

TUTORIAL

by Richard E. Esposito and Raymond W. Rowe

DATAU4E2F2
R3D2G1L2,BU1F1R
R2F1D4G1L2,R3L
R2D2U3R3,U5E1R1F1H1
U6D3R3U3D6,BR1R2L1U
U5D3R1E3G3F3,U6D6F
U1F1E2E1U4H1L2G1D4,U6F
D6L3R3H1F2,U6R3D3L3R1F
U2L3U2E1R1F1,BR1U6R2
BU2U4D4F2E2U4,U6D6R
U4U1D1G2H2U1D1F4D1,B
2U1,R3L3U2E3U1L3,BU
U1D4,BR1R2L1U6G1,R3
U1F1R1E1U2L2R2U2H1L
U4U3,BU1F1R1E1U1H1
U1F1H1L1G1D4F1R1E1
U4,BU1U4E1R1F1D4G1
U1E1U4H1L2G1D1F1P
EK(65280)=12
THENJ=1:SOUND2
SCREEN1,SM:LINE
6),PSET,B:GOSUB10:
U8L2D8R2U8L4H2U12R2D1
0R2D10U10R2D10U10R2D12G2L2":P

64K Modification Revisited

Get the scoop on upgrading your CoCo no matter what version you have.

Still struggling with 32, 16, or even 4K of memory in your Color Computer? In most cases, increasing your CoCo's memory capacity is easy enough for anyone with the confidence to try.

Some upgrading techniques, however, are too involved for novices. Opening your Color Computer voids its warranty, and if you make a mistake such as cutting a land on a printed circuit (PC) board, your local Radio Shack technician might refuse to service it. Each upgrade method in this article is rated for level of difficulty.

When RadioShack was charging \$150 for the 64KRAM chip set, the do-it-yourself upgrade could save the user up to \$100—worth risking damage to the CoCo. Today the savings are less, so if anything in this article is not clear to you, don't go ahead with the procedure. A wrong assumption could be costly. Read all instructions thoroughly and make sure the modification does not exceed your level of experience. And make sure you have all the parts on hand.

Ground Rules

Chips are numbered counterclockwise from the notch (Fig. 1). Where soldering is called for, use a 25- to 40-watt iron with small-gauge (71 mm. is fine), 50/50 or 60/40 rosin-core solder. Where wire is called for, use 30-gauge insulated wire. Where 4164 dynamic RAMs are called for, they must be of the 128-cycle variety. NEC chips are suitable, but TMS chips will not work because they require a 256-cycle refresh.

The difficulty ratings are easy (no soldering required), moderate (some soldering required), and difficult (cutting of lands or desoldering necessary).

Use the following procedure to prepare your CoCo for each upgrade modification. Open the CoCo by turning it upside down and removing the five (six on the CoCo 2) screws. Be sure to remove the screw under the warranty seal. Place a piece of masking tape over each hole so you won't lose the screws. Turn the machine upright and lift off the top.

Unplug the keyboard and put it aside. After clipping the two plastic cable ties, remove the metal shield covering the section of the board that contains the 6809 CPU (central processing unit), 6883 SAM (synchronous address multiplexor), the 6821 PIAs (peripheral interface

adaptors), the 6847 VDG (video display generator), and the memory chips. (Note: The CoCo 2 may not have a shield.)

4K to 16K—Easy

This modification applies only to C, D, or E boards.

Remove the metal shield. The eight 4K RAMs are identified as U20 to U27 on the board. Replace these with 4116 dynamic RAMs, making sure that the rounded notches on the chips match the notches on the sockets. (The notches point toward the rear of the computer.)

Move all jumpers to the 16K position. Replace the shield and close up the machine and you are done.

Continued on p. 54

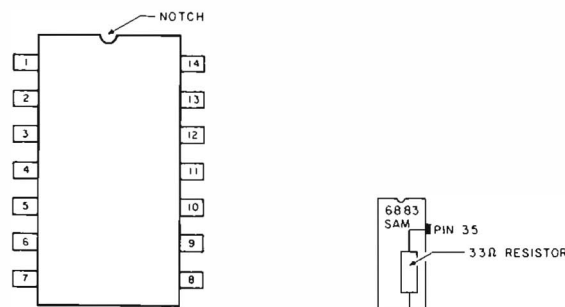


Fig. 1. Chip Pin Location

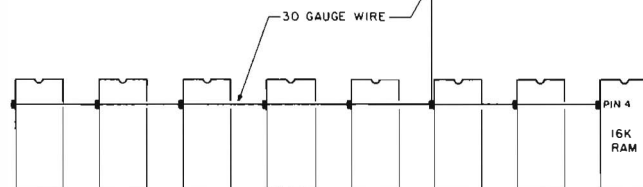


Fig. 2. The Piggyback Upgrade

HOT CoCo's Pull-Out Program Listings

64K Modification Revisited	45
80K Color Computer	45
Doctor ASCII	46
Mindbusters	47
Missile Defense	48
Pie In The Sky	50
Investment Analysis	51
Fabulous Fonts For The Gemini 10X	52
How Your CoCo Adds Up	52



All program listings are available on our Instant CoCo cassette.

64K Modification Revisited

Program Listing 1.

Memory Test

```

10 FOR I= 32382 TO 32407
20 READ X
30 POKE I,X
40 NEXT I
50 EXEC32382
60 DATA 26,80,142,128,0,166,132,183
70 DATA 255,223,167,128,183,255,222,140
80 DATA 255,0,38,241,183,255,223,28
90 DATA 175,57
100 CLS:PRINT
110 PRINT"E-BOARD TEST - E"
120 PRINT "WRITE TO ROM AREA - R"
130 PRINT"WHICH ONE?";
140 AS=INKEY$:IFAS="" THEN 140
150 PRINTAS
160 IF AS="R" THEN 240
170 IF PEEK(49152)=68 THEN S=&HE
000 ELSE S=&HC000
180 FOR I=S TO &HFEFF
190 POKEI,0: IF PEEK(I)<>0 THEN
PRINT"TEST FAILS AT $";HEX$(I):G
OTO110
200 POKEI,255: IF PEEK(I)<>255 T
HEN PRINT"TEST FAILS AT $";HEX$(
I):GOTO110
210 PRINT00,HEX$(I):NEXTI
220 PRINT"E-BOARD TEST CHECKS OK
": PRINT

```

```

230 GOTO110
240 POKE&HA147,99
250 IF PEEK(&HA147)<>99 THEN PRI
NT"TEST FAILS":GOTO110
260 PRINT"OS-9, FLEX, 64K CHECKS
OUT": PRINT
270 GOTO110

```

Program Listing 2.

