Serving the CoCo Community for

The RAINBOW 12 YEARS

THE COLOR COMPUTER MONTHLY MAGAZINE

March 1993 Vol. XII No. 8 Canada $4.95 U.S. $3.95

Feature Program

VEF Graphics for Disk BASIC

by Thomas Wong

One of the CoCo 3's best features has always been its ability to create very detailed graphics images through BASIC or a graphics editor such as Spectra 3 (THERAINBOW, October 1990, Page 10). Over the years, many different file formats for saving these creations on disk have been introduced. However, as the disks also fill up with the wide assortment of the picture savers and viewers necessary for the different formats, it becomes obvious that a standardized format should have been developed earlier. Few of the numerous formats invented are programmer-friendly or provide support for more than one screen resolution. Meanwhile, OS-9 Level II users have been able to sit back and enjoy the VEF format that essentially came with the operating system.

The VEF format has gained wide acceptance in the OS-9 world for good reason: it's a straightforward format that fits the intended job. With support for five resolutions, four of them perfect for BASIC's HSCREEN resolutions, it is easy to see that VEF could have saved Disk BASIC programmers a great deal of development time.

The program presented here, VEFit, allows CoCo 3 users to take advantage of this format. VEFit allows you to save any HSCREEN in memory as a VEF-format disk file, then view the image at a later time. With the exception of Type-2 VEF images, any VEF picture, including those created under OS-9, can be viewed. The Type-2 image uses a resolution of 160 by 200, with 16 colors -- a configuration for which there is no HSCREEN equivalent. Thankfully, Type-2 VEF's are rarely used under OS-9.

Getting Started

VEFit requires a CoCo 3 with at least 128K and one disk drive. To get started, enter the listing for VEFIT, save the program to disk, then enter RUN.

After a short pause, you are asked to select the type of monitor you are currently using. This is an important factor when loading or saving a picture's palette values; since composite color monitors (or TVs) and RGB monitors differ in the ways they

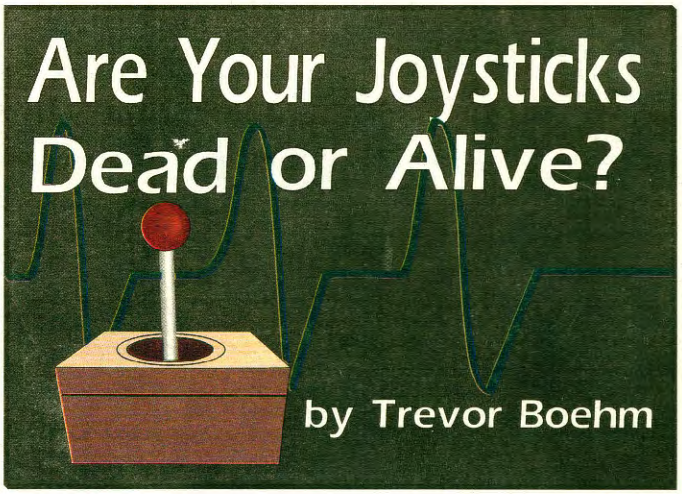
VEF Graphics continued on Page 10.

Byte #	Bytes	Description
1	1	Compression: 0=none, 128=squashed Note: VEFit supports no compression
2	1	Picture resolution: 0 = 320-by-200, 16 colors: HSCREEN2 1 = 640-by-200, 4 colors: HSCREEN4 2 = 160-by-200, 16 colors: No support 3 = 320-by-200, 4 colors: HSCREEN1 4 = 640-by-200, 2 colors: HSCREEN3
3-18	16	Palette values (RGB)
19-658	640	8 blank scan lines: HSCREEN2 or HSCREEN4
19-1298	1280	8 blank scan lines: HSCREEN1 or HSCREEN3
659-16018	15360	192 scan lines if HSCREEN2 or HSCREEN4
1299-32018	30720	192 scan lines if HSCREEN1 or HSCREEN3

Figure 1: VEF Format

Feature Program

Are Your Joysticks Dead or Alive?



by Trevor Boehm

Over the years, my Color Computers have seen over 25 joysticks come and go. But until I wrote Joystick Checker, I had no easy way to determine if the joysticks were dead or alive. More recently, this short utility has helped me with joystick repairs.

To use Joystick Checker, enter the program as shown in the listing and save it to tape or disk as JOYTEST. After running the program, simply follow the screen prompts to determine if your joystick works. If the joystick does not have a second button (like the one on the Tandy Deluxe Joystick), the program will break out of the loop after it tries to read that button. Other failures will cause the program to loop endlessly -- press BREAK to abort. If the joystick performs fine, the program reports the success on the screen.

Users with CoCo 1's or 2's must convert the program so it uses the PEEK(65280) function to read the button since the BUTTON command is available only on the CoCo 3. Information about doing this appears in the BASIC manual.

Joystick Checker is a useful program that should be in every CoCo user's bag of tricks.

Trevor Boehm is a tenth-grade student whose greatest passion is challenging computers with new programs. He has participated in several science fairs and has received numerous awards for his work. He can be contacted at 77 Inwood Cres., Winnipeg, MB R2Y 1A2, Canada. Please include an SASE when requesting a reply.

See program listing on Page 3

In this issue:

THE RAINBOW

Editor and Publisher
Lawrence C. Falk

Managing Editor
Cray Augsburg
Associate Editor
Sue Fomby
Submission/Reviews Editor
Julie Hutchinson
Technical Editor
Greg Law
Technical Assistant
Ed Ellers
Contributing Editors
Martin Goodman, M.D.,
Tim Kientzle, Eddie Kuns
Consulting Editor
Laura Falk
Art Director
O'Neil Arnold
Designers
Sharon Adams, Heidi Nelson
Editorial Director
John Crawley

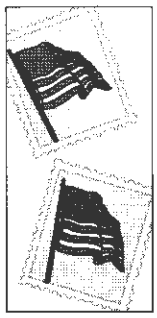
Falsoft, Inc.
President
Lawrence C. Falk
General Manager
Judi Hutchinson
Asst. General Manager for Finance
Donna Shuck
Asst. General Manager for Administration
Tim Whelan
Administrative Asst. to the Publisher
Ellen Patterson

Bookkeeper/Dealer Accounts
Debra Wilson
Customer Service Manager
Marisa Crouch
Customer Service Representative
Lauren Yates
Business Assistant
Shannon Yoffe
Corporate Business Technical Dir.
Calvin Shields
Chief of Building Security & Maint.
Lawrence Johnson

Advertising:
Western Sales Director
Ira Barsky (312) 587-1818
Eastern Sales Director
Kim Lewis (502) 228-4492

For RAINBOW Advertising and Marketing office information, see Page 15.

THE RAINBOW is published every month of the year by FALSOFT, Inc., The Falsoft Building, 9509 U.S. Highway 42, P.O. Box 385, Prospect, KY 40059, phone (502) 228-4492. **THE RAINBOW, RAINBOWfest** and **THE RAINBOW** and **RAINBOWfest** logotypes are registered trademarks of FALSOFT, Inc. ● Second class postage paid Prospect, KY and additional offices. USPS N. 705-050 (ISSN No. 0748-4797). POSTMASTER: Send address changes to **THE RAINBOW**, P.O. Box 385, Prospect, KY 40059. Authorized as second class postage paid from Hamilton, Ontario by Canada Post, Ottawa, Ontario, Canada; GST No. R125434092. ● Entire contents copyright © by FALSOFT, Inc., 1993. **THE RAINBOW** is intended for the private use and pleasure of its subscribers and purchasers and reproduction by any means is prohibited. Use of information herein is for the single end use of purchasers and any other use is expressly prohibited. All programs herein are distributed in an "as is" basis, without warranty of any kind whatsoever. ● Tandy, Color BASIC, Extended Color BASIC and Program Pak are registered ® trademarks of the Tandy Corp. ● Subscriptions to **THE RAINBOW** are $31 per year in the United States. Canadian rates are U.S. $36. Surface mail to other countries is U.S. $68, air mail U.S. $103. All subscriptions begin with next available issue. ● Limited back issues are available. Please see notice for issues that are in print and their costs. Payment accepted by VISA, MasterCard, American Express, cash, check or money order in U.S. currency only. Full refund after mailing of one issue. A refund of 10/12ths the subscription amount after two issues are mailed. No refund after mailing of three or more magazines.



LETTERS

Looking for CoCo 1's and 2's

Editor:

To support our R&D work with deaf children, I am looking for working CoCo 1's and 2's in good cosmetic shape. If any readers of THE RAINBOW have spare units they'd be willing to sell, they may contact me by phone, letter or fax. Thank you for your kind assistance.

Norman Lederman
Oval Window Audio
33 Wildflower Court
Nederland, CO 80466
(303) 447-3607 (Phone/Fax)

Wants Telecommunications Articles

Editor:

I want to send my kudos to THE RAINBOW for its fine work. Through time, I've managed to collect every back issue. I also want to point out that it looks like the future of the CoCo and its community is going to reside in telecommunications. Already Delphi and Fido-NET play a big part in keeping us connected. It would be nice to see THE RAINBOW provide some more articles on telecommunications, maybe in preparation for the time when CoCo users no longer have THE RAINBOW, or in the event THE RAINBOW moves its existence onto the Delphi services. Such articles would help solidify the community by helping us keep connected. I don't deny the need for other articles, but it seems the subject of telecommunications is underrated or overlooked (not covered as frequently), yet it is one of the things that holds our community together. It would be a terrible thing if our community broke up — it certainly would be the demise of our powerhouse 2-MHz CoCo.

Jason Gross
1122 Crosstown Boulevard
Chaska, MN 55318

We agree that telecommunications is very important to all CoCo users. This is why we see "Delphi Bureau" as such an important part of THE RAINBOW each month. And we encourage all CoCo owners to actively use the communications resources available.

OS-9 Help and Pen Pals

Editor:

I want to know how to build fonts under OS-9 Level II and create music using BASIC09. I would also like to receive letters from prospective pen pals between the ages of 10 and 13.

Nathan Price
658 Bond Avenue
Valparaiso, IN 46383

Several programs for creating and editing Level II fonts are available for downloading from the OS9 Online SIG on Delphi. We imagine there are at least one or two for creating music through BASIC09, as well.

Pascal and Trouble Booting OS-9

Editor:

Regarding the inquiry by B.N. Gregoire in the December issue of THE RAINBOW, Tandy did market a Pascal compiler for OS-9 Levels I and II. The product (Cat. No. 26-3-34) was reviewed in the December 1989 issue of THE RAINBOW. In addition, it is currently in stock and available through Radio Shack Express Order.

I have a Radio Shack FD-502 disk controller that runs with Disk BASIC. When I try to boot OS-9 with this controller by entering `DOS`, however, I get a syntax error. I don't have this problem with my Disto Super Controller II. Any suggestions?

John Kolb
20-4F Mosholu Pkwy. S
Bronx, NY 10468

We'll venture a guess: Since you have the Disto controller, it is likely you have installed and swapped a few different DOS chips between the controllers. Could you have installed Disk BASIC 1.0 (which displays as 2.0 on the CoCo 3 startup screen) in the FD-502 controller? Disk BASIC 1.0 does not support the DOS command, whereas Disk BASIC 1.1 (which displays as 2.1 on the CoCo 3) and most other DOSs do.

Dueling Computers

Editor:

I liked the idea you presented about putting a PC next to your CoCo ("Print#-2," May 1992) and decided to imitate it, though I'm not ready to invest the money required for a new Intel-based PC as you suggested. I decided to purchase a 640K PC-XT with a monochrome monitor, a hard drive, a mouse and some software. XT systems have been advertised in this area from $275 to $350 with various accessories.

I hope to use my Tandy DMP-107 printer with both the CoCo 3 and the XT. The DMP-107 is supposed to support IBM codes if the DIP switches are set appropriately. Radio Shack sold me a DB25-to-36-pin printer cable (Cat. No. 26-288) that is supposed to connect the XT to the DMP-107. I want to know if a switch box is available that would enable me to control which computer is connected to the printer without disconnecting all the cables. I'd also like to know if and how programs saved on tape for the CoCo 3 might be loaded into the XT (other than by retyping them).

R.L. Aldrich
2505 Bernard
Denton, TX 76205

In addition to supporting DIP-switch control, it should be possible to put the DMP-107 into the IBM mode by sending (printing) a few control codes on a command line from either computer. Many electronics outlets offer parallel printer switches. Look through the ads in any electronics magazine for sources. We suggest you get the kind of switch that accepts 36-pin Centronics inputs. Also, you'll need an extra cable to go between the switch and the printer.

Printing Sideways

Editor:

I have a CoCo 3, an FD-502 disk drive, an RGB monitor and a DMP-107 with a Blue Streak Ultima serial-to-parallel interface. I am having a problem with the program in "Printing Sideways" (April 1992). Every time I run this program, after the prompt "Condensed—Standard" appears, I receive an `IF ERROR IN 670` message.

Also, does anyone have the address for Sugar Software? I have some disks from Sugar and am having problems with them. My letters are returned marked "No Forwarding Address."

John W. Anderson
1709 W. Union Street
Lancaster, OH 43130

We've tried to duplicate the problem you are having, and the only way we've been successful is to use an empty input file. Make sure the file you intend to print is a standard text file that actually contains text. In the meantime, we have forwarded your letter to the author.

Needs Help With OS-9

Editor:

I have had a 128K CoCo 3 for about five years now. Although I am very experienced with the CoCo 3, there are many things about it that I don't understand about OS-9. When entering commands (such as `date`) at the OS-9 prompt, I frequently get errors (especially Error #249). Am I doing something wrong? Do I need special software?

Also, what is BASIC09? And how do I type in those weird listings in THE RAINBOW? You know the ones that look like

```
setend lda #$0d  carriage return
       sta .y+   to end of path
```

Jesse Burt
2 Lombard Lane
Eastham, MA 02642

Unlike Disk BASIC, OS-9 does not always keep all of its commands in memory — there are simply too many of them. Generally, some commands are automatically loaded into memory when you boot OS-9. The others are saved as separate programs on disk and are loaded when you need them.

When you enter an OS-9 command, the operating system first checks to see if the program is in memory. If so, OS-9 runs the command. If not, the system looks for the program in the current execution directory on disk. For this reason, you should keep your OS-9 system disk in the current drive — at least until you gain a clear understanding of the directory structure OS-9 uses.

An Error 249 is generated when you attempt a disk operation on a disk with a format OS-9 does not understand (e.g., a Disk BASIC disk, or an OS-9 disk with a different number of tracks or sides). The most common errors received by novice users are 214 and 216. Error 214 is a "no permission" error, which tells you that you don't have permission to do what you want to do. The most common cause for this error is trying to `list` a directory or perform a `dir` on a single file. Error 216 means "pathname not found," which tells you a) you mistyped the command line (or included/omitted necessary spaces or slashes), or b) the file/program does not exist in memory or the current data and execution directories.

Whew! We'll leave you other questions for other readers and a future issue. In the meantime, for more information, consult the "Getting Started" section of your OS-9 Level II manual. The knowledge you'll gain is well worth the time it'll take to read.

Help With the Lingo

Editor:

Being a novice in the computer world, I am confused by the many different letters used to describe computers (e.g., EX, DX, AT, PS/2, CAD/CAM, etc.). I am sure these designations have some meaning,

but what they stand for is anything but clear to me at this time. Please be assured that I have asked around only to find that I am not alone in this. What I need is a list of words to go along with the letters so I can have some idea of what people are talking about.

David Donnelly
1612 Diplomat Drive
Fayetteville, NC 28304

Some of the terms you mention are simply names used by different manufacturers to designate different computer models. For example, EX is the last part of the name for a specific model of Tandy 1000 (as are HX, TX and RLX), and PS/2 is the name of a line of computers marketed by IBM.

The other items you mentioned are acronyms for different computer techniques, terms and applications. For example, CAD/CAM stands for Computer-Aided Design/Computer-Aided Manufacturing. You can find definitions for this latter group of terms/acronyms in any good computer dictionary.

THE RAINBOW welcomes letters to the editor. Mail should be addressed to: Letters to Rainbow, The Falsoft Building, 9509 U.S. Hwy 42, P.O. Box 385, Prospect, KY 40059. Letters should include the writer's full name and address. Letters may be edited for clarity or to conserve space.

Letters to the editor may also be sent to us through our Delphi CoCo SIG. From the CoCo SIG> prompt, enter RAI to get to the Rainbow Magazine Services area of the SIG. At the RAINBOW> prompt, enter LET to reach the LETTERS> prompt, then select Letters for Publication. Be sure to include your complete name and address.

