For The Serious User Of Apple][Computers

Hardcore COMPUTIST

Issue No. 14 $2.50

Softkey For Seadragon
Pg. 8

Super IOB: UPDATE
Pg. 10

Putting Locksmith 5.0 Fast Copy Into A Normal Binary File
Pg.15

Tracking Down Rocky's Boots
Pg. 22

Batman Decoder Ring
Pg. 16

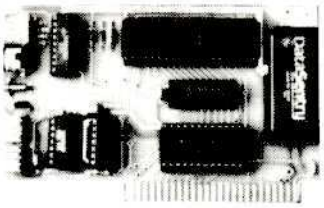
Hardcore COMPUTIST
PO Box 110846-T
Tacoma, WA 98411

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

APPLIED ENGINEERING

The BEST Peripherals for the BEST Computer!

THE NEW TIMEMASTER II H.O.



- Absolutely, positively, totally PRO-DOS and DOS 3.3 compatible.
- Time in hours, minutes, seconds and milliseconds (the ONLY PRO-DOS compatible card with millisecond capability).
- 24 hour military format or 12 hour with AM/PM format.
- Date with year, month, day of week and leap year.
- The easiest programming in BASIC.
- Eight software controlled interrupts so you can run two programs at the same time (many examples are included).
- Compatible with ALL of Apple's languages. Many sample programs for machine code, Applesoft, CP/M and Pascal on 2 disks.
- On-board timer lets you time any interval up to 48 days long down to the nearest millisecond.
- Rechargeable nickel-cadmium battery will last over 20 years.
- Two BSR/serial ports for future expansion.

Full emulation of all other clocks. Yes, we emulate Brand A, Brand T, Brand P, Brand C, Brand S and Brand M too. It's easy for the H.O. to emulate other clocks, we just drop off features. That's why the H.O. can emulate others, but none of the others emulate us.

The Timemaster II H.O. will automatically emulate the correct clock card for the software you're using. You can also give the H.O. a simple command to tell it which clock to emulate. This is great for writing programs for those poor unfortunates that bought some other clock card.

Of course, most programs will use the Timemaster II H.O. in its native mode, but its comforting to know that you can use programs written for other products without any modification.

REMOTE CONTROL

Our BSR X-10 interface option for the H.O. allows you to remotely control up to 16 lights and electrical appliances through your BSR X-10 home control system in your home or office. You're already wired because a BSR system sends its signals over regular 120 volt wiring. That means you can control any electrical device in your home or office without any additional wiring.

PRICE $129.00 BSR Option $49.00

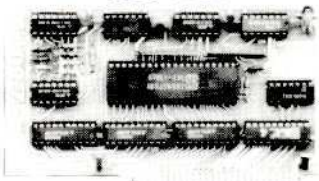
MemoryMaster IIe 128K RAM Card

- Expands your Apple IIe to 192K memory.
- Provides an 80 column text display.
- Compatible with all Apple IIe 80 column and extended 80 column card software (same physical size as Apple's 64K card).
- Can be used as a solid state disk drive to make your programs run up to 20 times FASTER (the 64K configuration will act as half a drive).
- Permits your IIe to use the new double high resolution graphics.
- Automatically expands Visicalc to 95K storage in 80 columns! The 64K config. is all that's needed, 128K can take you even higher.
- PRO-DOS will use the MemoryMaster IIe as a high speed disk drive.
- The 64K MemoryMaster IIe will automatically expand Apple Works to 55K storage. The 128K MemoryMaster IIe will expand Apple Works to 101K storage.
- High Speed disk emulation for BASIC, Pascal and CP/M is available at a very low cost. NOT copy protected.
- Documentation included, we show you how to use all 192K.

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

MemoryMaster IIe with 128K $249
Upgradeable MemoryMaster IIe with 64K $169
A. E. Extended 80 Column Card with 64K $139

Z-80 PLUS



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

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

PRICE $139.00

VIEWMASTER 80

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

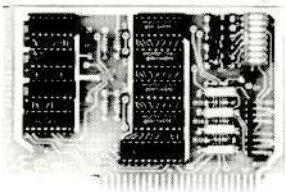
- TOTALLY Videx Compatible.
- 80 characters by 24 lines, with a sharp 7x9 dot matrix.
- On-board 40/80 soft video switch with manual 40 column override.
- Fully compatible with ALL Apple languages and software—there are NO exceptions.
- Low power consumption through the use of CMOS devices.
- All connections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

	PRICE	BUILT-IN SOFTSWITCH	SHIFT KEY SUPPORT	LOW POWER DESIGN	80 COLUMN MODE	7x9 DOT MATRIX	LIGHT PEN INPUTS	80 COLUMN OVERRIDE	INVERSE CHARACTERS
VIEWMASTER	159	YES	YES	YES	YES	YES	YES	YES	YES
SUPERTERM	MORE	NO	YES	NO	NO	NO	NO	YES	YES
WIZARD80	MORE	NO	NO	NO	NO	YES	NO	YES	YES
VISION80	MORE	YES	YES	NO	NO	YES	NO	NO	NO
OMNIVISION	MORE	NO	YES	NO	NO	NO	NO	YES	YES
VIEWMAX80	MORE	YES	YES	NO	NO	YES	NO	NO	YES
SMARTERM	MORE	YES	YES	NO	NO	NO	YES	YES	NO
VIDEOTERM	MORE	NO	YES	YES	NO	YES	YES	NO	YES

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

Thousands **SOLD** at $179 **NOW ONLY $159.00**

SUPER MUSIC SYNTHESIZER - END MOCKINGBOREDOM



- Complete 16 voice music synthesizer on one card. Just plug it into your Apple, connect the audio cable (supplied) to your stereo, boot the disk supplied and you are ready to input and play songs.
- It's easy to program music with our compose software. You will start right away at inputting your favorite songs. The Hi-Res screen shows what you have entered in standard sheet music format.
- Now with new improved software for the easiest and the fastest music input system available anywhere.
- We give you lots of software. In addition to Compose and Play programs, 2 disks are filled with over 30 songs ready to play.
- Easy to program in Basic to generate complex sound effects. Now your games can have explosions, phaser zaps, train whistles, death cries. You name it, this card can do it.
- Four white noise generators which are great for sound effects.
- Plays music in true stereo as well as true discrete quadraphonic.
- Full control of attack, volume, decay, sustain and release.
- Our card will play notes from 30HZ to beyond human hearing.
- Automatic shutoff on power-up or if reset is pushed.
- Many many more features.

PRICE $159.00

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

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

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

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

Public Domain Software

From the Computer Learning Center's Library

★SPECIAL OFFERS★

(Limited time only)
Order now!!!

S1 EAMON Collectors Offer

Every EAMON scenario and all EAMON utilities. Includes E1 through E76, U1 through U4, D5, D6. *82 EAMON volumes for only $250.*

EAMON adventures are fantasy role-playing games. Your character, which you create using the Master Disk, wanders through a fantasy world full of dangers and rewards. You are the master of your own fate; the course of the adventure is of your own making. Your decision to bargain, steal, fight or run affects the scenario and the action and eventual outcome is rarely the same.

S2 EAMON Introductory Offer

You get the EAMON Master with the Beginner's Cave, The Orb of Polaris, Operation Crab Key, The Feast of Carroll, EAMON Utility 1 and the Dungeon Designer version 6.0. Includes volumes E1, E33, E44, E45, U1, D6. *A popular sampling of 6 EAMON volumes for only $20.*

S3 Art & Graphic

Every Art & Graphic volume. Includes volumes 4 through 13, 93, 94. *12 volumes for only $40.*

S4 Business & Finance

All of the very popular Business & Finance volumes. Includes: 18 through 25. *8 volumes for only $28.*

S5 Games

Golf, hockey, ping-pong, poker, Hi lo, blackjack, cribbage, craps, chess, bowling, parachute, pinball, dragon maze, star trek, fizz bin, tank, keno, war lords, roulette, lunar lander and more. Includes volumes 36 through 57. *22 game-packed volumes for only $75.*

S6 Utilities

Disk utilities, printer programs, assemblers, and disassemblers. It's all here. Includes volumes 74 through 87. *Get 14 Utility volumes for only $48.*

S7 Tutor and Math Combo

From BASIC programming to multiple linear regression, you'll find what you need with these tutor and math volumes. Includes: 1, 2, 3, 59 through 63. *An 8 volume selection from Apple Tutor and Math & Statistics for only $28.*

S8 Chemistry and Math Combo

A choice selection for the scientifically inclined. Includes one Chemistry & Biology and five Math & Statisticss: volumes 26, 59 through 63. *All six volumes for only $20.*

S9 Public Domain Software (PDS) Introductory Offer

Includes one volume from each category: Apple Tutor, Art & Graphic, Astronomy, Aviation, Business & Finance, Chemistry & Biology, Demonstration, Education & School, Electronic & Radio, Food, Game, Hello & Menu, Math & Stats, Music & Sound, Passion, Pastime & Other, Unknown, Utility, Apple Bank, and Library. *22 volumes for only $75.*

There are over 175 volumes in the Computer Learning Center's Public Domain Library collection. All of these volumes will run on Apple][Plus computers and Apple-compatibles. Most will also run on the //e and //c. Each program in the collection has been donated to the public and has no copyrights attached. Therefore, each may be copied and distributed by anyone without regard for origin or ownership.

Public Domain Software is not commercial quality and is supplied as-is.

For more information on Public Domain Software from the Computer Learning Center(CLC), check the box on the order form to receive a copy of our NEW catalog.

Rush me the PDS items I have checked below. I understand that the volumes in each SPECIAL OFFERS package are packed on double-sided disks.(Offer expires March 31, 1985)

☐	$250	S1 EAMON Collectors Offer	☐	$48	S6 Utilities
☐	$20	S2 EAMON Introductory Offer	☐	$28	S7 Tutor & Math Combo
☐	$40	S3 Art & Graphic	☐	$20	S8 Chemistry & Math Combo
☐	$28	S4 Business & Finance	☐	$75	S9 PDS Introductory Offer
☐	$75	S5 Games			

☐ Please send me the NEW PDS Catalog

Name _____

Address _____

City _____ St _____ 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 & handling.

Many 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 generally 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? A softkey is a term which we coined to describe a procedure that removes, or at least circumvents, any copy protection that may be present on a disk. Once a softkey procedure has been performed, the disk can usually be duplicated by the use of Apple's COPYA program which is on the DOS 3.3 System Master Disk.

Following A Softkey Procedure: The majority of the articles in Hardcore COMPUTIST which contain a softkey will also include a discussion of the type of copy protection present on the disk in question and the technique(s) necessary to remove that protection. Near the end of the article, a step-by-step "cookbook" method of duplicating the disk will appear. Generally, the appropriate actions for the reader to perform will appear in boldface type. Examples are:

1) Boot the disk in slot 6

PR#6

or

2) Enter the monitor

CALL -151

It is assumed that the reader has some familiarity with his or her Apple, i.e. knowing that the RETURN key must be hit following the commands illustrated above.

Hardcore COMPUTIST tries to verify the softkeys which are published, although occasionally this is not possible. Readers should be aware that different, original copies of the same program will not always contain an identical protection method. For this reason, a softkey may not work on the copy of a disk that you own, but it may work on a different copy of the same program. An example of this is Zaxxon, by Datasoft, where there are at least 3 different protection methods used on various releases of the game.

Requirements: Most of the programs and softkeys which appear in Hardcore COMPUTIST require an Apple][+ computer (or compatible) with a minimum 48K of RAM and at least one disk drive with DOS 3.3. Occasionally, some programs and procedures have special requirements such as a sector editing program or a "nonautostart" F8 monitor ROM. The prerequisites for deprotection techniques or programs will always be listed at the beginning article under the "Requirements:" heading.

Software Recommendations: Although not absolutely necessary, the following categories of utilities are recommended for our readers who wish to obtain the most benefit from our articles:

1) **Applesoft Program Editor** such as Global Program Line Editor (GPLE).
2) **Disk Editor** such as DiskEdit, ZAP from Bag of Tricks or Tricky Dick from The CIA.
3) **Disk Search Utility** such as The Inspector, or The Tracer from The CIA.
4) **Assembler** such as the S-C Macro Assembler or Big Mac.
5) **Bit Copy Program** such as COPY II+, 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.

Three programs on the DOS 3.3 System Master Disk, COPYA, FID and MUFFIN, also come in very handy from time to time.

Hardware Recommendations: Certain softkey procedures require that the computer have some means of entering the Apple's system monitor during the execution of a copy-protected program. For Apple II+ owners there are three basic ways this can be achieved:
1) Place an INTEGER BASIC ROM card in one of the Apple's slots.
2) Install an old monitor or modified F8 ROM on the Apple's motherboard. The installation of a modified F8 ROM is discussed in Ernie Young's, "Modified ROMS", which appeared in Hardcore COMPUTIST No.6.
3) Have available a non-maskable interrupt (NMI) card such as Replay or Wildcard.

Longtime readers of Hardcore COMPUTIST will vouch for the fact that the ability to RESET into the monitor at will, greatly enhances the capacity of the Apple owner to remove copy protection from protected disks.

A 16K or larger RAM card is also recommended for Apple][or][+ owners. A second disk drive is handy, but is not usually required for most programs and softkeys.

Recommended Literature: The Apple II and II+'s come bundled with an Apple Reference Manual, however this book is not included with the purchase of an Apple //e. This book is necessary reference material for the serious computist. A DOS 3.3 manual is also recommended.

Other helpful books include:

Beneath Apple DOS, Don Worth and Peter Leichner, Quality Software. $19.95.

Assembly Lines: The Book, Roger Wagner, Softalk Books. $19.95.

What's Where In The Apple, William Lubert, Micro Ink. $24.95.

Typing In BASIC Programs: When typing in basic programs, you will often encounter a delta ("▲") character. These are the spaces you MUST type in if you wish your checksums to match ours. All other spaces are merely printed for easier reading and don't have to be keyed in. Any spaces after the word DATA that aren't delta characters MUST be ommitted!

It is a good idea to SAVE your BASIC program to disk frequently while typing it in to minimize the loss of data in the event of a power failure.

Checksoft: Checksoft is a Binary program that checks Applesoft programs to ensure that you have keyed them in properly. Every bin program we print has companion checksums which consist of the Applesoft program's line numbers and a hexadecimal (base 16) number for each line. After keying in a BASIC program, BRUN checksoft and compare the checksums for every line that Checksoft generates with those at the end of the program. If you use Checksoft and make a typing error, your checksums will differ from ours beginning at the line where you made the error.

Typing In Binary Programs: Binary programs are printed in two different formats, as source code and as object code in a hexadecimal dump. If you want to type in the source code, you will need an assembler. The S-C Macro Assembler is used to generate all the source code which we print. In our source code listings, the memory address of the each instruction is printed at the beginning of every line (instead of the line number).

Binary programs can also be entered directly with the use of the Apple monitor by typing in the bytes listed in the hexdump at the appropriate addresses. Be sure to enter the monitor with a **CALL -151** before entering the hexdump. Don't type the checksums printed at the end of each line of the hexdump and don't forget to BSAVE binary programs with the proper address and length parameters listed in the article.

Checkbin: Like Checksoft, Checkbin also generates checksums, but was designed to check binary (machine language) programs.

Whenever Hardcore COMPUTIST prints a hexdump to type in, the associated Checkbin generated checksums are printed after every 8 bytes and at the end of every line.

Checksoft and Checkbin were printed in Hardcore COMPUTIST No. 1 and the Best Of Hardcore Computing and are sold on Program Library Disk No. 1 and the Best Of Hardcore Library Disk.

Let Us Hear Your Likes And Gripes: New and longtime readers of Hardcore COMPUTIST are encouraged to let us know what they like and don't like about our magazine by writing letters to our INPUT column. Our staff will also try to answer questions submitted to the INPUT column, although we cannot guarantee a response due to the small size of our staff. Also, send your votes for the softkeys you would like to see printed to our "Most Wanted List."

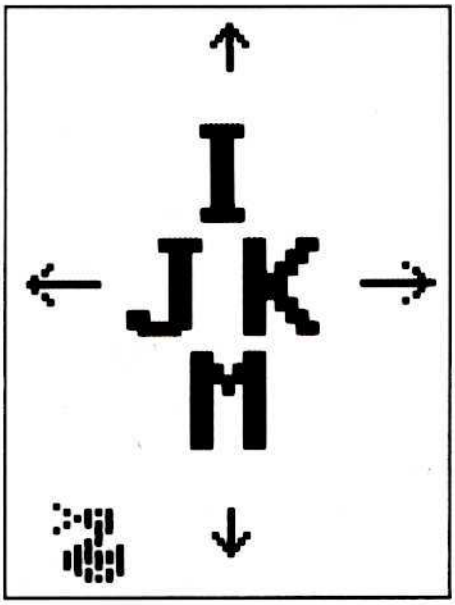
How-To's Of Hardcore

Welcome to Hardcore COMPUTIST, a publication devoted to the serious user of Apple][and Apple][compatible computers. We believe 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 back up 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 over the rest of this page carefully in order to avoid frustration when following the softkeys or typing in the programs printed in this issue. Longtime readers should know what to do next: Make a pot of coffee, get out some blank disks and settle in for a long evening at the keyboard.

Pg. 8



Pg. 22

Address all advertising inquiries to Hardcore COMPUTIST, Advertising Department, 3710 100th St. SW, Tacoma, WA 98499. 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 1984 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][or][Plus Computer, and is a trademark of Apple Computers, Inc.

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

DOMESTIC DEALER RATES: Call (206) 581-6038 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.

Issue No. 14

Hardcore COMPUTIST

Publisher/Editor: Charles R. Haight **Technical Editors:** Gary Peterson, Ray Darrah
Production & Graphics: Lynn Campos-Johnson **Circulation:** Michelle Frank
Business Manager: Valerie Robinson **Advertising:** (206) 581-6038 **Printing:** Grange Printing, Inc., Seattle, WA
Hardcore COMPUTIST is published monthly, except December, by SoftKey Publishing, 3710 100th St SW, Tacoma, WA 98499
Phone: (206) 581-6038

8 Softkey For Seadragon
Here is a simple method for the deprotection of this arcade-action submarine game. You'll find the remainder of this article devoted to some interesting and very helpful APTs. *By Jeff Rivett.*

10 Super IOB: Version 1.2 Update
The new version of Super IOB features hexadecimal track and sector display and easy sector editing at the controller level. Compatibility has been maintained with most of the controllers printed thus far in Hardcore COMPUTIST. *By Ray Darrah.*

15 Putting Locksmith 5.0 Fast Copy Into A Normal Binary File
At the completion of this technique, the reader will appreciate the resulting shortened waiting time to induce the 16-sector utility fast backup portion of Locksmith 5.0. The newly modified program can even be stored on hard disk. *By C.V. Fields.*

CORE SECTION

16 Batman Decoder Ring
With this CAD (Computer Aided Decoding) program, you can decipher cryptograms in a fraction of the time it takes to do it by hand. Batman Decoder Ring can also be used to encrypt messages, and the resulting work may be saved on disk. *By Ray Darrah.*

20 Boulder Dash by Microfun: Review
By Ray Darrah.

21 A Fix For DiskEdit
For all those poor souls who keyed in the misprint of the DiskEdit hexdump originally published in The Best of Hardcore Computing, here is the corrected version (with many apologies for our error).
* OK on disk

22 Tracking Down Rocky's Boots
In deprotecting Rocky's Boots, you will discover much useful information about the program and have the opportunity to use the first Super IOB controller with the "MOVE S PHASES" subroutine. *By Jerry Caldwell.*

DEPARTMENTS

4 INPUT

6 READERS' SOFTKEY & COPY EXCHANGE

Softkey For Knoware
By Doni G. Grande

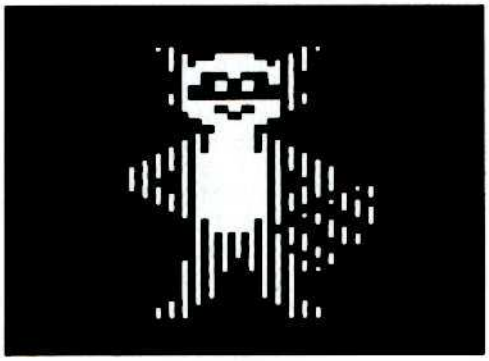
Deprotecting PFS Software
By Gary J. Wolfe

Deprotecting Computer Preparation: SAT
By Eddie Fang

Softkey For MatheMagic
By Doni G. Grande

9 BUGS

14 ADVENTURE TIPS



INPUT INPUT INPUT

How Compatible is Compatible?

For more than a year I have been the Editor for a religious news service, serving more than 15 member churches and providing ecumenical news for a community having better than 300 parishes.

For equipment, I have been using an Apple][+ computer equipped with a 16K card, Centronics parallel interface, Networker Modem, Applied Engineering 80-column Card, Axiom graphix printer interface, an unknown but possibly Microtek disk interface card, a USI Amber Monitor, an Apple Disk II, a Micro-Sci A2 disk drive, a Riteman printer, and a Seikosha GP-100A printer.

For programs, I have been using Magic Window, Magic Window II, Magic Mailer, and Magic Word. My modem program was Networker. I have several others that work with the Hayes modem, but do not work with the Networker.

As long as I was content to ask of my printer only upper and lower case letters I was able to produce, on schedule, a rather lengthy and involved series of publications. Subscriber response indicated that this was not sufficient and my reading of the manuals indicated that I should, with my programs and equipment, be able to produce some kind of graphics. I am now grievously behind schedule because the graphics do not work.

The plan to have the publication available on Source also does not work because the Networker was incapable of transmitting files to the source. But the main problems in publication are caused by Magic Window and the other programs for Artsci. The behind-publication-date issue I am now working on has the following problems:

A. Magic Window is so poorly designed that any keying in error can, at times, lead to major disasters, such as the time I thought I was loading my subscription file records and instead was copying a Newsletter out to the disk to overwrite the entire subscription records. This was a major disaster. Magic Window II corrects this problem by not using the arrow keys as the only means of select file functions.

B. However, Magic Window II did not correct some previous problems and brought on a few of its own. For example, spurious line feeds. In the use of the Title Page function, whenever a title is placed in the Top Margin, Magic Window and Magic Window II insert at least one extra line feed. On Magic Window, I simply lied to the format file and told the program that I had 65-line paper. This does not work with Magic Window II when I have expanded and emphasized print in the Title Line. In Magic Window II, despite 2 days and many experiments, I still am not certain how many extra line feeds I am getting or why.

C. Printer control characters: Magic Window II shifts them from whatever it is put in to something else based on the driver programs. My advertising literature straight from the salesman indicates that the Riteman is Epson-compatible so I selected the Epson driver. Now I simply do not know what the printer is doing or going to do with any code because the manual for Magic Window does not explain all the codes it uses. Magic Window II changes the codes, so I cannot trust my Printer Manual.

As the Editor I am responsible to see that the publication is readable and that I attract new readers. Unfortunately, my readers want outstanding copy that is easy to read. To expand my readership, I need to have good print output. The answer seems to require money spent on a good word processor and a better printer. The money I spent on Magic Window, the Seikosha GP100, the Riteman, and the Networker seems to have been a total loss and a total waste.

So, why am I writing this letter? To inform all my readers of the reason that their subscription is late, to inform all the computer clubs and magazines to which I belong or subscribe of these problems with this set of equipment and programs, and to solicit information as to any possible fixes or repairs to my current problem. Considering the rather fiendish delight that ARTSCI has in copy protection, I rather suspect that my cause is lost.

I hate to write entirely negative letters, so as a positive note I would like to say that my 3-year-old (at least) Apple][+ has never given me any problems at all, and that my Apple Disk II drive has only required speed adjustment in that time. WABASH disks work well and I really like my Riteman printer, if only other people could read the rather faint grey marks that the ribbon makes for copy.

Pierre Testart, Editor
BASOR
San Francisco CA

Mr. Testart: *We are rather surprised that you are having so many problems with Magic Window and Magic Window II because we use a close word processing relative of theirs, Acewriter, to edit and transmit to the typesetter, all text for each issue of Hardcore COMPUTIST.*

It sounds as if many of the problems you are having with the Magic Window programs could be traced to an improper printer driver. Although we have no experience with the Riteman printers, compatibility is not always everything it is cracked up to be. It would be well worth your while to call or write both Artsci and the Riteman corporations and describe to them in detail all of the problems you are having. In the meantime, you might try prayer.

Parms for PFS Graph

Re The Most Wanted List.

I have been successful in making a backup of Software Publishing's PFS using Locksmith 4.1 with the following parameters:

```
10 = 04, 16 = 40, 46 = 96, 51 = 00
53 = 08, 54 = 12, 81 = CF, 82 = F3
83 = FC
Copy tracks 0-22, Normal, Increment = 1
```

Bud Chambers
South Bend IN

I have been successful in making a backup copy of PFS Graph 2e (NEW) by Software Publishing Corp using Copy II Plus 4.4:

1. Drive speed is critical- set second drive to 200.5ns.
2. Copy track 0 in sync and nibble count mode with parm 3E = 2 (this will help in making a faster copy with fewer errors).
3. Copy track 1-22 in sync mode. Keep parm 3E = 2.
4. Write-protect the disk before booting it.

S.L. Grisham
Fayetteville AR

Double Hi-Res on the][+

I have been a subscriber for the past few issues and have found your magazine to be most helpful in backing-up my software. I think that it is the best thing to come along since the Apple. As to the question you asked about IBM and MAC softkeys in Hardcore, I say no. I don't care if you publish separate magazines covering them, but don't pollute our magzine. It's short enough already.

Regarding the softkey for Exodus: Ultima III, I was not able to back it up using the method outlined. The program kept running after the second time I typed in 8600G. Using my limited knowledge of machine language, I managed to track down a jump to $8848. So, for reasons that elude me at the moment, I changed the steps that say 8846:4C 59 FF to 8848:4C 59 FF. When I used this method I found that it was not necessary to use the third step. I then moved the RWTS down to $2400 and saved it as directed.

