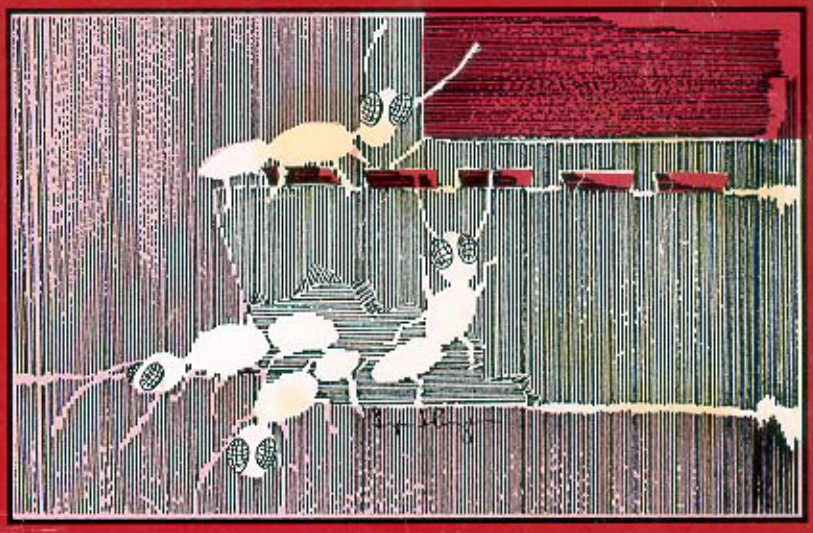
HOW TO BACKUP, UNLOCK, OR MODIFY COPY-PROTECTED SOFTWARE

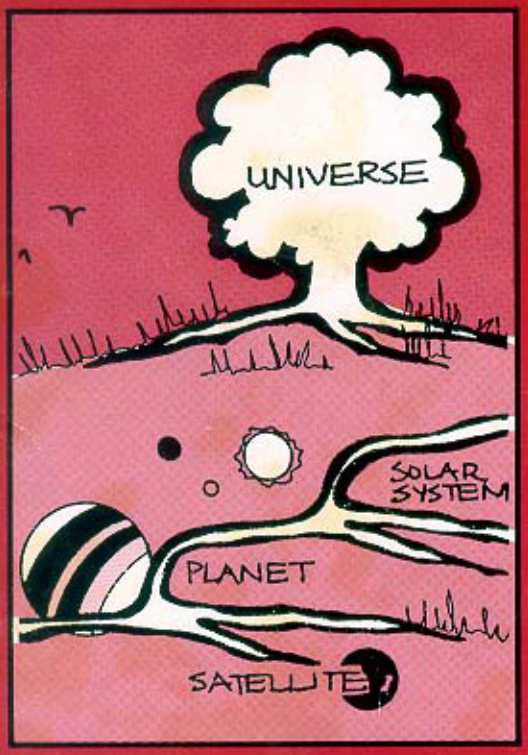
Hardcore **Issue No.6** **$2.50**

COMPUTIST

MODIFIED ROMS:

Turn your Apple into the ultimate softkey machine.





QUICKBUG:

Add three new and helpful commands to your "Quicktrace" 6502 debugging program (or just use them via monitor).

CORE INSERT: DATA BASES

Just what is this strange thing called a Data Base?

SoftKey Publishing
P.O. Box 44549
Tacoma, WA 98444

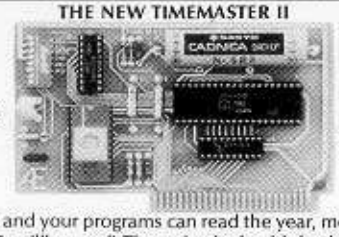
BULK RATE
U.S. Postage
PAID
Tacoma, WA
Permit No. 269

APPLIED ENGINEERING IS 100% APPLE

That's Why We're So Good At It!

THE NEW TIMEMASTER II

Automatically date stamps files with PRO-DOS



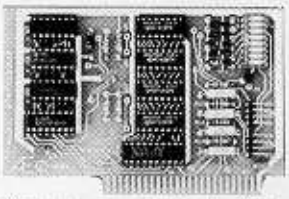
NEW 1984 DESIGN
An official PRO-DOS Clock

- Just plug it in and your programs can read the year, month, date, day, and time to 1 millisecond! The only clock with both year and ms.
- A rechargeable NiCad battery will keep the TIMEMASTER II running for over ten years.
- Powerful 2K ROM driver — No clock could be easier to use.
- Full emulation of most other clocks, including Thunderclock and Appleclock (but you'll like the TIMEMASTER II mode better). We emulate other clocks by merely dropping off features. We can emulate them but they can't emulate us.
- Basic, Machine Code, CP/M and Pascal software on 2 disks!
- Eight software controlled interrupts so you can execute two programs at the same time (many examples are included).
- On-board timer lets you time any interval up to 48 days long down to the nearest millisecond.

The TIMEMASTER II includes 2 disks with some really fantastic time oriented programs (over 40) including appointment book so you'll never forget to do anything again. Enter your appointments up to a year in advance then forget them. Appointment book will remind you in plenty of time. Plus DOS dater so it will automatically add the date when disk files are created or modified. The disk is over a $200.00 value along—we give the software others sell. All software packages for business, data base management and communications are made to read the TIMEMASTER II. If you want the most powerful and the easiest to use clock for your Apple, you want a TIMEMASTER II.

PRICE $129.00

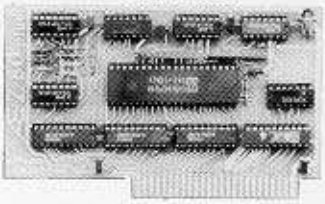
Super Music Synthesizer
Improved Hardware and Software



- Complete 16 voice music synthesizer on one card. Just plug it into your Apple, connect the audio cable (supplied) to your stereo, boot the disk supplied and you are ready to input and play songs.
- It's easy to program music with our compose software. You will start right away at inputting your favorite songs. The Hi-Res screen shows what you have entered in standard sheet music format.
- Now with new improved software for the easiest and the fastest music input system available anywhere.
- We give you lots of software. In addition to Compose and Play programs, 2 disks are filled with over 30 songs ready to play.
- Easy to program in Basic to generate complex sound effects. Now your games can have explosions, phaser zaps, train whistles, death cries. You name it, this card can do it.
- Four white noise generators which are great for sound effects.
- Plays music in true stereo as well as true discrete quadraphonic.
- Full control of attack, volume, decay, sustain and release.
- Will play songs written for ALF synthesizer (ALF software will not take advantage of all our card's features. Their software sounds the same in our synthesizer.)
- Our card will play notes from 30HZ to beyond human hearing.
- Automatic shutoff on power-up or if reset is pushed.
- Many many more features.

PRICE $159.00

Z-80 PLUS!



- TOTALLY compatible with ALL CP/M software.
- The only Z-80 card with a special 2K "CP/M detector" chip.
- Fully compatible with microsoft disks (no pre-boot required).
- Specifically designed for high speed operation in the Apple IIe (runs just as fast in the II+ and Franklin).
- Runs WORD STAR, dBASE II, COBOL-80, FORTRAN-80, PEACHTREE and ALL other CP/M software with no pre-boot.
- A semi-custom I.C. and a low parts count allows the Z-80 Plus to fly thru CP/M programs at a very low power level. (We use the Z-80A at fast 4MHZ.)
- Does EVERYTHING the other Z-80 boards do, plus Z-80 interrupts.

Don't confuse the Z-80 Plus with crude copies of the microsoft card. The Z-80 Plus employs a much more sophisticated and reliable design. With the Z-80 Plus you can access the largest body of software in existence. Two computers in one and the advantages of both, all at an unbelievably low price.

PRICE $139.00

Viewmaster 80

There used to be about a dozen 80 column cards for the Apple, now there's only ONE.

- TOTALLY Videx Compatible.
- 80 characters by 24 lines, with a sharp 7x9 dot matrix.
- On-board 40/80 soft video switch with manual 40 column override
- Fully compatible with ALL Apple languages and software—there are NO exceptions.
- Low power consumption through the use of CMOS devices.
- All connections are made with standard video connectors.
- Both upper and lower case characters are standard.
- All new design (using a new Microprocessor based C.R.T. controller) for a beautiful razor sharp display.
- The VIEWMASTER incorporates all the features of all other 80 column cards, plus many new improvements.

	PRICE	BUILT IN SOFTSWITCH	SHIFT KEY SUPPORT	LOW POWER DESIGN	80 COLUMN VIDEO	7x9 DOT MATRIX	LIGHT PEN INPUTS	80 COLUMN OVERRIDE	INVERSE CHARACTERS
VIEWMASTER	169	YES	YES	YES	YES	YES	YES	YES	YES
SUPRTERM	MORE	NO	YES	NO	NO	NO	NO	YES	YES
WIZARD80	MORE	NO	NO	NO	NO	YES	NO	YES	YES
VISION80	MORE	YES	YES	NO	NO	YES	NO	NO	NO
OMNIVISION	MORE	NO	YES	NO	NO	NO	NO	YES	YES
VIEWMAX80	MORE	YES	YES	NO	NO	YES	NO	NO	YES
SMARTERM	MORE	YES	YES	NO	NO	NO	YES	YES	NO
VIDEOTERM	MORE	NO	NO	YES	NO	YES	YES	NO	YES

The VIEWMASTER 80 works with all 80 column applications including CP/M, Pascal, WordStar, Format II, Easywriter, Apple Writer II, VisiCalc, and all others. The VIEWMASTER 80 is THE MOST compatible 80 column card you can buy at ANY price!

PRICE $179.00

MemoryMaster IIe 128K RAM Card

- Expands your Apple IIe to 192K memory.
- Provides an 80 column text display.
- Compatible with all Apple IIe 80 column and extended 80 column card software (same physical size as Apple's 64K card).
- Can be used as a solid state disk drive to make your programs run up to 20 times FASTER (the 64K configuration will act as half a drive).
- Permits your IIe to use the new double high resolution graphics.
- Automatically expands Visicalc to 95 K storage in 80 columns! The 64K config. is all that's needed, 128K can take you even higher.
- PRO-DOS will use the MemoryMaster IIe as a high speed disk drive.
- Precision software disk emulation for Basic, Pascal and CP/M is available at a very low cost. NOT copy protected.
- Documentation included, we show you how to use all 192K.

If you already have Apple's 64K card, just order the MEMORYMASTER IIe with 64K and use the 64K from your old board to give you a full 128K. (The board is fully socketed so you simply plug in more chips.)

MemoryMaster IIe with 128K	**$249**
Upgradeable MemoryMaster IIe with 64K	**$169**
Non-Upgradeable MemoryMaster IIe with 64K	**$149**

Our boards are far superior to most of the consumer electronics made today. All I.C.'s are in high quality sockets with mil-spec. components used throughout. P.C. boards are glass-epoxy with gold contacts. Made in America to be the best in the world. All products work in the APPLE IIE, II, II+ and Franklin. The MemoryMaster IIe is IIe only. Applied Engineering also manufactures a full line of data acquisition and control products for the Apple; A/D converters and digital I/O cards, etc. Please call for more information. All our products are fully tested with complete documentation and available for immediate delivery. All products are guaranteed with a no hassle **THREE YEAR WARRANTY.**

Texas Residents Add 5% Sales Tax
Add $10.00 If Outside U.S.A.
Dealer Inquiries Welcome

Send Check or Money Order to:
APPLIED ENGINEERING
P.O. Box 798
Carrollton, TX 75006

Call (214) 492-2027
8 a.m. to 11 p.m. 7 days a week
MasterCard, Visa & C.O.D. Welcome
No extra charge for credit cards

THIS ISSUE:

Hardcore **Volume 3** **Number 2**

COMPUTIST

Hardcore

COMPUTIST

Publisher/Editor
Charles Haight

Circulation
Valerie Robinson

Advertising
Attn:Tanyalee Erwin
Advertising Department
3710 100th St. SW
Tacoma, WA 98499

Copy Editor
Bassam Bishuti

Technical Editing
Gary Peterson

Business Manager
Ken Fields

Programmer
Ray Darrah

Printing
Grange Printing, Inc.
Seattle, WA

Production & Graphics
Lynn Campos-Johnson

Publishing
SoftKey Publishing
P.O. Box 44549
Tacoma, WA 98444
USA

Address all advertising inquiries to SoftKey Publishing, Advertising Department, 3710 100th St. SW, Tacoma, WA 98499. Address all manuscripts and editorials to: SoftKey Publishing, Editorial Department, P.O. Box 44549, Tacoma, WA 98444. All subscription inquiries should be directed to: Subscription Department, Hardcore COMPUTIST, P.O. Box 44549, Tacoma, WA 98444. See page 29 for subscription rates and information.

MAILING NOTICE: change of address must be postmarked at least 30 days prior to move. Paste your present mailing label on postal form **3576** and supply your new address for our records. Issues missed due to non-receipt of change of address may be acquired at the regular back issue rate.

Return postage must accompany all manuscripts, drawings, photos, disks, or tapes if they are to be returned. No responsibility can be assumed for unsolicited manuscripts. We suggest you send only copies.

DOMESTIC DEALER RATES sent upon request, or call (206) 581-6038.

INPUT INPUT INPUT

The Fourth Dimension

I am writing about the articles on Ultima II in Hardcore Computist no.4. The programs supplied were a great aid to my gaming. The table on page 14 of that issue displays a few "unknowns" which led me to discover that a certain address, $4E13, refers to the time period. Assigning these values to this location will place the player into the following periods:

Value in $4E13	Time Period
00	Legends
01	Pangea
02	B.C.
03	A.D.
04	Aftermath

Using this information, the information from an Ultimap or two, and that from the table on page 14, it is possible to locate a place and time, and project the character there!

Many thanks to your magazine for a great group of articles.

Peter Thorpe
Hong Kong

Search or Destroy?

I have a problem that I hope you can help me with. I copied a 99 sector program from a magazine on my Apple II+ with 64K memory. I saved it, saw a mistake towards the last and corrected it, saved it again and wound up with only one line in my program! Yet if I use the view file utility on my Copy II+ disk, I can see the text of the program still there. I just can't retreive it or bring it back. This isn't the first time I've done this boo-boo and I don't know why it happened or what I can do about it.

Do you have any suggestion outside of blowing up my Apple? Can you recommend any utility program that would bring it back for me? I dread to think of re-typing all that over again. I am very interested in computing and would also appreciate it if you could list 6 or 8 of the best programs or utilities available to help. I am in my 70's and do not intend to compete with the younger hackers, but I like to keep my brain agile, even if the rest of me is falling apart. Using my Apple sure beats sitting in my rocking chair!

Another thing. In magazines, most writers assume that the average reader must be a computer genius. I have printed out entire programs, and at the end the author might say it is for the Apple III ... but with a few minor corrections or changes it will work on a II+ or it is made to work on a Brother printer but with a few minor changes it can be made to work on an Epson. Then they drop it. If they know the changes, why in the "#%#$$#%$#" don't they put them in the article? It's discouraging and I'm not the only one who feels that way. A lot of the members in our Apple club have the same beef.

Well, thanks for letting me bend your ear. I hope you can help me with my problem.

Andrew Paranya
Winterhaven. CA

Mr. Paranya:

From the description you have given of your problem, it sounds as if one of the Applesoft line pointers has been destroyed. Unfortunately, there are no commercial utilities that we know of that will remedy this problem. However, your problem can be fairly easily solved if you are willing to learn how Applesoft stores the lines of a program.

An Applesoft program normally begins at $801 (2048 decimal) in memory. Each line of a program is stored as a series of hexadecimal bytes with a 00 indicating the end of each line. The first two bytes of every line contain, in lo/hi format, a pointer to the memory location of the next line. The program line number is stored as the third and fourth bytes of each line, again in lo/hi format. The remainder of the bytes in the line, up to the end-of-line 00, are the encoded statements of the program. If a line

DEAR AUTHOR:

Would you like to be published in Hardcore COMPUTIST? We would like to hear from you.

Hardcore COMPUTIST welcomes articles and submissions on a variety of subjects of interest to users of the Apple (or compatible) computers and would like to publish well-written material on the following:

*** Softkeys * Hardware Modifications**
*** Advanced Playing Techniques**
*** DOS modifications * Utilities**
*** Product reviews * Adventure Tips**
*** Original programs of interest**
*** Do-it-yourself hardware projects**
*** General interest articles**
*** Bit-Copy Parameters**

We prefer to see your submission on a DOS 3.3 disk using an Apple (or compatible) editing program. Please enclose a double-spaced hardcopy (paper) manuscript using a dot-matrix or letter-quality printer (or typewriter). Submissions will be mailed back if adequate adequate return packaging is included.

Hardcore COMPUTIST pays on acceptance. Rate of payment depends on the amount of editing necessary and the length of the article. Payment ranges between $10 for a short softkey, and $50 per typeset page for a full-length article. We pay more for softkeys if the original commercial disk is enclosed for verification. We guarantee the disk's return.

Softkey Publishing buys all rights as well as one-time reprint rights (for upcoming BEST OF Hardcores) on general articles, and exclusive rights on programs. We may make alternate arrangements with individual authors, depending on the merit of the contribution.

At present we are not accepting fiction or poetry submissions, but Hardcore COMPUTIST may make an exception for an outstanding computer-related short story or poem.

For a copy of our Writer's Guide, send a business-sized (20-cent) SASE (self-addressed, stamped envelope) to:

Hardcore COMPUTIST
WRITER'S GUIDE
P. O. BOX 44549
TACOMA WA 98444

INPUT *continued from page 3*

is the last line of the program then the pointers to the next program line point at 00's. Take for instance the following simple two line program:

10 PRINT
20 END

This program would be stored starting at $801 as:

Addr.	Byte	Function
801:	07	- Lo byte of pointer.
802:	08	- Hi byte of pointer.
803:	0A	- Line # lo byte (10).
804:	00	- Line # hi byte.
805:	BA	- Token for PRINT.
806:	00	- Indicates end of line.
807:	0D	- Lo byte of pointer.
808:	08	- Hi byte of pointer.
809:	14	- Line # lo byte (20).
80A:	00	- Line # hi byte.
80B:	BA	- Token for END.
80C:	00	- Indicates end of line.
80D:	00	- The two 00's here
80E:	00	- indicate the end of the program.

Apparently the pointers in the second line of your program somehow got set to zeroes, instead of pointing to the next line in memory.

To remedy your problem you need to load in your program, enter the monitor and correct the pointers in the second line of the program (look for 3 consectutive bytes of 00) so that they point to the next line in memory. You should only have to change these two bytes before resaving your program (you may want to save it under a different name, in case of a problem).

If this does not work then you may consider blowing up your computer (or just pouring Pepsi over the motherboard with the power turned on). But seriously, we hope this solves your problem. For more information on how Applesoft stores programs, see the Call A.P.P.L.E. publication "All About Applesoft."