This listing is from "Are Your Joysticks Dead or Alive" on Page 1.

CoCo 3, CoCo 1/2 Modification

The Listing: JOYTEST

```
1 'JOYTEST
2 'BY TREVOR BOEHM
3 'COPYRIGHT (C) 1993
4 'BY FALSOFT, INC.
5 'RAINBOW MAGAZINE
10 'CHECK YOUR JOYSTICK
20 'FOR COCO1/2 ALTER BUTTON
30 'TO PEEK STATEMENT
40 CLS
50 PRINT"JOYTEST 1.0"
60 PRINT"<C> 1992 BY FALSOFT, INC."
70 PRINT"ALL RIGHTS RESERVED"
80 PRINT:PRINT"PLUG JOYSTICK INTO RIGHT PORT"
90 PRINT"AND PRESS BUTTON..."
100 IF BUTTON(0)<>1 THEN GOTO 100
110 PRINT"NOW PRESS SECOND BUTTON..."
120 FOR X=1 TO 5000:IF BUTTON(1)<>1 THEN NEXT:PRINT"NO SECOND BUTTON, I ASSUME..."
130 PRINT"GOOD, NOW MOVE STICK TO:"
140 PRINT" UPPER LEFT..."
150 GOSUB 240:IF X<>0 OR Y<>0 THEN GOTO 150
160 PRINT" LOWER LEFT..."
170 GOSUB 240:IF X<>0 OR Y<>63 THEN GOTO 170
180 PRINT" UPPER RIGHT..."
190 GOSUB 240:IF X<>63 OR Y<>0 THEN GOTO 190
200 PRINT" LOWER RIGHT..."
210 GOSUB 240:IF X<>63 OR Y<>63 THEN GOTO 210
220 PRINT:PRINT"JOYSTICK CHECKS OUT OK!"
230 END
240 X=JOYSTK(0):Y=JOYSTK(1)
250 RETURN
```

Feature Program

Skipper Shows Fast GET/PUT Operation

by Keiran Kenny

Skipper is a short program that uses "speed" GET/PUT commands to produce an animated simulation of a girl jumping rope on the PMODE4 screen. The program requires at least 32K of RAM and Extended BASIC.

Part of *Skipper*'s speed is a result of the way storage space is allocated. In Line 20, space is set aside for ten 128-by-128 frames. The DIM value of 409 was arrived at by dividing the square of 128 by 32, then reducing the value until I got an FC error, which occurred at DIMA(408). Using this method in other programs, I have succeeded in dimensioning space for twenty-six 48-by-48 frames, using 57 as the DIM value.

Another aspect of *Skipper*'s speed comes from the way even multiples are used for defined locations. As shown in Line 110, the values for the top-left corner of a frame are multiples of eight, and the values for the lower-right corner are one less than a multiple of eight. All GET/PUT statements in the listing conform to this protocol, and G or PSET options are not needed.

Line 100 stores a 128-by-128 area of blank screen, labeled A, for use as an eraser. Lines 130 through 160 draw the basic figure, labeled B. Lines 180 through 400 add the skipping rope in eight different positions, stored with labels C to J. All drawing is done behind the scenes. If you want to see the frames being drawn, add SCREEN1,1 to the end of Line 90. The 132-by-132 square in Line 120 was for my own guidance. To keep my figure within a 128-by-128 rectangle, I made sure there was a visible margin of at least one pixel between any extremity of the figure and the sides of the square.

Lines 1000 through 1060 put frames C to J on the screen in succession. In Line 1060, the skipper jumps to allow the rope to pass under her feet — the eraser is needed before and after this. The short delay loop in the subroutine at Line 60 slows the action very slightly. If you change the value of 10 to a larger number, like 500 or 1000, you'll get time-delay animation.

The high-speed poke is in effect throughout program operation. Line 30 sets the values SP and SL according to whether you are using a CoCo 3 or an earlier model CoCo. Then the appropriate poke is issued in Line 40. During the display you can press CLEAR to stop the program and restore operating speed to normal.

If you use any sort of boot utility on your disks, you may find that this program freezes at Line 80. If so, do a cold start and run the program directly from the disk.

If you want to substitute your own graphics image for mine, add SCREEN1,1 to Line 90 and enter a "stop" line: 980 GOTO 980. Draw your basic figure beginning at Line 130 and, when it is complete, store the frame labeled B, as in Line 170. The subroutine at Line 70 puts the basic figure on the screen after you complete and store the subsequent frames.

I hope you'll agree that, despite the large size of the frames, the animation is fairly smooth and flicker-free.

Keiran Kenny's interests lie mainly with the Color Computer's graphics and math capabilities. But in his own words, "I like to try everything." He may be contacted at van Montfoortlaan 31, 2596 SP The Hague, Holland. Please include a self-addressed envelope with sufficient postage when requesting a reply.

32K ECB

The Listing: SKIPPER

```
1 'SPEED GET/PUT SKIPPER
2 'BY KEIRAN KENNY
3 'COPYRIGHT (C) 1993
4 'BY FALSOFT, INC.
5 'RAINBOW MAGAZINE
10 CLS
20 DIMA(409),B(409),C(409),D(409),E(409),F(409),G(409),H(409),I(409),J(409)
30 IFPEEK(33021)=50 THENSP=65497:SL=65496ELSESP=65495:SL=65494
40 POKESP,0
50 GOTO80
60 FORDL=1TO10:NEXT:RETURN
70 PUT(X1,Y1)-(X2,Y2),B:RETURN
80 PRINT@224,"ONE MOMENT PLEASE...."
90 PMODE4,1:COLOR0,5:PCLS
100 GET(0,0)-(127,127),A
110 X1=64:Y1=32:X2=191:Y2=159
120 LINE(62,30)-(193,161),PSET,B
130 DRAW"BM128,150L6U35R12D35L6U35NL15R15M-6,-25BM113,115M+6,-25M-6,-17M95,99U6M113,67E2R5E2U2"
140 DRAW"BM+11,+0D2F2R5F2M162,94D6M141,73M137,90"
150 CIRCLE(128,52),8,,1.4:PSET(125,51):PSET(131,51):LINE(128,52)-(128,55),PSET:DRAW"BM128,57NL2NR2"
160 DRAW"BM120,65M128,70M135,65BM114,75M118,65BR20M141,75BM121,49R3E2R2NU6RF2R3":PAINT(126,45),0,0:PAINT(130,45),0,0
170 GET(X1,Y1)-(X2,Y2),B
180 CIRCLE(128,96),34,,1.9,,5,1
190 GET(X1,Y1)-(X2,Y2),C
200 GOSUB70
210 CIRCLE(128,96),34,,1.5,,5,1
220 GET(X1,Y1)-(X2,Y2),D
230 GOSUB70
240 CIRCLE(128,96),34,,1.1,,5,1
250 GET(X1,Y1)-(X2,Y2),E
260 GOSUB70
270 CIRCLE(128,96),34,,,5,,5,1
280 GET(X1,Y1)-(X2,Y2),F
290 GOSUB70
300 CIRCLE(128,96),34,,,5,0,,5
310 GET(X1,Y1)-(X2,Y2),G
320 GOSUB70
330 CIRCLE(128,96),34,,1.1,0,,5
340 GET(X1,Y1)-(X2,Y2),H
350 GOSUB70
360 CIRCLE(128,96),34,,1.5,0,,5
370 GET(X1,Y1)-(X2,Y2),I
380 GOSUB70
390 CIRCLE(128,96),34,,1.9,0,,5
400 GET(X1,Y1)-(X2,Y2),J
990 PCLS:SCREEN1,1
1000 PUT(X1,Y1)-(X2,Y2),C:GOSUB60
1010 PUT(X1,Y1)-(X2,Y2),D:GOSUB60
1020 PUT(X1,Y1)-(X2,Y2),E:GOSUB60
1030 PUT(X1,Y1)-(X2,Y2),F:GOSUB60
1040 PUT(X1,Y1)-(X2,Y2),G:GOSUB60
1050 PUT(X1,Y1)-(X2,Y2),H:GOSUB60
1060 PUT(X1,Y1)-(X2,Y2),I:GOSUB60
1070 PUT(X1,Y1)-(X2,Y2),A:PUT(X1,Y1-16)-(X2,Y2-16),J:GOSUB60:PUT(X1,Y1-16)-(X2,Y2-16),A
1080 IFPEEK(135)=12THEN1100
1090 GOTO1000
1100 POKESL,0:CLS:END
```

EDDIE KUNS

Delphi Adds More Internet Services

Delphi has added both FTP and Telnet! These options are available only from the Internet SIG, so to find out more, just enter `GO REFERENCE INTERNET` to get to the Internet SIG. From there, enter `FORUM` and read Message 1197.

What are FTP and Telnet? FTP stands for File Transfer Protocol, and is the Internet protocol for transferring files from one computer to another. Like Internet mail, FTP file transfers count toward your monthly 10-megabyte Internet transfer limit. As I explained a few months back, Telnet is the service that allows you to logon to a computer connected to the Internet from any other computer also connected to the Internet. This means you can use this service *from* Delphi as well as *to* Delphi.

Yon still pay the same Delphi and telecommunications charges while Telnetted to another computer. However, most users can reach Delphi without paying long-distance rates. Once on Delphi, they could connect to another computer that *would* otherwise incur long-distance telephone charges. These people may benefit by using the link from Delphi to Telnet to the remote computer. For example, college students who are home on vacation might be able to use Telnet to connect to a school computer without having to call long distance.

If you are interested in using these features, you must read and follow the instructions given in Message 1197 in the Internet SIG Forum. There is no additional charge for using FTP or Telnet, but you must already have registered to use Internet services. Also remember that there are guidelines for any use of Internet services — read these guidelines before you sign up for any services. To learn more about exactly what services are available, enter `USING INTERNET SERVICES` at the main menu of the Internet SIG, then read the file `I'M NEW! WHAT DO I DO?`.

Eddie Kuns is pursuing a doctorate in physics at Rutgers University. He lives in Aurora, Illinois, and works as a programmer and researcher at Fermilab. Eddie is the database manager of the OS9 Online SIG and can be reached online as EDDIEKUNS.

MARTY GOODMAN

What is MNP?

Please explain the significance of MNP level numbers in modem communication. How can MNP be of practical use to me?

John L. Wilkerson (JWILKERSON)
Reynoldsburg, Ohio

AMNP, which stands for Microcom Network Protocol (no relation to Microcom Software), capability typically is built into the *firmware* (software in ROM chips) of a modem. MNP levels 2 through 4 enable the modem to automatically check for and correct errors that occur due to line noise when "talking" to another modem with the same capability. Telenet and Tymnet (telecommunications services that can be used to connect to Delphi's CoCo and OS9 online SIGs) support MNP error correction if you are connected at 2400 bps. This means that if you use an MNP modem that is correctly set to use its MNP capability, all noise characters that occur when it is connected to Telenet or Tymnet at 2400 bps will be eliminated. The tricky part is telling the modem to use its MNP capability.

In my experience, different modems require different (and often confusing) commands to enable this capability, so it is usually essential that you have the manual for your modem — read the section on MNP operation several times. You need to set your modem to "Auto-Reliable" mode, in which it turns on its MNP capability when it sees the same capability in the modem it is calling. For my Racal Vadic 2400-bps modem, I must enter (in command mode) `AT &*E1` to enable automatic error control, whereas on my Multitech 2400-bps modem, the proper command is `AT &E1`. (After entering either of these, I entered `AT &W` to make the setting a power-up default.)

There are other MNP options that many users find confusing. For instance, MNP 5 and up are designed to be used for data compression. Neither Telenet nor Tymnet currently support MNP 5, and using MNP data compression can be problematical. MNP 5 data compression is usually not helpful if you are transferring files that have already been compressed using an archive program. Indeed, on occasion, MNP 5 can actually expand a compressed file, resulting in an *increase* in the time it takes to transmit or receive it. Some modems support other forms of error correction (CCITT V.42) and other forms of data compression (CCITT V.42 bis).

I use simple MNP error correction when logging on to the CoCo SIG, and I find it a nice convenience since it eliminates the occasional noise characters I *used to* get when listing longer Forum and Mail messages. I don't find it all that important for Delphi use, though, because in a case where you absolutely must get an error-free transmission (e.g., downloading of binary files), you're going to use a protocol download (Xmodem, Ymodem or Zmodem) anyway. Such protocols assure error detection and error correction. Still, users who find line noise to be a bit of a problem will appreciate what MNP error correction can do.

Model I Monitor for the CoCo 3

Can I use a TRS-80 Model I monitor with my CoCo 3?

George Hill
Haverhill, New Hampshire

AYes, but it's tricky. The TRS-80 Model I monitor uses a 5-pin DIN plug for connection with the computer. One line on this plug is ground, and another line provides composite video. You can use these with the composite-video port on the CoCo 3. However, the Model I monitor also requires a well-regulated source of 5 volts DC on another pin of the 5-pin plng. Thus, yon'll need an external 5-volt power supply to make the monitor work.

Unfortunately, I don't recall what the pinout for the 5-pin connector is, and I have no references for it here. If you are able to track this information down, be sure that you hook both 5 volts and ground from the power supply to the appropriate pins of the monitor. If the monitor is working, it should accept the standard NTSC 1-volt peak-to-peak, composite-video signal that the CoCo produces.

Depending on your situation, it may not be worth the effort to resurrect the Model I monitor. Those monitors were of fairly poor quality even when new, and you can obtain much better used composite monitors for pretty low prices in most urban areas and through the classified-ads section on Delphi.

Double-Sided Access

What are the pokes for enabling Disk BASIC to gain access to the back side of a double-sided disk drive?

Bob Williams (BAWILLIAMS)
Cedar Hill, Montana

AIf you are using Disk BASIC 2.0 on a CoCo 3, enter the following command line:

```
POKE &HD89F,&H41:POKE &HD8A0,&H42
```

These pokes define Drive 2 as the back side of Drive 0 and Drive 3 as the back side of Drive 1. If you are using Disk BASIC 2.1, use addresses `D7AC` and `D7AD` instead of `D89F` and `D8A0` above. These pokes work only on a CoCo 3.

A much better modification for using double-sided drives is found in *ADOS 3*, which not only redefines the drives but fixes some other subtle problems that cause the head of the drive to get "lost" whenever you switch between sides on the drive.

Making a Modem Cable

How do I construct a cable to connect my RS-232 Pak to a modem?

Fred Trivett (FREDT)
Augusta, Georgia

AModems with DB-25 connectors connect "straight through" to the RS-232 Pak. Actually, only pins 2, 3, 4, 5, 6, 7, 8 and 20 are used, so if you are soldering your own cable, you really need only a nine-conductor cable. You may find, however, that your modem does not echo characters when you enter commands in the command mode before connecting. There are two ways to fix this. One is to issue the `AT &C0` command, then make this setting a default by entering `AT &W`. This works with many modern 2400-bps Hayes-compatible modems that have internal non-volatile RAM for parameter settings. Alternatively, instead of hooking Pin 8 of the modem to Pin 8 of the RS-232 Pak, try hooking Pin 8 of the RS-232 Pak to Pin 20 of the RS-232 pak, leaving Pin 20 also hooked to Pin 20 of the modem. This is another way to force the carrier detect line high in the RS-232 Pak, which is required to turn on its internal receiver.

Slot/Multi Pak Repair

I've got a dead Howard Medical Slot Pak. Do you have any suggestions on where to start when repairing it? I believe it was blown when someone unplugged a cartridge from it while the power was on. Do you have any tips for fixing a Radio Shack Multi-Pak Interface with a similar history?

Dennis McMillan (COCOKIWI)
Pittsburg, California

AHoward Medical released a service notice a while back that indicated the most likely chip to get blown when a user removes a cartridge from the Slot Pak with the power on is the 74LS08 chip. Thus, I'd suggest you start by desoldering that chip, then installing a socket and a new 74LS08. Howard Medical also noted that the Slot Pak powers slots A and B from its external power supply, but the last slot is powered from the CoCo's power supply. This information may be helpful to those choosing which card to put in which slot, for you don't want to draw too much power from the CoCo (which has only 100 to 250 milliamps to spare).

For the Multi-Pak Interface, after testing the power supply, I'd consider replacing IC1 (an LS245 chip) and IC2 (an LS367 chip). These chips have the same IC numbers on both models of the Multi-Pak Interface.

Parallel Converter on the Blink

I have a Microfazer serial to parallel converter with 64K buffer built in that is now malfunctioning. Half the characters come out wrong. Any ideas?

