

For The Serious User Of Apple ][ Computers

Hardcore

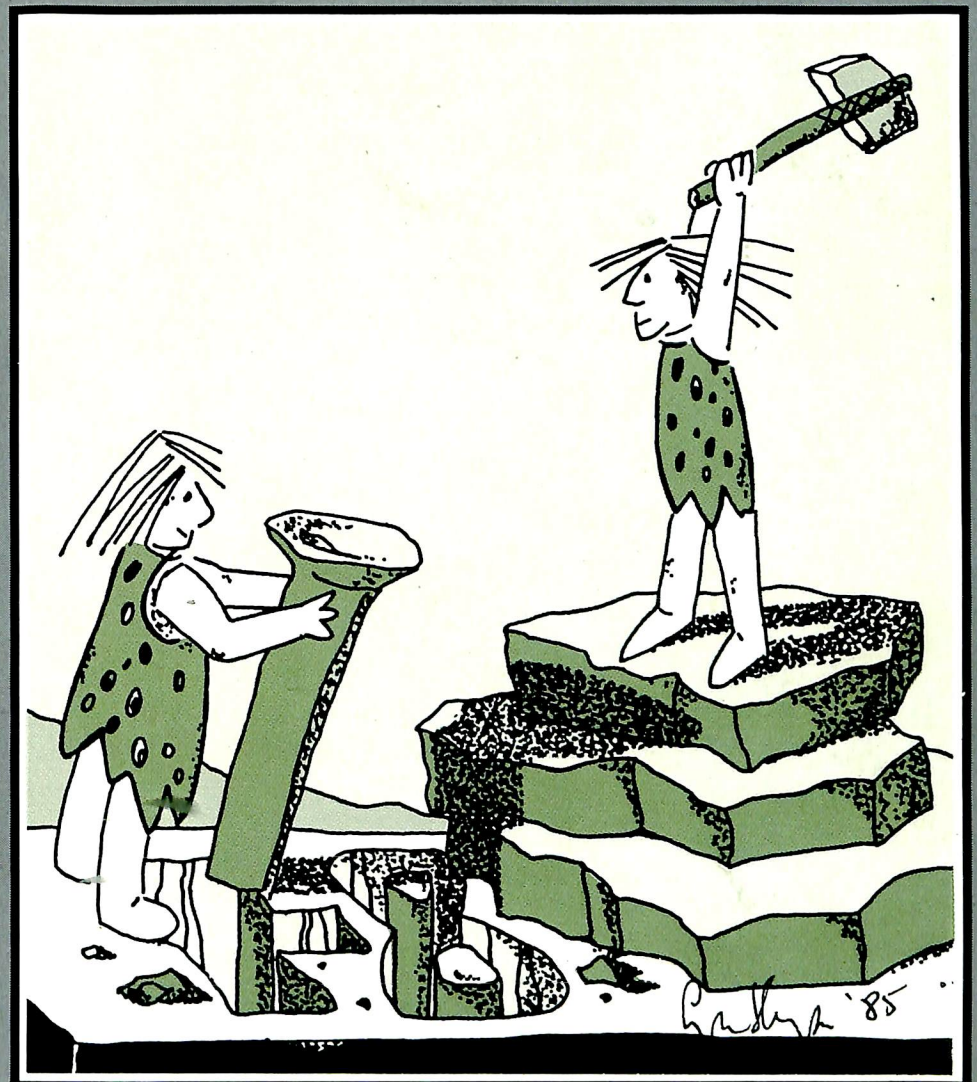
# COMPUTIST

Issue No. 19 \$3.75

The Games  
Of 1984: In Review  
Part II  
Pg. 12

Towards A  
Better F8 ROM  
Pg. 18

The Nibbler:  
A Utility Program  
To Examine Raw  
Nibbles From Disk  
Pg. 25



Double Your ROM Space

Pg. 9

BULK RATE  
U.S. Postage

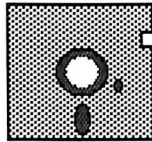
**PAID**

Tacoma, WA  
Permit No. 269

Hardcore COMPUTIST  
PO Box 110846-T  
Tacoma, WA 98411



APPLE USERS— FIX  
BROKEN DISKS WITH  
**MR. FIXIT**



- EDIT ANY SECTOR
- FIND T/S LISTS
- MAKE CATALOG ENTRIES
- FORMAT SINGLE TRACKS

SEND \$14.95 TO:

**THE SOFTWARE ASSET**  
133 RIDGEVIEW CIRCLE  
PRINCETON, NJ 08540

DEALER INQUIRIES INVITED

**WHO  
LIKES  
THE**



**HERE ARE JUST A  
FEW OF THE MANY**

- "an essential part of the Apple-user's repertoire" - APPLE USER
- "a valuable buy ... manual is practically worth having on its own" - WASHINGTON APPLE PI
- "the folks at Golden Delicious should be commended ... worth waiting for" - HARDCORE COMPUTIST
- "multifaceted" - NIBBLE
- "the most comprehensive disk accessor I have ever come across" - A.B., VERNON, CANADA
- "its ability to unlock other programs will greatly help me" - DR. B.P., SAN FRANCISCO, CALIFORNIA
- "an excellent set of programs ... just great - and good value too" - E.A.S., MILTON KEYNES, ENGLAND
- "very, very educational ... great manual ... it is FANTASTIC!!!!!!!" - J.C., TUCSON, ARIZONA
- "a very enlightening piece of software/book ... top of my list for good buys" - H.S., BLAINE, MINNESOTA
- "I like yours the BEST" - R.R., CHICAGO

Why all the excitement about the CIA (confidential information advisors)? Probably because it is the ONLY set of utilities (5 in all) which enable even a beginner to investigate, edit, locate, list, trace, rescue, translate, patch, repair, verify, examine, protect, unprotect, analyse, encrypt, and decrypt programs on normal AND protected disks. You also get the "CIA Files", a 65000+ word book which contains detailed instructions for using the C.I.A. plus easy-to-follow, hand-holding tutorials about patching, repair, formatting, encoding, protection, and numerous other disk topics. You'll find plenty of material here which has never before appeared in print. PROGRAMS NOT COPY PROTECTED

To put the 5 C.I.A. utilities, plus book, on the trail of your Apple II+, IIe, & IIc disks, send \$65.00 by check or money order to:

GOLDEN DELICIOUS SOFTWARE LTD.  
350 Fifth Avenue, Suite 3308, Dept H, New York, New York 10118



Now-the ultimate back-up system!

**EDD III<sup>®</sup> and  
TRAK STAR<sup>™</sup>**

**COMPLETE PACKAGE:  
159<sup>95</sup>\***  
Includes • TRAK STAR • 2-Drive Adaptor  
• EDD III • Trak Star Patching Software

PRICED SEPARATELY:

**TRAK STAR 99<sup>95</sup>**

2-Drive Adapter (required for 2-drive systems) \$12

Documentation: \$3

Refundable with the  
purchase of TRAK STAR

\* Please add \$3 for shipping  
& handling. Foreign airmail &  
handling, add \$8

**Save copying time  
with nibble programs**

- Works with nibble copy programs to display tracks and half tracks that the program accesses.
- Operates with any Apple<sup>®</sup>-compatible program.
- Save time by copying only the tracks being used.
- Displays up to 80 tracks and half-tracks; compatible with high density drives.
- If copied program doesn't run, Trak Star displays track to be recopied.
- Includes patching software for Trak Star.
- Compact size permits placement on top of disk drive.
- Does not use a slot in the Apple<sup>®</sup> computer.
- For Apple II, II+, IIe and compatibles.



Apple is a registered trademark of Apple Computer, Inc. EDD III is a trademark of Utilico Microware

**Midwest Microsystems**

To order, phone:  
913 676-7242

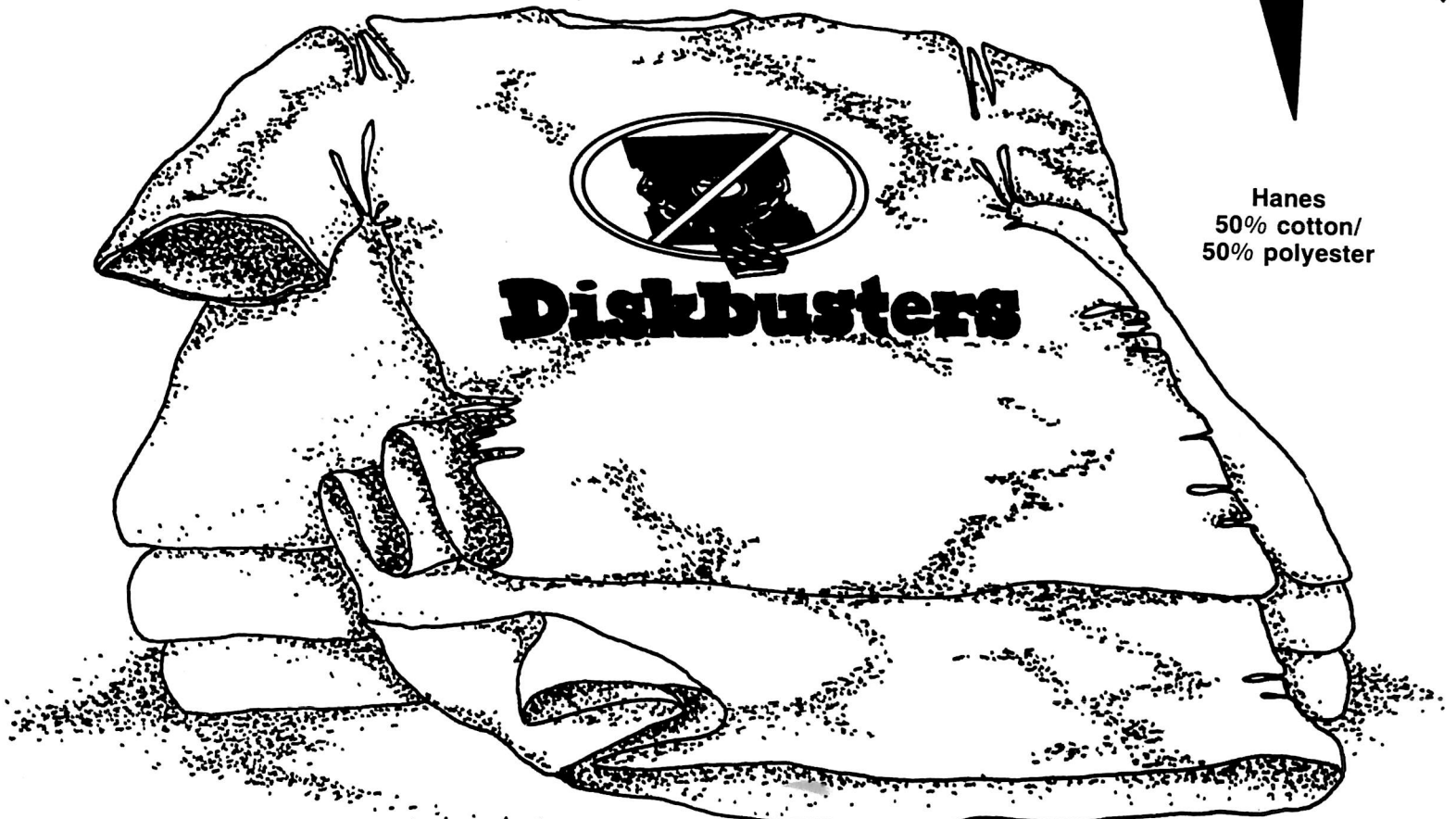
9071 Metcalf / Suite 124  
Overland Park, KS 66212

Spring  
is  
here!

**\$9.95**  
Only

# That means T-SHIRT weather!

What better way to show your friends  
*just* what you think of software copy-protection  
than to wear the ORIGINAL Hardcore COMPUTIST  
DISKBUSTERS t-shirt?!!!



Hanes  
50% cotton/  
50% polyester

Hardcore  
COMPUTIST  
says:

**“We ain’t afraid  
of no disk!”**

**YOU  
CAN,  
TOO!**

Rush me \_\_\_\_\_ (total) DISKBUSTERS T-shirts in the sizes indicated  
below. I have enclosed \$9.95 (plus applicable tax & shipping) for each shirt.

ADULT MENS:     Small     Med.     Large     X-Large

Name \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ St \_\_\_\_\_ Zip \_\_\_\_\_

Phone \_\_\_\_\_

VISA/MC \_\_\_\_\_ - \_\_\_\_\_ - \_\_\_\_\_ - \_\_\_\_\_ Exp \_\_\_\_\_

Signature \_\_\_\_\_

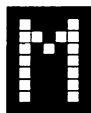
**Send check or money order to:** Hardcore T-shirts, PO Box 110816, Tacoma, WA 98411.  
Washington residents add 7.8% sales tax. Foreign orders add 20% shipping and handling. US funds  
drawn on US bank.

Use your  
VISA/MC,  
personal check  
or money order.

Shirts available  
in adult sizes  
**ONLY!**  
(black & red  
on gray).

**Hardcore  
COMPUTIST**

PO Box 110816 Tacoma, WA 98411



any of the articles published in Hardcore COMPUTIST detail the removal of copy protection schemes from commercial disks or contain information on copy protection and backup methods in general. We also print bit copy parameters, tips for adventure games, advanced playing techniques (APT's) for arcade game fanatics and any other information which may be of use to the serious Apple user.

Hardcore COMPUTIST also contains a center CORE section which focuses on information not directly related to copy protection. Topics may include, but are not limited to: tutorials, hardware/software product reviews and application and utility programs.

**What Is a Softkey Anyway?** Softkey is a term which we coined to describe a procedure that removes, or at least circumvents, any copy protection on a particular disk. Once a softkey procedure has been performed, the resulting disk can usually be copied by the use of Apple's COPYA program (on the DOS 3.3 System Master Disk).

**Commands and Controls:** In any article appearing in Hardcore COMPUTIST, commands which a reader is required to perform are set apart from normal text by being indented and bold. An example is:

**PR#6**

Follow this with the RETURN key. The RETURN key must be pressed at the end of every such command unless otherwise specified.

Control characters are indicated by being boxed. An example is:



To complete this command, you must first type the number 6 and then place one finger on the CTRL key and one finger on the P key.

**Requirements:** Most of the programs and softkeys which appear in Hardcore COMPUTIST require one of the Apple ][ series of computers and at least on disk drive with DOS 3.3. Occasionally, some programs and procedures have special requirements. The prerequisites for deprotection techniques or programs will always be listed at the beginning of the article under the "Requirements:" heading.

**Software Recommendations:** The following programs (or similar ones) are strongly recommended for readers who wish to obtain the most benefit from our articles:

- 1) **Applesoft Program Editor** such as Global Program Line Editor (GPLE).
- 2) **Sector Editor** such as DiskEdit, ZAP from Bag of Tricks or Tricky Dick from The CIA.
- 3) **Disk Search Utility** such as The Inspector, The Tracer from The CIA or The CORE Disk Searcher.
- 4) **Assembler** such as the S-C Assembler or Merlin/Big Mac.
- 5) **Bit Copy Program** such as Copy ][ Plus, Locksmith or The Essential Data Duplicator
- 6) **Text Editor** capable of producing normal sequential text files such as Applewriter ][, Magic Window ][ or Screenwriter ][.

You will also find COPYA, FID and MUFFIN from the DOS 3.3 System Master Disk useful.

**Super IOB:** This program appeared in Hardcore COMPUTIST No. 9, No. 14 and The Best of Hardcore Computing. Several softkey procedures will make use of a Super IOB controller, a small program that must be keyed into the middle of Super IOB. The controller changes Super IOB so that it can copy different disks. It is recommended that you get the latest version of this program (only appearing in Hardcore COMPUTIST No. 14).

**RESET Into The Monitor:** Many softkey procedures require that the user be able to enter the Apple's system monitor during the execution of a copy protected program. Check the following list to see what hardware you will need to obtain this ability.

**Apple ][ Plus - Apple //e - Apple compatibles:** 1) Place an Integer BASIC ROM card in one of the Apple slots. 2) Use a non-maskable interrupt (NMI) card such as Replay or Wildcard.

**Apple ][ Plus - Apple compatibles:** 1) Install an F8 ROM with a modified RESET vector on the computer's

motherboard as detailed in the "Modified ROM's" article of Hardcore COMPUTIST No. 6 or the "Dual ROM's" article in Hardcore COMPUTIST No. 19.

**Apple //e - Apple //c:** Install a modified CD ROM on the computer's motherboard. Don Lancaster's company (Synergistics) sells the instructions necessary to make this modification. Making this modification to an Apple //c will void its warranty but the increased ability to remove copy protection may justify it.

**Recommended Literature:** The Apple ][ Reference Manual and DOS 3.3 manual are musts for any serious Apple user. Other helpful books include: *Beneath Apple DOS*, Don Worth and Peter Lechner, Quality Software, \$19.95; *Assembly Language For The Applesoft Programmer*, Roy Meyers and C.W. Finley, Addison Wesley, \$16.95; and *What's Where In The Apple*, William Lubert, Micro Ink., \$24.95.

**Keying in Applesoft Programs:** BASIC programs are printed in Hardcore COMPUTIST in a format that is designed to minimize errors for readers who key in these programs. To understand this format, you must first understand the formatted LIST feature of Applesoft.

An illustration- If you strike these keys:

**10 HOME:REMCLEAR SCREEN**

a program will be stored in the computer's memory. Strangely, this program will *not* have a LIST that is exactly as you typed it. Instead, the LIST will look like this:

**10 HOME : REM CLEAR SCREEN**

Programs don't usually LIST the same as they were keyed in because Applesoft inserts spaces into a program listing before and after every command word or mathematical operator. These spaces usually don't pose a problem except in line numbers which contain REM or DATA command words. The space inserted after these command words can be misleading. For example, if you want a program to have a list like this:

**10 DATA 67,45,54,52**

you would have to omit the space directly after the DATA command word. If you were to key in the space directly after the DATA command word, the LIST of the program would look like this:

**10 DATA 67,45,54,52**

This LIST is different from the LIST you wanted. The number of spaces you key after DATA and REM command words is very important.

All of this brings us to the Hardcore COMPUTIST LISTING format. In a BASIC LISTING, there are two types of spaces; spaces that don't matter whether they are keyed or not and spaces that must be keyed. Spaces that must be keyed in are printed as delta characters (Δ). All other spaces in a Hardcore COMPUTIST BASIC listing are put there for easier reading and it doesn't matter whether you type them or not.

There is one exception: If you want your checksums (See "Computing Checksums" section) to match up, you *must not* key in any spaces after a DATA command word unless they are marked by delta characters.

**Keying In Hexdumps:** Machine language programs are printed in Hardcore COMPUTIST as both source code and hexdumps. Only one of these formats need be keyed in to get a machine language program. Hexdumps are the shortest and easiest format to type in.

To key in hexdumps, you must first enter the monitor:

**CALL -151**

Now key in the hexdump exactly as it appears in the magazine ignoring the four digit checksum at the end of each line (a "\$" and four digits). If you hear a beep, you will know that you have typed something incorrectly and must retype that line.

When finished, return to BASIC with a:

**E003G**

Remember to BSAVE the program with the correct filename, address and length parameters as given in the article.

**Keying In Source Code** The source code portion of a machine language program is provided only to better explain the program's operation. If you wish to key it in, you will need an assembler. The S-C Assembler is used to generate all source code printed in Hardcore COMPUTIST. Without this assembler, you will have to translate pieces of the source code into something *your* assembler will understand. A table of S-C Assembler directives just for this purpose was printed in Hardcore COMPUTIST No. 17. To translate source code, you will need to understand the directives of your assembler and convert the directives used in the source code listing to similar directives used by your assembler.

**Computing Checksums** Checksums are four digit hexadecimal numbers which verify whether or not you keyed a program exactly as it was printed in Hardcore COMPUTIST. There are two types of checksums: one created by the CHECKBIN program (for machine language programs) and the other created by the CHECKSOFT program (for BASIC programs). Both programs appeared in Hardcore COMPUTIST No. 1 and The Best of Hardcore Computing. An update to CHECKSOFT appeared in Hardcore COMPUTIST No. 18. If the checksums these programs create on your computer match the checksums accompanying the program in the magazine, then you keyed in the program correctly. If not, the program is incorrect at the line where the first checksum differs.

1) To compute CHECKSOFT checksums:

**LOAD filename  
BRUNCHECKSOFT**

Get the checksums with

**&**

And correct the program where the checksums differ.

2) To compute CHECKBIN checksums:

**CALL -151  
BLOAD filename**

Install CHECKBIN at an out of the way place

**BRUN CHECKBIN, A\$6000**

Get the checksums by typing the starting address, a period and ending address of the file followed by a

**xxx.xxx**

And correct the lines at which the checksums differ.

## How-To's Of Hardcore

Welcome to Hardcore COMPUTIST, a publication devoted to the serious user of Apple ][ and Apple ][ compatible computers. Our magazine contains information you are not likely to find in any of the other major journals dedicated to the Apple market.

Our editorial policy is that we do NOT condone software piracy, but we do believe that honest users are entitled to backup commercial disks they have purchased. In addition to the security of a backup disk, the removal of copy protection gives the user the option of modifying application programs to meet his or her needs.

New readers are advised to read this page carefully to avoid frustration when attempting to follow a softkey or when entering the programs printed in this issue.



# Hardcore COMPUTIST

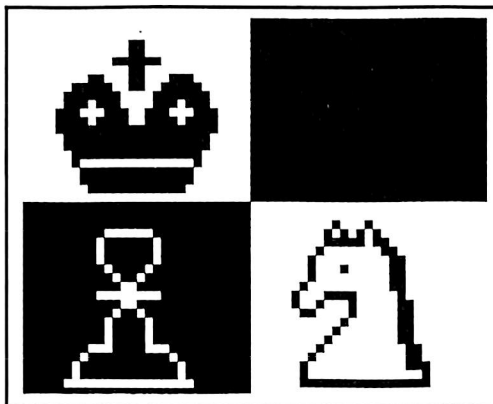
Publisher/Editor: Charles R. Haight Technical Editors: Ray Darrah, Gary Peterson  
Production & Graphics: Lynn Campos-Johnson Circulation: Michelle Frank

Advertising: (206) 474-5750 Printing: Grange Printing, Inc., Seattle, WA

Hardcore COMPUTIST is published monthly, except December, by SoftKey Publishing, 5233 S. Washington, Tacoma, WA 98409  
Phone: (206) 474-5750



Pg. 12



Address all advertising inquiries to Hardcore COMPUTIST, Advertising Department, PO Box 110816, Tacoma, WA 98411. Mail manuscripts or requests for Writers Guides to Hardcore COMPUTIST, PO Box 110846-K, Tacoma, WA 98411.

Return postage must accompany all manuscripts, drawings, photos, disks, or tapes if they are to be returned. Unsolicited manuscripts will be returned only if adequate return postage is included.

Entire contents copyright 1985 by SoftKey Publishing. All rights reserved. Copying done for other than personal or internal reference (without express written permission from the publisher) is prohibited.

The editorial staff assumes no liability or responsibility for the products advertised in the magazine. Any opinions expressed by the authors are not necessarily those of Hardcore COMPUTIST magazine or SoftKey Publishing.

Apple usually refers to the Apple II or II Plus Computer, and is a trademark of Apple Computers, Inc.

**SUBSCRIPTIONS:** Rates: U.S. \$25.00 for 12 issues, Canada \$34.00, Mexico \$39.00, Foreign (airmail) \$60.00, Foreign (surface mail) \$40.00. Direct inquiries to: Hardcore COMPUTIST, Subscription Department, PO Box 110846-T, Tacoma, WA 98411. Please include address label with correspondence.

**DOMESTIC DEALER RATES:** Call (206) 474-5750 for more information.

**Change Of Address:** Please allow 4 weeks for change of address to take effect. On postal form 3576 supply your new address and your most recent address label. Issues missed due to non-receipt of change of address may be acquired at the regular back issue rate.

## 9 Double Your ROM Space

This procedure, which utilizes a 2732 EPROM as a replacement for your F8 ROM, will allow you to drop into the monitor at will. Even if you are *already* able to pop into the monitor, you will now even be able to boot a program which checks for a modified F8 ROM. Be warned, however, that this article is not for people who get queasy around a soldering iron. *By Ray Darrah.*

## CORE SECTION

## 12 The Games Of 1984: In Review - Part II

Looking for entertainment value? Before you spend any more of your hard-earned dollars on computer games that *aren't quite* what you wanted, look through this second installment of Hardcore COMPUTIST's Games of 1984: In Review. You'll find critiques for several of the most recently released top-of-the-line games for the Apple computer. Last month we reviewed adventure games; this month, arcade games. *By Jeff Hurlburt.*

## 18 Towards A Better F8 ROM

Are you ready to explore one of the best enhancement packages available for the Apple's F8 ROM? When used in conjunction with, "Double Your ROM Space" by Ray Darrah, the techniques outlined in this article will transform your computer into a super Apple that can boot ANY program and still pop into the monitor. Extra added features: a search routine and continuous disassembly. An absolute MUST for the serious machine language programmer. *By Earl Taylor.*

## 25 The Nibbler: A Utility Program To Examine Raw Nibbles From Disk

Examine disk formatting and protection with this utility which can read the raw data from any full or half-track. Use the Nibbler's search option to help you quickly spot those altered address and data marks. *By Jan G. Eugenides & Ray Darrah.*

## DEPARTMENTS

### 4 INPUT

Removing Boot-Up Nibble Counts

*By Philby Burgess*

### 6 READERS' SOFTKEY & COPY EXCHANGE

## 17 ADVENTURE TIPS

Softkey For Rendezvous With Rama

*By Jeff Lucia*

A Backup For Peachtree's "Back To Basics" Accounting System

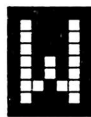
*By Clay Harrell*

Softkey for The HSD Statistics Series

*By Dr. Phillip Romine*



## RAMcard and Alphabet Zoo



With the aid of Ken Greenlaw's article, "Secret Weapon: RAMcard" and the information from Hardcore COMPUTIST No.'s 8 and 10, my Apple //c was able to produce a backup copy of Essential Data Duplicator III. However, while the program was happily running in auxiliary memory, no text was visible and the graphics were only visible after a minor modification to XFER.BOOT. The change I had to make was

```
306:8D 5F C0 STA ANN3.OFF ;  
Double Hi-res Off!
```

Unfortunately, I was unable to find the correct switches to enable me to display text. I would appreciate any suggestions that you might have to correct this problem. Otherwise, XFER.BOOT and RESTORE work quite well. Many thanks, and now on to another subject.

