

COLOR COMPUTER NEWS

February 1982
Issue No. 6

REMARKS.....	3	A Basic Word Processor.....	39
Mail Call.....	5	Disabling the Break Key.....	46
1981 Tax.....	9	Making Education More Colorful.....	49
Skiing.....	17	Those Friendly Folks at Frank Hogg Labs.....	53
Tape to Disk.....	19	Processing Numeric Data.....	54
32K for Free.....	23	MC6809 Software and Programming..	57
Comment Corner.....	27	RS-232 The physical Connection.....	59
Running Machine Language Programs	33	Putting Numbers in Strings.....	61
Interfacing a Non-Standard Printer...	36		

TRS-80 is a trademark of the Tandy Corporation.
Color Computer News Copyright © 1982 by REMarkable Software.

BULK RATE
U.S. POSTAGE
PAID
MUSKEGON, MI
PERMIT #220

REMarkable Software
P.O. BOX 1192
MUSKEGON, MI 49643

16 PLUS
MEMORY EXPANSION FROM 16K TO 32K FOR THE
**COLOR
COMPUTER**



- No soldering! Easy to plug in!
- Fits under RF shield — complete instructions included
- Allows video display (graphics) to reside anywhere in the 32K memory
- Call or write for our complete catalog of Color Computer products!



6809 Specialists

COMPUTERWARE®

Box 668 • Encinitas, CA 92024
(714) 436-3512

Computerware is a trademark of Computerware.

REMARKS
by Bill Sias

Everyone has noticed, and lots of people have written about the fact that we are behind schedule. We were completely and thoroughly snowed-in. We are now attempting to get back on schedule as rapidly as possible. At first I thought it would be best to get back on at the rate of a week per month, which would get us back on in about four months. But after reading the letters people have been sending I realize that the information available from other sources is limited and you need CCN, you also need it on time. So please bear with us we're trying to be back on schedule with the April issue. That means that you should get the March issues within the next two to three weeks and the April two to three after that.

Happy Birthday to us, CCN will be one year old in May and we're really going to celebrate. The advertisers have gotten behind us and we're going all out. We are having four contests and every reader is eligible. Prizes so far total over \$1,000 and I still have more than half of the advertisers to contact.

Contest number 1, remember the old magic square? No, well a magic square is a four by four array containing the numbers one through sixteen. If you draw a straight line from any one side or corner to the opposite the total will be 34. There are 880 solutions to a 4 by 4 magic square and the person that submits a BASIC program that solves all 880 possibles in the least amount of time wins. It must be an all BASIC program, no hidden machine language, no funny tricks. This contest will be judged by Mr. Phranc Hannum, owner of the 8 Bit Corner. Entries must be submitted on cassette with a description of the algorithm included on paper. This is the big banana, prizes include things like a life subscription and literally hundreds of dollars in software.

Contest number 2, everyone likes graphics so this one is the "best graphic portrait" contest. It must be a recognizable person. Either BASIC or machine language and the judging will be done by our staff. Programs must be submitted on cassette machine language programs must include Source code.

Number 3, the best rendition of the William Tell Overture. This can be either BASIC or machine language. Again all entries must be on cassette.

Number 4. We all have hobbies other than computing. So this contest is the best computer

program that assists with another hobby. All entries must be on cassette and must be accompanied by a complete explanation of the other hobby and how this program assists in its performance.

As a bonus we will be accepting postcards until May 1, 1982 on May 8, 1982 we will draw 5 of the received cards from a box and award those folks with life subscriptions.

The fine print. All entries become the property of REMarkable Software, Inc. and none will be returned. All entries must be received on or before May 1, 1982 and winners will be announced in the June 1982 issues of CCN. Winner's programs will be published and receive normal author compensation in addition to the prizes received. Ties will be settled by postmark with the earliest submission breaking the tie.

Silly Syntax

By Sugar Software

A hilarious and outrageous story game for one to ten players. This game will become one of your favorites to play and show off. Create your own stories with the built-in screen editor or order story tapes from the selection below. Silly Syntax features include creating, modifying, printing, saving and loading of stories. Included is the Silly Syntax game, two stories and the user guide.

\$19.95 - Requires Extended Basic.

Silly Syntax stories - Ten stories per cassette.

SS-001-Fairy Tales	SS-004-Current Events
SS-002-Sing Along	SS-005-Gothic Romance
SS-003-X-rated	SS-006-Adventure/Sci-Fi

\$9.95 - 10% off for 3 or more story cassettes.

All products are available now.
Ohio residents add 5.5% sales tax.
Add \$1.00 per cassette for postage and handling.

Sugar Software

2153 Leah Lane
Reynoldsburg, OH 43068
(614) 861-0565

EXCITING NEWS FOR COLOR COMPUTER USERS

FLEX, OS-9 and the Radio Shack Disk System ALL on the SAME Color Computer

Would you believe that you can run FLEX, OS-9 and Radio Shack disk software on the same Color Computer, and all you have to do is change the disk? That's right, just change the disk. If you have a 32K Color Computer with the Radio Shack disk system, all you need to do is make a trivial modification to access the hidden 32K, as described in the Feb. issue of COLOR COMPUTER NEWS and the March issue of '68' Micro. You can get FLEX from us right now. OS-9 will be ready by summer. Please note that this will only work with the Radio Shack disk system and 32K/64K memory chips that RS calls 32K. Maybe they put 64K's in yours, too. If you don't have a copy of the article, send a SASE and we'll send it to you.

Using this system to run FLEX and OS-9 has many advantages. First, it gives you 48K from zero right up to FLEX. This means that *ALL* FLEX compatible software will run with *NO MODIFICATIONS* and *NO PATCHES!* There are no memory conflicts because we moved the screen up above FLEX which leaves the lower 48K free for user programs.

What you end up with is 48K for user programs, 8K for FLEX and another 8K above FLEX for the screens and stuff. We are working on a multi screen format so you can page backward to see what scrolled by and a Hi-Res screen that will enable us to have 24 lines by 42 character display. That's better than an Apple!

We also implemented a full function keyboard, with a control key and escape key. All ASCII codes can now be generated from the Color Computer keyboard!

We also added some bells and whistles to Radio Shack's Disk system when you're running FLEX or OS-9. We are supporting single or double sided, single or double density, 35, 40 and 80 track drives. If you use double sided drives, the maximum is three drives because we use the drive 3 select for side select. When you are running the Radio Shack disk, it will work with the double sided drives but it will only use one side and only 35 tracks. Using 80 track drives is okay, but will not be compatible with standard Radio Shack software. You can also set each drive's stepping rate and drive type. (SS or DS - SD or DD)

In case you don't understand how this works, I'll give you a brief explanation. The Color Computer was designed so that the roms in the system could be turned off under software control. In a normal Color Computer this would only make it go away. However, if you put a program in memory to do something first (like boot in FLEX or OS-9), when you turn off the roms, you will have a full 64K RAM System with which to run your program (FLEX or OS-9). When the roms are turned off, it is as if you have removed them from the computer. They are gone!

Now, we need the other half of the 64K ram chip to work, and this seems to be the case most of the time, as the article states. Of course, you could also put 64K chips in.

We decided that this was the best way to run FLEX and OS-9 on the Color Computer because it does remove the roms from the memory map and leaves the full 48K for user programs. If you just put in memory for FLEX and use the Basic hooks for I/O, all you have is a little over 30K for user programs. In addition, very few FLEX programs will run without being modified and some won't run very well, if at all (our DATAMAN+ for example). Let me state it again. **ALL FLEX COMPATIBLE PROGRAMS WILL RUN WITHOUT MODIFICATION!!!** and the same goes for OS-9!

It is also the **ONLY** way OS-9 will run because 30K is just not enough.

Some neat utilities are included.

MOVEROM moves Color Basic from ROM to RAM. Because it's moved to RAM you can not only access it from FLEX, you can run it and even change it! You can load Color Computer cassette software and save it to FLEX disk. Single Drive Copy, Format and Setup commands are also included.

If you don't have a Color Computer, we can sell you one complete with 64K ram, 24K rom, Single RS disk drive and FLEX for only \$1,490, set up and ready to go.

FLEX with Edit, Asmb and installation disk is \$199.

FRANK HOGG LABORATORY, INC.

130 MIDTOWN PLAZA • SYRACUSE NEW YORK 13210 • (315)474-7856

I have a 16K Extended Basic "CoCo" with tape recorder, and have been considering additional memory expansion. I have seen ads for several types of RAMs, both 32K and 64K. I would like to see reviews on these, and the problems with such RAMs. i.e. (the Radio Shack 32K expansion requires changing from Microsoft Basic 1.0 to 1.1). Is this required with other RAMs? If this change is not made does it cause difficulties? Also, I would like to see a review on stringy floppies and comparison to disc. (Do they respond to a sort of Disc Basic, or do they use cassette commands, etc.). Have you considered a Basic command per issue, expounding on it limitations and full capabilities. (For example, using IF-THEN). (Or using LINE, and incorporating CIRCLE). (Or, when is spacing necessary? i.e. THEN) and Data statements, and their use in color GRAPHICS. My best wishes to you and your new magazine, and I hope you have a long and successful production.

Kelly G. Morris
Kansas City, MO

Dear Kelly,

First, I've never seen a stringy floppy for the CC. I've considered articles about each command but first we need someone to write them. Are you volunteering? Radio Shack's memory expansion will operate without BASIC 1.1, it's the revision E circuit board that you need, which happens to come with BASIC 1.1. The revision D boards can be made to work and they seem to be throwing in the BASIC 1.1 as a gift. None of the 32K upgrades really require 1.1.

Dear Bill,

I have a question I would like to pose. Rumor has it that phonetic sounds can be generated using software only. Has anyone out there figured out how this works?

Dave L. Landom
Bellbrook, OH

I'm sure that it's quite possible but it would have to be an extremely intelligent program to calculate all of the sounds necessary to create speech (which is what I assume your real question to be). Personally I'm quite happy with my Speaker from Alford and Associates (P.O. Box 6743, Richmond, VA 23230 (804-320-6722)).

Dear Sir:

Your article on Disks in the Nov./Dec. issue was very interesting to me as far as it went. I've never even seen a disk close up.

What are double-headed drives? What is a "Flippy"? What about compatibility of all the various sizes, number of tracks, etc? What do I need to have a complete system? I have the impression that sometimes a drive means a cabinet and power supply and sometimes it doesn't. What is the relation between the controller and the operating system?

I hope someone comes up with a good add-on system (without using extended BASIC) with Flex 9.

I see that most of the programs advertised in your magazine require EXTENDED BASIC. I think that is bad.

I bought the extended BASIC manual to determine if I wanted to buy the extended BASIC. The manual, in my opinion is lousy. So I still don't know whether extended BASIC is worthwhile for me.

You advertised a program exchange but I've never seen anything about it in the Color Computer News. I have written many programs for myself that I am sure others would be interested in but it takes too much time to document them for a magazine. Trading them for a good program I need would appeal to me.

Also I don't know any practical way of getting a cassette back if you turn down a program submitted on cassette. Do you have any suggestions?

Sincerely,
Charles C. Worstell
Auburn, WA

I wasn't sure for a long time if extended BASIC was worth it either, but after retyping entire lines on several occasions I decided that the editor alone was worth the price. We are working on the program exchange and the situation is this, if we accept the program we fill up your cassette with whatever is on the cassette that your has been accepted for and return it to you in your SASE, if the program isn't accepted we return your cassette in your SASE with an offer for reduced price on the accepted programs. The only programs that aren't accepted are violations of copyright, or items similar to something we already have (or a real dog). The answers to your questions about disks would require more space

than I have available here, but I'll work on another article going into more depth about disks in general, better yet, how about someone else!

Dear Bill,

I've discovered a few things about using a printer that I should pass on.

Warning!! On page 209 of Going Ahead With Extended Color Basic, the manual says that decimal address 115 is "line printer width" (LPTWID) and that on power up, it should have a value of 132. That's not the case, on power up, a PEEK(115) returns a value of 192.

If you are using a 80-column printer and "POKE 115,80," it doesn't seem to do anything special. But if you have to do a "RESET", everything in memory goes away (a complete power up condition).

On power up, a PEEK(115) results in a value of 132. So it seems that decimal address 155 is "line printer width". As to what location 115 is, I just don't know.

You can get a print out while in highspeed (POKE 65495,0). Since the clock is running at twice normal speed you just set up for 1/2 the desired baud rate. Take a look at this chart for the different rates.

Baud Rate	Low Speed	High Speed
	POKE 65494,0	POKE 65495,0
75	POKE 149,2	POKE 150,235
120	POKE 149,1	POKE 150,202
150	POKE 149,1	POKE 150,115
300	POKE 149,0	POKE 150,180
600	POKE 149,0	POKE 150,87
1200	POKE 149,0	POKE 150,41
2400	POKE 149,0	POKE 150,18

I do not know what sets up the data rate to the cassette, but it might be possible to apply this idea to the cassette also. Who knows where to "POKE" to make the cassette work in highspeed? In response to Ralph Coleman's question, about using the "DRAW" command for angles other than multiples of 45: Use the "M" without the "B".

```

10 ' SET SCREEN
20 PMODE 4,1 :PCLS: SCREEN 1,0
30 ' DRAWS A TRIANGLE
40 DRAW "BM 100,100 M+40,-60M + 40, + 60
M -80, +0"
50 GOTO 50
    
```

Bill, I hope you guys can get enough out of this to make it printable. I'm still trying to learn to type! Before I got my computer I didn't have a need to type.

Thanks to your article on upgrading to 32K, I now have a 32K machine for only \$21 more.

Keep up the good work.

Sincerely,

Robert Joe Harrison
El Dorado, AR

Mr. Editor:

Earlier this year I purchased a Radio Shack Color Computer from Computer Plus in Littleton Massachusetts. I ordered a 32K system. This was achieved by Computer Plus through "piggybacking" 16K RAM chips and running a jumper wire. The procedure was described in the March issue of BYTE.

Several months later I ordered the Extended Basic ROM and a Line Printer VIII from Computer Plus. My Color Computer has 1.0 version of ROM and I can't generate graphics with it. Computer Plus sent me some free software that was supposed to cause my computer to send 8 bit signals to the printer and thus be able to print graphics characters. I have not been able to do this. I have tried using the 4K and the 16K versions of the software. I have loaded these versions into memory at 3967; 16,255; and 32,639. After calling Computer Plus about this problem, I was told to contact you to see if you had experienced a similar problem and, if so, how you "fixed" it.

I would be very grateful for any help or guidance you may have to offer

Sincerely,

Charles Goad
103 Faye Ct.
King, NC 27021

If anyone has solved this one let us know and we'll print it here.

Dear Bill:

Recently my son and I bought a TRS-80 Color Computer, and shortly thereafter, we suscribed to the Color Computer News. Needless to say, we have found the CCN invaluable in learning how to use the Color Computer. No doubt many owners of Color Computers are like us and own another personal computer of some sort. We happen to have an S-100 bus system with either an 8080 or Z-80 CPU in it as the spirit moves us. So we are interested in having the Color Computer communicate with the S-100 system.

Mail Call

This prompts me to write you to see if you would be interested in an article about our explorations of hooking the Color Computer to the S-100 through the RS232 port. Our S-100 has an 8" soft sector, single density disk drive which operates with CP/M. We just completed a program that allows us to ship a file from the Color Computer to the S-100 which picks it up and records it on the disk. Or visa versa, we can send a file from disk to the memory of the Color Computer. That is, from the S-100 to the Color Computer.

Like any of these hobbist experiments, there are a few little quirks to overcome here and there in order to get something like that to work. And since your magazine is very much "down to earth", do you think you readers would be interested in something like that? If so, let us hear from you.

I might say that we have purchased monitors from all three companies, the Mico Works (mostly that one, on your favorable review), Computerware (the Power Pak) and DataSoft (the SIGMON monitor). We find all three to be most useful. We also bought a crossassembler that runs under CP/M (6809 cross-assembler), which is why we wanted to ship files back and forth.

In any case, keep up the good work, as we certainly appreciate the CCN.

Sincerely,

R.L. Froemke

1516 Argonne Road

Tallahassee, FL 32312

I'm very interested in your experiences so far.

I have one of the first Color Computers serial #1959 and have problems with CLOADing tapes and almost any volume level. This seems to occur only with tapes that I have programmed and doesn't happen with the tapes from Chromasette Magazine.

By the way I have 16K with extended BASIC using a Centrex Cassette Recorder by Pioneer, Model KD-11 with volume and tone controls. I've tried different tapes, Radio Shack and even the reverse side of the Chromasette Cassette. Because of this I am at a loss and it has reduced my understanding of the terminal and BASIC.

Thank you,

Jonathan Leawitt

5359 W. Arcadia

Skokie, IL 60077

It sounds like you need to re-align the head on the recorder or try moving the tone control. We've discovered that the CTR-80 is quite reliable and recommend them to everyone that has experienced difficulties with other brands.

My computer is the DISK EXTENDED COLOR BASIC 16K. The disk and the disc software work well. I just found one bug; when 2 random files are opened at the same time and you use the UNLOAD command to DOS the computer will hang up! If you reset it and type LIST, all you see is garbage instead of your BASIC program!?

Can you explain the purpose and how to use the DLOAD (down load) command. This EXTENDED BASIC command is not explained in RS books. Thank you.

Richard Bussiere

84 Parisse

Laval, Quebec

Canada

H7N 3S3

Continued from Page 53

Our Forth is written entirely in assembler so the execution and compilation time is much shorter. Forth was originally written to be easy to program and I feel it is.

Is your CC Forth an abridged version of XForth?

No, in fact it's an expanded version. It's really the first release of our XForth version 1.2 and is bigger and faster than the Flex versions.

Any final comments?

Well I guess I could close by saying that very few people realize how powerful the CC is. We have attempted to inform folks and to make it possible to overcome the few limitations that it does have. I think people would be shocked to realize that you could take the board out of the CC and put it inside a terminal and have one of the most powerful machines on the market today. The only limitations were the 32K RAM limit, the keyboard and the small display. By making the 64K modification and putting it into a terminal you have resolved all of the problems for less money than any comperable computer. The manuals are available separately.

ACTION GAMES

The fastest growing producer of computer games for your 6809 has the products you have waited for!!

NEW!

CAVE HUNTER

ARCADE GAMES FOR THE COLOR COMPUTER

COLOR BERSERK

Fast paced action • Super Hi-Res Graphics
Dynamite sound effects • Runs in 16K of memory

These games will astonish you with their Detail and Quality.
They set a standard for others to follow.

— ADVENTURES —

Calixto Island • The Black Sanctum

Highly acclaimed by reviewers • Challenging situations
Fast, efficient machine language • Runs in 16K of memory
Save game in progress

Adventures on 5¼ TSC FLEX disc (specify 6800 or 6809)	ea. \$24.95
Both adventures on single disc	\$39.95
Adventures for color computer	ea. \$19.95
Color Berserk for color computer	ea. \$24.95
Cave Hunter for color computer	ea. \$24.95

Shipped prepaid in continental U.S. California residents, please add 6% tax.



— MORE COMING SOON —



MARK DATA PRODUCTS

23802 Barquilla, Mission Viejo, CA 92691 • (714) 768-1551

TRS 80 IS A TRADEMARK OF TANDY CORP.

1981 TAX
Warren S. Napier

16K Regular COLOR BASIC
Form 1040, Schedule A
and Schedule G

APRIL 15--day of gloom if you owe IRS, day of joy if you have money coming back! This program will let you find out in a hurry which it's going to be--assuming you've got your act together, that is.

If you're like me, your act never gets together as early as it should; yet you want to at least get some idea of the damage. The problem then is redoing all the calculations every time you get a new piece of information--the interest statement from Sears, your spouse's W-2 (finally!) or a copy of your July contribution to your dentist's retirement fund.

This program won't collect your papers for you, but it will let you check their effect on your taxes as often as you want with minimal pain. Should you itemize? Can you save money by averaging your income? What if...?

Naturally, once you have all your material in final form, it'll compute that in short order, too. There's also a little music and color surprise to match the results of your calculations.

I designed the program to be usable by friends who only want to know how to put in the numbers and get an answer. Each major section can be reached from any other via the MENU. A mistake in most sections--e.g., the "Interest" category in Schedule A--can be corrected before going on. If you start a subroutine without first coming through the preceding section(s), the program will ask for the data it needs to complete its work.

All this requires a lot of memory, of course, and I had to pick and choose what calculations would be included. For example, every line in the "Adjustments" section of the first page of Form 1040 is included, but only the total from "Credits" on page 2 is accepted.

You may, of course, adapt it for your own needs. It frankly is weighted toward people who are married, filing a joint return and who itemize. It includes Schedule A, Schedule G (income averaging) and will completely calculate the tax for married folks filing jointly (and qualified widows/widowers). If you are single or married filing separately, the program will do all the work except for asking you to look at the Tax Table for Line 35 of the 1040.

If you want to change the program to do the calculations for a different filing status, you need to do the following:

1. Replace DATA lines in subroutine 13000 with the data in the appropriate Tax Schedule (not the Table--the program makes the necessary adjustments). The Schedule has 5 items in 4 columns (e.g., line 3 of the Schedule is 5500,7600,294,16%,5500). The DATA in program lines 13300-13370 consists of all items 4 and 3, read as SN and VA.

2. Change line 6160 to reflect your filing status (e.g., if you are married filing separately, this line could read, "IF FS=3 THEN 6162 ELSE 6170").

You can recapture a lot of RAM by eliminating the MENU, deleting lines which allow corrections, tightening up the formatting, etc. This in turn could allow you to add more Tax Schedules instead of replacing the one in the program. NOTE: you folks with EXTENDED COLOR BASIC will have to clear out your graphics pages whether you make changes or not (POKE 25,6;NEW<ENTER>).

The program is largely self-documenting, with REM where appropriate. Sound prompts exist with major changes in the routine. Obvious errors for the most part will be refused; for example, you will not be allowed to claim more than one exemption for your spouse.

Be careful when you type in the DATA, especially decimal points. Money inputs running the program should be rounded to the nearest dollar; less than 50 cents, round down--50 cents or over, round up.

With some adjustments, the program could handle the short 1040A. I have used the program for my taxes and got a neighbor to try it out. I have run a number of checks on it and made every reasonable effort to debug it. However, given my lack of free time and publishing deadline, I cannot guarantee that the umpteen possible combinations will all work out--although the math all seems okay. The responsibility for an accurate return is yours, so please check your figures carefully.

Please send me any corrections, improvements or comments. Typing this one in is a real bear, so if you're short on time and hate to debug, CCN readers can get a copy of the program tape from me for \$7.95. The address is RD 1, Box 475, Mars, PA 16046. Please note that personal checks must clear before shipment; so if you're in a hurry, please send a money order. In the meantime, many happy returns!

```

5 CLS
10 PRINT@ 71, "1981 INCOME TAX"
15 PRINT@416, "WARREN S. NAPIER"
30 PRINT@448, "RD 1, MARS, PA 16046
35 FOR X= 1 TO 2000NEXT
40 CLSPRINT TAB(8)"1981 INCOME TAX
"
45 PRINTPRINT TAB(13)"MENU"
50 PRINTPRINT"1.ALL"
60 PRINT"2.INCOME"
70 PRINT"3.TAX COMPUTATION"
80 PRINT"4.SCHEDULE A"
90 PRINT"5.SCHEDULE G"
115 PRINTPRINTINPUT"NO ITEM YOU WANT (1-5)";I
120 IF I>5 THEN 40
130 ON I GOSUB1000,3000,6000,11000,12000
1000 REM*FILING STATUS*
1005 SOUND 75,2CLSPRINT@73, "FILIN
G STATUS"
1010 PRINTPRINT"1.SINGLE"
1020 PRINT"2.MARRIED-JOINT"
1030 PRINT"3.MARRIED-SEPARATE"
1040 PRINT"4.OTHER"
1050 PRINTINPUT"YOUR STATUS (1-4)"
;FS
1060 IF FS=1 THEN E=200FS$="SINGLE
"
1070 IF FS=2 THEN E=400FS$="MARRIE
D-JOINT"
1080 IF FS=3 THEN E=200FS$="MARRIE
D-SEPARATE"
1090 IF FS=4 THEN FS$="OTHER"GOTO
1130
1100 IF FS>4 THEN1050
1110 PRINTPRINT"YOUR STATUS: "FS$
1115 PRINTINPUT"PRESS <ENTER> TO C
ONTINUE";C$
1120 GOTO2000
1130 PRINTFOR X= 1 TO 3SOUND50,2
NEXT X
1135 PRINT"YOUR STATUS IS 'FS$' .
SEE IRS INSTRUCTIONS BEFORE PR
OCEEDING."
1140 PRINT"THIS PROGRAM DOES NOT AU
TOMATICALLY COMPUTE FOR 'OTH
ER'. YOU MAY GO TO 'SCHEDULE
A' AND EN- TER";
1150 PRINT" THE NUMBER OF DEPENDENT
S AT THE END OF THOSE CALCULAT
IONS."
1160 PRINTINPUT"PRESS <ENTER> TO C
ONTINUE";X
1170 GOTO40

```

```

2000 REM*EXEMPTIONS*
2010 SOUND 75,2CLSPRINT@42, "EXEMP
TIONS"
2015 PRINT@69, "TYPE <1> IF EXEMPTIO
N
APPLIES, <0> IF NO
T"
2020 PRINT@160, "1.SELF"; INPUTBA
2025 IF BA>1THEN2020
2030 PRINT@192, "2.SPOUSE"; INPUTBB
2035 IF BB>1THEN2030
2040 PRINT@224, "3.SELF--OVER 65"; I
NPUTBC
2045 IFBC>1THEN2040
2050 PRINT@256, "4.SPOUSE--OVER 65";
INPUTBD
2055 IFBD>1THEN2050
2060 PRINT@288, "5.SELF--BLIND"; INP
UTBE
2065 IFBE>1THEN2060
2070 PRINT@320, "6.SPOUSE--BLIND"; I
NPUTBF
2072 IFBF>1THEN2070
2074 CLSPRINTINPUT"NO. OF DEPENDE
NTS OTHER THAN SELF AND SPO
USE"; BG
2075 ET=BA+BB+BC+BD+BE+BF+BG
2080 PRINT@416, "NO. EXEMPTIONS = "E
T
2090 PRINT@448, "WANT ANY CHANGES (Y
/N)"; INPUTBG$
2100 IF BG$="Y"THEN2000 ELSE3000
3000 REM*CALCULATE GROSS INCOME*

```

ExIBMer (NOW RETIRED)
For the FIRST TIME — Makes available to the PUBLIC
His personal collection of superior programs for the

TRS-80 COLOR

SEE HOW THE PROFESSIONALS DO IT!!

12 MINUTE TALKING GRAPHIC DEMONSTRATION

SHOWS & TELLS "What's inside the TRS-80C and how it works" — PLUS 12 self-contained, auto-start, artistic, hi-res, full color graphic demonstrations in fantastic motion — all from 1 "CLOAD". A must if you want to show off your computer at its best to your friends and A MUST FOR ANY RS SALESMAN.

ALL GRAPHIC \$24.95
T

SLOT MACHINE

Looks, sounds, feels and operates as good as any BIG CASINO machine. Watch and listen to coins and arm drop

ALL GRAPHIC \$24.95
T/D

BLACK JACK

4 suits — 52 cards — card counter displays all remaining cards and % odds to HIT 21 — call for new deck before any BET or HIT

ALL GRAPHIC \$24.95
T/D

CRAP TABLE

up to 4 players can bet the full field before every roll of the dice — LAS VEGAS pay-off odds given on all 12 table bets

ALL GRAPHIC \$24.95
T/D

ALL 3 GAMES

Ideal for Rumpas Room, Clubs, Parties and Social Events.

\$49.95
T/D

CHECK-BOOK

1 to 3 banks and/or credit card accounts — Automatic Bank Reconciliation — Automatic IRS expense listings and tabulations by major or minor expense account number.

\$39.95
D

STOCK PORTFOLIO MGMT.

Complete from daily P & L to IRS 1040D — Charts all your stocks — DOW JONES CHARTS from 1900 to 1982 are worth this price alone.