Ted Jaeger (TEDJAEGER)
Fulton, Missouri

AOften when half the characters printed are wrong, the problem is that a single bit in the parallel port is "stuck" either high or low. Compare the ASCII codes for the misprinted characters to see if you can find a single bit in the 7-bit ASCII representations that, when forced high or low, would explain what you are getting. After this, you must find where that bit is stuck! Often the problem lies in the output latch (usually an LS373 or LS374 chip) on the parallel port. Determine which chip is supplying pins 2 through 9 of the parallel ouput of the buffer, and see if replacing that chip helps.

Martin H. Goodman, M.D., a physician trained in anesthesiology, is a longtime electronics tinkerer and outspoken commentator — sort of the Howard Cosell of the CoCo world. On Delphi, Marty is the SIGop of THE RAINBOW's CoCo SIG. His non-computer passions include running, mountaineering and outdoor photography. Marty lives in San Pablo, California.

Advanced Forum Details

Last month I described many of the key Workspace features and commands. Most of those we didn't cover relate to uploading and downloading files, but we'll first take a look at a few general commands.

If you want to create a short file online, entering just a few lines of text, use the CREATE command. For example, suppose I was throwing a party and wanted to inform five local friends. I could use CREATE to build a Mail distribution list as follows (the italic text is what I enter):

```
WS> CREATE PARTY.DIS

Ok, enter data, terminate with a
 CTRL/Z, or abort with a CTRL/C.

GBROOKS
MITHELEN
JOELHEGBERG
LUCKYONE
RAGTIMER
^Z

PARTY.DIS 5 lines
```

When I pressed CTRL-Z, Delphi saved the file in my Workspace as PARTY.DIS and reported the number of lines I entered. Notice that the editor used is the same one you use to edit Forum messages; you can enter /EXIT to exit, /LIST to see what you have entered so far, /EDIT to use your chosen editor (EDT or OLDie), or /QUIT to abort the create process.

Another file manipulation command is APPEND. If you enter

```
APPEND MAIL.TXT ARCHIVE.TXT
```

the contents of MAIL.TXT are added to the end of ARCHIVE.TXT, then the whole thing is saved as a *new version* of ARCHIVE.TXT. After entering this command, the original MAIL.TXT file is unchanged, and you'll have two versions of ARCHIVE.TXT.

The DIR command supports many more options than I listed last month. One of the more useful directory options is DIR/GRAND_TOTAL/SIZE. This command tells you how many files you have in Workspace as well as how much disk space is taken up by these files. (Delphi may charge you for less disk space than is shown by this command, depending on the amount of unused space in your MAIL.MAI file.)

DIR supports exclusion modifiers that tells it to ignore certain files when listing the directory. For example, to tell Delphi to not display mail files, you would enter

```
DIR/EXCLUDE=.MAI
```

The /EXCLUDE modifier forces the directory command to ignore these files. If you have multiple patterns (different file types, extensions, etc.) to specify, separate them with commas and put the group within parentheses as follows:

```
DIR/EXCLUDE=(*.MAI, MEMO*.*)
```

If you also want to see the file sizes and dates, you can add the qualifiers /SIZE/DATE to the command line:

```
DIR/SIZE/DATE/EXCLUDE=.MAI
```

Two more useful qualifiers for DIR are /SINCE and /BEFORE, which are used to Delphi to display a directory of the files created or modified in a specific time interval. If no interval is specified, the default is TODAY (/SINCE=TODAY and /BEFORE=TODAY displays all files). You can specify /SINCE=YESTERDAY, or you can specify a date in the format /SINCE=28-MAR-1992. You can also use both /SINCE and /BEFORE with dates to specify a range in time.

You can combine the /SIZE, /SINCE, /BEFORE, /EXCLUDE and /DATE qualifiers arbitrarily. The /GRAND_TOTAL qualifier is incompatible with the /DATE qualifier but can be nsed with any combination of the others.

The PURGE command also has more qualifiers than I mentioned last month. If you want to purge old versions of one or more files but want to keep more than just the latest version, yon can specify PURGE/KEEP=2 to keep the two most-recent versions of the file(s) you purge.

The SETTINGS command drops you into the Settings menu, which is also available from the main SIG prompts. (From the CoCo SIG or OS9 Online prompt, enter SET PREFERENCES, followed by SETTINGS. This can be abbreviated as SET SET). The Settings menu allows you to change various parameters; we'll take a closer look at its options another time.

The final file-manipulation command available in the workspace is the EDIT command, which we have covered in detail in past issues.

The most common use of Workspace is for uploading and downloading files. You can use most any file-transfer protocol. Batch file-transfer protocols function in Workspace, so you can upload or download several files with one file transfer operation.

If you want to use Kermit to transfer files, tell Delphi to start the Kermit server on its end by entering KERMIT. Now start Kermit on your computer and use the Kermit GET and SEND commands to download and upload (respectively) files with Delphi. Terminate the Kermit server by sending the Kermit Finish command or by pressing three consecutive CTRL-Cs.

The UPLOAD and DOWNLOAD commands initiate a file transfer using your current default file transfer. Enter the /FX_METHOD slash command to see what your current default file-transfer protocol is. If you prefer to specify the file-transfer method, the OTHER command shows you the commands that use a specific file-transfer protocol. Alternatively, enter UPLOAD MENU to see a menu of all supported upload methods (entering DOWNLOAD MENU works the same way). If you are downloading with a batch protocol — one that allows you to transfer more than one file at a time — Delphi prompts you for filenames or file patterns until yon press ENTER without entering a filename.

DATABASE REPORT

OS9 Online:

General Information:
6309 ADDITIONAL INDEXED MODES
CURTISBOYLE Curtis Boyle
KIX\30 MANUAL
FHOGG Frank Hogg
NEW FARNA ITEMS
DSRTFOX Francis Swygert
MICRO CHARTS
JSUTEMEIER Jim Sutemeier
KIX/30 EURO-K BUS PINOUTS
FHOGG Frank Hogg
ADVANCED UTILS FOR OSK INFO
EDELMAR Ed Gresick

Applications (6809):
ESP TESTER
MIKE_GUZZI Mike Guzzi
PRINTFORM 2.3 (REVISED)
WOAY Jim Martin
BRU 1.1: HARDDRIVE BACKUP UTIL
WOAY Jim Martin

System Modules (6809):
KRNLUTILS: KERNEL CHANGING UTILS
WOAY Jim Martin

Games & Graphics:
GWINDOWS ICON FOR TEXT EDITORS
JSUTEMEIER Jim Sutemeier
JACK-O-LANTERN FLICKER ANIMATION
GRAPHICSPUB Bob Montowski
KWINDOWS SCREEN SNAPS IN GIF
MIKEHAALAND Mike Haaland
IMASTER 1.01: IMAGE VIEWER/PRINT
DODGECOLT Mike Sweet
GIFSHOW 2.2 FOR THE MM/1
MIKEHAALAND Mike Haaland
GWINDOWS SAMPLE SCREEN (GIF)
JSUTEMEIER Jim Sutemeier
RUN PACOS9 FROM MULTIVUE
REDCOAT Don Joyce

Music & Sound:
SYSTEM_TEST: DIGITIZED SOUND
MODEL299 Mark Steiner

Programmers Den:
GUI.L: C GUI LIBRARY
WTHOMPSON Wayne Thompson
C++ LIB UPDATE
VAXELF John Donaldson

OSK Applications:
QFED: FONT EDITOR FOR GWINDOWS
PAULTESCH Paul Tesch
GWINDOWS FILE RECOGNIZERS
JSUTEMEIER Jim Sutemeier

OSK Telecom:
ATERM V2.6 (EXEC/SRC)
SMARTCOCO Daniel Boulanger
STERM: MODIFIED FOR ZMODEM
JSUTEMEIER Jim Sutemeier
TERMINAL VERSION 1.0
DODGECOLT Mike Sweet

Tutorials & Education:
PRENV: PRINT ENVIRONMENT
DPHILIPSEN Dave Philipsen

Standards:
DSHELL: GUIB STANDARD DEMO
ILLUSIONIST Michael Graffam

CoCo SIG:

General Information:
ATLANTA COCOFEST3 REPORT
SUBETHA Allen Huffman

CoCo 3 Graphics:
SKETCH3:DEMO VERSION
RACINEGUY David Potter
TALKING PUMPKIN PROGRAM
DRILLMASTER Johnny Williams
LEGENDS IN DS69B FORMAT (PIX)
DRILLMASTER Johnny Williams
PEROT IN CM3
DRILLMASTER Johnny Williams

Source for 6809 Assemblers:
6309 NEW INDEXED MODES
CURTISBOYLE Curtis Boyle
DSKCON & DSKINI
MARTYGOODMAN Marty Goodman

Product Reviews & Announcement:
NEW FARNA PRODUCTS
DSRTFOX Francis Swygert

Uploads at a Glance

In the OS9 Online General Information database, **Curtis Boyle** (CURTISBOYLE) released a file describing some recently discovered addressing modes (extras) available in the 6309 CPU. Curtis also submitted this information in the CoCo SIG's Source for 6809 Assemblers database. In the System Modules (6809) database, **Jim Martin** (WOAY) uploaded a pair of utilities designed to make it easier to update the Kernel track of a boot disk.

In the Games & Graphics database, **Mike Haaland** (MIKEHAALAND) contributed some screen captures of typical windows under *K-Windows* on the MM/1. **Jim Sutemeier** (JSUTEMEIER) contributed a screen capture of a typical window using *G-Windows*. All of these images are in .GIF format. Jim also released "file recognizers" for .AR and .LZH files under *G-Windows* — using these files provides icons for these file types.

No less than three OSK terminal programs were uploaded this month to the OSK Telecom database! Two of the three were updates to previous uploads, but it shows the life in the databases.

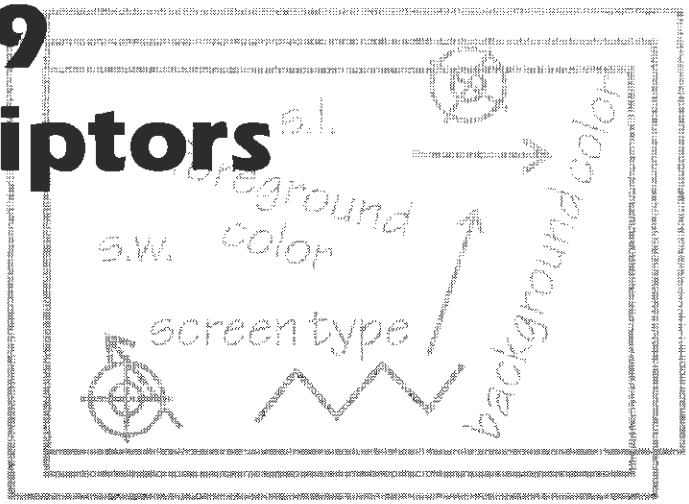
Dave Philipsen (DPHILIPSEN) released into the Tutorials & Education database the assembler source code for an OSK program that lists all environment variables. Anyone who wonders how to gain access to environment variables from within a program may find this source code useful.

In the CoCo SIG's CoCo 3 Graphics database, **Johnny Williams** (DRILLMASTER) uploaded a program that uses the Tandy Speech/Sound Pak to show a talking jack-o-lantern.

In the Source for 6809 Assemblers database, **Marty Goodman** (MARTYGOODMAN) provided an article that examines the DSKCON and DSKINI routines. If you are interested in the guts of disk I/O, you'll want to download this database gronp.

Feature Article

Modifying OS-9 Window Descriptors



By Ernest Bazzinotti, Jr.

OS-9's windowing system is great, but entering all those display codes can be a real pain. Have you ever wished you could simply initialize a window and it would already be set up just the way you want without your having to enter a bunch of display or `wcreate` codes? If so, read on . . .

The approach described here involves making changes to the actual device descriptors used for OS-9 Level II's windows. To do this, you'll use the `modpatch` command that comes with OS-9. You'll also need the `dump` and `save` commands, which are included with the OS-9 *Development System*. If you don't have the *Development System*, download similar utilities from Delphi or use an OS-9 disk editor. Though the approach will differ somewhat if you use a disk editor, the information provided here is enough to get you started. Some of you may even prefer to use a disk editor.

Needless to say, since we're going to change the modules on disk, the first thing you need to do (if you haven't already done so) is make a backup of your OS-9 system master. Put the original disk away for safety's sake, and use the copy.

With that little bit of housekeeping out of the way, you need to see what parameters are "hard-coded" into the window descriptors you want to change. Use the `dump` command to dump the contents of the desired window descriptor to the printer or screen.

The offsets to the hard-coded para-meters in a window descriptor are shown in Figure 1. To see what values the desired window descriptor uses for a particular parameter, use Figure 1 to determine the proper offset, then look at the dump of the descriptor and find the value at that offset. Armed with this information, it should be a simple matter to build a patch file to make the necessary changes.

First, jot down the offsets to the parameters you want to change, as well as the current (old) values at those offsets and the (new) values you want to use (remember that these numbers must be in hexadecimal). Then use `build` or a text editor to build a standard patch file that uses `modpatch`. To get you started, Listing 1 shows a standard procedure file designed to change the descriptor for Window 5 to a Type-7, 80-column graphics window. Let's use this sample patch file and step through the process.

First, use `dump` to view the contents of the standard `/w5` descriptor, which is stored in the `MODULES` directory of the Boot/Config/`BASIC09` disk as `w5.dw`. The result is shown in Figure 2. Now make a table of the parameters you want to change. The changes we'll make to the /w5 descriptor are shown in Figure 3. Once you have this table together, use it to build a `modpatch` file as shown in Listing 1.

To make the changes, just enter the name of the patch file (in this case, `changew5`), executing the script. To see your handiwork, enter the following commands to activate the window:

```
iniz w5
shell i=/w5&
```

Press the CLEAR key to switch to the modified window.

Once the changes are made to the module in memory, you can delete the original descriptor file from the `MODULES` directory and use the `save` command to save the new module there with the same name as the original. (Alternatively, instead of deleting the original, you could simply rename it to something like `w5.dw.old`.) At this point, create a new boot disk so the changed window will be available when you boot the system.

You can use this approach to change as many or as few of the window descriptors as you like. And remember, not all the parameters need to be changed — only the ones you want. Refer to Page 1-3 of the "Windows" section of the OS-9 Level II manual. The listing on this page will tell all you need to know about each of the windows supplied with OS-9. To give a better feel for the method I use to make the changes, refer to listings 2 and 3, which change the `/w4` and `/w8` descriptors to Type 2, 80-column text windows.

If you're using a disk editor to make the changes, you will be altering the actual data on disk. First copy the original file, say `w5.dw`, to a "reserve" file, say `w5.dw.old`. Then use the editor and the offset information to make the changes to `w5.dw`. Once the modified module is written to disk, you still need to make a new boot disk in order for the modifications to be available when you boot the system.

One of OS-9's strong features is its ability to be altered for specific uses and preferences. I hope you find the modifications presented here to be useful.

Ernest Bazzinotti works for Raytheon Co. and has used computers for three years, both as a hobby and for his work. He may be contacted at 91 Huggins Rd., Rockland, MA 02370, (617) 982-2412. Please include an SASE when requesting a reply.

```
002C screen width
002D screen height
0030 screen type (text or graphics)
0031 horizontal coordinate
0032 vertical coordinate
0033 foreground color
0034 background color
0035 border color
```

Figure 1: Window Parameters

```
Addr   0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0 2 4 6 8 A C E
------ ---- ---- ---- ---- ---- ---- ---- ---- ----------------
0000   87CD 0043 0036 F181 B000 3800 3803 07FF  .M.C.6q.0.B.;...
0010   A51A 0000 0100 0101 0000 1808 1800 1804  %...............
0020   0117 0305 0807 8000 0036 0D00 1308 0501  .........6......
0030   FF3D 0002 0704 57B5 5343 C643 4333 49CF  .=....W5SCFCC3IO
0040   C109 F3                                  A.s
```

Figure 2: Dump of `w5.dw` Descriptor

Parameter	Offset	Old Value	New Value
Screen Type	0030	FF	07
Horiz. Coordinate	0031	3D	00
Screen Width	002C	13	50
Screen Height	002D	0B	18
Background	0034	07	05
Border	0035	04	02

Figure 3: Changes to `w5.dw`

OS-9 Level II

Listing 1: `changew5`

```
modpatch -s
1 w5
c 0030 ff 07
c 0031 3d 00
c 002c 13 50
c 002d 0b 18
c 0034 07 05
c 0035 04 02
v
```

Listing 2: `changew4`

```
modpatch -s
1 w4
c 002c 3c 50
c 002d 0b 18
c 0033 00 02
c 0034 01 00
c 0035 04 07
v
```

Listing 3: `changew8`

```
modpatch -s
1 w8
c 002c 28 50
c 0030 01 02
c 0033 00 02
c 0034 01 00
c 0035 01 02
v
```

Yes! They're still available!

