```
==      ==  ========  ======
===    ===  ========  =======
====  ====     ==     ==   ==
== ==== ==     ==     =======
==  ==  ==     ==     ======
==      ==     ==     ==
==      ==     ==     == INC.
```

coco

"ROML"

ROM PAK Loader Program

Software for the
Radio Shack Color Computer
and the TDP-100.

Rev 1.0
Oct. 26, 1982

Micro Technical Products, Inc.
123 N. Sirrine, Suite 106
Mesa, Arizona   85201 USA
Phone: 602-834-0283

Introduction

The Radio Shack Color Computer is an excellent system. One of the most objectionable aspects, however, is the need to unplug the disk controller whenever you want to use one of your ROM PAKs. Micro Technical Products' ROML program can open a whole new world to your Color Computer. ROML will allow you to do the following:

If you have a Radio Shack disk and 32K or less of RAM:

> Load from disk or tape ANY machine language program -- even programs that are incompatible with the disk system. Now you can play all of your old tape version games without having to unplug your disk. In fact, you can save these programs on your disk for even more convenience.

If you have a Radio Shack disk and 64K of good RAM:

> Load from tape or disk ANY machine language program as described above. Additionally, ROML allows you to load ROM PAK software from tape or disk and execute it WITHOUT having to remove your disk controller. Please note -- ROML is for your convenience in loading your own ROM PAKs -- not pirated copies of someone else's! Also, some ROM PAKs may require RAM write protection in order to function correctly. Refer to the RAM write protect section of this manual for more details. If you are not sure whether or not your system has 64K RAM installed, then read the enclosed copy of Frank Hogg's article.

If you have 64K of good RAM but don't have a disk:

> Load ROM PAK software from tape as described above.

How to load ROML

Loading the tape version:

1) Turn on your computer.
2) Place the ROML tape in your recorder and depress the PLAY button.
3) Enter 'CLOADM "ROML"' and press ENTER.
4) When the tape has finished loading rewind and remove it from the recorder.
5) Enter 'EXEC' and press ENTER and ROML will begin.

Loading the disk version:

1) Turn on your computer.
2) Place the ROML disk in any disk drive.
3) Enter 'LOADM "ROML"' if the disk is in drive zero, otherwise enter 'LOADM "x:ROML"' where 'x' is the correct drive number. Then press ENTER. ROML will start execution automatically after it is loaded.

How to use ROML

When ROML begins execution it clears the CRT screen, outputs the MTP logo and displays the following:

```
PROGRAM NAME:
```

You should now enter the name of the program you wish to have loaded. The name may be a maximum of eight characters long. If you make a typographical error, the back-arrow key may be used to delete incorrect characters. DO NOT attempt to specify a file name extension -- ROML will always use a default of '.BIN'. This is inconsequential for tape operation since tape files do not have extensions, but, does mean that all disk files to be loaded via ROML MUST have extensions of '.BIN'.

For tape loading only, if you wish to load the 'next' program on the tape, it is not necessary to specify the file name -- merely press ENTER.

Once you have entered the desired file name (or not, if loading from tape) press ENTER and ROML will ask:

```
LOAD FROM (T)APE or (D)ISK?
```

Before proceeding, make sure that the correct tape or disk is now installed. If you are going to load from tape place the tape in the recorder and depress the PLAY button. If you are going to load from disk, insert the disk into the SAME drive from which ROML was loaded.

Once the proper tape or disk has been installed, press either the 'T' key or the 'D' key, depending upon whether you want to load the file from the tape or the disk, respectively. You do not need to press the ENTER key. As soon as you press either 'T' or 'D' the loading will commence.

Information displayed

If loading from tape the screen will clear and a blinking 'S' will appear in the upper left corner to indicate that the file is being Searched for. When the correct file is found the blinking letter will change into an 'F'.

If loading from disk the screen will not change until the file is completely loaded.

After the file has been loaded ROML will display:

```
DISK ROM DISABLED.

PROGRAM LOADS:
FROM         $XXXX
THROUGH      $YYYY
EXECUTES AT $ZZZZ
```

Note that the disk ROM has been disabled. This is because the purpose of ROML is to load programs which are incompatible with the disk system, and hence, the disk ROM must be gotten out of the way. But, don't fear -- it has only been disabled with software and when you press Reset your system will return to normal.

The dollar signs ($) preceeding the X's, Y's and Z's indicate that the numbers are in hexidecimal. The numbers XXXX and YYYY indicate the range of memory address into which the program is supposed to load. The number ZZZZ is the memory address at which the program is to begin execution.

