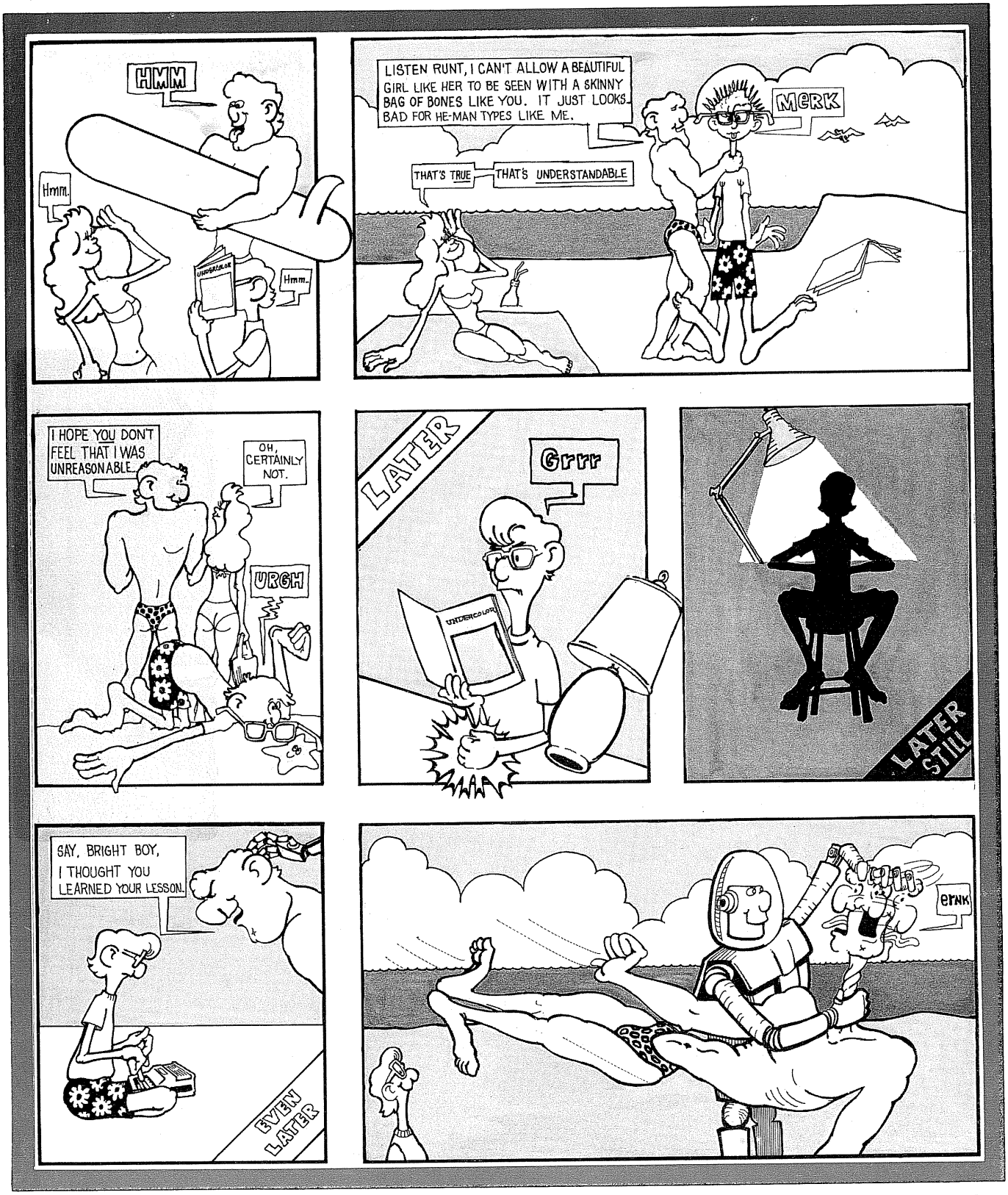
UNDERCOLOR

Vol. 1 No. 8

April 1, 1985



POSTAGE PAID
MANCHESTER, NH
PERMIT #1926

$2

A SPECTRUM PROJECTS SHOPPING LIST

A CHIP OFF THE OLD...

6821 Standard PIA $9.95
68766 (Fits Disk Basic Skt) Eprom.$24.95
27128 16K DOS Eprom (Or 2 DOS's!).$24.95
New SAM Chip w/heatsink (74LS785).$29.95
6809E CPU Chip $29.95
Ext Basic 1.1 ROM - NEW LOW PRICE.$29.95
28 pin Ext Basic upgrade 26-3134A.$34.95
Basic ROM 1.2 Chip (30% FASTER) ..$39.95
Disk ROM 1.1 (New DOS Command) ..$39.95
Eprom Eraser - 3 min erasure time.$49.95
Lower Kit Bd - Specify CoCoI/II**.$59.95
CoCo First Aid Kit - includes 2 PIAs, 6809E & SAM (Be Prepared!!!) $59.95
26-3134A & 26-3136A 64K Upgrade - 2 chip set (ONLY for new Korean CoCoIIA).$69.95
Eprom Prgmr (2ms speed/2K - 16K).$139.95
64K CoCo II - w/NEW keyboard $169.95

COCO LIBRARY...

CoCo Memory Map $12.00
Rainbow Book & Tape of Adventures.$14.95
Basic Programming Tricks Revealed.$14.95
The FACTS - Inside "guts" of CoCo.$14.95
500 Pokes, Peeks 'N Execs $16.95
Basic 09 Tour Guide $18.95
Disk Basic (1.0/1.1) Unraveled ...$19.95
Color Basic 1.2 Unraveled $19.95
New! CoCo II Service Manual $19.95
The Complete Rainbow Guide to OS9.$19.95
W/Two Disk Package of demo pgms ..$49.95

MORE GOOD STUFF...

27128/68764 Adapter - Allows you to plug-in 28pin EPROMs into 24pin sockets! Fits all controllers for 16K DOSs $19.95
CoCo Freeze Frame - Stop your CoCo dead in its tracks! Put games on "HOLD"$19.95
CoCo Light Pen with 6 programs ...$24.95
Spectrum Voice Pak - SAVE $30!! ..$39.95
PBH Parallel Interface - Beats Botek! 300-9600 baud w/ptr-modem switch .$69.95
PBJ WORD-PAK II - Hi-Res 80x24 display w/smooth scrolling & 8x10 matrix $139.95
PBJ C-C Bus - 6 slot interface ..$149.95
NOMAD - The CoCo Robot $199.95

All orders plus $3.00 S/H (Foreign $5.00)
COD add $2.00 extra
NYS Residents add Sales Tax

COCO CABLES AND...

Printer/Modem 15' Extender Cable .$14.95
Tired of unplugging devices from your RS232 port? Try a RS232 "Y" Cable.$19.95
Joystick/Mouse 10' Extender Cable.$19.95
Null Modem Cable - 40 pin to DB25.$24.95
Disk Interface/Rom Pak Extender - Move your disks & ROM Paks (3 feet) ..$29.95
Triple RS232 Switcher - Now select one of any three RS232 peripherals ...$29.95
40 Pin Dual "Y" Cable - Hook up a Disk w/Voice, Word Pak, CoCo Max,etc ..$29.95
Triple "Joyport" Switcher - Joystick, Mouse, Touch Pad and/or Light Pen $39.95
40 Pin Triple "Y" Cable - Hook up any 3-Voice/Word/RS232/Digitizer PAKs ..$39.95
Finally! 24" Multi-Pak Extender ..$39.95

OTHER GOOD STUFF...

C-10 tapes in any quantity .....49 cents
5 1/4" Diskettes in any quantity ..$1.49
Joystick, Cassette or Serial plug .$3.99
32K, 64K or 128K RAM Button $4.99
Rompak w/Blank PC Brd 27xx series .$9.95
The Disk Doubler - Doubleside your 5 1/4 diskettes for 160K more storage ..$19.95
Video Clear - This cable will reduce TV interference created by CoCo! $19.95
Video Reverser -Reduce eyestrain w/green letters on black background** $24.95
The Magic Box - Load Mod I/III Basic program tapes into the CoCo $24.95
DOS Switcher - Select from any two DOSs (Disk 1.0 1.1, JDOS) in J&M ctlr .$24.95
Stereo Pak - Hardware synthesizer used w/Musica 2...superb stereo sound! $39.95
CoCo Cooler- State D,E or CoCoII .$49.95
MicroWorks Video Digitizer $149.95
** N O T for 26-3134/36 CoCo II's

SPECTRUM PROJECTS
PO BOX 21272
93-15 86TH DRIVE
WOODHAVEN NY 11421

718-441-2807

SPECIAL 64K RAM CHIPS $29.95

Who's Who

Editor
Debra A. Marshall

Technical
Dennis Bathory Kitsz

Contributing Photographer
Charley Freiberg

Production Manager
Clare McCarthy

Paste-Up
Janet Patterson

Logistics Coordinator
JoAnn Trottier

Technical Consultant and Publisher
Dennis Kitsz

Advertising:
Judy Knapp
(802)485-6139

Advertising:
Spencer Knowlton
(207)785-4151

Office
(802)485-6440

Editorial Office
(207)785-5148

Published By: ColorPlus Inc., Box 6809, Roxbury, VT 05669. (802)485-6440.

Circulation Problems? 2nd Class permit pending. Contact David Gunn, Box 6809, Roxbury, VT 05669, 802-485-6440.
Subscriptions: United States 1 year—$33; 2 years—$66. First class mail 1 year—$47; 2 years—$94. Canada and Mexico 1 year $43; 2 years $86. Overseas 1 year $53 (US dollars); Airmail $65 (US dollars). UnderColor is published 26 times a year. Call (802)485-6440, or write to UnderColor, Box 6809, Roxbury, VT 05669, Attn: Perley Swann.
Advertisers: rates available.
Editorial Office: c/o Debra Marshall, RFD 1, Box 786, Union, ME 04862, 207-785-5148. Submissions are welcome; include S.A.S.E. for returns. To receive author's guide, write: Author's Guide, Box 6809, Roxbury, VT 05669. Letters to Editor, manuscripts, article queries, etc. should be sent to editorial office.

Table of Contents

• **Defusr** . . . . . page 2
By Terry Kepner *Q&A*

• **The Data Gatherer** . page 4
By Dennis Kitsz *Interim Information*

• **Hints, Tips & Tricks** . page 7
By Terry Kepner *Good stuff.*

• **Nimble Fingers** . . . . page 8
By Stewart J. Peck *The end of hunt and peck.*

• **Off Color: On Computer Porn** . . . . . page 12
By M.K. Charbonne *Can computers have sex?*

• **On Sig—** . . . . . page 14
A CompuServe discussion continued.

• **Errors and Omissions** page 18

• **Transfer** . . . . . page 19
By Dale Keller *CCEAD to EDTASM.*

• **Boy Fix!** . . . . . page 20
By Don Hutchinson *Disk Basic flaws.*

• **Reviews** . . . . . page 22

DefUSR

By Terry Kepner

Is it possible to auto-run a program using a Basic code, and if so, how?

T.C. Taulli
Monrovia, CA

No, you can't do it just using Basic code. The Basic interpreter controls the loading and it always retains control when loading Basic programs. After the program loads, Basic returns to the keyboard scan routine and waits for your instruction.

Basic programs that seem to auto-run are actually using a special assembly code loader routine that preceeds them on the tape. This loader takes control away from the Basic interpreter and starts the Basic program after it finishes loading.

*I have recently received the assembly language Summer Programminng Project-winning game, "Bugs (*TCCM, *1983)." My compliments go to Mssrs Shewchun and Knight for an exceptional game.*

Not having a disk drive, I sent for the "Bugs" listing with the intent of having it available at a later date. I was very pleasantly surprised to have the program arrive on cassette, but even more surprised to find the 64K game loaded on my 32K computer!

Is my computer 64K? I purchased it in August 1983, have Extended Basic 1.1, an 'F' board, and serial number 0045860.

Will commercial programs designated for 64K run on my computer without a disk drive and allow the same available RAM as a 64K unit with disk?

Does educational literature exist that bridges the gap between Radio Shack's dancing computers in the Color and Extended Color Basic documentation and the literature on assembly language and hardware architecture that seems to asume we are all degreed in Computer Science? My specific interest is to learn more about general structure and design of hardware without actually learning top-to-bottom assembly language or computer electrical engineering.

What literature do you recommend that goes into a more detailed explanation of topics I feel were inadequately covered in Section IV of Getting Started with Color Basic? *It seems as though functions such as USR,DEFUSR, POLCAT, and ML subroutines are glossed over compared to other Basic commands and functions in this manual. Is there help for us slow learners?*

Donald G. Hills Jr.
Dunkirk, NY

Yes, the computer is 64K. All 32K computers purchased from Radio Shack have 64K chips installed, but not all of them allow access to the upper 32K bank.

Yes, commercial 64K programs will run on your computer, whether or not you have disk drives.

Bill Barden's books on assembly language (available at Radio Shack) seem to be about the easiest ones around for teaching assembly language programming.

The ML routines are covered pretty well in Bill Barden's books.

I own a 26-1191 Multi-Pen Plotter from Radio Shack which I use in my word processing business for creating graphs, etc. with my Model III. I have tried to convert Tandygraph (which is not available for the CoCo, though it was advertised, as the Plotter is now discontinued), and make several programs for the CoCo in order to use it as my "graphics" machine while doing word processing on the III (you can say all you want about the power of the 6809, but the high quality software and support and plug-in compatibility just isn't there for the CoCo without doing a lot of "kludging").

Enter my problem: The III makes good use of the plotter's RS-232 I/O port, and is a good friend, even to the point of doing repeated handshakes whenever they feel like it. Not so with the CoCo. Mr. Sixty-Four Key just won't stick out his paw, and inundates Mr. Plotter with instructions to the point of confusing him and causing him to do strange things with his pens. Multiple embedded For/Next loops to slow down the sending of data make the plotter move at a good clip at times, but then take forever to draw a letter or something short. Appropriate For/Next timing loop values just cannot take into account all the different kinds of graphics, etc., especially since Tandygraph (which works well in the CoCo) is so long that I have to do a PCLEAR 0 (using the appropriate methods) to get it to work (Tandygraph requires a lot of variable space). If I could only get the CoCo to recognize a clear to send signal from the Plotter, or something similar, which would halt the CoCo's transmission until the Plotter is ready to receive more data.

All the pertinent data follows. Help!