The RAINBOW Back Issues

BACK ISSUES STILL AVAILABLE

Have you explored the wealth of information in our past issues? From our very first, four-page issue to many with more than 300 pages of material, it's all just for CoCo users — a great way to expand your library!

A WORLD OF INFO AT A BARGAIN PRICE

All back issues sell for the single issue cover price. In addition, there is a $3.50 charge for the first issue, plus 50 cents for each additional issue for postage and handling if sent by United Parcel Service. There is a $5 charge for the first issue, plus a $1 charge for each additional issue on orders sent by U.S.Mail. UPS will not deliver to a post office box or to another country.

MOST ISSUES STILL AVAILABLE

Available issues through June 1982 are provided on white paper in a reprint form. All others are in regular magazine form. VISA, MasterCard and American Express accepted. Kentucky residents please add 6 percent sales tax; Canadian residents, 7 percent GST. In order to hold down costs, we do not bill, and no C.O.D. orders are accepted.

Due to heavy demand, we suggest you order the back issues you want now while supplies last.

To order, review and fill out the form below and mail it with your payment.

For greater convenience, order through the Rainbow Magazine Services area of our Delphi CoCo SIG.

RAINBOW INDEX

A complete index for July 1981 through June 1984, is printed in the July 1984 issue. Separate copies are available for $2.50 plus 50¢ handling. Indexes for subsequent years are published annually in the July issues of THE RAINBOW.

TOTAL ________
KY RESIDENTS ADD 6% ________
CANADIAN RESIDENTS ADD 7% GST ________
U.S. MAIL CHARGE ________
SHIPPING & HANDLING ________
U.P.S. CHARGE ________
TOTAL AMOUNT ________
ENCLOSED ________

Article Reprints

In instances where a given issue is now out of print and not available for purchase, we do provide photocopies of specific articles. The cost for this service is $1.50 plus 50 cents S/H per article. This service is provided *only* in the case of out-of stock issues.

Name ________
Address ________
City ________ State ________ Zip ________
❑ Payment Enclosed, or Charge to my: ❑ VISA ❑ MC ❑ AE
Card # ________
Expiration Date ________ Phone ( ) ________
Signature ________

TO ORDER BY PHONE (credit card orders only) call (800) 847-0309, 9 a.m. to 5 p.m. EST. All other inquiries call (502) 228-4492. **send to: THE RAINBOW, The Falsoft Building, P.O. Box 385, Prospect, KY 40059**

Please send me the following back issues:

	VOLUME 1	
JUL 81	Premier Issue	$2.00 ❑
FEB 82		$2.00 ❑
	VOLUME 2	
JUN 83	Printers	$2.95 ❑
	VOLUME 3	
AUG 83	Games	$2.95 ❑
SEP 83	Education	$2.95 ❑
OCT 83	Graphics	$3.95 ❑
MAR 84	Business	$3.95 ❑
APR 84	Gaming	$3.95 ❑
MAY 84	Primer	$3.95 ❑
JUN 84	Music	$3.95 ❑
JUL 84	Anniversary	$3.95 ❑
	VOLUME 4	
AUG 84	Games	$3.95 ❑
SEP 84	Education	$3.95 ❑
OCT 84	Graphics	$3.95 ❑
NOV 84	Data Comm.	$3.95 ❑
DEC 84	Holiday	$3.95 ❑
JAN 85	Beginners	$3.95 ❑
FEB 85	Utilities	$3.95 ❑
MAR 85	Business	$3.95 ❑
APR 85	Simulations	$3.95 ❑
MAY 85	Printer	$3.95 ❑
JUN 85	Music	$3.95 ❑
JUL 85	Anniversary	$3.95 ❑
	VOLUME 5	
AUG 85	Games	$3.95 ❑
SEP 85	Education	$3.95 ❑
OCT 85	Graphics	$3.95 ❑
NOV 85	Data Comm.	$3.95 ❑
DEC 85	Holiday	$3.95 ❑
JAN 86	Beginners	$3.95 ❑
FEB 86	Utilities	$3.95 ❑
MAR 86	Business	$3.95 ❑
APR 86	Home Help	$3.95 ❑
MAY 86	Printer	$3.95 ❑
JUN 86	Music	$3.95 ❑
JUL 86	Anniversary	$3.95 ❑
	VOLUME 6	
AUG 86	Games	$3.95 ❑
SEP 86	Education	$3.95 ❑
OCT 86	Graphics	$3.95 ❑
NOV 86	Data Comm.	$3.95 ❑
DEC 86	Holiday	$3.95 ❑
JAN 87	Beginners	$3.95 ❑
FEB 87	Utilities	$3.95 ❑
MAR 87	Business	$3.95 ❑
APR 87	Home Help	$3.95 ❑
MAY 87	Printer	$3.95 ❑
JUN 87	Music	$3.95 ❑
JUL 87	Anniversary	$3.95 ❑
	VOLUME 7	
AUG 87	Games	$3.95 ❑
SEP 87	Education	$3.95 ❑
OCT 87	Graphics	$3.95 ❑
NOV 87	Data Comm.	$3.95 ❑
DEC 87	Holiday	$3.95 ❑
JAN 88	Beginners	$3.95 ❑
FEB 88	Utilities	$3.95 ❑
MAR 88	Business	$3.95 ❑
APR 88	Home Help	$3.95 ❑
MAY 88	Printer	$3.95 ❑
JUN 88	Music	$3.95 ❑
JUL 88	Anniversary	$3.95 ❑
	VOLUME 8	
AUG 88	Games	$3.95 ❑
SEP 88	Education	$3.95 ❑
OCT 88	Graphics	$3.95 ❑
NOV 88	Data Comm.	$3.95 ❑
DEC 88	Holiday	$3.95 ❑
JAN 89	Beginners	$3.95 ❑
FEB 89	Home Help	$3.95 ❑
MAR 89	Hardware	$3.95 ❑
APR 89	Business	$3.95 ❑
MAY 89	Printer	$3.95 ❑
JUN 89	Summer Fun	$3.95 ❑
JUL 89	Anniversary	$3.95 ❑
	VOLUME 9	
AUG 89	Beyond BASIC	$3.95 ❑
SEP 89	Education	$3.95 ❑
OCT 89	Graphics	$3.95 ❑
NOV 89	Data Comm.	$3.95 ❑
DEC 89	Holiday	$3.95 ❑
JAN 90	Beginners	$3.95 ❑
FEB 90	Home Help	$3.95 ❑
MAR 90	Hardware	$3.95 ❑
APR 90	Business	$3.95 ❑
MAY 90	Printer	$3.95 ❑
JUN 90	Summer Fun	$3.95 ❑
JUL 90	Anniversary	$3.95 ❑
	VOLUME 10	
AUG 90	OS-9	$3.95 ❑
SEP 90	Education	$3.95 ❑
OCT 90	Graphics	$3.95 ❑
NOV 90	Data Comm.	$3.95 ❑
DEC 90	Holiday	$3.95 ❑
JAN 91	Beginners	$3.95 ❑
FEB 91	Home Help	$3.95 ❑
MAR 91	Hardware	$3.95 ❑
APR 91	Music	$3.95 ❑
MAY 91	Printer	$3.95 ❑
JUN 91	Summer Fun	$3.95 ❑
JUL 91	Anniversary	$3.95 ❑
	VOLUME 11	
AUG 91	Graphics	$3.95 ❑
SEP 91	Education	$3.95 ❑
OCT 91	OS-9	$3.95 ❑
NOV 91	Data Comm.	$3.95 ❑
DEC 91	Holiday	$3.95 ❑
JAN 92	Utilities	$3.95 ❑
FEB 92	Home Help	$3.95 ❑
MAR 92	Hardware	$3.95 ❑
APR 92	Music	$3.95 ❑
MAY 92	Printer	$3.95 ❑
JUN 92	Programming	$3.95 ❑
JUL 92	Anniversary	$3.95 ❑
	VOLUME 12	
AUG 92	Graphics	$3.95 ❑
SEP 92	Education	$3.95 ❑
OCT 92	OS-9	$3.95 ❑
NOV 92	Data Comm.	$3.95 ❑
DEC 92	Holiday	$3.95 ❑
JAN 93	Utilities	$3.95 ❑
FEB 93	Home Help	$3.95 ❑

T&D Software : Our Final Frontier!

Our ten year mission has been to seek out new Coco software and boldly provide the biggest and best variety of Coco programs. Our mission is complete! Our subscription service has been stopped. This is your final chance to order!

Our Last ten issues featured here.

ISSUE #111, SEPT. 1991	ISSUE #112, OCT. 1991	ISSUE #113, NOV. 1991	ISSUE #114, DEC. 1991	Issue #115 January 1992
PERSONAL FILE	BOWLER'S HELPER	GRADE BOOK	BORDER DESTROYER	FLIES
QUEST OF CHAOS	POOL PLAYER	BINGO CALLER	LOW RES 1+2	SPEECH HINTS
RETIREMENT ACCT.	MUSIC MAN	CAVERNS OF OH NO	THE CASTLE	MOUNTAIN WARS
ALPHABET SCRAM. 3	GUEST LIST	BEAUTY PAGEANT	SKETCH IV	SUP. COPY/KILL
POKER 3	SERPENT	TELESALE	ENEMY ATTACK	DARN
FINANCE BUNDLE	FOWARD DICE	CRYPTO	MATH SOLVER 2	MAZE
SILENCE SYNDROME	COCO TOOLS	CRIONAUNT WARRIOR	DREAM ESCAPE AD.	DISK DIR.
MR. BILL	PAKFIX	PICTURE 3	TEST MAKER	BACKUP
COAST TO COAST	GNOME QUEST	SOLITAIRE PUZZLE	CODES 2, CODES 3	ROMAN CHECKER
GOPHER 3	BIRDS	FROGGY	MUNCHMAN	GNOME QUEST 2
Issue #116 February 1992	**Issue #117 March 1992**	**Issue #118 April 1992**	**ISSUE #119 MAY 1992**	**ISSUE #120 JUNE 1992**
CHOPPER	BOXES	VOCAB TESTER	LARGE	NAMEFILE
RICHER OR POOR	TIC TAC TOE 3	NURESHED	PICTURE MAKER	COMICS
MENU MAKER	MENU	EYE WITNESS	SQUARE	ALPHABET SOUP
THE CASTLE	MUNCHER	TEMP. CONV	DISK CHECKER	BUDGET
HELP!!	RIDDLE	GREEN HOUSE	HINTS	LABELER
B.BALL DBASE	LOTTOGAME	ADDITION TEST	LOTTO PREDICTIONS	KAYAK ADV.
SNOW MOBILE	HCOPY 128-512	DEMON FIRE	POCKETBALL	JB-ROBOT
ASTROIDSMASH	RAHUL'S QUEST 3	UFO HUNTER	RAMTEST	DESTROYER
DAFFI-NITION	CARS TYPEING	SHADOW WORLD	COCO DIL MGR.	SATELITE
DREAMQUEST 3	PROTECT	SHOWDOWN	MUTANT ATTACK	CHRISTMAS CARD

Klingon to our back issues of subscription software

*All issues contain 10 ready to run programs.
*All original software includes documentation.
*We send first class! Free shipping!
*Personal checks welcome.
*Write for a free catalog.
*Programs are for a Coco 1, 2, or 3.

RAINBOW CERTIFICATION SEAL

It is logical to order some of these Public Domain disks now!

53 Public Domain Disks

Utilities 1-8
12 Programs Each. U1-U4 Require Disk
U1 - [illegible], Disk Zapper, [illegible] Copy, Basic Map, +
U2 - [illegible], Disk [illegible], Customize, Offset, [illegible], +
U3 - Convert, Disk Library, [illegible], +
U4 - [illegible], Disk [illegible], [illegible], +
U5 - Assembler, Basic 64, [illegible], Recover, +
U6 - 64 Look, HGR Colors, LLIST 32, Renovate, +
U7 - Letter head print with 30 mini pictures
U8 - Fig Forth language with tutorial!

Telecommunications 1-3
T1 - [illegible]
T2 - [illegible] Terminal Package
T3 - [illegible] Communications

Music 1-7
M1 - 8 Utilities & 8 Songs
M2 - 17 Musica Files
M3 - 16 Musica Files
M4 - 16 Musica Files
M5 - 25 Orchestra Files
M6 - 23 .Bin Files ready to play
M7 - 23 .Bin Files ready to play

Adventures 1,2
A1 - CIA, Death Ship, Gargoyle, Kingdom, [illegible] - Odyssey, [illegible], Werewolf, [illegible]
A2 - [illegible], Ghostship, [illegible] - [illegible], Quest, Sorcerer, Survival, Adv. Gen.

Graphics 1-14
GR1 - Rainbow, Celtic, Space +
GR2 - Donald, Snoopy, Workdman, +
GR3 - 9 Coco 3 Graphic programs.
GR4 - 22 Coco Max Pictures
GR5 - 22 Coco Max Pictures
GR6 - 22 Coco Max Pictures
GR7 - 13 Coco Max Pictures
GR8 - 22 .BIN Pictures
GR9 - 22 .BIN Pictures
GR10 - 14 Large .BIN Pictures
GR11 - 8 MGE Pictures
GR12 - Coco Max 3 Pictures
GR13 - Macpaint Graphic Editor
GR14 - 5 Macintosh Pictures

HOME MANAGEMENT 1-4
*12 Programs Each Disk/Tape
H1 - Chkbook, Webpage, Typing Tutor, +
H2 - Check, Last Will, Word Processor, +
H3 - Envelope, Life, Finance, Interest Rate, +
H4 - Speeding Fin, Spelling Checker, +

Education 1-4
E1 - 12 Programs for young kids.
E2 - 12 Programs for High School Kids
E3 - 11 Programs Teach the Coco's Commands
E4 - 5 Graphic Programs About Australia

GAMES 1-11
*Each Disk/Tape Contains 12 Programs *
GA1 - [illegible], Robotics, Startrek +
GA2 - Amazing, Rider, Rabble, Vapor worm, +
GA3 - [illegible], Pythons, Robots, +
GA4 - Hawks, Lunar World, Saucer, Shooters, +
GA5 - Battleship, Chicken, Raceway, Lines, +
GA6 - Kings Army, Navygame, Shipsub, StarFix, +
GA7 - Connect 4, F-14, Pizza, Rabble, Life +
GA8 - Football, Germ Attack, Othello, Slither, +
GA9 - Battle, 3DTTT, Max Graphics, Trio, +
GA10 - Blockade, Life, Missile2, Scramble, +
GA11 - Bunker 3, Tank, Nuke Attack, Tracers, +

70 GRAPHIC DISKS!

Set #1	***Set #2***	***Set #3***	***New Sets 4,5,6,7***
Clipart Animals More	*Celebrities Cartoons More*	*R-Rated Women Adults Only*	*Each set has a variety of: Business Clipart. Grin Graphics. Sports. Special Events Plus much more!*

VISA MasterCard

Prices:

Single Issue (Tape or Disk)...	$ 8.00	**$6.00**
2-5 ISSUES..........	$ 6.00	**$5.00 Ea.**
6-10 Issues..........	$ 5.00	**$4.50 Ea.**
11 or more Issues..........	$ 4.50	**$4.00 Ea.**
Graphic Sets..........	$20	**$15 Ea.**
All 120 Single Issues..........	$250	**$125**
All 53 Public Domain Disks.....	$145	**$75**
All Graphic Stes..........	$140	**$70**
All Our Software(243Disks)...	$379.00	**$199**

No Bones about it. These are the best prices yet!