Now I have a few questions I would like to ask you. First, I have been bothered by your softkeys that require an old monitor ROM because I have an Apple][+. An Integer card is out of the question because it is out of my price range. I do not want to mess around with the chips inside my Apple, so a replacement ROM will do me no good. The only other alternative, according to your magazine, is an NMI card. I have been

looking at the Wildcard and would like to know if it can be used with the softkeys that require an old monitor ROM (and if so, how?). I have some of the programs that require an old monitor ROM to be cracked and would like to know if this is the answer.

Second, I have seen that you can get double hi-res graphics on the //e with the use of an extended 80-column card. I was wondering if there was any easy way to get double hi-res graphics on a II+. Beagle Brothers say that you need at least 128K RAM for it to work. I would think that somewhere out there is a company that makes an "Extended 80-column card" for the][+ because slot 3 of the][+ is essentially the same as the auxiliary slot of the //e, is it not? If there is a company out there, please let me know.

Third, I would like to know if there is any chance of getting Hardcore COMPUTIST issues number 2, 3, 5 and 7 reprinted. I would very much like to have these issues.

Fourth, when is The Best of Hardcore Vol. II coming out, if ever? I could use some new parameters and old softkeys from the original Hardcore Computing.

Thanks again for the best Apple][magazine around. I hope you will be around for a long time to come.

William Howard
Oak Ridge TN

Mr. Howard: Thanks for the tip on Exodus. It looks as if we have once again encountered a problem caused by multiple versions of the same program. In response to your questions: The Wildcard or any of the other NMI cards which are currently available can indeed be substituted for an old-monitor ROM. Just press the button (or flip the switch) on the card when the article instructs you to perform the RESET.

We have good news and bad news in our response to your question about double hi-res graphics on an Apple][+. The good news is that you can indeed obtain a horizontal resolution of 560 pixels (double the normal 280) on your][+. The bad news is that it can only be done in black and white.

We do not know of any of the double hi-res graphics packages designed for the //e that can be used on a][+. This is due to the way in which the double hi-res graphics were implemented on the //e. The memory in the extended 80-column card is mapped into the same address range as the memory on the motherboard. When displaying double hi-res, the Apple grabs one byte from the main hi-res memory and grabs the adjacent byte to be displayed from the same location in memory on the extended 80-column card. It would be very difficult to duplicate this scheme on a][+. The auxilary slot on the //e is not just like slot 3 on the older Apples.

For an informative discussion on how to obtain the double black and white hi-res graphics on your][+, check out Bob Bishop's article, "Apple-][Hi-Res Graphics: Resolving the Resolution Myth" in the Fall 1980 edition of Apple Orchard.

Although we have not advertised this fact, back issues of Hardcore COMPUTIST No.s 3 and 7 are available in very limited quantities on a first come, first served basis. Hardcore COMPUTIST No.s 2 and 5 are no longer available.

We are working on what is tentatively titled, "The Book of Softkeys". This publication will contain all the softkeys which have appeared up through Hardcore COMPUTIST No. 5. When the Book of Softkeys becomes available, it will be announced in Hardcore COMPUTIST.

Examining Exodus

I just received my first issue of Hardcore COMPUTIST which was Issue No. 11. I was interested in the Softkey for Exodus: Ultima III by Tim Schaap. After borrowing Super IOB from a friend, I went to work on Exodus. Everything was working well until Step 7 where the jump to the monitor did not execute properly for me. I looked at the BOOT1 code and discovered that the JMP instruction had been placed in the middle of another set of JMP instructions.

On my version of Exodus, a better choice for Step 7 is 8812:80, 8848:4C 59 FF. At this point it is interesting to note that the RWTS is loaded and set up. Therefore, I just skipped Steps 9, 10 and 11 and went right to Step 12. Everything went smoothly from there. Thanks to Tim Schaap for laying the groundwork for the deprotection of Exodus. Thanks also to Softkey Publishing for a magazine for the rest of us.

Nathan Manlove
Columbus IN

A Very Strange Program

I think Ray Darrah's Super IOB program is great. It has great versatility and is easy to use. However, to me, the accompanying article is not very clear. It does not explain how to use Super IOB to copy disks. It gives a couple of controllers, examples of copy-protection schemes and how the program works, but no explanation of how to use it. For example, I would appreciate a detailed explanation of how to find out any altered data marks using Golden Delicious Software's "CIA". The reason I want to know this information is that I am trying to backup Zork II: The Wizard of Frobozz. I found that Track 1 of this disk is readable but that the rest of the tracks have altered marks.

I have read your magazine from its first issue and really enjoy it. Here's an interesting little exercise to try out:

CALL -151

```
800:00 25 08 0A 00 BA 22 C8
:C1 D2 C4 C3 CF D2 C5 A0 C3
:CF CD D0 D5 D4 C9 D3 D4 A0
:C9 D3 A0 C7 D2 C5 C1 D4 A1
:22 00 00 00
```

3D0G
LIST

It should look something like this:

```
10 PRINT" + TO SGN THEN SPC(
    > SGN AT COLOR = SPC(>
    AND = USR ABS - INT ABS
    COLOR = - INT COLOR = STEP
    SGN AT TO ABS POP "
```

To see what happens just type

RUN

Chris Windle
Morgan Hill CA

Chris: You seem to have picked up some very unorthodox programming practices somewhere (or from someone). Do you write all your Applesoft programs in this manner?

See the listing of a Super IOB v1.2 (UPDATE pg. 10 in this issue) controller below that will work for Zork II and other Infocom Adventures.

```
1000 REM INFOCOM CONTROLLER
1010 TK = 0 : ST = 0 : LT = 35 : CD = WR :
    ONERR GOTO 1110
1020 T1 = TK : GOSUB 490 : IF TK THEN
    RESTORE : GOSUB 210
1030 GOSUB 430 : GOSUB 100 : ST = ST + 1
    : IF ST < DOS THEN 1030
1035 IF TK = 0 THEN GOSUB 210
1040 IF BF THEN 1060
1050 ST = 0 : TK = TK + 1 : IF TK < LT
    THEN 1030
1060 GOSUB 310 : 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
1110 CALL 822 : ERR = PEEK (222 )
1120 IF ERR > 254 OR CD <> RD THEN
    10230
1130 IF ERR > 15 THEN POKE 216 ,0 :
    RESUME
1140 IF TK > 17 THEN LT = TK : GOTO
    1060
1150 IF TK <> 2 THEN 10230
1160 GOTO 580
5000 DATA 213 ,170 ,188
5010 DATA 2^CHANGES
5020 DATA 0 ,2 ,252 ,173
5030 DATA 0 ,2 ,93 ,173
```

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.

READERS' SOFTKEY & COPY EXCHANGE

* See No. 15, Page 5 - alternate version method.

Softkey for Knoware

By Doni G. Grande

Knoware
Knoware, Inc.
301 Vassar St.
Cambridge, MA 02139
$95.00

Requirements:
A way to Reset into the monitor
COPYA
A sector editor
Three blank disks

Knoware is a very interesting game - simulation - educational program. The player starts out in a company working in the mailroom. The objective is to become chairman of the board. In order to accomplish this, you must use computer application programs for spreadsheet analysis, wordprocessing, and database use. All of the application programs may be used on their own, so once you make chairman of the board, you have learned how to use all of the programs!

First of all, only the first (out of three) disk is protected. Secondly, Knoware checks the disk in the drive to be sure it is an original. However, it is fairly easy to short-circuit this check. The procedure to make a copyable version of Knoware is as follows:

1) COPYA the original Disk #1.
2) Use a sector editor to make the following changes:

Track	Sector	Byte	From	To
$17	$01	$14	$4C	$A9
$17	$01	$15	$8D	$00
$17	$01	$16	$20	$EA

These changes are to the file ONESHOT.OVR (A$2000,L$FBA), and prevent a call to a protection check subroutine by replacing the JSR $208D with a LDA #00 and a NOP, which tricks the program into thinking there is an original disk in the drive.
3) Now COPYA Disks 2 and 3, and you have the complete set.

One modification that you might want to make is to allow a fast DOS to be used. There is a check in the file ONESHOT.OVR to see if the I/O hooks have been changed. Some fast dos versions do change these. The following change will defeat this check:

Track	Sector	Byte	From	To
$10	$0F	$01	$D0	$EA
$10	$0F	$02	$0C	$EA
$10	$0F	$05	$C9	$A9
$10	$0F	$06	$9E	$00

Deprotecting PFS Software

By Gary J. Wolfe

PFS:File
PFS:File //e
PFS:Report
PFS:Report //e
PFS:Graph
PFS:Graph //e

Software Publishing Corp.
1901 Landings Drive
Mountain View, CA 94043
$125.00 each

Requirements:
Apple][+ or equivalent
One disk drive
COPYA
Sector Editor with Search Capability (Zap, Inspector, etc.)

Although most of PFS's software can be backed-up by using Copy][+'s normal copy (not bit copy) utility, I prefer to completely remove the copy protection from commercial software that I own. Even though the PFS series of programs are written in Pascal, the disk protection code is written in assembly language and can be fairly easily circumvented.

To remove the protection from any of the PFS programs listed above, you will need some sort of disk search utility such as ZAP or Inspector. The entire disk has to be searched for a byte sequence of D0 04 88 98 F0 27. This code is found in a routine which checks the disk for the presence of extra bits in the sync fields, a protection scheme cleverly called the "bit insertion technique". If the extra bits are present, an $FF will be pushed on the stack, otherwise, a $00 will be pushed onto the stack before the routine returns to its caller. By changing the second byte of the search sequence from an $04 to a $29, the protection code can modified so that it will always push an $FF onto the stack whether the extra bits are present or not. The step-by-step instructions for removing the PFS protection scheme are listed below.

1) First, run COPYA to make a copy of the PFS program

RUN COPYA

2) Get out your disk search utility and search the copy for the following sequence of bytes

D0 04 88 98 F0 27

3) Change the $04 in this sequence to a $29 and write the sector back to the disk.
4) Search the whole disk for this pattern and make the same change in every location that it is found.
5) Write protect the disk before trying to boot it.

Deprotecting Computer Preparation: SAT

By Eddie Fang

Computer Preparation: SAT
Harcourt, Brace and Jovanovich, Inc.
1250 Sixth Avenue
San Diego, CA 92101
$79.95

Requirements:
48K Apple][, Apple][Plus, or //e
One disk drive with DOS 3.3
Four blank disks
The SAT package
Super IOB and the swap controller
(Optional: File transfer program)

This computer preparation package by HBJ ranks up at the top with Barron's computer preparation course for the SAT. HBJ's package has a lot of the major words found on the SAT and a lot of similar mathematical and verbal problems, too.

My problem, however, is that I would like to change the words and problems for a younger brother or change the problems that I have finished studying to some newer ones. The frustration of copying this disk is immense. After trying all of the major bit copiers to no avail, I decided to try Super IOB. Here is how I did it!

First, get the disk that is labeled "A". All the disks follow the same process, except you will have to modify different programs on different disks to get each totally broken. Follow these steps carefully:

1) Boot the first protected disk. When you hear the drive head access the first track, hit reset.
2) Get into the monitor

CALL -151

3) We must move the RWTS down to a safe location for Super IOB to use

1900<B800.BFFFM

4) When you get the monitor prompt again, insert your Super IOB slave disk and boot it

C600G

5) Save the RWTS

BSAVE RWTS.SAT,A$1900,L$800

6) Install the controller at the end of this article (a modified version of the swap controller) into Super IOB and execute Super IOB

RUN

7) When asked if you wish to format the backup first, reply with a "Y." This will put DOS on the disk and set the bootup program to HELLO.
8) Copy the other three sides in the same manner (Steps six and seven).

Almost Finished

You now have a COPYAable version, but if you want to modify it, you will have to

change a few line numbers of some of the BASIC programs. Several of the main programs have a line number 0 which looks something like this:

```
0 REM [CTRL]H[CTRL]H[CTRL]H[CTRL]H[CTRL]H[CTRL]H[CTRL]H[CTRL]H[CTRL]M[CTRL]D FP
```

Whenever you try to list a line like this, DOS sees the [CTRL]M[CTRL]DFP and clears the program in memory. In other words, the program self-destructs when LISTed. This situation is easily circumvented by merely eliminating line zero from these programs. The general format for doing this is:

```
LOAD filename
DEL 0,1
SAVE filename
```

Remove line zero from these programs:

1) "Math Item Bank Display" on the mathematical side of your duplicate disk
2) "Vocabulary Flashcards" on the vocabulary flash cards side of your duplicate disk
3) "Verbal Item Bank Display" on the verbal side of your duplicate disk
4) "Frank" (I don't know why they called it that) on the diagnostic and testing side of your duplicate disk

Closing Comments

You should now have four bootable, completely deprotected and modifyable Computer Preparation: SAT disks. I hope this helps you as much as it has helped me.

SAT Controller

```
410 GOSUB 80 : HOME : A$ =
    "FORMATTING" : FLASH : GOSUB 450
    : NORMAL : PRINT : PRINT CHR$ (4)
    "INITHELLO,S"S2",D"D2",V"VL
    :VL = 0 : RETURN
1000 REM SWAP CONTROLLER
1010 TK = 3 :ST = 0 :LT = 35 :CD = WR
1020 T1 = TK : GOSUB 490 : GOSUB 360 :
    ONERR GOTO 550
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 490 :TK = T1 :ST = 0 : GOSUB
    360
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 "COPY^DONE" : END
10010 IF PEEK (6400) <> 162 THEN
    PRINT CHR$ (4) "BLOAD^
    RWTS.SAT,A$1900"
```

Controller Checksums

```
410   - $961A     1060  - $E21F
1000  - $D784     1070  - $EA1A
1010  - $EA19     1080  - $B3DE
1020  - $5DE7     1090  - $AF89
1030  - $4BE6     1100  - $E66A
1040  - $A19D     10010 - $BD80
1050  - $1F88     0     - $0000
```

Softkey for MatheMagic

By Doni G. Grande

MatheMagic
International Software Marketing
Suite 421, University Building
120 E. Washington Street
Syracuse, New York 13202

Requirements:
Apple with 48K
COPYA
Super IOB (HC No.9)
A blank disk

MatheMagic is a program which transforms your microcomputer into the ultimate calculator. With it, you can program your computer in the same way you would a programmable calculator.

The program disk itself is copy-protected, but it can use and initialize normal data disks. This is done by altering RWTS when access is needed to the original disk, and then changing it back to normal when a data disk is accessed.

The only parts of RWTS I found changed were the following locations:

Location	From	To
$B991	$DE	$DF
$BCAE	$DE	$DF
$BC60	$FF	$FE

The first two locations are the address epilog start bytes and the second location is the sync mark used in the protected DOS. The only problem with just changing these locations to their normal values is that the program keeps changing them back whenever it accesses the program disk, generating an I/O error. Also, since the program is a compiled basic program, it is very difficult to discover the location in the program at which this change is made.

The answer to the problem is to let the program change DOS to access the program disk, but patch the RWTS subroutine to change those locations back to normal whenever it is called. This works quite nicely.

The main entry to RWTS is at $BD00, and a JuMP to the patch can be made there if the original instructions at this entry point are duplicated in the patch. After DOS is put back to normal, the patch jumps back to RWTS and all operates normally!

The steps required to make a normal copy of MatheMagic are:

1) Boot a normal DOS disk.
2) Enter the monitor, move an image of the normal DOS 3.3 RWTS to $1900 and then modify it so it can be used by Super IOB to read the protected MatheMagic disk.

```
CALL -151
1900<B800.BFFFM
1A91: DF
1DAE: DF
BSAVE RWTS, A$1900, L$800
```

3) Run Super IOB with the Swap Controller (pg. 24) installed to make a copy of MatheMagic. The Swap Controller will use the RWTS created in Step 2 to read the disk.
4) Type in the following short patch

```
0360:84 48 85 49 A9 DE 8D 91
0368:B9 8D AE BC A9 FF 8D 60
0370:BC 4C 04 BD
```

This is the patch which will be executed just before RWTS is entered. The monitor listing is:

```
* Store Y and A as RWTS would
0360-   84 48       STY   $48
0362-   85 49       STA   $49
* Fix data and address marks
0364-   A9 DE       LDA   #$DE
0366-   8D 91 B9    STA   $B991
0369-   8D AE BC    STA   $BCAE
* Might as well fix the sync byte
  too
036C-   A9 FF       LDA   #$FF
036E-   8D 60 BC    STA   $BC60
* Return control to the RWTS
0371-   4C 04 BD    JMP   $BD04
```

5) Save this patch out to the copy of the MatheMagic disk

```
BSAVE DOSPATCH, A$360, L$14
```

6) Load the file MATH.HELLO from the copy disk and add the following

```
227 PRINT CHR$(4) "BLOAD^DOSPATCH"
228 POKE 48384,76 : POKE 48385,96 :
    POKE 48386,3 : POKE 48387,234
```

This places a JMP to $360 and a NOP as the first instructions in RWTS which causes DOSPATCH to be executed before every call to RWTS.
7) Save MATH.HELLO back to the copy disk

```
UNLOCK MATH.HELLO
SAVE MATH.HELLO
```

The disk is now deprotected. You will find that program operation is enhanced by the use of a fast DOS such as Diversi-DOS.

IMPORTANT: Dear Subscriber,

The staff here at SoftKey Publishing will be taking a Hardcore holiday in the month of December. As a result, you will receive the final 1984 issue in November. The next issue you receive will be January 1985. Your subscription will not be affected, and you will still receive a total of twelve information packed issues of Hardcore COMPUTIST.

Many thanks for your continued support and we hope you enjoy your holidays.

Softkey For SEADRAGON

By Jeff Rivett

Seadragon
Adventure International
Box 3435
Longwood, FL 32750
$34.95

Requirements:
Seadragon original
Super IOB v1.2

When I first encountered Seadragon, like most people, I was fairly impressed by its animation and sound routines. However, I didn't play it too much because I was not able to back it up, and I don't like using originals.

The program boots very much like a normal disk. You can get a good copy up to the title page with any bit copier, but when you try to run the game, it will load and then start to do a strange thing. It sounds as if the disk drive is having a spasm, continually moving from track Ø to track $22. This goes on until you hit reset or turn the machine off. But the designers have nicely set the reset vector so that you cannot stop the reboot by repeatedly hitting reset.

The program is actually doing a funny kind of nibble-count. If you look at track $22 with a nibble editor, you will see a repeating pattern of FF's and DD's. This pattern is what the copy-protection looks for, and it is fairly hard to copy. I found that Copy][plus will do it on default parameters if you fiddle with your drive speed.

Once I was able to copy the disk, I started to really get into the game. There, I ran into another old enemy, the Arcade Mentality: the game is just too difficult. I felt I had been let down, and started thinking about how I could get some satisfaction from the game. That's when I thought about modifying it. But the modifications are easier discovered and executed once the copy protection has been eliminated.

The Softkey

The protection on this disk comes in stages. First, the END OF DATA marks have been changed from DE AA to AA DE. The DOS is very much like normal DOS, and to allow it to read the different marks, the read error routine has been turned off. Pretty clumsy, right? The only sector on the disk without these different marks is the one used to store high scores, track 3, sector Ø. In any case, I dealt with this problem by writing a Super IOB controller which fits these requirements.

The disk's DOS does not need to be changed to read the new, unprotected format, but I wanted to normalize the disk as much as possible, so I made the Super IOB controller edit track Ø, sector 3 as shown in the list below.

The next problem was the nibble count. Since the disk was now easily read with a sector editor, I reasoned that it should be possible to locate the nibble count routine and turn it off. I found this routine at track $19, sector $ØE.

But the Protectors hadn't finished yet. After I turned off the nibble-count, I booted the copy. The game started to run and then some very funny things happened. The mines seemed to be all over the place. And when I got far enough into the cave, the game simply hung. There is an area that is used for game data at $95ØØ which seems to get scrambled when you turn the nibble count off. But the routine that does this can be found at track $19, sector $Ø8.

Now the game seemed to run perfectly, and continued to do so just as long as I never used torpedoes. Well that certainly wasn't much fun. Another routine had been activated that was forcing an early link to the next level, that is, the end of the cave where you meet the Seadragon. When I pushed the torpedo button, this part of the game loaded and I was actually able to play it. You might want to try this just for fun if you've never seen the Seadragon. The routine that messes up this part of the game is at track $15, sector $ØE.

After I located and turned off these routines, I found that the game ran normally with the exception that it took slightly longer to load. This was simply due to the fact that I had turned the read error routine on.

So to make a working, COPYAable copy, load Super IOB and install the controller listed at the end of this article. Next, run it on Sea Dragon.

When the controller has finished, it has made these sector changes:

Track	Sector	Byte	From	To
$ØØ	$Ø3	$36	$EA	$DØ
		$37	$EA	$ØA
		$3F	$ØØ	$AA
		$4Ø	$DØ	$FØ
$19	$ØE	$ØØ	$4C	$6Ø
$19	$Ø8	$25	$A2	$6Ø
$15	$ØE	$E7	$A9	$6Ø

The Advanced Playing Techniques

There are lots of things out to get you in this game. Using the list I have compiled, you can now customize the game so that the more insidious ones just disappear. There are also a few irritating noises, which may now be turned on and off at will. I suggest, however, that you make a backup of your new copy before you get too carried away.

Here are the noises you may get rid of with the following sector edits:

1) Voice at start of game.
2) Sub explosion sound.
3) Irritating noise.

No.	Track	Sector	Byte	From	To
1)	$Ø4	$Ø2	$B8	$AD	$6Ø
2)	$16	$ØD	$DF	$AD	$EA
			$EØ	$3Ø	$EA
			$E1	$CØ	$EA
3)	$17	$Ø7	$8F	$AD	$EA
			$9Ø	$3Ø	$EA
			$91	$CØ	$EA
			$BD	$AD	$EA
			$BE	$3Ø	$EA
			$BF	$CØ	$EA

Some other APTs you might be interested in are:
1) Mine speed ($ØØ Turns them off).
2) Eel speed ($FF Turns them off).
3) Seaweed speed ($FF Turns them off).
4) Seaflea speed ($ØØ Stops them).
5) Stalactite speed ($ØØ Stops them).

These are accomplished by the following

Sector modifications:

No.	Track	Sector	Byte	From	To
1)	$14	$0C	$DE	$05	$00-$09
2)	$14	$08	$0B	$04	$00-$FF
3)	$14	$09	$59	$01	$00-$FF
4)	$14	$06	$15	$02	$00-$02
5)	$14	$07	$47	$04	$00-$09

Some miscellaneous APTs I've discovered include:

1) Turn off force fields.
2) Turn off shooters.
3) Unlimited damage.
4) Unlimited air.
5) Free sonic disruptor.

No.	Track	Sector	Byte	From	To
1)	$14	$0A	$33	$46	$00
	$15	$07	$62	$05	$00
2)	$14	$0A	$E3	$01	$FF
3)	$18	$0B	$6E	$8D	$EA
			$6F	$2E	$EA
			$70	$43	$EA
4)	$18	$0B	$32	$01	$00
5)	$18	$0B	$59	$05	$00

Other Stuff

The main portion of the game is on tracks $12 to $19. After the sub blows up, the game restarts at $8868. The actual amount of air you have is stored at $432B-$432C. Damage is at $432E. There are many other interesting locations, including $4DB0 and $57F3.

How I Did It

Most of the detective work I had to do to de-protect this disk was done with the aid of my REPLAY II card and Nibbles Away II. Replay allowed me to enter the monitor at any time and look at the code. When I found a suspicious looking routine, I searched the disk using the disk scan utility in the sector editor part of Nibbles Away II. After locating the routine, I was able to disassemble it from within the sector editor, make a calculated change, and boot the copy to see what, if anything, had changed.

Closing Remarks

I found that being able to control the game really increased my enjoyment of it. For example, I was having a lot of trouble figuring out how to get past the Seafleas, so I turned everything else off so that I could concentrate on them alone. You may want to turn everything off just to try navigating the cave from beginning to end. Have fun.

Sea Dragon Controller

```
1000 REM SEA DRAGON CONTROLLER
1010 TK = 0 : ST = 0 : LT = 34 : CD = WR
1020 T1 = TK : GOSUB 490 : RESTORE :
     GOSUB 170
1030 GOSUB 430 : IF ST = 1 AND TK = 3
     THEN RESTORE : GOSUB 170
1035 GOSUB 100 : ST = ST + 1 : IF ST <
     DOS THEN 1030
1040 IF BF THEN 1060
1050 ST = 0 : TK = TK + 1 : IF TK = 3 THEN
     GOSUB 230
1055 IF TK < LT THEN 1030
1060 GOSUB 230 : GOSUB 310 : 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 222 ,170 ,170 ,222
5010 DATA 7^CHANGES
5020 DATA 25 ,14 ,0 ,96 ,25 ,8 ,37 ,96
5030 DATA 21 ,14 ,231 ,96
5040 DATA 0 ,3 ,54 ,208 ,0 ,3 ,55 ,10
5050 DATA 0 ,3 ,63 ,170 ,0 ,3 ,64 ,240
```

Controller Checksums

1000	- $356B	1080	- $3136
1010	- $6344	1090	- $E343
1020	- $5D5B	1100	- $549F
1030	- $3E9C	5000	- $2EF3
1035	- $BEC7	5010	- $937A
1040	- $B2A6	5020	- $8BA1
1050	- $8A78	5030	- $268E
1055	- $B8FB	5040	- $912E
1060	- $0E9D	5050	- $5B35
1070	- $0698		

Bugs In Hardcore COMPUTIST No. 11, 12 & 13

HC No. 11

Ultimaker III program, p. 18
Line 1500 should read as follows:

```
1500 PRINT B$CH$CJ$; :B = B + INT(2 ^
     X ) * (B$ = "Y" ) : NEXT :RETURN
```

Ultimapper.small, p. 20
Line 650 should read:

```
650 DATA ^^^UP^LADDER ,V^^DOWN^LADDER
    ,I^^UP/DOWN^LADDER ,$^^GOLD ,#^^
    WALL ,=^^HIDDEN^DOOR ,O^^VISIBLE^
    DOOR
```

HC No. 12

The CORE Disk Searcher program, p. 21
Line 940 should read:

```
940  INPUT""  ;P$(NS ):  IF  P$(NS )
     =""THEN 850
```

Line 1250 should read:

```
1250 R = R + INT (16 ^ B * C ) : NEXT B
     :RETURN
```

Listing 1 Pseudo ROMS Hello, p. 26
Line 70 should read:

```
70 FOR AD = 769 TO 810 : READ BY : POKE
   AD ,BY : NEXT
```

Line 270 should read:

```
270 DATA^20, 74, 141, 0, 208, 205, 0,
    208, 208, 11, 169, 1, 141, 0, 3,
    173, 129, 192, 173, 129, 192, 96
```

HC No. 13

Locksmith 5.0 Patch, p. 5
The second ($18) and third ($18) values in the Byte column should be $19 and $1A, respectively.