64K Enable

```

10 REM ENABLE 64K AND COPY BASIC
INTO RAM
20 FOR I= 32382 TO 32407
30 READ X
40 POKE I,X
50 NEXT I
60 EXEC32382
70 DATA 26,80,142,128,0,166,132,183
80 DATA 255,223,167,128,183,255,222,140
90 DATA 255,0,38,241,183,255,223,28
100 DATA 175,57
110 REM RESET PROTECTION
120 POKE&HA055,&H0D
130 FOR I=&HA068 TO &HA071
140 POKE I,&H12
150 NEXT I
160 REM USE RESET VECTOR
170 A1=PEEK(114): A2=PEEK(115)
180 POKE114,0: POKE115,&HF8
190 POKE&HF8,&H12
200 POKE&HF9,&H7F: POKE&HFA,&HFF
: POKE&HFB,&HDF
210 POKE&HFC,&H7E: POKE&HFD,A1:
POKE&HFE,A2
220 PRINT"BASIC NOW IN RAM"
230 POKE 44014,ASC("o")
240 POKE 44015,ASC("k")

```

The 80K Color Computer

Program Listing 1. Bank Switching Utility

```

5 CLEAR 100,32300
10 FOR I=65068 TO 65068+172
20 READ AS:POKE I,VAL("&H"+AS):N
EXTI
25 RESTORE
30 FOR I=32300 TO 32300+172:READ
AS:POKE I,VAL("&H"+AS)
40 NEXTI
41 RESTORE
42 DEFUSR0=32300:DEFUSR1=32326
45 IFPT>0 THEN 54
47 PT=1
50 XX=USR0(0)
52 GOTO10
54 POKE&HFFDE,1
55 YY=USR1(0)
60 END
199 'ML SUBROUTINE 1
200 DATA1A,50,8E,80,00,A6,84,B7
210 DATAFF,DF,A7,80,B7,FF,DE,8C
220 DATAFF,00,26,F1,B7,FF,DF,1C
230 DATAAF,39
235 'ML SUBROUTINE 2
240 DATA1A,50,8E,00,00,10,8E,00
250 DATA00,A6,80,B7,FF,D5,A7,A0
260 DATAB7,FF,D4,10,8C,80,00,25
270 DATAF0,1C,AF
275 'ML SUBROUTINE 3
280 DATAB7,FF,D5,86,39,B7,01,67
290 DATAB7,FF,C6,B7,FF,C9,B7,FF
300 DATACA,B7,FF,CC,B7,FF,CE,B7
310 DATAFF,D0,B7,FF,D3,39
315 'ML SUBROUTINE 4

```

Listing continued

Listing continued

```

320 DATAB7,FF,D4,86,7E,B7,01,67
330 DATAB7,FF,C6,B7,FF,C9,B7,FF
340 DATAA,B7,FF,CC,B7,FF,CE,B7
350 DATAFF,D0,B7,FF,D2,39
355 'ML SUBROUTINE 5
360 DATA1A,50,8E,00,00,10,8E,00
370 DATA00,A6,80,B7,FF,D5,A7,A0

```

```

380 DATAB7,FF,D4,10,8C,05,FF,25
390 DATAF0,1C,AF,7E,7E,61
395 'ML SUBROUTINE 6
400 DATA1A,50,8E,00,00,10,8E,00
410 DATA00,A6,80,B7,FF,D4,A7,A0
420 DATAB7,FF,D5,10,8C,05,FF,25
430 DATAF0,1C,AF,7E,7E,7F

```

Program Listing 2. Bank Switching Utility. Assembly Version

```

7E2C 00090 *MACHINE LANGUAGE SUBROUTINE 1
7E2C 00095 ORG $7E2C
7E2C 00100 ORCC $S50 INHIBIT INTERRUPTS
7E2E 8E 8000 00110 LDY $S0000 START OF ROM
7E31 A6 84 00120 LOOP1 LDA ,X GET ROM BYTE
7E33 B7 FFD5 00130 STA $FFDF MAP1
7E36 A7 80 00140 STA ,X+ STORE BYTE, INCREMENT X
7E38 B7 FFD5 00150 STA $FFDE MAP0
7E3B 8C FF00 00160 CMPX $SFF00 ALL ROM COPIED?
7E3E 26 F1 00170 BNE LOOP1 LOOP IF NO
7E40 B7 FFD5 00180 STA $FFDF LEAVE MAP 1 RUNNING
7E43 1C AF 00190 ANDCC $SAF ENABLE INTERRUPTS
7E45 39 00200 RTS
00210 *MACHINE LANGUAGE SUBROUTINE 2
7E46 1A 50 00220 ORCC $S50
7E48 8E 0000 00230 LDY $S0 INITIALIZE REGISTERS
7E4B 108E 0000 00240 LDY $S0 TO BOTTOM OF RAM
7E4F A6 80 00250 LOOP2 LDA ,X+ GET BYTE, INC. X
7E51 B7 FFD5 00260 STA $FFD5 SELECT RAM BANK TWO
7E54 A7 A0 00270 STA ,Y+ STORE BYTE, INC. Y
7E56 B7 FFD4 00280 STA $FFD4 SELECT RAM BANK 1
7E59 108C 8000 00290 CMPY $S8000
7E5D 25 F0 00300 BLO LOOP2
7E5F 1C AF 00310 ANDCC $SAF
00320 *MACHINE LANGUAGE SUBROUTINE 3
7E61 B7 FFD5 00330 STA $FFD5 EXIT IN RAM BANK 2
7E64 86 39 00340 LDA $S39 DISABLE SUBROUTINE WHICH
7E66 B7 0167 00350 STA $167 RESETS VIDRAM TO PAGE2
7E69 B7 FFC6 00360 STA $FFC6 SET VIDRAM REGISTER
7E6C B7 FFC9 00370 STA $FFC9 TO BIT PATTERN
7E6F B7 FFCA 00380 STA $FFCA 1000010
7E72 B7 FFCC 00390 STA $FFCC
7E75 B7 FFCE 00400 STA $FFCE
7E78 B7 FFD0 00410 STA $FFD0
7E7B B7 FFD3 00420 STA $FFD3
7E7E 39 00430 RTS
00440 *MACHINE LANGUAGE SUBROUTINE 4
7E7F B7 FFD4 00450 STA $FFD4 SELECT BANK 1
7E82 86 7E 00460 LDA $S7E ENABLE ROM SUBROUTINE
7E84 B7 0167 00470 STA $167 TO RESET VIDEO RAM TO PAGE 2
7E87 B7 FFC6 00480 STA $FFC6 SET VIDRAM
7E8A B7 FFC9 00490 STA $FFC9 REGISTER TO
7E8D B7 FFCA 00500 STA $FFCA BIT PATTERN
7E90 B7 FFCC 00510 STA $FFCC 0000010
7E93 B7 FFCE 00520 STA $FFCE
7E96 B7 FFD0 00530 STA $FFD0
7E99 B7 FFD2 00540 STA $FFD2
7E9C 39 00550 RTS
00560 *MACHINE LANGUAGE SUBROUTINE 5
7E9D 1A 50 00570 ORCC $S50
7E9F 8E 0000 00580 LDY $S0
7EA2 108E 0000 00590 LDY $S0
7EA6 A6 80 00600 LOOP3 LDA ,X+
7EA8 B7 FFD5 00610 STA $FFD5
7EAB A7 A0 00620 STA ,Y+
7EAD B7 FFD4 00630 STA $FFD4
7EB0 108C 05FF 00640 CMPY $S5FF
7EB4 25 F0 00650 BLO LOOP3
7EB6 1C AF 00660 ANDCC $SAF
7EB8 7E 7E61 00670 JMP $7E61
00680 *MACHINE LANGUAGE SUBROUTINE 6
7EBB 1A 50 00690 ORCC $S50
7EBD 8E 0000 00700 LDY $S0
7EC0 108E 0000 00710 LDY $S0
7EC4 A6 80 00720 LOOP4 LDA ,X+
7EC6 B7 FFD4 00730 STA $FFD4
7EC9 A7 A0 00740 STA ,Y+
7ECB B7 FFD5 00750 STA $FFD5
7ECE 108C 05FF 00760 CMPY $S5FF
7ED2 25 F0 00770 BLO LOOP4
7ED4 1C AF 00780 ANDCC $SAF
7ED6 7E 7E7F 00790 JMP $7E7F
00800 END
00000 TOTAL ERRORS

