For The Serious User Of Apple][Computers

COMPUTIST

Issue No. 48 **October 1987**

USA $3.75 Canada & Mexico $7.00 All Others $13.25

• Features:

Rich Etarip's softkey for... *Apple Business Graphics*

Jim S. Hart's *Dungeon Editor & Encounter Editor for Ultima III*

Christopher Dean's softkey & A.P.T. for... *Shadowkeep*

Readers Data EXchange Softkeys:

816 Paint GS
Addition Logician
Amnesia
Arctic Fox
Award Maker Plus
Bard's Tale II
Betterworking Word Processor
Beyond Castle Wolfenstein
Black Magic
Bookends Extended
Bop & Wrestle
Chess 7.0
Chessmaster 2000
Counting Critters
Deluxe Paint GS
Destroyer
Hacker II
Hacker II GS
Hardball
Infiltrator
Instant Music GS
J-Bird
Mabel's Mansion
Marble Madness
Math Critters
Mean 18 GS Golf
Megabots
Might & Magic
Miner 2049er II
Mouse Word
Music Construction Set GS
Music Studio GS
New Oregon Trail
Paintworks Plus 1.0 GS
Paintworks Plus 1.01 GS
Paul Whitehead Teaches Chess
PHM Pegasus
Pieman
Poetry Express
Print Shop color version
Quotient Quest
Rambo: First Blood part II
Rocky Horror Show
Sargon III*
Shanghai GS
Spindizzy
TelePorter
Temple Of Apshai trilogy
Top Draw GS
Transylvania
Ultima I
World's Greatest Baseball Game
Writing A Character Sketch
Writing A Narative

COMPUTIST
PO Box 110846-T
Tacoma, WA 98411

BULK RATE
U.S. Postage
PAID
Seattle, WA
Permit No. 211

Coping With COMPUTIST

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

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

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

■ **Commands And Controls:** In any article appearing in COMPUTIST, commands which a reader is required to perform are set apart by being in boldface and indented:

PR#6

The [RETURN] key must be pressed at the end of every such command unless otherwise specified.

Control characters are specially boxed:

6[CTRL-P]

Press [6]. Next, place one finger on [CTRL] and press [P]. Remember to enter this command line by pressing [RETURN].

■ **Requirements:** COMPUTIST programs and softkeys require one of the Apple][series of computers and a disk drive with DOS 3.3. These and other special needs are listed at the beginning of the article under "Requirements".

■ **Software Recommendations:**

1) Applesoft Program Editor such as Global Program Line Editor (GPLE).

2) Sector Editor such as DiskEdit (from the Book of Softkeys vol I) or ZAP from Bag of Tricks.

3) Disk Search Utility such as The Inspector, The CIA or The CORE Disk Searcher (from the Book of Softkeys vol III).

4) Assembler such as the S-C Assembler from S-C software or Merlin/Big Mac.

5) Bit Copy Program such as Copy][Plus, Locksmith or The Essential Data Duplicator

6) Text Editor (that produces normal sequential text files) such as Applewriter II, Magic Window II or Screenwriter II.

COPYA, FID and MUFFIN from the DOS 3.3 System Master Disk are also useful.

■ **Super IOB:** This powerful deprotection utility (COMPUTIST 32) and its various controllers are used in many softkeys. This utility is now available on each Super IOB Collection disk.

■ **RESET Into The Monitor:** Softkeys occasionally require the user to stop the execution of a copy-protected program and directly enter the Apple's system monitor. Check the following list to see what hardware you will need to obtain this ability.

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

Apple][Plus - Apple compatibles: 1) Install an F8 ROM with a modified RESET vector on the computer's motherboard as detailed in the "Modified ROM's" article (COMPUTIST 6 or Book Of Softkeys III) or the "Dual ROM's" article (COMPUTIST 19).

Apple //e - Apple //c: Install a modified CD ROM on the computer's motherboard. Cutting Edge Ent. (Box 43234 Ren Cen Station-HC; Detroit, MI 48243) sells a hardware device that will give you this important ability but it will void an Apple //c warranty.

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

■ **Keying In Applesoft Programs:** BASIC programs are printed in COMPUTIST in a format that is designed to minimize errors for readers who key in these programs. If you type:

10HOME:REMCLEAR SCREEN

The LIST will look like:

10 HOME : REM CLEAR SCREEN

because Applesoft inserts spaces into a program listing before and after every command word or mathematical operator. These spaces usually don't pose a problem except in line numbers which contain REM or DATA commands. There are two types of spaces: those that have to be keyed and those that don't. Spaces that must be keyed in appear in COMPUTIST as delta characters (▵). All other spaces are there for easier reading. NOTE: If you want your checksums (See "Computing Checksums" section) to match up, you must only key in (▵) spaces after DATA statements.

■ **Keying In Hexdumps:** Machine language programs are printed in COMPUTIST as both source code and hexdumps. Hexdumps are the shortest and easiest format to type in. You must first enter the monitor:

CALL -151

Key in the hexdump exactly as it appears in the magazine, ignoring the four-digit checksum at the end of each line (a "$" and four digits). A beep means you have typed something that the monitor didn't understand and must, therefore, retype that line.

When finished, return to BASIC with:

E003G

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

■ **Keying In Source Code** The source code is printed to help explain a program's operation. To key it in, you will need the S-C Assembler. Without this assembler, you will have to translate pieces of the source code into something *your* assembler will understand. A table of S-C Assembler directives appears in COMPUTIST 17.

■ **Computing Checksums** Checksums are four-digit hexadecimal numbers which tell if you keyed a program exactly as it appears in COMPUTIST. There are two types of checksums: one created by the CHECKBIN program (for machine language programs) and the other created by the CHECKSOFT program (for BASIC programs). Both appeared in COMPUTIST 1 and The Best of Hardcore Computing. An update to CHECKSOFT appeared in COMPUTIST 18. If the published checksums do not match those created by your computer, then you typed the program incorrectly. The line where the first checksum differs has an error.

■ **CHECKSOFT Instructions:**

LOAD *filename*
BRUNCHECKSOFT

Get the checksums with: **&** [RETURN] and correct the program where the checksums differ.

■ **CHECKBIN Instructions:**

CALL -151
BLOAD *program filename*

Install CHECKBIN at an out of the way place

BRUN CHECKBIN,A$6000

Get the checksums by typing the starting address, a period and ending address of the file followed by a [CTRL-Y] [RETURN].

xxx.xxx[CTRL-Y]

Correct the lines at which the checksums differ.

You have a LEGAL RIGHT to an unlocked backup copy

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

Furthermore, the copyright laws guarantee your right to such a DEPROTECTED backup copy:

..."It is not an infringement for the owner of a copy of a computer program to make or authorize the making of another copy or ***adaptation*** of that computer program provided:

1) that such a new copy or adaptation is created as an essential step in the utilization of the computer program in conjunction with a machine and that it is used in no other manner, or

2) that such new copy or adaptation is for archival purposes only and that all archival copies are destroyed in the event that continued possession of the computer program should cease to be rightful.

Any exact copies prepared in accordance with the provisions of this section may be leased, sold, or otherwise transferred, along with the copy from which such copies were prepared, only as part of the lease, sale, or other transfer of all rights in the program. Adaptations so prepared may be transferred only with the authorization of the copyright owner."

United States Code title 17, §117 (17 USC 117)

Be assured of receiving the latest issue of COMPUTIST each month without the hassle of making a trek to the local computer store and not finding COMPUTIST so of course you ask the clerk if they have COMPUTIST and they tell you they don't carry COMPUTISTbut maybe you could try the computer store down the block because they might have an issue of COMPUTISTso you go to the computer store down the block and ask them if they have COMPUTIST but of course they don't (they just ran out of COMPUTISTyesterday) so you ride your unicycle clear across town to see if the computer store across town has COMPUTIST but when you go inside and ask for COMPUTIST, they don't have COMPUTIST either and suggest that you try the store you originally went to in the first place so go home disgusted that you missed another issue of COMPUTIST.

Can't Find COMPUTIST Anywhere?

Stop searching and **subscribe now!**

Annual subscription rates (12 issues):

U.S. Subscriptions - sent third class - **$32**
U.S./Canada/Mexico - sent First Class - **$45**
U.S./Canada/Mexico First Class PLUS library disk - **$100**

All other Foreign Subscriptions - **$75**
All other Foreign PLUS library disk - **$140**

■ Use the form on the right to order or renew your subscription.

Mag & Disk Combo

■ You may upgrade your current subscription to a magazine & disk combination by sending $5.50 ($6.50 foreign) per remaining issue.

Is it time to renew?

■ Check your mailing label to see if you need to renew your subscription.

Are you moving soon?

■ If you're moving, let us know at least 30 days in advance.

Issues missed due to non-reciept of Change-of-Address may be acquired at the regular back issue rates.

Remember, the Post Office does not forward third class mail unless requested.

COMPUTIST is not responsible for replacing issues lost while forwarding order is in effect.

Yes, I want to subscribe to COMPUTIST.
Enclosed are funds for an annual (12 issue) subscription.

I am ☐ A new Subscriber
☐ Renewing my current subscription
☐ Changing my address (please include latest label)

☐ U.S. - $32
☐ U.S./Canada/Mexico First Class - $45
☐ U.S./Canada/Mexico First Class plus Library Disk - $100

☐ All other Foreign - $75
☐ All other Foreign plus Library Disk - $140

Name _____ ID# _____

Address _____

City _____ State _____ Zip _____

Country _____ Phone _____

VISA MC _____-_____-_____-_____ Exp. _____

Signature _____ CP48

U.S. Funds drawn on U.S. bank. Please allow 4 to 8 weeks for subscription to commence.

Mail to: **COMPUTIST PO Box 110846-T Tacoma, WA 98411** or phone (206) 474-5750

Take time now to save time later!

Editorial.....

RDEX

Well, everyone's had time to receive COMPUTIST 47 and check out the new format. I hope you like RDEX as much as I. We've really only just begun to use the real strength (timeliness) of this new format.

In this issue you'll find a lot more softkeys, some submitted within the last 30 days. There are softkeys for IIgs programs and PRODOS. You'll also find repeats on some softkeys. We can't verify the submissions so duplicates will be printed as long as there is some variation in the method used.

All of RDEX is unpaid material sent by readers to help other readers. Some of the material are previously submitted manuscripts that were donated when the authors learned of our problems. I'd like to thank those writers who responded so quickly to our letters. You'll find a lot of your material in this issue. If you sent a manuscript and haven't received an acknowledgement, hang-on a little longer. There is still a tall stack of letters sitting on my desk.

Hardware Corner

There is another installment of the Hardware Corner on page 40. A flashing light gizmo that's functional as well as impressive to watch. Bobby says there's a lot more where that came from. The direction of Hardware Corner is still up in the air. We need some reader input on what kind of hardware you would like to see.

Letters on disk

I'd like to make a plea to all our readers. If you send a letter or note to RDEX that is longer than one page, *please send it on disk*. It's a lot easier to transfer to the typesetter and improves your chances of getting into the next issue.

Don't send any original commercial software

If you do send in softkeys, please, whatever you do, **don't send any original disks**. There are too many people coming and going around here and your originals may get lost.

Help! We need more local volunteers

We could use some more volunteer help. Are there any local subscribers who would like to do some editing or programming work? We're in desparate need of computists who can type quickly and accurately. If you volunteer, you'll learn a lot about publishing a magazine.

Most Wanted

We've received numerous queries about the whereabouts of the most wanted list. It was lost in the shuffle, but we found it again. It's kind of dusty and way out of date but we're going to print it anyway. Write us if you see any errors or have any updates.

Thanks to all of you for your support.

I'd like to thank all of the volunteers who sat up nights putting together COMPUTIST 48.

Charles R. Haight
Publisher/Editor

Publisher/Editor.....Charles R. Haight
Managing Editor.....Ryuji
Circulation.....Karen Fitzpatrick

Advertising.....(206) 474-5750
Printing....Valco Printing, Seattle WA

- Address all advertising inquiries to:

COMPUTIST
Advertising Department
PO Box 110816
Tacoma, WA 98411

- Mail manuscripts or letters to:

COMPUTIST
PO Box 110846-K
Tacoma, WA 98411.

- ***Unsolicited manuscripts*** are assumed to be submitted for publication at our standard rates of payment published in the most recent issue of COMPUTIST magazine (see the inside of the back page). SoftKey purchases all and exclusive rights. For more information on submitting manuscripts, consult the Writer's guide on the inside of the back page.

- 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 COMPUTIST magazine or SoftKey Publishing.

- Apple usually refers to an Apple][computer and is a trademark of Apple Computers, Inc.

SUBSCRIPTIONS: Rates (for 12 issues):

U.S.	$32
U.S. 1st Class	$45
Canada & Mexico	$45
Other Foreign	$75

Direct your subscription inquiries to:

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

DOMESTIC DEALER RATES:
Call (206) 474-5750 for more information.
Change Of Address: Please allow 4 weeks for change of address to take effect. On postal form 3576 supply your new address and your most recent address label. Issues missed due to non-receipt of change of address may be acquired at the regular back issue rate.

COMPUTIST

Issue 48 October 1987

RDEX Contributors

Special Features:

20 Readers Data Exchange

***Softkeys: software title* (page #)**

• *816 Paint GS*—21 • *Addition Logician*—34 • *Amnesia*—31 • *Arctic Fox*—31,36 • *Award Maker Plus*—27 • *Bard's Tale II*—33,39 • *Betterworking Word Processor*—31 • *Beyond Castle Wolfenstein*—38 • *Black Magic*—31 • *Bookends Extended*—32 • *Bop & Wrestle*—37 • *Chess 7.0*—22 • **Chessmaster 2000*—35 • *Counting Critters*—34 • *Deluxe Paint GS*—21 • *Destroyer*—37 • *Hacker II*—20,38 • *Hardball*—25 • *Infiltrator*—23 • *Instant Music GS*—20 • *J-Bird*—36 • *Mabel's Mansion*—23 • *Marble Madness*—32,35 • *Math Critters*—34 • *Mean 18 GS Golf*—20 • *Megabots*—24 • *Might & Magic*—31 • *Miner 2049er II*—37 • *Mouse Word*—21 • *Music Construction Set GS*—21 • *Music Studio GS*—20,39 • *New Oregon Trail*—26 • *Paintworks Plus GS*—20,30,38 • *Paul Whitehead Teaches Chess*—26 • *PHM Pegasus*—36 • *Pieman*—38 • *Poetry Express*—34 • *Print Shop color version*—21 • *Quotient Quest*—34 • *Rambo First Blood part II*—25 • *Rocky Horror Show*—36 • **Sargon III*—35 • *Shanghai GS*—20,21,39 • *Spindizzy*—21,28 • *TelePorter*—24 • *Temple Of Apshai trilogy*—37 • *Top Draw GS*—21 • *Transylvania*—37 • *Ultima I*—22 • *World's Greatest Baseball Game*—25 • *Writing A Character Sketch*—34 • *Writing A Narative*—34

N: *notes* **APT: Advanced Playing Techniques** **A:** *adventure Tips* **BUG:** *mispsrints-errors*

• **N:** *Megabots*—24 • **APT:** *Ultima IV*—25,30 • **APT:** *Aztec*—28 • **N:** *Elite softkey*—29 • **N:** *Fantavision sofkey*—32 • **BUG:** *Auto Duel softkey*—33 • **N:** *Goonies softkey*—33 • **A:** *Bureaucracy*—33 • **A:** *King's Quest*—33

Softkey for.....

Apple Business Graphics

Apple Computers, Inc.

by Rich Etarip

■ **Requirements:**

- ☐ 64K Apple
- ☐ A blank disk
- ☐ A sector editor
- ☐ COPYA (on DOS 3.3 System Master)
- ☐ *Apple Business Graphics* diskette

When I was given the chance to try to deprotect *Apple Business Graphics*, I almost declined it. I told myself that it would probably be a lost cause as far as deprotection not only because it had been on COMPUTIST'S Most Wanted List for about the past 30 issues, but also that it was protected by Apple Computer themselves. One point I failed to realize was that this was a 1981 release and back then, copy protection schemes were not quite as advanced and complex as they are today.

The *Apple Business Graphics* package includes 4 diskettes:

1 The program disk called Apple Business Graphics Plot
2 A backup copy of the program disk
3 A data diskette which is not protected
4 A blank diskette

Apple Business Graphics Plot

The disk we will be working with will be the program diskette: *Apple Business Graphics Plot*.

The disk itself is easily copyable except for Track 1 which even today's bit copiers have problems copying. Right then I thought I knew what I was looking for. All I had to do was search the disk for accesses to $C08C which is the address to read a byte from the disk. From there, I could easily find the nibble-count routine and defeat it. The problem is that I couldn't find a nibble-count routine anywhere.

The next thing I did was remove the cover from my disk drive so I could watch the drive head. Then I booted up the copy I made. I couldn't even make a guess as to when Track 1 was accessed because it was going all over the place and continually sliding back to Track 0. My guess was that Track 1 was accessed a number of times during boot.

Then I booted the original diskette and listened to the disk drive. I had only booted up the original once before and this time I noticed something that I had failed to notice before.

Almost right before the title page comes up, the disk drive recalibrates. However, on the copy I made, it doesn't. That made me think of a game I had deprotected a couple of years ago. It was copyable except for one track and unless it encountered an error in reading that track, it would not work. In order to find out if this was the same case, I copied a protected Track 1 from one of my game disks to the copy of *Apple Business Graphics* and sure enough, it worked fine.

Apple Business Graphics must encounter a read error on Track 1 or it will not work correctly. It would be simple to just make a copy of the disk and then copy a protected track to Track 1 but we aren't going to bother doing that for a working backup copy.

My aim is not to find ways around the protection, but to REMOVE the protection. You can easily make copies of the disk with a copier that ignores errors (such as the *Locksmith* Fast Disk Backup) and when you boot the copied diskette, it will appear to boot fine but once it gets past the title screen it gives a PLOT: NOT ON LINE error. The reason it doesn't work is because of the check to Track 1.

What we have to do is find out where it reads from Track 1. Their disk loading routine is stored in the language card at $D000 and this can be found on Track 2 of the disk.

Take a look at Listing 1 which is a small section of their loader. At $D18E is a JSR $D324 followed by a BCS $D183. By trial and error, I found this to be where the read error from Track 1 is detected. If the carry flag is set (Branch if Carry Set) upon return from the subroutine at $D324, that means that there was a read error.

If you look at Listing 2 you will see that $D324 is the routine to read the address field from the disk. If no read errors occur, the carry bit will be cleared, but if it does encounter an error, the carry will be set.

We are going to have to fool the loader into thinking it has run into an error when it reads Track 1. In other words, the carry bit must be set when returning from the JSR $D324 at $D18E. We could change the CLC (CLear Carry) at $D37E to a SEC (SEt Carry) but it wouldn't work correctly.

Remember, the routine at $D324 is used for reading all tracks from the disk and if we do that, it will think that there was an error on every track. The carry bit must be set ONLY after reading Track 1. By further examination, I found that the current track to be read is stored in $3A4. It is basically simple now. Check if $3A4 is $01, and if it is, set the carry bit so it thinks there was an error in reading Track 1.

I looked through memory for available space to write a routine and $DF00 appered to have just 'garbage' memory so I cleared out the page (which is on Track 2 Sector F) and booted the disk and it worked fine.

Next, I wrote a short routine at $DF00 to cause an error on Track 1 and tried the copy. It appeared to work fine until it started using the data disk and user disk. Then I was getting errors. The problem here was that the routine I wrote was not only causing errors on Track 1 of the program disk but any Track 1 it tries to read or write.

I had a simple solution to that problem though. I simply modified my routine to store an RTS at the beginning of itself after the error was checked on Track 1 and the routine could not be used anymore. I then booted the disk again and tried it out. This time it appeared to work fine up until I exited the disk format section. It then went to load the main program in again and it also checked for Track 1 again and since my routine was no longer useable, it crashed right there. Once again I knew what I had to do. Find out where it begins reading the program disk and remove the RTS that I stored at the beginning of my routine earlier.

Well, I spent endless hours looking through the code, making changes trying to find out where the load begins. No luck at all. The code for the loader was very complex and difficult to trace and after a while, I finally gave up on that idea. It appears that they use a different routine to load in the program the second time around and I couldn't find it anywhere.

There was only one question running through my mind at the time. "What is the easiest way to tell two disks apart when you are reading from them?" I kept asking myself that for quite a while and finally it hit me. It was so easy, and all this time I never even thought about it.

All I have to do is copy the program disk with an odd volume number that neither of the other disks would contain and then check for that. That way, I can call my routine right after the address field is read (which contains Volume, Track, Sector and Checksum) and verify that the volume is 5 before causing an error on Track 1. As mentioned earlier, the routine to read the address field is at $D324 and the JSR $D324 is at $D18E. We will change this to a

JSR $DF00 and at $DF00, JSR to $D324 and then check the volume number (which will be in location $CD). Then if the volume is 5, check the track (which is in location $3A4 or $CC) and if it is a $01, set the carry and return so it thinks that there was a read error. This is what the routine will look like:

```
DF00- JSR $D324     -Read an address field
DF03- LDA $CD       -Get the volume number
DF05- CMP #$05      -Is the volume 05?
DF07- BNE $DF12     -No. Branch to exit.
DF09- LDA $03A4     -Get the track number
DF0C- CMP #$01      -Is the Track 1?
DF0E- BNE $DF12     -No. Branch to exit.
DF10- SEC           -It is Track 1 of the
                     program disk. Set the
                     carry flag to show there
                     was an error.
DF11- RTS           -Return to caller
DF12- CLC           -No error. Clear carry
DF13- RTS           -Return to caller
```

Before anything else can be done, though, the disk must be copied with COPYA. We will modify it to ignore the errors on Track 1 and to write back with a volume of 5.

The *Locksmith* Fast Disk Backup will not work for this because it will not change the volume number.

The Deprotection

1 Load COPYA and COPY.OBJ0 from your system master diskette.

LOAD COPYA
BLOAD COPY.OBJ0

2 Delete line 70 so it does not reload COPY.OBJ0.

70

3 Now make a modification so it writes to the duplicate disk with a volume of 5.

247 POKE 714,5

4 Next, enter the monitor.

CALL -151

5 Make a modification to COPY.OBJ0 so it will ignore any errors it encounters.

3A1:18

6 Modify DOS so it does not recalibrate the drive upon error. If this is not done, you will have an awfully noisy copying process.

BDD4:4C 04 BE

7 Exit back to BASIC.

C

8 Finally, type RUN to run the copy program and follow the instructions until the copying is finished.

9 All that is left is making the sector edits to the disk. First, read Track 2 Sector F and enter the following at byte $00. This is the routine that goes at $DF00 to check for the volume and track. Once this is entered, rewrite the sector.

```
xx00:  20 24 D3 A5 CD C9 05 D0
xx08:  09 AD A4 03 C9 01 D0 02
xx10:  38 60 18 60
```

10 Now read Track 2 Sector E and at byte $8F change the $24 and $D3 to a $00 and $DF which will change the JSR $D324 to a JSR $DF00. Write the sector back to the disk.

And that's all there is to it. You now have a copyable version of *Apple Business Graphics*.

Note: One thing I should point out is if you wish to make copies of your new backup, you must use a copier that copies the volume number.

Listing 1

```
D17E-  A0 80     LDY  #$80
D180-  8C FE DF  STY  $DFFE
D183-  CE FE DF  DEC  $DFFE
D186-  30 20     BMI  $D1A8
D188-  20 01 D0  JSR  $D001
D18B-  AE A1 03  LDX  $03A1
D18E-  4C 24 D3  JMP  $D324
D191-  B0 F0     BCS  $D183
D193-  A5 CC     LDA  $CC
D195-  CD FB DF  CMP  $DFFB
D198-  F0 38     BEQ  $D1D2
D19A-  A4 C7     LDY  $C7
D19C-  0A        ASL
D19D-  99 78 04  STA  $0478,Y
D1A0-  AD FB DF  LDA  $DFFB
D1A3-  CE FD DF  DEC  $DFFD
D1A6-  D0 1B     BNE  $D1C3
D1A8-  CE FF DF  DEC  $DFFF
D1AB-  F0 1C     BEQ  $D1C9
D1AD-  AD FB DF  LDA  $DFFB
D1B0-  48        PHA
D1B1-  A9 0A     LDA  #$0A
D1B3-  8D FD DF  STA  $DFFD
D1B6-  A9 60     LDA  #$60
D1B8-  A4 C7     LDY  $C7
D1BA-  99 78 04  STA  $0478,Y
D1BD-  A9 00     LDA  #$00
D1BF-  20 6B D2  JSR  $D26B
D1C2-  68        PLA
D1C3-  20 6B D2  JSR  $D26B
D1C6-  4C 7E D1  JMP  $D17E
D1C9-  A9 40     LDA  #$40
D1CB-  28        PLP
D1CC-  4C 59 D2  JMP  $D259
D1CF-  4C 57 D2  JMP  $D257
D1D2-  AD A3 03  LDA  $03A3
D1D5-  48        PHA
D1D6-  A5 CD     LDA  $CD
D1D8-  8D AE 03  STA  $03AE
D1DB-  68        PLA
D1EF-  00        BRK
```

Listing 2

```
D324-  A0 FC     LDY  #$FC
D326-  84 CF     STY  $CF
D328-  C8        INY
D329-  D0 04     BNE  $D32F
D32B-  E6 CF     INC  $CF
D32D-  F0 51     BEQ  $D380
D32F-  BD 8C C0  LDA  $C08C,X
D332-  10 FB     BPL  $D32F
D334-  C9 D5     CMP  #$D5
D336-  D0 F0     BNE  $D328
D338-  EA        NOP
D339-  BD 8C C0  LDA  $C08C,X
D33C-  10 FB     BPL  $D339
D33E-  C9 AA     CMP  #$AA
D340-  D0 F2     BNE  $D334
D342-  A0 03     LDY  #$03
D344-  BD 8C C0  LDA  $C08C,X
D347-  10 FB     BPL  $D344
D349-  C9 96     CMP  #$96
D34B-  D0 E7     BNE  $D334
D34D-  A9 00     LDA  #$00
D34F-  85 CE     STA  $CE
D351-  BD 8C C0  LDA  $C08C,X
D354-  10 FB     BPL  $D351
D356-  2A        ROL
D357-  85 CF     STA  $CF
D359-  BD 8C C0  LDA  $C08C,X
D35C-  10 FB     BPL  $D359
D35E-  25 CF     AND  $CF
D360-  99 CA 00  STA  $00CA,Y
D363-  45 CE     EOR  $CE
D365-  88        DEY
D366-  10 E7     BPL  $D34F
D368-  A8        TAY
D369-  D0 15     BNE  $D380
D36B-  BD 8C C0  LDA  $C08C,X
D36E-  10 FB     BPL  $D36B
D370-  C9 DE     CMP  #$DE
D372-  D0 0C     BNE  $D380
D374-  EA        NOP
D375-  BD 8C C0  LDA  $C08C,X
D378-  10 FB     BPL  $D375
D37A-  C9 AA     CMP  #$AA
D37C-  D0 02     BNE  $D380
D37E-  18        CLC
D37F-  60        RTS
D380-  38        SEC
D381-  60        RTS
```

More A.P.T. Utilities For...

Ultima III

Origin Systems

Dungeon Editor

by Jim S. Hart

■ **Requirements:**

☐ Ultima III
☐ An initialized disk
☐ A copy of your scenario disk

You have just finished battling a renegade group of thieves. It was a tough battle but your party managed to get away with no major losses. You now inventory what the party has left and discover that the thieves made off with a large percentage of your gold! The only way to get it back quickly and with a minimum of risk is to go into one of Sosaria's dungeons. Which one? The one by the main castle had some, or was it the one in the southwest corner of the continent? You seem to recall that some of the gold was hard to get to or was on the oppressive 8th level. Hmmm. What do you do?

This dilemma need never happen to you anymore for I present two of my creations:

The Ultima III...

Dungeon Editor and

Encounter Editor

The **Dungeon Editor** will allow you to reshape the way the dungeons look. Want to make the first level all gold chests? You can, now.

Want to make it so that you can travel from level to level without having to follow the passages to find the ladders? It's easy to put two-way ladders anywhere you want to now.

You can "personalize" and customize them anyway you want.

As an extra added bonus, the **Encounter Editor** lets you alter the appearance of the several different encounters you meet in the dungeons such as The Time Lord, The Shrine, The Hot Rod of Marks, and The Fountain. This has no practical use but again, you can "customize" it to meet your fancy. For me, this makes the game more enjoyable.

Keying In the Programs

There are three programs to be entered:

1 The Dungeon Editor

2 The Encounter Editor

3 a small machine language program to facilitate reading and writing to the disk called Sector Zap.

The editors are written in good ol' Applesoft BASIC and Sector Zap is written in machine language.

First, type in listing #1 and save it to disk with the command:

```
SAVE III.DUNGEON.EDITOR
```

Now type in listing #2 and save it to disk:

```
SAVE III.ENCOUNTER.EDITOR
```

Finally, enter the monitor, type in the hexdump using the procedure outlined on the inside front cover of this magzine.

Save it to disk with:

```
BSAVE SECTOR.ZAP, A$300, L$21
```

The Programs

The BASIC programs are well documented so altering them should not be too much of a problem. Each section of code is marked off by REM statements that explain what it does. The two sections of code that may be of interest to programmers are the **cursor bar** subroutine and the **cursor movement** routine.

Cursor Bar from Beagle Bros.

The cursor bar routine is a modified and adapted section of code from one of the Beagle Bros flyers. When I saw it, I knew this was what I wanted to use in my editing programs.

The routine itself has been modified to allow it to use the entire screen, not just rows 7 to 11, and have a programmable ESCape key function. A short example is needed to demonstrate this.

Suppose you have 4 items in the menu. Using the arrow keys, you can move among and highlight the 4 choices. When you press RETURN, the number of the choice highlighted will be returned in the variable ZK, i.e. choice #3 returns ZK=3.

Programmable ESC-key

However, if you want to allow the user to go back a screen and do not want to add a "GO BACK ONE SCREEN" menu option to every menu, then use the ESC key for this function. When the ESC key is pressed, a value of -1 is put into ZK. This can be checked for quite easily. You can make the ESC key perform any function this way.

By the way, the only two keys that get you out of the cursor bar routine are the RETURN and the ESC keys.

Here is a list of the variables you will need to set prior to calling the cursor bar subroutine if you plan to use it in one of your programs:

& Encounter Editor

Values Upon Entry:

```
ZV = Vertical top of menu
ZH = Horizontal left of menu
Z2 = # of items in this menu
ZI = Which menu in menu array
```

Value Upon Exit:

```
ZK = # of menu item selected
                (-1 if ESC was used)
```

The first thing to do is to figure out where on the screen that your menu (left justified) is going to reside. Now find out what the VTAB and HTAB values are for the uppermost left character. The VTAB value goes into variable ZV. The HTAB value goes into variable ZH. The number of items that are in this particular menu goes into the variable Z2. This subroutine was designed to be used by more than one menu. Therefore, the menues are stored in the array ZM$(# of menu , items in menu). The number of the menu that is to be displayed goes into the variable ZI. When the user makes a choice, the variable ZK will hold the number of the menu item that they chose (eg. if they chose item #4 then ZK=4). As noted before, the ESC key gives a value of -1 and can be used for any purpose. If it is not going to be used, a check should be made for it. If it was pressed (ZK = -1), then go back to the line that originally called the subroutine to begin with.