Most Wanted List

If you have been trying to backup a program, and have only ended up pulling your hair out as a result of the ordeal, let us know about it.

We have received softkeys for a number of programs previously in our list and these will be published as soon as each has been evaluated and edited by our staff.

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

If you know how to de-protect, unlock or modify any of the programs below, we encourage you to help other Hardcore COMPUTIST readers and earn some extra 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
2. **Flight Simulator II**
   Sub Logic
3. **DB Master 4.0**
   Stoneware, Inc.
4. **DB Master 4.0 +**
   Stoneware, Inc.
5. **Bookends**
   Sensible Software
6. **Visiblend**
   Micro Lab
7. **Dollars And Sense**
   Monogram
8. **Word Juggler**
   Quark, Inc.
9. **Catalyst**
   Quark, Inc.
10. **Gutenburg Jr. & Sr.**
    Micromation LTD
11. **Prime Plotter**
    Primesoft Corp.
12. **The Statistics Series**
    Human Systems Dynamics
13. **Sargon III**
    Hayden
14. **Zardax**
    Computer Solutions
15. **List Handler**
    Silicon Valley Systems

DEAR AUTHOR:

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

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

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

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

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

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

page 11

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

For a copy of our WRITER'S GUIDE send a business-sized (20-cent) SASE (self-addressed, stamped envelope) to:

Hardcore COMPUTIST
WRITER'S GUIDE
PO BOX 110846-K
TACOMA WA 98411

Update

Super IOB: Version 1.2

- See No. 17, page 9 - tutorial for use of Super IOB
- See No. 16, page 15 - The Controller Writer
* See No. 22, page 17 - Super IOB 1.5 (new version)

By Ray Darrah

"Through the course of time, Hardcore Computing (old series) and Hardcore COMPUTIST have published several IOB programs...Super IOB was an advanced version of the original IOB program. Presented here is the first in a series of updates to the Super IOB program."

Requirements:
An Apple][with 48K and Applesoft
Disks that need to be modified

As Hardcore COMPUTIST readers may recall, the IOB program is a simple BASIC program that performs softkeys. IOB stands for Input/Output control-Block which is a list of parameters used by the Read/Write Track/Sector (RWTS) subroutine. The original program was named IOB because it controlled the IOB in DOS to copy disks.

Through the course of time, Hardcore Computing (old series) and Hardcore COMPUTIST have published several IOB programs (or modifications). These were useful not only for copying different types of disks, but for configuring the program to different machines (ex. one drive or two?).

page 15

As it appeared in Hardcore COMPUTIST No. 9 and the Best of Hardcore Computing, Super IOB was an advanced version of the original IOB program. Included were the most useful subroutines from all the previous IOB programs. In addition, new and better features were added.

Presented here is the first in a series of updates to the Super IOB program. Some of the features included in this update are:

1) Two subtle bugs have been corrected.
2) Altered ending marks (only altered starting marks were supported before) are now easily accessed.
3) A subroutine for ignoring the address header checksum has been provided.
4) A subroutine for changing the data field checksum while reading is included.
5) A routine for automatically (and easily) performing sector edits has been provided.
6) The "PRINT TRACK & SECTOR#" routine now displays in hexadecimal.
7) Compatibility with old Super IOB controllers has been maintained.

Updating Super IOB

All of the modifications in this update are to the BASIC program only. Therefore, to update your Super IOB program, you need only to type the lines that have the corresponding checksum shaded.

Keying In Super IOB For The First Time

If you don't already have Super IOB then you need to know that before it can be fully used, a few files must be made. Begin by typing the Applesoft listing at the end of this article and

```
SAVE SUPER IOB
```

Next, enter the hexdump at the end of this article and

```
BSAVE IOB.OBJ0,A$300,L$5C
```

A third file is required in order to copy disks that have been protected with a 13-sector base format.

RWTS.13

To read the protected DOS 3.2 disks, Super IOB uses an image of the 3.2 RWTS. By performing a swap of the image (the 3.2 RWTS is moved to the location of the 3.3 RWTS and vice versa, simultaneously) with the RWTS currently in memory, diskettes with these formats can be broken.

To create RWTS.13, use a program on the system master disk called BOOT13. This will allow the booting of a DOS 3.2 disk.

Once DOS 3.2 is up, enter the monitor and move the RWTS to a safe place.

```
CALL -151
1900<B800.BFFFM
```

Next, boot a DOS 3.3 disk (with no greeting program) and save the RWTS.

```
BSAVE RWTS.13,A$1900,L$800
```

No DOS 3.2?

If you do not have an old DOS 3.2 disk to boot, then you can use the method starting on page 6 of Hardcore COMPUTIST No. 13. You should now have a completely functional Super IOB program. Enjoy!

Familiar With Super IOB?

If you are already familiar with Super IOB, then you may skip the following text and begin reading at the section entitled "The Deletions."

The Function Of Super IOB

Super IOB de-protects disks by pushing the RWTS subroutine in DOS to its uppermost limits. Because of this, it only copies disks with sectors that somewhat resemble normal ones.

Before a disk can be softkeyed, the sector alterations must be determined. The easiest way to do this is to use a program [like Bag of Tricks, DiskView or The Linguist (from CIA)] which shows these differences on a protected disk.

Once the protection has been discovered, all that needs to be done is to insert a controller program (lines 1000 through 9999) into Super IOB. Here is a list of the protection schemes Super IOB was designed to softkey:

1) Altered address, data, prologue, or epilogue marks
2) Strangely numbered sectors or tracks
3) Modified RWTS (with same entry conditions)
4) Half tracks for any of the above
5) 13 or 16-sector base format for any of the above

A Little Briefing

The following is a short discussion of the protection scheme and how each relates to Super IOB. Keep in mind that often more than one scheme is used at a time. This has the effect of complicating the Super IOB controller.

Altered Marks

DOS looks for specific marks when trying to read a sector. Changing these is a common practice, especially on older releases. As previously explained in the Whiz Kid column, DOS puts certain reserved bytes on the disk (during INITialization) so it can tell where a sector (and other valuable information) begins and ends.

For example, a normal 16-sector disk has the bytes D5 AA AD designating the start of the data field which contains the 256 bytes of a sector in encoded form. When a standard RWTS tries to find a sector, it looks for these marks. If they are not found, (either because they don't exist or they have been changed to something else) DOS returns with the dreaded I/O ERROR.

The sequences of the four reserved-byte marks (start of address, end of address, start of data, end of data) are handled by subroutines in Super IOB. These subroutines change the marks that the current RWTS looks for when reading.

Strangely Numbered Sectors

Within an address field, there are 8 bytes which tell the RWTS what sector is about to pass under the read/write head. On some disks, these are not standard. These disks are easily softkeyed with Super IOB. The controller instructs Super IOB to read the sectors using the strange sector numbers and then write them using the correct ones. This works because the RWTS merely compares the sector number found on the disk with the one the controller is looking for (even if it is higher than 15 and; therefore, illegal).

Modified RWTS

The disk-protectors will often rearrange and/or modify the standard RWTS subroutine. If this is the case, all one has to do is save this strange RWTS onto a normal DOS disk and use a controller which reads the protected disk using the strange RWTS and then writes via the normal 3.3 RWTS.

Such a controller is included in this article. It is called the SWAP controller. This is because of its use of the "Swap RWTS at $1900 with the one at $B800" routine in Super IOB.

Since the RWTS of a protected disk will be modified to read any altered DOS marks, this is an easy method to use if you are unable to determine the protection scheme.

Half Tracks

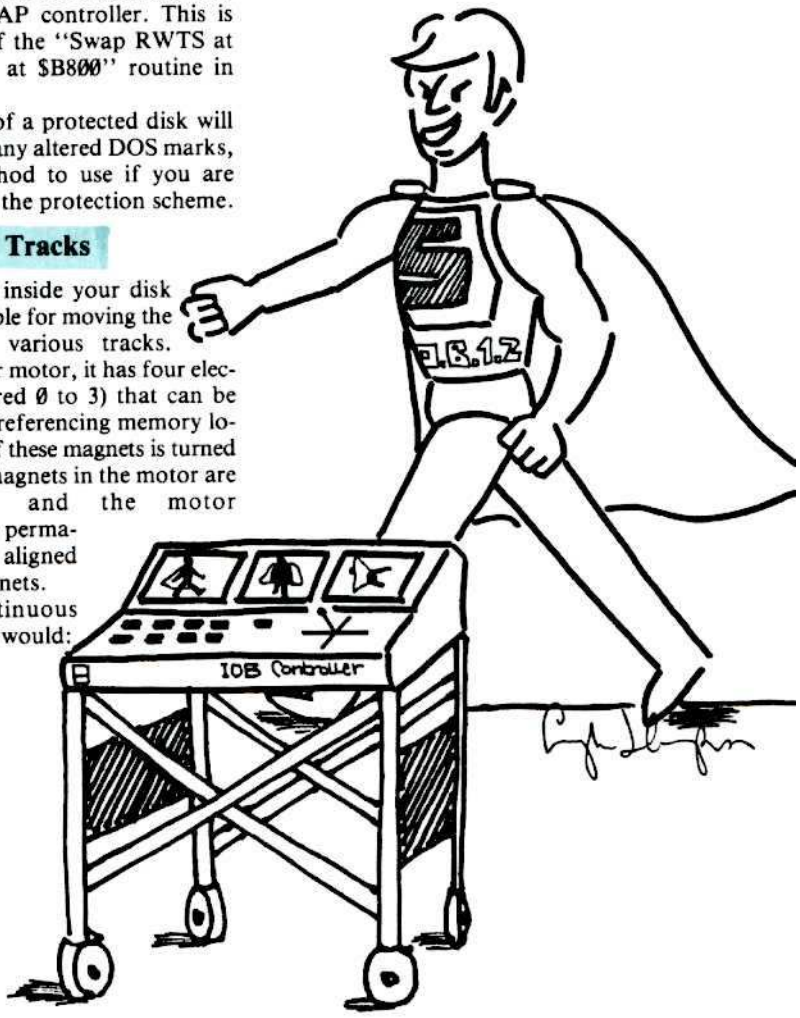
There is a motor inside your disk drive that is responsible for moving the disk head to the various tracks. Known as the stepper motor, it has four electromagnets (numbered 0 to 3) that can be turned on or off by referencing memory locations. When one of these magnets is turned on, the permanent magnets in the motor are attracted to it and the motor shaft turns until the permanent magnets are aligned with the electromagnets.

To obtain continuous motion, a program would:

1) Turn a magnet (called phase) on
2) Wait for the motor to get aligned (it doesn't take much time)
3) Turn off the magnet
4) Turn on the next adjacent magnet (the next magnet differs depending on whether you want to go to a higher track or a lower one)
5) Go to step 2

Because of the resolution of the disk head combined with the accuracy of the stepper motor, normal DOS tracks are placed only on the even phases. This means that for every track DOS moves, it references two magnets. As a result, the disk head never stops at any of the odd phases (i.e. aligned with magnets 1 or 3). Therefore, the odd phases are commonly called half tracks.

The disk-protectors will often put information on these phases that are inacessable to normal DOS. A routine called "MOVE S PHASES" (in Super IOB) handles the job of getting to these so called half tracks and can also be used (by a controller) to get to tracks that have been marked as other tracks. A complete discussion of how to use this routine appears later in this article.



Anatomy Of A Controller

Before we attempt to write a controller, let's look at the subroutines at the controllers disposal. During this explanation, it would be wise to refer to the listing of Super IOB to see how each is accomplished.

The Deletions

In this update, some subroutines have been removed from Super IOB Version 1 in order to fit the new ones in. They are:

1) All the 13-sector RWTS routines
2) The 16-sector "IGNORE CHECKSUMS AND & MARKS" routine

These routines were chosen because they are used infrequently. In fact, Hardcore COMPUTIST has never printed a controller that uses an "IGNORE CHECKSUMS & END MARKS" routine.

The New Routines

Here is a description of the new routines and how they work.

Name: **Altered Ending Marks**
Entry Conditions: Proper DATA pointers
Line Number(s): 17Ø

This routine changes the address field and data field epilogue markers in the normal RWTS. The values to change these to should be contained in a DATA statement. Because normal DOS only checks the first two bytes of these markers, only four values are required. The address field is changed first and; therefore, should appear first in the data statement.

Name: **Ignore Address Checksum**
Entry Conditions: none
Line Number(s): 27Ø

This routine modifies the RWTS subroutine so that it doesn't examine the checksum byte of the address field. This routine was incorperated in many controllers.

Name: **Altered Data Checksum**
Entry Conditions: Proper DATA pointers
Line Number(s): 29Ø

This routine alters the starting checksum byte that the RWTS subroutine will use when reading a DATA field. The normal value for the RWTS is Ø. The value to change the checksum to should be the next DATA element.

Name: **The Sector Editor**
Entry Conditions: Proper DATA pointers and Elements, T1 = lowest track in buffer, TK = highest track in buffer
Line Number(s): 31Ø - 34Ø

This routine automatically performs sector edits as the copy process goes on. It must be called (via GOSUB) just after reading a range of tracks. To indicate how many sector edits are to be performed, you must have a DATA element that has the number of sector edits followed by the word "CHANGES." For example:

```
11ØØ DATA 7 CHANGES,1,1,3,4
```

The above example would tell the sector editor that the next 28 DATA elements are sector edits. This is because each sector edit is defined in four DATA elements. The location of the "x CHANGES" element in the DATA string does not matter. The sector editor will search it out and use the elements immediately following it.

The format for the four bytes that define a sector edit is: TRACK, SECTOR, BYTE, CHANGE TO. Each element is decimal and should be within the correct ranges since no error checking is done.

The Older Routines

The following is an explanation of the remainder of the routines found in Super IOB Version 1.2. These routines were in the original Super IOB program.

Name: **Start Up**
Entry Conditions: Not Applicable
Line Number(s): 1Ø - 6Ø

The first few lines merely identify the program. However,line sixty sets HIMEM and LOMEM so that they fit the memory usage requirements (see memory map following). It then goes to "CONFIGURATION TIME".

Name: **Initial IOB Setup**
Entry Conditions: DV = drive to be accessed, VL = volume of disk to be accessed, SO = slot to be accessed
Line Number(s): 8Ø

This subroutine is normally GOSUBed via "TOGGLE READ/WRITE". Its purpose is to reset the buffer page and set the drive, slot and volume number to the disk to be accessed next.

Name: **R/W Sector**
Entry Conditions: TK = Track to be accessed, ST = Sector to be accessed, CD = Command code for the RWTS
Line Number(s): 1ØØ - 11Ø

This subroutine is GOSUBed directly from the controller. It reads or writes (depending upon CD) at the specified track and sector.

Name: **Move S Phases**
Entry Conditions: SO = Slot of drive to move, DV = Drive number of drive to move, PH = Phase number that the disk head is currently over, S = Number of phases to move
Line Number(s): 13Ø - 14Ø

This routine moves the disk read head the number of phases specified by S. One phase equals one half track. It is capable of moving in either direction up to 128 phases (or 64 tracks). Care should be taken that PH + S isn't greater than 255 or less than Ø or an error will occur.

Name: **Altered Address Marks**
Entry Conditions: Proper DATA pointers
Line Number(s): 19Ø

This routine modifies the RWTS (via POKE) so that it looks for a different sequence of address field prologue marks. The decimal values of the marks to look for should be stored as the next DATA elements.

Name: **Altered Data Marks**
Entry Conditions: Proper DATA pointers
Line Number(s): 21Ø

Same as previous subroutine except for DATA field prologue marks.

Name: **Normalizer**
Entry Conditions: None
Line Number(s): 23Ø - 25Ø

This routine restores the values in the RWTS subroutine that are changed by any routine in Super IOB. This routine should be called just before writing in order to fix the RWTS so that it can access normal DOS disks.

Name: **Exchange RWTSs**
Entry Conditions: A RWTS at $19ØØ
Line Number(s): 36Ø

This is the standard swap RWTS's routine. It uses a routine in IOB.OBJØ to exchange the RWTS at $19ØØ with the one at $B8ØØ, which is the normal place for an RWTS. To tell the machine language swap routine (which is invoked by a CALL 832) what to exchange, a few POKEs must be executed. They are:

POKE 253, start of first location
POKE 255, start of second location
POKE 224, number of pages to exchange (a standard RWTS is eight pages long)

Name: **Format Disk**
Entry Conditions: S2 = slot of disk to format, D2 = drive number of disk to format
Line Number(s): 38Ø - 41Ø

This routine formats the target disk. It was meant to be used before the controller takes hold of Super IOB (and is GOSUBed by "CONFIGURATION TIME") but can be called by the controller should the need arise.

Name: **Print Track & Sector#**
Entry Conditions: TK = The track number to display, ST = The sector number to display
Line Number(s): 43Ø

This is the subroutine that puts the current track and sector number at the top of the screen in hexadecimal during the softkey operation. It should be invoked just before reading or writing each sector.

Name: **Center Message**
Entry Conditions: A$ = The message
Line Number(s): 45Ø

This routine prints a message in the center

of the screen at the current VTAB position. Care should be taken that the message to print is not longer than 40 characters. Otherwise an error will result.

Name: **Print Message And Wait**
Entry Conditions: A$ = The message
Line Number(s): 470

This routine uses "CENTER MESSAGE" to print the intended message at a VTAB of 11 and then it prints "PRESS ANY KEY TO CONTINUE." It then waits for a keypress before RETURNing.

Name: **Toggle Read/Write**
Entry Conditions: CD = current command code
Line Number(s): 490 - 530

This routine toggles the state of CD (from ReaD to WRite and vice versa) and prints the current mode in flashing letters at the very top of the screen. In addition, if the user has only one drive, it asks him to swap disks. It then exits via "INITIAL IOB SETUP," thus making the sector buffer ready for the next operation.

Name: **Ignore Unreadable Sectors**
Entry Conditions: Not Applicable
Line Number(s): 550 - 590

If the controller should pay no attention to unreadable sectors, then somewhere in the beginning of it should be an "ONERR GOTO 550." This is used usually with RWTS.13 (since DOS 3.2 sectors are unreadable until they have been written to) but can be used with any disk that has unreadable sectors which should be ignored.

The Remainder Of The Program

Lines 1000 through 9999 are meant for the controller and all DATA statements it contains. All lines greater than 9999 are used by the error trapper or the configurer which consists of all the prompts when the program is run. The error trapper will print a disk error and stop the program. If the error wasn't a disk error, then the error trapper will let it occur.

Now that you have an idea of the subroutines, take a look at the following variable usage table and note how the variables relate to them.

Table 1

A - general temporary usage, scrambled by "MOVE S PHASES" and "THE SECTOR EDITOR."

A$ - holds message to pass to the user via "CENTER MESSAGE" and "PRINT MESSAGE AND WAIT", scrambled by "TOGGLE READ/WRITE."

A1,A2,A3,A4 - scrambled by "ALTERED ADDRESS MARKS", "ALTERED DATA MARKS", "THE SECTOR EDITOR", "ALTERED ENDING MARKS" and "ALTERED DATA CHECKSUM", they are READ from DATA statements and POKEd into the appropriate RWTS to change it.

B$ - altered only by "CONFIGURATION TIME."

BF - Buffer Full holds the status of the sector buffer, set to 1 if the buffer is either full or empty and 0 if neither, changed only by "R/W SECTOR."

BUF - BUFfer constant holds the address where the RWTS is expecting to find the page number of the sector, used by "INITIAL IOB SETUP" and "R/W SECTOR", a PEEK(BUF) will return the current sector buffer page number.

CD - CommanD code is used by the controller, "TOGGLE READ/WRITE" and "R/W SECTOR", holds the current RWTS command code (see RD, WR, and INIT).

CMD - CoMmanD code constant holds the address where the RWTS is expecting to find the previously stated command code, used by "R/W SECTOR", a POKE CMD,CD will change the IOB command.

D1 - Drive 1, set during configuration to the drive number of the source drive, used by "TOGGLE READ/WRITE."

D2 - Drive 2, same as above except for target drive.

DOS - Disk Operating System, the number of sectors to read or write, initialized to 16.

DRV - DRiVe constant, holds the address where the RWTS is expecting to find the drive number of the drive to be accessed, used by "INITIAL IOB SETUP", a PEEK(DRV) will return the drive last accessed.

DV - current DriVe, used by "INITIAL IOB SETUP", "TOGGLE READ/WRITE", and "MOVE S PHASES", holds the drive number of the drive to be accessed next.

ERR - ERRor code, used by "DISK ERROR" to determine the error that has just occured.

INIT - INITialize command code, a CD = INIT will set the command code to format the diskette.

IO - Input/Output constant, holds a 768 (set during configuration), CALLed by "R/W SECTOR" to induce the RWTS subroutine.

MB - Maximum Buffer page holds the last page of memory for the sector buffer, used by "R/W SECTOR", initialized (during configuration) to 151 and should be changed to 130 only when a 13-sector disk is read or written.

OVL - Old VoLume constant, a PEEK(OVL) will return the volume number of the previously accessed diskette.

PH - current PHase, if "MOVE S PHASES" is referenced (by the controller), this variable must contain the disk arms current phase number (PH = 2*TK).

RD - ReaD command code, a CD = RD will set the command to read the disk.

S - Step, used to tell "MOVE S PHASES" how may phases to step through (-120 to 120).

S1 - Slot 1, set to the slot number of the source drive during configuration, used by "TOGGLE READ/WRITE".

S2 - Slot 2, same as above except for target drive.

SCT - SeCTor number constant holds the address where the RWTS is expecting to find the sector to be accessed, used by "R/W SECTOR" to tell the RWTS which sector is to be read or written, a PEEK(SCT) will return the last accessed sector number.

SLT - SLoT number constant holds the address where the RWTS is expecting to find the slot number of the disk to accessed next, used by "INITIAL IOB SETUP", a PEEK(SLT) will return the last accessed disks slot number.

SO - SlOt number is used by "TOGGLE READ/WRITE" and "INITIAL IOB SETUP", holds the slot number of the disk to be accessed next.

ST - SecTor number is used by the controller to tell "R/W SECTOR" what sector number is to be read or written next.

TK - TracK number is used by the controller to tell "R/W SECTOR" what track is to be accessed next.

TRK - TRacK number constant holds the memory location where the RWTS is expecting to find the track to be accessed, a PEEK(TRK) will return the last accessed track number.

VL - VoLume number is used by the controller to tell "TOGGLE READ/WRITE" (which passes it to "INITIAL IOB SETUP") the volume number of the disk to be accessed next.

VL$ - altered only by "FORMAT DISK".

VOL - VOLume number constant holds the memory location where the RWTS is expecting to find the volume to be accessed, a PEEK(VOL) will return the volume number last used by the controller.

WR - WRite command code, a CD = WR will set the command to write.

ADVENTURE TIPS ADVENTURE TIPS

*** Zork II**
Infocom, Inc.,

The answer to the riddle is used for getting water.
Imagine what you could do if the demon gave you the Wizard's wand!
You might be able to get past the huge rock if you could somehow make it float.
Don't go too high in the balloon; you might never come down.
Something as pretty as that rose must be a treasure.
Make sure that you have a treasure to give to the bank teller. You'll be able to get it back later.
You can learn a lot from dying. Read the descriptions carefully.
Try putting the black sphere in the middle of the pentagram.

*** Enchanter**
Infocom, Inc.

Look for mouse tracks in the library; they might lead to something.
Sometimes mighty monsters will kill themselves if they are changed into other smaller forms.
Examine the underside of those lily pads.
The pencil can be used to create passages, but don't use up the eraser too soon.
Make sure the unseen terror doesn't escape from his prison; if he does, you're doomed.

* Contributed by Chris W. Cowell

Mask Of The Sun
Ultrasoft, Inc.

Bones are easily broken by hitting them with something hard.
The old woman with the flute is hungry.
Going southwest is like Rolaids when it comes to gas.
Either a coffin, a fire, or a man is the answer to the riddle.

Contributed by Andrew T. Welch

† Zork II
Infocom, Inc.

There are two parts to the riddle, one outside the door and one inside.
Let the robot touch the buttons.
Turn off the sound.

† Planetfall
Infocom, Inc.

Not all the robots are ruined.
What does a U-shaped object remind you of?
Treat the cards like you treat your disks.
Microbes don't like to eat hot things.

† Contributed by Tan Wee Meng

‡ Exodus
Origin Systems

Dawn is near the brothers.
People in prisons play cards.
Being exotic is best when you are in death.
Your friends need marks and cards, too.

‡ Ultima II
Sierra On-Line

In the time of A.D. there is a village in which you can find shuttles.
To fly a shuttle, your skin must be "Power"ful.

‡ Mystery House
Sierra On-Line

To find the matches, look where the chef hangs out.
To light the lamp, you must light a match first.

‡ Contributed by Dr. Duplicate

Softporn Adventure
Sierra On-Line

Everyone needs money on an adventure.
Go to the casino and win about $65Ø0.
Listen carefully to what the bum tells you.
If you're going to be drinking, don't drive.
Have you met the Funky Hooker? Be careful.