Starting the loaded program

If the loaded program is to reside in the lower 32K of RAM then ROML will display:

```
HIT ANY KEY TO START PROGRAM
```

Pressing any key will start the program executing.

If the loaded program is to reside in the upper 32K of RAM (such as all ROM PAK software does) then ROML will display:

```
DISABLE RAM WRITE PROTECT
HIT ANY KEY TO PROCEED
```

If you have installed the RAM write protect option, this is a reminder for you to turn it off so that the program may be loaded where it belongs. Press any key and after a brief time ROML will display

```
** 64K RAM SELECTED **

ENABLE RAM WRITE PROTECT
HIT ANY KEY TO START PROGRAM
```

The '64K' message is to let you know that your computer has been switched into the 64k RAM mode. If you have installed the RAM write protect option you may now turn it back on. Pressing any key will then start the program executing.

Error messages and recovery

There are two basic types of errors which can occur; recoverable and non-recoverable.

1) Recoverable errors:

System errors such as NE
These errors are displayed in the form

```
XX ERROR!!
```

where XX is replaced with a two letter code indicating the error. Refer to your BASIC programming manuals for an explanation of these errors.

After the error is displayed the following message is then output

```
PRESS ANY KEY TO START OVER
```

Pressing any key will then return you to the request to enter a file name.

2) Non-Recoverable errors:

BAD FILE TYPE
Indicates that you tried to load a file which was NOT a machine language program.

NULL FILE
Indicates that you tried to load a file which contained nothing.

RAM BAD AT $XXXX
Indicates that a bad RAM location was found which would prohibit the loaded program from executing correctly. $XXXX is the failing address in hexidecimal.

FILE TOO LARGE
Indicates you tried to load a program to large to fit into your system's available RAM. Remember that ROML takes up some RAM too. So even if the program fits into your system without ROML, it may not fit with ROML.

When ever any of these errors occur the message

```
NON-RECOVERABLE ERROR!!
  ==> PRESS RESET <==
```

will be displayed. The only way out of the error situation is to press the Reset switch on the back of your computer.

RAM write protect

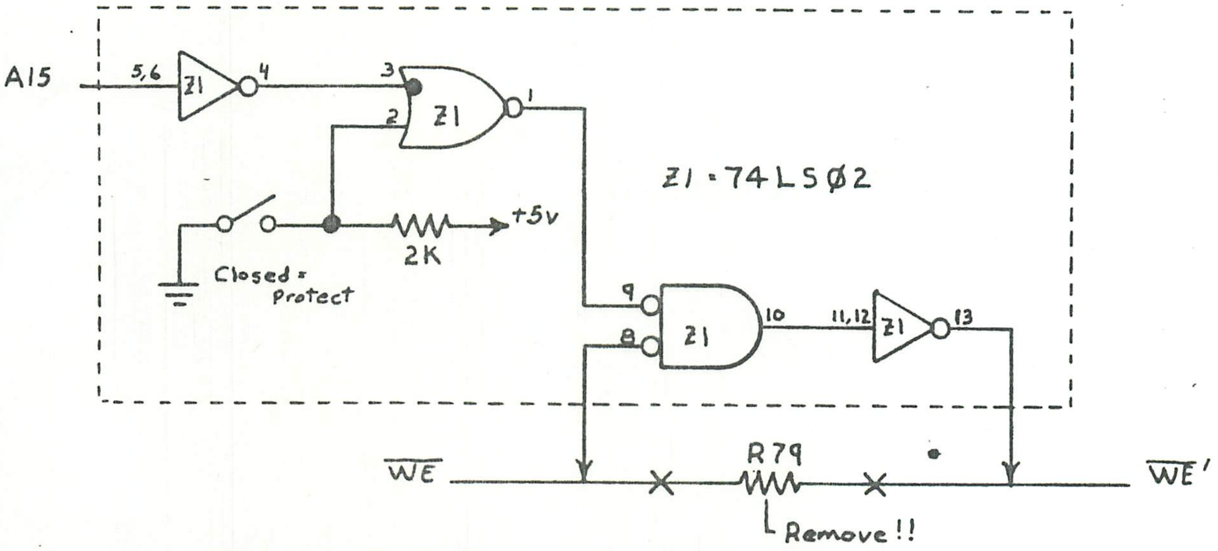
The Color Computer, when used with 64K RAM, is an amazingly powerful system. It's normal mode of operation, however, is to use only 32K RAM from $0000 through $7FFF. From $8000 on up is ROM and system I/O hardware. ROM, even by its own definition (Read Only Memory), is write protected. That is, there is nothing the microprocessor or a programmer can do to alter its contents. Sometimes this is a very desirable feature -- it keeps some run away program or a careless programmer from crashing the system.