To see how all of this works, examine the programs themselves and try them out.

Cursor Movement

The section of code that governs the movement of the cursor on the editing screen may be of some interest to the readers. These few lines allow the user to move around the editing screen using the "IJKM" diamond.

The cursor has a wrap-around feature - if you go off the top, you will appear in the same column at the bottom. The same goes for horizontal movement except you are either raised or dropped one depending on which way you are going. This cursor movement can be thought of as what would be used on a single screen text editor.

A FAST Endless FOR-NEXT Loop

Some of you may already have noticed the endless FOR-NEXT loop in the program that starts "FOR LOOP = 1 TO 2 STEP Ø".

Why set up an endless loop? *Speed!*

When your programs start to grow in size, you immediately look for two things: a way to compact or modularize the code, and a way to speed up execution.

An endless loop, like the one in the program, speeds up execution of a block of repeating code like the editing screen because no GOTOs are used to return to the start of the loop. Hence, Applesoft does not have to go hunting through a lot of line numbers to find the one you want. This saves time.

The way you get out of the loop is to change the value of LOOP so that it equals or exceeds the loop limit which in this case is two (2). Try out an endless loop in your spare time to get a feel for it. Who knows, you may use one in your next submission to COMPUTIST.

Faster Code VS Compact Code

Compacting and speeding up executing time do not always work hand in hand.

In the editing screen loop, there is a subloop which checks to see if the user has entered in a key to change one on-screen item to another, such as replacing a section of lava with water. The check goes through a string which contains the legal keypresses and if a match is found, the appropriate action is taken.

I originally used a long series of IF-THEN statements to accomplish this. When the code was rewritten to make it shorter, I found that the execution speed had diminished. This goes to show that good programming does not necessarily mean faster execution time.

Oddities On Editing The Dungeons

Editing the dungeons can be quite an interesting feat. The actual dungeon functions a lot like the cursor on the editing screen: it has a wrap-around feature. You could really get someone lost if you had the inclination to do so.

Another interesting feature (actually a bug - see definition of FEATURE in COMPUTIST No. 38) is what happens when you put a two way ladder on level 8 of any dungeon. This allows you to go to both level 7 and level 9 (!). I didn't know there was a level 9.

Level 9 ???

Obviously, the code that the folks at Origin Systems wrote was written so that more than 8 levels could be used, possibly as a future enhancement. I suggest you go down to level 9 and then P)eer at a gem. What you see will be most confusing and there are no guarantees that you will get back once you enter. A quick reboot can take care of the problem.

The enterprising adventurer might want to find out what sector is read in as level 9 for the dungeon they are editing, see if it is being used for anything else, and if it isn't then create a new level complete with gold and gremlins. Make your Ultima III different from everyone else's!

Final Notes

I hope readers out there gain some enjoyment from these programs. It might even be possible to learn something from them in the form of subroutines or techniques.

Budding programmers take note: comment your programs to death and make the flow of logic clear to all who examine the program. This is needed for any professional venture such as submitting a program to a magazine such as COMPUTIST or working as a programmer in a company.

Dungeon Editor

```
10 REM * ================================*
20 REM *     Ultima III Dungeon Editor    *
30 REM *          by : Jim S. Hart        *
40 REM * ================================*
50 REM
60 REM * --------------------------------*
70 REM *  Initialize arrays and constants *
80 REM * --------------------------------*
90 :
100 HIMEM: 38144
110 IF PEEK (768) + PEEK (770) <> 329 THEN
    PRINT CHR$ (4 ); "BLOAD^SECTOR.ZAP,
    A$300"
120 BUF = 38143 :RWTS = 768 :KB = - 16384 :CKB
    = - 16368
130 IN$ = "1234567890QWERT" :ZERO = 0 :CODE =
    ZERO
140 DIM DN$(8 ) ,MAP$(8 ) ,ZM$(2 ,9 ) ,TRK(8
    ,9 ) ,SEC(8 ,9 ) ,CH$(15 ) ,CH(15 )
150 :
160 REM * --------------------------------*
170 REM *       Read data into arrays      *
180 REM * --------------------------------*
190 :
200 GOSUB 1590 : HOME
210 INPUT "INSERT^SCENARIO^DISK^&^
    PRESS^RETURN^" ;A$
220 :
230 REM * --------------------------------*
240 REM *     Select dungeon and level     *
250 REM * --------------------------------*
260 :
270 GOSUB 1730
280 :
290 REM * --------------------------------*
300 REM *  Read in dungeon level selected  *
310 REM * --------------------------------*
320 :
330 CODE = 1 : GOSUB 2180
340 :
350 REM * --------------------------------*
360 REM *  Clear screen and print headers  *
370 REM * --------------------------------*
380 :
390 TEXT : HOME
400 PRINT DN$; "^(" ;LN; ")"
410 FOR I = 1 TO 19 : PRINT "-" ; : NEXT : PRINT
    "-" : PRINT
420 PRINT SPC( 12 ); "1111111"
430 PRINT SPC( 3 ); "1234567890123456"
440 PRINT : PRINT TAB( 4 );
450 :
460 REM * --------------------------------*
470 REM *     Calculate what is in each    *
480 REM *     location and print it out    *
490 REM * --------------------------------*
500 :
510 FOR J = 1 TO 256 :CHAR = PEEK (BUF + J )
520 FOR I = 1 TO 15
530 IF CH(I ) < > CHAR THEN NEXT I
540 POKE 50 ,255 - ((CHAR = 1 ) * 128 ) -
    ((CHAR = 128 ) * 192 )
550 PRINT CH$(I ); : NORMAL : I = 15
560 NEXT I
570 IF (J / 16 ) = INT (J / 16 ) THEN PRINT "^"
    : PRINT TAB( 4 );
580 NEXT J : PRINT "^"
590 FOR I = 1 TO 16 : VTAB I + 6 : PRINT SPC( I < 10
    ); I : NEXT
600 POKE 32 ,22 : POKE 33 ,18
610 :
620 REM * --------------------------------*
630 REM *      Print options on screen     *
640 REM * --------------------------------*
650 :
660 HOME :VT = 7 :HT = 4 :V = 1 :H = 1
670 VTAB 2
680 PRINT "1>HALLWAY"
690 PRINT "2>SOLID^WALL"
700 PRINT "3>CHEST"
710 PRINT "4>DOOR"
720 PRINT "5>TIME^LORD"
730 PRINT "6>FOUNTAIN"
740 PRINT "7>STRANGE^WIND"
750 PRINT "8>TRAP"
760 PRINT "9>ROD"
770 PRINT "0>GREMLINS"
780 PRINT "Q>WRITINGS"
790 PRINT "W>UP^LADDER"
800 PRINT "E>DOWN^LADDER"
810 PRINT "R>2^WAY^LADDER"
820 PRINT "T>HIDDEN^DOOR"
830 PRINT "=>SAVE"
840 PRINT "ESC>QUIT" : PRINT
850 PRINT SPC( 7 ); "I"
860 PRINT SPC( 6 ); "J" ; : INVERSE : PRINT "^" ;
    : NORMAL : PRINT "K"
870 PRINT SPC( 7 ); "M"
880 POKE CKB ,ZERO : POKE 32 ,ZERO : POKE 33 ,40
890 :
900 REM * --------------------------------*
910 REM *  Main processing loop. It uses   *
920 REM *  infinitely repeating FOR-NEXT   *
930 REM *   loops. This approach is much   *
940 REM *     faster than using GOTO's.    *
950 REM * --------------------------------*
960 :
970 FOR ALOOP = 1 TO 2 STEP ZERO
980 VTAB VT : HTAB HT
990 :
1000 REM * --------------------------------*
1010 REM *           Get keypress          *
1020 REM * --------------------------------*
1030 :
1040 GET K$ : POKE CKB ,ZERO
1050 :
1060 REM * --------------------------------*
1070 REM *       Respond to keypress       *
1080 REM * --------------------------------*
1090 REM
1100 REM * --------------------------------*
1110 REM *       Cursor movement keys      *
1120 REM * --------------------------------*
1130 :
1140 IF K$ = "J" THEN HT = HT - 1 :H = H - 1 : IF H
     < 1 THEN H = 16 :HT = 19 :V = V - 1 :VT = VT -
     1 : IF V < 1 THEN V = 16 :VT = 22
1150 IF K$ = "K" THEN HT = HT + 1 :H = H + 1 : IF H
     > 16 THEN H = 1 :HT = 4 :V = V + 1 :VT = VT + 1
     : IF V > 16 THEN V = 1 :VT = 7
1160 IF K$ = "M" THEN VT = VT + 1 :V = V + 1 : IF V
     > 16 THEN V = 1 :VT = 7
1170 IF K$ = "I" THEN VT = VT - 1 :V = V - 1 : IF V
     < 1 THEN V = 16 :VT = 22
1180 :
1190 REM * --------------------------------*
1200 REM *    If cursor key then go back    *
1210 REM * --------------------------------*
1220 :
1230 IF K$ = "I" OR K$ = "J" OR K$ = "K" OR K$ = "M"
     THEN NEXT ALOOP
1240 :
1250 REM * --------------------------------*
1260 REM *      Check for dungeon item      *
1270 REM * --------------------------------*
1280 :
1290 FOR I = 1 TO LEN (IN$ )
1300 I$ = MID$ (IN$ ,I ,1 )
1310 IF I$ < > K$ THEN 1360
1320 POKE 50 ,255 - ((CH(I ) = 1 ) * 128 ) - ((CH(I
     ) = 128 ) * 192 )
1330 PRINT CH$(I ) ; NORMAL
1340 POKE BUF + (16 * (V - 1 ) ) + H ,CH(I )
1350 I = LEN (IN$ )
1360 NEXT I
1370 :
1380 REM * --------------------------------*
1390 REM *save dungeon/exit editing screen*
1400 REM * --------------------------------*
1410 :
1420 REM - Need to quit & save dungeon level?
1430 :
1440 IF K$ < > "=" THEN 1490
1450 CODE = 2 : GOSUB 2180
1460 :
1470 REM -- Need to quit w/o saving?
1480 :
1490 IF K$ < > CHR$ (27 ) THEN 1520
1500 ALOOP = 2
1510 :
1520 NEXT ALOOP
1530 GOTO 270
1540 :
1550 REM * --------------------------------*
1560 REM *           Read Data in           *
1570 REM * --------------------------------*
1580 :
1590 FOR I = 1 TO 7
1600 READ DN$(I ) ,MAP$(I )
1610 FOR J = 1 TO 8
1620 READ TRK(I ,J ) ,SEC(I ,J )
1630 NEXT J ,I
1640 FOR I = 1 TO 8 : READ ZM$(1 ,I) : NEXT
1650 FOR I = 1 TO 9 : READ ZM$(2 ,I ) : NEXT
1660 FOR I = 1 TO 15 : READ J ,CH(I ) :CH$(I ) = CHR$
     (J ) : NEXT
1670 RETURN
1680 :
1690 REM * --------------------------------*
1700 REM *     Choose Dungeon and Level     *
1710 REM * --------------------------------*
1720 :
1730 HOME : VTAB 2 : HTAB 8 : INVERSE
1740  PRINT "^ULTIMA^III^DUNGEON^EDITOR^" :
     NORMAL
1750 VTAB 5 : PRINT "CHOOSE^DUNGEON^TO^EDIT^:"
1760 ZV = 8 :ZH = 8 :Z2 = 8 :ZI = 1 :GOSUB 2020 :A = ZK
1770 IF A = - 1 THEN A = 8
1780 IF A = 8 THEN POP : TEXT : HOME : PRINT
     "PROGRAM^TERMINATED." : END
1790 DN$ = ZM$(1 ,A ) : HOME
1800 VTAB 2 : HTAB 5 : PRINT DN$
1810 VTAB 4 : PRINT "CHOOSE^DUNGEON^LEVEL^
     TO^EDIT^:"
```

```
1820 ZV = 8 :ZH = 10 :Z2 = 9 :ZI = 2 : GOSUB
     2020 :B = ZK
1830 IF B = - 1 THEN B = 9
1840 IF B = 9 THEN A = 8 : GOTO 1780
1850 LN = B :TRK = TRK(A ,B ) :SEC = SEC(A ,B )
1860 RETURN
1870 :
1880 REM * ------------------------------*
1890 REM *        Cursor Bar Routine      *
1900 REM * ------------------------------*
1910 REM *                                *
1920 REM *        Values upon entry :     *
1930 REM *    ZV = Vertical top of menu   *
1940 REM *    ZH = Horizontal left of menu *
1950 REM *    Z2 = # of items in this menu *
1960 REM *    ZI = Which menu in menu array*
1970 REM *         Value upon exit :      *
1980 REM *    ZK = # of menu item selected *
1990 REM *  (i.e. 4=fourth item in menu)   *
2000 REM * ------------------------------*
2010 :
2020 Z = ZV - 1 : VTAB ZV
2030 FOR ZJ = 1 TO Z2 : HTAB ZH : PRINT ZM$(ZI
     ,ZJ ) : NEXT
2040 POKE CKB ,ZERO
2050 FOR BLOOP = 1 TO 2 STEP ZERO : INVERSE
2060 VTAB ZV : HTAB ZH : PRINT ZM$(ZI ,ZV - Z )
2070 IF PEEK (KB ) < 128 THEN 2070
2080 ZK = PEEK (KB ) : POKE CKB ,ZERO : NORMAL
2090 VTAB ZV : HTAB ZH : PRINT ZM$(ZI ,ZV - Z )
2100 IF (ZK = 141 OR ZK = 155 ) THEN ZK = (ZV -
     Z ) * (ZK = 141 ) - (ZK = 155 ) :BLOOP =
     2 : NEXT : RETURN
2110 IF (ZK = 136 OR ZK = 139 OR ZK = 138 OR ZK
     = 149 ) THEN ZV = ZV + (ZK = 138 OR ZK =
     149 ) - (ZK = 136 OR ZK = 139 ) : IF (ZV = Z
     OR ZV = Z + Z2 + 1 ) THEN ZV = (Z + Z2 ) *
     (ZK = 136 OR ZK = 139 ) + (Z + 1 ) * (ZK =
     138 OR ZK = 149 )
2120 NEXT
2130 :
2140 REM * ------------------------------*
2150 REM *    Read/Write sector of data   *
2160 REM * ------------------------------*
2170 :
2180 POKE 790 ,CODE : POKE 791 ,ZERO
2190 POKE 782 ,TRK : POKE 783 ,SEC
2200 CALL RWTS : POKE 72 ,ZERO : RETURN
2210 :
2220 REM * ------------------------------*
2230 REM *  Data for each dungeon: name,   *
2240 REM * Map Coordinates ,Trk-Sct lists *
2250 REM * ------------------------------*
2260 :
2270 DATA "DARDIN'S^PIT" , "2E-07"
2280 DATA 6 ,0 ,6 ,1 ,6 ,2 ,6 ,3 ,6 ,4 ,6 ,5 ,6
     ,6 ,6 ,7
2290 DATA "MINES^OF^MORINA" , "09-1C"
2300 DATA 6 ,9 ,6 ,10 ,6 ,11 ,6 ,12 ,6 ,13 ,6
     ,14 ,6 ,15 ,7 ,0
2310 DATA "PERINIAN^DEPTHS" , "38-06"
2320 DATA 7 ,2 ,7 ,3 ,7 ,4 ,7 ,5 ,7 ,6 ,7 ,7 ,7
     ,8 ,7 ,9
2330 DATA "ISLAND^DUNGEON" , "3A-2C"
2340 DATA 7 ,11 ,7 ,12 ,7 ,13 ,7 ,14 ,7 ,15 ,8
     ,0 ,8 ,1 ,8 ,2
2350 DATA "TIME^LORD" , "3A-1E"
2360 DATA 8 ,4 ,8 ,5 ,8 ,6 ,8 ,7 ,8 ,8 ,8 ,9 ,8
     ,10 ,8 ,11
2370 DATA "FIRES^OF^HELL" , "31-22"
2380 DATA 8 ,13 ,8 ,14 ,8 ,15 ,9 ,0 ,9 ,1 ,9 ,2 ,9
     ,3 ,9 ,4
2390 DATA "WELCOME^FOOLS" , "13-39"
2400 DATA 9 ,6 ,9 ,7 ,9 ,8 ,9 ,9 ,9 ,10 ,9 ,11 ,9
     ,12 ,9 ,13
2410 :
2420 REM * ------------------------------*
2430 REM *       Data for menus array     *
2440 REM * ------------------------------*
2450 :
2460 REM -- First menu
2470 :
2480 DATA "^DARDIN'S^PIT^" ,"^MINES^OF^
     MORINIA^"
2490 DATA "^PERINIAN^DEPTHS^" ,"^ISLAND^
     DUNGEON^"
2500 DATA "^TIME^LORD^" , "^FIRES^OF^HELL^"
2510 DATA "^WELCOME^FOOLS^" , "^EXIT^PROGRAM^"
2520 :
2530 REM -- Second menu
2540 :
2550 DATA "^1^" , "^2^" , "^3^" , "^4^" , "^5^" ,
     "^6^" , "^7^" , "^8^" , "^EXIT^PROGRAM^"
2560 :
2570 REM -- Dungeon character data
2580 :
2590 DATA 32 ,0 ,32 ,128 ,36 ,64 ,127 ,160 ,84 ,1
     ,70 ,2 ,83 ,3
2600 DATA 84 ,4 ,82 ,5 ,71 ,6 ,87 ,8 ,94 ,16 ,118
     ,32 ,37 ,48 ,47 ,192
```

Checksums

```
10    - $BADD     1310  - $82E5
20    - $9B13     1320  - $C0A7
30    - $4D3B     1330  - $D719
40    - $AD92     1340  - $7366
50    - $C899     1350  - $7140
60    - $FF65     1360  - $B426
70    - $A3BF     1370  - $44A9
80    - $A900     1380  - $0720
90    - $1A89     1390  - $9A6C
100   - $239F     1400  - $563A
110   - $A8A8     1410  - $5827
120   - $65CD     1420  - $F1F4
130   - $A8D6     1430  - $C6A2
140   - $9C85     1440  - $1CC6
150   - $139C     1450  - $3AC5
160   - $46AE     1460  - $E372
170   - $3CDA     1470  - $E074
180   - $3093     1480  - $B664
190   - $9DE8     1490  - $B0C1
200   - $FBE2     1500  - $1A5E
210   - $2DC0     1510  - $5C69
220   - $E2D2     1520  - $FBDD
230   - $A0A8     1530  - $F968
240   - $7A31     1540  - $25DC
250   - $7FA0     1550  - $01CD
260   - $F0D5     1560  - $2C7C
270   - $B57B     1570  - $D9EE
280   - $67B7     1580  - $1B34
290   - $6B18     1590  - $670E
300   - $2061     1600  - $8B10
310   - $8DCD     1610  - $9170
320   - $A99F     1620  - $BC10
330   - $980B     1630  - $5F98
340   - $8F70     1640  - $31F7
350   - $0018     1650  - $E003
360   - $680C     1660  - $A582
370   - $04D5     1670  - $0424
380   - $6964     1680  - $E12F
390   - $7320     1690  - $4E9D
400   - $FC44     1700  - $B489
410   - $DE2C     1710  - $08A2
420   - $0822     1720  - $BF5B
430   - $54D3     1730  - $7126
440   - $69B1     1740  - $1536
450   - $0D56     1750  - $2F74
460   - $E6B3     1760  - $E59E
470   - $A707     1770  - $B896
480   - $135D     1780  - $FB0B
490   - $A99C     1790  - $717E
500   - $4CBB     1800  - $32E5
510   - $1D64     1810  - $47B5
520   - $1AB2     1820  - $F809
530   - $1FB5     1830  - $24F8
540   - $4E92     1840  - $4690
550   - $7F40     1850  - $0441
560   - $8FDB     1860  - $C963
570   - $2891     1870  - $A88A
580   - $68E4     1880  - $3D5E
590   - $FB40     1890  - $F95E
600   - $1A8F     1900  - $E191
610   - $D486     1910  - $BA4E
620   - $39BD     1920  - $1950
630   - $EEAB     1930  - $4F52
640   - $C09B     1940  - $8D6B
650   - $E0B8     1950  - $68C2
660   - $4A2B     1960  - $9596
670   - $3611     1970  - $110E
680   - $9E90     1980  - $F1B1
690   - $B491     1990  - $5A66
700   - $8479     2000  - $D1E8
710   - $019F     2010  - $9F86
720   - $273D     2020  - $9DE3
730   - $6F40     2030  - $40DA
740   - $D4FA     2040  - $A56C
750   - $6DCF     2050  - $C6BE
760   - $5CF7     2060  - $507F
770   - $965B     2070  - $ABC5
780   - $12D6     2080  - $07BB
790   - $E6EB     2090  - $45A7
800   - $22DE     2100  - $8F81
810   - $AFE9     2110  - $EC36
820   - $B17F     2120  - $38CD
830   - $0C17     2130  - $5FDA
840   - $004F     2140  - $9CD0
850   - $D560     2150  - $4E62
860   - $94E1     2160  - $FCBB
870   - $1576     2170  - $71A6
880   - $AB4F     2180  - $D897
890   - $FE4E     2190  - $3D1D
900   - $230A     2200  - $05F8
910   - $B73A     2210  - $96E1
920   - $BFB9     2220  - $C3D3
930   - $98FE     2230  - $F98F
940   - $C3D0     2240  - $DD32
950   - $E582     2250  - $9CED
960   - $9F8B     2260  - $C7E1
970   - $FADD     2270  - $9D14
980   - $7239     2280  - $3A43
990   - $0805     2290  - $BECF
1000  - $445C     2300  - $CBC9
1010  - $71AB     2310  - $5612
1020  - $CABF     2320  - $DC7B
1030  - $3B8A     2330  - $D578
1040  - $ECFE     2340  - $B3C7
1050  - $3E80     2350  - $A0A8
1060  - $5841     2360  - $9FE4
```

.....continued.....

```
.....checksums continuation.....
1070  - $852C     2370  - $B7C8
1080  - $EC78     2380  - $0FDE
1090  - $48D4     2390  - $976B
1100  - $4CA3     2400  - $3C11
1110  - $AF60     2410  - $311B
1120  - $A0FE     2420  - $8BC1
1130  - $9E24     2430  - $C136
1140  - $39D3     2440  - $622D
1150  - $9DD7     2450  - $EDC9
1160  - $A03A     2460  - $516D
1170  - $5119     2470  - $E5C4
1180  - $B548     2480  - $26A2
1190  - $E033     2490  - $6033
1200  - $87E3     2500  - $2123
1210  - $07A9     2510  - $B49A
1220  - $15BA     2520  - $1E98
1230  - $E3A6     2530  - $2433
1240  - $76E6     2540  - $4368
1250  - $FE0A     2550  - $C851
1260  - $DA61     2560  - $7E5D
1270  - $553A     2570  - $52FE
1280  - $486D     2580  - $EDD3
1290  - $8769     2590  - $CF75
1300  - $C266     2600  - $C2DA
```

Ultima Encounter Editor

```
10 REM * ================================*
20 REM *      ULTIMA ENCOUNTER EDITOR     *
30 REM *           By : Jim S. Hart       *
40 REM * ================================*
50 REM
60 REM * -------------------------------*
70 REM *           Set constants         *
80 REM * -------------------------------*
90 :
100 HIMEM: 38144
110 DIM ZM$(1 ,5 ) ,TRK%(4 ) ,SEC%(4 ) ,CH$(6
    ) ,CH%(6 )
120 IF PEEK (768) + PEEK (770) <> 329 THEN
    PRINT CHR$ (4); "BLOAD^SECTOR.ZAP,
    A$300"
130 IN$ = "123456" :ZERO = 0 :CODE = ZERO
140 KB = - 16384 :CKB = - 16368 :BUF = 38143
    :RWTS = 768
150 :
160 REM * -------------------------------*
170 REM *       Load data into arrays     *
180 REM * -------------------------------*
190 :
200 GOSUB 1600 : HOME
210 INPUT " INSERT^SCENARIO^DISK^&^PRESS^
    RETURN^" ;A$
220 :
230 REM * -------------------------------*
240 REM *   Choose Dungeon level to edit  *
250 REM * -------------------------------*
260 :
270 GOSUB 1710
280 :
290 REM * -------------------------------*
300 REM *       Read Dungeon level in     *
310 REM * -------------------------------*
320 :
330 CODE = 1 : GOSUB 1840
340 :
350 REM * -------------------------------*
360 REM *       Print screen headers      *
370 REM * -------------------------------*
380 :
390 HOME : PRINT MID$ (DN$ ,2 ); "ENCOUNTER"
400 FOR I = 1 TO 19 : PRINT "-" ; : NEXT
410 PRINT "-" : PRINT
420 PRINT TAB( 13 ); "11"
430 PRINT TAB( 4 ); "12345678901"
440 PRINT : PRINT TAB( 4 );
450 :
460 REM * -------------------------------*
470 REM *    Print what was in dungeon    *
480 REM * -------------------------------*
490 :
500 FOR J = 1 TO 121 :CHAR = PEEK (BUF + J )
510 FOR I = 1 TO 6
520 IF CH%(I ) < > CHAR THEN 550
530 POKE 50 ,255 - ((CHAR = 70 ) * 192 ) : IF
    CHAR = 68 THEN FLASH
540 PRINT CH$(I ); : NORMAL :I = 6
550 NEXT I
560 IF (J / 11 ) < > INT (J / 11 ) THEN 580
570 PRINT "^" : PRINT TAB( 4 );
580 NEXT J : PRINT "^"
590 FOR I = 1 TO 11 : VTAB I + 6 : PRINT I : NEXT
600 POKE 32 ,22 : POKE 33 ,18
610 :
620 REM * -------------------------------*
630 REM *  Print rest of editing screen   *
640 REM * -------------------------------*
650 :
660 VT = 7 :HT = 4 :V = 1 :H = 1
670 HOME : VTAB 2
680 PRINT "1>WATER"
690 PRINT "2>LAVA"
700 PRINT "3>FORCE^FIELD"
710 PRINT "4>MOONGATE"
720 PRINT "5>SOLID^WALL"
730 PRINT "6>DARK^SPACE"
740 PRINT "=>SAVE^ENCNTR"
750 PRINT "ESC>QUIT^EDIT" : PRINT
760 PRINT "-----------------"
770 PRINT "USE^I-J-K-M^KEYS"
780 PRINT "TO^MOVE^CURSOR"
790 PRINT "-----------------"
800 PRINT : PRINT "^-^^WATER"
810 PRINT "^#^^LAVA"
820 PRINT "^@^^FORCE^FIELD"
830 PRINT "^" ; : FLASH : PRINT "^" ; : NORMAL
    : PRINT "^^MOONGATE"
840 PRINT "^" ; : INVERSE : PRINT "^" ; :
    NORMAL : PRINT "^^SOLID^WALL"
850 PRINT CHR$ (34 ); "^" ; CHR$ (34 );
    "^DARK^SPACE"
860 POKE CKB ,ZERO : POKE 32 ,ZERO : POKE 33
    ,40
870 :
880 REM * ================================*
890 REM *          Main Control Loop      *
900 REM * ================================*
910 :
920 FOR ALOOP = 1 TO 2 STEP ZERO
930 VTAB VT : HTAB HT
940 GET K$ : POKE CKB ,ZERO
950 :
960 REM * -------------------------------*
970 REM * Check keypress/ act accordingly *
980 REM * -------------------------------*
990 REM
1000 REM * -------------------------------*
1010 REM *       Cursor movement keys      *
1020 REM * -------------------------------*
1030 :
1040 IF K$ = "J" THEN HT = HT - 1 :H = H - 1 : IF
     H < 1 THEN H = 11 :HT = 14 :V = V - 1 :VT =
     VT - 1 : IF V < 1 THEN V = 11 :VT = 17
1050 IF K$ = "K" THEN HT = HT + 1 :H = H + 1 : IF
     H > 11 THEN H = 1 :HT = 4 :V = V + 1 :VT = VT
     + 1 : IF V > 11 THEN V = 1 :VT = 7
1060 IF K$ = "M" THEN VT = VT + 1 :V = V + 1 : IF
     V > 11 THEN V = 1 :VT = 7
1070 IF K$ = "I" THEN VT = VT - 1 :V = V - 1 : IF
     V < 1 THEN V = 11 :VT = 17
1080 :
1090 REM * -------------------------------*
1100 REM * If cursor, go back (for speed)* *
1110 REM * -------------------------------*
1120 :
1130 IF K$ = "J" OR K$ = "K" OR K$ = "M" OR K$ =
     "I" THEN NEXT ALOOP
1140 :
1150 REM * -------------------------------*
1160 REM *           Check for item        *
1170 REM * -------------------------------*
1180 :
1190 FOR I = 1 TO LEN (KEY$ )
1200 I$ = MID$ (IN$ ,I ,1 )
1210 IF I$ < > K$ THEN 1260
1220 POKE 50 ,255 - ((CH%(I ) = 70 ) * 192 ) :
     IF CH%(I ) = 68 THEN FLASH
1230 PRINT CH$(I ) : NORMAL
1240 POKE BUF + (11 * (V - 1 ) ) + H ,CH%(I )
1250 I = LEN (KEY$ )
1260 NEXT I
1270 :
1280 REM * -------------------------------*
1290 REM *           Save or exit?         *
1300 REM * -------------------------------*
1310 :
1320 REM -- Quit w/o saving?
1330 :
1340 IF K$ < > CHR$ (27 ) THEN 1390
1350 ALOOP = 2
1360 :
1370 REM -- Quit and save encounter?
1380 :
1390 IF K$ < > "=" THEN 1490
1400 TEXT : HOME : VTAB 5
1410 PRINT "WRITING^ENCOUNTER^BACK^TO^
     DISK..."
1420 CODE = 2 : GOSUB 1840
1430 ALOOP = 2
1440 :
1450 REM * -------------------------------*
1460 REM * Goto start of Main Control Loop *
1470 REM * -------------------------------*
1480 :
1490 NEXT ALOOP
1500 GOTO 270
1510 :
1520 REM * ================================*
1530 REM *           Subroutines           *
1540 REM * ================================*
1550 REM
1560 REM * -------------------------------*
1570 REM *           Read Data in          *
1580 REM * -------------------------------*
1590 :
```

```
1600 FOR I = 1 TO 4
1610 READ ZM$(1 ,I ) ,TRK%(I ) ,SEC%(I )
1620 NEXT : READ ZM$(1 ,5 )
1630 FOR I = 1 TO 6
1640 READ CH ,CH%(I ) :CH$(I ) = CHR$ (CH )
1650 NEXT : RETURN
1660 :
1670 REM * ------------------------------*
1680 REM *     Choose Dungeon and Level     *
1690 REM * ------------------------------*
1700 :
1710 HOME : VTAB 1 : PRINT : INVERSE
1720 PRINT "^ULTIMA^III^ENCOUNTER^EDITOR^" :
     NORMAL
1730 VTAB 5 : PRINT "^CHOOSE^ENCOUNTER^TO^
     EDIT^:"
1740 ZV = 8 :ZH = 8 :Z2 = 5 :ZI = 1 : GOSUB 1930
1750 IF ZK = - 1 THEN ZK = 5
1760 IF ZK = 5 THEN TEXT : HOME : END
1770 DN$ = ZM$(1 ,ZK ) :TRK = TRK%(ZK ) :SEC =
     SEC%(ZK )
1780 RETURN
1790 :
1800 REM * ------------------------------*
1810 REM *         Read/Write sector        *
1820 REM * ------------------------------*
1830 :
1840 POKE 792 ,CODE : POKE 791 ,ZERO
1850 POKE 782 ,TRK : POKE 783 ,SEC
1860 CALL RWTS : POKE 72 ,ZERO : RETURN
1870 :
1880 REM * ------------------------------*
1890 REM * Cursor bar routine adapted from *
1900 REM *  B. Kersey & Beagle Bros gang   *
1910 REM * ------------------------------*
1920 :
1930 Z = ZV - 1 : VTAB ZV
1940 FOR ZJ = 1 TO Z2 : HTAB ZH : PRINT ZM$(ZI
     ,ZJ ) : NEXT
1950 POKE CKB ,ZERO
1960 FOR BLOOP = 1 TO 2 STEP ZERO : INVERSE
1970 VTAB ZV : HTAB ZH : PRINT ZM$(ZI ,ZV - Z )
1980 IF PEEK (KB ) < 128 THEN 1980
1990 ZK = PEEK (KB ) : POKE CKB ,ZERO : NORMAL
2000 VTAB ZV : HTAB ZH : PRINT ZM$(ZI ,ZV - Z )
2010 IF (ZK = 141 OR ZK = 155 ) THEN ZK = (ZV -
     Z ) * (ZK = 141 ) - (ZK = 155 ) :BLOOP =
     2 : NEXT : RETURN
2020 IF (ZK = 136 OR ZK = 139 OR ZK = 138 OR ZK
     = 149 ) THEN ZV = ZV + (ZK = 138 OR ZK =
     149 ) - (ZK = 136 OR ZK = 139 ) : IF (ZV = Z
     OR ZV = Z + Z2 + 1 ) THEN ZV = (Z + Z2 ) *
     (ZK = 136 OR ZK = 139 ) + (Z + 1 ) * (ZK =
     138 OR ZK = 149 )
2030 NEXT BLOOP
2040 :
2050 REM * ==============================*
2060 REM *           Program data          *
2070 REM * ==============================*
2080 REM
2090 REM * ------------------------------*
2100 REM *  Encounter menu name, trk-sct#  *
2110 REM * ------------------------------*
2120 :
2130 DATA "^ROD^OF^MARKS^" ,1 ,15
2140 DATA "^TIME^LORD^" ,2 ,0
2150 DATA "^FOUNTAIN^" ,2 ,1
2160 DATA "^SHRINE^" ,2 ,2
2170 DATA "^EXIT^PROGRAM^"
2180 :
2190 REM * ------------------------------*
2200 REM *    Items (character , value )    *
2210 REM * ------------------------------*
2220 :
2230 DATA 45 ,0 ,35 ,64 ,64 ,66 ,32 ,68 ,32 ,70
     ,32 ,72
```

