

For The Serious User Of Apple][Computers

Hardcore

COMPUTIST

Issue No. 20 \$3.75

Apple][Boot ROM
Disassembly

Pg. 9

The Graphic
Grabber v3.0

Pg. 15

An Improved
BASIC/Binary
Combo

Pg. 25



Softkey for: The Wizardrys

- I. Proving Grounds of the Mad Overlord
- II. Knight of Diamonds

Pg. 27

Hardcore COMPUTIST
PO Box 110846-T
Tacoma, WA 98411

20 ID#NJ 24 7 issue(s) left in SUB
RON LUKACS
10 BELL AVENUE
FORDS NJ 08863

PEEL HERE

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

SIMPLY SOFTWARE

DISCOUNT SPECIALS ON APPLE SOFTWARE • ALL PRICES 30% OFF

LIST \$20.00 FONTRIX Art & Tech. App. FONTRIX Asst. Fancy Sets FONTRIX Bus. & Corr. Sets FONTRIX Headline & Dec. Sets	SPECIAL \$14.00 FONTRIX Foreign Lang. Alphabets FONTRIX Letter Forms & Poster Sets Tip Disk #1 Typefaces	LIST \$39.95 Brain Game Build a Book About You Cash Receipts Journal Chipwits Compuzzler Copy II+ D Code Expense Journal/Dist. Game Show	SPECIAL \$27.97 Learn to Type Lucky's Magic Hat Master Match Mines of Qyntarr Reading Comprehension Skills II Tic-Tac-Show Uptown Double Crostics Welcome Aboard Willy Byte in the Digital Dimension	LIST \$129.95 ASCII Express Pro DOS Basic Building Blocks BPI Job Cost BPI Speed Reading Creative Writer Filer Calc Data Factory 6.0 Dragon Fire Dreadnaughts Fontrix Free Trader Get Rich! Strategies Home Accountant Impossible Mission Legal Office Manager Lucky's Foreign Language Mach II Joystick Magic Office System Magic Window II Magicalc Mastering the GRE Original Boston Computer Diet S-C Macro Assembler Sensible Speller for Pro DOS Sideways Silicon Salad Tax Advantage Tax Manager Tax Preparer 1985 Tax Preparer California Supplement Telengard Trivia Fever Vol. 2 Trivia Savant Turbo Toolbox Versaform	SPECIAL \$90.97 \$ 55.97 \$416.15 \$ 69.30 \$ 97.97 \$139.30 \$ 13.97 \$ 21.00 \$ 52.50 \$ 17.50 \$ 34.97 \$ 52.47 \$ 24.50 \$174.30 \$ 13.97 \$ 31.47 \$206.50 \$104.97 \$104.97 \$110.60 \$ 55.97 \$ 70.00 \$ 87.50 \$ 42.00 \$ 17.47 \$ 48.97 \$126.00 \$175.00 \$ 66.50 \$ 19.60 \$ 17.47 \$ 31.47 \$ 38.47 \$ 48.30
LIST \$26.95 Alice in Wonderland Below the Root	SPECIAL \$18.87 Swiss Family Robinson Treasure Island Wizard of Oz	LIST \$40.00 Heist Robots of Dawn	SPECIAL \$28.00 Rogue Summer Games II		
LIST \$29.50 Beagle Bag Diskquik	SPECIAL \$20.65 Flex Type Frameup Utility City	LIST \$49.95 Barron Decimals Made Simple Get Rich! Insurance Planning Get Rich! Real Estate Planning Get Rich! Retirement and Estate Planning Mastering Units of Measurement	SPECIAL \$34.97 Newsroom Ortho's Gardening Print Shop Questron Reading Keys Return of Werdna Tycoon Uptown Trivia Molecules and Atoms		
LIST \$29.95 Gumball I.O. Silver	SPECIAL \$20.97 Pronto DOS Trivia Fever Super Sports	LIST \$59.95 Arcade Machine	SPECIAL \$41.97 Dazzle Draw Magic Memory		
LIST \$34.95 Championship Lode Runner Defender Dig Dug Double Take Drol F-15 Strike Eagle Felony Gemstone Warrior Karateka Lode Runner	SPECIAL \$24.47 Math Run Rescue Raiders Spare Change Stargate Step by Step Keyboard Sword of Kadash Transylvania Turbo Tutor Universal File Conversion Word Scrambler/Spelling Tutor	LIST \$395.00 BPI Accounts Payable BPI Accounts Receivable	SPECIAL \$276.50 BPI Inventory Control BPI Payroll		

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.

BACK UP YOUR DISKS

NOW
AVAILABLE
AT YOUR LOCAL
COMPUTER STORE

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.

REMEMBER!
Always
be
prepared
whenever
you
enter the
adventure-
filled world
of...

EAMON



Six different text-adventures
for the Apple.
All for only \$20.00!

Please send me the EAMON Introductory Offer. I have enclosed \$20.00 plus applicable sales tax and shipping & handling. I understand that Public Domain Software is not commercial quality and is supplied as-is and that orders are filled on double-sided disks.

Name _____
Address _____
City _____ State _____ Zip _____
Country _____ Phone _____
VISA/MC _____ Exp _____

Signature _____
Send check or money order (US funds drawn on US bank) to: Computer Learning Center, PO Box 110876-HC, Tacoma, WA 98411. Washington residents add 7.8% sales tax. Foreign orders add 20% shipping and handling.

Don't **TYPE IN**
programs that appear in
Hardcore **COMPUTIST**.

Order the Library Disk, instead!

Each month a Library Disk with all the programs that appeared in the previous issue of Hardcore COMPUTIST is prepared for **SMART READERS** like you who have *better* things to do with their time than type in program listings.

Disk No. 19 for Hardcore COMPUTIST 19 includes controllers for Rendezvous with Rama and HSD Stats Series, Better F8 ROM source code, Make New ROM, and Nibbler source code and BASIC program.

Disk No. 18 for Hardcore COMPUTIST 18 includes controller for Bank Street Writer, FreSec 1 & FreSec 2 (source code for Applewriter //e softkey), Checksoft source code, VTOC Mover source code, and RK Copy source code and BASIC program.

Disk No. 17 for Hardcore COMPUTIST 17 includes controllers for Skyfox, Zaxxon, The Print Shop and Crossword Magic, Super Controller, Graphic Grabber source code and BASIC program, & Lone Catalog Arranger Part II source code and BASIC program.

Disk No. 16 for Hardcore COMPUTIST 16 includes controllers for Sheila, AceCalc (for Artsci programs) and Beyond Castle Wolfenstein (revised version), Controller Writer BASIC program, and Lone Catalog Arranger Part I source code and BASIC program.

Disk No. 15 for Hardcore COMPUTIST 15 includes controllers for Master Type and Disk Head Move (Whiz Kid), and Boot From Drive 2 source code and BASIC program.

Disk No. 14 for Hardcore COMPUTIST 14 includes controllers for SAT, Sea Dragon, Batman Decoder Ring, Rocky's Boots and Infocom, Super IOB v1.2 (Standard & Swap controllers) and BASIC program.

Disk No. 10 for Hardcore COMPUTIST 13 includes controllers for Penguin Software, Snooper Troops, Electronic Arts and DLM Software, CSavers source code, New DOS 3.3 Command (MREAD/MWRT) source code and BASIC program.

Disk No. 9 for Hardcore COMPUTIST 12 includes controller for Lion's Share, The Armonitor source code, Zoom Grafix BASIC program, CORE Disk Searcher source code and BASIC program, Psychedelic Symphony source code and BASIC program.

Disk No. 8 for Hardcore COMPUTIST 11 includes controller for Ultima III, Ultimaker III source code and BASIC program, Ultimapper.large and Ultimapper.small (BASIC listings), and Sensible Speller source code.

Disk No. 7 for Hardcore COMPUTIST 10 includes controllers for Arcade Machine and Minit Man, Sensible Speller Boot 2 source code, MakeSaver, ApplEar source code, Bank Street Writer Emulation program, Crunchlist II source code.

Disk No. 3 for CORE Games issue includes Destructive Forces and Dragon Dungeon.

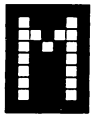
Disk No. 2 includes programs etc. for CORE Utilities issue and Hardcore COMPUTIST 3 & 4.

Disk No. 1 includes programs etc. for CORE Graphics issue and Hardcore COMPUTIST 1 & 2.

Get each
Library Disk
for the
low price of
\$9.95

Please send me the Library Disk(s) I have marked below. I have enclosed \$9.95 (plus applicable tax & shipping) for each disk.

<input type="checkbox"/> Disk No. 19	Name _____
<input type="checkbox"/> Disk No. 18	Address _____
<input type="checkbox"/> Disk No. 17	City _____ St _____ Zip _____
<input type="checkbox"/> Disk No. 16	Phone _____
<input type="checkbox"/> Disk No. 15	VISA/MC _____ Exp _____
<input type="checkbox"/> Disk No. 14	Signature _____
<input type="checkbox"/> Disk No. 10	Send check or money order to: Hardcore COMPUTIST, PO Box 110846-1, Tacoma, WA 98411. Most orders shipped UPS; please use street address. Washington state residents add 7.8% sales tax. Foreign orders add 20% shipping & handling. US funds drawn on US bank.
<input type="checkbox"/> Disk No. 9	SPECIAL! Order 5 Library Disks (of your choice) and pay only \$40.00.
<input type="checkbox"/> Disk No. 8	
<input type="checkbox"/> Disk No. 7	
<input type="checkbox"/> Disk No. 3	
<input type="checkbox"/> Disk No. 2	
<input type="checkbox"/> Disk No. 1	



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:

6 **P**

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 II 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 II Plus, Locksmith or The Essential Data Duplicator
- 6) **Text Editor** capable of producing normal sequential text files such as Applewriter II, Magic Window II or Screenwriter II.

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 II Plus - Apple IIe - 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 II 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 IIe - Apple IIc: 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 IIc will void its warranty but the increased ability to remove copy protection may justify it.

Recommended Literature: The Apple II 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 Leichner, 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 **CMY**.

xxx.xxx **CMY**

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 II and Apple II 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.

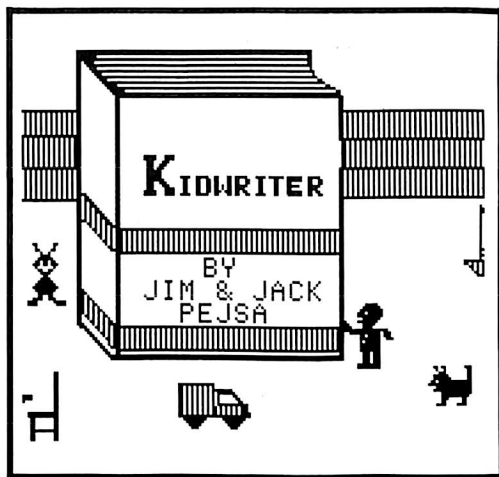
Hardcore COMPUTIST

Publisher/Editor: Charles R. Haight Technical Editors: Gary Peterson, Ray Darrah

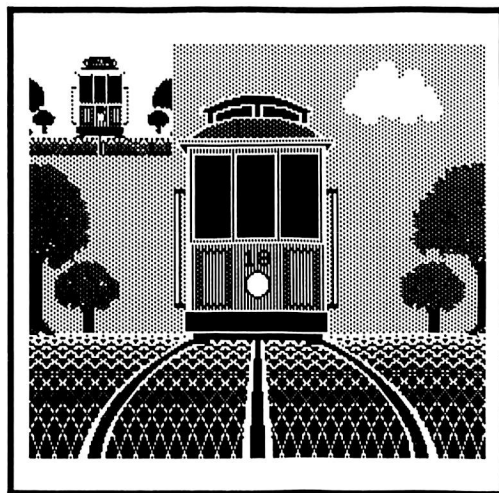
Production & Graphics: Lynn Campos-Johnson Circulation: Michelle Frank

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. 6



Pg. 15

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. 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 Apple II Boot ROM Disassembly

An in-depth look at the boot ROM, this article explains the 6 & 2 encoding method and the use of the disk controller's hardware addresses. *By Ray Darrah & Gary Peterson.*

15 The Graphic Grabber v3.0

If you liked "The Graphic Grabber" for the Print Shop in Hardcore COMPUTIST No. 17, you'll love "The Graphic Grabber v3.0". This update introduces several new commands and the ability to shrink hi-res areas to fit into the graphic window used by The Print Shop. *By Ray Darrah III.*

22 Copy II+ 5.0: A Review

Hardcore COMPUTIST takes out the magnifying glass and scrutinizes the latest version of Copy II Plus. *By Dr. Phillip Romine.*

24 The Know-Drive- A Hardware Evaluation

A look at this relatively unknown Titan/Legend compatible RAM card (from Abacus Enterprises, Inc.) that can simultaneously hold up to three application programs. *By Clay Harrell.*

25 An Improved BASIC/Binary Combo

"Where in the Apple can I safely put my machine code?" you ask. How about between DOS and its buffers? Read this article and put some little-used DOS space to good use. *By James L. Parham.*

CORE SECTION

26 Deprotecting Sargon III

Super IOB mows down another "unbeatable" protection scheme. And, with a backup of Sargon III, you'll sleep better knowing that your original no longer has to face the computer glitch witch. *By Kit LAU Yu-kit & Polly CHAN Yuk-yi.*

27 Softkey for: The Wizardrys

Are you tired of trying to find the correct bit copy parameters for Wizardry? Hardcore COMPUTIST is proud to present a step-by-step method to remove the copy protection from Proving Grounds of the Mad Overlord and Knight of Diamonds. *By Taco van Ieperen.*

DEPARTMENTS

4 INPUT

7 READERS' SOFTKEY & COPY EXCHANGE

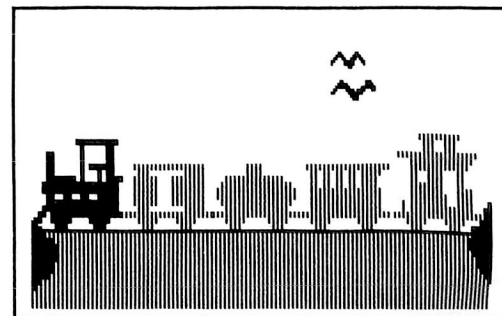
Softkey for Sensible Software's, The Report Card (Ver. 1.1)

By John Howard

Deprotecting Kidwriter

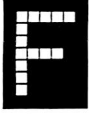
By Mike Stafford

29 ADVENTURE TIPS



Input

RAMcard Secrets

 Finally, those of us without the Integer firmware card and those with Apple //e's or //c's can "continue on" when the instruction tells us to "Reset and enter the monitor". Our salvation is the article in Issue No. 16 by Ken Greenlaw, "RAMCard". Unfortunately, in attempting to use RAMCard, I ran into a problem that I couldn't solve.

First of all, in attempting to use the program XFER.BOOT, I do not always get a BOOT upon typing CALL 768. After some investigative work I noticed that the hexdump and listing did not match. By changing byte 310 to 8D (was 9D) the program seemed to work; at least it booted the protected disk. Once the program disk had been booted, as explained earlier, I then attempted to move or transfer memory as described; however, nothing happened. Can you tell me how to determine what addresses to use with RESTORE when transferring between auxiliary and main memory? My question then: Is there a typo in the programs that are listed? If so, what is the correct listing? I would appreciate your response. As I said, this utility is possibly the greatest breakthrough for owners of //e's and //c's that has come through the media.

I would also like to request that the program "Fun Bunch" be added to your Most Wanted List of programs for backup. Fun Bunch is published by Unicorn and is an extremely interesting and educational program for children. Unfortunately, it has the disadvantage of being protected.

Larry Workman
Anaheim, CA

Mr. Workman: You are absolutely right. There was a typo in the listing of XFER.BOOT. Your fix of the typo corrects the problem. As for your questions about using XFER.BOOT and RESTORE: For many protected disks you will only need to transfer the foreign RWTS at \$B800-\$BFFF back down into main memory. Assuming that the program has been booted into the auxiliary memory and the RESTORE program has been typed in and connected to the Control-Y vector, the following command will perform the transfer

1900<B800.BFFF 

If you are trying to determine what portions of memory a protected program uses, usually the best way to proceed is to fill memory from

\$800 to \$BFFF with a specific value (11 is a good choice) before booting the disk. The protected program is then halted with a RESET to see what portions of memory have been overwritten.

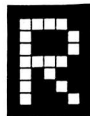
However, filling up the auxiliary memory with a specific value can get a little bit messy because of all the soft switches involved. Probably the easiest thing to do is to modify the RESTORE so that it transfers from main to auxiliary memory, rather than vice-versa. Here is what you can do:

BLOAD RESTORE
300:38 N 3F8:4C 00 03
800:11 N 801<800.63FFM
800<800.63FF 
6400<800.63FF 

This will fill up auxiliary memory from \$800 to \$BFFF with 11's, but will leave the DOS in main memory intact so that you can BLOAD XFER.BOOT.

After the protected program has been booted into auxiliary memory and halted, RESTORE can be used to transfer auxiliary memory to main memory where the areas of memory which were loaded over can be determined.

A Wayout Backup



e. Mr. Hogarth's letter in Issue No. 17. Wayout can be successfully copied using Nibbles Away IIb (I suppose version IIc should also work):

Trks: \$0-\$1C Addr Mark AD DA DD
Trks: \$22 Addr Mark AA D5 D5 FF D6 FF FD
Trk: \$21 Addr Mark AA
Sync size = 0A
Match # = 06
Nibble count ENABLED

Track \$21 is the sensitive track he was referring to. However, it copied fine for me and I have no problems with the backup.

Rocky Giovinazzo
Nashua, NH

Franklin Lives!



I am trying to set up a Franklin User Group. While information about the Apple II is usually of use to those of us who use clones, it is not always. There is also information dealing with

our computers which is seldom found in print.

The group would be connected by a newsletter and disk exchange (an interested person in Connecticut has indicated that we could probably use his bulletin board, as well). I would like to hear from any Franklin, clone or Apple user who would be interested. I can be reached at the address below:

Douglas Trueman
East Wind Rd. Apt. L
Tecumseh, MO 65760

It would be nice if, after telling people that Franklin Corporation was going out of business, you would inform them that they have changed their plans and are come back out of Chapter 11 protection. In fact, they plan to release a computer, the Franklin 2000, an Apple //e-//c compatible by the end of the year.

I am also interested in information on the 65C02 (I own two Franklin 1200 OMS's). I have read that the chip will improve the speed of the computer by 10% with no change in programs. However, the only person I have found who knows anything about this chip had no idea if it would do the same for the Franklin or if it would be possible to put it in the Franklin. I am not interested in programming in assembly and so am only interested in other speed improvements. I would also need to know what other chips might be needed to put the 65C02 in my computer.

Douglas M. Trueman
Tecumseh, MO

Mr Trueman: We haven't had any problems crop-up from installing 65C02's manufactured by Western Design into the Franklin Ace 1000's that we own, so you should not have problems putting one into an Ace 1200. However, don't expect any dramatic speed-up when using commercial software since it would not take advantage of the 65C02's expanded instruction set.

Capturing LS 5.0 Fast Copy



I was very pleased to see the article by C.V. Fields on putting Locksmith 5.0 Fast Copy into a normal binary file in Issue No. 14. This is a very useful utility which I use all the time to backup data diskettes, and I was

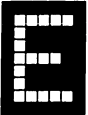
getting tired of changing disks and wading through two menus to get to it. Therefore, although I am relatively inexperienced and do not have a copy card, I decided to attempt it anyway. I thought your readers might be interested in learning how I was able to reset into the monitor with a //e and no copy card.