\$69.95
D

ALL PROGRAMS ARE OVER 14K LONG!!

* T = 16K-EXTENDED

* D = 32K-DOS

* POSTAGE PAID

* ALLOW 2 TO 3 WEEKS


```

3005 IF E=0 THEN CLSPRINT "AMOUNT OF
      INTEREST/DIVIDEND EX- CLUSION
      S (SEE FILING STATUS"; INPUT E
3010 SOUND75,2CLSPRINT@12,"INCOME
      "
3015 PRINTPRINT "IF A LOSS IS SHOWN
      , USE <-> IN FRONT OF THE FIG
      URE"
3020 PRINTPRINT "LINE 7--WAGES, SALA
      RIES, TIPS, ETC."; INPUT CA
3030 PRINT "LINE 8A--INTEREST INCOME
      "; INPUT CB
3040 PRINT "LINE 8B--DIVIDENDS"; INP
      UTCC
3045 CD=CB+CC
3050 PRINT "LINE 8C--TOTAL = "CD
3060 PRINT "LINE 8D--EXCLUSIONS = "E
      'FROM1060-80
3065 CE=CD-EIFCE<0THENCE=0
3070 PRINT "LINE 8E--TAXABLE INCOME/
      DIVI- DENDS ="CE
3080 PRINT "LINE 9--REFUNDS: STATE/L
      OCAL TAXES"; INPUT C
      F
3090 PRINT "LINE 10--ALIMONY RECEIVE
      D"; INPUT CG
3100 PRINT "LINE 11--BUSINESS INCOME
      /LOSS (SCHED C)"; I
      NPUTCH
3110 PRINT "LINE 12--CAPITAL GAIN/LO
      SS (SCHED D)"; I
      NPUTCI
3120 PRINT "LINE 13--40% CAP. GAIN D
      ISTR."; INPUT CJ
3130 PRINT "LINE 14--SUPP. GAINS/LOS
      SES"; INPUT CK
3140 PRINT "LINE 15--TAXABLE PENSION
      S (NOT ON LINE 16)";
      INPUT CL
3150 PRINT "LINE 16A--OTHER PENSIONS
      /ANNUI- TIES"; INPUT
      CM
3160 PRINT "LINE 16B--TAXABLE FROM P
      .10 WORKSHEET";
      INPUT CN
3170 PRINT "LINE 17--RENT, ROYALTIES,
      ETC. (SCHED E)"; I
      NPUTCO
3180 PRINT "LINE 18--FARM INCOME/LOS
      S"; INPUT CP
3190 PRINT "LINE 19A--UNEMPLOYMENT C
      OMP."; INPUT CQ

3200 PRINT "LINE 19B--TAXABLE UNEMP.
      COMP. FROM P.10";
      INPUT CR
3210 PRINT "LINE 20--OTHER INCOME (P
      .11)"; INPUT CS
3220 CT=CA+CE+CF+CG+CH+CI+CJ+CK+CL+
      CN+CO+CP+CR+CS
3225 PRINT
3230 PRINT "LINE 21--GROSS INCOME =
      $"CT
3240 PRINTPRINT "WANT ANY CHANGES (
      Y/N)"; INPUT CA$
3250 IF CA$="Y" THEN 3000
3260 PRINTPRINT "CHOOSE <1> OR <2>:
      "
3270 PRINT TAB(5)"1.60 TO ADJUSTMEN
      TS "
3280 PRINT TAB(5)"2.ANOTHER SECTION
      "
3290 INPUT CU
3300 IF CU=1 THEN 4000 ELSE 40
4000 REM*ADJUSTMENTS TO GROSS INCOM
      E*
4010 SOUND75,2CLSPRINT TAB(2)"ADJ
      USTMENTS TO GROSS INCOME"
4020 PRINTPRINT "LINE 22--MOVING EX
      PENSE"; INPUT DA
4030 PRINT "LINE 23--EMPLOYEE BUSINE
      SS EX- PENSES (FORM
      2106)"; INPUT DB
4040 PRINT "LINE 24--IRA PAYMENTS";
      INPUT DC
4050 PRINT "LINE 25--KEOGH PAYMENTS"
      ; INPUT DD
4060 PRINT "LINE 26--INTEREST PENALT
      Y"; INPUT DE
4070 PRINT "LINE 27--ALIMONY PAID";
      INPUT DF
4080 PRINT "LINE 28--DISABILITY INCO
      ME EX- CLUSION"; INP
      UT DG
4090 PRINT "LINE 29--OTHER ADJUSTMEN
      TS (P.12)"; INPUT DH
4100 DT=DA+DB+DC+DD+DE+DF+DG+DH
4110 PRINTPRINT "LINE 30--TOTAL ADJ
      USTMENTS = $"DT
4120 PRINTPRINT "WANT ANY CHANGES (
      Y/N)"; INPUT DI$
4130 IF DI$="Y" THEN 4000
5000 REM*CALCULATE ADJUSTED GROSS I
      NCOME*
5010 SOUND 75,2CLSPRINTTAB(5)"ADJ
      USTED GROSS INCOME"

```

```

5015 IF CT=0 THEN SOUND 75,2PRINT
PRINT"WHAT IS YOUR GROSS INCOM
E (LINE 21)";INPUT CT
5020 IF DT=0 THEN SOUND 75,2PRINT
PRINT"WHAT ARE YOUR ADJUSTMENT
S (LINE 30)";INPUTDT
5030 FT=CT-DT
5040 PRINTPRINT"ADJUSTED GROSS INC
OME = $"FT
5050 IF FT<10000 THENPRINTPRINT"SE
E 'EARNED INCOME TAX CREDIT',
LINE 57--P.15"
5060 PRINTINPUT"PRESS <ENTER> TO C
ONTINUE";Z$
6000 REM*TAX COMPUTATION*
6010 SOUND75,2CLSPRINT TAB(8)"TAX
COMPUTATION"
6020 PRINTPRINT"DO YOU WANT TO ITE
MIZE (Y/N)";INPUTGA$
6030 IF GA$="Y"GOSUB11000
6035 IF FT=0THEN6040ELSE6045
6040 PRINTSOUND 75,2INPUT"WHAT IS
YOUR ADJUSTED GROSS IN- COME
(LINE 32A)";FT
6045 IF PT=0THEN6050ELSE6055
6050 INPUT"DEDUCTIONS FROM SCHED A"
;PT
6055 GB=PT' FROMLINE41SCHEDA
6060 PRINT"LINE 32B="GB
6065 GC=FT-GBPRINTPRINT"ADJ INCOM
E (LINE 32C) ="GC
6070 IF ET=0 THEN6080 ELSE 6090
6080 INPUT"NO. EXEMPTIONS";ET
6085 IF FS=0THENINPUT"FILING STATUS
(1-5)";FS
6090 GD=ET*1000PRINTPRINT"EXEMPTI
ON DEDUCTION (LINE 33) = "GD
6100 GE=GC-GDPRINTPRINT"TAXABLE I
NCOME (LINE 34) ="GE
6110 PRINTINPUT"PRESS <ENTER> TO C
ONTINUE";Z$
6120 CLSINPUT"WANT TO AVERAGE YOUR
INCOME";Z$
6130 IFZ$="Y"GOTO 6195
6140 PRINTINPUT"WANT TO USE TAX TA
BLES";Z$
6150 IFZ$="Y"THEN6160ELSE6170
6160 IF FS=2 OR FS=5 THEN6162ELSE61
70
6162 GP=INT(GE*.01+.5)*100
6163 IF GP>GE THEN6165
6164 IF GP<GE THEN6166
6165 GE=GP-25GOTO6167
6166 GE=GP+25
6167 V=GEGOSUB13000
6168 GF=VA+((GE-WN)*SN)GF=INT(GF+.
5)GF=GF-(.0125*GF)
6169 GF=INT(GF)PRINT"LINE 35--TAX
= $"GFGOTO 6200
6170 PRINT"COMPUTE TAX FROM TABLE/S
CHED ANDENTER ON LINE 35"
6180 PRINTINPUT"PRESS <ENTER> TO C
ONTINUE";Z$
6190 INPUT"ENTER TAX (LINE 35)";GF
GOTO 6200
6195 GOSUB12000
6197 GF=FD
6200 INPUT"LINE 36--ADDITIONAL TAXE
S";GG
6210 GH=GF+GG
6220 PRINTPRINT"TOTAL (LINE 37)="G
H
6230 PRINTINPUT"LINE 46--CREDITS";
GI
6240 GJ=GH-GI
6250 PRINTPRINT"LINE 47 ="GJ
6260 PRINTINPUT"LINE 53--OTHER TAX
ES";GK
6270 GL=GJ+GKPRINT"TOTAL TAX(LINE
54)="GL
6280 PRINTINPUT"FED TAX WITHHELD";
GM
6290 IF GM=GL THEN CLSPRINT@228,"N
O REFUND BUT
NOTHING OWED EITHER!
6300 IF GM>GL GOSUB 6400
6310 IF GL>GM GOSUB6600
6320 END
6340 GN=GL-GMPRINTPRINT"BALANCE O
WED = $"GN
6345 RETURN
6400 CLS4FORX=1TO2SOUND89,2
6410 SOUND133,2FOR Y=1TO10NEXTY
6420 NEXT X SOUND176,2FORX=1TO50N
EXTX
6430 FORX=1TO2SOUND176,2NEXTX
6440 FORX=1TO25NEXTX
6450 SOUND176,8FOR Y=1TO10
6460 CLS5PRINT@232,"
";
6520 FORX=1TO100NEXTX
6530 PRINT@232,"REFUND = $"GM-GL;
6540 FORX=1TO250NEXTXNEXTY
6550 PRINT@456,"HOW ABOUT THAT!";
6560 FORX=1TO1500NEXTX
6570 RETURN

```

```

6600 CLS0SOUND89,12
6605 FORX=1T075NEXT
6610 SOUND89,12FOR X=1T075NEXTX
6615 SOUND89,4FORX=1T050NEXT
6620 SOUND89,12FORX=1T075NEXT
6625 SOUND117,12FORX=1T050NEXT
6630 SOUND108,4FORX=1T050NEXT
6635 SOUND108,12FORX=1T040NEXT
6640 SOUND89,4FORX=1T040NEXT
6645 SOUND89,12FORX=1T050NEXT
6650 SOUND69,4FORX=1T050NEXT
6655 SOUND89,20
6660 PRINT@230,"BALANCE OWED: $"GL-
GM;
6665 FORX=1T02000NEXT
6670 RETURN
11000 REM*SCHED A*
11005 SOUND 75,2CLSPRINT@10,"SCHE
DULE A"
11010 IF FT=0THEN11015ELSE11020
11015 INPUT"ADJUSTED GROSS INCOME(L
INE 32A)";FT
11020 INPUT"LINE 1--1/2 (BUT NOT >
$150) MED INSUR PREM";
HA
11030 INPUT"LINE 2--MEDICINE/DRUGS"
;HB
11040 HC=INT((FT*.01)+.5)
11050 PRINT"LINE 3--1% ADJ GROSS IN
COME = "HC
11060 IF HC>=HB THEN11070ELSE11080
11070 HD=0GOTO 11090
11080 HD=INT(C-D)
11090 PRINT"DEDUCTIBLE MED/DRUGS (L
INE 4)=""HD
11100 PRINTINPUT"LINE 5--BALANCE I
NSUR PREM";HE
11110 INPUT"LINE 6A--DOCTORS,DENTIS
TS,ETC.";HF
11120 INPUT"LINE 6B--HOSPITALS";HG
11130 INPUT"LINE 6C--TRANSPORTATION
";HH
11140 INPUT"LINE 6D--OTHER (TOTAL)"
;HI
11145 HJ=HD+HE+HF+HG+HH+HI
11150 PRINTPRINT"TOTAL (LINE 7) ="
;HJ
11160 HK=INT((FT*.03)+.5)
11170 PRINT"LINE 8 ="HK
11180 IF HK>=HJ THEN 11190 ELSE 112
00
11190 HL=0GOTO 11210
11200 HL=INT((HJ-HK)+.5)
11210 PRINT"LINE 9 = "HL
11220 HT=HA+HL
11230 PRINTPRINT"TOTAL MED/DENTAL
(LINE 10) ="HT
11232 PRINTINPUT"ANY CHANGES";Z$I
FZ$="Y"THEN11000
11235 PRINTINPUT"PRESS <ENTER> TO
CONTINUE";Z$
11240 PRINTINPUT"LINE 11--STATE/LO
CAL INCOME TAX";JA
11250 INPUT"LINE 12--REAL ESTATE TA
X";JB
11255 INPUT"LINE 13A--GENERAL SALES
TAX";JC
11260 INPUT"LINE 13B--VEHICLE SALES
TAX";JD
11270 INPUT"LINE 14--PERSONAL PROPE
RTY TAX";JE
11280 INPUT"LINE 15--OTHER (TOTAL)"
;JF
11290 JT=JA+JB+JC+JD+JE+JF
11300 PRINTPRINT"TOTAL TAXES (LINE
16) ="JT
11302 PRINTINPUT"ANY CHANGES";Z$I
FZ$="Y"THEN11240
11305 PRINTINPUT"PRESS <ENTER> TO
CONTINUE";Z$
11310 PRINTINPUT"LINE 17--MORTGAGE
INTEREST";KA
11320 INPUT"LINE 18--CREDIT/CHARGE
CARDS";KB
11330 INPUT"LINE 19--OTHER (TOTAL)"
;KC
11340 KT=KA+KB+KC
11350 PRINTPRINT"TOTAL INTEREST (L
INE 20) ="KT
11352 PRINTINPUT"ANY CHANGES";Z$I
FZ$="Y"THEN11310
11355 PRINTINPUT"PRESS <ENTER> TO
CONTINUE";Z$
11360 PRINTINPUT"LINE 21 (EXCLUDIN
G LINE 22)-- CASH C
ONTRIBUTIONS";LA
11370 INPUT"LINE 21B--CASH CONTRIBU
TIONS-- OVER $3000
";LB
11380 INPUT"LINE 22--OTHER THAN CAS
H";LC
11390 INPUT"LINE 23--CARRYOVER";LD
11400 LT=LA+LB+LC+LD
11410 PRINTPRINT"TOTAL CONTRIBUTIO
NS ="LT

```


1981 Tax

```

11412 PRINTINPUT"ANY CHANGES";Z#I
      FZ#="Y"THEN11360
11415 PRINTINPUT"PRESS <ENTER> TO
      CONTINUE";Z#
11420 PRINTINPUT"LINE 25--LOSS BEF
      ORE REIMBURSE";MA
11425 IFMA<=0THEN11540
11430 INPUT"LINE 26--INSURANCE/OTHE
      R REIM-          BURSEMENT";
      MB
11440 IF MB>=MA THEN11450 ELSE 1146
      0
11450 MC=0GOTO 11470
11460 MC=INT((MA-MB)+.5)
11470 PRINT"LINE 27="MC
11480 IF MC>100THEN 11490 ELSE 1150
      0
11490 MD=100GOTO11510
11500 MD=MC
11510 PRINT"LINE 28="MD
11520 MT=MC-MD
11530 PRINTPRINT"TOTAL CASUALTY/TH
      EFT LOSSES ="MT
11532 PRINTINPUT"ANY CHANGES";Z#I
      FZ#="Y"THEN11420
11535 PRINTINPUT"PRESS <ENTER> TO
      CONTINUE";Z#
11540 PRINTINPUT"LINE 30A--UNION
      DUES";NA
11550 INPUT"LINE 30B--TAX RETURN PR
      EP FEE";NB
11560 INPUT"LINE 31--OTHER (TOTAL)"
      ;NC
11570 NT=NA+NB+NC
11580 PRINTPRINT"TOTAL MISC ="NT
11582 PRINTINPUT"ANY CHANGES";Z#I
      FZ#="Y"THEN11540
11585 PRINTINPUT"PRESS <ENTER> TO
      CONTINUE";Z#
11590 PRINTPRINT"LINE 33 = "HT
11600 PRINT"LINE 34 = "JT
11610 PRINT"LINE 35 = "KT
11620 PRINT"LINE 36 = "LT
11630 PRINT"LINE 37 = "MT
11640 PRINT"LINE 38 = "NT
11650 OT=HT+JT+KT+LT+MT+NT
11670 PRINT"LINE 39 ="OT
11675 PRINTINPUT"PRESS <ENTER> TO
      CONTINUE";Z#
11680 IFFS=2OR FS=5 THEN11710
11690 IF FS=1 OR FS=4 THEN11720
11700 IF FS=3 THEN11730
11710 DA=3400GOTO 11740
11720 DA=2300GOTO 11740
11730 DA=1700GOTO 11740
11740 PRINT"LINE 40 ="DA
11745 IFOA>OT THEN11770
11750 PT=OT-DA
11755 PRINT"LINE 41 ="PT
11760 PRINTINPUT"PRESS <ENTER> TO
      CONTINUE";Z#
11765 PRINTPRINT"IF YOU STARTED SC
      HED 'A' WITHOUTDOING THE TAX
      COMPUTATION SEC- TION, YOU A
      RE ABOUT TO RECEIVE AN ERROR
      MESSAGE. IF SO, TYPE <GOTO
      6000>."RETURN
11770 PRINTPRINT"ARE YOU REQUIRED
      TO ITEMIZE (SEEP.12)";INPUTO
      B#
11780 IF OB#="Y" THEN 11790 ELSE 11
      820
11790 PRINTPRINT"ENTER 'TC' ON LIN
      E 41 OF SCHED A"
11795 PRINT"THEN USE WORKSHEET ON P
      .12 OF INSTRUCTIONS"
11800 INPUT"PRESS <ENTER> TO CONTIN
      UE";Z#
11810 INPUT"LINE 32C = ";GCGOTO 60
      70
11820 GOTO6000
12000 'INCOME AVERAGING
12005 SOUND75,2CLSPRINT TAB(9)"SC
      HEDULE G"
12010 IFGE=0THEN12020ELSE12030
12020 PRINTINPUT"1981 TAXABLE INCO
      ME (LINE 34 OF 1040)";GE
12030 INPUT"1980 BASE INCOME";QD
12040 INPUT"NO. 1980 EXEMPTIONS";PA
12050 PRINTINPUT"1979 BASE INCOME"
      ;QC
12060 INPUT"NO. 1979 EXEMPTIONS";PB
12070 PRINTINPUT"1978 BASE INCOME"
      ;QB
12080 INPUT"NO. 1978 EXEMPTIONS";PC
12090 PRINTINPUT"1977 BASE INCOME"
      ;QA
12100 INPUT"NO. 1977 EXEMPTIONS";PD
12105 QE=PD*750QF=PC*750QG=PB*100
      OQH=PA*1000'EXEMPT DEDUCT
12110 CLSPRINTPRINT"LINE 2A:"
12120 PRINT TAB(5)"1977 - $"QE
12130 PRINT TAB(5)"1978 - $"QF
12140 PRINT"LINE 2B:"
12150 PRINT TAB(5)"1979 - $"QG
12160 PRINT TAB(5)"1980 - $"QH

```

```

12170 'BASE TAXABLE INCOMES
12180 RA=QA-QE'77
12190 RB=QB-QF'78
12200 RC=QC-QG'79
12210 RD=QD-QH'80
12220 PRINTPRINT"LINE 3:"
12230 PRINT TAB(4)"1977 - $"RA
12240 PRINT TAB(4)"1978 - $"RB
12250 PRINT TAB(4)"1979 - $"RC
12260 PRINT TAB(4)"1980 - $"RD
12270 PRINTPRINT"EXCLUDABLE INCOME
        OUTSIDE U.S."
12290 PRINT TAB(2);INPUT"1977 - $"
        ;SA
12300 PRINT TAB(2);INPUT"1978 - $"
        ;SB
12310 PRINT TAB(2);INPUT"1979 - $"
        ;SC
12320 PRINT TAB(2);INPUT"1980 - $"
        ;SD
12330 CLSPRINTPRINT"LINE 5:"
12340 'BASE INCOMES
12350 LH=RA-SALI=RB-SBLJ=RC-SCLK
        =RD-SD
12360 PRINT TAB(4)"1977 - $" ;LH
12370 PRINT TAB(4)"1978 - $" ;LI
12380 PRINT TAB(4)"1979 - $" ;LJ
12390 PRINT TAB(4)"1980 - $" ;LK
12400 PRINTPRINT"LINE 6 = $"GE
12410 INPUT"LINE 7 = $" ;LL
12420 LM=GE-LL
12430 PRINT"LINE 8 = $"LM
12440 INPUT"LINE 9--EXCESS COMMUNIT
        Y INCOME";LN
12450 LO=LM-LN
12460 PRINT"LINE10--ADJ TAXABLE INC
        OME ="LO
12470 INPUT"PRESS <ENTER> TO CONTIN
        UE";Z$
12480 LT=LH+LI+LJ+LK'TOTAL BASE
12490 CLSPRINT
12500 PRINT"LINE 11 = $" ;LT
12510 YY=INT((LT*.3)+.5)
12520 PRINT"LINE 12 = $"YY
12530 XX=GE-YY'AVERAGEABLE INCOME
12540 PRINT"LINE 13=$"XX
12550 IFXX<=3000 THEN12560ELSE12610
12560 FORX=1T03SOUND100,2
12570 NEXTXPRINTPRINT"YOU DO NOT
        QUALIFY FOR AVERAGING"
12580 PRINT"DO YOU WANT TO REDO SCH
        ED 6 (Y/N)"INPUTZ$
12600 IFZ$="Y"THEN12000ELSE40
12610 WW=INT((XX*.2)+.5)'20%LINE 13
12620 PRINT"LINE 14=$"YY
12630 PRINT"LINE 15=$"WW
12640 VV=YY+WW
12650 PRINT"LINE 16=$"VV
12660 PRINT"LINE 17=$"LN
12670 VT=VV+LN
12680 PRINT"LINE 18=$"VT
12690 PRINTINPUT"PRESS <ENTER> TO
        CONTINUE";Z$
12700 CLSPRINT"WORKING..."
12710 V=VTGOSUB13000'TAXLINE18
12720 UU=VA+((VT-WN)*SN)
12730 UU=INT(UU+.5)
12740 V=VVGOSUB13000'TAXLINE16
12745 UV=VA+((VV-WN)*SN)
12750 UV=INT(UV+.5)
12755 V=YYGOSUB 13000'TAXLINE14
12760 UT=VA+((YY-WN)*SN)
12765 UT=INT(UT+.5)
12770 CLSPRINTPRINT"LINE 19 = $"UU
12780 PRINT"LINE 20 = $"UV
12790 PRINT"LINE 21 = $"UT
12800 TT=UV-UT
12810 PRINT"LINE 22 = $"TT
12820 SS=TT*4
12830 PRINT"LINE 23 = $"SS
12835 IF LL=0 THEN12870ELSE12840
12840 V=GEGOSUB13000'TAXLINE6
12845 UW=VA+((GE-WN)*SN)UW=INT(UW+
        .5)
12850 V=LMGOSUB13000'TAXLINE8
12855 UX=VA+((LM-WN)*SN)UX=INT(UX+
        .5)
12860 PRINT"LINE 24 = $"UWPRINT"LI
        NE 25 = $"UX
12865 UY=UW-UXPRINT"LINE 26 = $"UY
12870 FF=UU+SS+UYPRINT"LINE 27 = $"
        FF
12880 EE=INT((FF*.0125)+.5)
12890 PRINT"LINE 28 = $"EE
12900 FD=FF-EE
12910 PRINTPRINT"TAX (LINE 29)=$"F
        D
12920 PRINTPRINT"ENTER TAX ON LINE
        35 OF 1040 AND CHECK SCHED 6
        BOX"
12925 PRINTINPUT"PRESS <ENTER> TO
        CONTINUE";Z$

```

```

12930 PRINTPRINT"IF YOU ENTERED SC
      HED B DIRECTLY (INSTEAD OF TH
      RU THE TAX COMPU- TATION PROC
      EDURE), YOU ARE ABOUTTO RECEI
      VE AN 'RG' ERROR NOTE.
12935 PRINT"IF SO, TYPE IN <GOTO 40
      >"
12940 PRINTINPUT"PRESS <ENTER> TO
      CONTINUE";Z$
12945 CLS
12950 RETURN
13000 REM#SCHED Y AND TAX TABLE
13010 IF V>500000 THEN PRINT"OOPS..
      .I CAN'T HANDLE THIS!      BET
      TER SEE AN ACCOUNTANT."GOTO6
      000
13020 DATA 0,3400,3400,5500
13030 DATA 5500,7600,7600,11900
13040 DATA 11900,16000,16000,20200
13050 DATA 20200,24600,24600,29900
13060 DATA 29900,35200,35200,45800
13070 DATA 45800,60000,60000,85600
13080 DATA 85600,109400,109400,1624
      00
13090 DATA 162400,215400,215400,500
      000
13100 FOR X=1 TO 16
13110 READ WN,WN(1)
13120 IFV>WN ANDV<WN(1)THEN13150
13130 NEXTX
13140 REM
13150 FORY=1 TO 16
13160 READ SN,VA
13170 NEXTY
13180 RESTORE
13190 PRINT @ 0, ""
13200 RETURN
13300 DATA 0,0,.14,0
13310 DATA .16,294,.18,630
13320 DATA .21,1404,.24,2265
13330 DATA .28,3273,.32,4505
13340 DATA .37,6201,.43,8162
13350 DATA .49,12720,.54,19678
13360 DATA .59,33502,.64,117504
13370 DATA .68,81464,.7,117504
13380 END

```

A white, snow-covered ski slope provides the scenario for Radio Shack's Skiing. The player views the slope as if he were actually on a slope in Switzerland, Vail or Lake Placid. After a voice says, "get ready, get set..." and a gunshot sounds, the skier begins his descent. The object is to maneuver between 29 pairs of gates on a course marked by red poles on each side. An illusion of up and down motion is created as the player skis through gates marked by alternating green and purple flags. A counter in the upper right hand corner counts the number of gates missed and an indicator on the left appears when the skier veers off course. A timer in the middle ticks off the seconds while the skier progresses. As the skier approaches the finish line, a crowd of spectators appears and the individuals grow larger as they become closer. The crowd "roars" upon the skier's completion of the course. The final time is displayed on the menu and the best time of the session is recorded.

Unlike a number of other program paks, the Skiing joystick controls are easy to use. The game has two types of controls--simple and complex. Simple joystick controls use a front-back motion to control speed. Side-to-side motion is used to direct the skier's path through the gates. Complex joystick controls use the same side-to-side steering as the simple controls; however, the speed of the skier is determined by several different factors. First, the button on the joystick acts as ski poles by giving the skier a shove down the slope. Second, hills, slopes, moguls and turns affect the skier's speed. For example, going uphill and making wide turns will naturally take more time than going downhill and making tighter turns.

Overall, Skiing is a truly entertaining game. We particularly like the simulated up and down movement as we glided down the course. Also, actually seeing the course from the skier's viewpoint rather than watching a skier as he descends the mountain provides a unique and challenging effect.

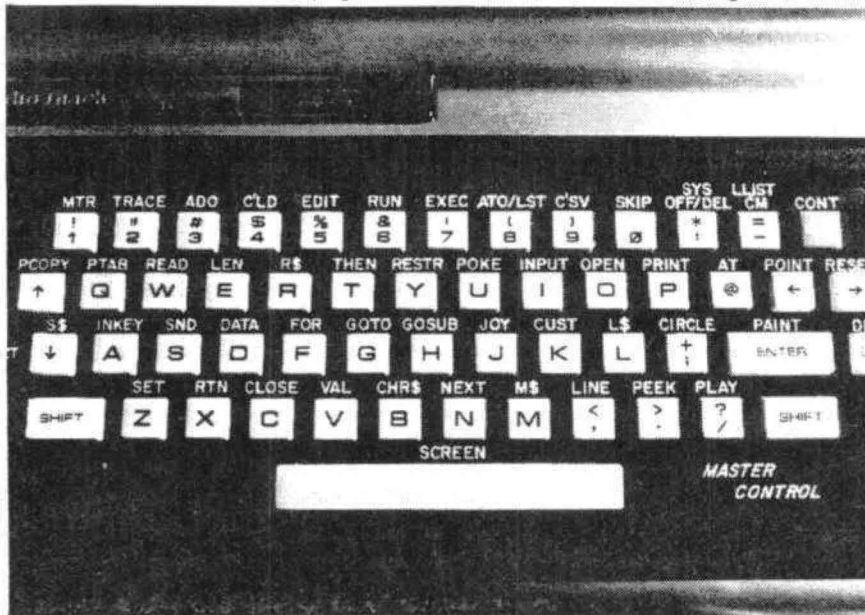
A few aspects of the game bothered us, however. First, missing just one gate or straying off course if only for an instant disqualifies the player's time from being the top score. Perhaps a penalty for each miss would have been a better solution. In addition, the step between simple to complex controls is a huge one. No skill level exists in between.