Plotter: Handshakes always occur between data blocks of 256 bytes or less and the computer should transmit a CR (0D16) at data block end. Plotter will respond with a CR (0D16) if it can receive 256 bytes. Commands are in the form of standard LPRINT (or PRINT #-2, "place commands here" statements. Baud is 2400 firm.

Plotter RS-232 pin-out: 1—chassis ground; 2—receive data from computer; 5—CTS from plotter; 6—DSR from plotter; 7—signal common; 8—CD from plotter; and 22—RI from plotter (ring indicator). Signals 2 and 3 are high = −3 to −25v low = ±3 to ±25v. All others TTL H = 2.4 − 5v L = 0 to .4v.

I have used the standard CoCo 26-3014 DB25 to DIN plug cable.

Enclosed is a Basic listing (not included, see page 31 of the plotter manual) which works perfectly and seems to be an accurate disassembly which appears to be primarily a 2400 baud drive, though I confess to be lacking in assembly skills.

So. What I need, then, is some way to get some handshaking done, with the plotter letting the computer know when it can or can't send data! Easy, I bet!—but not for me. Thank you for any help you can give.

Henry C. Gernhardt Jr.
Huntingtonn, WV

The locations $FF20 and $FF21 control the RS-232 port. Bit 1 of $FF20 must be set to one for RS-232 data output. Bit one of $FFew controls how the computer responds to the CD signal on pin 1 of the 4-pin DIN connector. If bit one is zero, the

register flag is set on the falling edge of the CD (when CD goes down). If the bit is one, the register flag is set on the rising edge of the CD (when the plotter raises CD).

What you need to do is wire pin 8 from the plotter to pin 1 on the computer. Now you'll need a short assembly language routine you can call with USR that will simply check $FF21 by loading it into the prime register and seeing if the register flag indicates if the plotter is ready.

Since I don't know if the plotter holds CD high or low to indicate ready, you'll have to experiment.

I have a Color Computer 2 with Extended Color Basic, 16K RAM. What are the chip numbers on the 16K RAM chips and the rest of the ICs in the CoCo 2? (By "chip numbers" I mean the number like "6809E MICROPROCESSOR," or "4164-150 DYNAMIC RAM CHIPS.") How many ICs are in the CoCo 2, and what are their uses? How is the voltage divided from the transformer?

Is the Model 4 character generator compatible with the CoCo 2?

What are the different circuit parts in the CoCo 2?

What is a breadboarding kit, and where can I find one? Does Radio Shack carry the 6821PIA?

What are the POKEs to disable the Break key and List, and who on the market has a power supply for the Coco 2, or a numeric keypad?

Stephen Mahan
Williamsburg, KY

Rather than go into a technical description, I'll refer you to the *Color Computer 2 Technical Manual,* available from Tandy National Parts, 900 E. Northside Drive, Fort Worth, TX, 76102, 817-870-5662. Mastercharge and Visa are accepted and each order includes a $1.50 handling charge. The manual number is the same as your computer's order number except you add the letters MS in front.

If you mean the Model IV character generating chip, the answer is no. If you mean the assembly language character generator driver, again the answer is no. The 6809 machine codes are incompatible with the Z-80 codes.

A breadboarding kit is an exposed circuit board that brings the various electronic lines out so you can work directly with them without messing around with the computer's circuit board. Typically, people use these kits to design devices for use with the Color Computer. After the circuit is put together (usually by just plugging the components into special solderless pads), the entire board is plugged into the Color Computer and tested. Modifications are easy to make, due to the breadboarding kit design. Once a design is thoroughly tested and approved it can be transferred to a custom circuit board that's much smaller and lighter than the breadboard.

You can get spare chips from Spectrum Products, 93-15 86th Dr., Woodhaven, NY, 11421, (212) 441-2807.

There isn't just a simple POKE to disable the Break key or List command, you have to use a short assembly language routine that intercepts the ROM keyboard scan and checks it for your "undesirable" key strokes.

And I don't know of anyone who manufacturers power supplies for the Color Computer, or numeric key pads. Sorry.

As I understand it, the Radio Shack Color Computer 2 has a much simpler power supply set-up than the earlier models. If this is so, would it be possible to set up this machine using battery power?

The advantages would be numerous. Such variables as power line surge would be eliminated. The power transformer, which is the prime source of heat in this machine, would be eliminated. Most of all, it would be possible to set up a portable computer at a fraction of the cost of the portable units.

While this unit would certainly not be as small as other portable units, it would be more than adequate for a number of applications where portability is necessary but size is not as critical, such as use in a camper or other vehicle.

Since cassette operation is fairly straightforward on low-voltage applications and numerous 12-volt television sets are available for use as a monitor, it looks like it would be fairly straightforward. Please advise.

Ken Hogan
St. Louis, MO

You certainly could rig the Color Computer to work with battery power. But while the power requirements are simpler, the current drain is still very high.

Most of the portabale-battery computers use low-power CMOS chips, which use only 1/10th the current of the standard chips. About the only batteries capable of powering the Color Computer for extended periods of time are 12-volt car batteries. If you intend to operate the unit off a car or camper, then you shouldn't have too many problems. Just be careful you don't accidentally kill the car's battery.

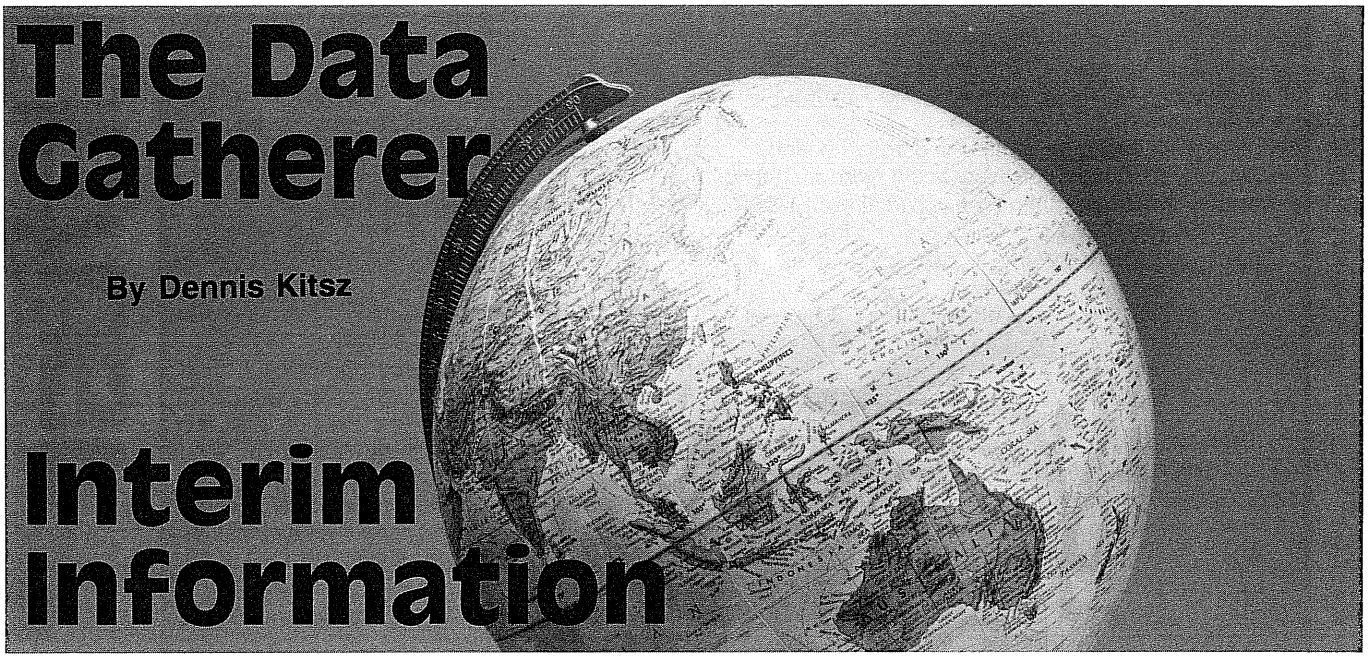
If the car or camper has the motor running, don't plug the computer into the system: there's a lot of static and electrical garbage floating around that could destroy the delicate chips in your Color Computer.

I own both a CoCo and a Dragon 64K. I have the Dragon connected to my DMP-200 printer, which is set up in the parallel mode. My problem is that when I use the Radio Shack Graphic Pak cartridge, which seems to work fine on the Dragon otherwise, it won't give me a printed copy. Can you tell me why?

Charles M. MacLeod
Taunton, MA

It depends on whether or not you're using the RS-232 or the Dragon's parallel port for printer output. The Graphics cartridge sends its output directly to the RS-232 port, not the parallel port (which doesn't exist on the Color Computer).

The Data Gatherer



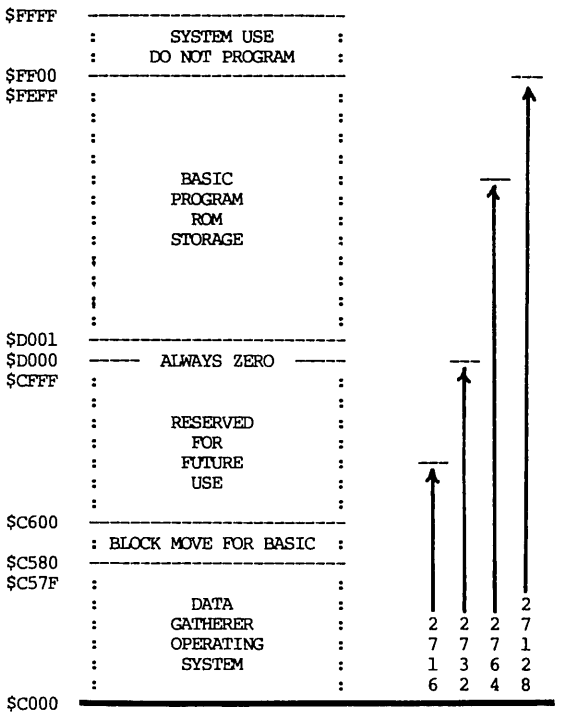
By Dennis Kitsz

Interim Information

Options Summary

The Data Gatherer has several sections (ROM, Clock, DAC) which may be used or combined in a variety of ways. These options are chosen by jumpering points on the circuit board.

Point	Function
a	from SCS*, output from CPU
b	to SCS*, output to feed-thru
y	from $FF40-4F SCS*, output from decoder USE: Jumper y to b for $FF40 SCS* to disk via feed-thru. Jumper a to b for complete SCS* to disk via feed-thru (no clock circuitry in Data Gatherer).
c	from CTS*, output from CPU
d	to CTS*, output to feed-thru USE: Jumper c to d for CTS* from $C000 to $E000 via feed-thru (no operating ROM in Data Gatherer).
e	from INT*, output from clock
f	to NMI*, input to CPU USE: Jumper e to f for NMI* interrupt generated by clock circuitry.
g	SPAN, 20v span DAC control line
h	10V span select
k	5V span select USE: Jumper g to h to select 0-10V analog input. Jumper g to k to select 0-5V analog input.
m	from SWAPSCS*, output from decoder
n	to CLKSEL*, clock select line
o	from $FF40-4F SCS*, output from decoder USE: Jumper m to n when using clock circuitry and disk system. Jumper o to n when using clock circuitry without disk.
p	from IRQB*, output from U2
q	to NMI*, input to CPU
v	from IRQA*, output from U2 USE: Jumper p to q to use IRQB* as interrupt input to CPU. Jumper v to q to use IRQA* as interrupt input to CPU. Both jumpers may be used.
r	to E*, enable input to ROM
s	from CTS*, output from CPU
t	from ECTS*, $E000 output from decoder USE: Jumper r to s for Data Gatherer ROM selection at $C000. Jumper t to r for Data Gatherer ROM selection at $E000. Latter to be used when disk system is in place; do not jumper c to d (viz.)
J	from IRQA*, output from U3
u	to NMI*, input to CPU
z	from IRQB*, output from U3 USE: Jumper z to u to use IRQB* as interrupt input to CPU. Jumper J to u to use IRQA* as interrupt input to CPU. Both jumpers may be used.
w	from Q, output from CPU
x	to CART, input to CPU USE: Jumper w to x when autostart at $C000 is needed. Not used for disk system. Must be used with jumper r to s (viz.)