```

```

LOOP1 7E31
LOOP2 7E4F
LOOP3 7EA6
LOOP4 7EC4

```

Program Listing 3. Basic Transfer Demonstration

```

100 CLS:PRINT"THIS SECTION OF THE PROGRAM IS EXECUTED IN THE FIRST RAM BANK."
110 INPUT"<ENTER> TO GO TO THE SECOND RAM BANK";ANS
120 EXEC 32413
130 CLS:PRINT"THIS SECTION IS EXECUTED IN THE SECOND BANK."
135 INPUT"<ENTER> TO RETURN TO BANK ONE";ANS
140 EXEC 32443
150 CLS:PRINT"NOW YOU ARE BACK WHERE YOU STARTED IN THE FIRST RAM BANK"

```

Doctor ASCII

Program Listing 1. PUTLDR

```

10 ' PUTLDR.BAS V1.0
20 CLEAR 1000
30 CLS:PRINT" PUTLDR PROGRAM"
40 PRINT" BY
50 PRINT" J.W.JACKSON
60 PRINT" FOR
70 PRINT" HOT COCO
80 PRINT" <C> 1984
90 IF YC<0 THEN 140
100 REM ADDS AN AUTOMATIC LOADER
110 REM TO A BINARY FILE ON DISK
120 REM SO THAT WHEN THE FILE IS
130 REM LOADED,IT AUTO STARTS.
140 PRINT:PRINT"SELECT OPTIONS"
150 PRINT"1. DISPLAY FILE ADDRESSES
160 PRINT"2. PUT AUTO LOADER
170 PRINT"3. REMOVE AUTO LOADER
180 PRINT"4. QUIT
190 INPUT" YOUR CHOICE <1,2,3,4>";YC
200 IF YC<4 THEN GOSUB 270:IF FE=0 THEN 300
210 IF YC<1 OR YC>4 THEN 300
220 IF YC=3 THEN UNS="Y" ELSE UNS="N"
230 ON YC GOSUB 360,620,380,260
240 IF YC=2 OR YC=3 THEN GOSUB 360
250 GOTO 300
260 END'*****
270 CLS:PRINT"ENTER FILENAME":PRINT" <DRIVE:>FILENAME<EXT>":LINE INPUT" ";AS
280 C0=INSTR(1,AS,":")
290 IF C0<3 THEN 320
300 DR$=MID$(AS,C0+1,1)
310 AS=DR$+" "+LEFT$(AS,C0-1)
320 IF (INSTR(1,AS,"/") = 0) AND INSTR(1,AS,":") = 0 THEN AS=AS+ "/BIN"
330 GOSUB 810'CHECK FILE EXISTS
340 IF FE=0 THEN GOSUB 960:RETURN
350 RETURN
360 GOSUB 380'GET FILE ADDRESSES
370 INPUT" PRESS A KEY ";ZZ$:RETURN
380 EA=0:L=0:S=0:OS=1:UA=0'GET FILE ADDRESSES
390 OPEN "D",#1,AS,1
400 FIELD #1,1 AS BS
410 GET #1,OS:TI=ASC(BS)
420 GET #1,1+OS:LH=ASC(BS)
430 GET #1,2+OS:LL=ASC(BS):L=LH*256+LL
440 GET #1,3+OS:AH=ASC(BS)
450 GET #1,4+OS:AL=ASC(BS):A=AH*256+AL
460 IF TI=255 THEN A:TH=AH:TL=AL:GOTO 520
470 IF UNS="Y" AND TI=00 THEN IF A=159 THEN UA=OS
480 IF EA=0 OR EA=A THEN 490 ELSE GOSUB 590:GOSUB 580
490 EA=A+L
500 IF S=0 THEN GOSUB 560:GOSUB 580
510 IF TI=00 THEN S=S+L:OS=L+5+OS:GOTO 410
520 IF UNS="Y" AND UA>0 THEN GOSUB 730'REMOVE LOADER
530 CLOSE #1
540 GOSUB 590:GOSUB 600
550 PRINTSTRING$(30,"*"):RETURN
560 GOSUB 550:PRINTAS," HEX/ DEC"
570 PRINTSTRING$(30,"*"):RETURN
580 PRINT"LOADS",HEX$(A)"/A:RETURN

```

Listing continued

16K to 32K Piggyback—Moderate

This upgrade applies to all CoCos, but note that the CoCo 2 uses different chips.