Skiing is certainly one of the best Program Paks we've seen from Radio Shack so far. We hope Radio Shack continues to produce new software for the CC with the same degree of excellence as Skiing.

For Your Color Comp

MASTER CONTROL

Copyright ©1981 Soft Sector Marketing, Inc. - Written by A. Swartz



Requires 16-32K

1. 50 preprogrammed color keys. Standard and Extended commands.
2. Direct control of motor, track and audio from keyboard.
3. Automatic line numbering.
4. Programmable Custom Key.
5. Direct Run Button.
6. Keyboard overlay for easy program use.
7. Easy entry of entire commands into computer.

Load Master Control into your machine then either type in a BASIC program or load one in from tape to edit. Cuts programming time by 50% or more **\$24.95**

Coming for Christmas!

COLOR BONANZA

50 Programs for the Color Computer.
Less than \$1.00 a program!

Some 4K, some 16K, some extended BASIC, some for non-extended.

Games - Personal Programs - Utility Programs
List **\$49.95**

Expected shipping date December 1 or before.
Any prepaid orders received before that date will pay only **\$39.95**

CONCENTRATION & CONNECT UP 4

Twin Pack - 16K Extended
\$14.95

Expected Shipping Date - December 1 or before

From Other Vendors

Space Invaders, Spectral Associates	\$21.95
Space War*, Spectral Associates, *req. Joy Sticks	\$21.95
Meteoroids*, Spectral Associates, *req. Joy Sticks	\$21.95
Battle Fleet, Spectral Associates	\$14.95
Space Traders, Spectral Associates	\$14.95
Software Development System Rom Pack	\$89.95
C Bug Monitor	\$29.95
Disassembler (Source Generator)	\$49.95
Color Computer News (Latest Issue)	\$2.50

Animated Hangman

He winks... he blinks... he almost lives!
An outstanding game for the whole family!



Non Extended BASIC
Requires 16K
\$12.95

C.O.D. - certified check, M.O. or cash only. Most orders shipped next day. All orders must have shipping included. Please add 2% or \$2.50, whichever is higher for shipping. Michigan residents, please add 4% tax. Add extra \$1.50 for C.O.D. Personal checks take 3 weeks to clear. Out of the country orders add \$10.00 extra shipping.
*TRS-80 is a product of Radio Shack, div. of the Tandy Corp.

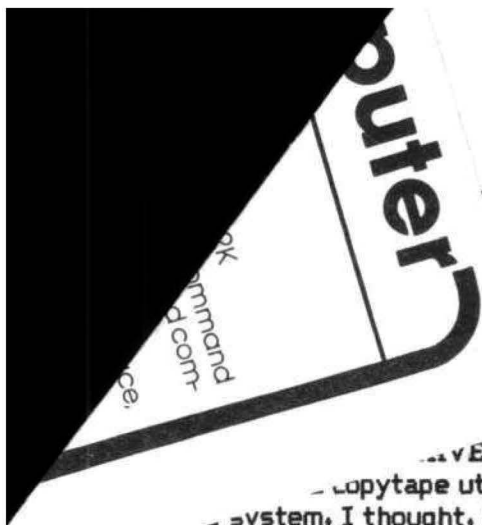
SSM SOFT SECTOR MARKETING,
INCORPORATED
6250 Middlebelt • Garden City, Michigan 48135
Order Line **800-521-6504**
Michigan Orders & Questions **313-425-4020**

THE AMERICAN EXPRESS
The American Express® Card
Don't leave home without it™



TAPE TO COLOR DISK

by Jack L. Aker



The first week of the project started to load programs to disk. For the next eight weeks, I wrote forward programs and programs to show where the programs were executed. I worked in with the SAVEM command, the COPYTAPE utility included in the system, I thought. Now where did I find the Owners Manual? Here it is. Quick, to the commands list. What? No COPYTAPE? Here's a copy command. Hmm, says it works if you have two drives. Why couldn't I use my cassette recorder for the second drive? Well now, if you could do that you wouldn't want a second drive would you? Oh, I see now, I have to write my own COPYTAPE command don't I?

Almost two months later I had my copytape command done, in the form of machine language program which I called TPTODSK. This effort involved disassembly of the operating system. The Owners Manual gives the machine language calls to DSKCON, which allows you to write individual sectors on the disk. But, then you have to do the file allocation table update to show which granules are allocated, and also you have directory entries to make. I thought it would be easier if I let the Operating System take care of those things.

I had previously disassembled the BASIC and EXTENDED BASIC code to do a tape to tape copy

program in machine language, so this job wouldn't be too difficult, I thought. It turns out the Disk Operating System does a lot of subroutine calls to BASIC and EXTENDED BASIC code to take advantage of routines already in the machine. This adds a further level of complexity to disassembly of the Disk Operating System. To understand what is happening in a particular area of code you must go to the called subroutine to find out what happens there. That subroutine may call some other routine, which tests a location and makes a branch decision. You can get lost quickly and forget how you got to where you are in the code.

The machine language program for tape to disk takes advantage of major functions in the Disk Operating System. All references to \$CXXX addresses in the EQU table at the beginning of the listing are to Disk rom code. The subroutines are those used by the SAVE and SAVEM commands for the disk. The \$AXXX addresses are calls to the BASIC interpreter rom code. Taking advantage of these calls makes the using program much smaller, as well as easier to debug.

I used a MICROWORKS SDS80C Editor Assembler to assemble the code in the listing which follows. If you don't have an assembler I would recommend the one I have used. Since no ram is needed by the assembler editor it allows assembly of very large programs.

For those of you who don't want to key in or assemble the program, I will supply a cassette tape with the object code for \$11.95 postpaid.

```

0001 0600                                NAM TPTODSK
*
* (C) 1981 J.L. AKER
* 12/29/81 11:20AM
*
0002 0600                                ORG $E00
0003 0E00                                BLKTYP EQU $7C
0004 0E00                                BLKLEN EQU $7D
0005 0E00                                BLKADR EQU $7E
0006 0E00                                CLOSE EQU $A42D          CLOSE FILE
0007 0E00                                GETBLK EQU $A701          SYNC-READ
0008 0E00                                BLKIN EQU $A70B          READ
0009 0E00                                CSRDON EQU $A77C          SYNC
0010 0E00                                MTROFF EQU $A7E9
0011 0E00                                MTRON EQU $A7CA
0012 0E00                                CLS EQU $A92B            CLEAR SCREEN
0013 0E00                                BINSTR EQU $C297          "BIN" ADR
0014 0E00                                OPFIL EQU $C956          OPEN FILE
0015 0E00                                TOBFR EQU $CB52          WRITE "A"
    
```

Tape to Color Disk

0016	0E00	C601	START	LDB #1	
0017	0E02	BDA99E		JSR \$A99E	AUDIO ON
0018	0E05	BDA92B		JSR CLS	
0019	0E08	B654		LDA #'T	TAPE IND
0020	0E0A	BDA30A		JSR \$A30A	PRINT
0021	0E0D	B620		LDA ##20	BLANK
0022	0E0F	BDA30A		JSR \$A30A	
0023	0E12	308D00D4		LEAX HEADER,PCR	GET ADR
0024	0E16	9F7E		STX <BLKADR	FOR HEADER
0025	0E18	BDA701	A@	JSR GETBLK	SYNC-READ
0026	0E1B	262F		BNE ERROR	
0027	0E1D	0D7C		TST BLKTYP	HEADER=0
0028	0E1F	26F7		BNE A@	LOOP TILL HDR
0029	0E21	7D0EF3		TST FILMOD	CHK FILMODE
0030	0E24	2BF2		BMI A@	ASCII=\$FF->SKIP
0031	0E26	9F7E		STX <BLKADR	DATA ADR
0032	0E28	C608		LDB #8	NAME LENGTH
0033	0E2A	108E094C		LDY ##94C	ADR FOR NAME
0034	0E2E	308D00BB		LEAX HEADER,PCR	NAME ADR
0035	0E32	A680	B@	LDA ,X+	
0036	0E34	BDA30A		JSR \$A30A	PRINT NAME
0037	0E37	A7A0		STA ,Y+	SAVE NAME
0038	0E39	5A		DECB	
0039	0E3A	26F6		BNE B@	LOOP 8 TIMES
0040	0E3C	BDA77C		JSR CSRDON	GET IN SYNC
0041	0E3F	BDA70B	C@	JSR BLKIN	READ REC
0042	0E42	260B		BNE ERROR	
0043	0E44	0D7C		TST BLKTYP	
0044	0E46	2B0A		BMI RDFIN	EOF=\$FF
0045	0E48	9F7E		STX <BLKADR	NEXT BLK ADR
0046	0E4A	20F3		BRA C@	READ MORE BLKS
0047	0E4C	BDA7E9	ERROR	JSR MTROFF	
0048	0E4F	7EA619		JMP \$A619	I/O ERROR
0049	0E52	BDA7E9	RDFIN	JSR MTROFF	STOP CASSETTE
0050	0E55	BD05		BSR WRTIT	DO THE DISK
0051	0E57	7FFF40		CLR \$FF40	STOP DRIVE
0052	0E5A	20A4		BRA START	DO IT AGAIN
* WRITE THE FILE TO DISK					
0053	0E5C	B644	WRTIT	LDA #'D	DISK IND
0054	0E5E	B70400		STA \$400	TO SCREEN
0055	0E61	327A		LEAS -6,S	MAKE ROOM ON S
0056	0E63	AF62		STX 2,S	SAVE END ADR
0057	0E65	308D0090		LEAX DATA,PCR	ADR
0058	0E67	AF64		BTX 4,S	TO STACK
0059	0E6B	AE8D0086		LDX EXECADR,PCR	EXEC ADR
0060	0E6F	AFE4		STX 0,S	STACK IT
0061	0E71	8EC297		LDX #BINSTR	GET BIN STR
0062	0E74	108E0954		LDY ##954	EXT ADR
0063	0E78	C603		LDB #3	
0064	0E7A	A680	A@	LDA ,X+	MOVE IN EXT
0065	0E7C	A7A0		STA ,Y+	

Tape to Color Disk

```

0066 0E7E 5A          DECB
0067 0E7F 26F9       BNE A@
0068 0EB1 CC0200     LDD ##200          FILETYPE & MODE
0069 0EB4 EDA4       STD ,Y             @ $957
0070 0EB6 7D0EF2     TST FILTYP        BASIC = 0?
0071 0EB9 2610       BNE BIN           NO, BINARY

* BASIC PROGRAM OPEN
0072 0EBB 8E4153     LDX ##4153        "AS" FOR EXT
0073 0EBE BF0955     STX $955          STO IT
0074 0E91 7F0957     CLR $957          FILTYPE IS BAS
0075 0E94 BDC956     JSR OPFIL         OPEN FILE
0076 0E97 86FF       LDA ##FF          1ST BYTE
0077 0E99 2004       BRA W1

* MACHINE LANGUAGE OPEN
0078 0E9B BDC956     BIN JSR OPFIL      OPEN FILE
0079 0E9E 4F         CLRA

0080 0E9F 8D43       W1 BSR WRTA        WRITE 1ST BYTE
0081 0EA1 EC62       LDD 2,S           END+1
0082 0EA3 A364       SUBD 4,S          CALC FILE LENGTH
0083 0EA5 1F02       TFR D,Y           COUNT IN Y REG
0084 0EA7 8D39       BSR WRTAB         WRITE 2 BYTES
0085 0EA9 7D0957     TST $957         CHK FILETYPE
0086 0EAC 2706       BEQ D@           IS BASIC
0087 0EAE ECBD0045    LDD LOADA,PCR    LOAD ADR
0088 0EB2 8D2E       BSR WRTAB         WRITE 2 BYTES
0089 0EB4 30BD0041    D@ LEAX DATA,PCR  DATA ADR
0090 0EBB A680       B@ LDA ,X+
0091 0EBA BDCB52     JSR TOBFR        A REG TO BFR
0092 0EBD 313F       LEAY -1,Y        DECR COUNT
0093 0EBF 26F7       BNE B@           LOOP TILL DONE
0094 0EC1 7D0957     TST $957         IS BASIC ?
0095 0EC4 2605       BNE ML           NO, FIN MACH LANG
0096 0EC6 3536       PULS A,B,X,Y    DUMP STACK
0097 0ECB 7E0ED7     JMP DONE         FINISHED BASIC

* END OF FILE FOR MACH LANGUAGE
0098 0ECB 86FF       ML LDA ##FF
0099 0ECD 8D15       BSR WRTA         WRITE EOF BYTE
0100 0ECF 4F         CLRA
0101 0ED0 5F         CLRB
0102 0ED1 8D0F       BSR WRTAB        WRITE 2 BYTES
0103 0ED3 3536       PULS A,B,X,Y    EXEC ADR->D
0104 0ED5 8D0B       BSR WRTAB        WRITE 2 BYTES

* CLEAR BASIC PROGRAM START
0105 0ED7 4F         DONE CLRA
0106 0ED8 5F         CLRB
0107 0ED9 9E19       LDX $19          GET BASIC START
0108 0EDB ED84       STD ,X           CLR BASIC ADR
0109 0EDD A782       STA ,-X          CLEAR BASIC ZERO
0110 0EDF 7EA42D     JMP CLOSE

```

Tape to Color Disk

```
* WRITE ONE OR TWO BYTES
* TO DISK FILE BUFFER

0111 OEE2 BD00      WRTAB  BBR WRTA
0112 OEE4 BDCB52   WRTA    JSR TOBFR
0113 OEE7 1EB9     EXG   A,B
0114 OEE9 39       RTS

0115 OEEA          HEADER RMB 15
0116 OEF9          FILTYP EQU HEADER+8
0117 OEF9          FILMOD EQU HEADER+9
0118 OEF9          EXEC   EQU HEADER+11
0119 OEF9          LOAD   EQU HEADER+13
0120 OEF9          DATA  EQU *
0121 OEF9          END   START

BIN    OE9B  BINSTR C297  BLKADR 007E  BLKIN  A70B
BLKLEN 007D  BLKTYP 007C  CLOSE  A42D  CLS    A92B
CSRDON A77C  DATA  0EF9  DONE   0ED7  ERROR  0E4C
EXEC   0EF5  FILMOD 0EF3  FILTYP 0EF2  GETBLK A701
HEADER 0EEA  LOAD   0EF7  ML     0ECB  MTROFF A7E9
MTRON  A7CA  OPFIL  C956  RDFIN  0E52  START  0E00
TOBFR  CB52  W1     0E9F  WRTA   0EE4  WRTAB  0EE2
WRTIT  0E5C
```

NOTHING FANCY — JUST GOOD SOFTWARE

GRAPHIC SCREEN PRINT PROGRAM

For use with TRS-80® Line Printer VII and VIII. We think ours is better. Supports all PMODES (not just two color ones). Distinguishes between colors by a shift of dot patterns - horizontal boundaries are easily seen. Relocatable code will allow you to use all of your 32K machine. Shift image anywhere between left and right margin or even cut off the left part of your image if you like. It does require that you have an eight bit serial interface to your printer (which is fully supported by Color BASIC release 1.1).

\$7.95 In machine language for greatest speed.

BOTH PROGRAMS require Extended Color BASIC and are delivered on cassette.

MATH TUTOR

Starts with addition and multiplication fact drill. Then goes on to full addition, subtraction, multiplication, and division at four levels of difficulty. Requires student to think through process step by step and make carry and regroup decisions. Provides correction of repeated errors and audio or visual rewards for good performance.

\$13.95 In BASIC

WE WANT YOUR SUGGESTIONS! Let us know what software you need. We don't promise to develop it, but if we do, you will be offered it at one half our retail price. No obligation on your part!

TRS-80® is a trademark of Tandy Corp.

Custom Software Engineering, Inc.

807 Minutemen Causeway
Cocoa Beach, Florida 32931
(305) 783-1083

Add \$1.00 per order for shipping.
Florida residents add 4% sales tax.
Return within two weeks if not completely satisfied.



For VISA and Master Card orders:
Include type, account number,
expiration date and phone number.
Sorry! No COD's.

80-U.S.

THE TRS-80 USERS JOURNAL

80-U.S. Journal is a monthly publication for the TRS-80 computer owner. The Journal covers Business, Scientific, Educational, and Recreational areas.

80-U.S. will keep you up to date on new products, software and hardware. Each issue will have listings of programs, reviews, tutorials. 80-U.S. is the complete "How to" Journal for the TRS-80!

If you haven't taken a look at 80-U.S., here is a no-risk opportunity to do it now. Become a trial subscriber now under the protection of a full money-back guarantee!



PLEASE enroll me as a trial subscriber to 80-U.S. and bill me just \$16. I understand I reserve the right to cancel my subscription any time, for any reason, and receive a refund for the balance of my subscription. If I should decide to cancel after receiving my first issue I will simply mark my bill *cancel* and keep the first issue *FREE!*

Name _____

Address _____

City _____ State _____ Zip _____

I prefer to enclose payment now.

Visa/MC # _____

Exp. Date _____

Canada & Mexico: \$25 per year, other foreign subscriptions \$30 surface mail, \$72 per year via airmail.

80-U.S. Journal
3838 South Warner Street
Tacoma, Washington 98409
(206) 475-2219

Order # CCN

32K RAM FOR FREE!!!

By Frank Hogg

"How to run Pascal, C, and Cobal, not to mention X-FORTH, 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. 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 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, untouched 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. Similarly 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 has developed 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. All FLEX compatible software will run as is without patches or modifications. 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.

If this isn't enough, we are planning to have the powerful OS-9 system on the Color Computer in the same way by early summer. Besides supporting several high level languages like Basic09, Pascal, C, and Cobal, OS-9 is Multi-User, Multi-Tasking, and Multi programming as well. The CC and OS-9 will be more powerful than all of the other TRS-80's put together!

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 has been 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.

In addition, the OS-9 operating system is being installed now and when finished (early summer '82') will allow the use of Radio Shack disk software, FLEX software, and OS-9 software on the same computer by merely changing Disks. No other computer made has those capabilities!

This article was prepared, using a preliminary version of the FLEX package, on a COLOR COMPUTER.

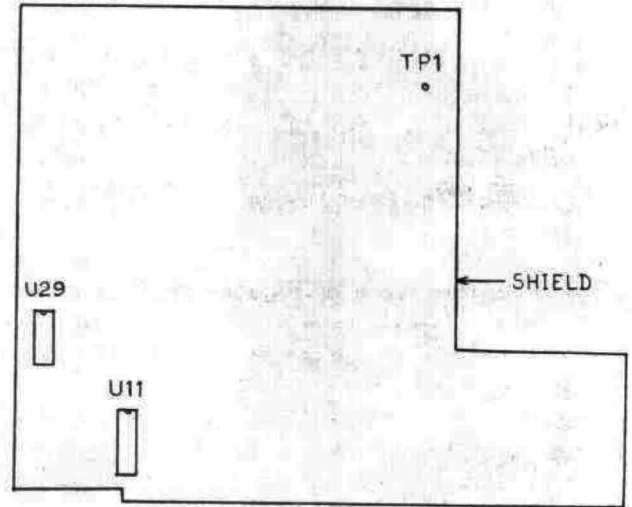


FIGURE 2. LOCATION OF COMPONENTS

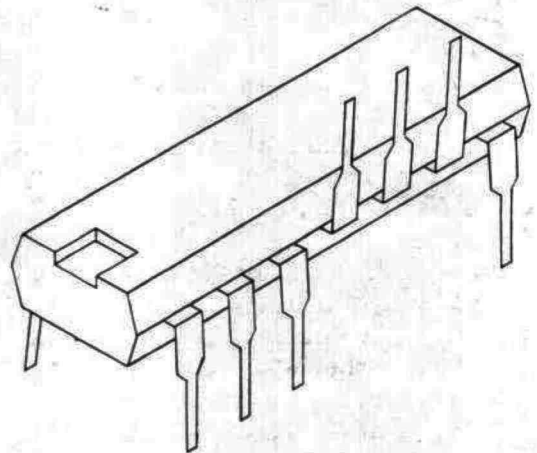


FIGURE 3. MODIFIED 74LS02 PACKAGE

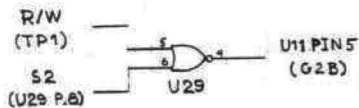


FIGURE 1. MODIFICATION TO TRS80CC FOR 64K RAM

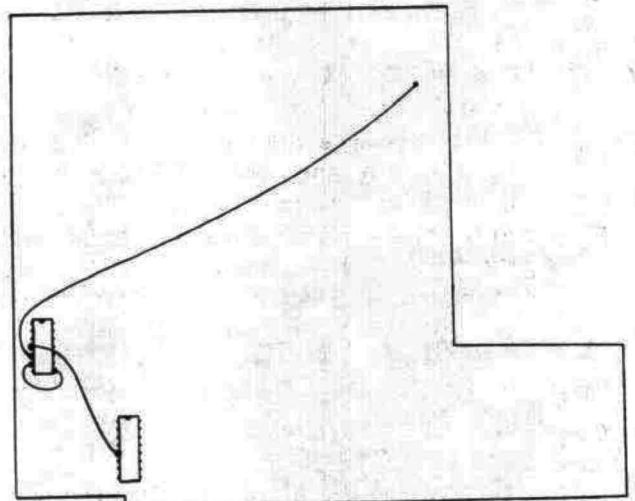


FIGURE 4. INSTALLED MODIFICATION

64K CC

```

100 ' THIS PROGRAM TESTS THE
110 ' MEMORY CHIPS IN A MODIFIED
120 ' 32K TRS80 COLOR COMPUTER
130 ' FOR FUNCTIONALITY. THE
140 ' TEST DETERMINES WHETHER
150 ' FULL 64K RAM CHIPS WERE
160 ' USED IN THE RAM UPGRADE.
170 '
180 ' REQUIRES EXTENDED COLOR
190 ' BASIC.
200 '
210 ' REQUIRES ABOUT 65 SECONDS
220 ' TO SET UP THE MACHINE
230 ' LANGUAGE PROGRAM BEFORE
240 ' RUNNING.
250 '
260 ' RESERVE RAM
270 '
280 CLEAR 256,&H3FFB
290 '
300 ' SET UP MACHINE LANGUAGE
310 ' PROGRAM
320 '
330 GOSUB 1660 : SA=H
340 GOSUB 1660 : EA=H
350 GOSUB 1660 : E0=H
360 GOSUB 1660 : E1=H
370 '
380 ' SA=START ADDRESS
390 ' EA=END ADDRESS
400 ' E0=ENTRY POINT 0
410 ' E1=ENTRY POINT 1
420 '
430 FOR A=SA TO EA
440 GOSUB 1660
450 POKE A,H
460 NEXT A
470 '
480 DEFUSR0=E0
490 DEFUSR1=E1
500 '
510 ' TEST THE MEMORY
520 '
530 X=USR0(0)
540 '
550 ' CHECK FAILURE ADDRESS
560 '
570 FA=PEEK(&H3FFC)*256
+PEEK(&H3FFD)
580 '
590 IF FA=0 THEN 760 : ' END TEST
600 '
610 ' REPORT FAILURE
620 '
630 DW=PEEK(&H3FFE) : ' DATA WRIT
640 DR=PEEK(&H3FFF) : ' DATA READ
650 '
660 PRINT "AT ADDRESS ";HEX$(FA)
670 PRINT "WROTE ";HEX$(DW);
"; READ ";HEX$(DR)
680 '
690 ' RESUME TEST
700 '
710 X=USR1(0)
720 '
730 GOTO 550
740 '
750 '
760 ' END OF TEST
770 '
780 END
790 '
800 ' MACHINE LANGUAGE PROGRAM
810 '
820 DATA 4000,4054
830 DATA 4000,4046
840 '
850 ' THE MACHINE LANGUAGE
860 ' MEMORY TEST PROGRAM IS AS
870 ' FOLLOWS:
880 '
890 ' E0 ORCC #50 DIS. INTS.
900 ' STA %FFDF MAP TYPE 1
910 ' LDX #4100
920 ' P1 CLR ,X+ CLEAR TO ZERO
930 ' CMPX %FF00
940 ' BNE P1
950 ' CLR %3FFE DW
960 ' LDX #4100
970 ' P2 LDA ,X
980 ' BNE RE REPORT ERROR
990 ' P2A LDA %FF
1000 ' STA ,X+
1010 ' CMPX %FF00
1020 ' BNE P2
1030 ' COM %3FFE DW
1040 ' LDX #4100
1050 ' P3 LDA ,X
1060 ' CMPA %FF
1070 ' BNE RE REPORT ERROR
1080 ' P3A LDA #00
1090 ' STA ,X+
1100 ' CMPX %FF00
1110 ' BNE P3
1120 ' LDX #0000 END TEST
1130 ' RE STX %3FFC FA
1140 ' STA %3FFF DR
1150 ' STA %FFDE MAP TYPE 0
1160 ' ANDCC %AF ENBL INTS.
1170 ' RTS
1180 '
1190 ' RESUME TESTING
1200 '
1210 ' E1 ORCC #50 DIS. INTS.
1220 ' STA %FFDF MAP TYPE 1
1230 ' LDX %3FFC FA
1240 ' TST %3FFE DW
1250 ' BEQ P2A
1260 ' BRA P3A
1270 '
1280 ' ACTUAL CODE
1290 '
1300 DATA 1A,50
1310 DATA B7,FF,DF
1320 DATA 8E,80,00
1340 DATA 8C,FF,00
1350 DATA 26,F9
1360 DATA 7F,3F,FE
1370 DATA 8E,41,00
1380 DATA A6,84
1390 DATA 26,21
1400 DATA 86,FF
1410 DATA A7,80
1420 DATA 8C,FF,00
1430 DATA 26,F3
1440 DATA 73,3F,FE
1450 DATA 8E,41,00
1460 DATA A6,84
1470 DATA 81,FF
1480 DATA 26,0C
1490 DATA 86,00
1500 DATA A7,80
1510 DATA 8C,FF,00
1520 DATA 26,F1
1530 DATA 8E,00,00
1540 DATA BF,3F,FC
1550 DATA B7,3F,FF
1560 DATA B7,FF,DE
1570 DATA 1C,AF
1580 DATA 39
1590 '
1600 DATA 1A,50
1610 DATA B7,FF,DF
1620 DATA BE,3F,FC
1630 DATA 7D,3F,FE
1640 DATA 27,C6
1650 DATA 20,D9
1660 '
1670 ' READ A HEX NUMBER TO H
1680 '
1690 READ A$
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

```

Comment Corner
 by Andrew Phelps
 The Micro Works

The following are comments which could be added to a disassembly listing of the Color Computer Rom. The subject this time is floating point

arithmetic. Given below are the routines which do addition, subtraction, multiplication, and division in the Basic interpreter.