checksums

```
10   - $BADD     1130 - $2647
20   - $9B13     1140 - $8BFB
30   - $4D3B     1150 - $0F4F
40   - $AD92     1160 - $9324
50   - $C899     1170 - $DC27
60   - $FF65     1180 - $1B8D
70   - $A3BF     1190 - $0C00
80   - $A900     1200 - $A968
90   - $1A89     1210 - $A7AB
100  - $239F     1220 - $485F
110  - $CC3E     1230 - $DDF6
120  - $94AD     1240 - $DA6C
130  - $DE1D     1250 - $7136
140  - $2D22     1260 - $C77A
150  - $A0A7     1270 - $42A1
160  - $09A7     1280 - $8FD4
170  - $69BB     1290 - $6ED6
180  - $7D50     1300 - $1484
190  - $3A17     1310 - $7E67
200  - $F14D     1320 - $11D0
210  - $7773     1330 - $E290
220  - $4D86     1340 - $09E6
230  - $DC13     1350 - $E56A
240  - $CDCB     1360 - $9835
250  - $D53F     1370 - $F3D7
260  - $43E9     1380 - $B173
270  - $CCC2     1390 - $CD74
280  - $F258     1400 - $5B1C
290  - $9007     1410 - $3CFA
300  - $9B48     1420 - $5778
310  - $D8C8     1430 - $B7BC
320  - $1094     1440 - $461B
330  - $1149     1450 - $55B5
340  - $8D59     1460 - $A385
350  - $9D40     1470 - $CBDA
360  - $B855     1480 - $F8BF
370  - $B9F7     1490 - $5E6B
380  - $2B69     1500 - $03CA
390  - $6D4B     1510 - $503C
400  - $56B8     1520 - $362C
410  - $08C2     1530 - $7EE3
420  - $6686     1540 - $3295
430  - $6B1B     1550 - $D414
440  - $BFBE     1560 - $394F
450  - $8EC6     1570 - $8EC9
460  - $2E1C     1580 - $90B5
470  - $F4B1     1590 - $0CD6
480  - $E52E     1600 - $884D
490  - $2E90     1610 - $7082
500  - $4BBE     1620 - $D868
510  - $C375     1630 - $2112
520  - $EAD0     1640 - $8FCA
530  - $87AE     1650 - $0161
540  - $89C1     1660 - $281A
550  - $5EB6     1670 - $972C
560  - $6E32     1680 - $D924
570  - $96E8     1690 - $A953
580  - $7DBA     1700 - $F2AA
590  - $C11B     1710 - $AA89
600  - $4103     1720 - $A384
610  - $B0A9     1730 - $F897
620  - $8A17     1740 - $42C7
630  - $7480     1750 - $CF56
640  - $973F     1760 - $439F
650  - $6CFB     1770 - $EC52
660  - $8554     1780 - $1434
670  - $F00A     1790 - $89B4
680  - $3BA2     1800 - $B30F
690  - $073A     1810 - $5C92
700  - $29DB     1820 - $F5E0
710  - $BB1B     1830 - $67A4
720  - $1482     1840 - $8806
730  - $C433     1850 - $A112
740  - $92B4     1860 - $2E14
750  - $5C90     1870 - $3B1F
760  - $3D84     1880 - $64AB
770  - $F7C4     1890 - $1089
780  - $1052     1900 - $2272
790  - $A770     1910 - $CD47
800  - $B4B5     1920 - $042D
810  - $A675     1930 - $D934
820  - $5DD4     1940 - $65E9
830  - $234E     1950 - $DA44
840  - $7E31     1960 - $0763
850  - $73B1     1970 - $8860
860  - $8091     1980 - $885D
870  - $7043     1990 - $9C00
880  - $BA34     2000 - $1B4C
890  - $6929     2010 - $9B5F
900  - $600F     2020 - $949D
910  - $36BB     2030 - $3BA5
920  - $CA5D     2040 - $3E0B
930  - $8442     2050 - $7904
940  - $258C     2060 - $4A5A
950  - $7180     2070 - $9C20
960  - $BD8F     2080 - $662D
970  - $C67C     2090 - $A734
980  - $593E     2100 - $AAF0
990  - $C348     2110 - $3E50
1000 - $6521     2120 - $EEC7
1010 - $F0F4     2130 - $EF3F
1020 - $E184     2140 - $67CC
1030 - $B47B     2150 - $DC50
1040 - $7B30     2160 - $C357
1050 - $BE47     2170 - $AD20
1060 - $F897     2180 - $F606
1070 - $BFD5     2190 - $30A8
1080 - $A99C     2200 - $4E0D
1090 - $144D     2210 - $67A8
1100 - $51BD     2220 - $AEB0
1110 - $2135     2230 - $00EA
1120 - $5926
```

Sector Zap

```
0300: A9 03 A0 0A 20 D9 03 60   $6D5D
0308: 00 00 01 60 01 00 00 00   $37AD
0310: 1B 03 00 95 00 00 01 00   $08D4
0318: 00 60 01 00 01 EF D8 00   $C3C5
0320: 00                        $4343
```

Softkey for.....

Shadowkeep

Trillium Corp.

by Christopher Dean

■ **Requirements:**

- ☐ Apple][+, //e, or //c with 64K memory
- ☐ 9 blank disk sides
- ☐ 1 temporary disk side
- ☐ A sector editor
- ☐ Two hours of spare time

Shadowkeep is a large, highly detailed fantasy role-playing game in which you must traverse the many corridors of the gigantic Shadowkeep. Your object is to rescue the wizard Nacomedon who has been captured by the evil demon Dal'brad.

Shadowkeep is one of the best-designed adventure games I have seen. It is written in an adventure gaming language known as Ultra][and uses hi-res graphics and animation. Unfortunately, its protection is also one of the best designed and most complex protection schemes.

The Protection

Each sector's address information is in a different order. The normal order is the volume #, track #, sector #, and finally the checksum. On *Shadowkeep*, however, the track is listed first, followed by the sector, volume number ($5B), and the checksum.

Address Epilogues

disk A	B7 BF
disk B	97 BF
disk C	9A BF
disk D	9B BF
save-game/character disk	9D BF

The first byte of the address epilogue is used to determine which side is in the drive. The byte is read off the disk each time a sector is read and is indexed with a data table. A $B7 yields a $00, $97 a $01, $9A a $02, $9B a $03, and $9D a $04. These are the numbers of the sides.

The address field is further protected. On even tracks, the prologue is the normal D5 AA 96, but on odd tracks it is D4 AA 96.

Shadowkeep can read the disk by performing a Logical Shift Right (LSR which divides the value in half). Thus, no matter what track is under the read head, a value of $6A will be returned for a $D5 or a $D4.

The data prologue is protected in the same way as the address prologue but has some added changes. The prologue is followed by nine extra bytes that are not part of the data.

These bytes are a **false address field**. The first three bytes are D5 (or D4) AA 96, followed by the track number, sector number and volume number ($00, except on the save game/character disk it is $80). The data epilogue is BF AA. Both the address field and data field have invalid checksums.

On the disk, the sectors are stored in sequential order instead of the normal sector-skewing. Refer to ***Beneath Apple DOS*** by Quality Software for more information on sector-skewing.

Also, the sector number in the false address field is not the same as the real sector number, but is written backwards from normal (except on the save game/character disk every sector is a $0F).

Table 1: Sector-skewing

False	Normal	Shadowkeep Address Field
0	0	0
7	1	8
E	2	1
6	3	9
D	4	2
5	5	A
C	6	3
4	7	B
B	8	4
3	9	C
A	A	5
2	B	D
9	C	6
1	D	E
8	E	7
F	F	F

Furthermore, each sectors volume is A5 EF ($5B), where the $A5 is a sync byte.

The program checks for these two bytes because they are written differently on a deprotected copy and would not be in the correct location.

Shadowkeep also uses a timing routine to check the $A5 to make sure it is a sync byte.

On track zero of the boot side, the track is only half *Shadowkeep* format.

Sectors $0, and $7 through $E are in a normal format with variable address epilogues, a volume of $4B, invalid data checksums and a BF AA for the data epilogue.

On the remaining sides, sector zero on track zero is the only sector in a normal format, with a volume of $4B, address epilogue of AA EB, data epilogue of BF AA, and invalid data checksums.

The game is protected still further. The Read-Write Translate tables have been changed so data will be written and read differently than normal.

Thus the disk is encoded and would be garbage when read by a normal RWTS. While this is not really a form of protection, it keeps the disk protected from prying eyes, makes it impossible to edit characters or perform sector edits of any kind.

On track zero of the boot side, the normal sectors are encoded in two ways.

The first way is after the sectors are loaded into memory they are decoded by Exclusively ORing (EOR) the first byte of the sector with a given value to get the byte wanted. Then, this result is EORed with the next byte to get the value wanted and so on.

The second way is upon disk bootup, Boot ROM creates a translate table at $356-$3FF which is used to convert the 342 disk bytes to 256 data bytes.

Boot 1 (sector zero) increases each byte of this table by one starting at $36C and ending at $3D5, so that the sectors cannot be sector-edited without extreme difficulty. These sectors are very important because they are *Shadowkeep*'s DOS and protection routines.

Shadowkeep has yet a few more protection schemes. It performs a nibble-count on track zero upon bootup and performs a check for a modified ROM.

Also, the program checks to see if any changes were made to the program's DOS. This DOS check is used by the 'initialize save-game/character disk' routine and the 'copy disk sides' routine.

Each of the above routines displays a "This is an invalid system" message when the routine is used.

To play the deprotected *Shadowkeep*, these functions could not be used even if they worked so that protection routine will not be removed.

Disk Operations

Shadowkeep has no DOS, no CATALOG track, and no RWTS to speak of. It only has certain parts of the RWTS that access the disk.

This semi-RWTS is contained in the last five pages loaded in by BOOT1. These are decoded and stored in memory at $130—$730. As you can see, these routines are stored in the stack, input buffer, and the text screen, making the code difficult to investigate.

Table 2 lists the major parts of *Shadowkeep*'s DOS:

Table 2

$14D—$223	*Sector read/write routine and parameter set-up*
$273—$298	*Pre-nibble routine*
$299—$2B6	*Post-nibble routine*
$2B7—$30E	*Seek track routine*
$30F—$326	*Disk arm move delay table*
$327—$337	*Arm delay subroutine*
$3A8—$411	*Write Translate table*
$3EA—$453	*Read Translate table*
$533—$5DF	*Write sector with address field and data field*
$5F1—$5FA	*Write a byte subroutine*
$5FB—$663	*Read data field with sector data*
$6D0—$6DF	*Sector Skewing table*
$700—$7FF	*Primary Data Buffer*
$800—$855	*Secondary Data Buffer*

The Boot Process

Upon bootup, the boot ROM first loads track zero, sector zero into memory at $800. The first few instructions of BOOT1 decodes the rest of the sector by using the EOR command (mentioned earlier) and stores the decoded code at $80-$16D.

The boot then jumps to $A0 where the program checks for a modified ROM.

Next, the code sets all the DOS page three vectors and the ROM vectors. The vectors include the reset vector, break vector, non-maskable interrupt vector, and maskable interrupt vector. When finished, the code erases memory from $2000-$6000 and sets the stack pointer at $FF so the stack does not interfere with the boot code.

Now, the code loads in the eight normal format sectors off of track zero. When finished, the sector $0E is decoded and stored at $A000.

The boot then jumps to $A000 which decodes sector $0D at $A100. The remaining sectors are decoded at $130-$730. At this point, the boot is almost finished. The stack pointer is set to $2F and a nibble-count is performed. Then the next three tracks are read in.

The next portion of the code is the protection against boot-tracing. The code erases memory from $6680 to $BFFF, which erases moved up code used in a boot trace. Thus, there will be a sudden break in the boot and program flow will suddenly *be transferred to the reboot routine via the break vector.*

Finally, the boot moves the disk arm over track 22 and jumps to $860 which is the main program.

The Deprotection Procedure

1 **Make *Shadowkeep* copy itself.**

The first thing that must be done before the deprotection process can begin is to use *Shadowkeep* to make protected copies of three of its disk sides. The copies are slightly different than the originals. These protected copies must therefore be deprotected.

So, use *Shadowkeep* and copy sides B-D.

2 **Make a save-game disk by saving a character.**

For a save game/character disk all that is needed is a blank disk with a volume of $04. However, when trying to restore saved characters, garbage will be printed under your characters. To avoid this annoyance, make a save game/character disk with *Shadowkeep* and save one of the characters onto the disk.

3 **Use my SHADOW.COPY**

Using the copy program included with this article, copy the protected *Shadowkeep* copies sides B-D and the save game/character disk by selecting choices 2 through 5 from the menu.

Write-protect all these sides.

4 Next, copy side A by selecting choice 1 from the menu of the copy program. The program will only copy half of track zero. The rest must be obtained by performing a boot trace.

5 First, Insert side A in drive one and move Boot ROM down in memory so it can be altered:

```
CALL -151
9600<C600.C6FFM
```

6 Alter BOOT0 to break after loading in the first sector; then execute the boot.

```
96F8:4C 59 FF
9600G
```

7 Now move BOOT1 up in memory so it will not be erased upon bootup and alter the code to break after the sector data has been decoded; then execute the boot.

```
9500<800.8FFM
950F:4C 59 FF
96F8:4C 01 95
9600G
```

8 The decoded data of BOOT1 is now at $80-$16D. The code needs to be moved up and altered.

```
9380<80.16DM
9463:4C 59 FF
950F:4C A0 93
```

9 Take out the jump to reboot vector and execute the boot

```
93C4:FF
93C6:59
9600G
```

10 On a //c, the boot will not break and a "check disk drive" message will be displayed but it does not matter. If the boot did not break then hit reset and enter the monitor.

```
(CTRL) RESET
CALL -151
```

11 Now the code of BOOT2 is in memory at $900-$10FF. However, this code is encoded and thus must be decoded and moved up to a safe place so a disk can be booted and the code saved. Type in and execute the following routine to decode the sectors:

```
300: A9 69     LDA #$69
302: A2 00     LDX #$00
304: 5D 00 09  EOR $900,X
307: 9D 00 60  STA $6000,X
30A: CA        DEX
30B: D0 F7     BNE $304
30D: A9 CD     LDA #$CD
30F: A2 00     LDX #$00
311: 5D 00 0A  EOR $A00,X
314: 9D 00 61  STA $6100,X
317: E8        INX
318: D0 F7     BNE $311
31A: A0 06     LDY #$06
31C: 5D 00 0B  EOR $B00,X
31F: 9D 00 62  STA 6200,X
322: E8        INX
323: D0 F7     BNE $31C
325: EE 1E 03  INC $31E
328: EE 21 03  INC $321
32B: 88        DEY
32C: D0 EE     BNE $31C
32E: 60        RTS
```

```
300G
```

12 Now make the changes to the DOS listed in table 1, so the copy will read and write normally and ignore any disk checks.

Ignore volume and sync byte check

```
6600:18
```

Ignore reading false address field

```
66EE:4C 27 06
```

Ignore BF AA epilog

```
6732:18
```

Put track, sector, volume, and checksum values in correct location; ignore checking address epilogue

```
677D:A9 00 85 FC EA EA A5 FF
     85 33 24 E9 A5 FE 85 FF
     A5 FD 85 FE A9 5B 85 FD
     D0 07 D5
```

Ignore nibble count

```
6116:EA EA EA
```

Write and ignore writing false address field

```
6603:4C 52 05 EA EA EA EA EA
     D5 AA AD
     EA EA EA EA EA EA EA EA
     EA EA EA EA EA EA EA EA
     EA EA EA EA EA EA EA 38
     A6 A0 8E 6B 03 BD 8D C0
     BD 8E C0 10 03 4C DC 05
     AD 00 08 85 F7 A9 FD 9D
     8F C0 1D 8C C0 48 68 EA
     A0 04 48 68 20 F2 05 88
     D0 F8 AD C7 06 20 F2 05
     A9 AA 20 F1 05 A9 AD 20
     F1 05
```

Write data epilogue

```
669B:DE
```

Ignore volume and sync byte check

```
6041:EA EA EA
```

Read DE AA data epilogue

```
672F:DE
```

Ignore decoding the sectors

```
6000:EA EA
6004:BD
600F:BD
```

13 Now the sectors are ready to be written to the copy. Boot a blank disk with a normal DOS, enter the monitor, type in the following code, and insert the copy of Side A in drive one. The code sets up the RWTS to write the sectors to disk. So, when finished, execute the routine.

```
C600G
CALL -151
300:A9 01 8D E8 B7 8D EA B7
    A9 60 8D E9 B7 A9 00 8D
    EB B7 8D EC B7 8D F3 B7
    8D F0 B7 A9 02 8D F4 B7
    A9 5F 8D F1 B7 A2 0E 8E
    ED B7 8E 40 03 EE F1 B7
    A9 B7 A0 E8 20 B5 B7 AE
    40 03 CA E0 06 D0 E8 60

300G
```

14 Now the sectors just written must be sector edited so the read and write translate tables will be correct. Sector edit the copy of Side A.

Track	Sector	Byte(s)	From	To
$00	$09	$50	$13	$00
$00	$09	$58	$15	$05
$00	$09	$61	$27	$08
$00	$09	$68	$37	$0C
$00	$09	$71	$00	$13
$00	$09	$74	$05	$15
$00	$09	$85	$31	$1B
$00	$09	$A3	$3B	$2C
$00	$09	$A8	$1B	$31
$00	$09	$B0	$0C	$37
$00	$09	$B5	$2C	$3B
$00	$0A	$78	$B7	$96
$00	$0A	$7D	$BA	$9E
$00	$0A	$80	$DE	$A7
$00	$0A	$84	$F6	$AE
$00	$0A	$8B	$96	$B7
$00	$0A	$8D	$9E	$BA
$00	$0A	$93	$EE	$CB
$00	$0A	$9F	$A7	$DE
$00	$0A	$A4	$FB	$E9
$00	$0A	$A9	$CB	$EE
$00	$0A	$AF	$AE	$F6
$00	$0A	$B3	$E9	$FB

15 The last step is to rewrite the boot-sector. Sector edit track zero, sector zero and enter the following hex dump starting at byte $01:

```
1:A0 00 B9 00 08 99 71 00
  C8 D0 F7 4C 83 00 4C 59
  FF A9 00 A0 80 8C F2 03
  8D F3 03 49 A5 8D F4 03
  A2 60 8E 01 08 A2 FF 9A
  A9 00 85 00 A6 00 E0 08
  B0 16 BD B2 00 85 3D A6
  2B 20 5C C6 E6 00 D0 EC
  02 04 06 08 0A 0C 0E 01
  A2 00 BD 00 09 9D 00 A0
  E8 D0 F7 20 E2 F3 2C 52
  C0 4C 00 A0
```

16 Write-protect Side A. You now have a deprotected copy of *Shadowkeep*.

To play the game, copy all the sides instead of the master deprotected ones (Do not write-protect these copies).

To copy them, use a whole track copier like *Locksmith* Fast Copy. You can also use Super IOB but you must format the disk with the appropriate volume number:

Side A - Volume $00
Side B - Volume $01
Side C - Volume $02
Side D - Volume $03
Save game.character disk - Volume $04

17 Remember, the 'copy disk' and 'initialize save game/character disk' functions will not work. Also, occasionally the boot side will not boot correctly on a warmstart (CTRL RESET). If this happens, turn the computer off and then on again with the disk in the drive.

About The Copy Program

Simply put, the copy program ignores the address field. It reads in the data field and uses the false address field to find the correct sector. This is done by skewing the value found with *Shadowkeep*'s normal skewing. (See Skewing Table 1)

Then the sectors are skewed back to normal and written to disk. The program uses the RWTS to seek tracks and write sectors.

The program SH1.OBJ is similar to SH.OBJ but is written differently so the save game/character disk can be copied. (It does not use the false address field).

To save these programs on your temporary blank disk, type:

```
SAVE SHADOWKEEP.COPY
BSAVE SH.OBJ,A$1400,L$11D
BSAVE SH1.OBJ,A$1400,L$127
BSAVE TRANSLATE TABLE,A$6096,L$6A
```

MAY SHADOWKEEP REST IN PEACE!

SHADOWKEEP.COPY

```
10 TEXT :HOME : INVERSE :VTAB 7 :HTAB 12 :
   PRINT "SHADOWKEEP"COPY" :NORMAL :VTAB 9
   :HTAB 8 :PRINT "[1]"COPY"SIDE"A"
20 VTAB 11 :HTAB 8 :PRINT "[2]"COPY"SIDE"B"
   :VTAB 13 :HTAB 8 :PRINT "[3]"COPY"
   SIDE"C" :VTAB 15 :HTAB 8 : PRINT "[4]"
   COPY"SIDE"D"
30 VTAB 17 :HTAB 8 :PRINT "[5]"COPY"SAVE"
   GAME"DISK"
40 VTAB 19 : HTAB 8 : PRINT "CHOOSE:"" ; : GET A$
   :S = VAL (A$ ) :SS = S - 1 : IF S < 1 OR S > 5
   THEN CALL 65338 : GOTO 40
50 IF S = 5 THEN 150
70 PRINT CHR$ (13 ) :PRINT CHR$ (4 ) "BLOAD"
   SH.OBJ" :SK = 5303 :WR = 5350
80 HOME :HTAB 12 :PRINT "1"OR"2"DRIVES?"";:GET
   A$ :A = VAL (A$ ) : IF A < 1 OR A > 2 THEN CALL
   65338 : GOTO 80
90 DR = A :HOME :VTAB 10 :HTAB 9 :PRINT
   "INSERT"DUPLICATE"IN"DRIVE"" DR :VTAB 12
   :HTAB 10 :PRINT "PRESS"ANY"KEY"TO"
   CONTINUE" :WAIT -16384,128 :POKE -16368,0
110 HOME :VTAB 11 :FLASH :HTAB 14 :PRINT
   "INITIALIZING" :NORMAL :PRINT CHR$ (13)
   :PRINT CHR$ (4)"INIT"HELLO,D" DR",V"SS
120 HOME
130 TR = 1 :BUF = 5376 :Z = 0 :NT = 7 : GOTO 170
150 PRINT CHR$(13) :PRINT CHR$ (4) "BLOAD"
   SH1.OBJ" :SK = 5305 :WR = 5352 :HOME :HTAB
   12 :PRINT "1"OR"2"DRIVES?"";:GET A$ :A =
   VAL (A$ ) : IF A < 1 OR A > 2 THEN CALL 65338
   : GOTO 150
160 GOTO 90
170 IF DR = 1 THEN VTAB 11 :HTAB 2 :PRINT
   "INSERT"SHADOWKEEP"COPY"DISK"IN"DRIVE"1"
   :VTAB 13 :HTAB 14 :PRINT "PRESS"ANY"KEY"
   :WAIT -16384 ,128 :POKE -16368 ,0
190 PRINT CHR$ (13 ) : PRINT CHR$ (4 )
   "BLOAD"TRANSLATE"TABLE,D1" : FOR X = 0 TO
   105 :Y = PEEK (24726 + X ) : POKE 47766 + X
   ,Y : NEXT X
200 HOME : POKE 47082 ,DR : VTAB 10 : HTAB 9 :
   PRINT "INSERT"SIDE"" CHR$ (192 + S )
   ""IN"DRIVE"1"
210 VTAB 12 : HTAB 9 : PRINT "PRESS"ANY"KEY"
   TO"CONTINUE" : WAIT - 16384 ,128 : POKE -
   16368 ,0 : HOME : GOSUB 530
220 REM * READ TRACK ZERO *
230 VTAB 9 : HTAB 16 : FLASH : PRINT "READING" :
   NORMAL : IF DR = 1 THEN VTAB 11 : HTAB 13 :
   PRINT DR; : HTAB 32 : PRINT DR
240 IF DR = 2 THEN VTAB 11 : HTAB 13 : PRINT "1"
   ; : HTAB 32 : PRINT DR
250 IF N = 1 THEN 380
260 IF S = 1 THEN X = 2 : GOTO 280
270 X = 1
280 POKE 2 ,0 : CALL SK : FOR L = 1 TO 15 STEP X :BUF
   = BUF + 256 : VTAB 13 : HTAB 13 : PRINT ""
   ; : HTAB 13 : PRINT W; : HTAB 33 : PRINT "";
   :HTAB 33 :PRINT L
290 POKE 0 ,0 :POKE 1 ,0 :POKE 3 ,L :POKE 4 ,BUF
   - INT (BUF/256) *256 :POKE 5 ,INT
   (BUF/256) :CALL 5120 :NEXT L :BUF =5376 : IF
   DR =1 THEN VTAB 15 :HTAB 6 :PRINT
   "INSERT"DUPLICATE"IN"DRIVE"ONE" : VTAB 17
   :HTAB 14 :PRINT "PRESS"ANY" KEY"
300 IF DR = 1 THEN WAIT - 16384 ,128 : POKE - 16368
   ,0 : VTAB 15 : HTAB 1 : PRINT SPC( 38 ) : VTAB
   17 : HTAB 1 : PRINT SPC( 38 )
320 POKE 47082 ,DR : VTAB 9 : HTAB 16 : FLASH :
   PRINT "WRITING" : NORMAL : FOR L = 1 TO 15
   STEP X :BUF = BUF + 256 : VTAB 13 : HTAB 13
   : PRINT "" ; : HTAB 13 : PRINT W; : HTAB 33
   : PRINT "" ; : HTAB 33 : PRINT L
330 POKE 2 ,0 : POKE 3 ,L : POKE 4 ,BUF - INT (BUF
   / 256 ) * 256 : POKE 5 , INT (BUF / 256 )
   : CALL WR : NEXT L :BUF = 5376
```

```
340 VTAB 9 : HTAB 16 : FLASH : PRINT "READING"
    : NORMAL
350 IF DR = 1 THEN VTAB 15 : HTAB 6 : PRINT
    "INSERT^ORIGINAL^IN^DRIVE^ONE" : VTAB 17
    : HTAB 14 : PRINT "PRESS^ANY^KEY" : WAIT
    - 16384 ,128 : POKE - 16368 ,0
360 IF DR = 1 THEN VTAB 15 : HTAB 1 : PRINT
    SPC(38) :VTAB 17 :HTAB 1 :PRINT SPC(38)
380 N = 1 : FOR W = TR TO (TR + NT ) : POKE 2 ,W :
    CALL SK
390 FOR L = 0 TO 15 :BUF = BUF + 256 : VTAB 13 :
    HTAB 13 : PRINT "^^" ; : HTAB 13 : PRINT W;
    : HTAB 33 : PRINT "^^" ; : HTAB 33 : PRINT L
400 POKE 0 ,0 : POKE 1 ,0 : POKE 3 ,L : POKE 4
    ,BUF - INT (BUF / 256 ) * 256 : POKE 5 ,
    INT (BUF / 256 ) : CALL 5120
410 IF PEEK (1 ) = 1 THEN HOME : VTAB 12 : HTAB
    16 : FLASH : PRINT "DISK^ERROR" : NORMAL
    : END
420 NEXT L : NEXT W :BUF = 5376 :TR = W - 1
430 VTAB 9 : HTAB 16 : FLASH : PRINT "WRITING"
    : NORMAL
440 POKE 47082 ,DR : IF DR = 1 THEN VTAB 15 :
    HTAB 6 : PRINT "INSERT^DUPLICATE^IN
    ^DRIVE^ONE" : VTAB 17 : HTAB 14 : PRINT
    "PRESS^ANY^KEY" : WAIT - 16384 ,128 :
    POKE - 16368 ,0 : VTAB 15 : HTAB 1 : PRINT
    SPC( 38 ) : VTAB 17 : HTAB 1 : PRINT SPC(
    38 )
460 TK = TR : FOR W = (TR - NT ) TO TR : FOR L = 0
    TO 15 :BUF = BUF + 256 : VTAB 13 : HTAB 13
    : PRINT "^^" ; : HTAB 13 : PRINT W; : HTAB
    33 :PRINT "^^"; :HTAB 33 :PRINT L
470 POKE 2 ,W : POKE 3 ,L : POKE 4 ,BUF - INT
    (BUF / 256 ) * 256 : POKE 5 , INT (BUF /
    256 ) : CALL WR : NEXT L : NEXT W :Z = Z + 1
    : IF Z = 4 THEN NT = 1 :TR = TK + 1 :BUF =
    5376 : IF DR = 1 THEN GOSUB 550 : GOTO 230
480 IF Z = 4 THEN 230
490 IF NT = 1 THEN HOME :HTAB 17 :PRINT "ALL
    ^DONE" :VTAB 3 :PRINT "TO^USE^THIS^COPY
    ^PROGRAM^AGAIN, ^YOU^MUST" :VTAB 4 :PRINT
    "REBOOT^THE^SHADOWKEEP^COPY^DISK^AND
    ^RE-" :VTAB 5 :PRINT "RUN^THE^COPY
    ^PROGRAM." : END
500 IF DR = 1 THEN TR = TK + 1 :BUF = 5376 :
    GOSUB 550 : GOTO 230
510 TR = TK + 1 :BUF = 5376 : GOTO 230
530 VTAB 3 :HTAB 13 :INVERSE :PRINT
    "SHADOWKEEP^COPY" :NORMAL :VTAB 9 :HTAB
    5 :PRINT "-ORIGINAL-^^^^^^^^-DUPLICATE-"
540 VTAB 11 : HTAB 6 : PRINT "DRIVE:" ; : HTAB
    25 : PRINT "DRIVE:" : VTAB 13 : HTAB 6 :
    PRINT "TRACK:" ; : HTAB 25 : PRINT
    "SECTOR:" : RETURN
550 VTAB 15 :HTAB 6 :PRINT "INSERT^ORIGINAL
    ^IN^DRIVE^ONE" :VTAB 17 :HTAB 14 :PRINT
    "PRESS^ANY^KEY" :WAIT -16384 ,128 : POKE
    - 16368 ,0
560 VTAB 15 : HTAB 1 : PRINT SPC( 38 ) : VTAB
    17 : HTAB 1 : PRINT SPC( 38 ) : RETURN
```

checksums

```
10   - $BB9E     290  - $F045
20   - $EFA8     300  - $740E
30   - $7E3E     320  - $F326
40   - $6B3F     330  - $AABC
50   - $256C     340  - $60A1
70   - $7C1D     350  - $1AF5
80   - $4E93     360  - $4F58
90   - $DD87     380  - $503B
110  - $2ED9     390  - $8C94
120  - $BAA3     400  - $EED4
130  - $F3B9     410  - $ABFD
150  - $B76F     420  - $3E14
160  - $FE51     430  - $A3AE
170  - $6859     440  - $E9A1
190  - $18FE     460  - $A7CF
200  - $6A43     470  - $64D1
210  - $0A9C     480  - $B001
220  - $7E15     490  - $6F77
230  - $614A     500  - $7BCC
240  - $02B3     510  - $DCA3
250  - $55F4     530  - $139B
260  - $B639     540  - $20C1
270  - $1EFC     550  - $CF57
280  - $E605     560  - $D02D
```