Original Software								Public Domain					Graphic Disks Packages
1	16	31	46	61	76	91	106						
2	17	32	47	62	77	92	107	M1	GR1	E1	U1	GA1	
3	18	33	48	63	78	93	108	M2	GR2	E2	U2	GA2	
4	19	34	49	64	79	94	109	M3	GR3	E3	U3	GA3	Set 1
5	20	35	50	65	80	95	110	M4	GR4	E4	U4	GA4	Set 2
6	21	36	51	66	81	96	111	M5	GR5	U5		GA5	Set 3
7	22	37	52	67	82	97	112	M6	GR6	H1	U6	GA6	Set 4
8	23	38	53	68	83	98	113	M7	GR7	H2	U7	GA7	Set 5
9	24	39	54	69	84	99	114		GR8	H3	U8	GA8	Set 6
10	25	40	55	70	85	100	115	A1	GR9	H4		GA9	Set 7
11	26	41	56	71	86	101	116	A2	GR10			GA10	
12	27	42	57	72	87	102	117		GR11			GA11	
13	28	43	58	73	88	103	118	T1	GR12				
14	29	44	59	74	89	104	119	T2	GR13				
15	30	45	60	75	90	105	120	T3	GR14				

Warp Speed them orders in To:

T&D Subscription Software
2490 Miles Standish Drive
Holland MI 49424
(616) 399-9648
Fax: (616) 396-2744

Name ________
Address ________
City ________ State ____ Zip ____
Credit Card # ________
Expiration ________ Total $ ________
Please Circle: TAPE or DISK

Feature Program

Alphabet Unscrambler

by Keiran Kenny

Alphabet is a short game intended for young children just learning the alphabet. Parental or teacher gnidance may be necessary to reinforce the learning process. Alphabet requires at least 16K and Extended BASIC.

When Alphabet is run, the alphabet appears in on the PMODE4 screen in big letters (32 pixels wide and 30 pixels high), but all the letters are out of order. The goal is to get them in the right order. To do this, the user moves the cursor until it surrounds the letter A. Then he presses the space bar, moves to where the letter A should be in the alphabet, then presses the space bar again. The A changes places with whatever letter was already in that position. With the A in the right place, the user follows the same procedure with B, then C, and so on, until the alphabet is properly organized. When all 26 letters are in the right place, the user should press ENTER, then he may choose whether to play again or end the game.

As an alternative to using the arrow keys, pressing J when prompted at the title screen enables joystick control of the cursor. This may be easier for some users.

I designed the program in such a way that the movement of the cursor box is fairly slow. This is intended to accommodate the reaction capacity of a younger child. To increase the speed of the cursor box, reduce the value of 500 (Variable DT) in Line 40.

If you are not interested in using the program to unscramble the alphabet, delete lines 50 through 100, 170 through 470 and 520 through 540, then rewrite the drawing portion to display your owu messages on the screen in big letters. However, make sure you add a stop line such as

```
990 POKE SL,0:GOTO990
```

The character set, which appears in lines 1000 through 1400, includes all letters and numbers, a space, a period, an exclamation sign, a dollar sign and an apostrophe. The strings for the characters are stored in Array L$, and the ASCII value of the characters are the subscripts to the array.

Yon can nse up to eight characters in each screen line for a total of five lines on the screen. Characters I, 1, the period and the exclamation sign occupy less space, so you could get away with more characters if a line includes any of these.

Follow the examples in lines 470 through 510. Variable B sets the horizontal position and C dictates the vertical. The characters are drawn upward, so C must be set to at least 30 to draw on the top line. The vertical coordinate of each subsequent line should be set to not less than 32 points more than that used for the previous line. I used 40 when writing Alphabet.

As with most of my other programs, the high-speed mode is used in Alphabet.

Line 30 sets the values for variables SP and SL to suit either a CoCo 3 or an earlier model of the Color Computer. The operating speed is returned to normal when the user presses E on the end screen to end the program.

Keiran Kenny's interests lie mainly with the Color Computer's graphics and math capabilities. But in his own words, "I like to try everything." He may be contacted at van Montfoortlaan 31, 2596 SP The Hague, Holland. Please include a self-addressed envelope with sufficient postage when requesting a reply.

16K ECB

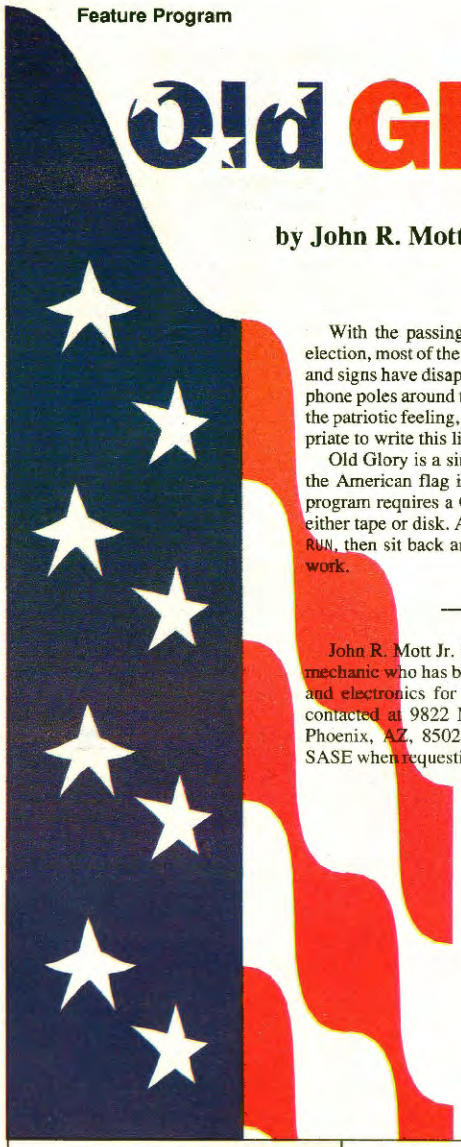
The Listing: ALPHABET

```
1 'ALPHABET UNSCRAMBLER
2 'BY KEIRAN KENNY
3 'COPYRIGHT (C) 1993
4 'BY FALSOFT, INC.
5 'RAINBOW MAGAZINE
10 CLS:CLEAR500
20 DIMC(25),D(25),L$(90),LN(90)
30 IFPEEK(33021)=50 THENSP=65497
:SL=65496ELSESP=65495:SL=65494
40 POKESP,0:DT=500
50 PRINT@38,''ALPHABET UNSCRAMBLE
R'
60 PRINT@128,''BY KEIRAN KENNY, T
HE HAGUE, 1992"
70 PRINT@225,''(A)RROWS OR RIGHT
(J)OYSTICK?'';PRINT@298,''PRESS A
OR J''
80 K$=INKEY$:IFK$<>''A''ANDK$<>''J''
THEN80
90 IFK$=''J''THENJY=1
100 GOTO140
110 FORDL=1TO100:NEXT:RETURN
120 DRAW''BM''+STR$(B)+'',''+STR$(C)
130 FORT=1TOLEN(W$):IFMID$(W$,T,
1)=''I''THENDRAW''BL12'':DRAWL$(ASC(
MID$(W$,T,1)))+''BR3'':NEXTELSE:DR
AWL$(ASC(MID$(W$,T,1)))+''BR3'':NE
XT:RETURN
140 PMODE4,1:COLOR0,5:PCLS:SCREE
N1,1
150 GOTO1000
160 H=8:V=39
170 FORZ=65TO90:LN(Z)=Z:NEXT
180 FORT=65TO90
190 R=64+(INT(RND(-TIMER)*26)+1)
200 IFLN(R)=0THEN190
210 DRAW''BM=H;,=V;''+L$(R)
220 LN(R)=0
230 H=H+40
240 IFH>208THENH=8:V=V+32
250 NEXT
260 A=6:B=8
270 IFJY THEN330
280 IFPEEK(341)=247THENB=B-32
290 IFPEEK(342)=247THENB=B+32
300 IFPEEK(343)=247THENA=A-40
310 IFPEEK(344)=247THENA=A+40
320 GOTO380
330 J0=JOYSTK(0):J1=JOYSTK(1)
340 IFJ0<63THENA=A-40
350 IFJ0>0THENA=A+40
360 IFJ1<31THENB=B-32
370 IFJ1>0THENB=B+32
380 IFA<6THENA=6
390 IFA>206THENA=206
400 IFB<8THENB=8
410 IFB>=136ANDA>46THENB=136:A=4
6ELSEIFB>136THENB=136
420 LINE(A,B)-(A+32,B+32),PSET,B
:FORDL=1TODT:NEXT:LINE(A,B)-(A+3
2,B+32),PRESET,B
430 IFP=0 ANDPEEK(135)=32THENP=1
:X1=A+2:X2=A+33:Y1=B+1:Y2=B+32:G
ET(X1,Y1)-(X2,Y2),C:POKE135,0
440 IFP ANDPEEK(135)=32THENX3=A+
2:X4=A+33:Y3=B+1:Y4=B+32:GET(X3,
Y3)-(X4,Y4),D:PUT(X3,Y3)-(X4,Y4)
,C:PUT(X1,Y1)-(X2,Y2),D:MV=MV+1:
P=0:POKE135,0
450 IFPEEK(135)=13THEN470
460 IFJY THEN330ELSE280
470 PCLS:B=0:C=31:W$=STR$(MV)+''
MOVES!'':GOSUB120
480 B=0:C=71:W$=''HIT 'M' FOR'':GO
SUB120
490 B=48:C=111:W$=''MORE!'':GOSUB1
20
500 B=20:C=151:W$=''HIT 'E' TO'':G
OSUB120
510 B=0:C=191:W$=''END GAME.'':GOS
UB120
520 K$=INKEY$:IFK$<>''M''ANDK$<>''E
''THEN520
530 IFK$=''M''THENMV=0:PCLS:GOTO16
0
540 POKESL,0:CLS:END
1000 L$(32)=''BR5"
1010 L$(33)=''BRHU3ENR3BU3HU20ER3
FD20GNL3BD3FD3GNL3BR''
1020 L$(36)=''BU3BR4H4U3R5D2F2R5B
R4R6E2U2H2L6BL4L10H2U9E4R8BR4R8F
4D3L5U2H2L5BL4L6G2D5F2R6BR4R8F4D
4G4L8D3L4U3BU3U6BU3U9BU3U3R4D3BD
3D9BD3D6BD3BL4L8BR24BD3"
1030 L$(39)=''BU25U5R5D5G3L2E3L3B
R5BD25"
1040 L$(46)=''BRHU3ER3FD3GL2BR3"
1050 L$(48)=''BR4H4U22E4R20F4D22G
4L20BE5BH4NM+14,-15U12E4R10BF4NM
-14,+15D12G4L10BD5BR19"
1060 L$(49)=''BR2U28L2E2R5D30NL5"
1070 L$(50)=''NR28U10E4R16E4U4H4L
11G4D2L4U5E5R18F5D10G5L17G3D3R24
D4"
1080 L$(51)=''BR4H4U6R5D4F3R12E3U
3H3L10U4R10E3U4H3L12G3L5U3E4R20F
4D9G3F3D7G4NL20BR4"
1090 L$(52)=''BR20U10L20U5M+20,-1
5BD5D11L14M+14,-11BU5R5D16R3D4L3
D10L5BR8"
1100 L$(53)=''BR4H4U6R5D4F3R12E3U
7H3L20U14R28D5L23D5R19F4D12G4L20
BR24"
1110 L$(54)=''BR4H4U22E4R20F4D6L5
U4H3L12G3D10E3R16F4D9G4L19BE5BL2
H2U2E2R14F2D2G2L14BD5BR20''
1120 L$(55)=''U4M+23,-21L23U5R28D
8M-20,+18D4L8BR28"
1130 L$(56)=''BR4H4U7E4H4U7E4R20F
4D7G4F4D7G4L20BE3H2U5E2R14F2D5G2
L14BU15H2U5E2R14F2D5G2L14BD18BR2
1"
1140 L$(57)=''BR4H4U4R5D2F3R12E3U
6BU6U5H3L12G3D5F3R12E3BM-19,+18R
20E4U22H4L20G4D12F4R16E3BD13BR5"
1150 L$(65)=''M+11,-30R6M+11,+30L
6M-3,-8NL8BM-2,-6M-1,-5L3M-1,+5N
R4BM-2,+6M-3,+8L6BR28"
1160 L$(66)=''U30R23F5D5G5F5D5G5L
23BE5U8R15F2D4G2L15BU13U8R15F2D4
G2L15BD18BR23"
1170 L$(67)=''BR4H4U22E4R20F4G4H3
L13G3D13F3R12E3F5G4L19BR23"
1180 L$(68)=''U30R23F5D20G5L23BM+
6,-5U20R13F3D14G3L13BD5BR22"
1190 L$(69)=''U30R28D5L23D7R20D5L
20D8R23D5NL28"
1200 L$(70)=''U30R28D5L23D7R20D5L
20D13NL5BR23"
1210 L$(71)=''BR4H4U22E4R20F4G4H3
L13G3D15F3R12E3L3U3R8D6G4NL20BR4
'
1220 L$(72)=''U30R5D12R18U12R5D30
L5U13L18D13L5BR28"
1230 L$(73)=''BR12U30R5D30NL5"
1240 L$(74)=''BR4H4UBR5D4F3R12E3U
22R5D26G4NL20BR4"
1250 L$(75)=''U30R5D12M+18,-12R5D
4M-12,+8M+12,+14D4L5M-13,-15M-5,
+4D11L5BR28"
1260 L$(76)=''U30R5D25R23D5NL28"
1270 L$(77)=''U30R6FBE8R6D30L6U22
G8H8D22L6BR28"
1280 L$(78)=''U30R5M+18,+12U12R5D
30L5U13M-18,-12D25L5BR28"
1290 L$(79)=''BR4H4U22E4R20F4D22G
4L20BE5H4U12E4R10F4D12G4L10BD5BR
20"
1300 L$(80)=''U30R23F5D5G5L18BU5U
5R15F2DG2L15BD5D15L5BR28"
1310 L$(81)=''BR4H4U22E4R20F4D16G
5M+5,+5L4M-3,-3G3L14BM+5,-4BL2H3
U16E3R15F3D10G5L6D3R3G2L7BD3BR20
'
1320 L$(82)=''U30R23F5D5G5L8M+13,
+15L7M-13,-15L3BU5U5R15F2DG2L15B
D5D15L5BR28"
1330 L$(83)=''BR4NR20H4U5R5D2F3R1
1E4U2H3L16H4U10E3R22F3D5L5U2H2L1
3G3D2F3R16F4D11G3BR3"
1340 L$(84)=''BR11U25L11U5R28D5L1
1D25L6BR17"
1350 L$(85)=''BR4H4U26R5D22F4R10E
4U22R5D26G4NL20BR4"
1360 L$(86)=''BR11M-11,-30R5M+9,+
25M+9,-25R5M-11,+30L5BR16"
1370 L$(87)=''U30R5D25M+9,-15M+9,
+15U25R5D30L8M-6,-9M-6,+9L8BR28"
1380 L$(88)=''U5M+9,-10M-9,-10U5R
5M+9,+10M+9,-10R5D5M-9,+10M+9,+1
0D5L5M-9,-10M-9,+10L5BR28"
1390 L$(89)=''BR11U10M-11,-15U5R5
M+9,+13M+9,-13R5D5M-11,+15D10L6B
R17"
1400 L$(90)=''U5M+22,-20L22U5R28D
7M-21,+18R2105NL28"
1410 GOTO160
```

Feature Program

Old Glory

by John R. Mott Jr.



With the passing of the recent presidential election, most of the red, white and blue banners and signs have disappeared from yards and telephone poles around town. To keep in touch with the patriotic feeling, I thought it might be appropriate to write this little BASIC program.

Old Glory is a simple program that displays the American flag in all its glory. Though the program reqnires a CoCo 3, you can save it on either tape or disk. After doing so, simply enter RUN, then sit back and watch the CoCo 3 go to work.

John R. Mott Jr. is a retired truck driver and mechanic who has been interested in computers and electronics for several years. He may be contacted at 9822 North 15th Street, Apt. B, Phoenix, AZ, 85020-1810. Please include an SASE when requesting a reply.