As to your question about what we consider the best utility programs for the Apple, our favorites are:

Global Program Line Editor (Beagle Brothers)
Bag of Tricks (Quality Software)
The CIA (Golden Delicious Software)
Copy II+ (Central Point Software)

If you want to program in Assembly language, our favorite assembler is the S-C Macro Assembler from S-C Software Corp.

Back to Basics

I have really enjoyed your magazine and I think that it is about time that somebody start publishing the things you publish.

I have only one comment; your softkeys always assume that we know what you are talking about. I can follow the steps with no problem, but I have no idea of what I am doing. I would like to see a column or two that starts at the basics and works its way up. How about it?

Seth Miller
Sacramento, CA

Seth: Your point is well taken. Because we have received so many suggestions similar to yours, we will be publishing a column on copy-protection which begins with the basics. See page 29 of this month's issue.

Time Zone

For anyone who has Time Zone from Sierra On-Line, a word of caution: Do not try to copy it with the built-in copier on side A of the disk. It is very unreliable and it would take a lot of playing before you run across the error(s). I suggest using COPYA instead.

After using COPYA to copy the disk(s), use a bit copier with a Nibble-counter option to copy track 0 of side A. This will save you a lot of headaches later on.

Eric Kinney
Walla Walla, WA

Rumor Mill

Have you looked closely at any of the pictures of Apple's new Macintosh computer and noticed that all of its integrated circuits are soldered directly to the motherboard? Apparently the Apple IIe is to suffer the same fate

The January, 1984 edition of Apple Assembly Lines (S-C Software) reports that future Apple IIe's will have most of their chips soldered in, rather than mounted in sockets. Supposedly this will increase the reliabilty of the IIe's. Needless to say, this will make things hard for those who like to make their own repairs and/or modifications.

READERS' SOFTKEY & COPY EXCHANGE

Deprotecting Pandora's Box

By Clay Harrell

Pandora's Box
Datamost, Inc.
8943 Fullbright Ave
Chatsworth, CA 91311

Requirements:
Apple, with 48K
One disk drive, with DOS 3.3
A sector editor, such as the Inspector or Disk Zap
A blank disk
Pandora's Box

When program publishers buy a protection scheme, they generally use it for as many programs as possible to get the most for their money (yes, protection schemes are just programs that people write and sell).

The advantage of this is that once you learn what they are doing, it is easy to backup many of their programs.

Datamost and Infocom are two examples of this. If you can backup ZORK I, then you can backup ZORK II and ZORK III and DEADLINE and the rest.

The people at Datamost used a modified DOS for many of their programs until about April 1983. After that they started using a different scheme of protection on a lot of their programs. But the company published at least seven good games before April 1983 and Pandora's Box was one of them.

The deprotection method I am about to describe will apply to many of them, but we will be using Pandora's Box as an example.

Modified DOS

As we said, Datamost uses a modified DOS for its protection scheme. Normally, this is apparent from the BASIC prompt that appears on the screen after a few seconds into the boot.

Datamost, however, turns on the hires screen right when the boot starts to hide this. But we still know there is a modified DOS because of the way the boot sounds.

Listen to your normal DOS disks boot. You will hear the same sound every time. First, the drive spins for half-a-second or so. This is track 0, sector 0, through sector 9 getting loaded into $B600 to $BFFF.

Then, you hear the drive's read-write head slide up to track 2 to load in the rest of DOS. Tracks 2 and 1 are read in quickly and the read-write head slides up to the catalog, track 17, and the hello program is located and run.

Now, listen to Pandora's Box load in. You will hear the same sounds. This is a dead giveaway that a modified DOS is being used.

What to do

Whenever a modified DOS is used the first thing you should do is boot a normal DOS disk and defeat the DOS error-checking. DOS checks the carry bit to determine if any errors have occurred in a disk access.

If the carry bit is clear, DOS assumes that everything is OK and just keeps on going. The routine that gets jumped to if an error is suspected is at $B942. This simply sets the carry bit and returns to the calling routine.

To defeat the error-checking, we only have to change address $B942 from $38 to $18. This simple modification will allow us to copy previously uncopyable disks with COPYA.

All that is left to do is to change the Datamost DOS just slightly so that it will live in a normal DOS 3.3 environment. At track 0, sector 3, change byte $91 from $DF to $DE. What the manufacturers have done to make their disk "uncopyable" is to change the epilog byte from the normal $DE to $DF. This will sufficiently confuse the copy program, preventing easy copies. (If you do not know what is meant by an "epilog" byte, please refer to Beneath Apple DOS by Don Worth and Pieter Lechner. This manual is indispensible for further understanding of DOS).

In addition, byte $42 must be changed from $38 to $18 on the same track and sector. This is an insurance policy, more or less, that everything will work correctly in the normal DOS 3.3 environment.

What you are actually doing is changing byte $B942 in DOS, as we did before to make the COPYA copy, but you are doing it directly to the disk for permanence.

These two modifications are all we need to do to make Pandora's Box a straight COPYA disk.

The Steps

In step-by-step fashion, here's what you should do to make Pandora's Box copyable with COPYA:

1) Boot normal DOS 3.3

2) Enter the monitor

CALL-151

3) Change byte $B942 from $38 to $18 by typing:

B942:18

4) Execute the COPYA program

RUN COPYA

5) Copy Pandora's Box to a blank disk

6) Re-boot normal DOS 3.3 and run your sector editor. Change the following bytes:

Trk	Sct	Byte	From	To
00	03	42	38	18
00	03	91	DF	DE

7) Write the sector back out to your COPYA disk version of Pandora's Box.

Softkey For Donkey Kong

By Dan Lui

Donkey Kong
Atari, Inc.
P. O. Box 61657
Sunnyvale, CA 94086
$29.95

Requirements:
Apple, with 48K
COPYA
The Inspector, or similar program
One blank disk

Donkey Kong, another popular game from Atarisoft, contains all the four different structures (or screens) and plays in a similar fashion to the game that is featured in the arcades. It, also, uses a similar (if somewhat different) format of protection to the other programs that Atarisoft produces.

The procedure to unlock it is as follows:

1) Start the COPYA program

RUN COPYA

2) Once loaded, terminate the program with

CTRL-C

3) Delete line #70 to avoid the reload of the copy routine

70

4) Enter the monitor

CALL-151

5) Type the following:

B928: 18 60
B988: 18 60
3D0G

6) Execute the program

RUN

7) After the disk is copied, use the Inspector to read track 0, sector 3

8) Change the following:

Trk	Sct	Byte	From	To
0	3	$35	AA	DE
0	3	$3F	DE	AA
0	3	$91	AA	DE
0	3	$9B	DE	AA

The procedure is now finished and you can copy the disk with COPYA.

Backup For The Caverns Of Freitag

By C.J. Singer

The Caverns of Freitag
Muse Software
347 N Charles
Baltimore, MD 21201
$29.95

Requirements:
Apple II, with 48K and Applesoft in ROM
One disk drive
FID from 3.3 System Master disk
One blank disk
The Caverns of Freitag original disk

Being the parent of three kids who like to play games on the computer, my first thought after purchasing a program is how to back it up before the kids have an accident with the original.

What I usually first try to do is backup the disk with Locksmith, but I don't really care for this method because it doesn't allow you to look at the program to see how it works and it does not allow you to modify it.

Here's my first, albeit somewhat tedious, explanation of how to backup the Caverns of Freitag. I will provide another, easier method below.

The game comes from Muse Software, of Castle Wolfenstein fame.

The Hard Way

1) Boot the Caverns of Freitag disk from drive 6

PR#6

2) While the disk is booting, stop the HELLO program from running

CTRL-C

3) To make sure the HELLO program is in memory, type

LIST

You should see the Caverns of Freitag listed on your monitor. If no program is listed, go back to step 1 and try again.

4) The Caverns of Freitag uses a modified DOS, which has changed some of the DOS commands. The INIT command has not been changed, however, so we can INITialize a blank disk, which will contain the modified DOS along with the HELLO program currently in memory. So, insert a blank disk into your drive and type

INIT HELLO

If you want to see how Muse has modified the normal DOS commands you can use a sector editing program, such as ZAP from Bag of Tricks or DISKEDIT, to view track 1, sectors 7 and 8. If you compare what you find with a list of the normal DOS commands you will find that the SAVE command has been changed to LSDK, CATALOG to KSJFLKA, MON to 983 and BSAVE to 87364.

5) Now put your original disk back in the drive and use Muse's modified CATALOG (KSJFLKA) to see what files are on the disk

KSJFLKA

6) Write down the list of files which are displayed so that you can save them to the disk you just initialized. You don't have to write down the HELLO file since it has already been transferred.

7) The first file on your list should be a binary file called OILER. Load it into memory

BLOAD OILER

8) When a new binary file is loaded into memory, its address is stored at $AA72 and $AA73 (hi/lo format) and its length is stored at $AA60 and $AA61. Enter the monitor and display the address and length of the file OILER

CALL-151
AA72.AA73
AA60.AA61

For the OILER file you should get an address of $02DE and a length of $00D5.

9) Put your backup disk in the drive and use the 87364 (the modified BSAVE command) to save the file, using the address and length parameters you just determined in the previous step

87364 OILER,A$02DE,L$00D5

10) Repeat the process of BLOADing each binary file from the Caverns of Freitag original disk, determining its address and length and using the 87364 (BSAVE) command to transfer the files onto the backup disk.

11) For the Applesoft files on the list, you can just LOAD each one of them from the original disk and, then, LSDK (SAVE) them onto the backup disk.

Like I said; the above procedure is long and tedious, but it works.

Now for the easy method!

The Easy Way

1) Go through steps 1 through 4, outlined above.

2) Insert your DOS 3.3 System Master and boot it

PR#6

3) Load the FID program into memory at $6800 where it won't get overwritten when we boot the Caverns of Freitag disk in the next step

BLOAD FID,A$6800

4) Boot the original Freitag disk and, again, stop it from running the HELLO program

PR#6
CTRL C

5) Enter the monitor, memory move FID from $6800 back down to $803 and, then, run FID to copy the files

CALL-151
803<6800.9000M
803G

6) Copy the files from the original disk to the blank, initialized disk using FID's wildcard option.

Now that you have a backup copy you can look at the Caverns of Freitag, using CTRL C and KSJFLKA for CATALOG. You can LOAD or BLOAD to study or modify any of the files on the disk.

If you have a sector editor you may even want to change track 1, sectors 7 and 8, so that they will contain the normal DOS commands, instead of the ones which Muse put on the disk.

Short Softkey For Visifile

By C. Masters

Visifile
VisiCorp.
2895 Zanker Rd.
San Jose, CA 95134
$250.00

Requirements:
Apple, with 48K
One blank disk and a sector editing program

1) Copy the disk, using COPYA

2) Use the sector editor to read track $22, sector 4, of the backup disk

3) Change byte $2D from $0A to 0F and write the sector back to the disk.

Quicktrace
Anthro-Digital
103 Bartlett Ave
Pittsfield, MA 01201
$50.00

Requirements:
Apple II Plus
QUICKTRACE (program)

I'm sure that many of you who are Assembly language buffs have bought either The Bug (by Sensible Software) or Quicktrace (by Anthro-Digital) or both, like me. Each is a great tool for debugging or analyzing your own or another's programs.

But this isn't a review...

Quicktrace can display, "D," up to six memory locations for monitoring as well as having a full-speed JSR command, "X," which is excellent. The Bug has great display and disassembly commands, "Z" and "L."

The following program is an addendum to Quicktrace. It adds three new commands: L, Z, and Q, which are similar in function to those found in The Bug. Quicktrace and this addendum form Quick.Bug which has all the advantages of both programs together without suffering from the ommissions.

If you are wondering why I chose to modify or add to Quicktrace instead of The Bug, I can tell you that the latter is protected and, thereby, much harder to inspect or analyze.

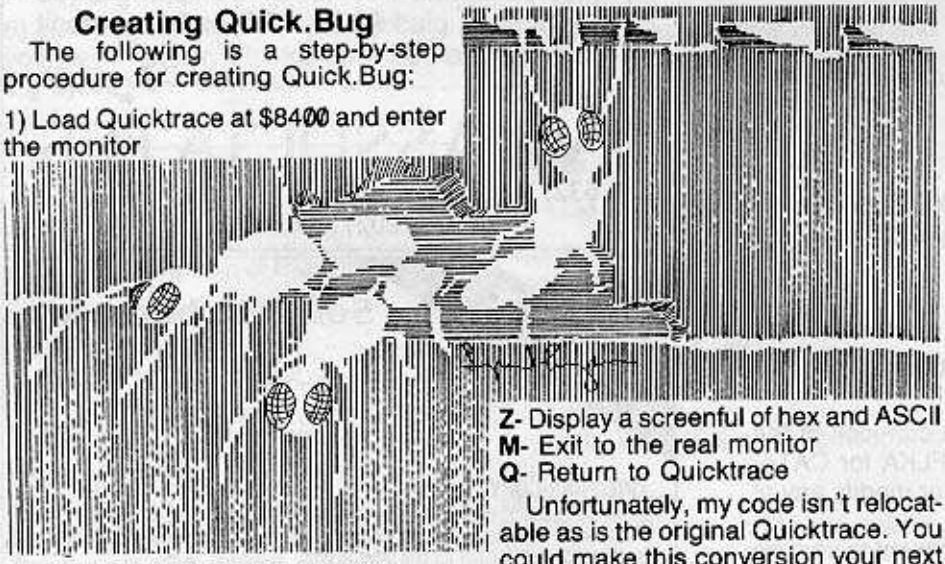
Program Enhancements: QUICK.BUG

By Enrique Gamez

Creating Quick.Bug

The following is a step-by-step procedure for creating Quick.Bug:

1) Load Quicktrace at $8400 and enter the monitor



BLOAD QUICKTRACE,A$8400
CALL-151

The default for Quicktrace's monitor command, "M," is $FF69. We must change it to $9200, where our addendum begins. The low byte is at $8551; the high byte is at $8564.

2) Make the default monitor vector point to the addendum

8551:00
8564:92

Now, the first time you press "M" from Quicktrace you will jump to the starting address of Quick.Bug.

3) At this point you should enter the hexdump on page 9.

4) Now all you have to do is save it

BSAVE QUICK.BUG,A$8400,L$EA8

That's it!

Here are the added commands you have once you enter Quick.Bug:

L- Disassemble 14 lines
Z- Display a screenful of hex and ASCII
M- Exit to the real monitor
Q- Return to Quicktrace

Unfortunately, my code isn't relocatable as is the original Quicktrace. You could make this conversion your next project, but I must report that I have had only one instance where memory conflicts forced me to load Quicktrace elsewhere. One is usually just concerned with analyzing sections of a program. Rarely would you ever want to step through a 120-sector game!

Try This One

But while we are on the subject of enhancements, here is one for Quick.Bug:

85EC:03 1A 09 04 02 2D 16 0E

Try this one too:

8575: 01 00 84 00 18 00 8D 00 19 00 E9 00 13 10 00 02 10 13

These two little modifications highlight the registers and other information with inverse characters. This greatly improves the display's readability but they also mess up a printout, alas!

Can't have everything...or can you?

Quick.Bug Source Code

```
1000 * QUICKTRACE ADDED COMMANDS
1010 * BY ENRIQUE GAMEZ
1020 * L= DISASSEMBLE $14 CMDS.
1030 * Z= DISPLAY HEX & ASCII
1040 * M= GO TO MONITOR
1050 * Q= RETURN TO QUICKTRACE
1060 * RTN= DO LAST CMD, DEFAULT
1070 *      TO 'Z'
1080 *--------------------------------
1090                .OR  $9200
1100                .TA  $800
1110 *--------------------------------
1120 CV             .EQ  $25
1130 INV.FLAG       .EQ  $32
1140 PROMPT         .EQ  $33
1150 PC             .EQ  $3A
1160 BGN.ADDR       .EQ  $3C
1170 GETLNZ         .EQ  $FD67
1180 ZMODE          .EQ  $FFC7
1190 GETNUM         .EQ  $FFA7
1200 BELL           .EQ  $FF3A
1210 CROUT          .EQ  $FD8E
1220 LIST           .EQ  $FE5E
1230 MONITOR        .EQ  $FF69
1240 XAM8           .EQ  $FDA3
1250 PRA1           .EQ  $FD92
1260 PRTBLNK        .EQ  $F94A
1270 COUT           .EQ  $FDED
1280 QUICKTRACE     .EQ  $8400
1290 *--------------------------------
1300                LDA  #$A4      ;USE '$' PROM
PT
1310                STA  PROMPT
1320                DEC  CV        ;DON'T SCROLL
 SCREEN
1330 ENTRY          JSR  GETLNZ    ;INPUT ROUTIN
E FROM MONITORROM, PG. 169
1340                JSR  ZMODE
1350                JSR  GETNUM
1360 *--------------------------------
1370 CMPARE         CMP  #$C6      =$39+$8D =CR
1380                BEQ  RETURN
1390                CMP  #$F3      =$19+$DA ='Z'
1400                BEQ  Z.ENTRY
1410                CMP  #$05      =$39+$CC ='L'
1420                BEQ  L.ENTRY
1430                CMP  #$EA      =$19+$D1 ='Q'
1440                BEQ  Q.ENTRY
1450                CMP  #$06      =$39+$CD ='M'
1460                BEQ  M.ENTRY
1470                JSR  BELL      ;NOT L,Z,M,Q
1480                DEC  CV
1490                JMP  ENTRY     ;TRY AGAIN
1500 *--------------------------------
1510 RETURN         LDA  CMD.FLAG  ;GET LAST CMD
 USED
1520                JMP  CMPARE    ;BRANCH TO EI
THER L OR Z
1530 L.ENTRY        STA  CMD.FLAG  ;SAVE IT
1540                LDA  #$0A      ;DISASSEMBLE
10 OPERATIONS
1550                JSR  LIST
1560                JSR  CROUT
1570                JMP  ENTRY
1580 Q.ENTRY        JMP  QUICKTRACE
1590 M.ENTRY        JMP  MONITOR
1600 *--------------------------------
1610 Z.ENTRY        STA  CMD.FLAG
1620
1630                LDA  #$14      ;# OF LINES T
O DECODE
1640                STA  ROWCOUNTER
1650 .7             LDA  BGN.ADDR+1
1660                STA  PC+1
1670                LDA  BGN.ADDR
1680                STA  PC
1690                EOR  #$07      ;TO ROUND OUT
 A LINE TO
1700                AND  #$07      ;NEAREST 8 BY
TES
1710                STA  COLCOUNTER
1720                EOR  #$07
1730                BEQ  .1
1740                JSR  PRA1      ;PRINT CR AND
 BGN.ADDR IN HEX
```

```
1750 .1   JSR XAM8      ;DOES ALL THE
 DECODING WORK!
1760      LDX #$2       ;FORMATTING
1770      JSR PRTBLNK   ;2 SPACES
1780      LDY #0
1790 .6   LDA (PC),Y
1800      BPL .2        ;<$80 ?
1810      CMP #$A0      CONTRL CHAR?
1820      BCC .3        ;YES, SKIP IT
1830 .8   CMP #$E0      NORMAL CHAR?
1840      BCC .4        ;YES
1850      LDX #$FF      ;SET NORMAL V
IDEO
1860      STX INV.FLAG  ;SEE PG.168 (
SFEB4)
1870 .3   LDA #$A0      ;A SPACE
1880 .4   JSR COUT      ;PRINT IT
1890      LDX #$FF      ;SET NORMAL V
IDEO
1900      STX INV.FLAG
1910      CPY COLCOUNTER ;ADJUST PRI
NT LOCATION
1920      BEQ .5        ;IF DONE
1930      INY
1940      BNE .6
1950 .5   DEC ROWCOUNTER
1960      BNE .7        ;IF NOT DONE
1970      JMP ENTRY
1980 *-------------------------------
1990 .2   CMP #$21
2000      BCC .3        ;IF CODE <$21
, SKIP IT
2010      ORA #$80      ;ALL OTHERS M
AKE NORMAL
2020      LDX #$3F      ;SET INVERSE
VIDEO
2030      STX INV.FLAG  ;SEE PG. 168,
 SFE80
2040      BNE .8
2050      JMP ENTRY     ;DONE, GO BAC
K AND WAIT FOR CMD.
2060 *-------------------------------
2070 CMD.FLAG   .DA #$F3      ;LAST CMD
2080 ROWCOUNTER .BS 1
2090 COLCOUNTER .BS 1
```