SH.OBJ

```
1400: A9 00 8D 2E B9 A9 BF 8D   $337D
1408: 35 B9 A2 60 BD 8E C0 BD   $62C0
1410: 8C C0 BD 8A C0 BD 89 C0   $D105
1418: A2 60 BD 8C C0 10 FB 4A   $3184
1420: C9 6A D0 F6 BD 8C C0 10   $35A4
1428: FB C9 AA D0 ED BD 8C C0   $218C
1430: 10 FB C9 AD D0 E4 BD 8C   $2739
1438: C0 10 FB 4A C9 6A D0 DA   $5436
1440: BD 8C C0 10 FB C9 AA D0   $7F90
1448: D1 A0 02 BD 8C C0 10 FB   $F240

1450: C9 96 D0 C6 A9 00 85 27   $1B16
1458: BD 8C C0 10 FB 2A 85 26   $32D3
1460: BD 8C C0 10 FB 25 26 99   $DE63
1468: 2C 00 45 27 88 10 E7 A0   $725A
1470: 56 20 FF B8 A6 2D BD A7   $C537
1478: 14 C5 03 F0 0B E6 00 A5   $D908
1480: 00 C9 20 F0 1A 4C 18 14   $49F2
1488: A5 04 85 3E A5 05 85 3F   $A564
1490: A9 00 85 26 20 C2 B8 A9   $7AC2
1498: 00 85 01 8D E8 C0 60 A9   $CDF5

14A0: 01 85 01 8D E8 C0 60 00   $63F7
14A8: 02 04 06 08 0A 0C 0E 01   $52EF
14B0: 03 05 07 09 0B 0D 0F A9   $72CB
14B8: 60 8D E9 B7 A9 01 8D EA   $1992
14C0: B7 A9 00 8D EB B7 8D ED   $FF00
14C8: B7 8D F3 B7 8D F4 B7 A5   $0BDF
14D0: 02 8D EC B7 A9 FB 8D EE   $BD1C
14D8: B7 A9 B7 8D EF B7 A0 E8   $7112
14E0: A9 B7 20 B5 B7 60 A9 02   $9E55
14E8: EA EA EA 8D F4 B7 A5 02   $529D

14F0: 8D EC B7 A6 03 BD 0D 15   $7A08
14F8: 8D ED B7 A5 04 8D F0 B7   $E57B
1500: A5 05 8D F1 B7 A0 E8 A9   $3562
1508: B7 20 B5 B7 60 00 07 0E   $E66D
1510: 06 0D 05 0C 04 0B 03 0A   $6293
1518: 02 09 01 08 0F            $0BEF
```

SH1.OBJ

```
1400: A9 00 8D 2E B9 A9 BF 8D   $337D
1408: 35 B9 A2 60 BD 8E C0 BD   $62C0
1410: 8C C0 BD 8A C0 BD 89 C0   $D105
1418: A0 FC 84 F7 C8 D0 04 E6   $28FF
1420: F7 F0 00 BD 8C C0 10 FB   $2795
1428: 4A C9 6A D0 EF BD 8C C0   $3E74
1430: 10 FB C9 AA D0 F2 A0 03   $988B
1438: BD 8C C0 10 FB C9 96 D0   $9581
1440: E7 A9 00 85 F8 BD 8C C0   $A1FC
1448: 10 FB 2A 85 F7 48 68 BD   $2AC4

1450: 8C C0 10 FB 25 F7 99 FC   $4642
1458: 00 45 F8 88 10 E5 A8 D0   $AA16
1460: 4E A0 20 88 F0 49 BD 8C   $909F
1468: C0 10 FB 4A C9 6A D0 F3   $7AE4
1470: BD 8C C0 10 FB C9 AA D0   $31E2
1478: F2 A0 09 BD 8C C0 10 FB   $2AFC
1480: C9 AD D0 E7 EA BD 8C C0   $1ECE
1488: 10 FB 88 D0 F7 A0 56 20   $FBAA
1490: FF B8 A5 FE C5 03 D0 80   $0D47
1498: A5 04 85 3E A5 05 85 3F   $21C1

14A0: A9 00 85 26 20 C2 B8 A9   $3E77
14A8: 00 85 01 BD E8 C0 60 A9   $79A8
14B0: 00 85 01 BD E8 C0 60 38   $9F8F
14B8: 60 A9 60 8D E9 B7 A9 01   $1166
14C0: 8D EA B7 A9 00 8D EB B7   $F251
14C8: 8D ED B7 8D F3 B7 8D F4   $851E
14D0: B7 A5 02 8D EC B7 A9 FB   $89BA
14D8: 8D EE B7 A9 B7 8D EF B7   $6F36
14E0: A0 E8 A9 B7 20 B5 B7 60   $2AE7
14E8: A9 02 8D EA B7 8D F4 B7   $CFD1

14F0: A5 02 8D EC B7 A6 03 BD   $8809
14F8: 0F 15 8D ED B7 A5 04 8D   $CE18
1500: F0 B7 A5 05 8D F1 B7 A0   $600A
1508: E8 A9 B7 20 B5 B7 60 00   $9B0F
1510: 07 0E 06 0D 05 0C 04 0B   $A1C3
1518: 03 0A 02 09 01 08 0F C5   $1875
1520: 01 08 0F C5 01 08 0F      $D703
```

Translate Table

```
6096: 13 01                     $A0DF
6098: 98 99 02 03 9C 04 15 06   $3B12
60A0: A0 A1 A2 A3 A4 A5 07 27   $5953
60A8: A8 A9 AA 09 0A 0B 37 0D   $8049
60B0: B0 B1 0E 0F 10 11 12 00   $0596
60B8: B8 14 05 16 17 18 19 1A   $A46F
60C0: C0 C1 C2 C3 C4 C5 C6 C7   $645F
60C8: C8 C9 CA 31 CC 1C 1D 1E   $45E7
60D0: D0 D1 D2 1F D4 D5 20 21   $7884
60D8: D8 22 23 24 25 26 08 28   $053E

60E0: E0 E1 E2 E3 E4 29 2A 2B   $970A
60E8: E8 3B 2D 2E 2F 30 1B 32   $2200
60F0: F0 F1 33 34 35 36 0C 38   $90BD
60F8: F8 39 3A 2C 3C 3D 3E 3F   $B3D2
```

A.P.T. for...

Shadowkeep

Trillium Corp

Use Your Sector-And Become The Of Shadowkeep's

by Christopher Dean

■ **Requirements:**

☐ A sector editor
☐ Cracked copy of Shadowkeep (*see previous article:* **Softkey for... *Shadowkeep.***)

Shadowkeep, like many fantasy role-playing games, is a game in which it is very hard to become powerful and stay alive. There are dozens of deadly creatures wandering throughout the keep that will kill you in the wink of an eye.

Because I decoded the disk when I cracked *Shadowkeep* (by correcting the Read/Write translate tables), I decided to see if I could alter my characters.

What I Found Out

The first thing I always do to cheat on a role-playing game is to search the disk for the name of my character. I found my character and it looked like it could be edited.

About Table 1

By using the trial and error method I discovered the locations of the character data listed in **Table 1**.

Using a sector editor, you can alter your characteristics, attributes, skills, and all your items and spells.

The byte values in Table 1 are offsets from the first byte of the character name. It is done in this way because the character can start anywhere in the sector (sometimes causing part of the character to be in another sector) but will start at a $x0 or a $x8 address, where *x* is a value from $00 to $0F.

Also, any byte not listed are locations that I could not discover the use for. For example,

Table I

Byte offset	Description/Valid values
$00-$0B	Name
$0F	Class (Table II)
$10	Strength ($00-$FF)
$11	Intelligence ($00-$FF)
$12	Dexterity ($00-$FF)
$13	Leadership ($00-$FF)
$14	Power-Base ($00-$FF)
$15	Power-Now ($00-$FF)
$16	Hit Points-Base ($00-$FF)
$17	Hit Points-Now ($00-$FF))
$18	Sex/Condition (Table III)
$19	Attack Percentage ($00-$FF)
$1A	Parry Percentage ($00-$FF)
$1B	Magic Percentage ($00-$FF)
$1C	Search Percentage ($00-$FF)
$1E	Open Percentage ($00-$FF)
$32	Goldens-low byte ($00-$FF)
$33	Goldens-high byte ($00-$FF)
$35-$39	Active Items (Table IV)
$3B-$52	Items/Spells (Table IV))
$53-$57	Items/Spells if no active items (Table IV)

Table II

Value	Description
$12	Runemage
$1B	Necromancer
$00	Warrior
$14	Shadowmage
$09	Monk

Table III

Value	Description
$00	Male
$01	Dead male
$02	Female
$03	Dead female
$04	Poisoned male
$05	Poisoned female
$08	Zombie

Editor Master Dangers

byte offset $1D seems to be a duplicate of the Search Percentage value, but I could not figure out if it was or why it was needed.

Byte offset $33 needs some explanation. The maximum number of Goldens that can be obtained is 32,767. If you put a $80 in this location, you will have -32513 Goldens. Values from $80-$FF cause the number of Goldens to be negative, values from $00-$7F cause the value to be positive.

Edit Your Character

First, using a sector editor with disk-search capability, search the boot side or save game/character disk of *Shadowkeep* for the name of your character. (I suggest that after you create your character, you should transfer it from the copy of your boot side to the copy of your save game/character disk and sector-edit this disk.)

Once found, use the tables I provided to edit your character. The valid value for each location is listed to the right of the description in parenthesis.

When done editing, write the sector back to the disk.

About Table II & III

Table II is a listing of your possible class values. Placing an invalid value will ruin the display when viewing your character.

Table III lists your sex and/or condition. There are three possible conditions I found, dead, poisoned, or zombie.

Finally, Table IV lists every item, weapon, armor, scroll, and spell available in the game. Byte values in parenthesis are the values for the magical version of that item.

Now you are ready to conquer *Shadowkeep*!

Table IV

Value	Description	Value	Description
$00 ($80)	No object	$40 ($C0)	Sanctuary Scroll
$01 ($81)	Goldens	$41 ($C1)	Ward Scroll
$02 ($82)	Key	$42 ($C2)	Fortress Scroll
$03 ($83)	Hammer	$43 ($C3)	Heal Scroll
$04 ($84)	Torch	$44 ($C4)	Haste Scroll
$05 ($85)	Mace	$45 ($C5)	Luminance Scroll
$06 ($86)	Axe	$46 ($C6)	Reveal Scroll
$07 ($87)	Broad Sword	$47 ($C7)	Revive Scroll
$08 ($88)	Bastard Sword	$48 ($C8)	Zombie Scroll
$09 ($89)	Great Sword	$49 ($C9)	Decay Scroll
$0A ($8A)	Tiny Sword	$4A ($CA)	Cure Scroll
$0B ($8B)	Souleater Sword	$4B ($CB)	Death Scroll
$0C ($8C)	Kinslayer Sword	$4C ($CC)	Maim Scroll
$0D ($8D)	Valkam sword	$4D ($CD)	Dissolve Scroll
$0E ($8E)	Breaker Bar	$4E ($CE)	Fear Scroll
$0F ($8F)	Medium Shield	$4F ($CF)	Slay Scroll
$10 ($90)	Great Shield	$50 ($D0)	Percieve Scroll
$11 ($91)	Quarter Staff	$51 ($D1)	Cantrip Scroll
$12 ($92)	Eldritch Staff	$52 ($D2)	Fire Scroll
$13 ($93)	Rogarth's Staff	$53 ($D3)	Moonfire Scroll
$14 ($94)	Staff of Power	$54 ($D4)	Illusion Scroll
$15 ($95)	Staff of Quiet	$55 ($D5)	Shift Scroll
$16 ($96)	Gloves of Cold	$56 ($D6)	Twilight Scroll
$17 ($97)	Salve of Aid	$57 ($D7)	Darkfire Scroll
$18 ($98)	Shadow Cloak	$58 ($D8)	Starflare Scroll
$19 ($99)	Ragged Cloak	$59 ($D9)	Flame Spell
$1A ($9A)	Culdron's Wand	$5A ($DA)	Thrust Spell
$1B ($9B)	Wand of Travel	$5B ($DB)	Fireball Spell
$1C ($9C)	Rod of Power	$5C ($DC)	Sunburst Spell
$1D ($9D)	Silver Helm	$5D ($DD)	Freeze Spell
$1E ($9E)	Gem of Darkness	$5E ($DE)	Statis Spell
$1F ($9F)	Black Crystal	$5F ($DF)	Protect Spell
$20 ($A0)	Gem of Change	$60 ($E0)	Guard Spell
$21 ($A1)	Devil's Gem	$61 ($E1)	Barrier Spell
$22 ($A2)	Funny Rock	$62 ($E2)	Sanctuary Spell
$23 ($A3)	Sun Amulet	$63 ($E3)	Ward Spell
$24 ($A4)	Amulet of Evil	$64 ($E4)	Fortress Spell
$25 ($A5)	Charlice of Awe	$65 ($E5)	Heal Spell
$26 ($A6)	Smedly's Stick	$66 ($E6)	Haste Spell
$27 ($A7)	Soggy Stick	$67 ($E7)	Luminance Spell
$28 ($A8)	Plate Mail	$68 ($E8)	Reveal Spell
$29 ($A9)	Leather Armor	$69 ($E9)	Revive Spell
$2A ($AA)	Studded Mail	$6A ($EA)	Zombie Spell
$2B ($AB)	Chain Mail	$6B ($EB)	Decay Spell
$2C ($AC)	Silver Armor	$6C ($EC)	Cure Spell
$2D ($AD)	Scale Mail	$6D ($ED)	Death Spell
$2E ($AE)	Runic Armor	$6E ($EE)	Maim Spell
$2F ($AF)	Book of Darkness	$6F ($EF)	Dissolve Spell
$30 ($B0)	Book of Opening	$70 ($F0)	Fear Spell
$31 ($B1)	Book of Notes	$71 ($F1)	Slay Spell
$32 ($B2)	Silver Rose	$72 ($F2)	Perceive Spell
$33 ($B3)	Lock Pick	$73 ($F3)	Cantrip Spell
$34 ($B4)	Ornate Ring	$74 ($F4)	Fire Spell
$35 ($B5)	Ring of Life	$75 ($F5)	Moonfire Spell
$36 ($B6)	Black Ring	$76 ($F6)	Illusion Spell
$37 ($B7)	Flame Scroll	$77 ($F7)	Shift Spell
$38 ($B8)	Thrust Scroll	$78 ($F8)	Twilight Spell
$39 ($B9)	Fireball Scroll	$79 ($F9)	Darkfire Spell
$3A ($BA)	Sunburst Scroll	$7A ($FA)	Starflare Spell
$3B ($BB)	Freeze Scroll	$7B-$7F	Nothing
$3C ($BC)	Statis Scroll		
$3D ($BD)	Protect Scroll		
$3E ($BE)	Guard Scroll		
$3F ($BF)	Barrier Scroll		

Readers Data EXchange

when writing a... letter to the editor

- Remember that your letters or parts of them may be used in the new Readers Data Exchange even if you don't address it to the RDEX editor. Correspondence that gets published may be edited for clarity, grammer and space requirements .
- Because of the great number of letters we receive and the ephemeral and unpredictable appearance of our all-volunteer staff, any response to your queries will appear only in the RDEX, so it would be more appropriate for you to present technical questions to the readers and ask for their responses which will then be placed in the RDEX section.
- Address your letters to:

COMPUTIST
RDEX Editor
PO Box 110846-K
Tacoma, WA 98411

- Although COMPUTIST can no longer purchase short softkeys and articles, please continue to contribute them but place them in a letter to the editor so that they get published in the RDEX as soon as possible.

—RDEXed

input

The INPUT column has been made a part of the RDEX (Readers Data Exchange) forum.

*Opinions expressed are not necessarily those of COMPUTIST, its volunteer editorial staff or SoftKey Publishing. **RDEX** is a published forum for all readers where the open exchange of computer information (opinions, softkeys, copy parms and general discoveries) can occur. Such information cannot be verified by the COMPUTIST editorial staff. If errors are detected, please send corrections and updates to . . . . . —RDEXed.*

Alexis Gehrt

Deprotecting some Apple IIGS programs

Most of these programs will also run on a SCSI Hard Disk!

IIGS Softkey for...

Music Studio *(ProDOS 16 1.0)*
Activision

Deprotection:

Block	Byte	From	To
$44D	$16	82 82 00	EA EA EA

Hard Disk Patch:

Block	Byte	From	To
$44D	$0D	22	AF

$AF is used to fool OMF-Relocative Loader

IIGS Softkey for...

Shanghai *(ProDOS 16 1.1)*
Activision

Deprotection and Hard Disk Patch:

Block	Byte(s)	From	To
$27D	$162	18 FB C2 30	A9 01 00 6B

IIGS Softkey for...

Hacker II *(proDOS 16 1.0)*
Activision

Deprotection and Hard Disk Patch:

Block	Byte	From	To
$3D8	$3C	22	AF
$3D8	$43	D0 06	EA EA
$456	$3C	22	AF
$456	$43	D0 06	EA EA

IIGS Softkey for...

Paintworks Plus *ProDOS 16 1.0)*
Activision

Deprotection:

Block	Byte	From	To
$291	$1E0	D0 01	EA EA
$291	$1CF	D0 12	EA EA

Hard Disk Patch: or the program will always want the master disk. With this change it accepts the copy as ok!

Block	Byte	From	To
$291	$1D8	22	AF

IIGS Softkey for...

Mean 18 (IIgs Golf)
Accolade

Deprotection (Golf.sys16):

Block	Byte	From	To
$506	$174	8D	AD

Deprotection (Architect):

Block	Byte	From	To
$29C	$174	8D	AD

A Hard Disk patch is not possible because the programmers were so silly to use direct prefixes and not modifiable ProDOS 16 prefixes, which have preceding length byte!

IIGS Softkey for...

Instant Music
Electronic Arts

Deprotection:

Block	Byte	From	To
$111	$5E	22	AF
$111	$63	F0 02	EA EA

This program works on Hard Disk without the "Key Disk Question".

Readers Data EXchange

IIGS Softkey for...

Deluxe Paint
Electronic Arts

Deprotection:

Block	Byte	From	To
$412	$165	22	AF
$412	$16A	F0 02	EA EA

This program also works on Hard Disk without the "Key Disk Question".

IIGS Softkey for...

Top Draw
Styleware

Deprotection:

Block	Byte	From	To
$394	$1E2	D0 1D	80 1A

This program works on Hard Disk without any modifying, which is very positive!

IIGS Softkey for...

816 Paint
Baudville

Paint 320 Super Deprotection:

Block	Byte	From	To
$273	$11D	30	80

Paint 640 Super Deprotection:

Block	Byte	From	To
$2DB	$DF	30	80

This program also works on Hard Disk without any modifying.

D. T.

IIGS Softkey for...

Music Construction Set GS
Electronic Arts

■ **Requirements:**

- ☐ *Copy II Plus* or simular program to copy disks
- ☐ ProDOS sector editor (I prefer ZAP on *Bag Of Tricks II*)
- ☐ Second 3.5 drive (or 800K RAMDISK)

1 First open the file to edit:

OPEN/MUSICGS/JIMSCODE

2 Read Block 0.

3 Change bytes $1CF and $1D0 from 49 20 to EA EA.

IIGS Softkey for...

Shanghai
Activision

1 First open the file to edit:

OPEN/SHANGHAI/SYSTEM/START

2 Read Block $0.

3 Change byte $DF from 01 to 00.

This change to *Shangai* also makes it run faster!

Marshal P Brown

Softkey for...

The Print Shop (Color version)
Broderbund

The new color version of *The Print Shop* is still using the nibble count routine listed in the COMPUTIST 17. The track $22 nibble count rountine has been relocated to address $083A. The cure:

1 Copy *The Print Shop* with any copier that will ignore errors on track $22.

2 Search the disk for: 20 31 08 (JSR $083A)

3 Replace the JSR with: EA EA EA. On my version I found this call on track $3, sector $E, bytes $12, $13, $14

Track	Sector	Byte(s)	From	To
$3	$E	$12-14	$20 31 08	EA EA EA

Samer M Kurdi Amman

Softkey for...

Mouse Word
Version Soft

I was pleased to see the softkey for *Mouse Calc* in COMPUTIST 45. I have another Version Soft program, *Mouse Word*, which is also protected.

I made a copy of *Mouse Word* with COPYA and searched for the bytes: A2 00 E8 D0 07 C6 06. They were found on track 0F, sector 0E. I changed the $07 to a $48 as in the *Mouse Calc* softkey and, low and behold, it worked!

Softkey for...

Spindizzy
Activision

Thanks to a letter from John Nicholson in COMPUTIST 46, I was able to deprotect *Spindizzy*, a great 3-D game by Activision. Since Activision uses the same copy protection on several of their programs, I used Mr. Nicholson's softkey on *Spindizzy*.

1 Copy *Spindizzy* with COPYA.

2 Use your sector editor and make the following changes:

Track	Sector	Byte(s)	From	To
$1C	$3	$58-$97	?	all EA's
		$98	25	A9
		$99	FC	FF

David G. Alexander

Antique Softkey for...

Ultima I

California Pacific

The disk uses modified DOS 3.2. Each side has two catalogs. The back side has normal marks. The front side has marks that change for each track and sector.

The back side has a VTOC on track $11, sector $0 and a catalog on sectors $2 and $1 which includes the graphic screens used by the demo program in file "Player Disk". These files can be transferred to a DOS 3.3 disk using Muffin or a version of *Copy II Plus* that enables designation of a disk as DOS 3.2 or DOS 3.3.

To transfer the other files, use a sector editor to read track $14, sector $0 and write it to track $11, sector $0 and use Muffin again. The disk copy program in the Applesoft "Player Disk" file will not work with DOS 3.3 and can be replaced with a suitable message.

The DOS 3.2 RWTS on the front side is modified to read the altered marks. In normal DOS 3.2 the address header is read using EOR #$D5, CMP #$AA and CMP #$B5.

In *Ultima I* boot side the header is read using [AND $48, CMP #55], CMP #$AA and [AND #3D, CMP #$35]. I don't know what is stored at $48, but assume that it is some function of the track number.

Similarly, the data header is read using [AND $48, CMP #$55], CMP #$AA and [AND #$2F, EOR #$2D]. The logical operations on the first bytes of the headers somehow make them all come out to $55.

The logical operations on the third bytes of the headers somehow make them all come out to zero. Since the operations on the third bytes use fixed numbers, no modification to the RWTS is necessary.

However, the operations on the first bytes use changing values which are stored in location $48.

To normalize the front side, I used *Copy II Plus* nibble editor to determine and record the first bytes of the address and data headers on each track.

The headers were the same throughout a particular track whereas the trailers changed from sector to sector.

The next step was to boot the front side, enter the monitor and modify the RWTS so that the recorded values could be poked into the RWTS and used with straight CMP statements.

The modifications were $B965:C9 D5 EA EA for the address header (the D5 could be EA or anything else because it is replaced by a POKEd value).

The modifications for the data header were $B907:C9 D5 EA EA.

After making the modifications the RWTS was moved down to $1900 and basaved for use in the swap controller.

The final normalization step was to modify the DOS 13 controller to include a table of data values corresponding to the recorded first bytes of the address and data headers for each track and poke statements to read the values from the data table and poke them into $B966 (address) and $B908 (data) for each track.

The front side also has two catalogs in the same places as on the back side. The one on track $11 is a dummy.

After normalizing the disk, the files can be transferred by writing track $14, sector $0 to track $11, sector $0 and using *FID* or *Copy II Plus*

Controller

```
1000 REM  ULTIMA I CONTROLLER
1001 CD = WR
1002 POKE 775 ,96 : REM IGNORE UNREADABLE
     SECTORS
1017 POKE 900 ,12
1018 DO3 = 13
1020 TK = 0
1021 LT = 35
1031 ST = 0 :T1 = TK : GOSUB 490 : REM  READ
1032 GOSUB 360 : READ A1 ,A2 : POKE 47462 ,A1
     : POKE 47368 ,A2
1060 GOSUB 430 : GOSUB 100 :ST = ST + 1 : IF ST
     < DOS THEN 1060
1090 GOSUB 490 : GOSUB 360 :TK = T1 :ST = 0 :
     REM  WRITE
1100 GOSUB 430 : GOSUB 100 :ST = ST + 1 : IF ST
     < DOS THEN 1100
1120 IF TK < (LT - 1 ) THEN TK = TK + 1 : GOTO
     1031
1140 HOME : PRINT : PRINT "COPY DONE" : END
5000 DATA 213 ,213 ,223 ,223 ,221 ,215 ,221
     ,223 ,223 ,223
5010 DATA 221 ,221 ,223 ,221 ,223 ,221 ,223
     ,223 ,223 ,221
5020 DATA 223 ,223 ,223 ,221 ,223 ,223 ,223
     ,221 ,223 ,215
5030 DATA 223 ,223 ,221 ,223 ,223 ,215 ,223
     ,223 ,221 ,223
5040 DATA 221 ,223 ,223 ,215 ,221 ,215 ,221
     ,215 ,223 ,223
5050 DATA 223 ,215 ,223 ,215 ,223 ,221 ,223
     ,223 ,223 ,215
5060 DATA 221 ,223 ,223 ,215 ,223 ,223 ,221
     ,223 ,223 ,223
10010 PRINT CHR$ (4 );"BLOAD RWTS.ULTIMA I,
     A$1900"
```

Controller Checksums

1000	$356B	1100	$922E
1001	$D329	1120	$9FC8
1002	$1A65	1140	$A5B2
1017	$669A	5000	$9D5F
1018	$6F23	5010	$EBF5
1020	$27AB	5020	$C201
1021	$DD45	5030	$26DD
1031	$58A2	5040	$1053
1032	$A08D	5050	$8EC4
1060	$AF89	5060	$1F2E
1090	$A32E	10010	$BD11

Antique Softkey for...

Chess 7.0

Odesta

The Protection

The protection comprises changing the first byte of the address headers from $D5 to $D4 for alternating sectors on each track. Also, the trailers are changed from $DE $AA to $EE $AA.

The disk is converted to standard DOS using super IOB with POKE 47444,74: POKE 47445, 201: POKE 47446, 106: POKE 47447, 208: POKE 47448, 239 to accept $D5 or $D4 and POKE 47426, 24 to ignore the altered trailers (universal Penguin controller).

The copy disk must be formatted with a volume number of $00.

The protection further comprises a Dav Holle loader similar to the one for *Masquerade* (COMPUTIST 35, p. 24) which consists of the following sections:

- *$800—$82A* boot code loader which jumps to $86B
- *$82B—$82C* variables
- *$82D—$86A* reset routine (encrypted)
- *$86B—$8F5* various setup including reset vector pointing to $110.
- *$8F6—$93C* de-encryption routine. Decodes and stores the reset routine from ($82D—$86A to $110—$14D) and the disk access and setup routines from ($93D—$BDF to $53D to $7DF). Jumps to $788.
- *$93D—$$B6D* disk-access routines (encrypted)
- *$B6E—$B87* "Copyright Dav Holle" (encrypted)
- *$B88—$BDF* various setup including filling memory blocks with $02's. Jumps to $FC58 (monitor home routine). Returns to $B48F via stack pushes at ($BAA—$BAD). (encrypted)

The data is encrypted by Exclusive ORing (EOR) blocks of code with respective bytes in other blocks of code.

The next step is to modify the loader to read standard DOS trailers. Use SREAD/SWRITE or the like to read the following sectors from track $0 into memory.

Sector	Memory
$0	$800
$E	$900
$D	$A00
$C	$B00

The Deprotection

Modify the de-encryptation routines to decode the respective blocks and save them in the same memory locations rather than relocating them and then stop execution.

```
910:9D 2D 08  reset routine
91E:9D 00 09  disk access and setup routines
92D:60        stop execution
8F6G          execute de-encryptation)
```

Modify the program to load and relocate the decoded routines rather than decoding them again.

```
90D:BD 2D 08  reset routine
910:9D 10 01  reset routine
91B:BD 00 09  disk access and setup routines
91E:9D 00 05  disk access and setup routines
92D:A5        continue execution
```

Modify the decoded disk access routines to read standard DOS trailers.

```
984:DE
9EB:DE
```

The disk access routines comprise a disk check at ($B3A—$B4F) which compares the current sector number with the current address header ($D5 or $D4). Disable the routine.

```
B4C:09
```

Write the memory pages back to the disk using SREAD/SWRITE or the like.

The final step is to disable a second disk check routine that counts the number of self-sync $FF bytes preceding each address header.

The original disk is modified to comprise exactly three of said bytes.