I selected the 16-sector Fast Disk Backup from the utility menu of Locksmith 5.0 as per the instruction and, as soon as the disk drive light went out, I hit CTRL-Reset. As soon as the disk drive light illuminated on the reboot, I hit CTRL-Reset again and this caused an exit to BASIC. Then it was just a matter of CALL -151 to get into the monitor and continue with the instructions in the article. I now have a normal binary file of this utility which I can FID to any of my other disks.

P.S. I would like to see a softkey for the Hayes Terminal Program and Frogger. Please add these to the Most Wanted List and keep up the good work.

Jerry Battenhouse
Williamson, GA

User's Rights

 nclosed find a check for \$25 for a subscription renewal. I'd much rather pay \$25 for a publication with 30 or so pages than \$15 for a "soft" magazine bloated with self-serving, groveling reviews for software that is featured in large ads. (Conflict of interest? Nahhh.)

Your publication is the only one with enough guts to print what the users really want to know. Until the software publishers realize that the Copyright Act of 1976 gives the buyers of their products a right to make an archival copy, they are going to spend higher and higher percentages of their revenue on increasingly complex and error-prone copy-protection schemes.

The money spent on such programming is borne totally by legitimate buyers and not at all by pirates who have already made the scheme worthless. The companies then escalate the war at an increasing cost in the hope that there is a light at the end of the tunnel, but with no luck.

Finally, formerly honest users see software prices hiked and become more tempted to use pirated works. Sales suffer and companies must charge more to keep profits at a reasonable level. The cycle repeats.

Further, some publishers follow the advice of a high-priced attorney and write Orwellian

shrink-wrap agreements with Doublespeak clauses (no, you didn't buy the program, you only bought the disk it came on). Prosecution can result if you try to list the code. Come on! Shouldn't agreements be enforceable?

If all companies were this bad, one might believe it *had* to end up this way. Yet, look at Beagle Brothers, Penguin, Quinsept, Term-Exec and others. Amid the paranoia and lunacy, they have prospered with unprotected works. Even Apple itself releases the best available integrated package, Appleworks, in an unprotected state.

At first, the cure looks unfair to authors and publishers who, after all, are to thank for the great selection of programs for the][series. But lowered prices and removal of copy-protection would both help to spur sales and decrease per-copy manufacturing costs. Maybe all of the high-priced advertising wouldn't be as essential to a program's success. I know that if a program cost less, I sure wouldn't need as much convincing to purchase it.

If nothing else, I hope Hardcore COMPUTIST will show the protectors that they are fighting an unwinnable war against both friend (buyer) and foe (pirate). Continued expenditures will merely force more to the true enemy camp. Naturally, surrender is out of the question as it would be tantamount to admitting that all of the cash spent on protection was wasted. Perhaps a slow, quiet trend toward forsaking the original intention of locking-up programs would be of interest. It seems such a strategy was tried once before in a slightly different conflict, and look who lost.

I don't mean to trivialize real war in the allusion, liken copy-protection removers to Communists, or to demean those who fought for their country, but to draw a parallel of leadership styles and modes of thinking. Perhaps the master plan in both was noble, but the practical application and effectiveness are certainly a lot different in real life than they were in the board room. The longer companies become entrenched in protection, the harder it will be for them to change. Are there any companies out there willing to remove the copy-protection from already protected works, or is the idea of user-as-potential-pirate already ingrained?

Brad Leonard
St. Louis, MO

(Amen)

Help for Franklin and IBM Users



he main reason that I am writing is to provide some feedback to two recent questions posed by letters in your Input column.

For the Franklin user (Tom Oldenhage), I must also confess to a lack of knowledge of any User Groups. However, right in my own backyard there is a Bulletin Board System devoted to Franklin (and Apple to a certain extent) users. It can be reached at (305) 671-4110, 24 hours on weekends. It is run by a professor at a local prep school and may lead the interested reader to other sources of Franklin information if he is willing to invest in one or more long distance phone calls.

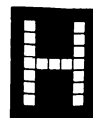
IBM-PC Hardcore-type data can be found on Remote Bulletin Board Systems (RBBB's), HostComm's and other BBS's. What are called 'unprotects' in the IBM-PC world, as well as PEEK/POKE lists, are usually found on these systems. A good starting place would be the Capital (Washington D.C.) PC User's Group (CPCUG). They can be reached at PO Box 3189, Gaithersburg, MD 20878. There are numerous BBS's that have a lot of public domain IBM-PC software as well as the above-mentioned technical info. CPCUG has lists of these on some of their bulletin boards.

As a late vote, PLEASE keep Hardcore devoted to the Apple][line. And finally, I am definitely interested in a group buy of the 65C02. I have an Apple][+ and two Apple //e's to support with this.

Thanks, as always, for keeping at it!

Bob Baker
Orlando, FL

Backup for Dazzle Draw



ere's one of the reasons that the more sophisticated protection methods can be more trouble than they're worth.

My father purchased the Dazzle Draw program for my daughter. We had made our one legal backup copy and everything was going fine; we were really enjoying the program. Then one day I went to boot-up the program and found that it would not work. It only made an odd sound. Strangely enough, I found later that I was able to boot it up on my father's computer.

To make a long story short, I found that this

Input cont...

program uses track arcing for its copy protection and is very speed dependent (the master would only boot-up on 2 out of 7 machines for a 29% bootability, and the copy I had made would boot-up on 3 out of 7 machines for a bootability of 43%).

I subsequently sent my original back to Broderbund explaining my problem with their original and my copy. They sent me a new disk within a week and told me that it had come to their attention that the Apple Duo-Disk may write errors on the softboot (CONTROL, OPEN APPLE, RESET). I was very pleased with Broderbund's quick response until I tried to boot-up the new disk and found that it would not work at all. In fact, it worked worse than the disk I had returned to them.

Although I could get into the Utility Menu (I thought I would make a new copy using the internal copy program that is included with Dazzle Draw to see if it would work after being copied on my own disk drives), to my surprise, Broderbund had already made the one and only copy. I couldn't make another backup! It was then that I declared war on Broderbund and its Dazzle Draw program.

This brings me to the parameters I used to copy the original Dazzle Draw disk while using EDD III. I went back to an older parameters list for EDD on Choplifter and successfully made another copy. Note: I was unable to make a successful copy using the newer parameter list for Choplifter.

Parms 28=2, 00=3

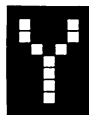
t 0 - t B
t C.25 - t 21.25
t 22

To make a copy from your new copy (not from the original disk), I used EDD III in the normal mode and ONE disk drive and did t 20.25 and t 21.25 over again until they had no read or write errors. I then put in the parms 28=2 and 00=3 for t 22 only.

Since my copies were done at a faster speed, I've been able to boot-up on 5 out of 7 machines or 71%. If I turn on the 80-column card first (PR#3) then use PR#6 to start the boot, I am able to make 100% boots. I also have been able to backup the educational program Spell It, from Davidson & Associates, using EDD III. The parameters are the same as for Dazzle Draw but for tracks 0 through 22 (28=2, 00=3).

Ed Roberts
Ventura, CA

More RAMcard Secrets



our magazine is excellent and I don't know how I ever got along without it. Since receiving my first issue I have learned infinitely many things about the computer. Anyone who isn't subscribing is wasting their Apple.

Regarding the article "Secret Weapon: RAMcard" for deprotection using the Apple //e with 128K (Hardcore COMPUTIST No. 16), I have found a slight problem using that technique with the Applied Engineering (an excellent company, by the way) MemoryMaster //e 128 RAM card.


When RESET is pressed it switches memory banks and, unless you hit RESET again, you will have to manually switch banks. After a few bank switches things can become quite confusing. My solution is this little routine:

```
9500:2C 5E C0 2C 5F C0 4C BF 9D
9500:2C 5E C0 BIT $C05E ;the first of the
                                two toggle switches
9503:2C 5F C0 BIT $C05F ;the second
9506:4C BF 9D JMP $9DBF ;reconnect
                                DOS
```

Then

```
03F2:00 95 30 (changes reset vector to jump
                to $9500 instead of $9DBF)
```

This is a simple routine, but quite effective. It doesn't keep the RESET key from switching banks, but merely switching banks again. I hope this helps some people out.

P.S. To add to your APT for Rescue Raiders:  gives a status display.

Safaa Abdulla
Richmond, VA

In Defense of Quarter Tracks



n Hardcore COMPUTIST No. 17's Whiz Kid column you missed an important point: quarter tracks are EXTREMELY useful against track arcing.

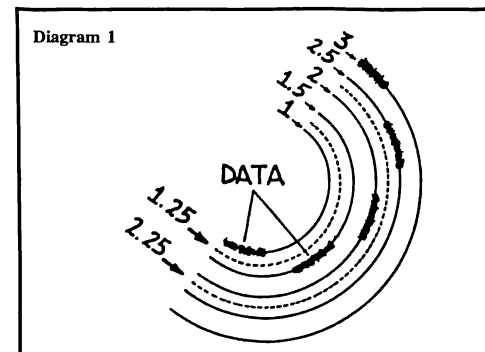
Diagram 1 represents a portion of some tracks formatted using track arcing.

Because adjacent 1/2 tracks overwrite each other, this disk would be almost impossible to copy. But, since 1/4 tracks read 1/2 tracks on BOTH SIDES AT ONCE, copying tracks 1.25 and 2.25 will easily copy the disk. Still in doubt? See pages 92-93 in the Copy][Plus

5.0 manual.

M.M. McFadden
Sunnyvale, CA

Mr. McFadden: Thank you for informing us of this important use of quarter tracks. We suspected that the statement Ray made about quarter tracks "not being worth much" would get some response showing us what they were really good for.



APT Feedback



e. "APT for Advanced Lode Runner" which appeared in Hardcore COMPUTIST No. 17, Input pg. 4.

Many thanks to Warren Power for his Advanced Lode Runner APT. It saved me many hours of trying to copy ALR. Unfortunately, there was a typo in his letter. Tracks 3 to 8 should be copied, not tracks 3 to 6. Otherwise, the number of levels available for play/editing is reduced to 32. Copying through track 8 gets you all fifty ALR levels.

Note that this technique can also be used to copy the regular Lode Runner disk onto a data disk (but copying from tracks 3 through C this time) thus making the games available to the editor and other APT's.

Bring on more Hardcore!

Ken Burnell
Kingdom of Saudi Arabia

Please address letters to: Hardcore COMPUTIST, Editorial Dept., 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. In addition, because of the great number of letters we receive and the small size of our staff, a response to each letter is not guaranteed.

Readers' Softkey & Copy Exchange

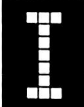
Softkey for Sensible Software's, The Report Card (Ver. 1.1)

By John Howard

The Report Card
Sensible Software Inc.
210 S. Woodward, Suite 229
Birmingham, Michigan 48011
(315) 258-5566
\$59.95

Requirements:

Apple][,][Plus, //e, or //c with 48K and Applesoft
One blank disk

 In Hardcore COMPUTIST No. 18, The Report Card from Sensible Software appeared on the Most Wanted List for softkeys. Since I had deprotected this program some time ago to modify grade calculation algorithms to suit my application, I decided to submit the method of deprotection to share with other users so that they too might modify it for their needs.

I am a subscriber to Hardcore COMPUTIST from almost the first issue and consider it to be the best Apple-specific magazine ever published. I have gleaned information from its pages not available anywhere else. It is one of my most valuable computer tools.

About The Program

The Report Card is a grading system designed specifically for the teacher. It allows easy compilation and reporting of student's marks individually or as a class, and it is completely menu driven which allows the first time user to use the program without reference to the manual. All of the normal bells and whistles of a good grade book program—weighted scores, name transfers, and numerous reports are available.

Nobody Knows The Incompatibilities I've Seen

When I first ran the program I immediately encountered a problem with grade algorithms—I like to compute mine in a certain way, and this program just wouldn't do it. So, rather than having a useless program sitting on the shelf, I thought that maybe I could deprotect it and change the grade algorithms. Following

is the result of my efforts.

About The Protection

The copy protection on the report card disk consists of two different schemes. One is an altered DOS that reads non-standard address prologs. The other is a nibble count check on track 6 while the disk is booting to check for an original disk. No other checks are made for an original disk after the program is loaded into memory. The only disk access after the program loads is to read from and write to the data disk. This makes our job easier.

Advanced Snooping

Using the sector editor and nibble editor of Copy][plus Version 4C, I found that Tracks \$3 - \$22 used an address field prolog of D7 AA 96 instead of the normal D5 AA 96. By patching these values into the Copy][plus sector editor I was able to scan the disk to see what was on it. I found a normal VTOC and CATALOG on track \$11 with two Applesoft files named HELLO and THE REPORT CARD. Hallelujah!!

I then booted a normal DOS disk to see if the deprotection on this disk was going to be that simple. When I dropped into the monitor with a CALL -151, changed \$B955 (in the RDADR routine in DOS) from its normal value of D5 to D7 and typed A56EG (CATALOG routine in DOS), lo and behold, I got a CATALOG listing. This meant that I could manually load and transfer the two report card files by changing the value in location \$B955 to D7 for a LOAD from the protected disk and to D5 for a SAVE to a normal DOS 3.3 Disk.

Step-By-Step Procedure

- 1) Initialize a blank disk with your favorite DOS (preferably a fast one).
- 2) Modify DOS to read the protected disk
POKE 47445,215
- 3) Load a file from the protected disk
LOAD HELLO
- 4) Restore DOS to its original form to allow saving the file to the unprotected disk
Poke 47445,213
- 5) Save the file to the freshly initialized disk
SAVE HELLO
- 6) Repeat steps 2 through 5, substituting "THE REPORT CARD" for "HELLO".

In Closing...

You will now have a normal DOS 3.3 Disk, preferably with a fast DOS on it, and two Applesoft files of 4 and 109 sectors. This will leave plenty of space to use the program disk as a data disk. The HELLO program sets up a few pointers, displays the introductory screen and message, and loads the main program.

One of the first things you may want to modify in the program is line #6 which puts the computer into an infinite loop after you select to quit the program. You must turn off the computer or hit RESET, which kills all your variables, to get your computer back. Just replace the loop code with END to exit cleanly to BASIC. You can then start the program again with all variables intact.

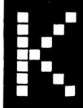
Numerous other possibilities for modification are now at your fingertips. Happy computing!

Deprotecting Kidwriter

By Mike Stafford

Requirements:

Apple][Plus or equivalent
Kidwriter
Super IOB v1.2

 idwriter is a "Kiddie Word-Processor" that allows children to create their own stories and illustrate them with colorful pictures. Unfortunately this program does not come with a backup disk and is not copyable with COPYA. Luckily, the copy protection on this disk is not too extensive and the disk can be easily de-protected.

The copy protection scheme consists of changing the address field prologue from the normal D5 AA 96 to D5 AA FE, and the data field prologue from D5 AA AD to D5 AA FF. This disk can be bit copied, but I prefer to have my disks unprotected (especially this program since I had some modifications in mind for it.)

To de-protect this disk, all that is necessary is to use a Super IOB controller which will read the altered address and data headers and then write the tracks to another disk with their normal values. After you're done making the copy you'll need to add a DOS to the disk. I prefer a fast DOS such as Pronto DOS, as

Readers' Softkey & Copy Exchange cont...

this program will load very slowly with the standard DOS 3.3. Finally, you will have to change the boot program because Kidwriter does not boot up with the normal HELLO program.

Here is the step-by-step procedure to softkey Kidwriter:

1) Initialize a blank disk with a fast DOS (preferably) and "KIDWRITER" as the boot filename

INIT KIDWRITER

2) Type and save the controller at the end of this article.

3) Install the controller into your Super IOB program and execute it:

RUN

3) Type in the hexdump at the end of this article and save it to your freshly created disk

BSAVE KIDWRITER,A\$9000,L\$33

You should now have a COPYAable version of Kidwriter.

Print-Out: The Modification

You can add a hi-res printout option for every page in a story if you have a sector editor and know a little bit of assembly language. To do this you will first have to modify your copied disk. Get out your sector editor and make the following alterations:

Track	Sector	Bytes	To
\$1A	\$06	\$0D-\$0F	\$20 \$1C \$90
\$1A	\$06	\$24-\$26	\$20 \$1C \$90
\$1A	\$07	\$F7-\$F9	\$20 \$1C \$90

This alters the program so that it calls \$901C when it needs to wait for a keypress between pages of a story. Once here, our KIDWRITER file takes over and tests for a **[P]**. If this key is pressed, program execution falls through to \$902F. Right now, \$902F just returns to the main program and no hi-res printing occurs.

As a result, the second thing you must do is add a machine language hi-res printout routine at \$902F and re-BSAVE the KIDWRITER file. This routine will differ depending on your printer-printer card combination. In any case, the routine should save the COUT vector (\$36 and \$37 (I saved them on the stack)) and restore it when finished.

KidWriter Controller

```

1000 REM KIDWRITER CONTROLLER
1010 TK = 0 : ST = 0 : LT = 35 : CD = WR
1020 T1 = TK : GOSUB 490 : RESTORE : GOSUB 190
      : GOSUB 210
1030 GOSUB 430 : GOSUB 100 : ST = ST + 1 : IF ST
      < DOS THEN 1030
1040 IF BF THEN 1060
1050 ST = 0 : TK = TK + 1 : IF TK < LT THEN 1030
1060 GOSUB 230 : 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 : IF BF = 0 AND TK < LT THEN
      1070
1090 IF TK < LT THEN 1020
1100 HOME : PRINT "DONE^WITH^COPY" : END
5000 DATA 213 ,170 ,254
5010 DATA 213 ,170 ,255
    
```

Controller Checksums & Hexdump

```

1000 - $356B    1070 - $B1B6
1010 - $3266    1080 - $1FDD
1020 - $8CB9    1090 - $F88D
1030 - $AAB8    1100 - $7AA8
1040 - $A841    5000 - $196A
1050 - $9AA6    5010 - $F27C
1060 - $B9B3

9000: A0 0F B9 0C 90 20 ED FD    $CAD5
9008: 88 10 F7 60 8D 85 88 88    $D88F
9010: 88 88 88 88 88 C7 CE D5    $F7CE
9018: D2 C2 84 8D 2C 00 C0 30    $3B05
9020: 01 60 48 AD 00 C0 C9 90    $303B
9028: F0 05 68 2C 00 C0 60 68    $19B9
9030: 2C 00 C0 60
    
```

KidWriter Source Code

```

*-----*
*                               KIDWRITER BOOT PROGRAM                               *
*-----*

FDDED- COUT      .EQ $FDED

*-----*

      .OR $9000
      .TF KIDWRITER

9000: A0 0F          LDY #ENDMSG-MSG
9002: B9 0C 90      PRNTMSG LDA MSG,Y
9005: 20 ED FD      JSR COUT
9008: 88           DEY
9009: 10 F7        BPL PRNTMSG
900B: 60          RTS

900C: 8D 85 88
900F: 88 88 88
9012: 88 88 88    MSG      .HS 8D8588888888888888
9015: C7 CE D5
9018: D2 C2          .AS -/GNURB/
901A: 84            .HS 84
901B: 8D          ENDMSG  .HS 8D

901C: 2C 00 C0      BIT $C000
901F: 30 01        BMI CTRL.P
9021: 60          RTS
9022: 48          CTRL.P PHA
9023: AD 00 C0      LDA $C000
9026: C9 90        CMP #$90
9028: F0 05        BEQ DUMP.IT
902A: 68          PLA
902B: 2C 00 C0    BIT $C000
902E: 60          RTS
902F: 68          DUMP.IT PLA      START PRINTOUT ROUTINE HERE
9030: 2C 00 C0    BIT $C000      REMEMBER TO SAVE COUT
9033: 60          RTS      POINTERS AND RESTORE THEM WHEN FINISHED
    
```

Apple][Boot ROM

Disassembly

By Ray Darrah and Gary Peterson

Carefully read this article and examine the accompanying disassembly and you can learn how the disk controller hardware addresses (\$C0n0-\$C0nF) are used for direct access of the disk drive, how 6 & 2 encoding works, how the address and data fields are read, and generally increase your overall knowledge of the Apple][disk drive.



