For The Serious User Of Apple][Computers

Hardcore COMPUTIST

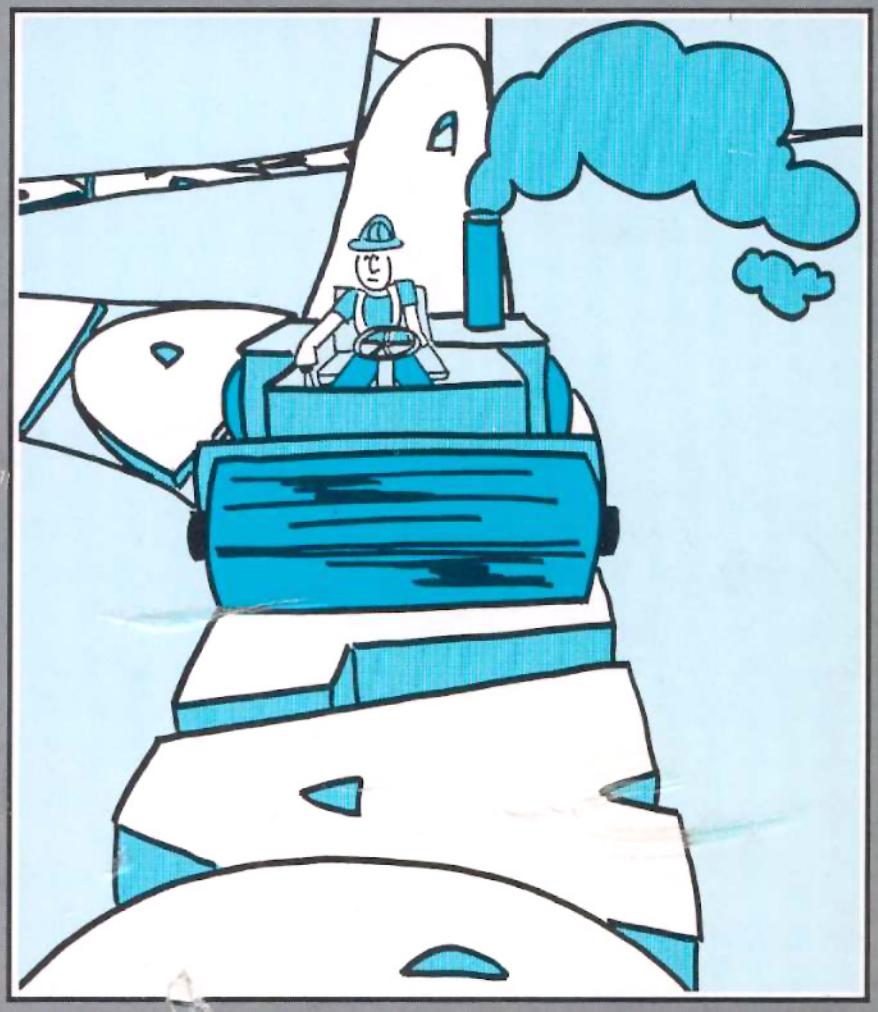
Issue No. 15 $2.50

Mastertype Softkey
Pg. 8

Stickybear BOP Softkey
Pg. 10

Cumulative INDEX to Hardcore Publications 1981-1984
Pg.13

A Boot From Drive 2
Pg. 17



DB Master's Data Compression Techniques
Pg. 19

Hardcore COMPUTIST
PO Box 110846-T
Tacoma, WA 98411

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

TURBO CHARGE YOUR IIe
AND EXPAND APPLE WORKS TO 101K

MEMORYMASTER IIe™ 128K RAM CARD

As an Apple user, you already know all the things your IIe can do. Now Applied Engineering expands that list with MemoryMaster IIe™.

With the MemoryMaster IIe, you'll have up to 192K RAM at your command. Designed expressly for the auxiliary slot in the IIe, the MemoryMaster IIe provides an 80 column video display and up to 128K of memory using just one slot. But the MemoryMaster IIe differs from other large memory cards in one important way, both the 64K and 128K configurations are TOTALLY compatible with software written for the Apple IIe 80 column and extended 80 column cards. In fact, it's not possible to write a program that uses the smaller cards from Apple that won't work with the MemoryMaster IIe.

But the MemoryMaster IIe is not just another piece of unsupported hardware. Each MemoryMaster IIe includes a multi-programming environment program which will enable you to have three different programs ready to run at any moment.

Many of today's software packages for data-base management, word processing, business applications and spread sheets either require or are enhanced by the MemoryMaster IIe. And more and more programs are using double high resolution graphics for which the MemoryMaster IIe is required.

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

- Expands your Apple IIe to 192K memory.
- Provides an 80 column text display.
- TOTALLY compatible with all Apple IIe 80 column and extended 80 column card software—there are NO exceptions.
- Available in 64K and 128K configurations.
- The 64K configuration is USER upgradeable to 128K.
- Can be used as a solid state disk drive to make your programs run over 20 times faster (the 64K configuration will act as half a drive).
- Bank select LED's for each 64K bank.
- Permits your IIe to use double high resolution graphics.
- Uses the same commands as the Apple 80 column card.
- Plugs into the auxiliary slot in the Apple IIe.
- The 64K MemoryMaster IIe will automatically expand VisiCalc to 95K storage, in 80 columns! The 128K MemoryMaster IIe will expand VisiCalc to 141K storage with optional pre-boot disk.
- The 64K MemoryMaster IIe will automatically expand Apple Works to 55K storage, in 80 columns! The 128K MemoryMaster IIe will expand Apple Works to 101K storage with optional expand disk.
- Fully PASCAL and CP/M compatible.
- Lowest power consuming 128K card available.
- Complete documentation included.
- PRO-DOS will automatically use the MemoryMaster IIe as a high speed disk drive.
- Three year warranty.

Low Cost Options.

Apple Works Expand™
Although Apple Works is compatible with both a 64K and 128K MemoryMaster IIe, Apple Works only "sees" it's first 64K bank. Our Apple Works expand program will make a modification to Apple Works that simply lets it know you've got more memory, giving you 101K work space.

$29

Ram Drive IIe™
Sure-fire wait-reduction! Time was, you had to wait for your disk drives. But Ram Drive IIe has changed all that, giving you a large, extremely fast, solid-state disk drive, much faster than floppies or hard disks.

Ram Drive IIe works with either the 64K or 128K MemoryMaster IIe to give you a high speed solid-state disk drive. The Ram Drive IIe software features audio-visual access indicators, easy setup for turnkey operation, and easy menu driven documentation. The program can be modified and is copyable. If you have a 64K MemoryMaster, Ram Drive IIe will act as half a disk drive. If you have a 128K MemoryMaster, Ram Drive IIe will act as a full disk drive. Ram Drive IIe is compatible with APPLESOFT, DOS3.3, PRO-DOS, and PASCAL. Disk also includes a high speed RAM disk copying program. Ram Drive is another disk drive only 20 times faster.

$29

CP/M Ram Drive IIe
CP/M Ram Drive IIe is just like the Ram Drive IIe above, only for CP/M. CP/M Ram Drive IIe runs on any Z-80 card that runs standard CP/M i.e. Applied Engineering Z-80 Plus or Microsoft Soft Card. CP/M Ram Drive will dramatically speed up the operation of most CP/M software because CP/M normally goes to disk fairly often. Fast acting software like dBase II and Wordstar become virtually instantaneous when used with CP/M Ram Drive.

$29

VC IIe Expander
VC IIe Expander will give owners of VisiCalc IIe and the 128K MemoryMaster a total of 141K work space. This disk will pre-boot VC IIe in 1.5 seconds.

$29

Advanced VC IIe Expander
Advanced VC IIe Expander is just like VC IIe Expand only it is designed for Advanced VC IIe. Your work space will be increased to 131K. This disk will pre-boot in 1.5 seconds.

$29

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

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

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

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

Public Domain Software

From the Computer Learning Center's Library

★SPECIAL OFFERS★

(Limited time only)
Order now!!!

S1 EAMON Collectors Offer
Every EAMON scenario and all EAMON utilities. Includes E1 through E76, U1 through U4, D5, D6. *82 EAMON volumes for only $250.*

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

S2 EAMON Introductory Offer
You get the EAMON Master with the Beginner's Cave, The Orb of Polaris, Operation Crab Key, The Feast of Carroll, EAMON Utility 1 and the Dungeon Designer version 6.0. Includes volumes E1, E33, E44, E45, U1, D6. *A popular sampling of 6 EAMON volumes for only $20.*

S3 Art & Graphic
Every Art & Graphic volume. Includes volumes 4 through 13, 93, 94. *12 volumes for only $40.*

S4 Business & Finance
All of the very popular Business & Finance volumes. Includes: 18 through 25. *8 volumes for only $28.*

S5 Games
Golf, hockey, ping-pong, poker, Hi lo, blackjack, cribbage, craps, chess, bowling, parachute, pinball, dragon maze, star trek, fizz bin, tank, keno, war lords, roulette, lunar lander and more. Includes volumes 36 through 57. *22 game-packed volumes for only $75.*

S6 Utilities
Disk utilities, printer programs, assemblers, and disassemblers. It's all here. Includes volumes 74 through 87. *Get 14 Utility volumes for only $48.*

S7 Tutor and Math Combo
From BASIC programming to multiple linear regression, you'll find what you need with these tutor and math volumes. Includes: 1, 2, 3, 59 through 63. *An 8 volume selection from Apple Tutor and Math & Statistics for only $28.*

S8 Chemistry and Math Combo
A choice selection for the scientifically inclined. Includes one Chemistry & Biology and five Math & Statistics: volumes 26, 59 through 63. *All six volumes for only $20.*

S9 Public Domain Software (PDS) Introductory Offer
Includes one volume from each category: Apple Tutor, Art & Graphic, Astronomy, Aviation, Business & Finance, Chemistry & Biology, Demonstration, Education & School, Electronic & Radio, Food, Game, Hello & Menu, Math & Stats, Music & Sound, Passion, Pastime & Other, Unknown, Utility, Apple Bank, and Library. *22 volumes for only $75.*

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

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

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

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

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

☐ Please send me the NEW PDS Catalog

Name _____

Address _____

City_____ St_____ Zip_____

Country _____ Phone _____

VISA/MC _____-_____-_____-_____ Exp _____

Signature _____

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

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

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

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

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

1) Boot the disk in slot 6

PR#6

or

2) Enter the monitor

CALL -151

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

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

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

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

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

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

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

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

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

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

Other helpful books include:

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

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

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

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

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

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

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

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

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

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

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

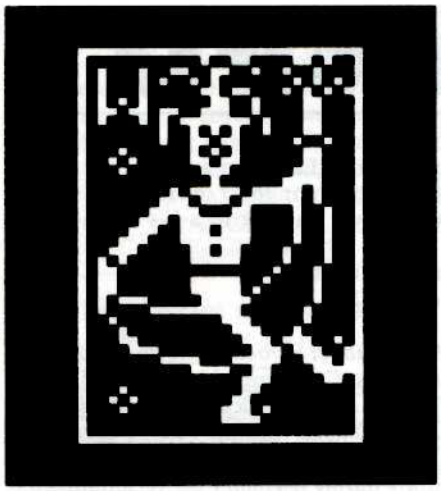
Let Us Hear Your Likes And Gripes: New and longtime readers of Hardcore COMPUTIST are encouraged to let us know what they like and don't like about our magazine by writing letters to our INPUT column. Our staff will also try to answer questions submitted to the INPUT column, although we cannot guarantee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

How-To's Of Hardcore

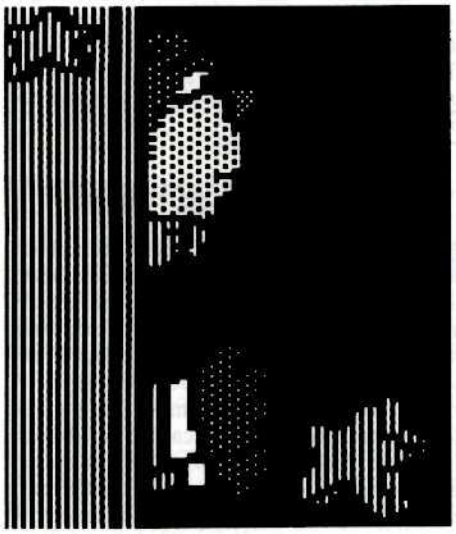
Welcome to Hardcore COMPUTIST, a publication devoted to the serious user of Apple][and Apple][compatible computers. We believe our magazine contains information you are not likely to find in any of the other major journals dedicated to the Apple market.