The routine is loaded at ($BC14—$BC2A) and encrypted on the disk at track $22, sector $3 by EORing the memory page $BC00 with page $6200 which is stored on the disk at track $17, sector $0 in unencrypted form. It appears that the routine is normally encrypted in memory and de-encrypted as required as another form of protection.

The routine loads the X-register with $FC and increments the register each time an $FF is read from the disk. If three $FF bytes are read the register will be incremented to $00 which is stored in $4E.

The disk check is disabled by the following modifications to the unencrypted routine.

$BC18—change from $FC to $00 (load the X register with $00 rather than $FC).

$BC1A—change from $E8 to $EA (disable incrementation of X register).

However, since the routine is encrypted on the disk, it is necessary to encrypt the altered bytes. The values to EOR the new bytes with are the corresponding values in page $6200.

The following sector edits will disable the encrypted routine track $22, sector $3 on the disk. Byte $18: change from $B6 to $4A (EOR the desired byte $00 with $4A from $6218 in memory) Byte $1A: change from $A2 to $A0 (EOR the desired byte $EA with $4A from $621A in memory

How about a nice deprotected game of Chess 7.0?

Still another Softkey for...

Infiltrator

Mindscape

The Protection

Track $0, sectors $0 to $9 on both sides of the disk can be read by normal DOS. However, the remaining data is scrambled on the disk using drastically altered translate tables (read - $BA96 to $BAFF, write - $BA29 to $BA68). The data checksums are also altered. There is a disk check routine on track $22, sector $E which is scrambled on the disk and loaded into page $2 of memory.

The Deprotection

To deprotect the disk, boot the original, break into the monitor, move the RWTS to $1900 (1900<B800.BFFFM) and save it to the Super IOB disk

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

Then, copy both sides of the disk using the swap controller with $B942 changed to $18 to ignore the checksum errors. The main data on the copy will be unscrambled.

Next, use a sector editor or whatever to copy track $0, sectors $0 to $9 from the original sides to the copy sides (sector $4 can be skipped). Then, copy track $0, sector $4 from a standard DOS disk to both sides of the copy so that the altered DOS on the copy will read data using the standard translate tables.

The final step is to disable the disk check routine. Edit track $22, sector $E, start byte $75 from $BD 88 C0 $4C 71 02 on both sides of the copy so that when the disk check fails the program will jump to the same instruction as when the disk check passes.

Softkey for...

Mabel's Mansion

Datamost

Thanks to Isaac de Pig for his crack of *Mychess II* in COMPUTIST 40. *Mabel's Mansion* uses a similar disk check routine and encryption scheme.

The Protection

The disk check routine is located at $A3B8 to $A42B and jumps to a reboot routine at $A448 if the check is unsuccessful.

Both routines are included in 41 ($29) pages of data which are encrypted on the disk, loaded at $7E00 to $A6FF and decoded by exclusive-ORing each byte with the preceding byte.

The de-encryptation or decoding loop is loaded at $7D22 to $7D41 and located at track $18, sector $F on the disk.

In addition, there are several jumps to the reboot routine from other parts of the program.

The Deprotection

The crack is as follows.

1 Use COPYA or whatever to make two copies of the original disk. Label one "Mabel's Mansion Cracked" and the other "Mabel's Mansion Working". The names on the labels are critical (just kidding).

2 Sector edit the "...Working" disk by changing track $18, sector $F, start byte $42 from 4C 02 7E to 4C 59 FF. This will allow the boot process to proceed through execution of the decoding loop and then jump into the monitor.

3 Boot the "...Working" disk and press the RETURN key as required until the program jumps into the monitor. The drive can be stopped by typing $C0E8.

Now the fun starts

In order to open the program to CPT's (cheat playing techniques) and the like it is desirable to save the de-encrypted code to disk.

This requires saving 41 ($29) pages of memory between $7E00 and $A6FF to the proper sectors on the disk.

I used SREAD/SWRITE from COMPUTIST 42 and wrote each sector manually, which took about 10 minutes. *Inspector* (which I don't have) would probably work too.

Persons who consider RML (repetitive manual labor) unacceptably repungent should be able to automate the procedure by writing, debugging, and verifying the results of a machine language program to perform the same function in a shorter period of time.

Since the block from $9600 to $A6FF is overwritten by booting DOS, I performed the procedure in two stages.

4 Boot a disk with DOS on it and write the 24 memory pages $7E00 to $95FF to the "Cracked" disk using the following table.

Track	Sector	Memory
18	E	7E00
18	D	7F00
18	1	8B00
18	0	8C00
19	F	8D00
19	E	8E00
19	8	9400
19	7	9500

5 Boot the "Working" disk again and type $C0E8 from the monitor to stop the drive.

6 Relocate the block from $9600 to $A6FF to lower memory to prevent it from being overwritten by DOS.

$6600<$9600.A6FFM

7 Boot the DOS disk again and write the 17 memory pages $9600 to $A6FF (relocated to $6600 to 76FF) to the "Cracked" disk using the following table.

Track	Sector	Memory
19	6	6600
19	5	6700
19	1	6B00
19	0	6C00
1A	F	6D00
1A	E	6E00
1A	7	7500
1A	6	7600

The following sector edits should be made to the "Cracked" disk.

8 To bypass the decoding loop.

Track	Sector	Start Byte	From	To
$18	$F	$22	A0 FF B9	4C 02 7E

9 To disable the disk check routine.

Track	Sector	Start Byte	From	To
$1A	$8	$1B	86 F1	EA EA
$1A	$9	$F9	86 F1	EA EA

10 To disable the jumps to the reboot routine.

Track	Sector	Start Byte	From	To
$19	$2	$74	4C 48 A4	EA EA EA
$19	$A	$78	4C 48 A4	EA EA EA
$1A	$9	$86	4C 48 A4	EA EA EA
$1A	$E	$73	4C 48 A4	EA EA EA

11 Boot your "Cracked" disk and enjoy the delights of the mansion. Be sure to tell Mabel that "Dave sent you".

Roman Drozd

Softkey for...

TelePorter

■ **Requirements:**

- ☐ Your Apple II
- ☐ Copy program that can copy selected tracks
- ☐ Sector Editor

TelePorter is a rather old communications program, but I got my hands on an original for couple of days and decided to make a back-up. I used *Essential Data Duplcator (EDD)* and *Copy II Plus* to break my copy, and it works fine, but any other equivalents will do.

The Crack

1 Disable the checking routine at the end address markers.

2 Copy the disk normally except for Track $0F.

3 Now boot your sector editor and make the following changes:

Track	Sector	Byte(s)	From	To
$00	$03	$35	ED	DE 02
	02	$E9	90	D0 02
	03	$61-62	00 BF	84 9D 0B
	0F	$12-13	D0 0B	EA EA 10
	08	$5E-5F	D0 01	EA EA

4 All done!.

The volume number on your copy must be 004 of the program will not run properly, so watch out !! You may also wish to re-format track $0F for better copies. Enjoy the crack.

Softkey for...

Megabots

Neosoft

■ **Requirements:**

- ☐ Apple II, II Plus, IIe, or IIc
- ☐ Blank disk
- ☐ A file copier
- ☐ DOS 3.3

Megabots is a keyboard-controlled graphic adventure in which you are a robot searching to find the hidden power source in a five level complex. While the graphics and sound are good, the game is rather slow, and lacks good playability. The game uses much disk access, and if it isn't backed up, the disk will wear out inside of a few months of play.

The Protection

The original is written in near DOS 3.3 format with only slight self-sync byte alterations. The actual game program does checks on DOS to see if it's written in the exact same format as the original. When you copy the disk normally, it will just hang into an endless drive spin. This suggests that the original could have been recorded with a very fast or slow drive.

The Procedure

1 Boot a DOS 3.3 disk

2 Enter the monitor:

CALL-151

3 Change a clear-carry instruction to a set carry instruction.

B942:18

4 Tell DOS to ignore any read errors.

B98A:00

5 Modify track $0's error-checking:

B99C:18 60
BE48:18

6 Back to BASIC:

3D0G

7

INIT HELLO,V254,D1

8

DELETE HELLO

10 Now run a file copier such as FID, and copy all files from the original to the INITialized disk.

11

PR#6

All finished!

You are now spared all the drive moaning and groaning of sync-track copies. Although it is deprotected, this program likes it's own DOS, and will not work with fast DOSes without some modifications to the ANAL.DROIDS files' DOS checking routine. Quite a task!

Enjoy the cracked version, and the time saved sending for a replacement! If only all wares had this protection...

Command Summary

SPACE	Select weapon
1	Fire weapon #1
2	Fire weapon #2
3	Look for power source in this room - go right one room
4	Go left one room
5	Move up one room
6	Move down one room
7	Ask robot for directions to the power source
8	Get recharge off dead robot

1 Brian A. Troha

Here are a few softkeys I would like to pass on.

First, I must have a different version of *World's Greatest Baseball Game*. It says: enhanced version.

To softkey this disk you must get it in a normal format (see COMPUTIST 33). I checked the protection code on track $0, sector $1 and found the protection code resides at $D700-D7FF and is run right after the first boot stage. When the protection passes it loads physical sector $8 in $D700-D7FF and runs it. I just loaded in physical sector $8 (DOS 3.3 sector $B) and wrote it out to sector $1 of track $0. The sector should start with "8E E9 D7 8E F7 D7". After comparing the softkey for *G.I. Joe*, a similar method should work, although the sector might have D4's instead of D7's.

Softkey for...

World's Greatest Baseball Game

Epyx

1 Boot DOS 3.3

2 Type:

```
CALL -151
B942:18
RUN COPYA
```

3 Copy BOTH sides of the program disk

4 With your sector editor (side 1, disk A), read sector $0B, track $0 and write it to sector $01, track $0

Softkey for...

Hardball

Accolade

1 Run COPYA

2 Change:

Track	Sector	Byte(s)	From	To
$0	$0	$4C-4E	20 60 B2	EA A9 00

This overwrites the call to the protection and sets the Branch EQual to pass.

Softkey for...

Rambo: First Blood part II

Mindscape

1 Boot DOS 3.3

2 Type:

```
CALL-151
B942:18
RUN COPYA
```

3 Copy the disk

4 Change:

Track	Sector	Byte(s)	From	To
$0	$0	$1B-1D	20 00 0B	8D 7B 04

See *Racter* in COMPUTIST 42 and also *American Challenge* in COMPUTIST 40. All three programs are from Mindscape and all use the same protection. The article about *American Challenge* is another way to break the same protection as both methods will work on *Rambo*, the only difference is the code is on Sector $A.

Using information from previous issues I was able to softkey the above programs. So it helps to keep (and reread) all older issues of COMPUTIST. With that I have one question: What happened to the Most Wanted List? One last thing: If any reader would like to talk "COMPUTIST" talk please send a letter to:

Brian A. Troha
2745 Alice Circle
Stoughton, WI 53589-3353

S. Todd Grant

a note on Doctor Destructions's...

Ultima IV APTs

COMPUTIST 42, Page 21

Doctor Destruction's *Ultima IV* APT's (COMPUTIST 42 page 21) should come in handy for those who don't already know them, but he or she left out a couple of details.

From Jimmying to Dispelling

First, you will need a key to (J)immy the lock of the door that leads to the dungeon entrance.

Second, by having a character stand on the **lower right hand corner** of the "treasure room" the energy field is automatically dispelled; no need for a spell. You only need Dispell if you want to kill the Balron in the next room over, to help your valor virtue, and to avoid the poison field on your way in.

By the way, I vote to keep COMPUTIST strictly Apple. Any effort to cover other computers would be fruit-less.

Readers Data EXchange

Jim S. Hart

Softkey for...

New Oregon Trail

MECC

■ **Requirements:**

- ☐ The *New Oregon Trail* original disk
- ☐ 2 blank disk sides
- ☐ Super IOB v1.5
- ☐ a sector editor

The New Oregon Trail is an educational game by MECC. Your job is to pilot a group of settlers through the old west's Oregon Trail. Along the way you must make decisions concerning the supplies to take along, when to hunt for food, and which trail to take, among others.

The game is a delight to play with the quality hi-res graphics screens that are generously sprinkled throughout it.

A few things, such as starvation and mechanical problems with the wagon, occur randomly and at times will drive you up the wall. For example, you die of starvation just 20 miles from the end even though your food supplies are well stocked. Aside from that, the game is a gem for those who have "young-uns".

The back of the package tells you that the diskette is unconditionally guaranteed but nowhere in the accompanying booklet does it say anything about backups.

I'd rather not pay extra money anyhow so I went about the task of deprotecting the disk. Not only is it cheaper than sending off for a backup, the deprotected copy is more reliable than the original and I also get the chance to do this writeup for COMPUTIST! It's a great way to get your friends to be envious since it's YOUR name that is in a nationally distributed magazine.

Onward!

The Protection

Looking through my back issues of COMPUTIST , I found a letter which said that MECC software could be backed up by using the disk's RWTS in a swap controller and then adding a normal DOS (3.3, of course).

Easy enough, I thought. An hour later I still couldn't get the darn controller to work.

The RWTS looked standard enough to use but closer inspection ruled it out. Well, I spent the next hour using my trusty nibble editor to check out the disk's format on every track. The epilogs and checksums were fine on both sides but the prologs had been altered as follows:

Table 1

Side #	Track(s)	
1:	$00	normal format
	$01 - $22	address prolog (AA D5 AD)
		data prolog (D5 96 AA)
2:	$00	normal format
	$01 - $22	address prolog (AA D5 AD)
		data prolog (D5 96 AA)

It isn't hard to install these changes into a controller. I made the required changes and ignored the first three tracks (we don't want their DOS, do we?).

After the disks had been converted to normal DOS 3.3 format, I added normal DOS to the disk and made HELLO the boot up program. Now it was time for the acid test: try out the deprotected disk.

A couple of hours of code searching and many unpleasant words later, I resigned myself to the fact that this disk needed its own DOS to operate. *Copy II Plus*' track/sector mapper showed that tracks $03-$08 were marked as in use but there were no corresponding files for those tracks.

The *Oregon Trail* DOS must be loading something important off of these tracks.

Going back and modifying the Super IOB controller so that it would now copy the whole disk, I deprotected the original once again. I knew that I would have to find where the disk's read routines were so that they could be modified to work with normal DOS disks. A search for the byte sequence 8C C0 ($C08C - the common memory reference for disk reading reversed), found the routines in track $00, sectors $06-$07.

A few sector edits later, I had a perfectly functioning backup.

To see what is contained in the BASIC program files, boot up normal DOS 3.3 and load any one you want.

Whoever wrote *Oregon Trail* had a slick ampersand package because the programs are littered with ampersand calls that merge BASIC programs, check the disk to see which side it is, and many other jobs.

Some enterprising COMPUTIST reader may want to find out where the ampersand package resides, how to save it as a BLOADable file, and find out all of its functions. That would make a great article.

1 Write-protect both sides of the original *Oregon Trail* disk.

2 The controller will deprotect both sides. Install it into Super IOB. Answer **YES** to 'Format Backup?'.

3 Boot up your favorite sector editor and make the following changes to side #1 of the copy:

Track	Sector	Byte(s)	From	To
$00	06	86	96	AA
		$8B	AA	AD
$00	07	$1F	96	AA
		$2A	AA	AD
		$83	AA	D5
		$8D	D5	AA
		$98	AD	96

4 Hide the original and enjoy your backup.

Controller

```
1000 REM OREGON TRAIL
1010 TK = 0 :LT = 1 :ST = 15 :LS = 15 :CD = WR
     :FAST = 1
1020 GOSUB 490 : GOSUB 610
1030 GOSUB 490 : GOSUB 610 : IF PEEK (TRK ) =
     LT THEN 1050
1040 TK = PEEK (TRK ) :ST = PEEK (SCT ) : GOTO
     1020
1050 TK = 1 :LT = 35 :ST = 15 :LS = 15 :CD = WR
     :FAST = 1
1060 RESTORE : GOSUB 190 : GOSUB 210
1065 GOSUB 490 : GOSUB 610
1070 GOSUB 230 : GOSUB 490 : GOSUB 610
1075 IF PEEK (TRK ) = LT THEN 1090
1080 TK = PEEK (TRK ) :ST = PEEK (SCT ) : GOTO
     1060
1090 HOME : PRINT "COPYDONE." : END
5000 DATA 170 ,213 ,173 ,213 ,150 ,170
```

Controller Checksums

```
1000  - $356B      1065  - $6210
1010  - $EA41      1070  - $E051
1020  - $3164      1075  - $002D
1030  - $5E3F      1080  - $1AAA
1040  - $3A08      1090  - $BDEB
1050  - $5B67      5000  - $49CA
1060  - $A3CE      0     - $0000
```

Softkey for...

Paul Whitehead Teaches Chess

Enlightenment

■ **Requirements:**

- ☐ *Paul Whitehead Teaches Chess* original
- ☐ 4 blank disk sides
- ☐ COPYA
- ☐ a sector editor

Paul Whitehead Teaches Chess is a chess tutor for beginning to middle level chess players. The program is well laid out and with the Road Map that comes in the package, it is easy to get to any feature that the program offers.

Also included with the tutor is a chess opponent named "The Coffeehouse Chess Monster". It's been a few years since I have played the game of kings and "The Monster" gave me quite a tussle when we played.

This package is recommended for beginners and players who think they know it all.

If your disk dies within three months of purchase, a five dollar processing fee and the blown disk will get you a backup. Complete backups (includes disk and docs) are available for $25.

While this is not a bad policy, I prefer my own: remove the protection and not send them any money when my original dies. What? This sounds like something for Computist!

The Protection

In the pamphlet with the warranty and liability statement, Enlightenment Inc. states that "We have chosen a copy protection scheme that will not harm your disk drive". Nowhere is it mentioned what the poor disk must go through each time it is booted, however.

Loading up the trusty nibble editor, you find that track $00 is normal and that the rest of the tracks are protected. The protection consists of the third byte of the data prolog being a different value on each track. No problem to fix that:

POKE 47358,0
RUN COPYA

Follow the above procedure to copy each of the four sides onto the four blank sides. The poke disables the check for the third data prolog byte which means that it now accepts any byte found there. Put the originals away now and let's concentrate on the copies.

Nonstandard data prolog

Since the originals were written with a non-standard data prolog, there must be a routine to read this format on the disk. Looking for disk access code on the copy, it shows up in two areas: track $00, sectors $0A and $0B.

After an hour of studying the code, I figured that the code on sector $0A was a protection scheme (nibble count perhaps?) and the code on sector $0B was responsible for loading the program into memory.

Disassemble the code in sector $0B and you come to a section that reads the data prolog (around bytes $40-$60).

Notice that in the check for the third byte, at bytes $58-$5A, a distinct value is not used. A memory location is used instead. Lets fix it to read a normal DOS 3.3 format:

Sector edits:

Track	Sector	Byte(s)	From	To
$00	$0B	$58 - $5A	CD 00 BD	C9 AD EA

The copy now has the ability to read itself. Booting it up reveals a nasty surprise: it loads a bit of code and then reboots.

Ah Hah! Sounds like a nibble count/signature check not being satisfied. Scanning the disk for a jump to the disk controller card (4C 00 C6) turns up nothing.

An alternative is to disable the protection scheme located in track $00, sectors $0D & $0A. Several fruitless hours were spent trying to do this.

However, if you look at track $00, sector $0A, around bytes $A0-$C0, you will notice that the RESET vector is changed and then an indirect jump is made to the ROM RESET vector at $FFFC-$FFFD.

Study this code for a minute and you will discover that this is where the rebooting takes place. The RESET vector at $3F2-$3F3 is scrambled and then jumped to indirectly through $FFFC-$FFFD. This causes a reboot. Looking up a bit earlier in the code shows that the RAM card is turned off and the motherboard ROMs enabled (at bytes $9F-$A1).

This looks like a good place to change and fix it so that it returns with the carry flag cleared (as a measure of safety):

Sector edits:

Track	Sector	Byte(s)	From	To
$00	$0A	$9F - $A0	AD 82	18 60

The reward for all of this work is that the copy now functions fine.

Cookbook Method

1

POKE 45358,0

2

RUN COPYA

and copy all four sides

3 Sector Edit:

Track	Sector	Byte(s)	From	To
$00	$0A	$9F-$A0	AD 82	18 60
$00	$0B	$58-$5A	CD 00 BD	C9 AD EA

That's all! Enjoy your backups.

Softkey for...

Award Maker Plus

Baudville

■ **What's needed:**

- ☐ *Award Maker Plus* original disks
- ☐ 4 blank disk sides
- ☐ a whole-disk copier like COPYA
- ☐ ProDOS User's Disk, //c System Utilities, or ProDOS *Copy II Plus*

Award Maker Plus is Baudville's entry into certificate making. Those who have used *Certificate Maker* will immediately recognize the similarities in the two programs. *Award Maker Plus* does one thing that *Certificate Maker* will not: **it will print your certificates in color and supports color printers such as the ImageWriter II for the printing.**

This is an excellent program for those who want to reward others for achievements or just to say "thanks".

The disk is in ProDOS format (ProDOS 8 to be exact) so naturally one would think there was no protection.

COPYA has no trouble making a copy of all four sides. Everything seems to be working out just fine until you try to print an Award. This is where you discover that the disk is protected. The message "Award Maker Plus by Baudville - Use original disk" appears on the certificate instead of the one you selected.

Hmmm. Reading through the *Award Maker Plus* Special Instructions reveals that a hard disk version is available upon receipt of your Warranty Registration card. "Hard Disk" implies unprotected but I would rather not have to wait. Sounds like it's COMPUTIST time.

The Protection

Since the disk is COPYA-able, no format alterations have been made so we don't have to worry about that.

A nibble count/signature check is the likely culprit. Looking at a disk map of side A with *Copy II Plus*, it seems a little odd that the ProDOS system file is on the last couple of tracks. It usually is on the first couple of tracks.

Also odd is that although every block is marked as used, a few on track $00 are not used by anything, directory or file-wise. A search of the disk for direct disk access code reveals some on track $00 (in those "unused" areas) and some in the ProDOS system file.

After many hours of studying and changing the track $00 code, I still could not get the disk to work. Getting a bit desperate, I tried something that looked to have a good chance.

Since the ProDOS system file was located in an odd place, and there was some odd code on track $00, why not format a ProDOS disk, copy a normal ProDOS system file to it, and then copy all of the other files from side A to the newly formatted disk?

Maybe it was during the boot that the disk check was taking place and if a normal boot was used, the program would think everything was OK. I did this and sure enough, the copy functioned perfectly now.

Maybe there is a "cute" way of circumventing the protection scheme, but this method allows you to put your *Award Maker Plus* on a hard drive if you want, and there is no waiting period!

Cookbook Method

1 Copy sides B,C, and D of the originals onto 3 of the blank sides using your whole-disk copier.

2 Using your User's Disk/System Utilities/*Copy II Plus*, format the other blank disk side and name it /**A**.

3 Copy the ProDOS system file off of your copier onto /**A**.

4 Copy all of the files, except the ProDOS system file, off of side A of the original onto /**A**.

5 Rename /**A** to /**AM.A1** and you will be done. Enjoy!

APT for...

Aztec

Datamost

■ **Needed:**

- ☐ Unprotected copy of *Aztec*
- ☐ Sector Editor/Disk Searcher

If you've ever played Datamost's *Aztec*, you know how frustrating it is to die because you have run out of strength points.

I usually get halfway through a level before my points run out. Wanting to fix this, I went about looking through the *Aztec* code to see if there was a solution. Here is a short synopsis of what I found:

$800-$9FF is where game variables are kept

$9ED is where the strength points are kept

$8E5 is where the sticks of dynamite are kept

$6506 is where the 'death' routine starts

$1B00 is where the game setup starts for a new game

$1B05 is where the game setup starts for a restarted game

Tracks $12-$22 is where the program code resides

Tracks $03-$05 and $07-$10 are the game boards

Track $11 is a dummy catalog for hackers to look at

Track $00 is the real catalog

2 versions of Aztec

There are two versions of *Aztec* out: One that is easily deprotected (POKE 47426,24 style) and one that is extremely nasty to back up.

My version is the easily deprotected version. If you have this version, copy a fast DOS onto it with *Copy II Plus* or equivalent and copy Track $06 to Track $11. Now change the catalog prointer in Track $11, Sector $00, Byte $01 from $06 to $11. There are two good files on the disk and one unrecoverable deleted file. You will only see the file "STARTER" (binary file) on the catalog, but using a program that scans the disk for Track-Sector lists will reveal another file written in BASIC.

The strength points and stick of dynamite are both initialized in the same place. The following code does the work:

```
A9 03       LDA #$03
8D ED 09    STA $09ED
8D E5 08    STA $08E5
```

This code is found (on my disk) at Track $21, Sector $0B, Byte $89. Changing the 03 to 09 will give you 9 strength points and 9 sticks of dynamite to start the game. Do not, however, use a number higher than 9. Stange things happen.

Now, to find the code that decrements the strength points, we must look for (you guessed it) a decrement instruction that decrements the memory location $09ED. Using the disk searcher, I came up with the following spots on the disk:

Track	Sector	Byte(s)	From	To
$1D	$01	$A7-A9	CE ED 09	EA EA EA
		$AF-B1	CE ED 09	EA EA EA
		$B5-B7	CE ED 09	EA EA EA
		$C2-C4	CE ED 09	EA EA EA

Making these changes has the net result of your strength points will not be decremented. You can, however, still die via a vicious attack or the natives getting you with their blow darts. Hmmmm.

I followed the code's logic and found that a death (except by blowing yourself up with dynamite) would send you to location $6506. This is the infamous "death routine". To get rid of it, you must take out the JMP $6506 commands and replace them with good code. Here are the sector edits to make:

Track	Sector	Byte(s)	From	To
$21	$09	$23-25	4C 06 65	38 B0 FB
		$52-54	4C 06 65	18 EA EA
$1D	$05	$06-07	A9 00	18 60

If you're wondering why there are different bytes in each of the above edits, I suggest that you look at the code around the edits.

For example, the change made at Track $21, Sector $09, Bytes $23-25 are from 4C 06 65 to 38 B0 FB. Right before the JMP $6506 is taken, a comparision is made and if the carry flag is set, everything is OK. If not, the JMP $6506 is taken. This tells us that the carry flag should be set in place of the jump and the program flow should be rerouted to the check again.

Looking at a disassembly of the code I'm talking about will clear things up, so take a peek.

Hope you enjoy the longer life you now have. The only "death" condition that I did not mess with was the one that occured when you were blown up by a stick of dynamite. This leaves some excitement in the game.

Softkey for...

Spindizzy

Activision

■ **Needed:**

- ☐ *Spindizzy* original disk
- ☐ one blank disk
- ☐ A fast DOS (not necessary but helpful)
- ☐ File copy program

Spindizzy is a three dimensional game in which you control a marble or top. The objective of the game is to collect all of the jewels that are scattered throughout the 386 screens. The animation is good and the graphics get high marks for being realistic. If you like this type of game, *Spindizzy* will give you many hours of fun and discovery.

I had been hacking at this game from time to time and getting nowhere. Looking through back issues of COMPUTIST for Activision softkeys, I had hoped to find one that would work for *Spindizzy*.

Looking at *Alter Ego*'s sector edits, the FROM bytes were exactly the same bytes as were found on the *Spindizzy* disk. Making these sector edits did not, however, produce a working copy.

I gave up on it until COMPUTIST 43 arrived. Jeff Rivett's softkey for *Murder on the Mississippi* rekindled my interest in cracking the program.

In his softkey, the file with the protection scheme was BOOT. Looking through all of the files on the *Spindizzy* disk revealed that "SHELL" contained the protection code. I had hacked with this particular file before but had gotten nowhere.

Keeping a mental note that the changes in Jeff's softkey started at byte $58, I looked through the SHELL file at location $C58, $D58, etc.

The instruction located at $C58 was a LDA command, so I tried making the changes listed in the *Murder...Mississippi* softkey.

Well, well...it worked! Made me feel grateful that I kept my back issues of COMPUTIST.

Cookbook Method

1 Boot up DOS, preferably a fast one like *Pronto-DOS* or *Diversi-DOS*.

2 Initialize the blank disk:

```
INIT HELLO
```

3 Using your file copier, copy all of the files from the original *Spindizzy disk to the newly initialized disk.*

4 *Put away the original in a safe place.*

5 *Get the protection code into memory and enter the monitor:*

```
BLOAD SHELL, A$C00
CALL -151
```

6 Bypass the protection check:

from: A9 08 C6 FC D0 04 C6 FD F0

to: A9 00 85 FC A9 55 4C 98 0C

7 Save the docile code back to the disk:

```
BSAVE SHELL,A$C00,L$228
```

Note: notice that the changes are the same ones that are in Jeff Rivett's softkey except at a different place in memory.

8 You're done! The disk now boots 3 times as quick and can be copied by your favorite disk copier.

H. Joseph Dobrowolski

A Note From...

Apple THREE Group International

First, congratulations on lasting another year; one that saw several good publications with poor management practices cease (many were TOO specific emphasizing only one machine, application or group of users; some were TOO general providing one program with 14 modifications).

Forgotten? Apple ///s

We also wish to commend your "balanced" coverage; for not abandoning the over 100,000 "forgotten" Apple /// & Appple /// Plus owners in the U.S. alone. Many publications (& their advertisers) don't even acknowledge the Apple /// as a 48K Apple][Plus emulator so it was a pleasure to see you spreading the word that Sara is neither dead nor forgotten.

Thanks for acknowledging the Apple ///'s still alive (COMPUTIST 24), for the generic hardware articles such as those dealing with disk drive modifications (Write-Protection switch, Speed Adjustment Control, etc.) allowing us to participate in your activities and your most recent mention of the Apple ///, "Modified F8 ROMs On the Apple ///." It provided much needed information and continues to let us know that COMPUTIST hasn't forgotten us.

Apple /// owners seem to be the "forgotten" people (many ex-customers) in Apple's corporate growth. The Apple /// is truely a fantastic machine; in our minds the best Apple ever produced. Advertised incorrectly as a "small business" machine only, the Apple /// has increasingly become a "personal" computer to tens of thousands. Many Apple Apple ///s were originally sold to businesses. When Apple decided to discontinue the Apple /// (and the rise of IBM/MS-DOS) many of these businesses looked for a "tax-advantage" in switching. Many decided to "donate" them to schools and non-profit organizations so their full list price could be written off. As a result many new Apple ///ers have "inherited" computers without the original manuals, without dealer support, without ... I'm sure you get the picture. As a result, users groups such as ours have joined together to help the NEW Apple ///er and through exchanging memberships, to help ourselves (including Apple ///ers in foreign countries).

I would like to share the following description of our users group with your Apple /// readers and welcome them to call or drop us a line:

The Apple THREE News & Views

The Apple THREE Group International, formerly the Apple /// Owners & Users Group International, is an independent, non-profit organization for all Apple ///ers (if you belong to a LOCAL Users Group or are connected to one via a modem, great, if not we'll try to make you feel like you belong).

Started in 1983 in Naples, Italy, we publish a monthly newsletter, the "Apple THREE News & Views," containing /// news gleaned from every source possible, attempt to answer or obtain answers to members' questions, and are building a "library" of EVERY piece of Apple /// Public Domain software available. COST? Dues are $15.00 annually in the U.S. ($16 in Canada, $25 foreign). Software is $3 per disk (Members only, U.S. postage included; $3.50 in Canada, $7.50 foreign; all payments are in U.S. funds). Ask a question or leave a message on our phone recorder now (1-804-865-7520) and look for our BBS to go "on line" soon!