If you shop, you can get a set of 64K RAM chips for under \$30, so this upgrade makes sense only if you received the 16K chips free. This modification requires soldering, preferably with the memory chips outside of your machine. The original CoCos require 4116 dynamic RAM chips, and the CoCo 2 requires 2118 dynamic RAMs.

Open the CoCo. To be on the safe side, first remove the original RAM chips and replace them with the ones you plan to piggyback. If they prove to be good, proceed as follows: Place the original chips in a piece of conductive foam in approximately the same configuration as they go in the board.

With each of the new chips, gently bend pin 4 straight up (the fourth pin down from the notch on the left side) using long-nose pliers. Gently squeeze the remaining pins inward a small amount so that when placed over a chip in the foam, it will hug it and not fall off. With the exception of pin 4, solder each pin of the new chip to the corresponding pin of the chip beneath it.

Caution: Do not use excessive amounts of solder or hold the iron too long. Solder a piece of wire to each of the bent-up pin 4's connecting them all and leaving enough slack so that you can replace the piggybacked chips in their sockets.

Remove the 6883 SAM chip and place it in the conductive foam. Pin 40 is the first pin to the right of the notch. Counting backwards on the right side, locate pin 35 (the sixth pin from the top). Solder one end of a 33-ohm, 1/4-watt resistor to the top of the SAM chip's pin 35 so that you can replace it in its socket.

Bend the resistor lead so that the resistor rests on top of the SAM

and the lead points toward the RAM chips. Return the SAM to its socket and solder a wire connecting the unused lead of the resistor to the most convenient pin 4 of the RAMs (Fig. 2). Do not move the jumpers. Close up the machine and you are done.

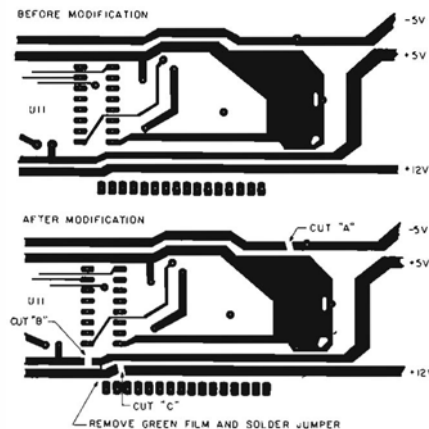


Fig. 3. Foil Side of PC Board

4 or 16K to 64K, D Boards—Difficult

You must first convert the D board to an E board. Check to see if you have a 1.1 or 1.2 Color Basic ROM by typing EXEC 41175. If you have the 1.0 version, order the updated ROM, Radio Shack no.

The Other Korean CoCo—A Different Story

Until recently, upgrading all models of the Color Computer involved, among other things, replacing the eight memory chips with eight 64K by 1-bit dynamic RAM chips, usually designated by the number 4164.

Around January of this year, Tandy quietly introduced a new version of the Color Computer 2 with a new catalog number, 26-3134A for the 16K Color Basic model and 26-3136A for the 16K Extended Basic model. The A-version boards have been extensively redesigned. What follows is the 64K upgrade procedure for these boards and a technical rundown.

The Newest 64K Upgrade

Tandy now uses 4-bit wide 16K dynamic RAM chips in its CoCo 2's. This means that there are only two RAM chips in the CoCo rather than eight. The chips in the current 16K A models are Texas Instruments 4416 chips. They are socketed and located more or less in the center of the board. On either side of this pair of 18-pin chips is a white female connector. This connector will allow for a plug-in satellite board for 64K upgrades.

But this satellite board would be quite clumsy, getting in the way of other hardware add-ons (e.g., lowercase kits and video drivers). (Ed. note: Apparently Radio Shack is upgrading these latest CoCos by replacing the entire board.)

But I know of two IC manufacturers who are delivering those 64K RAM chips at this writing: Hitachi (part number 50464) and NEC (part number 41254). Do not confuse the NEC number with the 41256 1-bit wide by 256K dynamic RAM chip. These chips currently sell for \$25 to \$35 each. Very soon at least two other chip manufacturers will offer similar chips. Texas Instru-

ments will introduce its 4464 chip and Micron Technologies will introduce its 4064 chip. Intel might offer its own version, too, so I expect prices to drop on the 4-bit by 64K RAM chips.

All of the above chips should work for the CoCo 2A 64K upgrade, which is simplicity itself. First, remove the two socketed 16K RAMs (labeled TMS4416) and replace them with your two 4-bit by 64K RAMs, aligning the notches on both chips and sockets. Then solder a jumper between the two solder points on the front (i.e., closer to the keyboard) right part of your CoCo board: they are clearly marked "64K RAM" and enclosed in a little rectangular white outline. That's all there is to it!

I used the NEC chips for this upgrade and have experienced no problems.

Technical Details

The CoCo 2A uses a new SAM chip, 74LS285. It is the only chip other than memory socketed on the main board and is very similar to the old SAM, but it possesses 256-cycle refresh circuitry rather than the 128-cycle circuitry of the old SAM. This is to accommodate the new 4-bit by 64K RAMs. The new SAM will work with older CoCos, but the old SAM will not work with the 2A CoCos.

The new SAM also has extra circuitry so that it can support a full 64K RAM, 32K ROM CoCo without the extra NOR gate added externally to all older CoCos.

There is a new buffer chip between the memory and CPU. Older CoCo used a 74LS244 octal buffer; the 2A uses a TI 74LS623 octal transceiver chip. It serves a similar function.

The 2A's ROM hardware is much changed. It has one 28-pin socket for the on-board ROM operating system. On the Color

AXX3052. With the correct ROM version in the computer or in hand, open the CoCo.