Data Gatherer Memory Map (not to scale)

$\overline{CS}$	A3	A2	A1	A0	Operation
1	X	X	X	X	No Operation
X	1	1	1	1	No Operation
0	1	1	1	0	Enable 4 LSBs of First Rank
0	1	1	0	1	Enable 4 Middle Bits of First Rank
0	1	0	1	1	Enable 4 MSBs of First Rank
0	0	1	1	1	Loads Second Rank from First Rank
0	0	0	0	0	All Latches Transparent

"X" = Don't Care

AD667 Truth Table

Parts List

U1	74S139N decoder/demultiplexer
U2-3	MC6821P peripheral interface adapter
U4	AD667JN digital-to-analog converter
U5-6	CD4529B dual analog multiplexer
U7	74LS73N positive edge flip-flop
U8, 12	74LS32N quad two-input OR gate
U9	7417N hex open-collector buffer
U10	MM58274J real-time clock/calendar
U11	74LS04N hex inverter
U13	LM311N precision voltage comparator
U14	LM7805CT positive 5-volt regulator
U15	2732/64/128 operating system (USER SUPPLIED)
C1, 3-18, 21-22, 29-31	0.1 mF monolithic ceramic capacitor
C2	100 pF ceramic disk capacitor
C19	10 mF / 16V radial electrolytic
C20	0.01 mF ceramic disk capacitor
C23	5-50 pF subminiature variable
C24-25	20 pF dipped mica 5% capacitor
C26	56 pF dipped mica 10% capacitor
C27-28	47 mF / 35V radial electrolytic
R1, 4	1K, 1/4 W, 5% resistor
R2	100 ohms, 1/4 W, 5% resistor
R3	100K, 1/4 W, 5% resistor
R5	470 ohms, 1/4W, 5% resistor
R6	4.7K, 1/4 W, 5% resistor
R7	2.2K, 1/4 W, 5% resistor
R8-14	10K, 1/4 W, 5% resistor
R15	3.3K, 1/4 W, 5% resistor
VR1	50K variable resistor
VR2	100 ohm variable resistor
VR3	5K variable resistor
D1-2	1N4148 small-signal diode
D3	1N5227B 3.6V zener diode
D4-5	1N5242B 13V zener diode
Q1	2N3904 NPN transistor
Q2	2N3906 PNP transistor
X1	32768 Hz crystal
B1	3.6V NiCad battery stack
J1	40-pin 0.05" IDC transition connector
J2	36-pin 0.1" 90-degree 3M-type latching connector
J3	40-pin 0.1" 90-degree female edge card connector
J4	20-pin 0.375" screw connector
J5	1-pin 0.375" screw connector
(off board)	40-pin 0.1" IDC male connector
(off board)	18" 40-conductor shielded ribbon cable, 0.05" spacing

Miscellaneous:

1	8-pin low-profile socket
5	14-pin low-profile sockets
4	16-pin low-profile sockets
1	24-pin low-profile socket
1	28-pin low-profile socket
2	40-pin low-profile sockets
1	TO-220 heat sink and hardware
1	Printed circuit board
1	+/-12 volt power supply (USER SUPPLIED)

Register Selected		Address (Binary) AD3	AD2	AD1	AD0	(Hex)	Access
0	Control Register	0	0	0	0	0	Split Read and Write
1	Tenths of Seconds	0	0	0	1	1	Read Only
2	Units Seconds	0	0	1	0	2	R/W
3	Tens Seconds	0	0	1	1	3	R/W
4	Units Minutes	0	1	0	0	4	R/W
5	Tens Minutes	0	1	0	1	5	R/W
6	Units Hours	0	1	1	0	6	R/W
7	Tens Hours	0	1	1	1	7	R/W
8	Units Days	1	0	0	0	8	R/W
9	Tens Days	1	0	0	1	9	R/W
10	Units Months	1	0	1	0	A	R/W
11	Tens Months	1	0	1	1	B	R/W
12	Units Years	1	1	0	0	C	R/W
13	Tens Years	1	1	0	1	D	R/W
14	Day of Week	1	1	1	0	E	R/W
15	Clock Setting/ Interrupt Registers	1	1	1	1	F	R/W

Address Decoding of Real-Time Clock Internal Registers

Subroutine Reference

JSR INIT

Initialize all ports and vectors after reset or power-on, and display sign-on message. Turns on clock. Does not initialize interrupt. Use JSR CLKOFF first before JSR INIT.
Entry: none.
Exit: none.

Basic: EXEC CF : EXEC IN

JSR REINIT

Same as above, but do not display sign-on message. Use from software or after reset to assure proper parameters. Turns on clock. Does not initialize interrupt. Use JSR CLKOFF first.
Entry: none.
Exit: none.

Basic: EXEC CF : EXEC RE

JSR CHANNL

Turn on one channel of 16-channel multiplexer. Value 0 turns off all channels.
Entry: channel (0-16) in MXCHAN.
Exit: none

Basic: POKE MX with 0 to 16, then EXEC CH

JSR GETVAL

Get one value from ADC channel specified.
Entry: channel (1-16) in MXCHAN.
Exit: data in MVALUE (2-byte integer).

Basic: POKE MX with 1 to 16, then EXEC GE. Value found in MV: PRINT 256*PEEK(MV) + PEEK(MV+1)

JSR GET16

Get 16 values from ADC channels 1 to 16.
Entry: none.
Exit: values returned as 16 2-byte integers at STOR32; order is channel 16 first in, channel 1 last in.

Basic: EXEC G6. Values found at ST: FOR X=0 TO 31 STEP 2 : PRINT 256* PEEK(X+ST) + PEEK(X+ST+1) : NEXT X

JSR CONVRT

Get one value from ADC channel already open (opened by JSR CHANNL). 3mS conversion. Repetitive (modular) version. Use JSR FASTAD for 1.6 mS conversion.
Entry: none.
Exit: data in MVALUE (2-byte integer).

Basic: EXEC CO, then: PRINT 256*PEEK (MV) + PEEK(MV+1)

JSR DACOUT

Send value to DAC and strobe through to output of converter. Use JSR FASTDA for 80 uS version.
Entry: value to send (0-4095) as 2-byte integer in VALHI.
Exit: none.

Basic: Break value into integer format, and POKE VA,MSB : POKE VA+1,LSB. Then: EXEC DA

JSR FASTDA

80 uS output routine through DAC, similar to JSR DACOUT.
Entry: data to send (0-4095) as 2-byte integer in VALHI.
Exit: none.

Basic: Break value into integer format, and POKE VA,MSB : POKE VA+1,LSB. Then: EXEC FA.

JSR FASTAD

1.6 mS input routine through DAC. Assumes channel is already open. Fast, nonrepetitive version. Use JSR CONVRT for modular version.
Entry: none.
Exit: data in MVALUE (2-byte integer).

Basic: EXEC FD. Then: PRINT 256*PEEK(MV)+ PEEK(MV+1)

JSR CLKSET

Reset time and date in real-time clock.
Entry: 16 bytes in STRSTR. See Basic format below.
Exit: none.

Basic: Prepare time and date in the form A$="DYYMMDDHHMMSSS" where D=1 to 7 (Sunday is day 1), YY is the year, MM is the month (January is 01), DD is the day, HH is the hour (24 hour clock is used in this version) MM is the minute and SSS are the seconds and tenths of seconds. POKE SR,PEEK(VARPTR(A$)+2) : POKE SR+1,PEEK(VARPTR(A$)+3) : EXEC CS

JSR DISCLK

Display time and date on screen at top right.
Entry: DISPLC contains address for display; if not specified, $0409 is used (top right). Use $05E9 for bottom right, but scrolling will occur if Basic programming reaches the end of screen.
Exit: none.

Basic: EXEC DK. To change display, break display address (1024 to 1513) into integer format, and POKE DP,MSB : POKE DP+1,LSB.

JSR GETCLK

Get time and date into memory.
Entry: none.
Exit: CLKSAV contains time/date in 14 bytes. See Basic version below for format.

Basic: EXEC GK. Time and date are found unconverted at CL (1=SUN, 7=SAT) in the order: day, tens of years, years, tens of months, months, tens of days, days, tens of hours, hours, tens of minutes, minutes, tens of seconds, seconds, tenths of seconds. No punctuation in this format.

JSR CLKON

Turn clock on, with 60 Hz update, and display on screen.
Entry: DISPLC contains address for display. Default address is $0409 (top right).
Exit: none.

Basic: EXEC CN

JSR CLKOFF

Turn clock off and stop screen display. No action taken if clock is already off.
Entry: none.
Exit: none.

Basic: EXEC CF

JSR PRINTI

Print one character and return. DOES NOT WAIT for printer ready.
Entry: PRINTC contains value to print.
Exit: none.

Basic: POKE PC,ASC(A$), where A$ is character to print. Then: EXEC PI.

JSR PRINTS

Print a string of characters. WAITS for printer ready. (Printer format is parallel printer, Centronics-type, busy on pin 11, busy when LOW).
Entry: STRSTR points to string of ASCII characters. String must terminate with $00 or $0D.
Exit: none.

Basic: Where A$ is a string of ASCII characters. Set up: POKE SR, PEEK(VARPTR(A$)+2) : POKE SR+1, PEEK(VARPTR(A$)+3) : EXEC PS.

JSR PRINTV

Print contents of video screen (text screen only). WAITS for printer ready. (Printer format is parallel printer, Centronics-type, busy on pin 11, busy when LOW).
Entry: SSTART points to screen of $400 bytes.
Exit: none.

Basic: EXEC PV

```
JSR SWAPD       Swap to disk; for recovery purposes.
                Entry:  none.
                Exit:   none.

                Basic:  EXEC SD

JSR SWAPC       Swap to clock; for recovery purposes.
                Entry:  none.
                Exit:   none.

                Basic:  EXEC SC

JSR COPYRT      Display copyright message, owner and
                version number.
                Entry:  none.
                Exit:   none.

                Basic:  EXEC CT

Vectors (indirect) beginning at $C008:
IPL, INIT, REINIT, CHANNL, CONVRT, GETVAL, GET16, DACOUT,
FASTAD, FASTDA, CLKSET, DISCLK, CLKON, CLKOFF, GETCLK,
PRINTI, PRINTS, PRINTV, SWAPD, SWAPC, COPYRT
```

Function	Comments	Control Word			
		DB3	DB2	DB1	DB0
No Interrupt	Interrupt output cleared, start/stop bit set to 1.	X	0	0	0
0.1 Second	DB3 = 0 for single interrupt DB3 = 1 for repeated interrupt	0/1	0	0	1
0.5 Second		0/1	0	1	0
1 Second		0/1	0	1	1
5 Seconds		0/1	1	0	0
10 Seconds		0/1	1	0	1
30 Seconds		0/1	1	1	0
60 Seconds		0/1	1	1	1
Timing Accuracy: single interrupt mode (all time delays): ± 1 ms Repeated Mode: ± 1 ms on initial timeout, thereafter synchronous with first interrupt (i.e., timing errors do not accumulate).					

Interrupt Control Register

Function	Data Bits Used				Comments	Access
	DB3	DB2	DB1	DB0		
Leap Year Counter	X	X			0 Indicates a Leap Year	R/W
AM/PM Indicator (12-Hour Mode)			X		0 = AM 1 = PM 0 in 24-Hour Mode	R/W
12/24-Hour Select Bit				X	0 = 12-Hour Mode 1 = 24-Hour Mode	R/W

Clock Setting Register Layout

Hints, Tips & Tricks

By Terry Kepner

Programming—If you want to slow down program scrolling in response to list, type POKE 359,60. To restore it, type POKE 359,126. If you want an orange screen border, type SCREEN 0,1.

Programming—And if you want to disable list entirely, type POKE 383,158. To restore it, type POKE 282,0.

Programming—If you want your program to switch to lowercase input for some prompts and uppercase for others, use POKE 282,0 to go into lowercase and POKE 282,1 to return to uppercase. This way the user doesn't have to remember to press the shift key for certain prompts.

Disk Basic—To speed up single-disk back-ups and reduce the number of disk swaps, type PCLEAR:FILES:BACKUP.

Programming—Want to draw a line at a non-standard angle while in Extended Basic graphics mode? Use the M (move) command without using the B option first.

Programming—If you have to evaluate two variables in an If . . . Then statement, it is faster, in execution, to use two If . . . Then statements than one with a Boolean operator. For example, IF X = 1 AND Y = 1 THEN PRINT Z will take a longer determination whether to print Z than will IF X = 1 THEN IF Y = 1 THEN PRINT Z. This is because the If statement with an And must evaluate the values of both X and Y before it knows not to print Z, while the second If statement knows it can't print after looking at only X. If both X and Y are true, both statements take the same amount of time to execute.

Programming—If you want to achieve the execution times of two pieces of code, try this:

```
10 PRINT"First Code";TIMER
20 FOR I=1 TO 1000
30 code to be tested . . .
. . .
90 NEXT I:PRINT"End code ";timer
100 PRINT"Second Code";TIMER
110 FOR I=1 TO 1000
120 code to be tested . . .
. . .
190 NEXT I:PRINT"Code ended";TIMER
```

For short pieces of code, just subtract the times from one another and you'll get a relative speed comparison. For longer pieces, set the timer values to variables and have the program subtract them. Don't forget the timer resets at 65535, so if you get a negative number, add 65336 twice.

Programming—Rather than use a For . . . Next loop as a time delay, use the timer function. First set timer to zero (TIMER=0), then use an If statement to test for the value equal to the delay you want. One second is 60 ticks of the timer function.

more on page 18

Nimble Fingers

By Stewart J. Peck

This program is an educational exercise which can be used happily by older persons. It provides a method which is fun and active to learn the typewriter keyboard. It supplies immediate visual and audio feedback which is both stimulating and encouraging.

The slowness of hunt and peck typing can be agonizing. The Type Tutor program can help children or adults lack of knowledge of the keyboard and provide amusement while doing so.

The screen starts with a title, and there is a delay of about five seconds while initialization takes place. Then the playing screen will be created and the game will start. The first letter to fall is selected randonly to get things going. The object is to prevent the inverted letters from reaching the bottom row of letters. To prevent this, you must select a letter which will rise from the bottom, erase all the inverted letters in its column, and play a scale as it rises. At the end of the game, your score and time is shown and there is a prompt to play again. Your score is accumulated by the number of inverted letters you erase. Say there are four inverted R's in the R column and you select R on the keyboard. An R will rise and erase the four inverted R's and you will receive 40 points. If any key is selected besides an alphabetic character, no response will be made (the Break key should never be selected).

The program consists of two arrays; one is loaded with the notes to be played for each letter as it rises on the screen, and the other tracks the number of inverted letters in each letter's column. The rest of the program acts upon values determined from the ASCII value of the letter selected from the keyboard. See the Table for a line by line description of the program's execution.

As an alternative to playing scales as the letters rise to the top of the screen, you might try substituting a series of 15 notes which make a short tune. If you have a problem getting the data statements in correctly, the following should help. Insert the line: 95 CLS(0):FORX=1TO26STEP2:Y=(X-1)*16:T2=(X-1)*16+16:PRINT@T,P$(X,1);:PRINT@T2,P$(X+1);:NEXTX:END This should put two identical columns on the screen of the beginning notes of each letter. Be sure to delete this line when ready to run the game. *(end)*

Editor's Note: This game is so good it kept 150/wpm Dennis mesmerized for hours!

Table. Line Run-down

Lines 70-120	Initializes values, arrays and creates game screen.
Line 130	Randomly selects the first letter and sets up the variables to play it.
Line 140	Keyboard selection of letter; sets up the letter's scale to play and its starting print position.
Lines 150-160	Increments timer cycles if timer has started over.
Lines 170-280	Prints the rising letters (followed by erasing of previous letter) and plays the scale. Also, randonly selects the inverted letters, prints it in the next available position in the column, and increments its row value. Sets flag if inverted letter is as bottom row.
Lines 290-300	Displays points awarded and time of game. Also, prompts to start a new game or to end.
Lines 400-910	Data statements for the scales played.
Lines 920-1090	Remarks on variables used in the program.