Alphabet Zoo from Spinnaker Software is a wonderful set of 3 programs designed for small children. Of course, dirty fingers, bubble gum and peanut butter could instantly ruin this disk. It is a disk which cries out to be copied, but none of my nibble copiers will copy it. Due to the disk's heavy copy protection, it more often than not pronounces, "THIS IS A DEFECTIVE DISK" upon booting on my old Apple ][+, making me wonder if the kids have finally done the disk in for good. Even though I have written Spinnaker about the problem, I have not yet received a reply.

An examination of track \$3 of this disk reveals an unusual track which is responsible for foiling my nibble copiers. It is full of FF's, except for a couple of widely spaced D5's, an F7, and, I think, an AF. A boot trace revealed some amazing code which transforms itself and a routine which is used to count the number of FF's between the two D5's. There is also a nasty routine which overwrites memory and prints the defective disk message if the count is not exactly correct; however, this nasty routine does not transform itself, and a very small fix to the disk can remove the problem of defective booting. Formatting track \$3 of a copy of Alphabet Zoo also permits making a COPYable backup.

For you fretting parents, this plain and simple fix should work for you. First, format a blank disk and then use a nibble copier to copy tracks \$0-\$2 and \$4-\$22. Then use a sector editor to place the following code starting at

```
byte $4B of track $00, sector $01  
20 58 FC A9 10 8D 0F BC 4C C9 BB
```

This translates to:

```
20 58 FC JSR $FC58 ; Home  
A2 10 LDX #$10  
8D 0F BC STA $B00F  
4C C9 BB JMP $B0C9
```

Don't forget to write this sector back to the disk and put a write protect tab on the backup copy.

David Parkhurst  
Glendale, CA

*Mr. Parkhurst: You are absolutely right. The correct softswitch to turn off double hi-res graphics is \$C05F. As for the problem of not being able to see any of the text in the auxiliary memory- this is probably due to the protected program writing to location \$C00C during its boot up, with the effect that 80-column text is turned back off. You will also have this same problem if you boot the version of DOS 3.3 that is distributed with the Apple //e into the auxiliary memory because of the patch in the \$BA69 area (formerly free-space) that does a STA \$C00C.*

*Short of trying to patch your original disk to remove the write to \$C00C, we can't think of any way to get around this problem. However, in most cases, this really should not limit the usefulness of Mr. Greenlaw's technique.*

## Wolfenstein APT



I think that your magazine is the best thing that ever happened to the computer industry. I read my old copies almost daily, looking for bits and pieces of info. It could be a little longer (perhaps 32 pages longer), but I can understand the search for material. Personally, I would like to see some more APT's, or even a disk of APT's. Anyway, I've come up with my own for Castle Wolfenstein that I haven't seen in your mag before. It's a little wordy, but it does work (at least, on my version):

When an SS is following you, he always takes the same path as you. Make sure you have a bulletproof vest so you can do this:

Let him follow you into an empty (no live guards) room, and stand one space away from the exit/entrance with your gun pointed in the direction from which you came. Unless there is a body between you and the doorway, he will follow you shortly. When he does, he will

run into your gun (don't move!) and stop with his hands up. Now, press "U". It will either say "searching..." or "Putting on bullet proof vest...10 seconds". The latter is fine, and then he doesn't have a vest, so you can shoot him (or whatever). The former, however, requires a little digital dexterity.

First let it finish the search of the SS. Then, run into him (yes, RUN INTO HIM), and before he can react, pull out your gun again. This can be done quite easily although it sounds nearly impossible. Then, when he is held up again, you can safely remove his vest with "U".

P.S. I especially enjoyed Ray Darrah's, "Batman Decoder Ring" article in Issue No. 14. Thank you, and keep up the excellent work.

Andrew Lundsten  
Buffalo, MN

## Dollars & Sense Backup



e. Your Most Wanted List. I haven't had problems with these programs: Dollars & Sense: For Version III.12, use the sector mod in the documentation to Copy II+, v4.4c (i.e. put the following code starting at byte \$8C of track \$0, sector \$3:EA A5 02 38 E9 40 85 04 A5 03 E9 00 85 05 A0 3F B1 04 91 02 88 10 F9). For Version III.14, the above method won't work, but I had no problem using Locksmith 5.0, default modes (that was Rev. Level F).

Word Juggler: On my version (2.6) for the //e, this also copied straight on LS 5.0f.

Harry J. Seaman  
Milwaukee, WI

## What's a ROGRAM?



I am writing this letter to ask for your assistance. One day, while fooling around on my Apple ][+ I got a "ROGRAM TOO LARGE" error. I had booted up a normally initialized slave diskette, entered the monitor and typed, "A54FG", the INIT command handler, without giving the necessary parameters. The disk drive started to spin for awhile and suddenly the "ROGRAM TOO LARGE" error popped up. What does it mean?



# Input cont...

How did it get there?

Christopher Hamlin  
Park Ridge, NJ

**Mr. Hamlin:** I guess you must have been sleeping when your Computer Science 101 teacher explained what a rogram is (just kidding, Chris).

Basically what has happened is that an error (probably **VOLUME MISMATCH** because the volume number in the File Manager parameter list did not match the volume No. 254 that the INIT handler defaults to, or a **FILE NOT FOUND** error because the **HELLO** file has not been opened by DOS) occurred during the initialization of the disk. Normally, when a DOS error occurs, the I/O hooks at \$36-\$39 point to the monitor input/output routines and not the DOS input/routines. However, when you call one of the DOS commands directly, the I/O hooks are abnormally set to the DOS I/O routines. This causes the index into the text of the input commands to get messed up as the error handling progresses.

When the error handling routine at \$A6D2 is entered, one of the first things that happens is that the error code number is saved at \$AA5C. Next, 3 characters are printed, **RETURN, BELL, RETURN**. Usually these characters are passed directly to **COUT** at \$FDED but, with the I/O hooks pointing to DOS, the printing goes through \$9EBD where the final value that the accumulator held is stored into \$AA5C. Thus, after the printing has occurred, \$AA5C incorrectly holds an \$8D (ASCII for **RETURN**) instead of the error code (6 for File Not Found, 7 for a Volume Mismatch).

The end result of \$AA5C holding this \$8D is that the pointer to the text of the error message points one byte beyond the text for the **"PROGRAM TOO LARGE"** and your mysterious, **"ROGRAM TOO LARGE,"** message is displayed, no matter what the error that actually occurred happened to be.

To get around this problem, make sure that DOS is disconnected by performing a call to \$9EE0 before you try to call any of its command handlers directly. To see the difference that disconnecting DOS makes, open the door of your disk drive, enter the monitor and then directly call the **CATALOG** command handler by typing

**A56EG**

Does this error message look familiar?

Then, to correct the error message problem,

type

**9EE0G N A56EG**

The correct message, **"I/O ERROR"**, should be displayed.

For a more in-depth discussion of the **"ROGRAM TOO LARGE"** problem, see either Rick Sutcliffe's article, **"ROGRAM and Other DOS Traps"** in **"All About DOS"** (Call **A.P.P.L.E** in Depth No. 3) or, **"ROGRAM TOO LARGE???"** by Lee Meador in the May 1983 issue of **Apple Assembly Lines**.

## Mind Prober

If any other new subscribers out there are interested, I used the Super IOB procedure for Millionaire from Issue No. 15 to deprotect Mind Prober. After making the "normal" copy, it's only necessary to load the **HELLO** program, delete all but the first and last lines and resave **HELLO**. If one has a penchant for customizing, the title screen is **PICTR.TEXT**. Fontrix, used in the normal mode with the foreground set to blue (03) and a black background (default if not already changed), will blend right into the existing screen. It took me longer to customize the screen to my satisfaction than was required to deprotect the program!

One additional note to the novice: it may seem obvious (it does to me, now), but the **RWTS** from the original disk must be saved to the disk containing the Super IOB. I don't care to recall how long I stumbled around the first time I tried to use Super IOB and I didn't even have to type it in!

Keep up the good work! I look forward to future issues and to receiving the program disks regularly.

H.E. Ford  
Lilburn, GA

## Calling Brian

Let me direct your attention to an article in **Hardcore COMPUTIST** No. 16, page 7, column 1, in which an old friend's name is mentioned—**"The Fabled B. Fitzgerald"**, whom I had the good fortune (and misfortune) to meet in the late **"Softline"** Magazine.

If Brian should be within earshot, please convey my thanks for several most interesting program listings and articles. I was really hooked and waited breathlessly for each new

issue. Then it happened. Somewhere in the middle of the machine language program, after successfully getting airborne and marveling at the crisp "bird's-eye view" through the window of my instrumentless cockpit, the **"Image Maker"** disappeared! Now, once again, there is hope?!

Brian, if you are there, **PLEASE** get me some instruments so I can get my feet back on the ground.

E. Killingbeck  
Romulus, MI

You  
will  
**MISS**  
**YOUR NEXT**  
**ISSUE**  
of **Hardcore**  
**COMPUTIST...**

...if your mailing label  
**ON THIS ISSUE** tells you  
that you have 0 issues  
remaining in your  
subscription.

**CHECK IT OUT NOW!**

We **DO NOT** send any other  
subscription renewal notice.

**Please address letters to:**  
**Hardcore COMPUTIST**, Editorial  
Department, PO Box 110846-K,  
Tacoma, WA 98411. Include your  
name, address and phone.  
Correspondence published in the  
**INPUT** section may be edited for  
clarity and space requirements.

# Readers' Softkey & Copy Exchange

## Softkey For Rendezvous with Rama

By Jeff Lucia

### Requirements:

Four blank disks  
Super IOB  
Rendezvous With Rama  
COPYA



endezvous with Rama is an interactive adventure based on the novel by Arthur C. Clark. The player's objective is to board an unknown space ship which is approaching Earth. After boarding the ship, you must talk to the aliens and find out as much as possible about them. Because the ship is drifting further into space, if you don't leave the alien ship in time, you will never be able to return home.

### The Protection

After trying to copy my Rendezvous disks, I was happy to discover that Trillium (recently renamed Telarium) only protected side one of disk one. The other disks are accurately copied with COPYA or any other normal DOS disk copier.

In my opinion, protecting even one out of four sides of a software package is too many. The side that is protected checks for a nibble count on track \$10. If you copy the disk and even nibble count track \$10, the copy will probably come up with the message, "THIS DISK IS A DEFECTIVE DISK". However, after much code searching I discovered how to circumvent the nibble count routine.

### The Cookbook

- 1) Copy sides two, three and four using COPYA or a normal DOS disk copier. Be sure to label the disks appropriately.
- 2) Copy Side one by using the Super IOB controller at the end of this article or some other copier (such as Locksmith Fast Copy) that will ignore the errors on track \$10.
- 3) After the disk has been copied, boot normal DOS. insert you copy of side one and load the program named "IO"

#### BLOAD IO

- 4) Enter the monitor

#### CALL -151

- 5) Make this small modification to disable the nibble count

```
1BF5:20 29 1C
```

(Note: Originally these bytes were AD 82 C0. The change bypasses the call to the nibble count routine that resides in the language card.)

- 6) Save "IO" back to your copy

**BSAVE IO, A\$A00, L\$1512**

### Final Comments

This procedure, or a variation of it, might work on other Trillium programs. And although the graphics in Rendezvous with Rama are good, if you are tired of looking at them, try typing "PICTURESOFF".

### Rama Controller

```
1000 REM RENDEZVOUS WITH RAMA
1010 TK = 0 : ST = 0 : LT = 35 : CD = WR
1020 T1 = TK : GOSUB 490
1030 GOSUB 430 : GOSUB 100 : ST = ST + 1 : IF ST
< DOS THEN 1030
1040 IF BF THEN 1060
1050 ST = 0 : TK = TK + 1 + (TK = 15) : IF TK < LT
THEN 1030
1060 GOSUB 490 : TK = T1 : ST = 0
1070 GOSUB 430 : GOSUB 100 : ST = ST + 1 : IF ST
< DOS THEN 1070
1080 ST = 0 : TK = TK + 1 + (TK = 15) : IF BF = 0
AND TK < LT THEN 1070
1090 IF TK < LT THEN 1020
1100 HOME : PRINT "DONE^WITH^COPY" : END
```

### Controller Checksums

1000	- \$356B	1060	- \$4A39
1010	- \$3266	1070	- \$423C
1020	- \$C11A	1080	- \$3D8F
1030	- \$D71B	1090	- \$A6B7
1040	- \$D748	1100	- \$60A2
1050	- \$6AE3		

## A Backup for Peachtree's, "Back to Basics" Accounting System

By Clay Harrell

### Requirements:

Apple //e  
An 80-column card (required by Peachtree)  
A sector editor program  
Back to Basics from Peachtree Software



inally, there's an accounting system that performs well and even has an excellent check printing program to boot. Peachtree has created this nice accounting package which operates quite efficiently and comes with excellent documentation. The only drawback is that the package does require that you have an Apple //e.

After your initial enthusiasm for the accounting package (shooting down aliens was becoming rather boring, anyway), you may try to make backups of your new (and expensive!) accounting system. Lo and behold! COPYA makes copies of the originals with no problems. But when you boot them up, guess what? They seem to load and then just hang there spinning the disk drive like you put a DOS 3.2 disk in by mistake!? Obviously, Peachtree has put some protection in to provide some fun (or, should I say, aggravation).

But Peachtree has really done us a favor by not wiping-out memory and re-booting after failing to read our COPYA copy of their program. Instead, the program just sits there and keeps trying to read the disk, expecting some perverted format to magically appear from our copy. This gives us the opportunity (at our leisure, thanks to Peachtree) to interrupt the program, locate the checksum routine, and defeat it. Thus, we will end up with a COPYA version of their accounting system.

To test our theory, after making a COPYA copy of one of the sides of the accounting system, boot the disk. After the program has loaded and it is just sitting there spinning the disk drive, open the disk drive door and put the original disk in. The program immediately rewards you by stopping the drive and continuing with the program's main menu.

All we need to do now is defeat this nasty routine and we will have deprotected the accounting system (yes, we have to do this for all three disks).

Now the normal and ordinary man with a normal and ordinary Apple could spend many hours looking for the routine that is checking the disk. One way of doing this is to do a boot code trace, but this is as much fun as contracting an incurable disease and is best avoided.

If you have a Replay II card, this can be most handy in locating the offending code. Simply make a COPYA copy, boot the copy and, when the program hangs, press the Replay II button. Now enter the Replay II monitor and write down the location of the Program Counter. This is the current address being executed. Finally, reset into the monitor and examine the code there!

The code should look like this (assuming you are using the General Ledger disk):

8CF7-	LDA #S00	load accum with 0
8CF9-	STA \$B7F4	command code = "seek"
8CFC-	STA \$B7EC	to track 0.
8CFF-	STA \$B7EB	match any volume number



```
8D02- LDY #E8
8D04- LDA #B7
8D06- JSR $B7B5 do the seek to track 0
8D09- LDA #00
8D0B- STA $48
8D0D- LDX $B7E9 X = disk slot# * 16
8D10- LDA $C089,X
8D13- LDA $C08E,X read the disk
8D16- LDY #02 and compare the
8D18- LDA $C08C,X accumulator to $8CE9
8D1B- BPL $8D18 indexed by Y ($8CEB)
8D1D- CMP $8CE9,Y
8D20- BNE $8D16 Go to $8D16 if not equal
8D22- DEY
8D23- BPL $8D18
8D25- JSR $8CE6
8D28- JSR $8CE6
.
.
.
8D39- JMP $0846 jump to start of program
```

The above code seeks the read/write head of the disk drive to track zero, then reads the disk and compares a byte to the contents of location \$8CEB. If the byte read from the disk and location \$8CEB are not the same, it goes back to the beginning of the routine and starts over. That is why our COPYA copy hangs and keeps spinning the drive.

Peachtree would have been more wise to keep track of how many times the two bytes do not match up and after so many tries, clear memory and reboot. This would have made finding the checksum routine more difficult (at least you would have to be more timely).

To defeat this routine, at \$8D1B we can insert a JMP (jump to) \$8D25 which bypasses the whole read and check routine. To see if this works, type

```
8D1B:4C 25 8D
8CF7G
```

Sure enough, the main menu appears! Now, all that we have to do is to find this code on the disk and use a sector editor to change it. On the General Ledger disk you'll find the offending code at track \$13, sector 0, byte \$AA. Change this from \$10 FB D9 to \$4C 25 8D and write the sector back out.

The two other disks use the exact same protection code, but at different locations, and you can use the same methods to find the code as just described. You will find the code at \$7F67 on the Accounts Receivable disk and \$82A8 on the Accounts Payable disk.

## In Cookbook Fashion

1) Boot your DOS 3.3 System Master and execute COPYA

## RUN COPYA

2) Copy all three sides of the Peachtree Accounting System to three blank disks  
3) Run your sector editor and make the following changes to the COPYA version of the disks:

General Ledger:

```
-----
Track $13, Sector 0, Byte $AA
change from $10 FB D9
to $4C 25 8D
```

Accounts Receivable:

```
-----
Track $12, Sector $C, Byte $A9
change from $10 FB D9
to $4C 95 7F
```

Accounts Payable:

```
-----
Track $13, Sector 8, Byte $A1
change from $10 FB D9
to $4C D6 82
```

4) Write the sectors back out and you're all done!

## Softkey for The HSD Statistics Series By Dr. Phillip Romine

HSD ANOVA II (V1.1)  
REGRESS II  
STATS +

Human Systems Dynamics  
9010 Reseda Blvd., Suite 222  
Northridge, CA 91324  
(800) 451-3030  
ANOVA II and REGRESS II \$150  
STATS+ \$200

### Requirements:

Apple ][ Plus or equivalent  
One disk drive  
Super IOB v1.2  
Tricky Dick or Copy ][ Plus 5.0  
Original HSD program(s)  
One blank disk per copy



The HSD Statistics Series can be easily deprotected using Super IOB in conjunction with a sector editor which permits header and trailer changes (like the one in the Copy II+ Utilities or Tricky Dick from the CIA Files). One of these sector editors is a great help because it eliminates the need to write a complex Super

IOB Controller for the HSD disks.

The general procedure is to copy tracks \$03 thru \$22 of the HSD disk using Super IOB and the HSD Controller listed at the end of this article. The controller is configured to read the HSD disks with address marks of D5 AA CE & CE AA and data marks of D5 AA B5 & D6 AA but will not be able to read two sectors on each disk because these sectors have normal data marks. Rather than modifying the controller to take these sectors into account, I just used Tricky Dick to manually write them to the destination disk.

That's all. Here is the general HSD procedure that you can use to deprotect ANOVA II, REGRESS or STATS+.

1) Use Super IOB and the HSD Controller to copy Tracks 3 thru 22 to an initialized disk without a HELLO program.

## RUN SUPER IOB

Ignore the disk drive complaints which occur while Super IOB is reading some sectors of the original disk.

2) Boot up Tricky Dick (or another disk editor) and set the address prologue to D5 AA CE and the address epilogue to CE AA.

3) Read in the desired sector (use the data listed below) from the original HSD disk.

HSD ANOVA II - Track \$11, Sector \$E  
& Track \$21, Sector \$E  
HSD REGRESS - Track \$11, Sector \$E  
& Track \$1D, Sector \$4  
HSD STATS + - Track \$11, Sector \$E  
& Track \$0B, Sector \$E

or Alternatively

HSD STATS + - Track \$11, Sector \$0  
Track \$11, Sector \$C  
Track \$11, Sector \$D  
Track \$11, Sector \$E  
Track \$0B, Sector \$E

4) Restore the address prologue to D5 AA 96 and the address epilogue to DE AA.

5) Write the sector out to the Super IOB copy of the HSD disk.

6) Repeat steps 2 through 5 for the second sector of the disk in question.

That's all there is to it. Your copy is completely deprotected and awaits your modifications. The use of a fast DOS (I used Pronto DOS) for initializing your blank disks will improve the performance of the finished product markedly. I also might mention that Nick Galbreath's, "The Controller Writer" (HC No. 16) can be used to create the HSD Controller below.

# Exchange cont...

## HSD Controller

```

1000 REM HSD STATS SERIES CONTROLLER
1010 TK = 3 : LT = 35 : CD = WR : MB = 151 : ONERR
      GOTO 550
1020 ST = 0 : T1 = TK : GOSUB 490 : RESTORE : GOSUB
      190 : GOSUB 210 : GOSUB 170
1030 GOSUB 430 : GOSUB 100 : ST = ST + 1 : IF ST
      < 16 THEN 1030
1040 IF BF THEN 1060
1050 ST = 0 : TK = TK + 1 : IF TK < LT THEN 1030
1060 GOSUB 230 : TK = T1 : ST = 0 : GOSUB 490
1070 GOSUB 430 : GOSUB 100 : ST = ST + 1 : IF ST
      < 16 THEN 1070
1080 ST = 0 : TK = TK + 1 : IF BF = 0 AND TK < LT THEN
      1070
1090 IF TK < LT THEN 1020
1100 HOME : AS = "ALL^DONE" : GOSUB 450 : END
5000 DATA 213 , 170 , 206 , 213 , 170 , 181 , 206
      , 170 , 214 , 170
  
```

## Checksums

1000	- \$356B	1060	- \$6AE6
1010	- \$5E3F	1070	- \$22FD
1020	- \$B92C	1080	- \$54D8
1030	- \$E2AA	1090	- \$7FC2
1040	- \$2463	1100	- \$F952
1050	- \$E2BC	5000	- \$33FC

## Removing Boot-Up Nibble Counts

By Philby Burgess

Arithmetickle  
Arithmekicks  
Houghton Mifflin Software  
One Beacon Street  
Boston, MA  
\$39.95 each

Early Games for Children  
Springboard Software  
7807 Creekridge Cr.  
Minneapolis, MN 55435  
\$34.95

### Requirements:

Apple II Plus or equivalent  
One disk drive  
A disk nibble utility (optional)  
Super IOB 1.2 & Swap Controller



I have found that many programs on the educational software market cannot be copied with a bit copier, but can be deprotected quite easily with the use of Super IOB and the Swap Controller. Quite easily, that is, as long as the

protected program can be halted by some means (see page 2 if you don't know how to obtain this ability) in order to capture the protected disk's RWTS.

However, with some programs, Super IOB will copy merrily along until it reaches a certain track from which it cannot read any of the sectors. Because the Swap Controller's uses the "Ignore Unreadable Sectors" subroutine, Super IOB will try to read each of the sectors on the track in question before giving up and moving on to the next sector.

The most common reason that this track cannot be read is that it is a "nibble count" track (a track that is used to verify the presence of an original disk). Normally, one of these tracks will not contain any sectors at all, but will consist mainly of sync bytes (usually FF's), possibly with a few other values interspersed here and there. Trying to read this nibble count track can greatly increase the time required to copy a disk.

On some disks that exhibit this phenomena, I have found that modifying the Swap Controller so that the nibble count track is skipped over when reading and writing that the Super IOB copy of the disk will work perfectly. This is because these disks verify their nibble count only once, during boot-up, and not later when it would be much harder to track down and disable. Just moving the programs over to a disk that has normal DOS removes the access of the nibble count track.

I have used this procedure on Arithmekicks and Arithmetickle from Houghton Mifflin Software and Early Games for Children from Springboard Software. Coincidentally, all three of these disks check track \$A in order to verify the presence of an original disk. It was an easy task to modify lines 1050 (reading) and 1080 (writing) of the Swap Controller so that track \$A was never accessed.

With other disks, of course, the nibble count track could be practically anywhere. I suggest using The CIA's Linguist module to first find out what the address and data marks have been changed to and, once you have this data, incorporate it into Tricky Dick and use the Tracer module to verify the entire disk. As long as the disk has only used a single set of address and data marks, the nibble count track will be a track which has no verifiable sectors. Alternatively, the Linguist or another nibble utility could be used to manually scan the disk for the nibble count track. Look for a track that is comprised mainly of sync bytes (FF's).

Once the nibble count track has been determined, modify lines 1050 and 1080 of

the Swap Controller so this track will be skipped. To do this, you have to add an expression like "+ (TK=9)" to the statements where the track number variable (TK) is manipulated. This sets up a "Boolean" (True or False) expression that evaluates as a zero (false) except when the nibble count track is to be accessed. When TK=9, the expression evaluates as a 1, or true, with the effect that the nibble count track is stepped over. Notice that the variable TK must be checked to see if it contains a value of one less than actual the nibble count track.

Here is the general procedure for you to follow:

1) Boot up the protected disk and halt it by whatever means are available to you (RESET, NMI etc.). If you have an Apple //e or //c you can use the auxiliary memory technique that was described in Ken Greenlaw's article, "Secret Weapon: RAMCard," (Hardcore COMPUTIST No. 16).

2) From the monitor, move the protected disk's RWTS to a location that is safe from a reboot

**1900<B800.BFFF**

or, if you have booted the program into auxiliary memory

**1900<B800.BFFF** 

3) Boot up a DOS 3.3 slave disk. (Note: This step is not necessary if you booted the program into auxiliary memory.)

**C600G**

4) Save the protected RWTS using an appropriate file name

**BSAVE PROTECTED.RWTS,  
A\$1900,L\$800**

5) Install the Swap Controller into Super IOB. If necessary, modify the controller so that the nibble count track is skipped over and the correct file name (ie. assigned in Step 4) is referenced by line 10010. Below I have listed lines 1050 and 1080 of the Swap Controller, modified so that track \$A will be skipped over.

```

1050 ST = 0 : TK = TK + 1 + (TK = 9) : IF TK < LT
      THEN 1030
1080 ST = 0 : TK = TK + 1 + (TK = 9) : IF BF = 0 AND
      TK < LT THEN 1070
  
```

6) Run Super IOB and copy to a formatted disk, preferably one that contains a fast DOS.



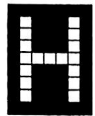
# Double Your ROM Space

By Ray Darrah

## Requirements:

An Apple ][ Plus or compatible  
Access to an EPROM burner  
A SPDT toggle switch with no center position  
2732-type EPROM  
Wire Wrap wire  
A soldering iron with a fine tip

**Warning:** The procedure described in this article should only be attempted by those persons having computer hardware experience. SoftKey Publishing assumes no liability for damage done to the computer if this procedure is followed.