Disconnect the power supply from the PC board. It has five plug-in leads. Remove the 10 Phillips-head screws that hold the PC board. Then pop the 16 metal fasteners that protrude through the board. Remove the board from its base. Now follow these steps in order:

- Remove the eight RAM chips, U20 through U27. These will be labeled 4027 if you have 4K or 4116 if you have 16K. If you have already performed the piggyback modification, remove the wire from the SAM's pin 35.

- Remove the 6883 SAM chip, U10, exercising great care since it is highly susceptible to static discharge.

- Turn the PC board over so that you are looking at the foil side. Referring to Fig. 3, cut the following lands: the -5-volt land at label A, the +5-volt land at label B, and the +12-volt land at label C. Make the cuts cleanly and about 1/16-inch wide. Use a high-quality cutting tool. We used an X-Acto knife with a #11 blade.

- Remove a portion of the green film at point D to expose the lands for soldering. Solder a small wire jumper at point D. We used approximately 1/4 of an inch from the lead of a 1/4-watt resistor. Any solid conductor wire of about the same diameter would suffice. This connects the +5 volts to the old +12-volt line, which will provide the 64K RAM chips with +5 volts to their pin 8's.

- Turn the PC board back over to the component side. Locate capacitors C48, C70, C45, C67, C35, C64, C61, and C31. Using a soldering iron, carefully remove these capacitors.

- On the PC board's component side, locate the area above capacitor C75, to the right of the SAM's pin 26, and below the screw hole. After you have found this area, hold the PC board up to a light and mark a dot in the center, being careful not to mark over a land. Then drill a

hole at this mark just large enough to pass an insulated wire.

- Prepare a 33-ohm, 1/4-watt, 10-percent resistor as follows:

- Cut one lead to 3/8 of an inch and solder it to about 6 inches of insulated wire as used above and cover the solder with spaghetti insulation or electrical tape.

- Bend the other lead at a 90-degree angle and install it into the hole previously occupied by capacitor C48. This is the hole closest to the board's bottom edge.

- Solder the resistor in place.

- Route the insulated wire through the previously drilled hole to the PC board's foil side. This wire should be routed to the land that connects to the SAM socket's pin 35.

- Cut the insulated wire to length and remove approximately 1/8 of an inch of insulation from the end.

- Very carefully solder the stripped wire end to pin 35 on the SAM socket. This now connects the 4164's pin 9's to address line A7.

- Check all connections, cuts, and so on to ensure that no solder bridges or metal fragments remain to short the lands.

- Remove the 4K/16K jumper that lies between the two PIAs, U4 and U8 on top of the board.

- Wire-wrap a length of 30-gauge wire to the center post and connect the other end of the wire to PIA U8's pin 17.

- If you have the 1.0 ROM, replace it with a 1.1 or 1.2 version.

- If it is not already set, place the other 4K/16K jumper to the 16K position.

- Install the new 64K RAM chips and replace the SAM chip. Be sure to line up the notches in the sockets with the notches on the chips.

- Proceed to this article's ROM-Write Disabling section.

64K Upgrade, E Boards—Moderate

If you already have a board that Radio Shack calls 32K, make certain that the jumper to the left of U8 connects the center pin with one marked "LOW", and then continue to the ROM-Write Disabling section.

If your E board has 4 or 16K, replace the RAMs with 4164s and set all four jumpers to the 32K position. One jumper is to the right of U10 (the SAM chip) and the other three are just above the keyboard connector. Remove capacitors C48, C70, C45, C67, C35, C64, C61, and C31.

If it is not already installed, add a jumper connecting the LOW and the unmarked center pin to the left of U8 (a PIA). The other pin, marked "HIGH", is not used. Locate the two pins without anything connecting them labeled R83 to the right of U10. Solder in place a 33-ohm, 1/4-watt, 10-percent resistor connecting the two pins. You now have what Radio Shack calls a 32K machine. Proceed to the ROM-Write Disabling section below.

ROM-Write Disabling

This procedure applies to E or modified D boards only.

You must disable the ROMs during a write cycle. RAM will occupy the entire address space from \$C000 to \$FDFF. The interrupt vectors will not be affected since they reside in the protected area from \$FFF0 to \$FFFF. Follow these steps:

- On the component side of the board, locate and remove the two ICs (integrated circuits) labeled U11 and U29. They should be a 74LS138 and a 74LS02.

- Prepare the chips as follows:

- Carefully bend pins 4, 5, and 6 of the 74LS02 straight up (Fig. 2).

- Carefully bend pin 5 of the 74LS138 straight up.

- Solder a short piece of wire to pin 8 on the 74LS02 at the point where the pin enters the chip, as you must plug this pin back into the socket.

- Solder the other end of the wire to pin 6 of the same chip.

- Replace the chips in their sockets.

- Solder a piece of wire from pin 5 of the 74LS138 to pin 4 of the 74LS02.

- Locate TP1 and cut a length of wire that will reach from pin 5 of the 74LS02 to TP1.

by Martin H. Goodman

Basic model, they have plugged into this a 24-pin, 8K by 8 ROM with Color Basic 1.3. When this model is upgraded to Extended Basic, Tandy unplugs the 24-pin ROM and replaces it with a 28-pin, 16K by 8 ROM that contains both Color Basic 1.3 and Extended Basic 1.1.

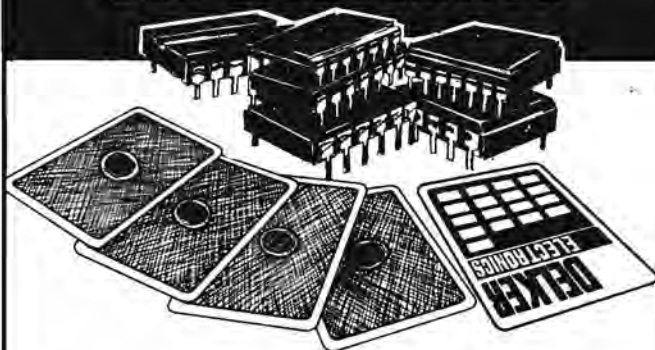
Tandy also has to switch four or five jumpers located near the ROM socket. These jumpers are labeled 64K/128K. The switching of jumpers might require desoldering. The labeling of the jumpers suggest that the 2A can handle 128K of RAM. This is not true. Those designations refer to the number of bits in the ROM chip. They are used to select the different wiring for the 16K by 8 (128K bit) ROM chip. On the Extended Basic 2A, the single 16-pin ROM is soldered directly to the board.