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

New readers are advised to read over the rest of this page carefully in order to avoid frustration when following the softkeys or typing in the programs printed in this issue. Longtime readers should know what to do next: Make a pot of coffee, get out some blank disks and settle in for a long evening at the keyboard.



Pg. 23



Pg. 10

Address all advertising inquiries to Hardcore COMPUTIST, Advertising Department, 3710 100th St. SW, Tacoma, WA 98499. Mail manuscripts or requests for Writers Guides to Hardcore COMPUTIST, PO Box 110846-K, Tacoma, WA 98411.

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

Entire contents copyright 1984 by SoftKey Publishing. All rights reserved. Copying done for other than personal or internal reference (without express written permission from the publisher) is prohibited.

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

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

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

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

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

Issue No. 15

Hardcore COMPUTIST

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

8 Mastertype Softkey
Are you typing away the short life of your Mastertype disk? Follow this article instead and learn to copy your disk before it requires a costly replacement. *By Peter Rongays.*

10 Stickybear BOP Softkey
Save those Sticky Bears from your youngster's sticky fingers! Make all the backups you need with this softkey for the Sticky Bear series of disks. *By Jerry Caldwell.*

12 MREAD/MWRT Update
MREAD/MWRT may appear to be in good working order, but recent testing has revealed an incompatibility with the DOS file manager. Read this article for a simple fix to the MREAD/MWRT code. *By Gary Peterson.*

13 Cumulative INDEX to Hardcore Publications: 1981-1984
"In which issue was that article on mapping Ultima][published?" "Did a softkey for Apple Galaxian appear in a previous issue?" Questions like these have prompted the staff here at SoftKey Publishing to compile a cumulative index which lists every article, commentary and column found in issues of Hardcore Computing (including UPDATES), Hardcore COMPUTIST, and CORE from 1981-1984.

CORE SECTION

17 A Boot From Drive 2
When performing boot code traces that require repeated booting of normal DOS, you'll appreciate having the capability to boot the disk in drive 2. This helpful program allows you to boot a slightly modified disk. *By Michael Phillips.*

19 DB Master's Data Compression Techniques
This easy to understand article examines the unique and practical method employed by the authors of DB Master (the popular data base program) to greatly increase disk storage. *By Alan & Valerie Floeter.*

23 Boot Code Trace For Tic Tac Show
If you're tired of waiting around to play Tic Tac Show or worried about the short lifespan of your original diskette, you should check out this article. *By Steve Fillipi.*

DEPARTMENTS

4 INPUT
5 BUGS
6 READERS' SOFTKEY & COPY EXCHANGE

Deprotecting The Financial Cookbook
By Pete Levinthal

Softkey For Escape From Rungistan
By Chris Chenault

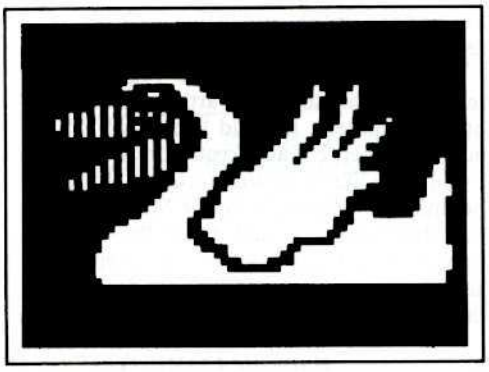
Putting The Byte On Alien Munchies
By Tom Phelps

Softkey For Millionaire & Plato Computer Literacy
By Doni G. Grande

9 ADVENTURE TIPS

26 WHIZ KID

Part One: DOS & The Disk Drive. This month's column examines a portion of the soft switches used by DOS to control the inner workings of the disk drive. *By Ray Darrah.*



INPUT INPUT INPUT

Echo Returns

I am writing for several reasons. First, to offer help to Mr. Carl Meyer (Issue No. 13, page 5) as well as to all readers of Hardcore COMPUTIST. Many people I know have backed up more programs using the ECHO 1.Ø program *without parameter changes* than with any other program available. The story of the program and its author, Ian Agranat, is really quite interesting.

Mr. Agranat wrote ECHO 1.Ø while he was in high school and, when he attempted to market it at a very reasonable price, it was ripped off before he made more than a few dollars.

I admired the program very much and wanted to contact the author. Since his address is not shown in the program, I traced all leads and called around the country searching for him. When I finally located him, I immediately purchased a copy of the program. One of the benefits, as is usually the case, was that I also received the documentation and was able to make even greater use of the program. The author also received some compensation for a program that has been so useful, and at the same time, he gained an incentive to improve his program even more.

The above leads me to the best news of all. Mr. Agranat recently wrote to tell me that he has been working on an even better version of the program, ECHO 2.Ø, which he hopes to have available early next year (it is now being beta tested). I am sure he would be interested in hearing from readers of Hardcore COMPUTIST who want more information on his new program.

Mr. Agranat can be reached at (and the original ECHO program can be ordered from) the address below:

Agranat Systems
1Ø Winthrop Circle
Weston, MA Ø2193

I hope that my information is helpful and of interest. I am especially happy to pass on the information about Ian Agranat since his program has been so effective (I have every other backup program that I know of) and he has received so little acclaim or remuneration for his effort.

Finally, in regard to your request for reader response to adding articles related to other computers. I would like to say (as a non-subscriber, but as a frequent reader) that I would not like to see such a change. I might be slightly interested in Apple Mac articles, but IBM articles would dilute the overall impact of each issue. I feel that a separate sister publication would be very desirable, interesting, and of great value to IBM owners, but not to most Apple owners.

Michael Reese
Murphysboro IL

Mr. Reese: *Thank you very much for solving the Echo mystery for us.*

Undocumented Locksmith Features

I just wanted to write and tell you how much I enjoy Hardcore COMPUTIST. It is definitely my favorite magazine. The thing I appreciate most is the way the softkeys sometimes explain the deprotection process so I can learn something that can be applied in other areas, rather than merely giving a cookbook recipe that leaves the hows and whys unexplained. Of course, I understand that many readers just want to get on with it, and the softkeys provide that, too. But I like to understand why I am doing things.

Which brings me to the main reason for writing. In Issue No. 13, Thomas Dragon wrote a fine article on Locksmith 5.Ø in which he mentioned an undocumented unprotect feature. After tantalizing the readers with this tidbit, he failed to elaborate any further. What is this feature and how is it implemented? He mentions that methods for making it work have not been provided to the general population of users, and that is exactly what I rely on Hardcore COMPUTIST to do. Mr. Dragon said he had not seen this feature work, so perhaps he did not know how to implement it.

Can any of the readers help? A call to Alpha Logic failed to get me anywhere, even though I am a registered Locksmith 5.Ø owner and never (just like they asked) use it except to backup my own legitimately purchased software, or convert it to hard disk. They said they couldn't reveal that information (maybe they really didn't know themselves).

I purchased 5.Ø when it first came out and could not, until recently, find anything I could copy with it. The saying going around is that if you can figure out how to copy something with Locksmith by reading their manual, then you don't need Locksmith. Well, I can relate to that, because it's taken me nearly a year to learn how to use their Locksmith Programming Language. It's definitely not for someone who doesn't even know BASIC. But, after reading many issues of Hardcore COMPUTIST, Beneath Apple DOS, etc. it's starting to make sense to me.

So, keep up the good work, and put Mask of the Sun (the old softkey won't work on the new version) on the Most Wanted List for me please.

P.S. I can't find enough good things to say about Don Lancaster's Absolute Reset Modification for the //e. It works great. You'll never go back to a][+.

James Faubus
Godley TX

Mr. Faubus: *Documentation for the "unprotect" feature of Locksmith 5.Ø will probably remain offically unavailable until Alpha Omega decides to release this information. Mr. Dragon learned of this feature during telephone conversations with the author of Locksmith 5.Ø, but has never actually seen it work. Hopefully it, and other features of Locksmith, will be implemented and documented by Alpha Omega in the near future. Notice that in the Locksmith 5.Ø manual, algorithmns 6Ø-6F are defined as being reserved.*

Hardware Conflict #876

I am writing to you for help with The Networker by Zoom Telephonics. A short time ago I bought a copy of the program to use with my Hayes. It booted up fine, and was all ready to run after I had changed the default slot of the modem (default = 4). But, after I bought the program, I bought an 8Ø-column card and Z-8Ø card (slots 3 and 4, respectively). Now when I boot it up, it defaults to slot 4 where the Z-8Ø card is and hangs the system. Do you know of a sector edit that I can use to change the default? Any help that you or your readers can give me would be much appreciated. My other alternative is to pull the card when I want to use the program (and that's a pain).

The other reason that I am writing is to express an opinion. In regard to your question in Issue No. 11, PLEASE don't expand to other computers. Selfish? Maybe, but I think that it is really nice to be able to read through your magazine and not have to worry whether or not a particular article has to do with Apples or something else. I've also seen many magazines fade too far away from Apples (Compute, for example).

Finally, do you have any plans for reprinting old back issues (No.s 2, 3, 5, & 7)? I wanted to get those, but didn't move fast enough. When you say limited supplies, you really mean it!

Anyway, thank you for any help, and for putting out such a great magazine.

Paul Hopkins
Tolland CT

Mr. Hopkins: *Sorry that we don't have any tips to offer you in solving the hardware conflict you describe. Hopefully, one of our readers will be able to offer you a suggestion.*

Thanks for your input on the content of our magazine. Judging from the responses we have received thus far, the majority of our readers seem to hold a similar opinion when it comes to covering other micros.

At this time we have no plans to reprint any of the back issues of Hardcore, although in 1985 we do plan to reprint the softkeys contained in our publications up through Hardcore COMPUTIST No. 6.

Likes, Gripes and An Offer You Can't Refuse

I am writing for several reasons. First, I would like to know if there are any readers out there who would be willing to sell any of the following out-of-print issues of Hardcore:

1981-1982

Hardcore Computing 1
UPDATE 1.1-Newsletter
Hardcore Computing 2

1983

Hardcore COMPUTIST 2

Second, I have a set of tutorials written that discuss basic techniques used for cracking protected disks. I have found that they are invaluable for starting to crack your own programs. I will sell them to anyone who is interested if they will send me a disk and $4.00 to cover copying and return postage costs.

Third, I have compiled an index for all the Hardcore COMPUTISTs, CORE that I have (1-3), the promotional flyer and all Hardcore COMPUTISTs after Issue No. 3. I will put it on the disk mentioned above on request.

The fourth thing that I wish to say is that I would disapprove of expanding this magazine to cover IBM and MacIntosh. If there is enough demand, then starting another magazine would be the appropriate thing to do. That way, no one would have to pay for articles that are totally useless, even to the extent that techniques used in deprotecting disks would be useless.

Finally, I have two complaints regarding the general format of your magazine. First, why must you split articles with a "continued on page X"? It is terribly annoying to the reader to have to page through a magazine to find the rest of a softkey, program or article. The second complaint I have regarding the format of your magazine is more serious, at least in my mind. When you print an assembly language listing of a program, please don't use all the special features of your assembler. It would greatly increase the usefulness of those listings if you would employ only standard commands and less complicated labels. One example:

```
LDA #$3D0-END.AMP+BEG.AMP
```

This label was used with the CSaver program, line 4005. If you continue the same manner of usage of your assembler, could you at least tell me what the "/" means? It was used in the CSaver article in several expressions including:

```
LDA /$3D0-END.AMP+BEG.AMP
(line 400A)
```

I have checked the S-C Assembler directives that you printed in Hardcore COMPUTIST No. 5 and it is not listed.