Variables, areas, and routines -

Addr	Comments
----	-----
0003	GENERAL COUNTER
0013	PRODUCT AREA
004F	FPAC1 EXPONENT
0050	FPAC1 MANTISSA
0054	FPAC1 SIGN
005B	NORMALLY ZERO
005C	FPAC2 EXPONENT
005D	FPAC2 MANTISSA
0061	FPAC2 SIGN
0062	SIGN COMPARISON
0063	EXTENDED PRECISION BYTE
00AB	EXTENDED PRODUCT AREA
B9B9	SUBTRACT: FPAC1 = [X] - FPAC1
B9C2	ADD: FPAC1 = [X] + FPAC1
B9E2	UNNORMALIZE [X]
B9FB	ADD MANTISSAS
BA1C	NORMALIZE RESULT & LEAVE
BA39	SET FPAC1 TO +0.0
BAC5	CONSTANT: +1.0
BACA	MULTIPLY: FPAC1 = [X] * FPAC1
BB2F	MOVE [X] TO FPAC2
BB48	ADD EXPONENTS
BB7D	CONSTANT: +10.0
BB8F	DIVIDE: FPAC1 = [X] / FPAC1
BC14	MOVE [X] TO FPAC1
BC35	MOVE FPAC1 TO [X]
BC4A	MOVE FPAC2 TO FPAC1

Line-by-line comments -

Addr	Comments
----	-----
B9B9	MOVE [X] TO FPAC2
B9BC	CHANGE SIGN OF FPAC1
B9BE	CHANGE SIGN COMPARE FLAG
B9C0	GO ADD FPAC2 TO FPAC1
B9C2	MOVE [X] TO FPAC2

B9C5	TEST EXP OF FPAC1
B9C6	IF ZERO, JUST COPY FPAC2
B9CA	X -> FPAC2
B9CD	COPY EXP OF FPAC2
B9CF	TEST EXP OF FPAC2
B9D0	IF ADDING ZERO, JUST RTS
B9D2	SUBTRACT EXPONENTS
B9D4	IF SAME, NO NORMALIZATION
B9D6	SKIP IF EXP1 > EXP2
B9D8	RESULT EXP = EXP2
B9DA	SIGN OF FPAC2
B9DC	TO SIGN OF RESULT
B9DE	X -> FPAC1
B9E1	EXP DIFFERENCE NEGATIVE
B9E2	ARE WE A WHOLE BYTE OFF?
B9E4	GO MOVE BY BYTES
B9E6	CLEAR EXTENDED BYTE
B9E7	GET FIRST SHIFT STARTED
B9E9	GO MOVE BY BITS
B9EC	CHECK SIGN COMPARISON
B9EE	IF SAME SIGN NO COMPLEMENT
B9F0	COMPLEMENT THE MANTISSA
B9F2	.. OF THE SMALLER NUMBER
B9F4	.. TO SUBTRACT BY ADDING
B9F6	.. (NOTE "COM" SETS CARRY)
B9F8	COMPLEMENT EXTENDED BYTE
B9F9	+1 FOR TWO'S COMP
B9FB	SAVE EXTENDED BYTE
B9FD	GET LSB OF FPAC1
B9FF	ADD LSB OF FPAC2
BA01	STORE IN LSB OF FPAC1
BA03	AND THE SAME FOR THE REST
BA15	CHECK SIGN COMPARE AGAIN
BA16	IF SAME, GO SHIFT RIGHT
BA18	IF CARRY SET, GO NORMALIZE
BA1A	SIGN WRONG, GO COMPLEMENT
BA1C	INITIALIZE SHIFT COUNTER
BA1D	CHECK MSB (HIGH BYTE)
BA1F	IF NOT ZERO, GO SHIFT BITS
BA21	MOVE EVERYTHING UP A BYTE
BA2D	GET EXTENDED BYTE
BA2F	MOVE INTO LOW BYTE
BA31	CLEAR EXTENDED BYTE
BA33	COUNT 8 BITS SHIFTED
BA35	HAVE WE SHIFTED 40 BITS?

Comment Corner

BA37	IF NOT, GO CHECK FOR MORE	BAA0	MOVE TO LOWEST
BA39	FORGET IT; SET FPAC1 TO 0	BAA2	MOVE ALL BYTES DOWN ONE
BA3A	SET EXPONENT TO 0	BAAA	NORMALLY 0 (USED BY IFIX)
BA3C	SET SIGN TO +	BAAC	CLEAR MSB (UPPER BYTE)
BA3E	RETURN	BAAE	COUNT 8 BITS MOVED
BA3F	GO UNNORMALIZE (NO EFFECT IF B=0 EXCEPT LDA 0063)	BAB0	IF MORE TO MOVE, LOOP
BA41	CLEAR CARRY	BAB2	GET EXTENDED BYTE TO A
BA42	GO BACK UP AND ADD	BAB4	CORRECT COUNTER BACK DOWN
BA44	COUNT UP ONE BIT SHIFTED	BAB6	IF ZERO, RETURN
BA45	GET BIT FROM EXTENDED BYTE	BAB8	MOVE RIGHT BY ONE BIT
BA47	ROTATE INTO MANTISSA	BABA	ROTATE WHOLE MANTISSA
BA49	ROTATE WHOLE MANTISSA	BAC0	ROTATE EXTENDED BYTE
BA4B	BY ONE BIT	BAC1	COUNT BITS SHIFTED
BA4D	TO TRY FOR "1" IN LAST BIT	BAC2	LOOP IF MORE TO SHIFT
BA4F	IF STILL 0, KEEP SHIFTING	BAC4	RETURN
BA51	GET EXPONENT	BAC5	0.1 (BASE 2) * 2 ¹ = 1.0
BA53	SAVE SHIFT COUNT	BAC6	SIGN = +
BA55	EXP = EXP - SHIFT COUNT	BACA	MOVE [X] TO FPAC2
BA57	SAVE NEW EXPONENT	BACC	GO RETURN IF TIMES ZERO
BA59	IF UNDERFLOW, GO CLEAR	BACE	CALCULATE EXP OF RESULT
BA5B	SKIP NEXT INSTRUCTION	BAD0	CLEAR A
BA5C	IF CARRY, GO ROTATE RIGHT	BAD2	CLEAR PRODUCT AREA
BA5E	CHECK BIT FOR ROUNDING UP	BADA	GET LOWEST BYTE OF FPAC1
BA60	DON'T "CLR A"; SAVE CARRY	BADC	MULTIPLY BY FPAC2
BA62	CLEAR EXTENDED BYTE	BADE	GET BYTE SHIFTED OFF END
BA64	GO CHECK ROUNDING UP	BAE0	SAVE IT
BA66	INCREMENT EXPONENT	BAE2	GET NEXT BYTE OF FPAC1
BA68	CHECK OVERFLOW	BAE4	MULTIPLY BY FPAC2
BA6A	ROTATE THE MANTISSA	BAE6	GET BYTE SHIFTED OFF END
BA6C	.. BACK ONE BIT TO GET	BAE8	SAVE IT
BA6E	.. THE HIGHEST "1" BIT	BAEA	ETC.
BA70	.. BACK INTO THE MANTISSA	BAFA	MOVE PRODUCT AREA TO FPAC1
BA72	IF NO CARRY, RETURN	BAFD	NORMALIZE AND EXIT
BA74	ROUND UP: INCREMENT	BB00	IF BYTE = 0 JUST GO SHIFT
BA76	IF CARRY, GO SHIFT AGAIN	BB02	SET CARRY
BA78	RETURN	BB03	GET TOP BYTE OF PRODUCT
BA79	CHANGE SIGN	BB05	ROTATE MULTIPLIER
BA7B	COMPLEMENT MANTISSA	BB06	IF ZERO, WE'VE DONE ALL 8
BA7D	(THIS IS A 2'S COMP SINCE	BB08	IF CARRY CLEAR, DON'T ADD
BA7F	IT FALLS THROUGH TO THE	BB0A	GET LAST BYTE OF PRODUCT
BA81	INCREMENT ROUTINE)	BB0C	ADD LAST BYTE OF FPAC2
BA83	GET LOWER TWO BYTES	BB0E	STORE IN PRODUCT
BA85	INCREMENT	BB10	GET NEXT BYTE
BA87	SAVE LOWER TWO BYTES	BB12	ETC.: ADD ALL BYTES
BA89	IF NOT EQUAL, NO CARRY	BB20	ROTATE RIGHT PRODUCT
BA8B	GET UPPER BYTES	BB21	STORE MSB OF PRODUCT
BA8D	INCREMENT	BB23	ROTATE WHOLE PRODUCT 1 BIT
BA8F	SAVE UPPER TWO BYTES	BB25	ETC.: ROTATE ALL BYTES
BA91	RETURN	BB2B	CLEAR CARRY
BA92	ERROR CODE FOR "OV"	BB2C	LOOP FOR EIGHT BITS
BA94	GO TO ERROR ROUTINE	BB2E	RETURN
BA97	PRODUCT AREA - MOVE 1 BYTE		
BA9A	GET LOWEST BYTE		
BA9C	MOVE TO EXTENDED BYTE		
BA9E	GET NEXT TO LOWEST BYTE		

Comment Corner

BB2F	GET TWO BYTES OF MANTISSA	BBC8	IF ON LAST ONE, SET B=\$40
BB31	SAVE SIGN	BBCA	IF NOT LAST ONE, SET B=\$01
BB33	SET TOP BIT OF MANTISSA	BBCC	RESTORE CARRY BIT
BB35	PUT INTO FPAC2	BBCE	IF CARRY, SUBTRACT DIVISOR
BB37	GET SIGN	BBDO	SHIFT DIVIDEND LEFT
BB39	EXCLUSIVE-OR FPAC1'S SIGN	BBD2	ROTATE LEFT ALL
BB3B	STORE INTO SIGN-COMPARE	BBD4	.. FOUR
BB3D	GET OTHER 2 BYTES OF MANT.	BBD6 BYTES.
BB3F	PUT INTO FPAC2	BBD8	IF CARRY SET, SUBTRACT OK
BB41	GET EXPONENT	BBDA	IF MINUS, NEED TO COMPARE
BB43	STORE IN FPAC2; LEAVE IN A	BBDC	NO CARRY; NEED NO COMPARE
BB45	GET OTHER EXPONENT IN B	BBDE	SUBTRACT FROM DIVIDEND
BB47	LEAVE	BBE0	... THE DIVISOR.
BB48	TEST EXPONENT OF FPAC2	BBE2	AND SAVE AS DIVIDEND
BB49	TIMES 0? GO CLEAR FPAC1	BBE4	ETC.: ALL FOUR BYTES
BB4B	ADD TO OTHER EXPONENT	BBF6	GO BACK & SHIFT DIVIDEND
BB4D	ROTATE RIGHT FOR A MOMENT	BBF8	B=\$40 ("DONE" MARKER)
BB4E	ROTATE BACK TO CHECK OV	BBFA	BACK UP TO RESTORE CARRY
BB4F	SKIP IF OVER- OR UNDERFLOW	BBFC	MOVE THE LAST TWO BITS
BB51	CORRECT TOP BIT	BBFD	.. OF THE RESULT
BB53	SAVE AS NEW EXP	BBFE TO THE LEFT END OF B
BB55	IF ZERO, UNDERFLOW	BBFF	SAVE AS EXTENDED BYTE
BB57	GET X-OR OF SIGNS	BC01	MOVE RESULT TO FPAC1
BB59	USE AS SIGN OF RESULT	BC03	NORMALIZE AND LEAVE
BB5B	RETURN	BC06	ERROR CODE: "/0"
BB5C	(SOME CODE CALLED ONLY BY	BC08	GO TO ERROR ROUTINE
	THE EXP ROUTINE AT \$8501)	BC0B	GET TWO BYTES OF RESULT
BB61	REMOVE RETURN ADDRESS	BC0D	MOVE TO FPAC1
BB63	IF PLUS, SET FPAC1 TO ZERO	BC0F	OTHER TWO BYTES
BB67	GO SAY "OVERFLOW"	BC11	TO FPAC1
BB8F	MOVE [X] TO FPAC2	BC13	RETURN
BB91	IF DIVIDE BY ZERO, ERROR	BC14	SAVE A REGISTER
BB93	NEGATE EXP1 SO SUBTRACTS	BC16	GET TWO BYTES OF MANTISSA
BB95	GO CALCULATE NEW EXPONENT	BC18	SAVE SIGN
BB97	ADJUST RESULT BY ONE	BC1A	SET FIRST BIT
BB99	IF OVERFLOW GO COMPLAIN	BC1C	SAVE IN FPAC1
BB9B	POINT X AT PRODUCT AREA	BC1E	CLEAR EXTENDED PRECISION
BB9E	INIT COUNTER FOR 4 BYTES	BC20	GET EXPONENT TO B
BBA0	COUNTER IN \$0003	BC22	GET LAST TWO BYTES
BBA2	1->B TO FLAG 8TH SHIFT	BC24	SAVE IN FPAC1
BBA4	FIRST BYTE FPAC1	BC26	STORE EXPONENT IN FPAC1
BBA6	COMPARE WITH FPAC2	BC28	RESTORE A AND RETURN
BBA8	SKIP IF GREATER OR LESS	BC35	GET EXPONENT
BBAA	GET NEXT BYTE	BC37	GIVE TO USER
BBAC	KEEP COMPARING	BC39	GET SIGN
BBAE	TILL ONE LARGER THAN OTHER	BC3B	SAVE ONLY SIGN BIT
BBB0	ETC.: COMPARE ALL BYTES	BC3D	COMBINE WITH FIRST BYTE
BBBC	IF ALL EQUAL, SET CARRY	BC3F	GIVE TO USER
BBBD	SAVE STATE OF CARRY BIT	BC41	GET NEXT BYTE
BBBF	ROTATE CARRY INTO QUOTIENT	BC43	GIVE TO USER
BBC0	IF MORE ROOM IN B, SKIP	BC45	GET LAST TWO BYTES
BBC2	SAVE QUOTIENT IN AREA	BC47	GIVE TO USER
BBC4	COUNT DOWN BYTES	BC49	RETURN
BBC6	SKIP IF WE'VE DONE ALL 4	BC4A	GET SIGN OF FPAC2

```

BC4C STORE IN FPAC1
BC4E GET EXP AND MANTISSA 1
BC50 STORE IN FPAC1
BC52 CLEAR EXTENDED BYTE
BC54 GET MANTISSA 2
BC56 STORE IN FPAC1
BC58 SET A = SIGN
BC5A GET MANTISSA 3 AND 4
BC5C STORE IN FPAC1
BC5E RETURN
    
```

QUESTION: How is a floating point number represented?

Each floating point number has an exponent, a mantissa, and a sign. The mantissa is a 32-bit binary fraction which is shifted by the number of bits specified by the exponent. For example, if the mantissa is .101, and the exponent is two, then the number is 10.1 (base 2) or two and a half.

What format are these numbers stored in in the Color Computer?

Each number takes five bytes. The first byte is the exponent; it has \$80 added to it so that \$7F means -1, \$80 means 0, \$81 means +1, etc. The other four bytes are the mantissa. Since the first bit of the mantissa is always 1, it is not needed and the sign bit is stored in its place.

Why is the first bit of the mantissa always 1?

Well, if it were zero, we could shift the whole mantissa left by one bit, and subtract one from the exponent, and end up with the same number. By convention, this is done to each number until the leftmost bit is a one. This is called "normalizing" a number.

What about zero?

Zero is a special case, since it cannot be normalized. To store zero, the exponent is set to zero. This leaves exponents in the range \$01 to \$FF, which correspond to -127 through +127.

What are FPAC1 and FPAC2?

These names stand for "Floating Point Accumulator" 1 and 2. This is where the arithmetic is done. FPAC1 is in memory locations \$004F through \$0054, and FPAC2 is in \$005C through \$0061.

Why are the floating point accumulators six bytes instead of five?

When a number is moved into these areas, the format is changed slightly. To make it easier to do operations, the sign bit is moved to a separate byte (\$0054 or \$0061) and the first bit of the mantissa (which is a 1) is put back.

How do I do floating point arithmetic from assembly language programs?

The floating point routines are called with the X register pointing to a five byte area. You would call a routine to move a number into FPAC1, then call routines to add to FPAC1, multiply by it, etc. Finally, you would call the routine to move FPAC1 to wherever you wanted your result.

How is addition performed?

The mantissa of the smaller number is shifted until its exponent is the same as that of the other number. Then the mantissas can be added. The

Comment Corner

resulting number is then shifted left or right to normalize it (that is, to put its leftmost bit into the first bit of the mantissa).

How is subtraction performed?

The sign of FPAC1 is reversed, and the numbers are then added.

What is the Extended Byte?

To minimize round-off errors, an extra byte is used in the calculations, which can end up being shifted into the mantissa when the result is normalized. The leftmost bit of this extra byte is also used to do rounding on the mantissa instead of truncating it at four bytes.

How is multiplication performed?

The exponents are added, and the mantissas are multiplied. The two 32-bit mantissas yield a 64-bit product. Although the lower 32 bits of this is saved (at \$00AB) only the upper 32 bits are used as the resulting mantissa.

How is division performed?

The exponents are subtracted, and the mantissas are divided.

How about functions like SIN or SQR?

These are "derived functions" and do their work by calling the four main arithmetic functions.

Now! AN AFFORDABLE LIGHT PEN
FOR YOUR TRS-80 COLOR COMPUTER.

Only **\$39⁹⁵** each

Programs for home, school, office include:

- Shuttle
- Bible Quiz
- Hangman
- Meteor Shower (Joysticks required)
- Chex (balance your bank account)
- Tic Tac Toe
- Moon Lander (from inside the LEM)
- Photon (Artificial Intelligence)
- Night Flight

Many more! From Kindergarten through graduate courses. All cassettes \$4⁰⁰ each. Write for free list.

MOSES ENGINEERING COMPANY

Route 7, Regent Drive
Greenville, S.C. 29609
(803) 834-7974

Color Computing

A one stop shopping center for your Color Computer, distributes software and hardware from some of the best companies around — The Micro Works, Spectral Associates, Computerware, Soft Sector Marketing, Exatron, Color Computer News and others.

And, by special arrangement with CCN, now producing CCN Sampler series at \$7.95 each.

Dealer Inquiries Invited
Write for Catalog
Back Issues of CCN Available

3166 Ardmore Ave.
Southgate, CA 90280
(213) 564-7458

COLOR COMPUTER SYSTEMS SOFTWARE

EDITOR/ASSEMBLER

The Micro Works Software Development System (SDS80C) is a complete 6809 editor, assembler and monitor package contained in one Color Computer program pack! Vastly superior to RAM-based assemblers/editors, the SDS80C is non-volatile, meaning that if your application program bombs, it can't destroy your editor/assembler. Plus it leaves almost all of 16K or 32K RAM free for your program. Since all three programs, editor, assembler and monitor are co-resident, we eliminate tedious program loading when going back and forth from editing to assembly and debugging!

The powerful screen-oriented Editor features finds, changes, moves, copies and much more. All keys have convenient auto repeat (typamatic), and since no line numbers are required, the full width of the screen may be used to generate well commented code.

The Assembler features all of the following: complete 6809 instruction set; complete 6800 set supported for cross-assembly; conditional assembly; local labels; assembly to cassette tape or to memory; listing to screen or printer; and mnemonic error codes instead of numbers.

The versatile ABUG monitor is a compact version of CBUG, tailored for debugging programs generated by the Assembler and Editor. It features examine/change of memory or registers, cassette load and save, breakpoints and more. **SDS80C Price: \$89.95**

MODEM COMMUNICATIONS

Make your Color Computer an intelligent printing terminal with off-line storage! The Microtext module is just what you'll need for:

- Talking to a timeshare system or information service
- Printing out what is received as it is received
- Saving received text to cassette tape
- Re-displaying the received text even while on-line
- Communications with other computers
- Using your computer as a general-purpose 300-baud terminal
- Downloading programs from other computers

The Microtext module is a program pack containing not only firmware but a second serial port so that both your printer and modem can be connected at the same time. Microtext can be configured for any serial printer that will work with the Color Computer, even if it requires line feeds! But even if you don't have a printer, you can keep a permanent copy of your data by storing to cassette tape. Also, any Radio Shack/Centronics-compatible parallel printer may be used by adding the Micro Works' PI80C parallel interface.

For those of you with special terminal applications, Microtext has selectable parity; it sends odd, even, mark or space. With mark parity (which is default) you can send to computers requiring either seven or eight bits. All 128 ASCII codes can be sent. Exchange programs with other Color Computer users! Basic programs may be downloaded from other computers or timesharing systems.

You'll find many uses for this versatile module! Available in ROMPACK, ready-to-use, for **\$59.95**.

MACHINE LANGUAGE

MONITOR TAPE: A cassette tape which allows you to directly access memory, I/O and registers with a formatted hex display. Great for machine language programming, debugging and learning. It can also send/receive RS232 at up to 9600 baud, including host system download/upload. 19 commands in all. Relocatable and reentrant. **CBUG Tape Price: \$29.95**

MONITOR ROM: The same program as above, supplied in 2716 EPROM. This allows you to use the entire RAM space. And you don't need to re-load the monitor each time you use it. The EPROM plugs into the Extended Basic ROM Socket or the Romless Pak I. **CBUG ROM Price: \$39.95**

SOURCE GENERATOR: This package is a disassembler which runs on the color computer and generates your own source listing of the BASIC interpreter ROM. Also included is a documentation package which gives useful ROM entry points, complete memory map, I/O hardware details and more. A 16K system is required for the use of this cassette. **80C Disassembler Price: \$49.95**

LEARN 6809!

6809 ASSEMBLY LANGUAGE PROGRAMMING, by Lance Leventhal, contains the most comprehensive reference material available for programming your Color Computer. **Price: \$16.95**

HARDWARE

PARALLEL O!

USE A PARALLEL PRINTER with your Color Computer! Adaptor box plugs into the serial port and allows use of Centronics/Radio Shack-compatible printers with parallel interface. Assembled and tested. **PI80C Price: \$69.96**

ROMLESS PAK I — is an empty program pack capable of holding two 2716 or 2732 EPROMS, allowing you up to 8K of program! The PC board inside comes with sockets installed, ready to go with the addition of your custom EPROMs. **Price: \$24.95**

SPARE PARTS — SAMs, 6809Es, RAMs, PIAs. Call for prices.

32K RAM!

MEMORY UPGRADE KITS: Consisting of 4116 200ns. integrated circuits, with instructions for installation. **4K-16K Kit Price: \$39.95. 16K-32K Kit (requires soldering experience) Price: \$39.95**

GAMES

Pak Attack — Try your hand at this challenging game by Computerware, with fantastic graphics, sound and action! Cassette requires 16K. **Price: \$24.95**

Berserk — Have fun zapping robots with this Hi-Res game by Mark Data Products. Cassette requires 16K. **Price: \$24.95**

Adventure — *Black Sanctum* and *Calixto Island* by Mark Data Products. Each cassette requires 16K. **Price: \$19.95 each.**

Star Blaster — Blast your way through an asteroid field in this action-packed Hi-Res graphics game! Available in ROMPACK; requires 16K. **Price: \$39.95**

THE MICRO
WORKS



GOOD STUFF!

MasterCharge/Visa Accepted
California residents add 6% tax.

P.O. BOX 1110, DEL MAR, CA 92014 [714] 942-2400

Running Machine Language Programs from Disks
by Tony Di Stefano

So, you just bought a disk drive for your color computer and a lot of your machine language programs will not work. They just hang-up, crash, or even write all over your diskette. If this happens to you, you have the 'incompatibility blues' and I have the cure. It's a short machine language program that locks your computer into believing that the disk drive is not there. For that matter, it will not even acknowledge EXTENDED BASIC. It, in fact, does a 'cold start' routine. Well, part of one anyway.

About the Program:

First I'll explain the program then I'll show you how to tack it on to your program (the one that won't run from disk).

The program starts up by clearing the screen. Then it loads and stores all the pointing hooks and other odds and ends that basic needs to operate. After that is done, it sets the start of a basic program to \$600 (the \$ meaning hex). Next, it checks if you have 16K or 32K memory. It also puts an RTI instruction to the non-maskable interrupt location. The stack is then moved to the top of memory and the logo "COLOR 1.0" is printed. It sets the reset hook to the same place basic does. It then clears location #0071. That location is hooked by the basic reset routine. If this location contains \$55, a warm start routine is done. If not, a cold start is done.

What is the difference you might ask? Well a cold start will do the same as powering down and powering back up. A warm start will just give control back to the keyboard. In my case, I want a cold start so that the disk drive and extended basic will be reconnected. I can then load another program.

Back to the end of my program. What? It doesn't end. All it has is NOPS. That is where your program comes in. This program is written in PIC (Position Independent Code) so you can tack this on to the front of any machine language program anywhere in RAM. If your program does not execute from the beginning of the program; ie, CSAVEM "PROG", &H2000, &H3000, &H2100, then the NOP'S should be changed to LBRA XX where XX = the offset between the end of my program and the EXEC point of yours.

After you have done all that, SAVE the whole program on DISK. When you load the program and EXEC it, all will run properly. When you press reset, your disk is reconnected and you are then able to load in another program.

One more point. If you want to go to basic, meaning no EXTENDED BASIC, add a JUMP \$A37C to the end of the program. You will get the color basic logo and have all the regular basic commands only. When you press reset DISK EXTENDED BASIC will return and you simply load your program. If you don't want to loose your program, change "CLR CHECK" opcode in the program to LDA #\$55, STA CHECK. Then when you press reset you will stay in BASIC and not loose your program. In this mode, if you want to return to DISK EXTENDED BASIC, you can either power down or in the command mode (no line number) POKE 113,0 and press reset.

One last note. This program is completely PIC, but make sure that you don't load it between \$600 and \$E00. DOS uses this area for a scratch pad. If you must use this area make sure you don't have any DISK I/O routine to do. Good Luck!

	NAM	POWER-UP	ROUTINE
CLS	EQU	\$A928	CLEAR SCREEN
TRAN	EQU	\$A59A	TRANSFER BLOCK
PRINT	EQU	\$B99C	PRINT TEXT
NEW	EQU	\$AD19	CLEAR PROGRAM
GET	EQU	\$B277	GET OPERAND
DATA1	EQU	\$A10D	BASIC DATA
STORE1	EQU	\$008F	START OF HOOKS
STORE2	EQU	\$010C	MORE HOOKS
STORE3	EQU	\$015E	START OF TRAPS
STORE4	EQU	\$02D9	ANTHER TRAP
END3	EQU	\$01A9	END OF TRAPS
BEGIN	EQU	\$0600	START OF BASIC
RAM	EQU	\$0019	START OF RAM
TES32K	EQU	\$4000	START OF 2ND
MEM32K	EQU	\$7FFE	TOP OF 32K
MEM16K	EQU	\$3FFE	TOP OF 16K
PNT1	EQU	\$0074	MEM POINTER
PNT2	EQU	\$0027	MEM LIMIT
PNT3	EQU	\$0023	BASIC LIMIT
PNT4	EQU	\$0021	TOP OF STACK
NMI	EQU	\$0109	NMI HOOK
TEXT	EQU	\$A146	LOGO TEXT
RESET	EQU	\$A0E8	RESET ROUTINE
REHOOK	EQU	\$0072	RESET HOOK
CHECK	EQU	\$0071	COLD START HOOK

Running Machine Language Programs from Di

START	JSR CLS	
	LDX #DATA1	BASIC DOWNLOAD
	LDU #STORE1	
	LDB ##1C	LENGTH
	JSR TRAN	TRANSFER DATA
	LDU #STORE2	
	LDB ##1E	LENGTH
	JSR TRAN	
	LDX #GET	OPERAND LOCATION
	STX 3,U	OPERAND HOOK
	STX 8,U	OPERAND HOOK
	LDX #STORE3	
	LDA ##39	RTS OPCODE
LOOP1	STA ,X+	
	CMPX #END3	
	BNE LOOP1	
	STA STORE4	ANTHER TRA
	LDX #BEGIN	
	CLR ,X+	
	CLR ,X	
	CLR 1,X	
	CLR 2,X	
	STX RAM	
TEST	LDX #TES32K	CHECK MEM
	LDB ,X	SAVE BYTE
	LDA ##AA	
	STA ,X	
	LDA ,X	
	CMPA ##AA	
	BNE LD16K	
	STB ,X	
	LDX #MEM32K	MEM=32K
	BRA MEM	
LD16K	LDX #MEM16K	MEM=16K
MEM	STX PNT1	
	STX PNT2	
	STX PNT3	
	LEAX \$FF38,X	
	STX PNT4	
	LDA ##3B	RTI OPCODE
	STA NMI	
	TFR X,S	MOVE STACK
	JSR NEW	
	ANDC ##AF	
	LDX #TEXT	
	JSR PRINT	PRINT LOGO
	LDX #RESET	RESET ROUTINE
	STX REHOOK	RESET HOOK
	CLR CHECK	
	NOP	
	NOP	
	NOP	
	END	

Computer Program Books for Beginners

Everything you need to know to get started programming your own computer. These handy books of programs, each jam-packed with easy-to-understand info for beginners, are crammed with hundreds of tips, tricks, secrets, hints, shortcuts, techniques, plus hundreds of tested ready-to-run programs. For the TRS-80 Color Computer. For the TRS-80 Pocket Computer and Sharp PC-1211, PC-1500, Casio FX-702P pocket computers, as well as for Apple and IBM.

