  DSBLAT FDB $0000
0022 7013 0000  DSBEG FDB $0000
0023 7015 0000  DSEND FDB $0000
0024 7017 0000  RETLAT FDB $0000
0025 7465 454E44  FCE "END"
0026 00F6  ZSIZ EQU PGMEND-PGMBEG
0027 7467  ZEND EQU $-1
0028 7468  END

--------------------------------------------------------------------------
## APPENDIX A: BANK SWITCH ADDRESSES

<table>
<thead>
<tr>
<th>PEEK ADDRESS</th>
<th>CPU BANK SELECTED</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&amp;HFFC0</td>
<td>0 &amp; 0</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFC1</td>
<td>1 &amp; 1</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFC2</td>
<td>2 &amp; 2</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFC3</td>
<td>3 &amp; 3</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFC4</td>
<td>4 &amp; 4</td>
<td>512k versions</td>
</tr>
<tr>
<td>$&amp;HFFC5</td>
<td>5 &amp; 5</td>
<td>512k versions</td>
</tr>
<tr>
<td>$&amp;HFFC6</td>
<td>6 &amp; 6</td>
<td>512k versions</td>
</tr>
<tr>
<td>$&amp;HFFC7</td>
<td>7 &amp; 7</td>
<td>512k versions</td>
</tr>
<tr>
<td>$&amp;HFFC8</td>
<td>0 &amp; 0</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFC9</td>
<td>0 &amp; 0</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFCA</td>
<td>0 &amp; 0</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFCB</td>
<td>0 &amp; 3</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFCC</td>
<td>0 &amp; 4</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFCD</td>
<td>0 &amp; 5</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFCE</td>
<td>0 &amp; 6</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFCF</td>
<td>0 &amp; 7</td>
<td>256k &amp; 512k versions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PEEK ADDRESS</th>
<th>VD6 BANK SELECTED</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&amp;HFFD0</td>
<td>0</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFD1</td>
<td>1</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFD2</td>
<td>2</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFD3</td>
<td>3</td>
<td>256k &amp; 512k versions</td>
</tr>
<tr>
<td>$&amp;HFFD4</td>
<td>4</td>
<td>512k versions</td>
</tr>
<tr>
<td>$&amp;HFFD5</td>
<td>5</td>
<td>512k versions</td>
</tr>
<tr>
<td>$&amp;HFFD6</td>
<td>6</td>
<td>512k versions</td>
</tr>
<tr>
<td>$&amp;HFFD7</td>
<td>7</td>
<td>512k versions</td>
</tr>
<tr>
<td>$&amp;HFFD8</td>
<td></td>
<td>Reserved for future expansion</td>
</tr>
<tr>
<td>$&amp;HFFD9</td>
<td></td>
<td>Reserved for future expansion</td>
</tr>
<tr>
<td>$&amp;HFFDA</td>
<td></td>
<td>Reserved for future expansion</td>
</tr>
<tr>
<td>$&amp;HFFDB</td>
<td></td>
<td>Reserved for future expansion</td>
</tr>
<tr>
<td>$&amp;HFFDC</td>
<td></td>
<td>Reserved for future expansion</td>
</tr>
<tr>
<td>$&amp;HFFDD</td>
<td></td>
<td>Reserved for future expansion</td>
</tr>
<tr>
<td>$&amp;HFFDE</td>
<td></td>
<td>Reserved for future expansion</td>
</tr>
<tr>
<td>$&amp;HFFDF</td>
<td></td>
<td>Reserved for future expansion</td>
</tr>
</tbody>
</table>

---

Page 24
APPENDIX B : BANKER SOFTWARE MEMORY MAP

* SYSTEM SOFTWARE AND HARDWARE ( RADIO SHACK OPERATING SYSTEM )

BANK 0 - MAXBNK $FB00-$FEFF Utility/permanent program storage
$FFC0-$FFDF PEEK programming addresses

* BANKRISK & BANKERBAK

BANK 0
$6FF0-$7FFF Initial load and preparation

* RAMDISK A :

BANK 0-3
$FD00-$FEFF Permanent usage in BANKS 0-3
BANK 1
$0000-$F8FF Tracks 0-13
BANK 2
$0000-$F8FF 14-27
BANK 3
$0000-$7DFF ($D7FF) 28-34 (39)

* RAMDISK B : ( 512k versions only )

BANK 0,4-7
$FD00-$FEFF Permanent usage in BANKS 0,4-7
BANK 4
$0000-$F8FF Tracks 0-13
BANK 5
$0000-$F8FF 14-27
BANK 6
$0000-$7DFF ($D7FF) 28-34 (39)
BANK 7
---- NOT USED ----

* BANKTEST

BANK 0
$7B00-$7DFF Initial load and preparation
BANK 0 - MAXBNK $FB00-$F9FF Transfers to test lower 32K

* PCOPYMDR

BANK 0
$6FF0-$7DFF Initial load and preparation
BANK 0 - MAXBNK $FB00-$F8FF Permanent usage in BANKS 0-3(7)
BANK 1
$0000-$EFFF PAGES 9-48
BANK 2
$0000-$EFFF 49-88
BANK 3
$0000-$EFFF 89-128
BANK 4
$0000-$F7FF CUSTOMIZING OPTION ( 512k only )
BANK 5
$0000-$F7FF "" "" ""
BANK 6
$0000-$F7FF "" "" ""
BANK 7
$0000-$F7FF "" "" ""

* BANKSPL

BANK 0
$6FF0-$7DFF Initial load and preparation
BANK 0
$FA00-$FCFF Permanent usage in BANK 0
BANK 3
$7E00-$F7FF "STOCK" 35 track RAM DISK compatible configuration
BANK 4
$0000-$F7FF CUSTOMIZING OPTION ( 512k only )
BANK 5
$0000-$F7FF "" "" ""
BANK 6
$0000-$F7FF "" "" ""
BANK 7
$0000-$F7FF "" "" ""
APPENDIX C: WARRANTY

WARRANTY

All products assembled and tested by J & R Electronics are warranted for a period of 90 days against defects not caused by user negligence, misuse or abuse. The aforesaid defects will be repaired free of charge, provided the product is returned, postpaid to J & R Electronics within the warranty period. J & R Electronics reserves the right to determine which repairs are unwarranted when user negligence, abuse, misuse or shipping damages are in question. This warranty is limited to the repair or replacement of defective parts. All software supplied by J & R Electronics are believed to be accurate as of the date of publication and are without warranty.

J & R Electronics assumes no responsibility for any damages caused by the buyer during assembly, installation or use of the Banker. J & R Electronics will not be liable for damages, direct or consequential, general or special, nominal or exemplary, resulting from use of any products or documentations supplied by J & R Electronics.

This manual, the Banker and any documentation pertaining to the Banker is the property of J & R Electronics. Reproduction by any means, electrical or otherwise, is strictly prohibited except by prior written permission from J & R Electronics.