When the Color Computer is switched into its 64K RAM mode this protection disappears, as RAM may be altered at will. When ROML loads a copy of a ROM PAK into your system it must switch into the 64K RAM mode. Unfortunately, some ROM PAKs are dependent upon memory above $8000 being ROM -- or at least being protected.

The only ROM PAK that we are aware of that requires this protection is SPECTACULATOR. When it first begins its execution it sizes available RAM. Without the RAM being write protected, and since we are in the 64K RAM mode, it thinks that available RAM exists throughout itself which causes it to crash.

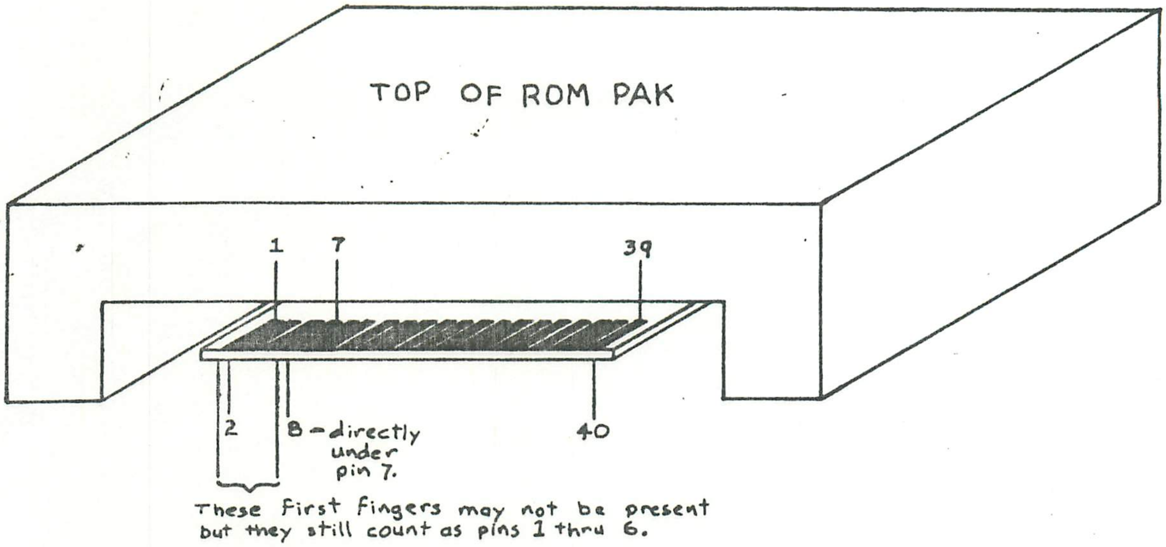
If you want to run your ROM PAK version of SPECTACULATOR (or any others that may exhibit this symptom) via ROML, then you have two choices: 1) Modify the SPECTACULATOR program so that it no longer sizes memory in this manner, which should not be too difficult to do once you have saved it on disk, or 2) Add the RAM write protect option to your computer.

The following schematic is for a circuit which you can build to perform the necessary RAM write protect. It has no other effects on the system.



Saving ROM PAKs to tape or disk

1) Cover pins 7 and 8 of the edge connector in the ROM PAK with a small piece of Scotch tape.


TOP OF ROM PAK
1
7
39
2
8 - directly under pin 7.
40
These first fingers may not be present but they still count as pins 1 thru 6.

2) With your computer turned off, plug the ROM PAK into the expansion port on the right side of the computer.

3) Turn your computer on. It should come up in BASIC the same as it would if the ROM PAK wasn't plugged in.

4) Find the ending address for the particular ROM PAK in the table on the following page. If it is not listed there then you can use $DFFF, which should be adequate for most all programs. If this still does not work you might try $E7FF.

5) Install a blank tape in the recorder and press the RECORD and PLAY buttons.

6) Enter 'CSAVEM "progname",&HC000,&Hendadr,&HC000' and press ENTER. 'progname' is to be the name you want to assign to the file, and 'endadr' is to be the ending address from the table on the next page.

7) When the computer is done saving to the tape, press the STOP button on the recorder.

8) Turn off the computer and remove the ROM PAK. Plug the disk controller back in (if you have one).

9) Use the TAP2DSK program to transfer the program from the tape to a disk if you desire.

ROM PAK addresses

The following table lists the ending address for several of Radio Shack's ROM PAKs. They all have a beginning address and an execution address of $C000.