Color Computer

101 Color Computer Programming Tips & Tricks, learn-by-doing instructions, hints, secrets, shortcuts, techniques, insights, for TRS-80 Color Computer, 128 pages **\$7.95**
55 Color Computer Programs for Home, School & Office, practical ready-to-run software with colorful graphics for TRS-80 Color Computer, 128 pages. **\$9.95**
55 MORE Color Computer Programs for Home, School & Office, sourcebook of useful type-in-and-run software with exciting graphics, for TRS-80 Color Computer, 112 pages. **\$9.95**
The Color Computer Songbook, 40 favorite pop, classical, folk & seasonal songs arranged for TRS-80 Color Computer; ready-to-run music programs, 96 pages. **\$7.95**
My Buttons Are Blue and Other Love Poems from the Digital Heart of An Electronic Computer, for poetry lovers, computer lovers, a high-tech classic, 66 heartwarming poems written by a TRS-80 Color Computer, 96 pages. **\$4.95**
Color Computer Coding Form, handy preprinted programming worksheets make writing software easy, fun, 40-sheet pad. **\$2.95**

Pocket Computer

101 Pocket Computer Programming Tips & Tricks, secrets, hints, shortcuts, techniques from a master programmer, 128 pages. **\$7.95**
50 Programs in BASIC for Home, School & Office, sourcebook of tested ready-to-type-in-and-run software for TRS-80 and Sharp pocket computers, 96 pages. **\$9.95**
50 MORE Programs in BASIC for Home, School & Office, ideal source for lots more useful software for TRS-80 and Sharp pocket computers, 96 pages. **\$9.95**
Murder in The Mansion and Other Computer Adventures, with 24 game programs: murder mystery, space, adventure, loads of fun for TRS-80 and Sharp pocket computers, 96 pages. **\$6.95**
Pocket Computer Programming Made Easy, new fast 'n easy way to learn BASIC, make your computer work for you, for TRS-80, Sharp, Casio pocket computers, 128 pages. **\$8.95**
35 Practical Programs for the Casio Pocket Computer, book of useful type-in-and-run software for the FX-702P, 96 pages. **\$8.95**
Pocket-BASIC Coding Form, preprinted program worksheets make writing programs a breeze; for TRS-80, Sharp, Casio pocket computers, 40-sheet pad. **\$2.95**
Universal BASIC Coding Form, programming worksheets for anybody writing in BASIC for any computer system, make writing program lines easy and fun. 40-sheet pad. **\$2.95**

Order direct from this ad. Send check or money order. Include \$1 shipping for each item ordered up to a maximum of \$3 shipping. Or write for our free catalog. Mail your order to:

ARCsoft Publishers

Post Office Box 132V
 Woodsboro, Maryland 21798

(301) 663-4444

TELEWRITER

Provides your COLOR COMPUTER with:

REAL LOWER CASE CHARACTERS ■ A POWERFUL TEXT FORMATTER
51 COLUMN × 24 LINE SCREEN DISPLAY ■ SPECIAL DRIVER FOR EPSON MX-80
ADVANCED CASSETTE HANDLING FEATURES ■ A SOPHISTICATED FULL-SCREEN TEXT EDITOR

and requires absolutely no hardware modifications

TELEWRITER

Telewriter is a powerful word processor designed specifically for the Color Computer. It can handle almost any serious writing job and it is extremely easy to use. It has all the advanced features you need to create, edit, store, format and print any kind of text. With Telewriter you can quickly produce perfect, finished copy for letters, reports, term papers, articles, technical documentation, stories, novels, screenplays, newsletters. It is also a flexible and efficient way to take notes or organize ideas and plans.

51 × 24 DISPLAY

The Color Computer is an incredibly powerful and versatile computer, but for text editing it has some major drawbacks. The small 32 character by 16 line screen format shows you too little of the text and, combined with its lack of lower case letters, bears little resemblance to the way text really looks on the page. Reverse video in place of lower case just adds confusion.

Telewriter eliminates these shortcomings with **no hardware modifications required**. By using software alone, Telewriter creates a new character set that has **real lower case letters**, and puts 24 lines of 51 characters on the screen. That's more on-screen characters than Apple II, Atari or TRS-80 Model III. That's more than double the Color Computer's standard display.

FULL SCREEN EDITOR

The Telewriter editor is designed for maximum ease of use. The commands are single key (or single key plus control key), fast, and easy to remember. There is no need to switch between insert modes and delete modes and cursor movement modes.

You simply type. What you type is inserted into the text at the cursor, on the screen. What you see on the screen is always the current state of your text. You can move quickly through the text with one key cursor movement in all 4 directions, or press the shift key simultaneously for fast, auto-repeat. You can jump to the top or bottom of the text, the beginning or end of a line, move forward or backward a page at a time, or scroll quickly up or down. When you type past the end of a line, the wordwrap feature moves you cleanly to the next.

You can copy, move or delete any size block of text, search repeatedly for any pattern of characters, then instantly delete it or replace it with another. Telewriter gives you a tab key, tells you how much space you have left in memory, and warns you when the buffer is full.

FORMAT FEATURES

When it comes time to print out the finished manuscript, Telewriter lets you specify: left, right, top, and bottom margins; line spacing and lines per page. These parameters can be set before printing or they can be dynamically modified during printing with simple format codes in the text.

Telewriter will automatically number pages (if you want) and automatically center lines. It can chain print any number of text files from cassette without user intervention. You can tell it to start a new page anywhere in the text, pause at the bottom of the page, and set the Baud rate to any value (so you can run your printer at top speed).

You can print all or any part of the text buffer, abort the printing at any point, and there is a "Typewriter" feature which allows you to type straight to your printer. Because

Telewriter lets you output numeric control codes directly (either from the menu or during printing), it works with any printer. There's even a special driver for the Epson MX-80 that lets you simply select any of its 12 fonts and do underlining with a single underline character.

CASSETTE HANDLER

Telewriter makes cassette as simple to use as possible. It will search in the forward direction til it finds the first valid file, so there's no need to keep retyping a load command when you are lost in your tape. You can save all or any part of the text buffer, and you can append pre-existing files to what you have in the buffer already. You can abort an append or filesearch without harming the program or the text in the buffer.

Telewriter will maintain compatibility with popular Color Computer disk systems, but, since it makes using cassette almost painless, you can still have a powerful word processor without the major additional cost of a disk.

AVAILABLE NOW

Telewriter turns your Color Computer into the lowest cost hi-power word processor in the world today. It runs in 16K or 32K (32K recommended) and is so simple you can be writing with it almost immediately. It comes with complete documentation and is fully supported by Cognitec. Telewriter costs \$49.95 (California residents add 6% tax). To order or request more information write:

Cognitec
704 Nob Ave.
Del Mar, Ca. 92014

Or call (714) 755-1258 (weekdays, Saturdays, and early morning). We will gladly answer your questions.

Interfacing a Printer to the Color Computer by Mark Rothstein

The BASIC routines on the Color Computer are capable of supporting a variety of non-standard serial interfaces via the RS232 port. I touched briefly on this point in my review of Micro Work's Disassembler last month. There I described how I modified DISASM to work with my printer. Actually my printer is a terminal and it does not perform a line feed when the computer sends a return. (Printers usually have this option). In this article I will describe how it is possible to use a completely non-standard printer (or other RS-232 device) with BASIC.

When BASIC performs a PRINT command, as part of its machine code it jumps to location \$167 where BASIC has reserved three locations and stored a return from subroutine (RTS=\$39) in the first location. This is done only during cold start, that is reset on power-up. It is possible to interrupt the PRINT command when it goes to this location! Insert a jump instruction to a user routine. This will make the computer execute the user routine before it performs the print. This user routine can do a lot of things - it can add a line feed to a return, as my routine did, or it can even do a complete code conversion, say from ASCII to Baudot!! This technique of incorporating a section of executable code in RAM like this is called leaving "hooks". You can see why the technique got this name and why it is so useful. The changes to RAM can be done via a POKE, or a program like Micro Works CBUG.

When I first discovered my printer problem, I called Radio Shack on their toll-free number and they sent the program excerpt which is given in Listing 1.

LISTING 1 Radio Shack Printer Patch

```
110 DATA 52,20,214,111,193,254
120 DATA 38,11,129,13,38,7,190
130 DATA 160,2,173,3,134,10,53
140 DATA 20,57
150 FOR D = 1000 to 1021
160 READ E: POKE D,E: NEXT E
190 POKE 1021,PEEK(359)
200 POKE 1022,PEEK(360)
210 POKE 1023,PEEK(361)
220 POKE359,126,:POKE360,3:POKE361,232
```

This is incomprehensible, isn't it? Well this program is equivalent to the machine language routine in Listing 2 and commented in Figure 1.

```
ORG $03E8
X006F EQU $006F
XA002 EQU $A002
P03E8 PSHS B,X
LDB <X006F
CMPB #$FE
```

```
BNE P03FB
CMPA #$0D
BNE P03FB
LDX XA002
JSR 3,X
LDA #$0A
P03FB PULS B,X
RTS
END
```

Figure 1 Some program comments are:

line 1 This output is the result of reverse assembling the Radio Shack code using DISASM.
line 2 This is the start of the user routine. \$3E8 = 1000

line 3 This is the address in RAM of a variable which indicates where the PRINT output is to be sent. For this, a -2 causes output to be sent to the RS-232 device, a -1 causes output to be sent to the cassette, a 0 causes a output to be sent to the screen and a 1 to 15 causes output to be sent to a file (for DISK BASIC).

Other user routines are possible. I had a letter from a reader whose printer performs a "return - line feed" only when it receives a "line feed". In this case, the routine given in Listing 3 should do the trick. Note that this routine doesn't bother to send the "return". It just exchanges it for a "line feed".

```
ORG $0167
X03E8 EQU $03E8
P0167 JMP X03E8
END
ORG $03E8
X0000 EQU $0000
X006F EQU $006F
P03E8 PSHS B
LDB <X006F
BNE P03F6
CMPA #$0D
BNE P03F6
LDA #$0A
P03F6 PULS B,PC
END
```

listing 3

The above routine is just fine for a single character exchange. What if the entire alphabet is to be changed, as in a code conversion? Then the general exchange routine given in Listing 4 should solve the problem. Here the user routine sets up a table and compares the given character against alternate table entries. When it finds a match it replaces the BASIC PRINT character with the adjacent table entry. The routine also checks to see if the character is not in the table.

Interfacing a Non-Standard Printer

In this version of the routine, if the character is no in the table a null (00) will be sent out instead. A user may adjust this to an appropriate value.

Happy Printing.

```

NAM PRCONV
*A PROGRAM TO ADAPT THE COLOR
*COMPUTER TO A NON-STANDARD
*RS-232 DEVICE
*(FOR INSTANCE,
*A NON-SUPPORTED PRINTER)
*
*THIS ROUTINE HAS BEEN EDITED &
*ASSEMBLED ON MICROWORKS SDS80C
*EDITOR/ASSEMBLER/MONITOR

*BASIC RAM ALLOCATIONS
DEST EQU $6F          DESTINATION
PRBAUD EQU $96        BAUD RATE
HOOK EQU $167

*SET THE ORIGIN FOR HIGH MEMORY
*IN A 16K SYSTEM
ORG $3F00
PRCONV PSHS B,X      SAVE THE B-ACC
LDB DEST            OUTPUT TO PRINT

CMPB #-2
BNE EXIT            NO. NORMAL RTN

*YES. OUTPUT TO THE PRINTER:
LDX #TABLE

*CHECK THE CHARACTER AGAINST
*THE TABLE ENTRY
CHKCHR CMPA 0,X
BEQ COMPAR

*DID NOT COMPARE. INCREMENT
*THE TABLE POINTER
LEAX 2,X
CMPX #TBLEND      END?

*IF NOT EQUAL, THEN
*GO CHECK THE NEXT TABLE ENTRY
BNE CHKCHR

*ERROR EXIT
LDA #0            SEND NULL
PULS B,X,PC      ERROR RETURN

*NORMAL RETURN
COMPAR LDA 1,X    SEND REPLACEMENT
EXIT PULS B,X,PC FIX UP STACK
    
```

```

*TABLE
TABLE FCC 'A
FCB $41
*
*
*
FCC 'Z
FCB $5A
*END OF TABLE
TBLEND EQU *
    
```

C. C. Writer

[Word Processing for the TRS-80 Color Computer]

Features Page Formatting, Block Moves, Tabs, Sentence Deletion and Insertion, Global Search and Replace, Centering, Indenting, Page Pause, ASCII Code Transmission, Justification, Scrolling Review, Keyboard Stops, and File Chaining. 16-32K cassette-\$35. 32K Disk-\$40 (Shipping late Feb.)

NEW! --> Check Rec Plus

Reconciles your Checkbook AND allows you to keep Memo Entries of cash and credit card expenses without affecting your Checkbook balance. The History Files may be Listed and Totaled by Income or Expense Category for Budgeting or Tax Preparation. On screen Trial Balancing and printed Trial Balances, Audit Trails, and Summaries (REQUIRES PRINTER). Current balance is displayed on the Screen Menu and Reports. Save yourself frustration and perhaps some money too.

--> Prices until April 15th: <--

16K Version-\$25, 32K with extensive prompts-\$30

16-32K Disk System with Prompts-\$35

All versions include System Binder, Cassettes or Disk with storage pockets, and documentation. (Shipping late Feb.)

For information or orders write:

TransTek BSC 2-2
194 Lockwood

Bloomington, IL 60108

† TRS-80 is a Trademark of Tandy Corporation

**TRS-80 COLOR COMPUTER
GAUNTLET!**

A fast action, machine language space game;
Brilliant colors, exciting sounds.....

Only \$10.00

For 4K (and larger)

Sold on cassette, joysticks required.

BRITT MONK, CDP
P.O. BOX 802
ELYRIA, OHIO 44036



CHEAP THRILLS

QUALITY SOFTWARE FOR TRS-80 COLOR!



ADVENTURES!!!!

For TRS-80, and COLOR-80. These Adventures are written in BASIC, are full featured, fast action, full plotted adventures that take 30-50 hours to play. (Adventures are inter-active fantasies. It's like reading a book except that you are the main character as you give the computer commands like "Look in the Coffin" and "Light the torch.")

Adventures require 16K on COLOR-80 and TRS-80. They sell for \$14.95 each.

ESCAPE FROM MARS

(by Rodger Olsen)

This ADVENTURE takes place on the RED PLANET. You'll have to explore a Martian city and deal with possibly hostile aliens to survive this one. A good first adventure.

PYRAMID (by Rodger Olsen)

This is our most challenging ADVENTURE. It is a treasure hunt in a pyramid full of problems. Exciting and tough!

TREK ADVENTURE (by Bob Retelle)

This one takes place aboard a familiar starship. The crew has left for good reasons - but they forgot to take you, and now you are in deep trouble.

DEATH SHIP (by Rodger Olsen)

Our first and original ADVENTURE, this one takes place aboard a cruise ship - but it ain't the Love Boat.

VAMPIRE CASTLE (by Mike Bassman)

This is a contest between you and old Drac - and it's getting a little dark outside. \$14.95 each.

SPACE SHUTTLE

ONE OR TWO PLAYER HIGH RES GAME - Your mission is to dock with an orbiting space platform - but you may have to land on the planetary surface for refueling first. A real value in a high res real time game. \$6.95.

KILLERBOT - (Available in 4K) - Real time action at 20 levels of difficulty as you run, sneak, and dodge your way through a bloody field of Killer Robots. Get across or die! Joysticks or Keyboard controls. TRS-80 COLOR (ANY BASIC 4K or more.). \$9.95.

SLASHBALL (Available in 4K) - This one is best described as a thinkers arcade game. It rewards fast reflexes and clear thinking - like nothing you have ever seen before. It is one of our best family games for one or two players. \$9.95.

TIMETREK - A REAL TIME, REAL GRAPHICS STARTRECK. See your torpedoes hit and watch your instruments work in real time. No more unrealistic scrolling displays! \$14.95.

STARFIGHTER - This one man space war game pits you against spacecruisers, battlewagons, and one man fighters, you have the view from your cockpit window, a real time working instrument panel, and your wits. Another real time goody. \$9.95

BATTLEFLEET - This grown up version of Battleship is the toughest thinking game available on OSI or 80 computers. There is no luck involved as you seek out the computers hidden fleet. A topographical toughie. \$9.95

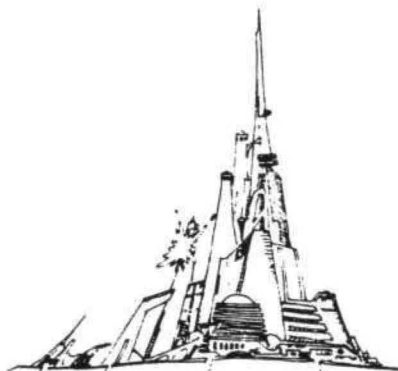
LABYRINTH - 16K EXTENDED COLOR BASIC - With amazing 3D graphics, you fight your way through a maze facing real time monsters. The graphics are real enough to cause claustrophobia. The most realistic game that I have ever seen on either system. \$14.95.



QUEST - A NEW IDEA IN ADVENTURE GAMES! Different from all the others, Quest is played on a computer generated map of Alesia. Your job is to gather men and supplies by combat, bargaining, exploration of ruins and temples and outright banditry. When your force is strong enough, you attack the Citadel of Moorlock in a life or death battle to the finish. Playable in 2 to 5 hours, this one is different every time. 16K COLOR-80 OR TRS-80 ONLY \$14.95.



SPACE ZAPPER - Protect your central Star Base from ships that attack from all four sides. Fast reflexes are required as the action speeds up. Great for kids or Dads. This game has high speed high resolution graphics and looks as if it just stepped out of the arcades. - 16K extended or 32K disk. BASIC TRS-80 Color only. \$14.95.



Please specify system on all orders

This is only a partial listing of what we have to offer. We have arcade and thinking games, utilities and business programs for the OS1 and TRS-80 Color. We add new programs every week. Send \$1.00 for our complete catalog.



TRS 80

2352 S. Commerce, Walled Lake, MI 48088
(313) 669-3110

TRS 80 COLOR

CC WORD PROCESSOR

by David M. Dacus

Since acquiring my Color Computer (CC) in December of 1980 I have been looking for software to allow the CC to be used as a tool rather than a toy. One of the most practical uses of any microcomputer is as a word processor. Six months ago (in the dark ages of the CC) nothing was available for the CC but a few Radio Shack games. This left two options, write a word processor myself, or modify a good existing program to operate on the CC. My lack of knowledge and ability precluded the first option, and at about that time I found a good BASIC word processor by Delmer Hinrichs in the May 1980 edition of 80 Microcomputing.

Mr. Hinrichs' program was written for a 16K tape based TRS-80 Model 1. The translation of the program to operate on the CC has turned out to be not as trivial as I had anticipated. It seems that the ASCII codes for all the non-alphanumeric keys are not identical between the Model I and the CC. It was necessary to interpret the ASCII values into the keyboard entries for the Model I, and then translate them into the correct ASCII values for the CC. There were other changes necessitated by errors or omissions in the original program. For example, the documentation stated that the SHIFT down arrow will center the text of the line and add a <do not justify> mark. This function was not in the original code for the ADD function. I made this and other corrections. Another article I found most helpful in modification of Mr. Hinrich's program was an adaption of the program for the TRS-80 Model II by Mike Kilroy in the July, 1981 80 Microcomputing. The changes from the original indicated areas I needed to check for necessary changes for the CC.

My modification of Mr. Hinrich's program is designed for a 32K CC with Radio Shack CCDOS, one disk drive, and an Epson MX-80 serial printer. It should be pretty straight-forward to modify to a 16K system by changing the size of the CLEAR statement, NL, and DIMENSION statements in line 20. The program can be converted from disk to tape storage by modifying the disk I/O formats in lines 1570-1660 and 1860-1940 to the Open'I' and OPEN 'I' -1 formats for tape I/O. If you are not using an Epson MX-80 printer you may find it necessary to modify lines 1740 and 1750 to match your printer. These values came from Howard Potvin's short article in the October 1981 80

Microcomputing.

The CC Word Processor provides very useful manipulating power for preparation of correspondence and technical writing. Text can be entered from the keyboard either into a new file or added to an existing file. It provides a means of moving lines of text from one part of the file to another inserting lines between lines of text, and replacing one line with another entered from the keyboard. Lines can be deleted from the text, and blank lines removed by a one letter command. Straight right margins can be produced by the Justify command. Line text can be shifted to the right margin for such purposes as entering the inside address for a letter, or the line text centered and <end of page> and/or <do not justify> marks added to the line. The line oriented editor has a full list of functions including changing, adding, deleting, and inserting characters including the control characters. The program stores text on disk, and recalls text on command. Stored text can be recalled for editing, re-formatting adding text to the file, and printing. The print format is user modifiable while in the program by entering the F command. Pagination includes printing the title and page number if desired. Single or multiple spacing is allowed. Since the program is written in BASIC it is user adaptable to meet any requirement. This article was prepared using the CC Word Processor.

The program is self prompting and if all else fails enter the command H for help. It lists all available command. The commands supported by the word processor are:

A - add - start a new file or add to an existing file.

B - blank - delete all blank line (lines with one blank space are not blank).

C - compile - move all text to fill the available space from the last line designated to the first line designated. Compile one paragraph at a time to avoid burying end of page and do not justify marks.

D - delete - deletes first designated line to last designated line and leaves blank lines (delete blank line with the B command).

E - edit - enters the edit mode (edit commands discussed below).

F - format - change the default format parameters for printing the file.

H - help - lists all available commands.

I - insert - inserts a line at the line designated.

1670 Valencia
Las Cruces, NM
88001

CC Word Processor

J - justify - right justifies all lines not marked <do not justify>.

K - kill - kills the letter or file in memory.

L - load - load a file from disk.

M - move - moves lines specified to a specified location. Location must be blank line (use I command to insert blank lines if necessary).

P - print - sends the file in memory to the printer using the format prescribed by command F.

R - replace - replaces specified line with keyboard input.

S - save - saves the file to disk.

V - video - displays the file on the screen.

X - exit - ends the program.

The ADD mode has some single key commands to perform certain functions. These are:

LEFT ARROW - moves cursor one space left and erases the last entry.

SHIFT LEFT ARROW - erases the whole line.

RIGHT ARROW - TABs the cursor five spaces to the right.

SPACE BAR - moves the cursor one space to the right.

ENTER - ends the current line and goes to the next line.

SHIFT RIGHT ARROW - moves all text on the line to the right margin and goes to the next line.

DOWN ARROW - ends the line and adds an <end of page- do not justify> mark.

CLEAR - ends the line and marks it <do not justify>.

SHIFT DOWN ARROW - centers the line of text and marks it <do not justify>.

SHIFT CLEAR - ends the ADD command.

The EDIT mode has some commands requiring further definition. Another feature of the edit mode that is slightly different is that all letter commands in EDIT are in lower case. If this bothers you simply change the appropriate ASCII values in line 790-1240. The commands for the editor are:

#LEFT ARROW - moves the cursor # spaces to the left (non-destructive).

#RIGHT ARROW - moves the cursor # spaces to the right (non-destructive).

a - ignore previous changes and go to the start of the line.

#d - delete the next # characters.

h - hack - chops the line at the cursor and enters the input mode.

i - input mode - inserts the keyboard entries at the cursor location.

l - lists the line with all changes made and moves the cursor to the first position.

#s char - searches for the #th occurrence of char in the line.

x - go to the end of the line text and input.

SHIFT UP ARROW - exit h, i, or x function and re-enter edit.

SHIFT RIGHT ARROW - moves all text in the line to the right margin and ends the line.

SHIFT DOWN ARROW - centers the line text and marks it <do not justify>.

CLEAR - ends the line and marks it <do not justify>.

ENTER OR SHIFT CLEAR - exits the edit mode.

As written, the program will run on a 32K extended basic CC only if the PCLEAR 1 command is entered. This can be avoided by reducing the cleared string space in line 20 to approximately 10,000. If you are using a non-disk system enter - POKE 25,6;NEW in command mode before loading the program. This will free a maximum of 31015 BYTES for use on a 32K machine, and an appropriate amount on a 16K machine. The POKE 25,6 command can not be used on a disk machine because it frees the disk address space and will not allow communication with the disk.

The disk I/O is tailored to a 60 space line to save disk storage space. If, for some reason, it is necessary to change the line length longer than 64 characters then lines 1590 and 1880 must be modified. To store an even number of line per sector 85 or 128 characters per line will optimize disk storage.

REFERENCES:

Hinrichs, Delmer D., BASIC Word Processor - Cheap and Simple Word Processing, 80 Microcomputing, May 1980, PP 50-55.

Kilroy, Mike, Everyman's Mod II Word Processor - An Alternative to Expensive Model II Letter Crunchers, 80 Microcomputing, July 1981, PP 226-233.