Hardcore COMPUTIST has received quite a number of inquiries from Apple ][ Plus owners asking, "How can I RESET into the monitor so that I can perform the softkey procedures described in your magazine?" Basically, there are three different methods for ][ Plus owners to obtain this RESET ability:

- 1) Install an Integer BASIC firmware card in one of the Apple's slots.
- 2) Install a NMI card such as Replay or Wildcard in one of the Apple's slots.
- 3) Install a modified F8-ROM on the Apple's motherboard.

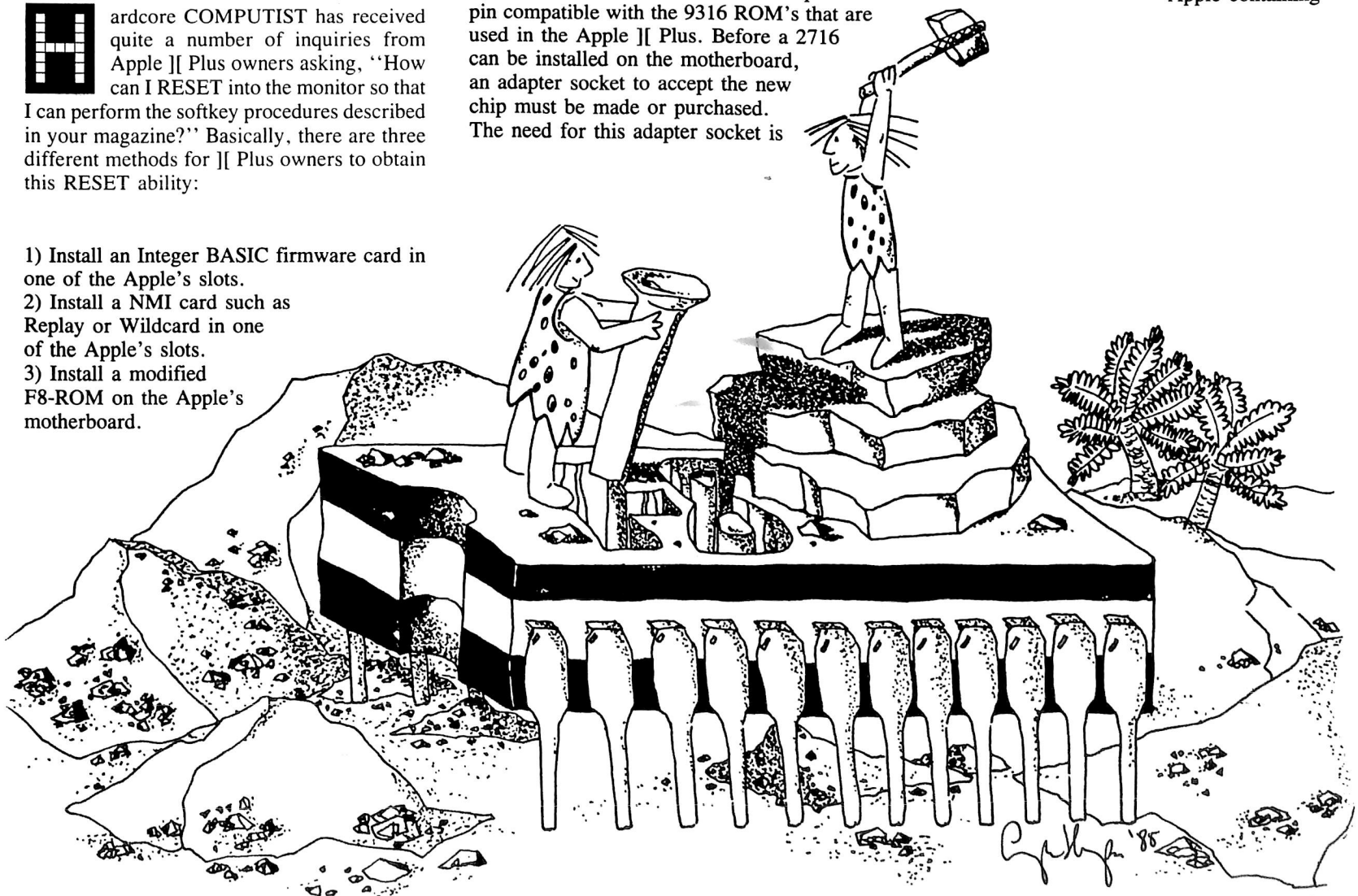
If it is only necessary for you to enter the Apple's monitor upon demand, options 1 and 2 from above are rather costly solutions (about \$100) to the RESET problem. A much more economical approach is to install a modified F8-ROM (in the form of a 2716 EPROM) on the Apple's motherboard. An article in Hardcore COMPUTIST No. 5 entitled, "Modified ROMS" by Ernie Young, detailed how to go about making this change. However, installing a 2716 EPROM in place of the autostart F8-ROM does have its drawbacks.

## The Problem

First of all, 2716 EPROM's are not pin-for-pin compatible with the 9316 ROM's that are used in the Apple ][ Plus. Before a 2716 can be installed on the motherboard, an adapter socket to accept the new chip must be made or purchased. The need for this adapter socket is

really more of an inconvenience than anything else.

A second problem which occurs when installing a modified F8 is caused by commercial programs that perform a "checksum" of the F8-ROM or check for the presence of certain bytes within the F8-ROM as they are booting. If these programs do not like what they see, the disk may just keep rebooting or some enlightening message to the user will be printed, ie. "UNABLE TO LOAD PRODOS". For example, the Apple ProDOS system file checks to make sure that it has been loaded into a genuine Apple containing



unaltered ROM's before it will proceed with its execution. With ProDOS, it is fairly easy to circumvent this check but, for protected programs, getting around this sort of thing can be more difficult.

One nice thing about having an Integer card in an Apple ][ Plus is that the switch to enable/disable the old monitor ROM can be flipped at nearly any time. A program can then be booted up with the autostart ROM enabled, and the old monitor ROM can be switched in when you want to perform the RESET. Wouldn't it be nice if there were some method of switching in a modified F8-ROM when it is needed without having to install a card in one of the Apple's slots?

### The Solution

After looking over the Apple's circuitboard schematics in, "The Apple ][ Circuit Description", I came up a method of accomplishing this very feat. My method involves replacing the F8 autostart ROM with a 2732, or 32K bit EPROM that contains an image of both the autostart and "modified" F8. A toggle switch is then installed so that the desired F8-ROM can be enabled. I must mention that this method is not for the faint-hearted because it involves soldering and removal and modification of the motherboard in addition to access to an EPROM burner.

Before I go into the details of installing a 2732 EPROM into your computer, allow me to refresh your memory as to why the F8-ROM is so important to the operation of your Apple.

### The Apple ][ ROM's

Back in the mid-70's when the Apple ][ computer was designed, the 16K bit or 2K byte ROM's used in it were the highest capacity memory devices that could be obtained in large quantities. The use of these chips allowed them to pack 12K of code into the six 24-pin ic's that can be found in row F of the motherboard of an Apple ][ Plus.

With this setup, one chip alone holds all the input-output routines of the computer. This chip is known as the F8-chip, and its eight pages of data are mapped into your computer from \$F800 through \$FFFF, inclusive.

To the 6502 microprocessor used in the Apple ][ series of computers, the top six bytes in memory (addresses \$FFFA-\$FFFF) are vectors or pointers to the addresses of the routines which respectively handle NMI's (Non-Maskable Interrupts), RESETs and IRQ's (Interrupt Requests). When the Apple ][ (Integer BASIC, only) was released, its RESET vector at \$FFFC-\$FFFD pointed to \$FF59 so that the monitor was entered whenever the RESET key was pressed.

With the release of the Apple ][ Plus, the "Autostart" F8-ROM was introduced to make the booting up of a disk drive automatic whenever the computer was turned on. This was accomplished by changing the RESET to point to \$FF6A where a routine checks to see if the computer has just been powered up. If a power-up has not occurred, an indirect JMP to the

routine pointed at by memory locations \$3F2 and \$3F3 is performed.

Establishing this second RESET vector on page \$3 makes it easy to have the computer perform just about any task, including rebooting or wiping memory, when the RESET key is pressed. Needless to say, most copy-protected software takes advantage of the page \$3 RESET vector. The advantage of placing a "modified" F8-ROM into an Apple ][ Plus is that the user can decide what happens when the RESET key is pressed.

### A New ROM

Basically, the pin-out of a 2716 EPROM and a 2732 EPROM are the same except that the 2732 has an extra address input. This extra address bit can be thought of as an input which selects either the upper or lower 16K bits of the chip. If this bit is a 0, the lower half of the EPROM is selected. If the bit is a 1, the upper half of the chip is enabled.

What I suggest is burning two F8-ROM images into a single 32K bit EPROM and then connecting a switch to the chip which will activate the image you desire (either autostart or modified F8). This method is not only inexpensive but it also solves the problem presented by disks which will not boot when a modified ROM is detected. You simply use the normal image when booting and switch to the other image if you want to perform a RESET.

The 24-pin ROM's on the ][ Plus's motherboard can be found in row F at positions F1 through F6. If you have some sort of Apple-clone, the ROM's are usually found just below the peripheral slots and the large horizontally positioned chip (the 6502 microprocessor). If your chips have a window on the top of them (usually covered by a sticker) then they are EPROM's (2716's) otherwise they are ROM's (generally 9316's). Note that if there are less than six of these chips, then your computer uses something other than 2716's and this procedure will not work.

### Creating the New Monitor

The first step in replacing the F8-ROM (usually the leftmost ROM) is to burn the EPROM to replace it. The EPROM that we are going to replace the F8-ROM with is a 2732, 32K bit EPROM. One-half of it is going to contain the autostart image and the other half is going to contain a nonautostart image. To create the data to burn into our new EPROM, follow these steps:

1) Move the autostart image into RAM

**2000<F800.FFFF**

2) Either BLOAD your modified F8 image (created using the steps in Hardcore COMPUTIST No. 5 or, better yet, the enhanced monitor described by Earl Taylor in his article on page 18 of this issue) at \$2800 or move the autostart image there and modify it.

**2800<F800.FFFF**  
**FFFC:59 FF**

3) Now save this dual image

**BSAVE NEW F8-ROM,A\$2000,L\$1000**

4) Burn this image into a blank 32K bit EPROM (2732). Make sure that if you are using a 2732A EPROM that you program it at 21, not 25 volts.

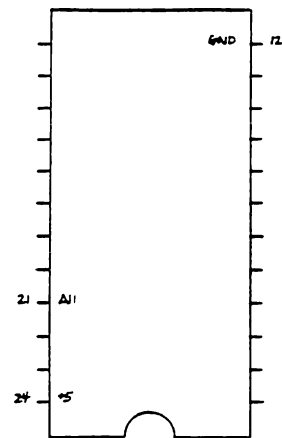
### Modifying the Chip

Before inserting this chip it must be modified so that either half of it can be externally selected. To do this you must solder one end of three wires to the chip and the other ends to a SPDT switch with no center position.

When soldering wires to the chip, try to avoid any prolonged contact with the soldering iron, as excess heat can damage integrated chips. Secondly, solder the wires to the very tops of the pins, right before they disappear into the chip housing.

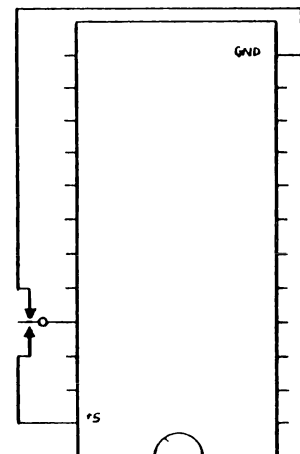
Now that you have been briefed on how to solder, you are ready to solder a wire to pins 12, 21 and 24. Here is a diagram that will help you locate these pins:

Diagram A



Next, solder the wire connected to pin 24 to the center contact of the SPDT switch and solder the other two wires to the edge contacts of the switch. When you have done this, the schematic of it should look something like this:

Diagram B



## Function of the Switch

When the switch is in position 1, pin 21 is connected to pin 12 (Ground) putting a zero on the uppermost bit of the address with the effect of enabling the lower 2k bytes of the chip. When the switch is in position 2, pin 21 is connected to pin 24 (+5V), pulling the highest address bit of the chip to a 1, thereby enabling the upper 2k bytes of the chip.

## Franklin Aces and Compatibles

It is easiest to perform this modification on the Franklin Aces (older versions only) or other compatibles that use 2716's because they are already wired to accommodate EPROM's instead of ROM's. All you have to do is bend pin 21 so that it isn't inserted into the socket when the chip is in position. After bending the pin, your chip should look similar to this illustration:

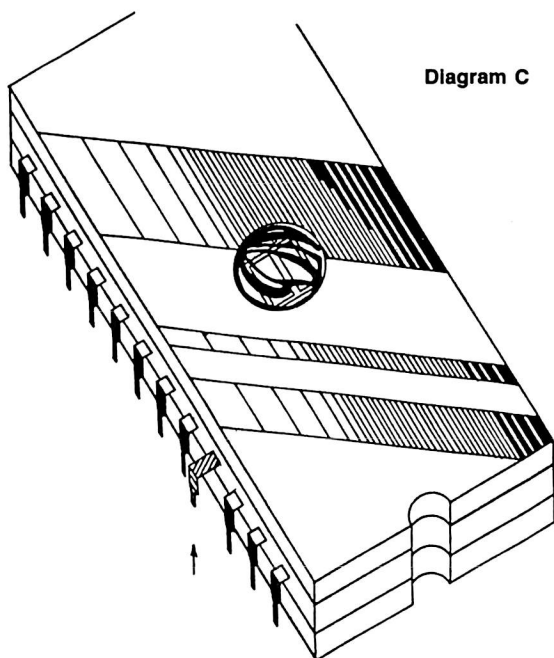


Diagram C

Your chip is now ready for operation. Just plug it in and enjoy.

## Apple II's

If you have a real Apple or some compatible that doesn't use EPROM's, the procedure isn't so straight forward.

In addition to bending pin 21 as explained in the above section, you must also invert the signal that is applied to pin 18 of the EPROM. This has to be done because your computer is wired to accommodate ROM's rather than EPROM's.

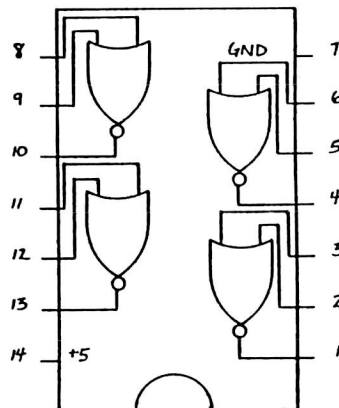
## Inversion

The quickest way to invert the signal on pin 18 is to use the patch area on your computer (at A-14 on Apples). This is an area on the motherboard where two additional ic's can be placed for revision changes or custom modifications.

Unless you have a revision 0 motherboard, there should be a socketed chip in this area that has some wires (as opposed to etched circuitry)

running to it. This chip should be a 74LS02 (quad two input NOR gate). Its pin-out looks like this:

Diagram D

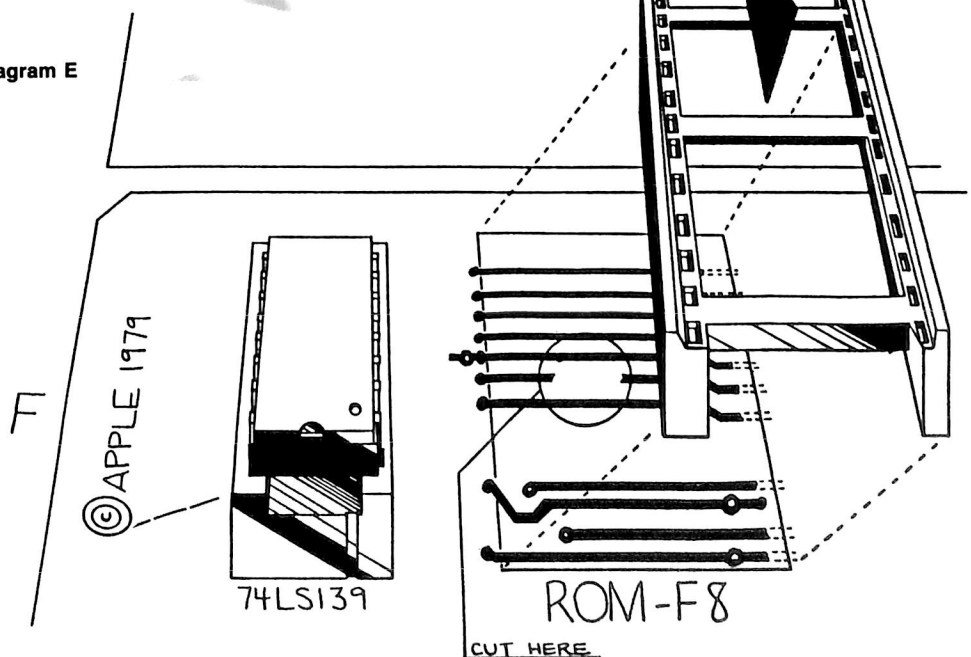


If you have a revision 0 motherboard, there probably are no chips in the patch area and you will have to wire in a 74LS02 of your own.

On my computer, the only free NOR gate on the 74LS02 was the one connected to pins 11-13 (Diagram D). I will assume that this gate is free on your computer also. By tying the two inputs of the NOR gate together, an inverter which can be used to fix the signal going to pin 18 of the EPROM can be created.

For this bit of trickery, you will need to remove your motherboard from your case. To do this, turn your computer upside down and remove the screws around the edges of the bottom. Next, remove the motherboard from the base of your computer by using pliers to pinch the small plastic posts (called standoffs) protruding through the motherboard. When they are pinched, you may nudge the board up a little and then do another one. Finally, with the motherboard removed, grab some small wire and remove your two leftmost ROM's (F0 and F8).

Diagram E



## Wiring the Inverter

- 1) Turn your motherboard upside down and *very carefully* solder a wire to pin 18 of both your F8-ROM socket and your F0-ROM socket. Remember that when the circuit board is upside down, the pin numbers are reversed.
- 2) Solder the wire that comes from the F0-ROM socket to pins 11 and 12 of the quad two-input NOR gate (74LS02) chip at A-14 (lower right hand corner) of your computer.
- 3) Solder the other wire to pin 13 of the quad two-input NOR gate chip.
- 4) Turn the motherboard back over and re-insert the F0-ROM.
- 5) Now you *must* cut the trace that goes from pin 18 of your F0-ROM to pin 18 of your F8-ROM. I cut it using an exacto blade right under the F8-ROM socket as illustrated in Diagram E below:

## Finalities

Finally, insert your PROM into the F8 socket, re-assemble your computer and test it out. Once you are satisfied that everything is working properly you will probably want to mount the switch in a convenient location and mark it so that you can tell at a glance which F8 image is enabled.

Although this procedure may take a lot of time if you own an Apple, it certainly costs less than a firmware or copy card and has a switch that can be conveniently mounted.

My thanks to C.R. Haight

## References

Gayler, Winston, *The Apple II Circuit Description*. Howard W. Sams. 1983.

Apple Computer Inc. *Apple II Reference Manual*. Part #A2L0001A. 1977,1981.



# The Games Of 1984: In Review -PART II

By Jeff Hurlburt

CORE

**T**his month we finish up our look at some of the games of 1984 with the following reviews of arcade and strategy games. Just as in the adventure/fantasy category, 1984 was a very good year for arcade game releases, even though it has become increasingly difficult to dream up fresh themes for new software.

A couple of the recently-released arcade games, namely Broderbund's Karateka and Electronic Arts' Skyfox, have set new standards against which future arcade games will be measured. Both Broderbund and Electronic Arts should be commended on the high quality of the majority of their recent releases. Hopefully, both of these software publishers will continue to prosper so that they never have the word "Chapter" associated with their names.

Many of the recent game releases are beginning to follow a trend: they require a minimum memory of 64K. As the installed base of Apple //c's and 128K Apple //e's increases, look for this trend to continue as software continues to take greater advantage

of the additional memory and double hi-res graphics.

As in last month's issue, each review begins with a "game type" descriptor (such as "Picture-Text Adventure") and the number of players for which the game is designed. Compatibility, hardware requirements, options, and recommended peripherals, if any, are described next. Lacking convenient access to an Apple //c, I was unable to test games for //c compatibility. Thus, only products so designated by the publisher are labeled as //c compatible in the reviews. Game list price and publisher phone number are also provided in the event that you are unable to find the game you want in a local software shop.

I've played each game listed in this review, and in many cases, watched others play it as well. The results of this observation appear towards the end of each review, following the game description.

Each game is rated in five areas and "overall" on a ten point scale: 10=

SUPERIOR, 07= GOOD, 05= FAIR, and 03= POOR. "NA" means "NOT APPLICABLE".

"Graphics" (GRFX) is the first area rated. It rates quality of artwork, clarity, impact, smoothness, speed, and realism. Good "Support Materials" (S.M.) include clear, thorough directions for play. In some cases attractiveness, tutorial value, or effectiveness in creating 'atmosphere' may also be important. "Playability" (PLAY) is determined by the amount of extraneous activity that the player must perform to play the game. Good parsing, rapid "save" and "restore" functions, efficient menus, smooth controls, and readily available "Help" screens are features which enhance playability.

"Difficulty" (DIFF) is self-explanatory for single-player games. For others, it evaluates the difficulty in reaching a fairly high level of play. High "Interest" (INTR) games effectively attract and hold the player's attention. Typically, these are the adventures you can't wait to continue and the arcades you play, and replay, for hours at a time. The "Overall" (GAME) rating amounts to a summary of player reaction(s) during tryouts. For educational games, a second rating (/ED) of educational value is included.

As I mentioned earlier, the overall quality of computer games released this past year has been the best ever and the evaluations reflect this generally favorable climate. Even products which fare poorly here will appeal to some readers, and many of the more superior games will undoubtedly soon rank among your favorites.

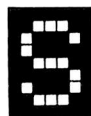


## Ape Escape (Arcade)

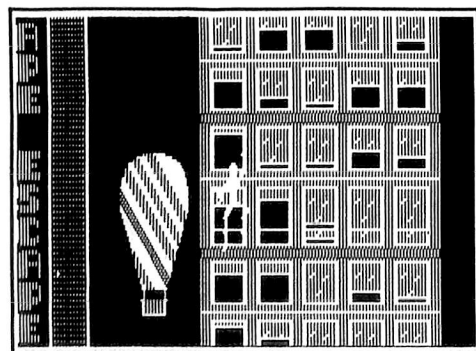
**Compatible:** 48K Apple ][+, //e

**Requirements:** One disk drive

KB input (Mockingboard Speech/ Sound I optional; will not boot if later model Mockingboard in slot 4)



Someone slipped up at the zoo and Harry the ape has escaped. So, let's go get 'em! Right? Not this time. In **Ape Escape** you are Harry, and in the middle of the big city the



only way out is up the side of the tallest building you can find.

**Ape Escape** demands a special sort of dexterity since both lateral movement and the left-right climbing motion is controlled, two-handed, from the keyboard. Nor is scaling a building merely a matter of avoiding closed (or closing) windows. Falling bowling balls, a net-dropping helicopter, the zookeeper on a painter's platform, and earthquakes quickly separate the apes from the monkeys. Fortunately, catching a hot air balloon or rescuing a damsel rewards you with a free ride for several floors. The higher you get and the more hazards you avoid, the higher your score (the top ten are saved).

Though this is one of those games that appears inane, it just feels good to play. Graphics work smoothly and it's fun to have such puppet-like control of a character.

Available from: H.W. Sams & Co., Inc., 4300 W. 62nd Street, Indianapolis, IN 46206, (800) 426-3696. Cost: \$19.95.

GRFX	S.M.	PLAY	DIFF	INTR	GAME
07	05	09	06	06	06

## Arcade Boot Camp (Arcade)

**Compatible:** 48K Apple ][+, //e, //c

**Requirements:** One disk drive

Joystick

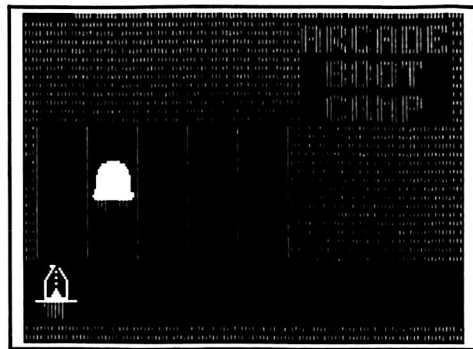


tired of feeding quarters to the local game room arcades and never getting a free game or seeing your name in lights? Well maybe it's time you signed up for the **Arcade Boot Camp!**

In what amounts to a cleverly integrated collection of mini arcades, boot camp training includes copter flying, jeep racing, and bomb dodging, to name but a few areas of practical concern. Success at any level is rewarded by advanced rating (up to five players' ranks may be saved) and the chance to tackle more demanding tasks. By the time you reach Sergeant Major, you'll be ready to take on any machine in the game room.

**Boot Camp's** sequences are generally too brief or too simple to satisfy most veterans, except possibly in the context of a

multi player tournament. Mainly for the beginner, this is a cute, nicely crafted piece of software which does, in fact, sharpen arcade expertise.



Available from: Penguin Software, 830 Fourth Avenue, PO Box 311, Geneva, IL 60134, (312) 232-1984. Cost: \$29.95.

GRFX	S.M.	PLAY	DIFF	INTR	GAME
08	07	08	06	06	07

### Bruce Lee (Arcade for 1 or 2 players)

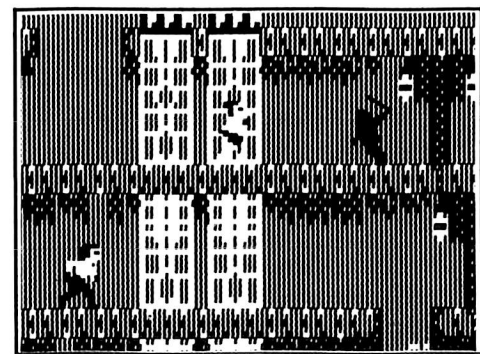
Compatible: 48K Apple ][+, //e, //c  
Requirements: One disk drive  
Joystick



ntold wealth and the secret of immortality lure you, Bruce Lee, to challenge the perils of the evil wizard's sprawling fortress. You begin your quest in the outer courtyard with passage from one setting to the next obtained by collecting enough lanterns to trigger the opening of secret panels. Deadly ninjas and yamos put your mastery of the martial arts to a stern test, while exotic traps require that you command super keen awareness.