Interested? Write for an application! Already a member; why not let other Apple /// owners/ users know about us? Post copies in your local computer stores and on any BBS's you log on to!:

Apple THREE Group International
c/o H. Joseph Dobrowolski
PO Box 913
Langley AFB, VA 23665

Michael A. Horton

a note on the...

softkey for... *Elite*

Since you gave *Elite* such a high rating on your games review, I thought it would be worth my money. Well you know what I think of it, I think that it is one heck of a good game. I just hope that there is a sequel coming and that it is even better than *Elite*.

Also the softkey for *Elite* that you published does NOT work for my copy.

What I would like to see is a book of APT's. I am sure your other readers would like one too.

Readers Data EXchange

David Wreggit

APT for...

Ultima IV

This APT concerns *Ultima IV* and a rather bizarre use of the balloon that can be found just outside of the dungeon of Hylothe (see COMPUTIST 42 for how to get there).

Unusual Balloon Ride

Aside from allowing a bird's eye view of the world of Britannia (which is quite useful for searching out the secrets hidden in mountains ranges and dark forests) it allows a person to seemingly 'walk upon the air' when properly used.

Technically, the balloon is suppose to be **B**oarded and then flown and landed with the **K**limb and **D**escend commands and finally e**X**ited.

Landing can only occur upon the grasslands, which really limits the usefulness of the balloon since there isn't a great deal of grassland in Britannia.

However, the trick to using the balloon is not to climb and descend with it, but to **B**oard, **K**limb and then e**X**it. Eliminating the **D**escend step in the command sequence provides several fabulous advantages for the adventurer.

Magical Advantages

1 The balloon can literally be landed anywhere. The top of a mountain range, the middle of the ocean, or in a lava pool are not unreasonable places to land. Once you've landed, the balloon can be left and returned to no matter what kind of terrain it is on.

2 After leaving the balloon in this unusual manner, many of the benefits of floating in the air still apply. For instance, the adventurer can still see into dark forests or over mountain ranges (You can't walk on water, though).

Poison swamp, fire/lava zones and other nasty surprises (such as the sleep area that you've got to walk though to talk to Hawkwind) don't affect you at all.

Monsters stand still and blithly watch as you waltz by them (they still think that you're in the balloon).

In towns and cities, the problem of seeing beyond walls is eliminated since the adventurer can see over any obstacle.

Odd Disadvantages

There are two small glitches to all this wonderful, new-found power of observation.

Invisibility?

Upon entry to any village, no one will move around to acknowledge the group's presence.

This gets to be an irritation very quickly when one wants to talk to Hawkwind or any of the other inhabitants of the towns that have to move towards the adventurers to communicate with them.

This may not seem terribly irritating, but it is occassionally a bother.

Dungeon Travel

The second oddity involves dungeon travel. To make a long explanation short, as long as the group travels in a straight line the screen is not updated - this makes dungeon travel virtually impossible.

Back To Normal

To return to a normal mode of play, **B**oard the balloon again, **K**limb to an altitude, and then **D**escend and e**X**it as you normally would.

Of course, NOW you have to land on a plains area.

Walk Through The Air

There are a few useful things that can be accomplished with this awesome "walk in the clouds" ability.

1 **Entering the Abyss**. With the use of a couple of wind spells, or a lucky south wind from the take off point of the original balloon location, the party of adventurers can set down in the middle of the fire zone surrounding the final dungeon of the Abyss without having to wade through the waters of Pirate's Cove; that's the place with about a billion pirate ships that sink your ship faster than a piece of lead unless you have the famous "Wheel".

Speaking of which, the wheel doesn't really help enough. You've really got to hop between pirate ships and subdue them hand-to-hand or else try to run through all the ships before they start attacking your ship in force rather than trading cannon shots.

2 **Getting to the Shrine of Humility** on the Isle of the Abyss. Anyone who has fought their way to this shrine without the silver horn, or even considered fighting the forty plus battles with daemons necessary to do it, will be happy to know that floating into the area around the shrine and e**X**iting will allow a safe trip to the shrine and back to the balloon.

Well, that's about all there is to tell about the useful auxilary functions of the ballon.

The ability to get around certain areas and see everything that there is to see is extremely useful and I've only pointed out the most obvious cases. I trust that everyone will find it as useful as I did - and after over 100,000 moves of the game, a unique way of playing is rather refreshing.

Mark Swanson

IIGS Softkey for...

Paintworks Plus

Activision

Piantworks Plus is a paint program for the Apple IIGS by Activision. At a retail price of $59-$69, I hated using the original master, so I set out to make a copy.

This was my first attempt at deprotection and also my first experience working with the GS in native mode.

First, I tried making a copy using the Apple IIGS SYSTEM disk desktop copy function, the system utilities will also work. Everything copied except block 7, which generated an error when it was read.

I booted the copy (minus block 7), and to my surprise everything started as it usually does with the original. It finally started drawing the *Paintworks*' screen, and when it finished, a message came up that said "Please insert master disk".

Well at this point all you could do was click the 'OK' box and the program would check the disk (I assume a nibble count on block 7). From this I deduced that the entire program had been copied and I was missing some kind of identifing marks on block 7.

I don't have many GS utilities yet, so trying to manually copy block 7 was impossible. Also, copying block 7 would only duplicate the master verbatim, not deprotect it. So I decided to dismantle the subroutine that checks block 7.

I used ByteZap.Pro by Beagle Bros. as a sector editor for a 3.5 drive. By changing line 72 to read: 72 BMAX=1600, ByteZap will recognize all 1600 blocks on the 3.5 disk. After ByteZap is running, press 'Ctrl S' and then press '5'. this will change the drive being used to slot 5. Now ByteZap will function normally, but it will work on your GS 3.5 disks!!

By single-stepping through *Paintworks Plus*, (using *Apple Programmers Workshop*, available through A.P.D.A.) I was able to pinpoint the JSR to the subroutine which checks block 7.

Next, I scanned the disk with ByteZap for that JSR. After I found it, I changed three consecutive bytes (the call routine) to NOP's (EA'a).

This didn't work. Somehow the last two bytes (the address of the subroutine) are restored after *Piantworks* is loaded from the disk.

Rather than spend more time trying to find out how the 2 bytes were restored, I decided to to change the JSR ($20) to something ng else which would not cause the program to brake when those 2 bytes are restored.

I used $8D (STA) since this would put the contents of the accumlator into the first locationb of the subroutine that check block 7. I did this so the subroutine would be destroyed and any later jump to this subroutine would cause a break and allow me to trace the stack and find where the jump came from.

To my surprise, this worked and there seemed to be no further callls to the subroutine. My patched copy seems to work fine and all functions work normally (sometimes things go haywire, but the original does this also). Obviously, the newness of the GS is showing in protection schemes (they seem simple).

I also copied *Tass Times in Tonetown* with the Sysytem Utilities and it works fine. However, a friend of mine can't get his copy of *Tass Time* to load and save, but my copy does.

I hope your readers can use this patch. I cringe every time I have to boot an original disk. And I will keep deprotecting programs for my own personal use as long as companies keep protecting them.

■ Requirement:

- ☐ *Paintworks Plus*
- ☐ a blank 3.5 disk
- ☐ A sector editor (must be able to write to a 3.5 disk)
- ☐ * Apple IIGS System Disk or any 3.5 disk copy program
- ☐ About 5 minutes of your time

Step by Step

1 Using the GS System Disk, (or 3.5 copy program), copy the original *Paintworks Plus* disk onto the blank disk. Ignore the error on block 7. Use this copy for the following steps.

2 Using a 3.5 disk sector editor, go to block 688 (hex $2B0), part A (the first 256 bytes). Next find byte #43 (hex $2B) and change it from 32 (hex $20) to 141 (hex $8D). Write this block back to the copy. *

3 Boot and use your deprotected *Paintworks Plus*!

* If you have trouble getting a sector editor to work for the 3.5 disks, try this. First copy (using Apple II GS System Disk) the file called "Paintworks" onto a 5 1/4 disk. Now use your favorite sector editor to do the patch described above. The byte to change won't be in the same location, so use the editor to scan for the occurance of the bytes 32 106 09 (hex $20 6A 09), and then change the $20 to $8D as described above in step 2. After you finish step 2, copy the patched *Paintworks* file back to the 3.5 disk.

John Wiegley

■ Requirements:

- ☐ A Sector Editor
- ☐ A copier which will ignore errors
- ☐ *Copy II Plus* Version 6.0 or better
- ☐ A blank formatted disk
- ☐ COPYA

Softkey for...

Arctic Fox
Electronic Arts

This program took me seven months to deprotect. Copy with a copier which will ignore errors. Make the following sector edits:

Track	Sector	Byte(s)	To
$01	$06	08	62
$01	$0A	5A	EA EA

Softkey for...

Might and Magic
Activision

This program however took two minutes to deprotect. Copy with a copier which will ignore errors. Make the following Sector Edits on Side A:

Track	Sector	Byte(s)	To
$0C	$0E	03	90

Copy the other sides with any copier.

Softkey for...

Amnesia
Electronic Arts

Copy this disk with a copier that ignores errors. Sector Edit:

Track	Sector	Byte(s)	To
$09	$03	27	60

Softkey for...

Black Magic

Enter *Copy II Plus*'s Sector Editor and go into the Patch screen. Choose DOS 3.3 PATCHED and then custom change the last Address prolog byte to AD and the last Data prologue byte to 96. Make all yes/no questions answer no. Read Track 1, Sector 0 to see if it worked. Then go to Manual Sector Copy and change these parameters: 59=AD, 77=FF. Copy tracks 1 to 10 and then track 12 to 22 onto the blank formatted disk.

Go back to the sector editor and go to the Patch screen. Choose DOS 3.3 PATCHED and go back to Manual Sector Copy. Don't change any parameters and copy Track 0. Then go back to the sector edit and make the same patch changes you made the first time and read track 11, sector 0 on the original. Write this sector on the copy.

The copy is **not** deprotected but will work.

1 William Thex

I have been trying for quite a while to copy some Spinnaker Products. Well, guess what I learned from just looking at COMPUTIST 29, the only one in my possession? I was just glancing through the "readers' softkey and copy exchange" section and in one paragraph, titled "Cookbook Instructions", it said to enter the monitor and defeat DOS's error checking. I thought to myself hummmm, tricky!

Softkey for...

Betterworking Word Processor
Spinnaker

Readers Data EXchange

Well, to get to the point, I was able to remove the copy protection from my *Betterworking Word Processor* from Spinnaker.

■ **Requirements:**

- ☐ DOS 3.3
- ☐ COPYA
- ☐ FID
- ☐ 1 blank double-sided disk

The way I removed the copy protection may seem a little amateurish but it works!

1 Enter the monitor.

CALL-151

2 Kill DOS error checking.

B942:18

3 Brun FID. Use the wildcard option (=) to copy all the files.

Ta-da! No copy-protection (neato!).

4 Run COPYA, make sure DOS's error checking is still dead. Copy side two of the disk.

You can just copy without removing the protection by following only steps 1, 2 and 4.

Well I just had to share this with you people since you are the ones that have inspired me to really work at copying that program. You know what is funny, I never really liked that word processor anyway, I use *Bankstreet Writer* and *Appleworks* more than I ever use *BetterWorking*.

Nother Anonymous

Softkey for...

Marble Madness

Electronic Arts

■ **Requirements:**

- ☐ 64K Apple][and up
- ☐ Copy program
- ☐ Sector Editor

First, without the article by Steve and Rod Smith in COMPUTIST 24, Page 10, this would not have been possible for me. It was through their complete explanations that this softkey was possible.

Marble Madness can be copied with any copy program that can accept track errors. For example, *Copy II Plus* disk copy or *Locksmith* Fast Copy.

First, format a blank disk.

Second copy both sides of the original to the formatted disk. Track 6 of the first side will not copy, but is not needed, so do not be concerned. The second side is not copy-protected, so no modifications are needed.

Using your sector editor scan the disk for the bytes $4C 69. I found them at the following locations:

Track	Sector	Bytes	Original Hex Values
$01	$0C	$00-04	4C 69 05 A0
$01	$0C	$69-72	4C 69 05 04
$01	$0F	$00-04	4C 69 A0 A0
$01	$0F	$69-72	4C 69 A0 04

Using the information for Mssrs. Smith's article, the next job is to delete the call to the nibble count while maintaining the balance they discuss on page 12.

Not being able to think in hex, I converted all hex to decimal. These hex-decimal conversion tables are found in most copy program manuals. Since the number was small enough when added, I needed to utilize only the first three bytes. The conversions of hex to decimal are shown below.

```
HEX 4C + 69 + 05 = DEC 76 + 105 + 5 = 186
HEX 4C + 69 + A0 = DEC 76 + 105 + 160 = 341
```

To skip the Electronic Arts protection code we need to change the call to the protection (the 4C 69) to an 18 60, but we also need to add a number to it to make it "balance". To do this we must use the first three bytes, substituting 18 60 for the 4C 69. Where the first three bytes total to 186 (Track $01, sector $0C) we must add a hex number whose decimal equivalent will bring the total to 186.

```
HEX 4C + 69 + 05 = DEC 76 + 105 + 5 = 186
HEX 18 + 60 + ? = DEC 24 + 96 + ? = 186
```

Solving for the unknown value, we find it to be 42 decimal which looking in the conversion table is hex $2A. Thus where the protection is 4C 69 05, we now substitute 18 60 2A.

We must do the same where the code totals are 341 (Track $01, Sector $0F).

```
HEX 4C + 69 + A0 = DEC 76 + 105 + 160 = 341
HEX 18 + 60 + ? = DEC 24 + 96 + ? = 341
```

Solving for the unknown value, we find it to be 221 which looking in the conversion table is hex $DD. Thus where the protection is 4C 69 A0 we now substitute 18 60 DD.

Congratulations! You just made it possible to let the kids play with another game without worrying that they might ruin the disk.

Summary

1 Format a disk with DOS 3.3 (Both sides).

2 Copy both sides with a disk copy program which ignores track errors.

3 With a sector editor, make the following modifications to your copy of side one:

Track	Sector	Byte(s)	From	To
$01	$0C	$00	4C	18
		$01	69	60
		$02	05	2A
		$69	4C	18
		$70	69	60
		$71	05	2A
01	0F	$00	4C	18
		$01	69	60
		$02	A0	DD
		$69	4C	18
		$70	69	60
		$71	A0	2A

Softkey for...

Bookends Extended

Bookends Extended may be copied by searching for the sequence BD 8C C0 10 FB 49 D5 D0 F7 and changing the final byte ($F7) to $56. On mine it was on Track 00, Sector 0F, Byte 0D.

Also, why isn't there a most wanted list anymore?

Walter Scott

A note on the softkey for...

Fantavision

The *Fantavision* softkey (COMPUTIST 30) will not work on the "authorized" backup copy of the disk. So, using TRAX from *Bag of Tricks*, I found that the epilog marks have been changed.

To make the softkey work you just change line 1130 to:

```
1130 DATA 222,170,222,170
```

The rest of the softkey works like a charm.

a note on...

The softkey for... *Masquerade*

COMPUTIST 35

Next, I would like to comment on the *Masquerade* softkey (COMPUTIST 35). Whoever wrote that softkey, was not very clear. I still don't understand what he was trying to say.

A misprint in the softkey for...

Auto Duel

COMPUTIST 36

Also the *Auto Duel* softkey (COMPUTIST 36) had a misprint. In step 4, it said to make some sector edits to the disk but failed to say which track and sector to modify!

I found out that it is supposed to be Track $17, Sector $07.

In closing, I would like to request a reprint of the "C SAVER" program. In the reprint of Super IOB V 1.5 in COMPUTIST No. 32, it should have been included, in my opinion.

P.S. Does anyone know how to backup *Infiltrator* by Mind Scape ...

Check out COMPUTIST 47 and this issue for Infiltrator softkeys. . . . . . RDEXed

James E. Mueller

a note on Clay Harrel's...

Softkey for... *Goonies*

COMPUTIST 44, Page 22

On my copy of *Goonies*, there is a check for the original disk after level one, and it appears to be the last one (at least through level 4).

If Clay Harrell would look into this, I would really appreciate it.

Eric D.

Adventure Tips for...

Bureaucracy

Infocom

1 To get the mail at the mansion, ring the bell, hot foot to the porch, and enter the house. Take the picture, exit, show it to the macaw, take the mail.

2 To get mail from the llama, open the bag of llama treats. Put them in the mail box, then take the mail.

3 On all the mail you get, notice the postage marks.

4 Eat whatever you get at the restaurant. You can't pay, so slip out the back door.

5 The object of part one is to get the neighbor's mail + $75 + 7 points.

6 To get mail at the old house, knock on the locked door. Go South. Show the man the stamp on the leaflet you received. He will leave, then you can take his mail.

Adventure Tips for...

King's Quest II

Sierra On-Line

1 First, get the cross by playing at the monastery.

2 Then find the door by the mountains.

3 Read the door, go back to the beach.

4 Get the trident, also make sure you have jewelry with you (found under the clam).

5 Find the mermaid (on one of the beach scenes).

6 Give her the bracelet. Ride the seahorse. Give trident to King Neptune. Get the first key.

Tyler Van Gorder

Softkey for...

The Bard's Tale II

Electronic Arts

■ **Requirements:**

- ☐ *Bard's Tale II*
- ☐ A copier (fast copier recommended)
- ☐ A sector editor
- ☐ 4 blank disk sides or 2 double-sided disks

The Bard's Tale II is the sequel to *The Bard's Tale*. It features very good animation and is good for many hours of fun. The object of this excellent adventure is to collect 7 pieces of the destiny wand and reforge them, to bring peace to the world again.

The protection was simple. Looking back to Steve and Rod Smith's article on how to crack Electronic Arts software (COMPUTIST 24), I scanned the disk for $4C 69. I found the following $4C 69 A0 in track $1 sector $F bytes $00-02. Well how convenient, I didn't even have to do any math, since the *Archon II* softkey uses the exact same bytes. After a little investigation, I discovered I had only to change those 3 bytes.

1 Copy all four sides of the *Bard's Tale II* with any copier you desire. I used the *Locksmith* fast copier. Be sure to label the sides.

2 Make the following sector edit changes to the BOOT side!

Track	Sector	Byte(s)	From	To
$01	$0F	$00	$4C	$18
		$01	$69	$60
		$02	$A0	$DD

3 That's all Folks!!

Readers Data EXchange

Michael Coffey

Softkey for...

Poetry Express
Mindscape

■ **Requirements:**

- ☐ Learning Well: *Poetry Express* original
- ☐ blank disk
- ☐ COPYA

Poetry Express is an easy-to-use program that makes poetry writing fun for even me.

There are explanation and line-by-line guidance providing help for the beginner in creating and printing nine different styles of poetry. Unfortunately, the disk is copy-protected, and if you are having kids using this, well...

I set out to discover the ins and outs of the disk and lo-and-behold if the copy-protection isn't an old trick we have seen before in Computist.

The only protection it had was to change some addresses.

On odd tracks the headers D4 AA 96 instead of the normal D5 AA 96.

You can get around this using the following method of having the computer copy the disk while ignoring the different address header.

1 Boot your computer with a normal DOS 3.3 disk.

2 Enter the monitor and alter the Read Address Header routine by typing:

```
CALL -151
B954:4A C9 6A D0 EF
```

3 Disable the address epilogue by typing:

```
B98B:18 60
```

4 Make a copy of the disk using COPYA .

5 Enjoy your new unprotected disk.

Michael David

More softkeys for...
MECC software

MECC has long been a respected name in educational software. However, MECC appears to have little regard for the teacher's dilemma of not enough money to buy back-up software and the inability to copy MECC software quickly and easily.

Softkeys for...

Addition Logician

Writing A Character Sketch

Writing A Narative
MECC

I was recently given three MECC programs that could not easily be copied. They had 1983-1984 title pages, and appeared worthwhile enough to attempt to crack them. I prepared for a long, arduous cracking session on the following: *Addition Logician*, *Writing a Character Sketch* and *Writing a Narrative*.

This was not to be the case. In fact, the use of a nonmaskable interrupt and RWTS saving was not necessary. Using Super IOB and the lines listed below, the programs were cracked in less than 2 minutes.

Step-by-step

1 Load Super IOB V1.5 into memory.

2 Delete lines 1000 through 9999 by typing:

```
DEL 1000,9999
```

3 Enter the new controller.:

Controller

```
1000 REM MECC CONTROLLER WITHOUT RWTS
1010 TK = 3 :LT = 35 :CD = WR :MB = 151 : ONERR
     GOTO 550
1020 ST = 0 :T1 = TK : GOSUB 490 : RESTORE :
     GOSUB 190 : GOSUB 210 : GOSUB 170
1030 GOSUB 430 : GOSUB 100 :ST = ST + 1 : IF ST
     < 16 THEN 1030
1040 IF BF THEN 1060
1050 ST = 0 :TK = TK + 1 : IF TK < LT THEN 1030
1060 GOSUB 230 :TK = T1 :ST = 0 : GOSUB 490
1070 GOSUB 430 : GOSUB 100 :ST = ST + 1 : IF ST <
     16 THEN 1070
1080 ST = 0 :TK = TK + 1 : IF BF = 0 AND TK < LT THEN
     1070
1090 IF TK < LT THEN 1020
1100 HOME :A$ = "ALL^DONE" : GOSUB 450 : END
5000 DATA 170 ,213 ,150 ,213 ,170 ,173 ,222 ,170
     ,222 ,170
```

Controller Checksums

1000	- $356B	1060	- $6AE6
1010	- $5E3F	1070	- $22FD
1020	- $B92C	1080	- $54D8
1030	- $E2AA	1090	- $7FC2
1040	- $2463	1100	- $F952
1050	- $E2BC	5000	- $ECD8

4 Copy your favorite DOS onto the disk and boot it.

Softkeys for...

Quotient Quest

Counting Critters

Math Critters
MECC

Although this method worked well on the three programs as listed, I was still faced with the task of cracking additional MECC wares with copyright dates varying from 1985-1986: *Quotient Quest*, *Counting Critters* and *Math Critters*.

Step-by-step

1 Boot up the original program and wait until the title page stops.

2 Using a non-maskable interrupt such as *Wildcard*, stop the program and get into the monitor.

3 Type:

```
1900<B800.BFFFM
```

4 Boot up a disk with basic on it by:

```
6(CTRL-P)
```

Readers Data EXchange

Keith Parker

Softkey for...

Marble Madness
Electronic Arts

■ **Requirements:**

- ☐ *Marble Madness* disk
- ☐ Whole disk copier that will ignore errors (such as *Locksmith*'s fast back-up)
- ☐ sector editor
- ☐ a desire to unlock software

When I purchased *Marble Madness*, I was a bit skeptical as it might not be as neat as the arcade version (which I dearly love). But to my surprise, it was.

Marble Madness is a game in which you, a marble, must roll your way through a 3-D maze. While dodging vicious marble-munching oozes (living pools of acid), steelies (aggressive bully marbles), slinkies (cute hungry springs), and birds. The game is quite challenging, especially if you play with the keyboard. Using the joystick is much easier.

Now on with the softkey.

The protection is quite simple since it's almost exactly like the protection on *The Bard's Tale*.

1 Copy both sides with your whole disk copier. Ignore any errors on track $6.

2 On "Side A" use your sector editor to do the following:

Track	Sector	Byte(s)	To
$01	$0B	$47-48	18 60
01	0E	47-49	18 60 40

3 Write the sectors back to the disk.

That's all! If there's anybody out there that knows how to get to the secret level, please let the rest of us in on it. Also, I found some interesting text about the secret level. The text just said "Congratulations!" and then told the password to his next game. I also found somthing about a "water level", possibly the secret level? Somehow, there's a way to choose which level you want to play, too.

Dave Richmond

I have several questions concerning disk drives and copy protection. The first is fairly simple, what exactly are parameters and how does a bit copier use them while copying a protected disk?

Second, How exactly, does the Boot ROM work when booting up a disk? I have heard several versions of how the controller ROM works. The first is that ROM only loads track $00, sector $00 into $800 and immediately jumps to $801 in memory. The second version is that the ROM finds the first data byte on track $00, sector $00 and sequentially loads in that many sectors into pages 8 and up starting with track $00, sector $00 and so on. However, a study of the boot process shows that this is not ture. The first byte at track $00, sector $00 is indeed the number of pages to be loaded in and they are loaded into pages 8 and up, but they do not come sequentially on the disk. In fact, some of them even come from different tracks altogether and some of the pages loaded in to the computer seem to be nonexistent on the disk. (Note: I did snow pages $8 - $20 with zeros and did a warm boot [C600G] to avoid getting bad results). What is happening?

Also, I wanted a backup of *Guitar Wizard* by Baudville, so I looked up all your softkeys for Baudville software that I had. Although I only had one softkey (for *Take 1*) it worked! I got it from COMPUTIST 37 on page 14. Just follow the instructions for deprotecting *Take 1*. COMPUTIST is great despite the cost but I wish you'd put in more articles on copy-protection and how to crack it.

Roger Roberts

A problem with the softkey for...

Sargon III
COMPUTIST 20

After deprotecting *Sargon III* with the method described by Kit LAU Yu-kit and Polly CHAN Yuk-yi in COMPUTIST 20, I encountered the same problems that Phil Boling of Dallas, Texas mentioned in COMPUTIST 9. *Sargon III* would begin to boot, the title "SARGON III" would appear at the top of the screen and then the disk would boot again and again and again ... ad nauseum.

Since I had just replaced the Autostart F8 ROM in my Apple][Plus with an old F8 ROM, I figured that the program was performing an F8 ROM check.

Using *Core Disk Searcher*, I searched the first three tracks of *Sargon III* for a reference to $FFFC, which is the address for the low byte of the reset vector in the Apple][Plus F8 ROM. Happily, I came up with a match in track 0, sector 1, Byte 13.

The code looked like this:

```
LDA $FFFC
CMP #$62
BEQ ......
```

It seems that in the Autostart F8 ROM, location $FFFC contains a $62. But in the Old F8 ROM, location $FFFC contains a $59. Therefore the loop was not taken and a reboot resulted.

In order to correct this, use the procedures outlined in COMPUTIST 20 in order to get rid of most of the protection and then use *Disk Edit* to make the following changes on Track 0, Sector 1, beginning with Byte 13:

Track	Sector	Byte(s)	From	To
$0	$1	$13	AD	A9
		$14	FC	00
		$15	FF	EA
		$16	C9	EA
		$17	62	EA

This translates to:

```
LDA #$00
NOP
NOP
NOP
```

This will eliminate the F8 ROM check and then allow you to play the game with either the Autostart or Old F8 ROMs since the branch will always be taken.

A Solution To Software That Do A RAMcard Check... such as:

Chessmaster 2000
Electronic Arts

Another chess game, *Chessmaster 2000* also does an F8 ROM check. But if you have a RAM card in slot 0, you can fool it into thinking you have a normal Apple][Plus.

First, start with a clean machine by booting DOS 3.3.

Then, put Applesoft in a file by typing:

```
BSAVE APPLESOFT ROM,A$D000,LS2800
```

Then take that disk and go to someone who has an Autostart F8 ROM and after booting DOS 3.3, type:

```
BSAVE AUTOSTART F8 ROM,A$F800,LS800
```

Now put both files on a regular DOS 3.3 disk and get out a text editor that writes regular DOS 3.3 text files.

Now you need to construct an EXEC file that will load these files into your RAM card in slot 0 whenever you need to fool a program. Start up your text file writer and type in the following:

```
NEW
BLOAD APPLESOFT ROM,A$1000
BLOAD AUTOSTART F8 ROM,A$3800
CALL -151
C089 N C089
D000<1000.3FFFM
C08B N C08B
3D0G
```

Save the file on your System Master Disk. Now whenever you need a regular Apple II Plus even if the protected program loads into the Slot 0 RAM card like *Chessmaster 2000*!

Leo & Eric Van Der Loo

Softkey for...

Arctic Fox

Electronic Arts

The protection is similar to the protection used on *The Bard's Tale*. Copy the disk with *Locksmith* fast copy or any other copier that ignores read errors. Edit the copy:

Track	Sector	Byte(s)	From	To
$1	$E	$47	20	18
		$48	F8	60
		$49	A0	40

Softkey for...

PHM Pegasus

Electronic Arts

PHM Pegasus uses the well known Electronic Arts protection, but on both sides of the disk this time. Copy both sides of the disk and use the same sector edits as in the *Arctic Fox* softkey above but be sure you edit both sides of the copy.

Softkey for...

The Rocky Horror Show

Activision

This is another of the new Activision games that can be copied with COPYA. The protection is similar to the protection used on *Labyrinth* and *Great American Cross Country Race*. After reading Mr. Nicholson and Mr. Rando's softkeys I was able to deprotect this one. The protection is found on track $22, sector $09.

1 Copy *The Rocky Horror Show* with COPYA or *Locksmith* Fast Copy.

2 Make the following edits:

Track	Sector	Byte(s)	From	To
$22	$9	$58-95	??	EA
		$96	38	EA
		$97	2A	EA
		$98	25	A9
		$99	FC	FF

If you have a new game by Electric Dreams or Activision and are unable to make a backup, scan your disk for the above shown bytes for the same copy protection. If you find the same protection let us all know.

Softkey for...

J-Bird

Cosmy Corp

At one time I made a working copy of the *J-Bird* game, then my kids got a hold of it and guess what? The copy wouldn't boot anymore. After two unsuccessful attempts to make another bit copy I decided it was time to deprotect the game.

The Protection

After examining the disk I found that, except for track $0 sector $0, all sectors had altered markers, the VTOC and the catalog on track $11 were altered and DOS command and error codes removed.

I figured Super IOB should be able to do most of the work. I made 2 controllers with the Super IOB Controller Writer (COMPUTIST 16). The first was for track $0, sectors $1 to $F; the second controller was for the rest of the disk. After running the IOB's I edited sectors $0 and $3 of track $0 and booted the disk. IT WORKED!!