One last question. Do the programs that you sell on your disks work even if there has been an error in the article? Are the programs updated if you make alterations in them (Modified ROMs and Super IOB, for example)?

Despite these complaints, I like your magazine and hope you will continue the high quality of work you have shown so far.

Kevin Lepard
102 S. Lakeview
Sturgis MI 49091

Mr Lepard: *We hope the following will clarify the meaning of the two lines from the CSaver source code you refer to.*

The S-C Macro Assembler, which was used to produce the source code for CSaver, uses the pound sign (#) and the slash (/) to the indicate the lower byte and the upper byte, respectively, of the address to which a label or an expression is equated. In these two lines the expression $3D0-BEG.AMP+END.AMP is used to calculate where the routine which switches the RAM card on and off can be placed without overwriting the DOS vectors which start at $3D0. Using an expression in this manner makes the program easier to modify, but does make the source code a bit more difficult to understand. Other assemblers such as Merlin/Big Mac can also handle expressions of this sort, except they will use characters other than # and / (< and >, for example) and may not accept periods (.) within labels.

Beginning with Hardcore COMPUTIST No. 14, we have been printing listings of the source code as it is assembled so that readers can immediately determine the values which result from the assembly of an an instruction. We are also trying our best to to hold down the number of continued on page xx's in our magazine.

In response to your question about the Hardcore Library Disks: Any bugs in the programs on the Library disks are corrected as soon as we become aware of them. The programs are not, however, updated if new versions of the program are printed.

Sensible Speller Clarification

Gerald Gibson's letter in Issue No. 13 of Hardcore COMPUTIST made me realize that I hadn't adequately identified a parameter I provided in my article, "Copy][Plus (4.4c): Update of an Old Friend" which appeared in Hardcore COMPUTIST No. 11. My apologies.

The parm in question is for Sensible Speller, Version 4.1c, not for 4.0k as Mr. Gibson surmised.

Philip G. Romine
Cookeville, TN

Bugs In Hardcore COMPUTIST No. 13 & 14

HC No. 13 In "Beneath Beyond Castle Wolfenstein" (pg. 12) we omitted a necessary Super IOB controller. (See the slightly modified version of the Swap Controller below).

```
1000 REM BEYOND WOLFENSTEIN
     CONTROLLER
1010 TK=3:ST=0:LT=35:CD=WR
1020 T1=TK:GOSUB 490:GOSUB 360
     :ONERR GOTO 550
1030 GOSUB 430:GOSUB 100:ST=ST
     +1:IF ST<DOS THEN 1030
1040 IF BF THEN 1060
1050 ST=0:TK=TK+1:IF TK<LT
     THEN 1030
1060 GOSUB 490:TK=T1:ST=0:
     GOSUB 360
1070 GOSUB 430:GOSUB 100:ST=ST
     +1:IF ST<DOS THEN 1070
1080 ST=0:TK=TK+1:IF BF=0
     AND TK<LT THEN 1070
1090 IF TK<LT THEN 1020
1100 HOME:PRINT "EVERYTHING^
     O.K.^NO^DOS^ON^COPY":END
10010 IF PEEK(6400)<>162 THEN
     PRINT CHR$(4)"BLOAD^
     RWTS.BEYOND^WOLF,A$1900"
```

HC No. 14 In "Softkey for Knoware" (pg. 6), the procedure may not work on all versions of the program. An alternate procedure is printed below.

1) COPYA the original disk #1.
2) Insert the copy of disk #1 into drive 1 and type the following

```
CALL -151
BLOAD ONESHOT.OVR
2010:A9 00 EA
BSAVE ONESHOT.OVR,
A$2000,LSFBA
BLOAD BOOTKW
9FD:EA EA
A01:A9 00
BSAVE BOOTKW,A$800,L$852
```

3) Use COPYA to make copies of Disks 2 and 3.

In the article "Putting Locksmith 5.0 Fast Copy Into A Normal Binary File" (pg. 27), add this instruction to Step 7:

```
B60:00
```

READERS' SOFTKEY & COPY EXCHANGE

Deprotecting The Financial Cookbook from Electronic Arts

By Pete Levinthal

Requirements:
A Replay card, Wildcard or other 48K copy card
A blank disk
COPYA from the DOS 3.3 System Master
A sector editor
Financial Cookbook from Electronic Arts

Here is a quick and dirty method to copying the Financial Cookbook from Electronic Arts. This method requires a copy card such as the Replay or Wildcard.

Since the Financial Cookbook requires 64K of memory to run, it is no big deal to use a copy card to load the program back in. The trick to this method is to copy memory at just the right time. The object is to escape the protection but catch the program before it starts loading into the RAM card (thus preventing us from having to copy 64K of memory).

The Cookbook disk is completely copyable with COPYA except for track 6, which is the nibble count track. The boot proceeds like any other Electronic Arts release: it loads a title page and then does a nibble count and loads the program. Listen to the program load and you will hear the nibble count (sounds weird, eh?). We want to copy memory after the nibble count but before the RAM card is activated. Just after the mibble count, the drive will stop and the text page will flutter for a second. Press the copy card switch at this instant. You will only have a second or two to do it. You may have to practice a few times.

After this process, use your copy card utility disk to make Binary files of the memory.

Now just copy tracks $12 to $14 of the original Financial Cookbook to a blank initialized disk using a slightly modified COPYA. Now run a sector editor and change bytes $80 to 00 00, bytes $84 to 00 00, bytes $88 to 00 00, bytes $3C to 00 00 and bytes $40 to 00 00 to allocate tracks $12 to $14 and the DOS tracks as used in the VTOC. Finally, copy the bfiles of memory to this disk using FID or some other utility.

You're all done! Just BRUN the memory files and, if all went correctly, the program will restart, read in some data from tracks $12 to $14 and all is fine.

Step-By-Step

1) Boot the Financial Cookbook and after the title page and the nibble count, and just after the drive stops (for just a second!), hit the copy card switch.
2) Copy all 48K of memory.
3) Process the copied memory using your copy card utility disk to create normal Binary files from it.
4) Start COPYA going and then break it when the title page comes up

```
RUN COPYA
CTRL-C
```

5) Enter the monitor

```
CALL -151
```

6) Type the following so that COPYA will only copy specific tracks

```
302:16
35F:16
2DE:20 B0 02
2B0:A9 0F 8D D1 02 8D D2 02 60
3D0G
70
RUN
```

7) When the copy is done, get out your sector editor and make the following changes to your freshly created disk.

Track	Sector	Bytes	To
$11	$00	$80,$81	$00 $00
$11	$00	$84,$85	$00 $00
$11	$00	$88,$89	$00 $00
$11	$00	$3C,$3D	$00 $00
$11	$00	$40,$41	$00 $00

8) Finally, transfer the memory files to this disk using FID.

Now you're all done! Just BRUN the memory files to restart the program.

Softkey For Escape From Rungistan

By Chris Chenault

Escape From Rungistan
Sirius Software
10364 Rockingham Drive
Sacramento, CA 95827
$29.95

Requirements:
48K Apple or an Apple //e
One disk drive with DOS 3.3
Escape From Rungistan
A sector editor

Escape from Rungistan is a unique adventure featuring graphics, sound and animation. At the beginning of the game, you awake in a foreign prison and hear the guard say that you are to be shot at sunrise. There is no other choice but to attempt to break out. As the game progresses, you battle bears, snakes and test your skiing ability. This adventure is written with an Old West flavor and the author shows a great sense of humor. To aid your efforts at playing this game, the author added a clue file to keep you from getting too flustered.

The locking procedure on this disk consists of a combination of many fairly simple procedures, but together they give you endless trouble. I used DiskView to discover that this is a DOS 3.3 disk with the last byte of the address prologue marker changed to $F7 from $96 on tracks $03-$22.

Using this pre-analysis, it seems to be a disk that a slightly modified COPYA would deprotect. Unfortunately, because some of the last few tracks have been damaged for protection, this won't work.

I finally came up with the following method after discovering that it had a normal CATALOG on the correct track and a hello program called START. I discovered this with the help of DiskEdit. Now for the procedure.

Step-By-Step

1) Boot the DOS 3.3 system master disk.
2) Insert a blank disk and format it with START as the boot file

```
INIT START
```

3) Re-insert Master and load FID

```
BLOAD FID
```

4) Make sure your Escape From Rungistan disk is write protected.
5) Drop into the monitor

```
CALL -151
```

6) Modify DOS so that the last byte of the address prologue is ignored

```
B969:29 00
```

7) Start FID going

```
803G
```

8) Copy all the files using the "=" feature to the disk you INITed in Step 2.
9) When FID informs you that a file named START already exists, hit RETURN so that we will get the file from the Rungistan disk.

The copy at this point won't work because the DOS on the original disk has strange DOS commands. This is how they have been altered:

Normal Command	Rungistan Command
RUN	ARC
CLOSE	SVRTT
READ	DNRT
OPEN	CBSE
MAXFILES	FILMAXES
BSAVE	AVESB
BLOAD	ODABL
ALL OTHERS	(RUBBED OUT)

10) use your sector editor to read track $01, sector $07 from the original disk and write it to your new unlocked disk. This changes the commands to "Rungistan" Commands).

You're done! To CATALOG your new

disk, study the code and do APTs, simply boot a normal disk before starting. Your copy of Escape From Rungistan is COPYAable.

Putting The Byte On Alien Munchies

By Tom Phelps

Alien Munchies
Gentry Software
9421 Winnetka Avenue
Chatsworth, CA 91311
(213) 701-5161
$29.95

Requirements:
Apple with 48K
Means of resetting into the monitor
One slave disk with no HELLO program

Alien Munchies by Gentry Software is your common, everyday, run of the mill "Fry the aliens on your barbecue grill!" arcade game. (Sure, everyday huh?) This game is slow at the beginning and patient playing is needed to get to the much more exciting 2nd and 3rd type of aliens (10,000 and 20,000 points respectively).

The method used to deprotect this game illustrates a very useful move routine you can use in your own cracks. The problem of running out of men before reaching these more challenging stages is solved with an example of the art of Advanced Playing Techniques.

Step-By-Step

1) Boot Alien Munchies.
2) Reset into the monitor after the picture has come onto the screen.
3) Move page eight out of the way for a boot

2000<800.8FFM

4) Boot a slave disk with no HELLO program.
5) Enter the monitor

CALL -151

6) Move page eight back

800<2000.20FFM

At this point, the Alien Munchies program is in memory and in its proper memory locations. Rather than save a huge chunk of memory, let's save some disk space with a move routine. Furthermore, to bring the game under control a little better, let's include an APT routine.