**Bruce Lee** features fast, lifelike action in interesting, attractive settings. Punching, kicking, leaping, etc. motions are under quick responding joystick control- a necessity since you have only five lives (when two play, turns change on the loss of a life). Scoring is based mainly on lanterns taken, guardians defeated, and level of penetration.

Though high scores are not saved (too



bad), **Bruce Lee** is a good entertainment value; a real toughie, yet fair and fun to play.

Available from: Data Soft, Inc., 19808 Nordoff, Chatsworth, CA 91311, (818) 701-5161. Cost: \$34.95.

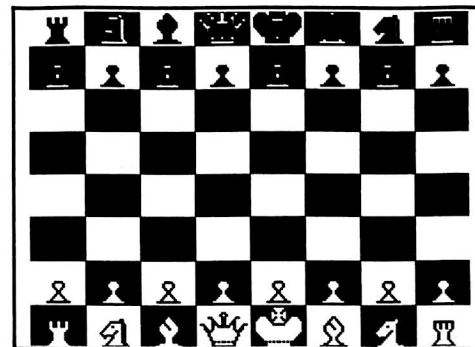
GRFX	S.M.	PLAY	DIFF	INTR	GAME
08	05	08	09	07	07

### How About A Nice Game Of Chess? (Chess Tutor)

Compatible: 48K Apple ][+, //e, //c  
Requirements: One disk drive  
KB or Joystick



desta's, **How About a Nice Game of Chess?** is intended as a tutorial package for players ages 8 through adult. Thus, besides a decent chess playing program, you get a manual and six chapters of on-disk instruction covering rules, piece movement, and major phases of the game. 'Nice' Chess also incorporates several valuable in-play assistance and analysis aids but unfortunately doesn't allow for saving games.



Although there may be some more advanced and better-playing programs (e.g. Sargon and Mychess) which provide some beginner oriented aids, 'Nice' Chess is decidedly ahead when it comes to getting the non-player started. Instead of having to enter moves in algebraic notation, pieces are 'picked up' and moved using a cursor. The tutorials appear to work well and, of course, it says "beginner" right on the box (a non-player, overwhelmed by Sargon, could easily pick up and use the Odesta package.)

Featuring an easy-to-use boardside menu, this program can be utilized with a minimum of hassle. For many would-be wood-pushers, **How About a Nice Game of Chess?** is a worthy introduction to the game of kings.

Available from: Odesta, 3186 Doolittle Drive, Northbrook, IL 60062, (312) 498-5615. Cost: \$34.95.

GRFX	S.M.	PLAY	DIFF	INTR	GAME
07	07	08	4-7	08	07

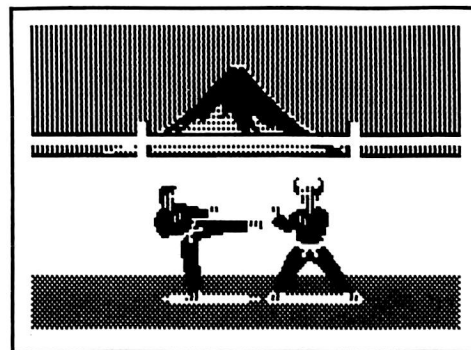
### Karateka (Arcade)

Compatible: 48K Apple ][+, //e  
Requirements: One disk drive  
Joystick



kuma, the evil warlord, has seized the beautiful Princess Mariko and is holding her prisoner in his mountain bastion. As a loyal karateka (a kind of martial arts samurai) it falls to you to rescue Mariko. That this means facing the warlord's skilled minions, his trained hawk, and even Akuma himself, and that you will almost certainly perish in the attempt-- all of this matters not. For such is the way of the karateka.

**Karateka** is the most recent and best illustration of a trend towards computer games that look like movies. Featuring superb artwork and animation as well as superior sound effects, the game opens with a motion-picture-like scenario setter. Between bouts (assuming you win) there are even scene switches to Akuma ordering yet another warrior to combat or the princess languishing in her cell.



As the Karateka, you engage successively more skilled adversaries, moving along walkways and through archways towards the heart of the fortress. Punching, kicking, running and other motions are under reasonably tight control and two sets of counters show the number of hits remaining for you and your opponent. Each victory restores some counters as does resting; however, the number of warriors you must face depends on how much ground you cover between fights.

Simply put, **Karateka** is a breakthrough. Art merges with technology to produce a game almost as much fun to watch as to play.

Available from: Broderbund, 17 Paul Drive, San Rafael, CA 94903, (415) 479-1170. Cost: \$39.95.

GRFX	S.M.	PLAY	DIFF	INTR	GAME
10	05	09	07	07	09

### Lost Tomb (Arcade for 1 or 2 players)

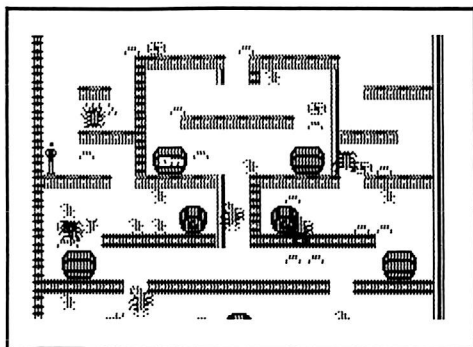
Compatible: 48K Apple ][+, //e  
Requirements: One disk drive  
Joystick



apidly paced, even for an arcade, **Lost Tomb** features 13 mazes utterly crammed with unfriendly mummies, bats, scorpions, and

spiders. The format is reminiscent of Lady Tut except that walls don't flip and figures are not as well defined. Besides bullets, you also have whips that can wipe out nearby monsters and walls (an advantage amply balanced by deadly earthquakes which can be stopped only by opening a treasure chest).

Ultimately, of course, your aim is to plunder the innermost tomb chamber. Short of this, there's the satisfaction of outscoring a competitor in a two-person game. Some points are awarded for each kill, but chests yield the most as well as supply weapons. High scores, unfortunately, are not saved.



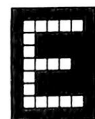
Nominally a maze arcade, **Lost Tomb** allows far too little time for assessment and planning. I was soon frustrated by poor control response and bored by the unrelieved sameness of action.

Available from: Data Soft, Inc., 19808 Nordoff, Chatsworth, CA 91311, (818) 701-5161. Cost: \$29.95.

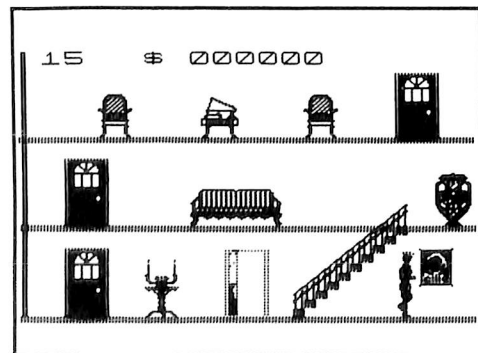
GRFX	S.M.	PLAY	DIFF	INTR	GAME
05	05	05	08	03	04

### Mabel's Mansion (Extended Arcade)

Compatible: 48K Apple II+, IIe, IIc  
Requirements: One disk drive  
KB input



eccentric Aunt Mabel has passed away leaving you her entire estate, if you can but prove your mettle by collecting all of the treasures hidden in her 90-room puzzle box mansion. With Aztec-like movement and manipulation



of small objects, you obtain treasures and keys by pushing, opening, unlocking, etc. the

correct item (usually an ornament or piece of furniture) in the correct order. The same goes for the weapons you will need to fend off the assorted critters which try to munch away your "life points". Zero life points means you're dead and the game is over.

**Mabel's Mansion** is only moderately challenging but it is fun to explore such a colorful place. The game's major weakness is an emphasis on randomness and luck, with little real opportunity for problem-solving.

Available from: Datamost, Inc., 8943 Fullbright Avenue, Chatsworth, CA 91311, (818) 709-1202. Cost: \$29.95.

GRFX	S.M.	PLAY	DIFF	INTR	GAME
07	05	08	06	06	06

### MicroLeague Baseball (Arcade for one or two players)

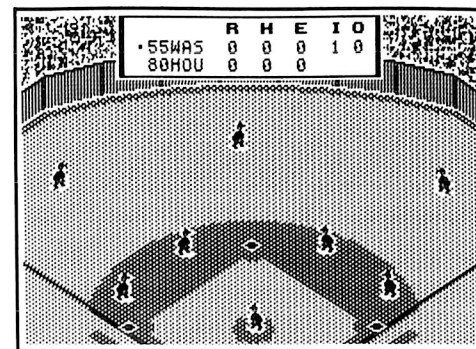
Compatible: 48K Apple II+, IIe  
Requirements: One disk drive  
KB input



inally you can boot out the bum who's been mismanaging your favorite team and do things right.

**MicroLeague Baseball** is a non-arcade strategic simulation which determines outcomes on the basis of probabilities using player statistics. With **MLB** you may take on another player, play the computer, manage both teams, or have the computer play both sides. The better you know your team and the greater your baseball savvy, the more likely you are to chalk up a win.

The **MLB** game disk includes a selection of 25 solid teams spanning several decades from the 1927 Yankees through the 1984 All-Star teams. Team Disks, purchased separately, include world series teams, all star rosters, and all 26 NL and AL teams for the past three seasons. Team Disks allow player trade and game save options not otherwise available.



Once teams are chosen, you need to select a starting pitcher and set the lineup. On-screen stats make it unnecessary to refer to booklets or the newspaper. Any legal in-play changes (such as changing pitchers, pinch hitting, etc.) are also allowed in **MLB**. Each at bat normally involves selection of one offensive and one defensive play. The ball

is thrown and the results (strikeout, walk, base hit, pop fly, etc.) are played out. Quick, smooth graphics approach TV broadcast realism.

Though what I know about baseball would fit comfortably on the back of the box, I've enjoyed **MLB**. More impressive, a couple of real fans regularly spend hours playing the game. Described as "amazingly accurate", without qualification the local 'experts' proclaim **MLB** the best of computer baseball.

Available from: MicroLeague Sports, 28 East Cleveland Ave., Newark, DE 19711, (800) PLA-YBAL. Cost: \$39.95 (Team Disks: \$19.95).

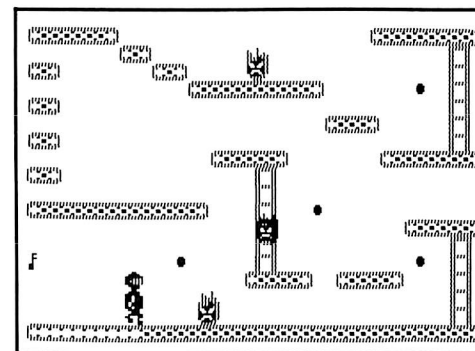
GRFX	S.M.	PLAY	DIFF	INTR	GAME
08	08	08	05	08	08

### Mr. Robot And His Robot Factory (Extended Arcade)

Compatible: 48K Apple II+, IIe, IIc  
Requirements: One disk drive  
KB or Joystick



r. Robot is the best yet in a long line of chutes & ladders/ power pill arcades. Besides treadmills, ladders, and escalators, there are poles to slide down, transporters to zip you through the air, trampolines to bounce on, and magnets to give you a boost (unless you get too close!)



Your objective on each of 22 levels is to maximize your score by collecting power pills, igniting bombs, and using energizers to zap the otherwise deadly Alien Fires. Not especially original in concept, **Mr. Robot** is distinguished by superior execution resulting in tight control and natural (for a robot) movement.

Almost as much fun as the actual play of the game is the Robot Factory where you can design and test (ala Pinball Construction Set) up to 26 of your own screens which can be saved and played just like the original game.

Available from: Datamost, Inc., 8943 Fullbright Avenue, Chatsworth, CA 91311, (818) 709-1202. Cost: \$34.95.

GRFX	S.M.	PLAY	DIFF	INTR	GAME
08	06	09	06	08	08



## Mychess II (Chess Player for 1 or 2 players)

**Compatible:** 48K Apple ][+, //e, //c  
**Requirements:** One disk drive  
 KB input

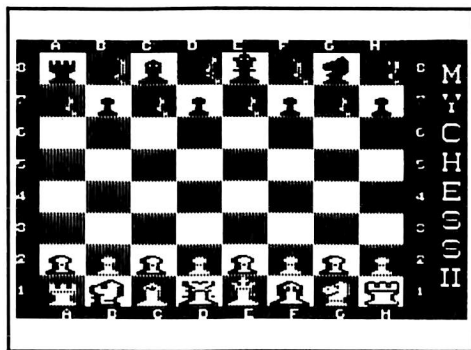
## Sargon III (Chess Player for 1 or 2 players)

**Compatible:** 48K Apple ][+, //e  
**Requirements:** One disk drive  
 KB input (SpeeDemon optional)

In just the past few months, several new programs have considerably advanced the quality of computer competition for Apple wood-pushers. Representative of the best are Hayden's, **Sargon III** and Datamost's, **Mychess II**.

Both feature multiple, time-controlled playing levels with choice of color and options to take back moves or switch sides in mid-game. Modes for setting up and analyzing positions, solving problems (**Mychess** adds a special high-speed "solve mate" mode), and game replay are also included.

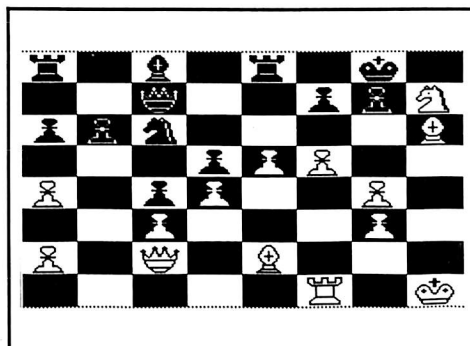
**Sargon's**, "Window on the Search" offers by far the better in-play analysis aid, showing **Sargon's** current "best line", depth of search, and the moves being considered. There is even a score reflecting **Sargon's** current evaluation of its position. Both programs permit saving and loading games as well as listing games and positions to a printer. The hires board displays are attractive, with **Sargon's** B&W getting the nod for clarity and **Mychess's** color (you get a choice of three piece sets) being prettier. The **Mychess** 3-D option with front or side view is also very impressive.



Each package offers a good selection of "great games" which may be loaded and replayed. **Sargon** adds 45 problems as well, whereas **Mychess** includes several recent computer chess contests. Nicely done booklets provide aids for beginners and detailed explanations of features. **Sargon's** is better organized with a much fuller description of program operation; however,

**Mychess** offers a very handy in-program "Help" screen.

I conducted a simple experiment to compare the relative playing strength of the games by pitting the two against each other (I used two computers and a chess clock with each program set at two minutes per move. **Sargon** opened P-K4 and **Mychess** replied P-Q4 (the Center Counter Gambit). Nearly three hours later, **Mychess** mated **Sargon** on move No. 43. Each program used nearly identical amounts of time per move, averaging almost exactly two minutes. Having played both, I would safely assume that, over several games, **Sargon** and **Mychess** would score about equal in wins.



Both programs devoured an assortment of "suggest a move" and "find the mate" problems and, playing at tournament speed, they have performed admirably against their human counterparts who hold USCF ratings ranging from "C" through "A". Capable of maintaining a "B" (or perhaps low "A") rating, **Mychess II** and **Sargon III** are good, solid, occasionally brilliant adversaries guaranteed to sharpen your game.

*Mychess available from: Datamost, Inc., 8943 Fullbright Avenue, Chatsworth, CA 91311, (818) 709-1202. Cost: \$39.95. Sargon available from: Hayden Software, 600 Suffolk Street, Lowell, MA 01854, (201) 393-6000. Cost: \$49.95.*

	GRFX	S.M.	PLAY	DIFF	INTR	GAME
S III :	07	08	08	5-9	08	09
M II :	08	07	08	5-9	08	09

## Regatta (Arcade for 1 or 2 players)

**Compatible:** 48K Apple ][+, //e  
**Requirements:** One disk drive  
 KB or KB and Paddle

Recognizing that even the toughest arcade warrior needs a bit of R & R between missions, H.W. Sams suggests a day or so of sail racing on a peaceful lake. **Regatta** puts you at the tiller of a one-man boat with the option of racing the clock and/or another player around one of four courses (the game saves course records.)

This is a real-time simulation requiring quick response to wind changes while avoiding such hazards as running aground,

hitting a marker buoy or capsizing. A set of indicators makes adjusting sail angle and tiller a fairly simple process, even for a 'lubber', though proficiency comes only with practice.

**Regatta's** realization is approximately state of the art circa 1981. The graphics are unspectacular and, though a joystick could easily handle all control functions, no such option exists. Even so, the game is still fun to play. With so many variables, no two races are alike and there's always the lure of running the wind to that one unbeatable record.

*Available from: H.W. Sams & Co., Inc., 4300 W. 62nd Street, Indianapolis, IN 46206, (800) 426-3696. Cost: \$29.95.*

	GRFX	S.M.	PLAY	DIFF	INTR	GAME
	05	07	07	07	07	06

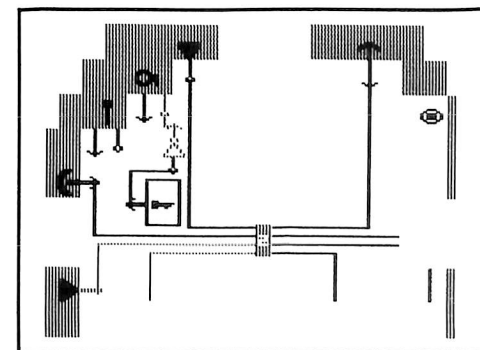
## Robot Odyssey I (Adventure/Digital Logic Tutorial)

**Compatible:** 64K Apple ][+, //e, //c  
**Requirements:** One disk drive  
 Color monitor  
 KB or Joystick



**Robot Odyssey** is an educational package designed for ages 13 to adult. Its main purpose is to teach principles of logic design. Beginning with several Rocky's Boots-type tutorials, you learn about logic gates, sensors, and other robot control components. In the Innovation Lab you can try your hand at some 'real world' applications and even design, test, and "burn-in" your own control IC's which may be saved for future use.

A robot, it turns out, is a nifty little mechanism complete with vision and touch sensors, thrusters for propulsion, an antenna, a "grabber", and a power source. You can move inside a robot and, using your toolkit, wire it to perform all sorts of interesting tasks. Keys, tokens, and even other robots can be moved into a robot.



The game phase is called Robotropolis. Robotropolis is not such a bad place except that no one lives here except robots and a few unfortunate creatures who, like yourself, got here by mistake and would just as soon be elsewhere. Escaping entails the creative

implementation of your three robots to navigate mazes, slip through barriers, and solve other problems.

Each of five levels, from the "City Sewers" through the "Skyways", presents special challenges. Fortunately, wherever you may be in Robotropolis, the resources of the Innovation Lab are but a bolt's throw away.

An educational masterpiece, the package does a good job of communicating some fairly sophisticated content. As the title implies, **Robot Odyssey** is a multi-session venture you will have to seriously 'get into' to even get started in Robotropolis. However clever, I doubt that an adventure with such complex mechanics will have much leisure appeal.

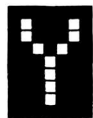
Available from: *The Learning Company, 545 Middlefield Road, Ste. 170, Menlo Park, CA 94025, (415) 328-5410. Cost: \$49.95.*

GRFX	S.M.	PLAY	DIFF	INTR	GAME/ED
08	08	05	09	07	07/09

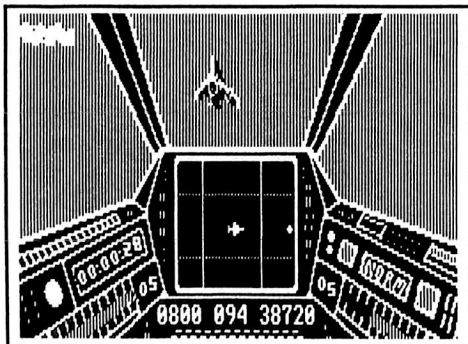
### Skyfox (Arcade)

**Compatible:** 48K Apple ][+, //e

**Requirements:** One disk drive  
KB or Joystick with KB (Mockingboard optional but essential for best results)



You are all that stands between enemy forces and the destruction of Federation bases, you and **Skyfox**. Skyfox is a Mach IV fighter that can turn on a dime while raking the enemy with a deadly barrage of laser cannon fire, radar-guided missiles, and heat seekers. This plus shields and a sophisticated electronics package just about evens the odds against hordes of attacking tanks, fighters, and motherships.



Learning to handle this high-performance craft requires practice and you are provided with five levels of difficulty (cadet through ace). In addition, five of the game scenarios are training missions. In the eight combat scenarios (i.e. the Real Thing) you must face invasion forces ranging from small probes to massive onslaughts in a variety of configurations. Your mission: knock out enemy units before they can reach friendly

installations and blast the motherships before they launch so many attackers that not even Skyfox can stop them.

After selecting level and scenario, you'll find yourself seated in the Skyfox cockpit watching the base computer feed with your clock counting off elapsed scenario time. The invasion is on! Even now enemy tanks and planes are moving against friendly installations. A button press reveals the high-tech instrument cluster and a forward view down the launch tube (if your Mockingboard is hooked up to an external amp, check to make sure the volume is set to a sane level). A second press and, ala Battlestar Galactica, Skyfox hurtles forth to meet the challenge.

Swooping down on advancing tanks or dueling aggressor fighters, full color high-speed graphics and spectacular sound put you THERE. A new prototype in long-play combat strategy and tactics, **Skyfox** makes anything else look like an arcade game.

Available from: *Electronic Arts, 2755 Campus Drive, San Mateo, CA 94403, (415) 571-7171. Cost: \$40.00.*

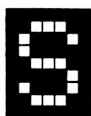
GRFX	S.M.	PLAY	DIFF	INTR	GAME
09	07	08	08	09*	10*

\*with Mockingboard sound (otherwise, delete 3 points)

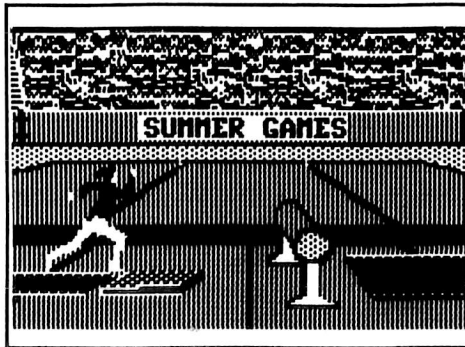
### Summer Games (Arcade for 1 to 8 players)

**Compatible:** 64K Apple ][+, //e

**Requirements:** One disk drive  
KB or Joystick



**Summer Games** features Olympic style competition in diving, gymnastics, pole vault, and skeet shoot as well as two running and two swimming events. Gold, silver, and bronze medals are awarded and, at the end of the games, the player with the most medal points is honored as Grand Champion. There's also an opportunity to establish a "world record" (saved on disk) in each event.



Epyx goes the distance to make participation in **Summer Games** a realistic experience. Doves fly by during opening

ceremonies and your country's anthem accompanies the award of a gold medal (you can represent any one of eighteen countries). Animation is smooth and contestant figures are even large enough to show detail.

In tryouts, players awarded high marks to gymnastics and diving, and the skeet shoot proved a worthwhile change of pace. Otherwise, there was a boring sameness and limitation of control and, too often, **Summer Games** left me feeling like a spectator not really in charge of my own performance.

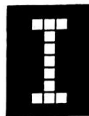
Available from: *EPYX, 1043 Kiel Ct., Sunnyvale, CA 94089, (408) 745-0700. Cost: \$40.00.*

GRFX	S.M.	PLAY	DIFF	INTR	GAME
08	07	08	07	06	06

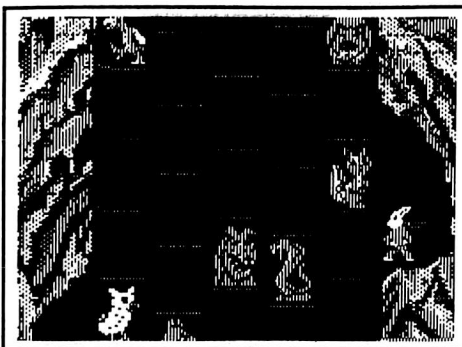
### Super Bunny (Arcade)

**Compatible:** 48K Apple ][+, //e, //c

**Requirements:** One disk drive  
KB or Joystick



In **Super Bunny** you are Reginald Rabbit, out to even the score against the wolves, snakes, and other predators terrorizing your clan. To accomplish this you must hop your way across five moving columns of predators and eat the Power Carrots which transform you into Super Bunny. Then, before the effects wane, you must punch out as many baddies as you can. Get them all and you advance to the next of six levels. Miss a hop, or land on a predator as Reginald and one of your five lives is gone. Success at level six is rewarded by a spectacular display plus the opportunity to recycle and enhance your score (also your bragging rights, since the top ten scores are saved.)



**Super Bunny** is an excellent blend of smooth graphics, sound effects, and humor (you even get an Origins of Super Bunny comic book)! Finely crafted and highly addictive, this one ranks among the best.

Available from: *Datamost, Inc., 8943 Fullbright Avenue, Chatsworth, CA 91311, (818) 709-1202. Cost: \$39.95.*

GRFX	S.M.	PLAY	DIFF	INTR	GAME
08	07	09	07	09	09

## T-Rex (Educ. Arcade/Simulation)

**Compatible:** 48K Apple ][+, //e, //c  
**Requirements:** One disk drive  
 KB or Joystick (color monitor recommended)



-Rex is designed to help players (ages 8 and up) better understand the habitat as well as the strengths and limitations of dinosaurs. In the game, your objective is to keep T-Rex (Tyrannosaurus rex, that is) alive through four increasingly difficult levels of play.



Contrary to the image developed over years of "Lost World" type movies, you do not simply walk around plucking prey at will. Small animals are often too quick and larger creatures come with tank-like armor and/or horns the size of small trees. Either way, you've got to expend energy and risk your food reserves to accomplish anything. Chases, especially through swamps, gobble energy and you can't afford to ignore such hazards as quicksand and tar pits. Roaming country which is too hot or too cold requires

careful management of body heat. The longer you last, the more experience points you pile up and the easier it is to survive. You can save promising dinosaurs between sessions.