The 256 byte PROM at position P5 on the Apple's disk controller card contains machine language code that all bootable disks, protected or not, must execute as the first stage of their boot. Many people take advantage of this fact to perform what is known as "boot-code tracing" or the "front door method" in order to track down the location of a disk's protection scheme. Several articles previously published in the pages of this magazine have elaborated upon the technique of boot code tracing.

Because the boot PROM is so widely utilized, we would like to present a disassembly of the code in order to clear up some of the mystery of how it works. By carefully reading this article and examining the accompanying disassembly you can learn how the disk controller hardware addresses (\$C0n0-\$C0nF) are used for direct access of the disk drive, how 6 & 2 encoding works, how the address and data fields are read, and generally increase your overall knowledge of the Apple][disk drive.

This article and the commented disassembly assume a disk drive controller in slot 6.

The Nybble Conversion Table

The code at \$C600-\$C620 is responsible for generating a nybble conversion table at \$36C-\$3D5. This table is used to convert the encoded (or nybbled) bytes read from the disk into a six-bit hexadecimal value between \$0 and \$3F. To say the least, this is a very obscure piece of code that is extremely difficult to follow. If you know of someone who frequently boasts of his or her knowledge of 6502 programming, give them this little chunk of code and ask them to tell you what it does. More likely than not, you will soon notice a marked increase in this person's overall humility quotient.

This code tests for values which meet the requirements imposed by the 6&2 encoding method (high bit set, no more than one set of consecutive zeroes and at least one set of consecutive ones.) When a value corresponding to a valid disk byte is found, the translated equivalent of that value is stored at \$2D6 plus the value of that valid disk byte. Since this is bound to sound pretty confusing, lets look at a couple of examples:

The first hex value that meets the requirements of 6&2 encoding is a \$96. In the read translate table, 96=00; therefore, a 00 will be stored at \$2D6+\$96 or \$36C. In the same

The code starting at \$C65C is really the meat of the disk controller's code. The code here is used by the DOS 3.3 BOOT1 code at \$84A (JMP (\$003E)) to read the RWTS into memory. Some protected disks, like Rocky's Boots, rely entirely on this piece of the disk controller code when reading from a disk.

manner, since \$ED is a valid disk byte and from the read translate table, ED=30, a 30 will be stored at \$2D6+\$ED or \$3C3 (later we will get back to how this table is used to convert the encoded values read from the disk back into their unencoded form). If you want more details about the nybble conversion table, examine the commented source code accompanying this article.

Stack Extraction and Recal

The next part of the ROM at \$C621 to \$C63A is responsible for determining which slot the disk controller is in and for recalibrating the head of the disk drive to track \$0. The slot number is determined by doing a JSR to an RTS instruction in the Apple's monitor. After the RTS has been executed, the code retrieves the return address from the stack and uses it to calculate which slot the controller is in. The slot number is then multiplied by 16 and stored in zero page at \$2B. Once the slot number has been calculated, it is transferred to the X register where it can be used to access the correct controller hardware locations. Drive one is enabled and the motor is turned on before stepping over 80 phases by accessing the various phase on/phase off locations in descending order. This has the effect of moving the head outward to track zero (track \$0 is the outermost track), no matter where it happened to start from.

Boot ROM Source Code

```

*-----*
*                                     *
*                                     *
*                                     *
*-----*
                                .OR $C600   FOR SLOT 6
                                .TA $800    STORE IT HERE FOR NOW

*-----*
*                                     *
*                                     *
*-----*
                                ZERO PAGE USAGE

*-----*
*                                     *
*-----*
0026- SECTOR.PTR .EQ $26   POINTS TO THE LOCATION IN MEMORY WHERE THE CURRENT SECTOR IS
                                TO BE STORED
002B- SLOT      .EQ $2B   HOLDS BOOTING SLOT NUMBER TIME 16
003C- TEMP0     .EQ $3C   USED BY DIFFERENT ROUTINES FOR DIFFERENT THINGS
003D- SECTOR    .EQ $3D   HOLDS THE SECTOR NUMBER WE ARE LOOKING FOR
0040- TRK.READ  .EQ $40   HOLDS THE TRACK NUMBER FOUND FROM DISKETTE
0041- TRACK     .EQ $41   HOLDS THE TRACK NUMBER WE ARE LOOKING FOR

*-----*
*                                     *
*-----*
                                DISK CONTROLLER BASE ADDRESSES

*-----*
*                                     *
*-----*
C080- MAG0.OFF .EQ $C080  STEPPER MAGNET 0 OFF
C081- MAG0.ON  .EQ $C081  STEPPER MAGNET 0 ON
C082- MAG1.OFF .EQ $C082  STEPPER MAGNET 1 OFF
C083- MAG1.ON  .EQ $C083  STEPPER MAGNET 1 ON
C084- MAG2.OFF .EQ $C084  STEPPER MAGNET 2 OFF
C085- MAG2.ON  .EQ $C085  STEPPER MAGNET 2 ON
C086- MAG3.OFF .EQ $C086  STEPPER MAGNET 3 OFF
C087- MAG3.ON  .EQ $C087  STEPPER MAGNET 3 ON
C088- MOTOR.OFF .EQ $C088 DRIVE AND MOTOR OFF
C089- MOTOR.ON  .EQ $C089 DRIVE AND MOTOR ON
C08A- SEL.DRV1 .EQ $C08A  ROUTE POWER TO DRIVE 1
C08B- SEL.DRV2 .EQ $C08B  ROUTE POWER TO DRIVE 2
C08C- I01      .EQ $C08C  STROBE LATCH FOR I/O
C08D- I02      .EQ $C08D  LOAD DATA LATCH
C08E- I03      .EQ $C08E  PREPARE LATCH FOR INPUT
C08F- I04      .EQ $C08F  PREPARE LATCH FOR OUTPUT

*-----*
*                                     *
*-----*
                                MONITOR LOCATIONS USED

*-----*
*                                     *
*-----*
FF58- IORTS    .EQ $FF58  A KNOWN RTS INSTRUCTION
FCA8- WAIT     .EQ $FCA8  WAIT FOR TIME SPECIFIED BY A-REG, X & Y UNALTERED

*-----*
*                                     *
*-----*
                                MASS DATA STORAGE AREAS

*-----*
*                                     *
*-----*
0100- STACK    .EQ $0100  6502 STACK AREA STARTS HERE
02D6- NIBBLE.TBL .EQ $02D6 6 & 2 DECODE TABLE FROM HERE TO $3D5 (FIRST NIBBLE AT $
36C)
0300- BUFFER1  .EQ $0300  EXTRA 86 BYTES (342-256=86) OF EACH SECTOR STORED FROM
HERE TO $355
0800- BUFFER2  .EQ $0800  FIRST SECTOR OF DATA GOES TO $0800

*-----*
*                                     *
*-----*
                                PR# OR IN# GOES HERE

```

C600: A2 20 LDX #520 DOES NOTHING EXCEPT IDENTIFY DISK CONTROLLER

```

*-----*
*           SET UP A READ TRANSLATE TABLE           *
*           A = GENERAL USAGE                         *
*           X = NIBBLE TO TEST FOR VALIDITY (-$80)    *
*           Y = WHAT NEXT VALID NIBBLE TRANSLATES INTO *
*-----*

```

```

C602: A0 00 LDY #500 FIRST BYTE TRANSLATES TO A 500
C604: A2 03 LDX #503 COULD HAVE USED ANYTHING FROM 500 - 516
C606: 86 3C TST.NIBB STX TEMP0 PUT NIBBLE HERE SO IT CAN BE MANIPULATED
C608: 8A TXA PUT NIBBLE IN A SO WE CAN SHIFT IT
C609: 0A ASL TEST FOR ADJACENT ONES
C60A: 24 3C BIT TEMP0 IF A ^ M = 0 THEN Z=1
C60C: F0 10 BEQ NXT.NIBB NO ADJACENT ONES, GET NEXT NIBBLE
C60E: 05 3C ORA TEMP0 THE ASL, ORA, EOR SEQUENCE WILL
C610: 49 FF EOR #5FF CHANGE ALL ADJACENT 0-BITS TO ONE 1-BIT
C612: 29 7E AND #57E IGNORE BIT 7 AND 0
C614: B0 08 ADJ0.TST BCS NXT.NIBB NEVER TAKEN THE FIRST TIME THROUGH, ON SUBS
EQUENT TIMES, THIS BRANCH IS TAKEN IF THERE STILL ARE BITS SET IN THE NIBBLE
AND A SET BIT HAS BEEN PUT INTO THE CARRY: THEREFORE MORE THAN ONE PAIR OF ADJ
ACENT 0-BITS
C616: 4A LSR SHIFT A BIT INTO THE CARRY, COULD HAVE USED A
SL
C617: D0 FB BNE ADJ0.TST TAKEN IF THERE ARE STILL SOME BITS SET (THERE
FORE SOME ADJACENT 0-BITS) LEFT IN NIBBLE
C619: 98 TYA NIBBLE PASSED TEST, A=WHAT NIBBLE TRANSLATES
INTO
C61A: 9D 56 03 STA NIBBLE.TBL+$80,X STORE IT IN TRANSLATE TABLE + $80
C61D: C8 INY NEXT NIBBLE TRANSLATES INTO Y + 1
C61E: E8 NXT.NIBB INX GET NEXT NIBBLE TO TRY (-$80)
C61F: 10 E5 BPL TST.NIBB IF NOT LAST NIBBLE $7F=$FF THEN KEEP GOING

```

```

*-----*
*           FIND WHAT SLOT WE ARE IN                 *
*-----*

```

```

C621: 20 58 FF JSR IORTS JSR TO A KNOWN RTS INSTRUCTION
C624: BA TSX PUT STACK POINTER IN X
C625: BD 00 01 LDA STACK,X GET MSB OF RETURN ADDRESS FROM STACK
C628: 0A ASL MOVE LOWER FOUR BITS INTO UPPER FOUR BITS
C629: 0A ASL IN OTHER WORDS, MULTIPLY BY 16 MOD 256
C62A: 0A ASL
C62B: 0A ASL
C62C: 85 2B STA SLOT SAVE SLOT NUMBER TIME 16 IN ZERO PAGE
C62E: AA TAX AND PUT SLOT NUMBER * 16 IN X

```

```

*-----*
*           TURN ON DRIVE                            *
*-----*

```

```

C62F: BD 8E C0 LDA I03,X SET DRIVE IN READ MODE
C632: BD 8C C0 LDA I01,X
C635: BD 8A C0 LDA SEL.DRV1,X MAKE SURE DRIVE ONE!
C638: BD 89 C0 LDA MOTOR.ON,X TURN ON MOTOR

```

```

*-----*
*           RECALIBRATE DRIVE HEAD                   *
*-----*

```

```

C63B: A0 50 LDY #80 MOVE BACKWARDS OVER 80 PHASES (40 TRACKS)
C63D: BD 80 C0 NXT.PHASE LDA MAG0.OFF,X TURN CURRENT PHASE MAGNET OFF
C640: 98 TYA CALCULATE NEXT PHASE

```

Continued on next page

Initialize Zero Page

The next section of the disk controller ROM code at \$C652-\$C65B is very short. All it does is set up the necessary zero page locations so that track \$0, sector \$0 is read into page \$8. Zero page locations \$26-\$27 point to the data buffer, \$3D holds the sector number and \$41 holds the track number.

Reading the Disk

The code starting at \$C65C is really the meat of the disk controller's code. The code here is used by the DOS 3.3 BOOT1 code at \$84A (JMP (\$003E)) to read the RWTS into memory. Some protected disks, like Rocky's Boots, rely entirely on this piece of the disk controller code when reading from a disk.

When entered, this routine first clears the Carry flag to signal that an address header is to be read. Once the address header pattern of D5 AA 96 is found, a branch to \$C683 is taken

If you take a look at the listing from \$C65C-\$C6A5, you will notice that the address epilog or checksum is never verified. Some copy-protected disks take advantage of this fact and alter the address epilogs or the checksum to prevent people from reading track \$0, sector \$0 with a normal sector editor.

where the 4&4-encoded volume, track and sector data is read. The track and sector numbers are compared to the values at \$3D (sector) and \$41 (track). If they are correct, the carry flag is set and the code branches back to \$C65D, this time to read a data header. If the track and sector are not correct, then the carry flag is cleared and the code returns to \$C65C to again try to find the desired track and sector.

If you take a look at the listing from \$C65C-\$C6A5, you will notice that the address epilog or checksum is never verified. Some copy-protected disks take advantage of this fact and alter the address epilogs or the checksum to prevent people from reading track \$0, sector \$0 with a normal sector editor. However, the address or data prologues on track \$0, sector \$0 cannot be altered from D5 AA 96 and D5 AA AD.

Decoding the Data

The code at \$C6A6-\$C6EA is responsible for reading encoded data from the desired sector into the proper buffer. Because it takes \$156 (342 in decimal) 6&2 encoded bytes to store

one sector of data, the reading occurs in two stages.

First, \$56 bytes are read into \$300-\$356. The remaining \$100 bytes are read into the buffer pointed at by \$26-\$27 (\$800 upon booting). Notice at \$C6AA and \$C6BC that when a byte of data is read from the disk, it is loaded into the Y register. This is so that the value read from the disk can be used as an index into the nybble conversion table that was generated at \$36C-\$3D5. This is how the valid byte read from the disk is converted into a six-bit hex value.

The Data Checksum

Just before each of the \$156 bytes is stored into the buffers, an Exclusive OR operation is performed. This is necessary because the data was Exclusive ORed with itself (offset by one byte) before it was written to disk. All of this EORing produced a cumulative checksum byte which was stored as the final byte of the encoded sector. When the sector is read, the EORing again occurs and, if the accumulator does not hold a value of zero after the final EOR with the checksum byte, an error has occurred and the code at \$C683 branches back to try again.

If no checksum error occurred, then the code falls through to \$C6D5 where the 6&2 encoded data in the two buffers is restored to its original, unencoded form. This involves a lot of bit-shifting to restore each byte with the two bits that were stripped off of it before it was written. These two bits come from that data which is stored at \$300-\$355.

Once the data in the primary buffer (\$800-\$8FF) has been restored to its original form, the code falls through to \$C6EB where a check is made to see if any more sectors have to be read. The number of sectors to be read is contained in \$800. Normally this byte will contain a 1 but it could be any value up to an \$F. I have seen very few disks, aside from the BASIC's disk, that will load in more than one sector during their stage 0 boot. Once the desired number of sectors have been read, the JMP \$801 is executed to start the next stage of the boot.

After the jump to \$801 has been taken, a disk is pretty much free to proceed with its boot as the programmer has seen fit, although most disks will use the routine starting at \$C65C to read in several more sectors from track \$0. Normal DOS 3.3 disks will read the RWTS into memory from track \$0, sectors \$0-\$9. Sector \$0 is reread so that its image is available when a new disk needs to be formatted.

References:

- Worth, Don & Peter Lechner, Beneath Apple DOS. Quality Software. 1981.
- Worth, Don & Peter Lechner, Beneath Apple ProDOS. Quality Software. 1984.
- Apple Assembly Lines, S-C Software. 09/81, pages 17-20.

Continued from previous page

```

C641: 29 03      AND #$03      MOD 4
C643: 0A         ASL           *2
C644: 05 2B      ORA SLOT      MERGE IN TO ZERO PAGE SLOT HOLDER
C646: AA         TAX           PUT NEXT PHASE INTO X
C647: BD 81 C0    LDA MAG0.ON,X TURN ON NEXT MAGNET
C64A: A9 56      LDA #$56      WAIT DELAY OF 19.2 MILLISECONDS
C64C: 20 A8 FC    JSR WAIT      WAIT A WHILE
C64F: 88         DEY           NEXT PHASE OR HALF TRACK
C650: 10 EB      BPL NXT.PHASE KEEP GOING IF NOT DONE

```

```

* ----- *
*                               *
* SET UP ZERO PAGE TO READ TRACK 0, SECTOR 0 *
* ----- *

```

```

C652: 85 26      STA SECTOR.PTR LSB OF POINTER = 0
C654: 85 3D      STA SECTOR SECTOR 0
C656: 85 41      STA TRACK TRACK 0
C658: A9 08      LDA /BUFFER2 MSB OF POINTER = $08
C65A: 85 27      STA SECTOR.PTR+1 PUT SECTOR AT $0800

```

```

* ----- *
*                               *
* FIND EITHER THE ADDRESS HEADER *
* OR DATA HEADER OF THE CURRENT SECTOR *
* ----- *

```

```

C65C: 18         GET.ADDR.HDR CLC      C=0 MEANS LOOK FOR ADDRESS HEADER
C65D: 08         GET.DATA.HDR PHP    SAVE CARRY
C65E: BD 8C C0   P1      LDA I01.X      READ DISK
C661: 10 FB      BPL P1        WAIT FOR FIRST BYTE
C663: 49 D5     TRY.D5   EOR #$D5      IS IT A $D5?
C665: D0 F7      BNE P1        NOPE. GET NEXT BYTE
C667: BD 8C C0   .1     LDA I01.X      READ DISK
C66A: 10 FB      BPL .1        WAIT FOR SECOND BYTE
C66C: C9 AA      CMP #$AA      IS IT A SAA?
C66E: D0 F3      BNE TRY.D5   NOPE. MAYBE IT WAS A $D5
C670: EA         NOP           WAIT 2 CYCLES
C671: BD 8C C0   .2     LDA I01.X      READ DISK
C674: 10 FB      BPL .2        WAIT FOR LAST BYTE IN EITHER ADDRESS OR DATA
          HEADER
C676: C9 96      CMP #$96      IS IT THE ADDRESS HEADER?
C678: F0 09      BEQ GET.SCT.NUM YES. SEE IF RIGHT ONE
C67A: 28         PLP           WERE WE LOOKING FOR THE DATA MARK?
C67B: 90 DF      BCC GET.ADDR.HDR NOPE. TRY FOR ADDRESS HEADER AGAIN
C67D: 49 AD      EOR #$AD      LOOKING FOR DATA HEADER. IS THIRD BYTE AND SA
          D?
C67F: F0 25      BEQ READ.SCT YES. GET THE DATA FROM THE SECTOR
C681: D0 D9      BNE GET.ADDR.HDR NOPE. FIND ANOTHER ADDRESS HEADER

```

```

* ----- *
*                               *
* READ VOLUME, TRACK AND SECTOR *
* ----- *

```

```

C683: A0 03      GET.SCT.NUM LDY #$03 THREE 4+4 ENCODED BYTES
C685: 85 40      STORE.44 STA TRK.READ STORE CURRENT DECODED BYTE HERE, FIRST VOLUME
          NUMBER WHICH WILL BE OVERWRITTEN BY THE TRACK NUMBER
C687: BD 8C C0   .1     LDA I01,X      READ DISK
C68A: 10 FB      BPL .1        WAIT FOR FIRST HALF OF BYTE
C68C: 2A         ROL           RE-POSITION EVERY OTHER BIT
C68D: 85 3C      STA TEMP0     SAVE IT HERE
C68F: BD 8C C0   .2     LDA I01,X      READ DISK
C692: 10 FB      BPL .2        WAIT FOR SECOND HALF OF BYTE
C694: 25 3C      AND TEMP0     MERGE TWO HALVES TOGETHER TO DECODE BYTE
C696: 88         DEY           DONE WITH ADDRESS FIELD (EXCEPT CHECKSUM)?