7) Compact the code

3000<6000.6FFFM

8) Enter the space saving, relocatable move routine

```
2000: A0 00 A9 00 85 00 85 02
2008: A9 30 85 01 A9 60 85 03
2010: B1 00 91 02 E6 02 E6 00
2018: D0 F6 E6 03 E6 01 A5 01
2020: C9 40 D0 EC EA
```

If you disassemble this code, it should look like this:

```
2000-   A0 00      LDY   #$00
2002-   A9 00      LDA   #$00
2004-   85 00      STA   $00
2006-   85 02      STA   $02
2008-   A9 30      LDA   #$30
200A-   85 01      STA   $01
200C-   A9 60      LDA   #$60
200E-   85 03      STA   $03
2010-   B1 00      LDA   ($00),Y
2012-   91 02      STA   ($02),Y
2014-   E6 02      INC   $02
2016-   E6 00      INC   $00
2018-   D0 F6      BNE   $2010
201A-   E6 03      INC   $03
201C-   E6 01      INC   $01
201E-   A5 01      LDA   $01
2020-   C9 40      CMP   #$40
2022-   D0 EC      BNE   $2010
2024-   EA         NOP
```

9) Type in the following APT routine. Unlike the Sammy Lightfoot APT (Hardcore COMPUTIST No. 5), let's make ours optional so you can play normal style or with **infinite men** and a counter in your number of barbecues to tell you what level you're on.

```
2025: 20 2F FB
2028: 20 58 FC A2 0B 20 4A F9
2030: A2 00 BD 62 20 20 ED FD
2038: E8 E0 12 D0 F5 2C 10 C0
2040: AD 00 C0 C9 D9 F0 07 C9
2048: CE D0 F5 4C 00 08 A9 EA
2050: 8D C9 10 8D CA 10 8D CB
2058: 10 A9 01 8D FD 12 4C 00
2060: 08 EA C9 CE C6 C9 CE C9
2068: D4 C5 A0 CD C5 CE A0 A8
2070: D9 AF CE A9 00
```

A disassembly of this would show:

```
2025-   20 2F FB    JSR   $FB2F
2028-   20 58 FC    JSR   $FC58
202B-   A2 0B       LDX   #$0B
202D-   20 4A F9    JSR   $F94A
2030-   A2 00       LDX   #$00
2032-   BD 62 20    LDA   $2062,X
2035-   20 ED FD    JSR   $FDED
2038-   E8          INX
2039-   E0 12       CPX   #$12
203B-   D0 F5       BNE   $2032
203D-   2C 10 C0    BIT   $C010
2040-   AD 00 C0    LDA   $C000
2043-   C9 D9       CMP   #$D9
2045-   F0 07       BEQ   $204E
2047-   C9 CE       CMP   #$CE
2049-   D0 F5       BNE   $2040
204B-   4C 00 08    JMP   $0800
204E-   A9 EA       LDA   #$EA
2050-   8D C9 10    STA   $10C9
2053-   8D CA 10    STA   $10CA
2056-   8D CB 10    STA   $10CB
2059-   A9 01       LDA   #$01
205B-   8D FD 12    STA   $12FD
205E-   4C 00 08    JMP   $0800
2061-   EA          NOP
2062-   C9 CE       CMP   #$CE
2064-   C6 C9       DEC   $C9
2066-   CE C9 D4    DEC   $D4C9
2069-   C5 A0       CMP   $A0
206B-   CD C5 CE    CMP   $CEC5
206E-   A0 A8       LDY   #$A8
2070-   D9 AF CE    CMP   $CEAF,Y
2073-   A9 00       LDA   #$00
```

10) Finally, add a JMP to our special routines before the actual program starts

7FD:4C 00 20

11) Save the new and improved Alien Munchies

BSAVE ALIEN MUNCHIES, A$7FD,L$5803

Don't stop with these modifications. Do some of your own!. For example, the title page is just sitting on page two of hi-res waiting to be altered or replaced with one of your own!

Softkey for Millionaire & Plato Computer Literacy

By Doni G. Grande

Millionaire
Blue Chip Software
19818 Ventura Blvd.
Woodland Hills, CA 91364
$59.95

Plato Computer Literacy Introduction
Control Data Publishing, Inc.
PO Box 261127
San Diego, CA 92126
$45.00

Requirements:
A way to Reset into the monitor
Super IOB w/Swap Controller (HC No.9)
One blank disk

Both the above programs, while using very different protection schemes, may be copied very easily using the Swap Controller from Hardcore Computist No. 9. The procedure is as follows:

1) Boot the original disk. When the disk stops spinning, use your method to reset into the monitor.
2) Move the modified DOS to a safe area of RAM

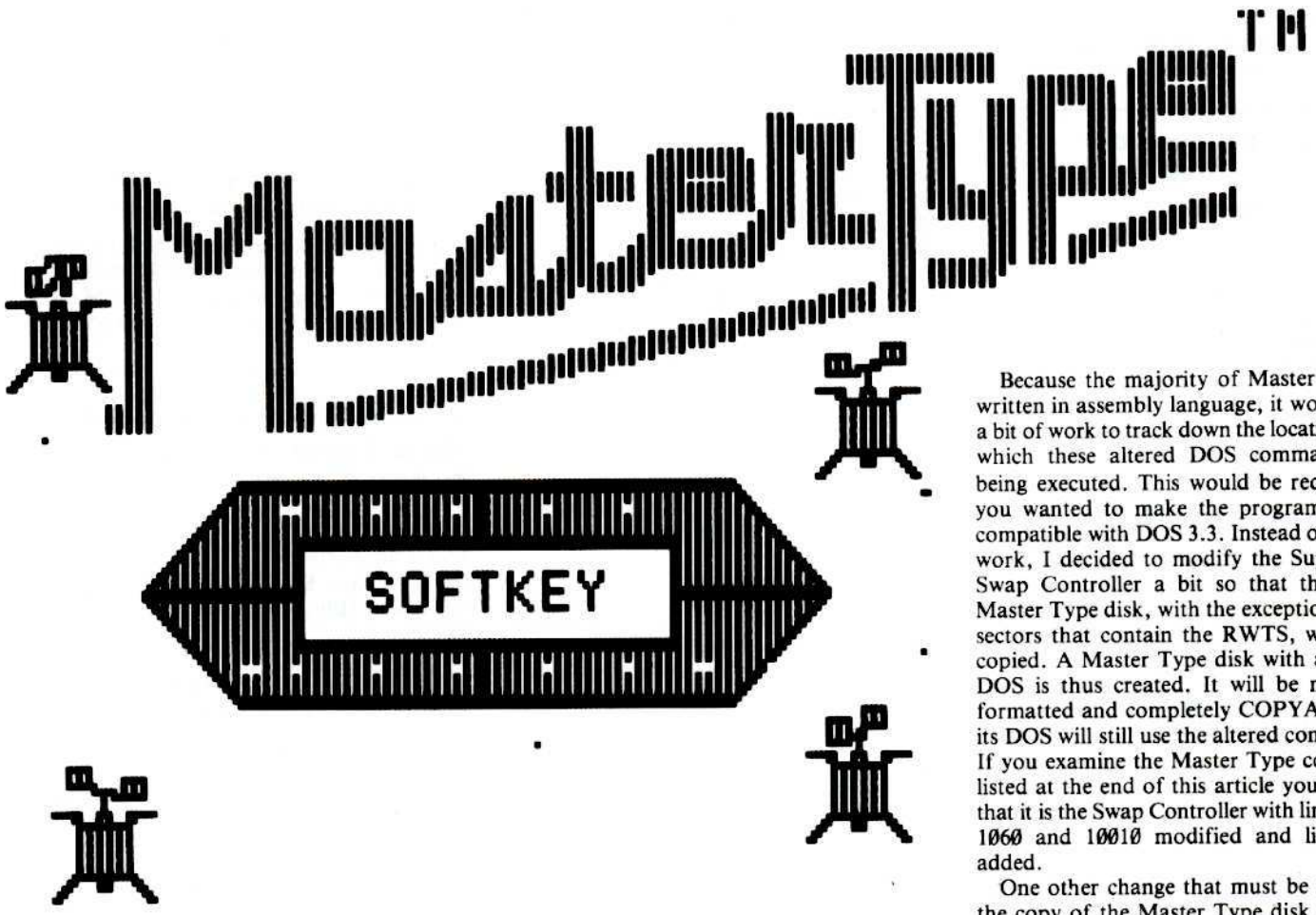
1900<B800.BFFFM

3) Boot with a normal slave disk.
4) Save the modified DOS to your Super IOB disk

BSAVE RWTS, A$1900, L$800

5) Run the Swap Controller version of Super IOB. The disks will copy and work normally.

You now have a normal copy of the disk. If you use the above procedure to copy Millionaire, you will find that you can cut the startup time (6 minutes normally!) by using a fast DOS such as Diversi-DOS.



By Peter Rongays

Master Type v1.7
Lightning Software
PO Box 11725
Palo Alto, CA 94306
$40.00

Requirements:
48K Apple][Plus or equivalent
Super IOB v1.2
Old monitor F8 ROM or Copycard
One blank disk

Bruce Zweig's Master Type has been one of the best-selling pieces of software for the Apple][ever since it was released back in 1981. The program's most unfortunate drawback is a common one: it is copy protected. Since the program is undoubtedly being used in a fair number of schools thoughout the USA and elsewhere, there is a corresponding demand for information on how to backup this disk. The reluctance of an instructor to turn over the only copy of a $40 program to a group of precocious eight-year-olds is entirely understandable.

Luckily, as I found out, the program Super IOB has little difficulty in copying Master Type. This article will explain how to make the backup.

A Titanic Fantasy

Judging from the EDD III parameter list for Master Type, it appears to have been protected by "Lock-It-Up", one of those copy-protection utilities that you may have seen advertised as being able to produce an "uncopyable" disk (Note: The "uncopyable disk" is close kin to the "unsinkable ship"). Indeed, Master Type is protected fairly well, as evidenced by its several month stay on the Hardcore "Most Wanted List" and the difficulty of backing it up with a bit copier.

Upon booting Master Type, the familiar Applesoft prompt will appear while the game loads. The appearance of this prompt is usually an indication that a disk can be copied by Super IOB with the Swap Controller installed, as long as its RWTS can be captured by some means. This is the case with Master Type, but the Super IOB copy made with the standard Swap Controller will not work because the text of some the DOS commands has been altered. You can verify this by booting Master Type, halting it with an old monitor RESET or NMI and then examiming memory from $A884 to $A908. About half of the normal DOS commands have been blanked out entirely and, of those that remain, nine are the same as with normal DOS and five have had their text changed. Table 1 shows the Master Type DOS commands that remain and their DOS 3.3 counterparts.

Because the majority of Master Type is written in assembly language, it would take a bit of work to track down the location from which these altered DOS commands are being executed. This would be required if you wanted to make the program totally compatible with DOS 3.3. Instead of all this work, I decided to modify the Super IOB Swap Controller a bit so that the entire Master Type disk, with the exception of the sectors that contain the RWTS, would be copied. A Master Type disk with a hybrid DOS is thus created. It will be normally formatted and completely COPYAble, but its DOS will still use the altered commands. If you examine the Master Type controller listed at the end of this article you will see that it is the Swap Controller with lines 1010, 1060 and 10010 modified and line 1065 added.

One other change that must be made to the copy of the Master Type disk is to fill in two of the free areas in the RWTS with hexadecimal 60's. This is necessary because Master Type can store user-created lessons on normally formatted disks. On the original Master Type disk, this free space contains some code which modifies the RWTS depending upon whether the original or a data disk has to be read. This code won't be needed, or present, on the copy. Therefore, we will just replace it with a bunch of machine language RTS's.

With a little more effort, I'm sure Master Type could be made to work with a totally normal DOS, but it probably is not worth the effort to do so unless you have some modifications or enhancements you would like to add to it. I will leave that chore up to the more ambitious readers.

Theory out of the way, let's get cracking!

Making the Copy

1) Boot up the original Master Type disk and, after the program has been loaded, stop it with a RESET or NMI.
2) From the monitor, move the Master Type RWTS to a "safe" location

1900<B800.BFFFM

3) Boot up a DOS 3.3 slave disk and then BSAVE the Master Type RWTS onto a disk which contains Super IOB

BSAVE RWTS.MASTER TYPE, A$1900,L$800

4) Enter the monitor and fill in two of the "holes" in the RWTS with 60's (RTS's) before initializing a blank disk

```
CALL -151
BA69:60 N BA6A<BA69.BA94M
BCDF:60 N BCE0<BCDF.BCFEM
INIT HELLO
```

5) Load Super IOB, install the Master Type Controller and then make a copy to the disk initialized in Step 4 above. Note: *Do not* reformat the disk.

You now have no excuse for not learning how to touch type.