Program. Type Tutor

```
10 '    ****  TYPE TUTOR  ****
20 '
30 '      SEE END OF LISTING
40 '        FOR REMARKS ON THE
50 '        VARIABLES USED.
60 '
70 DIMR(26),P$(26,15):B=128
80 PLAY"T32;V30;A":T=0:TIMER=0:TIME=1:POKE359,13:C
LS(0):SCREEN0,1:PRINT@492,"TYPE TUTOR";:CON=1:THOLD
=0
90 FORP1=1TO26:FORP2=1TO15:READP$(P1,P2):NEXTP2:NE
XTP1:RESTORE
100 PLAY"T32;G;B":FORX=492TO502:PRINT@X,CHR$(B);:N
EXTX:FORX=1TO26:R(X)=0:NEXTX
110 FORX=2TO483STEP32:PRINT@X," ";:PRINT@X+27," ";
:NEXTX
120 FORX=1TO26:PRINT@X+2,CHR$(X+64);:PRINT@X+482,C
HR$(X+64);:NEXTX
130 A2=RND(26):Z=A2+479:IN$=CHR$(A2+64):GOTO170
140 IN$=INKEY$:IFIN$=""THEN140ELSEZ=ASC(IN$):IFZ>6
4ANDZ<91THENA2=Z-64:Z=Z+415ELSE140
```

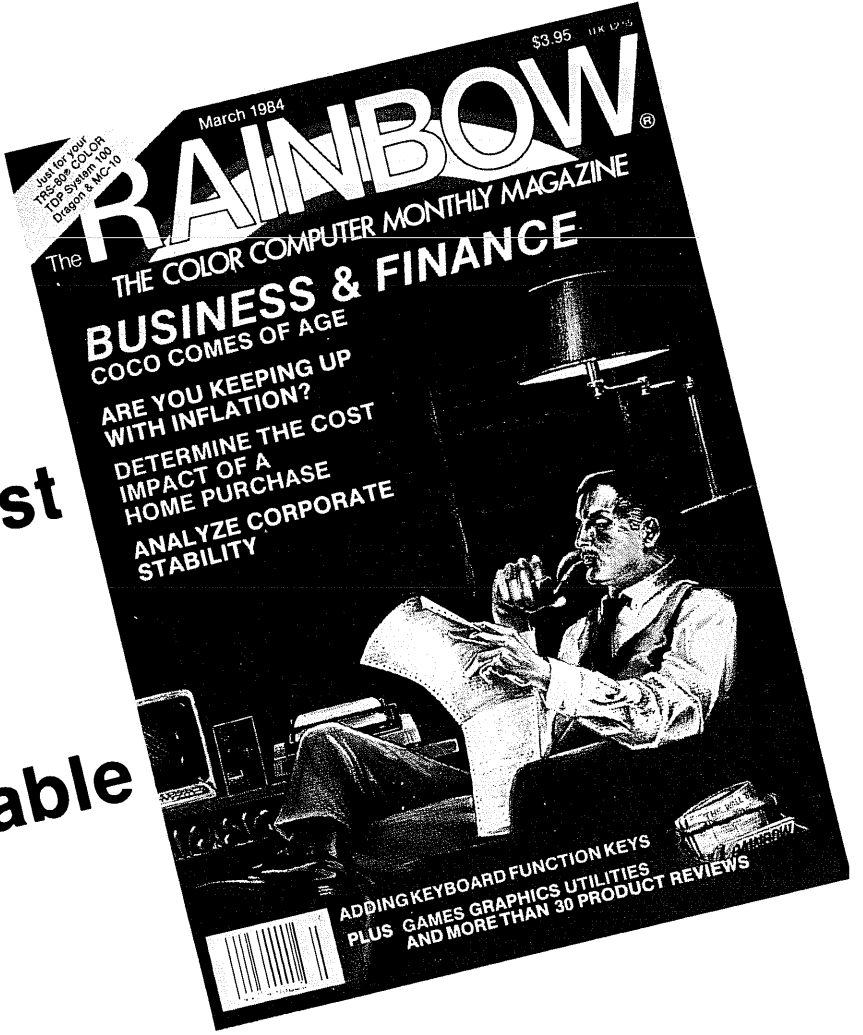
```
150 IF TIMER<THOLD THEN TIME=TIME+1
160 THOLD=TIMER
170 FORX1=1TO15
180 X=Z-(X1*32)+3
190 PRINT@X,IN$;
200 IFX1<>1THENPRINT@X+32,CHR$(B);
210 PLAY P$(A2,X1)
220 A=RND(26)
230 IF A=A2 THEN 220
240 R(A)=R(A)+1
250 IFR(A)>14THENCON=0:GOTO280
260 A1=R(A)*32+A+2
270 PRINT@A1,CHR$(A+96);
280 NEXTX1
290 T=T+R(A2):R(A2)=0
300 IF CON THEN 140
310 FORX=0TO8:CLS(X)
320 PLAY"T64;V15;O5;C;B;A;G;F;E;D;O4;C;B;A;G;F;E;D
;O3;C;B;A;G;F;E;D;O2;C"
330 NEXT X
340 CLS(4):PRINT@200,"CONGRATULATIONS!";:PRINT@232
,"YOU HAVE SCORED:";:PRINT@249,T*10/TIME;
350 T8=TIMER:T9=INT(T8/3600):T0=(T8/3600)-T9:T0=IN
T(T0*60):IFTIME>1THENT9=T9+18
360 PRINT@268,"TIME IS:";:PRINT@280,T9;":";T0;
370 PRINT@448,"TYPE N TO END GAME  ";:PRINT@480,"T
YPE Y TO PLAY AGAIN";
380 IN$=INKEY$:IFIN$="Y"THEN80ELSEIFIN$<>"N"THEN38
0
390 CLS(1):SCREEN0,0:POKE359,126:END
400 REM  C MAJOR
410 DATA T32;O1;C,D,E,F,G,A,B,O2;C,D,E,F,G,A,B,O3;
C
420 REM  C MINOR
430 DATA T32;O1;C,D,E,F,G-,A,B,O2;C,D,E,F,G-,A,B,O
3;C
440 REM  D MAJOR
450 DATA T32;O1;D,E,F#,G,A,B,O2;C#,D,E,F,G#,A,B,O3
;C#,D
460 REM  D MINOR
470 DATA T32;O1;D,E,F#,G,A-,B,O2;C#,D,E,F,G#,A-,B,
O3;C#,D
480 REM  E MAJOR
490 DATA T32;O1;E,F#,G#,A,B,O2;C#,E-,E,F#,G#,A,B,O
3;C#,D,E
500 REM  E MINOR
510 DATA T32;O1;E,F#,G#,A,B-,O2;C#,E-,E,F#,G#,A,B-
,O3;C#,E-,E
520 REM  F MAJOR
530 DATA T32;O1;F,G,A,B-,O2;C,D,E,F,G,A,B-,O3;C,D,
E,F
540 REM  F MINOR
550 DATA T32;O1;F,G,A,B-,B,O2;D,E,F,G,A,B-,B,O3;D,
E,F
560 REM  G MAJOR
570 DATA T32;O1;G,A,B,O2;C,D,E,F#,G,A,B,O3;C,D,E,F
#,G
580 REM  G MINOR
590 DATA T32;O1;G,A,B,O2;C,D-,E,F#,G,A,B,O3;C,D-,E
,F#,G
600 REM  A MAJOR
610 DATA T32;O1;A,B,O2;C#,D,E,F#,G#,A,B,O3;C#,D,E,
F#,G#,A
620 REM  A MINOR
630 DATA T32;O1;A,B,O2;C#,D,E-,F#,G#,A,B,O3;C#,D,E
-,F#,G#,A
640 REM  C MAJOR
650 DATA T32;O3;C,D,E,F,G,A,B,O4;C,D,E,F,G,A,B,O5;
C
660 REM  C MINOR
670 DATA T32;O3;C,D,E,F,G-,A,B,O4;C,D,E,F,G-,A,B,O
5;C
680 REM  D MAJOR
690 DATA T32;O3;D,E,F#,G,A,B,O4;C#,D,E,F#,G,A,B,O5
;C#,D
700 REM  D MINOR
710 DATA T32;O3;D,E,F#,G,A-,B,O4;C#,D,E,F#,G,A-,B,
O5;C#,D
720 REM  E MAJOR
730 DATA T32;O3;E,F#,G#,A,B,O4;C#,E-,E,F#,G#,A,B,O
5;C#,E-,E
740 REM  E MINOR
750 DATA T32;O3;E,F#,G#,A,B-,O4;C#,E-,E,F#,G#,A,B-
,O5;C#,E-,E
760 REM  F MAJOR
770 DATA T32;O3;F,G,A,B-,O4;C,D,E,F,G,A,B-,O5;C,D,
E,F
780 REM  F MINOR
790 DATA T32;O3;F,G,A,B-,B,O4;D,E,F,G,A,B-,B,O5;D,
E,F
800 REM  G MAJOR
810 DATA T32;O3;G,A,B,O4;C,D,E,F#,G,A,B,O5;C,D,E,F
#,G
820 REM  G MINOR
830 DATA T32;O3;G,A,B,O4;C,D-,E,F#,G,A,B,O5;C,D-,E
,F#,G
840 REM  A MAJOR
850 DATA T32;O3;A,B,O4;C#,D,E,F#,G#,A,B,O5;C#,D,E,
F#,G#,A
860 REM  A MINOR
870 DATA T32;O3;A,B,O4;C#,D,E-,F#,G#,A,B,O5;C#,D,E
,F#,G#,A
880 REM  B MAJOR
890 DATA T32;O3;B,O4;C#,D#,E,F#,G#,A#,B,O5;C#,D#,E
,F#,G#,A#,B
900 REM  B MINOR
910 DATA T32;O3;B,O4;C#,D#,E,F,G#,A#,B,O5;C#,D#,E,
F,G#,A#,B
920 '
930 '   * VARIABLE DEFINITION *
940 '
950 '   A           :RANDOM GENERATION OF INVERTED
LETTERS
960 '   A1          :POINTER TO PRINT POSITION OF I
NVERTED LETTERS
970 '   A2          :POINTER TO LETTER'S SCALE TO P
LAY
980 '   B           :BLANK SCREEN CHARACTER
990 '   CON         :CONTINUE FLAG
1000 '   IN$        :INKEYS$ VALUE
1010 '   R(26)      :INVERTED LETTERS' ROW VALUE
1020 '   P$(26,15) :MUSICAL NOTES FOR EACH LETTERS
' POSITION
1030 '   T          :ACCUMULATOR OF POINTS DURING G
AME
1040 '   T8         :TIMER AT END OF GAME
1050 '   T9         :TIMER IN MINUTES
1060 '   T0         :TIMER IN SECONDS
1070 '   THOLD      :TIMER AS OF PREVIOUS PASS
1080 '   TIME       :# OF TIMES THE TIMER CYCLED TH
RU 64K
1090 '   Z          :POINTS TO STARTING POSITION TO
 PRINT LETTER SELECTED
1100 REM
1110 REM  ***  GOOD LUCK  ***

1060 '   T0         :TIMER IN SECONDS
1070 '   THOLD      :TIMER AS OF PREVIOUS PASS
1080 '   TIME       :#
```

Editor's note: Disk users—delete the POKE and Screen command in line 80.

The Biggest

The Best

The Indispensable



The latest news on the Color Computer grapevine is that more and more people are discovering **The Rainbow.**®

Now in its third year, **The Rainbow** has become the standard by which other Color Computer magazines are compared. And no wonder! **The Rainbow** towers above the crowd, now offering up to 300 pages each month, including as many as two dozen type-in-and-run program listings, a host of articles and in excess of 30 hardware and software product reviews.

We lead the pack in Color Computer publications and are devoted *exclusively* to the TRS-80® Color, TDP-100 and Dragon-32. We made our climb to the top by continually offering the best and the most by such well-known authors and innovators as Bob Albrecht and Don Inman, and games from top programmers like Robert Tyson, Fred Scerbo and John Fraysse. **The Rainbow** offers the most in entertainment and education, home uses, technical details and hardware projects, tutorials, utilities, graphics and special features like Rainbow Scoreboard and our CoCo Clubs section.

For only $28* a year, you get the keys to all the secrets locked in your CoCo!

Are you searching through the jungle of claims and clamor? Climb above it all. Look up. Find **The Rainbow.**

THE RAINBOW
(502) 228-4492

9529 U.S. Highway 42
The Falsoft Bldg.
P.O. Box 385
Prospect, KY 40059

THE RAINBOW is a publication of Falsoft, Inc.

YES! Sign me up for a year (12 issues) of **THE RAINBOW.**

Name _____

Address _____

City _____ State _____ ZIP _____

Payment Enclosed ☐

Charge Visa ☐ MasterCard ☐ American Express ☐

My Account# _____

Signature _____ Card Expiration Date _____

In order to hold down non-editorial costs, we do not bill for subscriptions.

*Subscriptions to **the RAINBOW** are $28 ($31 after Jan. 1, 1985) a year in the United States. Canadian rate $35 U.S. ($38 after Jan. 1, 1985). Surface rate to other countries $65 U.S. ($68 after Jan. 1, 1985); air rate $100 U.S. ($103 after Jan. 1, 1985). All subscriptions begin with the current issue. Please allow up to 5-6 weeks for first copy. U.S. FUNDS only.

Prices subject to change.

In 1965 I returned from a stay in Europe aboard a military flight. The customs official was a comedian. "Got any drugs to share?"