Backgammon	$CFFF
Bingo Math	$CFFF
Bustout	$D7FF
Checkers	$CFFF
Chess	$DFFF
Color Cube	$CFFF
Color File	$CFFF
Diagnostics	$C7FF
Dino Wars	$D7FF
Edit-Asm	$E7FF
Football	$D7FF
Handyman	$CFFF
Music	$CFFF
Personal Fin.	$DFFF
Pinball	$CFFF
Polaris	$D7FF
Poltergeist	$DFFF
Quasar Command	$CFFF
Roman Checkers	$DFFF
Skiing	$CFFF
Scripsit	$DFFF
Space Assult	$CFFF
Spectaculator	$DFFF
Spectrum Anal.	$C7FF
Typing Tutor	$CFFF

64K RAM

The article on the following pages was written by Frank Hogg of Frank Hogg Laboratory, Inc. It was published in "68 Micro Journal" and "Color Computer News". It is reprinted here with permission from Frank Hogg Laboratory, Inc.

32K FOR FREE!

"How to run Pascal, C, and Cobal, not to mention XFORTH, esther, and spelltest, on the TRS-80 COLOR COMPUTER"

Someday, as the Honeywell advertisement would say, integrated circuit processing will become so inexpensive that computer memory will be available for free.

That day is today, for owners of the Radio Shack TRS-80 COLOR COMPUTER.

The story begins with my early production model (with a 3-digit serial number) of the 4k color computer. Its logic board had some extra wires and things on it, indicating that the design was not quite perfected when it was produced. I heard that radio shack would replace the board with a newer version if I purchased their 32k ram upgrade for $149.00, so I decided to give it a try.

When I took the computer to the local computer center, I was told that the upgrade would only cost $99.00. I did learn, however, that radio shack is unwilling to work on a computer which has a modification in it, even if the mod is electrically disconnected. They did complete the upgrade, and indeed they did install a new logic board, containing eight memory chips with unrecognizable part numbers on them.

Various rumors have been circulating about how the 32k upgrade is accomplished. it is not done by piggybacking 16k rams! Neither is it done by installing 32k rams, as radio shack contends.

The 32k dynamic ram was actually only available for a short time. These parts were actually attempts at 64k parts that were only half-good, or they had some bad bits in one half or the other. The 32k upgrade was originally designed to take advantage of these parts - a jumper exists on revision E of the color computer circuit board to select which half of the 64k dynamic ram is accessed.

Since then, memory manufacturers have learned how to produce 64k chips with sufficient yield to drive the cost lower than you or I, or radio shack, expected. these chips are available by mail order, in small quantities, for less than $12.00 each. radio shack can certainly buy them in quantity at a lower price.

The astute reader will have guessed the punch line by now. The 32k color computer actually contains 64k rams! I am not in a position to guarantee this, of course, but so far it seems to be the case. I will now tell you how the "other 32K" might be useful to you.

USING THE FULL 64K RAM.

None of the versions of radio shack color basic know how to use the other 32k. As a matter of fact, this memory is not available to the cpu at all in an unmodified color computer. This is due to an easily correctible omission in the design of the computer.

The dynamic memory in the color computer is controlled by a chip known as the sam, or synchronous address multiplexer. The sam bears the Motorola part number 6883, or 74ls783. The sam takes care of refreshing the rams and interlaces the access cycles of the cpu and the video display so that no "specks" occur on the screen. The sam must be programmed differently for 4k, and 16k, and 64k rams. (this is why color basic 1.1 was written - version 1.0 didn't know about 64ks.) the sam also provides address decoding for the three roms, as well as the i/o hardware.

As the sam was being designed, Motorola considered the possibility that it might be useful in systems which did not use rom, but might want to use 64k of ram (minus 256 bytes for I/O, etc.) For this reason the selection of rom in the sam is programmable. If you whisper the right thing to the sam (POKE &HFFDF,anything), the roms will go away, at least in theory, leaving behind nearly 32k of clean, untouhed ram.

Well, we need a more sophisticated theory, because it doesn't quite work. The sam will still try to select the roms if the cpu writes to those addresses, regardless of how it is programmed. I guess motorola must have thought that this decoding might be used for something -

clearly it wouldn't hurt, since the system designer would have to provide logic to prevent the roms from being turned on in a write cycle anyway. (the rams are "selected" for write purposes all the time.)

Radio shack, on the other hand, didn't see things the same way; they figured they would avoid writing to that area, so no problems would result. As a matter of fact, the first thing color basic does (after programming the sam) is to test the memory from zero until it finds a byte that won't write. when this test hits address &H8000, the cpu tries to write the roms with exactly the opposite data they contain, and at the same time the roms are reading - resulting in two different chips trying to put different data onto the same bus at the same time.