```

C697: D0 EC

BNE STORE.44 NOPE, GET NEXT BYTE

* SEE IF THIS IS THE SECTOR WE WANT *

C699: 28 PLP FIX STACK BECAUSE OF THE PHP AT \$C65D
C69A: C5 3D CMP SECTOR A-REG HAS SECTOR READ, IS IT THE RIGHT ONE?
C69C: D0 BE BNE GET.ADDR.HDR NOPE, LOOK FOR NEXT ADDRESS FIELD
C69E: A5 40 LDA TRK.READ GET TRACK NUMBER READ
C6A0: C5 41 CMP TRACK IS IT THE ONE WE WANT?
C6A2: D0 B8 BNE GET.ADDR.HDR NOPE, LOOK FOR NEXT ADDRESS FIELD
C6A4: B0 B7 BCS GET.DATA.HDR . ALWAYS TAKEN, LOOK FOR DATA FIELD HEAD
R

* READ SECTOR AND STORE IT IN MEMORY *

C6A6: A0 56 READ.SCT LDY #86 READ FIRST EIGHTY SIX BYTES
C6A8: 84 3C NXT.BUF1 STY TEMP0 SAVE NUMBER OF BYTES OF BUFFER1 LEFT TO READ
C6AA: BC 8C C0 .1 LDY I01,X READ DISK
C6AD: 10 FB BPL .1 WAIT FOR 6 & 2 ENCODED NIBBLE OF DATA
C6AF: 59 D6 02 EOR NIBBLE.TBL,Y DECODE THE BYTE (A=0 FIRST TIME THROUGH)
C6B2: A4 3C LDY TEMP0 GET NUMBER OF BYTES LEFT
C6B4: 88 DEY DONE WITH BUFFER1?
C6B5: 99 00 03 STA BUFFER1,Y SAVE 6 BIT NIBBLE
C6B8: D0 EE BNE NXT.BUF1 NOT DONE WITH FIRST 86 BYTES, CONTINUE

* READ REMAINING DATA FIELD BYTES *

C6BA: 84 3C NXT.BUF2 STY TEMP0 SAVE THE CURRENT BYTE NUMBER (FIRST=0)
C6BC: BC 8C C0 .1 LDY I01,X READ DISK
C6BF: 10 FB BPL .1 WAIT FOR BYTE
C6C1: 59 D6 02 EOR NIBBLE.TBL,Y TRANSLATE NIBBLE
C6C4: A4 3C LDY TEMP0 GET CURRENT BYTE INDEX (NUMBER)
C6C6: 91 26 STA (SECTOR.PTR),Y SAVE BYTE IN BIG BUFFER
C6C8: C8 INY NEXT BYTE INDEX
C6C9: D0 EF BNE NXT.BUF2 KEEP READING BYTES UNTIL ALL 256 HAVE BEEN RE
AD
C6CB: BC 8C C0 .2 LDY I01,X READ DISK
C6CE: 10 FB BPL .2 WAIT FOR DATA FIELD CHECKSUM
C6D0: 59 D6 02 EOR NIBBLE.TBL,Y TRANSLATE THE BYTE
C6D3: D0 87 GOTO.NXT.SECTOR BNE GET.ADDR.HDR IF CHECKSUM NOT RIGHT THEN START A
LL OVER

* DECODE THE 6 & 2 ENCODED SECTOR *

C6D5: A0 00 LDY #00 START WITH BYTE 0 OF THE BUFFER
C6D7: A2 56 STRT.BUF1 LDX #86 86 BYTES IN BUFFER1
C6D9: CA NXT.62.DEC DEX GET NEXT BUFFER1 INDEX
C6DA: 30 FB BMI STRT.BUF1 IF = \$FF THEN START BACK AT END OF BUFFER1
C6DC: B1 26 LDA (SECTOR.PTR),Y GET BYTE MISSING LOWER TWO BITS
C6DE: 5E 00 03 LSR BUFFER1,X GET BIT FROM BUFFER1
C6E1: 2A ROL PUT INTO BUFFER2 BYTE
C6E2: 5E 00 03 LSR BUFFER1,X GET THE OTHER BIT FROM BUFFER1
C6E5: 2A ROL PUT INTO BUFFER2 BYTE
C6E6: 91 26 STA (SECTOR.PTR),Y BYTE IS DECODED, STORE IT
C6E8: C8 INY GET NEXT BUFFER2 INDEX
C6E9: D0 EE BNE NXT.62.DEC NOT DONE WITH WHOLE PAGE, DO NEXT BYTE

Continued on next page

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.

- | | |
|---|--|
| 1. Apple Business Graphics
Apple Computer | 13. Robot Odyssey
The Learning Company |
| 2. Flight Simulator II
Sub Logic | 14. Zardax
Computer Solutions |
| 3. Apple LOGO II
Apple Computer | 15. The Handlers
Silicon Valley Systems |
| 4. Jane
Arkrtronics | 16. Milliken Math Series (NEW)
Milliken Publishing |
| 5. Bookends
Sensible Software | 17. College Entrance Exam Prep
Borg Warner |
| 6. Visiblend
Microlab | 18. Bank Street Speller
Broderbund |
| 7. Sundog
FTL Games | 19. Karateka
Broderbund |
| 8. Lifesaver
Microlab | 20. King's Quest
Sierra On-line |
| 9. Catalyst
Quark, Inc. | 21. Hayes Terminal Program
Hayes |
| 10. Gutenberg Jr. & Sr.
Micromation LTD | 22. Fun Bunch
Unicorn |
| 11. Prime Plotter
Primesoft Corp. | |
| 12. The Newsroom
Springboard Software | |

The Graphic Grabber v3.0

By Ray Darrah III

Requirements:

Apple II Plus or equivalent
One disk drive and DOS 3.3
The Print Shop (By Broderbund)



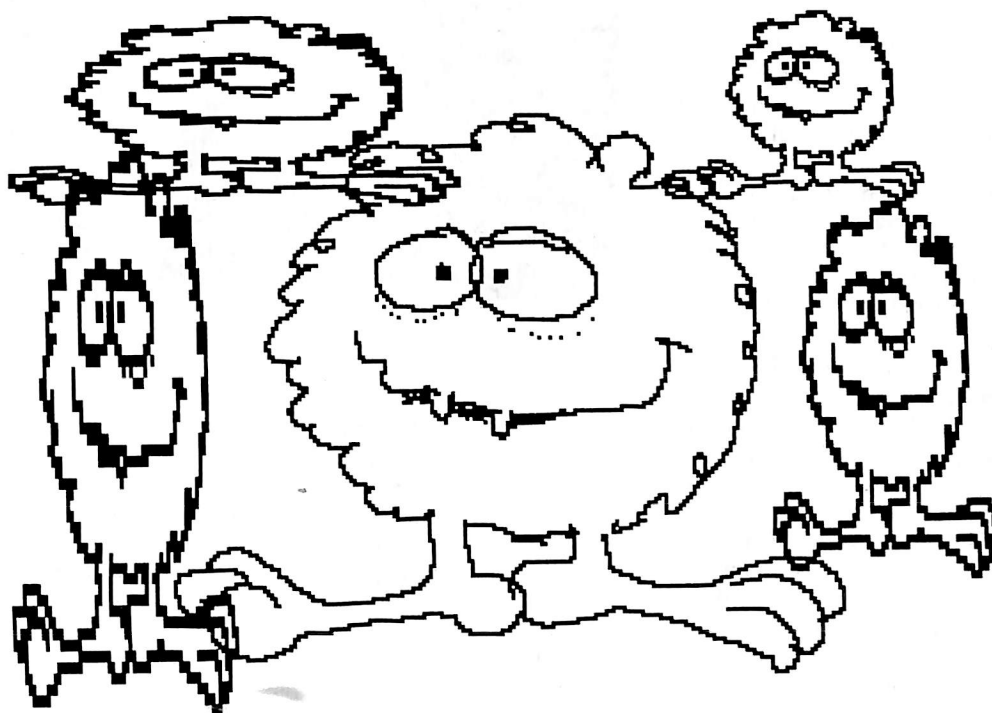
ew printer-graphics programs have the versatility and ease of use which are incorporated in Broderbund's recent release, The Print Shop.

Often I find myself behind the controls of this program sculpting yet ANOTHER creative work. Judging from the great number of letters that Hardcore COMPUTIST receives on stationary created by The Print Shop, our readers also put this program to good use.

One thing I particularly appreciate about this program is the fact that the user is allowed to create his or her own graphics and store them on an unprotected disk. My few criticisms of the package are that user-definable fonts are not available and that its graphic editor is not up to the standards of some other programs currently on the market.

My initial disappointment in The Print Shop's graphic editor led me to create a program called The Graphic Grabber (featured in Hardcore COMPUTIST No. 17). This program allows the user to capture a small part of any hi-res screen (88 x 52 pixels) and save it for use by The Print Shop.

After using the program to "grab" many graphics, I decided that some additional features would make the Graphic Grabber even more useful. For example, wouldn't it be nice to be able to take a graphic and place it on a hi-res screen along with several other graphics to create sheets of available graphics similar to those that come with The Print Shop? Or, how about shrinking a large picture to the size that The Print Shop requires? With



these ideas I began to revise the program, and the new version (which accompanies this article) enables you to "grab graphics" even more efficiently than before. I call this The Graphic Grabber 3.0.

Typing It In

Unfortunately, since so many revisions and overhauls were made to the original program, you cannot simply change some lines in the Graphic Grabber program you typed in from Hardcore COMPUTIST No. 17. Instead, you must type in the two files (one Applesoft and one Binary) which comprise The Graphic Grabber 3.0 (see page 2 of this issue if you are not familiar with the process of entering and saving programs).

Save the Applesoft BASIC portion with:

SAVE GRAPHIC GRABBER 3.0

Save the Binary hexdump with:

**BSAVE OBJ.GRAPHIC GRABBER,
A\$4000,L\$1AB**

Using The Graphic Grabber 3.0

When The Graphic Grabber 3.0 is RUN, the hi-res screen page 1 is left intact. This is so that you may use what is currently on the screen as a graphic rather than going through the hassle of saving the screen, running The Graphic Grabber, and then re-loading the screen.

If all goes well, The Graphic Grabber will come up with a help screen of all the

commands available to you. At the bottom of the screen there should be a display of some of the important parameters.

Moving Right Along...



When you are viewing the help screen, notice that the command list indicates to type “?” to toggle between the help screen and the hi-res screen. Actually, hitting any key that does not invoke one of the other functions will toggle between the text and hi-res screens.

When viewing the hi-res screen, you will notice a flashing square that defines the “window” upon which The Graphic Grabber 3.0 operates. When considering the size of the window, be sure to include the boundaries of the window itself as these are included in any action of the window.

The Window

Motion is accomplished by pressing the standard ESCape keys. Pressing “I,J,K,M” will move the window up, left, right and down, respectively.

Pressing “P” will invoke a routine which will ask you for a new window position. You may type in a couple of hi-res coordinates and the window will immediately jump there. The upper left-hand corner of the window will be placed at the coordinates you specify.

A third way to move the cursor is to type “I” or the “TAB” key if you have a //e. This will move the window to the next graphic position. This feature was designed primarily to aid in the creation of sheets of various graphics. If you first use the “P” command to place the window at 4,9, every time you press the I key the window will jump to the next available position on the screen that does not overlap the space occupied by an adjacent graphic. This makes it easy to evenly space the various graphic images you wish to place on the screen, and is also useful for quickly moving the window to a general area on the screen.

Toggle Parameters

Invoke this option by pressing the **spacebar**. and the parameters at the bottom of the screen will become either visible or invisible. When the parameters are invisible, the bottom 32 lines of the hi-res screen are visible.

Clear Screen

If you type “C”, the hi-res screen will be cleared to black. This is most used to simply clear the screen before creating a sheet of available graphics. As a safety precaution

against inadvertently clearing the screen by mistake, “OK to clear hi-res screen?” will appear and you will have to verify that you *really do* want to perform this function.

Reverse Buffer

A graphic saved with The Graphic Grabber 3.0 will always be printed by The Print Shop exactly as it appears on the hi-res screen; unlit pixels are printed as black and lit pixels are printed as white (actually, not printed at all). If you wish a graphic to be printed in inverse, you may reverse the current graphic buffer (lit pixels become unlit and vice versa) by pressing “R”.



Changing X and Y Scale


The horizontal and vertical size of the window may be altered by pressing “X” or “Y”, respectively. The three available x scales and y scales translate into nine possible window sizes. The size or scale of the window is very important because when the screen is moved to a graphic buffer or a graphic buffer

is moved to the screen, The Graphic Grabber uses the entire window. Note that the sensitivity parameter changes when you alter the x or y scale. Also note that the window wraps around if it is larger than the screen can accommodate.

Changing the Graphic Buffer

The Graphic Grabber 3.0 utilizes two different graphic buffers. All functions (i.e. Load, Save, Reverse, Move buffer to screen, Move screen to buffer) dealing with a graphic buffer will be performed on the current buffer selected. To select a graphic buffer, press “1” for buffer one or “2” for buffer two.

Move Graphic to Buffer

Pressing “U” or the **right arrow key** will invoke a routine that examines the pixels in the window and creates an 88 x 52 Print Shop graphic out of them. When using the larger windows, more resolution is lost. With the window at its smallest size (xscale=1, yscale=1), there is a one-to-one correspondence between the pixels on the screen and the pixels that make up a

Print Shop graphic; therefore, no resolution is lost when the window is at its smallest size.

Lit-Dot Sensitivity

The lit-dot sensitivity (press “S” to change) is used when a graphic is moved to the current buffer area. When the window is larger than

The Graphic Grabber 3.0 Source Code

```

*-----*
*          GRAPHIC GRABBER 3.0          *
*          MACHINE LANGUAGE STUFF      *
*          COPYRIGHT                    *
*          HARDCORE COMPUTIST 1985     *
*-----*

00E2- HGR.Y      .EQ $E2
00E0- HGR.X      .EQ $E0
0030- MON.HMASK .EQ $30
00E5- HGR.HORIZ .EQ $E5
00FE- ROWS      .EQ $FE      NUMBER OF ROWS LEFT
00FF- COLUMNS .EQ $FF      NUMBER OF COLUMNS LEFT
0026- MON.GBASL .EQ $26
00FD- BITS      .EQ $FD      NUMBER OF BITS LEFT
    
```

```

00FC- XSAVE .EQ $FC
00E7- HGR.SCALE .EQ $E7
00FB- STATUS.SQ .EQ $FB

F417- HPOSN .EQ $F417
F467- MOVE.LEFT .EQ $F467 MOVES HGR CURSOR LEFT
F48A- MOVE.RIGHT .EQ $F48A
F4D5- MOVE.UP .EQ $F4D5 MOVES HGR CURSOR UP
F504- MOVE.DOWN .EQ $F504 MOVES CURSOR DOWN
0000- DOT.STAT .EQ $0 HOLDS NUMBER OF UNLIT DOTS IN THREE BY THREE AREA
0001- XSCALE .EQ $1
0002- YSCALE .EQ $2
0003- XSCALE1 .EQ $3
0004- YSCALE1 .EQ $4
0005- STARTX .EQ $5 AND $6
F65D- XDRAW .EQ $F65D

```

```

.OR $4000 HGR2 IS UNUSED MOSTLY
.TF OBJ.GRAPHIC GRABBER

```

```

-----*
* MOVE SCREEN TO BUFFER *
*-----*

```

```

4000: 20 B0 40 JSR SETUP INITIALIZE STUFF
4003: 20 D3 40 DO.ROW JSR STARTROW SETUP FOR EACH ROW
4006: A9 00 DO.NEXT LDA #0 ZERO UNLIT DOT COUNTER
4008: 85 00 STA DOT.STAT
400A: A5 30 COUNT.EM LDA MON.HMASK GET MASK BYTE
400C: 29 7F AND #$7F IGNORE MSB
400E: 31 26 AND (MON.GBASL),Y MASK THE BYTE
4010: D0 02 BNE ALONG LIT, NO COUNT
4012: E6 00 INC DOT.STAT UNLIT, COUNT IT
4014: 20 42 40 ALONG JSR MOV.ALONG DO ALL THE DOTS DEPENDING ON THE SCALE
4017: 90 F1 BCC COUNT.EM NOT DONE COUNTING THE DOTS, KEEP GOING
4019: A5 00 LDA DOT.STAT NUMBER OF DOTS LIT
401B: C9 01 CMP #1 SENSITIVITY C=1 - DOT LIT
401D: 3E 00 58 ROLLER ROL $5800,X PUT BIT INTO PICTURE
4020: C6 FD DEC BITS BYTE DONE?
4022: D0 0A BNE NXT.COL NOPE!
4024: A9 08 LDA #8 FIX BITS
4026: 85 FD STA BITS
4028: E8 INX NEXT POS
4029: D0 03 BNE NXT.COL IF NO PAGE CROSSING
402B: EE 1F 40 INC ROLLER+2 NEXT PAGE
402E: C6 FF NXT.COL DEC COLUMNS DONE WITH ROW?
4030: D0 D4 BNE DO.NEXT NOPE!
4032: 20 3A 40 JSR NEXT.Y GET NEXT Y POSITION
4035: C6 FE DEC ROWS DONE WITH GRAPHIC?
4037: D0 CA BNE DO.ROW NOPE!
4039: 60 RTS

403A: 18 NEXT.Y CLC ADD Y SCALE TO Y POSITION
403B: A5 E2 LDA HGR.Y
403D: 65 02 ADC YSCALE DEFAULT SCALE = 1
403F: 85 E2 STA HGR.Y
4041: 60 RTS

4042: 20 8A F4 MOV.ALONG JSR MOVE.RIGHT GO RIGHT
4045: C6 03 DEC XSCALE1 DONE GOING RIGHT?
4047: D0 28 BNE CLC.RTS NOPE,

```

Continued on next page

CORE

its smallest size, The Graphic Grabber examines an area of pixels on the hi-res screen for each pixel that will appear in the current graphic buffer. The area of pixels that are examined differs and can be found by multiplying the x-scale by the y-scale. The lit-dot sensitivity tells The Graphic Grabber 3.0 how often to make the corresponding graphic pixel "lit". When the sensitivity is set as high as it will go (this changes with each window size), then there only has to be one lit dot in order for the corresponding graphic pixel to be lit. Likewise, when the sensitivity is set to the lowest setting, all of the dots have to be lit in an area before the corresponding graphic pixel will be lit.

You will discover that different types of pictures require different sensitivity settings to be properly "grabbed". For example, digitized pictures or pictures that have a lot of gray scale in them usually work best on a sensitivity that is half of the maximum. Pictures that have few lit dots will probably require high sensitivity settings and pictures that have many lit dots will almost always require low sensitivity settings.

Move Buffer to Screen

Pressing "**←**" or **left arrow key** will fill the window with the graphic found in the current graphic buffer. If the window is large than normal (x-scale or y-scale greater than 1) then the graphic will be stretched accordingly. In other words, one pixel in the graphic buffer could be placed on the screen as several pixels if the window is larger than normal.

Disk Access

When the disk access routine is invoked (press "**D**") you will be presented with a mini menu which provides 4 options: 1) Load hi-res screen, 2) Save hi-res screen, 3) Load graphic buffer and 4) Save graphic buffer. You must then press the number of your choice (if you accidentally typed the "**D**" command, you may exit this routine by pressing **ESCAPE**).

When asked for a filename, you may press **RETURN** to get a directory. To exit the current function, press **ESCAPE**. After the filename, you may put slot, drive and volume parameters. For example, if you wanted to load a file called **EXODUS** from drive two, you would type

EXODUS,D2

as a filename. Just typing a slot drive or volume number will display the directory of that disk. For example, if you wanted a **CATALOG** of the disk in slot 6, drive one you would type

,S6,D1

as the filename.