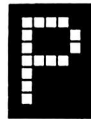
Marginal graphics technique and low action level make **T-Rex**, at best, a mediocre arcade. However, the module's game, tutorials, and nicely done text make an attractive, relatively painless way to learn some significant facts about dinosaurs.

*Available from: CBS Software, One Fawcett Place  
 Greenwich, CT 06836, (203) 622-2525. Cost: \$49.95.*

GRFX	S.M.	PLAY	DIFF	INTR	GAME/ED
06	09	07	07	06	06/08

## Word Challenge (Word Game)

**Compatible:** 48K Apple ][+, //e  
**Requirements:** One disk drive  
 KB input

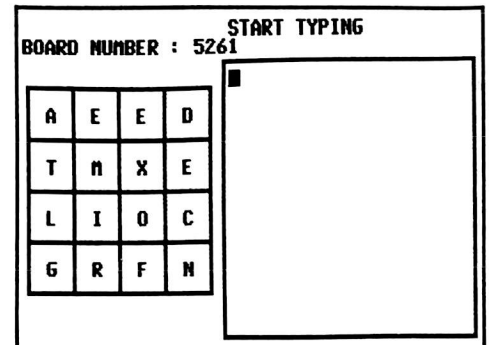


robably because they tend to require pencil and paper, a dictionary, or other special supplies, word games have always been under-represented in computer play. In contrast, Hayden's, **Word Challenge** needs only your Apple and employs advanced packing techniques to maintain a 90,000 word 'dictionary' on-disk. Chiefly a test of vocabulary, the object is to score points by extracting and entering as many words as possible from a given letter square.

Squares range from 3x3 to 5x5 and only words formed from adjacent letters count. So, for a 3x3 square, the longest possible word has nine letters. Long words, of course, count more than short ones. The standard

game uses a 4x4 square and three minute time limit.

In **Word Challenge** your opponent is LEX, a word finding routine with 26 user-selectable skill levels. At the end of each game the program figures both your scores and those of LEX and updates a 'match score' box on the screen. The first to reach 100 points wins the match. Practically every game parameter including square size, winning match score, time limit, and scoring criteria, is open to modification.



This is a well designed package, simple in concept yet able to accommodate players of varied scholarship. Boasting a repertoire of nearly 200,000 different squares, **Word Challenge** is an excellent entertainment value.

*Available from: Hayden Software, 600 Suffolk Street,  
 Lowell, MA 01854, (201) 393-6000. Cost: \$49.95.*

GRFX	S.M.	PLAY	DIFF	INTR	GAME
05	07	08	6-9	08	08

## ADVENTURE TIPS

### \* Infidel Infocom

You must use something as a counter-weight in the Statue Room.

At the panel, 3 odd numbers will work. Be careful not to set the torch on the floor in the Barge.

Once you find the pyramid, you may discard the shovel, but not the axe.

Unlock the statues according to the hieroglyphics found in the Rooms of the Gods.

### \* Zork I Infocom

So, you haven't got a prayer? Try it anyway. Careful with the torch as it will vaporize your candles.

Remember the story of the Greek hero and the Cyclops?

Don't bother the skeleton.  
 "Echo" for the bar.

### \* Zork III Infocom

Touch the table, and ZAP!!! You're somewhere else.

776 sounds like a good time.

Greet the sailor with the **CORRECT** words, and he will give you a present.

Under the seat is a good place to keep things. Careful not to kill your "friend" at the cliff.

All you're after is the staff.

The mysterious figure cannot keep you from taking his hood if he is badly wounded.

### \* Savage Island Part 1 Adventure International

Coconuts are useful things with which to move to an overhead lever.

You need some salt to cure the bear.

Keep the bandanna with you all the time.

\* Contributed by The Sinexus.

### Colossal Caves Adventure International

You'll need a green thumb in the two pit room. Or, just a bottle of water or two will do.

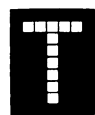
Can't get past the alcove? You have too many possessions.

Nasty troll, huh? Throw some eggs.



# Towards A Better F8 ROM

By Earl Taylor



The Modified F8-ROM articles in Hardcore COMPUTIST Issues No. 6 and 8 presented a useful change to the RESET handling code but, after reading these, I began to wonder what other changes might be made to improve the ROM's usefulness without making it incompatible with most software. My objective was to produce a ROM which could stay in its socket 90% of the time and which solved most of the problems I experienced using the normal F8. The changes I present here are a step towards this goal. I hope that they will encourage you to examine the F8 and lend your efforts, comments and suggestions to the task.

## Features

- Blinking underline cursor** is easy on the eyes and eliminates those strange characters that show up when moving over lowercase characters.
- No automatic cancel** when typing in long input lines.
- Improved disassembler** allows continuous disassembly a screen at a time or a line at a time.
- Three-way RESET** lets you either enter the monitor, move low pages of memory to higher addresses and enter the monitor or do the normal reset action.

**Non-Maskable Interrupt Handler** saves the contents of A, X, Y, and S, moves low pages of memory into higher addresses and enters the monitor.

**Binary Search** with wildcard lets you scan a range of memory for any byte pattern you choose.

## Making the ROM

As described in the article, "Modified ROM's" (Hardcore COMPUTIST No. 6), you require access to an EPROM programmer (and probably an eraser), some blank 2716-type (single voltage) EPROM's, and an adapter socket or modified motherboard in order to make and use the new F8-ROM described below. Modifying the motherboard so that the signal going to pin 18 of the 2716 will be inverted (as opposed to continually held high) is really the best way to go in order to avoid problems with such things as CP/M cards. This modification is described under the section, **Apple II's and Wiring the Inverter** in Ray Darrah's article on page 9 of this issue.

Please refer to Issue No. 6 or The Best of Hardcore Computing for details on using EPROM's in an Apple.

**IMPORTANT:** The changes which follow are for the Autostart F8-ROM only.

- 1) With your normal Autostart F8-ROM in your Apple, boot a slave diskette and enter the monitor

**CALL -151**

- 2) Move a copy of the Autostart into RAM  
**2800<F800.FFFF**

- 3) Take care when typing in the following modifications to the relocated F8-ROM

2A6F: 4C \$B8C9  
2A70: 7B FA 8D 87 D7 AF D2 A0 \$CDD0  
2A78: CF CE A0 \$2724

2B09: 38 49 C8 29 DF F0 50 \$ECF6  
2B10: 20 2C FC 20 0C FD 4C 09 \$3E70  
2B18: FB \$F7C3

2B60: 4C 58 FC 8D 00 29 8E 01 \$9E75  
2B68: 29 8C 02 29 4C DE FC \$2320

2B97: A0 \$A547  
2B98: 00 B9 02 02 CD 00 02 F0 \$74B1  
2BA0: 04 D1 3C D0 13 C8 CC 01 \$83EA  
2BA8: 02 90 EE 20 80 FE 20 1A \$37F2  
2BB0: FF A2 00 20 C8 FE F0 05 \$3903  
2BB8: 20 BA FC 90 DA 4C 2F FB \$FCEE  
2BC0: EA \$FC09

2C2C: 69 FD 90 EA \$E738  
2C30: F0 DE 69 FD 90 BE F0 04 \$DC65  
2C38: 69 FD 90 2A 68 68 60 EA \$65C8  
2C40: EA EA \$AA1D

2CC9: AD 00 C0 10 FB 2C 10 \$665A  
2CD0: C0 29 DF C9 D3 F0 07 C9 \$C62D  
2CD8: CD F0 2D 4C 62 FA BA 8E \$52CF  
2CE0: 03 29 A0 00 B9 00 00 99 \$DF51  
2CE8: 00 20 B9 00 01 99 00 21 \$9DD7

## New ROM Source Code

0000-	ZEROPG	.EQ 0	Zero page
0022-	WNDTOP	.EQ \$22	Window top
0024-	CH	.EQ \$24	Cursor horizontal position
0024-	TEMP	.EQ \$24	Temporary storage. Note: same as CH (cursor horizontal)
0028-	BASL	.EQ \$28	Base address of current screen line
0038-	KSWL	.EQ \$38	Character Input SwitchLow
003A-	PCL	.EQ \$3A	Program counter pointer for Disassembler
003B-	PCH	.EQ \$3B	Program counter pointer for Disassembler
003C-	A1L	.EQ \$3C	Parm A1 low byte- matching location from SEARCH routine
003D-	A1H	.EQ \$3D	Parm A1 high byte- matching location from SEARCH routine
003E-	A2L	.EQ \$3E	Memmove endLow
003F-	A2H	.EQ \$3F	Memmove endHi
0042-	A4L	.EQ \$42	Memmove destLow
0043-	A4H	.EQ \$43	Memmove destHi
004E-	RNDL	.EQ \$4E	Low byte of "random" counter
004F-	RNDH	.EQ \$4F	Hi byte of "random" counter
0100-	STAKPG	.EQ \$0100	Stack page start
0200-	WILDCD	.EQ \$0200	Users choice of wildcard
0201-	STRGLN	.EQ WILDCD+1	Length of binary string to match
0202-	STRING	.EQ STRGLN+1	
03F8-	USRADR	.EQ \$03F8	Ctrl-Y vector
2000-	SAVZPG	.EQ \$2000	Save page for zero page
2100-	SAVSTK	.EQ \$2100	Save page for stack
2900-	SAVEA	.EQ \$2900	Save registers starting with A here
2901-	SAVEX	.EQ SAVEA+1	X goes here
2902-	SAVEY	.EQ SAVEA+2	and Y goes here
2903-	SAVES	.EQ \$2903	Stack pointer save
C000-	KBD	.EQ \$C000	Keyboard data

Continued on next page

2CF0: C8 D0 F1 84 3C 84 3E 84 \$C646  
 2CF8: 42 A9 02 85 3D A9 09 85 \$3EE8  
 2D00: 3F A9 22 85 43 20 2C FE \$BDE1  
 2D08: 4C 59 FF EA A4 24 B1 28 \$A56A  
 2D10: 6C 38 00 20 3A FF 4C 81 \$3823

2D18: FD EA EA 20 FD FE 2C 00 \$BBA7  
 2D20: C0 10 F8 91 28 84 24 EA \$B63C  
 2D28: AD 00 C0 2C 10 C0 60 20 \$9493  
 2D30: 0C FD 20 09 FB 20 0C FD \$5CEC  
 2D38: C9 9B F0 F3 60 \$6786

2D58: E0 FE 90 03 4C 13 FD E8 \$A8E9

2D7E: 4C 84 \$665F  
 2D80: FD E8 A9 88 \$26AF

2E74: AA \$6CA1

2EC2: 60 EA EA 4C F8 03 \$4A4C  
 2EC8: 20 5E FE A9 01 20 63 FE \$0988  
 2ED0: AD 00 C0 10 FB 2C 10 C0 \$9662  
 2ED8: C9 95 F0 EC C9 A0 F0 EB \$1855  
 2EE0: C9 9B 60 A9 02 85 22 4C \$7983  
 2EE8: 97 FB A0 08 B9 72 FA 20 \$1ECB  
 2EF0: ED FD 88 10 F7 60 20 00 \$527B  
 2EF8: FE 68 68 D0 6C 48 E6 4E \$C69A  
 2F00: D0 16 E6 4F A5 4F 29 20 \$1F91  
 2F08: C5 24 F0 0C 85 24 A9 FF \$D2B5

2F10: D1 28 D0 02 68 48 91 28 \$96F4  
 2F18: 68 60 20 92 FD A5 3C E9 \$D4FC  
 2F20: 0F 85 3A A5 3D E9 00 85 \$2094  
 2F28: 3B 4C 84 FE EA \$ECC0

2FD2: EC \$6724

2FE4: C4 \$2306

2FE9: E2 \$1F18

2FF2: C7 E9 \$E0F8

2FF5: E9 \$2C01

2FFA: 63 FB C9 FC \$C0F2

#### 4) Save the new version to disk

##### **BSAVE NEW F8,\$2800,L\$800**

Burn a new EPROM with this file and replace your normal Autostart F8-ROM with the NEW F8-ROM after ensuring that the necessary hardware changes to the F8-ROM socket have been made.

### How to Use the New F8-ROM

Following are some examples to illustrate the enhanced features of the ROM.

#### Booting Up

Insert a normal DOS 3.3 slave disk in your drive. It should have a simple Applesoft HELLO program which you can exit easily. When you power up your Apple, you should not get any beep or action at all except possibly some random display on your monitor. This is normal with the new ROM; just press return to start things going. The disk should boot up normally except no "APPLE ][]" message will appear during the cold-start (you should know what kind of computer it is by now!).

#### Continued from previous page

C010- KBDSTB .EQ \$C010 Keyboard clear strobe  
 FA62- RESET .EQ \$FA62 Normal autostart reset  
 FB2F- INIT .EQ \$FB2F Monitor text screen window initializer  
 FBF4- ADVANCE .EQ \$FBF4 Cursor advance  
 FC10- BS .EQ \$FC10 Cursor backspace  
 FC1A- UP .EQ \$FC1A Cursor up  
 FC66- LF .EQ \$FC66 Linefeed  
 FC58- HOME .EQ \$FC58  
 FCBA- NXTA1 .EQ \$FCBA Monitor routine to increment A1 and compare to A2  
 FD84- ADDINP .EQ \$FD84  
 FD92- PRA1 .EQ \$FD92 Prints parameter A1 as 4 hex digits  
 FDED- COUT .EQ \$FDED Character output routine  
 FE00- BL1 .EQ \$FE00 Print a blank  
 FE2C- MONMOV .EQ \$FE2C Monitor move routine  
 FE5E- LIST .EQ \$FE5E Old disassembler entry  
 FE63- LIST2 .EQ \$FE63 Disassemble a single line  
 FE80- SETINV .EQ \$FE80  
 FE84- SETNRM .EQ \$FE84 Set normal video  
 FF3A- BELL .EQ \$FF3A Beep routine  
 FF59- MON .EQ \$FF59 Monitor entry point  
 FF69- MONZ .EQ \$FF69 Silent monitor entry

\*-----\*  
 \* TAPE MESSAGE \*  
 \*-----\*

.OR \$FA6F On top of old INITAN  
 .TA \$2A6F

FA6F: 4C 7B FA JMP TAPMSG+9 Skip over message  
 FA72: 8D 87 D7  
 FA75: AF D2 A0  
 FA78: CF CE A0 TAPMSG .HS 8D87D7AFD2A0CFCEA0 The message "NO R/W ctrl-G ctrl-M"  
 in reverse order

\*-----\*  
 \* NEWESC ROUTINE \*  
 \*-----\*

.OR \$FB09 On top of old TITLE and XLTBL  
 .TA \$2B09

FB09: 38 NEWESC SEC for ESC1  
 FB0A: 49 C8 EOR #\$C8 base at "H"  
 FB0C: 29 DF AND #\$DF lower to upper  
 FB0E: F0 50 BEQ HOMEV "ESC-H" so do HOME and return  
 FB10: 20 2C FC JSR ESC1 other characters handled by ESC1  
 FB13: 20 0C FD JSR RDKEY if it was IJKM then get another key  
 FB16: 4C 09 FB JMP NEWESC and handle it as before

\*-----\*  
 \* HOMEV AND NEWNMI \*  
 \*-----\*

.OR \$FB60 On top of old TITLE  
 .TA \$2B60

FB60: 4C 58 FC HOMEV JMP HOME  
 FB63: 8D 00 29 NEWNMI STA SAVEA  
 FB66: 8E 01 29 STX SAVEX  
 FB69: 8C 02 29 STY SAVEY  
 FB6C: 4C DE FC JMP SAVE

\*-----\*  
 \* BINARY SEARCH \*  
 \*-----\*

.OR \$FB97 On top of old ESCOLD etc.  
 .TA \$2B97

FB97: A0 00 SEARCH LDY #\$00 Start at first byte of search string  
 FB99: B9 02 02 TEST LDA STRING,Y Get byte to match  
 FB9C: CD 00 02 CMP WILDCCD Is it wildcard?  
 FB9F: F0 04 BEQ MATCH Yes, simulate match  
 FBA1: D1 3C CMP (A1),Y Compare to memory byte  
 FBA3: D0 13 BNE NXTBYTE No match?,next memory byte  
 FBA5: C8 MATCH INY Matched, point to next string byte  
 FBA6: CC 01 02 CPY STRGLN Any more?  
 FBA9: 90 EE BCC TEST Yes, keep testing  
 FBAB: 20 80 FE JSR SETINV Complete match, prepare for display

Continued on next page

Continued from previous page

```
FBAE: 20 1A FF      JSR R0YDSP  and display the matching code
FBB1: A2 00        LDX #000   Prepare for listing
FBB3: 20 C8 FE      JSR NEWLST  and disassemble
FBB6: F0 05        BEQ DONE   Forced exit?
FBB8: 20 BA FC      NXTBYTE    JSR NXTA1  No, inc A1 and test against range
FBBB: 90 DA        BCC SEARCH Go back if more to search
FBD0: 4C 2F FB      DONE      JMP INIT   Reset window and return to monitor
FBC0: EA          .HS EA
```

-----\*  
\* MODIFIED ESC1 \*  
-----\*

```
.OR $FC2C  Modifies old ESC1
.TA $2C2C
```

```
FC2C: 69 FD      ESC1  ADC #0FD  Ready to test I,J
FC2E: 90 EA      BCC UP    I, do up & return to NEWESC+10
FC30: F0 DE      BEQ BS    J, do backspace & return to NEWESC+10
FC32: 69 FD      ADC #0FD  Ready to test K,L
FC34: 90 BE      BCC ADVANCE K, do advance & return to NEWESC+10
FC36: F0 04      BEQ POP   L, pop back to caller of NEWESC
FC38: 69 FD      ADC #0FD  Ready to test M,N
FC3A: 90 2A      BCC LF    M, do linefeed & return to NEWESC+10
FC3C: 68          POP      PLA      pop return address off stack
FC3D: 68          PLA      and return to caller of NEWESC
FC3E: 60          RTS
FC3F: EA EA EA   .HS EAEAEA Fill up hole with NOP's
```

-----\*  
\* NEWRESET \*  
-----\*

```
.OR $FCC9  On top of old HEADR etc
.TA $2CC9
```

```
FCC9: AD 00 C0    NEWRESET LDA KBD   Wait for key
FCCC: 10 FB      BPL NEWRESET
FCEE: 2C 10 C0    BIT KBDSTB Clear keyboard
FCD1: 29 DF      AND #0DF  Convert to uppercase
FCD3: C9 D3      CMP #'S+$80 Wanna save?
FCD5: F0 07      BEQ SAVE  Yes...
FCD7: C9 CD      CMP #'M+$80 Right to mon?
FCD9: F0 2D      BEQ GOMON Yes...
FCDE: 4C 62 FA   JMP RESET No. Just reset
FCDE: BA          SAVE    TSX      Save stack pointer
FCDF: 8E 03 29   STX SAVES for NMI users
FCE2: A0 00      LDY #00   Setup for whole page
FCE4: B9 00 00   MOV0N1    LDA ZEROPG,Y Get a zpg byte
FCE7: 99 00 20   STA SAVZPG,Y Save it..
FCEA: B9 00 01   LDA STAKPG,Y Get a stack byte
FCED: 99 00 21   STA SAVSTK,Y and save it too
FCF0: C8          INY
FCF1: D0 F1      BNE MOV0N1 More?
FCF3: 84 3C      STY A1L   Clear low bytes
FCF5: 84 3E      STY A2L   of pointers
FCF7: 84 42      STY A4L
FCF9: A9 02      LDA #002  Start at page2
FCFB: 85 3D      STA A1H
FCFD: A9 09      LDA #009  End at page 9
FCFF: 85 3F      STA A2H
FD01: A9 22      LDA #022  Put at page 22
FD03: 85 43      STA A4H
FD05: 20 2C FE   JSR MONMOV Use monitor move now
FD08: 4C 59 FF   GOMON    JMP MON   Enter monitor
FD0B: EA          NOP      Fill up space so that RDKEY starts at $FD0C
```

-----\*  
\* MODIFIED RDKEY \*  
-----\*

```
FD0C: A4 24      RDKEY    LDY CH    Get cursor position as index
FD0E: B1 28      LDA (BASL),Y Get character off screen
FD10: 6C 38 00   JMP (KSWL) and vector to character input routine
FD13: 20 3A FF   EOLBEL   JSR BELL  Beep for End of Line
FD16: 4C 81 FD   JMP STOP  and stop at end of line
FD19: EA          NOP
FD1A: EA          NOP
FD1B: 20 FD FE   KEYIN    JSR NEWCUR insert new cursor routine
FD1E: 2C 00 C0   BIT KBD  key pressed?
FD21: 10 F8      BPL KEYIN No, keep looping
```

Continued on next page

## The New Cursor

Exit to Applesoft immediate mode. You should notice that the cursor is a flashing underline instead of the flashing block. Type a few characters and, before pressing RETURN, use the backspace arrow and forward space arrow keys and observe the action of the cursor. If you have lowercase capability, type some lowercase letters and backspace the cursor over them. Notice that the new cursor does not produce odd characters when sitting on lowercase letters.

## The ESC codes

The "ESC" codes have been changed, somewhat. The familiar recursive I, J, K, M moves still work fine. Try them out. ESC @ has been changed to ESC H. This change is actually a happy coincidence (it's easy to remember "H" stands for "HOME"). The escape handler requires consecutive codes be used and, luckily, H comes before I in the ASCII table in the same way that @ comes before A. If you forgot to switch off your lowercase, you will be pleased to notice that the ESC codes also work with h, i, j, k, and m. I have eliminated the non-recursive A, B, C, D, E and F codes altogether since they are used very little, if at all.

## Long Lines

To try out the new oversized line handling, hold down a letter and the repeat key. The screen line will begin to fill up as usual. If you continue to add characters, the cursor will eventually come to a stop when you reach the 254th character. At this point, the only two keys accepted are backspace or return. Any other key will cause a beep and will overtype the previous character. With this arrangement, you can enter those long lines of machine code or Applesoft without fear of losing the entire line just trying to fit in that last character. You need only back up to the nearest logical spot, hit RETURN and carry on with the next line.

## Disassembler Improvements

Type RETURN to prepare for a new input line and enter the monitor by typing

### CALL-151

You should get the asterisk (\*) prompt as usual. Type

### FEC8L

and you will get a screenful of disassembled instructions as you might expect but, instead of returning to the \* prompt, the cursor does not reappear. If you press the space bar at this point, the next disassembled instruction will appear at the bottom of the screen. Try it a few times. Now press the forward arrow key. You should see another screenful of instructions. Any other key will return you to the monitor prompt.

You no longer need to use LLLLLL and control-S to scan through a program. This also works with your printer to let you print disassembled code without breaks for more L's.



## Handling RESET

The RESET key works in a manner that is similar to the Super Saver ROM, but not exactly the same. When you press reset (or at power up) the computer will 'hang'. Actually, it is waiting for a keypress. Almost any key will let it carry on with the normal reset cycle (either booting the disk, entering BASIC or passing control to a resident program). However, two keys are treated differently. If you press the "M" key, you will enter the monitor directly, regardless of the contents of RAM. If you press the "S" key, pages \$00 through \$08 will be moved to pages \$20 through \$28. After the moves are done, the monitor is entered. Incidentally, "m" and "s" are also accepted.

## Non-Maskable Interrupts

The Non-Maskable Interrupt vector is handled directly without waiting for a keypress. If you generate an NMI through some means, locations \$2900 to \$2903 will be loaded with the values of A, X, Y, and S, respectively, following the interrupt. Pages \$00 to \$08 are then moved up to pages \$20 to \$28 and the Apple enters the monitor.

The value at \$2903 allows you to locate (in page \$2100) the position of the stack pointer following the interrupt. Its value points to the byte after the status value (P) was pushed when the NMI was received. For example, after you generate a non-maskable interrupt, the stack pointer value will be found at \$2903. If \$2903 contained, for example, \$FA, then you should examine \$21FB, \$21FC, and \$21FD. In these locations will be found P, PCL, and PCH, respectively, at the time the NMI occurred. By examining these values, the associated code and the contents of the registers, you have everything needed to duplicate "copycard"-like restarts.

## The Binary Search

A very useful feature of the new ROM is its built-in binary search. Try this example to see how it works.

### ESC H

to clear the screen (Don't hit return!). Now type

**0200:FF 01 EA N F800.FFFF**

and then hit RETURN. You should see several things:

First, notice that the command line stays at the top of the screen because the scroll window has been set below it. Second, "FAD5-" is printed in inverse below the command line. And third, you should have a disassembled listing starting at FAC5.

Look through the listing and find the byte at FAD5. It is an EA (NOP). The command line in the example specified that you wished to find all the occurrences of the byte EA in the range from F800 to FFFF.