Quick.Bug

Hexdump

```
9200: A9 A4 85 33 C6 25 20 67 $2EF6
9208: FD 20 C7 FF 20 A7 FF C9 $EF99
9210: C6 F0 18 C9 F3 F0 2E C9 $1A8A
9218: 05 F0 16 C9 EA F0 20 C9 $96BD
9220: 06 F0 1F 20 3A FF C6 25 $B653
9228: 4C 06 92 AD A5 92 4C 0F $BB37
9230: 92 8D A5 92 A9 0A 20 5E $8BB4
9238: FE 20 8E FD 4C 06 92 4C $8D6F
9240: 00 84 4C 69 FF 8D A5 92 $6250
9248: A9 14 8D A6 92 A5 3D 85 $9D72

9250: 3B A5 3C 85 3A 49 07 29 $4680
9258: 07 8D A7 92 49 07 F0 03 $CD22
9260: 20 92 FD 20 A3 FD A2 02 $3E48
9268: 20 4A F9 A0 00 B1 3A 10 $4C88
9270: 25 C9 A0 90 08 C9 E0 90 $7E69
9278: 06 A2 FF 86 32 A9 A0 20 $1E32
9280: ED FD A2 FF 86 32 CC A7 $DDFC
9288: 92 F0 03 C8 D0 DF CE A6 $874B
9290: 92 D0 BA 4C 06 92 C9 21 $CFBB
9298: 90 E3 09 80 A2 3F 86 32 $05E3

92A0: D0 D3 4C 06 92 F3       $6424
```

ADVANCED PLAYING TECHNIQUES

Offing the SS

Contributed by Sean Williams

Castle Wolfenstein
Muse Software

If you are tired of running into S.S. guards, especially when you no longer have any grenades to destroy them with, use the following technique and any S.S. can be killed with the use of bullets.

1) As the S.S. guard approaches, point your gun at him and run into him.

2) As the screen is going through the collision routine, press the key which makes the gun point at the S.S. guard again.

3) To take his bullet-proof jacket away, press "U."

4) If the screen replies with "SEARCHING...," run into the guard again. (Remember to point your gun at him again during the collision routine).

5) Press "U" again and the guard will lose his bullet-proof jacket.

6) The guard can now be killed with plain bullets.

Note: This procedure works best if you are wearing a vest yourself!

Bountiful Bobs

Contributed by Johnny Yukon

Miner 2049'er
Microlab

For this APT you need to have an old F8 ROM on the motherboard or INTEGER card. This APT will let you choose the starting level and will give an unlimited number of Bounty Bobs to player number one.

1) Boot up Miner 2049'er and go through the joystick adjustment routine.

2) When the game asks "One or Two Players?" hit RESET to get into the monitor.

3) Enter the desired starting level minus 1 at $814 and the actual starting level at $812. For instance to start on level 05 type:

814:04 N 812:05

4) Enter the following:

0972:A9 03 8D 16 08 8D 17 08 4C 81 09 N 981G

5) The game will start up and player number one will have an unlimited number of Bounty Bobs.

Personalizing A Program

By Enrique Gamez

Once a program is unlocked by myself or someone else, I quickly dive in and "personalize" it.

What I do is change title pages or delete them to make the program load faster; I change or add default responses to entry prompts or I change the general layout of screen information to suit my particular application. I might even substitute my own shapes for someone else's shape tables (just for grins and giggles).

In games, I may want to change the scoring criteria, what level I am on, alter mazes, substitute sound effects, etc. There's all sorts of fun stuff one can do without being a programmer of genius. After all, it's a lot easier to alter another person's program than to write one from scratch.

You already know the program works in its original state (or we'll assume it does) so the trick is to change one thing at a time then run it and see if it still works. You may need to analyze a program a bit to make these alterations but you don't need to analyze, or even understand, everything to get started.

The very useful modification I will explain below shows how to change the controls of your favorite game. I'm sure most of you out there in Pac-land have played those games (mostly made in Japan) which force us to contort and/or disfigure our hands in order to reach the movement, thruster and fire keys.

There is a remedy

I'll venture to say that every game ever written for the Apple, which can be played with the keyboard, contains the following three bytes:
AD 00 C0.

As it turns out, these bytes are so common that I usually use their location as a starting point in my analyses of a program (game or otherwise). It is a good point of reference since these bytes are usually accompanied by several CoMPares (CMP) to see if certain keys were pressed.

A sample disassembly listing might look like this:

```
LDA $C000   ;GET A KEY CODE
BPL NO.KEY  ;EXIT, NO KEY HAS BEEN
            PRESSED
STA $C010   ;RESET THE KEY LATCH
CMP #UP     ;WAS IT THE UP KEY?
BEQ MOVE.UP ;YES, MOVE THE SHIP
            UPWARDS
CMP #DOWN   ;WAS IT THE DOWN
            KEY?
BEQ MOVE.DOWN ;YES, MOVE THE SHIP
              DOWNWARDS
```

All this brings us to my program and why I wrote it. Most commercial programs are upwards of 80 sectors long and a chore to flip through, 20 instructions at a time, using the monitor "L" command. My program, CRUCIAL CODE FINDER, allows you to type in the bytes you wish to search for and then speedily does this otherwise boring task. I suggest the use of this program whenever you're beginning an analysis and see how much time and eyestrain you'll save!

Once CRUCIAL CODE FINDER is installed all you need to do is (from the monitor) type the beginning address to search, the ending address and the bytes to search for. Of course, it may find erroneous code, because the occurance of the specified bytes might be part of a data table or something. But the valid code it finds is usually quite useful.

Using the Code Finder

Before we begin, type in the hexdump on page 12 and

BSAVE CRUCIAL CODE FINDER, A$2A1,L$FC

To install the code finder, type

BRUN CRUCIAL CODE FINDER

Once installed, the code finder is ready to take the drudgery out of program analyzing.

Here is the format to use when calling the code finder:

HHHH.GGGG^{CTRL}Y J1 J2 J3 J4...

"HHHH" and "GGGG" are four-digit hexadecimal numbers like 092F and D000, separated only by a period. These numbers, respectively, designate the beginning and ending memory locations you wish to search through. "^{CTRL}Y" is obtained by pressing the key marked "CTRL" and the key marked "Y" simultaneously and should come immediately after "GGGG." The "J1 J2 J3 J4..." is a string of any length, consisting of two-digit hexadecimal numbers represent-

ing the bytes you wish to find in the program. If you want, you may use X's instead of some of these numbers. In this case, the search routine will allow anything to match those bytes in the string.

For example, let's change that finger-contorting game I talked about.First, you would install the code finder, then you would BLOAD the menace and type something like this:

0800.9500CTRL-Y X 00 C0

The above command would find any references to location $C000. By changing the machine-language compares that undoubtedly follow to the values representing the keys you wish to use, your fingers will live happily ever after.

The first time through, it is a good idea to write down the places it finds that seem to be valid code and then examine them with the monitor.

After it finds the specified pattern, the program will display the instructions, starting with the matched bytes, and wait for a key to be typed. At this time, the "ESC" key will exit the search and place you back in the monitor; any other key will continue the search.

My theory is simple: let the computer do all the tedious work, leaving the creative, fun part for us. Have fun!

Crucial Code Finder Source Code

```
1000            .OR $2A1        ;JUST RIGHT S
O THAT THE REAL CODE WILL END UP AT $300
1010            .TF  CRUCIAL CODE FINDER
1020 * ------------------------------------
1030 * CRUCIAL CODE FINDER
1040 * BY ENRIQUE GAMEZ
1050 *
1060 * ENHANCED BY RAY DARRAH
1070 * ------------------------------------
1080
1090 A1L        .EQ $3C         ;LSB OF 'FROM
' LOCATION
1100 A1H        .EQ $3D         ;MSB
1110 A3L        .EQ $40         ;LSB OF 'TO'
LOCATION
1120 A3H        .EQ $41         ;MSB
1130 A2L        .EQ $3E         ;LSB OF TEMPO
RARY DATA
1140 A2H        .EQ $3F         ;MSB
1150 BUFF       .EQ $200
1160 BUFPTR     .EQ $FF
1170 GETNUM     .EQ $FFA7
1180 WILD.VAL   .EQ $DB         ;VALUE TO STO
RE FOR WILDCARD
1190 MONITOR    .EQ $FF69       ;ENTER THE MO
NITOR ROUTINE
1200 KEYBOARD   .EQ $C000
1210 KEY.RESET  .EQ $C010
1220 ESCAPE     .EQ $1B         ;KEY CODE FOR
 ESC KEY
1230 PCL        .EQ $3A         ;PROGRAM COU
NTER LSB FOR DISASSEMBLING
1240 PCH        .EQ $3B         ;MSB
1250 HOME       .EQ $FC58       ;ROUTINE TO C
LEAR THE SCREEN
1260 DISASEMBLE .EQ $FE63       ;ROUTINE THAT
 LISTS 6502 INSTRUCTIONS STARTING AT PCL
& PCH
1270 MODE       .EQ $31         ;TELLS MONITO
R WHAT HAS TRANSPIRED SO FAR
1280 CTRL.Y     .EQ $3F8        ;CONTROL Y VE
CTOR
1290 COUT1      .EQ $FDF0       ;PRINT ASCII
1300
1310 * ------------------------------------
1320 * CHANGE <CTRL>Y VECTOR
1330 * ------------------------------------
1340
1350 SETUP      LDA #$4C        ;MAKE THE CON
TROL Y VECTOR JMP
1360            STA CTRL.Y
1370            LDA #START      TO THE START
 OF THE PROGRAM
1380            STA CTRL.Y+1
1390            LDA /START      ;GET MSB
1400            STA CTRL.Y+2
1410            LDY #0          ;PRINT THE IN
SATLLED MESSAGE
1420 PRINT1     LDA MESSAGE1,Y  ;GET A BYTE
 TO PRINT
1430            PHA             ;SAVE IT IN C
ASE IT WAS THE LAST ONE
1440            ORA #$80        ;MAKE IT NORM
AL
1450            INY             ;NEXT OFFSET
1460            JSR COUT1       ;PRINT IT
1470            PLA             ;RESTORE WHAT
 IT WAS
1480            BMI PRINT1      ;IF GREATER T
HAN 128, THEN PRINT ANOTHER
1490            RTS             ;WERE DONE, R
ETURN
1500 MESSAGE1   .HS 8D8D        ;CARRIAGE RET
URNS BEFORE MESSAGE
1510            .AS -'CRUCIAL CODE FINDER INSTAL
LED:'
1520            .HS 8D8D        ;CARRIAGE RET
URN AFTER FIRST SENTENCE
1530            .AS -'USE<CTRL>Y FROM THE MON
ITOR'
1540            .HS 8D00        ;CARRIAGE RET
URNS AFTER ENTIRE MESSAGE
1550
1560 * ------------------------------------
1570 * BEGINING OF PROGRAM
1580 * ------------------------------------
1590
1600 START      LDA A2L         ;MOVE A2L & A
2H TO A3L & A3H
1610            STA A3L
1620            LDA A2H
1630            STA A3H
1640            INC MODE        ;TELL MONITOR
 NOT TO OVER-WRITE THE ADDRESSES
1650
1660 * ------------------------------------
1670 * CONVERT THE SEARCH BYTES
1680 * ------------------------------------
1690
1700            LDY #0          ;MAKE THE BUF
FER OFFSET ZERO
1710            STY BUFPTR
1720 FINDFIRST  INY
1730            LDA BUFF,Y      ;FIND THE FIR
ST DATA BYTE
1740            CMP #$99        ;IS IT A CONT
ROL Y?
1750            BNE FINDFIRST
1760 .1         INY             ;START WITH N
EXT BYTE
1770            LDA BUFF,Y      ;KEEP GOING P
AST THE SPACES
1780            CMP #$A0        ;SPACE
1790            BEQ .1
1800 CONVERT    JSR GETNUM      ;CONVERT THE
BYTE INTO HEX
1810            LDA A2L         ;RETRIVE THE
CONVERTED BYTE
1820            LDX BUFPTR      ;GET THE POIN
TER TO WHERE IT GOES
1830            STA BUFF,X      ;STORE IT
1840            INC BUFPTR      ;POINT TO NEX
T BYTE
1850            DEY             ;REPLACE Y WI
TH THE LAST CHARACTER GOTTEN
1860            LDA BUFF,Y      ;EXAMINE THE
NEXT BYTE
1870            INY
1880            CMP #$8D        ;FINISHED?
1890            BEQ START.LOOKIN;YES, BEG
IN THE SEARCH
```

```
1900              CMP#'X'+$80     ;IS IT A WILD CARD?
1910              BEQ WILDCARD
1920              CMP#$A0         ;IS IT A SPACE?
1930              BEQ CONVERT     ;YES, KEEP CONVERTING
1940 WILDCARD     LDA #WILD.VAL   ;STORE TWO CONSECUTIVE VALUES
1950              LDX BUFPTR
1960              INC BUFPTR      ;NEW LOCATION IN THE BUFFER
1970              STA BUFF,X
1980              DEX
1990              STA BUFF,X
2000              INY             ;GET THE NEXT BYTE FROM THE BUFFER
2010              BNE CONVERT     ;ALWAYS
2020
2030 *-------------------------------------------------------
2040 * SEARCH FOR THE SEQUENCE
2050 *-------------------------------------------------------
2060
2070 START.LOOKING
2080              LDY #0          ;START AT BEGINING OF POINTER
2090              LDX #0          ;START AT BEGINING OF INPUT BUFFER
2100 TRY1         LDA BUFF,X      ;GET BYTE TO LOOK FOR
2110              CMP#WILD.VAL    MIGHT BE A WILDCARD
2120              BNE NOT.WILD
2130              INX             ;SEE IF TWO CONSECUTIVE WILD VALUES
2140              CMPBUFF,X       ;MATCH?
2150              BEQ FOUND1.MATCH;YES, SIMULATE MATCH
2160              DEX             ;NOT A WILDCARD SO RESTORE X
2170 NOT.WILD     CMP(A1L),Y      ;DOES IT MATCH THE CURRENT LOCATION?
2180              BNE NEXT.BYTE   ;NO, TRY NEXT BYTE
2190 FOUND1.MATCH
2200              INX             ;GET NEXT BYTE FROM BUFFER
2210              INY             ;GET NEXT BYTE FROM MEMORY
2220              CPX BUFPTR      ;COMPLETE MATCH?
2230              BCC TRY1        ;NO, TRY TO MATCH THE NEXT BYTE
2240              JSR HOME        ;CLEAR THE SCREEN
2250              LDA A1L         ;MOVE A1L & A1H TO PCL & PCH
2260              STA PCL
2270              LDA A1H
2280              STA PCH
2290              LDA #23         ;DISASSEMBLE23 LINES
2300              JSR DISASEMBLE  ;GO DO IT
2310
2320 WAIT         BIT  KEYBOARD   ;WAIT FOR A KEY
2330              BPL WAIT
2340              STA KEY.RESET   ;RESET THE KEYBOARD STROBE
2350              LDA KEYBOARD    ;GET KEY VALUE
2360              CMP#ESCAPE      ;IS IT AN ESCAPE?
2370              BNE NEXT.BYTE
2380 EXIT         JMP MONITOR     ;ESCAPE WAS PRESSED SO JUMP TO THE MONIT
OR
2390
2400 *-------------------------------------------------------
2410 * GET THE NEXT ADDRESS TO SEARCH
2420 *-------------------------------------------------------
2430
2440 NEXT.BYTE    INC A1L         ;GET THE NEXTO BYTE TO SEARCH
2450              BNE .1          ;START BY INCREMENTING LSB THEN MSB IF NE
EDED
2460              INC A1H
2470 .1           LDA A1H         ;COMPARE T SEE IF RINISHED
2480              CMPA3H          ;REACHED DESTINATION?
2490              BCC START.LOOKIN ;NO, SEARCH HERE TOO
2500              LDA A1L         ;DOES THE LSB MATCH AS WELL
2510              CMPA3L
2520              BCC START.LOOKIN ;NO, KEEP SEARCHING
2530              BCS EXIT        ;IF DONE, JUMP INTO THE MONITOR
```

Crucial Code Finder Hexdump

```
02A1: DA FD 68 6A 45 0B 2A          $6BDB
02A8: 45 0C 85 0B 45 0C 6A 85       $3027
02B0: 0C 20 DC 02 90 BD A2 03       $7149
02B8: 20 4A F9 A9 A4 20 ED FD       $795E
02C0: A5 0B A6 0C 20 41 F9 20       $9EAB

02C8: 8E FD 60 A4 FD A6 FC 20       $5C3B
02D0: 8E FD 20 40 F9 A0 00 A9       $87D6
02D8: BA 4C ED FD A5 FC C5 FE       $B30E
02E0: A5 FD E5 FF E6 FC D0 02       $2E86
02E8: E6 FD 60 BE D9 A0 C6 D2       $5BB3
```



```
02F0: CF CD A0 D4 C8 C5 A0 CD       $157A
02F8: CF CE C9 D4 CF D2 8D 0D       $DA98
0300: A5 3E 85 40 A5 3F 85 41       $4E46
0308: E6 31 A0 00 84 FF C8 B9       $7639
0310: 00 02 C9 99 D0 F8 C8 B9       $7809
0318: 00 02 C9 A0 F0 F8 20 A7       $51FA
0320: FF A5 3E A6 FF 9D 00 02       $7A22
0328: E6 FF 88 B9 00 02 C8 C9       $142A
0330: 8D F0 18 C9 D8 F0 04 C9       $FC1F
0338: A0 F0 E3 A9 DB A6 FF E6       $2CB2

0340: FF 9D 00 02 CA 9D 00 02       $8687
0348: C8 D0 D3 A0 00 A2 00 BD       $CC63
0350: 00 02 C9 DB D0 07 E8 DD       $4955
0358: 00 02 F0 05 CA D1 3C D0       $62F0
0360: 28 E8 C8 E4 FF 90 E8 20       $A97B
0368: 58 FC A5 3C 85 3A A5 3D       $9625
0370: 85 3B A9 17 20 63 FE 2C       $62F6
0378: 00 C0 10 FB 8D 10 C0 AD       $3EBC
0380: 00 C0 C9 1B D0 03 4C 69       $659F
0388: FF E6 3C D0 02 E6 3D A5       $5811
0390: 3D C5 41 90 B6 A5 3C C5       $D4D3
0398: 40 90 B0 B0 E9                $3EC0
```

MODIFIED ROMS

By Ernie Young

Requirements:
48K Apple II Plus
One disk drive
A supply of 2716 16K EPROM chips
Several 24-pin, low-profile sockets
Access to an EPROM burner

SoftKey Publishing assumes no responsibility for any damage done to the computer while following this procedure.