Table 1

DOS 3.3 Command	Master Type Command
INIT	SAVE
DELETE	KILLDE
CLOSE	CLOSE
READ	READ
EXEC	EXEC
WRITE	WRITE
OPEN	OPEN
CATALOG	CATNDOG
NOMON	NOMON
PR#	PR#
IN#	IN#
FP	FP
BLOAD	YZ123
BRUN	YZ23

Master Type Controller

```
1000 REM MASTER TYPE CONTROLLER
1010 TK = 0 : ST = 10 : LT = 35 : CD = WR
1020 T1 = TK : GOSUB 490 : GOSUB 360 :
     ONERR GOTO 550
1030 GOSUB 430 : GOSUB 100 : ST = ST + 1
     : IF ST < DOS THEN 1030
1040 IF BF THEN 1060
1050 ST = 0 : TK = TK + 1 : IF TK < LT
     THEN 1030
1060 GOSUB 490 : TK = T1 : ST = 0 : IF TK
     = 0 THEN ST = 10
1065 GOSUB 360
1070 GOSUB 430 : GOSUB 100 : ST = ST + 1
     : IF ST < DOS THEN 1070
1080 ST = 0 : TK = TK + 1 : IF BF = 0 AND
     TK < LT THEN 1070
1090 IF TK < LT THEN 1020
1100 HOME : PRINT "EVERYTHING^O.K.^
     NO^DOS^ON^COPY" : END
10010 IF PEEK (6400 ) <> 162 THEN
     PRINT CHR$ (4 ) "BLOAD^
     RWTS.MASTER^TYPE,A$1900"
```

```
1000  - $356B      1065  - $316D
1010  - $4E99      1070  - $3968
1020  - $DD03      1080  - $C1E9
1030  - $CB02      1090  - $A08B
1040  - $6939      1100  - $058F
1050  - $BFFC      10010 - $CD27
1060  - $5359
```

ADVENTURE TIPS ADVENTURE TIPS

Time Zone
Sierra On-Line

Bothered by Tyrannosaurus Rex? Just avoid him.
The mastadons can't reach you if you're up a tree.
You'll have to spear the Sabertooth Tiger with a hefty weapon. You'll find it in the distant past.
Desert people eat dates to keep from going hungry. You'll need some money from Cleopatra to buy them.

Mission Asteroid
Sierra On-Line

Can't get the spaceship going? Try pushing the throttle.
You'll need a flight plan to find the asteroid. Push these: black, black, orange, white, white, white, blue, white, orange, orange.
The asteroid has no atmosphere, and you'll die if you go out without "wearing" protective gear.

* Enchanter
Infocom, Inc.

Pay attention to those dreams!
Don't get caught with your hands full.
You're going to need the sacrificial dagger.
Follow the tracks.

* Zork I
Infocom, Inc.

Don't kill the thief until you've given him a gift.
Learn to operate the dam.
Read the whole book.
Ever heard of inflatable PLASTIC rafts?

* Contributed by Tan Wee Meng

Ultima
Origin Systems

To get lots of gold, go to Lord British and offer gold. Then enter the town and purchase food. Pick a good spot. Hold the attack button down or, if you are in a craft, hold down the fire button. The monster should come and go quickly while you rake in the gold.

Ultima II
Origin Systems

To get past the annoying title pages, hit RETURN and ESC a couple of times. This will bypass the rest of the pages and jump you right to the menu and ask you whether you want to play, create a character, or see a demo.

Contributed by Dr. Duplicate

Death in the Caribbean
Microlab

Get the wagon to move that rock and load 'er up.
The only way to stop the ants is to plug their hole.
Clear the rockslide away and you'll discover something useful.
If you're going to pull the rope, you'd better be prepared for the worst.

Gruds in Space
Sirius Software

Don't worry about picking up items on your ship. You won't need them.
Read the transmission? Set coordinates for Saturn 64-18-52 , then go West and set the Teleport coordinates 77-34-40.
Take everything you can. Even from the Gruds.
Knock to get in at Lord Deebo's.
You must pay tribute to Lord Deebo. Try giving him your ship.

† Zork III
Infocom, Inc.

"Knock and the door shall be opened. Seek and ye shall find."
Read H.G. Wells before entering the technology museum.
Go jump in a lake.
Getting past the Guardians of Zork can be done, but it's a trick. It's all done with mirrors.
Look very carefully at the Dragon Master. You want to be like him in every way.

† Starcross
Infocom, Inc.

Why does the ray gun shoot silver rays when it misfires, but orange rays if you fire it again?
When floating weightless, remember that every action has an equal and opposite reaction.

† Contributed By Cullen Johnson

*see No.17, page 5 - additional info (letter)

Stickybear Series
Xerox Educational Software
Computer Software Division
Dept. B-I
245 Long Hill Road
Middletown, CT 06457
$39.95 each

Requirements:
Apple][Plus or equivalent
Blank disks
Disk Search Utility (Inspector, ZAP, etc.)
COPYA
Copycard or Integer card (optional)

For those who have looked into the excellent Stickybear series, the disks can be read by a normal sector editor or copied by COPYA with the exception of one sector. The protection scheme is based solely upon the data residing on this one sector. On the different programs in the series, the location of this protection will vary, but it is generally sector $0F on track $01 or $02. This sector is checked at various intervals to validate the presence of the original Stickybear disk and, as many of you probably know, is very hard to duplicate even with the latest bit copy programs.

Analysis of the booting process reveals that various tracks are loaded into memory, and then a jump is taken to a sequence of machine language instructions that are nearly identical on several of the Stickybear disks. In this series of instructions, an IOB (Input/Output Block, see pages 94-98 of the DOS 3.3 Manual or Chapter 6 of Beneath Apple DOS) is created for reading the protected sector. Next, a subroutine which alters the RWTS "post-nibblization" routine (used to convert the 6&2 encoded nibbles read from the disk into 8 bit bytes) at $B8C2 is called. This change to the postnibblization routine makes the first instruction of it a jump to a different postnibblization routine located on hi-res page 2 (at $4E9F on Stickybear Bop). When the RWTS is called, the protected sector is read into memory. To restore the postnibble routine to its original form, a second call is made to the routine which changed it in the first place.

The technique for producing a backup is as follows:

Using a modified COPYA, the original disk will be copied, ignoring the the read error on the protected sector. The COPYAd backup will then be searched with a disk search utility for the code which reads the protected sector. Once this code has been found, the memory where the protected data is read into must be identified by examining the code which sets up the IOB. The original Stickybear disk is then booted up, allowed to read the protected sector into memory and the program is then halted by an old monitor RESET, an NMI card, or by modifying one of the copies of the disk. This translated data can then be recovered and written back to the proper sector on the COPYAd disk with a sector editor. Finally, because this sector on the backup will no longer be protected, the routine which alters the postnibblization routine must be disabled.

Modifying COPYA

As pointed out in the documentation for Bag of Tricks, COPYA can be modified to permit disks with I/O errors to be copied (except for the sectors which are unreadable). This can be done by changing byte $E1 of file COPY.OBJ0 to an EA (NOP instruction). The procedure to do this is as follows:

1) Insert the disk with COPYA on it and load the COPY.OBJ file

BLOAD COPY.OBJ

2) Alter COPY.OBJ so that it will ignore unreadable sectors

POKE 929,234
BSAVE COPY.OBJ0,
A$2C0,L$10B

(Note: BSAVE this to another disk which contains the Applesoft program COPYA, unless you want to have the modified version on your System Master.)

Now, run the modified COPYA and, when prompted, copy the original disk to a blank. You should hear the drive recalibrate twice when it comes across the protected sector. If you do not have any means of resetting into the monitor, make two copies of the Stickybear disk at this time.

The backups made with the altered COPYA will be *almost* identical to the original now, except that they will not work. The sector that is protected on the Stickybear original will be formatted, but will contain no data. We will rectify this situation shortly.

The next step in creating the backup is to locate the sector which contains the instructions for setting up the IOB to read the protected sector. These instructions are as follows:

```
A9 tt     LDA #$tt    Track with protected
                      sector
8D EC B7  STA $B7EC   Store it in the IOB
A9 ss     LDA #$ss    Protected sector
8D ED B7  STA $B7ED   Put it in the IOB,
                      too
```

You will need a disk search utility like ZAP or Inspector to search the backup for the instruction STA $B7EC (8D EC B7). This code could be located almost anywhere on the disk, but tracks $1, $2 and $11 are likely candidates. Once you find this code, disassemble the sector and look for some more code which has the form:

```
Low byte of data buffer
   A9 ll     LDA #$ll
Store it in the IOB
   8D F0 B7 STA $B7F0
High byte of data buffer
   A9 hh     LDA #$hh
Store it in the IOB
   8D F1 B7 STA $B7F1
```

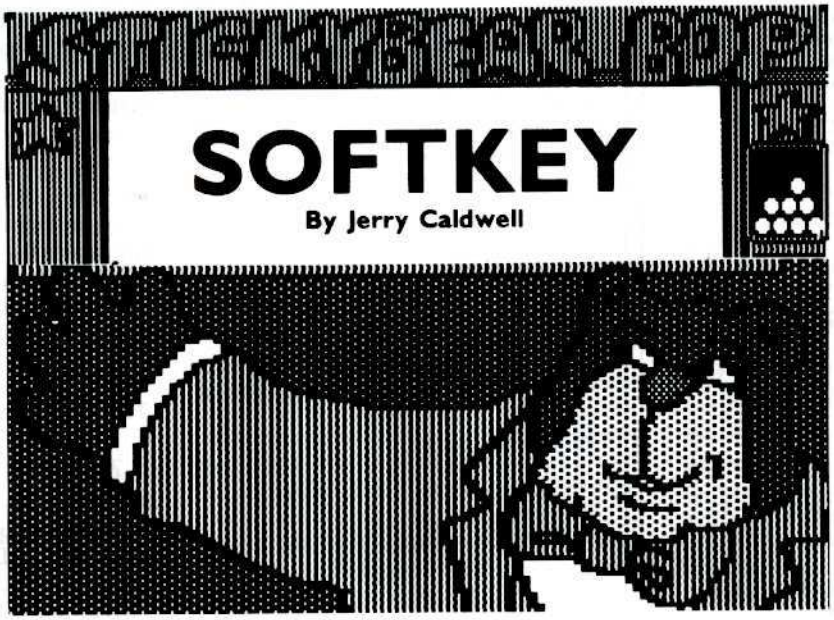
This set of instructions will indicate to you the location in memory that the protected sector will be read into. For example, if the code you find reads:

```
A9 00       LDA #$00
8D F0 B7    STA $B7F0
A9 03       LDA #$03
8D F1 B7    STA $B7F1
```

then the protected sector will be read into memory starting at address $300. However, different Stickybear programs will store the data at different locations. Make a note of whatever address you decide that the protected sector is being loaded into.

Next, disassemble a little bit further and look for the next three JSR's. (Note: On some programs in the Stickybear series, the three JSR's will be on a sector adjacent to the one where the code which sets up the IQB is found). You should find a JSR $B7B5 which is sandwiched between two other JSR's to the same address. The call to $B7B5 is the main entry point to RWTS and the other two JSR's call the routine which alters the postnibblization routine. The first call to the routine will alter the postnibble routine before reading the protected sector and the second call to the routine restores it back to its normal form.

Just past the second call to the table alteration routine you should see the instructions:



```
60          RTS
A2 00       LDX #$00
```

The RTS marks the end of the routine which reads the protected sector and the LDX #$00 is the first instruction of the code responsible for altering the postnibble routine. Later, we will negate this subroutine by storing a $60 in the place of the $A2 so that no alteration of the postnibble routine will occur when it is called.

Recovering the Data

The data on the protected sector can be recovered in one of two ways. The first method requires an old-monitor F8 ROM or NMI card so that the running Stickybear program can be interrupted. If you do not have any means of halting the program, then you will have to use the second, brute force method. I'll describe the easy method first.

The Easy Way

The "RESET" method of recovering the data is very straightforward. First, the original Stickybear program is booted up and allowed to read the protected sector. The program is then halted and the portion of memory that the data was read into is moved to a "safe" location in memory before booting up a DOS 3.3 slave disk. The translated data from the protected sector can then be saved onto disk for later use. For instance, if you discover that the protected sector is being read into $3Ø0, boot up the original Stickybear disk, halt the program after it has started running and from the monitor type:

9Ø0Ø<3Ø0.3FFM
C6ØØG (After inserting a DOS 3.3 slave disk)
BSAVE PROTECTED SECTOR, A$9ØØØ,L$1ØØ

The data from this sector will later be written to a backup copy of the Stickybear disk.

The Hard Way

Lacking an Integer or NMI card, we must alter one of the backups we have made to allow us to boot the backup, wait for us to insert the original, read the encoded sector and then exit to the monitor.