The real tragedy is that a few unused nor gates exist on the color computer circuit board. You only need one of these to solve this problem. (radio shack designers - take note.)

THE MODIFICATION IS REVERSIBLE.

One of the extra nor gates must be connected into the circuit as shown in figure 1. this modification disables the selection circuitry (G2B high) if a write is attempted (r/w low) and a rom is addressed (r/w low). If you have some experience with fine soldering, you can accomplish this modification in a reversible fashion, allowing you to run to radio shack if your color computer breaks. Warning - you must remove that nasty sticker on the back, thus voiding your warranty (if you're still covered), to get inside.

The procedure is as follows. remove the case and the top of the rf shield, on the right behind the keyboard. you should be able to find the ic's and TP1 as shown in figure 2. They are also marked on the board. U11 is a 74LS138, and U29 is a 74LS02.

You may wish to obtain a new 74LS138 and a 74LS02, so you can save the "originals" for a rainy day. in reality radio shack probably doesn't remember what brand of ic it put in your computer, but precautions are cheap. Anyway, carefully remove those two ic's. (they are not especially sensitive to static.) bend pins 4, 5 and 6 of the 74LS02 up in the air, as shown in figure 3. They must be almost straight up so they don't touch the shield. similiarly disfigure pin 5 of the 74LS138. (be gentle!)

Next, using a short piece of 30-gauge wire, connect pin 6 of the 74LS02 to pin 8. pin 8 must plug back in, so try not to get solder down on the pin - you should tack the wire on the very top of the pin, where it enters the package. if it doesn't come out right, buy another 74LS02 - it costs much less than a new computer.

You can do the rest of your soldering either before or after you plug the chips back in; use your own judgement. Pin 4 of the 74LS02 must be connected to pin 5 of the 74LS138, and pin 5 of the 74LS02 must be connected to TP1. I recommend that you do not solder to TP1. just use a wire wrap tool to wrap the wire around the pin, so it can be pulled off.

After you have reinstalled the ic's, the wiring should appear as in figure 4. check carefully for shorts!

At this point you can turn on the computer and do a "PRINT MEM." if it says the usual number, all is probably well, so put it back together.

TESTING YOUR NEW FREE MEMORY.

The extended color basic program in listing 1 will test the ram which you have just made available. save it before you try to run it, because if you mistype one of those data statements, anything can happen. the program will take about a minute to get set up, after which it will print "ok" if your memory is good. if you do have a problem, it will tell you the address and the data read from the ram, compared to what was expected. I would like to hear from you if you do find errors. If the errors occur in only one or two bit positions, they can be fixed with one or two 64k rams, for one or two ten-dollar bills. No big deal.

WHAT DO YOU DO WITH IT?

You now own a computer with almost 88k of memory, in a box no bigger than a typewriter. this fact alone may be enough for some of you. However, a large collection of software exists which can now be run on your computer.

The most important item in this collection is the popular FLEX operating system. (FLEX is a trademark of Technical Systems Consultants, Inc.) Frank Hogg Labs is developing a package which will allow FLEX to be run on the 32k color computer, with the radio shack disc system, and the modification described above. FLEX will reside in memory at addresses &HC000-&HDfff, as always. addresses 0-&HBFFF will be available for user programs. addresses &HE000-&HFFEF will be available for utility programs. (we are working on an enhanced display package, using hi-res graphics to simulate a 41-by-24 screen. that's better than an apple!)

With FLEX you have a whole cosmos of software available to you. besides the items mentioned in the subtitle, there are basic compilers, business programs, adventure games, assemblers and text editors, word processing software, machine-language debug programs, disc system diagnostic packages, and too much more to mention. FLEX is an excellent system which is widely supported.

another way to do it.
=====================

Frank Hogg Labs is working on a neater way to do the hardware modification. This would be a small printed circuit board which replaces U11, as shown in figure 5. A jumper wire would clip onto TP1 to complete the circuit.

Figure 6 shows the logic diagram of the board. This circuit is identical to the other method, because U11 pin 3 is connected to U29 pin 8 inside the color computer. one warning - we have not yet tested this board.

Figure 7 hows the layout of the board. Figure 8 shows the placement of components on the board.

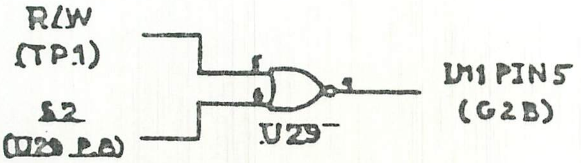


Figure 1: Modification to TRS80CC for 64K RAM.