The search routine identifies the location of the start of the matching string and prints it in inverse. It then disassembles starting fifteen bytes before that address. This allows the

*Continued from previous page*

```

FD23: 91 28          STA (BASL),Y Yes, so restore original character to screen
FD25: 84 24          STY CH       and restore CH to original value
FD27: EA           NOP
FD28: AD 00 C0      LDA KBD      Now get the key
FD2B: 2C 10 C0      BIT KBDSTB   and clear flag
FD2E: 60           RTS
FD2F: 20 0C FD     ESC JSR RDKEY    Get next key for ESC handler
FD32: 20 09 FB     JSR NEWESC   and process it
FD35: 20 0C FD     RDCHAR JSR RDKEY    Get a key
FD38: C9 9B        CMP #$9B     Is it ESC?
FD3A: F0 F3        BEQ ESC      Yes, so handle it specially
FD3C: 60           RTS          No, so pass it to caller

```

```

*-----*
*                                     MOD.NOTCR
*-----*

```

```

.OR $FD58      On top of end part of NOTCR
.TA $2D58

```

```

FD58: E0 FE        CPX #$FE     End of line?
FD5A: 90 03        BCC NOTCR1   No, keep going
FD5C: 4C 13 FD     JMP EOLBEL   Yes, handle end of line
FD5F: E8          NOTCR1 INX       Carry on

```

```

*-----*
*                                     STOP PATCH
*-----*

```

```

.OR $FD7E      On top of old CAPTST
.TA $2D7E

```

```

FD7E: 4C 84 FD     JMP ADDINP   Skip over new STOP code
FD81: E8          STOP        INX       Compensate for DEX in BCKSPC code
FD82: A9 88        LDA #$88     Overwrite with backspace character

```

```

*-----*
*                                     MAKE LIST2 ROUTINE DROP THROUGH TO A1PC
*-----*

```

```

.OR $FE74
.TA $2E74

```

```

FE74: AA          TAX

```

```

*-----*
*                                     MISCELLANEOUS CODE
*                                     -NEWLST,SRCHV,NOTAPE,CRMON,
*                                     NEWCUR,RDYDSP ROUTINES
*-----*

```

```

.OR $FEC2      Old TRACE start
.TA $2EC2

```

```

FEC2: 60          TRACE  RTS       No trace, but
FEC3: EA          NOP       allow space
FEC4: EA          NOP       for a JMP
FEC5: 4C F8 03    USR     JMP USRADR Ctrl-Y vector moved up a little

```

```

*-----*
*                                     MODIFIED DISASSEMBLER PATCH
*-----*

```

```

FEC8: 20 5E FE     NEWLST JSR LIST    Do a screenful
FECB: A9 01        NEWLST1 LDA #$01    Disassemble
FECD: 20 63 FE     JSR LIST2 one line only
FED0: AD 00 C0      CHKKEY   LDA KBD     Wait for a key
FED3: 10 FB        BPL CHKKEY
FED5: 2C 10 C0      BIT KBDSTB Got one
FED8: C9 95        CMP #$95   Was key "->"?
FEDA: F0 EC        BEQ NEWLST Yes, another screenful
FEDC: C9 A0        CMP #$A0   Was key a space?
FEDE: F0 EB        BEQ NEWLST1 Yes, one more line
FEE0: C9 9B        CMP #$9B   Test ESC for search routine
FEE2: 60          RTS        And return to caller

```

```

*-----*
*                                     WINDOW SET AND VECTOR TO SEARCH
*-----*

```

```

FEE3: A9 02        SRCHV   LDA #$02    Freeze top 2

```

*Continued on next page*

Continued from previous page

```
FEE5: 85 22      STA WNDTOP   screenlines
FEE7: 4C 97 FB      JMP SEARCH   and goto binary search
```

```
*-----*
*                MESSAGE FOR R AND W COMMANDS                *
*-----*
```

```
FEEA: A0 08      RDORWR   LDY #08      Nine characters
FEEC: B9 72 FA      RDWR1    LDA TAPMSG,Y Get a character
FEED: 20 ED FD      JSR COUT   Print it
FEF2: 88          DEY       Next char...
FEF3: 10 F7      BPL RDWR1  Loop till done
FEF5: 60          RTS
```

```
*-----*
*                UNCHANGED CRMON ROUTINE                      *
*-----*
```

```
FEF6: 20 00 FE      CRMON    JSR BL1     Unchanged CRMON routine
FEF9: 68          PLA
FEFA: 68          PLA
FEFB: D0 6C      BNE MONZ
```

```
*-----*
*                NEW CURSOR PATCH                             *
*-----*
```

```
FEFD: 48          NEWCUR   PHA         Save Screen chr
FEFE: E6 4E      INC RNDL   Count Low
FF00: D0 16      BNE EXIT  Exit if no change to hi byte
FF02: E6 4F      INC RNDH   Count High
FF04: A5 4F      LDA RNDH   Get new Hi byte
FF06: 29 20      AND #20    Discard all but 1 bit
FF08: C5 24      CMP TEMP   Same as last time?
FF0A: F0 0C      BEQ EXIT  Exit if no change
FF0C: 85 24      STA TEMP   Different so save for test next time
FF0E: A9 DF      LDA #'_+$80 Get underline
FF10: D1 28      CMP (BASL),Y Already on screen?
FF12: D0 02      BNE PUTSCN No, so put it on screen
FF14: 68          PLA        Yes, so get chr back
FF15: 48          PHA        but don't disturb stack
FF16: 91 28      PUTSCN    STA (BASL),Y Put chr or underline on screen
FF18: 68          EXIT      PLA        Get original chr back in A
FF19: 60          RTS        and return to KEYIN
```

```
*-----*
*                ROUTINE TO SET UP FOR SEARCH DISPLAY          *
*-----*
```

```
FF1A: 20 92 FD      RDYDSP    JSR PRA1    Print matching location for SEARCH
FF1D: A5 3C      LDA A1L    Get low byte
FF1F: E9 0F      SBC #0F    and backup
FF21: 85 3A      STA PCL    disassembler
FF23: A5 3D      LDA A1H    Get hi byte
FF25: E9 00      SBC #00    Include borrow
FF27: 85 3B      STA PCH    for disassembler
FF29: 4C 84 FE      JMP SETNRM Set normal video and return to SEARCH
FF2C: EA          NOP
```

```
*-----*
*                MAKE S COMMAND VALID                          *
*-----*
```

```
.OR $FFD2
.TA $2FD2
```

```
FFD2: EC          .HS EC      An "S" plus $89 and 0Red with $B0
```

```
*-----*
*                FIX USR COMMAND                              *
*-----*
```

```
.OR $FFE4
.TA $2FE4
```

```
FFE4: C4          .DA #USR-1
```

```
*-----*
*                RE-ROUTE S COMMAND                            *
*-----*
```

```
.OR $FFE9
```

Continued on next page

disassembler to be synchronized with the instruction stream well before the located address and also gives you a look at some of the code on either side of the address.

Since you are in the disassembler, the space bar and forward arrow keys work as described above. If you press ESC, the search will be terminated immediately. Any other key permits the search to proceed beginning at the byte immediately after the start of the previously found location (regardless of the location to which you disassembled).

The syntax of the command line is as follows:  
0200:ww nn bb ... bb N ssss.eeeeS

which you should type after first using ESC H to clear the screen.

The "0200:" specifies storage of bytes at location \$0200 (the keyboard buffer).

Byte "ww" is your chosen 'wildcard' value for this search. You must give a value here even if you don't use the wildcard value in the search string. If you use the chosen wildcard value in the search string, any memory byte will match it.

Byte "nn" is set to the number of consecutive bytes you wish to match (01 matches 1 byte, 0A would match 10 bytes, etc.).

The sequence "bb ... bb" is the specific string of bytes you wish to search for in memory. There should, of course, be nn of these bytes. You may use your chosen wildcard value in the string wherever you wish (or not at all).

The "N" is used to separate the storage function from the range specification which follows (be sure to include a space on either side).

"ssss.eeee" is the standard form for a range specification where "ssss" is the starting address and "eeee" is the ending address identifying the range to be searched.

Finally, the "S" specifies that you want the new search command to be executed. It is important that no space be typed between the end address and the S.

### Wildcard Examples

0200:AA 03 4C AA FE N F800.FFFFS

finds all JMPs anywhere into page \$FE (the sequence 4C xx FE) in the range \$F800 to \$FFFF

0200:BB 05 A9 A0 BB ED FD N FB00.FE00S

finds the code which prints a space to the current output device (the sequence A9 A0 xx ED FD)

### Sorry, No Cassette

Trying to read a range of memory with the R or W commands now results in the message: "NO R/W" and a beep.

### Details of the New Routines

There is very little space in the F8-ROM for user routines. For almost any significant new routine, something must be given up. The cassette routines are the first to go since

(almost!) every Apple owner has at least one disk drive and has long abandoned his cassette recorder, except as a means of saving sections of cracked code or loading in search utilities when booting a disk might overwrite some important memory contents. Hopefully, this new ROM will minimize the need for cassette support even further.

An important consideration when changing the F8 is that many of the existing routines are heavily used by external programs and, therefore, cannot have their entry points changed without causing drastic incompatibilities with existing software. I have tried to minimize the problem by attacking only sections which are very rarely used outside of the monitor itself.

Although deleting the cassette routines yields a good sized slice of memory, I found several additional sites for new code. Following is a list of the addresses I have used and a description of their contents.

**FA6F-FA7A (12 bytes)** - the INITAN entry point. This code set the game port annunciator outputs to initial states during a reset cycle. Very few people use the annunciator outputs and those that do may not require initialization during reset. I used the first three bytes to JMP over the next nine bytes where I stored the message "NO R/W" which is issued if the user attempts to use the monitor cassette commands. If you need the annunciator initialization, you can leave this section

unchanged and make the following change:

```
FEEA:4C 2D FF EA EA EA EA EA
FEF2:EA EA EA EA
```

This will simply cause "ERR" and a beep if R or W commands are used.

**FB09.FB10 (8 bytes)** - this was the TITLE message which appears during coldstart. I used this space for part of the new escape handler code and changed the cold start message printer (APPLEII) to JMP HOME.

**FB11.FB18 (8 bytes)** - this was the XLTBL (translate table) used to convert the IJKM keycodes into the old ABCD codes before they were passed to the old escape handler code. Since the entire escape handler has been rewritten to handle IJKM directly, this table is no longer needed. In conjunction with the previous group of bytes, the entire new escape handler now resides between FB09 and FB18.

**FB60.FB6E (15 bytes)** - the APPLEII routine that was used to put the coldstart title on the screen is now used to hold the JMP HOME instruction for the coldstart and the rest is used for the register save code of the NMI handler.

**FB97.FBC0 (42 bytes)** - the ESCOLD, ESCNOW and ESCNEW routines followed by 14 NOPs were completely

*Continued on next page*

*Continued from previous page*

```
.TA $2FE9
FFE9: E2 .DA #SRCHV-1
```

```
*-----*
* RE-ROUTE LIST COMMAND (L) *
*-----*
```

```
.OR $FFF2
.TA $2FF2
FFF2: C7 .DA #NEWLST-1
```

```
*-----*
* RE ROUTE READ & WRITE COMMANDS (R & W) *
*-----*
```

```
FFF3: E9 .DA #RDORWR-1
.OR $FFF5
.TA $2FF5
```

```
FFF5: E9 .DA #RDORWR-1
*-----*
* FIX NMI AND RESET VECTORS *
*-----*
```

```
.OR $FFFA
.TA $2FFA
FFFA: 63 FB .DA NEWNMI
FFFC: C9 FC .DA NEWRESET
```

## Most Wanted List

### Need help backing-up a particularly stubborn program?

Send us the name of the program and its manufacturer and we'll add it to our Most Wanted List, a column (updated each issue) which helps to keep Hardcore COMPUTIST readers informed of the programs for which softkeys are MOST needed. Send your requests to:

**Hardcore COMPUTIST  
Wanted List  
PO Box 110846-K  
Tacoma, WA 98411**

If you know how to deprotect unlock, or modify any of the programs below, let us know. You'll be helping your fellow Hardcore COMPUTIST readers and earning MONEY at the same time. Send the information to us in article form on a DOS 3.3 diskette.

- |   |  |
|---|--|
| <b>1. Apple Business Graphics</b><br>Apple Computer   | <b>13. Robot Odyssey</b><br>The Learning Company             |
| <b>2. Flight Simulator II</b><br>Sub Logic            | <b>14. Zardax</b><br>Computer Solutions                      |
| <b>3. Apple LOGO II</b><br>Apple Computer             | <b>15. The Listers</b><br>Silicon Valley Systems             |
| <b>4. DB Master 4.0+</b><br>Stoneware, Inc.           | <b>16. Milliken Math Series (NEW)</b><br>Milliken Publishing |
| <b>5. Bookends</b><br>Sensible Software               | <b>17. College Entrance Exam Prep</b><br>Borg Warner         |
| <b>6. Visiblend</b><br>Microlab                       | <b>18. Bank Street Speller</b><br>Broderbund                 |
| <b>7. Sundog</b><br>FTL Games                         | <b>19. Karateka</b><br>Broderbund                            |
| <b>8. Lifesaver</b><br>Microlab                       | <b>20. Microzine Series</b><br>Scholastic                    |
| <b>9. Catalyst</b><br>Quark, Inc.                     | <b>21. Report Card</b><br>Sensible Software                  |
| <b>10. Gutenberg Jr. &amp; Sr.</b><br>Micromation LTD | <b>22. Frogger</b><br>Sierra On-Line                         |
| <b>11. Prime Plotter</b><br>Primesoft Corp.           | <b>23. Hayes Terminal Program</b><br>Hayes                   |
| <b>12. The Newsroom</b><br>Springboat Software        | <b>24. Fun Bunch</b><br>Unicorn                              |



**Hardcore COMPUTIST** welcomes articles of interest to users of the Apple II (or compatible) computers and would like to publish well-written material including:

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

Send your submission on a DOS 3.3 disk using an Apple (or compatible) editing program. Enclose a double-spaced hardcopy manuscript (typewritten or computer printed). Submissions will be returned only if adequate packaging is enclosed.

**Have you  
written  
an ARTICLE or  
PROGRAM  
you'd like to see  
published in  
Hardcore  
COMPUTIST?**

**We would like to hear  
from you!**

**Hardcore COMPUTIST** pays on publication. Rate of payment depends on the amount of editing necessary and the length of the article- generally between \$10 for a short softkey, and \$50 per typeset page for a full-length article. For a higher pay rate, enclose the original commercial disk for verification of softkeys. We guarantee the disk's return.

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

For a copy of our WRITER'S GUIDE, send a business-sized (22-cent) SASE (self-addressed, stamped envelope) to:

**Hardcore COMPUTIST**  
Writer's Guide  
PO Box 110846-K  
Tacoma, WA 98411

*Continued from previous page*

replaced by rewriting the escape handler. This space now houses most of the binary search routine.

**FC2C.FC40** (22 bytes) - the ESC1 routine which actually handled the old cursor moves, etc. has been modified to handle LJKM directly.

**FCC9.FD0B** (66 bytes) - this section handled the timing and control of the cassette ports to read and write bits. It now contains the new reset handler and move-memory-up routines.

**FEC2.FEF5** (49 bytes) - the cassette Write routines at this spot have been replaced with the disassembler controller code, the window set-up and jump to the search routine and the "NO R/W" message printer.

**FEFD.FF2C** (48 bytes) - the cassette Read command handler is now replaced by the new cursor routine and the RDYDSP subroutine for the search.

### Adding Your Own Changes

You can try other characters for the cursor by changing location \$FF0F to the desired character (\$FF looks good to some people). Changing the mask byte at \$FF07 will change the cursor flash rate. Use bytes with only one bit set (\$01, \$02, \$40, \$80, etc.).

Another possibility is to accept L or N as valid ESC codes since they are easily detected in the ESC1 routine. ESC L might, for example, 'autotype' LIST or LOAD into the keyboard buffer.

How about supporting the shift key modification right in the F8?

For further modifications, you may wish to use the space housing the low resolution graphics support routines at F800.F881 (129

bytes). This will, of course, destroy the GR, COLOR, PLOT, HLIN, VLIN and SCRN commands used by APPLESOFT for lo-res graphics. Since lo-res is not used quite as much as it used to be, this may be acceptable to some.

The most practical way to develop changes to the F8, is to move it into a RAM card in slot 0 and set the card for read/write access. An example of this technique is given in NIBBLE magazine Vol. 4/No. 2/1983 pg.167.

### About Compatibility

The only F8-ROM's fully compatible with all software are the true Autostart ROM and/or the original Monitor ROM. Presumably, any change to the F8 could be detected by doing a checksum and checking against the standard checksums for unmodified F8-ROM's. Most programs do not go to this effort, but some do (like ProDOS). If you run into one of these programs, you will have to change back to the normal F8-ROM. A method of defeating the ProDOS checksum was printed in Hardcore COMPUTIST No. 9, pg. 18.

Naturally, any program which uses the sections of code which were modified, perhaps by entering at non-standard addresses, may also have trouble. Again, this should not happen often since those sections affected are unlikely to be used directly by any program.

### A Hardware Suggestion

Since you may, on occasion, still need the original F8-ROM, you could use the procedure outlined on page 9 of this issue to install a dual ROM where the original and the new ROM can be just a flip of the toggle switch away.

## Bugs

### Hardcore COMPUTIST No. 18, pg. 25

The hexdump which accompanied the article "Simple Copy Protection" by Rohn Smith was incorrect. The corrected hexdump is as follows:

```

0300: 60 A9 11 8D EC B7 A9 01 $06DA
0308: 8D F4 B7 20 79 03 20 93 $23E9
0310: B7 AD 01 10 C9 11 D0 60 $CAAE
0318: A9 00 85 FE A9 1F 85 FF $2653
0320: A2 0E A0 01 A9 23 91 FE $D720
0328: C6 FF CA D0 F9 A9 24 8D $B6EE
0330: 34 10 A9 FF 8D 7C 10 8D $32E5
0338: 7D 10 A9 23 8D 01 10 A9 $5F2F
0340: 02 8D F4 B7 A9 23 8D EC $07A9
0348: B7 20 79 03 20 93 B7 A9 $F4C4

0350: 1F 85 FF A9 00 85 FE A2 $3E35

```

```

0358: 10 A8 91 FE C8 D0 FB C6 $1BB7
0360: FF CA D0 F6 A9 02 8D F4 $3683
0368: B7 A9 11 8D EC B7 20 79 $653E
0370: 03 20 93 B7 A9 00 85 48 $9033
0378: 60 A9 00 8D EB B7 A9 0F $05F5
0380: 8D ED B7 A9 00 8D F0 B7 $CCF8
0388: A9 1F 8D F1 B7 A9 10 8D $A7F0
0390: E1 B7 60 $58EE

```

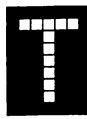
### pg. 22

The article, "Checksoft v2" by Ray Darrah contains an error in the command found under the subtitle, "Typing It In Using Checksoft v2". The correct command should be:

**BSAVE CHECKSOFT,A\$2E7,L\$E9**

# The Nibbler: A Utility Program To Examine Raw Nibbles From Disk

By  
Jan Eugenides  
& Ray Darrah

 The Nibbler is a utility program which allows you to read any track on a disk into memory in its raw form. Unlike a sector editor, The Nibbler makes no assumptions about the formatting of the disk in the drive and does not decode the data it reads from a disk. It simply reads a full track of encoded data and stores it in a buffer where it can be examined in order to determine the exact type of formatting that has been used. The Nibbler comes in very useful if you are trying to acquire some of the details on what copy-protection techniques have been used on a particular disk.

Even though The Nibbler suits my current needs, there is room for you to add your own bells and whistles. For those interested in modifying The Nibbler, source code for the machine language portion of the program begins on this page.

As it now stands, The Nibbler will read any track of the disk into a buffer which extends from \$4000-8000. This is approximately twice the length of a track, just to be sure we really get the whole track in memory. Once it's there, you can choose from several options to learn more about the formatting of the disk.

For those of you not familiar with the manner in which data is stored on an Apple disk, here is a brief overview.

## Disk Formatting - An Overview

The Apple disk uses what is called "soft sectoring." This means that a sector is found through software (i.e. DOS 3.3 or another operating system), rather than by some hardware device. A "hard sectored" disk drive will generally utilize the small holes in a floppy disk and its jacket in conjunction with a light

sensing device to detect the beginning of a sector. The Apple does not use these holes for anything at all.

So, you ask, how does the drive know where sectors begin and end? When a disk is formatted a pattern of bytes is written to

it beginning with a series of special "autosync" bytes. For normal DOS 3.3 these special bytes have a value of \$FF. Probably the best way to see these bytes is to type in The Nibbler and use it to read in track zero of a regular DOS 3.3 disk (or a ProDOS disk; the

### The Nibbler Source Code

```

*-----
* NIBBLER MACHINE ROUTINES
* INSPIRED BY JAN EUGENIDES
*-----

                .OR $800
                .TF OBJ.NIBBLER

0026- BUFPTR   .EQ $26           POINTS TO CURRENT BUFFER BYTE
FD8E- CROUT    .EQ $FD8E        PRINTS A CARRIAGE RETURN
FDDA- PRBYTE   .EQ $FDDA        PRINTS ACCUM. AS HEX
FBC1- VTAB     .EQ $FBC1        CALCULATES BAS ADDRESS FOR SCREEN LINE
0024- CH       .EQ $24           CURRENT CURSOR HORIZONTAL POSITION
0025- CV       .EQ $25           CURSOR VERTICAL POSITION
C08E- SETRD    .EQ $C08E        DISK READ MODE
C08C- READ     .EQ $C08C        DISK READ LATCH
00FC- STATUS   .EQ $FC          WHETHER THE STRING WAS FOUND OR NOT
00FD- NUMBYTES .EQ $FD          NUMBER OF BYTES TO SEARCH FOR
00FE- X        .EQ $FE          XPOSITION OF CURSOR
00FF- Y        .EQ $FF          YPOSITION OF CURSOR
00A1- INTLSB   .EQ $A1
00A0- INTMSB   .EQ $A0
FBC1- BASCALC  .EQ $FBC1
0028- BASL     .EQ $28          BAS ADDRESS OF SCREEN
FDED- COUT     .EQ $FDED
0300- INBUFF   .EQ $300        INPUT BUFFER FOR SEARCH BYTES

*-----
* FAKE BASIC PROGRAM
*-----

0800: 00 0D 08
0803: 0A 00                .HS 000D080A00
0805: 8C 32 30
0808: 36 33 3A
080B: AC                  .HS 8C323036333AAC
080C: 00 00 00            .HS 000000

*-----
* MOVE POINTER TO BEGINNING OF
* BASIC PROGRAM
*-----

```

*Continued on next page*

Continued from previous page

```
080F: A9 75          LDA #END.OBJ
0811: 85 67          STA $67
0813: A9 09          LDA /END.OBJ
0815: 85 68          STA $68
0817: 60             RTS
```

\*-----  
\* ZERO BUFFER POINTER  
\*-----

```
0818: A0 00      ZBUFTR LDY #0      ZERO THE Y OFFSET ALSO
081A: 84 26              STY BUFPTR
081C: A9 40      LDA #$40      BUFFER STARTS AT $4000
081E: 85 27      STA BUFPTR+1
0820: 60             RTS.1      RTS
```

\*-----  
\* INCREMENT BUFFER POINTER  
\*-----

```
0821: C8          NXTBPTR INY          NEXT OFFSET
0822: D0 FC          BNE RTS.1      NO PAGE CROSSING, EXIT
0824: E6 27          INC BUFPTR+1   PAGE CROSSING SO NEXT MSB
0826: A5 27          LDA BUFPTR+1   ARE WE PAST THE $7FFF
0828: C9 08          CMP #$80       LIMIT?
082A: B0 EC          BCS ZBUFTR     YES, BUFFER WRAP AROUND
082C: 60             RTS
```

\*-----  
\* PRINT 19 LINES OF BUFFER DATA  
\* VIA USR  
\*-----

```
082D: 20 0C E1      JSR $E10C
0830: A5 A0      ML.ENTRY LDA INTMSB
0832: 85 27      STA BUFPTR+1
0834: A4 A1      LDY INTLSB
0836: A9 03      LDA #3      VTAB4
0838: 85 25      STA CV
083A: 20 8E FD      JSR CROUT   NEXT VTAB, HTAB 1
083D: 20 21 08 PRINT19 JSR NXTBPTR GET NEXT ADDR
0840: A5 25      LDA CV      DONE?
0842: C9 17      CMP #23     BOTTOM LINE
0844: B0 08      BCS WAIT.KEY YUP!
0846: B1 26      LDA (BUFPTR),Y GET BYTE
0848: 20 DA FD      JSR PRBYTE  PRINT IT
084B: 4C 3D 08      JMP PRINT19 KEEP GOING
```

\*-----  
\* WAIT FOR A KEY  
\*-----

```
084E: A5 FF      WAIT.KEY LDA Y          YPOSITION
0850: 18          CLC            ADD 4 TO Y
0851: 69 04          ADC #4
0853: 20 C1 FB      JSR BASCALC GET ADDR OF LINE
0856: A4 FE          LDY X          GET XPOSITION OF HIGHLIGHTED BYTE
0858: A2 02          LDX #2         INVERSE TWO BYTES
085A: B1 28      INV1 LDA (BASL),Y
085C: 48          PHA            SAVE FOR RESTORE
085D: 29 3F          AND #$3F       MAKE INVERSE
085F: 91 28          STA (BASL),Y
0861: C8          INY
0862: CA          DEX
0863: D0 F5          BNE INV1
0865: AE 00 C0      WAIT1 LDX $C000      WAIT FOR KEYPRESS
```

Continued on next page

formatting is the same.) See the instructions below on using The Nibbler.

After you have read in track 00, you might see something like this:

```
SLOT=>6          SEARCH=>000000000000 (U)
DRIVE=>1
TRACK=>00.0      ?=HELP

969FA79AB5CE969FA79AB596CE96B2D96DD96DD
96DEDD9696E59696969796DF9BDF9696DF969697
96EC96EC96A6969696E6E79696969B9696F59696
96979696969B9696F3F4F39696F29696F2EF96FD
96FEFE96979696969BEF96969697F29696EDEE96
9696A69696ED96ED979696969BEDED9696979696
96B3DEAAEBFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFD5AA96FFFEAAAAEAAFBFDEAAAFAB5FFFF
FFFFFFFFD5AAD96969696969696969696969696
96969696969696969696969696969696969696
96969696969696969696969696969696969696
96969696969696969696969696969696969696
96969696969696969696969696969696969696
96969696969696969696969696969696969696
96969696969696969696969696969696969696
96969696969696969696969696969696969696
96969696969696969696969696969696969696
96969696969696969696969696969696969696
96969696969696969696969696969696969696
```

Notice that about halfway down there is a series of \$FF bytes (commonly 7 or 8, but occasionally as many as 15 or 16). These are the autosync bytes. The longer sequence indicates the beginning of the track. A shorter sequence is used before address and data fields.

Directly following each series of \$FF bytes you'll see D5 AA 96- the address prologues. They mark the beginning of the address fields which contain all the information on which sector follows, the track volume number, and a checksum (not in that order!).

Next you will see DE AA EB- the address epilogue, or trailer. This marks the end of the address field. The EB on the end is sometimes incorrect and may be an E7, EC or some similar value. DOS is not interested in this when reading a disk, so we don't need to be concerned, either.

After a few more autosync bytes, you will see D5 AA AD. This is the data prologue (which indicates the beginning of the actual data field) and is followed by one sector of data. Because the disk drive is unable to accurately read two consecutive zero bits, each sector has to be encoded when written. Due to the method of encoding (known as 6&2 encoding) it actually takes 342 bytes on the disk to store one sector of data.