16K ECB

The Listing: OLDGLORY

```
1 'AMERICAN FLAG
2 'BY JOHN R. MOTT JR.
3 'COPYRIGHT (C) 1993
4 'BY FALSOFT, INC.
5 'RAINBOW MAGAZINE
10 PCLEAR4:WIDTH80
20 HSCREEN4
30 HCLS5
40 PALETTERGB:PALETTE0,63:PALETTE1,36
50 HCOLOR0,1
60 HLINE(0,5)-(640,187),PSET,B:HLINE(220,5)-(220,103),PSET:HLINE(0,103)-(220,103),PSET:HLINE(220,19)-(640,19),PSET:HLINE(220,33)-(640,33),PSET:HLINE(220,47)-(640,47),PSET:HLINE(220,61)-(640,61),PSET:HLINE(220,75)-(640,75),PSET
70 HLINE(220,89)-(640,89),PSET:HLINE(220,103)-(640,103),PSET:HLINE(0,117)-(640,117),PSET:HLINE(0,131)-(640,131),PSET:HLINE(0,145)-(640,145),PSET:HLINE(0,159)-(640,159),PSET:HLINE(0,173)-(640,173),PSET
80 HPAINT(320,12),3,12:HPAINT(320,26),12,12:HPAINT(320,40),3,12:HPAINT(320,54),12,12:HPAINT(320,68),3,12:HPAINT(320,82),12,12:HPAINT(320,96),3,12:HPAINT(320,110),12,12
90 HPAINT(320,124),3,12:HPAINT(320,138),12,12:HPAINT(320,152),3,12:HPAINT(320,164),12,12:HPAINT(320,180),3,12:PALETTE1,9:HPAINT(136,54),1,12
100 FORX=8TO198STEP38
110 FORY=14TO94STEP20
120 HLINE(X,Y)-(X+5,Y),PSET:HLINE(X+5,Y)-(X+7,Y-5),PSET:HLINE(X+7,Y-5)-(X+9,Y),PSET:HLINE(X+9,Y)-(X+14,Y),PSET:HLINE(X+14,Y)-(X+9,Y+2),PSET:HLINE(X+9,Y+2)-(X+14,Y+5),PSET:HLINE(X+14,Y+5)-(X+7,Y+3),PSET
130 HLINE(X+7,Y+3)-(X,Y+5),PSET:HLINE(X,Y+5)-(X+5,Y+2),PSET:HLINE(X+5,Y+2)-(X,Y),PSET:HPAINT(X+7,Y),12,12
140 NEXTY,X
150 FORH=27TO179STEP38
160 FORV=24TO84STEP20
170 HLINE(H,V)-(H+5,V),PSET:HLINE(H+5,V)-(H+7,V-5),PSET:HLINE(H+7,V-5)-(H+9,V),PSET:HLINE(H+9,V)-(H+14,V),PSET:HLINE(H+14,V)-(H+9,V+2),PSET:HLINE(H+9,V+2)-(H+14,V+5),PSET:HLINE(H+14,V+5)-(H+7,V+3),PSET
180 HLINE(H+7,V+3)-(H,V+5),PSET:HLINE(H,V+5)-(H+5,V+2),PSET:HLINE(H+5,V+2)-(H,V),PSET:HPAINT(H+7,V),12,12
190 NEXTV,H
200 PALETTE2,54:HLINE(0,0)-(640,5),PSET,B:HPAINT(320,2),14,12:HLINE(0,187)-(640,192),PSET,B:HPAINT(320,189),14,12
210 GOTO210
220 'REM 1317 BYTES
```

VEF Graphics *from Page 1*

produce colors, all palette values need to be converted from their RGB values. Although the conversion does not create exact equivalents, the results should be close enough for most images. Still, these values can always be edited with the Change Palette option found on *VEFit*'s Main menu.

Menu Options

To select an option from a *VEFit* menu, use the arrow keys to point to the desired option, then press ENTER. To exit *VEFit* and return to BASIC, press BREAK. To return to the Main menu from a submenu, press Q. Errors encountered while the program is running are reported, but they may result in altered palette values due to BASIC's error-handling routines. If such a situation arises, you may need to use the Change Palette option to reset the proper palette values. Following are *VEFit*'s Main menu options:

Load Picture: *VEFit* searches for filenames with a .VEF extension. If none are found, you are prompted to return to the Main menu. Otherwise a list is displayed for selection. Since a regular disk can hold up to only nine images (depending on their resolution), a maximum of 16 filenames are displayed onscreen. Once you make a selection, the filename and a status bar are displayed to indicate loading progress. When loading is complete, the picture is displayed in the resolution specified in the VEF file. When you finish viewing the picture, press any key to return to the file selection menu. Loading time varies from 45 to 90 seconds, depending on picture resolution.

Change Palette: The 16 current palette values are displayed for editing. Upon choosing a palette to edit, use the up-arrow to increase or down-arrow to decrease the palette value. After you load a picture, you may toggle between the graphics and text modes by pressing the space bar. This toggle feature is also active after you select a default HSCREEN with the Memory Display option below. In the graphics mode, changes to the palette are seen onscreen as you edit. Press ENTER to record the changes and return to the palette selection menu.

Save (four menu choices): Before saving, ensure that the image to be saved is in HSCREEN memory and that the proper palette values are set. You can use the Memory Display option to preview the picture first. After you select the appropriate Save option from the Main menu, the disk is checked for free space. If there is sufficient disk space, you are asked to enter a filename, or you can press ENTER to abort the save. If a duplicate filename exists on the disk, you are prompted to abort the save or overwrite the existing file. After passing all these checkpoints, the filename and a status bar are displayed to indicate saving progress. When saving is complete, you are returned to the Main menu. Save times vary from 20 to 45 seconds, depending on the save resolution you choose.

Memory Display: This option allows you to view HSCREEN memory in four differ-

VEF Graphics *continued on Page 12.*

```
00100          ORG $5FDA
00110          LDY 2,X        Make Y: 2+X (address of STRING)
00120          LDX #$6000     Make X: Address of memory block
00130          INC COUNT      Increase count for # 128 bytes
00140          LDA COUNT      Load count...
00150          CMPA #$40      Has 64 (64x128 = 8192) been read?
00160          BNE P2         If not, jump to P2
00170          CLR COUNT      Else clear count for begin screen
00175          DECA           And decrease for last time around
00180 P1       LEAX 128,X     Make X: X+128 bytes/find right line
00190 P2       DECA           Decrease count
00200          BNE P1         Loop back to P1 if not done adding
00210          LDA #$80       Load 128 for counter
00220 P3       LDB ,X+        Load byte from screen memory
00230          STB ,Y+        Store in return string
00240          DECA           Decrease count
00250          BNE P3         Done? If not, back to P3 for next byte
00260          RTS            Return to BASIC
00270 COUNT    RMB 1          Counter of 128 multiples
00280          END $5FDA
```

Figure 2: VEFIT **Assembly-Language Routine**

```
00100          ORG $5FBA
00110          JSR $B3ED      Put USR(PRINT@) location in D
00120          ADDD #$0400    Add $0400 to point to screen memory
00130          TFR D,X        Transfer D into X
00140          LDA #$08       Characters to hilite (POKE &H5FC3,#)
00150 H1       LDB ,X         Read character from screen memory
00160          CMPB #$7F      If video code>127 then...
00170          BHI H3         Leave and write back to screen (H3)
00180          CMPB #$3F      If video code>63 (capital) then...
00190          BHI H2         (-)64 from video code to lowercase (H2)
00200          ADDB #$40      Else (+)64 to video code to uppercase...
00210          BRA H3         And write to screen (H3)
00220 H2       SUBB #$40      Subtracts 64 from video code
00230 H3       STB ,X+        Write character to screen
00240          DECA           Decrease count to hilite
00250          BNE H1         Done? If not, back to H1 to convert next
00260          RTS            Return to BASIC
00270          END $5FBA
```

Figure 3: HILITE **Assembly-Language Routine**

CoCo 3 Disk

The Listing: VEFIT

```
1 'VEFIT
2 'BY THOMAS WONG
3 'COPYRIGHT (C) 1993
4 'BY FALSOFT, INC.
5 'RAINBOW MAGAZINE
10 '==========================
    vefIT 1.1:RSDOS PIX UTILITY
   ==========================
12 WIDTH 32:CLEAR 1500,&H5FB9:PO
KE &HE6C6,&H33
14 DIM PA(15),AA$(2),AB$(68):LA=
Ø:MA=2
16 AA=1:AA$="BDB3EDC304001F01860
8E684C17F220AC13F2204CB402002C04
ØE7804A26EB3910AE028E60007C5FFFB
65FFF814026087F5FFF4A308900804A2
6F98680E680E7A04A26F939"
18 FOR AB=&H5FBA TO &H5FFE
20 AC=VAL("&H"+MID$(AA$,AA,2)):P
OKE AB,AC:AA=AA+2
22 NEXT AB
24 DEFUSRØ=&H5FBA:DEFUSR1=&H5FDA
26 ON BRK GOTO 458:ON ERR GOTO 4
52
28 AA$=CHR$(174)+STRING$(30,172)
+CHR$(173)+CHR$(170)+STRING$(30,
32)+CHR$(165)+CHR$(171)+STRING$(
30,163)
30 LA$="ØØØCØ2ØEØ7Ø9Ø51Ø1C2CØØ1D
ØB1BØA2B2211122103Ø113321E2D1F2E
ØF3C2F3D17Ø815Ø627162636192A1A3A
1829283B14Ø4233325352434203B313E
3739303F"
32 SA$="ØØ15Ø2143106230421050E0C
Ø1ØAØ31CØ7111216302225202C282AØD
Ø8ØB181A3813103236342624 2E2D29ØF
Ø9191B1E3E3A17333735273C2F3D2B39
1D1F3B3F"
34 CLS:PRINT@Ø,AA$;CHR$(167):PRI
NT@416,AA$;:POKE &H5FF,&HA7:PRIN
T@33,"vefIT - RSDOS PICTURE UTI
LITY";
36 PRINT@449,"CHOOSE A MONITOR T
YPE TO BEGIN";
38 GOSUB 320:GOSUB 388
40 GOSUB 408
42 PRINT@131,"LOAD A PICTURE":PR
INT@163,"PALETTE CHANGE":PRINT@1
95,"SAVE [320 X 200: 4 COLORS]":
PRINT@227,"SAVE [320 X 200: 16 C
OLOR]":PRINT@259,"SAVE [640 X 20
Ø: 2 COLORS]":PRINT@291,"SAVE [6
40 X 200: 4 COLORS]"
44 PRINT@323,"MEMORY DISPLAY":PR
INT@355,"MONITOR RESET":PRINT@44
9,"[ARROWS] TO SELECT AND [ENTER
]";
46 AD=1:AE=8:AF=28:AG=130:GOSUB
358
48 IF AB$<>CHR$(13) THEN 46:ELSE
 GOSUB 388
50 ON AJ GOSUB 56,148,196,196,19
6,196,294,318
52 GOSUB 388:GOTO 42
54 '* LOAD PIX *
56 PRINT@449,"[ARROWS] SELECT FI
LE OR [Q]UIT";
58 GOSUB 428
60 IF AN=Ø THEN SOUND 200,1:PRIN
T@228,"NO PICTURES ON THIS DISK"
:PRINT@260,"[PRESS ANY KEY FOR M
AIN]";:GOSUB 394:RETURN
62 IF AN>16 THEN AN=16
64 LB=Ø:LC=Ø
66 PRINT@(128+(LB*32)+((LC*16)+2
)),AB$((LB*2)+LC+1);
68 LC=LC+1
70 IF AN=((LB*2)+LC) THEN 74
72 IF LC=3 THEN LB=LB+1:LC=Ø:GOT
O 66:ELSE 66
74 AD=2:AE=8:AF=14:AG=129:AH=16:
GOSUB 358
76 IF AB$="Q" THEN RETURN
78 LD=((AJ-1)*2)+(AI-1)+1
80 IF LD<=AN THEN LB$=AB$(LD):GO
SUB 388:ELSE SOUND 200,1:PRINT@4
49,"INVALID SELECTION: PRESS A K
EY";:GOSUB 394:RETURN
82 PRINT@225,"LOADING PICTURE ["
;LB$;"]":PRINT@257,"STATUS: [";S
TRING$(20,191);"]":PRINT@449,STR
ING$(30,128);
84 OPEN "D",#1,LB$,1
86 FIELD #1,1 AS LC$
88 GET #1,2:LA=ASC(LC$):IF LA<3
THEN LE=250:LF=12:ELSE LE=125:LF
=7
90 LA=VAL(MID$("24013",LA+1,1))
92 FOR LG=3 TO 18
94 GET #1,LG:IF MA=Ø THEN PA(LG-
3)=VAL("&H"+MID$(LA$,(ASC(LC$)*2
)+1,2))+64:ELSE PA(LG-3)=ASC(LC$
)+64
96 NEXT LG
98 LD$=""
100 FOR LG=LOF(1)-17 TO LOF(1)
102 GET #1,LG:LD$=LD$+LC$
104 NEXT LG
106 CLOSE #1
108 IF LA=Ø THEN GOSUB 388:SOUND
 200,1:PRINT@227,"TYPE 2 VEF'S N
OT SUPPORTED":PRINT@259,"[PRESS
A KEY TO MAIN MENU]":GOSUB 394:R
ETURN
110 OPEN "D",#1,LB$,128
112 FIELD #1,128 AS LC$
114 LH=Ø:LI=48
116 POKE &H5FF8,&HAØ:POKE &H5FFA
,&H8Ø:POKE &H5FFF,Ø
118 GET #1,LF-1:LE$=RIGHT$(LC$,1
1Ø)
120 POKE &HFFA3,LI
122 IF LEN(LE$)>128 THEN LF$=LEF
T$(LE$,128):LE$=RIGHT$(LE$,LEN(L
E$)-128):ELSE 128
124 LF$=USR1(LF$):LH=LH+1
126 IF LH=64 OR LH=128 OR LH=192
 THEN LI=LI+1:GOTO 120:ELSE 122
128 IF LF>LE THEN LF$=USR1(LE$+L
D$):GOTO 140
130 LF$=LC$
132 GET #1,LF:LF=LF+1:LE$=LC$
134 LJ=128-LEN(LF$):LF$=LF$+LEFT
$(LE$,LJ):LE$=RIGHT$(LE$,LEN(LE$
)-LJ)
136 LK=INT((LF/LE)*20):PRINT@266
,STRING$(LK,175);
138 GOTO 124
140 CLOSE #1
142 GOSUB 418:HSCREEN LA:GOSUB 3
94:GOSUB 402:HSCREEN Ø
144 GOSUB 388:GOTO 56
146 '* PALETTES *
148 IF LA>Ø THEN PRINT@449,"[ARR
OWS] ROTATE [SPACE] [Q]UIT";:ELS
E PRINT@449,"[ARROWS] ROTATE PAL
ETTE [Q]UIT";
150 FOR PA=Ø TO 7
152 PB=PA(PA)-64:PA$=RIGHT$(STR$
(PB),2):IF PB<10 THEN MID$(PA$,1
,1)="Ø"
154 PB=PA(PA+8)-64:PB$=RIGHT$(ST
R$(PB),2):IF PB<10 THEN MID$(PB$
,1,1)="Ø"
156 PC$=RIGHT$(STR$(PA+8),2):IF
(PA+8)<10 THEN MID$(PC$,1,1)="Ø"
158 PRINT@(131+(PA*32)),"PAL Ø";
RIGHT$(STR$(PA),1);": ";PA$;"
   PAL ";PC$;": ";PB$
160 NEXT PA
162 AD=2:AE=8:AF=13:AG=130:AH=16
:GOSUB 358
164 IF AB$="Q" THEN RETURN
166 PC=USRØ(AG):PD=AJ+((AI-1)*8)
168 IF PD<9 THEN PE=139:ELSE PE=
155
170 PE=PE+((AJ-1)*32):PF=Ø
172 PG=PA(PD-1)-64:PD$=RIGHT$(ST
R$(PG),2):IF PG<10 THEN MID$(PD$
,1,1)="Ø"
174 PRINT@PE,PD$;
176 GOSUB 394
178 IF AB$="Q" THEN GOSUB 402:HS
CREEN Ø:RETURN
180 IF AB$=CHR$(10) THEN PG=PG-1
:IF PG<Ø THEN PG=63
182 IF AB$=CHR$(94) THEN PG=PG+1
:IF PG>63 THEN PG=Ø
184 PA(PD-1)=PG+64
186 IF AB$=CHR$(32) THEN PF=PF+1
:IF PF>1 THEN PF=Ø:GOSUB 402:ELS
E PF=LA
188 IF PF>Ø THEN GOSUB 418
190 HSCREEN PF
192 IF AB$<>CHR$(13) THEN 172:EL
SE GOSUB 402:HSCREEN Ø:PC=USRØ(A
G):GOTO 150
194 '* SAVE PIX *
196 IF AJ=3 OR AJ=5 THEN SA=7:EL
SE SA=14
198 IF FREE(Ø)<SA THEN SOUND 200
,1:PRINT@228,"NOT ENOUGH SPACE O
N DISK":PRINT@260,"[PRESS ANY KE
Y FOR MAIN]";:GOSUB 394:RETURN
200 PRINT@449,"TYPE FILENAME: [E
NTER] TO MAIN";:PRINT@227,"ENTER
 FILENAME: [         ]"
202 SA=Ø:SB$="":PRINT@244,CHR$(1
75);
204 GOSUB 394
206 IF AB$<>CHR$(8) THEN 214
208 IF SA>Ø THEN SA=SA-1:SB$=LEF
T$(SB$,SA):PRINT@244+SA,CHR$(175
);CHR$(32);:ELSE SOUND 200,1
210 IF SA=7 THEN PRINT@252,"]"
212 GOTO 204
214 IF AB$<>CHR$(13) THEN 218
216 IF SA=Ø THEN RETURN:ELSE 224
218 IF SA=8 THEN SOUND 200,1:GOT
O 204:ELSE SB$=SB$+AB$
220 PRINT@244+SA,AB$;:SA=SA+1:IF
 SA<8 THEN PRINT CHR$(175);
222 GOTO 204
224 IF INSTR(1,SB$,".")>Ø OR INS
TR(1,SB$,"/")>Ø THEN SOUND 200,1
:PRINT@449,"NO EXTENSIONS... PRE
SS ANY KEY";:GOSUB 394:GOTO 200
226 SB$=SB$+STRING$(8-SA,32)+".V
EF"
228 GOSUB 428
230 IF AN=Ø THEN 244
232 FOR SB=1 TO AN
234 IF SB$<>AB$(SB) THEN 242
236 SOUND 200,1:PRINT@449,"FILE
EXISTS: GO ON? [N]O [Y]ES";:GOSU
B 394
238 IF AB$<>"N" AND AB$<>"Y" THE
N 238
240 IF AB$="N" THEN 200:ELSE 244
242 NEXT SB
244 PRINT@225,"SAVING PICTURE: [
";SB$;"]":PRINT@257,"STATUS: [";
STRING$(20,191);"]":PRINT@449,ST
RING$(30,128);
246 AJ=AJ-2:IF AJ=2 OR AJ=4 THEN
 SC=240:SD=160:ELSE SC=120:SD=80
248 OPEN "D",#1,SB$
250 SE=VAL(MID$("3041",AJ,1))
252 PRINT #1,CHR$(Ø);CHR$(SE);
254 FOR SF=Ø TO 15
256 SG=PA(SF)-64
258 IF MA=Ø THEN SG=VAL("&H"+MID
$(SA$,(SG*2)+1,2))
260 PRINT #1,CHR$(SG);
262 NEXT SF
264 FOR SF=1 TO 8
266 PRINT #1,STRING$(SD,Ø);
268 NEXT SF
270 SH=Ø:SI=48
272 POKE &H5FF8,&H8Ø:POKE &H5FFA
,&HAØ:POKE &H5FFF,Ø
274 POKE &HFFA3,SI
276 SC$=USR1(STRING$(128,32))
278 PRINT #1,SC$;
280 SH=SH+1
282 IF SH=64 OR SH=128 OR SH=192
 THEN SI=SI+1:GOTO 274
284 SJ=INT((SH/SC)*20):PRINT@266
,STRING$(SJ,175);
286 IF SC>SH THEN 276
288 CLOSE #1
290 RETURN
292 '* MEMORIZE *
294 PRINT@449,"[ARROWS] SELECT O
R [Q]UIT MAIN";
296 IF LA=Ø THEN SOUND 200,1:PRI
NT@134,"NO HSCREEN SELECTED!":EL
SE PRINT@134,"CHOOSE AN HSCREEN.
.."
298 PRINT@195,"VIEW [320 X 200:
4 COLORS]":PRINT@227,"VIEW [320
X 200: 16 COLOR]":PRINT@259,"VIE
W [640 X 200: 2 COLORS]":PRINT@2
91,"VIEW [640 X 200: 4 COLORS]"
300 IF LA>Ø THEN PRINT@323,"VIEW
 DEFAULT HSCREEN:NOW";LA
302 AD=1:AF=28:AG=194:IF LA>Ø TH
EN AE=5:ELSE AE=4
304 GOSUB 358
306 IF AB$="Q" THEN RETURN:ELSE
GOSUB 418
308 IF AJ<5 THEN LA=AJ
310 HSCREEN LA
312 GOSUB 394:GOSUB 402:HSCREEN
Ø
314 GOTO 296
316 '* MONITORS *
318 PRINT@449,"[ARROWS] SELECT T
YPE OR [Q]UIT";
320 PRINT@128,"SELECT A MONITOR
TYPE: NOW = ";:IF MA=2 THEN PRIN
T "???":GOTO 324
322 IF MA=Ø THEN PRINT "CMP":ELS
E PRINT "RGB"
324 PRINT@198,"COMPOSITE COLOR (
TV)":PRINT@230,"RGB COLOR MONITO
R"
326 AD=1:AE=2:AF=22:AG=197:GOSUB
 358
328 IF MA<2 THEN 336
330 IF AB$="Q" THEN 326
332 IF AJ=1 THEN MA=Ø:CMP:ELSE M
A=1:RGB
334 GOSUB 408:RETURN
336 IF AB$="Q" THEN RETURN
338 IF AJ=2 THEN 348
340 IF MA=Ø THEN RETURN:ELSE MA=
Ø:CMP
342 FOR AL=Ø TO 15
344 PA(AL)=VAL("&H"+MID$(LA$,((P
A(AL)-64)*2)+1,2))+64
346 NEXT AL:RETURN
348 IF MA=1 THEN RETURN:ELSE MA=
1:RGB
350 FOR AL=Ø TO 15
352 PA(AL)=VAL("&H"+MID$(SA$,((P
A(AL)-64)*2)+1,2))+64
354 NEXT AL:RETURN
356 '* SELECTOR *
358 AI=1:AJ=1:POKE &H5FC3,AF
360 AK=USRØ(AG)
362 GOSUB 394:AK=USRØ(AG)
364 IF AB$=CHR$(8) THEN AI=AI-1:
IF AI<1 THEN AI=AI+1:ELSE AG=AG-
AH:GOTO 360
366 IF AB$=CHR$(9) THEN AI=AI+1:
IF AI>AD THEN AI=AI-1:ELSE AG=AG
+AH:GOTO 360
368 IF AB$=CHR$(94) THEN AJ=AJ-1
:IF AJ<1 THEN AJ=AJ+1:ELSE AG=AG
-32:GOTO 360
370 IF AB$=CHR$(10) THEN AJ=AJ+1
:IF AJ>AE THEN AJ=AJ-1:ELSE AG=A
G+32:GOTO 360
372 IF AB$="Q" THEN RETURN
374 IF AB$<>CHR$(13) THEN SOUND
200,1:GOTO 360
376 FOR AL=1 TO 6:AK=USRØ(AG):GO
SUB 382:NEXT AL
378 RETURN
380 '* PAUSES *
382 FOR AM=1 TO 20:NEXT AM
384 RETURN
386 '* CLS PART *
388 FOR AI=3 TO 12:PRINT@(AI*32)
,STRING$(32,32);:NEXT AL
390 RETURN
392 '* GET KEYS *
394 FOR AL=341 TO 344:POKE AL,25
5:NEXT AL
396 AB$=INKEY$:IF AB$="" THEN 39
6
398 RETURN
400 '* PAL BACK *
402 IF MA=Ø THEN CMP:ELSE RGB
404 RETURN
406 '* PAL READ *
408 FOR AL=Ø TO 15
410 PA(AL)=PEEK(&HFFBØ+AL)
412 NEXT AL
414 RETURN
416 '* PAL VIEW *
418 FOR AL=Ø TO 15
420 POKE &HFFBØ+AL,PA(AL)
422 NEXT AL
424 RETURN
426 '* DIR READ *
428 AN=Ø
430 FOR AO=3 TO 11
432 DSKI$ Ø,17,AO,AA$(1),AA$(2)
434 FOR AP=1 TO 2
436 AC$=AA$(AP)
438 FOR AQ=1 TO 128 STEP 32
440 AD$=MID$(AC$,AQ,8)+"."+MID$(
AC$,AQ+8,3)
442 IF MID$(AD$,1,1)=CHR$(255) T
HEN 448
444 IF MID$(AD$,1,1)<>CHR$(Ø) AN
D RIGHT$(AD$,3)="VEF" THEN AN=AN
+1:AB$(AN)=AD$
446 NEXT AQ,AP,AO
448 RETURN
450 '* TRAP ERR *
452 GOSUB 388:SOUND 200,1:PRINT@
224,"WARNING: ERROR HAS OCCURRED
 >";ERNO:PRINT@256,"[PRESS ANY K
EY: RETURNS TO MENU]"
454 UNLOAD:GOSUB 394:GOSUB 388:G
OTO 40
456 '* LEAVE-IT *
458 CLS:PRINT@5,"THANKS FOR USIN
G vefIT":UNLOAD:END
```