SUMMARY

The 32k upgrade of the radio shack trs-80 color computer is accomplished by installing 64k dynamic ram chips. With a simple, reversible modification, nearly all 64k of this ram can be utilized. A package is being developed which will allow the FLEX operating system to be run on the modified 32k TRS-80CC with disc. You can do a lot of stuff with that.

This article was prepared, using a preliminary version of the FLEX package, on a color computer.

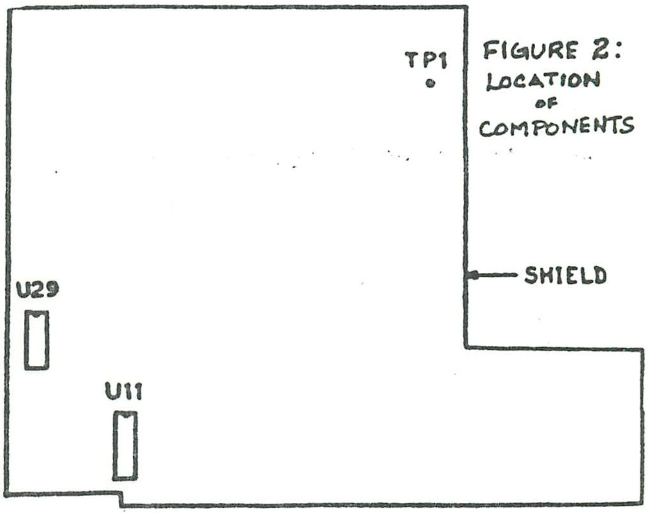


FIGURE 2: LOCATION OF COMPONENTS

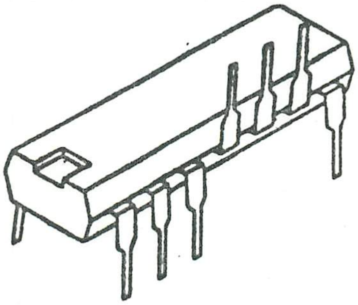


FIGURE 3. MODIFIED 74LS02 PACKAGE

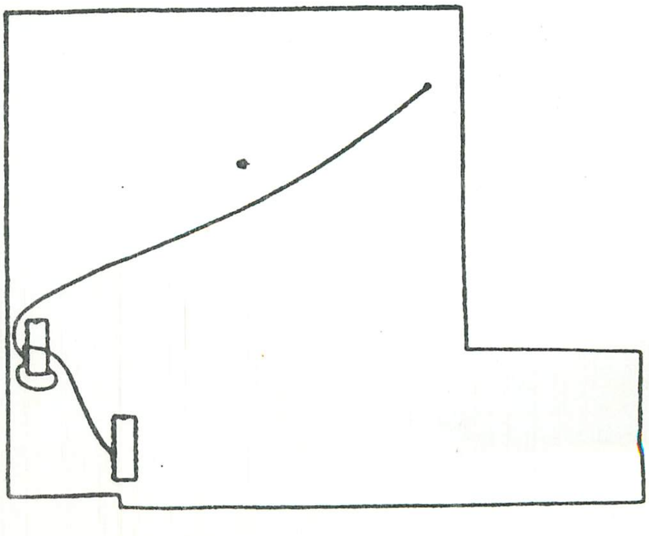


FIGURE 4. INSTALLED MODIFICATION

```
' THIS PROGRAM TESTS THE
' MEMORY CHIPS IN A MODIFIED
' 32K TRS80 COLOR COMPUTER
' FOR FUNCTIONALITY. THE
' TEST DETERMINES WHETHER
' FULL 64K RAM CHIPS WERE
' USED IN THE RAM UPGRADE.
'
' REQUIRES EXTENDED COLOR
' BASIC.
'
' REQUIRES ABOUT 65 SECONDS
' TO SET UP THE MACHINE
' LANGUAGE PROGRAM BEFORE
' RUNNING.
'
' RESERVE RAM
'
CLEAR 256,&H3FFB
'
' SET UP MACHINE LANGUAGE
' PROGRAM
'
GOSUB 1660 : SA=H
GOSUB 1660 : EA=H
GOSUB 1660 : E0=H
GOSUB 1660 : E1=H
'
' SA=START ADDRESS
' EA=END ADDRESS
' E0=ENTRY POINT 0
' E1=ENTRY POINT 1
'
FOR A=SA TO EA
GOSUB 1660
POKE A,H
NEXT A
'
DEFUSR0=E0
DEFUSR1=E1
'
' TEST THE MEMORY
'
X=USR0(0)
'
' CHECK FAILURE ADDRESS
'
FA=PEEK(&H3FFC)*256
 +PEEK(&H3FFD)
'
IF FA=0 THEN 760 :' END TEST
'
' REPORT FAILURE
'
DW=PEEK(&H3FFE) :' DATA WRIT
DR=PEEK(&H3FFF) :' DATA READ
'
PRINT "AT ADDRESS ";HEX$(FA)
PRINT "WROTE ";HEX$(DW);
 "; READ ";HEX$(DR)
```

```
 680 '
 690 ' RESUME TEST
 700 '
 710 X=USR1(0)
 720 '
 730 GOTO 550
 740 '
 750 '
 760 ' END OF TEST
 770 '
 780 END
 790 '
 800 ' MACHINE LANGUAGE PROGRAM
 810 '
 820 DATA 4000,4054
 830 DATA 4000,4046
 840 '
 850 ' THE MACHINE LANGUAGE
 860 ' MEMORY TEST PROGRAM IS AS
 870 ' FOLLOWS:
 880 '
 890 ' E0   ORCC #$50 DIS. INTS.
 900 '      STA $FFDF MAP TYPE 1
 910 '      LDX #$4100
 920 ' P1   CLR ,X+ CLEAR TO ZERO
 930 '      CMPX #$FF00
 940 '      BNE P1
 950 '      CLR $3FFE DW
 960 '      LDX #$4100