Potvin, Howard, Everyman's Debug in 80 Debug, 80 Microcomputing, October 1981, P 22.

```

10 CLS:PRINT TAB(5)"BASIC WORD P
ROCESSOR"
20 CLEAR 13000:NL=360:DIM A$(NL)
,X$(NL),S(25),T(25)
30 B#=CHR$(30):C#=CHR$(191):N#=""
Y":PN#="Y":P1#="Y"
40 S#="" :H#=S#:LA=-1:P=1:FP=1:P
L=26:LL=60:LM=10:U=32:G=10:H=6:V
=27
50 L=LA:IT=0:R=0:A#="":PRINT:INP
UT"COMMAND";A#:IF A#="" THEN 70
60 A=ASC(A#)-64:IF A > 0 THEN ON
A GOTO 80,480,510,760,790,1250,
70,1370,1400,1440,1560,1570,1670
,70,70,1730,70,1840,1860,70,70,1
50,70,2090
70 PRINT"**ENTRY ERROR**":GOTO 5
0
80 CLS:D=0:N#="Y":IF LA < 0 THEN
L = 0:GOTO 120
90 IF NL=LA+1 THEN 200 ELSE IF L
> FL+12 THEN B = L - 12 ELSE B
= FL
100 FOR I = B TO L:X = LEN(A$(I)
):D=D + INT((X+4)/64 -.1)
110 GOSUB 2030:NEXT I:L=L+1
120 C=(L-FL+D)*64:IF C > 447 THE
N PRINT:PRINT:C=446
130 PRINT@C,USING"### ";L;:PRINT
A$(L);:P=LEN(A$(L))+1:C=C+P+3:K=
L+1
140 PRINT@C,C#;:A#=INKEY#:PRINT@
C,S#;:IF A#="" THEN 140
150 GOSUB 280:ON A-7 GOTO 340,39
0,290
160 IF A=13 THEN A#=S#:GOTO 200
ELSE IF A=21 THEN 360 ELSE IF A
= 12 THEN 440
170 IF A = 93 THEN 410 ELSE IF A
= 10 THEN 310 ELSE IF A = 91 TH
EN 460
180 IF A = 92 THEN IF LA < L THE
N LA = L:GOTO 50 ELSE 50

```

```

190 PRINT@C, A#;:A$(L)= A$(L) +A
#:IF P <= LL THEN P = P + 1:C =
C + 1:GOTO 140
200 IF R THEN 50 ELSE IF NL <= K
THEN PRINT:PRINT"FILE FULL":LA
= NL - 1:GOTO 50
210 IF LEN(A$(K)) THEN L = K:GOS
UB 1410
220 IF K > LA THEN LA = K
230 IF A# = S# THEN 270
240 FOR M = LL+1 TO 2 STEP -1:A#
= MID$(A$(L),M,1):IF A#<> S# TH
EN NEXT M:GOTO 270
250 A$(K)= RIGHT$(A$(L),LL-M+1):
A$(L) = LEFT$(A$(L),M-1)
260 PRINT@C-LL+M-1, B#;:L=K:GOTO
120
270 A$(L)=LEFT$(A$(L),LL):L=K:GO
TO 120
280 A = ASC(A#):RETURN
290 IF P > LL THEN 200
300 PRINT@C, CHR$(92);:A$(L) = A
$(L) + CHR$(17):A#=S#:GOTO200
310 IF P > LL THEN 200
320 C = (L-FL + D)*64+4:IF C > 4
52 THEN C = 452
330 GOSUB 1210:P = 1:A# = S#:GOT
D 200
340 IF P = 1 THEN 140
350 C = C - 1:PRINT@C,B#;:P = P
- 1:A$(L) = LEFT$(A$(L),P-1):GOT
D 140
360 IF P = 1 THEN 140
370 A$(L)="" :P = 1:C = (L - FL +
D)*64+4:IF C > 452 THEN C = 452
380 PRINT@C, B#;:GOTO 140
390 IF P > LL - 6 THEN 140
400 A$(L)=A$(L) + STRING$(5,S#):
C = C + 5:P = P + 5:GOTO 140
410 IF P > LL THEN 200
420 C = (L - FL + D)*64+4:IF C >
452 THEN C = 452
430 GOSUB 1230:P = 1:A# = S#:GOT
D 200
440 IF P > LL THEN 200
450 PRINT@C, CHR$(95);:A$(L) = A
$(L) + CHR$(20):A#=S#:GOTO 200
460 C = C -(LEN(A$(L))):A$(L) =
STRING$((LL-LEN(A$(L)))/2,32) +A
$(L) + CHR$(20)
470 PRINT@C, B#; (A$(L)); CHR$(9
3);:A#=S#:GOTO 200

```

```

480 CLS:PRINT"DELETING BLANK LIN
ES";FOR J = LA TO 0 STEP -1
490 IF A$(J) = "" THEN FOR I = J
TO LA:A$(I)=A$(I+1);NEXT I:A$(LA
) = "";LA = LA - 1
500 NEXT J: IF R THEN RETURN ELS
E 1950
510 INPUT "FIRST LINE TO COMPILE
";F:IF F < 0 THEN F = 0
520 INPUT "LAST LINE TO COMPILE"
;Z: IF Z > LA THEN Z = LA
530 IF F >= Z THEN 70 ELSE CLS:P
RINT"COMPILING":FOR L = F TO Z-1
;K = L + 1
540 X=LEN(A$(L)):X$ = "":IF X <
2 THEN 620 ELSE IF X <= LL GOTO
600
550 FOR I = X TO 1 STEP -1 :A$ =
MID$(A$(L),I,1)
560 IF A$ <> S$ THEN X$ = A$+ X$:
NEXT I:GOTO 600 ELSE IF X$ = ""
THEN NEXT I
570 A=ASC(RIGHT$(X$,1)):IF A = 3
3 OR A = 46 OR A = 58 OR A = 63
THEN X$ = X$ + " "
580 A$(L) = LEFT$(A$(L),I-1):IF
LEN(A$(K)) = 0 THEN A$(K) = X$:G
OTO 540
590 A$(K) = X$+ S$ + A$(K):GOTO
540
600 X = LEN(A$(L)):IF X < 2 THEN
620 ELSE FOR I = X TO 2 STEP -1
610 IF RIGHT$(A$(L),1) = S$ THEN
A$(L) = LEFT$(A$(L),I-1):NEXT I
620 NEXT L:FOR L = F TO Z -1:K =
L + 1
630 X=LEN(A$(L)):Y=LEN(A$(K)):X$
 = "":IF X = 0 OR Y = 0 THEN 750
640 A = ASC(RIGHT$(A$(L),1))
650 IF A = 33 OR A = 46 OR A = 5
8 OR A = 63 THEN A$(L)= A$(L) +
" ":X = X + 2
660 FOR I = 1 TO Y:A$= MID$(A$(K
),I,1)
670 IF A$ <> S$ THEN X$ = X$ + A
$:NEXT I ELSE IF X$ = "" THEN NE
XT I
680 IF LL - X < I THEN 710
690 Y = Y - 1:IF Y < 0 THEN Y =
0
700 A$(L) = A$(L) + S$ + X$:A$(K
) = RIGHT$(A$(K),Y):GOTO 630
710 X = LEN(A$(L)):IF X < 2 THEN

```

```

730 ELSE FOR I.= X TO 2 STEP -
1
720 IF RIGHT$(A$(L),1) = S$ THEN
A$(L) = LEFT$(A$(L),I-1):NEXT I
730 IF Y < 2 THEN 750 ELSE FOR I
= Y TO 2 STEP -1
740 IF LEFT$(A$(K),1) = S$ THEN
A$(K) = RIGHT$(A$(K),I-1):NEXT I
750 NEXT L:X = LEN(A$(Z)):GOTO90
0
760 INPUT"FIRST LINE TO DELETE";
F:IF F < 0 THEN F = 0
770 INPUT"LAST LINE TO DELETE";Z
;IF Z > LA THEN Z = LA
780 IF F > Z THEN 70 ELSE FOR I
= F TO Z:A$(I) = "":NEXT I:GOTO
1950
790 INPUT"EDIT LINE";L:IF L < 0
OR L > LA OR A$(L) = "" THEN 70
800 C = 4:P = 1:X$(0) = A$(L):N$
 = "Y"
810 CLS:I = L:GOSUB 2030:N =1:Q$
 = ""
820 GOSUB 910:IF A > 47 AND A <
58 THEN Q$ = Q$ + A$:N = VAL(Q$)
:GOTO 820
830 M = 0:IF A = 8 THEN Y = -1:G
OSUB 940 ELSE IF A = 9 OR A = U
THEN Y = 1:GOSUB 940
840 IF A = 97 THEN A$(L) = X$(0)
:GOTO 800
850 IF LEN(A$(L)) >= LL THEN 870
860 IF A = 93 THEN GOSUB 1230 EL
SE IF A = 91 THEN GOSUB 1210
870 IF A > 98 THEN ON A - 98 GOS
UB 960,1030,2080,2080,2080,1050,
1060
880 IF A = 115 THEN GOSUB 1150 E
LSE IF A = 120 THEN GOSUB 1200 E
LSE IF A = 108 THEN 800
890 IF M = 1 THEN N = 1:Q$ = "":
GOTO 820 ELSE IF R THEN PRINT@16
0,; ELSE 810
900 IF LL < X THEN PRINT "LINE";
L;"HAS";X;"CHARACTERS":GOTO 50
ELSE 1950
910 X$ = MID$(A$(L),P,1)
920 PRINT@C, C$;:A$ = INKEY$:PRI
NT@C, X$;:IF A$ = "" THEN 920
930 GOSUB 280:X=LEN(A$(L)):IF A
= 13 OR A = 92 THEN R = 1:RETURN
ELSE IF A = 91 OR A = 10 OR A =
12 THEN GOSUB 1000 ELSE RETURN

```

```

940 M = 1:FOR I = 1 TO N:P = P +
Y:IF P > X THEN P = X:RETURN
950 IF P < 1 THEN P = 1:RETURN E
LSE C = C + Y:NEXT I:RETURN
960 Q = P:D = C:FOR I = 1 TO N:G
OSUB 910:IF R OR A = 27 THEN P =
Q:C = D:RETURN
970 PRINT@C, A#;:GOSUB 1130:P =
P + 1:GOSUB 1140:A$(L) = L$ + A$
+ R$
980 A = U:C = C + 1:IF P <= X TH
EN NEXT I
990 P = Q:C = D:RETURN
1000 IF A = 91 THEN GOSUB 1210 E
LSE IF A = 10 THEN GOSUB 1070
1010 IF A = 12 THEN A$(L) = A$(L
) + CHR$(20):R = 1
1020 RETURN
1030 IF P + N - 1 > X THEN N = X
- P + 1
1040 GOSUB 1130:Q = P:P = P + N:
GOSUB 1140:A$(L) = L$ + R$:P = Q
:RETURN
1050 GOSUB 1130:A$(L) = L$ + S$:
PRINT@C, B#
1060 GOSUB 910:IF R OR A = 27 TH
EN RETURN
1070 IF A = 10 THEN A$(L) = A$(L
) + CHR$(17):R = 1:RETURN
1080 IF A = 95 THEN 790 ELSE IF
A = 12 THEN A$(L) = A$(L) + CHR$
(20):R = 1:RETURN
1090 PRINT@C, A#;:IF A = 8 THEN
Y = -1:GOSUB 940:GOTO 1060
1100 IF A = 9 THEN Y = 1:GOSUB 9
40:GOTO 1060 ELSE IF P > X THEN
X = P
1110 GOSUB 1130:GOSUB 1140:A$(L)
= L$ + A$ + R$:PRINT@C, B#;A$ +
R$
1120 C = C + 1:P = P + 1:GOTO 106
0
1130 L$ = "":IF P < 2 THEN RETUR
N ELSE L$ = LEFT$(A$(L),P - 1):R
ETURN
1140 R$ = "":IF P > X THEN RETUR
N ELSE R$ = RIGHT$(A$(L),X-P+1):
RETURN
1150 GOSUB 910:Q = P:D = C
1160 FOR I = 1 TO N:F = 0:FOR J
= Q + 1 TO X:D = D + 1
1170 IF MID$(A$(L),J,1) = A$ THE
N F = 1:Q = J:J = X

```

```

1180 NEXT J:NEXT I:IF F THEN P =
Q:C = D
1190 A = U:RETURN
1200 A$(L) = A$(L) + S#:P = X +
1:C = P + 3:GOTO 1060
1210 C = C - (LEN(A$(L))):A$(L)
= STRING$( (LL - LEN(A$(L))) / 2, 32
) + A$(L) + CHR$(20):R = 1
1220 PRINT@C, B#; (A$(L)); CHR$(
93):RETURN
1230 A$(L) = STRING$(LL - LEN(A$
(L)), 32) + A$(L)
1240 PRINT@C, B#; A$(L);:RETURN
1250 CLS:PRINT"LINE LENGTH =";LL
,:INPUT"NEW =";LL
1260 PRINT"LINE SPACES =";S,:INP
UT"NEW =";S
1270 PRINT"LINE #S =";N#;"",:IN
PUT"NEW (Y/N)";N#
1280 PRINT"FIRST LINE =";FL,:INP
UT"NEW =";FL
1290 PRINT"LEFT MARGIN =";LM,:IN
PUT"NEW =";LM
1300 PRINT"PAGE LENGTH =";PL,:IN
PUT"NEW =";PL
1310 PRINT"PAGE #S =";PN#;"",:I
NPUT"NEW (Y/N)";PN#
1320 PRINT"FIRST PAGE =";FP,:INP
UT"NEW =";FP
1330 PRINT"PAGE 1 # = ";P1#;"",
,:INPUT"NEW (Y/N)";P1#
1340 PRINT "CHAR/INCH =";G:INPUT
"NEW (5,10,16.5) =";G
1350 PRINT "LINES/INCH =";H:INPU
T "NEW (6,8) =";H
1360 PRINT"HEADING = ";H#;"",
,:INPUT"NEW =";H#:GOTO 50
1370 CLS:PRINT"LEGAL COMMANDS AR
E":PRINT
1380 PRINT"A ADD","B BLANK","C
COMPILE","D DELETE","E EDIT"
,"F FORMAT","H HELP","I INSER
T","J JUSTIFY","K KILL","L LO
D","M MOVE","P PRINT","R REPL
ACE","S SAVE","V VIDEO","X EX
1390 PRINT:PRINT"PRESS 'SHIFT CL
EAR' TO RETURN FROM A,E,I,R TO
COMMAND MODE":GOTO 50
1400 INPUT"INSERT AT LINE";L:IF
L < 0 OR L > LA THEN 70
1410 IF NL = LA + 1 THEN PRINT"F
ILE FULL":GOTO 50 ELSE IF R THEN
50

```

```

1420 FOR I = LA TO L STEP -1:A$(
I+1) = A$(I):NEXT I
1430 A$(L) = "":LA = LA + 1:L =
L - 1:IF IT THEN RETURN ELSE IT
= 1:GOTO 80
1440 CLS:PRINT"JUSTIFYING":FOR L
= 0 TO LA:X = LEN(A$(L))
1450 IF X < 2 THEN 1550 ELSE FOR
I = X TO 2 STEP -1:A = ASC(RIGH
T$(A$(L),1))
1460 IF A = U THEN A$(L) = LEFT$(
A$(L),I-1):X = X - 1:NEXT I
1470 IF X >= LL OR A = 17 OR A =
20 THEN 1550 ELSE J = 0:K = 1:F
OR I = 1 TO X
1480 IF MID$(A$(L),I,1) <> S$ TH
EN K = 0 ELSE IF K = 0 THEN K =
1:S(J) = I:J = J + 1
1490 NEXT I:IF J = 0 THEN 1550
1500 K = RND(J) - 1:IF INT(J/2)
=J/2 OR J = 1 THEN N = 1 ELSE N
= 2
1510 FOR I = 1 TO LL -X:T(K) = T
(K) + 1:K = K + N:IF K > J - 1 T
HEN K = K - J
1520 NEXT I:FOR I = J - 1 TO 0 S
TEP -1:A$ = STRING$(T(I),S$):T(I
) = 0
1530 A$(L) = LEFT$(A$(L),S(I))+A
$+RIGHT$(A$(L),LEN(A$(L))-S(I))
1540 NEXT I
1550 NEXT L:GOTO1950
1560 CLS:INPUT"REALLY KILL (Y/N)
":A$:IF A$ = "Y" THEN 10
1570 INPUT "WHAT LETTER DO YOU W
ANT?":Z$
1580 CLS:PRINT "LOADING FROM DIS
K:":Z$
1590 OPEN "D", #1, Z$,64:REM IF
LINE LENGTH > 64 THIS LINE SHOUL
D BE CHANGED TO 85 OR 128 CHARAC
TERS AS REQUIRED
1600 GET #1, 1
1610 INPUT #1, LA,LL,S,N$,FL,LM,
PL,PN$,FP,P1$,H$
1620 FOR I = 0 TO LA
1630 GET #1
1640 INPUT #1, A$(I)
1650 NEXT I
1660 CLOSE #1:GOTO50
1670 INPUT"FIRST LINE TO MOVE":F
:IF F < 0 THEN F = 0
1680 INPUT"LAST LINE TO MOVE":Z:
IF Z > LA THEN Z = LA

```

```

1690 IF F > Z THEN 70 ELSE INPUT
"FIRST NEW LINE":N:FOR I = F TO
Z
1700 IF LEN(A$(N)) THEN PRINT"LI
NE":N;"NOT EMPTY":GOTO 50
1710 A$(N) = A$(I):A$(I) = "":N
= N + 1:IF N > LA THEN LA = N
1720 NEXT I:GOTO 1950
1730 X = FP:M = FL:GOTO 1740
1740 IF G = 5 THEN W = 14 ELSE I
F G = 10 THEN W = 18 ELSE IF G =
16.5 THEN W = 15
1750 IF H = 6 THEN T = 50 ELSE I
F H = 8 THEN T = 48 ELSE IF H =
10 THEN T = 49
1760 INPUT"WANT TO CHECK FORMATS
(Y/N)":Z$:IF Z$ = "Y" THEN 1250
ELSE R = 1:GOSUB 480:CLS:PRINT"
PRINTING"
1770 IF PN$ = "Y" THEN B1 = 63 E
LSE B1 = 66:PRINT#-2, CHR$(W), C
HR$(V); CHR$(T);: IF PN$ <> "Y"
OR (P1$ = "N" AND X = 1) THEN 17
0
1780 PRINT#-2, TAB(LM);H$:TAB(LL
+ LM - 7)"Page":PRINT#-2,USING
"###":X:PRINT#-2,CHR$(10)
1790 FOR P = M TO M + PL -1:IF P
> LA THEN 1830
1800 M = M + 1:IF S THEN PRINT#-
2, STRING$(S,138)
1810 PRINT#-2, TAB(LM);:IF N$ =
"Y" THEN PRINT#-2, USING "### ";P
;
1820 PRINT#-2, A$(P):IF ASC(RIGH
T$(A$(P),1)) <> 17 THEN NEXT P
1830 IF S THEN S1 = B1 -((S+1)*P
L) ELSE S1 = B1 - PL:FOR I = 1 T
O S1:PRINT#-2, CHR$(28):NEXT I :
IF P > LA THEN 50 ELSE X = X + 1
GOTO 1770
1840 INPUT"REPLACE LINE":L:IF L
< 0 OR L > LA THEN 70
1850 R = 1:A$(L) = "":L = L - 1:
GOTO 80
1860 INPUT "PLEASE NAME THIS LET
TER":Z$
1870 CLS:PRINT "SAVING PRESENT L
ETTER ON DISK AS:":Z$
1880 OPEN "D", #1, Z$, 64:REM IF
LINE LENGTH > 64 LINE MUST BE C
HANGED TO 85 OR 128 CHARACTERS A
S APPROPRIATE

```





```

1890 WRITE#1, LA,LL,S,N$,FL,LM,P
L,PN$,FP,P1$,H$
1900 PUT #1, 1
1910 FOR L = 0 TO LA
1920 WRITE #1, A$(L)
1930 PUT #1, L+2:NEXT L
1940 CLOSE #1:GOTO 50
1950 CLS:L = LA:X = FP -1:FOR M
= FL TO LA STEP PL:X = X + 1
1960 IF P1$ = "N" AND X = 1 THEN
1980
1970 IF PN$ = "Y" THEN PRINT H$;
TAB(LL-7) "Page";:PRINTUSING"##
#";X:PRINT
1980 FOR I = M TO M + PL -1:IF I
> LA THEN 2010
1990 IF S THEN PRINT STRING$(S-1
,10)
2000 GOSUB 2030
2010 NEXT I:A$ = "":IF I <= LA T
HEN INPUT"PRESS ENTER";A$:IF A$
<> "" THEN M = LA
2020 NEXT M:L = LA:GOTO 50
2030 Y = LEN(A$(I)):IF Y THEN A
= ASC(RIGHT$(A$(I),1)) ELSE A =
0
2040 IF N$ = "Y" THEN PRINT USIN
B "### ";I;
2050 PRINT A$(I);:IF A = 17 THEN
PRINT CHR$(92);
2060 IF A = 20 THEN PRINT CHR$(9
5); ELSE IF A = 32 THEN PRINT CH
R$(93);
2070 IF N$ <> "Y" OR Y <> 60 THE
N PRINT
2080 RETURN
2090 CLS:INPUT"REALLY EXIT (Y/N)
";A$:IF A$ <> "Y" THEN 50
2100 CLS:PRINT"THE END":END
    
```

COCOBOG: 8804 debugging monitor for TRS 80 Color Computer. 11 commands and 4 control characters to change memory registers. VIX, SAM (hex), MATH. Reference book card \$19.95 + 2.00 shipping.

ALLEN GELDER SOFTWARE
Box 11721 Main Post Office
San Francisco, CA 94101
TRS-80™ Radio Shack/Tandy Corp.

VEICO INTERNATIONAL

3870 WEST 143RD STREET, CLEVELAND, OHIO 44111, U. S. A.

COMPUTER GAMES == GOOD NEWS AGAIN FOR == RADIO SHACK TRS-80™

COLOR COMPUTER OWNERS == CHOOSE ANY ONE GAME FROM THE LIST BELOW FOR \$7.00 OR ANY THREE FOR \$15.00 == WHAT A BARGAIN !!!!

ALL GAMES REQUIRE ONLY A MINIMUM OF "4K" TO RUN. CASSETTE-PLAYER AND JOYSTICKS ARE REQUIRED. THIS IS AN INTRODUCTORY OFFER FROM "PRO-GAMES" DON'T MISS THIS GREAT SALE == SAVE \$\$\$\$ == SEND ORDER(S) TODAY

1) SLOT-MACHINE	7) BREAK-THE-CODE
2) REFLEX	8) BIORHYTHM
3) BLACK-JACK	9) POKER-DICE (YAMTZE LIKE)
4) CRAPS	10) LUNA-LANDER
5) ROULETTE	11) TIC-TAC-DRAGON
6) HANG-MAN	12) RUSSIAN-ROULETTE

NOTE: WE STILL HAVE "BRICKAWAY" (GREAT CHALLENGE) FOR \$7.00 EACH, AND "GAUNTLET" (SPACE GAME) FOR \$10.00 EACH. SUPPLY IS LIMITED !!!!

L O T S O F F U N ! ! !

SEND ORDER(S) TO: VEICO INTERNATIONAL (DEPT.03.)
3870 WEST 143RD STREET
CLEVELAND, OHIO 44111

(TRS-80 IS A REGISTERED "TRADE MARK" OF TANDY CORP.)

BAKER'S DOZEN SALE

UNTIL JANUARY 31, 1982
TO INTRODUCE OUR

CHEAP BLANK COMPUTER TAPES

	12 + 1 FREE	24 + 2 FREE
C-10	75¢ ea.	65¢ ea.
C-20	85¢ ea.	75¢ ea.

- Premium Tape
- 100% Error Free
- Short Rewind
- Mu Metal Shield
- Superior Five Screw Construction

Specifically Designed for the Higher Baud Rates of the Color and Mod III Computers

**the little computer store
of cincinnati**
7785 ELBROOK
CINCINNATI, OHIO 45237
(513) 631-4555

BREAK DISABLE FOR THE COLOR COMPUTER
by C J Roslund

Did you know that after every instruction in a BASIC program the BASIC interpreter in the COLOR COMPUTER polls the keyboard and checks for "BREAK" or "SHIFT-@" being pressed. This can add considerably to the execution speed for a BASIC program. Speeding up your BASIC programs is reason enough to investigate disabling "BREAK". Other reasons include use in programs that children may be running (educational programs) or important programs that you wish to protect from accidental interruption. This article describes an easy method to disable "BREAK" and "SHIFT-@" by adding 5 lines to the beginning of your BASIC program. By the way, this method is only applicable to EXTENDED COLOR BASIC computers although a similar procedure could be implemented on a standard COLOR BASIC computer.

There is a block of code in the EXTENDED BASIC ROM (\$82B9-\$831E) that controls execution of a BASIC program. This loop is returned to after each BASIC instruction in the program is executed. It pulls successive bytes out of the BASIC program area and decides what to do based on each byte. One function of this loop is to check if the "TRON" (trace feature) is activated and handle that function if necessary. It also calls the routine that polls the keyboard for "BREAK" or "SHIFT-@". This block of code is entered after executing a "RUN" command via a "JMP" instruction located at addresses \$019A, \$019B, and \$019C. These addresses contain what I call a "TRAP VECTOR". It contains an "RTS" instruction in a standard COLOR BASIC computer. It contains the instruction "JMP \$82B9" in EXTENDED COLOR BASIC computers. This "TRAP" gives us the means to take over control of the execution of a BASIC program.

Disabling "BREAK" and "SHIFT-@" involves 4 steps:

First - Copy the block of code that controls execution of a BASIC program (\$82B9-\$831E) to an area in RAM where you can modify it.

Second - Patch the copied block so that it can run where it's been relocated and also delete the call to the check for "BREAK" and "SHIFT-@" routine.

Third - Modify the "TRAP VECTOR" at \$019A, \$019B and \$019C to point to the relocate block of code.

Fourth - Execute a "RUN" command to cause the BASIC interpreter to jump to the relocated block of code via the modified "TRAP VECTOR".

One restriction to the use of this "BREAK Disable" is that it does not disable "BREAK"

during an "INPUT" command. All input to the program must be via "INKEY\$" or an "INKEY\$" loop rather than "INPUT".

The program to disable "BREAK" and "SHIFT-@" is listed below. It should appear as the first 5 lines in your program. The "CLEAR" statement in line 1 may be modified to reserve the desired string storage space, but the memory reserved by this "CLEAR" statement should not be used since this is where the block of code (see first step above) is copied into. Line 10 of the program is included just as a test. Your program should begin with line 10.

16K VERSION

```
1 IF PEEK(&H3EB9) <> &H32 THEN CLEAR 200,
&H3EB0: FOR I=&H82B9 TO &H831E: POKE
I-&H4400, PEEK(I): NEXT ELSE 5
2 FOR I=0 TO 2: POKE &H3EBD+I,18: NEXT:
I=&H3F1E
3 POKE I,&H26: POKE I+1,3: POKE I+2,&H7E:
POKE I+3,&H83: POKE I+4,&H22: POKE I+5,&H7E
4 POKE I+6,&HA4 :POKE I+7,&H4C
5 POKE &H19B,&H3E: RUN 10
10 PRINT"BREAK DISABLED": GOTO 10
```

32K VERSION

```
1 IF PEEK(&H7EB9) <> &H32 THEN CLEAR 200,
&H7EB0: FOR I=&H82B9 TO &H831E: POKE
I-&H400,PEEK(I): NEXT ELSE 5
2 FOR I=0 TO 2: POKE &H7EBD+I,18: NEXT:
I=&H7F1E
3 POKE I,&H26:POKE I+1,3:POKE I+2,&H7E:POKE
I+3,&H83:POKE I+4,&H22:POKE I+5,&H7E
4 POKE I+6,&HA4:POKE I+7,&H4C
5 POKE &H19B,&H7E:RUN
10 PRINT"BREAK DISABLED":GOTO10
```

Having "BREAK" disabled increases execution speed of a simple "FOR NEXT" loop by about 40%. More complicated commands will realize a lesser increase in execution speed.

Once this program is run, the only way to interrupt it is to push the "RESET" button.

Charles J. Roslund
150 Hochberg Road
Pittsburg, PA 15235



Jan Mix Software

3424 College N.E.
Grand Rapids, MI 49505

MOON LANDER

2 Programs - A real treat!

Train on "Moon Lander" then move up to "Lander II". An outstanding flying experience. Visit the moon on your CC. Ext. Basic.

\$15.95

ML RABBIT

No serious programmer can afford to pass this up!
Make copies of any machine or basic program effortlessly. Even copies programs that automatically execute. Completely automatic.
(Caution-Intended to make back-up tapes only).

**only
\$14.95**



ABC 123 ABC 123

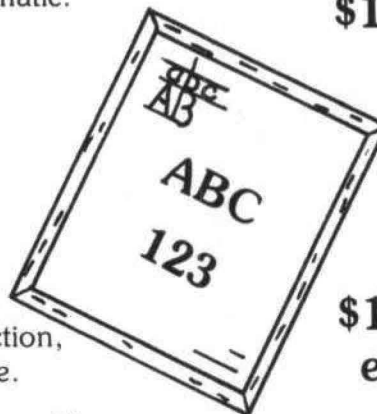
EDUCATIONAL PROGRAMS

(16K Ext. Basic)

★ **Math Drill** - Designed for teaching addition, subtraction, multiplication and division to the student of any age.

Recommended by teachers as an outstanding learning aid.

★ **Spelling Test** - Hear your CC talk to you! An outstanding learning tool. Students love to learn with this program. For any age.



**\$19.95
each**

**\$19.95
each**

CCN-SAMPLER SERIES

\$7.95 ea.

CONNECT FOUR

Challenging game played either with two players or against the computer. 4K

\$12.95

3 GAME PAKS - 4K

Battleship, Othello, Life or Mastermind, Pig, Spidersweb

**\$19.95
ea. pak**

Add \$1.00 Postage & Handling.

★ **LOOKING FOR NEW SOFTWARE** ★
— TOP ROYALTIES PAID —

Money Order's sent same day. Personal checks must clear your bank before shipment.
Michigan residents add 4% Sales Tax.