Although the new SAM saves Tandy the use of one NOR gate on the 74LS02 chip, an extra gate is needed to blend together the two chip-select lines from the 74LS138, which formerly selected the two separate ROM chips, into a single select line for the new single ROM chip. Also, the new memory-chip arrangement requires the use of another NOR gate on that 74LS02 in order to properly work the gate timing on the 74LS623.

What about this Basic 1.3? Fortunately, the only changes that Tandy made from the 1.2 ROM were in the SAM setup routine and in the copyright notice logo. The changes for the SAM tell it how to recognize the new RAM chips. Other than that, the two ROM versions are identical.

If you have any questions, feel free to contact me on CompuServe's Color SIG. Type GO PCS 126 at the ! prompt, and then leave a note addressed to my PPN number, 70007,2246. ■

It's time we put our chips on the table



**and showed you our best
deals on computer hardware.**

HARDWARE SPECIALS

Extended Basic w/bk	\$ 39.95	26-1276 DMP-105 Printer 80 cps	\$179.95
64K (DEI) Memory Upg	\$ 49.95	C. Itoh 8510 BPI Printer 120 cps	\$399.95
26-3029 CoCo Drive 0	\$275.95	C. Itoh 7500 Printer	\$249.95
26-3023 CoCo Drive 1	\$229.95	Gorilla/NAP Video Monitor (Grn)	\$109.95
HJL Keyboard (D.E.F. 2)	\$ 79.95	Video Monitor Adapters	\$ 29.95
Super Pro Keybd: (D.E)	\$ 64.95	26-3024 RS Multi-Pac Interface	\$ 89.95
26-3127 64K Extended CoCo2	\$179.95	Botek Ser/Par Interface	\$ 69.95
26-3134 16K Standard CoCo2	\$ 89.95	CGP-220 Ink Jet Printer	\$379.95
26-3136 16K Extended CoCo2	\$129.95	DMP 110 Printer	\$349.95
26-3802 Model 100 24K	\$449.95	26-1278 DWP 220 Printer	\$539.95
26-3816P 8K Upgrade Model 100	\$ 55.95	26-3860 Model 200	\$899.95
26-1192 CGP-115 Printer/Plotter	\$119.95		

ACCESSORIES

Volksmodem 1200	\$299.95	26-3030 OS-9 (64k)	\$ 64.95 (disk)
RS D.C. Modem 1B	\$ 89.95	Basic-09 (req. OS-9)	\$ 87.95 (disk)
Novation J-Cat Modem	\$129.95	"C" Compiler (OS-9)	\$ 87.95 (disk)
USR Password 300	\$179.95	FHL O-Pak (req. OS-9)	\$ 34.95 (disk)
Hayes SM 300 Modem	\$239.95	Elite Word	\$ 59.95 (d&c)
USR Password 1200	\$399.95	Elite Calc	\$ 59.95 (d&c)
USR Password 2400	\$599.95	Color Term Plus	\$ 29.95 (cass)
CoCo Switcher	\$ 39.95	Delt Pascal	\$ 79.95
Elephant Disks ssdd	\$ 18.95	26-3012 Deluxe Joystick	\$ 34.95
Mach II Joystick	\$ 39.95		

NEW! Dual Double Sides Drives including case, power supply & cable \$375.95

NEW! 26-3128 64K Direct Video CoCo2 \$219.95

MSI SOFTWARE

MSI DISKUTIL	NEW	\$ 19.95	MSI CALENDAR	NEW	\$ 19.95
COLOR FINANCE II	NEW	\$ 69.95	MSI COLOR PAYROLL	NEW	\$ 99.95
MSI NAMEFILE		\$ 24.95	COLOR FINANCE 4		\$ 69.95

Call for prices and availability of your favorite software and hardware. All advertised items subject to availability. Prices do not include shipping and handling. All of the above units are covered by our 120 day carry-in warranty.

TRS-80 Trademark Tandy Corporation. Prices subject to change without notice. Write for our FREE newsletter.

Circle Reader Service card #397.



DELKER ELECTRONICS, INC.

P.O. Box 897
408C Nissan Blvd. Smyrna, TN 37167

Call Toll Free:
800-251-5008
800-545-2502 (TENNESSEE)
615-459-2636 (TENNESSEE)
615-254-0088 (NASHVILLE)



- Solder one end of the wire to pin 5 and wire-wrap the other end to TP1.
- Recheck your wiring and look for any shorts, especially between the 74LS02 and the shield.
- Reassemble your machine.

16K to 64K, 285 (a.k.a. F) Board —Easy

If you have a 32K F board, you need only the 64K Enable software described later in this article. If your F board has 16K, remove capacitors C58, C60, C62, C64, C66, C68, C70, and C72 with a pair of diagonal cutters. Replace your 16K chips with 4164 dynamic RAMs and set the jumpers to the 64K position. You must add another jumper at a location clearly marked "64K".

16K to 64K, U.S.A.-made CoCo 2—Moderate

Open your machine, noting that there are six screws. Remove the 16K chips from the sockets labeled U14 to U21. Solder a wire connecting the two solder pads to the right of W1, which is located diagonally between U6 and U7. Install the new 4164 RAM chips in sockets U14 to U21, aligning the notches. Reassemble the machine.

16K to 64K, Type 1 Korean CoCo 2 —Moderate/Difficult

The level of difficulty for this modification depends on whether the chips are socketed or soldered. If the 16K chips are not socketed, you must remove the PC board from the machine and desolder the chips, replacing them with IC sockets with their notches pointing in the same direction as those on the chips. Note: Desoldering is difficult, and not for amateurs.

The eight 2118 dynamic RAMs are located in two places: a group of three and a group of five. Solder a wire connecting the two solder pads inside the box marked "64K". Install the new 4164 RAM chips in the sockets, aligning the notches. Reassemble your computer.

Testing Your Handywork

Memory Test, Program Listing 1, tells you if you did the upgrade correctly, and it will work with any Color Computer. The E Board test checks to see if your RAMs are active; it writes to the area of RAM that is not in the same address range as your ROMs. If you have a 32K machine and this test checks out, you really have 64K RAMs. The Write to ROM Area test is for those with D or E boards. If it fails, recheck the ROM-Write Disabling section.