 970 ' P2   LDA ,X
 980 '      BNE RE REPORT ERROR
 990 ' P2A  LDA #$FF
1000 '      STA ,X+
1010 '      CMPX #$FF00
1020 '      BNE P2
1030 '      COM $3FFE DW
1040 '      LDX #$4100
1050 'P3   LDA ,X
1060 '     CMPA #$FF
1070 '     BNE RE REPORT ERROR
1080 'P3A  LDA #$00
1090 '     STA ,X+
1100 '     CMPX #$FF00
1110 '     BNE P3
1120 '     LDX #$0000 END TEST
1130 'RE   STX $3FFC FA
1140 '     STA $3FFF DR
1150 '     STA $FFDE MAP TYPE 0
1160 '     ANDCC #$AF ENBL INTS.
1170 '     RTS
1180 '
1190 ' RESUME TESTING
1200 '
1210 'E1   ORCC #$50 DIS. INTS.
1220 '     STA $FFDF MAP TYPE 1
1230 '     LDX $3FFC FA
1240 '     TST $3FFE DW
1250 '     BEQ P2A
1260 '     BRA P3A
```

```
1270 '
1280 ' ACTUAL CODE
1290 '
1300 DATA 1A,50
1310 DATA B7,FF,DF
1320 DATA 8E,41,00
1330 DATA 6F,80
1340 DATA 8C,FF,00
1350 DATA 26,F9
1360 DATA 7F,3F,FE
1370 DATA 8E,41,00
1380 DATA A6,84
1390 DATA 26,21
1400 DATA 86,FF
1410 DATA A7,80
1420 DATA 8C,FF,00
1430 DATA 26,F3
1440 DATA 73,3F,FE
1450 DATA 8E,41,00
1460 DATA A6,84
1470 DATA 81,FF
1480 DATA 26,0C
1490 DATA 86,00
1500 DATA A7,80
1510 DATA 8C,FF,00
1520 DATA 26,F1
1530 DATA 8E,00,00
1540 DATA BF,3F,FC
1550 DATA B7,3F,FF
1560 DATA B7,FF,DE
1570 DATA 1C,AF
1580 DATA 39
1590 '
1600 DATA 1A,50
1610 DATA B7,FF,DF
1620 DATA BE,3F,FC
1630 DATA 7D,3F,FE
1640 DATA 27,C6
1650 DATA 20,D9
1660 '
1670 ' READ A HEX NUMBER TO H
1680 '
1690 READ A$
1700 LZ=LEN(A$)
1710 H=0
1720 IF LZ<=0 THEN RETURN
1730 C$=LEFT$(A$,1)
1740 FOR I=0 TO 15
1750 IF (I=0)AND(C$="0")
     THEN 1790
1760 IF C$=HEX$(I) THEN 1790
1770 NEXT I
1780 RETURN
1790 H=H*16+I:LZ=LZ-1
1800 A$=RIGHT$(A$,LZ)
1810 GOTO 1730
```

```
1000 ' THIS PROGRAM MOVES
1010 ' RADIO SHACK COLOR BASIC,
1020 ' EXTENDED COLOR BASIC, AND
1030 ' DISK EXTENDED COLOR BASIC
1040 ' FROM ROM TO RAM IN A
1050 ' MODIFIED 32K TRS80
1060 ' COLOR COMPUTER. WHEN THE
1070 ' PROGRAM TERMINATES, BASIC
1080 ' WILL BE RUNNING IN RAM
1090 ' AND THE ROM WILL BE
1100 ' DISABLED.
1110 '
1120 ' REQUIRES EXTENDED COLOR
1130 ' BASIC VERSION 1.0.
1140 '
1150 ' REQUIRES ABOUT 20 SECONDS
1160 ' TO SET UP THE MACHINE