VEF Graphics *from Page 10*

ent resolutions. After making a selection, the HSCREEN is displayed. When you finish viewing, press any key to return to the display selection menu. Note that an additional option now appears: View default HSCREEN. This option allows you to reselect the most-recent viewing choice you made, as indicated. After using Load Picture, this option also appears for reviewing an image in its intended resolution. However, the default value changes each time you choose a different viewing option than View Default HSCREEN. You must use this option at least once to enable graphics mode toggling with the Change Palette option, as this is the default HSCREEN used.

Monitor Reset: This function is automatically performed after you start *VEFit*. However, if you made an error in selection or decide to switch the type of monitor being used, you may select this option. Choosing the wrong monitor type causes *VEFit* to interpret the palette values incorrectly when loading or saving a picture.

Program Notes

A description of the VEF format is shown in Figure 1. The various parts of *VEFit* are shown in figures 2 through 4. In reverse order, Figure 4 shows the various portions of the BASIC program, Figure 3 shows the HILITE machine-language subroutine used to reverse the video codes to produce a highlight effect, and Figure 2 shows the source code for VEFIT.ASM, another machine-language subroutine. VEFIT.ASM reads or writes 128-byte blocks of data in the address range $6000 through $7FFF.

Since the four 8K blocks of HSCREEN memory ($60000 through $67FFF, or blocks 48 through 51) are not mapped into BASIC's regular workspace, *VEFit* must use Page Address Register 3 at $FFA3 to map these blocks into the $6000-to-$7FFF range one at a time as required. To gain a better understanding of how the MMU is used to switch 8K blocks of memory in BASIC's workspace, refer to "Barden's Buffer" (THE RAINBOW, May 1990, Page 78) and "New Clear-screen Routines: ML and the CoCo 3" (THE RAINBOW, June 1992, Page 1).

Conclusively, *VEFit* provides a simple tool to support the VEF format under Disk BASIC. The longevity of this format is a result of its simplicity. To learn more about VEF, refer to Tim Kientzle's series of articles, "Displaying Picture Files" (THE RAINBOW, October 1990 through December 1990). I hope VEF gains wider recognition under Disk BASIC, as there are many advantages to following a good standard.

Lines	Function
10-40	Initialization
42-52	Main Menu
54-144	Load Picture
146-192	Palette Change
194-290	Save Picture
292-314	Memory Display
316-354	Monitor Reset
356-384	Option Select
386-390	Clear Screen
392-398	Keyboard Input
400-404	Default Palette
406-414	Read Palette
416-424	View Palette
426-448	Read Directory
450-454	Error Trap
456-458	Exit/[BREAK]

Figure 4: *VEFit* Subroutines

Thomas Wong is an undergraduate pursuing a degree in commerce. He enjoys cycling, stamp collecting and playing Badminton. He may be contacted at 29 Page Ave., Red Deer, AB T4P 1J7, Canada. Please include an SASE with sufficient return postage when requesting a reply.

Feature Program

Twenty One and Over

By Trevor Boehm

Twenty One is a twist on the classic game Blackjack. Instead of dealing cards, the object is to stop three numeric tumblers in such a way that the total value of the displayed numbers is 21 or greater. But it isn't easy . . .

Twenty One works on any CoCo with at least 16K and Extended color BASIC. Enter the listing as shown, save it to tape or disk as TWENTY1, then enter RUN. After reading the short instruction screen, press any key to proceed. Displayed in the middle of the playing screen is a spinning tumbler with numbers on it. Press any key to stop the tumbler; the value displayed is added to your total. After you have stopped three tumblers, if your total is 21 or greater, you've won!

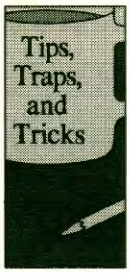
This delightfully simple game is remarkably entertaining. I hope it finds a good home in your software library.

Trevor Boehm is a tenth-grade student whose greatest passion is challenging computers with new programs. He has participated in several science fairs and has received numerous awards for his work. He can be contacted at 77 Inwood Cres., Winnipeg, MB R2Y 1A2, Canada. Please include an SASE when requesting a reply.

16K ECB

The Listing: TWENTY1

```
1 'TWENTY-ONE
2 'BY TREVOR BOEHM
3 'COPYRIGHT (C) 1993
4 'BY FALSOFT, INC.
5 'RAINBOW MAGAZINE
10 CLS
20 PRINT"TWENTY-ONE...":PRINT
30 PRINT"THE OBJECT OF THE GAME
IS TO    SCORE 21 POINTS BY STOP
PING EACHOF FOUR COUNTERS. YOUR
SCORE IS THE SUM OF THE DIGITS O
N THE    COUNTERS. GOOD LUCK!"
40 EXEC44539
50 PLAY "O1T5CDEFGP4"
60 PLAY "O2CDEFGP4"
70 PLAY "O3CDEFG"
80 A$=INKEY$:IF A$<>"" THEN 80
90 CLS:PRINT"PRESS ANY KEY TO ST
OP COUNTERS"
100 CC=1:GOSUB 140
110 CC=2:GOSUB 140
120 CC=3:GOSUB 140
130 CC=4:GOSUB 140:GOTO 170
140 C(CC)=RND(10)-1
150 PRINT@256+14,C(CC)
160 IF INKEY$<>"" THEN PRINT@454
+CC*3,C(CC):RETURN ELSE GOTO 140
170 PRINT@448,"TOTAL SCORE WAS:"
;C(1)+C(2)+C(3)+C(4);
180 IF C(1)+C(2)+C(3)+C(4)=>21 T
HEN FOR X=1 TO 10:CLS RND(9)-1:N
EXT:PLAY"CCDDEEFFGG":PLAY"P402CC
DDEEFFGG":PLAY"P403CCDDEEFFGG":C
LS:PRINT@256+5,"YOU BEAT THE COM
PUTER!"
190 EXEC44539:RUN
```



TIM KIENTZLE

How to Manage Strings, Part II

Last month we discussed six different approaches to handling strings in C or assembly language. To refresh your memory, they are

- **Allocation of a fixed number of fixed-length blocks for storing strings:**
  Advantages: fast and easy.
  Disadvantages: limits number and size of strings.
- **Allocation of space for each string from `malloc`:**
  Advantages: easy; no limits on size or number of strings.
  Disadvantages: can be slow when there are many strings; not usually available for assembly programmers.
- **Fixed-length blocks, dynamically allocated from the heap:**
  Advantages: unlimited number of strings.
  Disadvantages: limited size.
- **Static string pool:**
  Advantages: no size limits.
  Disadvantages: deleting strings doesn't reclaim space.
- **Combination of previous two:**
  Advantages: easy to implement; allows fast handling of common shorter strings; uses string pool to handle occasional longer ones.
  Disadvantages: deleting longer strings doesn't reclaim space.
- **String pool with compaction (or "garbage collection"):**
  Advantages: size and number of strings are limited only by pool size; memory is used fairly efficiently.
  Disadvantages: harder to implement; can't use pointers to strings.

As I pointed out last month, the last method is the most general, and there are times when it is the only one that will work well. But the others are faster and/or easier to implement, and are usually preferred when they will do the job. This month I'll develop a compacting string manager in some detail since it will bring up some interesting points.

A Compacting String Manager

The idea behind a compaction system is simple: whenever we can't easily find enough space, we move all the strings that are still allocated down to the bottom of the string pool. The result is that all the remaining free space once again appears in a block at the top of the pool, where it can be easily allocated.

In order to simplify the following discussion, let's establish one bit of terminology: a *client* is any program, function or other code that uses our *string manager*. The idea is that our string manager is providing some service to its client, and we need to distinguish between things for which the client is responsible and things that the string manager must handle. The general rule is that the client and the manager never make changes to something they don't owu.