Adventureland
Adventure International

Got a chigger bite? Put mud on it to stop the swelling.
Squirrels hide their treasure in stumps.
Maybe you can, too.
You'll need to light a lamp to see down the bottomless hole.
Swamp gas is really methane, isn't it?
Could be useful for "blasting" brick walls.
You must "jump" over the chasm, but bricks will weigh you down, so don't take them.

Memory Usage

Before actually looking at some controllers, let's say a few words about memory usage.

Following is a memory allocation table for the various parts of Super IOB. It is extremely important to stay within the boundaries when writing a controller, otherwise, horrible things might happen (the least of which would be the production of an incorrect copy).

Memory Allocation Table

$Ø8ØØ.$18FF (2Ø48-6399) intended for the Applesoft part of Super IOB
$19ØØ.$2ØFF (64ØØ-8447) space allocated for a moved RWTS (RWTS.13 or other)
$21ØØ.$26FF (8448-9983) Super IOB Applesoft variable space
$27ØØ.$96FF (9984-38655) enough space for 7 tracks, this is the sector buffer

First, notice the amount of space available for the BASIC program. The Super IOB program as listed (with all REMs) ends about 32ØØ bytes short of the $2ØFF (or 8447) limit. This leaves ample room for any controller that doesn't use an alternate RWTS. However, if a controller does use an alternate RWTS, then there are only about 12ØØ bytes free for it. In view of the space requirement, the end of program should be checked by typing

PRINT PEEK(175) + PEEK(176)*256

before a controller with an alternate RWTS (swap controller etc) is used. If it has exceeded the 6399 limit, I suggest DELeting all subroutines not referenced by the controller and all REM lines until it fits within the allocated space.

Second, observe the 1534 bytes for variables. This should be enough space for the simple softkey procedure. It is impossible to allocate more memory for variables and use an alternate RWTS file. If you find that you need more memory and the program does not use RWTS.13 or some other moved RWTS, the LOMEM: 8448 command may be removed from line 6Ø. This will allocate what isn't used (by the BASIC program) of the 2K area reserved for the relocated RWTS as variable space.

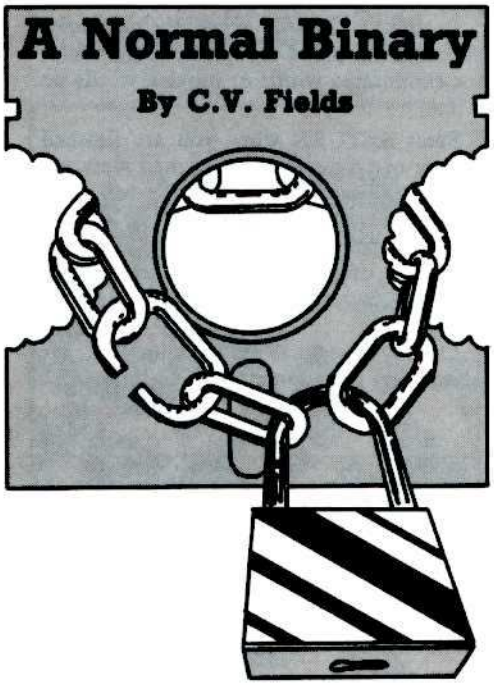
Never omit the "HIMEM:" statement! This could cause variables to overflow into the sector buffer, thus making a faulty copy.

With all this new knowledge, we are finally ready to scrutinize some sample controller programs. Keep in mind that protection schemes can be used with one another. Therefore, a more sophisticated controller for Super IOB will probably be required for most softkeys. Even so, developing new controllers isn't difficult.

Continued on page 24

Putting Locksmith 5.0 Fast Copy Into A Normal Binary File

By C.V. Fields



Locksmith 5.0
Alpha Logic Business Systems Inc.
4119 North Union Road
Woodstock, IL 60098
(815) 568-5166
$99.95

Requirements:
Locksmith 5.0, Rev F
A way to RESET into the Apple Monitor
A blank disk

In the past, I have seen two procedures for placing the "16 SECTOR FAST DISK BACKUP" portion of Locksmith 5.0 into a normal binary file that you can BRUN. However, neither procedure was easy to follow and both required that you write memory move routines and save memory from Page 0 through the end of the program. This method resulted in a program that was much longer than necessary, in addition to the procedures being difficult to follow.

Most readers of this magazine should be able to follow this procedure which will produce a 46-sector BRUNable version of the fast copy program. It may be a problem for some readers to enter the Apple monitor at just the right time. I use a REPLAY card, but several methods should work, for example the Old Monitor ROM or moving your RAM card to Slot 1 (See Chris Rys' article "Softkey for Sensible Speller" in Hardcore COMPUTIST No. 9).

What follows is a narrative of what I did. Those of you who can't wait or don't care about the detail can skip to the step-by-step procedure.

I first stored $11 in all memory locations from $0800 through $95FF so I could determine where the program loads; (CALL -151 N 800:11, 801<800.95FFM). I then booted LS 5.0 and selected the "16 SECTOR FAST DISK BACKUP" from the utility menu. The instant the disk drive light went out I pressed the button on my REPLAY card and then selected M to go into the APPLE monitor. A quick scan of memory, (800.BFFF), showed that the program extended from approximately $800 through $3FFF, although some of the code was suspect. There was also some code above $8000 that is moved there during the program initialization and is not needed as part of the binary file we will save.

Because Page $60 was empty, I moved Page $8 there so I could boot a DOS 3.3 slave disk, (6000<800.8FFM). I then booted a DOS 3.3 slave disk, with the HELLO deleted, and went back to the monitor so I could move Page $8 back down to where it belongs; (CALL -151, 800<6000.60FFM). Then I saved the file with, BSAVE LS 5.0 FAST COPY,A$800,L$3800.

Now, I knew this file wouldn't work "as is" because when you BRUN a program DOS BLOADs it and then JuMPs to the starting address. So the next thing I had to do was locate the entry point of the program so I could put a jump to it at the beginning of the file.

I used the INSPECTOR/WATSON utility to examine the code. If you BLOAD the file and then step through the buffers on I/W, you can view the program in memory with all the ASCII text identified. As I scanned the program, I made notes on likely starting points, where all the ASCII text was located and possible code at the beginning and end that could be eliminated.

I then started testing the entry points that I had on my list. As luck would have it, one of the first points I tested, $2002, bombed me into Zero Page (sometimes a bomb means luck) at an address of $AB. Since the monitor shows the address two steps beyond the break I subtracted $2 and booted the original LS 5.0 disk again. After it loaded I used the REPLAY card to check location $A9 and found a $60, which is a "ReTurn from Subroutine" instruction.

Again I BLOADed my test file and this time I went into the monitor and placed a $60 at $A9 before I typed 2002G. Much to my surprise, it worked perfectly.

As I said earlier, some of the code looked unnecessary to the program so I started cutting off the suspect code a page at a time and testing the program until I determined that only Pages $0A through $36 were required. There was also enough room at the beginning of Page $0A for our starting code and at the end of Page $36 so we could cut to an even full sector. When you BSAVE a file, the starting address and length are saved at the beginning of the first sector. This extends the code by 4 bytes. The length then becomes $36FF-$A00 = $2CFF and $2CFF-4 = $2CFB. The code will, therefore, take up $2D or 45 sectors. Add one sector for a track/sector list, and a total of 46 sectors will be needed.

After some code searching, I also discovered that the slot number for all the copying is stored at location $A4B. Adding this to the fact that there is some free space at the beginning of page $A, I wrote a little startup routine that will allow you to change slot numbers. Using this routine you can place the 16 SECTOR FAST BACKUP file on a hard disk.

Therefore, to put your 16 SECTOR FAST BACKUP in a normal DOS 3.3 file, follow these steps:

Step-By-Step Procedure

1) Init a DOS 3.3 slave disk and delete the HELLO program. I recommend one of the rapid DOS programs such as Diversi-DOS, PRONTO DOS or my favorite, RAPIDOS][.

```
INIT HELLO
DELETE HELLO
```

2) Boot your original of Locksmith 5.0, Rev F, and select the 16 SECTOR FAST DISK BACKUP from the utility menu.

3) The instant the disk drive light goes out (while loading the Fast Disk Backup) exit to the Apple monitor using your favorite method.

4) Boot your DOS 3.3 slave disk (Note: Since Page $8 is not needed, we don't have to move it out of the way)

```
6CTRL-P
```

5) Enter the monitor and key in the hexdump on the following page

```
CALL -151
```

Continued on page 27

* See No. 16, page 5 - (letter) - correction & comment

Batman Decoder Ring

By Ray Darrah

CORE

A short while ago, I intercepted a note passing from Mark Fuchs to Jeff Power during my French class. After eagerly opening the message, here is what I saw:

Vgx ci xio sgemj u Billiciqa rewsx dioq er liqa nivaqdok sgum um Unnka svi?

Foiled Again

The message was obviously in some sort of code. It appeared as though letter substitution (where each letter has been translated into a different one) had been used. Since I didn't have enough time to figure out the code during class, I decided to create a computer program which would aid me in the cracking of this mysterious message.

The Program

With a little help from a teacher named Mr. Jacobson. I was able to create a small program which did the job. This really sparked my interest in deciphering messages. Soon I was wanting more from my program. So I updated it, made it more user friendly, increased its power and made it faster. The resulting program and complete instructions for its use are presented below.

Typing It In

When keying in the Applesoft program be sure to SAVE it before you RUN it. Otherwise you may lose a lot of typing. The Applesoft portion must be saved like this:

SAVE BATMAN DECODER RING

Also, do a FP before keying in the hexdump since it occupies the area occupied by a normal Applesoft program. The hexdump should then be saved with

BSAVE BDR.OBJ,A$800,L$12A

Step-By-Step Through The Program

After you press a key to get past the title page, you will see something like this:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ

E)DIT                Q)UIT
C)OUNT               R)EMOVE
H)ARD COPY           P)UNCTUATE
D)ISK ACCESS         S)TART OVER
```

Each of these options is selected by typing the control character corresponding to the first letter of the options. That is: If you want to quit the program, type CTRL-Q.

The the top three-fourths of the screen is blank because that is where the encrypted message would go if you had one. There are two ways to get an encrypted message into the computer. You may load a previously saved message or you may type in a new message.

You may type in a new message by selecting **EDIT**. This option is normally used to edit the encoded message in the event of error. However, it may be used to enter a message if you have none.

While editing the message, a specialized input routine that allows editing is used. Typing a CTRL-I will insert a space at the cursor location. This is helpful for inserting words or moving a piece of a word to the next line.

Pressing CTRL-D will delete the character directly to the left of the cursor. This is useful for eliminating words or moving words up a line.

Press RETURN when you are finished editing or typing in an encrypted message. Batman Decoder Ring (BDR) will then decode the message using the current key and count the most used letter in the message. The most often used letter in the English language (and, therefore, most cryptograms) is E.

Once a message is loaded into BDR, the screen will look something like this:

```
------ -------- -- --- -------- ----
AYXWVU TYVSRYZ, QY XWY PXONYWYZ NYWY

----- -- --- ------- ---- ----- ------
OSZXU OS PYO ONWSGPN ONML ONMKP JXVVYZ

----- -------- ---- ----- -- -----
VMIY. YVYJOWMJ QSWZ VMIY, MO HYXKL

------- --- ------ ------ ---------
ISWYRYW XKZ ONXO'L HMPNOU VSKPOMHY.

--- --- ---- -- ---- --- -------
TGO M'H NYWY OS OYVV USG ONYWY'L

--------- ----- --- ----------
LSHYONMKP YVLY: ONY XIOYWQSWVZ

     ABCDEFGHIJKLMNOPQRSTUVWXYZ

  E)DIT                Q)UIT
  C)OUNT               R)EMOVE
  H)ARD COPY           P)UNCTUATE
  D)ISK ACCESS         S)TART OVER
```

BDR can be used to encode messages as well as decode them. If you wish to encode a message use the EDIT function to type in the unencoded message and start encoding.

Letter Substitution

At this point you may begin the process of decoding (or encoding). You merely type the letter (of the encoded message) you wish to change followed by the letter you wish to change it to. After you assign a letter in this fashion, the second letter you typed will disappear from the alphabet above the menu. This means that you may not assign any other letter to that same letter.

Let's say you decided that X should be E and then later you decided that P should be E. You will first have to use the **REMOVE** function and remove the assignment to E.

Once you type CTRL-R to invoke the remove routine, you press the letters of the alphabet above the menu you wish to bring back. This will cancel the letters they are assigned to. If you type CTRL-A, BDR will erase all the substitutions you have done so far. You may leave the remove routine by typing CTRL-E (or space or return).

The **QUIT** routine clears the screen and exits the program.

The **START OVER** routine will erase any unsaved work and start you back at the title page.

Pressing CTRL-P will invoke the **PUNCTUATE** routine. This routine will go through the encoded string and replace all the non-letters with the same value. This is a faster method of assigning a period to a period, a comma to a comma etc.

The **COUNT** routine is invoked by pressing CTRL-C this routine will print a question mark and wait for you to type the letter you wish to count. As soon as you press the letter, BDR will count the occurrences of this letter in the encoded string. The result is displayed directly below the cursor.

To obtain a printout of the message, you would type CTRL-H to invoke the **HARD COPY** routine. The first question you will be asked is which slot your printer is in. Next, you will be asked what you want printed. Press one through three.

The most complex of the menu items is the **DISK ACCESS** routine. This routine will allow the loading and saving of messages and keys. A key is a list of which characters get translated to what.

When you see the menu:

```
1= ENCODED MESSAGE
2= UNENCODED MESSAGE
4= KEY
```

you should add the values to the left of the words corresponding to which things you want to load or save. For example, if you wanted to load the key and the encoded

message, you would press 5. On the other hand if you wanted to load all of them you would press 7 (4 + 2 + 1 = 7).

When working on a cryptogram, it is best to load and save all of them (press 7). However, if you want to just save one string so that someone else may decode it, you could press another number.

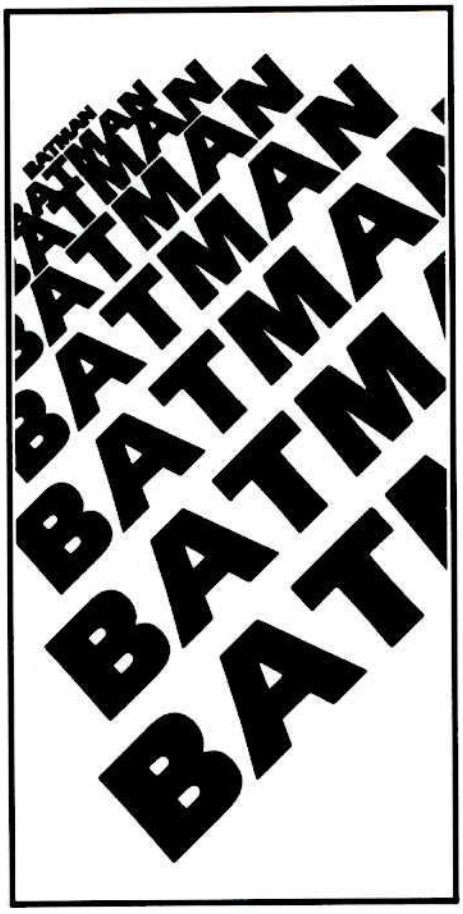
One For The Road

Here is something for you to attempt to decode (if you have already decoded the two examples above). It was encoded by BDR and is probably very dear to the hearts of all Hardcore COMPUTIST readers:.

```
TALKING TLGREHG:
'ALOLOPLA, NGJM MNE SUG TALCLGI KNWIDUAL
THAUSM.''
```

Batman Decoder Ring

```
10 REM [++++++++++++++++++++++]
20 REM [                      ]
30 REM [ BATMAN DECODER RING  ]
40 REM [                      ]
50 REM [     BY RAY DARRAH    ]
60 REM [                      ]
70 REM [++++++++++++++++++++++]
80 REM
90 IF PEEK (104) <> 10 THEN POKE 103
    ,1 : POKE 104 ,10 : POKE 2560 ,Z :
    PRINT CHR$ (4) "RUN^BATMAN^
    DECODER^RING"
100 PR# 0 : IN# 0 : CALL 1002 : IF PEEK
    (2048) = Z THEN PRINT CHR$ (4)
    "BLOAD^BDR.OBJ,A$800"
110 TEXT : NORMAL : SPEED= 255 :
    NOTRACE : POKE 10 ,76 : POKE 11 ,2
    : POKE 12 ,9
120 HOME : POKE 48 ,174 : GOSUB 190
    :A$ = "!/-" + CHR$ (92)
130 VTAB 3 : HTAB 11 : PRINT "BATMAN^
    DECODER^RING" : VTAB 6 : HTAB 14
    : PRINT "BY^RAY^DARRAH"
140 VTAB 12 : HTAB 10 : PRINT "PRESS^
    ANY^KEY^TO^BEGIN" :W = 1 : DIM
    C(63) :Z = 0
150 FOR A = W TO LEN (A$) : POKE 48 ,
    ASC ( MID$ (A$ ,A ,W) ) + 128 :
    GOSUB 190 : POKE - 16336 ,Z
160 IF PEEK ( - 16384 ) < 128 THEN NEXT
    : GOTO 150
170 E3$ = "^" : POKE - 16368 ,Z : FOR A
    = W TO 64 :E3$ = E3$ + CHR$ (95) :
    NEXT
180 E4$ = "^" : FOR A = 65 TO 90 :E4$ =
    E4$ + CHR$ (A) : NEXT :E4$ = E4$ +
    "^" : GOTO 210
190 VLIN Z ,47 AT Z : VLIN Z ,47 AT 39 :
    HLIN Z ,39 AT Z
200 HLIN Z ,39 AT W : HLIN Z ,39 AT 46 :
    HLIN Z ,39 AT 47 : RETURN
210 HOME :P$ = E2$ : GOSUB 700 : VTAB 2
    :P$ = E1$ : GOSUB 700
220 VTAB 19 : HTAB 7 : PRINT E4$; :
    VTAB 21 : HTAB W : PRINT "E)DIT"
    TAB( 30) "Q)UIT"
230 PRINT "C)OUNT" TAB( 30)
    "R)EMOVE" : PRINT "H)ARD^COPY"
    TAB( 30) "P)UNCTUATE"
240 PRINT "D)ISK^ACCESS" TAB( 30)
    "S)TART^OVE" ; : POKE 2039 ,210 :
    POKE 35 ,18 : POKE 34 ,0
250 K$ = CHR$ (5) + CHR$ (17) + CHR$
    (3) + CHR$ (18) + CHR$ (8) + CHR$
    (16) + CHR$ (4) + CHR$ (19)
260 VTAB 21 : HTAB 18 : GET A$ : FOR A =
    W TO LEN (K$) : IF A$ <> MID$ (K$
    ,A ,W) THEN NEXT : GOTO 280
270 ON A GOSUB 370 ,810 ,480 ,570 ,720
    ,520 ,830 ,790 : VTAB 21 : HTAB 18
    : PRINT SPC( 4); : GOTO 260
280 IF A$ < "!" OR A$ > CHR$ (95) OR
    E2$ = "" THEN 260
290 PRINT A$ "=>" ; : GET B$ : IF B$ <
    "!" OR B$ > CHR$ (95) THEN HTAB 18
    : PRINT SPC( 4); : GOTO 260
300 IF B$ < CHR$ (91) AND B$ > "@"
    THEN IF MID$ (E4$ , ASC (B$) - 63
    ,W) = "^" THEN HTAB 18 : PRINT
    SPC( 4) CHR$ (7); : GOTO 260
310 C$ = MID$ (E3$ , ASC (A$) - 31 ,W)
    : IF C$ < CHR$ (91) AND C$ > "@"
    THEN E4$ = LEFT$ (E4$ , ASC (C$) -
    64) + C$ + RIGHT$ (E4$ ,91 - ASC
    (C$) )
320 PRINT B$; :E3$ = LEFT$ (E3$ , ASC
    (A$) - 32) + B$ + RIGHT$ (E3$ ,64
    - ASC (A$) + 32)
330 POKE 253 , ASC (A$) + 128 : POKE
    254 , ASC (B$) :X = USR (E2$) :
    VTAB W :P$ = E2$
340 IF B$ < CHR$ (91) AND B$ > "@"
    THEN E4$ = LEFT$ (E4$ , ASC (B$) -
    64) + "^" + RIGHT$ (E4$ ,91 - ASC
    (B$) )
350 GOSUB 700 : VTAB 19 : HTAB 7 :
    PRINT E4$; : HTAB 18 : VTAB 21 :
    PRINT SPC( 4); : GOTO 260
360 REM EDIT ENCODED STRING
370 HOME : VTAB 2 :P$ = E1$ : GOSUB 700
    : POKE 255 , LEN (E1$) : VTAB 2 :
    HTAB W : PR# Z
380 POKE 56 ,Z : POKE 57 ,8 : POKE 512
    ,Z : INPUT "" ;E1$ : POKE 253 ,
    PEEK (131) : POKE 254 , PEEK (132
    )
390 A = PEEK (253) + PEEK (254) * 256
    : POKE A , PEEK (255) : POKE A + W
    ,128 : POKE A + 2 ,2
400 P$ = E1$ : VTAB 2 : GOSUB 700 :E2$
    = "................"
410 FOR A = W TO 4 :E2$ = E2$ + E2$ :
    NEXT :E2$ = LEFT$ (E2$ , LEN (E1$
    )) : FOR A = W TO LEN (E3$)
420 POKE 253 ,159 + A : POKE 254 , ASC
    ( MID$ (E3$ ,A ,W) ) :X = USR (E2$
    ) : NEXT
430 FOR A = Z TO 63 :C(A) = Z : NEXT :
    FOR A = W TO LEN (E1$)
440 X = ASC ( MID$ (E1$ ,A ,W) ) - 160
    :C(X) = C(X) + W : NEXT :X = W :
    FOR A = W TO 63 : IF C(A) > C(X)
    THEN X = A
450 NEXT :P$ = E2$ : VTAB W : GOSUB 700
460 VTAB 23 : HTAB 18 : PRINT C(X) ">"
    CHR$ (32 + X) "^" ; : IN# Z : CALL
    1002 : RETURN
470 REM COUNT A LETTER
480 PRINT "?=>" ; : GET A$ : PRINT A$;
    : IF A$ < "!" OR A$ > CHR$ (95)
    THEN RETURN
490 A$ = CHR$ ( ASC (A$) + 128) :X = Z
    : FOR A = W TO LEN (E1$) : IF MID$
    (E1$ ,A ,W) = A$ THEN X = X + W
500 NEXT : VTAB 23 : HTAB 18 : PRINT X
    ">" A$ "^" ; : RETURN
510 REM PUNCTUATE
520 IF E1$ = "" THEN RETURN
530 PRINT "PUNC" ; : FOR A = 161 TO 192
    : POKE 253 ,A : POKE 254 ,A - 128
540 X = USR (E2$) : NEXT :P$ = E2$ :
    VTAB W : GOSUB 700 :A$ = "" : FOR A
    = 32 TO 64
550 A$ = A$ + CHR$ (A) : NEXT :E3$ = A$
    + RIGHT$ (E3$ ,32) : RETURN
560 REM REMOVE MODE
570 IF E1$ = "" THEN RETURN
580 POKE 34 ,20 : POKE 35 ,24 : HOME :
    FLASH : PRINT "REMOVE" : NORMAL :
    PRINT : PRINT "A)LL" TAB( 30)
    "E)XIT"
590 VTAB 21 : HTAB 18 : PRINT "^" CHR$
    (8); : GET A$ : IF (A$ < "!" OR A$
    > CHR$ (94) ) AND A$ <> CHR$ (5)
    AND A$ <> CHR$ (W) THEN A$ = CHR$
    (5)
600 PRINT A$; : IF A$ = CHR$ (5) THEN
    POP : TEXT : GOTO 210
610 IF A$ = CHR$ (W) THEN 660
620 FOR A = W TO LEN (E3$) : IF MID$
    (E3$ ,A ,W) <> A$ THEN NEXT :
    PRINT CHR$ (7); : GOTO 590
630 POKE 253 ,159 + A : POKE 254 ,95 :X
```