1 Run Super IOB with the following controller:

Controller

```
1000 REM  J.BIRD CONTROLLER
1010 TK = 0 :LT = 1 :CD = WR :MB = 144
1020 ST = 1 :T1 = TK : GOSUB 490 : RESTORE :
     GOSUB 190 : GOSUB 210 : GOSUB 170
1030 GOSUB 430 : GOSUB 100 :ST = ST + 1 : IF ST
     < 16 THEN 1030
1040 IF BF THEN 1060
1050 ST = 1 :TK = TK + 1 : IF TK < LT THEN 1030
1060 GOSUB 230 :TK = T1 :ST = 1 : GOSUB 490
1070 GOSUB 430 : GOSUB 100 :ST = ST + 1 : IF ST
     < 16 THEN 1070
1080 ST = 1 :TK = TK + 1 : IF BF = 0 AND TK < LT
     THEN 1070
1090 IF TK < LT THEN 1020
1110 TK = 1 :LT = 35 :CD = WR :MB = 151
1120 ST = 0 :T1 = TK : GOSUB 490 : RESTORE :
     GOSUB 190 : GOSUB 210 : GOSUB 170
1130 GOSUB 430 : GOSUB 100 :ST = ST + 1 : IF ST
     < 16 THEN 1130
1140 IF BF THEN 1160
1150 ST = 0 :TK = TK + 1 : IF TK < LT THEN 1130
1160 GOSUB 230 :TK = T1 :ST = 0 : GOSUB 490
1170 GOSUB 430 : GOSUB 100 :ST = ST + 1 : IF ST
     < 16 THEN 1170
1180 ST = 0 :TK = TK + 1 : IF BF = 0 AND TK < LT
     THEN 1170
1190 IF TK < LT THEN 1120
1200 HOME :A$ = "ALL^DONE" : GOSUB 450 : END
5000 DATA 170 ,213 ,171 ,170 ,213 ,235 ,222
     ,171 ,237 ,170
```

Controller Checksums

1000	- $356B	1110	- $6835
1010	- $A9AC	1120	- $32D5
1020	- $B159	1130	- $1EE7
1030	- $9748	1140	- $0DE0
1040	- $516B	1150	- $469B
1050	- $1676	1160	- $9689
1060	- $28D9	1170	- $4883
1070	- $1DC1	1180	- $2F1E
1080	- $4ACE	1190	- $B6DD
1090	- $0D14	1200	- $7EFD
1100	- $F0B3	5000	- $1DB3

2 Using a sector such as DiskEdit, read track 0, sector 0 from the original disk. Remove the original and edit the following:

Track	Sector	Byte(s)	From	To
$0	$0	$4A	AA	D5
		$53	D5	AA
		$5D	AB	96
		$88	AA	D5
		$91	D5	AA
		$9B	EB	AD

3 Write the sector to track 0, sector 0 of your deprotected disk.

4 Edit track 0, sector 3:

Track	Sector	Byte(s)	From	To
$0	$3	$1A	AA	D5
		$23	D5	AA
		$2D	AB	96
		$57	AA	D5
		$60	D5	AA
		$6A	EB	AD

You should now have a COPYAble copy.

Jack R. Nissel

Softkey for...

Miner 2049er II
Micro Lab

1 Boot your DOS 3.3 system disk

2 Type:

CALL-151

3 Type:

B942:18

4 Run COPYA

5 Load your sector editor and make the folowing changes to the copy you just made.

Track	Sector	Byte(s)	From	To
$00	$02	$9E	D5	DE
$00	$03	$35	D5	DE
		$9B	E7	AA
		$91	9E	DE
$00	$08	$38	4C	08
		$39	6A	B0
		$3A	BA	8E

Softkey for...

Bop 'n Wrestle
Mindscape

1 Boot your DOS 3.3 system disk

2 Type:

CALL -151

3 Type:

B942:18

4 Run COPYA

5 Load your sector editor and make the follwoing changes to the copy you just made.

Track	Sector	Byte(s)	From	To
$00	$03	$42	38	18
$00	$08	$EA	20	EA
		$EB	00	EA
		$EC	02	EA

Softkey for...

Destroyer
Epyx

1 Boot you DOS 3.3 system disk

2

CALL-151

3 **Type:**

B942:18

4 Run COPYA

5 Load you sector editor and make the following changes to the copy you just made.

Track	Sector	Byte(s)	From	To
$00	$0A	$7C	A5	A9
		$7D	F4	E7
		$93	6C	4C
		$94	FC	7A
		$95	xx	BB

Sorry, but I cannot remember what the xx byte was.

Softkey for...

Temple Of Apshai Trilogy
Epyx

1 Boot your DOS 3.3 system disk

2

CALL -151

3 Type:

B942:18

4 Run COPYA The copy you make should run with no additional changes needed.

Softkey for...

Translyvania
Penguin Software

1 Install the controller listed below into Super IOB and copy the original to a blank disk

2 Copy DOS from your 3.3 system disk to the copy you just made

3 Boot your DOS 3.3 system disk

4 Type:

NEW

5 Type:

MAXFILES1

6 Put the copy you just made in your drive

7

BLOAD TPAR

8

CALL -151

9 Type:

943D:EA EA EA

10 Type:

BSAVE TPAR, A$9400, L$69D

11 Enjoy the game!

Controller

```
1000 REM TRANSYLVANIA
1010 TK = 3 :LT = 35 :ST = 15 :LS = 15 :CD = WR
     :FAST = 1
1020 GOSUB 490 :T1 = TK :LT = TK + 1 : RESTORE
     : GOSUB 170 : GOSUB 2000
1025 GOSUB 610 : IF PEEK (BUF ) < MB AND LT < >
     35 THEN LT = LT + 1 :TK = TK + 1 : GOSUB
1030 GOSUB 230 :TK = T1 :LT = 35 : GOSUB 490 :
     GOSUB 610 : IF PEEK (TRK ) = LT THEN 1050
1040 TK = PEEK (TRK ) :ST = PEEK (SCT ) : GOTO
     1020
1050 HOME : PRINT "COPY*DONE" : END
2000 POKE 47445 ,212 + (TK / 2 = INT (TK / 2 ) )
     : RETURN
5000 DATA 218 ,170 ,218 ,170
```

Readers Data EXchange

Controller Checksums

1000	- $356B	1040	- $FEC9
1010	- $2445	1050	- $F4DE
1020	- $FFC1	2000	- $133E
1025	- $CA14	5000	- $9388
1030	- $83A3	0	- $0000

Softkey for...

Beyond Castle Wolfenstein
Muse

1 INITialize a blank disk using the name HELLO

2 Install the controller listed below into Super IOB and copy the original to the initialized disk

3 Put the copy you just made into your drive

4 Type the following:

```
FP
10 HOME : D$ = CHR$ (4)
20 PRINT D$ "BRUN @INIT"
SAVE HELLO
```

Controller

```
1000 REM  BEYOND CASTLE WOLFENSTEIN
1010 TK = 3 :LT = 35 :ST = 15 :LS = 15 :CD = WR
     :FAST = 1
1020 RESTORE : GOSUB 190 : GOSUB 210 : POKE
     47426 ,24 : POKE 47786 ,35 : GOSUB 490 :
     GOSUB 610
1030 GOSUB 230 : POKE 47426 ,56 : POKE 47786
     ,170 : GOSUB 490 : GOSUB 610 : IF PEEK
     (TRK ) = LT THEN 1050
1040 TK = PEEK (TRK ) :ST = PEEK (SCT ) : GOTO
     1020
1050 HOME : PRINT "COPY*DONE" : END
5000 DATA 213 ,218 ,150 ,213 ,218 ,173
```

Controller Checksums

1000	- $356B	1040	- $E635
1010	- $2445	1050	- $7412
1020	- $F6C6	5000	- $0631
1030	- $7FBB	0	- $0000

Softkey for...

Pieman
Penguin Software

Use the controller for *Sword of Kadash* in COMPUTIST 27. When prompted, press "F" to run the controller (for the front side of *Sword of Kadash*).

Most of these softkeys were found by me trying the softkeys in COMPUTIST on other software by the same companies, and making minor, if any, modifications to them.

If you haven't already tried it, log onto the Cutting Edge BBS at (313) 349-2954 for some additional deprotection tips.

One last thing. I am sure there are alot of us out there that this magazine has helped out. Now let us help out the magazine. If you're reading someone else's copy, then subscribe to it, and if you already subscribe to it, get someone else to subscribe. I am sure none of us want this magazine to fold up. So, let's get out there and really support it any way we can.

Steve Marvin

Softkey for...

Hacker II: The Doomsday Papers
Activision

Run COPYA or any fast sector copier and the copy APPEARS to work normally, until you try to enter a logon ID. Then it does some disk access followed by a crash. The "HELLO" file HACKER II HELLO (Binary type) checks the disk for a sequence of nibbles (Signature) that begins with a normal nibble $FB and ends with a self sync nibble $FF. After nibble examination of the disk, track by track, I found this sequence on track 0 only, immediately following one data sector:

DE AA EB FB (normal)
BF FD BB FB FF (self sync)

After the signature check, which is located at $63C9 (The file "BRUNS" at $6000), the result is placed in location $6435 and the code exits with a CLC RTS. I discovered by placing a short intercept routine in the code that the byte should be $55 for my version.

Near the end of the protection code you'll see:

```
$6405: BD 8C C0 LDA C08C,X
$6408: 10 FB    BPL 6405
$640A: 38       SEC
$640B: 2A       ROL
$640C: 25 FC    AND $FC
Byte value is ready
$640E: 49 AA    EOR #$AA
Save it for check later
$6410: 8D 35 64 STA $6435
```

Change $6410 to jump to the intercept routine at $2D0, which will save the Accumulator, read-enable the motherboard ROMs, turn off the disk drive and enter the monitor. The value we want is in the Accumulator:

```
6410:4C D0 02
2D0:8D DC 02
2D3:8D 82 C0
2D6:9D 88 C0
2D9:4C 59 FF
2DC:00
$6000G
```

The patch is as follows: (Use the byte you found above)

```
BLOAD HACKER II HELLO,A$6000
CALL-151
6435:55
640E:A9 55
BSAVE HACKER II HELLO,A$6000,L$589
```

You can replace the DOS with a Fast DOS, but after the initial load of files "BLOAD/BRUN DOS.D500" ends up at $D500 in bank 1 of the upper 16K and it replaces DOS for all other file access during the game. Happy Hacking !

Scuzzy Port

Deprotection for Apple IIGS

Although the following deprotection schemes are not very informative, they are nevertheless very effective.

Copy the 3.5" disk and ignore the bad blocks (ie:"protection"). It's usually block $7 that is bad. Then, using a sector editor, make the following changes:

IIGS Softkey for...

Paintworks Plus 1.0, 1.01
Activision

Search the disk for: C9 06 09 D0 01 and replace these five bytes with: EA's.

On version 1.0, it's in block $291.

On version 1.01, it's in block $4B.

Readers Data EXchange

IIGS Softkey for...

Music Studio
Activision

Change byte $14 in block $44D from F0 to 80. (If it's not there, search for 0C 00 C9 01 00 **F0**.)

Block	Byte	From	To
$44D	14	F0	80

IIGS Softkey for...

Shanghai
Activision

Change:

Block	Byte	From	To
$243	$E1	F0	80

Joaquin Berrios

Softkey for...

Bard's Tale II
Electronic Arts

■ **Requirements:**

- ☐ *Bard's Tale II*
- ☐ Apple II series computer
- ☐ Super IOB v1.5
- ☐ Four blank disk sides

If you can copy a disk by a certain company, and you have another made by the same company, try the method you used on the first disk before anything else.

This proved to be the case with Electronic Arts' new release *Bard's Tale II (The Destiny Knight)*. After carefully loooking at the *Copy II Plus* parm for *Bard's Tale*, I set out to write a softkey to deprotect the new *Bard's Tale II*. I was surprised to see that the method of protection had not changed. All that was needed to do was skip track $6 and a few other sector edits to by-pass the protection routine.

1 Install the Super IOB controller listed at the end of this softkey

2 Copy the disk saying "yes" to the format option

3 The controller will make all the neccesary sector edits

4 Copy the other sides with COPYA or a fast copier

5 You now have a deprotected version of *Bard's Tale II*!

A Little Help

I also figured out what "The Dreamspell" spell code is: ZZGO. Also you can get more money than you will ever need by doing a simple sector edit. The bank accounts are stored in track $1, sector $1 of the character disk. The first four bytes contain the account number, which can be seen in the text with a sector editor, while the amount of gold is stored after it in a digit by digit format. For example, 56782 gold would be stored as $05 $06 $07 $08 $02. You can change the amount of money in your account by changing these bytes with a sector editor. Putting a $09 after the account number will give you all the gold you will ever need.

Controller

```
1000 REM  BARD'S TALE II CONTROLLER
1010 ST = 15 :LS = 15 :FAST = 1 :CD = WR
1020 TK = 0 :LT = 5 : GOSUB 1050
1030 TK = 7 :LT = 35 : GOSUB 1050
1040 HOME : PRINT "COPY*DONE!" : END
1050 GOSUB 490 : GOSUB 610
1060 T1 = TK :TK = PEEK (TRK ) - 1 : RESTORE :
     GOSUB 310 :TK = T1
1070 GOSUB 490 : GOSUB 610 : IF PEEK (TRK ) =
     LT THEN RETURN
1080 TK = PEEK (TRK ) :ST = PEEK (SCT ) : GOTO
     1050
3000 DATA 4*CHANGES
3010 DATA 1 ,6 ,8 ,98
3020 DATA 14 ,7 ,117 ,234
3030 DATA 14 ,7 ,118 ,234
3040 DATA 14 ,7 ,119 ,234
```

Controller Checksums

1000	- $356B	1070	- $547D
1010	- $16BA	1080	- $AD4D
1020	- $B211	3000	- $BE41
1030	- $5DC8	3010	- $E8CF
1040	- $D703	3020	- $35E6
1050	- $3601	3030	- $E707
1060	- $2FDF	3040	- $1304

If you know how to deprotect, unlock, or modify any of the programs listed below,

let us print your know-how in the **Readers Data Exchange.** Just send the information to us in a letter to the RDEX editor preferably on a DOS 3.3 diskette.

MOST WANTED LIST

Visiblend Microlab
Gutenburg Jr. & Sr. Micromation LTD
Prime Plotter Primesoft Corp.
The Handlers Silicon Valley Systems
Fun Bunch Unicorn
Willy Byte ... Data Trek
Snoggle Broderbund
ABM Muse
Agent U.S.A. Scholastic
Handicapping System Sports Judge
Odin Odesta
Brain Bank The Obsevatory
Crypt of Media Sir Tech
The Works First Star Software
Cross Clues Science Research
Peeping Tom Microlab
Jigsaw Microfun
Miner 2049er II Microfun
Create with Garfield DLM
Print Master Unision World
Bandits Sirius Software
Bank Street Filer Broderbund
Operation Frog Scholastic Software
Work Force II Core Concepts
Super Boulder Dash Electronic Arts
Zorro Datasoft
Goonies Datasoft
Kitchen's Game Maker Activision

A Real-time

by Bobby

It's too early to tell how many of you are interested in hardware. No letters have reached me yet. If you want more, please write; you hold the life of this column in your pens.

I'm going to assume that at least one of you is interested and write another column.

First, I have a suggested change to the schematic for the interrupt card. The resistor connecting PB7 to PB6 should be deleted and the circuit in figure 1 used instead. This way the AND gate will allow either PB7 or the external input (a jumper to OE on the video ROM, I hope) to clock PB6.

I have an Apple IIe with only four cards (extended 80 column, 64K printer buffer, hard disk controller and floppy disk controller). Recently it began clearing memory at odd intervals and in conjunction with a disk access. I'd seen this problem before; the wimpy Apple power supply ain't hacking it. My power supply was getting old too fast and couldn't put out enough current to satisfy my peripheral cards and the disk drive together. Removing any one card cleared the problem, so I knew the power supply wasn't completely bad, just marginal. Where could I get some extra milliamps?

Well, I could plunk down $35 for a new after-market power supply from JAMECO Electronics. That would give me 2500 extra milliamps. But I only needed a few, so I decided to "finess it". Hardware hackers probably already know about CMOS (Complimentary Metal Oxide Semiconductor) chips and about the new 74HCTxx series (a direct replacement for 74LSxx) used in the Apple. But do the rest of you know that not only is it a direct replacement but the HCT chips draw only microamps of current instead of milliamps. Here was a way to get that extra current and still be able to take in a movie tonight.

Pop open the top of your IIe and check inside. (Turn off the power first!) We're looking for socketed 74LSxx chips the we can replace without soldering. Let's see now, there's a 74LS245, a 74LS138, a 74S109 and a 74S10 on the motherboard. (Some idiot soldered all the other 74LSxx chips.) Then there's a 74LS245 and a 74LS374 on the extended 80 column card. OK! Close it up.

The 74Sxx chips draw a lot of current but they are also very very fast, too fast for HCT so we leave them alone. The 74LS138 only draws 10 milliamps so let's leave it alone too. But the 74LS245 (85 milliamps) and the 74LS374 (45 milliamps) are ideal for our purpose. Together (in a worse case) they draw a current of 215 milliamps. To get the right perspective, you need to read the Apple IIe reference manual (page 172) where it says "+5-volt power supply. A total of 500ma is available for *all* accessory cards." Only 500 milliamps for all of your power hungry cards in all your slots. Pretty lean diet. But we may be able to get up to 215 more milliamps just by replacing three chips. It's worth a try.

JAMECO Electronics lists the 74HCT245 and the 74HCT374 at $1.19 each. That comes to $3.57 (two 74HCT245 and one 74HCT374). The minimum order is $20 so I also ordered a few other chips that I needed.

I swapped the chips and so far, it works perfectly. The old power supply can still put it out with a little help from HCT.

Before anyone decides to replace all the chips in their computer, let me cautions you. HCT is listed as a direct replacement for LSTTL chips but there are some differences. HCT is marginally slower than LS and some chips have a reduced drive capability.

If you replace LS with HCT, it won't always work. You need to look at the timing and the loading of each individual chip and even then there is some confusion. For example. The 74LS138 (on the motherboard) drives the I/O Select ($CnXX) lines to the peripheral slots. The reference manual states that each I/O Select line will drive 10 LSTTL inputs. That means it will sink 4 milliamps of current for each slot or a total of 32 milliamps (worst case). The 74HCT138 is specified at only 8 milliamps output drive. That seems to leave it out. However, it's highly unlikely that each accessory card will use the maximum drive of 10 LSTTL loads. So the verdict is it will probably work on most of our Apples anyway, but I wouldn't use it on my Apple.

Also, if you have any of the oversize RAM boards that everyone seems to be selling now, 215 milliamps is just a drop in the bucket. You need a bigger power supply with four or even seven amps of current on the +5V line. That's 4000 or 7000 milliamps. Sorry, I don't have any reliability data on these power supplies since I haven't had to buy one. If anyone has experience, how about letting me know and I'll pass it along.

Activity Monitor

Now let's do something fun and nifty. I've got a simple circuit that everyone can build. I wanted a display of what addresses a program was accessing and where the program resided. I didn't want to stop the program so the display would also have to be in real time. Obviously the bus monitor circuit published previously with its hexadecimal displays would not work. A real-time display with hexadecimal readouts would be a blur. Besides, hexadecimal readouts cost to much. I like dirt cheap projects that don't use up my movie money. So I thought about it and came up with the circuit in figure 2.

The upper 4 bits of the address bus (A12, A13, A14 and A15) define 16 blocks of 4K each. This works out very neatly.

A15	A14	A13	A12	block	address range
1	1	1	1	F	$F000-FFFF
1	1	1	0	E	$E000-EFFF
1	1	0	1	D	$D000-DFFF
1	1	0	0	C	$C000-CFFF
1	0	1	1	B	$B000-BFFF
1	0	1	0	A	$A000-AFFF
1	0	0	1	9	$9000-9FFF
1	0	0	0	8	$8000-8FFF
0	1	1	1	7	$7000-7FFF
0	1	1	0	6	$6000-6FFF
0	1	0	1	5	$5000-5FFF
0	1	0	0	4	$4000-4FFF
0	0	1	1	3	$3000-3FFF
0	0	1	0	2	$2000-2FFF
0	0	0	1	1	$1000-1FFF
0	0	0	0	0	$0000-0FFF

Block 0 contains the zero page ($0000-00FF), the stack ($0100-01FF), the input buffer ($0200-02FF), text page 1 ($0400-07FF) and text page 2 ($0800-0FFF). Also, BASIC programs usually load starting at $0800.

Block 2 and 3 are hi-res page 1 ($2000-3FFF)

Block 4 and 5 are hi-res page 2 ($4000-5FFF).

DOS 3.3 starts in block 9 and uses block A and B ($9600-BFFF).

Block C is primarily used for I/O. The soft switches and on-board I/O are from $C000-C07F. The 16 byte I/O space for each slot use $C080-C0FF. The peripheral ROM space from $C100-C7FF is broken into 7 pages with each slot getting its own page. That leaves $C800-CFFF for the expansion ROM that each slot can use. The Apple IIe can disable I/O decoding

Part II

Activity Monitor

in the $C100-CFFF range to allow access to the CD ROM routines.

Applesoft in ROM ($D000-F7FF) uses block D, E and part of F.

The Apple monitor ($F800-FFFF) resides in the upper half of block F.

The other blocks are not assigned to any one particular function.

You've probably noticed that some blocks are a bit cramped and yes, a neat follow-up project would be to decode block 0 and block C.

How it works

The 4 upper address lines (bits A12-A15) are connected to input lines A thru D on the 74HC154 decoder.

These inputs select 1 of 16 outputs. The selected output goes low, all other outputs are high. G1 and G2 are enable inputs and are tied to ground. The decoder outputs are connected to LEDs (Light Emitting Diodes). All of the LEDs are connected, through a resistor, to the +5 volt supply. When any output (on the 74HC154) goes low, the corresponding LED will glow.

The resistor limits the current flowing thru the LEDs. Only one resistor is needed for all of the LEDs since only one will be on at any given moment.

But here's the nifty part, since the LEDs will be turning on and off at a very high rate, more than one will seem to be on at the same time indicating where the processor is accessing. And the intensity of each individual LED will indicate how much time is being spent in each block. The brighter LEDs are where the program resides and the dimmer LEDs are where the program is reading or writing. Nifty and all in real time.

The 74HCT373 transparent latch buffers the address lines (The 74HC154 input levels are not fully compatable with the Apple bus signals) and allows a special option.

On a IIe, pin 39 of each peripheral slot is connected to the microprocessor sync output (pin 7). Don't confuse this with the video sync. They are not the same.

The microprocessor sync is high during the first cycle of an opcode fetch. That means the sync is high only when the processor is reading the next program instruction. When we use this line to enable the latch, the LEDs will only glow on those blocks where the program resides.

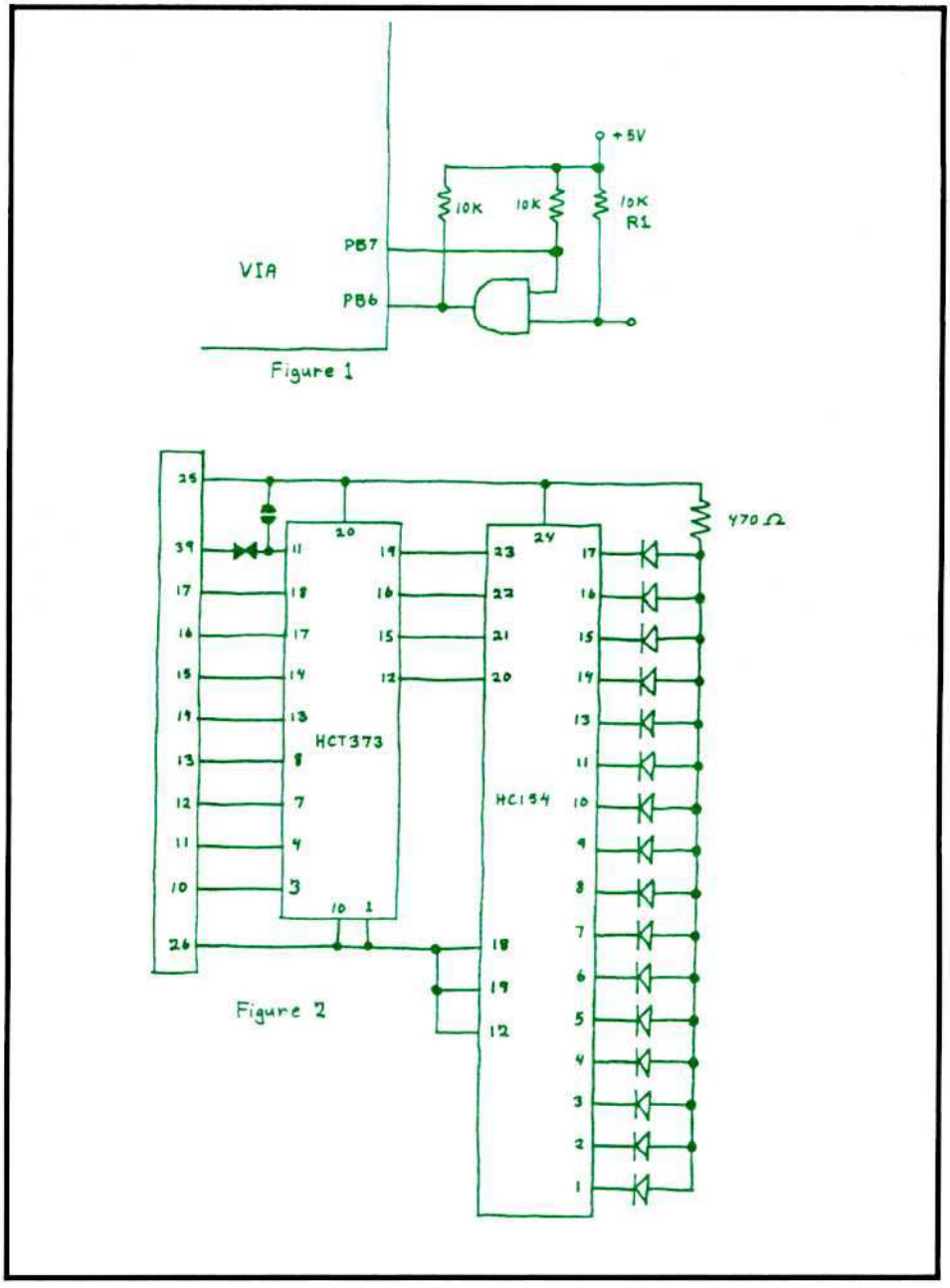
On a II and II+ the microprocessor sync line is not connected to anything.

The more adventurous reader can loosen the 6502, bend pin 7 outward, then connect a jumper from pin 7 to the display card. If you do this, don't forget to cut the connection to pin 39 of the slot.

Some Possible Problems

If you have a card that does DMAs (Direct Memory Access) the display will be erroneous while the card doing the DMAs is active. Accelerator cards will really give some wrong indications since the programs are running in the accelerator memory which is not reflected on the apple bus.

I'll put the circuit board layout for the interrupt card in the next issue. That's all for now. Oh yes, don't forget to write.



A Hackers Challenge

awaits those who enjoy "breaking" protection schemes for fun.

Knowledgeable with DOS 3.3 and associated utilities?

If so, this game is for you. Includes graphics routines from **The Graphics Magician®** by **Polarware™**.

Inside, protected by layers of graphical puzzles and DOS-tricks ***hides a secret***. Being among the 1st ten to discover the ***game secret*** wins you a computer-game prize! What are these prizes? I'm not saying, yet all the ***clues*** you need to win along with the ***prize-list*** are concealed in the game.

Find the clues by playing the game.

Find the clues by using your favorite utilities.

To answer this challenge send $10.00 to:

Mark V. Whitehurst
PO Box 485
Franklin Park, IL 60131

Requires APPLE IIe or IIc.

APPLE COMPATIBLES LOWEST PRICES ANYWHERE!

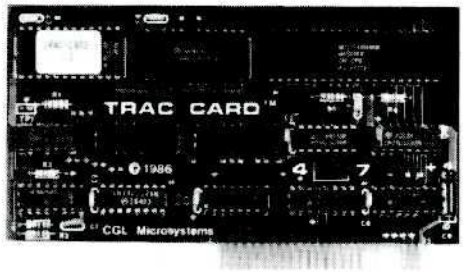
IMEG/80 COL BD w/256K+Software (IIe)	$99
IMEG RAM BD for IIGS w/256	$89
Above w/512K add $39 IMEG add $99	
64K/80 COL BD. New Lower Power (II+)	$35
16K RAM Board (II+)	$35
128K RAM BD. New Lower Power (II+)	$69
80 Column Board. Videx Comp. (II+)	$49
Super Serial Board (II+/e)	$49
Graphic Par BD w/6FT CBL (II+/e/gs)	$45
Z80 CP/M BD Microsoft Comp. (+/e/gs)	$38
Cooling Fan w/surge protect (II+/e)	$29
GS Super Cooling Fan (IIgs)	$25
Numeric Keypad. 16 Keys. (IIe)	$35
Joystick (Specify II+/c/e/gs)	$15
Joystick w/Fire on stick (+/c/e/gs)	$25
Mini Vacuum Cleaner w/Attachments	$10
A/B Switchbox. Parallel or RS-232	$29
Disk Drive H/H (Specify II+/c/e/gs)	$129
Disk Controller Board (II+/e/gs)	$39
Eprom Programmer (II+/e/gs)	$49

ONE YEAR WARRANTY ON ALL PRODUCTS
CALL/WRITE FOR COMPLETE LIST
ADD $3 SHIPPING (Per ORDER not per item)

NEXO DISTRIBUTION
914 East 8th Street, Suite 109
National City, CA 92050
(619) 474-3328
10am-6pm Mon-Fri

UNIV & SCHOOL P.O.'s WELCOME!
VISA/MC OKAY-C.O.D. ADD $2.00

TRAC CARD

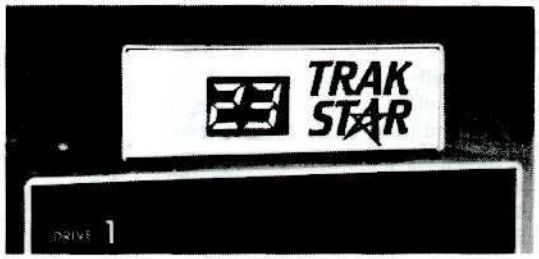


Boot Process Memory Card

+On-Board Memory Stores Up To 200 Disks Of Accessed Tracks While Powerd Up
+All Disks Are Automatically Monitored From The Moment You Power Up. The Tracks Are Divided Into Groups Of "Booted" Disks
+Save Time When Using Backup Software-The Tracks Accessed May Be Displayed In Numerical Order Or In The Order In Which They Are Read
+TRAC CARD Gives You Maximum Accuracy For Backing Up Software By Precisely Storing 1/4, 1/2 and 3/4 Tracks, As Well As Full Tracks
+You May Choose 40 or 80 Column On Monitor Or Dump Data To Printer. Name Each Disk When Printing Track List
+Choose Either Decimal Or Hexidecimal Readout
+Use In Any Slot, Including Slot #3 On //e
+Works With Any Apple Compatible 5 1/4" Drive
+Works With Apple II, II+ and //e, As Well As Compatibles

Price **$159.95** Plus $3.00 Shipping & Handling

TRAK STAR



Constant Digital Readout of Disk Drive Head Position

+Works With Any 5 1/4" Apple Compatible Drive
+Saves Copying Time With Nibble Programs
+Copy Only Tracks That Are Displayed
+If Copied Program Doesn't Run, TRAK STAR Displays Track To Be Recopied
+Displays Full and Half Tracks
+Operates With Any Apple Compatible Program, Including Protected Software
+Displays Up To 99 Tracks and Half Tracks; Compatible With High Density Drives
+Does Not Use A Slot in the Apple
+For Apple II, II+ and //e
+Simple One Minute Installation

Price **$99.95** Plus $3.00 Shipping & Handling
Adaptor Cable Required For 2 Drive System $12.00
DuoDdisk, 5 1/4" Unidisk and IIc Owners Please Write

Apple is a registered trademark of Apple Computer Inc

Personal checks, M.O.
Visa and Mastercard
Phone 913 676-7242

Midwest MM Microsystems

10308 Metcalf, Suite 355
Overland Park, KS 66212

the COMPUTIST shopper

Animate	$43.00	Legacy of Llylgamyn	$25.00	Rocky's Boots	$32.00
Award Maker	$23.75	Lode Runner	$22.00	Sargon III	$25.00
Bank Street Writer + (128K)	$46.75	**Magic Mindow II**	**$88.00**	Silent Service	$23.00
Black Cauldron	$25.00	**Magic Window //e**	**$88.00**	Summer Games II	$25.00
Certificate Maker	$28.00	Math Blaster	$27.00	**Super Macroworks**	**$29.50**
Copy II+	**$23.00**	Math Rabbit	$25.00	Where in the USA Carmen San Diego	$26.50
F15 Strike Eagle	$23.00	Might & Magic	$32.00	Where in the World Carmen San Diego	$30.00
Flight Simulator II	$35.00	Multiplan	$62.00	Wizardry	$32.00
GPLE	**$29.50**	Music Studio (GS)	$52.00	**Word Perfect w/ Spelling Checker**	**$95.00**
Gamemaker	$32.00	Newsroom	$35.00	WordPerfect (GS)	$95.00
Hacker II	$25.00	Paintworks + (GS)	$52.00	World Games	$25.00
Hacker II (GS)	$30.00	**Printshop**	**$32.00**	Writer Rabbit	$25.00
Hitchhiker Guide	$20.00	**Printshop Companion**	**$26.00**	Writers Choice Elite (GS)	$60.00
Karateka	$22.00	Reader Rabbit	$25.00	Zork I	$25.00
Knight Of Diamonds	$25.00	Reader Rabbit (GS)	$32.00	Zork Trilogy	$45.00

How To Order

- **US orders:** Circle your selection.
If total order is less than $200, please add $2 per item shipping & handling. Orders over $200 receive **free** shipping. Most orders shipped UPS, so use street address.
- In Washington state, please add 7.8% sales tax.
- Offer good while supplies last. All products are for the Apple II unless otherwise specified.
- **Foreign Orders:** Please inquire as to appropriate shipping fees.