Earlier in the article the "JSR sandwich" responsible for altering the postnibblization routine and reading the protected sector was described. What we will do is alter the first and third JSR's. The first JSR will be altered so that it calls a routine which we will write. This routine will wait for a keypress (while we insert the original disk) before calling the postnibble alteration routine. The second JSR will be changed to a JMP $FF65 so that the Apple's monitor will be entered and the data from the protected sector can be recovered.

Due to of the number of different Stickybear programs, the location on the disk that these alterations have to be made will vary. As an example, I will show you how to make the changes to Stickybear Bop. For other programs in the series, the changes will be pretty much the same, except that they will have to be made to different sectors on the disk.

On Stickybear Bop, the code which reads the protected sector starts on track $11, sector $Ø2 and ends on track $11, sector $Ø1. A disassembly of this routine is printed below.

```
4DD8-  A9 Ø2       LDA  #$Ø2
4DDA-  8D EC B7    STA  $B7EC
4DDD-  A9 ØF       LDA  #$ØF
4DDF-  8D ED B7    STA  $B7ED
4DE2-  A9 ØØ       LDA  #$ØØ
4DE4-  8D FØ B7    STA  $B7FØ
4DE7-  A9 BØ       LDA  #$BØ
4DE9-  8D F1 B7    STA  $B7F1
4DEC-  A9 ØØ       LDA  #$ØØ
4DEE-  8D EB B7    STA  $B7EB
4DF1-  A9 Ø1       LDA  #$Ø1
4DF3-  8D EA B7    STA  $B7EA
4DF6-  A9 B7       LDA  #$B7
4DF8-  AØ E8       LDY  #$E8
4DFA-  A2 ØØ       LDX  #$ØØ
4DFC-  48          PHA
4DFD-  8A          TXA
4DFE-  48          PHA
4DFF-  98          TYA
4EØØ-  48          PHA
4EØ1-  20 12 4E    JSR  $4E12
4EØ4-  68          PLA
4EØ5-  A8          TAY
4EØ6-  68          PLA
4EØ7-  AA          TAX
4EØ8-  68          PLA
4EØ9-  20 B5 B7    JSR  $B7B5
4EØC-  Ø8          PHP
4EØD-  20 12 4E    JSR  $4E12
4E1Ø-  28          PLP
4E11-  6Ø          RTS
4E12-  A2 ØØ       LDX  #$ØØ
4E14-  AØ ØØ       LDY  #$ØØ
4E16-  BD 3B 4E    LDA  $4E3B,X
4E19-  E8          INX
4E1A-  85 68       STA  $68
4E1C-  BD 3B 4E    LDA  $4E3B,X
4E1F-  E8          INX
4E2Ø-  85 69       STA  $69
4E22-  C5 68       CMP  $68
4E24-  DØ Ø5       BNE  $4E2B
4E26-  C9 ØØ       CMP  #$ØØ
4E28-  DØ Ø1       BNE  $4E2B
4E2A-  6Ø          RTS
4E2B-  BD 3B 4E    LDA  $4E3B,X
4E2E-  48          PHA
4E2F-  B1 68       LDA  ($68),Y
4E31-  9D 3B 4E    STA  $4E3B,X
4E34-  E8          INX
4E35-  68          PLA
4E36-  91 68       STA  ($68),Y
4E38-  4C 16 4E    JMP  $4E16
4E3B-  C2          ???
4E3C-  B8          CLV
4E3D-  4C C3 B8    JMP  $B8C3
4E4Ø-  9F          ???
4E41-  C4 B8       CPY  $B8
4E43-  4E DC B8    LSR  $B8DC
4E46-  4C DD B8    JMP  $B8DD
4E49-  4F          ???
4E4A-  DE B8 4E    DEC  $4EB8,X
4E4D-  ØØ          BRK
```

As a result of examining this code you should be able to see that sector $F on track $Ø2 is the protected sector and that it is read into $BØØØ-$BØFF. The JSR $4E12 instructions are the calls to the postnibblization alteration routine. The first of these calls will be changed to a JSR $BCDF. At $BCDF, which is the beginning of 33 free bytes within the

RWTS, we will put the routine which waits for the keypress before reading the protected sector. This routine will look like this:

```
2Ø ØC FD  JSR $FDØC  Wait for a keypress
2Ø 12 4E  JSR $4E12  Alter "Postnibble"
6Ø        RTS        Return
```

The second JSR $4E12 will be changed to JSR $FF65 for entry into the Apple's monitor. So, on Stickybear Bop make the following sector edits to one of your backup copies.

Track	Sector	Byte	From	To
ØØ	Ø6	DF	88	2Ø
ØØ	Ø6	EØ	A5	ØC
ØØ	Ø6	E1	E8	FD
ØØ	Ø6	E2	91	2Ø
ØØ	Ø6	E3	AØ	12
ØØ	Ø6	E4	94	4E
ØØ	Ø6	E5	88	6Ø
11	Ø1	Ø2	12	DF
11	Ø1	Ø3	4E	BC
11	Ø1	ØE	12	65
11	Ø1	ØF	4E	FF

After making the changes, boot this disk. The patch at $BCDF will be executed momentarily. The drive will continue to spin, but all movement of the head will stop because the code is waiting for a key to be pressed. At this time, remove the modified backup, insert your write-protected original disk and press any key. When you hear a beep, the protected sector will have been read into memory. Depending upon the current setting of the screen soft-switches, you may or may not be able to see what you type at the keyboard. But fear not. Your Apple will accept input just the same. So, place a DOS 3.3 slave disk in the drive and recover the data from the protected sector by *carefully* typing

9Ø0Ø<BØØØ.BØFFM
C6ØØG
BSAVE PROTECTED SECTOR, A$9ØØØ,L$1ØØ

Remember, Stickybear Bop was used as an example here and you will have to modify the procedure depending upon which disk in the series you are using.

Once you have saved the data from the protected sector on disk, all you have to do is write this data back to the appropriate sector on the other COPYAd disk of the Stickybear disk and change the first byte of the routine that alters the postnibble routine from a $A2 (LDX) to a $6Ø (RTS). A generalized step-by-step procedure for

making backups of the Stickybear series follows below.

Stickybears in General

1) Make two copies of the Stickybear disk using the modified COPYA program which was described earlier.
2) Use a disk search utility, such as ZAP or Inspector to search the disk for the code which sets up the IOB for reading the

protected sector. Search for a hex pattern of

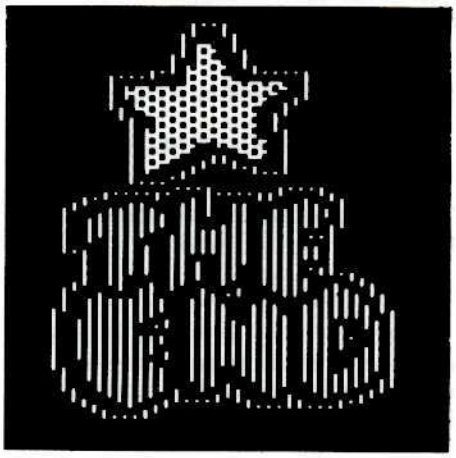
8D EC B7

Note: On Stickybear Bop, it will be found on track $11, sector $2.
3) Examine the code on the sector where this pattern was found and identify the portion of memory that the protected sector is read into. Note: On Stickybear Bop, it is read into $B000-$B0FF from track $02, sector $F.
4) Recover the data from the protected sector by using either the "Easy Way" or the "Hard Way" and save it to disk.
5) Write the data from the protected sector to the appropriate sector on the backup with a sector editor. Note: On Stickybear Bop, it should be written to track $02, sector $F.
6) Change the first byte of the routine which alters the "postnibble" routine from a $A2 to a $60 and write it back to the disk. Note: On Stickybear Bop, this is byte $12 on track $11, sector $1.

At this point, you should (hopefully) have a completely COPYAble version of your Stickybear disk that boots and runs exactly likes the original. Congratulations, if you were successful. If your backup does not boot, then carefully go back over the procedure and make sure that you did not forget to do anything along the way.

Although I have not had an opportunity to test this procedure on anything but Stickybear disks, I would not be surprised if it also works on other programs put out by Xerox.

(Hardcore COMPUTIST would like to hear from those readers who successfully use the procedure described in this article.)



MREAD/MWRT Update

by Gary Peterson

Looking through some back issues of Apple Assembly Lines recently, I noticed an article in the June 1983 issue entitled, "Replacing INIT Can Be Dangerous". The article, authored by Bill Morgan, piqued my interest because Hardcore COMPUTIST No. 13 contains my article, "Adding New Commands to DOS 3.3", which describes how to implement two new DOS commands, one of which replaces the INIT command code.

In his article, Bill describes how after adding a new DOS called SHOW (which replaced the INIT code), he began having problems with the S-C Word Processor. It seems that with the SHOW command installed, every time he tried to save a new file, a "FILE NOT FOUND" error would be generated. He eventually traced the problem to the S-C Word Processor's direct use of the DOS File Manager for OPENing files.

The File Manager can be called directly by setting up its parameter list and then using the page 3 vector at $3D6. If a new file is allowed to be created, the X register must be set to 0 on entry, otherwise the file being accessed must already exist. The $3D6 vector is a JMP to $AAFD where the X register is checked. If X = 0, then it is stored at $AA5F and if X is anything else, a $02 is stored at $AA5F. The value stored at $AA5F is what can cause a problem if the INIT code has been replaced.

The File Manager uses the value at $AA5F as an index into one of the normal DOS commands. A $00 indexes to the INIT command which is allowed to create new files (the HELLO file) and a $02 indexes to the LOAD command which cannot create new files. By replacing INIT with a command which cannot create new files, direct calls to the File Manager won't allow you to create new files either. This is the case with my MREAD command.

Bill Morgan suggests three methods to avoid the problem. The first (not replacing the INIT command) and the second (replacing INIT with a command that can create new files) are both rather restrictive. The third method (applying a patch to store a value of $04 at $AA5F when a new file needs to be created) seems much more reasonable.

Storing an $04 at $AA5F solves the problem by indexing into the SAVE command, which can create new files. The patch involves putting a JMP at $AAFD to a small piece of code which will check the X register and then return with it set to either $02 or $04. This code can be placed at the end of the MREAD/MWRT code at $AEED. Bill provides the following patch to fix the problem. Append it onto the MREAD/MWRT code and BSAVE the new file.

```
                      .OR $AEED
AEED:E0 02    PATCH CPX #$00     Create a
                                 new file?
AEEF:F0 04          BEQ .1       Yes.
AEF1:A2 02          LDX #$02     No new
                                 file. Use
                                 LOAD
                                 index.
AEF3:D0 02          BNE .2
AEF5:A2 04    .1    LDX #$04     Use SAVE
                                 index.
AEF7:4C 03 AB .2    JMP $AB03    Back to
                                 File
                                 Manager.
```

BSAVE MREAD/MWRT, A$AE8E,L$6C

The code at $AAFD also has to be changed so that it does a JMP $AEED whenever the File Manager is called through $3D6. This change can be POKEd in by the INSTALL MREAD/MWRT program. Just add this line to the Applesoft program before SAVEing it.

```
175 POKE 43773 ,76 : POKE 43774 ,237
    : POKE 43775 ,174 : REM
    APPLY FILE MANAGER PATCH
```

Thanks to Bill Morgan, these changes should solve any problems you may have been having with MREAD and MWRT installed. I only wish I had read his article sooner.

KEY	HC - Hardcore COMPUTIST* HCC - Hardcore Computing CORE- (previously published as an independent quarterly magazine) UPD - Update (additions to the early Hardcore Computing series) BHC - Best Of Hardcore Computing * Listings for Hardcore COMPUTIST No. 7 refer to the issue which appeared as Volume 3 No. 3.

Cumulative INDEX To Hardcore Publications: 1981-1984

Every new magazine is like a child. It grows and changes, developing new interests and dimensions. It seeks to gain improved skills for communication, attain wisdom, and acquire those good elements it sees in others while avoiding the mistakes often suffered in youth. The publications of SoftKey Publishing are no exception.