 * GENERAL SET UP ROUTINES *

```

40B0: A9 08      SETUP  LDA #8      EIGHT BITS IN A BYTE
40B2: 85 FD      STA BITS
40B4: A9 58      MSB    LDA #$58    GET BUFFER MSB
40B6: 8D 1F 40   STA ROLLER+2 STORE IT
40B9: 8D A0 40   STA GET.BIT+2
40BC: A2 00      LDX #0     INITIALIZE X
40BE: A9 34      LDA #52    52 ROWS
40C0: 85 FE      STA ROWS
40C2: A5 E0      LDA HGR.X  SAVE STARTING X
40C4: 85 05      STA STARTX
40C6: A5 E1      LDA HGR.X+1
40C8: 85 06      STA STARTX+1
40CA: A5 01      LDA XSCALE COPY SCALES
40CC: 85 03      STA XSCALE1
40CE: A5 02      LDA YSCALE
40D0: 85 04      STA YSCALE1
40D2: 60        RTS

40D3: A9 58      STARTROW LDA #88    88 COLUMNS
40D5: 85 FF      STA COLUMNS
40D7: A5 05      LDA STARTX MOVE STARTING X TO CURRENT X
40D9: 85 E0      STA HGR.X
40DB: A5 06      LDA STARTX+1
40DD: 85 E1      STA HGR.X+1
40DF: 20 77 41   JSR HPOSN1 CALCULATE CURRENT POSITION
40E2: 60        RTS
  
```

 * MOVE WINDOW *

```

40E3: 8D 10 C0   DONE.MOV STA $C010 CLEAR KEY
40E6: 20 3A 41   MOVE.W JSR DO.SQUARE
40E9: A0 50      LDY #80
40EB: A2 50      TEST.X LDX #80
40ED: 2C 00 C0   TEST.Y BIT $C000 KEYPRESS?
40F0: 30 08      BMI EVALUATE YUP
40F2: CA        DEX      WAIT A WHILE
40F3: D0 F8      BNE TEST.Y
40F5: 88        DEY
40F6: D0 F3      BNE TEST.X
40F8: F0 EC      BEQ MOVE.W

40FA: A5 FB      EVALUATE LDA STATUS.SQ SQUARE ON?
40FC: F0 03      BEQ SKIP.ER NOPE
40FE: 20 3A 41   JSR DO.SQUARE
4101: AD 00 C0   SKIP.ER LDA $C000 GET VALUE
4104: C9 C9      CMP #$C9 MOVE.UP?
4106: D0 09      BNE TST.J
4108: A5 E2      LDA HGR.Y
410A: F0 D7      BEQ DONE.MOV
410C: C6 E2      DEC HGR.Y
410E: 4C E3 40   JMP DONE.MOV
4111: C9 CA      TST.J   CMP #$CA MOVE LEFT?
4113: D0 09      BNE TST.K
4115: A5 E0      LDA HGR.X
4117: F0 CA      BEQ DONE.MOV
4119: C6 E0      DEC HGR.X
  
```

Continued on next page

```

120 IF PEEK (16384) + PEEK (16385) <> 208 THEN
  PRINT CHR$ (4) "BLOAD^OBJ.GRAPHIC^GRAB
  BER,A$4000"
130 HIMEM: 8191 : POKE 251 , 0 : W1 = 1 : W2 = 1 :
  POKE 230 , 32 : POKE 16723 , 43 : POKE 16702
  , 43
140 POKE 1 , 1 : POKE 2 , 1 : X = 96 : Y = 70 : W = 1
  : POKE 16716 , 51 : POKE 16737 , 51 : POKE
  16412 , W
150 A$ = "" : ESC$ = CHR$ (27) : K$ = CHR$ (9) +
  CHR$ (8) + CHR$ (21) + "DXY12SRP^C" +
  ESC$ + CHR$ (16)
160 GOSUB 860 : ONERR GOTO 820
170 REM GET A KEY
180 GOSUB 950 : POKE 225 , X / 256 : POKE 224 , X
  - PEEK (225) * 256 : POKE 226 , Y
190 CALL 16611 : X = PEEK (224) + PEEK (225) *
  256 : Y = PEEK (226) : GET A$
200 FOR A = W TO LEN (K$) : IF A$ <> MID$ (K$ , A
  , W) THEN NEXT : W1 = W - W1 : GOTO 180
210 ON A GOSUB 260 , 380 , 400 , 580 , 320 , 350 , 220
  , 230 , 470 , 450 , 510 , 240 , 420 , 990 , 1010
  : GOTO 180
220 POKE 16565 , 85 : RETURN
230 POKE 16565 , 88 : RETURN
240 W2 = W - W2 : RETURN
250 REM TAB TO NEXT POSITION
260 X = X + 92 : IF X > 192 THEN X = 4 : Y = Y + 61
  : IF Y > 140 THEN Y = 9
270 RETURN
280 REM DO MONITOR COMMAND
290 A$ = A$ + "N^D9C6G" : FOR A = W TO LEN (A$
  ) : POKE 511 + A , ASC ( MID$ (A$ , A , W) )
  + 128
300 NEXT : POKE 72 , 0 : CALL - 144 : RETURN
310 REM INCREMENT XSCALE
320 POKE W , PEEK (W) + W : IF PEEK (W) = 4 THEN
  POKE W , W
330 POKE 16702 , PEEK (W) * 88 - 45 : POKE 16723
  , PEEK (W) * 88 - 45 : POKE 16412 , 1 + PEEK
  (W) * PEEK (2) / 2 : RETURN
340 REM INCREMENT YSCALE
350 POKE 2 , PEEK (2) + W : IF PEEK (2) = 4 THEN
  POKE 2 , W
360 POKE 16737 , PEEK (2) * 52 - 1 : POKE 16716
  , PEEK (2) * 52 - 1 : POKE 16412 , 1 + PEEK
  (W) * PEEK (2) / 2 : RETURN
370 REM PUT GRAPHIC ON SCREEN
380 A$ = "4F00<5500.5A40M" : GOSUB 290 : CALL
  16499 : A$ = "5500<4F00.5440M" : GOTO 290
390 REM PUT GRAPHIC IN BUFFER
400 CALL 16384 : RETURN
410 REM CLEAR HI-RES SCREEN
420 POKE 34 , 20 : HOME : POKE - 16301 , 0 : PRINT
  : PRINT "OK^TO^CLEAR^HI-RES^SCREEN?^N"
  CHR$ (8) ; : GET A$ : PRINT A$ : IF A$ = "Y"
  THEN CALL 62450
430 TEXT : RETURN
440 REM REVERSE BUFFER
450 CALL 16785 : RETURN
460 REM GET NEW SENSITIVITY
470 VTAB 23 : HTAB 15 : INVERSE : PRINT "(1-"
  PEEK (W) * PEEK (2) )" : NORMAL : HTAB
  10 : POKE - 16301 , 0
480 HTAB 14 : GET A$ : VTAB 23 : HTAB 15 : PRINT
  SPC (15) : IF A$ < "1" OR A$ > STR$ ( PEEK
  (W) * PEEK (2) ) THEN RETURN
490 POKE 16412 , VAL (A$) : RETURN
500 REM JUMP TO NEW POSITION
  
```

CORE

```

510 POKE 34 ,20 : HOME : VTAB 22 : PRINT
"CURRENT^POSITION=>" X "," Y : PRINT
TAB( 25 ); : INVERSE : PRINT
"(0-192,0-140)"
520 NORMAL : POKE - 16301 ,0 : VTAB 24 : PRINT
"NEW^POSITION^(X,Y)=>" ;
530 FLASH : PRINT "" CHR$( 8 ); : NORMAL : WAIT
- 16384 ,128 : IF PEEK ( - 16384 ) = 155
THEN 560
540 INPUT "" ; A , B : A = INT ( A ) : B = INT ( B ) :
IF A < 0 OR A > 192 OR B < 0 OR B > 140 THEN 560
550 X = A : Y = B
560 HOME : TEXT : RETURN
570 REM DISK ACCESS
580 TEXT : POKE 34 , 2 : HOME : VTAB 4 : PRINT TAB(
14 ) "DISK^ACCESS" : VTAB 7
590 PRINT ""^1)^LOAD^HI-RES^SCREEN" : PRINT :
PRINT ""^2)^SAVE^HI-RES^SCREEN" : PRINT
600 PRINT ""^3)^LOAD^GRAPHIC^BUFFER^" ( PEEK
(16565 ) - 85 ) / 3 + 1 : PRINT : PRINT
""^4)^SAVE^GRAPHIC^BUFFER^" ( PEEK
(16565 ) - 85 ) / 3 + 1
610 PRINT : PRINT "WHICH?" CHR$( 8 );
620 GET A$ : IF A$ < "1" OR A$ > "4" THEN 660
630 ON VAL ( A$ ) GOTO 640 , 670 , 690 , 710
640 A$ = "LOAD^HI-RES^SCREEN" : GOSUB 740 : IF
F$ = ESC$ THEN 660
650 PRINT CHR$( 4 ) "BLOAD" F$ " , A$2000"
660 TEXT : GOTO 860
670 A$ = "SAVE^HI-RES^SCREEN" : GOSUB 740 : IF
F$ = ESC$ THEN 660
680 PRINT CHR$( 4 ) "BSAVE" F$
" , A$2000 , L$2000" : GOTO 660
690 A$ = "LOAD^BUFFER^" + STR$( ( PEEK (16565
) - 85 ) / 3 + 1 ) : GOSUB 740 : IF F$ = ESC$
THEN 660
700 PRINT CHR$( 4 ) "BLOAD" F$ " , A" 256 * PEEK
(16565 ) : GOTO 660
710 A$ = "SAVE^BUFFER^" + STR$( ( PEEK (16565
) - 85 ) / 3 + 1 ) : GOSUB 740 : IF F$ = ESC$
THEN 660
720 PRINT CHR$( 4 ) "BSAVE" F$ " , A" 256 * PEEK
(16565 ) " , L$240" : GOTO 660
730 REM GET A FILENAME
740 HOME : PRINT : HTAB 20 - LEN ( A$ ) / 2 : PRINT
A$ : VTAB 6 : PRINT TAB( 8 ) "RETURN^DO
ES^A^DIRECTORY"
750 VTAB 12 : PRINT "FILENAME=>" ; : FLASH :
PRINT "" CHR$( 8 ); : NORMAL
760 WAIT - 16384 , 128 : IF PEEK ( - 16384 ) = 155
THEN GET F$ : RETURN
770 INPUT "" ; F$ : IF PEEK ( 512 + LEN ( F$ ) ) THEN
VTAB PEEK ( 37 ) : CALL 64578
780 FOR A = 512 + LEN ( F$ ) TO 768 : IF PEEK ( A
) THEN F$ = F$ + CHR$( PEEK ( A ) ) : NEXT
790 IF F$ = "" OR LEFT$( F$ , W ) = " , " THEN HOME
: PRINT CHR$( 4 ) "CATALOG" F$ : PRINT :
PRINT "PRESS^A^KEY^" ; : GET F$ : GOTO 740
800 RETURN
810 REM ON ERR HANDLER
820 CALL 16773 : ERR = PEEK ( 222 ) : IF ERR > 254
THEN RESUME
830 TEXT : HOME : VTAB 12 : PRINT "ERROR^#" ERR
CHR$( 7 )
840 F$ = ESC$ : FOR A = 1 TO 1500 : NEXT
850 REM PRINT COMMANDS
860 HOME : PRINT "GRAPHIC^GRABBER^3.0" SPC(
4 ) "BY^RAY^DARRAH^III"
870 VTAB 4 : PRINT "P=JUMP^TO^NEW^POSITION"
: PRINT "1-2=GRAPHIC^BUFFER"

```

Continued from previous page

```

411B: 4C E3 40      JMP DONE.MOV
411E: C9 CB      TST.K    CMP #SCB    MOVE RIGHT?
4120: D0 0A      BNE TST.M
4122: A5 E0      LDA HGR.X
4124: C9 C0      CMP #SC0
4126: B0 BB      BCS DONE.MOV
4128: E6 E0      INC HGR.X
412A: D0 B7      BNE DONE.MOV
412C: C9 CD      TST.M    CMP #SCD    MOVE DOWN?
412E: D0 3D      BNE OTHER.KEY
4130: A5 E2      LDA HGR.Y
4132: C9 8C      CMP #140
4134: F0 AD      BEQ DONE.MOV
4136: E6 E2      INC HGR.Y
4138: D0 A9      BNE DONE.MOV

DO.SQUARE
413A: 20 77 41      JSR HPOSN1  GET FIRST POS
413D: A2 2B      LDX #43     SCALE=43
413F: A9 00      LDA #0      ROT=0
4141: 20 6E 41      JSR XDRAW1  DRAW IT
4144: A2 2C      LDX #44     SCALE=44
4146: A9 00      LDA #0      ROT=0
4148: 20 6E 41      JSR XDRAW1
414A: A2 33      LDX #51     SCALE=51
414D: A9 10      LDA #16     ROT=16
414F: 20 6E 41      JSR XDRAW1  DRAW IT
4152: A2 2B      LDX #43     SCALE=43
4154: A9 20      LDA #32     ROT=32
4156: 20 6E 41      JSR XDRAW1  DRAW IT
4159: A2 2C      LDX #44     SCALE=44
415B: A9 20      LDA #32     ROT=32
415D: 20 6E 41      JSR XDRAW1
4160: A2 33      LDX #51     SCALE=51
4162: A9 30      LDA #48     ROT=48
4164: 20 6E 41      JSR XDRAW1  DRAW IT
4167: A5 FB      LDA STATUS.SQ
4169: 49 01      EOR #01
416B: 85 FB      STA STATUS.SQ
416D: 60      OTHER.KEY RTS

416E: 86 E7      XDRAW1    STX HGR.SCALE
4170: A2 8F      LDX #SHAPE
4172: A0 41      LDY /SHAPE  MSB
4174: 4C 5D F6      JMP XDRAW

4177: 86 FC      HPOSN1    STX XSAVE   DON'T MESS UP X
4179: A5 E2      LDA HGR.Y
417B: A6 E0      LDX HGR.X
417D: A4 E1      LDY HGR.X+1
417F: 20 17 F4      JSR HPOSN  CALCULATE STARTING ADDR
4182: A6 FC      LDX XSAVE  RESTORE X
4184: 60      RTS

4185: 68      FIX.ERR   PLA        POP STACK
4186: A8      TAY
4187: 68      PLA
4188: A6 DF      LDX $DF    FIX POINTER
418A: 9A      TXS
418B: 48      PHA
418C: 98      TYA
418D: 48      PHA

```



```

418E: 60          RTS
418F: 05 00      SHAPE .HS 0500

```

```

-----*
*          REVERSE CURRENT BUFFER          *
-----*

```

```

4191: A2 03      LDX #3      NUMBER OF PAGES TO REVERSE
4193: A0 00      LDY #0       INITIALIZE Y
4195: AD B5 40    LDA MSB+1  GET CURRENT BUFFER MSB
4198: 85 FF      STA COLUMNS STORE ZERO PAGE
419A: 84 FE      STY ROWS   ZERO LSB
419C: B1 FE      REVERSE LDA (ROWS),Y GET BYTE
419E: 49 FF      EOR #$FF  REVERSE IT
41A0: 91 FE      STA (ROWS),Y PUT IT BACK
41A2: C8         INY       NEXT BYTE
41A3: D0 F7      BNE REVERSE NOT END OF PAGE YET
41A5: E6 FF      INC COLUMNS NEXT MSB
41A7: CA         DEX       DONE WITH GRAPHIC?
41A8: D0 F2      BNE REVERSE NOPE, CONTINUE
41AA: 60         RTS       YUP, EXIT

```

```

4160: A2 33 A9 30 20 6E 41 A5 $F9F3
4168: FB 49 01 85 FB 60 86 E7 $1F8B
4170: A2 8F A0 41 4C 5D F6 86 $C037
4178: FC A5 E2 A6 E0 A4 E1 20 $5AD1
4180: 17 F4 A6 FC 60 68 A8 68 $A252
4188: A6 DF 9A 48 98 48 60 05 $0C77

```

```

4190: 00 A2 03 A0 00 AD B5 40 $DA78
4198: 85 FF 84 FE B1 FE 49 FF $B403
41A0: 91 FE C8 D0 F7 E6 FF CA $B415
41A8: D0 F2 60          $DC44

```

The Graphic Grabber 3.0 BASIC Checksums

10	-	\$BADD	520	-	\$1167
20	-	\$9B13	530	-	\$DE9A
30	-	\$4D3B	540	-	\$26AC
40	-	\$AD92	550	-	\$1CB6
50	-	\$C899	560	-	\$A0C9
60	-	\$FF65	570	-	\$B200
70	-	\$A3BF	580	-	\$B615
80	-	\$A900	590	-	\$D543
90	-	\$924D	600	-	\$0C5A
100	-	\$CB63	610	-	\$C4AB
110	-	\$BD13	620	-	\$3178
120	-	\$FD22	630	-	\$AE11
130	-	\$C468	640	-	\$0BA6
140	-	\$E23E	650	-	\$330B
150	-	\$EA33	660	-	\$EBE1
160	-	\$2928	670	-	\$B453
170	-	\$DA05	680	-	\$D5FC
180	-	\$8A99	690	-	\$145E
190	-	\$68B3	700	-	\$E4FB
200	-	\$EBD8	710	-	\$BAEE
210	-	\$5FFF	720	-	\$BC1E
220	-	\$77CE	730	-	\$5972
230	-	\$DDBB	740	-	\$D866
240	-	\$6A19	750	-	\$5ECC
250	-	\$77A0	760	-	\$A4CB
260	-	\$2022	770	-	\$8002
270	-	\$21A6	780	-	\$32AA
280	-	\$E60D	790	-	\$7B8E
290	-	\$3113	800	-	\$5880
300	-	\$A0E6	810	-	\$5DA3
310	-	\$1643	820	-	\$4EC7
320	-	\$0F49	830	-	\$065E
330	-	\$0BB4	840	-	\$067F
340	-	\$DC15	850	-	\$4EB7
350	-	\$6E75	860	-	\$929E
360	-	\$7510	870	-	\$FC32
370	-	\$8084	880	-	\$5DD7
380	-	\$400E	890	-	\$9333
390	-	\$9ECB	900	-	\$027C
400	-	\$76AE	910	-	\$9C2A
410	-	\$7E53	920	-	\$E846
420	-	\$F908	930	-	\$208A
430	-	\$C187	940	-	\$A760
440	-	\$C33F	950	-	\$38B2
440	-	\$C33F	950	-	\$38B2
450	-	\$BB5A	960	-	\$074E
460	-	\$6AC5	970	-	\$8063
470	-	\$2A31	980	-	\$0A5E
480	-	\$6AC4	990	-	\$DDD5
490	-	\$3B6D	1000	-	\$7C7D
500	-	\$6147	1010	-	\$17B0
510	-	\$1288			

```

880 PRINT : PRINT "X=CHANGE^XSCALE" TAB (25)
      "C=CLEAR^SCREEN" : PRINT "Y=CHANGE^YSCALE" ;
890 PRINT TAB (25) "ESC=EXIT" : PRINT : PRINT
      "S=LIT^DOT^SENSITIVITY" : PRINT
      "<SPACE>=TOGGLE^PARAMETERS"
900 PRINT : PRINT "<=>MOVE^GRAPHIC^TO^SCREEN" : PRINT
      "->MOVE^GRAPHIC^TO^BUFFER"
910 PRINT : PRINT "D=DISK^ACCESS" : PRINT
      "R=REVERSE^BUFFER"
920 PRINT : PRINT "<CTRL>|=JUMP^TO^NEXT^GRAPHIC^POSITION" : PRINT
      "?=SWITCH^BETWEEN^THIS^SCREEN^AND^HI-RES"
930 RETURN
940 REM PRINT PARAMETERS
950 HTAB W : VTAB 22 : PRINT "XSCALE=>" PEEK (1)
      ) TAB (20) "YSCALE=>" PEEK (2) : PRINT
960 PRINT "SENSITIVITY=>" PEEK (16412) TAB (20)
      "BUFFER=>" ( PEEK (16565) - 85 ) / 3 + 1 ;
970 POKE - 16302 + W2 , W2 : POKE - 16304 + W1 , W1
      : POKE - 16297 + W1 , W1 : RETURN
980 REM EXIT PROGRAM
990 TEXT : HOME : END
1000 REM PRINTOUT HI-RES SCREEN
1010 RETURN

```

The Graphic Grabber 3.0 Hexdump

4000:	20 B0 40 20 D3 40 A9 00	\$43CC	40F0:	30 08 CA D0 F8 88 D0 F3	\$EFB3
4008:	85 00 A5 30 29 7F 31 26	\$9C92	40F8:	F0 EC A5 FB F0 03 20 3A	\$470F
4010:	D0 02 E6 00 20 42 40 90	\$DFC8	4100:	41 AD 00 C0 C9 C9 D0 09	\$1E9D
4018:	F1 A5 00 C9 01 3E 00 58	\$33BF	4108:	A5 E2 F0 D7 C6 E2 4C E3	\$5D62
4020:	C6 FD D0 0A A9 08 85 FD	\$1A68	4110:	40 C9 CA D0 09 A5 E0 F0	\$90C3
4028:	E8 D0 03 EE 1F 40 C6 FF	\$A888	4118:	CA C6 E0 4C E3 40 C9 CB	\$899F
4030:	D0 D4 20 3A 40 C6 FE D0	\$A1A2	4120:	D0 0A A5 E0 C9 C0 B0 BB	\$BE53
4038:	CA 60 18 A5 E2 65 02 85	\$09A8	4128:	E6 E0 D0 B7 C9 CD D0 3D	\$8A1B
			4130:	A5 E2 C9 8C F0 AD E6 E2	\$6223
			4138:	D0 A9 20 77 41 A2 2B A9	\$22C5
			4140:	00 20 6E 41 A2 2C A9 00	\$638C
			4148:	20 6E 41 A2 33 A9 10 20	\$CBD0
			4150:	6E 41 A2 2B A9 20 20 6E	\$01AB
			4158:	41 A2 2C A9 20 20 6E 41	\$4035

COPY II+ 5.0:

A Review

By Dr. Phillip Romine

IS IT REALLY NEW?