In the article "Hidden Locations Revealed" (HARDCORE COMPUTIST 3, p. 10) the author addressed the idea of what to do when performing a softkey on a program that uses the text page for valid code use.

He suggested a small hardware modification to the Apple to enable one to see the text page of a program while another page (i.e., hi-res page) was supposed to be displayed.

What I am about to discuss is an inexpensive modification to your computer that is about as good (maybe even better) than those cards that save all of memory. Needless to say, this is a very good means to achieve that end on programs which use volatile (easily eraseable) memory.

Volatile memory includes such locations as a great deal of page zero, page one, page two and the text page location at $400 to $7FF. For example, as you type and characters are echoed onto the text screen, locations $400-$7FF change. Each ASCII (American Standard Code for Information Interchange) character is represented on the screen in one of the locations in the range of $400 to $7FF.

In most softkeys it has been assumed that by hitting "RESET" (with the old-style monitor, of course) and dropping into monitor, we can snoop through memory and find what we need to save or disregard.

In the softkey entitled "Hidden Locations Revealed," the idea was addressed that the text page and the keyboard input buffer (page 2) were used for valid program storage and that upon resetting, these volatile memory locations would be destroyed.

Indeed, it is no new trick to many software publishers that using the text page and other volatile memory locations is a good way to keep the public from snooping through their programs and possibly reducing them to a more copyable form.

Hitting "RESET"

To understand this more clearly, let's examine what happens when you hit the "RESET" key.

Instead of going through the input latch at $C000 as the other keys on the keyboard do, the "RESET" key is connected directly to pin 40 of the 6502 microprocessor chip. This is the big long chip just in front of the slots on the motherboard.

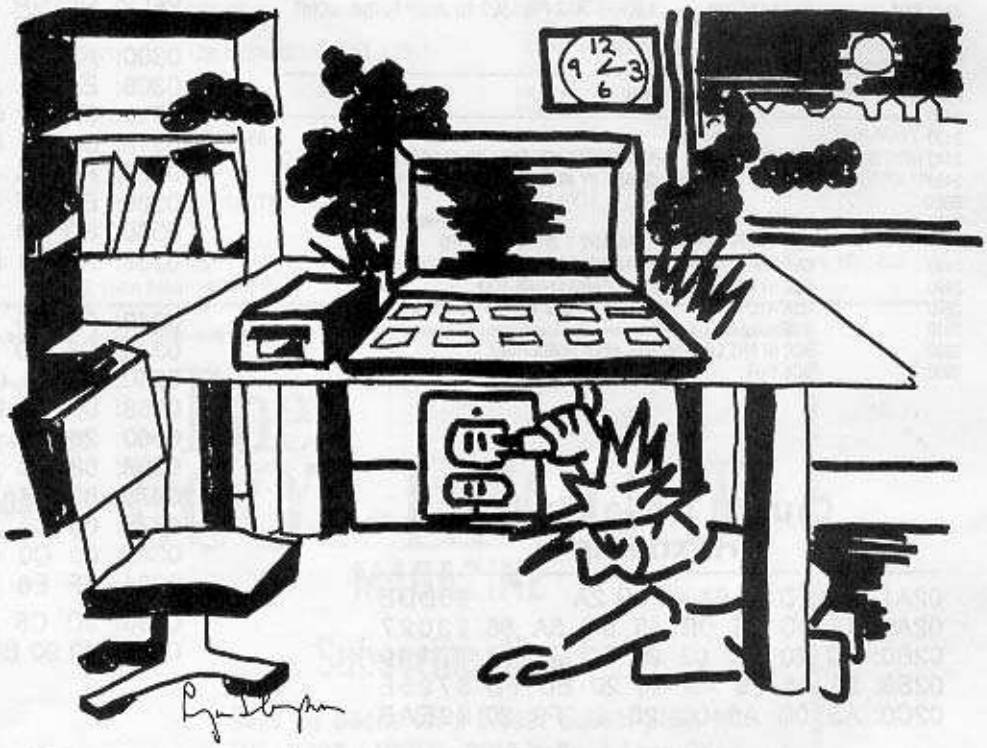
When "RESET" is pushed, pin 40 gets connected to ground and the computer unconditionally jumps to the address contained in locations $FFFC and $FFFD. These locations are in the F8-monitor ROM and, depending on which F8-monitor ROM you have, your computer can do one of two things.

There are two flavors of monitor ROMs in the Apple world, known as the "autostart" and the "old-style" monitor ROMs.

In the "autostart" F8-ROM, which is used in the Apple II Plus model, these locations point to a series of routines that check to see if the computer is just being powered up or if "RESET" had been typed before. If it finds the power already on, it jumps to the routine pointed to by locations $3F2 and $3F3. Normally, they point to the BASIC warmstart and, therefore, you get the Applesoft prompt.

The "old-style" monitor found in the older Apple II models has $59 and $FF stored in locations $FCCC and $FCCD. This causes the Apple to jump to the routines at $FF59. These routines set the keyboard for input, the monitor (or TV) for output, the text page for viewing and, finally, it puts you in the machine language monitor with an asterisk prompt.

In the "autostart" ROM, anyone can program where they want their Apple to go, when the "RESET" key is pressed, by changing the code at $3F2 accordingly. In the "old-style" monitor, however, there is no way to prevent a reset from occuring and, eventually, giving you the monitor prompt. This is obviously essential if you want to break into a program to examine the code but, as mentioned in "Hidden Locations Revealed," this is no guarantee to performing a working softkey.



Destructive "RESET"

The problem is that when we reset into the monitor, many locations are destroyed. These locations may or may not be essential to the program's operation.

For example, the following zero page locations are destroyed by an "old-style" reset: $20-$29, $32, $33, $35-$39 and $48.

But, even worse than that, the entire text screen scrolls up one line when the monitor prompt is printed. This scrambles locations $400-$7F8 and destroys locations $400-$427 completely.

Most software publishers know this and use it against us to keep us, "unwanteds," from snooping through their code. This becomes evident if you hit "RESET" and find that the text page is filled with garbage and other incomprehensible junk.

Ultimately, we would like to be able to save locations $00 to $8FF in non-volatile memory upon hitting "RESET" and, then, to exit to the monitor for examination. I include $800-$8FF, even though it doesn't get destroyed on resetting, because it gets wiped out when we do a 48K slave disk boot. The best place to store this information would be at $2000-$28FF, since this is normally the hi-res page used in most games and is not destroyed by booting a slave disk. (Remember that $00-$8FF and $9600-$BFFF are destroyed when booting a 48K slave disk).

In order to save these volatile memory addresses into locations $2000-$28FF, we need to change what normally happens when we hit "RESET." This may seem impossible to do since what happens when we hit that key is predetermined in ROM and, therefore, is not changeable.

Well; yes and no.

We can copy the code from the F8-ROM down to RAM, change it and burn it into a new ROM!

Help from EPROM

Of course, this assumes that you have access to an EPROM (Eraseable, Programmable, Read Only Memory) programmer and some 2716 EPROM chips.

Most computer stores that are worth anything will be willing to burn you a new 2716 chip for a reasonable fee if you do not have your own access to an EPROM programmer. These 2716 chips are available from many sources. See the back of any BYTE magazine for names of suppliers of the

EPROM if you can't find any locally in your area

So, assuming you have access to these two resources, all we have to do is alter the normal F8-code.

First, let's develop the code that we will need to jump to when the "RESET" key is pressed, moving memory from $00-$8FF to $2000-$28FF. This requires some knowledge of assembly language but if you are not familiar with machine code try and follow along anyway.

(While going to the extreme of burning new ROM, it would also be a good idea to change the NMI (non-maskable interrupt) vector to point to our new routine).

1. Let's start by moving the code at $F800-$FFFF down to RAM. Location $2800 would be a good place for it

2800<F800.FFFFM

For our next step, we need a place to put our "Super Saver" routine. We don't have any open memory locations, but we do have some routines in the monitor code which we never use, like the tape read and write routines. These will have to be sacrificed for our new routines which are printed page 28.

To enter this code, CAREFULLY type in the hexdump on page 28.

Now, change the RESET and the NMI vectors in our RAM version of the F8-ROM

2FFA:CD FE CD FE

Save the file onto a disk with the command:

BSAVE MODIFIED F8.ROM, A$2800,L$800

Now, burn your new EPROM!

Plugging-in your New F8 Monitor ROM

In order to use your new, modified, ROM, we must install it in the motherboard (or in an integer card).

But, first, we have to make one explanation: both the 2716 you just burned and the original 9316 ROM, used by Apple, are Read-Only-Memory devices, containing 2K bytes of information, which gives us 16K bits of information. Hence, 16K ROMs. But, unfortunately, they are not totally compatible. The arrangement of the pin numbers are slightly different.

To use your new EPROM, you must make these changes directly to the chip itself (not advisable), or to a jumper socket which your new chip will plug into. This, then, will be plugged into the motherboard, or integer card.

For the latter, you will need a 24-pin, low-profile socket, which is available from Radio Shack or similar stores.

With the socket upside-down and the pins looking you in the face, it should look like figure 1.

Figure 1

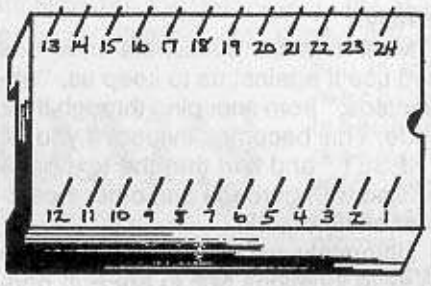
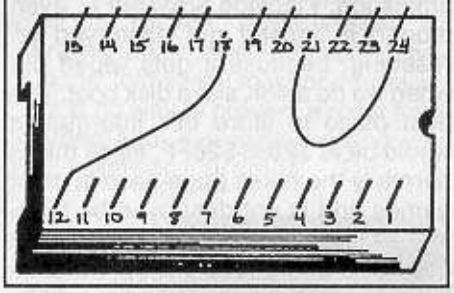


Figure 2



Your soldering skills come in handy now. Using some short, hi-gauge wire (wire-wrap is preferrable, but anything in the 26-30 gauge will work), solder a piece between pins 21 and 24 and then solder a piece between pins 12 and 18.

Be extremely careful not to short out the wire or to cross-solder any pins! Also, try and solder as close to the base of the socket as possible, since you have to cut off pins 18 and 21 after you have finished soldering them.

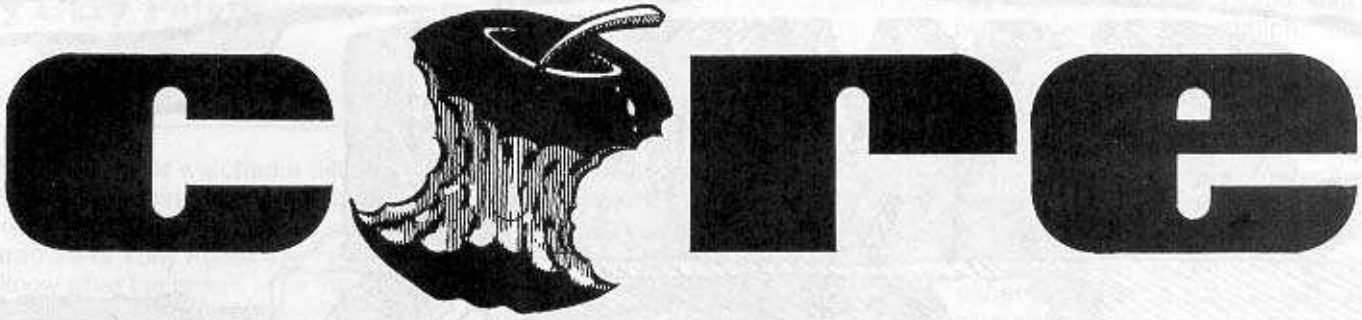
The next step is to cut off pins 18 and 21 as close to the base of the socket as you can, without cutting the freshly soldered wires. Remember that pins 18 and 21 should be short enough so they will not touch the socket you will be plugging this one into.

The socket should now look like figure 2.

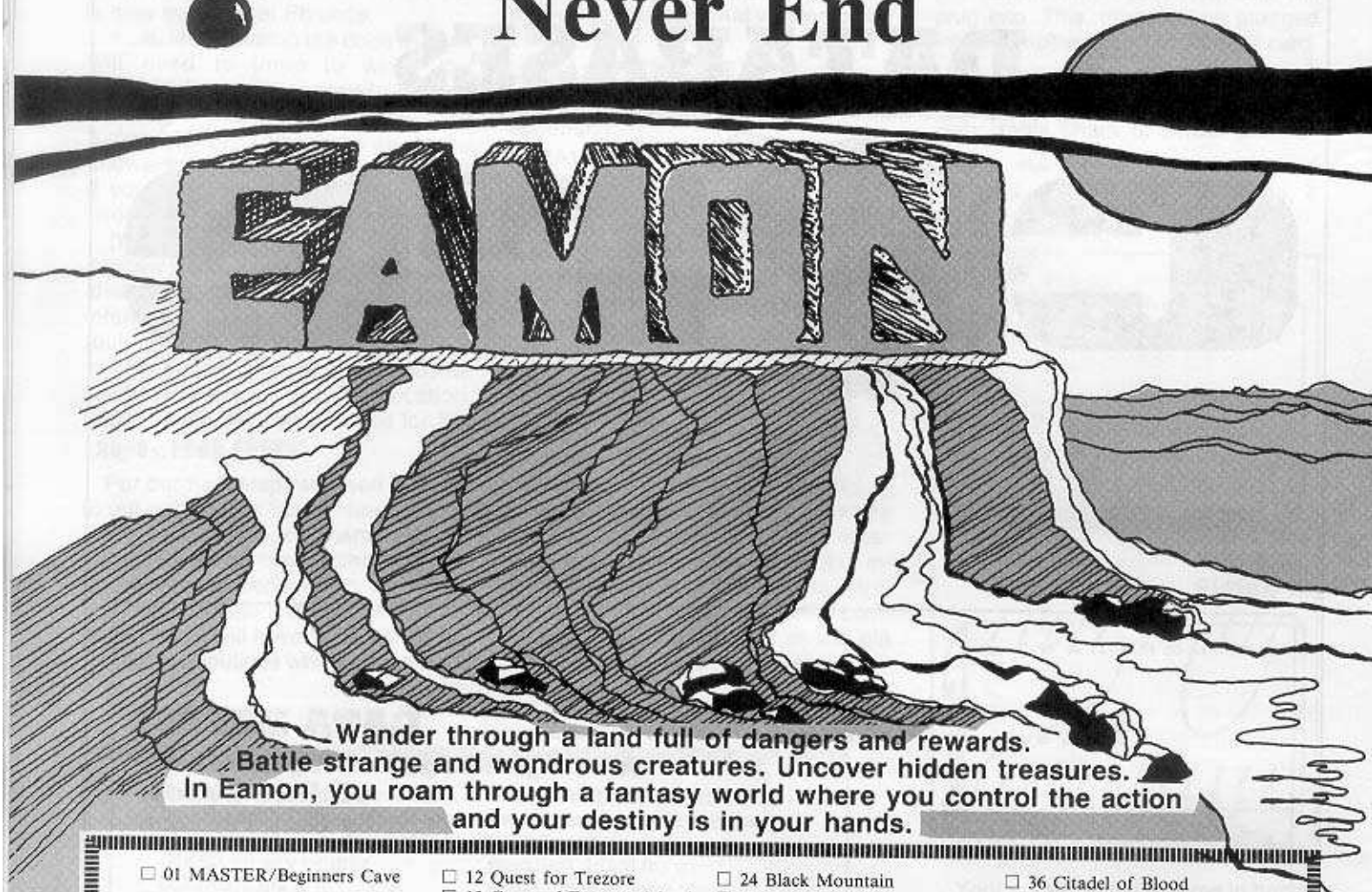
Double-check your soldering and

Hardcore continued on page 28

DATABASES



Where The Adventures Never End



**Wander through a land full of dangers and rewards.
Battle strange and wondrous creatures. Uncover hidden treasures.
In Eamon, you roam through a fantasy world where you control the action and your destiny is in your hands.**

- ☐ 01 MASTER/Beginners Cave
- ☐ 02 Lair of the Minotaur
- ☐ 03 Cave of the Mind
- ☐ 04 Zyphur River Venture
- ☐ 05 Castle of Doom
- ☐ 06 Death Star
- ☐ 07 Devil's Tomb
- ☐ 08 Abductor's Quarters
- ☐ 09 Assault of the Clone Master
- ☐ 10 Magic Kingdom
- ☐ 11 Tomb of Molinar
- ☐ 12 Quest for Trezore
- ☐ 13 Caves of Treasure Island
- ☐ 14 Furioso
- ☐ 15 The Heroes' Castle
- ☐ 16 Caves of Mondamen
- ☐ 17 Merlin's Castle
- ☐ 18 Hogarth Castle
- ☐ 19 Death Trap
- ☐ 20 The Black Death
- ☐ 21 Quest for Marron
- ☐ 22 Senators' Chambers
- ☐ 23 Temple of Ngurct
- ☐ 24 Black Mountain
- ☐ 25 Nuclear Nightmare
- ☐ 26 Assault on the Moleman
- ☐ 27 Revenge of the Moleman
- ☐ 28 Tower of London
- ☐ 29 Lost Island of Apple
- ☐ 30 Underground City
- ☐ 31 Gauntlet
- ☐ 32 House of Ill Repute
- ☐ 33 Orb of Polaris
- ☐ 34 Death's Gateway
- ☐ 35 Lair of the Mutants
- ☐ 36 Citadel of Blood
- ☐ 37 Quest for the Holy Grail

Tournament Adventures

- ☐ 60 Castle of Count Fuey
- ☐ 61 Search for the Key
- ☐ 62 The Rescue Mission

EAMON Utilities

- ☐ 01 EAMON Utilities
- ☐ 02 EAMON Utilities
- ☐ 03 EAMON Utilities
- ☐ Dungeon Designer Ver 5

Enclose $4.00 for each EAMON scenario volume that you order or see our special EAMON collectors offer.