```
    =USR (E2$) :E3$ = LEFT$ (E3$ ,A -
    W) + CHR$ (95) + RIGHT$ (E3$ ,65
    - A)
640 IF A$ < CHR$ (91) AND A$ > "@"
    THEN E4$ = LEFT$ (E4$ , ASC (A$) -
    64) + A$ + RIGHT$ (E4$ ,91 - ASC
    (A$))
650 VTAB W :P$ = E2$ : GOSUB 700 : VTAB
    19 : HTAB 7 : PRINT E4$; : GOTO 590
660 E4$ = "^" : FOR A = 65 TO 90 :E4$ =
    E4$ + CHR$ (A) : NEXT :E4$ = E4$ +
    "^" :E3$ = "^"
670 FOR A = 33 TO 95 :E3$ = E3$ + CHR$
    (95) : NEXT :E3$ = E3$ + "^" : FOR
    A = W TO LEN (E3$) - W
680 POKE 253 ,A + 159 : POKE 254 , ASC
    (MID$ (E3$ ,A ,W)) :X = USR (E2$
    ) : NEXT : TEXT : GOTO 210
690 REM PRINT SPACED OUT STRING
700 HTAB W : FOR A = W TO LEN (P$) STEP
    40 : PRINT MID$ (P$ ,A ,40); :
    VTAB PEEK (37) + 3 : NEXT : RETURN
710 REM HARD COPY
720 HOME : INPUT "PRINTER^SLOT=>"
    ;PR : IF PR > 7 OR PR < 0 THEN 210
730 VTAB 5 : PRINT "1)^" LEFT$ (E1$
    ,20) : PRINT : PRINT "2)^" LEFT$
    (E2$ ,20) : PRINT : PRINT "3)^
    BOTH" : PRINT : PRINT
740 PRINT "^WHICH?" CHR$ (8); : GET
    A$ : PRINT A$; :X = VAL (A$) : IF
    A$ > "3" OR A$ < "1" THEN 730
750 HOME : PR# PR : FOR A = W TO LEN
    (E1$) STEP 40 : IF X > W THEN
    PRINT MID$ (E2$ ,A ,40)
760 IF X = W OR X = 3 THEN PRINT MID$
    (E1$ ,A ,40)
770 PRINT CHR$ (13); : NEXT : PR# Z :
    HOME :P$ = E2$ : GOSUB 700 :P$ =
    E1$ : VTAB 2 : GOTO 700
780 REM START OVER
790 RUN
800 REM LEAVE PROGRAM
810 POP : TEXT : HOME : PRINT "BYE" :
    END
820 REM DISK ACCESS
830 POP : ONERR GOTO 1180
840 TEXT : HOME : PRINT TAB( 15 )
    "DISK^ACCESS" : VTAB 7 : PRINT
    "1)^LOAD^FILES"
850 PRINT : PRINT "2)^SAVE^FILES" :
    PRINT : PRINT "3)^DOS^COMMAND"
860 PRINT : PRINT : PRINT : PRINT :
    PRINT "^WHICH^?" CHR$ (8);
870 GET A$ : IF A$ > "3" OR A$ < "1"
    THEN 210
880 IF A$ = "3" THEN 1210
890 HOME :X = VAL (A$) : PRINT TAB( 16
    ) "LOAD^FILES" ; : HTAB 16 : IF X =
    2 THEN PRINT "SAVE" ;
900 PRINT : VTAB 7 : PRINT "1=^
    ENCODED^STATEMENT" : PRINT :
    PRINT "2=^UNENCODED^STATEMENT"
910 PRINT : PRINT "4=^KEY" : VTAB 14 :
    PRINT "ADD^THE^VALUES^AND^
    SELECT^#^(1^-^7)"
920 PRINT : PRINT : PRINT "^WHICH^?"
    CHR$ (8);
930 GET A$ : IF A$ < "1" OR A$ > "7"
    THEN 210
940 PRINT A$ :A = VAL (A$) : PRINT :
    PRINT : INPUT "FILENAME=>" ;C$ :
    HOME :C1$ = "" : FOR B = 512 + LEN
    (C$) TO 768
950 IF PEEK (B) THEN C1$ = C1$ + CHR$
    (PEEK (B)) : NEXT
960 PRINT :D$ = CHR$ (4) : IF X = W
    THEN PRINT D$ "OPEN" C$C1$ :
    PRINT D$ "READ" C$ : GOTO 1020
970 PRINT D$ "OPEN" C$C1$ : PRINT D$
    "DELETE" C$ : PRINT D$ "OPEN" C$
    : PRINT D$ "WRITE" C$
980 IF A / 2 <> INT (A / 2) THEN FOR B =
    W TO LEN (E1$) : PRINT CHR$ ( ASC
    ( MID$ (E1$ ,B ,W) ) - 128); :
    NEXT
990 PRINT CHR$ (13); :A = INT (A / 2)
    : IF A / 2 <> INT (A / 2) THEN
    PRINT E2$;
1000 PRINT CHR$ (13); :A = INT (A / 2
    ) : IF A / 2 <> INT (A / 2) THEN
    PRINT E3$E4$;
1010 PRINT CHR$ (13 )D$ "CLOSE" :
    GOTO 210
1020 X = A : GOSUB 1260 :C$ = A$ :
    GOSUB 1260 : IF A / 2 <> INT (A / 2
    ) AND A$ <> "" THEN E1$ = A$
1030 C = 1 :A = INT (A / 2) : IF A / 2 <>
    INT (A / 2) AND C$ <> "" THEN E1$
    = C$
1040 A = INT (A / 2) : GOSUB 1260 : IF A
    / 2 <> INT (A / 2) AND A$ <> ""
    THEN E3$ = LEFT$ (A$ ,65) :E4$ =
    RIGHT$ (A$ ,28) : GOTO 1060
1050 C = 0 :E4$ = "^" : FOR A = 65 TO 90
    :E4$ = E4$ + CHR$ (A) : NEXT :E4$
    = E4$ + "^"
1060 E2$ = "..............." : FOR A
    = W TO 4 :E2$ = E2$ + E2$
1070 NEXT :E2$ = LEFT$ (E2$ , LEN (E1$
    )) : IF X = 4 THEN 1100
1080 FOR A = W TO LEN (E1$)
1090 POKE 639 + A , ASC ( MID$ (E1$ ,A
    ,W) ) + 128 : NEXT
1100 B = LEN (E1$) :E1$ = "" : POKE
    253 , PEEK (131)
1110 POKE 254 , PEEK (132) :A = PEEK
    (253) + PEEK (254) * 256
1120 POKE A ,B : POKE A + W ,128 : POKE
    A + 2 ,2 : FOR A = W TO LEN (E3$)
1130 POKE 253 ,A + 159 : POKE 254 , ASC
    ( MID$ (E3$ ,A ,W) ) :X = USR (E2$
    ) : NEXT
1140 IF C THEN 1010
1150 FOR A = 34 TO 59 :B = ASC ( MID$
    (E3$ ,A ,W) ) : IF B <> 95 THEN
    E4$ = LEFT$ (E4$ ,B - 64) + "^" +
    RIGHT$ (E4$ ,91 - B)
1160 NEXT : GOTO 1010
1170 REM ERROR
1180 POKE 216 ,0 : TEXT : HOME : PRINT
    "ERROR^#" PEEK (222) CHR$ (7)
1190 FOR A = 1 TO 1000 : NEXT : GOTO
    1010
1200 REM DOS COMMAND
1210 HOME :D$ = CHR$ (13) + CHR$ (4)
1220 INPUT "COMMAND^->" ;A$ : IF A$ =
    "" THEN 840
1230 IF PEEK (512 + LEN (A$)) THEN
    VTAB PEEK (37) : CALL 64578 :B$ =
    ""
1240 FOR A = 512 + LEN (A$) TO 768 : IF
    PEEK (A) THEN B$ = B$ + CHR$ (
    PEEK (A)) : NEXT
1250 PRINT D$A$B$ : PRINT : GOTO 1220
1260 A$ = ""
1270 GET B$ : IF B$ <> CHR$ (13) THEN
    A$ = A$ + B$ : GOTO 1270
1280 RETURN
```

Hexdump With Checksums

```
0800: 85 FA 68 68 A9 10 85 38    $026C
0808: A9 08 48 A9 20 48 A5 FA    $FDD1
0810: 2C 00 C0 10 FB 91 28 AD    $9690
0818: 00 C0 2C 10 C0 60 20 0C    $AC62
0820: FD C9 89 D0 31 A4 FF F0    $4498
0828: F5 C0 EF B0 F1 E4 FF B0    $3A52
0830: ED 88 86 FA B9 80 02 C8    $CCDB
0838: 99 80 02 88 88 C4 FA D0    $84D4
0840: F3 B9 80 02 C8 99 80 02    $4DB1
0848: A9 A0 88 9D 80 02 E6 FF    $8C6A

0850: 20 D0 08 4C 1E 08 C9 84    $0513
0858: D0 23 A4 FF F0 C0 8A F0    $EC4D
0860: BD 86 FC C5 FF F0 0D BD    $07F7
0868: 80 02 CA 9D 80 02 E8 E8    $CFFB
0870: E4 FF D0 F3 C6 FF 20 D2    $F363
0878: 08 A9 88 D0 08 C9 88 D0    $B753
0880: 18 E0 00 F0 99 CA 20 5C    $2BD1
0888: DB A5 24 C9 27 D0 8F C6    $A061
0890: 25 C6 25 20 22 FC 4C 1E    $379F
0898: 08 C9 8D D0 03 A2 01 60    $B111

08A0: C9 95 D0 02 B1 28 C9 A0    $1805
08A8: 90 EC C9 DF B0 E8 9D 80    $114C
08B0: 02 E0 EF B0 E1 20 C1 08    $974F
08B8: E4 FF 90 02 E6 FF E8 D0    $488E
08C0: D5 20 5C DB A5 24 F0 01    $180D
08C8: 60 E6 25 E6 25 4C 22 FC    $5597
08D0: 86 FC A5 25 85 FA A5 24    $6191
08D8: 85 FB A9 01 85 25 20 22    $A496
08E0: FC A2 00 86 24 BD 80 02    $513D
08E8: 20 C1 08 E8 E4 FF 90 F5    $16F8

08F0: A9 A0 20 5C DB A5 FB 85    $74E3
08F8: 24 A5 FA 85 25 A6 FC 4C    $48AB
0900: 22 FC A0 00 B1 83 85 FF    $A01D
0908: C8 B1 83 85 FA C8 B1 83    $774E
0910: 85 FB A0 00 B9 80 02 C5    $0055
0918: FD D0 04 A5 FE 91 FA C8    $3A86
0920: C4 FF 90 F0 A9 00 A8 4C    $35A8
0928: F2 E2                      $4EC1
```

Source Code

```
*--------------------------------
* CBM ADVANCED ENCRYPTION INPUT ROUTINE
*
* BY RAY DARRAH
*--------------------------------

C000 KEYBOARD .EQ $C000
0038 KSWL     .EQ $38
C010 KEY.RESET .EQ $C010
0083 LAST.VAR .EQ $83       POINTER TO LAST
      USED VARIABLES VALUE
0028 BASL     .EQ $28
FD0C RDKEY    .EQ $FD0C
FC22 VTAB     .EQ $FC22     ADJUST BASL
0024 CH       .EQ $24       CURSOR HORIZONT
     AL
0025 CV       .EQ $25       CURSOR VERTICAL
DB5C OUTDO    .EQ $DB5C     BASIC PRINT ROU
     TINE
```

```
FC42 CLR.EOP   .EQ $FC42     ERASE REST OF S
     CREEN
FD8E CROUT     .EQ $FD8E     PRINT A CR
00FF LENGTH    .EQ $FF       MAXIMUM CHARACT
     ERS TYPED SO FAR
00FD STRING.PTR .EQ $FD      POINTS TO STRIN
     G TO PRINT
0069 VAR.PTR   .EQ $69       POINTER TO STAR
     T OF BASIC VARIABLES
0280 IN.BUFF   .EQ $280      INPUT BUFFER (S
     TRETCHES INTO PAGE THREE)
00FA TEMP1     .EQ $FA       TEMPORARY USAGE
00FB TEMP2     .EQ $FB
00FC TEMP3     .EQ $FC
00FD FROM      .EQ $FD       SUBSTITE THIS
00FE TO        .EQ $FE       WITH THIS

               .OR $800
               .TF BDR.OBJ

0800 START     STA TEMP1     SAVE SCREEN
0802           PLA           POP OFF RTS
0803           PLA
0804           LDA #KEYIN-START FIX KSWL
0806           STA KSWL
0808           LDA /GETCHAR+2 RTS 2 GETCHAR
080A           PHA
080B           LDA #GETCHAR+2
080D           PHA
080E           LDA TEMP1

0810 KEYIN     BIT KEYBOARD
0813           BPL KEYIN
0815           STA (BASL),Y
0817           LDA KEYBOARD
081A           BIT KEY.RESET
081D           RTS

081E GETCHAR   JSR RDKEY     GET A KEY

0821 CTRL.I    CMP #$89      INSERT?
0823           BNE CTRL.D
0825           LDY LENGTH    NO CHARS?
0827           BEQ GETCHAR   NO INSERT
0829           CPY #239      FULL?
082B           BCS GETCHAR
082D           CPX LENGTH    FULL
082F           BCS GETCHAR   YUP
0831           DEY
0832           STX TEMP1
0834 P2        LDA IN.BUFF,Y
0837           INY           INSRT SPACE
0838           STA IN.BUFF,Y
083B           DEY
083C           DEY
083D           CPY TEMP1     DONE?
083F           BNE P2        NOPE!
0841           LDA IN.BUFF,Y
0844           INY
0845           STA IN.BUFF,Y
0848           LDA #$A0
084A           DEY
084B           STA IN.BUFF,X
084E           INC LENGTH    ONE LONGER
0850           JSR PRINT.STRING
0853           JMP GETCHAR

0856 CTRL.D    CMP #$84      DELETE?
0858           BNE CTRL.H
085A           LDY LENGTH    NO CHARS?
085C           BEQ GETCHAR
085E           TXA
085F           BEQ GETCHAR
0861           STX TEMP3     SAVE X
0863           CMP LENGTH    END?
0865           BEQ DEC.ONLY
0867 P3        LDA IN.BUFF,X
086A           DEX
086B           STA IN.BUFF,X
086E           INX
086F           INX
0870           CPX LENGTH    DONE?
0872           BNE P3
0874 DEC.ONLY  DEC LENGTH    ONE SHORTER
0876           JSR PRINT.STRING+2 SKIP XSAV
     E
0879           LDA #$88      FAKE A CTRL.H
087B           BNE CTRL.H+8 ..ALWAYS

087D CTRL.H    CMP #$88      BACKUP?
087F           BNE CTRL.M
0881           CPX #0        NO CHARS?
0883           BEQ GETCHAR   YES, NO BACKUP
0885           DEX           ONE LESS CHAR
0886           JSR OUTDO     BACKUP
0889           LDA CH        WRAP AROUND?
088B           CMP #39       COL 40?
088D           BNE GETCHAR   NOPE!
088F           DEC CV        MOVE CURSOR UP
0891           DEC CV
0893           JSR VTAB
0896 GETCHR1   JMP GETCHAR

0899 CTRL.M    CMP #$8D      RETURN?
089B           BNE CTRL.U
089D           LDX #1        NULL INPUT
089F           RTS

08A0 CTRL.U    CMP #$95      MOVE FORWARD?
08A2           BNE LTR.CHK
08A4           LDA (BASL),Y GET SCREEN CHAR

08A6 LTR.CHK   CMP #$A0      < SPACE?
08A8           BCC GETCHR1
08AA           CMP #$DF      > UNDERLINE?
08AC           BCS GETCHR1
08AE           STA IN.BUFF,X
08B1           CPX #239      FULL?
08B3           BCS GETCHR1
08B5           JSR OUTDO1    PRINT IT
08B8           CPX LENGTH    EOL?
08BA           BCC P4        NO, NO INC LEN
08BC           INC LENGTH
08BE P4        INX           NEXT POS
08BF           BNE GETCHR1   ..ALWAYS

08C1 OUTDO1    JSR OUTDO     PRINT CHAR
08C4           LDA CH        WRAP AROUND?
08C6           BEQ MOVEDOWN
08C8           RTS
08C9 MOVEDOWN  INC CV
08CB           INC CV
08CD           JMP VTAB

     PRINT.STRING
08D0           STX TEMP3     SAVE POINTER
08D2           LDA CV
08D4           STA TEMP1
08D6           LDA CH
08D8           STA TEMP2
08DA           LDA #1        FIX BASL
08DC           STA CV
08DE           JSR VTAB
08E1           LDX #0        BEGIN TO PRINT
08E3           STX CH
08E5 P1        LDA IN.BUFF,X GET A CHAR
08E8           JSR OUTDO1
08EB           INX
08EC           CPX LENGTH
08EE           BCC P1        NOT DONE YET
08F0           LDA #$A0      SEND A SPACE
08F2           JSR OUTDO
08F5           LDA TEMP2
08F7           STA CH
08F9           LDA TEMP1
08FB           STA CV
08FD           LDX TEMP3     RESTORE X
08FF           JMP VTAB

*--------------------------------
* USR PATCH SUBSTITUTE ROUTINE
*--------------------------------

0902           LDY #0        LENGTH
0904           LDA (LAST.VAR),Y LAST USED V
     AR.
0906           STA LENGTH
0908           INY
0909           LDA (LAST.VAR),Y POINTER
090B           STA TEMP1     GET ADDR
090D           INY
090E           LDA (LAST.VAR),Y
0910           STA TEMP2
0912           LDY #0        START AT BEG
0914 P5        LDA IN.BUFF,Y GET BYTE
0917           CMP FROM      SWITCH?
0919           BNE NXT.1     NOPE

091B           LDA TO        YES!
091D           STA (TEMP1),Y

091F NXT.1     INY
0920           CPY LENGTH    DONE?
0922           BCC P5        NOPE
0924           LDA #0        RETURN A ZERO
0926           TAY
0927           JMP $E2F2
```

Basic Checksums

```
10   - $BADD     650  - $7CFF
20   - $9B13     660  - $E139
30   - $4D3B     670  - $BB88
40   - $AD92     680  - $4706
50   - $C899     690  - $59B1
60   - $FF65     700  - $0266
70   - $A3BF     710  - $41F2
80   - $A900     720  - $06BF
90   - $28BE     730  - $6C21
100  - $DE61     740  - $0941
110  - $52EF     750  - $6B83
120  - $6DE1     760  - $3BDA
130  - $27CF     770  - $E7E9
140  - $BA53     780  - $F06D
150  - $2491     790  - $AA98
160  - $1248     800  - $9DB8
170  - $8EFC     810  - $A81C
180  - $6F06     820  - $4998
190  - $6706     830  - $DB4E
200  - $F7BB     840  - $7F28
210  - $05CF     850  - $63D5
220  - $CD8B     860  - $0249
230  - $9B5A     870  - $F34E
240  - $5C8C     880  - $CE74
250  - $5D89     890  - $615C
260  - $A3A6     900  - $9139
270  - $11EA     910  - $1457
280  - $D6E6     920  - $691F
290  - $15A8     930  - $E494
300  - $1387     940  - $8B2A
310  - $BB13     950  - $0ED6
320  - $3797     960  - $B160
330  - $B5A4     970  - $8DA7
340  - $2B6C     980  - $B246
350  - $B8AB     990  - $F943
360  - $D732     1000 - $F728
370  - $757D     1010 - $4792
380  - $0213     1020 - $DC4E
390  - $DA00     1030 - $2A01
400  - $A0C7     1040 - $C101
410  - $175F     1050 - $C1F1
420  - $DB1E     1060 - $AF1F
430  - $112A     1070 - $FE47
440  - $0B14     1080 - $47C6
450  - $A119     1090 - $EC31
460  - $D6A1     1100 - $C5BF
470  - $7DD3     1110 - $C60E
480  - $500D     1120 - $4287
490  - $02CF     1130 - $965C
500  - $5465     1140 - $1A51
510  - $1995     1150 - $1503
520  - $7621     1160 - $BBC7
530  - $764F     1170 - $F369
540  - $59FB     1180 - $AAF9
550  - $1526     1190 - $3C5C
560  - $61AE     1200 - $B491
570  - $19C2     1210 - $8D21
580  - $7D80     1220 - $F53F
590  - $B790     1230 - $6295
600  - $1FFA     1240 - $6249
610  - $F383     1250 - $68F3
620  - $DE35     1260 - $B9FF
630  - $7554     1270 - $FFD8
640  - $8454     1280 - $F9EE
```

Boulder Dash by Microfun:

A Review

By Ray Darrah

MicroLab, Inc.
2699 Skokie Valley Road
Highland Park, IL 60035
$40.00

Hardware Required:
Apple][,][Plus, //e or compatible
One Disk Drive
Joystick

The folks at Microfun have once again designed a completely new game program. Rather than just slightly modifying some game already found in the arcades, Microfun continues to create new programs. Unfortunately, when a new game is designed, it often isn't much fun to play. I am happy to report that this is not the case with Microfun's latest release: Boulder Dash.

This game is comprised of 19 very different screens called caves. In a cave you may find dirt, boulders, jewels, an amoeba, fireflies or butterflies. The object in each cave is gather a specified amount of jewels by running into them. Once this is accomplished, an escape door appears somewhere in the cave. You must then get to it before the time clock expires.

The hi-res screen is a window that shows about one-ninth of the entire cave; consequently, you cannot view the whole of the cave you are in. You may move about in this screen quite rapidly and when you try to go to another part of the cave, the screen scrolls in the correct direction to reveal the new scene.

Rather than you being represented by a blip or creature on the computer screen, your marker has its own identity (Rockford) and you must instruct him (via joystick commands) on how to complete each screen. Rockford is your complete slave and has no will of his own.

Rockford is an expert mole, and has no trouble moving through (and eliminating) dirt. I don't know where it goes when he moves through it, but maybe he digests it. Everything else (except the walls of the cave itself) found in the cave will destroy Rockford if the conditions are right. Jewels and boulders must land on him to kill him, but the amoeba, butterflies, fireflies and enchanted walls carry the touch of death.

Watch For Falling Rocks?

Yes. If the dirt below a boulder or jewel is removed, that item (and the rocks or jewels resting on it) will fall downward until it hits something. This is one of the main obstacles of the game. The boulders fall predictably, but it takes a while for the player to get the feel of exactly what's going to go where when some dirt or jewels are digested.

The program responsible for the falling objects and the scrolling cave is top notch real-time programming. It is almost as if the programmer had timed interrupts at his disposal. But since the Apple has no such luxuries, some advanced programming with hi-res page flipping was used.

After the first few caves, when you are finally almost conditioned to the falling objects, the amoeba, butterflies, fireflies, and enchanted walls are introduced to make things quite a bit harder.

The amoeba is a green blob that will not kill you if you bump into it, but continually multiplies over dirt (but not through rock). If the amoeba gets trapped where it cannot grow anymore, it turns into jewels. Another way of turning the amoeba into jewels is to make the deadly butterflies run into it. Fireflies and butterflies glide about the tunnels in the cave using opposite algorithms, instantly destroying Rockford if they hit him. These can only be destroyed by dropping boulders on them. Fireflies demolish their surroundings when they explode while butterflies turn into jewels when they are destroyed.

The enchanted wall is the worst of the cave nasties. It flies around the tunnels and cannot be destroyed. When a boulder hits an enchanted wall for the first time, it vibrates for a short period of time. While it is vibrating, boulders that drop through it (providing there is an empty space below) are magically turned into jewels.

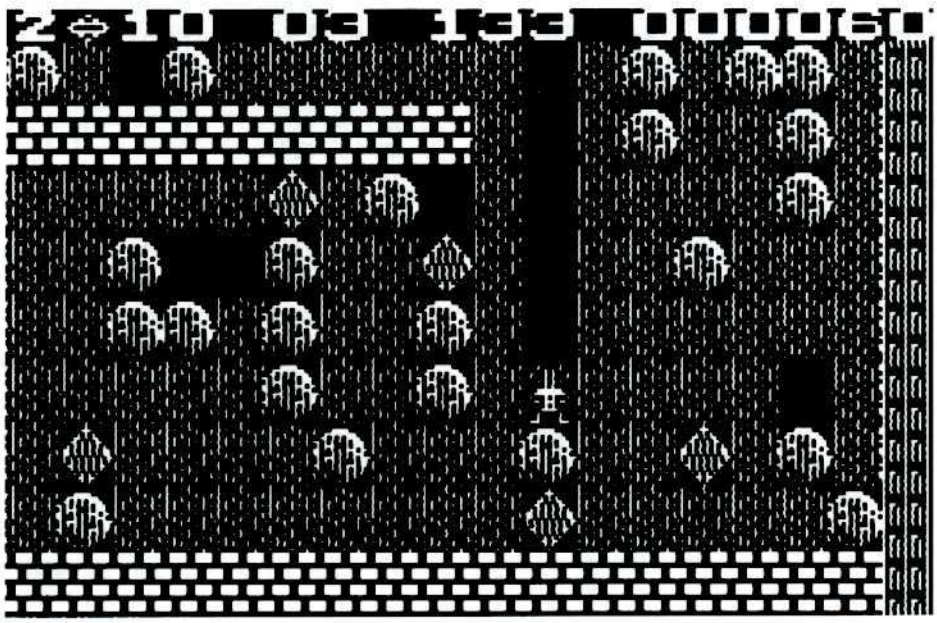
One Quibble

Perhaps the only downfall of Boulder Dash is its difficulty. The 4-1/2 pages of documentation state that every fifth screen is a playable intermission. At a playable intermission there is no penalty for not completing the screen, but an extra Rockford is awarded for successfully completing the cave.

I have played many many games of Boulder Dash and have never viewed one of these playable intermissions. Microfun allows the player to start on cave 1,6,11 or 16. This always puts you four screens away from one of these intermissions. If they didn't let you start on these different caves, you might not ever see more than the first four screens!

Final Comments

I enjoy a good game of Boulder Dash. I think that even its extreme difficulty is acceptable because the player doesn't ever have to deposit a quarter into his or her Apple. Overall, this game rates a B+ and would surely provide hours of entertainment to the person who doesn't give up easily.



A Fix For DiskEdit

(from the Best of Hardcore Computing)

We have recently discovered that the huge hexdump accompanying the DiskEdit article on page 10 of The Best of Hardcore was almost entirely incorrect. This was caused by a memory conflict between DiskEdit and the program which transmits checksums to our Compugraphic typesetter. We apologize to, and symphathize with those dedicated readers who typed in this hexdump.

Because the hexdump printed in The Best of Hardcore is incorrect from $800 through $B07, you can find a corrected hexdump for that portion of memory below. Since the change in the first part of the hexdump changes the checksums for the rest of it, after location $B07, the new checksums are also listed.