Happily, I can offer an unequivocal yes and no! The changes in Copy II+ 5.0 are only evolutionary in the bit copier's performance, but revolutionary in convenience and ease of use. Furthermore, the limited copy protection on the earlier versions has been removed and the price has remained at an astonishingly low \$39.95.

The improvements in Copy II+ Version 5.0 can be summarized under three headings: changes and additions to the bit copier program (extensive), changes in bit copier copying performance (modest), and changes or additions to the utility programs (few, but valuable).

Changes and Additions to the Bit Copy Program

The most revolutionary change to the bit copier is in the automatic loading of parameters. Although Nibbles Away has offered this feature for some time, with Locksmith adding it also more recently, I find the Copy II+ approach much more convenient because the parameters are on the disk with the bit copier; no disk swapping is necessary. Also, with the lack of copy protection, if the parameter list ever grows too large to fit onto the entire Copy II+ disk, the user can easily put the bit copier and parameters on a separate disk. The parameters are also very easy to find and retrieve. If you know the name of the parameter you want, you simply type it in. If not, Copy II+ will provide its directory for you to view and also make available a convenient search feature. If you don't know exactly how a parameter is listed - as is usually the case - simply enter the first few characters of the probable title. The program presents a list of all the parameters with the beginning you specified and allows you to choose among them with a keystroke. The user also retains the options of entering parameters manually or of

recopying specific tracks after the parameters have been entered either automatically or manually.

I must confess that I was unimpressed with this new auto-loading of parameters at first. I had used the Nibbles and Locksmith auto-load features and, furthermore, had never considered the entry of parameters manually any real chore. But after only a few uses, I was hooked. Now I would rather never again enter parameters manually.

A second valuable addition to the bit copier program is a high resolution disk scan. It seems identical to the Locksmith Quickscan feature which, in my opinion, is the best such function of its type. It gives the user a visual overview of which disk tracks contain data, allows you to spot spiraled tracks and half tracks, and doesn't cause the disk drives to run incessantly (unlike similar functions on some other popular bit copiers).

A new sector copy program has also been included in the bit copier package. This addition, like Back-It-Up's "Copy Sectors" option, is intended for more reliable copying of protected disks with a near-normal DOS. The Copy II+ Manual states that this program, "can handle a few protection schemes more readily than the manual bit copier," but does not

elaborate. The sector copy program also handles the disks with bit insertion protection that used to be copied by the copy disk from the main menu. Some additional parameters have been added for use by this program.

The remaining new bit copier functions are to be used in the creation, editing, storing, and use of your own parameters which you may want to add to the autolisting. A very minimal text editor is provided for working with the parameter files.

Bit Copier Performance

The Copy II+ manual and promotional literature make two basic claims about the bit copier's performance: 1) that it offers completely revised nibble counting and track analysis routines, and 2) has the ability to copy more disks than before. I was able to verify one of these.

The automatic nibble counting of the new Copy II+ is much improved over the earlier 4.4C version. On my disk drives, the optimum copying speeds seem to be 199.5 millisecond/revolution on the original drive, and a slower 201 m/r on the copy drive. Even with this discrepancy, the keep track length (nibble count) option was able to reconcile the track lengths most of the time. With earlier versions of the program, I always had to use one drive or turn to EDD for the nibble count. An added advantage is that the count is high speed. On some competing disks, especially Locksmith, the time required to nibble count a single track can seem interminable.

Unfortunately, I was not able to verify the claim that Copy II+ can now backup more disks. I don't question the assertion but, on the disks which I had available at the time, with but one exception, there was no discernible difference in copying performance between the 5.0 and 4.4C versions. In general, these disks ranged from difficult (Simon Says - MECC Sunburst) to very difficult (Pop 'R Spell - Milliken and the Britannica series). All of these, by the way, yield readily to Super IOB with the

Swap Controller. On these disks, and some others of similar difficulty, either 4.4C and 5.0 would both make a workable copy, or neither version would do so. The one exception was an easier disk (Sensible Speller - Sensible Software). The 5.0 version of Copy II+ would make a good copy almost every time, while the hit rate for 4.4C on that program was only about 33% without recopying certain tracks. So perhaps my disk sample was too hard for a fair test. But for the present, I must assume modest, if any, improvement in copying ability when nibble counting is not required. I should add, however, that 4.4C is already an excellent copier, one of the best available, so a lack of marked improvement after the recent upgrading to version 4.4C isn't surprising. A new programmable version of Nibbles Away is due any day now and Echo II is due out this spring. Performance comparisons of these two updates with Copy II+, E.D.D. and Locksmith should be fascinating.

Changes and Additions to the Utilities Section

All of the old utility program functions which have proven so useful and popular are still on the Copy II+ disk. There are also some welcome additions.

One of the first things that you might notice when viewing the main menu is that the order of the options has changed slightly. "CATALOG DISK" is no longer the first entry; it has been replaced by "COPY". In addition, "BIT COPY" no longer appears in the main menu at all but has been moved to the "COPY" submenu. The Sector Editor is, thankfully, much improved. There is now an option to read the sectors from a specified file, a disassembly option, a (text) screen dump, read next (+) or last (-) sector, and a search

function.

The patch option now offers a custom patch which allows the changing of address and data headers and trailers, ignoring of checksums, and selection of the different encoding methods. This custom patch ability should expand the number of protected diskettes which can be read and sector edited by the program. A new utility allows the user to alphabetize the disk catalog, view the rearranged titles, and make them permanent if the aesthetics are pleasing. One minor inconvenience of the option is that, once it has been used, there is no way to "unalphabetize" a disk's catalog short of getting out another disk utility such as Sensible Software's Disk Organizer.

The main menu COPY DISK has been updated so that it is faster on a //c or //e with 128K of memory and (according to the manual) more reliable. It is now strictly a fast-copy program for unprotected disks. The Locksmith 5.0 Fast Copy utility running on a 128K machine can copy a disk about 10 seconds faster than can Copy II Plus "COPY DISK" but, with LS 5.0 Fast Copy, you must reboot after you have finished making your copies.

The "VERIFY DISK" option also has been updated in that, when a bad sector is found, no recalibration of the disk arm will occur. Apparently, someone at Central Point realized that a recalibration is unnecessary since a program verifying a disk must know what track it is trying to read.

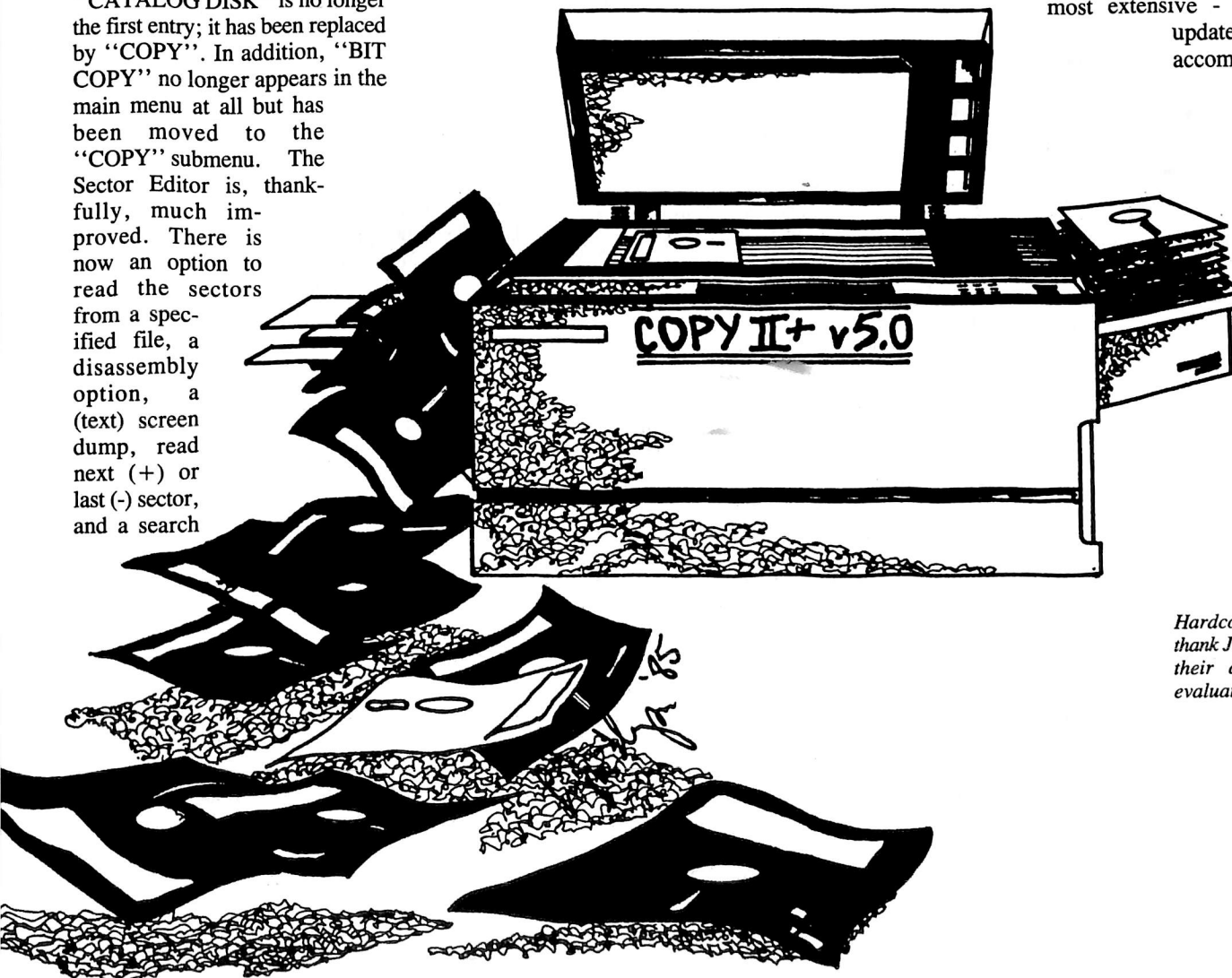
The final change that I noticed in the performance of the utilities involves those functions which use filenames. With the 5.0 update you can now specify a file type parameter when entering filenames. This makes it easy when, for instance, you want to copy all of the Applesoft files from one disk to another. To do this you could just type "E=,A" while in the "COPY FILES" function and all of the Applesoft BASIC would be transferred. This is a small, but quite useful, feature.

Miscellaneous Observations

The new manual is attractive and (with a few exceptions) clearly written, has large print, and is easy to follow. I did notice that the explanation of the sector editor's custom patching instruction could be more clear and that the discussion of the sector copy program does not adequately describe when this feature should be used.

Unfortunately, a few parameter listings in the manual show the wrong default values. For example, parm 00 (number of read retries) defaults to 01 according to the manual. On my copy, however, it defaults to 03 as it did in Copy II+ 4.4C. Likewise, the manual gives a default value of 02 for parm 01 (number of write retries); on the disk it too defaults to 03. But these are minor errors. Overall, the manual is beautifully set-up and extremely well written.

The Copy II+ parameter listing is still the most extensive - and the most frequently updated - of all parameter lists accompanying the popular bit copiers. The parameters and utilities alone are worth more than the price of the disk. Therefore, I remain convinced that Copy II+ is the most software available for the money in any category.



Hardcore COMPUTIST would like to thank Jeff Lucia and Joshua Wilson for their additional assistance in the evaluation of Copy II+ 5.0.

: A Hardware Evaluation

The Know-Drive

from Abacus Enterprises, Inc.

By Clay Harrell

Every once in a while I come across a real find. Like the time I stumbled across a grandmother selling her deceased grandson's "old '63 Chevy with low mileage" for \$500, and it turns out to be a split window Corvette! Well this time the "find" was some little known, but extremely useful, Apple hardware. While over at a friend's house, I noticed a little board stuck to the side of his Apple. Curious, I asked him what it was. He proceeded to give me a demo of the strange device, a "Know-drive".

He first removed the cover of his computer to reveal that the Know-Drive was a 128K RAM card, and more! From the card came an 18-inch ribbon cable connecting to a 3-inch by 1.5-inch card stuck to the side of the Apple case that contained several switches.

For the first demo, he used the 128K card like a RAM disk. But, so what? RAM disks are old hat, though in this case ABACUS' proprietary ABCdos resulted in a lightning fast load of binary files. Seeing that I still wasn't very impressed, my friend explained that this card alone used either the Address or Data lines for bank selects, thereby achieving compatibility with Titan and Legend RAM cards. No small feat as software written for one is incompatible with the other.

The next demonstration involved the use of Locksmith 5.0's Fast Copy program. One pass copies (yes, one pass!) were now available. This reduced the time required to make copies and, if you have a reason for producing mass copies, this is a real advantage!

Now came the real demos. My friend pulled out a couple of disks and booted one labeled "Know-Drive BACK-to-BACK". After the program was loaded, he pressed a push-button switch on the small auxiliary board attached to the Know-Drive and booted up Apple Panic. He pressed the switch again and booted Apple Galaxian. Once more, he pressed the switch and booted up Copy II Plus.


"So what?", I proclaimed. At this point, he then pressed the switch and three key presses later Apple Panic was back on the screen, just where we had left off! He pressed

the switch again, followed by three key presses, and Apple Galaxian was back up! He repeated the procedure once more and we were transferring files using Copy II Plus! Another push of the button and we were back to Apple Galaxian, right where we left off, in the matter of seconds and without booting any disks!

The last part of the demo involved deprotecting a program. My friend booted a disk called "Know-Drive PLAY-BACK". After PLAY-BACK was loaded, he booted his Apple Galaxian original disk. After loading was complete he pressed the switch on the small board attached to his Apple case. There appeared what looked very much like the menu on a Replay II card. All the registers were displayed and also a disassembly of the current address being executed and the next 9 lines of code! The menu at the bottom of the screen allowed him to set the video page desired, save the program, restore the program, continue the program, enter the monitor, or boot another disk. This was incredible!

Now came the most amazing part! First my friend saved the game to the Know-Drive, then continued the game and let an Alien kill him. He then pressed the switch and restored the program to continue playing. He was back in the position where he had been before the Alien had killed him!

Now he pressed the switch again and saved the program. This saved the program to the Know-Drive at his present position so we could continue the game from that point. If we died, we could always go back and try again in less than 2 seconds! The real advantage to this is that you essentially have an infinite number of lives and the possibility of ever climbing best scores. In other words, you could practice any part of a game over and over until you succeeded, and then press on to the next challenge.

The menu even offered the choice of entering the monitor. Using this handy option, you could enter the monitor, do some work, and then press  and return to the Know-Drive menu. You could even continue your program! A nice feature.

Now my friend booted the PLAY-BACK disk and saved the whole game to a normal DOS disk. No funny formats, just good ole unmodified DOS 3.3 or, if you prefer, fast ABCdos. To continue the game, he just booted this disk and pressed one key!

Finally, he demonstrated the program packer. This automatically searched the bootable disk and tried to pack the program into a file, only requiring a 48K Apple and

the single file to re-run the program. To pack the program required pressing only one key. Simple, and no special knowledge was needed at all! The contents of memory was saved into one file that could be BRUN from any Apple!

Other than the switch to use Data or Address line bank selects, there were two more switches on the remote card. One of these allowed write-protect of the RAM card. This feature is invaluable when trying to protect code you have saved in the RAM card from being destroyed by a protected program. The last switch allowed RESET to point to the motherboard ROM or your own monitor routines moved to the RAM card. The latter would allow entry to the monitor from any program.

Basically, I have discovered that the KNOW-DRIVE gives you a lot more than a 128K RAM card. With it you have:

- Titan or Legend 128K card compatibility
- One pass Locksmith 5.0 Fast Copies
- A concurrent programming computer (have 3 programs in memory at once)
- More available data memory when using spreadsheet programs
- Deprotection tools that rival the Replay II card
- A way to enter the monitor easily and without disturbing memory
- A tool for debugging and developing your own programs and applications

Pricing on the product line is: Know-Drive 128K \$335, Play-Back \$69, Back-to-Back \$59, but my friend got his package through a group purchase by his Apple club at substantially better than distributor cost!! Depending on the number of people ordering, you could cut the price in half for all the products. For more information contact ABACUS Enterprises at PO Box 1836, Detroit, Michigan 48231, (313) 524-2444. There is also a modem bulletin board available at (313) 524-0238 to get technical help 24 hours a day.

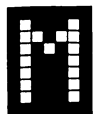
If you need a 128K RAM card for the Apple II series (except //c) and clones, look no further. The Know-Drive is simply the best. If you need a copy card, the Know-Drive is still the best. It offers everything a Replay II card offers but is far more versatile. After all, all you can do with a copy card is deprotect. With the Know-Drive you can do that, plus multi-programming, plus one pass copies, and the list goes on.

This is one hardware device that is certainly worth the money for any Apple enthusiast. ABACUS calls their KNOW-DRIVE: The smartest 128K card for the Apple. Own one and you'll get religion, too.

(Apple //e owners should be aware that if their computer does not have socketed RAM chips they will either have to modify their motherboard, or purchase a \$20 adapter from Abacus to install a Know-Drive.)

An Improved **BASIC/Binary Combo**

By James L. Parham



Most programmers need to use, at one time or another, a short binary routine with their BASIC programs. The problem is where to put it so that it will be safe from being overwritten. Aside from that well-used section of page \$3, a favorite place to hide code is below DOS, but above the DOS buffers. Here the code remains protected against most potential disasters such as HIMEM changes, RESET, etc. There are, however, three main disadvantages to this technique:

- 1) the binary file requires extra disk space,
- 2) the HELLO program or other program must load the binary file into memory from the disk, and
- 3) the binary file takes memory space away from your BASIC programs.