At the end of the data you will see D5 AA EB- the data epilogue. In the interest of space, the example above does not show a complete sector; however, The Nibbler does read in a whole track (and a bit more) and store it at \$4000 to \$8000.

By using these unique prologue and epilogue bytes as markers, the Apple is able to keep track of the location of the read/write head on the disk.



## A Common Copy-Protection Technique

Probably the most common copy protection technique used is that which changes the address and data epilogues and/or prologues. (If you read this magazine often, you are probably tired of hearing about epilogues and prologues. If you're new, you'll be tired of them soon!) The Nibbler can help you locate these changed markers by reading in the raw data, which you can then look over and easily pinpoint the markers that have been changed. This information can be used with a nibble copier or Super IOB to copy the disk.

Unfortunately, these days the copy protection rarely stops there. Programs can use different auto-sync bytes, and can even use different nibblizing techniques to effectively scramble the data on a disk and make it unrecognizable. Oh well, if it was easy there would be no challenge, right? Anyway, the data and address markers are a good place to begin your investigation of a protected disk.

One bit of useful information: On a bootable disk, the address and data prologues for track 0, sector 0 cannot be modified from D5 AA 96 and DE AA. This means that the first sector is usually open to scrutiny. I've seen some weird disk formats, but this is one part of the disk that is hard to hide.

### Typing It In

First type in the Hexdump (using the directions on pg. 2 of this magazine) at the end of this article and save it with

**BSAVE OBJ.NIBBLER,A\$800,L\$177**

Next, clear memory and fix the beginning of program pointer

**FP**

Third, type in the BASIC program (again, follow the directions on pg. 2) and save it with

**SAVE BAS.NIBBLER**

The machine language portion of this program was designed in a special way so that it could be merged with the BASIC portion in order to form one file (this process is explained in detail on pg. 20 of Hardcore COMPUTIST No. 17). Use the following procedure to merge the files.

First, clear memory and fix the beginning of program pointers:

**FP**

Next, load in the machine language portion:

**BLOAD OBJ.NIBBLER,A\$800**

A LIST at this point should reveal:

10 CALL 2063 : RUN

Now use BASIC to partially execute this file:

**RUN**

And load the BASIC portion of the file

*Continued from previous page*

```

0868: 10 FB          BPL WAIT1    NOT YET
086A: 8D 10 C0       STA $C010
086D: 88             DEY         NORMALIZE PLACE
086E: 68             PLA         GET BYTE
086F: 91 28          STA (BASL),Y PUT ON SCREEN
0871: 88             DEY         DO OTHER DIGIT
0872: 68             PLA         FIRST BYTE
0873: 91 28          STA (BASL),Y
0875: E0 C9        I          CPX #$C9    MOVE UP?
0877: D0 38        BNE J       NO, TRY LEFT
0879: C6 FF        MOVE.UP    DEC Y       MOVE UP
087B: 10 D1        BPL WAIT.KEY NOT OFF EDGE OF SCREEN
087D: E6 FF        INC Y       RESTORE TO ZERO
087F: A5 A0       LDA INTMSB  CAN SCROLL BACKWARDS?
0881: C9 41        CMP #$41    BEGINNING OF BUFFER+1
0883: 90 C9        BCC WAIT.KEY NO MOVE
0885: A5 A1       LDA INTLSB  LSB SBC
0887: E9 14        SBC #20
0889: 85 A1        STA INTLSB
088B: B0 A3        BCS ML.ENTRY MOVE, NO PAGE CROSSING
088D: C6 A0       DEC INTMSB
088F: D0 9F        ENTRYNE    BNE ML.ENTRY ..ALWAYS
0891: E0 CD        M          CPX #$CD    MOVE DOWN?
0893: D0 3E        BNE RTS.2  NOPE, LET BASIC FIX IT
0895: E6 FF        MOVE.DOWN  INC Y
0897: A5 FF        LDA Y
0899: C9 13        CMP #19
089B: 90 B1        BCC WAIT.KEY
089D: C6 FF        DEC Y       RESTORE TO 18
089F: A5 A0       LDA INTMSB  CAN DISPLAY MORE?
08A1: C9 7E        CMP #$7E
08A3: B0 A9        BCS WAIT.KEY NOPE!
08A5: A5 A1       LDA INTLSB  LSB + 20
08A7: 69 14        ADC #20     C=0 ALREADY
08A9: 85 A1        STA INTLSB
08AB: 90 83        BCC ML.ENTRY GO TO ML.ENTRY
08AD: E6 A0       INC INTMSB
08AF: D0 DE        BNE ENTRYNE GO TO ML.ENTRY
08B1: E0 CA        J          CPX #$CA    MOVE LEFT?
08B3: D0 0C        BNE K       NO, TRY RIGHT
08B5: 88             DEY
08B6: 88             DEY         X-2
08B7: 84 FE        STY X
08B9: 10 93        BPL WAIT.KEY
08BB: A0 26        LDY #38     WRAP AROUND
08BD: 84 FE        STY X
08BF: D0 B8        ???? MOVE.UP ..ALWAYS
08C1: E0 CB        K          CPX #$CB    MOVE RIGHT?
08C3: D0 CC        BNE M       NO, TRY DOWN
08C5: C8             INY
08C6: C8             INY         X+2
08C7: 84 FE        STY X
08C9: C0 27        CPY #39     NEXT LINE?
08CB: 90 81        BCC WAIT.KEY NOPE!
08CD: A0 00       LDY #0
08CF: 84 FE        STY X
08D1: B0 C2        BCS MOVE.DOWN ..ALWAYS
08D3: A5 A0       RTS.2      LDA INTMSB
08D5: A4 A1       LDY INTLSB
08D7: 4C F2 E2    JMP $E2F2

```

\* -----  
\* FILL BUFFER WITH DISK DATA  
\* -----

```

08DA: 20 E3 03       JSR $3E3    GET ADDR OF RWTS PARMS

```

*Continued on next page*

Continued from previous page

```
08DD: 20 D9 03      JSR $3D9      SEEK CORRECT TRACK
08E0: AE E9 B7      LDX $B7E9     SLOT NUMBER, USE 3 SECOND TURN OFF DELAY
08E3: BD 8E C0      LDA SETRD,X   INSURE READ MODE
08E6: 20 18 08      JSR ZBUFPTX   START STORING AT BEGINNING
08E9: BD 8C C0      LDA READ,X    START READING
08EC: 10 FB        BPL READ1     WAIT FOR VALID DATA
08EE: 91 26        STA (BUFPTX),Y SAVE IT
08F0: C8          INY          NEXT BUFFER POSITION
08F1: D0 F6        BNE READ1     NO PAGE CROSSING
08F3: E6 27        INC BUFPTX+1
08F5: A5 27        LDA BUFPTX+1
08F7: C9 80        CMP #$80
08F9: 90 EE        BCC READ1
08FB: 60          RTS
```

\*-----  
\* SEARCH FOR BYTES  
\*-----

```
08FC: A4 26      SEARCH LDY BUFPTX   FIX Y
08FE: A9 00      LDA #0       LSB=0!
0900: 85 26      STA BUFPTX
0902: A2 00      BYTE0 LDX #0       START WITH BYTE ZERO
0904: 18          CLC
0905: 20 21 08   JSR NXTBPTX
0908: B0 1D      BCS NO.MATCH END OF BUFFER REACHED, EXIT
090A: B1 26      LDA (BUFPTX),Y GET BYTE
090C: DD 00 03   CMP INBUFX,X MATCH?
090F: D0 F1      BNE BYTE0    NOPE, FIX X FOR FIRST CMP AGAIN
0911: 84 26      STY BUFPTX
0913: A0 00      LDY #0       COMPARE REMAINDER OF BYTES
0915: C8          NXT.CMP INY
0916: E8          INX          DONE?
0917: E4 FD      CPX NUMBYTES
0919: B0 09      BCS GOT.MATCH ALL BYTES MATCH, EXIT
091B: B1 26      LDA (BUFPTX),Y
091D: DD 00 03   CMP INBUFX,X
0920: F0 F3      BEQ NXT.CMP  MATCH, KEEP MATCHING
0922: D0 D8      BNE SEARCH   ..ALWAYS
0924: A9 01      GOT.MATCH LDA #1   ONE=GOT MATCH
0926: 2C          .HS 2C      SKIP NEXT TWO BYTES
0927: A9 00      NO.MATCH LDA #0    ZERO=NO MATCH FOUND
0929: 85 FC      STA STATUS   TELL BASIC
092B: 60          RTS
```

\*-----  
\* 4 PLUS 4 DECODE NEXT 8 BYTES  
\*-----

```
092C: A2 04      LDX #4       DECODES INTO 4 BYTES
092E: A0 00      LDY #0       START AT OFFSET ZERO
0930: B1 26      DEC1BYT LDA (BUFPTX),Y GET ONE HALF
0932: 38          SEC          ROL A ONE INTO LSB!
0933: 2A          ROL         EVERY OTHER BIT
0934: 85 FC      STA STATUS   TEMPORARY
0936: C8          INY         NEXT HALF
0937: B1 26      LDA (BUFPTX),Y
0939: 25 FC      AND STATUS
093B: 20 DA FD   JSR PRBYTE   PRINT IT
093E: A9 A0      LDA #$A0     FOLLOWED BY A SPACE
0940: 20 ED FD   JSR COUT
0943: C8          INY         NEXT POS
0944: CA          DEX         DONE?
0945: D0 E9      BNE DEC1BYT KEEP GOING
0947: 60          RTS
```

Continued on next page

## LOAD BAS.NIBBLER

Start up this part of the file

**RUN**

Finally, hit ESCape to leave the program and save the whole thing as one file

**SAVE THE.NIBBLER**

## Using The Nibbler

First, type any key to get past the title page. The Nibbler will next recalibrate the disk arm and read in track zero of whatever disk is in the drive. Look at the data it finds on this track by using the following commands:

“**I,J,K,M**” will move the cursor up, left, right and down respectively. These are just like the usual ESCape codes. If you press the CTRL key along with I,J,K or M, the cursor will jump to the extreme edge of the screen in the specified direction.

“**B**” will move the display to the beginning of the buffer.

“**E**” will move the display to the end of the buffer.

“**☐R**” will recalibrate the disk drive arm. Because of the way The Nibbler handles half tracks, it is recommended that you recalibrate the disk arm every time you change slot or drive numbers.

“**T**” will specify and read a new track. Half tracks are allowed by putting a five (5) after the decimal point.

“**S**” will specify a new slot number for reading disks. Again, it is recommended that you recalibrate the disk arm following the use of this option.

“**D**” will specify a new drive number for reading disks. Don’t forget that a recalibration is recommended here also.

“**\$**” will set up a search string. The Nibbler has two ways of searching for data. The first search method (unencoded) is invoked by pressing “**U**” when the cursor is between the parenthesis after the search string. In this search mode, The Nibbler will look for the exact bytes that you have specified as the search string. The use of this option is useful, for instance, if you want to find a series of sync bytes or some particular address or data markers.

The second search method (4 + 4 encoded) is invoked by pressing “**E**” when the cursor is between the parenthesis after the search string. In this search mode The Nibbler will look for some 4 + 4 encoded bytes that decode into the bytes you have specified in the search string. This is helpful for finding the start of address markers. You simply search for a track and sector number that has been 4 + 4 encoded. For example: if track \$11 has been read into the buffer, find the start of the track by specifying 11 00 as the search string. The bytes preceding it should be the start of address markers.

“**SPACE**” will instruct The Nibbler to find the next occurrence of the search string. If no match is found, you will hear a beep.

Using the search option, you may notice that

when the desired string is found, the cursor does not move. Instead, the data in the buffer is moved so that the string is placed under the cursor. I like to use this "feature" to more or less automatically find the end of data epilogues. To do this I first move the cursor so that it rests upon the third to the last byte in the first row of the data in the viewing window. Next, I instruct The Nibbler to search (unencoded) for the data prologue (usually D5 AA AD). Once the data prologue has been found, the data epilogue can be determined by looking at bytes 4 through 6 of the last line in the viewing window. This works because there are 342 bytes of encoded data between the data prologue and epilogue.

"4" will 4 + 4 decode the next eight bytes and print the results at the top of the screen. This is helpful if you want to examine the address field to determine the decoded volume, track, sector and checksum values.

"P" will print the current screen to a printer in slot one (1). Only the data that is currently on the screen will be dumped, not the entire contents of the buffer.

At any point in the program, you may press "ESC" to exit the current function. If you are not in a function (ex. typing in a new track number), then "ESC" will leave the program. Because of the way The Nibbler handles half tracks, it is recommended that you always leave the program via the ESCape key. Leaving the program any other way (ex. C) is not recommended because access to normal disks will be disabled.

## Modifying The Nibbler

Using the source code for The Nibbler, you may want to add some features of your own: displaying sync bytes and/or markers in inverse to the program. If you do reassemble the machine language portion of the program, keep in mind that the Applesoft portion of The Nibbler contains CALL's that will have to be modified to reflect the changes.

Happy nibbling!

## The Nibbler BASIC program

```

10 REM <<<<<<<<<<->>>>>>>>>
20 REM <<<<<<<<<<<<<<<<<>>>>>>>
30 REM <<<<<<<<<<<<<<<<<>>>>>>>
40 REM <<<<<<<<<<<<<<<<< THE >>>>>>>
50 REM <<<<<<<<<<<<<<<<< NIBBLER >>>>>>>
60 REM <<<<<<<<<<<<<<<<<>>>>>>>
70 REM <<<<<<<<<<<<<<<<<>>>>>>>
80 REM <<<<<<<<<<<<<<<<<>>>>>>>
90 REM <<<<<<<<<<<<<->>>>>>>>>
100 GOTO 70
110 GOSUB 900 : GOSUB 470
120 POKE 254 ,X : POKE 255 ,Y :AD=USR (AD) :AS
   =CHR$ ( PEEK ( - 16384 ) ) :X=PEEK ( 254 )
   :Y=PEEK ( 255 )
130 FOR A = W TO LEN (K$) : IF AS <> MID$ (K$ , A
   ,W) THEN NEXT : GOTO 120
140 ON A GOSUB 160 ,170 ,180 ,190 ,210 ,220 ,470
   ,1080 ,390 ,240 ,580 ,620 ,490 ,430 ,450
   ,270 ,970 ,970 ,1130 : GOTO 120
150 REM FAST CURSOR MOVEMENT
160 Y = 0 : RETURN
  
```

```

170 X = 0 : RETURN
180 X = 38 : RETURN
190 Y = 18 : RETURN
200 REM LEFT AND RIGHT ARROWS
210 AD = AD - 240 * (AD > 16862) : RETURN
220 AD = AD + 240 * (AD < 32024) : RETURN
230 REM 4 + 4 DECODE 8 BYTES
240 VTAB 3 : HTAB 29 : A = W + AD + Y * 20 + X / 2 :
   POKE 39 , A / 256
250 POKE 38 , A - PEEK (39) * 256 : CALL 2348 : POKE
   38 , 0 : RETURN
260 REM GET SEARCH STRING
270 VTAB W : FOR A = 0 TO 5
280 HTAB 25 + A * 2 : GOSUB 660 : IF AS = CH$ THEN
   A = A - (A > 0) : GOTO 280
290 IF AS = ESC$ THEN HTAB 25 : PRINT
   "0000000000000^U" ; : GOTO 950
300 IF AS = CM$ AND A = 0 THEN 280
310 IF AS = CM$ THEN 330
320 POKE 768 + A , B : NEXT
330 POKE 253 , A : PRINT TAB ( 38 ) "(^" CH$ ;
340 GET AS : IF AS <> "U" AND AS <> "E" AND AS <
   > ESC$ THEN 340
350 IF AS = ESC$ THEN 290
360 PRINT AS ; : IF AS = "U" THEN 390
370 CALL 2376
380 REM FIND NEXT SEARCH STRING
390 A = AD + Y * 20 + X / 2 + W : POKE 39 , A / 256 :
   POKE 38 , A - PEEK (39) * 256
400 CALL 2300 : IF PEEK (252) = 0 THEN PRINT CHR$
   ( 7 ) ; : RETURN
410 AD = PEEK (38) + PEEK (39) * 256 - Y * 20 - INT
   ( X / 2 ) - W : POKE 38 , 0 : RETURN
420 REM JUMP TO START OF BUFFER
430 AD = 16623 : RETURN
440 REM JUMP TO END OF BUFFER
450 AD = 32263 : RETURN
460 REM RECALIBRATE DISK ARM
470 POKE 47084 ,160 : CALL 2266 : POKE 47084 , 0
   : CALL 2266 : VTAB 3 : HTAB 8 : PRINT "00.0"
   : RETURN
480 REM GET NEW TRACK
490 VTAB 3 : HTAB 8 : S = PEEK (1287) : GOSUB 660
   : IF AS = ESC$ THEN POKE 1287 , S : RETURN
500 IF AS = CM$ AND B$ <> CM$ THEN CALL 2266 :
   RETURN
510 ON (AS = CH$) GOTO 490 : IF AS = CM$ THEN PRINT
   ". " ; : GOTO 540
520 HTAB 11 : GET AS : IF AS <> "0" AND AS <> "5"
   AND AS <> CH$ AND AS <> CM$ THEN 520
530 IF AS = CH$ THEN 490
540 IF AS = CM$ THEN AS = CHR$ ( PEEK (1290) - 128 )
550 PRINT AS ;
560 B = B * 2 + (AS = "5") : POKE 47084 , B : CALL
   2266 : RETURN
570 REM GET NEW SLOT NUMBER
580 VTAB W : HTAB 7 : GET AS : IF AS = CM$ OR AS =
   ESC$ THEN RETURN
590 IF AS < "1" OR AS > "7" THEN 580
600 PRINT AS : POKE 47081 , VAL (AS) * 16 : RETURN
610 REM GET NEW DRIVE NUMBER
620 VTAB 2 : HTAB 8 : GET AS : IF AS = ESC$ OR AS
   = CM$ THEN RETURN
630 IF AS < "1" OR AS > "2" THEN 620
640 PRINT AS : POKE 47082 , VAL (AS) : RETURN
650 REM GET A HEX NUMBER
660 B$ = "" : GET AS : IF AS = CH$ OR AS = CM$ OR AS
  
```

Continued on next page

## Continued from previous page

```

* -----
* DECODE (4+4) SEARCH STRING
* -----
0948: A2 00          LDX #0          ZERO X AND Y
094A: A0 00          LDY #0
094C: BD 00 03 ENC1BYT LDA INBUFF,X GET BYTE
094F: 4A           LSR           MAKE CORRECT BIT PATTERN
0950: 09 AA          ORA #$AA      ABCDEFGH BECOMES
0952: 99 10 03      STA INBUFF+$10,Y 1A1C1E1G + 1B1D1F1H
0955: C8           INY           NEXT STORAGE POSITION
-----
0956: BD 00 03      LDA INBUFF,X GET SAME BYTE AGAIN
0959: 09 AA          ORA #$AA      NO LSR THIS TIME
095B: 99 10 03      STA INBUFF+$10,Y SAVE IT
095E: C8           INY           NEXT STORAGE
095F: E8           INX           NEXT BYTE
0960: E4 FD          CPX NUMBYTES DONE?
0962: D0 E8          BNE ENC1BYT  NOPE, CONTINUE
0964: 06 FD          ASL NUMBYTES TWICE AS MANY BYTES TO SEARCH FOR
0966: A2 00          LDX #0        MOVE BYTES BACK TO $300
0968: BD 10 03 MOV1BYT LDA INBUFF+$10,X
096B: 9D 00 03      STA INBUFF,X
096E: E8           INX
096F: E4 FD          CPX NUMBYTES DONE?
0971: D0 F5          BNE MOV1BYT  NOPE, CONTINUE
0973: 60           RTS
-----
0974: 00           .HS 00
0975: 00 00          END.OBJ .HS 0000
  
```



Continued from previous page

```

= ESC$ THEN RETURN
670 IF (AS < "0" OR AS > "9") AND (AS < "A" OR AS
> "F") THEN 660
680 B = 16 * (ASC (AS) - 48 - 7 * (ASC (AS) > 64
)) : PRINT AS;
690 GET BS : IF BS = CH$ THEN PRINT BS; : GOTO 660
700 IF BS = ESC$ THEN AS = ESC$ : RETURN
710 IF BS = CM$ THEN PRINT CH$ "0" AS; : AS = BS : B
= B / 16 : RETURN
720 IF (BS < "0" OR BS > "9") AND (BS < "A" OR BS
> "F") THEN 690
730 B = B + ASC (BS) - 48 - 7 * (ASC (BS) > 64)
: PRINT BS; : RETURN
740 REM INITIALIZE VARIABLES
750 W = 1 : POKE 10 , 76 : POKE 11 , 45 : POKE 12 , 8
: PR# 0 : IN# 0 : HIMEM : 16383
760 KS = CHR$ (9) + CHR$ (10) + CHR$ (11) + CHR$
(13) + CHR$ (8) + CHR$ (21) + CHR$ (18) +
CHR$ (16)
770 KS = KS + "4SDTBES?/" + CHR$ (27) : AS(W)
= "*****" : FOR X = 2 TO 3
780 AS(X) = RIGHT$(AS(W) , X - W) + LEFT$(AS(W)
, 16 - X) : NEXT
790 AS(4) = AS(2) : AS(5) = AS(W) : AS(6) = AS(3)
800 FOR X = W TO 3 : FOR Y = W TO 3 : AS(X) = AS(X
) + CHR$ (10) + CHR$ (8) + MID$(AS(X) , Y
, W)
810 AS(X + 3) = MID$(RIGHT$(AS(X + 3) , Y) , W
, W) + CHR$ (8) + CHR$ (10) + AS(X + 3) : NEXT
: NEXT
820 CH$ = CHR$ (8) : CM$ = CHR$ (13) : ESC$ = CHR$
(27)
830 REM TITLE PAGE
840 TEXT : HOME : VTAB 3 : HTAB 15 : PRINT
"THE^NIBBLER" : VTAB 12 : HTAB 14
850 PRINT "BY^RAY^DARRAH" : VTAB 23 : HTAB 9 :
INVERSE : PRINT "PRESS^ANY^KEY^TO^BEGIN"
: NORMAL
860 FOR A = W TO 3 : VTAB W : HTAB 13 : PRINT AS(A
) : VTAB 2 : HTAB 13 : PRINT AS(A + 3)
870 IF PEEK (- 16384) < 128 THEN NEXT : GOTO 860
880 GET AS : GOTO 110
890 REM SCREEN SETUP
900 TEXT : HOME : FOR A = 0 TO 39 : POKE 1408 + A
, 32 : POKE 2000 + A , 32 : NEXT
910 PRINT "SLOT=>" PEEK (47081) / 16 : PRINT
"DRIVE=>" PEEK (47082)
920 PRINT "TRACK=>00.0" : VTAB W : HTAB 17 :
PRINT "SEARCH=>000000000000^U" : POKE
47084 , 0
930 PR# 0 : IN# 0 : VTAB 3 : HTAB 18 : PRINT
"?=HELP" : X = 18 : Y = 9
940 AD = 16623 : TK = 0 : POKE 47083 , 0 : POKE 47092
, 0 : POKE 47100 , 0
950 POKE 253 , 6 : FOR A = 0 TO 5 : POKE 768 + A , 0
: NEXT : RETURN
960 REM HELP SCREEN
970 POKE 34 , 4 : POKE 35 , 23 : HOME : POKE 35 , 24
: PRINT : PRINT SPC(6) "I , J , K , M" = ^CURSOR^
MOVEMENT"
980 PRINT "CTRL>^I , J , K , M" = ^MOVE^CURSOR^TO^E
DGES" : PRINT SPC(4) "<--^; ^-->^" = ^MOVE^
THROUGH^BUFFER^FAST" ;
990 PRINT SPC(12) "B" = ^JUMP^TO^START^OF^BUF
FER" : PRINT SPC(12) "E" = ^JUMP^TO^END^OF
^BUFFER" : PRINT
1000 PRINT SPC(7) "CTRL . R" = ^RECALIBRATE^DISK
^ARM" : PRINT SPC(12) "T" = ^READ^A^NEW^
TRACK"
1010 PRINT SPC(12) "S" = ^GET^NEW^SLOT^NUMBER"
: PRINT SPC(12) "D" = ^GET^NEW^DRIVE^NUM

```

```

BER" : PRINT
1020 PRINT SPC(12) "$" = ^DEFINE^SEARCH^STRING
" : PRINT SPC(6) "<SPACE>" = ^FIND^NEXT^S
EARCH^STRING"
1030 PRINT SPC(12) "4" = ^DECODE^8^BYTES^(4+4)
" : PRINT : PRINT SPC(7) "CTRL . P" = ^PRINT
OUT^OF^SCREEN"
1040 PRINT : PRINT SPC(10) "ESC" = ^EXIT^PROGR
AM^OR^FUNCTION" ;
1050 TEXT : WAIT - 16384 , 128 : IF PEEK (- 16384
) = 155 OR PEEK (- 16384) = 144 THEN GET AS
1060 ON (AS = CHR$ (16) ) GOTO 1080 : RETURN
1070 REM PRINT SCREEN
1080 PR# 1 : FOR A = W TO 3 : GOSUB 1100 : PRINT :
NEXT : PRINT
1090 FOR A = 5 TO 22 : GOSUB 1100 : PRINT : NEXT
: GOSUB 1100 : VTAB 1 : PRINT : PR# 0 : RETURN
1100 VTAB A : HTAB W : PRINT CHR$ (0) ; : B = PEEK
(40) + PEEK (41) * 256
1110 FOR B = B TO B + 39 : PRINT CHR$ ( PEEK (B)
) ; : NEXT : RETURN
1120 REM EXIT
1130 TEXT : HOME : PRINT "TO^RESTART" ; :
INVERSE : PRINT "THE^NIBBLER" ; : NORMAL
1140 PRINT " , ^TYPE" : PRINT : PRINT "JRUN" :
PRINT : CALL 1002 : POKE 47100 , W : POKE 103
, 1 : POKE 104 , 8 : END
1150 REM COPYRIGHT (C) 1985 , SOFKTEY PUB
LISHING