1170 ' LANGUAGE PROGRAM
1180 ' BEFORE RUNNING.
1190 '
1200 ' RESERVE RAM
1210 '
1220 CLEAR 256,&H7EFF
1230 '
1240 ' SET UP MACHINE LANGUAGE
1250 ' PROGRAM
1260 '
1270 GOSUB 2140 : SA=H
1280 GOSUB 2140 : EA=H
1290 GOSUB 2140 : EP=H
1300 '
1310 ' SA=STARTING ADDRESS
1320 ' EA=END ADDRESS
1330 ' EP=ENTRY POINT
1340 '
1350 FOR A=SA TO EA
1360 GOSUB 2140
1370 POKE A,H
1380 NEXT A
1390 '
1400 DEFUSR0=EP
1410 '
1420 ' RUN THE PROGRAM
1430 '
1440 X=USR0(0)
1450 '
1460 PRINT "BASIC IS NOW IN RAM"
1470 PRINT "ROMS ARE DISABLED."
1480 PRINT
1490 '
1500 FOR I=1 TO 1000
1510 NEXT I
1520 '
1530 ' CHANGE SPELLING OF PRINT
1540 '
1550 POKE &HAA79,ASC("W")
1560 POKE &HAA7A,ASC("R")
1570 POKE &HAA7B,ASC("I")
1580 POKE &HAA7C,ASC("T")
1590 POKE &HAA7D,ASC("E")+&H80
1600 '
1610 PRINT " THE SPELLING OF"
1620 PRINT " 'PRINT' HAS NOW"
1630 PRINT " BEEN CHANGED. "
1640 PRINT
1650 PRINT " LIST 1610-1670 TO"
1660 PRINT " SEE FOR YOURSELF:"
1670 PRINT
1680 '
1690 FOR I=1 TO 1000
1700 NEXT I
1710 '
1720 LIST 1610-1670
1730 '
1740 END
1750 '
1760 ' MACHINE LANGUAGE PROGRAM
1770 '
1780 ' SA, EA, EP
1790 '
1800 DATA 7F00, 7F19, 7F00
1810 '
1820 ' THE MACHINE LANGUAGE
1830 ' PROGRAM TO MOVE BASIC TO
1840 ' RAM IS AS FOLLOWS:
1850 '
1860 '
1870 ' EP   ORCC #$50 DIS. INTS.
1880 '      LDX #$8000 1ST ADDR.
1890 ' LOOP LDA ,X
1900 '      STA $FFDF MAP TYPE 1
1910 '      STA ,X+ IN RAM!
1920 '      STA $FFDE MAP TYPE 0
1930 '      CMPX #$FF00 LAST +1
1940 '      BNE LOOP
1950 '      STA $FFDF MAP TYPE 1
1960 '      ANDCC #$AF ENBL INTS
1970 '      RTS
1980 '
1990 ' ACTUAL CODE
2000 '
2010 DATA 1A,50
2020 DATA 8E,80,00
2030 DATA A6,84
2040 DATA B7,FF,DF
2050 DATA A7,80
2060 DATA B7,FF,DE
2070 DATA 8C,FF,00
2080 DATA 26,F1
2090 DATA B7,FF,DF
2100 DATA 1C,AF
2110 DATA 39
2120 '
2130 '
2140 ' READ A HEX NUMBER TO H
2150 '
2160 READ A$
2170 LZ=LEN(A$)
2180 H=0
2190 IF LZ<=0 THEN RETURN
2200 C$=LEFT$(A$,1)
2210 FOR I=0 TO 15
2220 IF I<>0 THEN 2240
2230 IF C$="0" THEN 2270
2240 IF C$=HEX$(I) THEN 227
2250 NEXT I
2260 RETURN
2270 H=H*16+I : LZ=LZ-1
2280 A$=RIGHT$(A$,LZ)
2290 GOTO 2200
```