The MASTER disk is required to play any EAMON scenario. HC6

Adventure scenarios are packed on double-sided disks. Minimum Order: 2 Volumes. ($8.00)

Name _____

Address _____

City _____ State _____ Zip _____

Country _____ Phone _____

☐ VISA ☐ MC #_____ Exp Date _____

Signature _____

U.S. funds only. Make checks payable to: Computer Learning Center

SPECIAL COLLECTORS'S OFFER
offer ends 4/31/84

Every EAMON disk (including Utilities) ALL for only $140.

☐YES! Send me the entire EAMON collection.

Make checks payable to:
Computer Learning Center
P.O. Box 45202
Tacoma, WA 98445

No purchase orders or C.O.D.
Washington state residents add 7.8% sales tax.
Foreign orders add 20% for shipping and handling.

Data Bases

By Gary Peterson

Have you ever watched a televised baseball game on Saturday afternoon and wondered how it is that Joe Garagiola or Tony Kubek just happens to know what the record is for the most inside-the-park home runs by a Presbyterian short stop? Or, perhaps, is it Howard Cosell who never ceases to amaze you with his total recall of the statistics of virtually every sport ever played by man?

Of course these sports announcers do not have super-human recall facilities. Instead, they have the services of a computerized data base, which is likely to be sitting in a van in the stadium's parking lot. Whenever the announcers need a particular tidbit of information they can relay their request to the van where the computer's operator can query the data base and, hopefully, produce the desired statistic.

Down on the playing field the players and their managers are not being left out of the computer revolution either. It is becoming more and more commonplace for professional baseball and football teams to have a personal computer containing a data base of information about both their team and the opposing team. The managers and coaches can call on the facilities of the data base in decision-making situations.

For instance, the manager of a baseball team may want to know which of his pinch hitters has the best night-game, clutch-hitting percentage against a particular left-handed pitcher known to throw screwballs. The defensive coach of a football team may want to know what play the opposing team is likely to run on a 3rd and 3 situation.

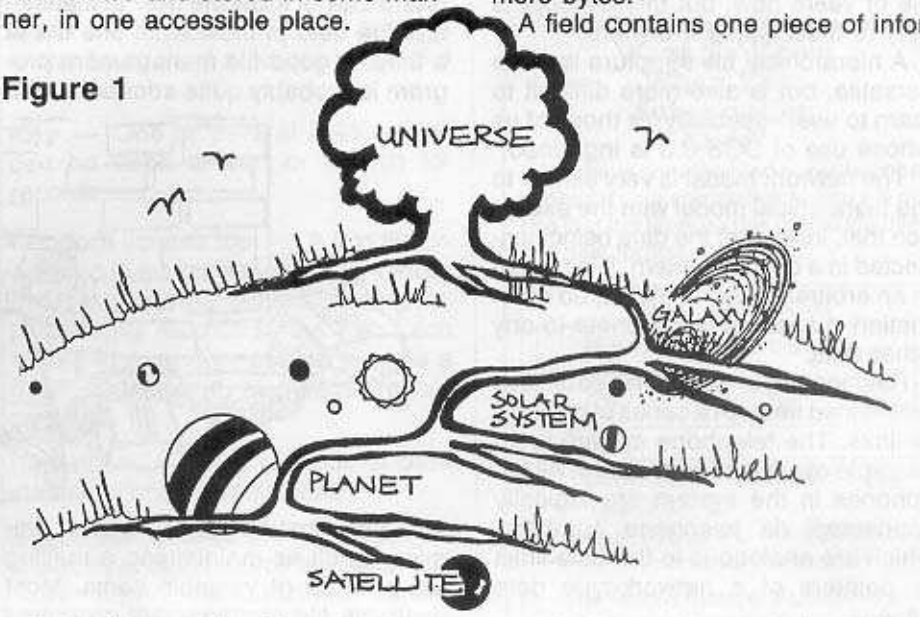
If the computer's data base is extensive enough this information can be obtained and in the long run it will give the team an overall percentage edge. This edge could be just enough for a team to make it to the post-season playoffs or bowl games.

Sports statistics are just one example of the use of a computerized "data base," a computer industry buzzword that is being heard quite frequently these days.

From the way everybody has suddenly started using it, one could easily get the idea that no data bases ever existed prior to the advent of the computer. In reality, however, data bases have existed since man first began to use written symbols to record grain inventories, herd counts, the number of wives in a harem, taxation data and any other information needed by society.

A data base is simply data which is interrelated and stored in some manner, in one accessible place.

Figure 1



A good example of a commonly used data base is the card catalog found in any library. Each card contains information pertaining to a book's title, author, subject, catalog number, publisher, etc. Every book in the library will have at least three entries in the card catalog: one filed by the title, one filed by the author and one filed by the subject of the book. Thus, each book is sorted alphabetically according to three different pieces of information (keys).

It is only a fairly recent innovation that computers have been used to store and manipulate the data contained in a data base. With the use of computers, vast amounts of information can be stored on tape or disk and that data can be searched for any specific information, or relationship between pieces of information, that is desired.

The computer software and hardware which allow this are called a data base system. A Data Base Management System (DBMS) is the software which allows the user to build, maintain and manipulate the data base.

A data base program can be written either to perform a specific task, such as maintaining a mailing list, or to enable the user to configure the data base for particular needs. Obviously, a data base program falling into the second category is much more versatile, but it can be more time-consuming to set up and learn to use.

The basic unit of information in a computerized data base is the byte. A byte, generally, will store one numeric or alphanumeric character. A field is the next highest piece of information and is composed of a series of one or more bytes.

A field contains one piece of information. For example, a simple mailing list would contain the following fields:

Field 1	(Name)	John Doe
Field 2	(Street)	220 Main St
Field 3	(City)	Fat City
Field 4	(State)	FL
Field 5	(Zip)	88888

These five fields, taken together, would constitute one record of the mailing list data base. A collection of

records constitutes a file. The number of records contained in the mailing list file would be equal to the number of names on the mailing list.

There are three generally-accepted, logical models on which a data base program can be established: hierarchical, network and relational.

Due to the similarities between the network and the hierarchical models some authors group the two together and call the combined scheme a network/hierarchical data base.

In the hierarchical model of a data base, the data is viewed as having a clearly-defined structure, or hierarchy, with various levels from top to bottom. This model is, sometimes, called a tree because of the resemblance to a tree which appears when the structure is inverted. A graphic example of this type of structure can be seen in figure 1 which shows a planet's "address" in the universe.

Apple's new operating system, ProDos, is designed around a hierarchical file structure. What this means is that files will be broken down into subcategories related to the type of information in a file. Each of these subcategories will have its own subdirectory. This type of file structure has been used on the Apple III for a couple of years now, but this concept is new to most Apple II owners.

A hierarchical file structure is more versatile, but is also more difficult to learn to use, especially for those of us whose use of DOS 3.3 is ingrained.

The network model is very similar to the hierarchical model with the exception that, instead of the data being connected in a definite pattern, it is related in an arbitrary manner. Thus, no information is seen as subordinate to any other data.

Relationships between data are maintained through a series of pointers or links. The telephone system is an example of a network structure. All telephones in the system are logically connected via telephone numbers, which are analogous to the data links or pointers of a network-type data base.

The relational model is the most frequently implemented and easily understood of the data base types. In the relational model, all the data is organized into a table of rows and columns. Record information is held in the rows of the table, while field information is held in the columns.

Sometimes the columns of such a data base table are called tuples, while the rows are termed domains. Personally, I have enough of a problem keeping the rows and columns straight without worrying about tuples and domains.

Because of the relative simplicity of using and writing a relational data base program, this is the type used most frequently on microcomputers. Network and hierarchical data bases are, ordinarily, used on mini- and mainframe computers, where the amount of data being stored and manipulated is fairly large. The majority of data base management packages offered for sale on the microcomputer market today are relational.

Relational data base programs can fall into one of two general categories, based upon the features incorporated into the particular package: either a fairly simple file management system or, if it has sufficient features, a relational data base management system (RDBMS).

A file management system is the simplest implementation of a data base program. A file management program will allow the user to define the file structure, add, edit or delete fields and records, search the file for specific information and produce printouts and reports of the information on file.

Generally, file management programs do not allow the user to manipulate the data in more than one file at a time. A good file management program is probably quite adequate for a home application of a data base program, such as maintaining a mailing list or a list of valuable items. Most available file management programs are menu driven, meaning that the user simply picks the desired operation from a menu presented when the program is run.

The public domain program File Cabinet, which has been available since the early days of the Apple computer, is an example of a file management program. Other file management programs for the Apple include Visifile (Visicorp) and PFS:File (Software Publishing Corp).

A relational data base management system (RDBMS) contains most of the features of a file management system, plus some additional features. Most relational data base systems are capable of manipulating more than one file at a time, have password protection and can be set up for multi-user access.

Other RDBMS features might include built-in report generators, a mode to restructure records, a query facility to enable non-programmers to access the data base information easily and a system of reading files of any format.

Most RDBMS packages have a built-in programming language and/or the ability to interface with a host language, such as BASIC or Assembler.

With a good RDBMS package, an experienced programmer can write a customized and integrated computerized accounting system for a small business. Some of the RDBMS packages available for the Apple are General Manager (Sierra On-Line), dBASE II (Ashton-Tate; Z80 card required) and DB Master (Stoneware).

It is not necessary for the home computer user to buy or write special software in order to use data bases. With the use of a modem and a subscription to one of the information services, such as The Source, Dialog or Compuserve, a person can access a variety of data bases containing information for both professional and personal use. Practically any information which is publicly available can be found in the proper data base.

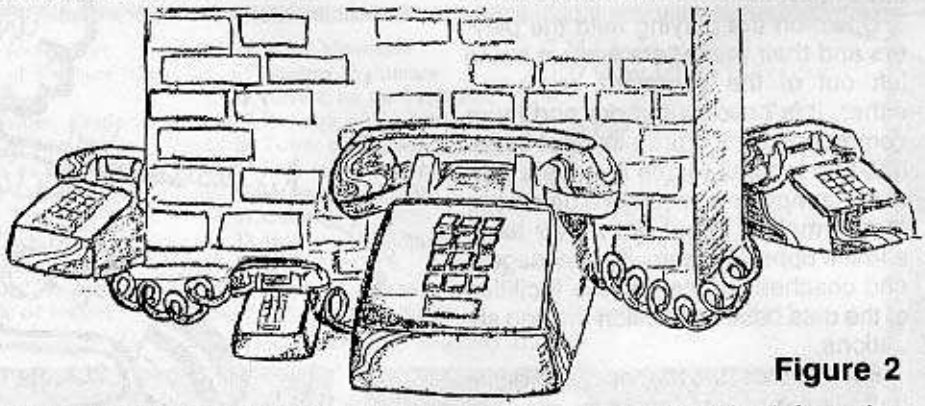


Figure 2

There are many other applications of data bases which cannot be covered in one short article such as this. Several books are available with information on microcomputer data base use, in addition to the many books aimed at users of mainframe data bases.

Several periodicals, also, contain information for data base users and programmers. One of the more popu-

lar data base packages, dBASE II, even has an entire magazine devoted to its use.

Other data base related topics covered in the contents of CORE this issue and the next few issues will include a review of a specific Data Base Management System along with a complete listing of a general-purpose file management system.

Data Base Definitions

ASCII — Acronymn for American Standard Code for Information Interchange. Each character in the system is assigned a unique number. For example, the ASCII code for the letter A is the decimal number 65 or hexadecimal $41. The Apple and most other computers use this method of character representation.

Byte — The smallest piece of information in a record. A byte normally holds one character.

Data base — A grouping of data which is interrelated. The data base may contain indices, pointers and a dictionary for the data. Generally, the entire data base is stored on either a floppy, or a hard disk system.

Database — Generally, database refers only to the information (i.e. the files) contained in the data base (i.e. the records and their related aspects). In other words, the database is contained in the data base.

Data base management program — A program or set of programs which allow the user to access and manipulate the information in the data base.

Domain — Refers to a column of the data table in the relational data base model.

Encryption — A method by which data can be transformed so that it is meaningless to someone who does not know the method of transformation.

Field — An individual data item within a record. Each field has a defined length and characteristic (numeric or alphanumeric). The field length is the number of bytes contained in the field.

File — A collection of data which is normally stored on disk. A file contains a series of individual records.

Index — A table containing pointers to the location of a record within the data base.

Key — One or several fields which can be used to sort or search for records.

Random access file — A text file organized in such a way that any record may be accessed without reading the preceeding records in the file. Each record in a random access file has a fixed length and a unique record number.

Record — A group of fields or data elements which are related.

Sequential access file — A text file whose individual records are not of any fixed length. A carriage return (CHR$(13)) indicates the end of each record. Individual records in a sequential text cannot be accessed easily without reading each preceeding record in the file.

Schema — The logical structure of a particular data base.

Tuple — Refers to a row of the data base table in the relational data base model.

Want To Know When Your Subscription Ends?

We have standardized our subscriber codes for your convenience. You will notice that the label on this magazine uses the new format.

Take A Look!

Name Record#
Address
City St Zip

Ex.

John Doe FL/599/10 (S)
123 Main St
Big Town, FL 33950

The record number FL/599/10 (S) reads: state/record/ENDING ISSUE/ and S indicates subscription. The ENDING ISSUE for this label is Hardcore COMPUTIST #10.

Now you can look at your label and determine when your subscription wil end. Consecutive numbering will begin with this issue: Hardcore COMPUTIST #6.....ending code 6.

Your record number may change, so please note the most current label when writing for information.

Data Base Management Systems

By Monty Lee

A Data Base Management System (DBMS) can be defined as a centralized collection of data stored for use in one or more applications, utilizing hardware and software facilities, to produce information in a display or report, hard-copy format. The goal of a DBMS is to process personal and business applications for the microcomputer's user, with some of the approaches employed by users of large computer systems.

The most basic building block of a data base consists of "fields." These are units of information, such as the name of a book in a library catalog. The next building block is the record which consists of a number of related fields, involving the item being cataloged.

Obviously, a DBMS is only as good as its features. What constitutes good features is an open debate among all DBMS programmers, but, generally speaking, there are several musts for an effective data base program. These include DOCUMENTATION, EASE OF USE, DATA ENTRY, DATA SEARCHING AND RETRIEVAL, REPORTING CAPABILITY, FILE STRUCTURE and other, MISCELLANEOUS FEATURES.

In order to understand each of these areas, a specific DBMS will be evaluated in the light of these options and the program will be assigned points in each area (on a scale of up to ten). The program we will be reviewing is DB Master, Version Four. Also included in this section are reviews of three utility packages which are available for DB Master: Utility Pak #1, Utility Pak #2 and DB Master Stat Pak.

DB Master, Version Four
Language: Applesoft
Manufacturer: Stoneware, Inc
50 Belvedere St
San Raphael, CA 94901
Suggested retail: $350.00

DB Master, Version Four, is the newest Data Base Management System from Stoneware. It improves upon the features of DB Master, Version Three and previous versions. Any file or "utility pak" that worked with Version Three will also work with Version Four.

Besides slightly different display screens, Version Four has an improved editor, higher capacity, faster performance in most areas and a much better manual. Additionally, some of the features that were on Utility Pak #1 have been included in Version Four.

DOCUMENTATION: 9 points

The documentation of DB Master, Version Four is excellent. It is over 300 pages long and consists of a beginners' section, a tutorial for all the various options, a reference guide, an index, an appendix of terms and sample files.

For beginners, it takes you in sequential steps through both the documentation and menu options. An example is used throughout the tutorial so that you can key in the same information and follow the hands-on discussion. Previous users of DB Master can skip the beginners' section and move right up to the reference guide to look up specific program functions.

EASE OF USE: 8 points

This program is relatively easy to use. Contrary to dBase, DB Master is a menu-driven DBMS. Screen design for data entry is done entirely by menu.

There is little flexibility for free form design, such as title banners on the data screen. Instead, you select the field type, length and location on the actual screen. You can have more than one page of data entry. Multiple field types are supported, as well as flexibility of field names.

Form development is very good in that there is user flexibility for placing a field anywhere on the screen. DB Master does not support any type of 80-column card and, thus, the field layout must be in 40 columns. The program also restricts you to a sequential layout for the primary fields, plus a limit of 35 characters.

DATA ENTRY: 6 points

DB Master will now format two diskettes: a utility disk and a data storage disk. Both are required for any given data file. Because of this, two drives are really needed to utilize DB Master efficiently. But, even with two drives, you will have some disk swapping to do.

There is no type-ahead buffer like the one which exists in most word processors. Thus, if you have to enter one piece of data at the bottom of the screen, you have to wait for the program to catch up. You cannot enter data at maximum typing speed without losing characters.

Defaults on entry are adequately provided. These defaults are defined at the time of file creation but can be modified temporarily at any time upon entering the data entry mode. Computed fields can also be defined to eliminate data entry where calculations are to be performed.

DATA SEARCHING & RETRIEVAL: 8 points

DB Master gets top points for flexibility in search criteria.

The selection capability is excellent, allowing for the Wildcard, ORs, ANDs, Less Than, Greater Than, ascending and descending searches, or any combination thereof.

Using key fields, data retrieval is almost instantaneous. Using secondary keys it is still very fast. If you search for an item that is not a key, the search process is very slow. Each record on the disk is searched until the required match is found. For a large data base, this can take some time.

Once the data is retrieved, DB Master has a very sophisticated editor that will allow you to edit the data on the screen and, then, re-save it as the old file, or even save it as a new one. If a record is deleted, you can recover it up to the point where you re-blocked the files to remove all deleted files from the disk.

REPORTING CAPABILITY: 5 points

The program's reporting capability is not more than fair. This is the one significant weakness of DB Master, Version Four.