Extended Basic Installation

This procedure is the same for all CoCos. Order the Extended Basic kit, Radio Shack catalog no. AXX7072. Open your CoCo or CoCo 2. Install the new ROM in the empty socket that is provided, aligning the notches.

Reassemble your CoCo and type PRINT MEM. You will have 6,144 fewer bytes because Extended Basic reserves that amount of memory for graphics. If you do not plan to use graphics and want to free memory for a large Basic program, type POKE 25,6:NEW for a tape system or POKE 25,14:POKE &HE00,0:NEW for a disk system.

If you have the latest CoCo version (catalog no. 26-3134A), you need a different 16K ROM to replace the 8K Color Basic 1.3. (See sidebar.)

Enabling 64K

Program Listing 2, 64K Enable, puts your computer into 64K (all RAM) mode and keeps it there even when you reset it. Your CoCo's OK prompt appears in reverse video to let you know you are in all-RAM mode. To return to the RAM/ROM mode, cycle power off and on again. ■

See program listings on page 45

Richard Esposito and Raymond Rowe are authors of HOT CoCo's Doctor ASCII column. Write them c/o HOT CoCo, 80 Pine St., Peterborough, NH 03458.

UTILITY

by Henry C. Grace, Jr.

The 80K Color Computer

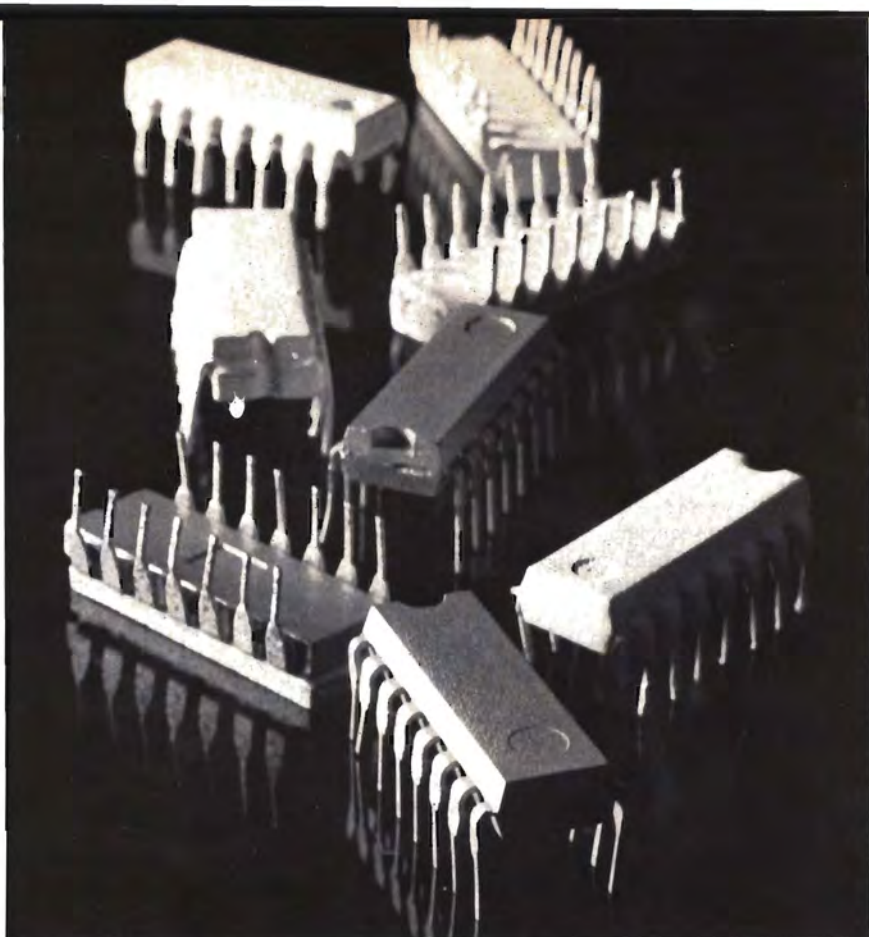


Photo by Edward Judice

*Find that hidden memory and
make the most of your CoCo.*

Those of you with 32K or 64K CoCos are probably painfully aware of what it costs to use all 64K. You need a disk drive and an alternate operating system to reside in the upper 32K of RAM.

You have probably also seen articles and notes on how to dump your ROM into RAM and activate it. This provides almost 16K of additional RAM for machine-language programming, and, for those who have modified boards, it allows RAM patches. My unmodified 32K machine does not allow such repairs.

These are unsatisfying restrictions on the use of all that memory capacity sitting under your fingertips, and there should be a better way to use it.

Program Listing 1 demonstrates that this extra RAM is accessible and can be used as a separate Basic operating environment. (Program Listing 2 gives the Assembly version of the DATA statements.) You can have two Basic programs loaded into the computer at the same time, one in each 32K bank of RAM. You can execute either program independently of the other, or cause them to interact with each other back and forth between the banks of RAM.

In fact, you can use this technique to manipulate your cassette files. Have you ever used multiple tape saves and loads to reorganize programs on a tape? Now you can simplify the process by loading two programs into the computer at the same time. Or have you ever needed information contained in a program on tape, but you don't want to dump the program presently in memory? Now you can put both programs into memory, select the one you want, and list it. With RAM and ROM, your computer becomes an 80K or more machine.

Background

Type POKE&HFFD5,0 into your 64K computer and it goes into limbo, responding only to the reset button, because memory location \$FFD5 selects the alternate bank of RAM. But, when you

select it, it is blank, and the Basic interpreter does not have its required pointers in low memory, so the computer rips off, absently doing its own thing. How can the Basic system be made to run in the alternate RAM bank? Give it the pointers it needs.

Before you do this, you must look at memory location \$FFDF, which selects memory map 1. When this address is POKED it deactivates ROM and activates the alternate RAM bank in place of ROM in the memory map. That is, the RAM becomes the upper 32K of the total of 64K in the computer.

Memory address \$FFDE reverses this process, returning the machine to memory map 0. The first machine-language subroutine in Listing 1 (called by USR0 to address 32300) makes use of these two addresses to sequentially load the entire ROM operating system into the alternate RAM bank. When the subroutine returns to Basic, memory map 1 is in effect and the RAM-based operating system is running things. (You might also have seen this technique before in "64K Modification," by Richard Esposito and Raymond Rowe, *HOT CoCo*, July 1983, p. 44.)