The first two disadvantages can be easily overcome if the binary program(s) are short and you are willing to keep your disks as "Slaves". All that is required is that you place your binary file or files in memory at \$9C00, make a few minor DOS changes and initialize as many disks as you need. Your binary file will be placed on the initialized disks in an unused part of the DOS track 0. Whenever the disk is booted, your binary file will automatically be reloaded into memory and the DOS buffers adjusted accordingly. The last disadvantage can be overcome, if necessary, by reducing the number of buffers with the "MAXFILES" command. The default is 3 but 2 will usually be sufficient and will leave you with more program memory than you originally had.

How To Proceed

First make sure you have a standard DOS in memory. Then, in immediate mode, type in the following commands:

```
POKE 40193,155  
CALL 42964  
POKE 40205,156
```

Place the disk containing your binary file(s) in drive 1 and type the following:

```
BLOAD(ANY BINARY FILE),A$9C00
```

Substitute the actual name of your binary file for (ANY BINARY FILE). Repeat for as many binary files as you have, adjusting the beginning address (\$9C00) as each file is loaded. Total length of all files must be no more than 256 bytes, but see the section below under the heading "Modifications" if you need to use a second page.

Now, make sure your HELLO program in memory is correct. Load or create one if necessary. Place a new disk in drive 1 and initialize it. Go ahead and INIT as many disks as you need.

Your initialized disks, when booted, will automatically load in your binary code at \$9C00. The DOS buffers will be created just under your code and will protect it from being overwritten. Now, all you have to do is make sure that any programs that use these binary files use their correct memory address.

How It Works

When you issue the first POKE command, the DOS buffer pointers at \$9D00 & \$9D01 are made to point one memory page lower than normal. Instead of pointing to \$9CD3, they will point to \$9BD3. A DOS call to \$A7D4 makes DOS re-create the buffers using the new buffer pointers. There is now room between the buffers and DOS itself to load in your binary code beginning at \$9C00. The second POKE command lowers the "DOS BEGINNING" pointers by one page. These pointers located at \$9D0C and \$9D0D normally point to \$9D00, but will now point to \$9C00. Now DOS thinks that your binary code at \$9C00 is a part of itself. Whenever a disk is initialized that code, along with your modified DOS, will be written to disk like any normal DOS.

To understand how this works, and why it

does not overwrite important DOS disk sectors, we must examine what happens when a "Slave" disk is booted. The book "BENEATH APPLE DOS" by Don Worth and Pieter Lechner was very helpful here. In the latter stages of a boot, 27 sectors from track 2, sector 4 backward to track 0, sector A are loaded into memory from \$B5FF down to \$9B00. Track 0, sectors A & B are blank, being reserved for the relocation code of a "Master" disk. Even so, these blank sectors are loaded into memory at \$9B00 & \$9C00. DOS then proceeds to create its buffers from \$9600 through \$9CFF which includes the two normally blank pages. However, DOS doesn't care what is loaded into this part of its buffer space on boot. Whenever the buffers are used, the data will be overwritten.

Since we have modified DOS to create its buffers one page lower than normal, the data loaded from track 0, sector B to memory \$9C00 is not subject to overwrite. If we place our binary file there, it is safe from buffer overwrite as well as from HIMEM changes resulting from basic or DOS commands.

Modifications

It is just as easy to use two pages as is to use one. Sector A of track 0 will be loaded in memory at \$9B00. Change the first POKE value to 154 and the second POKE value to 155. Also BLOAD your binary program(s) beginning at \$9B00. The total length of the binary program(s) must not exceed 512 bytes.

Systems Supported

These procedures have been verified only on an Apple][+ and //e. They should work equally well on any DOS 3.3 system, but will not work on other DOS versions including PRODOS.

Deprotecting Sargon III

By Kit LAU Yu-kit & Polly CHAN Yuk-yi

Requirements:

Apple][Plus or equivalent
Super IOB v1.2

My first attempt at removing the copy protection from Sargon III took the form of a boot code trace. I performed the usual steps of moving the controller ROM into RAM, modifying it so that it would JuMP into the monitor after loading track 0, sector 0 and examining the code starting at \$801. Following the code at \$801 wasn't an easy task.

I could see that the first thing it did was relocate itself into page \$1 (the 6502 stack area). Next it read a sector into \$400, did some tricky stuff and then continued with the boot at \$400.

I didn't care to find out what the boot code's next action was as this would involve patching the \$801 code and writing a memory move routine to examine the stuff that is supposed to execute on the text page. Instead, I noticed a disk read routine in page \$8 that resembled the normal DOS sector read code. Instead of looking for the usual sector markers, this code was looking for AA D5 AB (address field prolog), DE AB (address field epilog), AA D5 EB (data field prolog) and ED AA (data field epilog).

Taking out my handy dandy track nibble editor (Nibbles Away J), I checked to see if these were indeed the marks on the Sargon III disk. I found that track 0, sector 0 and tracks \$C through \$22 were normal and track 0, sector 1 through track B, sector F was protected using the marks I had discovered from the code at \$801.

Next, I wrote a Super IOB controller that would copy my Sargon III disk using the information I had gathered about it. The first time I used the controller, I got a drive error on track 1, sector 5. This made me think that I might have overlooked a few marker changes.

So, again I used my Nibbles Away J, to examine trk 1, sctr 5. It appeared to be in the same format as the remainder of the track, so I tried the controller again. This time I got an error on track 0, sector A.

My first thought was, "Perhaps these markers are hard to distinguish," so I added the extended error retry (from Hardcore

COMPUTIST No. 14) to my controller and tried it again. This time it copied the entire disk (with only a couple of growls from the disk drive) and I got no unreadable sectors.

Hooray!

Next I got out my sector editor and then modified track 0, sector 0 so that it would read normal markers. The resulting disk, when booted only, printed Sargon III at the top of the screen and the disk spun forever. I decided that there must be another read routine somewhere on the disk. Using my sector editor once again I located another read routine on track \$0, sector \$3. I promptly normalized this routine and booted the disk. To my surprise, the disk booted all the way up and allowed me to play an exciting game of chess!

The Controller

The resulting controller (after adding the sector edits) is listed at the end of this article. Note that while making the copy, Super IOB may recalibrate the disk arm on several sectors. This is OK as long as you don't get a drive error. If you should get an error, type "GOSUB 230 : RUN" and hope that next time it will be able to read that sector. Secondly, note that a pause of a couple seconds occurs when Super IOB goes to write the first time. This is due to all the sector edits that are performed, and is normal.

Step by Step

1) Get out your copy of Super IOB v1.2 (from Hardcore COMPUTIST No. 14) and install the controller printed at the end of this article.

2) Start up Super IOB

RUN

3) Answer the question and follow the prompts.

You should now have a fully deprotected copy of Sargon III (now put your Sargon III in a safe place along with your other original disks).

Many thanks to P.T. Boling for his assistance in the verification of this softkey.

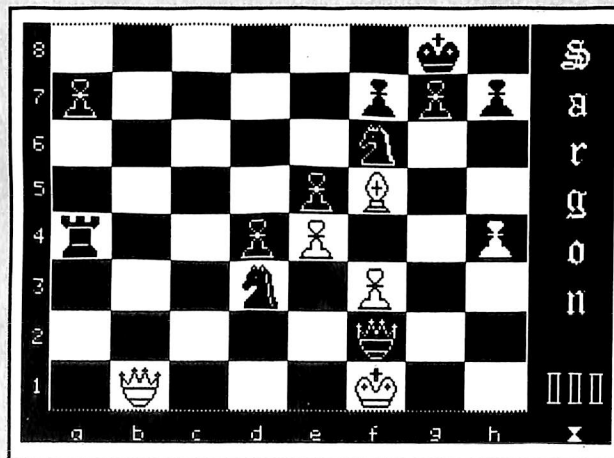
Sargon III Controller

```

1000 REM SARGON III
1010 TK=0:ST=0:LT=35:CD=WR:POKE 48573
,128
1020 T1=TK:GOSUB 490:RESTORE:IF TK<12
AND TK>0 THEN GOSUB 190:GOSUB 210:
GOSUB 170
1030 GOSUB 430:GOSUB 100:ST=ST+1:IF ST
=1 AND TK=0 THEN GOSUB 190:GOSUB 210
:GOSUB 170
1035 IF ST<DOS THEN 1030
1040 IF BF THEN 1060
1050 ST=0:TK=TK+1:IF TK=12 THEN GOSUB 230
1055 IF TK<LT THEN 1030
1060 GOSUB 230:GOSUB 490:TK=T1:ST=0:IF
TK=0 THEN GOSUB 2000
1070 GOSUB 430:GOSUB 100:ST=ST+1:IF ST
<DOS THEN 1070
1080 ST=0:TK=TK+1:IF BF=0 AND TK<LT THEN
1070
1090 IF TK<LT THEN 1020
1100 HOME:PRINT "DONE^WITH^COPY":END
2000 A$="274D:D5^N^2756:AA^N^2760:96^N^
278B:D5^N^2794:AA^N^279E:AD":GOSUB
2020
2010 A$="2A1A:D5^N^2A23:AA^N^2A2D:96^N^
2A57:D5^N^2A60:AA^N^2A6A:AD"
2020 A$=A$+"^N^D9C6G":FOR A=1 TO LEN(A$
):POKE 511+A,ASC(MID$(A$,A,1))
+128
2030 NEXT:POKE 72,0:CALL-144:RETURN
5000 DATA 170,213,171,170,213,235
5010 DATA 222,171,237,170
    
```

Controller Checksums

1000 - \$356B	1080 - \$E693
1010 - \$B166	1090 - \$7CE0
1020 - \$1B3A	1100 - \$8716
1030 - \$4594	2000 - \$5FBD
1035 - \$E523	2010 - \$153D
1040 - \$7DD4	2020 - \$5EAB
1050 - \$9302	2030 - \$07E8
1055 - \$1856	5000 - \$C8A3
1060 - \$054A	5010 - \$D56E
1070 - \$0D4F	





One way to remove copy protection that usually works (with perseverance) is to perform a process known as a boot code trace. This process is explained in detail in several previous issues of this magazine, but I will go over it anyway for the benefit of readers who are new.

When you turn on the computer, a series of subroutines is executed

to load in data from the disk. The first of these subroutines can be found at \$C600 in ROM (if your disk drive controller is in slot 6). This subroutine moves the disk drive head to track 0, and reads sector zero into memory locations \$800-\$8FF. It then jumps to memory location \$801 (this program and its function is explained in detail in the article starting on page 9 of this issue). Memory location \$801 normally contains a routine that loads in a much larger block of memory and then executes it.

When a boot code trace is performed, the subroutine at \$C600 is moved to RAM and modified in such a way that it will stop after loading all the correct data into page \$8. This allows you to list memory at \$801 and find out where the next stage of the boot is loaded in. The code at \$801 is then modified in such a way that it loads in all the data it normally would, but stops before it executes this code. This allows you to list the next stage of the boot and modify it as well. If you are able to follow the boot far enough into the program, you will eventually find the protection subroutine. Once this has been isolated, it is usually very easy to remove the protection by editing

the disk sectors on which the routine has been stored.

Applying This To Wizardry

On Wizardry, the first stage of the boot (\$800-\$900) loads in a large amount of data and then exits to \$B69E. At \$B69E there is a routine which clears memory and performs several memory moves before exiting to \$6827. At \$6827 more memory moves are performed before the program goes into a loop through memory location \$6E. This loop is what powers the Pascal language and is extremely difficult to trace. Rather than spend weeks doing a complex trace of the Pascal loop, I decided to try some other method of isolating the protection code.

Searching The Disk

When the Apple disk drive is reading data from a disk, the byte that is under the drive head can be retrieved by reading memory location SC0EC (for slot 6). Since this is the only location that shows what the disk drive is reading, it is constantly accessed by any routine that reads information from the disk drive. This is normally done with the command LDA SC08C,X.

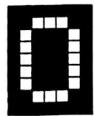
And since a nibble count routine accesses the disk drive, it seemed extremely likely to me that this command was used within the routine. With this in mind, I used The CORE Disk Searcher



Wizardry 1: Proving Grounds of the Mad Overlord
Wizardry 2: Knight of Diamonds
 Sir-Tech Software, Inc.
 6 Main Street
 Ogdensburg, N.Y. 13669
 (315) 393-6633
 \$49.95

Requirements:

Apple II Plus or equivalent
 A disk editor
 Wizardry I or II
 2 blank sides of a disk



One of my favorite adventure games is Sir Tech's Wizardry. It has given me many hours of enjoyment. However, I have always been concerned about a disk crash, since the sophisticated copy protection makes it almost impossible to make a backup copy of the disk. I have tried several times to duplicate the disk with EDD III, but even with the correct parameters it is difficult (the best I have been able to come up with is something that boots 2 out of 3 times). I decided, therefore, that the only solution was to find the copy protection and remove it.

Following is an in-depth explanation of how I was able to remove the protection from this fine fantasy/adventure game. If you wish to skip the explanation, merely proceed to the section entitled, "The Softkey".

Boot Code Trace

The first step of any deprotection job is to determine exactly what the copy protection is. As you may or may not know, Wizardry is in a completely COPYable format. But because a COPYA version will not boot, this reveals that there is a form of copy protection present on the disk.

When I examined the documentation for Wizardry, I noticed that the troubleshooting section contains a comment about adjusting your drive speed if your original does not work. This message is a clue that a protection technique (a nibble count in this case), is employed. This technique uses a routine to check the length of certain tracks on the disk. If these lengths are not what they should be, the disk will not boot.

Softkey for:

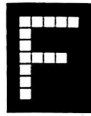
The **Wizardrys**

I. Proving Grounds of the Mad Overlord

II. Knight of Diamonds

By Taco van Ieperen

The Softkey



Following is a step-by-step procedure that describes how you can deprotect your original disks for Wizardry I & II. I have applied this technique only to the version of Wizardry 1 dated 10-MARCH-82.

Wizardry I: The Proving Grounds Of The Mad Overlord

1) Copy both sides of Wizardry 1 with COPYA or some other whole disk copier. Be sure to label the copy disks as to what they are.

2) Run any disk editor and edit track \$22, sector \$04 of your copied boot side. Starting at byte \$A4 enter the following sequence of bytes:

```
A0 00 AE 28 8B BD 29 8B
85 0D 91 02 E8 C8 BD 29
8B 91 02 85 0E E8 8E 28
8B 60 00 10 12 57 05 13
12 8C 05 08 12 53 05 E7
29 8D 05 17 12 53 05 12
12 92 05 D0
```

3) Write this sector to your disk.

4) Put a write protect tab on your copied boot side.

5) Use your sector editor to edit Track \$13, Sector \$00 of your SCENARIO MASTER COPY. Starting at byte \$C0, enter the following:

```
AE DC 20 BD DD 20 85 0D
8D 00 20 E8 BD DD 20 8D
01 20 85 0E E8 8E DC 20
8D E8 C0 60 00 23 12 61
06 23 12 62 06
```

6) Write the sector to your disk.

Your disk should now be completely deprotected. I have played the entire game through twice and have encountered no troubles at all.

Here's a hint about playing the game: when you find the level where Werdana is, do not move right in any of the rooms unless you want to go back to where you started.

Wizardry II: Knight of Diamonds

1) Copy both sides of Wizardry II with COPYA or some other whole disk copier. Be sure to label the copy disks as to what

they are.

2) Run any disk editor and edit track \$22, sector \$0E of your copied boot side. Starting at byte \$CB, enter the following sequence of bytes:

```
A0 00 AE 7F 8C BD 80 8C
85 0D 91 02 E8 C8 BD 80
8C 91 02 85 0E E8 8E 7F
8C 60 00 C9 09 42 09 57
09 2E 05 C7 09 4B 09 4B
09 26 05
```

3) Write this sector to your disk.

4) Put a write protect tab on your copied boot side.

5) Use your sector editor to edit Track \$12, Sector \$01 of your SCENARIO MASTER COPY. Starting at byte \$C0 enter the following:

```
AE D9 20 BD DA 20 8D 00
20 85 0D E8 BD DA 20 E8
8D 01 20 85 0E 8E D9 20
60 00 8D 09 20 05 27 05
81 05 B7 E0 00
```

6) Write the sector to your disk.

I have also played this game quite a bit, but have not been able to test the softkeyed version completely due to my inability to make it past level 4.

One hint (to anyone stuck in level one): If you find a passage that seems endless, you are being teleported back to where you moved from. Cast the MALOR spell for 3 east and you should be OK!

If The Procedure Does Not Work For You

Wizardry has been updated several times during the last few years. If your version is different, read the section on how I determined the game's protection. Use the techniques I have used to find the direct disk access routines and look for a routine that ends by loading some data from \$0D and \$0E and storing it in (\$02),Y. This routine is your nibble count. I have given a list of all the steps I used, but these should be done on a working copy of the disk, NOT THE ORIGINAL. To produce a backup of the disk use any bit copier with Synchronized copy and auto nibble count from tracks \$0-\$11. Use normal copy from tracks \$12-\$22. If the disk still does not boot, continue copying tracks \$0-\$11 with the above parameters until it copies.

Continued from previous page

(from Hardcore COMPUTIST No. 12) to isolate the track and sector on which routines that have a LDA \$C08C,X were located. I accomplished this by searching the entire disk for the bytes 8C C0.

On the Scenario Master I found these bytes in the following tracks and sectors:

```
Track 00 00 14 14 14 1E 1E 22
Sector 0D 0E 02 05 06 07 08 04
```

Now I had to determine which (if any) one of these eight sectors contained the nibble count. To do this I used a sector editor with disassembly capabilities and disassembled the data on these sectors. With the exception of track \$22, sector \$04, all of the routines seemed to be used for legitimate disk access.

Upon a close examination of the disassembly of track \$22, sector \$04 I found that this routine turned on the disk drive and then waited for sector zero to swing under the disk head. When this had happened, it did a series of complicated checksums and stored the result in several memory locations. This was, without a doubt, the nibble count routine.

Although I had found where the nibble count was stored and what it looked like, I had not yet determined how to get rid of it. Many nibble count routines verify the checksums before returning to the main program. This would allow you to put a RTS on the first byte, thus avoiding the verification of the checksums as well as the actual routine. Since the routine I had discovered produced checksums that would be verified by other parts of Wizardry, there was no chance of merely putting a RTS on the first line. If I had done this, no checksums would have been produced and Wizardry would have considered this incorrect and crashed.

Next, I decided to search for all the locations on the disk that read the memory addresses in which the checksums from the nibble count are stored. I did an extensive search of the disk but came up blank. There are so many ways to access a memory location from assembly language that it would take much too long to think of them all, let alone search an entire disk for them.

My only remaining option was to attempt to rewrite the copy protection itself. The modification I had in mind was rather straightforward. It involved applying a patch to the end of the routine so that the routine would look up the correct values for the various nibble counts on a table and store them in the memory locations where the nibble count value would normally be.