Report layout is very difficult to accomplish and very confusing. You must use worksheets to layout actual print output. This is very poor. One of the purposes of the computer is to eliminate hand calculations. Some of the simple commands in other DBMS programs allowing you to center text, left- or right-justify, etc. are nonexistent in DB Master.

Further, the various hierarchical print formats can often be confusing. There is very little flexibility for specifying output page options. For example, in designing label reporting outputs, only the standard size labels are supported.

Until you have worked with the tutorial several times and done some experimenting you will not be able to work effectively with the report options of DB Master.

FILE STRUCTURE: 5 points

Expandability is good. Several data diskettes for a given file are supported. In order to modify the file, however, you must use one of the optional utility paks. A special version of DB Master exists that allows you to use it with a hard disk. Although a special data storage technique is used, all data diskettes can be easily backed up with any copy program.

The biggest drawback to DB Master is the protection scheme. The disk is protected so much that it is extremely sensitive to drive speed and type. Stoneware only guarantees operation with Apple drives. Other drives will fail to boot the diskette or will result in system crashes during execution. The disk is so sensitive that it may frequently create I/O (Input/Output) errors and have to be replaced by Stoneware.

DB Master stores data in a special format that cannot be directly accessed with other programs. The only way to use the data is through a separately-purchased program called Utility Pak #1 and write the data out in a Data Interchange Format (DIF).

MISCELLANEOUS FEATURES: 6 points

DB Master supports mulitple drives in various slots. The only requirement is the boot disk in slot 6.

Single drive users will have a lot of disk swapping to do. If you purchase the option, a hard disk is supported. But the manual does not indicate that the program can be upgraded to support this at a later date.

Also, as indicated before, there is no 80-column support. This is unacceptable, particularly since a common standard allowing this in the Apple IIe does exist.

Lower case is not readily supported. If you can enter lower case via your keyboard, DB Master will accept it for data input but not for program responses.

Further, the search and sort routines are strictly ASCII (American Standard Code for Information Interchange) and so will not function correctly if a combination of upper and lower case is used.

Finally, error-trapping is very good. But there are some instances during file creation where screen error messages will be displayed that are incorrect. Also, partly due to the protection mechanism, the disk may occassionally dump to the monitor while loading various modules. This leaves the user with nothing to do but reboot.

SUMMARY

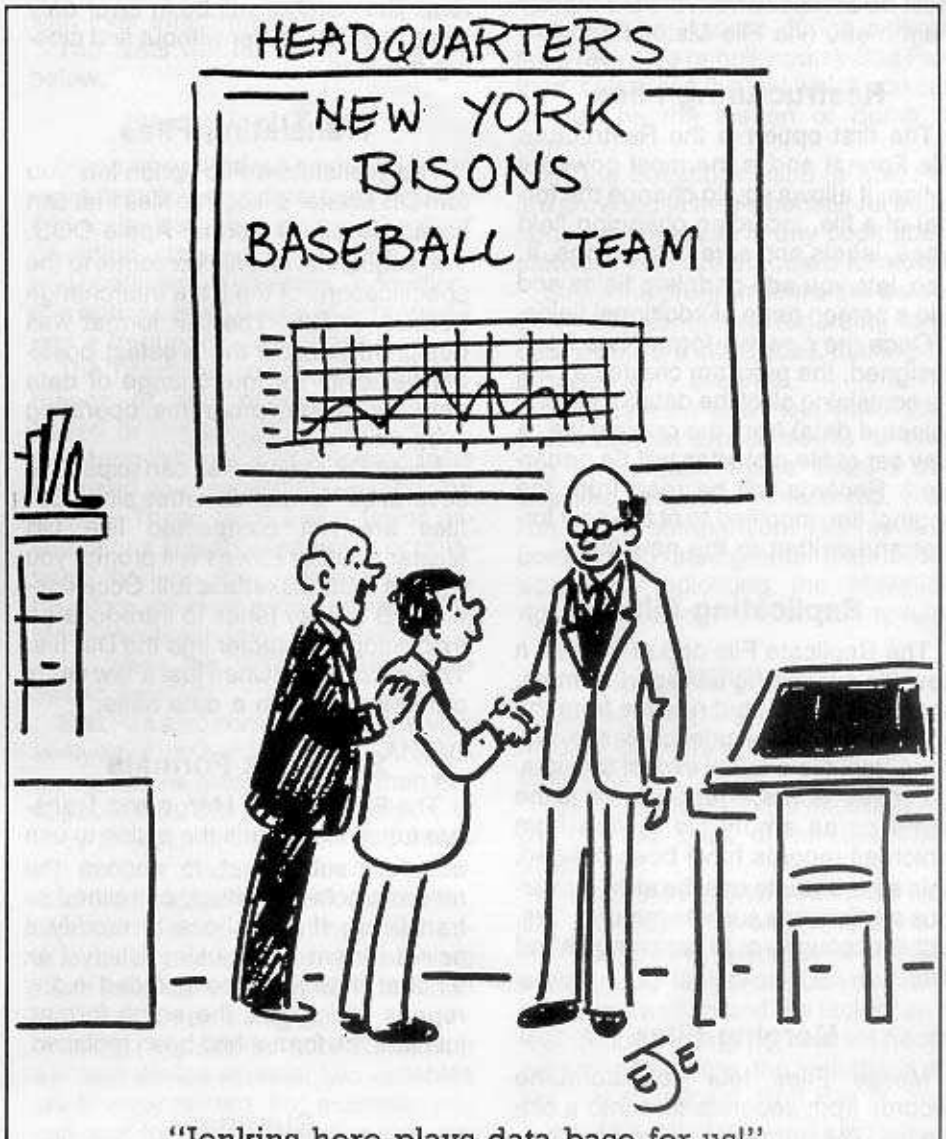
This results in a final score for DB Master of 47 points, or roughly 67%.

As you can see, the program has some very strong points, but the major weaknesses really detract from the program.

DB Master, Version Four is a powerful DBMS. It is not nearly as flexible as dBase, but, then, it does not require any programming knowledge. This makes it very attractive to a beginner or for a business only interested in establishing a data base.

The complexity of the output and reporting formats, combined with the inflexibility of design layout detract from the product. The failure of DB Master to use common hardware, such as 80-column cards, is the one major weakness of the program.

In short, DB Master, Version Four is



"Jenkins here plays data base for us!"

a very good data base program, but it still has room for improvement.

DB Master Utility Pak #1
Manufacturer: Stoneware, Inc.
Language: Applesoft, Machine
Hardware: 48K Apple, DB Master, Versions Three or Four, Two disk drives
Suggested retail: $129.00

DB Master Utility Pak #1 is the most useful of all DB Master "utility paks." Some of the options on this program include a Data Interchange Format (DIF) translator, the possibility of file restructure, replicate and merge and damaged-file recovery.

The DIF option lets you use standard text files and write your own special purpose programs to work with DB Master files or it lets you interchange data with many of the other programs which support DIF files, like Visicalc.

Utility Pak #1 cannot be booted but must be accessed from the DB Master main menu (via File Maintenance).

Restructuring Files

The first option is the Restructure File Format and is the most powerful option. It allows you to change the format of a file, including changing field types, labels and screen locations. It, also, lets you add or delete fields and add a screen page of additional fields.

Once the new file format has been designed, the program creates a new file containing all of the data (or just the selected data) from the original file. A new set of file diskettes will be generated. Records will be read from the original file, modified to fit the new format and written to the new file.

Replicating Files

The Replicate File option creates a new file containing all report formats, short forms, etc., but no data from the original file. Replicate copies everything from the original except the data.

In other words, a replicate file is the same as an empty file or one from which all records have been deleted. This allows you to use the shell for various applications such as monthly files, active accounts, or to recover crashed files.

Merging Files

Merge Files lets you combine records from separate files into a single file. The formats must be identical. Create or Change File defaults let you permanently change data entry defaults. The recover module is capable of recovering most records from files on diskettes which can no longer be read or accessed by DB Master.

An intact utility diskette is required for the file to be recovered. Unfortunately, there is no option to attempt to recover a bad "utility pak." This would have been an excellent option. As it is, only data diskettes can be recovered. Diskettes with I/O errors, damaged indexes, or multi-volume files where a data disk has been destroyed are all candidates.

Resetting Indicators

Option six allows you to reset the number of records in the file indicator. This option is included in DB Master, Version Four.

The relevant figure is displayed when checking file statistics and is only there for your convenience. The file will not be affected in any way if it is incorrect. The number will be in error only if you exit DB Master without first closing the files.

Translating Files

The Translate-a-File option lets you turn DB Master's files into files that can be accessed by normal Apple DOS. The sequential text file conforms to the specifications of the Data Interchange Format, or DIF. The DIF format was designed to allow the greatest possible flexibility for interchange of data between various programs, operating systems, hardware, etc.

Large DB Master files can expand to several DIF format diskettes since DIF files are not compacted like DB Master's. Utility Pak #1 will prompt you when a DOS diskette is full. Occasionally, DB Master tends to introduce an extraneous character into the DIF file. This occurs only when just a few fields are selected from a data base.

Selecting Formats

The Restructure, Merge and Translate functions include the option to use a select sub-format to choose the records to be modified, combined or translated. If you chose to modify a select format when using Utility Pak #1, that change will be included in any reports which use the same format, just as if the format had been replaced.

Summary

The biggest disadvantage to Utility Pak #1 is the inadequacy of the documentation. It fails to explain many of the screen prompts for user input, which can lead you into destroying a file.

For example, near the end of the Restructure Format option the documentation says to "simply follow the directions on your screen." Yet there are no explicit directions and you are presented with a menu of several options. If you select the wrong one, you can destroy all the work you had done under this option.

Overall, though, Utility Pak #1 is the most versatile and valuable program of its kind. Its applicability is almost universal among all DB Master users. If DB Master is re-evaluated using the same DBMS standards but including Utility Pak #1, then some of the problems associated with File Structure and the Miscellaneous Features (see accompanying review of DB Master) will disappear. DB Master would then rate 52 out of 70 points, or about 75%.

DB Master Utility Pak #2
Manufacturer: Stoneware, Inc.
Language: Applesoft, Machine
Hardware: 48K Apple, DB Master, Versions Three or Four, two disk drives
Suggested retail: $129.00

DB Master Utility Pak #2 provides additional capabilities for users of the DB Master program in its Versions Three or Four.

Some of these capabilities have been included in Version Four of the main program and, thus, Utility Pak #2 may not be needed by many new owners. Some of the features include a global editor, label printer and the possibilities of transaction file merge, reblock and file statistics.

To use Utility Pak #2 you have to access it from the file maintenance option in DB Master; you cannot boot the utility pak. As the documentation states, error-trapping in Utility Pak #2 is not that extensive so it is very important to first backup your data and utility diskettes before using the utilities.

The Features

The Global Editor lets you delete or replace all or selected records in a file. Up to five different edits in one pass can be made.

The Label Printer is a better label report option than that included in DB Master. You can print from one to five labels across. Blank lines and spaces

are eliminated and variable spacing is supported. Transaction file merge lets you edit data in one file based upon information stored in a different file.

Transaction File Merge only works with numeric and dollar-sign ($) fields. This option is only applicable to a limited number of specific user operations.

The Re-block option is used to create extra space and is included on the master disk of DB Master, Version Four but not on Version Three.

It is primarily used with large files or on one which already has more than one data diskette and is close to splitting onto another. It has limited applicability for hard disk users.

The Re-block option creates another data diskette that removes extra spaces from data storage to shrink the file to its optimum size. This leaves more room on the diskette for storage.

Change Protected Fields lets you change fields from Read Only status to normal and vice versa. This is for users that have established protected fields against other users of their data base.

Printing Data Entry Forms is included in Version Four. It lets you print data entry forms and print the DB Master short form.

Finally, File Statistics does the same thing as the File Maintenance option of DB Master.

Summary

Overall, Utility Pak #2 has some very good options with the Global Editor and Label Printer options.

Although these functions can be accomplished from DB Master in some form, they are done much more effectively using the utility program. If you do a lot of global editing or label printing, then Utility Pak #2 will streamline operations for you.

The other options are applicable to specific users only and, thus, may not be needed by most.

DB Master Stat Pak
Manufacturer: Stoneware. Inc.
Language: Applesoft, Machine
Hardware: 48K Apple, DB Master, Versions Three or Four, two disk drives
Suggested retail: $129.00

DB Master's Stat Pak extends the range of statistical information that can be calculated for data stored in DB Master files.

Stat Pak performs mean and standard deviation calculations and other powerful statistical tests, including linear regression, correlation and chi-square analysis. This program differs from the Statistics option of the DB Master Report Generator which generates mean and standard deviation figures, in that it calculates without printing a report.

The documentation is an excellent tutorial in the operation of Stat Pak as well as the interpretation of statistical test results and the use of the statistical tables. However, a basic understanding of probability and statistics is still required for any user of Stat Pak.

Like other DB Master "utility paks," Stat Pak cannot be booted and must be accessed via the File Maintenance option on the main DB Master program diskette.

Stat Pak is designed to help you determine when to use the various statistical tests, how to interpret the results of the tests, to employ procedures specific to running the tests and it, even, provides sample test results with their interpretations.

The tests will be briefly explained below.

Statistical Tests

Mean and standard deviation is the most common statistical test. This option calculates the mean, standard deviation, standard error of the mean and coefficient of variation. In addition, this options sorts data and arranges it into a frequency distribution.

The T-tests are used to compare the means of two populations, either paired or unpaired. Like the T-tests, the Mann-Whitney and Wilcoxon tests are used to compare two population means. These tests make fewer assumptions about the sampled populations than do the T-tests and can, therefore, be more widely applied. The Mann-Whitney test uses unpaired data, while the Wilcoxon test uses paired samples.

Stat Pak also computes the one-way analysis of variance (ANOVA). ANOVA compares the means of more than two populations. Chi-Square analysis is used to determine whether two different traits or characteristics in a population are independent. For example, chi-square analysis can help you decide whether hair color and sex are traits which are independent of each other in human beings.

Linear regression and correlation will help decide whether two variables are linearly related. For example, you can test for a linear relationship between calorie input and weight gain in people. Only first order analysis of the form $Y = mX + b$ is supported.

Linear regression tries to match a straight line to a set of data points, which can then be used for projection. Correlation determines the validity or accuracy of that line.

Numeric Data Only

To use data from a DB Master file, all data must be numeric. Up to 350 individual data items can be processed by DB Master. For large data files, this can result in numerous calculations. The user must then piece the different sets of calculations together.

This approach does not result in statistically correct answers and, so, unless a small sample is desired, you must determine your own selection criteria for your files in order not to exceed the limit of 350 data items.

After specifying the statistical test and data elements, you input the maximum number of decimal places to be reported in the output and the data values to be ignored during calculations (like zero or null inputs). Stat Pak, then, calculates the test which you can display on the screen or dump to printer.

All the equations used in Stat Pak are provided in the Appendix. All equations can be found in any book about statistics and are standard formulae.

Since the program limits the data to 350, no assumptions regarding large data inputs are introduced, making the program very accurate.

Results from each test were compared against similar results for tests run on a main frame system and checked by hand. A T-test was, then, run on the comparison. The resulting comparison rated greater than .999 in accuracy, indicating the statistical rigorousness of the Stat Pak results.

Summary

Overall, Stat Pak is an excellent module for users who are interested in statistics.

The documentation is excellent and by far the best of all the utility programs.

Most of the important statistical tests are provided. The only limitations of Stat Pak are the restriction to 350 data samples, to first order polynomial in linear regression and the lack of an F-test. Since the last two are not necessary to most users, the limitations are not that significant. Stat Pak is, thus, a valuable addition to DB Master.

CRUNCHLIST By Ray Darrah

Crunchlist is an ampersand (&) utility designed for maximum disk-space utilization when creating EXECable text files.

An EXECable text file is one in which part (or all) of a program has been saved as keystrokes instead of the usual way (as Applesoft tokens). Such a program is said to be "Captured." When done properly, EXECing the file (typing 'EXEC Filename') is just like keying the program in. Therefore, program lines are inserted between (or replace) the existing lines of program in memory.

DOS allows the commands necessary to create an EXECable file (i.e. OPEN and WRITE) only in deferred execution. Because of this, one must actually alter the program they wish to capture. Usually this alteration is the addition of a new first line number. Then, a RUN will accomplish the save. A typical line that accomplishes the capture of lines 1000 through 9999 would look something like this:

```
1 NM$ = "CAPTURED":D$ =  CHR$ (4)
    : PRINT D$"OPEN"NM$: PRINT D
    $"WRITE"NM$: POKE 33,33: LIST
    1000 - 9999: PRINT D$"CLOSE"
    : TEXT : END
```

Notice the LIST 1000-9999 statement. When the Applesoft interpreter comes across this, it does a regular listing. But, rather than each character of the listing being printed on the screen, they are sent to the disk and stored consecutively under the filename "CAPTURED". As you probably have already noticed, a lot of spaces are inserted into a regular listing to make it more readable. When you capture a program in this manner, all those spaces are saved on the disk.

Since these spaces aren't required when typing line numbers back in (even via EXEC), why have them laying around on your disk just taking up space?

Crunchlist eliminates this diskette waste by listing specified line numbers **without unnecessary spaces**! In addition to the disk storage gained because of this, the captured programs take less time to save and less time to EXEC back in.

Using Crunchlist

First, type in the hex dump on page 27 and

BSAVE CRUNCHLIST,A$300,L$98

Whenever you wish to install Crunchlist, simply type

BRUN CRUNCHLIST

With Crunchlist installed, one uses the ampersand instead of the command word "LIST" on those lines he wishes to Crunchlisted (i.e. captured). Therefore, using Crunchlist, the above line number which does the capturing would look like this:

```
1 NM$ = "CAPTURED":D$ =  CHR$ (4)
    : PRINT D$"OPEN"NM$: PRINT D
    $"WRITE"NM$: & 1000 - 9999: PRINT
    D$"CLOSE": END
```

In addtion to the word "LIST" being replaced by the ampersand, the "POKE 33,33" and "TEXT" statements have been removed. This is because Crunchlist, as opposed to Applesoft, doesn't indent line numbers (which is just more empty space). Try Crunchlisting some long lines in immediate execution for a good example.

Now your ready to start capturing EXECable files and conserve disk space at the same time. This is great for saving lines that appear frequently in your creations (like a title page or REMarks stating the illustrious author). I'm sure you can think of many other uses for Crunchlist.