You cannot POKE new instructions into the Basic area of an unmodified CoCo (like mine). You are concerned with memory locations from \$C000 to \$FEFF (\$E000 to \$FEFF in Disk Basic). You can POKE these addresses to gain almost 16K (almost 8K with Disk Basic) of extra RAM for machine-language programs.

One of the tricks to initializing RAM bank 2 for Basic lies with the second machine-language subroutine (address 32326) in List-

System Requirements

64 or 32K RAM

Extended Color Basic

Editor/Assembler optional

ing 1. The trick is in its placement in memory; the machine code is stored in duplicate in both RAM banks. Keep in mind that, although RAM bank 2 is now serving as the upper 32K, it will soon be returned to an equal status with RAM bank 1. The code is stored in identical address locations in each RAM bank, when you think of each bank as consisting of addresses 0-32767.

This program sequentially duplicates every byte of information from the first RAM bank into the second RAM bank. It works because of the duplicate machine code. When the code selects the second RAM bank, it finds the duplicate right where it should be to keep the CPU working on the task. When the code jumps back to the first RAM bank, it again finds the code right where it needs to be.

Getting Video

When you run Listing 1, you will not notice any change. The video screen will look as it normally looks, though a little massaging of the video RAM page-select register was required to get it. In RAM bank 1 the start of video RAM, VIDRAM, is \$400. This is page 2 (bit pattern 0000010) in the video-page-select register (locations \$FFC6 through \$FFD3). (See the back of the *Getting Started with Color Basic* manual for further information on this register.)

In RAM bank 2, Basic treats VIDRAM no differently; memory \$0 to \$7FFF are the RAM addresses in both banks, and \$400 is VIDRAM. However, the physical realities are different. In RAM bank 2, the video RAM is located in page 66 (bit pattern 1000010) in the video page-select register. If you leave VIDRAM at page 2, or \$400, you will not have a video display in RAM bank 2.

By properly setting the video page-select register, VIDRAM is changed to \$8400, or page 66, and RAM bank 2 has video. Of course, when you want to change back to RAM bank 1, VIDRAM must be reset to \$400, or page 2.

Machine-language subroutine 2 properly sets VIDRAM to \$8400 before leaving RAM bank 2 active. It also stores a \$39 (RTS) at memory location \$0167, disabling a ROM subroutine that resets VIDRAM to page 2 before every PRINT operation. This is why you see a proper video display after you run Listing 1.

Machine-language subroutines 3 and 4 in Listing 1 allow transfer from one RAM bank to the other. Subroutine 3 (address 32353) transfers from bank 1 into bank 2. Just type EXEC 32353 and you will be in bank 2 with proper video. (All this assumes that you have run Listing 1 to store the machine code.) Subroutine 4, called by EXEC 32383, transfers from bank 2 into bank 1, with the video properly set. You cannot use bank 2 for high-resolution graphics.

Using the Method

To set up for Basic in the second RAM bank, enter and run Listing 1. You will be in RAM bank 2 when the OK prompt returns. The Basic program operates in two passes. On the first run through, it stores the machine-language code starting at address 32300. It attempts to store at address 65068, but those addresses are not in RAM yet.

Next, the program uses the USR function to execute the first machine-language subroutine, which copies ROM into RAM and sets memory map 1. The program then loops back and stores the machine-language subroutines again. This time the addresses starting with 65068 are in RAM and the code is stored. After this, the program switches back to memory map 0.

You now have two equally ranked banks of RAM again, but now they have the machine code stored at identical locations. Finally the program calls subroutine 2, which copies RAM bank 1 identically into RAM bank 2 and exits back to Basic in RAM bank 2.

To reenter RAM bank 1, type EXEC 32383 and press enter. EXEC 32353 puts you into RAM bank 2. You can CLOAD, list, and run programs in each RAM bank independently. Just remember that high-resolution graphics will not respond properly in bank 2. In bank 2, avoid all commands like PCLS, PCLEAR, and so on. Also, if you press reset in RAM bank 2, you will return to bank 1. On one occasion I lost the program in RAM bank 2 after pressing reset. However, every other time the program was still there when I switched back to bank 2 again.

Cross Communications

The two RAM banks are independent, but it is possible for them to communicate with a Basic program. Machine-language subroutines 5 and 6 in Listing 1 allow this cross transfer using Basic programs.

Subroutine 5, called by EXEC 32413, transfers from RAM bank 1 into bank 2. Subroutine 6, called by EXEC 32443, transfers from RAM bank 2 into bank 1. Both calls should be made as part of a Basic program, or segment of a program, and stored identically in both RAM banks. Program Listing 3 is a demonstration program for Basic transfers between the two banks.

To use Listing 3, follow this sequence:

- CLOAD Listing 1 and run it;
- enter EXEC 32383 to reenter bank 1;
- CLOAD Listing 2 but do not run it yet;
- enter EXEC 32326 (subroutine 2) to duplicate Listing 2 in bank 2;
- enter EXEC 32383; and
- run.

The program will print messages and input prompts, under Basic control, from both RAM banks. Note from the above sequence that machine-language subroutine 2 duplicates all of RAM bank 1 into RAM bank 2 anytime you execute it. Subroutines 5 and 6, on the other hand, duplicate only the Basic pointers and video RAM in lower memory. This means that programs communicating back and forth can have altogether different variables-table data.

It should be possible, using either Basic or machine-language programs, to devise algorithms that would allow more complete use of all 64K. The trick is to have identically addressed duplicate code in both memory banks. It might even be possible to have two totally different Basic programs running at the same time.

Perhaps you could operate RAM bank 2 as a printer buffer, while another program runs in bank 1. Or perhaps you could write machine-language code to let you use bank 2 as a minidisk emulator to store data for a program running in bank 1. This would give more room in bank 1 for the program.

A simpler way to do all these things is to store required machine-language code in a cartridge ROM. This would allow all the operations I have described to be performed without the need for duplication in the RAM banks. The cartridge ROM would be independent of the RAM areas, and it could preside over both areas in the same way the Basic ROM does.

I would be glad to hear from anyone having ideas or questions about this method, or potential uses for it. (If you want a reply, please send a self-addressed, stamped envelope with your query.) ■

See program listing on page 45

Address correspondence to Henry C. Grace, Jr. 424 Ranchwood Drive, Baton Rouge, LA 70815.