Over the past four years, the informative (and sometimes a bit radical) publications of SoftKey Publishing have appeared under the names Hardcore Computing, CORE, and finally Hardcore COMPUTIST. Determined to provide a quality source for the deprotection of Apple software, Hardcore Computing encountered immediate criticism from software manufacturers and ad censorship by fellow computer publications. With stubborn resolve, Hardcore Computing bravely weathered the storm and its offspring, Hardcore COMPUTIST, now thrives in an atmosphere of appreciation from users of Apple software.

In the pages of Hardcore COMPUTIST, vital information on making backups (as an alternative to costly replacement of software which has been damaged or destroyed) has become available to those who need it most: Apple computer users.

To provide our readers with a key reference to the many articles, softkeys, commentaries and columns found in the pages of Hardcore publications, we have presented here a Cumulative Index which spans the years 1981-1984. *(We would like to thank those freelance indexers who contributed index lists at nearly the same time as we were beginning to compile our own.)* This index contains entries for articles which have appeared in Hardcore Computing, the Hardcore Computing Updates, CORE, Hardcore COMPUTIST and The Best of Hardcore Computing (a recently compiled edition which offers the "BEST" of the Hardcore Computing series).

The entries for those articles and programs which have required corrections to the original contain a reference to the issue in which the correction was printed. Please note that no back issues of Hardcore Computing are available, but that certain articles from the series have been reprinted in The Best of Hardcore Computing. Issues of Hardcore COMPUTIST No.s 2,3 and 5 are also currently out of print and unavailable.

See below the short chronological history which details the name and format changes experienced by Hardcore COMPUTIST before it reached its present form.

The premiere issue of Hardcore Computing appeared early in 1981 and was followed later that year by Update 1.1, Hardcore Computing No.2 and Update 2.1. The year 1982 saw the publication of Hardcore Computing No.3 and Updates 3.1 and 3.2.

In April of 1983, Hardcore Computing split into two magazines, Hardcore COMPUTIST and CORE, with the intention of publishing eight issues of Hardcore COMPUTIST and four issues of CORE annually. Four issues of Hardcore COMPUTIST and three issues of CORE were printed in 1983.

At the onset of 1984, it became clear that the time and effort required to publish two Apple-related magazines with only one production/editorial staff was beyond our capabilities. The decision was made to recombine Hardcore COMPUTIST and CORE into one monthly magazine under the banner of Hardcore COMPUTIST. Hardcore COMPUTIST No. 6 became the first issue with the combined Hardcore/CORE format. As a result, a total of 11 issues plus The Best of Hardcore Computing have rolled out the door in 1984.

The current issue of Hardcore COMPUTIST (No. 15) will be the final issue for this year. The month of December will be used to catch up on all those projects that have been neglected while we were busy putting the monthly magazines together. But, never fear! Hardcore COMPUTIST No. 16 will arrive in your mailbox in January of 1985.

The members of our small but diligent staff would like to take this opportunity to thank our readers and contributors for their continued support during this past year and to wish you all happy and safe holidays. See you again in January.

Chuck Haight, Publisher
Gary Peterson
Ray Darrah
Valerie Robinson
Michelle Frank
Lynn Campos-Johnson

Advanced Playing Techniques

Adventure Game

Beyond Castle Wolfenstein *by Ray Darrah*
HC No. 13, pg. 12
Castle Wolfenstein *by Robb Canfield*
UPD 3.1, pg. 9: See BHC, p. 52
Castle Wolfenstein *by Sean Williams*
HC No. 6, pg. 10
Castle Wolfenstein *by Eric Holman Whitaker*
HC No. 8, pg. 23: See HC No. 10, pg. 21
Let's Map Ultima *by Bobby*
UPD 3.2, pg. 13
Ultima][*by Wes Felty*
HC No. 8, pg. 26
Ultima][...The Rest Of The Picture *by Wes Felty*
HC No. 11, pg. 21
Wizard And The Princess *by Donald Oliveau*
HC No. 7, pg. 29
Wizardry: The Proving Grounds Of The Mad Overlord *by Staff,* HC No. 7, pg. 29

Arcade Game

Cannonball Blitz *by Staff*
HC No. 1, pg. 24: See HC No. 10, pg. 10
Choplifter *by Staff*
HC No. 1, pg. 20
Choplifter *by Sean Williams*
HC No. 3, pg. 23
Loderunner *by Staff*
HC No. 10, pg. 10
Miner 2049'er *by Johnny Yukon*
HC No. 6, pg. 10
Miner 2049'er *by Dan Rosenburg & Paul Anderson*
HC No. 8, pg. 13
Serpentine *by Paul Andersen*
HC No. 8, pg. 26
Star Maze *by Ferrell Wheeler*
HC No. 7, pg. 29

Backup Procedures

DB Master Version 4.0 *by Dr. Nibbles*
HC No. 10, pg. 10
Using Locksmith to Copy Wizardry *by John Samborski,* HC No. 3, pg. 7: See HC No. 5, pg. 7

Editorial/Opinion/Commentary

Censorship In Computer Magazines *by Bev R. Haight,* HCC No. 1, pg. 4
Computer Code Must Be Kept Hidden And Secret *by Richard Cornelius,* HCC No. 3, pg. 10
Copying Vs. Copy Protection *by Bill Parkhurst*
HCC No. 3, pg. 9
Grand Piracy: Imitation Or Adaptation *by Bev R. Haight,* HCC No. 3, pg. 8
Let's All Become Disney Pirates *by Staff*
HCC No. 2, pg. 8
Piracy On The High Keys *by Art Cohl*
HCC No. 2, pg. 9
Rebuttals *by Val Golding*
HCC No. 2, pg. 12
Software Insurance Policies *by Tod A. Wicks*
HCC No. 3, pg. 11
Software Piracy, The Other Side *by Joe Zuis*
HCC No. 2, pg. 10
Where Do First Amendment Rights Stop? *by Allen L. Wyatt,* HCC No. 3, pg. 10

Fiction

L300 Grindle Series 12 Faces Life *by Elizabeth Nieuwland,* HC No. 5, pg. 31

Information

General

Apple Advocation Alliance *by Staff*
HCC No. 3, pg. 16
Data Bases *by Gary Peterson*
HC No. 6, pg. 19
More On Using Both Sides Of Your Diskette *by Karen Fitzpatrick,* HCC No. 2, pg. 16
Using Both Sides Of Your Diskettes *by Staff*
HCC No. 1, pg. 20: See BHC, p. 51
Whiz Kid- Protecting BASIC Programs *by Ray Darrah,* HC No. 6, pg. 29
Whiz Kid- The RESET Vector *by Ray Darrah*
HC No. 7, pg. 31
Whiz Kid- Diskette Sectoring *by Ray Darrah*
HC No. 8, pg. 31

Whiz Kid- Encoding Disk Data Part 1 *by Ray Darrah,* HC No. 10, pg. 29
Whiz Kid- Encoding Disk Data Part 2 *by Ray Darrah,* HC No. 13, pg. 25

Programming

Animation *by Staff*
CORE No. 1, pg. 56
Block Graphics *by Staff*
CORE No. 1, pg. 38
Color *by Staff*
CORE No. 1, pg. 28
Compiling Games: Ways To Speed Them Up *by Gary Peterson,* CORE No. 3, pg. 10
Hi-Res Screens *by Jack Hewitt*
HCC No. 3, pg. 43
High Resolution *by Staff*
CORE No. 1, pg. 19: See CORE No. 1, pg. 47
Introduction To Customizable Modular Adventure-Arcade Game *by B. Leo Bryte,* UPD 1.1, pg. 8
Low Resolution Graphics *by Staff*
CORE No. 1, pg. 16
Normal Keyboard Entry *by Bev R. Haight*
HCC No. 3, pg. 28
ProDOS To DOS: Single Drive Conversion *by Jimmy Eubanks, Jr.,* HC No. 9, pg. 15
Text Graphics *by Staff*
CORE No. 1, pg. 12
Using Applesoft Hi-Res Routines From Machine Language *by Staff,* CORE No. 1, pg. 20
Utilities As Game Development Tools *by Staff*
CORE No. 3, pg. 50
Vector Graphics *by Bev R. Haight*
HCC No. 3, pg. 45
Vector Graphics *by Staff*
CORE No. 1, pg. 30
What Exactly Is A Program? *by Staff*
HCC No. 2, pg. 8

Software

Back It Up][Plus Technical Notes *by Staff*
UPD 3.2, pg. 9: See BHC, pg. 35
Compleat Guide To Locksmith Parms *by Staff*
HCC No. 3, pg. 57: See BHC, pg. 24
EAMON: Creating The Adventure *by Staff*
CORE No. 3, pg. 41
How To Make Parm Changes With Nibbles Away *by Staff,* HCC No. 3, pg. 64
How To Survive Night Falls *by Bev R. Haight*
UPD 3.2, pg. 8
How To Use String Plotter *by David C. Smith*
HC No. 2, pg. 24
Locksmith 5.0 And The LPL *by Thomas H. Dragon,* HC No. 13, pg. 23
S-C Assembler Directives *by Staff*
HC No. 5, pg. 25
Using Visicalc For Job Costing *by Jerry Scott, PhD*
HCC No. 3, pg. 51

Software Marketing/Legal

Introducing Softquest USA (User's Software Alliance) *by B. Bryte,* UPD 3.2, pg. 4
Legal Forum: Consumers, Computers & The Law *by Barry D. Bayer,* HCC No. 2, pg. 13
Legal Forum: Good Guys, Bad Guys & Split Personalities *by Barry D. Bayer*
HCC No. 3, pg. 14
Software Writer's Market *by Bev R. Haight*
HCC No. 3, pg. 48
Writer's Markets *by Staff*
HCC No. 1, pg. 30

Modifications

Hardware

Construct Your Own Joystick *by Gary Peterson*
CORE No. 3, pg. 8
Curing Those Auto-Start ROM Blues *by Charles R. Haight,* HCC No. 2, pg. 45: See BHC, pg. 46
Getting On The Right Track *by Robert Linden*
HC No. 5, pg. 11
Hidden Locations Revealed *by Enrique Gamez*
HC No. 3, pg. 10
Modified ROMs *by Ernie Young*
HC No. 6, pg. 14: See HC No. 8, pg. 26: See HC No. 9, pg. 5
Pseudo ROMs On The Franklin Ace *by Ken Stutzman,* HC No. 12, pg. 24: See HC No. 14, pg. 9

Software

Akalabeth Softfix And Additions *by Bobby*
HCC No. 1, pg. 18
Fixing a ProDOS 1.0.1 BSAVE Bug *by Cecil Fretwell*
HC No. 13, pg. 20
Program Enhancements: Quick Bug *by Enrique Gamez* HC No. 6, pg. 8
Putting Locksmith 5.0 Fast Copy Into A Normal Binary File *by C. V. Fields,* HC No. 14, pg. 15
Upper And Lower Case Output For Zork *by Brian Burns,* HC No. 7, pg. 27
Using ProDOS On A Franklin Ace *by Staff*
HC No. 9, pg. 18

News/Interviews

Caveat Emptor *by Staff*
HC No. 11, pg. 30
Chapter 7 Blues *by Staff*
HC No. 13, pg. 29
Defendisk Cracking Reward *by Staff*
HC No. 10, pg. 27
Defendisk Defeated? *by Staff*
HC No. 11, pg. 30
Interview With Dave Alpert *by Staff*
HCC No. 1, pg. 8
Mike Markkula Interview *by Staff*
HCC No. 3, pg. 12
Omega Micro Files For Bankruptcy *by Staff*
HC No. 10, pg. 27
Recently Developed Chips Promise New Life For Old Apples: The 65SC802 & 65SC816 *by Gary Peterson,* HC No. 10, pg. 18

Parameter Listings

Back It Up][Plus
UPD 3.2, pg. 9: See BHC, pg. 35
Copy][Plus
HC No. 1, pg. 6: See HC No. 3, pg. 14: See HC No. 4, pg. 25: See BHC, pg. 38
Essential Data Duplicator
HC No. 11, pg. 10
Locksmith 4.0/4.1
HCC No. 3, pg. 60: See HC No. 2, pg. 9: See BHC, pg. 24
Nibbles Away
HCC No. 3, pg. 64: See BHC, pg. 31