Crunchlist Source Code

```
1000 * ***************************************
1010 *                                       *
1020 *              CRUNCH LIST              *
1030 *                                       *
1040 *                                       *
1050 *            BY RAY DARRAH              *
1060 *                                       *
1070 * ***************************************
1080
1090
1100
1110
1120
1130 *                                       *
1140 *     APPLESOFT ROUTINES/LOCATIONS      *
1150 *                                       *
1160
1170 LINGET      .EQ  $DA0C        SETS "LINNU
M" TO NUMBER TYPED
1180 FNDLIN      .EQ  $D61A        MAKES "BASIND
EX" POINT TO THE LINE NUMBER CONTAINED IN "LIN
NUM"
1190 LINPRT      .EQ  $ED24        ROUTINE THAT
PRINTS X,A IN DECIMAL
1200 TOKENTABLE  .EQ  $D000        TABLE OF TOKE
NS SPELLED OUT
1210 GETCHR      .EQ  $D72C        ROUTINE THAT
GETS NEXT CHARACTER OF COMMAND WORD
1220
1230 *                                       *
1240 *          MONITOR ROUTINES             *
1250 *                                       *
1260
1270 COUT        .EQ  $FDED        PRINT A AS AS
CII
1280 CROUT       .EQ  $FD8E        PRINT A CARRI
AGE RETURN
1290
1300 *                                       *
1310 *          ZERO PAGE LOCATONS           *
1320 *                                       *
1330
1340 CHRGOT      .EQ  $B7          RETRIEVE THE
LAST CHARACTER FROM BASIC
1350 CHRGET      .EQ  $B1          GET A NEW BAS
IC CHARACTER
1360 LINNUM      .EQ  $50          NUMBER OF LAS
T LINUMBER TO LIST
1370 BASINDEX    .EQ  $9B          POINTER TO CU
RRENT LINE THAT WE ARE LISTING
1380 TEMP        .EQ  $85          TEMPORARY ST
ORAGE OF THE OFFSET FOR THIS LINE
1390 BYTENLINE   .EQ  $9D          USED BY GETCH
R SHOULD POINT TO WHICH TOKEN YOUR GETTI
NG
1400
1410 *                                       *
1420 *          PAGE THREE LOCATIONS         *
1430 *                                       *
1440
1450 AMPER       .EQ  $3F5         AMPERSAND
JMP LOCATION
1460
1470 *                                       *
1480 *          START OF PROGRAM             *
1490 *                                       *
1500
1510             .OR  $300         SQUEEZE IT IN
TO PAGE THREE
1520             .TF  CRUNCHLIST
1530             LDA #$4C          MAKE AMPERSA
ND JMP TO START
1540             STA AMPER
1550             LDA #START
1560             STA AMPER+1
1570             LDA /START
1580             STA AMPER+2
1590             RTS               HOOKUP COMPL
ETE
1600 START       JSR LINGET        SET LINNUM TO
 START OF RANGE
1610             JSR FNDLIN        POINT BASINDE
X TO 1ST LINE
1620             JSR CHRGOT        RANGE SPECIFI
ED?
1630             BEQ MAINLIST      YES, LIST IT
1640             JSR CHRGET        SKIP THE MINU
S
1650             JSR LINGET        SET LINNUM TO
 END RANGE
1660 MAINLIST    LDA LINNUM        IS THE END NU
MBER A NULL?
1670             ORA LINNUM+1
1680             BNE NEXTLIST      IF NO, START
LISTING
1690             LDA #$FF          YES IT IS, SO
 SET THE END NUMBER TO 65535
1700             STA LINNUM
1710             STA LINNUM+1
1720 NEXTLIST    LDY #1            LIST AN ENTIR
E LINE
1730             LDA (BASINDEX),Y  IS HIGH B
YTE OF LINK ZERO
```

```
          BEQ FINISHED       YES, DONE LISTING
          INY                GET LINE NUMBER
          LDA (BASINDEX),Y
          TAX
          INY
          LDA (BASINDEX),Y
          CMP LINNUM+1
          BNE .1
          CPX LINNUM
          BEQ STRTLISTN
.1        BCS FINISHED
STRTLISTN STY TEMP
          JSR LINPRT
LISTLOOP  LDY TEMP
          JMP PRINT1
SENDCHR   ORA #$80
          JSR COUT
PRINT1    INY
          LDA (BASINDEX),Y
          BNE TOKENCMP       NOT END OF LINE SO SEE IF TOKEN
          JSR CROUT          EVERY LINE IS TERMINATED WITH A CARRIAGE R
ETURN
          LDY #0             AT END OF LINE TO GET NEXT LINK
          LDA (BASINDEX),Y
          TAX
          INY
          LDA (BASINDEX),Y
          STX BASINDEX       POINT TO NEXT LINE
          STA BASINDEX+1
          BNE NEXTLIST       BRANCH IF NOT AT END OF PROGRAM
FINISHED  RTS                COMPLETELY DONE LISTING SO RETURN
TOKENCMP  BPL SENDCHR        IF NOT A TOKEN, JUST PRINT IT
          SEC
          SBC #$7F           MAKE IT A INDEX NUMBER FOR THE TABLE
          TAX
          STY TEMP           SAVE LINE POIINTER
          LDY #TOKENTABLE-$100
          STY BYTENLINE      POINT BYTENLINE TO TABLE
          LDY /TOKENTABLE-$100
          STY BYTENLINE+1
          LDY #$FF
SKIPTOKN  DEX                COUNT TOKENS VERSA X
          BEQ PRNTOKN
TOKLP     JSR GETCHR
          BPL TOKLP
          BMI SKIPTOKN
PRNTOKN   JSR GETCHR         GET A CHARACTER OF THE TOKEN
          BMI TOKDONE
          ORA #$80
          JSR COUT
          JMP PRNTOKN
TOKDONE   JSR COUT           SEND LAST CHARACTER OF TOKEN
          JMP LISTLOOP       ;GO DO NEXT THING IN LINE
```

Crunchlist Hex Dump

```
0300: A9 4C 8D F5 03 A9 10 8D $2C5A
0308: F6 03 A9 03 8D F7 03 60 $BB3A
0310: 20 0C DA 20 1A D6 20 B7 $5082
0318: 00 F0 06 20 B1 00 20 0C $6F0F
0320: DA A5 50 05 51 D0 06 A9 $6C42
0328: FF 85 50 85 51 A0 01 B1 $78CD
0330: 9B F0 36 C8 B1 9B AA C8 $12E6
0338: B1 9B C5 51 D0 04 E4 50 $2CBF
0340: F0 02 B0 25 84 85 20 24 $B676
0348: ED A4 85 4C 53 03 09 80 $8239

0350: 20 ED FD C8 B1 9B D0 12 $90A7
0358: 20 8E FD A0 00 B1 9B AA $082D
0360: C8 B1 9B 86 9B 85 9C D0 $76C6
0368: C4 60 10 E2 E9 7F AA 84 $8A2F
0370: 85 A0 D0 84 9D A0 CF 84 $074C
0378: 9E A0 FF CA F0 07 20 2C $B68D
0380: D7 10 FB 30 F6 20 2C D7 $8C3C
0388: 30 08 09 80 20 ED FD 4C $1F92
0390: 85 03 20 ED FD 4C 49 03 $5E75
```


CORE CORE CORE CORE CORE CORE CORE CORE CORE CORE CORE CORE CORE CORE CORE

Hardcore continued from page 16

the connections to be certain that pins 18 and 21 are cut off.

MAKE SURE THE APPLE POWER IS OFF BEFORE CONTINUING.

Now, carefully remove the ROM labelled F8 (it is the socket farthest to the left that has 24 pins, as you face the keyboard) and plug this jumper socket into the motherboard. (You could plug this socket into your integer card in the same fashion). Now, plug the modified EPROM into this jumper socket and you will all be done!

Go ahead and turn on your Apple and, if you had followed these instructions correctly, you will see the text page filled with "garbage." At this time, press "RETURN" to get things going as usual.

How to use the "Super Saver" F8 ROM

From now on, whenever you press "RESET," the computer will just freeze (no beep or anything). Then, you must press one of three keys, depending on what you want the computer to do.

First of all, by pressing "RETURN" the computer will just do the usual kind of reset (i.e. JuMP to BASIC). Secondly, by typing "-" the computer will act as if you have the "old-style" F8-ROM and JuMP into the monitor without any memory saves. Finally, a ":" will engage the "Super Saver," thus, moving the volatile memory (locations $0000 through $9000) into locations $2000 through $2900, with the stack pointer saved at $2901 and it will put you in the monitor.

Uses for your Modified ROM

Remember our original problem with DISK ORGANIZER II in the article "Hidden Locations Revealed?" Well, now we can boot the program and, at the desired time, hit "RESET," type ":" and save all of memory from $00-$900 at $200-$2900, as well as the stack pointer at $2901!

All that is left to do, now, is to find the starting location and what memory is used and the hard part of your soft-key is done!

Recapitulation

1) Acquire a blank 2716 and access to an EPROM programmer.

2) Boot a disk with normal DOS and enter the monitor

CALL-151

3) Move the memory from ROM to RAM

2800<F800.FFFFM

4) Type in the hexdump on page 28.

5) Alter the RESET and the NMI vectors

2FFA: CD FE CD FE

6) Save the modifications

BSAVE MODIFIED F8ROM, A$2800,L$800

6) Burn the blank 2716 with this saved code.

7) Using a low-profile, 24-pin socket, solder pin 12 to pin 18, then solder pin 21 to pin 24.

8) Cut off pins 18 and 21 as close to the socket body as possible.

9) Plug the jumper socket into the F8 socket on the motherboard or integer card.

10) Plug the modified 2716 into the jumper socket and you are done!

Source Code

```
1000 * ------------------------------
1010 *          SUPER SAVER
1020 *        MODIFIED F8-ROM
1030 *        BY ERNIE YOUNG
1040 *   ENHANCED BY RAY DARRAH
1050 * ------------------------------
1060
1070                .OR  $FECD
1080                .TA  $2ECD
1090
1100 * ------------------------------
1110 *          EQUATES
1120 * ------------------------------
1130
1140 KEY.RESET      .EQ  $C010
1150 KEYBOARD       .EQ  $C000
1160 A1L            .EQ  $3C        ;LSB ADDRESS
1
1170 A1H            .EQ  $3D        ;MSB ADDRESS
1
1180 A2L            .EQ  $3E        ;LSB OF ADDRE
SS 2
1190 A2H            .EQ  $3F        ;THE MSB OF A
DDRESS 2
1200 A4L            .EQ  $42        ;LSB ADDRESS
4
1210 A4H            .EQ  $43        ;THE MSB OF A
DDRESS 4
1220 MONITOR        .EQ  $FF59      ;ENTRY POINT
TO THE MONITOR
1230 RESET          .EQ  $FA62      ;ENTRY TO NEW
 F8 RESET ROUTINE
1240 MON.MOVE       .EQ  $FE2C      ;MONITOR "M"
ROUTINE
1250 COUT1          .EQ  $FDF0      ;ROUTINE TO P
RINT ASCII
1260 HOME           .EQ  $FC58      ;ROUTINE TO C
LEAR TEXT SCREEN
1270 TEXT           .EQ  $FB2F      ;ROUTINE THAT
 SELECTS THE TEXT SCREEN
1280
1290 * ------------------------------
1300 * EVALUATE WHAT TYPE OF RESET
1310 * IS DESIRED
1320 * ------------------------------
1330
1340 WAIT           BIT  KEYBOARD   ;WAIT UNTIL A
KEY IS PRESSED
1350                BPL  WAIT
1360                STA  KEY.RESET
1370                LDA  KEYBOARD   ;GET THE KEY
1380                CMP  #'-        ;IS IT AN "OL
D" F8 RESET
1390                BEQ  MONITOR    ;YES, JUMP IN
TO THE MONITOR
1400                BCS  NEW.STUFF  ;IF A COLON,
 THEN DO THE NEW RESET
1410                JMP  RESET      ;DO A REGULAR
 RESET
1420
1430 * ------------------------------
1440 *   SAVE THE STACK POINTER
1450 * ------------------------------
1460
1470 NEW.STUFF      TSX             ;GET THE VALU
E OF THE STACK POINTER
1480                STX  $2901      ;AND SAVE IT
AT $2901
1490
1500 * ------------------------------
1510 * MOVE PAGES $00 AND $01
1520 *    INTO PAGES $20 AND $21
1530 * ------------------------------
1540
1550                LDY  #0         ;START WITH B
YTE ZERO
1560 SAVE1          LDA  0,Y        ;GET A ZERO P
AGE BYTE
1570                STA  $2000,Y    ;SAVE IT
1580                LDA  $100,Y     ;GET A STACK
BYTE
1590                STA  $2100,Y    ;AND SAVE IT
1600                JMP  AROUND.CODE;JUMP AROU
ND THE MONITOR STUFF
1610                .OR  $FEFD      ;NEW ORIGIN F
OR THE CODE
1620                .TA  $2EFD
1630 AROUND.CODE    INY             ;GET OFFSET F
OR NEXT BYTE
1640                BNE  SAVE1      ;IF NOT DONE,
 MOVE ANOTHER BYTE
1650
1660 * ------------------------------
1670 * SET UP FOR MONITOR MOVE
1680 * ------------------------------
1690
1700                STY  A1L        ;MSB OF FROM
ADDRESS IS ZERO
1710                STY  A4L        ;THE MSB OF T
HE TO ADDRESS IS ALSO ZERO
1720                STY  A2L        ;MOVE CODE TH
ROUGH $900
1730                LDA  #9
1740                STA  A2H
1750                LDA  #2         ;MOVE CODE ST
ARTING AT $200
1760                STA  A1H
1770                LDA  #$22       ;MOVE CODE IN
TO PAGE $22
1780                STA  A4H
1790
1800 * ------------------------------
1810 *         EXIT
1820 * ------------------------------
1830
1840                JSR  MON.MOVE
1850                JSR  TEXT       ;SET THE TEXT
 SCREEN
1860                JSR  HOME       ;CLEAR SCREEN
1870                JMP  MONITOR
```

Hexdump

```
2ECD:  2C 00 C0                        $0E8B
2ED0:  10 FB 8D 10 C0 AD 00 C0         $B6A9
2ED8:  C9 2D F0 7D B0 03 4C 62         $1B98
2EE0:  FA BA 8E 01 29 A0 00 B9         $D15A
2EE8:  00 00 99 00 20 B9 00 01         $9B59
2EF0:  99 00 21 4C FD FE 20 00         $1FE5
2EF8:  FE 68 68 D0 6C C8 D0 E7         $C5C6
2F00:  84 3C 84 42 84 3E A9 09         $2A77
2F08:  85 3F A9 02 85 3D A9 22         $9E7A
2F10:  85 43 20 2C FE 20 2F FB         $F0AC
2F18:  20 58 FC 4C 59 FF 60            $66E7
```

WHIZ KID

BY RAY DARRAH

Editor's Note: Recently, Hardcore COMPUTIST received many requests for a simple, easy to follow guide that will take a beginning computer's user step-by-step through copy protection and deprotection.

Well folks, here it is!

Starting with this issue, Hardcore COMPUTIST will be featuring a column by staff programmer Ray Darrah, entitled "Whiz Kid." Its goal is to make computer whiz kids out of Hardcore COMPUTIST readers. It will mainly focus on copy protection and deprotection but may stray from the subject sometimes.

Before going any further, however, Whiz Kid assumes that you have read the Applesoft manual and possess a working knowledge of BASIC (on the Apple). Also, a general understanding of RAM (Random-Access Memory), ROM (Read-Only Memory) and the Apple DOS 3.3 disk operating system would help.

What is Copy Protection?

The answer is really quite simple. Copy Protection is a non-standard method of preserving data, programs or otherwise. This means that data stored on a protected disk you bought "looks" completely different from data stored on the disk you INITialized yesterday.

In later issues we'll discuss disk-based protection in depth. But for now, here are a few tricks you can do that protect programs, but not from being copied.

The RUN Flag

A little known feature exists in the Applesoft ROM which allows the programmer to "tell" the ROM (via POKE) that a specific program is not to be listed.

The memory location which holds the information of whether or not to list this particular program is at 214 (or $D6 in hexadecimal). If locations hold a number higher than 127, Applesoft will not evaluate any typed commands (i.e. commands issued for immediate execution). Instead, Applesoft will RUN the program as soon as any line at all is typed.

To try this out:

1) LOAD in your favorite program and add line zero

```
0  POKE 214,128
```

2) Now run and then stop the program

```
RUN
CTRL-C
```

3) Try typing something other than RUN

```
LIST
```

Notice how the program RUNs no matter what you type. This would be very useful in keeping unwanted eyes (and every other bodily part) out of your program.

Unfortunately, this protection scheme has two flaws. First of all, because the ROM has to encounter the POKE statement before this protection becomes active, simply LOADing the program will override it. The second is that although BASIC commands don't work, DOS (Disk Operating System) commands do (e.g. CATALOG and FP), so the program can be easily copied using normal DOS commands.

continued on page 32

The Essential Data Duplicator:

REVIEW

The Essential Data Duplicator
Manufacturer: Utilico Software
Language: Machine
Hardware: Apple II, with 48K and at least one disk drive
$79.95

A few weeks ago, I received a much talked about nibble copier called "The Essential Data Duplicator" from Utilico Software. I simply couldn't wait to try it out.

While booting, I heard the strangest disk arm noises I've ever encountered. Ah!, I thought, sophisticated disk arm movement routines. "It looks like COPY II Plus has met its match," I said to myself.

Putting that thought aside when the title page came up, I banged a couple of keys to get to the main menu. I then noticed the addition of quarter tracks to the standard read/write error table at the top of the screen. "That's what I heard when it booted," I thought.

Even though the addition of quarter-tracks seemed superfluous, because they can be easily read or written on from an adjacent half-track, I took off my drive cover to see if it actually positions the read head between half-tracks. When I saw that EDD (The Essential Data Duplicator) accomplished this, I was reminded of a pet rock; a novel idea, but really not very useful.

Before going any further, I thought I'd read the manual.

Documentation

I found myself lost a few times while reading the 43 page, non-typeset manual that accompanies EDD. It desperately needs a glossary to define the terminology it uses extensively throughout the pamphlet. In addition, procedures aren't explained clearly or adequately enough to be even understandable. In short, there is a lot of room for improvement in the EDD manual.

But, from this manual, I gathered that the Essential Data Duplicator is one of the "New Breed" of nibble copiers which do a lot more than the old copiers do during the default copying procedure.

For this reason, EDD is supposed to be able to copy more programs without changing parameters than the other bit-copiers. In practice, however, this seems untrue.

While using EDD, I found that the number of programs I could copy reliably was substantially less than claimed. Furthermore, because of the extra steps EDD goes through, the backup procedure takes a very long time (in comparison with the other bit-copier programs). This applies even to copying disks with simple protection.

While scrutinizing the generally inadequate explanation of EDD parameters presented, I soon noticed that some of the parameters seemed useless and the most important ones were not explained well enough for application.

As an example, parameter $14 controls whether the hi-res screen is shown during copying or not. This produces some neat hi-res effects, but can't actually be used for presenting data in an understandable format.