To apply this patch I first had to determine the correct values. To do this I went back to my disk editor and found that the last bytes of the nibble count routine were:

```
9D 88 C0 STA $C088,X ;TURN OFF DRIVE
60 RTS
```

I edited these bytes so that, instead of turning off the disk drive, the computer would go to

a subroutine that would keep track of all the values the nibble count came up with. This involved changing the BD 88 C0 to 4C 9E B6 (from STA \$C088,X to JMP \$B69E). I chose \$B69E because it was a location that contained a rather trivial routine when I attempted a boot code trace. The commands to do the boot code trace follow:

1) Move the boot code to RAM

9600<C600.C700M

2) Store a 0 (BRK) in \$890

96F8:A9 00 8D 90 08 4C 01 08

3) Execute the modified boot

9600G

4) Set up a JuMP into the monitor

9251:4C 59 FF

5) Execute the routine normally at \$B69E

B69EG

6) Enter the subroutine to save the nibble count values

**B69E:AE B6 B6 A5 0D 9D B7 B6
B9A6:E8 A5 0E 9D B7 B6 E8 8E
B9AE:B6 B6 60 00**

7) Continue the boot

6827G

I allowed the disk to continue booting until the message, "INSERT SCENARIO MASTER" appeared. I then hit RESET and examined the memory at \$B6B6. Stored there was a listing of all the correct nibble count values. I recorded the values and then scanned through memory to pinpoint where the nibble count routine was loaded. I found it at \$8B00. Finally, I edited the routine so that it ended with a subroutine, which replaced the nibble count values found on the disk with the correct values.

When I tried the disk, LO and BEHOLD! It booted! I then went through a similar procedure on the scenario master and found that it was also protected with a nibble count (located on track \$13, sector \$0). When I removed it in a similar way, I had a COPYAable version of Wizardry 1!

Wizardry II Protection

It has often been said that many companies use one protection method on several of their games. I was pleased to find that Sir Tech, the company that markets Wizardry, is one of these companies. Upon examining Wizardry 2, I found that the boot code was exactly the same, and that the nibble count routines were also the same. The only difference in the protection of Wizardry 1 and 2 was that the nibble count lies on a slightly different track and it generates different values with Wizardry 2.

ADVENTURE TIPS

* Masquerade Phoenix Software

Bras can be used as weapons.
When was the last time you searched popcorn for foreign substances?
Mr. Topp has a good sense of humor.
Do what the name of the game says you should do.

* Karateka Broderbund

Tip for the eagle: When you first hear the screech, stop in the middle of the screen. When the eagle flies out, punch in the direction he is flying. You should hit him on the third punch.

* Deadline Infocom

Follow George. He knows something important.
Check out the note pad and calendar carefully.
Use the pencil.
Even the Robners get mail.
Eavesdropping can be helpful.
For fun: Search the males, snort the sugar, drink the liquor twice, look in the toilet, read the end of the book!

* Zork I Infocom

Ever try climbing up trees?
Can't find the empire? Furnishings are movable.
Can't open the egg? Thieves are skillful.
Bats can smell, too.
Never fool with dead bodies.
Cyclops' got tired after nice hearty meals.

**Contributed by Michael Dinerman*

† Pirate Adventure Adventure International

Parrots get rid of snakes.
Crocodiles get rid of parrots.
Fish can be found North into the lagoon.
You can get nails out of the floor with a claw hammer.
Your crew is sleeping in a sleeping bag.
Walk "30" and dig outside the monastery.

† Contributed by Andy Welch

§ Mystery House Sierra On-Line

Have you been to the moist basement?
Once you light the candle, do not enter the dining room.
Look at the hole in the rug.
You will need the towel.
If you're stuck in the forest, try the command "up".
Something in the kitchen should be moved.

§ Contributed by Jim Rutherford

‡ Escape From Rungistan Sirius Software

Don't let the bear's good looks fool you.
Some books come before others.
Paper in the cash register is useless.
Don't catch with bare hands.
Natives can be amazed with predictions.

‡ Transylvania Penguin Software

Sailing without a passenger will do no good.
Religious articles can be very helpful.
The cloak may hold something useful.

‡ Contributed by Troy Bontrager

+ Beyond Castle Wolfenstein Muse

When you have the bomb and are trying to find Hitler, if there is a doorway going up, go "up". If there is a door going to the right and none going up, go "right". Only as a last resort, go "left" (if there are no other doorways).

Trying to get out? Go "down" if there is a doorway down. If there is no doorway, go "left". If there are no other doors, go "right".

+ The Hitchhiker's Guide to the Galaxy Infocom

Are they going to demolish your house? Do the most illogical thing. (Hint: What do you do when you're tired?)
Don't accept the towel from Ford.
Do what Ford says.
In the dark about something? It may smell.
Feeling weak? Eat something.

+ Contributed by Dwayne Claud

Rely on Hardcore COMPUTIST* to provide you with up-to-date deprotection methods for all the new games and programs available for the Apple II line of computers.

*The magazine for the serious user of Apple II computers.

Subscription rates:

Please check one

- US \$25
- Canada, US 1st Class \$34
- Mexico \$39
- Foreign \$60
- SAMPLE- US \$3.50
- SAMPLE- Foreign \$4.50

() Yes! Start my subscription now.

Name _____

Address _____

City _____ St _____ Zip _____

Country _____

VISA/MC _____ Exp _____

Signature _____

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

Are you a NEW SUBSCRIBER?

BACK ISSUES of Hardcore COMPUTIST and *CORE are PACKED with information that you won't want to miss.

Hardcore COMPUTIST 19: Softkeys for Rendezvous With Rama, Peachtree's Back To Basics Accounting System, HSD Statistics Series, Arithmetickle, Arithmekicks and Early Games for Children / Double Your ROM Space / The Games of 1984: In Review- Part II / Towards a Better F8 ROM / The Nibbler: A Utility Program to Examine Raw Nibbles From Disk

Hardcore COMPUTIST 18: Softkeys for the Scholastic Version of Bank Street Writer, Applewriter IIe, SSI's Non-RDOS Disks, BPI Accounting Programs and DesignWare Programs / Installing a Free Sector Patch Into Applewriter IIe / The Games of 1984: In Review / 65C02 Chips Now Available / Checksoft v2 / Simple Copy Protection

Hardcore COMPUTIST 17: Softkeys for The Print Shop, Crossword Magic, The Standing Stones, Beer Run, Skyfox, and Random House Disks / A Tutorial For Disk Inspection and the Use Of Super IOB / The Graphic Grabber For The Print Shop / The Lone Catalog Arranger Part Two / S-C Macro Assembler Directives (Reprint)

Hardcore COMPUTIST 16: Softkeys for Rescue Raiders, Sheila, Basic Building Blocks, Artsci Programs, Crossfire, Sensible Speller for ProDOS and Sideways / Secret Weapon: RAMcard / The Controller Writer / A Fix For The Beyond Castle Wolfenstein Softkey / The Lone Catalog Arranger Part 1

Hardcore COMPUTIST 14: Super IOB v1.2 Update / Putting Locksmith 5.0 Fast Copy Into a Normal Binary File / Softkeys for Seadragon, Rocky's Boots, Knoware, PFS Software, Computer Preparation SAT & MatheMagic / Batman Decoder Ring / REVIEW: Boulder Dash / A Fix For DiskEdit

Hardcore COMPUTIST 13: Softkeys for Laf Pak, Beyond Castle Wolfenstein, Transylvania and The Quest, Electronic Arts, Snooper Troops (Case 2), DLM Software, Learning With Leeper, & TellStar / CSaver: The Advanced Way to Store Super IOB Controllers / Adding New Commands to DOS 3.3 / Fixing A ProDOS 1.0.1 BSAVE Bug / REVIEW: Enhancing Your Apple / Locksmith 5.0 and the Locksmith Programming Language

Hardcore COMPUTIST 11: Copy II Plus 4.4C Update / PARMS for Essential Data Duplicator / Ultimaker III / Mapping of Ultima III / Ultima II...The Rest of the Picture / Softkeys for Sensible Speller, Ultima III, Softporn Adventure, The Einstein Compiler v5.3, & Mask of the Sun

Hardcore COMPUTIST 10: Controller Saver / Softkeys for The Arcade Machine, Bankstreet Writer, Minit Man, Sensible Speller IV, Essential Data Duplicator III, Krell LOGO, & Canyon Climber, ApplEar / REVIEWS: The 65SC802 & 65SC816 Chips and Dino Eggs / Examining Protected Applesoft Basic Programs / Crunchlist II / Parms for DB Master v4

Hardcore COMPUTIST 4: Ultima II Character Editor / Softkeys for Ultima II, Witness, Prisoner II, & Pest Patrol / Adventure Tips for Ultima II & III / Copy II Plus PARMS Update

Hardcore COMPUTIST 1: Softkeys for Data Reporter, Multiplan & Zork / PARMS for Copy II Plus / No More Bugs / APT's for Choplifter & Cannonball Blitz / Reviews: Replay, Crackshot, Snapshot & Wildcard copy cards

CORE 3 Games: Constructing Your Own Joystick / Compiling Games / GAME REVIEWS: Over 30 of the latest and best / Pick Of The Pack: All-time TOP 20 games / Destructive Forces / EAMON / Graphics Magician and GraFORTH / and Dragon Dungeon

CORE 2 Utilities: Dynamic Menu / High Res: Scroll Demo / GOTO Label: Replace / Line Find / Quick Copy: Copy.

CORE 1 Graphics: Memory Map / Text Graphics: Marquee, Boxes, Jagged Scroller / Low Res: Color Character Chart / High Res: Screen Cruncher, The UFO Factory / Color / Vector Graphics: Shimmering Shapes, A Shape Table Mini-Editor / Block Graphics: Arcade Quality Graphics for BASIC Programmers / Animation

(*CORE is no longer published as an independent quarterly magazine. Back issues not listed are no longer available.)

Send me the back issues indicated below:

HC 11	\$3.50	<input type="checkbox"/>
HC 19	\$3.50	<input type="checkbox"/>
HC 18	\$3.50	<input type="checkbox"/>
HC 17	\$3.50	<input type="checkbox"/>
HC 16	\$3.50	<input type="checkbox"/>
HC 14	\$3.50	<input type="checkbox"/>
HC 13	\$3.50	<input type="checkbox"/>
HC 10	\$3.50	<input type="checkbox"/>
HC 4	\$3.50	<input type="checkbox"/>
HC 1	\$3.50	<input type="checkbox"/>
CORE 3 Graphics	\$3.50	<input type="checkbox"/>
CORE 2 Utilities	\$3.50	<input type="checkbox"/>
CORE 1 Games	\$3.50	<input type="checkbox"/>

Name _____ ID# _____

Address _____

City _____ St _____ Zip _____

Phone _____

VISA/MC _____ Exp _____

Signature _____

Send check or money order to: Hardcore COMPUTIST, PO Box 110846-B, Tacoma, WA 98411. Most orders shipped UPS. Please use street address. Washington residents add 7.8% sales tax. Foreign orders add 20% shipping and handling. US funds drawn on US bank.

Void after August 30, 1985

.....An inside look at disk formats → **DISKVIEW**

.....Deprotecting disks with **SUPER IOB**

.....A quick and easy way to **UNLOCK HYPERSPACE WARS**

.....Taking a peek at **BOOT CODE TRACING**

.....List of Publisher abbreviations and **INTRODUCTION TO 'PARMS'**

.....The Compleat Guide to **LOCKSMITH PARAMETERS**

.....Step-by-Step guide to making backups using **NIBBLES AWAY II PARAMETERS**

5.....Technical notes and making backups using **BACK-IT-UP II + PARAMETERS**

38.....How to make backups using **COPY II PLUS PARAMETERS**

46.....Curing those Auto-Start ROM blue **HARDWARE SOLUTIONS**

47.....A MENU HELLO PROGRAM

51.....USING BOTH SIDES OF YOUR DISKETTES

.....Advanced Playing Techniques, or how to get **INSIDE CASTLE WOLFENSTEIN**

.....Understand Strings - **TEXT INVADERS**

If you took all the old Hardcores...
 Tore off the fancy covers...
 Deleted all the editorial material, out-of-date interviews and letters...
 Updated the remaining material and included
 an enormous list of bit copy parameters
 And packed it all into a single volume...
 You'd have the core of Hardcore Computing.

We call it:
The Best Of Hardcore Computing

Please send me:

The Best Of Hardcore Computing AND Program Disk.....\$19.95

The Best Of Hardcore Computing.....\$14.95

Program disk only.....\$9.50

Name _____ ID# _____

Address _____

City _____ St _____ Zip _____

Phone _____

Visa/MC _____ Exp _____

Signature _____

Send check or money order to: Hardcore COMPUTIST, PO Box 110846-B, Tacoma, WA 98411.
 Washington State residents add 7.8% sales tax. Foreign orders add 20% shipping & handling.
 US funds drawn on US bank.

Have you written an ARTICLE or PROGRAM you'd like to see published in Hardcore COMPUTIST ? We would like to hear from you!

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

Get 25 SS/DD 5 1/4" diskettes

That's ONLY 80¢ per disk!!
 (Sold in multiples of 25 ONLY.)

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

FOR ONLY \$20.00

Please send me _____ sets (25 disks per set) of 5 1/4" SS/DD diskettes. I have enclosed \$20.00 plus applicable shipping and handling for each set.

Name _____

Address _____

City _____ St _____ Zip _____

Country _____

VISA/MC _____ Exp _____

Signature _____

Send your disk 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.



Now-the ultimate back-up system!

EDD III[®] and TRAK STAR[™]

COMPLETE PACKAGE:
159⁹⁵*
 Includes • TRAK STAR • 2-Drive Adaptor
 • EDD III • Trak Star Patching Software

PRICED SEPARATELY:

TRAK STAR 99⁹⁵

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[®]-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[®] computer.
- For Apple II, II+, IIe and compatibles.

Personal checks, M.O., VISA & Mastercard accepted.



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



DISC COMMANDER

COPYRIGHT 1985 WILLIAM STEPHENS

DISK EDITING SYSTEM
 FOR THE APPLE II FAMILY OF COMPUTERS
 USING DOS 3.3 OR DOS 3.3 WITH
 PRONTO-DOS ENHANCEMENT

- 4 Editors • 13 Sub-editors • Completely menu driven • 1 or 2 drive operation • INSTRUCTORIALS and TECHNICAL INFORMATION on disk
- Sort catalog entries • Hide catalog entries • Create multiple heading catalogs • Create multi-column catalogs • Reassign lock/unlock and file type symbols, spaces, columns, catalog heading and program titles to any combination of NORMAL, FLASH, INVERSE, LOWER CASE or CONTROL characters • Rename all DOS commands • Rewrite all DOS error messages • Make a (non-VTOC dependent) disk map showing the TRACK/SECTOR LISTS, VTOC and CATALOG TRACK/SECTORS
- Trace out individual files • Change the HELLO program • Change any byte anywhere on a disk
- Make instructions, disk maps, track/sector displays and traces using your printer

AND MUCH

MUCH MORE

SO WHAT

SOFTWARE

10221 Slater Ave. Suite 103 Fountain Valley, Ca. 92708

TO ORDER: Send a check or money order (U.S. funds) for \$29.95 plus \$2.00 shipping (\$5.00 foreign) California residents add 6%.

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 June 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, Hardware COMPUTIST, PO Box 110846-C, Tacoma, WA 98411.

**Apple][,][+,][e,
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?

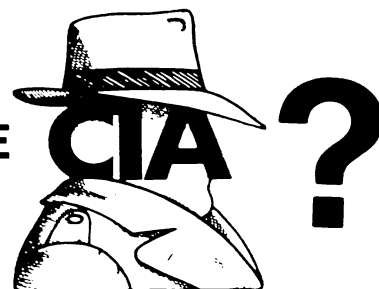
**It's simple now.
Just drop us a line.**

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.

**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+,][e, &][c 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

*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

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, PACK OF 25 \$32.50*
- STAND UP DISKETTE LIBRARY CASES \$2.75 EACH
4 for \$10.00
(SPECIFY COLOR CHOICES: BEIGE, BLACK, BLUE, GREEN, GREY, RED, YELLOW)
- JUMBO-SIZE FLIP TOP 70 DISKETTE FILE CASES \$11.00
- 140-DISKETTE LOCKING WOOD FILE CABINET \$33.00

PRINTERS

- PANASONIC P1090 \$199.00
- PANASONIC P1091 \$269.00*
- CANON PW 1080, 160 CPS \$339.00
- CITIZEN MSP-10, 160 CPS \$329.00*
- OKIDATA MICROLINE 92 \$369.00
- PANASONIC P1092 \$399.00
- EPSON RX-100, WIDE \$399.00
- CITIZEN MSP-15, WIDE \$469.00*
- CITIZEN MSP-20, 200 CPS \$469.00*
- SILVER REED 400, LETTER QUALITY \$269.00
- SILVER REED 500 \$299.00*
- AJ 831 w/KEYBOARD \$349.00
- TOSHIBA DOT MATRIX & LETTER QUALITY \$595.00

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 \$109.00*
- MICROFAZER PRINT BUFFER \$149.00
- PRINTER STAND \$14.00
- SWITCH BOX 3 PARALLEL PORTS \$79.00*
- IIc TO SERIAL PRINTER CABLE \$20.00*

FLOPPY DISK DRIVES

- FOURTH DIMENSION (FULL OR SLIMLINE) \$159.00
- MITAC \$139.00
- GAMMA \$129.00
- IIc CABLES FOR ABOVE DRIVES \$20.00
- DISK CONTROLLER \$59.00

HARD DISK DRIVES

- 10 MB \$695.00*

MONITORS

- GORILLA 12-INCH GREEN \$84.00
- USI 12-INCH AMBER \$99.00
- AVT AMERICA, 25 Mhz \$149.00*
- PANASONIC 1300 COMPOSITE & RGB \$239.00
- MONITOR STAND \$16.00*

MODEMS

- ZOOM TELEPHONICS 300 BAUD \$109.00
- IIe MODEM W/SOFTWARE \$159.00
- CENTAURI 300 BAUD \$145.00*
- PRO-MODEM 1200 \$349.00
- PRO-MODEM 1200A INTERNAL \$319.00

GRAPHICS DEVICES

- POWER PAD & STARTER KIT \$129.00

VIDEO & DISPLAY EQUIPMENT

- MICRO WORKS DIGITIZER \$299.00
- B & W CAMERA \$195.00
- COMPUTER EYES SYSTEM \$109.00*

GENERAL ITEMS

- 6-OUTLET POWER STRIP \$19.00
- 6-OUTLET WITH SURGE PROTECT \$25.00
- SURGE PROTECTOR \$11.00
- RF MODULATOR \$49.00
- CABLE GENDER CHANGER \$13.00*

COMPATIBLE COMPUTERS

- AX 5500 \$425.00
- LEADING EDGE IBM COMPATIBLE QUADLINK FOR APPLE SOFTWARE \$395.00

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.

GAME I/O DEVICES

- CH PADDLE STICKS \$37.00
- CH MACH II JOYSTICK \$37.00
- CH MACH III JOYSTICK \$45.00
- I/O PORT EXPANDER \$25.00
- 9-16 OR 16-9 ADAPTER \$9.00*

SLOT EXPANSION

- 16K RAM CARD \$49.00
- CENTAURI APS Z-80 CARD \$59.00
- SUPER SERIAL INT. CARD \$89.00
- TITAN ACCELERATOR IIe \$259.00*
- WILDCARD II COPY BOARD \$109.00
- MULTIPLE-SLOT EXPANSION CHASSIS \$149.00
- SINGLE-SLOT EXPANDER \$29.00
- QUICK-LOADER PROM BOARD \$149.00
- PROM BURNER \$119.00

SPECIAL PERIPHERALS

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

APPLE SOFTWARE

- PRINT SHOP \$39.00
- COPY II (5.0) \$29.00
- DISK DRIVE ANALYZER \$29.00*

SPECIAL!!!!!!
DISK DRIVE ANALYZER
\$29.00*

UPS SHIPPING: \$4.00 per order plus \$6.00 per printer and monitor.

CALL FOR OUR FREE CATALOGUE!

W ASSOCIATES (301) 652-4232

8231 WOODMONT AVENUE, BETHESDA, MARYLAND 20814

STORE HOURS: Monday through Thursday: 12 noon until 8 p.m.
Friday: 12 Noon until 6 p.m./Saturday: 11 a.m. until 5 p.m.

* DENOTES NEW PRICE OR ITEM