```

### The Nibbler Hexdump

```

0800: 00 0D 08 0A 00 8C 32 30 $C2F1
0808: 36 33 3A AC 00 00 00 A9 $106A
0810: 75 85 67 A9 09 85 68 60 $CAB1
0818: A0 00 84 26 A9 40 85 27 $BDD9
0820: 60 C8 D0 FC E6 27 A5 27 $6468
0828: C9 80 B0 EC 60 20 0C E1 $FB2B
0830: A5 A0 85 27 A4 A1 A9 03 $5046
0838: 85 25 20 8E FD 20 21 08 $A9EC
0840: A5 25 C9 17 B0 08 B1 26 $BBD9
0848: 20 DA FD 4C 3D 08 A5 FF $22B8

0850: 18 69 04 20 C1 FB A4 FE $7C2F
0858: A2 02 B1 28 48 29 3F 91 $4E7E
0860: 28 C8 CA D0 F5 AE 00 C0 $9DA6
0868: 10 FB 8D 10 C0 88 68 91 $758C
0870: 28 88 68 91 28 E0 C9 D0 $6271
0878: 38 C6 FF 10 D1 E6 FF A5 $9F46
0880: A0 C9 41 90 C9 A5 A1 E9 $EE75
0888: 14 85 A1 B0 A3 C6 A0 D0 $6202
0890: 9F E0 CD D0 3E E6 FF A5 $66B8
0898: FF C9 13 90 B1 C6 FF A5 $FFA7

08A0: A0 C9 7E B0 A9 A5 A1 69 $539F
08A8: 14 85 A1 90 83 E6 A0 D0 $BFC8
08B0: DE E0 CA D0 0C 88 88 84 $FBB4
08B8: FE 10 93 A0 26 84 FE D0 $F38F
08C0: B8 E0 CB D0 CC C8 C8 84 $5636
08C8: FE C0 27 90 81 A0 00 84 $E4DA
08D0: FE B0 C2 A5 A0 A4 A1 4C $F56F
08D8: F2 E2 20 E3 03 20 D9 03 $CB76
08E0: AE E9 B7 BD 8E C0 20 18 $0E79
08E8: 08 BD 8C C0 10 FB 91 26 $9450

08F0: C8 D0 F6 E6 27 A5 27 C9 $1B67
08F8: 80 90 EE 60 A4 26 A9 00 $A807
0900: 85 26 A2 00 18 20 21 08 $6D92
0908: B0 1D B1 26 DD 00 03 D0 $212A
0910: F1 84 26 A0 00 C8 E8 E4 $2A01
0918: FD B0 09 B1 26 DD 00 03 $CEB8
0920: F0 F3 D0 D8 A9 01 2C A9 $99A0
0928: 00 85 FC 60 A2 04 A0 00 $630E
0930: B1 26 38 2A 85 FC C8 B1 $CAF6

```

```

0938: 26 25 FC 20 DA FD A9 A0 $3C7F
0940: 20 ED FD C8 CA D0 E9 60 $6E5F
0948: A2 00 A0 00 BD 00 03 4A $5C9C
0950: 09 AA 99 10 03 C8 BD 00 $7FA9
0958: 03 09 AA 99 10 03 C8 E8 $BE0B
0960: E4 FD D0 E8 06 FD A2 00 $824D
0968: BD 10 03 9D 00 03 E8 E4 $6B30
0970: FD D0 F5 60 00 00 00 $D82D

```

### The Nibbler BASIC Checksums

10	- \$BADD	590	- \$1DC8
20	- \$9B13	600	- \$4CA3
30	- \$4D3B	610	- \$7014
40	- \$AD92	620	- \$412F
50	- \$C899	630	- \$F0D9
60	- \$FF65	640	- \$00F0
70	- \$A3BF	650	- \$AF6A
80	- \$A900	660	- \$51EB
90	- \$924D	670	- \$A48F
100	- \$5E1C	680	- \$ED9A
110	- \$27E0	690	- \$2CC3
120	- \$7F5C	700	- \$47E0
130	- \$0E41	710	- \$93E7
140	- \$7BA9	720	- \$D016
150	- \$3F7B	730	- \$4E68
160	- \$96C7	740	- \$9FC7
170	- \$FF5B	750	- \$C442
180	- \$D478	760	- \$8BCE
190	- \$2DB6	770	- \$82EC
200	- \$94D8	780	- \$6D12
210	- \$2846	790	- \$C700
220	- \$729F	800	- \$049E
230	- \$59E2	810	- \$B4E9
240	- \$F454	820	- \$4745
250	- \$0E1B	830	- \$2468
260	- \$A7D2	840	- \$DACE
270	- \$2916	850	- \$5121
280	- \$4459	860	- \$4658
290	- \$FA35	870	- \$DA89
300	- \$7628	880	- \$4BAD
310	- \$E897	890	- \$1C2E
320	- \$5B89	900	- \$76A7
330	- \$219E	910	- \$484D
340	- \$8A8F	920	- \$16C1
350	- \$3F11	930	- \$CF67
360	- \$C42C	940	- \$E51C
370	- \$6105	950	- \$95A0
380	- \$59F1	960	- \$30BB
390	- \$196B	970	- \$A5FB
400	- \$1DEA	980	- \$C556
410	- \$55F6	990	- \$7655
420	- \$A62F	1000	- \$2F7C
430	- \$22A4	1010	- \$0BAE
440	- \$7416	1020	- \$99E9
450	- \$D2E6	1030	- \$C293
460	- \$6E70	1040	- \$0FA7
470	- \$B1A1	1050	- \$A83E
480	- \$9D5F	1060	- \$37A1
490	- \$A133	1070	- \$9D3F
500	- \$767C	1080	- \$5F86
510	- \$9794	1090	- \$18F3
520	- \$8F8C	1100	- \$A9C6
530	- \$8BBB	1110	- \$F880
540	- \$A971	1120	- \$7835
550	- \$67F7	1130	- \$3960
560	- \$F7DB	1140	- \$6AE3
570	- \$502F	1150	- \$7F1A
580	- \$800E		

Get  
25  
SS/SD  
5 1/4"  
diskettes

(ANSI Spec  
100% guaranteed,  
with reinforced hubs.  
Tyvek sleeves and  
write-protect tabs  
included.)

For Only  
**\$18.75**

**That's only  
75¢ per disk!!**  
(Sold in multiples  
of 25 ONLY.)

Please send me \_\_\_\_\_ sets (25 disks per set) of 5 1/4" SS/SD diskettes. I have enclosed \$18.75 plus shipping and handling for each set.

Name \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ St \_\_\_\_\_ Zip \_\_\_\_\_

Country \_\_\_\_\_

VISA/MC \_\_\_\_\_ Exp \_\_\_\_\_

Signature \_\_\_\_\_

Send your order to: Disk Offer, SoftKey Publishing, PO Box 110816, Tacoma, WA 98411. Please enclose \$3 shipping and handling for the first set plus \$1 per additional set. Foreign orders add 20% shipping and handling. US funds drawn on US bank. Washington State residents add 7.8% sales tax. Orders shipped via UPS. Please use street address.

**Moving  
Soon?**

Let us know your new address **at least 30 days in advance** so you won't miss a single issue of Hardcore COMPUTIST (issues missed due to non-receipt of change of address may be obtained at the current back issue rate).

**Fill out this form, paste your present address label in the space provided, and send it to:**

Hardcore COMPUTIST, Subscription Department  
PO Box 110846-T, Tacoma, WA 98411

**My new address is:**

Address \_\_\_\_\_

City \_\_\_\_\_ St \_\_\_\_\_ Zip \_\_\_\_\_

Phone \_\_\_\_\_

**Paste present address label here or fill in old address:**

Name \_\_\_\_\_ ID # \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ St \_\_\_\_\_ Zip \_\_\_\_\_

Country \_\_\_\_\_

Phone \_\_\_\_\_

**What does Hardcore COMPUTIST offer its readers?**

More of what they need to know about backing up Apple software than ANY OTHER magazine on the market.

**Why should YOU subscribe?**

Because any *serious* Apple computer user knows he can get MORE out of the software he buys using the advanced deprotection techniques found in each issue of Hardcore COMPUTIST.

- \* **MORE backups!**
- \* **MORE modifications!**

**DON'T  
USE  
your  
valuable  
software  
without FIRST  
backing-it-up.**

INSTEAD, rely on Hardcore COMPUTIST to provide you with up-to-date deprotection methods for all the new games and programs available for the Apple ][ line of computers.

**Annual Rates: Please check one**

- U.S. .... \$25
- Canada, U.S. 1st Class ..... \$34
- Mexico ..... \$39
- Foreign Airmail ..... \$60
- Foreign surface mail ..... \$40
- SAMPLE, U.S. .... \$3.50
- SAMPLE, Foreign ..... \$4.50

( ) Yes, start my subscription now.  
( ) I would like to RENEW my subscription. (Please paste PRESENT MAILING LABEL below)

Name \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ St \_\_\_\_\_ Zip \_\_\_\_\_

Phone \_\_\_\_\_

VISA/MC \_\_\_\_\_ Exp \_\_\_\_\_

Signature \_\_\_\_\_

Send check or money order (US funds drawn on US bank) to: Hardcore COMPUTIST, PO Box 110846-T, Tacoma, WA 98411

**COMPUTIST**

**Hardcore**

**Apple II, II+, IIe,  
Franklin users:**

Do you have problems  
backing-up your  
copy-protected programs?  
Do you lack parameters for  
your copy programs?  
Are you looking for programs  
that you can AFFORD?  
Are you hesitating to  
upgrade your equipment due to  
expensive prices  
quoted in other ads?

**It's simple now.  
Just drop us a line.**

Send \$1.00 U.S. funds to:

**Reliant  
P.O. Box 33610  
Sheungwan, Hong Kong**

**IMPORTANT:** We have over 1000 PC name-brand programs and various hardware offers. Programs @ U.S. \$8.00/PC includes the disk and registered air-mail handling.

**ENHANCE your Apple!  
with a**

**65C02**

Upgrade to this  
pin-for-pin compatible CMOS version  
of the 6502 microprocessor.

*Advantages of the 65C02:*

- ▶ 27 NEW Machine Language Opcodes
- ▶ 8 Completely NEW Machine Language Instructions
- ▶ Low Power CMOS Design
- ▶ Invalid Opcodes Default to 1 Cycle NOP's
- ▶ The JMP(XXFF) Bug Has Been Fixed

**Only  
\$9.95**

These chips are the 2MHZ versions manufactured by Western Design Center of Mesa, AZ. Used in the Apple IIc, the 65C02 microprocessor is also included in the Apple IIe upgrade package recently released by Apple Computer.

Each order will include a data sheet and installation instructions.

**PLACE YOUR ORDERS NOW!**

Offer expires May 31

Please send \_\_\_\_\_ 65C02 chip(s) to the address below.  
I have enclosed \$9.95 plus shipping and handling for each chip.

Name \_\_\_\_\_

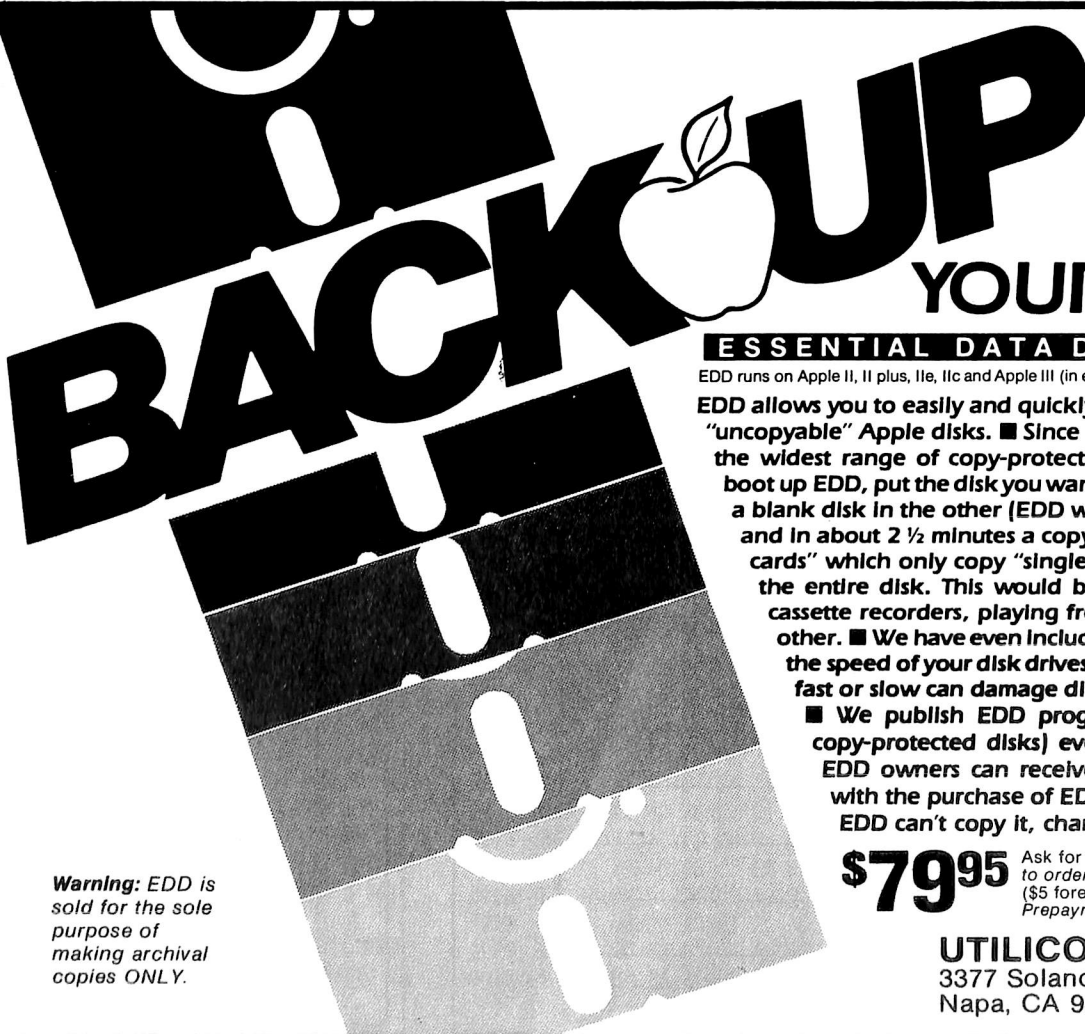
Address \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

VISA/MC \_\_\_\_\_ - \_\_\_\_\_ - \_\_\_\_\_ Exp \_\_\_\_\_

Signature \_\_\_\_\_

Domestic orders please enclose \$1.00 shipping and handling for the first chip + .25 per additional chip; foreign orders add 20%. US funds drawn on US bank. Washington state residents add 7.8% sales tax. Please allow 6 weeks for delivery. Send order to: Chip Offer, Hardcore COMPUTIST, PO Box 110846-C, Tacoma, WA 98411.



**NOW  
AVAILABLE  
AT YOUR LOCAL  
COMPUTER STORE**

**YOUR DISKS**

**ESSENTIAL DATA DUPLICATOR III™**

EDD runs on Apple II, II plus, IIe, IIc and Apple III (in emulation mode) using one or two disk drives.

EDD allows you to easily and quickly make back up copies of your "uncopyable" Apple disks. ■ Since EDD has been preset to copy the widest range of copy-protections possible, you just simply boot up EDD, put the disk you want to copy in one disk drive and a blank disk in the other (EDD will work using one drive also) and in about 2 1/2 minutes a copy is made. ■ Unlike the "copy-cards" which only copy "single load" programs, EDD copies the entire disk. This would be similar to hooking up two cassette recorders, playing from one, and recording to the other. ■ We have even included an option so you can check the speed of your disk drives because drive speeds running fast or slow can damage disks and cause other problems. ■ We publish EDD program lists (information about copy-protected disks) every couple of months, which EDD owners can receive. The current list is included with the purchase of EDD. ■ The bottom line is this; if EDD can't copy it, chances are nothing will.

**\$79.95**

Ask for EDD at your local computer store, or, to order direct; send \$79.95 plus \$2 shipping (\$5 foreign). Mastercard/Visa accepted. Prepayment required.

**UTILICO MICROWARE**  
3377 Solano Ave., Suite #352  
Napa, CA 94558 (707)257-2420

**Warning:** EDD is sold for the sole purpose of making archival copies ONLY.



*By Hackers  
For Hackers*

- ELITE BOARD DOWNLOADS
- CRACKING TIPS
- PHREAKING SECTION
- GAME CHEATS
- PARMS
- PROGRAMS
- INTERVIEWS
- ADVENTURE TRIPS
- HACKING TIPS
- MYSTERY SECTIONS

Published on both sides of  
an Apple diskette -  
4 times a year.

# The BOOT-LEGGER MAGAZINE

**Subscribe Now!**

Send 25 Bucks for a 1-Year Subscription  
**THE BOOT LEGGER**, 3310 Holland Loop  
Road, Cave Junction, Oregon 97523.  
Overseas Subscriptions \$50.  
Canadian \$30 U.S. Currency.

FOR AD INFO. & QUESTIONS  
CALL BOOTLEG AT (503) 592-4461

# SIMPLY SOFTWARE

DISCOUNT SPECIALS ON APPLE SOFTWARE • ALL PRICES 30% OFF

	LIST	SPECIAL				LIST	SPECIAL
Pixit Shape Library #1	\$20.00	<b>\$14.00</b>	Matchmaker U.S. Geography Facts	Welcom Aboard Willy Byte in the Digital Dim.		\$129.95	<b>\$103.96</b>
Pixit Shape Library #2			Matchmaker Vocab. Word Builders	Winnie the Pooh Dimension		\$ 75.00	<b>\$ 52.50</b>
Multiple Labels	LIST \$24.95	<b>SPECIAL \$17.47</b>				\$ 26.95	<b>\$ 18.87</b>
Vaults of Zurich						\$ 74.95	<b>\$ 52.47</b>
Arcade Album	LIST \$29.95	<b>SPECIAL \$20.97</b>	Heist	Pitstop II		\$195.00	<b>\$136.50</b>
Eliminator			Miner 2049er	Robots of Dawn		\$195.00	<b>\$136.50</b>
Figures and Formulas						\$199.00	<b>\$139.30</b>
Addition Sequences	LIST \$34.95	<b>SPECIAL \$24.47</b>	MACH II Joystick	Mission Algebra		\$495.00	<b>\$346.50</b>
BC's Quest for Tires				Paddlestick		\$299.00	<b>\$209.30</b>
Division Sequences						\$399.00	<b>\$279.30</b>
NATO Commander			AST Purchase Order	Pixit		\$499.00	<b>\$349.30</b>
Pathwords			Blazing Paddles	Return of Werdna		\$599.00	<b>\$419.30</b>
Percents Sequences			Millionaire	Wilderness		\$599.00	<b>\$419.30</b>
Queen of Phobos						\$699.00	<b>\$489.30</b>
Turbo Tutor			Breakthru in the Ardennes	Inspector		\$ 45.00	<b>\$ 31.50</b>
Agent U.S.A.	LIST \$39.95	<b>SPECIAL \$27.97</b>	Forecast	Take I		\$ 74.95	<b>\$ 52.47</b>
FactsBrain Game						\$495.00	<b>\$346.50</b>
Conan						\$369.00	<b>\$258.30</b>
Copy II +			ARCO Computer Prep. for SAT	Interior Design		\$ 99.00	<b>\$ 69.30</b>
Dinosaurs			Bank Street Filer	Landscape Design		\$ 30.00	<b>\$ 21.00</b>
Donald Duck's Playground				Magic Words		\$149.95	<b>\$104.97</b>
Goofy's Word Factory			Beginning Algebra	Bank Street Mailer		\$425.00	<b>\$297.50</b>
Imperium Galactium						\$149.00	<b>\$104.30</b>
Master Type			LIST \$99.95	<b>SPECIAL \$69.97</b>		\$ 26.95	<b>\$ 18.87</b>
Matchmaker Grammer			Alpha Type Faces	F.A.S.T.		\$250.00	<b>\$175.00</b>
Skills			Elite Controller	Professional Architectural Design		\$ 28.00	<b>\$ 19.60</b>
Matchmaker Spanish			Monte Carlo Simulations			\$319.00	<b>\$223.30</b>
Skills						\$175.00	<b>\$122.50</b>
						\$299.00	<b>\$209.30</b>
						\$ 79.95	<b>\$ 55.97</b>

PLEASE make check or M.O. payable to: **Simply Software Inc.** • P.O. Box 36068 • Kansas City, Missouri 64111  
Add \$3.00 shipping, Missouri residents add 6 1/8% sales tax. Allow 4-6 weeks for delivery.

# DISCOUNTS!

## 5 1/4 DISKETTES & STORAGE

- SS/DD, BOX OF 10 ..... \$10.00 \*
- SS/DD, 10 BOXES ..... \$89.00
- DOUBLE NOTCHED DS/DD, BOX OF 10 ..... \$14.00
- DOUBLE-NOTCHED DS/DD, 100 ..... \$125.00
- HARD PLASTIC STAND-UP DISKETTE LIBRARY CASES **\$2.75 EACH**  
**4 FOR \$10.00**  
(specify color choices beige black blue green grey red)
- SMOKED PLASTIC JUMBO-SIZE FLIP-TOP 75 DISKETTE FILE CASES ..... \$12.00 \*
- 140-DISKETTE LOCKING WOOD FILE CABINET ..... \$33.00

## PRINTERS

- PANASONIC P1090 ..... \$199.00
- PANASONIC P1091 ..... \$299.00
- PANASONIC P1092 ..... \$439.00
- EPSON RX-100 ..... \$439.00
- OKI-DATA MICROLINE 92A DOT MATRIX ..... \$369.00
- SILVER REED 400 LETTER QUALITY ..... \$269.00
- CITIZEN MSP-10 ..... \$359.00 \*
- CITIZEN MSP-15 ..... \$549.00 \*
- CANON PW1080 ..... \$339.00 \*

## PRINTER INTERFACES AND ACCESSORIES

- STANDARD PARALLEL INTERFACE CARD ..... \$49.00
- APPLE IIc TO PARALLEL/ GRAPHICS INTERFACE .... \$99.00
- GRAPHICS PARALLEL INTERFACE CARD ..... \$75.00
- FINGERPRINT PUSH-BUTTON GRAPHICS CARD ..... \$119.00
- MICROFAZER GENERAL PRINT BUFFER ..... \$149.00
- PRINTER STAND ..... \$14.00
- SWITCH BOX
- 3 PARALLEL PORTS ..... \$99.00
- SWITCH BOX
- 3 SERIAL PORTS ..... \$79.00
- 2700 SHEETS OF PAPER .... \$31.00 \*

## FLOPPY DISK DRIVES

- FOURTH DIMENSION (FULL OR SLIMLINE) ..... \$159.00
- DISTAR ..... \$149.00
- LASER/MITAC ..... \$139.00
- GAMMA ..... \$129.00
- IIc CABLE FOR ABOVE DRIVES ..... \$20.00 \*
- DISK CONTROLLER ..... \$59.00
- DOUBLE SIDED DRIVE ..... \$199.00
- 2-DRIVE SET ..... \$249.00

## HARD DISK DRIVES

- 10 MEGABYTE WITH CONTROLLER & SOFTWARE ..... \$995.00 \*

## MONITORS

- GORILLA 12-INCH GREEN ..... \$84.00
- USI 12-INCH GREEN ..... \$94.00
- USI 12-INCH AMBER ..... \$99.00
- INTRA 14-INCH COMPOSITE COLOR/80 COLUMN ..... \$199.00
- PANASONIC 1300 COMPOSITE/RGB ..... \$239.00 \*

## MODEMS

- ZOOM TELEPHONICS 300-BAUD ..... \$109.00
- IIc MODEM W/SOFTWARE ..... \$159.00
- CENTAURI 300 BAUD ..... \$159.00
- PRO-MODEM 1200 ..... \$349.00
- PRO-MODEM 1200A INTERNAL ..... \$299.00

## GRAPHICS DEVICES

- POWER PAD & STARTER KIT ..... \$129.00

## VIDEO & DISPLAY EQUIPMENT

- DIGITIZER ..... \$299.00
- B & W CAMERA ..... \$195.00

## GENERAL ITEMS

- 6-OUTLET POWER STRIP ..... \$19.00
- 6-OUTLET WITH SURGE PROTECT ..... \$25.00
- SURGE PROTECTOR ..... \$11.00
- RF MODULATOR ..... \$49.00
- COMPUTER STAND ..... \$24.00
- MODEM ELIMINATOR CABLE ..... \$21.00

## GAME I/O DEVICES

- CH PADDLE STICKS ..... \$37.00
- CH MACH II JOYSTICK ..... \$37.00
- CH MACH III JOYSTICK ..... \$45.00

\* DENOTES NEW PRICE OR ITEM

LONG DISTANCE • CALL TOLL FREE WITH TOUCH TONE PHONE. DIAL 950-1088. WAIT FOR TONE. DIAL 363-1313. NOTE: IF 950-1088 DOES NOT WORK IN YOUR LOCATION, CALL 1-800-446-4462. WAIT FOR TONE. DIAL 363-1313.

## SLOT EXPANSION

- 16 RAM CARD ..... \$49.00
- SEHI-ALL SERIAL INTERFACE CARD ..... \$119.00
- CENTAURI APS Z-80 CARD ..... \$59.00
- QUICK-LOADER PROM BOARD ..... \$149.00
- MULTIPLE-SLOT EXPANSION CHASSIS ..... \$149.00
- SINGLE-SLOT EXTENDER ..... \$29.00
- WILDCARD II COPY BOARD ..... \$109.00
- PROM BURNER ..... \$119.00

## SPECIAL PERIPHERALS

- COOLING FAN WITH SURGE PROTECTOR ..... \$39.00
- LIFETIME POWER SUPPLY ..... \$179.00
- SHIFT KEY MOD KIT ..... \$8.00
- SCREEN SWITCHER/ DRIVE STEPPER ..... \$74.00

## APPLE SOFTWARE

- WORD STAR ..... \$99.00 \*
- MAIL MERGE/SPELL STAR/ STAR INDEX ..... \$125.00
- INCOME TAX PREP ..... \$199.00
- PRINT SHOP ..... \$39.00 \*

**CLOSEOUT  
SPECIAL  
WORDSTAR  
\$99<sup>00</sup>**

### STORE HOURS:

12-8 M-TH  
12-6 FRI  
11-5 SAT



(301) 652-4232

**V ASSOCIATES**

8231 Woodmont Ave., Bethesda, MD/20814

UPS shipping,  
\$4.00 per order  
plus \$6.00  
per printer  
or monitor