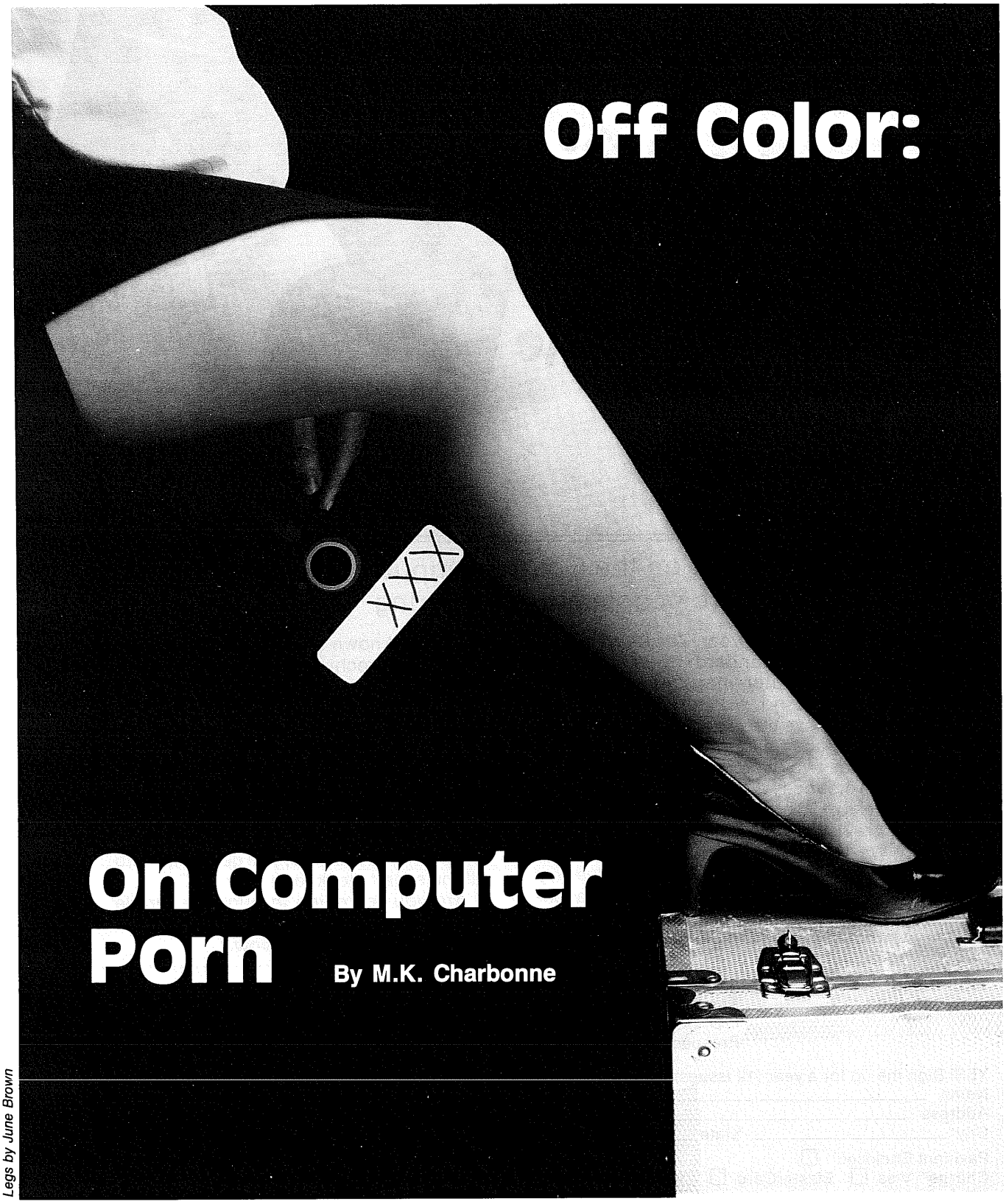
"No, sir!"

"Got any French funnybooks?"

"Sir?"

"You know: pornography."

"Oh, no sir! I don't even own a pornograph!"



Off Color: On Computer Porn

By M.K. Charbonne

Legs by June Brown

It was a joke I'd read in *Reader's Digest*. I'd waited years to use it.

In those days pornography was illegal.

* * * *

These days we are not sure what pornography is, and its legality depends on where we live and what our neighbors think.

I'd define pornography as boring, puerile representation of human sexuality. This excludes the erotic.

That won't stand up in court because I take a limited view.

* * * *

I had a contract to produce a children's computer book and needed the programs tested by kids. A 12-year-old boy, the son of a friend, spent an hour trying my programs with some interest, then asked to try *Eliza*, the tongue-in-cheek non-directive analyst program which purports to deal with emotional problems. I went away as he began. I returned when I heard him chuckling. He'd filled the screen with "fupooshicapi" words and was delighted by the program's responses. It seems Eliza is programmed to take the user to task for use of specific naughty words.

I was amazed! Eliza is not smut, but it responded obliquely to the use of smut. I think the reason is that Eliza's authors realized users would get rather testy when they discovered they couldn't direct the conversation into some areas. Eliza's responses to these words is by turns humorous and mildly castigating.

Babe in the woods that I was, it had not occurred to me a computer could be used for pornography. Now I know any medium can be used for any reason: religion, philosophy, pornography among them.

* * * *

A piece of commercial software appeared. It was an arcade game themed to Custer's Last Stand. It included the sexual attack of an Indian woman by a graphics character controlled by the player. I never saw the program; I read several reviews. One reviewer noted the program was made all the more sleazy because the pixilated graphics portrayed the act in a comic way.

The program was hooted out of existence. The publisher agreed to withdraw it.

* * * *

Other risque programs have appeared. Some pridefully advertise they were by their nature excluded from this or that software catalog. The reviews I've read with interest portray these programs as puerile without exception.

I'm led back to the image of the 12-year-old boy typing naughty words for Eliza.

* * * *

Soon after my first book was published, I received a tape from a chronological adult reader. In the book I'd written a program which combined different categories of words held in several arrays to generate playful titles for science fiction books. The reader had expanded and revised the program so it spewed vaguely disparate sentences describing the coupling of a male and female of some species or another. He wanted to know if I'd be interested in going in halves on promoting the program as commercial software.

I ran the program with some fascination. It made sense! Did I write that?

Then I realized that if computers can spawn raw pornography from very, very raw data, the problem is not in computers but in the quality of pornography. True sexual pornography is mechanical and killing to the spirit of love. Sometimes, eroticism—no matter how we perceive it—is not.

I wrote back to the chronological adult in a kindly way, disassociating myself from his efforts and relinquishing vestigial rights.

* * * *

I've read of bulletin boards with sexual content. Penthouse-type letters. Opportunities for strange types to exchange fantasies—"I do this to you (pant)." "And in return I do this to you (gurgle)."

* * * *

My conclusion:

What is advertised as adults-only material most often is nothing more than the tittering stuff of pre-adolescence. I do not begrudge pre-adolescents their approach, because I've been there and know we all can progress into warm, one-to-one relationships that don't depend on unlikely fantasies.

Much of what passes for the shockingly new is repetitive, puerile, vile chaff to us old gaffers who have passed age 40.

I'll take a hand-holding routine any day. *(end)*

Fm: Dennis Bathory Kitsz (to Marty)

I've been reviewing the discussion, and it seems to diverge when it reaches the question of, so to speak, programming ethics or morality, or responsibility. Who is responsible, and for what? If, Marty, every programmer has to re-invent the wheel, that is, write every disk I/O routine because they haven't been documented, or provide all kinds of shuffles and soft-shoes, then where does the end user benefit? From (1) pure friendliness and hassle-free use, or (2) from the lower price of a product that doesn't require the additional development time? And, furthermore, if the blame lies with the original authors (Tandy & Microsoft), then aren't you just reaffirming that commercial products cut corners and, like the hardware modifications we perform, are necessary to produce the product we are looking for? Is Tandy to blame for the need for outboard or software lowercase, for example? Could they sell the computer to Everyperson for $99 that way? Does "professional" operating system structure cost more too? In development time? Marketing strategy? Would the reason perhaps be the market pressure—for example, from Exatron, who may have been trying to set a standard—to get that DOS on the market, *now*! Is a computer going from $599 to $499 to $399 to $299 to $199 to $99 expected to provide this kind of environment? Is this an unwritten law of the programming world (you tell us 95 percent of machine language programmers would agree with you. How do you know? Are all the clowns working for Tandy and Microsoft, or do they know something you don't about the issues involved here?) Finally, is it laziness or economy? Ultimately, who is responsible? Could it be the end user, and neither Tandy nor Microsoft nor the third party software vendors?

Fm: A. Flexser (to D. Kitsz)

As far as I can see, the cost to Tandy/Microsoft of putting a good set of vectors into the Disk ROM would have been *extremely* minimal, when spread out over the number of units we are talking about. The main cost would have been in having to put out a slightly longer technical section of the Disk Basic manual to document the entry points. As to the responsibility of software vendors to the buying public, I think the responsibility is fulfilled if the product runs on all ROM versions available at the time of purchase, and if there is some reasonable upgrade policy for people who bought a second CoCo with newer ROMs, burned out a ROM and had to replace it, etc.—a tiny minority. I suspect that the group most impacted by incompatibilities due to ROM changes are those individuals who buy new CoCo's and then acquire older software by, shall we say, "unofficial" means.

With regard to cutting corners, I think a little perspective is necessary. In examining CoCo Basic closely, and comparing it to the Basics available on other comparable machines, one comes to realize that the folks at Microsoft who programmed the ROMs had a very different outlook on the CoCo than the one that shows in Tandy's "everyman" (more like "everykid") marketing strategy for the machine. Little Johnny's Christmas present does not need functions like HEX$, VARPTR, DSKI$, etc. CoCo DBasic was clearly designed to meet the needs of a fairly sophisticated user. ML utility programmers were well-served in the structure of low RAM with its ample set of hooks. Careful planning shows throughout. *Except* in the failure to provide sufficient ROM vectors, which sticks out like a sore thumb precisely because of the general excellence of the ROMs taken as a whole. Commodore 64 Basic, which is a wretched piece of trash compared to the CoCo's, boasts around 35 documented ROM vectors, versus a mere handful in the CoCo. I guess what I'm saying is, if they had to cut corners, stinting on documented entry points was a very ill-

considered place to do it, and entirely out of keeping with the general philosophy that appears to have characterized the design of the ROMs. The effort it would have taken to put them in is negligible compared to the effort spent on any number of vastly less valuable features of the ROM that *were* included. (The virtually useless DLOAD(M) command is just one of many possible examples.)

(A reply chain of two follows immediately:)

Fm: Marty (to A. Flexser)

Very eloquently put, Art. You really *should* try getting some of your discussions published.

and:

Fm: A. Flexser (to Marty)

Thanks. You might find 6139 *(Eds. Note: refers to a later message.)* of interest on this topic . . . perhaps you've infected me with a touch of paranoid ravings tendency, hee hee! Maybe I'd have made a good conspiracy theorist, back when those were in high demand!

Fm: Marty (to D. Kitsz)

My position was intentionally a very narrowly limited one: that Tandy and Microsoft screwed up royally, and are to be condemned for not providing disk file I/O vectors in disk Basic. And that they should at the next ROM revision rectify that mistake.

There are dozens of other vectors I'd like to see, sure. I'd have liked a *true* operating system, chock-full of vectors.

But I *do* understand that Microsoft set out to write a Basic that supports disk I/O, not a general-purpose operating system. Nevertheless, of all the vectors that are absent from R/S DOS, I maintain that the absence of disk file I/O vectors has caused vastly more trouble for would-be CoCo machine language program developers, and for the CoCo software market, and for CoCo users, than any other single vector that was "left out."

That one oversight, I argue, was disastrous! Peter and Michael agree fully with me. Steve Bjork agrees with me. And if you weren't being such a devil's advocate and trying to goad me to write more so you can print it in *Under Color*, you'd agree with me on this issue too!

I'm focusing on this one goof both because it is so serious, and because it would still be a relatively simple matter for them to correct that goof! Thus, I'm raising such a fuss about it not just because I love to release hot air (though it's true I *do* love to release hot air), but because on this matter I believe it still is within Tandy's power to correct it.

If, as John Roach was quoted as saying, Tandy's policy is going to be "CoCos forever," then if they corrected this single major flaw there still would be time to re-attract machine language programmers to support the machine—programmers who now don't want to bother to spend months writing stuff that will be made un-runnable at the whim of a new ROM revision.

Yes, such potential programmers *could* get around the problem by writing their own DOS.

But, as you know, that is a solution that is costly in time and raises the cost of the software to the public. And costly software for a $100 machine won't sell well!

Although I enjoy a good fight with anyone, especially so worthy an opponent as my old and dear friend Wayne Day, I honestly hope that if enough noise is made about this, it may result in something constructive.

Wayne sneered something to the effect that no matter how loud he or I squawked, the folks in Tandy Towers would ignore it. I disagree. I hear that the Deluxe CoCo was going to have file handling hooks in its software.

So Tandy *may* be on the verge of correcting this mistake.

Fm: A. Flexser (to D. Kitsz)

After leaving my last message, a possibility came to mind that previously hadn't occurred to me. Following up on my statement that failure to put in a good set of ROM vectors was uncharacteristic of the general caliber of the Microsoft CoCo ROMs, the possibility dawned on me that perhaps this omission was not an oversight or corner-cutting measure, or due to general laziness, but was rather a considered and deliberate decision. Why? Perhaps as a sort of advance counter-measure against anyone who might consider manufacturing a CoCo clone with its own non-Microsoft ROMs. If there were a good set of ROM vectors available, such a clone would be able to achieve a high degree of software compatibility with the CoCo. Without vectors, there'd be substantial incompatibility, making the clone much less attractive in the marketplace.

Fm: Marty (to J. Ross)

I maintain that that "warning" was laughable, coming from someone who sanctioned (or at least did nothing to correct) the failure to include proper file handling vectors.

As I wrote Wayne, I agree that some programmers were stupid and lazy (Howard Cohen comes to mind), and ignored what few meager vectors Microsoft in its infinite benevolence saw fit to give them. And such lazy programmers are indeed to be condemned. But their sins are as naught to the evil wrought by the lack of file handling vectors. And yea! Blame for that is upon the door of Tandy and Microsoft! Woe unto them, until they shall make atonement by installation of file handling vectors in disk Basic version 1.3!!

Fm: A. Flexser (to Marty)

Actually, I think you may be giving Howard Cohen a bit of a bad rap here. Howard chose to do his Disk I/O from Basic. This is inefficient and (worse) wastes vast amounts of memory space better allocated to buffer. But it *is* ROM-independent. It is only in a very few places in the TW64 code (like where it reacts to the "D" command to go to S/XXX) that there is ROM dependency. A measure of this is that the only difference between the old version of TW64 that ran under Disk 1.0 only, and the new one, that runs under both versions, is a single line containing a half dozen or so POKEs added to U/BAS.

Fm: Marty (to J. Ross)

As I said to Dennis, *if* John Roach and Tandy are serious about that comment regarding a policy of "CoCos forever," then it still may not be too late for them to correct that error. Very late? Yes. But *too* late? No, not if they plan to make CoCos available for a few more years.

Fm: John Ross (to Marty)

True, but I doubt very seriously whether you will ever see it.

Fm: Sandy Trevor (to Marty)

I'm not taking sides in this debate, mind you, but I have spent more than a few minutes cursing the decision to include only one pathetic vector in the disk ROM. Even took up the issue in Tandy Towers more than once. But, what's burned in silicon might as well be etched in stone. So, Jim Davenport and I worked out what we thought was the

best compromise solution under the circumstances. As Art said, writing your own file service has its own hazards too. In the case of Vidtex, I'm happy with the solution we used. If Tandy comes out with a new ROM, we can have a new Vidtex ready at the same time. And anyone with an old ROM who wants to upgrade can get the Vidtex that supports the new ROM without having to purchase a new copy.