The first goal in designing any package such as this is to figure out what the client has to deal with. The first hurdle appears when we realize that the client cannot use pointers to the strings, since compaction can happen at any time and this can cause the strings to move. Instead, we'll let the client have pointers to *handles*. A handle is really just a pointer, but it's a pointer that we own, not the client. Since the handle will never move, it's perfectly safe for the clieot to have a pointer to it. And since we own the handle, we are free to change it whenever we need to.

This means that our clieot has to be careful when actually accessing strings. Since the client's pointer is really a pointer to the handle, the client must first get a pointer to the actual string in order to manipulate the string directly. Be very careful with this since strings can move any time a new string is created. Usually we'll create routines in the string package to handle almost everything the client could need (copying strings, concatenating strings, finding substrings), so the client should hardly ever need to obtain a pointer to the actual string. If it did, it might look something like that in Figure 1.

(Most of the code for the basic string manager appears in the following paragraphs. In the interest of conserving space, I've deliberately left out some pieces, so you should carefully study this to make sure you understand what's going on.)

```
* String handle is pointed to by X, get first character
ldx  ,x     Get pointer to string
lda  ,x     Get character
```

Figure 1

One point that deserves some comment is that a handle in this case actually contains slightly more than just a pointer to the string. It also contains a pointer to the *next* handle. Depending on your needs, it might be reasonable to keep even more information in each handle. For example, you might want to keep the size of the string here as well.

All handles are in one of two lists. One list starts with the handle pointed to by the "first" variable, and it contains all the handles that are in use. This list is used whenever we do compaction. By keeping this list sorted, our compaction becomes much simpler. The second list is a roster of the string handles that are unused. A significant part of creating a new string or freeing old string space involves moving the handle from one list to the other.

First, we define the actual storage for the pool. Just as with the simple pool manager described last month, we need to know the addresses of the beginning and end of the pool and the address of the free space at the top of the pool, which is where new strings will be allocated. This can be accomplished using the code shown in Figure 2.

continued

Tips, Tricks. . . *continued*

We also have to define the handles. Since each handle is four bytes long, we allocate 400 bytes for 100 handles. If we were implementing this in C, we would probably use `malloc` to allocate space for the string pool and the string handles. The advantage is that it should be easy to create new string handles (or enlarge the pool) as we go along. We also need pointers to the two lists. The code for this is shown in Figure 3.

Unlike the string-pool manager discussed last month, the client needs to reset the string manager before actually trying to create any strings. The bulk of this routine (see Figure 4) is devoted to putting all the handles onto the free list.

Freeing a string is actually pretty simple since all it really amounts to is abandoning the string — the compaction will reclaim the space later. The complicated part is moving the handle from the list of strings in use to the list of free handles. As we have things now, this requires searching the list to find the handle just before ours. If desired, we could speed it up by augmenting each handle with a pointer to the previous handle. This could be made beefier by checking for various errors (see Figure 5).

Allocating a string is a simple process, too, if there's enough space. If there's not enough space, we do compaction and try again. If it fails twice, there really isn't enough space, and we must report an error. This would be faster if we didn't start from the beginning of the used list everytime we needed to add a new handle to the end. The code is shown in Figure 6.

Finally, there's the actual compaction routine (see Figure 7). Since the list of used strings was kept sorted, all we do is walk down that list, moving each successive string down to its proper location. In the code below, Register X points to the string handle we're considering. U points to where the string is being moved, and Y points to where the string is now.

To make this into a full-fledged string manager, we should write the routines to perform basic string operations. For example, a string-concatenate function would take pointers to two string handles and return a pointer to a new string handle which points to the new string. With a fully functioning string manager, it should be relatively simple to write a line-based editor by simply storing each line as one string. Hopefully, by the time you read this, I'll have completed a somewhat more sophisticated version of this string manager in C and uploaded it to Delphi. Look for it there.

Tim Kientzle is currently pursuing a doctorate in mathematics at the University of California at Berkeley. He is the author of V-Term *and has worked with the Color Computer since 1982.*

```
pool    rmb 1000    1000 bytes of string storage
pend    rmb 0       End of pool
palloc  rmb 2       Pointer to current allocation spot in pool
```

Figure 2

```
strings rmb 4*100  Handles for 100 strings
free    rmb 2      List of free handles
first   rmb 2      First handle in pool
```

Figure 3

```
* Reset the string pool, freeing all the strings
reset   pshs d,x,y
        ldx  #pool      Address of the beginning of the pool
        stx  palloc     That's the new allocation spot
        ldx  #0         There are no handles currently in use
        stx  first
        ldx  #strings   Get address of first handle
        stx  free       First handle is now first free handle
        lda  #99
reset1  leay 4,x        Y points to next string handle
        sty  2,x        first handle points to second handle
        tfr  y,x        Now point to second handle
        deca
        bne  reset1     Point 99 string handles each to the next
        ldy  #0
        sty  2,x        Last string handle points nowhere
        puls d,x,y,pc   Restore variables and return
```

Figure 4

```
* Free a string
* On entry, X points to a string handle
* There are two steps: remove it from used list, add it to free list
free    pshs d,x,y
* First remove from used list
        cmpx first    Was this one first?
        bne  free1
        ldy  2,x      If so, the second one becomes first
        sty  first
        bra  free4
free1   ldy  first    Otherwise, start at the first
        bra  free3    (Start by checking the first one)
free2   ldy  2,y      Get the next handle
free3   cmpx 2,y      Is this the handle before us?
        bne  free2    No, keep looking
        ldd  2,x      Get pointer to handle after us
        std  2,y      Make handle before us point to handle after us
* Now add to free list
free4   ldy  free     Get address of first free handle
        sty  2,x      old first handle is now after this one
        stx  free     This handle is now first free one
        puls d,x,y,pc
```

Figure 5

```
* Allocate next string in pool
* Size of requested string in D
* Return address of string in X
alloc   pshs d,y      Save registers, but not X
* Get address of string
        ldx  palloc   Address to store string
        leay d,x      Move end of pool address after this string
        cmpy #pend    Is this past the end of the pool?
        blo  alloc1   If not, we're okay
        bsr  compact  If it is, then compact
        ldx  palloc   Try allocating again
        leay d,x      New palloc
        cmpy #pend    Is it too big?
        lblo alloc1
        ldx  #0       Return 0 if we can't create the string
        bra  alloc4
alloc1  sty  palloc   Bump pointer for allocating next string
* Get a string handle, X holds address of string
        ldy  free     Address of first free handle
        pshs y        Save it, this will be our new handle
        ldy  2,y      Address of second free handle
        sty  free     That's now the first free handle
        puls y        Now we have our handle
        stx  ,y       This handle knows where our string is stored
        ldd  first    Get address of first handle into D.
        bne  alloc2   If there aren't any...
        sty  first    .. then this is the first one.
        bra  alloc3
alloc2  tfr  d,x      Put pointer to handle in X
        ldd  2,x      Is there another one?
        bne  alloc2   Yes, keep going
        sty  2,x      Found the end, add this one to the end
alloc3  ldx  #0       Mark this one as the last one
        stx  2,y
        tfr  y,x      Move pointer to X for returning
alloc4  puls d,y,pc
```

Figure 6

```
* Compact
* Start with first allocated handle
* Move each one where it should be
compact pshs d,x,y,u
        ldu  #pool      Where the first string should be
        ldx  first      Get the first string
        beq  garb9      If none, there's nothing to do
garb1   ldy  ,x         Get where string is now
        stu  ,x         Store where it will be in a minute
garb2   lda  ,y+        Move the string
        sta  ,u+
        bne  garb2      Null byte means end of string
        ldx  2,x        Get next handle
        bne  garb1
garb9   stu  palloc     Just after last string is where we allocate now
        puls d,x,y,u,pc
```

Figure 7

Feature Article

Monthly Calendar

by Harold H. Britten

Miniature calendars are handy when you need to know the day of the week on which a given date falls. Check registers often include such calendars for the current year as well as one or two years in the future, and most convenience stores carry plasticized wallet-size copies as impulse items at the counter. It's easy to lose those little cards, though.

Calendar is a short BASIC program I wrote so my CoCo 3 could give me this information, too. When you run *Calendar*, you are prompted to enter the month and the year. To do this, enter the first few letters of the month, followed by a comma and the year. Make sure you enter all four digits of the year (e.g., 1992 instead of just 92). For example, to see the calendar for November 1995, you would enter NOV,1995. After *Calendar* has displayed the appropriate month, press any key to see the calendar for another month.

Calendar is "perpetual" — that is, it can be used to display the monthly calendar for any month of any year. This sure helps make planning for the future much easier. The program can also be fun — use it to see what day of the week you were born on. I hope you find *Calendar* to be a useful (and fun) program.

Harold H. Britten is an avid bycyclist and currently works for a health and fitness club. He may be contacted at 949 N. Loudon Rd., Apt. #6, Latham, NY 12110. Please include an SASE when requesting a reply.

CoCo 3

The Listing: CALENDAR

```
1 'PERPETUAL CALENDAR
2 'BY HAROLD H. BRITTEN
3 '(COPYRIGHT (C) 1993
4 'BY FALSOFT, INC.
5 'RAINBOW MAGAZINE
10 WIDTH40:DIM M$(12),D(12):FORI=
1TO12:READM$(I),D(I):NEXT
20 PRINT:INPUT"      MONTH,YEAR";
M$,Y:FORI=1TO12:IFM$=LEFT$(M$(I)
,LEN(M$))THENM=I ELSENEXT:GOTO20
30 D=INT(Y-2000):D(2)=1-SGN(D-IN
T(D/4)*4)+SGN(D-INT(D/100)*100)-
SGN(D-INT(D/400)*400):D=INT((D+3
)/4)-INT((D+99)/100)+INT((D+399)
/400)+D:FORI=0TOM-1:D=D+D(I):NEX
T:D=D+6:D=D-INT(D/7)*7
40 CLS:M$=M$(M)+STR$(Y):PRINTTAB
(21-LEN(M$)/2);M$:LOCATE0,3:PRIN
T" SUN MON TUE WED THU FRI
  SAT":LOCATE5*D,6:FORI=1TOD(M)+
28:M$=STR$(I):PRINTLEFT$("  ",5
-LEN(M$))+M$;:D=D+1:IFD=7THENPRI
NT:PRINT:PRINT:D=0:NEXTELSENEXT
50 A$=INKEY$:IFA$=""THEN50ELSEPR
INT:IFA$=" "THENIFM<12THENM=M+1:
GOTO30ELSEM=1:Y=Y+1:GOTO30ELSE20
60 DATA JANUARY,3,FEBRUARY,0,MAR
CH,3,APRIL,2,MAY,3,JUNE,2,JULY,3
,AUGUST,3,SEPTEMBER,2,OCTOBER,3,
NOVEMBER,2,DECEMBER,3
```

Proven Technology

On the Razor's Edge of the Color Computer Frontier

OWL-WARE

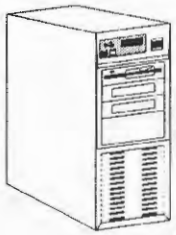
In our 10th Year !

NOW, 120MB Hard Drive Standard

A DECADE OF SERVICE TO THE COMPUTER USER!

486SX-20 SYSTEMS - $1495.00!

Now You can enter the world of 486 computing at a reasonable cost!



the OWL SUPER ATOM - 486

High Powered Computing from a local, well established company.

- 33MHz / 50MHz i486 based Systems with Socket for Weitek CoProcessor
- System and Video BIOS in Cache
- Large Tower Case : (33MHz, FCC Class B) - (50MHz, FCC Class A)
- 230 Watt Power Supply & 8 Option Slots
- System Price includes 120MB HD, 4MB RAM, Std. Resolution Color VGA Monitor, High Resolution VGA Card, 2 High Density FD's, MS-DOS 5.0

$1495 486SX-20 / $1795 486-33 / $2095 486DX-50 / $2185 486-DX2-66

- Super VGA Upgrade

3- YEAR WARRANTY on all systems! Including One Full Year on Parts and Labor. Manufactures 3-Year Warranty on All Hard Drives

OWL SUPER ATOM - 386

- 25/40MHz 386DX Based
- Small Footprint Case
- FCC Class B Approved
- 200 Watt Power Supply
- 7 Expansion Slots
- 4MB of RAM
- 120MB Hard Drive
- Std. Resolution VGA Color Monitor
- 2 High Density FD's
- 101 Keyboard
- MS DOS 5.0

$1495 40MHz

OWL SUPER ATOM - SX

- 16/25MHz 386SX Based
- Small Footprint Case
- FCC Class B Approved
- 200 Watt Power Supply
- 7 Expansion Slots
- 2MB of RAM
- 120MB Hard Drive
- Std. Resolution VGA Color Monitor
- 2 High Density FD's
- 101 Keyboard
- MS DOS 5.0

$1195 25MHz / $1295 33MHZ

386-SX Notebook Computers

20MHz, 60MB HD, 1.4MB FD, 2MB RAM(exp. to 5MB), VGA 640X480 LCD w/32 shades of gray. Ports: 2 Ser, 1 Par, 1 VGA, DOS & Windows, 7.7LBS!

$1395

386-DX Notebook Computers

33MHz, 120MB HD, 1.44MB FD, 32KB CACHE, 4MB RAM(exp. to 16MB), Std. VGA LCD w/32 gray, Ext. Keypad inc., DOS & Windows, 7.7LBS!

$1895

OWL COMPUTER SERVICES

5950 Keystone Drive
Bath, PA (215)-837-1917 (800) 248-6228

Kids & Us - RadioShack® Dealer	St. Onge Systems	Computers & Games
Pottstown Ave., RT. 663	Wescosville	Muhlenberg Shopping Plaza
Pennsburg (215)-679-3389	Call for Appt. (215)-481-9775	Reading (215)-929-0540

DISK DRIVES

Bonus! Special Bundled Software with any Disk Drive Purchase!

Floppy Drive Systems

The Highest Quality for Years of Service

Drive 0 Systems (Half Height, Double Sided,

SOLD OUT!
WE NEED CONTROLLERS! ~~$180.~~

IF YOU HAVE 502 CONTROLLERS, CALL US!

Drive 1 Systems (Half Height, Double Sided, Direct Drives) **$115.**

New 3.5", 720K Drives for OS-9 with case & Power Supply **$129.** SALE!

Drive 1 Systems have drive, case, power supply. (You may require optional cable and/or DOS chip to use)

Special for 0/1 Combos (0,1,2,3) $199.

(WITHOUT CONTROLLER)

HALF- HEIGHT DRIVE UPGRADES FOR RS HORIZONTAL CASES

Why only double the capacity of your system when you can triple in the same case? Kit includes: double-sided to fit your case, chip to run both sides of new drive, hardware, and detailed instructions. Easy! Takes only 5 minutes!

Model **Only $119.**
500 , 501, or 502

All drives are new and fully assembled. We ship only FULLY TESTED and CERTIFIED at these low prices. We use Fuji, YE Data, and other fine brands. No drives are used or surplus unless otherwise stated to you when you order. We appear to be the one of the few advertisers in Rainbow who can truly make this claim. We have 7 years experience in the CoCo disk drive market! We are able to provide support when you have a problem.

Drives 1 Year Warranty

OWL Phones

Order Numbers (only)
1-800-245-6228
1-215-682-6855
Fax: 1-215-837-1942
Technical Help
1-215-837-1917

OWL WARE Software Bundle

Disk Tutorial/Utilities/Games

DISK TUTOR Ver 1.1

Learn how to use your disk drive from this multi-lesson, machine language program. This tutor takes you through your lessons and corrects your mistakes for a quick, painless disk drive introduction. (This professionally written tutor is easily worth the bundle's total price.)

3 UTILITIES

A copy verify, copy, and DOS utility.

2 GAMES

We will select 2 games from our stock. These are sold for more than $20 each.

Do not mistake this software with cheap "Public Domain" software which others offer. All of this software is copyrighted and professional in quality. The tutor is unique with us and has helped thousands of new users learn their disk drive.

only $27.95
(or even better)
only $6.95 with
any Disk Drive Purchase!!

512K Upgrade

Again at a popular price. Fully assembled and tested before shipping. Easy to install. Uses fast 120 ns. chips.

SALE $79.

Now includes **memory test, Ram Disk Lighting, Printer Lighting, and Back-up Lighting.** All with an upgraded manual exclusive with OWL!

Our prices include a discount for cash but do not include shipping.

OWL-WARE has a liberal warranty policy. During the warranty period, all defective items will be repaired or replaced at our option at no cost to the buyer except for shipping costs. Call our tech number for return. Return of non-defective or unauthorized returns are subject to a service charge.

OWL-WARE
P.O. BOX 116
Mertztown, PA 19539