```
0800: 00 11 08 00 00 8C 32 30   $68DE
0808: 36 37 3A AB 31 30 3A B2   $2B6A
0810: 00 00 00 4C 73 08 01 60   $D0FC
0818: 01 00 00 00 27 08 00 09   $1EE0
0820: 00 00 01 00 00 60 01 00   $DE04
0828: 01 EF D8 00 00 00 00 00   $43A8
0830: 01 01 00 00 00 01 00 09   $6EDD
0838: 00 00 8B 0C 00 00 00 4C   $792A
0840: 90 08 4C 29 0A 4C 0E 12   $62DF
0848: 4C 7E 0F 4C 00 0A 4C 6B   $E0AD

0850: 0B 4C C9 0C 4C BD 0C 4C   $4637
0858: 5D 0E 4C AE 0B 4C 03 12   $8C0F
0860: 4C DA 11 4C 0A 0E 60 60   $457A
0868: 60 00 FF 01 00 00 00 01   $1CD9
0870: 10 23 00 20 E3 03 84 48   $8710
0878: 85 49 A0 01 B1 48 8D 17   $9AD8
0880: 08 C8 B1 48 8D 18 08 A9   $2778
0888: 1F 85 67 A9 12 85 68 60   $896E
0890: A9 08 A0 16 20 D9 03 90   $49B9
0898: 06 AD 23 08 8D 2E 08 60   $AAF8

08A0: 00 00 00 00 00 00 00 00   $8A78
08A8: 00 00 00 00 00 00 00 00   $AAF8
08B0: 00 00 00 00 00 00 00 00   $8A78
08B8: 00 00 00 00 00 00 00 00   $AAF8
08C0: 00 00 00 00 00 00 00 00   $8A78
08C8: 00 00 00 00 00 00 00 00   $AAF8
08D0: 00 00 00 00 00 00 00 00   $8A78
08D8: 00 00 00 00 00 00 00 00   $AAF8
08E0: 00 00 00 00 00 00 00 00   $8A78
08E8: 00 00 00 00 00 00 00 00   $AAF8

08F0: 00 00 00 00 00 00 00 00   $8A78
08F8: 00 00 00 00 00 00 00 00   $AAF8
0900: 00 00 00 00 00 00 00 00   $8A78
0908: 00 00 00 00 00 00 00 00   $AAF8
0910: 00 00 00 00 00 00 00 00   $8A78
0918: 00 00 00 00 00 00 00 00   $AAF8
0920: 00 00 00 00 00 00 00 00   $8A78
0928: 00 00 00 00 00 00 00 00   $AAF8
0930: 00 00 00 00 00 00 00 00   $8A78
0938: 00 00 00 00 00 00 00 00   $AAF8
0940: 00 00 00 00 00 00 00 00   $8A78
0948: 00 00 00 00 00 00 00 00   $AAF8
0950: 00 00 00 00 00 00 00 00   $8A78
0958: 00 00 00 00 00 00 00 00   $AAF8
0960: 00 00 00 00 00 00 00 00   $8A78
0968: 00 00 00 00 00 00 00 00   $AAF8
0970: 00 00 00 00 00 00 00 00   $8A78
0978: 00 00 00 00 00 00 00 00   $AAF8
0980: 00 00 00 00 00 00 00 00   $8A78
0988: 00 00 00 00 00 00 00 00   $AAF8

0990: 00 00 00 00 00 00 00 00   $8A78
0998: 00 00 00 00 00 00 00 00   $AAF8
09A0: 00 00 00 00 00 00 00 00   $8A78
09A8: 00 00 00 00 00 00 00 00   $AAF8
09B0: 00 00 00 00 00 00 00 00   $8A78
09B8: 00 00 00 00 00 00 00 00   $AAF8
09C0: 00 00 00 00 00 00 00 00   $8A78
09C8: 00 00 00 00 00 00 00 00   $AAF8
09D0: 00 00 00 00 00 00 00 00   $8A78
09D8: 00 00 00 00 00 00 00 00   $AAF8

09E0: 00 00 00 00 00 00 00 00   $8A78
09E8: 00 00 00 00 00 00 00 00   $AAF8
09F0: 00 00 00 00 00 00 00 00   $8A78
09F8: 00 00 00 00 00 00 00 00   $AAF8
0A00: A9 8D 20 ED FD A5 3A 8D   $79DA
0A08: 2B 08 20 0E 12 A9 04 85   $45AC
0A10: 24 A9 AD 20 ED FD A2 01   $2560
0A18: 20 4A F9 20 8C F8 20 D3   $461D
0A20: F8 20 53 F9 85 3A 84 3B   $D75B
0A28: 60 AD 35 08 0A 0A AA A0   $DEC9

0A30: 00 BD 95 0C 99 E7 00 E8   $2A16
0A38: C8 C0 04 90 F4 A9 00 85   $0EAE
0A40: E4 85 25 20 CF 0A E6 25   $6CF7
0A48: A5 25 C9 14 D0 F5 A5 25   $FD3E
0A50: 20 8F 0A A0 27 A9 A0 91   $5AC2
0A58: 28 88 10 FB 4C C9 0C 00   $0CED
0A60: 04 80 04 00 05 80 05 00   $6629
0A68: 06 80 06 00 07 80 07 28   $1411
0A70: 04 A8 04 28 05 A8 05 28   $DE15
0A78: 06 A8 06 28 07 A8 07 50   $5CC5

0A80: 04 D0 04 50 05 D0 05 50   $7681
0A88: 06 D0 06 50 07 D0 07 0A   $3600
0A90: AA BD 5F 0A 85 28 18 69   $C20B
0A98: 1B 85 26 BD 60 0A 85 29   $9DE2
0AA0: 85 27 60 A2 14 CA BD B8   $3A77
0AA8: 0A CD 34 08 90 05 F0 03   $0A7A
0AB0: 4C A5 0A 86 25 85 E4 60   $F356
0AB8: 00 0D 1A 27 34 41 4E 5B   $AB3E
0AC0: 68 75 82 8F 9C A9 B6 C3   $53C6
0AC8: D0 DD EA F7 20 A3 0A A5   $9520

0AD0: 25 20 8F 0A A9 0D 8D 6C   $81D9
0AD8: 08 A6 E4 BD 00 09 48 EC   $5368
0AE0: 34 08 D0 05 AE 6F 08 F0   $02FF
0AE8: 03 20 B1 0B A0 00 91 26   $EAE0
0AF0: E6 26 68 AE 6F 08 F0 44   $9C6B
0AF8: 48 4A 4A 4A 4A A6 E4 EC   $C880
0B00: 34 08 D0 0B 09 30 C9 3A   $75D4

0B08-$DE13   0D70-$0C3B   0FE0-$70E9
0B10-$C4B7   0D78-$3F3F   0FE8-$A984
0B18-$A6E0   0D80-$1E61   0FF0-$8502
             0D88-$DA03   0FF8-$E767
0B20-$9975   0D90-$D5EB   1000-$FB2F
0B28-$0EFE   0D98-$B658   1008-$100C
0B30-$B69C               1010-$4696
0B38-$D12D   0DA0-$E4F0   1018-$2B9D
0B40-$859D   0DA8-$041F
0B48-$D64D   0DB0-$8EDB   1020-$A2E6
0B50-$28EE   0DB8-$0FA2   1028-$9049
0B58-$AC00   0DC0-$03D7   1030-$CD28
0B60-$BE55   0DC8-$B2D8   1038-$2ADE
0B68-$3987   0DD0-$F4C5   1040-$4567
             0DD8-$B40A   1048-$6286
0B70-$12C9   0DE0-$95EA   1050-$A4DE
0B78-$FE56   0DE8-$2151   1058-$BF0F
0B80-$2822               1060-$CE90
0B88-$42A3   0DF0-$0719   1068-$3C8F
0B90-$586C   0DF8-$88C7
0B98-$7764   0E00-$AED5   1070-$FF81
0BA0-$963A   0E08-$1EBD   1078-$6C3E
0BA8-$5151   0E10-$E3CE   1080-$EFA6
0BB0-$3CB6   0E18-$0A9A   1088-$72A3
0BB8-$3097   0E20-$4EC5   1090-$3128
             0E28-$E4DB   1098-$9ED4
0BC0-$316B   0E30-$1CEF   10A0-$E6A9
0BC8-$11FA   0E38-$56D8   10A8-$5216
0BD0-$6398               10B0-$9FDE
0BD8-$F2D1   0E40-$4FEB   10B8-$9103
0BE0-$6366   0E48-$EA28
0BE8-$629D   0E50-$B9AC   10C0-$3359
0BF0-$420D   0E58-$DE6F   10C8-$59A7
0BF8-$C23D   0E60-$95F4   10D0-$AA0B
0C00-$29E3   0E68-$7786   10D8-$5294
0C08-$DBF5   0E70-$B198   10E0-$FC00
             0E78-$5E7D   10E8-$ABF8
0C10-$A1FF   0E80-$25B5   10F0-$D2ED
0C18-$018F   0E88-$3225   10F8-$CEEF
0C20-$719F               1100-$A28C
0C28-$D1CF   0E90-$7455   1108-$FAB1
0C30-$017F   0E98-$A66A
0C38-$D1CF   0EA0-$47BE   1110-$292E
0C40-$017F   0EA8-$47B3   1118-$0904
0C48-$D1CF   0EB0-$C440   1120-$89BA
0C50-$C15F   0EB8-$CB60   1128-$AB23
0C58-$214F   0EC0-$C37F   1130-$AFAB
             0EC8-$430A   1138-$39F3
0C60-$F1FF   0ED0-$7F8D   1140-$AE90
0C68-$214F   0ED8-$473E   1148-$9BD1
0C70-$F1FF               1150-$5211
0C78-$214F   0EE0-$73F5   1158-$760E
0C80-$F1FF   0EE8-$B11E
0C88-$214F   0EF0-$043C   1160-$33CA
0C90-$69BC   0EF8-$4443   1168-$A35E
0C98-$5D58   0F00-$2535   1170-$F0BB
0CA0-$5E21   0F08-$213C   1178-$F126
0CA8-$9E99   0F10-$4A53   1180-$7899
             0F18-$E450   1188-$D7A2
0CB0-$1EE1   0F20-$2479   1190-$26EA
0CB8-$A5BF   0F28-$EA36   1198-$EDA3
0CC0-$EF0C               11A0-$8F2E
0CC8-$F9E7   0F30-$6648   11A8-$D263
0CD0-$B3C3   0F38-$B9BE
0CD8-$FBA0   0F40-$9DA3   11B0-$6D6D
0CE0-$CA42   0F48-$8568   11B8-$6FC1
0CE8-$9FBA   0F50-$5F35   11C0-$9FE5
0CF0-$31B5   0F58-$5B3B   11C8-$1D96
0CF8-$F436   0F60-$67F7   11D0-$B5B4
             0F68-$1665   11D8-$EE35
0D00-$DCE4   0F70-$0541   11E0-$B602
0D08-$17B8   0F78-$D3E9   11E8-$F0D6
0D10-$DA82               11F0-$ECA7
0D18-$FB3B   0F80-$C4F4   11F8-$6E6C
0D20-$BD6A   0F88-$07BE
0D28-$2AD0   0F90-$F3F2   1200-$B9A3
0D30-$ACD9   0F98-$102A   1208-$ED42
0D38-$D60D   0FA0-$B90C   1210-$0FAE
0D40-$D3F8   0FA8-$D268   1218-$0313
0D48-$A45C   0FB0-$7DF8   1220-$1583
             0FB8-$2D7E
0D50-$E1BE   0FC0-$4329
0D58-$9F9A   0FC8-$E655
0D60-$DD57               0FD0-$71DB
0D68-$A0BA   0FD8-$8960
```

* see No. 16, page 26 - correction for source code - incorrect origin

Tracking Down Rocky's Boots

By Jerry Caldwell

Rocky's Boots
The Learning Company
4370 Alpine Road
Portola Valley, CA 94025
$75.00

Requirements:
Apple][+ or equivalent
Rocky's Boots
Super IOB v1.2
Sector Editor
Blank disk

Rocky's Boots is a very nice little educational program designed to instruct students of any age in the basic concepts of electronic digital logic. The user progresses from learning about electricity to the point where he/she is able to construct "machines" from wires, logic gates, clocks, sensors and other pieces of hardware that will "kick" targets of a specific color and/or shape. Unfortunately, or fortunately for those of us who like a challenge, Rocky's Boots is also copy-protected by its publisher, The Learning Company.

The Learning Company uses two techniques to copy-protect Rocky's Boots. First, every sector on the disk is marked as if it were on track 0. This prevents the use of COPYA or any other standard copier to make a backup. The second copy protection measure involves the use of half-tracks with track arcing during the boot process of the disk. On Rocky's Boots, tracks 3.0, 3.5 and 4.0 each contain five sectors of data.

While the first technique is easy to circumvent with the use of a bit copier, the latter technique is more difficult to defeat, even with a bit copier which can read half-tracks. If full tracks of data are written to a disk in half-track increments, the data on adjacent half-tracks will tend to be obliterated. However, the half-tracks on an original copy of Rocky's Boots are written in a special manner so that they are synchronized with one another with each of the half-tracks containing only five sectors of data. This pattern is generally very hard to duplicate with a bit copier but can be read with the proper software. Thankfully, with Rocky's Boots, both of the copy protection techniques I have described can be removed so that a backup which resides on a normally formatted disk can be produced.

There are two basic things which must be done to produce the backup. First, all of the data that is written to the original must be moved to a backup disk. The data that resides on the three half-tracks on the original disk will be written onto one full track on the backup disk. This transfer can be handled by Super IOB with the Rocky's Boots controller listed on this page installed.

Once Rocky's Boots has been moved to a normally formatted disk, a couple of changes need to be made to the disk so that it will run properly. This involves some changes to the disk I/O code so that no half-tracks will be accessed and I/O errors will not be generated when the program finds that the sectors are marked normally instead of as if they were all on track 0. One other change has to be made so that one can exit from Rocky's Boots without having it crash into the monitor.

Making The Copy

1) Begin by typing in the controller listed below.
2) Install it into Super IOB and

RUN

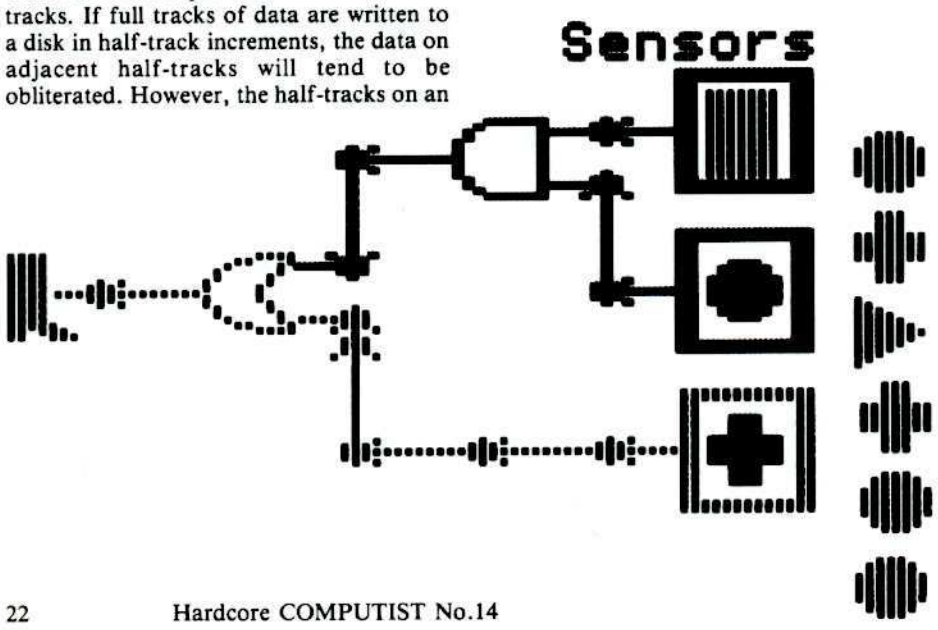
During the copy process, the disk head will recalibrate just before reading the original disk. If it should recalibrate in the middle of reading the original, then you will get a bad copy.

Rocky's Boots Controller

```
1000 REM ROCKYS BOOTS
1010 TK = 0 : ST = 0 : LT = 34 : CD = WR :
     POKE 48573 ,128
1020 GOSUB 490 : T1 = TK : TK = 0 : CD = 0
     : GOSUB 100 : GOSUB 80 : S = - 128 :
     GOSUB 130
1025 CD = RD : S = T1 * 2 : GOSUB 130 :
     RESTORE : GOSUB 170 : S = 2
1030 TK = PH / 2 : GOSUB 430 : TK = 0 :
     GOSUB 100 : ST = ST + 1 : IF ST < DOS
     THEN 1030
1040 IF BF THEN 1060
1050 ST = 0 : GOSUB 130 : IF PH = 6 THEN
     GOSUB 1110
1055 IF PH < LT * 2 THEN 1030
1060 GOSUB 310 : GOSUB 490 : TK = T1
     : ST = 0 : GOSUB 230
1070 GOSUB 430 : GOSUB 100 : ST = ST + 1
     : IF ST < DOS THEN 1070
1080 ST = 0 : TK = TK + 1 + (TK = 3 ) :
     GOSUB 1150 : IF TK = 3 THEN GOSUB
     1160
1085 IF BF = 0 AND TK < LT THEN 1070
1090 IF TK < LT THEN 1020
1100 HOME : PRINT "DONE^WITH^COPY" :
     END
1110 S = 1 : ST = 1 : GOSUB 1160 : GOSUB
     100 : ST = 0
1120 FOR A1 = 1 TO 5 : TK = 3 : ST = ST + 1
     : GOSUB 430 : TK = 0 : GOSUB 100 :
     NEXT
1130 GOSUB 130 : IF PH < 9 THEN 1120
1140 GOSUB 130 : S = 2 : ST = 0
1150 POKE 48683 ,185 : POKE 48684
     ,184 : POKE 48685 ,191 : RETURN
1160 POKE 48683 ,234 : POKE 48684
     ,234 : POKE 48685 ,234 : RETURN
5000 DATA 255 ,255 ,255 ,255
5010 DATA 1^CHANGES
5020 DATA 0 ,7 ,43 ,231
```

Controller Checksums

```
1000  - $356B      1090  - $C3A1
1010  - $B367      1100  - $163B
1020  - $3969      1110  - $8529
1025  - $78D8      1120  - $05A2
1030  - $6B11      1130  - $56FC
1040  - $0F9A      1140  - $961A
1050  - $44B6      1150  - $9410
1055  - $6AEF      1160  - $858D
1060  - $C5F8      5000  - $5B07
1070  - $CDFD      5010  - $F445
1080  - $C3C3      5020  - $77C6
1085  - $5B52
```



A Few Controller Words

Here is a list of the differences in this controller from the standard one which makes it successful.

1010 - Set the last track to be copied at 33, set extended error retry mode.
1020 - Tell DOS to position over track 0, recalibrate.
1025 - Fix command code, move to correct track via MOVE S PHASES, alter the ending marks (to FF FF, FF FF), fix step variable for whole tracks.
1030 - Calculate TK so that PRINT TRACK & SECTOR# will display the correct track, restore TK to zero since that is what the sectors are marked as.

1050 - If on track 3 then call the track arcing reader at 1110.

1060 - Call the sector editor to edit track 0, sector 7, byte $2B to $E7, normalize the DOS ending marks.

1080 - Skip track 4, tell DOS to write logical sectors unless on track three in which case, tell DOS to write physical sectors.

1110 - Set the step variable for half-tracks, tell DOS to read physical sectors, read sector 1 as a dummy, fix sector variable.

1120 - Read the next five physical sectors.

1130 - Step forward a half-track, if not done with the track arcing, then read five more sectors.

1140 - Move on to track five, fix step variable, start with sector zero of track five.

1150 - Restore DOS so that it reads logical sectors.

1160 - Alter DOS so that it reads physical sectors.

Once the Super IOB copy of Rocky's Boots has been made, there are a few changes that have to be made to the disk so that it will function properly on a normally formatted disk.

Rocky's Boot

During the boot-up of the disk, some code is written to page $04 (on the text page) that is responsible for accessing the data on the half-tracks. After this has been accomplished, the code exits to $500 to where the menu is read in. Putting code on the text page like this is also a form of copy-protection.

An interesting fact about Rocky's Boots is that it always uses the sector read routine at $Cx5C (x being the slot number) to read the disk. That is, the original Rocky's Boots uses this ROM subroutine. The modification we will make moves this routine into RAM where it can be modified to suit our purposes.

This change modifies the code that is written to page $04 so that it does not access the half-tracks, but instead, reads the data that was written on tracks 3.0, 3.5 and 4.0 all from track $03. After this has been done, the code will move the sector read routine from the disk controller card to $400-$49E and then modify it so that the normally marked sectors will not cause any errors. Our code will also make a modification at $508-$50A so that the code which was moved to $400-$49E will be used instead of the routine in the disk controller's ROM. This change has to be made to track $00, sector $07 (logical) on the Super IOB copy of Rocky's Boots.

To make this change, get out your sector editor and read in track $00, sector $07 of the backup. Move the cursor to byte $AE of this sector and start entering the bytes listed in the hexdump below. Write the sector back to the disk when you have finished entering all of the bytes. For those who are interested, the source listing of this code can be seen on this page.

```
A2 0F 20 0C 04 20 2A 04
46 4A A9 A4 85 76 A9 03
85 77 A9 00 85 78 A5 3F
85 79 A0 5C B1 78 91 76
C8 D0 F9 A9 41 8D 43 04
A9 04 8D 0A 05 A9 00 8D
09 05 A6 2B 86 EF 4C 00
05 E6 41 20 2D 04 60
```

A Graceful Exit

The final modification is made to track $1, sector $07 of the backup so that the drive will reboot properly when the "END" option is chosen from the main menu. This modification is necessary because with the previous sector edit we tricked the program into thinking the disk controller ROM was located on the text page (page $04). The program will just crash if it tries to reboot from there. So, to restore a graceful exit to Rocky's Boots, make the following change to the Super IOB copy.

Track	Sector	Byte	From	To
$01	$07	$03	$AC	$A4
$01	$07	$04	$0A	$3F
$01	$07	$05	$05	$EA

Once you have written the change back to the disk, you will have a fully functional backup of Rocky's Boots.

Those of you who would like to learn how Rocky's Boots works might be interested to know that each of the items on the menu (Rocky's Boots, Rocky's Challenge, etc) has an entry point of $A00. If you have some means of halting without a reboot (old F8 ROM or a NMI card, etc.) you can stop the program, snoop around memory to your heart's content and then restart the module in memory with a A00G. Just keep in mind that the code necessary for disk access resides on the text page and it will hit the proverbial bit bucket as soon as the monitor is entered. Once the program has been halted, you will have to reboot the disk in order use the main menu.

Other Learning Company Programs

Other Learning Company programs are copy-protected in a similar fashion to what I have described for Rocky's Boots. There are several ways to approach these programs.

One of the easiest approaches is to make a bit copy of the original and boot it with the cover of the disk drive removed. The head will probably move inward until it tries to read from a track that should have adjacent half-tracks. You may then replace the copy with the original, close the door briefly and then re-open it. As the Apple reads the data in, you can determine which are the half-tracks. By alternating between the bit copy and the original copy, you can

Continued on page 26

Source Code

```
*
*
* PATCH THIS CODE TO TRACK $0, SECTOR $7 OF THE IOB COPY OF ROCKY'S BOOT. THE CODE
* WILL MOVE THE DISK CONTROLLER ROM SECTOR READ ROUTINE ONTO THE TEXT PAGE WHERE IT
* WILL BE USED TO READ THE NORMALIZED COPY OF ROCKY'S BOOTS.
*
*
              .OR $4EA
              .TA $800
              .TF ROCKY CODE
*
*
* 04AE
04EA: A2 0F             LDX #$0F      START ON SECTOR $F
04EC: 20 0C 04          JSR $040C     READ SUBROUTINE
04EF: 20 2A 04          JSR $042A     MOVE A FULL TRACK
04F2: 46 4A             LSR $4A       RESET TO LOGICAL SECTORING
04F4: A9 A4             LDA #$A4      SET UP THE ZERO
04F6: 85 76             STA $76       PAGE POINTERS
04F8: A9 03             LDA #$03      TO MOVE THE ROM
04FA: 85 77             STA $77       SECTOR READ ROUTINE
04FC: A9 00             LDA #$00      AT $CX5C TO $400
04FE: 85 78             STA $78
0500: A5 3F             LDA $3F       $3F HOLDS HI BYTE OF
0502: 85 79             STA $79       CONTROLLER'S ROM ADDRESS
0504: A0 5C             LDY #$5C      OFFSET INTO ROM
0506: B1 78    MOVEIT   LDA ($78),Y   MOVE THE CODE
0508: 91 76             STA ($76),Y                          Note: Some versions
050A: C8                INY                                  may require the
050B: D0 F9             BNE MOVEIT                           change below
050D: A9 41             LDA #$41      MAKE A MOD TO PREVENT
050F: 8D 43 04          STA $0443     I/O ERRORS
0512: A9 04             LDA #$04      MODIFY THE CODE ON
0514: 8D 0A 05          STA $050A     PAGE $5 SO THAT          STA $507
0517: A9 00             LDA #$00      THE CODE AT $400         LDA #$00
0519: 8D 09 05          STA $0509     IS USED TO READ DISK     STA $506
051C: A6 2B             LDX $2B       $2B HOLDS SLOT #
051E: 86 EF             STX $EF
0520: 4C 00 05          JMP $0500     EXIT TO $500
0523: E6 41             INC $41       PATCH THAT INCREMENTS
0525: 20 2D 04          JSR $042D     THE TRACK #
0528: 60                RTS
```

Continued from page 14

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

Unique Variables

The following variables are used by the controller exclusively. Other variables used by the controller are for interaction with the various subroutines in Super IOB and; therefore, fit the requirements stated in Table 1.

LT - this variable holds the last track to be accessed (it is the last track, plus one). For example, if line 1010 were to have a LT=15 (instead of LT=35) then only tracks 0-14 would be copied.

T1 - holds the track number (TK) for the transition of read to write and vice versa

Line explanation

1000 - identifies the controller.

1010 - initializes variables.
- TK=0 - sets the starting track to zero.
- ST=0 - sets the starting sector to zero.
- LT=35 - sets the last track to be copied to 34.
- CD=WR - sets the command code to write.

1020 - the read routine. It begins by saving the current track number and then gets the source disk.

1030 - prints the current track and sector, reads the sector and increments the sector number. If it is less than DOS (in this case, 16) then read another sector.

1040 - if the sector buffer is full, go to the write routine.

1050 - resets the sector number to zero and increments the track number. If it is not past the last track, it reads the new one.

1060 - this is the beginning of the write routine. It gets the destination drive and starts at the previously saved track (T1), sector zero.

1070 - Prints the current track and sector, writes the sector to the disk and increments the sector number. If it is not finished with this track, it writes another sector.

1080 - resets the sector number and increments the track number. If the sector buffer isn't empty and it's not past the last track, it writes the new track.

1090 - if it is not finished copying the disk (not past last track), it reads some more tracks.

1100 - tells the user that everything is O.K. and that the disk is copied. It then ENDs.

Saving the Controller

Even though this controller only copies normal DOS 3.3 disks, I recommend saving it anyway. This controller is the basic (pun intended) building block for more complex controllers. You may use either The Controller Saver from Hardcore COMPUTIST No. 10 or CSaver from Hardcore COMPUTIST No. 13.

You now have the capability (I'm sure you did before) to copy an unprotected DOS 3.3 diskette.

The Swap Controller

The other controller used very frequently by Hardcore COMPUTIST is the Swap Controller. It is the one that reads using a foreign RWTS and then writes using the normal one. This is a fine example of what a few changes to the standard controller can do.