We review products from every source and carry ALL of THE BEST



We offer products from
Mark Data • Micro Works
Radio Shack • Book Publishers
Epson • NEC • Centronics • Moore

SOFTWARE: Color Invaders • Color Pac Attack
Monitor • Adventure Games • Text Editors
Assemblers • PASCAL • Magikube
Finance Programs • Color Data Organizer
Graphic Games • Disassembler

HARDWARE: 32K RAM Expansion Board • 16K RAM Set
Cables • Interfaces • Power Pack ROM Cartridge • Printers

ACCESSORIES: Books • Cassettes • Supplies • Service Manuals

Look to **COMPUTERWARE** for **DISK SOFTWARE**

★ ★ ★ **NEW PRODUCTS** ★ ★ ★

16 PLUS BOARD — just plug in to expand from 16K to 32K
PAC ATTACK — graphics action game — **PAC ATTACK**
32K versions of **Editor, Assembler, Monitor, PASCAL, BERSERK** game —
Micro Text (communications) — **STAR BLASTER**

Shipping from stock

**CALL
OR
WRITE
FOR
COMPLETE
INFORMATION**



Dept. C • Box 668
6809 Specialists Encinitas, CA 92024 • (714) 436-3512

Computerware is a trademark of Computerware.

Making Education More Colorful
by David Bodnar

For several years I have culminated the study of long division by challenging my students to solve a really difficult division problem. Something like this:

1234567890098760531124570 divided by 123432

Such an activity accomplishes several objectives. First it impresses upon the students the need for care and neatness in working division problems. If you have a tendency to be sloppy you will run into trouble in a very short time. Since I permit the use of calculators in solving the problem the children come to realize that even though the problem is too large for the calculator to handle at one time it may be entered and worked section by section. This also requires a good knowledge of the division algorithm (the step method by which a problem is solved). I have made up a new division problem each year and solved it myself several times to be sure of the correct answer. I would have liked to have been able to make up 30 different problems each year so that there wouldn't be any "sharing" of answers, but it required an unrealistic expenditure of time. That all changed this year due to the presence of several COLOR COMPUTERS in my classroom.

With the aid of the computer I was able to generate and solve 30 different problems, one for each student. This not only took care of the problem of "sharing" answers, but I found the children to be even more motivated than usual. This I attribute to the personalized nature of the assignment.

When I set out to use the computer I realized that the resident BASIC language was designed to shift into scientific notation when a number exceeded 9 digits. I got around this by treating the dividend and the quotient (answer) of the problem as strings. The divisor is less than 9 digits so I was able to use some of the computers built in math functions in such the same way as the children used the calculators.

The program that I used to "TEACH" the computer how to divide is based on the division algorithm that we all learned as children. It is very similar to a computer program as you can see from the step by step procedure outlined below:

DIVISION ALGORITHM

1. Is the divisor less than or equal to the first digit (from the left) of the dividend?

2. If it is go to step 3—if not keep adding digits, one at a time, until you have a section of the dividend that the divisor will divide into evenly.

3. How many times does the divisor go into the section of the dividend that we are working with? (this is the trial divisor)

4. Multiply the trial divisor times the dividend.

5. Subtract this product from the section of the dividend that we chose in step 2.

6. Is the remainder (answer from step 5) less than the divisor?

7. If it is go to step 8—if not increase the trial divisor by one and go to step 4.

8. Have we used all of the digits of the dividend?

9. If we have, we are done and the remainder at the bottom of the problem becomes part of the answer—if not add the next digit of the dividend to the remainder and go to step 3.

Before I could translate the division algorithm into BASIC so that the computer could answer the problem I had to have the computer come up with a problem. The random number function also shifts to scientific notation when the 9 digit threshold is passed so I again had to resort to strings. Lines 500 to 590 show the subroutine that is used to generate the problems.

The problem that the computer generates is displayed on the screen at line 80. It uses the low resolution graphics characters to make the division bracket. Lines 90 through 210 solve the problem in such the same way as I described above in the division algorithm. Line 110 is roughly equivalent to steps 1 & 2; line 130 to step 3; line 170 to steps 4 & 5; line 180 to step 7; line 190 to step 8; and line 200 to step 9. Step 6 is omitted because we assume that the computer does not make division errors!!

The program as it appears here is designed for classroom use. It asks you to type in each child's name and when it displays a problem there is a name printed above it. It also allows you to save a set of problems to tape and reenter them into the program by starting the program by typing RUN 3000 rather than just typing RUN. If you wanted to display just the problem, so that a child might copy it onto a piece of paper for solving, simply edit lines 160 and 220 so that a REM appears at the beginning of the line. This allows the line to remain but the computer will ignore it and print only the problem.

Making Education More Colorful

If you are fortunate enough to have a printer available you can have a separate personalized problem sheet printed for each child. You could also have a master key printed by having data in the ST\$,PR\$ and AN\$ arrays output to the printer.

While it is possible to have the computer display each intermediate step while it is computing the answer the Color Computer's small screen size causes most of it to scroll off of the screen very rapidly and I have found it to be of little value. A possible solution would be to send the output to your printer.

```

10 CLEAR 1000
20 CLS:
   PRINT:
   PRINT:
   PRINT:
   INPUT " HOW MANY STUDENTS";NS
30 DIM ST$(NS),PR$(NS),AN$(NS)
40 FOR X=1 TO NS:
   CLS:
   PRINT:
   PRINT:
   PRINT " NAME OF STUDENT"X:
   LINE INPUT ST$(X):
   NEXT X
50 FOR YY= 1 TO NS ' DO THE ENTIRE
   PROGRAM ONCE FOR EACH STUDENT
60 IF FLAG <> 1 THEN GOSUB 500 ELSE
   GOSUB 1000 ' GENERATE PROBLEM OR
   GET IT FROM ARRAY LOADED FROM T
   APE
70 CLS:
   PRINT@0,ST$(YY); ' PRINT THE STU
   DENTS NAME
80 PRINT@32*4+LEN(DV$),STRING$(LEN(
   DD$)+1,CHR$(140)):
   PRINT@32*5,DV$;CHR$(128);DD$ ' P
   RINT THE PROBLEM AND THE DIVISIO
   N BRACKET
90 FOR X=1 TO LEN(DD$) ' GO THROUGH
   THE DIVIDEND FROM LEFT TO RIGHT
   TO FIND THE FIRST SECTION THAT
   IS >= THE DIVISOR
100 DS=VAL(LEFT$(DD$,X)) ' FIND VAL
   UE OF FIRST X DIGITS OF DIVIDEN
   D
110 IF DS<DV THEN NEXT X ' DS=VALUE
   OF DIVIDEND SECTION-- IF DS<DV
   THEN GET THE NEXT DIGIT OF DIV
   IDEND
120 XX=X ' SAVE VALUE OF X WHICH DE
   TERMINES PLACEMENT OF QUOTIENT
   ABOVE DIVIDEND
130 Q=INT(DS/DV) ' FIND DIGIT OF AN
   SWER
140 QQ$=RIGHT$(STR$(Q),1) ' REMOVES
   SPACES FROM STRING
150 Q$=Q$+QQ$ ' BUILD THE QUOTIENT
   DIGIT BY DIGIT
160 PRINT@32*3+LEN(DV$)+XX,Q$; ' PR
   INT QUOTIENT DIGIT BY DIGIT
170 R=DS-Q*DV ' FIND REMAINDER
180 X=X+1 ' GET THE NEXT DIGIT OF T
   HE DIVIDEND
190 IF X>LEN(DD$) THEN 220 ' IF WE
   HAVE USED ALL OF THE DIGITS OF
   THE DIVIDEND WE ARE DONE
200 DS=R*10+VAL(MID$(DD$,X,1)) ' AD
   D THE NEXT DIGIT OF THE DIVIDEN
   D TO THE END OF THE REMAINDER
210 GOTO130 ' GO BACK FOR NEXT DIGI
   T OF DIVIDEND
220 PRINT@32*10,"":
   PRINT:
   PRINTQ$,"*R*";R ' PRINT ANSWER
   & REMAINDER ON SCREEN

230 PR$(YY)=DV$+" "+DD$ ' SAVE PROB
   LEM
240 AN$(YY)=Q$+" *R*"+STR$(R) ' SAV
   E ANSWER
250 DD$="":
   DV$="":
   Q$="" ' CLEARS STRINGS
260 NEXT YY
270 GOTO 2000
280 '
290 ' *****
   *****
300 '
500 ' GENERATES PROBLEM
510 DV=RND(999999):
   IFDV<100000THEN510 ' GENERATES
   6 DIGIT DIVISOR
520 FOR X=1TO25:
   ' GENERATES DIVIDEND OF 25 DIG
   ITS
530 N$=STR$(RND(10)-1) ' GENERATE E
   ACH DIGIT OF DIVIDEND

```

Making Education More Colorful

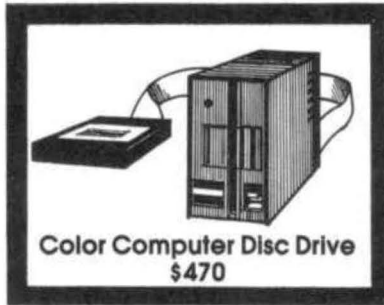
```

540 N$=RIGHT$(N$,1) ' REMOVES LEADI
    NG SPACE FROM EACH DIGIT
550 IF X=1 AND N$="0" THEN530 ' IF
    FIRST DIGIT = ZERO THEN REDO
560 DD$=DD$+N$ ' BUILD STRING DIGIT
    BY DIGIT
570 NEXTX
580 DV$=STR$(DV):
    DV$=RIGHT$(DV$,LEN(DV$)-1) ' MA
    KE DIVIDEND A STRING AND REMOVE
    LEADING ZERO
590 RETURN
600 '
610 '*****
    *****
620 '
1000 ' GETS PROBLEM FROM ARRAY THAT
    WAS LOADED FROM TAPE
1010 PO=INSTR(PR$(YY)," ") ' FIND P
    OSITION OF SPACE IN PROBLEM ST
    RING
1020 DV$=LEFT$(PR$(YY),PO-1) ' THE
    SECTION LEFT OF THE SPACE IS T
    HE DIVISOR
1030 DD$=RIGHT$(PR$(YY),LEN(PR$(YY)
    )-PO) ' THE SECTION RIGHT OF T
    HE SPACE IS THE DIVIDEND
1040 DV=VAL(DV$)
1050 RETURN
1060 '
1070 '*****
    *****
1080 '
2000 ' SAVE PROBLEMS TO TAPE
2010 CLS:
    PRINT@32*3," SAVE TO TAPE (Y/
    N)":
    INPUT EE$:
    IF EE$="N" THEN END
2020 CLS:
    PRINT@32*3," NAME OF TAPE FI
    LE":
    LINEINPUT NT$
2030 IF NT$="" THEN NT$="UNNAMED"
2040 CLS:
    PRINT@32*3," SET TAPE TO RECOR
    D":
    PRINT:
    PRINT" <ENTER> WHEN READY":
    LINE INPUT EE$
2050 CLS:
    PRINT@32*3," SAVING <"NT$">"
2060 OPEN"O",-1,NT$
2070 PRINT#-1,NT$ ' NAME OF FILE
2080 PRINT#-1,NS ' NUMBER OF STUDEN
    TS
2090 FOR X=1 TO NS:
    PRINT#-1,ST$(X);PR$(X);AN$(X):
    NEXT X ' NAME,PROBLEM AND ANSW
    ER
2100 CLOSE -1
2110 END
2120 '
2130 '*****
    *****
2140 '
3000 ' READ TAPE FILE ROUTINE
3010 CLS:
    PRINT@32*3," WHAT FILE NAME TO
    LOAD":
    LINE INPUT NT$
3020 IF NT$="" THEN RF$="UNNAMED" E
    LSE RF$=NT$
3030 CLS:
    PRINT@32*3," SEARCHING FOR <"R
    F$">....."
3040 OPEN"I",-1,NT$
3050 LINE INPUT#-1,NT$
3060 PRINT@32*3," FOUND <"NT$">...
    ..LOADING"
3070 INPUT#-1,NS ' NUMBER OF STUDEN
    TS
3080 FOR X=1 TO NS:
    LINE INPUT #-1,ST$(X):
    LINE INPUT #-1,PR$(X):
    LINE INPUT #-1,AN$(X):
    NEXT X ' NAME,PROBLEM AND ANSW
    ER
3090 CLOSE -1
3100 FLAG = 1
3110 GOTO 50
30000 CSAVE"CCN PRGM",A

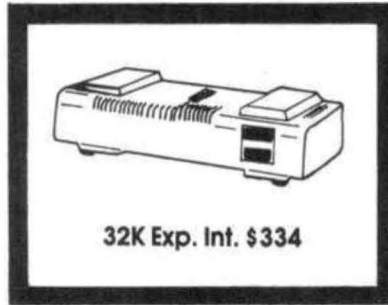
```

From Computer Plus to YOU...

PLUS after PLUS after PLUS



Color Computer Disc Drive
\$470



32K Exp. Int. \$334



Line Printer VII \$315



Color Computer 4K \$310
w/16K Ext. Basic \$459
w/32K Ext. Basic \$525



Model 16 128K
1 Drive \$4299



Model III 16K \$839
Model III 48K
2 Disc & RS232C \$2059

BUY DIRECT Here are just a few of our fine offers...
call TOLL FREE for full information.

COMPUTERS

Model II 64K	\$3300
Model III 4K LEV I	599
MODEL III 16K	839
MODEL III 32K	945.50
*MODEL III 32K	881.50
MODEL III 48K	1052
*MODEL III 48K	924
Model III 48K	
2 Disc & RS232 c	2059
Color Computer 4K	310
Color Computer 16K	416.50
*Color Computer 16K	352.50
Color Computer 16K	
w/extended basic	459
Color Computer 32K	
w/extended basic	525
Color Computer Drive 0	470
Pocket Computer	189
VIDEOTEX	310

*Computer Plus New Equipment,
with NEC RAM installed.
180 Day Computer Plus Warranty.

**We have the lowest possible
Fully Warranted Prices AND
a full complement of Radio Shack
Software.**

DEALER INQUIRIES ARE INVITED

Prices subject to change without notice.
Not responsible for typographical errors.
TRS-80 is a registered trademark of Tandy Corp.

PERIPHERALS

Expansion Interface OK	\$249
Expansion Interface 16K	355.50
*Expansion Interface 16K	291.50
Expansion Interface 32K	462
*Expansion Interface 32K	334
16K RAM N.E.C. 200 N.S. chips	25

MODEMS

Lynx Direct Connect MII/MIII	235
Auto Ans./Dial	
Telephone Interface II	169
R.S. Modem I D.C.	130
R.S. Modem II D.C.	210

PRINTERS

Daisy Wheel II	1695
Epson MX80	479
Epson MX80 FT	589
Epson MX100	759
Line Printer VII	315
Line Printer VIII	620
Line Printer V	1610

Microline 80	345
Microline 82A	479
Microline 83A	745
Microline 84 Parrabel	1090
Pocket Computer Printer	130

DISK DRIVES

R.S. Model III 1ST-Drive	712
Tandon 40 Track MI	329
R.S. 1 Drive Exp MII	999
R.S. 2 Drive Exp MII	1518
R.S. 3 Drive Exp MII	2040

SOFTWARE

R.S. Software 10% off list	
Newdos 80 MIII	149
ST80III	149

ETC.

Verbatim 5" Double Density	32
Verbatim 8" Data Life	49.95
Ctr-80A recorder	52
C. C Joysticks	22

call TOLL FREE 1-800-343-8124

computer plus

245A Great Road
Littleton, MA 01460
617-486-3193

Write for your
free catalog



As you know we missed the interview with Frank Hogg Labs last issue, due to the erasure of the telephone conversation, so Frank and I got on the phone again and made another attempt at bugging his phone.

Why did you adapt Flex to the CC and what is Flex?

Flex is the standard 68XX disk operating system, it's been available for about five (5) years and is, in fact, older than the 6809. There is a lot of software that operates under Flex, in fact by implementing Flex on the CC hundreds of software packages are immediately available, for example there are over 40 packages from us alone. This version of Flex is unique in that it has the ability to turn the BASIC ROMs off, essentially this is just like removing them! The 64K modification does some very slick things, like turning off the ROMs, but also causes some problems. The biggest problem is that a lot of software written for the CC uses code resident in the BASIC ROMs, when you turn off the Rom you can't call it so you must write all of the code from scratch. The advantage is that you can change anything you wish. Some of the things that we changed are the ability to move the text screen anywhere in memory, our keyboard routine allows the keyboard to generate all of the ASCII codes and the ability to read or write any combination of 5" disks, single or double density, any number of tracks, single or double sided or any head step rate. For example, you could keep the Radio Shack 35 track drive as 0 and have a double sided, double density, 80 track drive as drive 1. The optional paging screen will allow other neat features like real lower case which can be moved most anywhere is RAM.

How does it work?

First, you have to "Boot" the system. With this version of FLEX you simply put your FLEX disk in drive 0 the directory will only show one file, FLEX. Type RUN"FLEX". In a few moments you will be in the Flex operating system at the +++ prompt. What happens is the program loads from the disk a short routine which changes the SAM's map type bit to all RAM, "unfolds" the 64K RAMs and loads Flex from the disk starting at \$C000. As you know this is the area usually occupied by ROM Paks. The routine then jumps to the starting address of Flex. You then have a new directory not available from CCDOS. An interesting feature is that you don't have to give up BASIC, you can move it into RAM and jump to

it. This opens up the possibility of correcting some of the "bugs" in it. Another feature is that machine language programs that wouldn't run from RS disks will run from Flex. If the machine language program uses ROM calls just move BASIC to RAM and not RSDOS. The biggest advantage is that the CC's limitations are gone. The only limit now is the imagination of the programmer.

Your ads also mention OS9. What is it and why did you modify it for the CC?

OS9 was written specifically for the 6809 and therefore is as "state of the art" as the 6809 is. OS9 was originally written for BASIC09, which is really a combination of BASIC and Pascal. OS9 is like UNIX and version 1.2, which is the one modified for the CC, includes "pipes". Pipes are a difficult thing to explain, but essentially it allows more pathways for information to follow. For example, in OS9 you could have a directory in a directory in a directory. Confused? Suppose you had a disk that was for all your correspondence, you could have a "CCNDIR" which contained the directory of all letters you've written to CCN, on the same disk you could have a "MOMDIR" which contained all of the letters you've written to your mom, both of these are in the same directory and are like sub-directories in the directory. Pipes can also be used to reroute I/O, for example keyboard entries usually go to the screen, you can change it to: from the disk to the screen or disk to printer or whatever. OS9 is multi-user, multi-programming, multi-tasking. In comparing it to Flex, Flex has more software by virtue of it's being older but OS9 has more power. In fact, a CC running OS9 has 3.5K more RAM than an SS-50 computer. In fact it can do as much as an SS-50 machines. In about 3 months we'll have a board available to allow you to have all of this without opening the CC case. It will include a ROM monitor program and some ACIAs to allow you to do multi-user on the CC. Actually you could do multi-user as it sits but the RS-232 on the CC isn't really RS-232 and would be far too slow.

Tell me about Forth.

Forth is an interesting language in that if there is something about the language that you don't like, you change it. As an example, if you don't like ? as the abbreviation for PRINT you can change it to CCN or * or anything else you like. If it's missing a command that you need you add it. All without ever touching your assembler. Some versions of Forth were written in Forth.

PROCESSING NUMERICAL DATA
by Robert P. Bussell

The Color Computer is much more powerful than many people realize. As a software engineer I find that I use my computer at home to solve many work related problems. While working with numeric data, I have found some shortcuts not documented by Radio Shack and some problems and quirks in Extended BASIC which I would like to share with you.

Octal and hexadecimal numbers may be converted to decimal with this short routine:

```
10 DIM BI$(15)
20 DATA 0000, 0001, 0010, 0011, 0100, 0101,
0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101,
1110, 1111
30 FOR I=0 TO 15
40 READ BI$(I)
50 NEXT I
60 CLS
70 INPUT "OCTAL NO.:";O$
80 DEC = VAL("&O" + O$)
90 HE$ = HEX$(DEC)
100 PRINT "THE NUMBER IS:";PRINT
O$;"OCT";DEC;"DEC ";HE$;"HEX
110 FOR I = 1 TO LEN(HE$)
120 PRINT BI$(VAL("&H" + MID$(HE$,I,1)));
130 NEXT I
140 PRINT "BI$ "; GOTO 70
```

Similarly, hexadecimal numbers can be converted to decimal by using the statement DEC = VAL("&H" + HE\$). The program above works for all positive integers between 0 and 65535. There does not appear to be a function equivalent to HEX\$ for converting decimal numbers to octal. If you enter an octal number which contains an 8 you will get some interesting results. The numbers below illustrate this:

Octal	Decimal	Hexadecimal	Binary	Correct
8	8	8	1000	10
18	16	10	10000	20
88	72	48	10001000	110
181	129	81	10000001	201

109 SN ERROR IN 80

It appears that whenever Extended BASIC encounters the number 8 in the octal conversion routine it assumes it to be a ten octal. If we apply this assumption to the numbers in the table above and substitute a 10 for the 8 and evaluate the number we see:

```
8=10 18=10 88=10 181=1
=10 100 100
-- --- 100
20 110 ---
201
```

A more annoying problem is the entry of an octal number containing the digit 9. Instead of returning a zero entry the VAL function invokes the SN ERROR routine and halts program execution. I have tried entering the remainder of the character set and the VAL function returns a zero as you would expect for a non-numeric entry. An easy way to protect your program from an abnormal halt would be to insert the statement:

```
75 X=INSTR(O$,"9"); IF X<>0 THEN
DEC=0; GOTO 90
```

This line is inserted just before the VAL function and if a 9 is entered it sets the number to be converted to 0. This test could also be a branch to an error processing routine.

Accuracy of numbers used in equations is another concern which must be addressed when exact numerical values are needed. For example, if you run the short program listed below you will find that the number of significant digits displayed to the right of the decimal place will change as the value of B is incremented. This effect is caused because all numerical values are manipulated internally as floating point numbers. Rounding routines in this conversion to and from the internal format for certain numbers cause the resultant sum to be slightly different than the value expected. In accounting application, exact numbers are usually required and this slight rounding when compounded, could cause your final result to be incorrect. The second example illustrates a method to test for rounding errors and a compensation factor to correct the value.

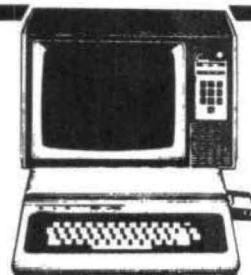
```
10 A=.01
20 B=B+A
30 PRINT B
40 GOTO 20
*** listing below goes beside listing
above!!!!!!!!!!!!
```

```
10 A=0.01
20 B=B+A
25 IF FIX(100*B)/100 < B THEN
B=FIX(100*B)/100+0.01
30 PRINT B
40 GOTO 20
```

CO-RESIDENT EDITOR/ASSEMBLER (CORES9)

CORES9 is a complete full function editor/assembler package that will allow you to create, edit and assemble 6809 machine language programs for the color computer. It features a powerful full function text editor and supports the entire 6809 instruction set with all addressing modes, forward and reverse label references, will output object code directly to memory or "CLOADM" compatible tapes and much more.

Price \$39.95



TEXT EDITOR

This program is a line/character oriented text editor for the color computer, that will enable you to create and edit text files for Basic programs, letters, text data files, or almost anything you might want to put on paper. It features functions for adding, inserting, deleting, moving and copying text lines or paragraphs; powerful string search and replace commands, single and automatic line numbers and line editing with 9 sub commands to insert, delete, change, add and remove individual or multiple characters. Tape commands allow you to save, load, append, and skip tape files; also it is compatible with Basic ASCII tape formats. A MUST HAVE PROGRAM!!

ONLY \$19.95

SYSTEM MONITOR (TRSMON)

Trsmon is a 2K system monitor program that will allow you to explore the workings of the color computer. It features 9 debugging commands, tape load and save compatible with Basic "CLOADM", up/down load via RS232 port, terminal package that allows the color computer to be used as a terminal at baud rates up to 9600 baud and a printer driver to direct display output to the printer for memory dumps, disassemblies etc. The program is position independent so it can be moved anywhere within the system memory. A very powerful tool at a very reasonable price.

ONLY \$19.95

5566 RICOCHET AVE.
Las Vegas, Nv. 89110

CER-COMP
(702) 452-0632

All Orders Shipped From Stock
Add \$1.00 Postage -
MC/VISA Add 3%

Ad-comp

provides the following

- Display Ads
- Complete Typesetting Services
- Artwork
- Camera
- Ad Placement

We can handle any advertising need.

For details call
(616) 452-8649

Monday-Friday 8:00-5:00 p.m.

A Division of

THE "WERE YOUR TYPE"
TYPESETTER

1113 Burton S.W.
Wyoming, Michigan 49509

Subscribe
to CCN

Color Computer News



Are you tired of searching the latest magazine for articles about your new Color Computer? When was the last time you saw a great sounding program listing only to discover that it's for the Model I and it's too complex to translate? Do you feel that you are all alone in a sea of Z-80's? On finding an ad for a Color Computer program did you mail your hard earned cash only to receive a turkey because the magazine the ad appeared in doesn't review Color Computer Software? If you have any of these symptoms you're suffering from Color Computer Blues!

But take heart there is a cure!

It's COLOR COMPUTER NEWS.

The monthly magazine for Color Computer owners and only Color Computer owners. CCN contains the full range of essential elements for relief of CC Blues. Ingredients include: comments to the ROMS, games, program listings, product reviews, and general interest articles on such goodies as games, personal finances, a Kid's page and other subjects.

The price for 12 monthly treatments is only \$21.00 and is available from:



Mail
Today!

REMarkable Software
P.O. Box 1192
Muskegon, MI 49443

NAME _____

ADDRESS _____

CITY _____ State _____ Zip _____

Allow 8-10 weeks for 1st issue.

Processing Numeric Data

Finally, I would like to address the entry of numerical data using the INPUT statement. If data is input into a numeric variable as shown in the example below the routine will accept all legal numeric values. However, not all non-numeric characters are trapped and flagged by the READ error function. Certain alphabetic and symbol keyboard entries have special uses in the input of numeric data. These characters are: E . (space) "down arrow" & + - &H &O. If any of these keys are entered and the ENTER key is pressed, the input variable will be set to the value 0. Entry of the symbols: or, and pressing the ENTER key will display the message EXTRA IGNORED. The input variable will be set to zero and processing will continue. If the character E is embedded in a number the input variable will be set to the exponential value if the number is within the legal range of -1038 to 1038. However, numbers that are out of range can cause some unusual results. Sample entries are shown below.

Program	Entered	Results
10 INPUT A	123E4	1230000
20 PRINT A	123E-4	.0123
30 GOTO 10	123E37	OV ERROR
	123E36	1,23E + 38
	123E777	1,23E + 11
	123E444	OV ERROR
	123E-444	OV ERROR

Basically, if a number is entered in exponent format that is out of range the results are uncertain and BASIC can not be depended upon to flag the entry. If it is necessary to enter numbers where only the digits 0-9 are acceptable entries the following subroutine could be used:

```

50000 B$=""
50010 A$=INKEY$:IF A$="" THEN 50010
50020 IF A$=CHR$(13) THEN 50100
50030 X=INSTR("0123456789",S$): IF X$; B$ = B$
+ A$
50040 LPRINT A$; B$ = B$ + A$
50050 GOTO 50010
50100 RETURN
    
```

Has your TRS-80 Color Computer READ A GOOD TAPE LATELY?

Trying to educate your CoCo can be a trying experience. Pounding on the keyboard is not the positive reinforcement your computer needs. **CHROMASETTE** Magazine is the civilized way to introduce your computer to the world of good software.

With **CHROMASETTE** Magazine, CoCo gets both quantity and quality. Every month, 6 to 8 programs arrive by First Class Mail. No need to type them in — **CHROMASETTE** Magazine is a cassette tape with educational, practical, utility, and game programs on it. Just load and run. Ah, the life of luxury! Give your computer a cultural lesson.

Get a subscription to **CHROMASETTE** Magazine.