Yes, there is no doubt that additional vectors in the disk ROM would have been a much more far-sighted and technically correct solution. But, that's show biz.

Fm: Marty (to S. Trevor)

Can't argue with that statement. A very balanced response, I'd say.

Fm: Roger Bouchard (to A. Flexser)

Disk command entry points are in the disk table of pointers, which has a pointer in RAM always at the same location. What I am saying is this. Get the table pointer. Calculate offset in table, then JSR to it as Basic would. Or better, since from version to version the way the parameters from a given command are taken from Basic the same way, see what offset has to be added to skip to the part of the command and set the parameters yourself. That's the way to, for example, print to a given device code.

Fm: A. Flexser (to R. Bouchard)

I am not nearly so certain as you that the offest one might want to add to the address of the start of a Basic command routine would remain constant from one ROM version to the next. Even if it is constant between Disk Basic 1.0 and 1.1, there is no guarantee that that won't change in the next version.

Fm: Marty (to everyone)

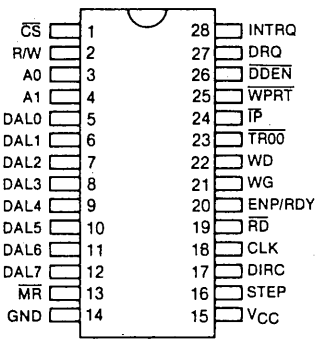
Over the last few days there has been a quite heated discussion of the "ethical" and "moral" aspects of who is to blame for certain technical limitations of the operating software supplied by Microsoft for Tandy's Color Computer. As is my style, I've argued my position with brutal frontal attacks on those who argued the other side. In this case, it happened that Wayne Day found himself as chief spokesman on the Sig for a position opposed to mine, and as such became the target of a number of my attacks. To set the record straight for newcomers to the Sig, I want to *emphasize* that I consider Wayne an old and dear friend. He is one of the more generally knowledgeable and articulate folks here, and has over the last few years made this Sig into the stunning success it is by his outstanding administration of it. Newcomers should understand that my salvos at Wayne in the current debate do not in *any* way at all reflect any personal negative feelings toward Mr. Day . . . but rather are a rhetorical device I frequently employ around this forum. Although I'm sure Wayne is not happy with my bitter attack on his position . . . or the fact that that attack got a little ad hominum at times . . . he knows me very well, and I'm sure realizes that any opposition I have toward him is limited to the *very* narrow confines of the academic issues of this debate over ROM calls in the CoCo's Basic Interpreter. *(end)*

****
*** NEWS FROM UNDER COLOR **** NEWS FOR UNDER THE COLORS ***
********** Compiled April 9, 1985 **********
** Send news to: Under Color, Box 6809, Roxbury, VT 05669 **
********** CompuServe 70136,1257 **********
****

Interesting DISK CONTROLLER news: Western Digital's 1773 controller is the heart of several new disk packs, including Tandy's. Under Color recently reported PROBLEMS WITH MIXED DENSITY MODES, a technique often used for software protection. However, one user group reports THOROUGH TESTING OF RECENT TANDY CONTROLLERS. Their discovery? EVERYTHING WORKS in the recent controllers. For your information, we reprint some of the WD1773 controller data here.

FEATURES

- 100% SOFTWARE COMPATIBILITY WITH WD1793
- BUILT-IN DATA SEPARATOR
- BUILT-IN WRITE PRECOMPENSATION
- SINGLE (FM) AND DOUBLE (MFM) DENSITY
- 28 PIN DIP, SINGLE +5V SUPPLY
- TTL COMPATIBLE INPUTS/OUTPUTS
- 128, 256, 512 OR 1024 SECTOR LENGTHS
- 8-BIT BI-DIRECTIONAL HOST INTERFACE

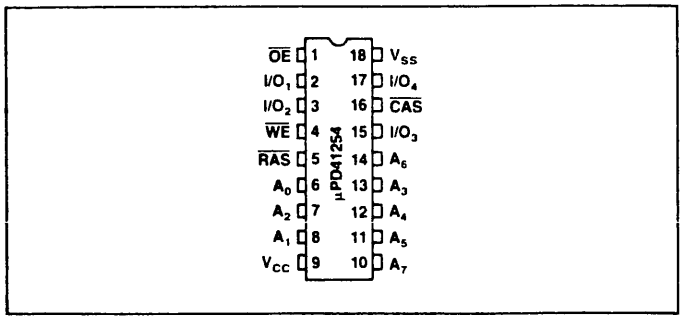


PIN DESIGNATION

* * * * *

The new Korean Color Computer (the NEW new one, that is ... 26-3134A/B, 3136A/B) is filled with differences. Under Color already told you about the lowercase 6847-T1 appearing in them; take heed that THIS LOWERCASE MUST BE TURNED ON BY SOFTWARE (details soon). There is also ONLY ONE ROM SOCKET. For Color Basic, this is the usual 24-pin 68364 Motorola type; but for Extended Basic, it is a 27128-compatible ROM in a 28-pin package. To upgrade, you need the new ROM and must MOVE FIVE JUMPERS FROM "64K" TO "128K". This computer is also the beast with JUST TWO MEMORY CHIPS ... for your information, we reprint the uPD41254 memory data here:

Pin Configuration



Pin Identification

Pin No.	Symbol	Function
1	$\overline{OE}$	Output Enable
2, 3, 15, 17	I/O_1–I/O_4	Data Input/output
4	$\overline{WE}$	Write Enable
5	$\overline{RAS}$	Row Address Strobe
6–8, 10–14	A_0–A_7	Address Inputs
9	V_{CC}	+5V Power Supply
16	$\overline{CAS}$	Column Address Strobe
18	V_{SS}	Ground

* * * * *

Errors & Omissions

Michael J. Lill, Box 6809, UnderColor, Issue 6

```
SPO 256 Speech Processor Test


10 REM MICHAEL J. LILL SEPT84
20 CLS:PRINT"1 = SPO256-117  CLOCK TEST
30 PRINT"2 = SPO256-AL2  ALLOPHONE TEST
40 PRINT"3 = SPO256-AL2  SPEAK SAMPLE
42 PRINT"4 = SPO256-AL2 RANDOM SPEEK
45 PRINT:PRINT"   PRESS SPACE BAR TO STOP
50 A$=INKEY$:IFA$=""THEN50ELSEA=VAL(A$)
60 PRINTA
70 ONA GOSUB120,250,380,2000
80 RESTORE
90 PRINT#-2,CHR$(0);
100 PRINT#-2,CHR$(64);
110 CLS0:SOUND100,1:RUN
120 REM SPO256-017 CLOCK TEST (SPR016-117)
130 FORA=0TO35
140 IFINKEY$<>""THENRETURN
150 FORB=1TO500:NEXT:'DELAY
160 PRINT#-2,CHR$(A);:'PUT DATA ON ADDRESS LINES
170 PRINT@7,A
180 C=A+64:'GET STROBE READY
190 PRINT#-2,CHR$(C);:'SEND STROBE
200 IFINKEY$<>""THENRETURN
210 FORB=1TO500:NEXT:'DELAY
220 NEXT
230 GOTO120
240 REM SPO256-AL2 TEST
250 REM ALLOPHONE TEST
260 FORA=5TO63
270 FORB=1TO500:NEXT
280 PRINT@39,A
290 PRINT#-2,CHR$(A);
300 C=A+64
310 PRINT#-2,CHR$(C);
320 FORB=1TO500:NEXT
330 PRINT#-2,CHR$(0);
340 PRINT#-2,CHR$(64+128);
350 IFINKEY$<>""THENRETURN
360 NEXT
370 GOTO250
380 REM THIS IS YOUR COMPUTER TALKING
390 RESTORE
400 PRINT#-2,CHR$(0);:PRINT#-2,CHR$(64);
410 IFINKEY$<>""THENRETURN
420 '
430 READA
440 PRINT@72,A
450 PRINT#-2,CHR$(A);:PRINT#-2,CHR$(A+64);
455 PRINT#-2,CHR$(A+128);
460 IFA=0THEN380
470 GOTO410
480 DATA27,7,45,53,4,3,18,12,55,2,12,43,2,18,15,2,42,15,
16,9,49,22,13,51,2,13,24,42,19,44,4,6,2,57,53,9,2,13,31,
63,19,2,20,49,2,27,7,7,7,45,9,2,4
490 DATA15,2,63,32,13,3,40,6,35,2
500 DATA25,47,43,2,15,1,61,53,2
510 DATA16,6,3,24,2,9,51,1,20,49,13,53,52,2
520 DATA63,19,1,42,26,20,49,16,2
525 REM VERY MUCH
530 DATA35,47,19,2,16,15,2,50,2
535 REM INTERESTED
540 DATA12,11,13,39,7,55,1,13,1,13,7,21,3
        545 REM IN SPEECH
550 DATA35,47,2,55,55,2,19,1,50,2
555 REM AND ROBOTS
560 DATA26,26,11,21,2,14,53,1,63,24,24,13,43,4,4,4,4
1999 DATA0
2000 REM SPO256-AL2 RANDOM SOUND
2010 A=RND(63)
2020 PRINT@103,A
2030 PRINT#-2,CHR$(A);
2040 PRINT#-2,CHR$(A+64+128);
2050 IFINKEY$<>""THENRETURN
2060 GOTO2000
```

more Hints, Tips & Tricks

Programming—When using data statements, if one of the numerical values is 0 you can just leave the zero out. Because nothing is typed between the commas, the computer will assume a value of zero; i.e., DATA 65,0,32 will give the same results as DATA 65,,32, and save one byte of RAM, too.

General—Save your eyes: when doing a lot of keyboard input turn down the color control until you have only black and white. It's much easier on the eyes.

Programming—If you want to save bytes, but still have comments in your program, you can eliminate the keyword REM in some cases, such as after GOTO. Since program control is changed to another line, the comment is never noticed by your computer; i.e., 10 GOTO 100:start initialization won't cause any problems, but 10 GOSUB 100:start will cause an SN error as the computer tries to decipher the word "start."

Programming—A frequent mistake of novice programmers is to use a stack of If statements for program control. In many cases this can be replaced by a single On . . .GOTO or On . . .GOSUB statement. For the sake of speed, the most common option should be placed first.

Programming—If you want to use letters as the menu control medium (B for beginning, E for end, etc., instead of 1. Beginning, 2. End, etc.) you can still use an On . . . GOTO or On . . . GOSUB if you use the INSTR command in the statement; i.e., ON (INSTR("bBeEcCdDfFrR//",X$)/2 + 1) GOTO 100, 200, 300, 400, 500, 600, 700, where X$ is the keyboard input.

Programming—If you want to use INKEY$ as an input to drive a GOTO or GOSUB sequence, try this: 100 ON ((INSTR$(" bBeE//",INKEY$)/2 + 1) GOSUB 100,200,300,400. With no keyboard input, the blank at the start of the string sequence forces line 100 to cycle continuously, pressing either "b" or "B" sends the program to line 200, and so forth.

One For the MC-10:

Printer—For faster printer bauds use POKE 16932, x; where x is 255 for 100 baud, 59 for 300 baud, 27 for 2400 baud, 11 for 4800 baud, and 2 for 9600 baud. *(end)*

Transfer

by Dale Keller

After writing several asembly language programs on a CCEAD assembler on cassette I upgraded my system with 64K, a disk drive, and a disk EDTASM assembler. Not wanting to retype my CCEAD programs into EDTASM, I wrote this program to copy the source files.

Since the two files have slightly different syntax, the program must also change a few things as it transfers data. The CCEAD files use a left bracket ([ ) at the beginning of each line which must be dropped. The EDTASM files must have line numbers, and a tab character can be added in many places.

The transfer program reads in a line from the CCEAD file on tape, drops the leading bracket, breaks the line into separate fields, inserts tabs, inserts line numbers, and then writes the data to a disk file. The disk file should then be loadable by EDTASM. The source code will not be perfect if all the CCEAD columns weren't even. Also, some mnemonics are different—ANDC in CCEAD will need to be changed to ANDCC for EDTASM to assemble it properly. The easiest way to find the differences is to assemble the file with the / WE (wait on error) switch.

Line 240 may need to be tailored to read your files properly. Keep the first statement (this drops the leading bracket) and then change the MID$ statements to break up the input line properly. W1$ is the symbol, W2$ is the op code, W3$ is the operand, and W4$ is the comment. See lines 30–0 for more information. *(end)*

Program. Transfer.

```
10 'TRANSFER
20 'CCEAD DATA TRANSFER FROM CASSETTE TO DISK EDTA
SM+
30 'FOR CCEAD PROGRAMS WHICH HAVE A 5 CHARACTER FI
ELD FOR SYMBOLS,
40 'A SPACE, A 4 CHARACTER OPCODE FIELD,
50 'A SPACE, A 6 CHARACTER OPERAND FIELD,
60 'A SPACE, A 14 CHARACTER COMMENT FIELD.
70 'LINES STARTING WITH '*' ARE COPIED INTACT.
80 'CLEAR ENOUGH MEMORY & STRING SPACE
90 PCLEAR 1
100 CLEAR 1000
110 'PROMPT TO BEGIN
120 CLS: INPUT "READY TO READ FROM CASSETTE     &
WRITE TO DISK";A$
130 'SET STARTING LINE NUMBER FOR OUTPUT FILE
140 I=10
150 'OPEN FILES
160 OPEN "I",-1,""
170 OPEN "O",1,"TFRFILE.ASM"
180 'GET A STATEMENT FROM CASSETTE FILE
190 INPUT #-1,IN$
200 TP$=""
210 'CHECK FOR END-OF-FILE
220 IF EOF(-1)<>0 THEN 500
230 'BREAK INPUT STRING INTO SEPERATE FIELDS
240 QQ$=LEFT$(IN$,1):W1$=MID$(IN$,2,5):W2$=MID$(IN
$,8,4):W3$=MID$(IN$,13,6):W4$=MID$(IN$,20,14)
250 'IF FULL COMMENT LINE, KEEP IT ALL
260 IF LEFT$(W1$,1)="*" THEN TP$=RIGHT$(IN$,LEN(IN
$)-1):GOTO340
270 'IF SYMBOL PRESENT, KEEP SYMBOL
280 IF LEFT$(W1$,1)>" " THEN TP$=TP$+W1$
290 'INSERT TAB
300 TP$=TP$+CHR$(9)
310 'ASSEMBLE REMAINING FIELDS WITH TABS
320 TP$=TP$+W2$+CHR$(9)+W3$+CHR$(9)+W4$
330 'BUILD LINE NUMBER
340 IF I<10 THEN SP=4 ELSE IF I>9 AND I<100 THEN S
P=3 ELSE IF I>99 AND I < 1000 THEN SP=2 ELSE IF I>9
99 AND I<10000 THEN SP=1 ELSE IF I>9999 THEN SP=0
350 IF SP=4 THEN LN$="0000"+RIGHT$(STR$(I),1)
360 IF SP=3 THEN LN$="000"+RIGHT$(STR$(I),2)
370 IF SP=2 THEN LN$="00"+RIGHT$(STR$(I),3)
380 IF SP=1 THEN LN$="0"+RIGHT$(STR$(I),4)
390 IF SP=0 THEN LN$=RIGHT$(STR$(I),5)
400 'BUILD OUTPUT STRING FROM ASSEMBLED FIELDS & L
INE NUMBER
410 OUT$=LN$+" "+TP$
420 'WRITE IT TO DISK & SCREEN
430 PRINT OUT$
440 PRINT #1, OUT$
450 'INCREMENT LINE NUMBER
460 I=I+10
470 'DO NEXT STATEMENT
480 GOTO 190
490 'CLOSE FILES AND END
500 CLOSE #-1
510 CLOSE #1
520 END
FROM ASSEMBLED FIELDS & LINE NUMBER
410 OUT$=LN$+" "+TP$
420 'WRITE IT TO DISK & SCREEN
430 PRINT OUT$
440 PRINT #1, OUT$
450 'INCREMENT LINE NUMBER
460 I=I+10
470 'DO NEXT STATEMENT
480 GOTO 190
490 'CLOSE FILES AND END
500 CLOSE #-
```

Bug Fix!