```
1000 REM SWAP CONTROLLER 1010 TK=3
     :ST=0:LT=35:CD=WR
1020 T1=TK:GOSUB 490:GOSUB 360:
     ONERR GOTO 550
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 490:TK=T1:ST=0:GOSUB
     360
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 "EVERYTHING^O.K.^
     NO^DOS^ON^COPY":END
10010 IF PEEK(6400)<>162 THEN
     PRINT CHR$(4)"BLOAD^
     RWTS,A$1900"
```

Closing Notes

Most of the old controllers printed in Hardcore COMPUTIST will work with Super IOB Version 1.2. Hence forth, all controllers printed in Hardcore COMPUTIST will work on Super IOB v1.2 and may not work on the original version.

Go out there and break some disks!

Super IOB

```
10 REM ************************
20 REM **     SUPER IOB 1.2    **
30 REM **     BY RAY DARRAH    **
40 REM ************************
50 REM SET HIMEM BELOW BUFFER AND
   XSET LOMEM ABOVE THE BLOADED
   XRWTS
60 LOMEM: 8448:HIMEM: 9983:GOTO
   10010
70 REM INITIAL IOB SETUP
80 POKE BUF,39:POKE DRV,DV:POKE
   VOL,VL:POKE SLT,SO*16:
   RETURN
90 REM R/W SECTOR
100 BF=0:POKE TRK,TK:POKE SCT
    ,ST:POKE CMD,CD:CALL IO:POKE
    BUF,PEEK(BUF)+1:IF PEEK
    (BUF)=>MB THEN BF=1
110 RETURN
120 REM MOVE S PHASES
130 POKE 49289+SO*16+DV,0:POKE
    49289+SO*16,0:A=PH-INT(PH
    /4)*4:POKE 1144,128+A:
    POKE 811,128+S+A:POKE 813
    ,SO*16:CALL 810:POKE 49288+
    SO*16,0:PH=PH+S:IF PH<0
    THEN PH=0
140 RETURN
150 REM
    16 SECTOR RWTS ALTERATIONS
    XX
160 REM ALTERED ENDING MARKS
170 READ A1,A2,A3,A4:POKE 47505
    ,A1:POKE 47515,A2:POKE 47413
    ,A3:POKE 47423,A4:RETURN
180 REM ALTERED ADDRESS MARKS
190 READ A1,A2,A3:POKE 47445,A1:
    POKE 47455,A2:POKE 47466,A3:
    RETURN
200 REM ALTERED DATA MARKS
210 READ A1,A2,A3:POKE 47335,A1:
    POKE 47345,A2:POKE 47356,A3:
    RETURN
220 REM NORMALIZER
230 POKE 47505,222:POKE 47515,170
    :POKE 47413,222:POKE 47423
    ,170
240 POKE 47445,213:POKE 47455,170
    :POKE 47466,150:POKE 47335
    ,213
250 POKE 47345,170:POKE 47356,173
    :POKE 47360,0:POKE 47498,183
    :RETURN
260 REM IGNORE ADDRESS CHECKSUM
270 POKE 47498,0:RETURN
280 REM ALTERED DATA CHECKSUM
290 READ A1:POKE 47360,A1:RETURN
300 REM THE SECTOR EDITOR
310 READ A$:IF RIGHT$(A$,7)<>
    "CHANGES" THEN 310
320 FOR A=1 TO VAL(A$):READ A1,A2
    ,A3,A4
330 IF A1<T1 OR A1>TK THEN NEXT:
    RETURN
340 POKE 9984+(A1-T1)*4096+A2*
    256+A3,A4:NEXT:RETURN

350 REM SWAP RWTS AT $1900 WITH T-
    HE ONE AT $B800
360 POKE 253,25:POKE 255,184:
    POKE 224,8:CALL 832:RETURN
370 REM FORMAT DISK
380 A$="VOLUME^NUMBER^FOR^
    COPY^=>254":HOME:GOSUB 450:
    HTAB 32:INPUT "";VL$:VL=VAL
    (VL$):IF VL$=""THEN VL=254
390 IF VL>255 OR VL<0 THEN 380
```

```
400 POKE CMD,INIT:SO=S2:DV=D2
    :A$="INSERT^BLANK^DISK^IN^
    SLOT^"+STR$(S2)+",^DRIVE^"
    +STR$(D2):GOSUB 470
410 GOSUB 80:HOME:A$=
    "FORMATTING":FLASH:GOSUB 450
    :NORMAL:CALL IO:VL=0:RETURN
420 REM PRINT TRACK & SECTOR#
430 VTAB 3:HTAB 10:PRINT "TRACK^$"
    MID$(HX$,TK*2+1,2)"^^
    SECTOR^$"MID$(HX$,ST*2+1,2
    )"^^":RETURN
440 REM CENTER MESSAGE
450 HTAB 21-LEN(A$)/2:PRINT A$;
    :RETURN
460 REM PRINT MESSAGE AND WAIT
470 HOME:VTAB 11:GOSUB 450:VTAB
    13:A$="PRESS^ANY^KEY^TO^
    CONTINUE":GOSUB 450:WAIT-
    16384,128:GET A$:RETURN
480 REM TOGGLE READ/WRITE
490 CD=(CD=1)+1:IF CD=RD THEN
    A$="INSERT^SOURCE^DISK.":SO
    =S1:DV=D1:GOTO 510
500 A$="INSERT^TARGET^DISK.":SO=
    S2:DV=D2
510 IF D1=D2 AND S1=S2 THEN GOSUB
    470:HOME
520 VTAB 1:HTAB 1:PRINT SPC(39);
    :FLASH:A$="READING":IF CD=
    WR THEN A$="WRITING"
530 GOSUB 450:NORMAL:GOTO 80
540 REM ONERR IGNORE UNREADABLE
    SECTORS
550 CALL 822:ERR=PEEK(222)
560 IF ERR=255 OR ERR=254 OR CD<>
    RD THEN 10230
570 IF ERR>15 THEN POKE 216,0:
    RESUME
580 PRINT CHR$(7);:POKE BUF,PEEK
    (BUF)+1:IF PEEK(BUF)=>MB
    THEN BF=1
590 RETURN
10000 REM CONFIGURATION TIME
10010 REM BLOAD RWTS HERE
10020 IF PEEK(768)*PEEK(769)=
    507 THEN 10060
10030 HOME:A$="*^SUPER^IOB^*":
    GOSUB 450:PRINT:PRINT:A$=
    "CREATED^BY^RAY^DARRAH":
    GOSUB 450
10040 VTAB 10:A$="INSERT^SUPER^
    IOB^DISK":GOSUB 450:PRINT:
    PRINT:PRINT:A$="PRESS^ANY^
    KEY^TO^CONTINUE":GOSUB 450:
    WAIT-16384,128:GET A$
10050 PRINT:PRINT CHR$(4)"BLOAD^
    IOB.OBJ0,A$300"
10060 TK=ST=VL=CD=DV=SO:RD=1
    :WR=2:INIT=4:ONERR GOTO
    10220
10070 IO=768:SLT=779:DRV=780
    :VOL=781:TRK=782:SCT=783
    :BUF=787:CMD=790:OVL=792
10080 HOME:DOS=16:MB=151:HX$=
    "000102030405060708090A0B0C0
    D0E0F101112131415161718191A1
    B1C1D1E1F202122"
10090 VTAB 8:PRINT:A$="ORIGINAL"
    :S2=6:D2=1:GOSUB 10140:S1=
    S2:D1=D2
10100 PRINT:PRINT:PRINT:D2=(D2
    =1)+1:A$="DUPLICATE":GOSUB
    10140
10110 A$="FORMAT^BACK^UP^FIRST?^
    N"+CHR$(8):HOME:VTAB 12:
    GOSUB 450:GET A$:IF A$="Y"
    THEN GOSUB 380
10120 HOME:A$="INSERT^DISKS^IN^
    PROPER^DRIVES.":GOSUB 470:
    HOME:GOTO 1000
10130 REM GET SLOT AND DRIVE#
10140 GOSUB 450:PRINT:PRINT:
    PRINT TAB(10)"SLOT=>"S2 SPC(
    8)"DRIVE=>"D2;
10150 HTAB 16:B$="7":GOSUB 10180
    :S2=VAL(A$)
10160 HTAB 32:B$="2":GOSUB 10180
    :D2=VAL(A$):RETURN
10170 REM GET A KEY
10180 GET A$:IF(A$<"1" OR A$>B$)
    AND A$<>CHR$(13) THEN 10180
10190 IF A$=CHR$(13) THEN A$=CHR$
    (PEEK(PEEK(40)+PEEK(41)*
    256+PEEK(36))-128)
10200 PRINT A$;:RETURN
10210 REM DISK ERROR
10220 ERR=PEEK(222):IF ERR>15
    AND ERR<254 THEN POKE 216,0:
    CALL 822:RESUME
10230 IF ERR=254 THEN PRINT "TYPE^
    AGAIN^PLEASE:":PRINT:RESUME
10240 IF ERR=255 THEN STOP
10250 IF ERR=0 THEN A$=
    "INITIALIZATION^ERROR"
10260 IF ERR=1 THEN A$="WRITE^
    PROTECTED"
10270 IF ERR=2 THEN A$="VOLUME^
    MISMATCH^ERROR"
```

```
*---------------------------------------------------------------
* Super IOB.OBJ0
*
* BY RAY DARRAH
*---------------------------------------------------------------

03D9- RWTS.B800 .EQ $03D9    ENTRY POINT TO RWTS @$B800
D412- INVOKERROR .EQ $D412   ROUTINE THAT CASES BASIC TO DO THE ERROR CONTAINED IN X
1E00- RWTS.1900 .EQ $1E00    ENTRY POINT TO THE RWTS AT $1900
B9A0- SEEKABS   .EQ $B9A0    ENTRY POINT TO THE SEEKABS ROUTINE AT $B800
00DE- BAS.ERR   .EQ 222      ;BASIC ON ERR ERROR CODE
00FC- SWFRM     .EQ $FC      ;EXCHANGE FROM PARAMTER
00FE- SWTO      .EQ $FE      ;EXCHANGE RWTS 'TO' PARAMETER
00E0- PAGES     .EQ $E0      ;NUMBER OF PAGES OF MEMORY TO EXCHANGE
                .OR $0300    STARTS AT PAGE THREE
                .TF IOB.OBJ0

*---------------------------------------------------------------
*    CALL RWTS               *
*---------------------------------------------------------------

0300: A9 03     IO       LDA /TABLETYP ENTRY POINT FOR CALLING THE RWTS THROUGH BAS
                                       IC
0302: A0 0A              LDY #TABLETYP A,Y POINT TO THE IOB TABLE
0304: 20 D9 03           JSR RWTS.B800 GO TO THE RWTS AT $B800
0307: B0 16              BCS DOS.ERR   IF THE CARRY SET THEN CAUSE BASIC ERROR
0309: 60                 RTS           OTHERWISE, ALL IS WELL SO RETURN
030A: 01        TABLETYP .HS 01        TYPE OF TABLE (1=IOB)
030B: 60        SLT      .HS 60        SLOT TO BE ACCESSED NEXT (VIA POKESLT,SO)
030C: 01        DRV      .HS 01        DRIVE TO BE ACCESSED NEXT (1 OR 2)
030D: 00        VOL      .HS 00        VLUME TO BE ACCESSED (0=ANYTHING WILL DO)
030E: 00        TRK      .HS 00        TRACK TO ACCESS
030F: 00        SCT      .HS 00        SECTOR TO ACCESS
0310: 1B 03     DCTPTR   .DA DCT       POINTER TO THE DEVICE CHARACTERISTICS TABLE
0312: 00        BUFFERLO .HS 00        ALWAYS MAKE LSB OF BUFFER POINTER ZERO!
0313: 27        BUF      .HS 27        SECTOR BUFFER PAGE POINTER
0314: 00        NOTHING  .HS 00        NOT USED
0315: 00        BYTCOUNT .HS 00        BYTE COUNT FOR PARTIAL SECTOR (0=256 BYTES)
0316: 00        CMD      .HS 00        COMMAND CODE (0=SEEK)
0317: 00        RWTS.ERR .HS 00        ERROR CODE THAT THE RWTS.B800 RETURNS WITH
0318: 00        OVL      .HS 00        VOLUME NUMBER OF LAST ACCESSED DISK
0319: 60        OLDSLT   .HS 60        SLOT PREVIOUSLY ACCESSED
031A: 01        OLDDRV   .HS 01        DRIVE PREVIOULSY ACCESSED
031B: 00        DCT      .HS 00        DEVICE TYPE OF DEVICE CHARACTERISTICS TABLE
031C: 01        PHASES   .HS 01        PHASES-1 PER TRACK, (0 OR 1)
031D: EF D8     MOTORCNT .HS EFD8      MOTOR-ON TIME COUNT
031F: AD 17 03  DOS.ERR  LDA RWTS.ERR DOS HAS HAD AN ERROR, GET THE ERROR CODE
0322: 4A                 LSR           DIVIDE IT BY 16
0323: 4A                 LSR
0324: 4A                 LSR
0325: 4A                 LSR
0326: AA                 TAX           TRANSFER IT TO X SO BASIC WLL INDUCE THE FALS
                                       E ERROR CODE
0327: 4C 12 D4           JMP INVOKERROR CAUSE A BASIC ERROR

*---------------------------------------------------------------
*    MOVE THE DISK ARM    *
*---------------------------------------------------------------
```

Continued on page 26

```
032A: A9 00     MOVPHASES LDA #$00    ROUTINE TO SET UP THE REGISTERS BEFORE CALLIN
                                      G SEEKABS
032C: A2 00               LDX #$00    X AND A HAVE DUMMY NUMBERS THAT WILL BE POKED
                                       INTO BY "MOVE S PHASES"
032E: 4C A0 B9            JMP SEEKABS

*-------------------------------------------------------------------
*   CAUSE ERROR IN CONTROLLER   *
*-------------------------------------------------------------------

0331: A6 DE     BASICERR LDX BAS.ERR  BASIC HAS MADE AN ERROR SO CAUSE THE ERROR NU
                                      MBER AT 222
0333: 4C 12 D4           JMP INVOKERROR
*-------------------------------------------------------------------
*    POP OFF RETURN   *
*-------------------------------------------------------------------
0336: 68        POP      PLA          ROUTINE TO POP OFF ONE RETURN (BASIC) ADDRESS
0337: A8                 TAY
0338: 68                 PLA
0339: A6 DF              LDX BAS.ERR+1 GET WHAT THE STACK WOULD BE IF THE GOSUB WAS
                                      N'T THERE
033B: 9A                 TXS          PUT THAT AS THE STACK POINTER
033C: 48                 PHA
033D: 98                 TYA          RESTORE THE LAST RETURN ADDRESS
033E: 48                 PHA
033F: 60                 RTS

*-------------------------------------------------------------------
*   EXCHANGE RWTS's   *
*-------------------------------------------------------------------

0340: A0 00              LDY #0       ;ZERO THE LSB's
0342: 84 FC              STY SWFRM    ;AND HAVE Y AT ZERO FOR START
0344: 84 FE              STY SWTO
0346: B1 FC     MOVE.PAGE LDA (SWFRM),Y ;GET A BYTE
0348: 48                 PHA          ;AND SAVE IT
0349: B1 FE              LDA (SWTO),Y ;GET THE BYTE WHERE THE SAVED ONE GOES
034B: 91 FC              STA (SWFRM),Y ;AND STORE IT WHERE THE SAVED ONE WAS
034D: 68                 PLA          ;GET THE SAVED BYTE
034E: 91 FE              STA (SWTO),Y ;AND STORE IT WHERE IT GOES
0350: C8                 INY          ;DONE WITH A PAGE
0351: D0 F3              BNE MOVE.PAGE ;NO KEEP WORKING ON IT
0353: E6 FD              INC SWFRM+1  ;GET NEXT MSB's
0355: E6 FF              INC SWTO+1
0357: C6 E0              DEC PAGES    ;DECREMENT THE NUMBER OF PAGES TO MOVE
0359: D0 EB              BNE MOVE.PAGE ;IF NOT DONE, MOVE ANOTHER PAGE
035B: 60                 RTS          ;FINISHED, RTS
```

Continued from page 23

determine where the half-tracks end and when full tracks are again being accessed. Once the half-tracks have been read, the bit copy will probably function just as well as the original.

The sector responsible for reading in the half-tracks probably will be sector $01 of track $00 on other Learning Company disks, just as it is on Rocky's Boots. You should be able to read this sector with the Inspector or Tricky Dick (set the end of address and data marks to 000000). Look for instructions like

```
B9 81 C0 LDA C081,Y
B9 80 C0 LDA C080,Y
```

These are instructions which control the head movement of the drive. Look also for instructions which call subroutine which are displaced three steps away in memory, such as

```
800:20 03 08 JSR $0803
803:48       PHA
804:98       TYA
805:48       PHA
```

If a call to $803 causes the head to increment a half-track, then a call to $800 will cause two half-track increments, in other words, a whole track (for an example of this, look at the code which starts at byte $2A of track $0, sector $7 on Rocky's Boots). Having found the appropriate sector, one still needs to recover the data from the half-track arcs. Knowing the location in memory to which the data goes, an Integer card could be used (as it could have in our example here) to recover that data. You would need to dissasemble the sector to trace that information. With other programs, it may not be easy to find a place in the RAM memory to place the disk controller ROM as we were able to do with Rocky's Boots. In such a case, you should plan on only modifying track 0 and the half-track arcs, and backup the remainder of the disk with a bit copier.

```
10280 IF ERR = 4 THEN A$ =
    "DRIVE^ERROR"
10290 IF ERR = 8 THEN A$ =
    "READ^ERROR"
10300 VTAB 12 : GOSUB 450 : PRINT CHR$
    (7) : END
```

IOB.OBJ0

```
0300: A9 03 A0 0A 20 D9 03 B0    $BD35
0308: 16 60 01 60 01 00 00 00    $9CF5
0310: 1B 03 00 27 00 00 00 00    $4320
0318: 00 60 01 00 01 EF D8 AD    $55A7
0320: 17 03 4A 4A 4A 4A AA 4C    $B42B
0328: 12 D4 A9 00 A2 00 4C A0    $8038
0330: B9 A6 DE 4C 12 D4 68 A8    $6E1C
0338: 68 A6 DF 9A 48 98 48 60    $FDD9
0340: A0 00 84 FC 84 FE B1 FC    $3777
0348: 48 B1 FE 91 FC 68 91 FE    $AAB9

0350: C8 D0 F3 E6 FD E6 FF C6    $921F
0358: E0 D0 EB 60                $3160
```

Checksums

```
10    - $BADD      460   - $4999
20    - $9813      470   - $C2B5
30    - $4D3B      480   - $41C8
40    - $AD92      490   - $E5B6
50    - $C899      500   - $9DF4
60    - $1FBA      510   - $D183
70    - $0061      520   - $8030
80    - $835F      530   - $DF21
90    - $E171      540   - $9A0C
100   - $AD0E      550   - $FFEA
110   - $57B6      560   - $5993
120   - $8472      570   - $5FAF
130   - $617E      580   - $2537
140   - $0F1F      590   - $4F3E
150   - $F1B3      10000 - $11D9
160   - $C59A      10010 - $B220
170   - $24E7      10020 - $CEAE
180   - $3991      10030 - $66D7
190   - $8D19      10040 - $7B90
200   - $87A6      10050 - $BC4A
210   - $F20A      10060 - $2603
220   - $7021      10070 - $B23B
230   - $B0F4      10080 - $22FC
240   - $B3A7      10090 - $7E0E
250   - $7FFD      10100 - $F226
260   - $51AB      10110 - $9682
270   - $E236      10120 - $5AC9
280   - $2EE2      10130 - $9A33
290   - $CBCA      10140 - $DFA8
300   - $BAE5      10150 - $B954
310   - $16EC      10160 - $0DF1
320   - $4E65      10170 - $5A0C
330   - $7F0F      10180 - $890F
340   - $9024      10190 - $5905
350   - $FCD3      10200 - $237F
360   - $582F      10210 - $FC1A
370   - $2B63      10220 - $5FC8
380   - $4C32      10230 - $4FFC
390   - $7147      10240 - $FF73
400   - $F86C      10250 - $6FFD
410   - $4377      10260 - $1381
420   - $9363      10270 - $4D68
430   - $7F2D      10280 - $F161
440   - $312D      10290 - $1217
450   - $D115      10300 - $664E
```

*See No.15, page 5 - correction

Continued from page 15

```
0A00: 20 58 FC A9 2C A2 0C 20    $AB0F
0A08: 35 0A A9 08 85 25 20 24    $5CED
0A10: FC A9 56 A2 11 20 35 0A    $D05D
0A18: A0 05 84 24 20 0C FD 91    $A2D8
0A20: 28 C9 B9 B0 F7 C9 B1 90    $A919
0A28: F3 E9 B0 8D 4B 0A A9 60    $A7AD
0A30: 85 A9 4C 02 20 85 FE 86    $0036
0A38: FF A0 00 B1 FE F0 05 91    $40CA
0A40: 28 C8 D0 F7 60             $6BBB
```

6) Check your typing against this listing