Name _____ **ID#** _____

Address _____

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

Country _____ **Phone** _____

VISA MC _____-_____-_____-_____ **Exp.** _____

Signature _____ **CP48**

Send orders to: **Softkey Publishing PO Box 110816-T Tacoma, WA 98411 (206) 474—5750**

Legends tell of the days when the ancient back issues of Hardcore COMPUTIST were readily available to anyone who wished to purchase them. Those days may be long since past, but the information contained in these ancient documents has been diligently transcribed to the pages of a modern reference work:

The Book of Softkeys

Volume I: Compiled from issues 1-5

contains softkeys for: Akalabeth •Ampermagic •Apple Galaxian •Aztec •Bag of Tricks •Bill Budge's Trilogy •Buzzard Bait •Cannonball Blitz •Casino •Data Reporter •Deadline •Disk Organizer II •Egbert II Communications Disk •Hard Hat Mack •Home Accountant •Homeword •Lancaster •Magic Window II •Multi-disk Catalog •Multiplan •Pest Patrol •Prisoner II •Sammy Lightfoot •Screen Writer II •Sneakers •Spy's Demise •Starcross •Suspended •Ultima II •Visifile •Visiplot •Visitrend •Witness •Wizardry •Zork I •Zork II •Zork III •**PLUS** how-to articles and program listings of need-to-have programs used to make unprotected backups.

Volume II: Compiled from issues 6-10

contains softkeys for: Apple Cider Spider •Apple Logo •Arcade Machine •The Artist •Bank Street Writer •Cannonball Blitz •Canyon Climber •Caverns of Freitag •Crush, Crumble & Chomp •Data Factory 5.0 •DB Master •The Dic*tion*ary •Essential Data Duplicator I & III •Gold Rush •Krell Logo •Legacy of Llylgamyn •Mask Of The Sun •Minit Man •Mouskattack •Music Construction Set •Oil's Well •Pandora's Box •Robotron •Sammy Lightfoot •Screenwriter II v2.2 •Sensible Speller 4.0, 4.0c, 4.1c •the Spy Strikes Back •Time Zone v1.1 •Visible Computer: 6502 •Visidex •Visiterm •Zaxxon •Hayden Software •Sierra Online Software •**PLUS** the complete listing of the ultimate cracking program...Super IOB 1.5 ••and more!

Volume III: Compiled from issues 11-15

contains softkeys for: Alien Addition •Alien Munchies •Alligator Mix •Computer Preparation SAT •Cut And Paste •Demolition Division •DLM (Developement Learning Materials) software •EA (Electronic Arts) software •Einstein Compiler version 5.3 •Escape From Rungistan •Financial Cookbook •Flip Out •Hi-Res Computer Golf II •Knoware •Laf Pak •Last Gladiator •Learning With Leeper •Lion's Share •Master Type v1.7 •MatheMagic •Minus Mission •Millionaire •Music Construction Set •One On One •PFS software •PS (Penguin) Software •The Quest •Rocky's Boots •Sabotage •Seadragon •Sensible Speller IV •Snooper Troops II •SoftPorn Adventure •Stickybear series •Suicide •TellStar •Tic Tac Show •Time Is Money •Transylvania •Type Attack •Ultima III Exodus •Zoom Graphics •Breaking Locksmith 5.0 Fast Copy •**PLUS** feature articles on •Csaver •The Core Disk Searcher •Modified ROMs.

To order: The Book of Softkeys

☐ **Volume I - $7.95 + $2 shipping/handling**
☐ **Volume II - $12.95 + $2 shipping/handling**
☐ **Volume III - $17.95 + $2 shipping/handling**

☐ **All three volumes! - $30.00 + $2 ship/handling**

Name _____ **ID#** _____

Address _____

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

Country _____ **Phone** _____

VISA MC _____-_____-_____-_____ **Exp.** _____

Signature _____ **CP48**

Foreign orders (except Canada and Mexico) please add $5 for shipping and handling. Washington residents add 7.8% sales tax. Most orders are shipped within 5 working days, however, please allow 4-6 weeks delivery. **US Funds drawn on US banks only.** Send to:

Book of Softkeys PO Box 110846-T Tacoma, WA 98411
(206) 474-5

COMPUTIST

47 *Features:* Infocom-text Reader Enhancement •Color Ultimapper mod to Ultimapper IV •Towne Mapper utility for Ultima IV •Dungeon Mapper utility for Bard's Tale •Hardware Corner: Interrupting Your Apple •Softkey for Charlie Brown's 1,2,3s ● *RDEX Softkeys:* •Guitar Wizard •Gemstone Warrior •Notable Phantom •Micro Wine Companion •Stickybear Printer •Note Card Maker •Starcross •Wishbringer •Dinosaur Dig •Dam Busters •Pirate Adventure •Infiltrator •MECC software •Banner Catch •Turtle Tracks •PFS File •Microzine #12, #13, #14 •Marble Madness •Writer Rabbit •Arcticfox •Age Of Adventure •Might And Magic •Space Station •Alternate Reality •Mindshadow •Gemstone Warrior •Strip Poker •Lucifer's Realm •Manuscript Manager •Bank Street Writer III •Kids On Keys •The Missing Ring •Graphic Solution •Empire I, II •Championship Golf.

46 *Softkeys* •Advanced Microsystems Technology programs •Word Attack •Star Blazer •Science Toolkit •The Color Enhanced Print Shop •Video Vegas •The Handlers •K.C. Deals On Wheels •Law Of The West •Break The Bank Blackjack •Foundation Course In Spanish •OGRE •Puzzles And Posters •*Features* •The Shift Key/Lower Case Option For II+ •Amazing Computer Facts •Shape Magic utility ●*review*: Multiscribe. . . . . .

45 *Softkeys* •Mouse Calc •Sands of Egypt •Number Farm •Agent U.S.A. •Wavy Navy •Kindercomp •Flight Simulator Update •Raid over Moscow •Crime Stopper •Key Perfect 5. •The Final Conflict •Miss Mouse •Snoggle •*Features* •Write Protecting the Microsoft RAM Card •Keys to Success on the Franklin Ace •Modified F8 ROMs on the Apple /// •*Core* •Owner's Review of Copy Master II

44 *Softkeys* •Arcade Boot Camp •Goonies •Zorro •Coveted Mirror •Crimson Crown •Compubridge •Fleet System 3 •Microwave •Escape •Catalyst 3.0 •Number Farm •Alphabet Circus •Joe Theisman's Pro Football •Black Cauldron •Intern. Gran Prix •*Features* •Making DOSless Utilities •Pixit Printer Drivers •*Review:* Z-RAM Memory Expansion Board •Reading the Joystick. . .

43 *Softkeys* •Graphics Expander •Information Master •Certificate Maker •Elite •Catalyst 2.0 and 3.0 •Murder On The Mississippi •Temple Of Apshai Trilogy •Troll Associates programs •Spell It •Regatta •Cdex Training programs •Think Fast •*Features* •How to Write-Protect your Slot Zero •Capturing Locksmith 6.0 Fast Copy •Revisiting DOS to ProDOS and Back •*Core* •Computer Eyes / 2: a Review •*APTs* •Sword of Kadash & Rescue Raiders •Ultimaker IV.

42 *Softkeys* •Light Simulator •Beach-Head •Monty Plays Scrabble •Racter •Winnie the Pooh •Infocom Stuff, Kabul Spy, Prisoner II •Wizardry I & 2 •Lucifer's Realm •The PFS Series •Dollars and Sense •Strip Poker •Coveted Mirror •Wizard's Crown •The Swordthrust Series •Axis Assassin •Manuscript Manager •The Crown of Arthain •Address Book •Decimals 3.0 •Dragonfire •*Features* •Auto Duel Editor •Wizard's Crown Editor •Questron Mapper •*Core* •The Games of 1986 in Review •*Adventure Tips* •Ultima IV

41 *Softkeys* •The Periodic Table •Gemstone Warrior •Inferno •Frogger •Story Maker •Adventure Writer •Mummy's Curse •Zaxxon •The Quest •Pitfall II •H.E.R.O. •*Features* •A Two-Drive Patch for Winter Games •Customizing the Speed of a Duodisk •Roll the Presses Part Two: Printshop Printer Drivers •The Games of 1986. . . . . .

40 *Softkeys* •Adventure Writer •E-Z Learner •Mychess II •Raster Blaster •Cranston Manor •Ghostbusters •Designer's Pencil •The American Challenge •Encyclopedia Britannica Programs •Crime Wave *Features* •Taking the Wiz out of Wizardry •Adding a Printer Card Driver to Newsroom •*Core* Games of 1986

39 *Softkeys* •MIDI/8+ •Homeword v2.1 •Borrowed Time •Amazon •Speed Reader][•Discovery! •M-ss-ng L-nks series •Donald Ducks's Playground •Mastering the SAT •Copy][Plus 4.4C •Master of the Lamps •One on One •Bridge Baron •A.E. •Great American Cross-Country Road Race •Computer Preparation for the SAT •Castle Wolfenstein •Luscher Profile •Skyfox •Silent Service •Echo Plus •Swashbuckler •Randamn •*Features* •Electronic Disk Drive Swapper •Abusing the Epilogues •Print Shop Companion's Driver Game •*Core* •Keyboard Repair •Fixing the Applesoft Sample Disk. . . . . .

38 *Softkeys* •Cyclod •Alternate Realty •Boulder Dash I & II •Hard Hat Mack (Revisited) •The Other Side •F-15 Strike Eagle •Championship Lode Runner •Gato V 1.3 •I, Damiano •Wilderness •Golf's Best •*Features* •The Enhanced/ Unenhanced //e •Looking into Flight Simulator's DOS •*Core* •Appavarex •Installing a RAM disk into DOS 3.3. . . .

37 *Softkeys* •Under Fire •Pegasus][•Take 1 (revisited) •Flight Simulator II v1.05 (part 2) •Magic Slate •Alter Ego •Rendezvous •Quicken •Story Tree •Assembly Language Tutor •Avalon Hill games •Dark Crystal •*Features* •Playing Karateka on a //c •Track Finder •Sylk to Dif •*Core* •Breaking In: tips for beginners •Copy][Plus 6.0: a review •The DOS Alterer. . . . . .

36 *Softkeys* •Flight Simulator II v 1.05 •AutoDuel •Critical Reading •Troll's Tale •Robot War •General Manager •Plasmania •Telarium Software •Kidwriter v1.0 •Color Me •*Features* •ScreenWriter meets Flashcard •The Bus Monitor •Mousepaint for non-Apples •*Core* •The Bard's Dressing Room •*APT* •Championship Lode Runner. . . . . .

35 *Softkeys* •Olympic Decathlon •Hi-res Cribbage •Revisiting F-15 Strike Eagle •Masquerade •The Hobbit •Pooyan •The Perfect Score •Alice in Wonderland •The Money Manager •Good Thinking •Rescue Raiders •*Feature:* Putting a New F8 on Your Language Card •*Core* •Exploring ProDOS by installng a CPS Clock Driver. . . . . .

34 *Softkeys* •Crisis Mountain •Terripin Logo •Apple Logo II •Fishies 1.0 •SpellWorks •Gumball •Rescue at Rigel •Crazey Mazey •Conan •Perry Mason: The Case of the Mandarin Murder •Koronis Rift •*Feature* •More ROM Running •*Core* •Infocom Revealed. . . . . .

33 *Softkeys* •Word Juggler •Tink! Tonk! •Sundog v2.0 •G.I. Joe & Lucas Film's Eidolon •Summer Games II •Thief •Instant Pascal •World's Greatest Football Game •Graphic Adventure #1 •Sensible Grammar & Extended Bookends •Chipwits •Hardball •King's Quest II •The World's Greatest Baseball Game •*Feature* •How to be the Sound Master •*Core* •The Mapping of Ultima IV. . . . . .

Available Back Issues

32 *Softkeys* •Revisiting Music Construction Set •Cubit •Baudville Software •Hartley Software •Bridge •Early Games for Young Children •Tawala's Last Redoubt •Print Shop Companion •Kracking Vol II •Moebius •Mouse Budget, Mouse Word & Mouse Desk •Adventure Construction Set •*Feature* •Using Data Disks With Microzines •*Core* •Super IOB v1.5 a Reprint . . . . . .

31 *Softkeys* •Trivia Fever •The Original Boston Computer Diet •Lifesaver •Synergistic Software •Blazing Paddles •Zardax •Time Zone •Tycoon •Earthly Delights •Jingle Disk •Crystal Caverns •Karate Champ •*Feature* •A Little Help With The Bard's Tale •*Core* •Black Box •Unrestricted Ampersand . . . . . .

30 *Softkeys* • Millionaire • SSI's RDOS • Fantavision • Spy vs. Spy • Dragonworld • King's Quest • Mastering the SAT • Easy as ABC • Space Shuttle • The Factory • Visidex 1.1E • Sherlock Holmes • The Bards Tale • *Feature* • Increasing Your Disk Capacity • *Core* • Ultimaker IV, an Ultima IV Character Editor . . . . . .

29 *Softkeys* • Threshold • Checkers v2.1 • Microtype • Gen. & Organic Chemistry Series • Uptown Trivia • Murder by the Dozen • Windham's Classics • Batter Up • Evelyn Wood's Dynamic Reader • Jenny of the Prairie • Learn About Sounds in Reading • Winter Games • *Feature* • Customizing the Monitor by Adding 65C02 Disassembly • *Core* • The Animator . . . . . .

28 *Softkeys* • Ultima IV • Robot Odyssey • Rendezvous • Word Attack & Classmate • Three from Mindscape • Alphabetic Keyboarding • Hacker • Disk Director • Lode Runner • MIDI/4 • Algebra Series • Time is Money • Pitstop II • Apventure to Atlantis • *Feature* • Capturing the Hidden Archon Editor • *Core* • Fingerprint Plus: A Review • Beneath Beyond Castle Wolfenstein (part 2) . . . . . .

27 *Softkeys* • Microzines 1-5 • Microzines 7-9 | Microzines (alternate method) • Phi Beta Filer • Sword of Kadash • Another Miner 2049er • Learning With Fuzzywomp • Bookends • Apple Logo II • Murder on the Zinderneuf • *Features* • Daleks: Exploring Artificial Intelligence • Making 32K or 16K Slave Disks • *Core* • The Games of 1985: part II . . . . . .

26 *Softkeys* • Cannonball Blitz • Instant Recall • Gessler Spanish Software • More Stickybears • Financial Cookbook • Super Zaxxon • Wizardry • Preschool Fun • Holy Grail • Inca • 128K Zaxxon • *Feature* • ProEdit • *Core* • Games of 1985 part I

25 *Softkeys* • DB Master 4.2 • Business Writer • Barron's Computer SAT • Take 1 • Bank Street Speller • Where In The World Is Carmen Sandiego • Bank Street Writer 128K • Word Challenge • Spy's Demise • Mind Prober • BC's Quest For Tires • Early Games • Homeword Speller • *Feature* • Adding IF THEN ELSE To Applesoft • *Core* • DOS To ProDOS And Back . . . . . .

24 *Softkeys* • Electronic Arts software • Grolier software • Xyphus • F-15 Strike Eagle • Injured Engine • Mr. Robot And His Robot Factory • Applecillin II • Alphabet Zoo • Fathoms 40 • Story Maker • Early Games Matchmaker • Robots Of Dawn • *Feature* • Essential Data Duplicator copy parms • *Core* •DOS-Direct Sector Access . . . . . .

23 *Softkeys* • Choplifter • Mufplot • Flashcalc • Karateka • Newsroom • E-Z Draw • Gato • Dino Eggs • Pinball Construction Set • TAC • The Print Shop: Graphics Library • Death In The Caribbean • *Features* • Using A.R.D. To Softkey Mars Cars • How To Be The Writemaster • *Core* • Wheel Of Money . .

22 *Softkeys* • Miner 2049er • Lode Runner • A2-PB1 Pinball • The Heist • Old Ironsides • Grandma's House • In Search of the Most Amazing Thing • Morloc's Tower • Marauder • Sargon III • *Features* • Customized Drive Speed Control • Super IOB version 1.5 • *Core* • The Macro System . .

20 *Softkeys* • Sargon III • Wizardry: Proving Grounds of the Mad Overlord and Knight of Diamonds • The Report Card V1.1 • Kidwriter • *Feature* • Apple][Boot ROM Disassembly • *Core* • The Graphic Grabber v3.0 • Copy II+ 5.0: A Review • The Know-Drive: A Hardware Evaluation • An Improved BASIC/Binary Combo . . . . . .

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

16 *Softkeys* • Sensible Speller for ProDOS • Sideways • Rescue Raiders • Sheila • Basic Building Blocks • Artsci Programs • Crossfire • *Feature* • Secret Weapon: RAMcard • *Core* • The Controller Writer • A Fix For The Beyond Castle Wolfenstein Softkey • The Lone Catalog Arranger Part I . . . . . .

1 *Softkeys* • Data Reporter • Multiplan • Zork • *Features* • PARMS for Copy II Plus • No More Bugs • APT's for Choplifter & Cannonball Blitz • *'Copycard' Reviews* • Replay • Crackshot • Snapshot • Wildcard

Use the handy Order Form on the back.

Are you missing a piece of the picture ???!!!

Issue	Mag $4.75	Disk $9.95	Both $12.95
47	☐	☐	☐
46	☐	☐	☐
45	☐	☐	☐
44	☐	☐	☐
43	☐	☐	☐
42	☐	☐	☐
41	☐	☐	☐
40	☐	☐	☐
39	☐	☐	☐
38	☐	☐	☐
37	☐	☐	☐
36	☐	☐	☐
35	☐	☐	☐
34	☐	☐	☐
33	☐	☐	☐
32	☐	☐	☐
31	☐	☐	☐
30	☐	☐	☐
29	☐	☐	☐
28	☐	☐	☐
27	☐	☐	☐
26	☐	☐	☐
25	☐	☐	☐
24	☐	☐	☐
★23	☐	☐	☐
22	☐	☐	☐
19	☐	☐	☐
16	☐	☐	☐
8	•	☐	•
★7	☐	☐	☐
Core 2	☐	☐	☐
1	☐	☐	☐
Core 1	☐	☐	☐
Computing 3	☐	•	•
Best of Hardcore Computing	•	☐	•
Core Special Combo $10 (All three CORE magazines)	☐		

Some disks apply to more than one issue and are shown as taller boxes. Special "Both" disk & magazine combination orders apply to one issue and its corresponding disk.

★ We have a limited supply of these issues.

• Back issue is no longer available

COMPUTIST

back issues and library disks are frequently referenced in current issues.

Back Issues and Library Disk Rates

- US, Canada and Mexico back issue rate - **$4.75** each.
- All other Foreign back issue rate - **$8.75** each.
- US, Canada, Mexico library disk rate - **$9.95** each.
- All other Foreign library disk rate - **$11.94** each.
- "Both" disk and magazine rates for:
  US, Canada & Mexico - **$12.95** each combination.
  All other Foreign - **$18.95** each combination.

What is a library disk?

A library disk is a diskette that contains programs that would normally have to be entered by the user. Documentation for each library disk can be found in the corresponding issue.

• Library disks are available for all issues of COMPUTIST 1 thru 45. A description of the softkeys and programs covered in each issue is available upon request. Please send your name and address along with a first class postage stamp (US $ .22).

Complete Your Collection!

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

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

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

Hardcore Computing 3 HyperDOS Creator• Menu Hello• Zyphyr Wars• Vector Graphics• Review of Bit Copiers• Boot Code Tracing• Softkey IOB• Interview with 'Mike' Markkula.

☐ **Please send the back issues indicated:**

☐ **Please send the following library disks:**

Name _____ ID# _____

Address _____

City _____ State _____ Zip _____

Country _____ Phone _____

VISA MC _____-_____-_____-_____ Exp. _____

Signature _____ CP48

US funds drawn on US banks

_____ _____ _____ _____

Send check or money order to:

COMPUTIST
PO Box 110846-T
Tacoma, WA 98411.
(206) 474-5750

Most orders are shipped within 5 working days, however please allow up to 4 weeks delivery for some orders. Most orders shipped UPS, so please use street address. Offer good while supply lasts. In Washington state, add 7.8% sales tax.

How To Write for COMPUTIST

COMPUTIST has two basic sections:
(1) the new Readers Data Exchange, and
(2) Special Features.

R D Ex
Readers Data Exchange

The old INPUT and the READERS SOFTKEY AND COPY EXCHANGE have now been combined into the new Readers Data Exchange (or RDEx). All short softkeys, comments, short articles, questions, answers, fixes and bugs and almost any other information that must be printed in a timely fashion will be placed in the Readers Data Exchange.

Although we will print letters and short article-softkey submissions that are received as only printed text, we would prefer that they be submitted on disk. For submission requirements, see ***How To Get Published***.

We no longer purchase short softkeys-articles unless they merit being placed in the SPECIAL FEATURES section.

COMPUTIST reserves the right to edit and publish any contribution (whether a letter-to-the-editor or short article-softkey) in the RDEx section. All published contributions become the property and are copyrighted as part of COMPUTIST.

COMPUTIST SPECIAL FEATURES

We will purchase **feature-length** original material on the following subjects:

1) Original program/article combinations;
2) General articles (Apple computing);
3) Softkeys;
4) Hardware modifications;
6) DOS modifications.

COMPUTIST pays upon publication, depending upon the amount of editing and typeset length of the article, from $20 to $100 per article.

Please submit completed manuscripts directly; do not send queries. Previously published material and simultaneous submissions will not be purchased.

HOW TO GET PUBLISHED

Whether you are contributing data to the new Readers Data Exchange or if you feel your submission is worthy of being published as a Special Features, we urge you to follow these guidelines and please PLEASE send contributions and submissions on disk as well as on paper!

All contributions-submissions should be submitted in BOTH printed and disk formats. Your printout must use only one side of the paper. Text should be double-spaced using a non-compressed font with both upper and lower case. A letter quality mode is preferred, with each page torn at the perforation only. Pages need not be stapled together.The first page should contain the following data:

TITLE OF WORK
FULL NAME OF AUTHOR
ADDRESS
PHONE NUMBER

Each page of the manuscript and program listing should include the author's name, the title of the work, and the page number in the upper right hand corner.

The article and any accompanying program should be submitted as a **standard text file on a DOS 3.3 disk.** Label the disk with the title of the work and the author's full name and address. ON DISK, TEXT MUST BE SINGLE-SPACED ONLY. Please identify your editing program.

Disks are always returned as soon as possible. Other materials will be returned only when adequate return packaging and postage is enclosed. We are not responsible for unreturned submissions.

HINTS ON WRITING STYLES

A. Always assume that your reader is a novice and explain all buzzwords and technical jargon. Pay special attention to grammar and punctuation; we require technical competence but also good, readable style.

B. A list of hardware and software **Requirements** should be included at the beginning of the manuscript. When published, this list will be offset from the main text.

C. Include the manufacturer's or publisher's name when a commercial program or product is mentioned.

D. When submitting your own programs, first introduce the purpose of the program and features of special interest. Include background information describing its use. Tips for advanced uses, program modifications, and utilities can also be included. Avoid long print statements and use TABs instead of spaces.

Remember: A beginner should be able to type the program with ease. A program will not accepted for publication without an accompanying article. These articles, as well as articles on hardware and DOS modifications MUST summarize the action of the main routines and include a fully remarked listing.

F. GENERAL ARTICLES may include advanced tips, tutorials, and explorations of a particular aspect of Apple computing.

G. SOFTKEYS must contain detailed step-by-step procedures. For each softkey, first introduce the locking technique used and then give precise steps to unlock the copy-protected program. Number each step. We accept articles which explain locking techniques used in several programs published by the same company.

H. APTs or ADVANCED PLAYING TECHNIQUES can deal with alterations to a program, deleting annoying sounds, acquiring more points in play and avoiding hazards. Again, provide step-by-step instructions to complete each APT and explain each step's function.

Please mail all letters, short articles and softkeys to the *RDEX Editor*. These will be IMMEDIATELY edited, typset and entered into the Readers Data Exchange. Remember that we cannot purchase these contributions. Feature-length articles should be sent to the *Features Editor*. You will be notifieid of the statusd of your submission within 6 to 8 weeks after it is received.

COMPUTIST
PO Box 110846-T
Tacoma, WA 98411

The Hacker's Ultimate Copy & Deprotection Utility

Super IOB Collection !

ALL of our Super IOB controllers (through 1986) in ONE package!

COMPUTIST developed the ultimate copy program to remove copy protection from software:

The Super IOB program.

Since the introduction of Super IOB, COMPUTIST has used this flexible program to deprotect (or partially deprotect) dozens of commercial programs with far ranging protection schemes.

Super IOB deprotects disks by using a modified RWTS (the subroutine in DOS which is responsible for the reading and writing of disk sectors) for reading from the protected disk and then using a normal RWTS for writing to the deprotected disk.

This package contains:

► TWO DISKS (supplied in DOS 3.3). Each disk contains at least 60 Super IOB Controllers including the standard, swap, newswap and fast controllers. Also included is **version 1.5 of Super IOB**, the Csaver program from COMPUTIST No. 13, and a Menu Hello Program that lists the available controllers and, when you select one, automatically installs it in Super IOB and RUNs the resulting program.*

► A reprint of **Disk Inspection and the Use of Super IOB**, from COMPUTIST No. 17. This article explains how to write your own Super IOB controllers.

► **COMPUTIST No. 32**, which contains an extensive article detailing the hows and whys of Super IOB v1.5 and at least 5 articles using the new Super IOB program.

- Several of the controllers deprotect the software completely with no further steps. This means that some programs are only minutes away from deprotection (with virtually no typing).
- The issue of COMPUTIST in which each controller appeared is indicated in case further steps are required to deprotect a particular program.**

The SUPER IOB Collection

Volume 1 of the Super IOB collection covers all the controllers from COMPUTIST No. 9 through No. 26. Also included are the newswap and fast controllers from COMPUTIST No. 32. The following 60 controllers are on volume 1:

Advanced Blackjack, Alphabet Zoo, Arcade Machine, Archon II, Archon, Artsci Software, Bank Street Writer, Barrons SAT, Beyond Castle Wolfenstein, BSW //c Loader, Castle Wolfenstein, Computer Preparation: SAT, Dazzle Draw, DB Master 4 Plus, Death in the Carribean, Dino Eggs, DLM Software, Electronic Arts, F-15 Strike Eagle, Fast Controller, Fathoms 40, Financial Cookbook, Gessler Software, Grandma's House, The Heist, In Search of the Most Amazing Thing, Instant Recall, Kidwriter, Lions Share, Lode Runner, Mastertype, Match Maker, Miner 2049er, Minit Man, Mufplot, Newsroom, Newswap controller, Penguin Software, Print Shop Graphic Library, Print Shop, Rendezvous with Rama, Rockys' Boots, Sargon III, Sea Dragon, Shiela, Skyfox, Snooper Troops, Standard controller, Stoneware Software, Summer Games, Super Controller, Super Zaxxon, Swap Controller, TAC, Ultima I II, Word Challenge, Xyphus, Zaxxon

Volume 2 of the Super IOB collection covers all the controllers from COMPUTIST No. 27 through No. 38. The following 65 controllers are on volume 2:

Alice in Wonderland, Alphabetic Keyboarding, Alternate Reality, Autoduel, Checkers, Chipwits, Color Me, Conan.data, Conan.prog, CopyDOS, Crisis Mountain, Disk Director, Dragonworld, Early Games, Easy as ABC, F-15 Strike Eagle, Fantavision, Fast controller, Fishies, Flight Simulator, Halley Project, Hartley Software (a), Hartley Software (b), Jenny of the Prarie, Jingle Disk, Kidwriter, Kracking Vol II, Lode Runner, LOGO II (a), LOGO II (b), Masquerade, Mastering the SAT, Microtype: The Wonderful World of Paws, Microzines 1, Microzines 2-5, Miner 2049er, Mist & View to a Kill, Murder on the Zinderneuf, Music Construction Set, Newswap controller, Olympic Decathlon, Other Side, Phi Beta Filer, Pitstop II, Print Shop Companion, RDOS, Robot War, Spy vs Spy, Standard controller, Sundog V2, Swap controller, Sword of Kadash, Synergistic Software, Tawala's last Redoubt, Terripin Logo, Threshold, Time is Money, Time Zone, Tink! Tonk!, Troll's Tale, Ultima IV, Wilderness, Word Attack & Classmate, World's Greatest Baseball, World's Greatest Football

☐ **Yes, please send me The Super IOB Collection**

Includes both disks with Super IOB version 1.5, COMPUTIST32, PLUS a reprint of "Disk Inspection and the Use of Super IOB"

☐ **US/Canada/Mexico for $16.00**
☐ **Other Foreign for $20.00**

Send to: **Super IOB Collection**
PO Box 110846-T
Tacoma, WA 98411
(206) 474-5750

Name _____ ID# _____

Address _____

City _____ State _____ Zip _____

Country _____ Phone _____

VISA MC _____-_____-_____-_____ Exp. _____

Signature _____ CP48

Most orders are shipped within 5 working days, however, please allow 4 to 6 weeks for delivery. Washington residents, please add 7.8% sales tax.

US funds drawn on US banks

*Requires at least 64K of memory.

**Although some controllers will completely deprotect the program they were designed for, some will not, and therefore require their corresponding issue of COMPUTIST to complete the deprotection procedure.