By Don Hutchinson

I recently became aware of a flaw in Version 1.0 of Disk Basic. The DSKINI routine contains an obscure bug which could cause problems with data written to disks that are formatted in succession. (If you're like me, you probably format the entire box when you break the cellophane.) A Radio Shack addenda sheet (furnished with all new drive 0's) advises one to hit the Reset button between DSKINI's, but that method doesn't solve the problem at all.

The problem is that write precompensation may remain enabled following the DSKINI, since the track pointer at location &HEC still contains a track number greater than $16. A successive DSKINI operation at this point will result in the low-numbered tracks (less than $16) being formatted with write precompensation still enabled, and this may make them difficult to read at a later time. (The tracks numbered greater than $16 should be formatted with write precompensation enabled.)

There are some who feel that write precompensation makes no difference whatsoever, whether it's always on, always off, or turned on and off as it was designed to be. However, write precompensation is necessary for 8-inch drives and was mandatory for the older model single density 5-¼ inch drives with early head designs that used wider gaps. Eight inch drives use twice the data density and are therefore much more in need of the greater readability that write precompensation provides on the inner tracks.

A "quick fix" for the bug (real or imagined) is to do a DIR after each DSKINI operation, but a better solution is to alter the Disk Basic ROM itself. Those of you with EPROM burning equipment may patch your Disk ROM as follows:

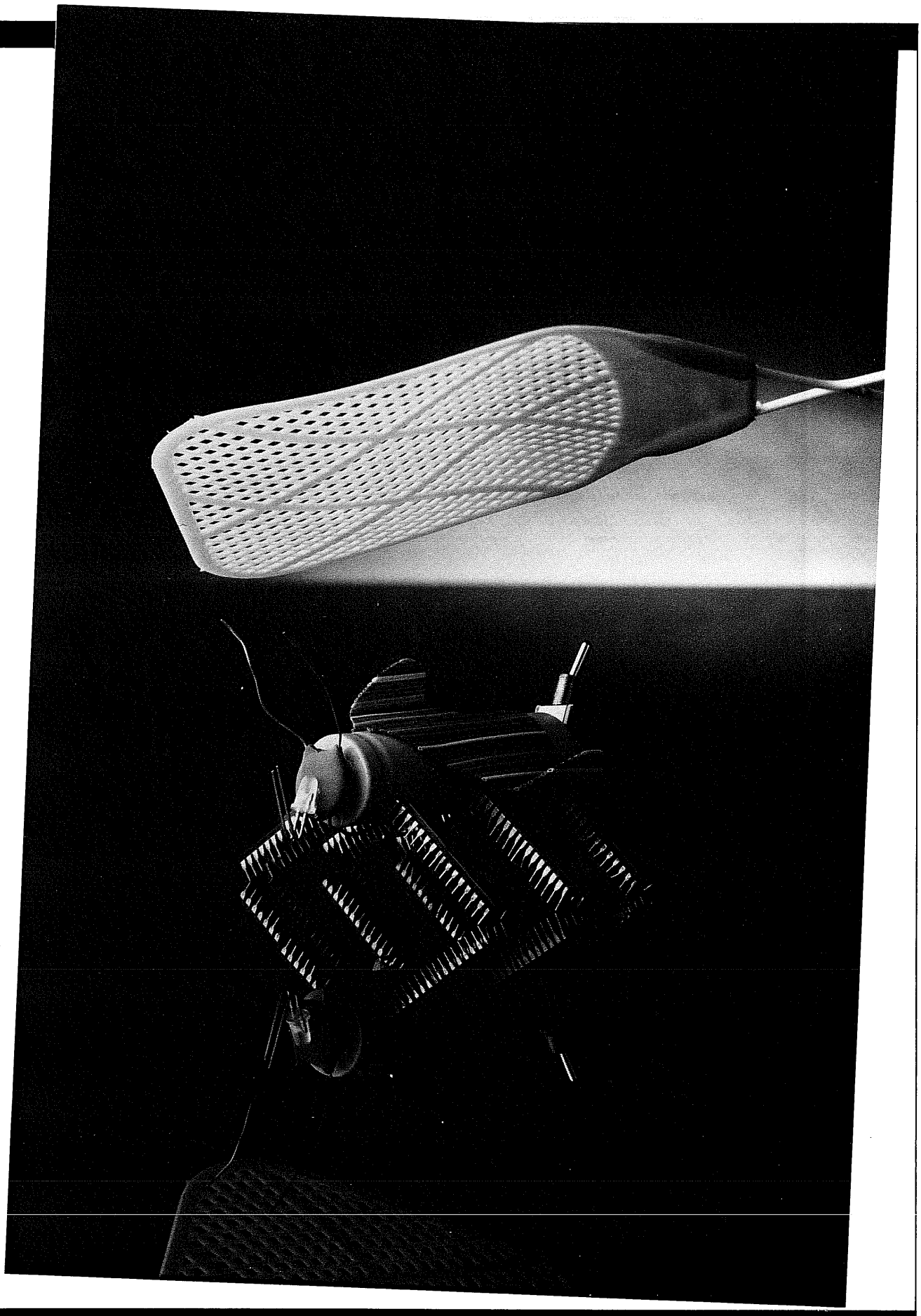
1. At location $D598 insert a JMP $#### instruction to some location above the area reserved for the Disk ROM. $D7DD and above are safe areas. (Remember that these addresses are for Disk Basic Version 1.0 only. Disk Basic Version 1.1 will operate correctly as the flaw in the DSKINI routine was corrected by Microsoft in that later version.)

2. At the chosen location insert the following code: CLR $EA Restore opcode CLR $EC. Clear the track pointer: JSR $D66C. Use DSKCON to restore the Head: JMP $D1E0.

Normal exit from DSKINI.

This code will restore the head of your drive to track zero following each DSKINI operation and "cure" the bug in the ROM. Alternately, you may opt to replace the LDA –O STA $EA code at location $D4FF with the instructions CLR $EA CLR $EC. This method has an advantage in that no additional space is needed to patch the DSKINI routine for proper operation, yet the user should be aware that some game programs protection schemes or other assembly language routines that are not properly written may circumvent this method. This is relatively unlikely, however, so the preceding fix may be used successfully most of the time.

For other useful patches to the Disk Basic ROM I refer you to Marty Goodman's article in the July 1983 issue of *Hot CoCo*. This article will tell you how to patch your Disk ROM to correct it for the "head banging" problem, the "head settle" delay, and let you add the code for an automatic program boot and/or a pseudo-DOS command. *(end)*



Review—EPROM Burners

By Martin Goodman, M.D.

PRODUCT
Intronics V2.5
DSS Peripheral Corp.
516-249-388
or
Spectrum Projects
718-411-2807
$140

Color Burner
Green Mountain Micro
802-485-6117
$70

• The Intronics EPROM programmer—The Intronics programmer costs twice as much as the Green Mountain Micro product. Its features include an ultra-high quality AMP brand ZIF socket which also has a gold plated on-board firmware driver in a 2716 or 2732 EPROM; full software-switching between EPROM types (no personality modules are used) and a well layed-out circuit board. All chips are socketed. It is capable of programming by far the widest range of EPROM types: 2732, 2732A, 2764, 27128, 2516, 2532, 2564, 25128, 68764 and 68766. It can be used with Multipak-based disk systems. Instructions are provided for using it with a Y cable and a disk controller, but those instructions are hard to follow and probably will result in an unreliable system due to lack of grounding on the disk controller.

The documentation is complete and includes electronic schematic information though it is at best marginally readable. The software that drives it has a rather crude user interface, though it is essentially self-documenting. This unit gets its 25 and 21 volt programming voltages by using a DC to DC converter that takes its juice off the 5 volt line on the system bus. Thus, it needs no external battery or supply. I have used this device for about a year and am overall extremely impressed with it. I have one *very* serious criticism of it: apparently in an effort to make his item still more "flashy" in its advertisements, George Indrof rewrote the software for Ver. 2.5 using a hi-sown idea of a "fast programming algorithm." Instead of the usual 6 to 10 minutes to program an 8K by 8 device, his software takes about 15 to 30 seconds. At first glance this seems terrific. But the "algorithm" he used is *not* the one recommended by the EPROM chip makers. It's *much* too fast. Although it appears to properly program the chip, chips programmed in that fashion are at risk of forgetting their data some weeks or months later. Even if George had used the "correct" algorithm (which gives a burn of at least five times longer than George's), the system would still not by fully meeting the specs of the fast programming information given by the EPROM chip manufacturers. Those manufacturers demand that Vcc be raised to 6 volts during fast programming. There is no provision on his unit for doing this. As I program a great many EPROMs and am very concerned about whether or not they'll stay programmed, I *rewrote* the software for that programmer so that it could be made to do "dumb" programming. I also added an option of trying the proper "Intel-igent" type programming algorithm, though I seldom use that except in rapid development work, as even that does not meet specs due to the lack of a provision to raise Vcc to 6 volts. I might add that this rewrite was facilitated both by George giving me a printed copy of the source code for his driver, and by Don Hutchison using that copy to patiently disassemble and fully comment George's driver routine using the Micro Works Macro 80C (the Editor Assembler I use). If I get permission from George I'll arrange to have my modified driver software posted on Compuserve in the CoCo Sig database.

• The Green Mountain Micro Color Burner—Dennis Kitsz introduced his Color Burner at a bargain price of $70 (assembled and tested). He's also made it available as a kit for $57. This makes it an extremely inexpensive way to acquire EPROM programming capability. Like the Intronics unit, it does have gold-plated contacts and socketed IC's on a professionally laid-out and produced circuit board. Unfortunately, there the similarities end. In order to economize, Dennis's unit uses three 9V alkaline batteries in series to supply the programming voltages. These batteries are capable of programming only about a dozen or so 2764's before they need to be replaced.* This makes one's operating costs climb. Worse yet, one always risks the possibility that the batteries will fail during a programming run, resulting in a marginally programmed EPROM that might later forget its data. There is no easy way to tell this has happened at the time of programming.

The unit is made to handle different EPROM types via a combination software *and* hardware programming "personality modules." The latter you must solder up yourself. The Color Burner comes with one blank one with your unit . . . you have to buy more (about $1 each at electronic supply houses) and solder them up yourself if you want to program more than one type of EPROM. You must both select the EPROM type you want with software *and* plug in the proper EPROM personality module. If you're programming several different types of EPROMs this can get very clumsy. The ZIF socket supplied for the EPROM is adequate, though of very cheap construction. But only a regular IC socket is supplied for the personality module. If you're changing modules a lot this can be a grand pain. Green Mountain Micro supports only the 27 series (2716, 2732, 2732A, 2762 and 27128) and the Motorola 66 series of

EPROM. The TI-25 series is not supported (though the TI series are weird EPROMs, expensive, and seldom used, and made by a company famous for using cheap labor in El Salvador and the Philippines to bond its IC's).

The software supplied for this unit is in the form of a pair of Basic and ML programs on tape. This combination is particularly ill-suited to transfer to disk. For this reason setting up the Green Mountain Color Burner to work on a disk based CoCo (even with a Multipak or CoCo bus expansion port system) is quite difficult and involves significant software rewrites. The programming time is slow (about 10 minutes for an 8K by 8 EPROM) but fully meets manufacturers specs for proper reliable programming. The software has some features that result in the EPROM getting less hot during programming than it does with the Intronics unit. On the positive side the documentation that comes with the Color Burner is *outstandingly* thorough and includes a complete schematics PC board layout and a full tutorial on how the unit works. All is written in exemplary style by Dennis Kitsz the designer. Full source code is provided for the software. And Dennis Kitsz personally is available for support of his unit, while George is a bit hard to reach through the suppliers of his unit.

Recommendations: The Intronics unit is by far the superior item if one corrects its programming algorithm (if you send me a disk and $10 to cover my time, shipping, and handling I'll send you both source and object does for my modified version of George's software: 1633 Bayo Vista Ave., San Pablo, CA 94806). If you are considering programming a large variety of EPROMs or doing small production runs, the Intronics one is clearly for you. If you are a hardware hacker and very short of cash, in need of the capability to program a few EPROMs now and then, consider the Green Mountain Micro unit. You might want to provide it with an external 27 volt DC power supply if you plan to do small production runs with it. You might also want to rewrite the software to support use on disk based system. *(end)*

**Eds. Note: GMM tells us that the betteries will reliably program at least 100 EPROMs before they need to be replaced.*

Update on new EPROMs

I just encountered some new 1764 type EPROMs from Intel. The new line called 2764A uses a die that is ⅓ the size of previous 2764 chips. More significantly, they require a programming voltage of *12.5* volts, *not* the usual 21 volts of the older line. They are apparently the new generation of EPROMs. THe new 27256 also requires 12.5 volts for programming, and I assume Intel used this newer technology to remachine its old 2764 line (and probably its 27128 line as well). The new series can be ordered in *faster* versions than the old. Intel Part No. D2764A—450ns; D2764A-3—300 ns; D2764A—250ns; D2764A-2—200 ns; D2764A-1—80 ns.