When playing your favorite arcade game, can you tell what the dots and lines actually are in memory? It's a good thing that you don't have to change parameters to copy programs with EDD, because you can't tell which ones to change anyway.

Performance

The makers of EDD say that it can copy timing or "sync", bits more accurately than other copiers. However, they don't say exactly what EDD does to determine a timing bit from a data bit.

I believe it is impossible to do this, but EDD actually chops off (or inserts more of) these, hopefully valid, timing bits during the normal copying mode.

What if the omitted bits aren't really "sync" bits? There goes the chance of a working backup!

To further complications, EDD is heavily copy-protected and has, supposedly, been programmed to make unreliable copies when it tries to backup itself. It definitely is a difficult program to copy. So it seems that the assurance of a backup doesn't apply to EDD. My guess is that Utilico Software believes other programs are more worthy of backing up than their own.

Interesting Stuff

All this aside, EDD has some very unique features.

For instance, it has two ways of keeping nibble count. One way (as mentioned) is by inserting and deleting sync bits. Using the other, EDD will keep writing the track and verifying it until the track has the same number of nibbles as the original.

While this is occuring, the user hurriedly adjusts his drive speed in an attempt to get the displayed difference down to zero. My first thought was that this would be an excellent way to make a near-perfect copy. However, I realized that if the drive speed changes while EDD is writing a track, the bits on that track will not be evenly spaced. This would result in a copy of poor quality.

Because EDD does not wait until you are ready for it to write the track, this feature is difficult (if not impossible) to operate as it was intended.

Also included is a feature that will allow you to adjust your drive speed to that of the drive on which the disk you are trying to copy was written. Using this feature before you start to copy a program should give you a better chance of a good copy, even on tough protection schemes.

Unfortunately, what exactly is being displayed during this option is not explained clearly. But, even after figuring out what I was looking at, I had a VERY difficult time using this feature. It didn't help me to copy programs more reliably, anyway.

Because the number of disks EDD failed to copy accurately for me was substantial, I put it through a little test. Using the normal copying mode, I copied five different (normal DOS 3.3) diskettes. I was amazed when EDD created unreadable sectors on two of them. Apparently, to EDD, normal DOS 3.3 is a very sophisticated protection technique.

In my opinion, EDD is a copying package that has some strong points, but some features need to be worked out better. I think it best to wait for Eddie's father (a more advanced, better documented version of EDD) which will, hopefully, be available in the not-too-distant future.

Editors note: As this issue of Hardcore Computist was going to press, we received an updated version of The Essential Data Duplicator. We plan to review this latest version of EDD in an future issue of Hardcore Computist.

Reviewed by Dave Thompson

The CIA

Golden Delicious Software
7 Sloan Avenue
London SW3 3JD, England
$65.00

Anyone who owns and uses a bit-copy program like Copy II + or Locksmith knows that without a set of parameters the bit copiers will usually not produce a bootable backup disk. The question is, how does one go about finding correct settings for the parameters without spending countless hours experimenting with different combinations of the more than fifty individual parameters?

The CIA: Review

"The CIA.....is a set of five disk utilities which allows the user to investigate the manner in which almost any copy-protected disk is put together.....Although there are other programs on the market.....never before have these functions been combined into one package and sold with a quality piece of documentation."

The CIA (Confidential Information Advisors) from Golden Delicious Software LTD is a set of five disk utilities which allow the user to investigate the manner in which almost any copy-protected disk is put together. These utilities are a great aid in the determination of parameter settings for copy protected disks. The CIA package also includes "The CIA Files", a manual of more than 60,000 words which serves as a reference manual for the experienced user and a tutorial for those new to the world of the Apple's disk format.

The five disk utility modules include: "Tricky Dick" (a sector editor), "The Linguist" (track nibble dump utility), "The Code Breaker" (translate table editor) , "The Tracer" (disk search utility) and "The Tracker" (displays a list of tracks and sectors being accessed on the disk). Although there are other programs on the market which can accomplish most of these tasks, never before have these functions been combined into one package and sold with a quality piece of documentation.

The CIA's sector editing module, Tricky Dick, is termed in the documentation as the "Chief Executive Officer" because of the way it works in conjunction with the other CIA ulitities. Tricky Dick is a stand-alone module, but the services of all the other modules, except "The Tracker", have to be invoked through Tricky Dick. More on this interaction in a minute.

When you are using Tricky Dick you can change the address and data header and trailer bytes to anything you want, including wild cards, so that disks with altered marks can be read. You also have the ability to defeat the use of data and address field checksums when reading or writing sectors.

Once a sector has been read by Tricky Dick you can list it in an Applesoft or Integer BASIC or machine language disassembly format. This ability comes in handy when performing such techniques as boot code tracing.

Of course, if you have a protected disk with data and address marks altered to unknown values Tricky Dick, like any other sector editor, will generate the familar "I/O ERROR" when trying to read such a sector. To make life easier for those who want to view the data on such protected disks, "Tricky Dick" can work with The CIA module which is capable of directly dumping the raw (untranslated) nibbles from any specified track. This module is called "The Linguist".

The Linguist cannot function by itself, but must be accessed through Tricky Dick. Normally, to use The Linguist, one would load both Tricky Dick and The Linguist into memory by choosing option 2 from the main disk menu. Once The Linguist has been loaded into memory, it can be invoked while in Tricky Dick by typing a CTRL E. This calls upon the services of The Linguist which, first of all, recalibrates the disk drive read/write head to track zero. You can then select, in increments of one half track, the track which you wish to read. A CTRLR will read in the entire contents of the selected track and display the contents in their raw, untranslated format. This allows you to scan the track and see exactly what values are being used for the address and data marks, sync bytes, etc. Once you have figured out what address and data marks are being used, Tricky Dick can be reentered by typing CTRLC. The appropriate values for the address and data marks can then be inserted into Tricky Dick and, hopefully, the sector read operation will proceed smoothly.

The Linguist also has the ability to pass one sector of information back to Tricky Dick for editing purposes. The Linguist is able to use any of the 3 encoding techniques (6 & 2, 5 & 3 or 4 & 4) to translate one sector of data and pass it to Tricky Dick. This is done in The Linguist by choosing the desired encoding scheme, placing the cursor on the first byte following a data sector header, and then typing CTRLT. Control is then passed back to Tricky Dick with the translated sector data being displayed on the screen. This data can then be viewed, edited or listed in its appropriate language.

The next CIA module, The Tracer, adds some of the capabilities of Omega's Inspector to The CIA. The Tracer, which again works in conjunction with Tricky Dick, allows you to verify a disk's formatting and to search it for a VTOC (Volume Table of Contents), Catalog, Track/Sector lists and up to six different strings (with "wildcards") in ASCII (high or low format) or hexadecimal. This verification and/or search can proceed over any range of tracks and sectors which you specify. When one of these items is found on the disk, the user is placed back into Tricky Dick with the cursor placed over the first byte of the desired information.

The Tracer's scan rate is very rapid; for instance to search an entire DOS 3.3 disk for six nonexistent strings takes72 seconds, or just slightly over 2 seconds per track.

With the facilities of The Tracer, it becomes an easy matter to find track/sector lists of deleted files, specific assembly lanuguage instructions, program menus, references to the keyboard or any other type of information you may want to sniff out. Because The Tracer works in conjunction with Tricky Dick, the disk scan can be performed on protected disks if their data and address marks can be found which, of course, you can do by using The Linguist.

My only criticism of The Tracer is that it is rather difficult to correct typing mistakes when entering the search strings or the track and sector search ranges. If you make a mistake when entering any of these items, you cannot back space over or delete these mistaken characters; all you can do is enter CTRL SHIFT@ and type the whole string or search range all over again. This is not exactly state-of-the-art "user friendliness."

Like the Linguist and the Tracer, the next module, The Code Breaker, also must be used in conjunction with "Tricky Dick". The Code Breaker allows you to change the RWTS translate tables to protect your own disks from being read or copied by others. These translate tables are needed to transform encoded or "prenibblized" bytes into values which can actually be stored on the disk (high bit set and no more than 1 pair of consecutive 0's in DOS 3.3). If the values in the table are altered and a disk is written using the altered table, then that disk is unreadable to normal DOS because the bytes

on the disk can not be properly translated back to their original values . This may all sound a bit bewildering unless you have read "Beneath Apple DOS" or The CIA manual, but basically, The Linguist allows you to alter the tables and then have Tricky Dick read or write using the new translate tables. The Code Breaker is, therefore, useful for both protecting your own disks and reading from protected disks written with altered translate tables. Reading from a disk written with an altered translate table is actually a bit tricky since you have to know how the translate table has been altered. The manual, however, does give examples of how to use The Tracer to find translate tables which have been relocated and/or altered from that of normal DOS.

The last module of The CIA, The Tracker, is the only one which does not work in conjunction with Tricky Dick. The idea behind The Tracker, at first glance, seems to be very good; the ability to display which tracks and sector on a disk are being accessed. In practice, however, The Tracker can only be used on disks with a more-or-less normal DOS 3.3. This precludes using The Tracker for such things as finding where a nibble count is stored on a protected disk. The main use of The Tracker would seem to be to demonstate how DOS allocates files and, possibly, to locate bad sectors on a damaged disk.

Whenever The Tracker is used in conjunction with a program it displays a list of all tracks and sectors accessed by the program at the top of the screen. If there is a lot of disk access being done the screen can get pretty crowded and confusing, especially if the program is also displaying information on the text screen.

The "CIA Files", a 127 page manual accompanying The CIA is a very good piece of documentation. Each of the five utilities has a chapter devoted to it which contains a tutorial for newcomers on how to get the best use out of The CIA. These tutorials contain information not only on how to use The CIA, but also general information the VTOC, sector and data headers, sync bytes, how to get extra sectors on a disk and several different protection schemes.

There are areas in which The CIA Files could be improved upon, however. My first gripe is the lack of an alphabetical index. The manual does have an extended table of contents, but I feel that no documentation of more than ten pages should lack an index in the back.

My other criticism of the documentation has to do with its discussion of software protection techniques. Although several different protection schemes are discussed, there are never any specific examples of what software products are using which protection schemes. To me, it appears that this omission was intentional. For whatever reason this information was not included, I think some specific examples of protection schemes would be helpful in learning how to best utilize The CIA package.

The only other disk utility package I have seen which compares with The CIA is "Bag of Tricks" by the authors of Beneath Apple DOS. While each of these packages has its strong points , The CIA utilities are not copy protected as are the Bag of Tricks programs. The folks at Golden Delicious should be commended for the decision not to copy protect their software.

Although The CIA programs themselves are not protected The CIA disk does come with a protected track and a half track so owners can hone their disk snooping skills. The tracks are not transferred by FID, so it is necessary to either practice on the disk original or to use The Linguist and Tricky Dick to transfer these tracks to a backup disk.

The manufacturer's suggested price of The CIA is $65.00 which, in my opinion, is rather reasonable given the quality of the software and documentation. Delivery of The CIA, when ordered directly from Golden Delicious, does take a minimum of six weeks. So be patient. The CIA is worth waiting for.

Reviewed by Gary Peterson

WHIZ KID *continued from page 29*

DOS Listing

DOS intercepts and evaluates every character printed. This includes program listings as they roll up your screen. Whenever a CTRL-M CTRL-D string is printed, DOS examines subsequent bytes up to the next RETURN character (CTRL-M) and tries to execute them (e.g. VERIFY HELLO).

If you could make your program contain the correct sequence of characters (i.e. CTRL-M CTRL-D FP), you could have the DOS obliterate the program whenever it is listed. For that matter, you could instruct DOS to do any of a number of things, like maybe DELETE the file from the disk.

To try this out:

1) Clear the program in memory and set DOS pointers to correct locations

FP

2) Get your favorite program from disk (or type one in).

3) Add line zero (it is important not to type a space after the REM)

0 REMMDFP

4) Change the first two characters after the REM statement to control characters

POKE2054,13:POKE2055,4

I suggest that you SAVE this file before LISTing it.

Now, as soon as you LIST it, the program will vanish. Try changing the FP in line 0 to CATALOG or some other disk command. DOS will attempt to execute any command placed here every time you LIST the program.

You can even make the actual number of the program line disappear by adding ASCII (American Standard Code for Information Interchange) code 08 (CTRL-H's). These move the cursor backward.

After the appropriate number of CTRL-H's (i.e., when the cursor reaches the left edge of the text line) use spaces to erase any words or numbers preceding the DOS command.

That's about all for this issue. Tune in next time for more of Whiz Kid.

Advertiser's Index

SURE, YOU HAVE PROBLEMS. BUT BEFORE YOU BUY THE ANSWER, YOU HAVE TO KNOW IT.

You can save hundreds of dollars on commercial software by first using Public Domain Software

Take a look around you.

The software industry is growing and changing daily.

High-powered marketing campaigns are underway to promote a vast array of software programs that may or may not be what YOU need.

Business owners, teachers, artists, entrepreneurs, and other computer users all have one thing in common: no one REALLY knows what will work for them.

What about you?

Do you know what you want in a business software program? Are you sure that graphics program you saw last week will produce what you need? And what about all those $30 games on the market?

Before you buy any commercial software, evaluate your situation. After all, there's a difference between what looks good and what you need.

When you try Public Domain Software (PDS), you develop a feel for the things you want to have in a fine-quality commercial software program. Running a PDS letter writer program or phone list gives you an immediate workable file while allowing you to note features or limitations that are absolutely necessary for your future needs.

There's a difference between what looks good and what you need.

In a $1 billion worldwide software industry, there is bound to be a lot of confusion. Similar software packages are competing fiercely for shelf space.

And who runs the biggest risk?

You, the computer user, who only wants to buy the right program at a fair price.

Public Domain Software gives you the freedom to choose.

If you're searching for a business software program, PDS Business & Finance categories offer payroll programs, stock analysis, Visicalc formulas, inventory models, loan schedules and tax plans, among others.

For educators, PDS Education & School categories are great jumping off points for developing a full software library for your institution. And, the Math & Statistics, Chemistry, and Astronomy disks are good learning tools for even the advanced student.

What about games?

Well, first there's Eamon, with over 30 adventure disks. These adventure or role-playing games let you experience a world apart from reality. Within each scenario, you are the master and your fate is up to you.

Public Domain Software gives you the freedom to choose.

A good portion of the other PDS disks are all GAMES! Dragon mazes, pinball, puzzle generators, war games, and more adventures. Every PDS game disk has at least 10 games on it and some have a whole lot more. And, if you went through all the PDS disks available from the Computer Learning Center, you would probably find a match for every game available on the commercial market.

On those crowded shelves.

For a lot more money.

And all you pay for each of these PDS disks is $4. Not much when you consider that the $4 covers first class postage, insurance, and the disk itself.

The Computer Learning Center distributes PDS disks as a service to the computer user because we believe there should never be a limit to what you can know or do. PDS disks allow you to sample a wide variety of software programs without the risk of spending last week's savings on a program that isn't going to work for you.

Try a names file or sort utility PDS disk. See how the programs work and THEN go to your favorite software retailer and experiment with the utility programs they sell. Now that you know something about files and sorts, you will make a much better buying decision and have some $$ left over for more disks.

Like Art & Graphics. Or Apple BASIC tutorials if you're just starting out on programming.

The point of spending money is to get true value for the item purchased.

If back-up documentation exists, it is either in the program itself or in a separate file. All of the disks are supplied in DOS 3.3 16 sector format. Most of the PDS disks will work on the Apple III in emulation mode.

The Computer Learning Center will also make you another offer: if you like programming and have some sparetime, order a PDS disk that interests you and work on it. Clean-up, fix or add to a program on the disk and then return it to us with a note describing your changes. We will trade the program for another program of your choice or an entire disk if the changes are substantial.

There is very little that money can't buy. But the point of spending money is to get true value for the item purchased.

Spend a few dollars today and save hundreds tomorrow. You'll build a fine program library for yourself, buy the right commercial software (the programs you really need) and still have $$ left over.

COMPUTER LEARNING CENTER. P.O. BOX 45202. TACOMA, WA. 98444

Special introductory offer - 4 volumes for $15

Four categories with over 170 different programs in all.

Business & Finance Art & Graphics Math & Statistics Games

( ) Please send me a PDS catalogue. Here's my $1.

( ) OK. I'll try your disks. Here's my $15. *Offer ends 3/31/84*

Send To: Computer Learning Center P.O.Box 45202 Tacoma, WA 98444

Name _____

Address _____

City _____ **State** _____ **Zip** _____

Signature _____

Visa/MC _____ **Exp date** _____

Avoid the I.R.S.

(Inadvertently Ruined Software)

You need software insurance.

Diskettes are fragile, and when a protected program is damaged, the results are expensive and inconvenient. If you have a backup diskette, though, you can have your Apple, IBM or compatible computer back on line within seconds . . . affordably. That's software insurance.

Copy II Plus

This is the most widely used backup program for the Apple. Rated as "one of the best software buys of the year" by **InCider** magazine, its simple menu puts nearly every disk command at your fingertips. The manual, with more than 70 pages, describes protection schemes, and our **Backup Book™** lists simple instructions for backing up over 300 popular programs. The Backup Book is expanded bimonthly, and is always available to Copy II Plus owners for only $1.00 (and a self-addressed, stamped envelope). Best of all, Copy II Plus is still only **$39.95.**

WildCard 2 (formerly The Alaska Card)

Designed by us and produced by Eastside Software, WildCard 2 is the easiest-to-use, most reliable card available. Making backups of your total load software can be as easy as pressing the button, inserting a blank disk and hitting the return key twice. WildCard 2 copies 48K, 64K and 128K software, and, unlike other cards, is always ready to go. No preloading software into the card or special, preformatted diskettes are required. Your backups can be run with or without the card in place and can be transferred to hard disks. $139.95 complete.

Important Notice: These products are provided for the purpose of enabling you to make archival copies only. Under the Copyright Law, you, as the owner of a computer program, are entitled to make a new copy for archival purposes only, and these products will enable you to do so.

These products are supplied for no other purpose and you are not permitted to utilize them for any other use, other than that specified.

Copy II PC

This is **THE** disk backup program for the IBM PC, PC/XT and PCjr that copies almost anything. Others may make similar claims, but in reality, nothing out performs Copy II PC . . . at any price. Copy II PC even includes a disk speed check and is another "best buy" at only **$39.95.** Specify machine when ordering.

We are the backup professionals. Instead of diluting our efforts in creating a wide variety of programs, we specialize in offering the very best in backup products. So, protect your software investment, **before** the I.R.S. gets you.

CENTRAL POINT Software, Inc.

The Backup Professionals

To order, call 503/244-5782, 8:00-5:30 Mon.-Sat., or send your order to: Central Point Software, 9700 SW Capitol Hwy, Suite 100, Portland, OR 97219. Prepayment is required. Please include $2 for shipping and handling.