```
0A00-   20 58 FC    JSR   $FC58
0A03-   A9 2C       LDA   #$2C
0A05-   A2 0C       LDX   #$0C
0A07-   20 35 0A    JSR   $0A35
0A0A-   A9 08       LDA   #$08
0A0C-   85 25       STA   $25
0A0E-   20 24 FC    JSR   $FC24
0A11-   A9 56       LDA   #$56
0A13-   A2 11       LDX   #$11
0A15-   20 35 0A    JSR   $0A35
0A18-   A0 05       LDY   #$05
0A1A-   84 24       STY   $24
0A1C-   20 0C FD    JSR   $FD0C
0A1F-   91 28       STA   ($28),Y
0A21-   C9 B9       CMP   #$B9
0A23-   B0 F7       BCS   $0A1C
0A25-   C9 B1       CMP   #$B1
0A27-   90 F3       BCC   $0A1C
0A29-   E9 B0       SBC   #$B0
0A2B-   8D 4B 0A    STA   $0A4B
0A2E-   A9 60       LDA   #$60
0A30-   85 A9       STA   $A9
0A32-   4C 02 20    JMP   $2002
0A35-   85 FE       STA   $FE
0A37-   86 FF       STX   $FF
0A39-   A0 00       LDY   #$00
0A3B-   B1 FE       LDA   ($FE),Y
0A3D-   F0 05       BEQ   $0A44
0A3F-   91 28       STA   ($28),Y
0A41-   C8          INY
0A42-   D0 F7       BNE   $0A3B
0A44-   60          RTS
```

This code will set up the display on the text screen, get the slot number and place it at $A4B, and store a $60 at $A9 before performing the aforementioned JMP $2002.

7) Make a couple of modifications to the main program so that it works with the hexdump above

2005:20 78 12
2008:AD 43 0B F0 0D 4C 1C 0A
1B13:EA EA EA
B60:60

Now all you have to do is save the file by typing

BSAVE FAST COPY.LS, A$A00,L$2CFB

You should now have a 46-sector, working copy of this great fast copy utility.

It should go without saying, but I will mention it anyway. This procedure should only be used on your own original copy of LS 5.0. Make sure that you have updated to Rev F since earlier versions had bugs in the fast copy program.

As you can see from the above I am certainly not a machine language wizard, but if you have a plan and work at it, you too can have more useful, unprotected software.

Moving Soon?

Let us know your new address **at least 30 days in advance** so you won't miss a single issue of Hardcore COMPUTIST.
Fill out this form, paste your present address label in the space provided, and send it to:

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

That's all there is to it!

My new address is:

Address _____

City _____ St _____ Zip _____

Phone _____

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

Name _____ ID # _____

Address _____

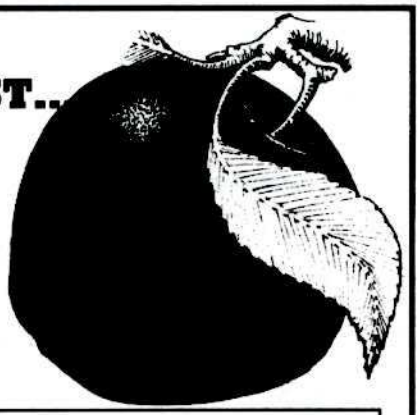
City _____ St _____ Zip _____

Country _____

Phone _____

Hardcore COMPUTIST..

- **Make backups more easily**
- **Move software from floppy to hard disk**
- **Add custom modifications such as fast-DOS to speedup LOADs and SAVEs**



NOT PIRACY

Just Good Sense!

Annual Subscription Rates:

Please check one

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

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

NAME _____

ADDRESS _____

CITY _____ ST _____ ZIP _____

COUNTRY _____

If renewal ID # _____ Phone _____

VISA/MC _____ - _____ - _____ - _____ EXP _____

SIGNATURE _____

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

WHO LIKES THE CIA ?

HERE ARE JUST A FEW OF THE MANY

"an essential part of the Apple-user's repetoire" - APPLE USER

"a valuable buy . . . manual is practically worth having on its own" -WASHINGTON APPLE PI"

"the folks at Golden Delicious should be commended . . . worth waiting for" - HARDCORE COMPUTIST

"multifaceted" - NIBBLE

"the most comprehensive disk accessor I have ever come across" -A.B., VERNON, CANADA

"its ability to unlock other programs will greatly help me" - DR. B.P., SAN FRANCISCO, CALIFORNIA

"an excellent set of programs . . . just great - and good value too" -E.A.S., MILTON KEYNES, ENGLAND

"very, very educational . . . great manual . . . it is FANTASTIC!!!!!!" - J.C., TUCSON, ARIZONA

"a very enlightening piece of software/book . . . top of my list for good buys" - H.S., BLAINE, MINNESOTA

"I like yours the BEST" - R.R., CHICAGO

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

To put the 5 C.I.A. utilities, plus book, on the trail of your Apple II+/IIe (R) 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

CALL 602-820-8819

MEMORY PLUS DISTRIBUTORS
2632 West Baseline
Mesa, Arizona 85202

Five year limited warranty on all products. ALL Orders Add $3.00 Shipping
VISA & M.C. OKAY

Other Apple & IBM products available. Call or write for FREE color brochure.

NEW Tilt & Swivel Diamond Monitor Stand **$34.95**

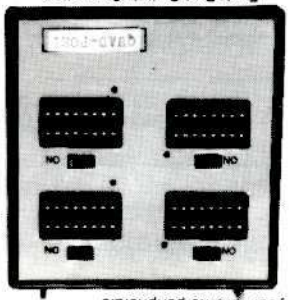
Dealers & Clubs Inquiries Invited!

Machine Gun Joystick
For Apple II and IIe
NEW Slim-Line Design!
DUAL FIRE BUTTONS

TRIM POTS
AUTO CENTERING
ONLY **36.95**
+ SHIPPING

Base Dimensions: 1 1/8" x 3 1/8" x 5 3/4"

Quadport
Multiple Game I/O For Apple II & IIe
NO more plugging and unplugging your favorite peripherals



Each Port Switch Controlled
ONLY **29.95**
+ SHIPPING

DEALERS
TURN YOUR PROFITS.

The Holidays Are ALMOST Here!

Can't decide what to give your friends?

How about a gift subscription to Hardcore COMPUTIST?

They'll receive 12 issues packed with information on deprotection and programs for the Apple][computer found in no other magazine. Your friends will think of you each time they open a new issue of Hardcore COMPUTIST.

ORDER NOW!

Send gift subscription to:

Name_____

Address_____

City _____ St _____ Zip _____

From: _____

Check one: ☐ $25 US ☐ $34 Canada & US 1st Class

Send gift subscription to:

Name_____

Address_____

City _____ St _____ Zip _____

From: _____

Check one: ☐ $25 US ☐ $34 Canada & US 1st Class

Send gift subscription to:

Name_____

Address_____

City _____ St _____ Zip _____

From: _____

Check one: ☐ $25 US ☐ $34 Canada & US 1st Class

For Mexico and Foreign annual subscription rates, refer to form at bottom of page 27.

My Name _____

Address _____

City_____ St_____ Zip_____

Phone _____

VISA/MC _____ Exp _____

Signature_____

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

SIMPLY·SOFTWARE

DISCOUNT SPECIALS ON APPLE SOFTWARE • *ALL PRICES 30% OFF*

LIST $34.95 **SPECIAL $24.47**

Beyond Castle Wolfenstein
Centipede
Donkey Kong
Fortress
Jungle Hunt
Lode Runner
Moon Patrol
Ms. Pacman
Murder by the Dozen
Night Mission Pinball
Quest for Tires
RDF 1985
Story Machine
Transylvania

LIST $39.95 **SPECIAL $27.97**

Algebra I
Battle for Normandy
Broadsides
Cartels & Cuthroats
Comp. Quarterback
Cosmic Balance
Cosmic Balance II
Crypto Cube
Eagles
Face Maker
50 Mission Crush
Galactic Adventure
Galactic Gladiators
Geopolitic 1990
Most Amazing Thing
Knights of the Desert
Mach II Joystick
Master Type
Objective Kursk
Pro. Golf Tour
Rails West
S.E.U.I.S.
Seastalker
Song Writer
Stickybear ABC
Stickybear Shapes
Tigers in the Snow
Type Attack
Warp Factor
Wizards of Words
Zaxxon
Zork I, II or III

LIST $29.95 **SPECIAL $20.97**

Alphabet Zoo
Castle Wolfenstien
Early Games for Y.C.
Kindercomp
Minute Man

LIST $40.00 **SPECIAL $28.00**

Dino Eggs
Heist
Micro Cookbook
Temple of Apshai

LIST $49.95 **SPECIAL $34.97**

Deadline
Enchanter
Flight Simulator II
Homeward Speller
Kraft Joysticks
Mach III Joysticks
Math Blaster!
Planetfall
Print Shop
Questron
Rocky's Boots
Sargon III
Sorcerer
Wizardry
Word Attack

LIST $59.95 **SPECIAL $41.97**

Arcade Machine
Bomb Alley
Carrier Force
Computer Ambush
Computer Bismark
Fighter Command
Germany 1985
Ultima II or III
Guadalcanal Campaign
Millionaire
Napoleon's Campaign
North Atlantic '86
Road to Gettysburg
S.A.M.
Shattered Alliance

LIST $69.95 **SPECIAL $48.97**

Bank Street Speller
Bank Street Writer
Homeward
Speed Reader II

LIST $125.00 **SPECIAL $87.50**

PFS: File
PFS: Graph
PFS: Report
PFS: Write
Sensible Speller

	LIST	SPECIAL
Appointments	$ 75.00	$ 52.50
ASCII Express the Prof.	$129.95	$ 90.97
BPI General Ledger	$395.00	$276.50
DB Master 4.0	$350.00	$245.00
DB Master Utility Pack #1	$129.00	$ 90.30
Death in the Caribbean	$ 35.00	$ 24.50
Dollars & Sense	$100.00	$ 70.00
Home Accountant	$ 74.95	$ 52.47
Jane	$295.00	$206.57
Mastering the SAT	$150.00	$105.00
Magic Window II	$149.95	$104.97
Multiplan	$195.00	$136.50
Snooper Troops #1	$ 44.95	$ 31.47
Study Prog. for SAT	$ 89.95	$ 62.97
Tax Prep. (Howard)	$250.00	$175.00
Telengard	$ 28.00	$ 19.60
Terrapin Logo	$ 99.00	$ 69.97
Think Tank	$150.00	$105.00
TK! Solver	$299.00	$209.30
Typing Tutor II	$ 24.95	$ 17.47
Wordstar	$495.00	$346.50
Visicalc	$250.00	$175.00

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 5 5/8% sales tax. Allow 4-6 weeks for delivery.

By Hackers For Hackers

The BOOT-LEGGER MAGAZINE

- ELITE BOARD DOWNLOADS
- CRACKING TIPS
- PHREAKING SECTION
- GAME CHEATS
- PARMS
- PROGRAMS
- INTERVIEWS

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

Subscribe Now!

Send 25 Bucks for a 1-Year Subscription to:
THE BOOT LEGGER, 3310 Holland Loop Road,
Cave Junction, Oregon 97523

Dealer Inquiries Invited

For information on how to obtain Hardcore COMPUTIST at a substantial savings, write:

Hardcore COMPUTIST, PO Box 110816, Tacoma, WA 98411 OR CALL (206) 581-6038.

BACK UP YOUR DISKS

NOW AVAILABLE AT YOUR LOCAL COMPUTER STORE!

$79.95

EDD runs on Apple II, II plus (including most compatibles), IIe, and IIi (in emulation mode), with one or two 3.3 disk drives.

EDD is the most powerful copy program available for backing up your protected Apple software.

Since EDD has been preset to copy a broad range of copy-protections, many disks can be copied easily, without changing messy parameters. Even though you rarely need to change them, each parameter is fully described in the operating manual. Unlike the copycards, which only copy single load programs, EDD backs up entire disks. Thus, not only copying single load, but, multi disk access programs as well. We feel on an average, EDD can back up many more protected disks than all other copy programs or copycards put together.

ESSENTIAL DATA DUPLICATOR III™

- *Automatically copies most protections.*
- *Rarely needs parameter changing*
- *Average duplication time 2½ minutes*
- *Accurately finds "self-sync" bytes and their lengths*
- *Can copy ¼ and ¾ tracks*
- *Updated program lists available*
- *Unlike copycards, EDD backs up entire disks, not just what's in memory*

To order direct; send **$79.95** plus $2 shipping ($5 foreign), California residents add 6%. Mastercard/Visa accepted. *Prepayment required.*

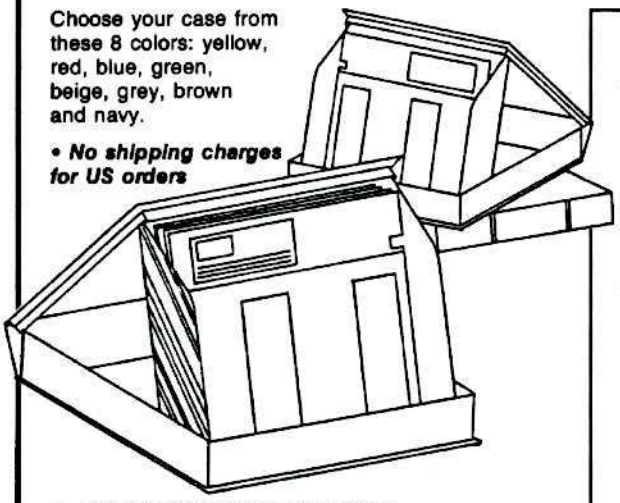
UTILICO MICROWARE 3377 Solano Ave., Suite #352 Napa, CA 94558 (707) 257-2420

Put some COLOR into your computer diskette library

Order 10 diskettes for the low price of $17.00 and get one color coded library case FREE!

Choose your case from these 8 colors: yellow, red, blue, green, beige, grey, brown and navy.

- ***No shipping charges for US orders***



- **10 SS/DD 5 1/4" diskettes**
- **Includes Tyvek sleeves**
- **Hub rings**
- **Guaranteed,** 100% certified, full surface tested

** Matching labels accompany each color coded library case.*

Please send me:

_____ set(s) 10 SS/DD diskettes with 1 FREE color coder library case per set

Number of sets × $17.00 $ _____

Applicable taxes or shipping $ _____

Specify color choices: TOTAL $ _____

___ yellow ___ red ___ blue ___ green

___ beige ___ grey ___ brown ___ navy

Name _____

Address _____

City _____ St _____ Zip _____

Phone _____

VISA/MC _____ Exp _____

Signature _____

Send order to: SoftKey Publishing, PO Box 110816, Tacoma, WA 98411. Washington residents add 7.8% sales tax. Foreign orders add 20% shipping and handling. US funds drawn on US bank. For faster service, send postal money order or certified check.

Know where your head is, at all times, with TRAK STAR constant digital readout

- **Saves copying time**
- **For 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.
+ Compact size permits placement on top of disk drive.
+ Does not use a slot in the Apple® computer.
+ For Apple® II, II+ and //e

Apple is a registered trademark of Apple Computer Inc.

FREE INTRODUCTORY BONUS with purchase of Trak Star

- **Trak Star disk contains patching software.**
- **Simple-to-operate, menu-driven Trak Star software automatically repairs a bad track without requiring technical expertise.**

99.95

Plus $3 shipping and handling charge

Foreign airmail & handling $8.00.

Adapter required for 2-drive systems: $12

Documentation only: $3
Refundable with purchase of Trak Star

Personal checks, M.O.
Visa and Mastercard

Midwest Microsystems

Phone 913 676-7242

9071 Metcalf / Suite 124
Overland Park, KS 66212

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 600 PC name-brand programs and various hardware offers. Programs @ U.S. $8.00/PC includes the disk and registered airmail handling.

.....
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
.....How to make backups using COPY II PLUS PARAMETER
38.....Curing those Auto-Start ROM blue HARDWARE SOLUTIO
46.....A MENU HELLO PROGR
47.....USING BOTH SIDES OF YOUR DISKE
51.....Advanced Playing Techniques, or he get INSIDE CASTLE WOLFEN
derstand String-
EXT INVADER>
uch

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 THEN
included the MOST RECENT and MOST COMPLETE LIST of parameters
for the major bit-copy programs...
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.

Do you need BACK ISSUES?

Are you tired of typing in programs that appear in Hardcore COMPUTIST?

Now, take advantage of our SPECIAL OFFERS!

(Interested in receiving the accompanying Library Disk each month with your magazine subscription? Write for details.)

Please send me the back issues and/or library disks I have checked:

- **Hardcore COMPUTIST 1**.....**$3.50** ☐
- **Hardcore COMPUTIST 4**.....**$3.50** ☐
- **Hardcore COMPUTIST 6***.....**$3.50** ☐
- **Hardcore COMPUTIST 8***.....**$3.50** ☐
- **Hardcore COMPUTIST 9***.....**$3.50** ☐
- **Hardcore COMPUTIST 10***.....**$3.50** ☐
- **Hardcore COMPUTIST 11**.....**$3.50** ☐
- **Hardcore COMPUTIST 12**.....**$3.50** ☐
- **Hardcore COMPUTIST 13**.....**$3.50** ☐
- **CORE 1 Graphics**.....**$3.50** ☐
- **CORE 2 Utilities**.....**$3.50** ☐
- **CORE 3 Games**.....**$3.50** ☐

* **Limited supplies**

SPECIAL OFFERS!

- **CORE 1, 2, 3**.....**$8.50** ☐
- **Hardcore COMPUTIST 1, CORE 1, & Library Disk #1**.....**$14.95** ☐
- **Hardcore COMPUTIST 4, CORE 2, & Library Disk #2**.....**$14.95** ☐
- **Hardcore COMPUTIST 1, 4, 6, 8-13, CORE 1, 2, 3, & Library Disks #1-9**
  (Limited supplies. Orders filled as received.).....**$105.00** ☐
- **Hardcore COMPUTIST 1, 4, 6, 8-13**
  (Limited supplies).....**$26.00** ☐

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.

Order five Library Disks of your choice for only $40.00 ☐

#_____

Library Disk #14.....**$9.95** ☐
Hardcore COMPUTIST 14:
SAT Controller, Sea Dragon Controller, Super IOB v1.2 with Standard Controller and Swap Controller, Batman Decoder Ring, Rocky's Boots Controller.

Library Disk #10.....**$9.95** ☐
Hardcore COMPUTIST 13:
Penguin Controller, Snooper Troops Controller, CSaver, New DOS 3.3 Command Listings, Electronic Arts Controller

Library Disk #9.....**$9.95** ☐
Hardcore COMPUTIST 12:
The Armonitor, Lion's Share Controller, CORE Disk Searcher, Psychedelic Symphony

Library Disk #8.....**$9.95** ☐
Hardcore COMPUTIST 11:
Ultimaker, Ultimapper, Ultima III Controller, Sensible Speller

Library Disk #7.....**$9.95** ☐
Hardcore COMPUTIST 10:
Sensible Speller, Arcade Machine Controller, Controller Saver, ApplEar, Crunchlist II, Minit Man Controller

Library Disk #6.....**$9.95** ☐
Hardcore COMPUTIST 9:
Super IOB and related Controllers, CORE Word Search Generator, Sensible Speller Loader & Saver

Library Disk #5.....**$9.95** ☐
Hardcore COMPUTIST 7:
Corefiler, Disk Directory Designer
Hardcore COMPUTIST 8:
Corefiler Formatter

Library Disk #4.....**$9.95** ☐
Hardcore COMPUTIST 6:
Modified ROMs, Crunchlist, Crucial Code Finder

Library Disk #3.....**$9.95** ☐
CORE Games issue:
Destructive Forces, Dragon Dungeon

Library Disk #2.....**$9.95** ☐
CORE Utilities issue
Hardcore COMPUTIST 4

Library Disk #1.....**$9.95** ☐
CORE Graphics issue
Hardcore COMPUTIST 1

NOTE: Library Disks need accompanying magazine for documentation. Those back issues not listed are no longer available.

DISCOUNTS!

5-1/4 DISKETTES & STORAGE

- SS/DD, BOX OF 10 ..... $15.00 *
- SS/DD, 10 BOXES ..... $139.00 *
- DOUBLE-NOTCHED DS/DD, EACH $1.65
- DOUBLE-NOTCHED DS/DD, 100 ..... $155.00
- HARD PLASTIC STAND-UP 10-DISKETTE LIBRARY **$2.75 EACH**
  CASES ..... **4 FOR $10.00**
  (specify color choices: beige, black, blue, green, grey, red, yellow)
- SMOKED PLASTIC JUMBO-SIZE FLIP-TOP 75 DISKETTE FILE CASES $18.00
- 70 DISKETTE FILE CASES $16.00
- 140 DISKETTE LOCKING WOOD FILE CABINET ..... $33.00 *

PRINTERS

- PANASONIC P1090 DOT MATRIX ..... $259.00 *
- EPSON RX-80 F/T DOT MATRIX ..... $289.00 *
- PROWRITER DOT MATRIX $349.00
- OKI-DATA MICROLINE 92A DOT MATRIX $399.00
- SILVER REED 400 LETTER QUALITY $309.00
- STARWRITER A-10 18CPS LETTER QUALITY ..... $495.00 *
- TOSHIBA 1340 DOT MATRIX AND LETTER QUALITY COMBINED $795.00

PRINTER INTERFACES AND ACCESSORIES

- STANDARD PARALLEL INTERFACE CARD ..... $44.00 *
- GRAPHICS PARALLEL INTERFACE CARD $75.00
- FINGERPRINT PUSH-BUTTON GRAPHICS CARD FOR IMAGEWRITER ..... $114.00 *
- MICROFAZER GENERAL PRINT BUFFER $149.00
- PRINTER STAND $14.00
- SWITCH BOX 3 PARALLEL PORTS ..... $129.00 *
- SWITCH BOX, 3 SERIAL PORTS ..... $79.00 *

FLOPPY DISK DRIVES

- FOURTH DIMENSION (FULL OR SLIMLINE ..... $179.00 *
- ADAMA TECH ..... $169.00 *
- DISTAR SLIMLINE ..... $159.00 *
- LASER SLIMLINE ..... $154.00 *
- DISK CONTROLLER $59.00
- DOUBLE-SIDED DRIVE ..... $249.00 *
- 650K RANA DRIVE ..... $439.00 *

HARD DISK DRIVES

- 5 MEGABYTE WITH CONTROLLER AND SOFTWARE ..... $749.00 *
- 10 MEGABYTE ..... $1175.00 *

MONITORS

- GORILLA 12-INCH GREEN ..... $84.00 *
- GORILLA 12-INCH AMBER ..... $89.00 *
- SYNCO 12-INCH AMBER ..... $74.00 *
- USI 12-INCH GREEN ..... $99.00 *
- USI 12-INCH AMBER ..... $104.00 *
- INTRA 14-INCH COLOR/80 COLUMN ..... $269.00 *

MODEMS

- ZOOM TELEPHONICS 300-BAUD $109.00
- CENTAURI 300 BAUD $179.00
- PRO-MODEM 1200 ..... $349.00 *
- SIGNALMAN MARK XII $279.00

GRAPHICS DEVICES

- POWER PAD & STARTER KIT $119.00

VIDEO & DISPLAY EQUIPMENT

- DIGITIZER $299.00
- B & W CAMERA $195.00
- COLOR PROCESSOR ..... $99.00 *
- COLOR PROCESSOR/ENHANCER STABILIZER/SYNTHESIZER .. $279.00 *

GENERAL ITEMS

- 6 OUTLET POWER STRIP ..... $19.00 *
- SURGE PROTECTOR ..... $11.00 *
- RF MODULATOR ..... $49.00 *
- COMPUTER STAND ..... $24.00 *

GAME I/O DEVICES

- 9-16-PIN ADAPTER FOR IIe OR IIc ..... $12.00 *
- TWIN PORT GAME I/O EXTENDER ..... $25.00 *
- SINGLE PORT GAME I/O EXTENDER ..... $18.00 *
- TG GAME PADDLES ..... $29.00 *
- TG JOYSTICK ..... $31.00 *
- SAMPSON JOYSTICK ..... $25.00 *
- HAYES MACH II JOYSTICK $37.00
- HAYES MACH II JOYSTICK $45.00

* DENOTES NEW PRICE OR ITEM

SLOT EXPANSION

- 16 RAM CARD $49.00
- 64K RAM & 80 COLUMN CARD FOR IIe ..... $115.00 *
- MEMORY MASTER IIE 64K+ RAM & 80 COLUMN CARD $145.00
- MEMORY MASTER IIE 128K RAM & 80 COLUMN CARD $195.00
- MICROTEK II + 128 K VISICALC AND MEMORY EXPANSION .. $219.00
- CCS 7710 SERIAL INTERFACE CARD $117.00
- MODEM CABLE FOR CCS CARD ..... $21.00 *
- SERI-ALL SERIAL INTERFACE CARD $149.00
- 80-COLUMN CARD (VIEWMASTER) WITH SOFT-SWITCH $139.00
- CENTAURI APS Z-80 CARD $59.00
- Z-80 PLUS CARD (CPM FOR APPLE) $115.00
- FAST Z-80 CARD APPLICARD ..... $195.00 *
- TIMEMASTER II CLOCK/CALENDAR CARD $109.00
- QUIK-LOADER PROM BOARD ..... $149.00 *
- ANALOG/DIGITAL BOARD .... $99.00 *
- SUPER I/O BOARD ..... $49.00 *
- MULTIPLE-SLOT EXPANSION CHASSIS ..... $149.00 *
- SINGLE-SLOT EXTENDER .... $29.00 *

SPECIAL PERIPHERALS

- COOLING FAN WITH SURGE PROTECTOR $39.00
- DELUX KEYBOARD ..... $249.00 *
- LIFETIME EXTERNAL POWER SUPPLY ..... $179.00 *
- SHIFT KEY MOD KIT ..... $8.00 *
- SCREEN SWITCHER/ DRIVE STEPPER ..... $74.00 *

APPLE SOFTWARE

- WORD STAR ..... $195.00 *
- MAIL MERGE/SPELL STAR/ STAR/INDEX ..... $135.00 *

SPECIAL!!!!
EPSON RX-80 F/T PRINTER
ONLY $289.00!!!!

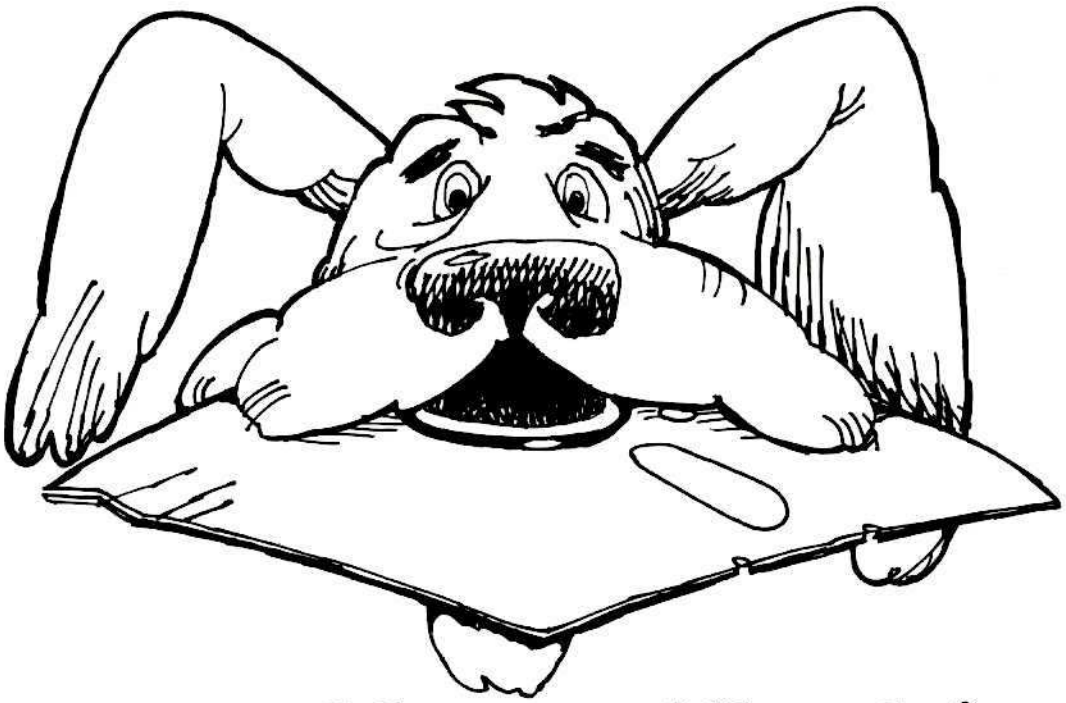
CALL FOR OUR FREE CATALOG

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

(202) 363-1313

VF ASSOCIATES

6327 WESTERN AVE. NW WASHINGTON, DC 20015



Having problems with retrieval?

You need software insurance.

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

Copy II Plus (Apple][,][Plus, //e)

This is the most widely used backup program for the Apple. Rated as "one of the best software buys of the year" by **InCider** magazine, its simple menu puts nearly every disk command at your fingertips. The manual, with more than 70 pages, describes protection schemes, and our **Backup Book™** lists simple instructions for backing up over 300 popular programs. A new version is now available that is easier to use and more powerful than before. Best of all, Copy II Plus is still only **$39.95.**

WildCard 2 (Apple][,][Plus, //e)

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

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

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

Copy II PC (IBM)

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

We are the backup professionals. Instead of diluting our efforts in creating a wide variety of programs, we specialize in offering the very best in backup products. So, protect your software investment, **before** your software meets its master.

CENTRAL POINT Software, Inc.

The Backup Professionals

To order, call 503/244-5782, 8:00-5:30 Mon.-Fri., or send your order to: Central Point Software, 9700 SW Capitol Hwy, Suite 100, Portland, OR 97219. **Prepayment is required. Please include $3 for shipping and handling ($8** outside U.S. or Canada). **Welcome**