One Color Computer specific application of the new chip line would be to replace the Basic ROM in a disk system. Until now while one could replace the Extended Basic and the Disk Basic ROM in all disk systems with even 450 ns EPROMs with no problem; replacing the Basic ROM with even a 250 ns 2764 would on some systems still result in occasional crashes when the disk is accessed. We here have speculated that this was related to the effect of the greater capacitance of the EPROMs (with respect to ROMs) on the function of the timing of the NMI and HALT functions.

In any case, these new EPROMs if ordered in the fast versions, may work more reliably as they are both faster and physically smaller (the latter perhaps meaning that they'll have lower capacitance on their lines.) Of course, owner, of the current generation of CoCo EPROM programmers will have to alter their EPROM programmers in order to lower the programming voltage to 12.5 volts if they wish to try programming these new hummers. *(end)*

Review—Flight Simulation

By Dennis Peterson

PRODUCT
Worlds of Flight
Tom Mix Software
4285 Bradford N.E.
Grand Rapids, MI 49506
616-364-8217
System requirements:32K
non-Extended, joysticks
$29.95 cassette, *$32.95* disk

Worlds of Flight is more than just another computer game program. You'll find it a realistic simulation of flight, whether you've got a few thousand cockpit hours or are content to do your flying from your living room armchair, Walter Mitty style.

Granted, it isn't designed to take the place of your favorite flight instructor or ground school class, but it could leave you with some "fuel for thought" as you work the pages of sample exams that lie between you and your private pilot written test.

I was definitely impressed that the worlds of flight creators approached the program as much more than a game, even referring to the user's guide as a flight manual. Whether they were serious or just inclined to some down-home leg-pulling, after paging through twenty-two pages of instructions and maps I was inclined to take them seriously, especially when I found the aircraft performance table. The

simulation had been designed around an imaginary single-seat sport trainer, with a six-hundred pound maximum gross weight, a 40 horsepower engine, and some very precise and believable performance specifications.

The simulation lets the pilot predetermine weather in any one of nine "worlds" into which he will fly. While he may take off from any world, beginners will want to start from the practice field on world 5 before exploring other worlds. Just getting off the runway is quite a trick, so the detailed, clearly-written manual becomes a mighty good copilot. Taxiing around the airfield before take-off is a fine idea, since you'll discover you can get into trouble without even being airborne.

Having defined the world and weather conditions on the first screen, your Enter button puts you in the cockpit of your joystick and keyboard-controlled aircraft. An instrument panel appears on the lower half of your screen, and a canopy window "wire graphic" runway scene on the upper half. The instrument panel includes brake, airspeed, stall, trim, flap and rate-of-climb indicators, as well as compass, altimeter, artificial horizon and others, even a stall-warning buzzer—enough to provide for the needs of a pilot flying under visual flight rules. The nicely-displayed read-outs are digital, rather than traditional dial, but we'll be seeing more and more digital on real aircraft, too.

Some very special features include instant weather information, a ground speed indicator, and an ability to view both sides without turning the airplane.

The sparse, wire graphic scenery display takes a little time to become familiar with, but I found it adequate as I floated over the wire (line drawing) countryside.

The program also has engine sound, started by pressing both joystick buttons simultaneously. RPM increases as the left joystick (throttle) is pushed forward, and also increases in a dive as it would in a real airplane.

If you're used to the throttle on your right, just move the joystick over, but the right-left motion of the rudder and nosewheel is also controlled by a corresponding motion of the left joystick, something you might not like if you're used to rudder pedals. However, since the right joystick controls roll, putting both sticks side-by-side might be a fun, although a somewhat unrealistic way to practice coordinated turns with your computer, while using only one hand.

There is a built-in delay on the controls which operate, rudder, ailerons and elevator. The manual says this copies the delay natural to a real airplane. Still, they seemed a little too sluggish, which caused me to over-control, maybe because I am used to things happening more quickly on a computer than in the pilot's seat.

A nice program option is a "radar" view of the world over which you're flying. Corresponding maps in the flight manual let you work on pilotage, the use of landmarks to keep you on course.

Another features lets the Enter key freeze the entire simulation, letting you stop long enough to get your favorite beverage or to think of a way to avoid a realistic-sounding crash.

For those of you who can't get the flying bug out of your system, this program will let the memories of Sunday afternoon's flight in the rental bird extend well into the week. While you'll never earn an air transport rating with it, it may give you a few things to think about when the canopy in front of you isn't just your TV screen. Even those of you who hate flying should try this program. You just might change your mind. *(end)*

Review—Emulator

By Jeffrey S. Parker

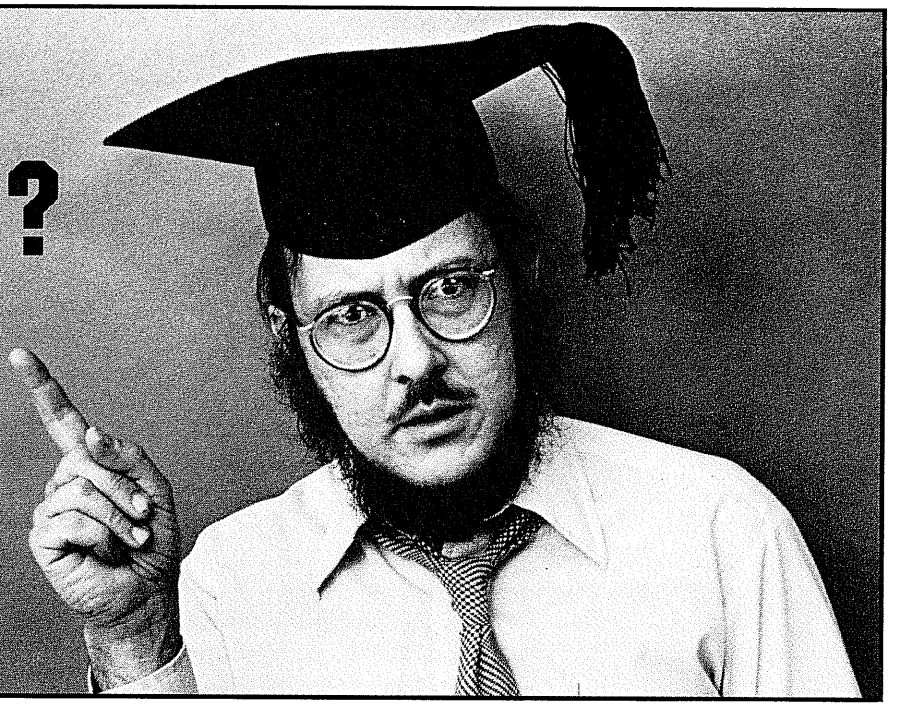
PRODUCT
64 Column Model I/III/IV Emulator
Spectrum Products
P.O. Box 21272
Woodhaven, NY 11421
718-441-2807
$19.95 disk
64K and disk drive

This is a program designed for those people wishing to be able to run Basic programs written for the Models I/III/IV computers from Tandy. The program emulates the screen and graphics allocations for these machines, although there is no fix for missing commands such as DEFDBL, DEFSNG, DEFINT. Essentially, this program is designed for those who want to load a Model I/III/IV Basic program (or type it) into a CoCo without having to change the graphics statements. Spectrum Projects notes that not all programs will necessarily run; they recommend the use of Spectral Associates Magic Box (also available through Spectrum Projects for $24.95, tape) to facilitate the baud and tape monitoring utilities as well as other differences between the languages.

This program emulates a blace and white tv monitor. Because of the character size and styling, the characters can be very difficult to read due to color flare and poor resolution. Spectrum is aware of this problam, and suggests several solutions: either turn the color off on the tv, or enter the command SCREEN,0, which will provide black characters on a green background for much better resolution. This problem will be less likely to occur on either a monochrome or composite video monitor.

The 64 Column Model I/III/IV Emulator from Spectrum Projects is a good stand-alone program for the user wishing to input Basic program designed for the Model I/III/IV. If the user has more sophisticated programming needs, other programs, such as Magic Box, would be necessary for a greater versatility of programming. *(end)*

FEELING HELPLESS? YOU NEED TO LEARN A LESSON!



MICRO LANGUAGE LAB: LEARNING THE 6809

Feeling at the mercy of a programmer somewhere? Mystified by a machine? My **Micro Language Lab** will give you the power to end those helpless feelings.

Here's why. The heart of *any* computer program *anywhere* is machine language. Every piece of software uses it — your favorite game, spreadsheet, word processor, data base manager, or recipe file. Every time you hit "Enter", it's working. All the languages and operating systems are created from it. Basic, Fortran, Pascal, Flex, OS-9, TRSDOS... they're 6809 machine language, the whole lot!

And you can learn the language of the 6809, the programming heart of your Color Computer, with my **Micro Language Lab.**

Not everyone can program. Writing a few lines in Basic now and then doesn't make you a programmer. But if you *can* program, then my **Micro Language Lab** will teach you — the right stuff, the right way.

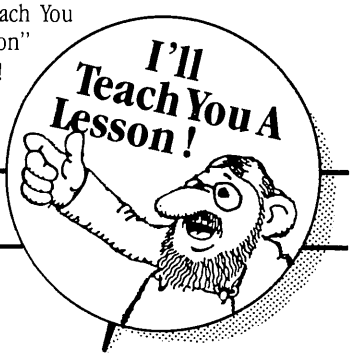
I'll teach you in 24 half-hour lessons on 12 audio cassettes, with a 220-page textbook, with data booklets, with 35 sample programs, and with a programming reference card. You'll spend 50 hours or more with my course, listening, watching, and working. And when you're done, you'll be programming your Color Computer in the 6809's language.

Stop ignoring those gnawing feelings of helplessness. The 6809 is where the power lies in your computer. Turn on that power with the **Micro Language Lab.**

- **Micro Language Lab,** $99.00 (plus $3.50 shipping and handling)

  Requires 16K Extended Basic EDTASM +

Not sure? Write or call for a Table of Contents and sample pages — and your "I'll Teach You A Lesson" button!



ALSO AVAILABLE FROM GREEN MOUNTAIN MICRO

(Add $2.50 shipping and handling to your order)

Lowerkit III

- Full-time upper and lowercase installs in 15 minutes.
- Normal and reverse video standard
- Fully compatible with all Alpha and Graphic modes

$79.95 assembled and tested
$49.95 complete kit of parts
Important! Specify Color Computer or Color Computer II

TV Buff II, clear image for video monitors, $19.95 (specify CoCo or CoCo2)

CoCoPort parallel interface (now includes plastic case), $54.95 / $44.95 kit

ColorPack ROM/RAM pack, $29.95 (specify configuration) / $19.95 kit / case $3.00 extra

64K memory upgrade kit with full instructions, $49.95 / with memory test on tape, $54.95

Color Quaver Software Music Synthesizer on tape (requires 32/64K), $19.95

Color Burner EPROM Programmer (2716/32/32A/64/64A/128, 68764/66) with software, $69.95 / $56.95 kit

User Group, Educational, Club and Dealer Discounts are available.

Green Mountain Micro
Bathory Road, Box U
Roxbury, Vermont 05669
802 485-6112
Hours: 9am - 5pm EST, Monday - Friday
COD/VISA/MASTERCARD

DYNACALC®

CoCo's Best & Fastest Spreadsheet System

ACCLAIMED BY THE EXPERTS

"DYNACALC is my choice for a CoCo spreadsheet."
Dan Downard, RAINBOW, September, 1984.

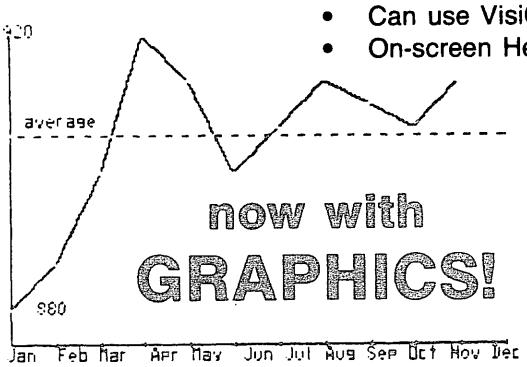
"Eat your heart out, Lotus 1-2-3!"
Scott Norman, HOT CoCo, October, 1984.

Built-in Features:

- 51 x 24 Display with Lower Case
- Super-fast Smart Screen Refresh
- Auto-Repeat Keyboard Driver
- Keysaver (TM) repeats last command x times
- Disk Operating System (works just like ROM DOS)
- Easy communication with BASIC/DOS programs
- Two-way communications with PRO-COLOR-FILE ★Enhanced★
- Outputs to ASCII Word Processors like Telewriter-64
- Fast 16-Digit Arithmetic with Scientific Functions
- Summation, Mean, and Standard Deviation Functions
- Logical Functions with String & Numeric Comparison
- String locate command to navigate large worksheets
- Sort full or partial worksheet by columns or rows
- Line, Bar, Hi/Lo/Close, Circle Graphs
- Full Graphics captioning and overlay facility
- Graphics Drivers for all popular Printers
- Joystick/Mouse Driver for Cursor Movement
- Works with any ROM versions, even JDOS
- 33k Available Worksheet Space
- Up to 256 Columns or 256 Rows
- Can use VisiCalc worksheets & training materials
- On-screen Help Messages

$99.95

RAINBOW CERTIFICATION SEAL



FOR 64K DISK SYSTEMS
available from

COMPUTER SYSTEMS CENTER
42 Four Seasons Center #122
Chesterfield, MO 63017
(314) 576-5020
10 to 6 Mon.-Fri.

NEW ADDRESS

CSC

or your local DYNACALC dealer

Price US$99.95
Outside North America add $10 postage

CANADA- CDN $129.95
RGS MICRO INC.
759, VICTORIA SQUARE 405
MONTREAL H2Y 2J3
TEL.: (514) 287-1563
ORDER LINE ONLY ★ ★ ★
QUEBEC-ONTARIO-MARITIMES
800-361-5338
WESTERN CANADA 800-361-5155

MasterCard

VISA