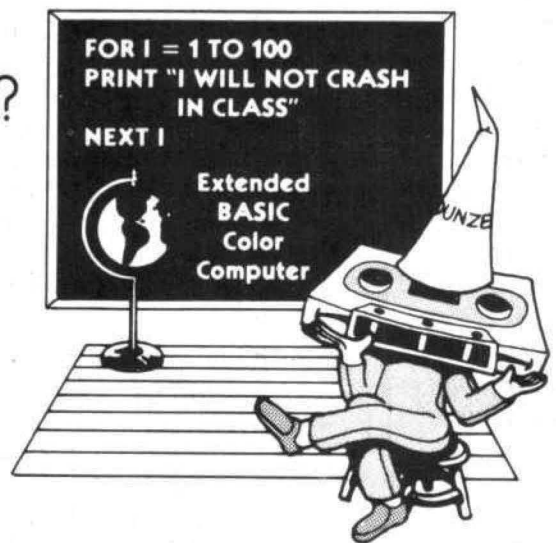
The Bottom Line:

1 year (12 issues) \$45.00
 6 months (6 issues) \$25.00
 Single copies \$5.00

Calif. residents add 6% to single copies.

Overseas — add \$10 to subscriptions, and \$1 to single copies. Sent AO rate.

The Fine Print: Issues are sent First Class Mail. All issues from July 81 on available — ask for list. Programs are for the Extended BASIC model only.
 TRS-80 is a trademark of Tandy Corp. MasterCard/Visa/Gold also welcome



Chromasette Magazine

P.O. Box 1087 Santa Barbara, CA 93102 (805) 963-1066

**MC6809 SOFTWARE
FEATURES AND TECHNIQUES
L.W. Zeliph**

A lot of gossip has been traveling around about the Color Computer's microprocessor chip; the MC 6809. You hear such remarks as 'interim processor', 'high-level language processor', and 'push/pull instructions' to name a few. But what does this mean to us users with our own software applications and information processing duties?

An attempt shall be made herein to clarify a few of these topics in an effort to increase your knowledge of the Color Computer's internals. Consequently, you may discover a new method by which your data processing tasks may be further enhanced.

The MC6809 is considered by its designers (MOTOROLA) to be an upward growth device from its predecessor, the MC6800. This means that applications originally designed for the MC 6800 will execute properly on a MC6809, but some modifications may be required. The term 'interim processor' applies to the MC6809's ability to perform either 8-bit or 16-bit arithmetic. Specifically, the instructions LOAD, ADD, SUBTRACT, COMPARE, and STORE are available in 16-bit format. Present day designers feel that 16-bit processors will become state-of-the-art in the near future, thusly the term 'interim processor'.

The MC6809 is a byte-oriented device. This means that of the 16 bits of data available on the address bus, 8 (a byte) are processed during each execution cycle rather than one at a time. Since high-level languages are designed to process data by specific word lengths (word-length being a certain number of bytes), this feature greatly enhances the MC6809's capability as a high-level language processor. It also lends itself very easily to double precision and string-oriented softwares.

Many of the features to be found in the MC6809 lie in it's powerful assembly language capabilities. Unfortunately, Radio Shack does not provide an assembler as standard software for the Color Computer. They will soon be marketing an optional assembler, and MOTOROLA and various software vendors across the nation have versions available. Care should be taken in selecting an assembler since various versions exist and no standard is available at this time.

For comparison purposes, the MOTOROLA assembler (catalog # ME6809EA) will be cited here for it's summary statistics. This particular

assembler has 59 instructions mnemonics, 268 opcodes, and 1464 instructions when the different types of addressing modes are employed. In addition, very powerful PUSH/PULL instructions, register transfers and exchanges, address-manipulation instructions, and extended-range long branches are provided. PUSH/PULL instructions are used to manipulate data on the two stacks (called system and user) is the MC6809. The register transfers and exchanges allow you to move control and/or data from various points in a program. Data may also be changed or updated and tested for certain results by which further processing can be controlled (for example, subroutine jumps may occur only when a certain value is found in the register of a predetermined control variable). The address-manipulation techniques are extremely exhaustive in variability and an explanation of them in this article would greatly increase it's bulk (perhaps a tutorial on the various address modes and their usage will be the topic of a future article). Lastly, the extended-range long branches are conditional transfers which use 4 bytes rather than the usual 2. They provide greater precision when comparing register contents with operand contents.

All of this comes into focus more readily when you begin to understand and employ the various software features inherent to the MC6809. The value of position-independent code is realized when you notice that older assemblers required that instructions using data in memory registers had to have the register's address affixed to the instruction. With the various modes of addressing available to MC6809 assemblers, memory addresses can be computed internally simply by specifying a starting index value. This value is used to determine the effective address (EA). The assembler will then move to this computed address and use it's contents as the data or as another register address.

The user stack in the MV6809 can be used to develop and execute stack-oriented comiler instructions. These instructions are simply PUSHed onto the stack in a pre-specified reverse order, and PULLed from the stack one by one and executed by the CPU. With this feature, small subroutines (commonly called MACROS) can be accessed through transfers from the system stack (which stores the contents of variables

that are passed between subroutines) and processed under control of the main program. This is referred to as structured high-level subroutine code.

Reentrancy and recursion are two other subroutine techniques available in the MC6809's assemblers. Reentrancy is the ability of a subroutine to be interrupted by a calling program, and still be able to 1) be reentered, 2) finish its processing tasks, and 3) return the correct results to both the interrupting and interrupted programs. Recursion is the ability of a subroutine to call itself. These are considered to be two of the most advanced techniques in structured program design.

Through the use of handshaking, the MC6809 is capable of performing multi-task and multi-processing operations. Handshaking implies the use of the clock strobe in conjunction with the peripheral interface adapter (PIA). Simply stated, each time the system clock pulses, the CPU can send data to a peripheral device, a peripheral device can send data to the CPU (via RAM), or the CPU can handshake - determine if a

peripheral is ready for transmission by sampling its bus port. Since the CPU is capable of speeds much greater than the peripherals, it can sample all I/O ports, handshake only with those which aren't ready, and transmit/receive with those which are ready. By optimization of the principle (along with employing the interrupt vectors and the techniques of reentrancy/recursion) your Color Computer is capable of emulating a full scale minicomputer.

This article was meant to be selective and informative rather than exhaustive. Many other features have not been touched upon, nor were the interrelating functions of those presented fully extrapolated. In the future, I hope to present an article on the addressing modes, then one on the MC 6809 CPU pinouts, and eventually we'll dig into that ROM port and find out how to introduce our new friend to the outside world. In that case, I better get busy studying the CC technical manual and punching the typewriter keys.

COLOR COMPUTER DISK SYSTEM

A complete disk drive system for the color computer, featuring the Tall Grass Technology Double density, buffered disk controller. This system will support up to 4 5 1/4 in. disk drives with a maximum capacity of 3.2 Mega bytes of storage using double sided 80 track drives. This is a minimum of 4 times the capacity of the "Standard" color computer disk drive system.



DISK OPERATING SYSTEM (CCMD + 9)

This is a full featured "Basic" compatible disk operating system which does "NOT" require extended Basic and will even run on a 4K color computer. It includes a complete dynamic allocation system that leaves no wasted or unused space on the disk. It will automatically repack disk space when files are deleted to reduce file fragmentation and increase access time.

This system features three operating systems in one, the first is a free standing system which has 11 commands for loading, saving, removing, changing, checking, analyzing and executing files on disk. It can be configured to allow any mixed combinations of 35, 40 and 80 track drives.

The second system is a completely supported external access system for interfacing with virtually any program requiring the use of the disk system. It includes 10 functions for opening, closing, reading, writing sequential and random access files. There are also 13 subroutine functions and 7 I/O subroutines accessible to the programmer.

The third system is a Basic interface system which includes 6 direct execute Basic commands and 6 indirect commands which conform to the standard Basic tape & printer I/O commands and allow use of string and numeric variables for disk parameters. Up to 9 files can be active at once, all disk file memory allocation is done automatically at run time. Also, Basic has access to all the free standing DOS commands either directly or under program control.

PRICES:

Controller w/CCMD + 9 Eprom	\$159.95
Disk Controller only	\$99.00
CCMD + 9 Dos on 2732 Eprom	\$69.00
CCASM9 disk assembler	\$34.95
CCEDT9 disk text editor	\$24.95
CCDISS disk disassembler	\$29.95
CCUTLY disk utilities	\$19.95
CDTPRO Text processor	\$39.95

Games from Spectral Associates

CGAME1 HI-RES Graphic Game includes Space Invaders Meteroids Space Wars	\$49.95	CGAME2 mixed game disk includes Battle Fleet Space Traders Adventure	\$39.95
---	---------	--	---------

5566 RICOCHET AVE.
Las Vegas, Nv. 89110

CER-COMP (702) 452-0632

All Orders Shipped From Stock
Add \$1.00 Postage - MC/VISA Add 3%

What is RS-232-C and what is it good for? As explained by Andrew Phelps in the November/December Comment Corner, RS-232-C is "your computer's link to the outside world." The outside world consists primarily of printers, terminals (input-output devices consisting generally of a keyboard and a CRT or printer) and other computers. The connection to other computers may be either direct or through a modem (MOdulator-DEModulator, a device which converts the RS-232-C signals for transmission over telephone lines or other media). In general you will require special, though not necessarily unique, hardware (cables) and software (I/O drivers) to communicate with these devices. I will discuss and review software in a subsequent issue. For now, I will try to explain how to get from the 4 pin DIN socket on the back of your Color Computer to the 25 pin plug or socket used by most RS-232-C devices.

Let's begin with a few basic definitions. RS232 generally refers to EIA (Electronics Industries Association) specification RS-232-C which defines an electrical interface for transmitting serial data between digital equipment. The -C denotes that this is the 3rd version of the original standard. There may still be some RS-232-B equipment alive in the world, but generally RS232 and RS-232-C are used interchangeably. Other serial transmission methods that you may occasionally run into are RS-449, which has officially been accepted by our government, but has, so far, not been able to dislodge RS-232 in the general industry, and current loops, which are an older method still used for local transmission in some environments. These alternate transmission standards are not strictly compatible with RS-232, and you will have to buy or build special conversion circuits if you need to use them.

Computer equipment is generally split into two classes: DTE and DCE. Terminals and printers are DTE (Data Terminal Equipment) while modems are DCE (Data Communications Equipment). The various devices are generally connected via cables consisting of 3-25 wires and DB-25 (pin) connectors. Theoretically, any DTE can be connected to any DCE by a cable which connects all 25 pins, or an appropriate subset of them, from end to end. Like equipment (DTE-to-DTE or DCE-to-DCE) can be connected only through an adapter cable which interchanges various pairs of pins. In practice, it is sometimes also necessary to connect jumper wires between

some of the pins at one end of the cable to accommodate handshake requirements. (Handshakes are essentially signals transmitted alongside the serial data which tell the equipment at one end of the cable whether the equipment at the other end is ready to send or receive more data.

Computers were deliberately left out of the preceding paragraph because there is some disagreement as to whether computers are DCE or DTE. While peripheral equipment tends to have a DB25 connector mounted on it, computers tend to have their DB25 connector at the end of a cable which is connected internally to the computer by a variety of means. The computer's DB25 may be either DTE or DCE, depending on which type of equipment it was intended to interface. According to standard convention DTE has a male DB25 and requires connection to a female socket, while DCE has a female DB25 and requires connection to a male plug. This convention, however, is frequently violated and should not be relied upon. A better rule of thumb is to consult the owner's manual for any piece of equipment. If pin 2 is defined as an output (Transmit), the equipment is DTE. If pin 2 is defined as an input (reCeive), the equipment is DCE. Note, however, that the direction, not the name, is the relevant criteria, since some modems will name pin 2 "Transmit Data" or "Data Out" but define it as an input to the modem.

Why do they use a 25 pin connector when only 2 or 3 wires are required to send and receive data between devices? As mentioned above, various handshake signals are transmitted and receive alongside the data. The major signals and pin designations are outlined in table 1. The EIA has also assigned standard definitions and functions to pins 12,13,14,15,16,17,19,21,22,23, and 24, but these are less commonly followed and will probably never concern you when working with the Color Computer. Note, also, that the EIA definitions are not enforced. Most manufacturers will follow the guidelines for the pins indicated in the table, but many have also defined other signals for some of the pins in the second group, as well as the remaining pins (9,10,11,18, and 25).

The TRS-80 Color Computer uses only 4 of these EIA signals:

Pin 1 - Carrier Detect (Equivalent to DTR): This is a control input to the Color Computer telling it that the equipment at the other end is on line. It is connected to a Pin interrupt channel in the Color Computer, but ignored by Color Basic. It

RS-232 The Physical Connection

may, however, be used by some communications software not supplied by Radio Shack.

Pin 2 - Data Input to the Color Computer. This line is used as a printer Not-Busy signal by Color Basic, but will generally be used as a data line by most communications software.

Pin 3 - Ground

pin 4 - Data output from the Color Computer.

The above definitions, along with table 1, indicate that the following cable(s) is needed to interface the Color Computer to another RS-232 device. (Select DTE or DCE as your application requires):

CC Pin	to	DTE Pin	DCE Pin
1	20	6 and/or 8	
2	2 and/or *	3	
3	7	7	
4	3	2	

* in the above table is used to accommodate the Printer Not-Busy signal. If your printer provides such a signal, jump pin 2 to that pin. Pin 4 or pin 20 may provide the necessary signal if not available elsewhere. If you can not find an appropriate signal, you will have to tie the line high (+3 volts or greater) to use your printer. To maintain the flexibility of your system, these jumpers should be made as far away from the CC as possible. I have solved this problem by jumping pin 2 to pin 15 on my DTE cable and then jumping my printer's Not-Busy signal to pin 15 on its side of the connection. Any pin could be used,

providing that it is not being used for another purpose by any equipment that may use the cable.

I have wired the above cables with a male DB25 on my DTE cable and a female DB25 on my DCE cable. The connectors you use should, of course be mated to the equipment you intend to use with the cable. Two auxiliary cables which I use in conjunction with these, as well as with other equipment, are a male-to-female adapter cable which routes pins 1,7, and 15 straight through and interchanges pins 2 & 3, pins 4 & 5, and pin 20 with both 6 and 8; and a male-to-male sex-change cable which connects pins 1-8, 15 and 20 end-to-end.

Using the above set of cables, and the appropriate software, I have been able to interface my Color Computer to all of the aforementioned devices (printer, terminal, modem, and computer). Since EIA standards are not rigorously enforced, however, there is no guarantee that any particular cabling scheme will always work. If you do run into difficulty you may want to try one or more of the following variations: (Always consult your owner's manual for specific signal requirements).

(1) Some equipment will use pin 1 for ground rather than pin 7.

(2) To satisfy handshake requirements you may have to install jumpers. The most commonly used ones are: pin 4 (RTS) to pin 5 (CTS), pin 20 (DTR) to either or both of pins 6 (DSR) and pin 8 (RSL).

Table 1: RS-232-C Standard Pin Designations for DB25 connectors

PIN	NAME	FUNCTION	DTE Dir	DCE Dir	CC PIN #
1	PGND Protective Ground	Ground	-	-	
2	TMI# TransMIT data	Data	Out	In	2
3	REC RECEive data	Data	In	Out	4
4	RTS Ready To Send	Control	Out	In	
5	CTS Clear To Send	Control	In	Out	
6	DSR Data Set Ready	Control	In	Out	
7	SGN# Signal Ground	Ground	-	-	3
8	RSL# Received Line Signal Detect	Control	In	Out	
20	DTR Data Terminal Ready Control	Control	Out	In	1

COLOR COMPUTER BACK

JOYSTICKS

CASSETTE

RS-232-C

RIGHT

LEFT



Putting Numerical Data into Strings
by Richard White

BACKGROUND

The Color Basic and Extended Basic manuals do a good job of defining the basic syntax of the BASIC string functions and commands. They do not, and cannot, cover most of the creative things that can be done with the screen or to other devices. Many of these "tricks" were not designed in, but result from a combination of other properties put in the language. Some of the fun of programming or reading and typing in programs of others is the discovery of a new technique or method that is faster, saves code or saves memory.

A string is primarily intended to hold letters and numbers associated with text. But, strings can also contain numerical data destined to be used in calculations or that results from calculations. Provided the numerical data can be retrieved later, a string can be a much more memory efficient way to keep such data compared with a numerical array. The same functions that are used to process text in strings are used to process and retrieve numerical data in a string. In fact, the numerical data and string characters that relate to that data can be carried in the same string up to the 240 byte limit imposed by Color Basic 1.0. It's this simple. If each variable uses five bytes in the variable table for its address and characteristics, and if 50 data items are related such that they can be stored in sequence in one string variable, then nearly, but not quite 250 bytes of memory can be saved. Savings are even more pronounced since the only way we can store a number as an interger using only one, two or what ever number of characters is to put it in a string.

PUTTING NUMBERS INTO STRINGS

Lets first look at putting numbers into a string. We will use some code from the Checkbook program published in the Nov.-Dec. CCN. In that program, I associated a number of values with each check number or other transaction. These were the check number or transaction, CK\$, the date, DA\$, the amount, QN\$, balance, BA\$, tax deductible flag, TD\$ and notes, PY\$. The amount, for example, is entered into a numeric variable, QN, and is used in finding the new balance, BA. It is then converted to a string QN\$ = STR\$(QN). When all data for the line entry and its calculations are complete, and the data put into strings, the program goes to a

string assembly subroutine to assemble the final data string like this:

```
630 SN$(K) = CK$ + DA$ + "$" + QN$ + "$" + TD$ + "$" + PY$ :RETURN
```

The "\$" serves to separate the data items which will vary in length and will serve as the "target string" when SN\$(K) is disassembled later. Memory overhead to store the data includes the five byte variable table listing, the five bytes for the \$'s and the spaces that VAL adds before each number when it makes it a string. Since the string variable names CK\$, DA\$ and the rest are used over, they tie up a fixed and reused block of memory.

GETTING NUMBERS BACK FROM STRINGS

We must be able to get the data back out of the string and here the key is a Color Computer Extended Basic statement that is absent in Apple and TRS'80 Level II Basics, INSTR. INSTR returns the numerical location of a target string in another string. The target string may range from a single character to a whole sentence. INSTR can be used as the heart of a simple editor in a basic word processor where it finds the start of the phrase to be changed. INSTR can also be used to find the start of the numerical data to be recovered from a string. We put the \$'s in SN\$(K) just so we would have target strings for INSTR to work on. For those without Extended Color Basic, I will give a subroutine that works in place of INSTR later. Read on and understand what INSTR does.

When we use INSTR on SN\$(K) there is one key point to remember. When INSTR finds a \$ and gives us its location, that number is one less than the start of the data in the string. INSTR found the target which precedes the number. Likewise the next \$ found will be the one after the data in question.

In the Checkbook program, an example of string disassembly starts at line 660. The length of the string is found and a number of variables are initialized.

```
660 LS=LEN(SN$(K)):K1=0:L(0)=0:PY$=""
```

K1 is the count variable. L(K1) is a location of the target string "\$" in the string as returned by INSTR. When INSTR finds a \$, K1>0 and the program goes back through the line again looking for the next \$. If INSTR does not find a target string before the end of the search string, it returns a 0. K1 is incremented at the start of

Putting Numerical Data into Strings

the line so we tell INSTR to start looking at location L(K1-1)+1. We also need to tell INSTR that the search string is SN\$(K) and the target string is \$. The final line looks like this:

```
675 K1=K1+1; L(K1)=INSTR(L(K1-1)+1,SN$(K),"$");  
IF L(K1)>0 THEN 675
```

We now have L(1), L(2), L(3) etc. holding the numerical positions of the \$'s in the string. LEFT\$ and MID\$ can now be used to recover the data between these points. The check number CK\$ is the first data in the string. We use LEFT\$, telling it to get the left portion of the string up to the position before the first \$. The form is CK\$=LEFT\$(SN\$(K),L(1)-1). The remaining data is recovered using MID\$ since we need to specify a start position and the length of the data. The form for DA\$ is typical and is DA\$=MID\$(SN\$(K),L(1)+1,L(2)-L(1)-1). Basically we are saying the data starts one to the right of the \$, L(1)+1, and ends one to the left of the next one L(2)-1. The final line is:

```
680 CK$=LEFT$(SN$(K),L(1)-1);  
DA$=MID$(SN$(K),L(1)+1,L(2)-L(1)-1);  
GN$=MID$(SN$(K),L(2)+1,L(3)-L(2)-1).
```

Line 690 is similar to 680 and return BA\$, TD\$ and PY\$. The data is still in strings, but can be easily converted using VAL. For example to get the balance, BA=VAL(BA\$). Since many programs work only with one set of data at a time, the variables like BA\$, BA, CK\$, Ck and the rest are redefined for each calculation and used over and over.

A SUBSTITUTE FOR INSTR

I hope those without Extended Color Basic have stayed with us this far. Now we will develop a subroutine for you to use in place of INSTR. Those with Extended Color Basic may want to read along and understand INSTR better.

We start with the target string. Call it TS\$. Get the length of TS\$, LS = LEN(TS\$). Starting at a specified location ST in the search string SN\$, we want to test character sequences of length LS. We want to repeat the test moving one position at a time along SN\$ until we find the target string character sequence or reach the end of the search string. The following subroutine does this.

```
1000 TT$ = MID$(SN$,ST,LS); IF TT$<>TS$ THEN  
ST=ST+1 ELSE PO=ST; RETURN  
1010 EP = LS + ST; IF EP>LEN(SN$) THEN PO=0;  
RETURN ELSE 1000
```

TT\$ is the sequence from the search string. If there is no match, ST is incremented by one and a test is made in 1010 to see if the end of the search string is past. If so, PO is set to 0 and we RETURN. Otherwise, the program goes to 1000 for another try. If a match is found, PO is

set equal to the current start location ST and we RETURN.

Now lets rewrite line 675 above without INSTR.

```
675 K1=K1+1; ST=L(K1-1)+1; SN$=SN$(K); TS="$";  
GOSUB 1000; L(K1)=PO; IF L(K1)>0 THEN 675
```

Its a bit longer this way, but memory usage is not bad. The main drawback is that the subroutine is in Basic rather than in machine language as is INSTR and is therefore much slower.

GETTING MORE FOR THE MEMORY - HEX\$

When a number is converted to a string using STR\$, it is stored and printed with a leading space, for example ' 678'. Obviously, these spaces can use valuable memory if large numbers of entries are involved. Where the numbers are all integers, memory savings and data recovery simplicity can result if they are converted to hex. The computer treats a hex number as a string, so putting data into strings is similar to what we did above except that the separators or target string, \$, can sometimes be eliminated saving even more memory.

Lets use as an example a grade averaging program where we want to save to tape all grades entered and load the file later to add more grades and find a new average. An elementary school teacher who assigns a lot of work book pages and grades then can generate as many as 50 grades per student per term. Under these conditions, memory is tight. Nearly all grades will be between 0 and 100, but sometimes values over 100 may be given for extra credit. A two character hex number can represent decimal values to 255. If we add leading 0's to those few grades below it, then we can save all grades as two character strings without leading spaces. We can eliminate the separators as well since we know the data is always two characters long.

SAMPLE LINES USING HEX\$ TO HANDLE DATA

In this example, N is the number of data points. It is necessary to test to see that each hex string is two characters long and to add a leading 0 if it is not. Otherwise the line is straight forward.

```
660 FOR K=1 TO N; G$(K)=HEX$(G(K)); IF  
1=LEN(G$(K)) THEN G$(K)="0"+G$(K);  
SN$(SO)=SN$(SO)+G$(K); NEXT
```

Notice that each value of G\$(K) was added to SN\$(SO) before the next value was obtained, getting double duty from the FOR...TO...NEXT loop. The trick of testing for data length and adding leading zeros or blanks works as well for decimal data as for hex.

String disassembly is easier for fixed length hex or decimal data since we don't have to deal with data separators. In the Gradebook

Putting Numeric Data into Strings.

program, the student's name is allotted the first 20 string positions. The number of grades in each string was tacked onto the right end of the string and recovered with `NO=RIGHT$(SN$(KJ),2)`. The code to recover the grades in hex was then `FOR J=1 TO NO: L=19+2*J: G$(J)=MID$(SN$(KJ),L,2)`. Note that since the student's name is allotted 20 positions, the first grade started in position 21, hence the `19+2*J` to define the start position of the next

grade. Converting the hex strings to numbers uses the code `G$(J)="&H"+G$(J): G(J)=VAL(G$(J))`; `NEXT J`.

These techniques are most valuable in storing fixed decimal or integer values and compensate in part for the absence of the `DEFINT` command in Color Basic. They do take some study to get "the hang of". Floating decimal values and data in scientific notation are better handled in numeric arrays.

ML Rabbit

Protect your software investment with ML Rabbit. Software for the Color Computer is too expensive to have only the original tape. ML Rabbit can make Backups of any Color Computer program. No knowledge of the program to be copied is required. ML Rabbit does all the WORK.

Only \$14.95

WORD CC7

Word Processing for the Color Computer. WORDCC7 coupled with your Color Computer & printer turns your machine into a typewriter. Modify and review letters before any ink touches paper.

Only \$19.95 (Ext. Basic)



TUBE CUBE

The Multi-colored cube invades the Color Computer. You can even substitute letters if your color set is busy. Cube Save feature if you can't solve it all.

Only \$9.95

MEMORY

16K RAM	\$25.00
16-32K Solderless Kit	\$49.95
4-32K Solderless Kit	\$74.95

MI Res. add 4% sales tax. Always looking for Great Color Programs. TOP royalties PAID.
Dealer inquiries invited.

DSL Computer Products - P.O. Box 1113 - Dearborn, MI 48121 - (313) 582-3406

COLORTERM (c)

The 16K Color Computer* as an intelligent terminal with 51 or 64 columns by 21 lines and lower case!

- 300 or 110 Baud
- user programmable keys
- automatic repeat when key is held down
- dump your files to host
- reverse video
- partial screen clear
- 4-way cursor control
- any data format (commercial systems, TSO, bulletins etc.)
- memory buffer for incoming data—save buffer—scroll through buffer
- preserve a "window" of any size; new material scrolls through remainder of screen.
- encode data for more secure storage
- macro buffers for often-used output
- patch the 51 or 64 column display to your own programs running above 9168 (23 D0 hex)

Cassette and Manual \$34.95 (U.S.) \$40.95 (Canadian)

Visa, Master Charge, Money Order.

Martin Consulting, 94 Macalester Bay, Winnipeg, Manitoba, R3T 2X5 Canada

*TM OF TANDY CORP

Color computer owners, 32K PLUS DISKS* \$298.00

Yes, that's right - for as little as \$298.00 you can add 32K of dynamic RAM, and a disk interface, to your TRS-80 Color Computer! If you just want the extra memory it's only \$199.00, and you can add the disk interface later for \$99.00.

Just plug the *Color Computer Interface (CCI)*, from Exatron, into your expansion socket and "Hey Presto!" - an extra 32K of memory. No modifications are needed to your computer, so you don't void your Radio Shack warranty, and Exatron give both a 30 day money-back guarantee and full 1 year repair warranty on their interface.

The *CCI* also contains a 2K machine-language monitor, with which you can examine (and change) memory, set break-points, set memory to a constant and block-move memory.

So what about the *CCI Disk Card*? Well as we said it's only an extra \$99.00, but you'll probably want Exatron's *CCDOS* which is only \$29.95 - unless you want to write your own operating system. The *CCI Disk*

Card uses normal TRS-80 Model I type disk drives, and *CCDOS* will even load Model I TRSDOS disks into your color computer - so you can adapt existing TRS-80 BASIC programs.

As a further plus, with the optional *ROM Backup* adaptor, you can dump game cartridges to cassette or disk. Once the ROM cartridge is on cassette, or disk, you can reload, examine and modify the software. The *ROM Backup* adaptor is only \$19.95.

For more information, or to place an order, phone Exatron on their Hot Line 800-538 8559 (inside California 408-737 7111), or clip the coupon.



excellence in electronics

exatron

DEALER ENQUIRIES INVITED

Exatron,
181 Commercial Street,
Sunnyvale, CA 94086



- Please send a 32K Color Computer Interface for \$199.00
- Please send a CCI Disk Card for \$99.00
- Please include CCDOS and manual for \$29.95
- Also include a ROM Backup adaptor for \$19.95

Please add \$5.00 for shipping to all orders, and 6 percent sales tax in California.

Name

Address

City

State Zip

Charge my:

MasterCard Interbank Code

Visa Expiration Date

Card

Check enclosed for

Ship COD (\$2.00 extra)

Signature