Programs

Business

COREfiler *by Staff*
HC No. 7, pg. 22
COREfiler Form *by Ray Darrah*
HC No. 8, pg. 17

Game

Amber's T's *by Bev R. Haight*
HCC No. 2, pg. 52
Customizable Modular Adventure Game *by B. Bryte,* Part 2: See HCC No. 2, pg. 20; Part 3: See HCC No. 3, pg. 19
Destructive Forces *by Ray Darrah*
CORE No. 3, pg. 31
Dragon Dungeon *by Ray Darrah*
CORE No. 3, pg. 52
Map Maker *by Robb Canfield*
HC No. 3, pg. 26: See HC No. 4, pg. 32
Mapping Of Ultima III *by Jeff Hurlburt*
HC No. 11, pg. 19: See HC No. 13, pg. 8: See HC No. 14, pg. 9
Site & Its Elements *by B. Bryte*
HCC No. 3, pg. 19: See CORE No. 1, pg. 64
Space Raid *by Rich Orde*
CORE No. 1, pg. 58: See CORE No. 2, pg. 47
Text Invaders 2.0 *by Bev. R Haight*
UPD 1.1, pg. 12
Text Invaders 2.0 *by Bev R. Haight*
HCC No. 2, pg. 32: See UPD 3.1, pg. 16: See BHC, pg. 55
Ultima][Super Character *by Gary Peterson, Dave Thompson, Mark Cal, Rocky Giovinazzo & Richard Kahn,* HC No. 4 , pg. 6
Ultimap *by Gary Peterson & Dave Thompson*
HC No. 4, pg. 16
Ultimaker III *by Jeff Rivett & Ray Darrah*
HC No. 11, pg. 15: See HC No. 14, pg. 9
Zephyr Wars 2.0 *by Bev R. Haight*
HCC No. 3, pg. 38: See UPD 3.1, pg. 16: See BHC, pg. 58

Graphics

Arcade Quality Graphics *by Robb Canfield*
CORE No. 1, pg. 39: See CORE No. 2, pg. 47
Artist's Easel *by Jack Hewitt*
HCC No. 2, pg. 47
High Resolution *by Michael Patrick Scanlin*
CORE No. 2, pg. 12
Ink Blots *by B. Bryte*
HCC No. 1, pg. 22
Page Flipper *by Robb Canfield*
HC No. 2, pg. 19
Psychedelic Symphony *by Ray Darrah*
HC No. 12, pg. 16
Screen Cruncher *by Robb Canfield*
CORE No. 1, pg. 22: See CORE No. 2, pg. 47
Shape Table Mini-Editor *by Enrique A. Gamez*
CORE No. 1, pg. 35
Shimmering Shapes *by Neil Taylor*
CORE No. 1, pg. 31: See CORE No. 2, pg. 47
Three-D Wall Draw *by Barry Vaughan*
HC No. 2, pg. 34
UFO Factory *by Bev R. Haight*
CORE No. 1, pg. 26

Softkey

Controller Saver *by Ray Darrah*
HC No. 10, pg. 11
Controlling The IOB *by Bobby*
HCC No. 3, pg. 36
CSaver: The Advanced Controller Saver *by Ray Darrah,* HC No. 13, pg. 16
Demuffins *by Staff*
HCC No. 1, pg. 28: See HCC No. 2, pg. 5
How To Create Demuffin Plus *by Staff*
HC No. 8, pg. 14: See HC No. 9, p. 18
Super IOB *by Ray Darrah*
HC No. 9, pg. 11: See HC No. 11, pg. 7: See BHC, pg. 15
Super IOB v1.2: Update *by Ray Darrah*
HC No. 14, pg. 10

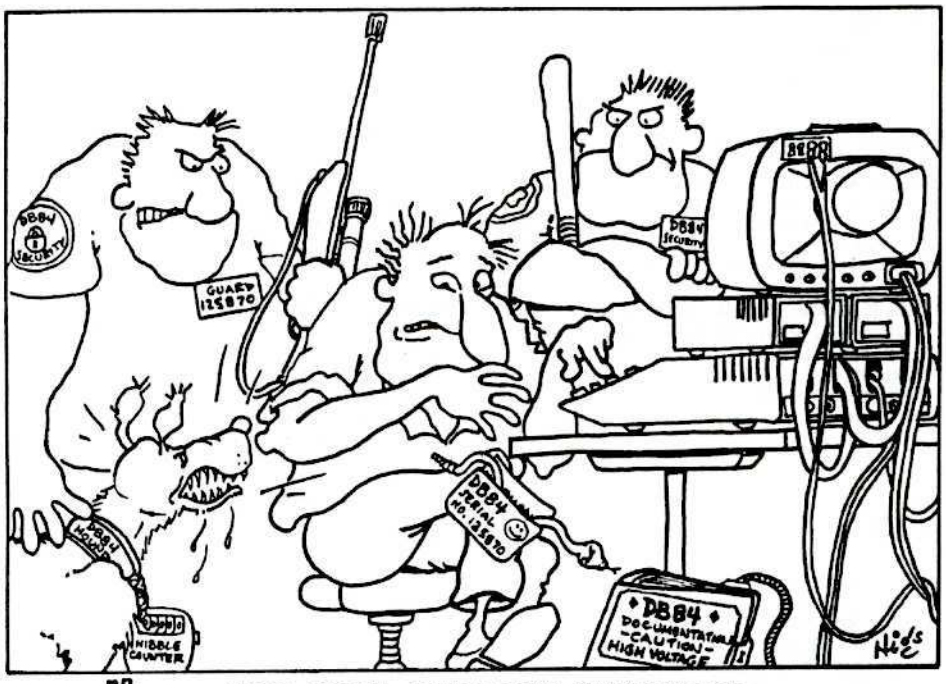
Sound Reproduction

ApplEar *by Ray Darrah*
HC No. 10, pg. 16

Utility

Adding New Commands To DOS 3.3 *by Gary Peterson,* HC No. 13, pg. 18
Armonitor *by Nick Galbreath*
HC No. 12, pg. 23
Batman Decoder Ring *by Ray Darrah*
HC No. 14, pg. 16
Checkbin *by Robb Canfield*
HC No. 1, pg. 21: See HC No. 3, pg. 4
Checksoft *by Robb Canfield*
HC No. 1, pg. 11: See HC No. 3, pg. 4
CORE Disk Searcher *by Bryce L. Fowler & Ray Darrah* HC No. 12, pg. 19: See HC No. 14, pg. 9
CORE Word Search Generator *by Barry Palinsky*
HC No. 9, pg. 19: See HC No. 11, pg. 4
Crunchlist *by Ray Darrah*
HC No. 6, pg. 26: See HC No. 10, pg. 26

Disk Directory Designer *by Tim Lewis*
HC No. 7, pg. 17
DiskEdit 2.1 *by Charles Haight*
HCC No. 2, pg. 60: See UPD 2.1, pg. 9: See BHC, pg. 1
Diskview 1.0 *by Charles R. Haight*
HCC No. 2, pg. 42: See UPD 2.1, pg. 9: See UPD 3.1, pg. 16: See BHC, pg. 12
Dynamic Menu *by Brent Millirans*
CORE No. 2, pg. 4: See CORE No. 3, pg. 64
&GOTO Label *by Robb Canfield*
CORE No. 2, pg. 24
HyperDOS *by John Bridges*
HCC No. 3, pg. 26
HyperDOS Without INIT *by Robb Canfield*
UPD 3.2, pg. 14
Line Find *by Robb Canfield*
CORE No. 2, pg. 34: See CORE No. 3, pg. 64
Menu *by Chuck Haight*
UPD 2.1, pg. 5
Menu Hello Program *by Robb Canfield*
HCC No. 3, pg. 32: See UPD 3.1, pg. 16: See BHC, pg. 47
Personalizing A Program *by Enrique Gamez*
HC No. 6, pg. 11: See HC No. 8, pg. 26
ProDOS Data Encryptor *by Gary Peterson*
HC No. 8, pg. 20
Quick Copy *by Robb Canfield*
CORE No. 2, pg. 40: See CORE No. 3, pg. 64
String Plotter *by Rich Hofmann*
HC No. 2, pg. 22
Using DiskEdit To Reformat Catalog *by Charles R. Haight,* HCC No. 1, pg. 25



HARVEY NIBBLOFF'S RECENTLY PURCHASED DATA BASE PACKAGE CAME WITH THE VERY LATEST IN COPY-PROTECTION SAFEGUARDS.

Reviews

Adventure Game

Critical Mass *by Staff*
CORE No. 3, pg. 22
Crystal Caverns *by Staff*
CORE No. 3, pg. 22
Death In The Caribbean *by Staff*
CORE No. 3, pg. 23
Doom Valley *by Staff*
CORE No. 3, pg. 23
Exodus: Ultima III *by Staff*
CORE No. 3, pg. 15
Gruds In Space *by Staff*
CORE No. 3, pg. 15
Jenny Of The Prairie *by Staff*
CORE No. 3, pg. 24
Quest For The Holy Grail *by Staff*
CORE No. 3, pg. 28
Wizardry *by Staff*
CORE No. 3, pg. 56

Arcade Game

Asteroid Belt *by Staff*
CORE No. 3, pg. 17
Boulder Dash *by Ray Darrah*
HC No. 14, pg. 20
Buzzard Bait *by Staff*
CORE No. 3, pg. 20
Caverns Of Callisto *by Staff*
CORE No. 3, pg. 20
Crime Wave *by Staff*
CORE No. 3, pg. 21
Dino Eggs *by Ray Darrah*
HC No. 10, pg. 21
Evolution *by Staff*
CORE No. 3, pg. 23
Grapple *by Staff*
CORE No. 3, pg. 24
Highrise *by Staff*
CORE No. 3, pg. 24
Kamikaze *by Staff*
CORE No. 3, pg. 26
Lady Tut *by Staff*
CORE No. 3, pg. 15
Miner 2049'er *by Staff*
CORE No. 3, pg. 27
Minit Man *by Staff*
CORE No. 3, pg. 27
Mission Escape *by Staff*
CORE No. 3, pg. 27
Pentapus *by Staff*
CORE No. 3, pg. 28
Repton *by Staff*
CORE No. 3, pg. 30
Sammy Lightfoot *by Staff*
CORE No. 3, pg. 30
Shuttle Intercept *by Staff*
CORE No. 3, pg. 30
Spider Raid *by Staff*
CORE No. 3, pg. 55
Wayout *by Staff*
CORE No. 3, pg. 56
Zargs *by Staff*
CORE No. 3, pg. 57
Zaxxon *by Staff*
CORE No. 3, pg. 57

Book

Enhancing Your Apple][, Volume 1, Second Edition *by Martin Collamore* HC No. 13, pg. 21
Graphic Software For Microcomputers *by Staff*
HCC No. 2, pg. 51

Business Software

Data Base Management *by Monty Lee*
HC No. 6, pg. 22

Educational Software

Alphabet Beasts & Co. *by Staff*
CORE No. 3, pg. 17
Micro Mother Goose *by Staff*
CORE No. 3, pg. 26
Type Attack *by Staff*
CORE No. 3, pg. 55

Hardware

Copy Cards *by Edward Burlaw (Reprint from Peelings II Vol. 4, No. 1),* HC No. 1, pg. 28
Replay Card *by Julie Joringdal*
UPD 3.1, pg. 13

Simulation Program

Alibi *by Staff*
CORE No. 3, pg. 17
Bermuda Race *by Staff*
CORE No. 3, pg. 17

Utility Program

Another Bit Copier: Nibbles Away *by Karen Fitzpatrick,* HCC No. 2, pg. 14
Best Of The Bit Copiers *by Dr. Phillip Romine*
HC No. 8, pg. 27
Bit Copy Programs *by Karen Fitzpatrick*
HCC No. 1, pg. 10
CIA *by Gary Peterson*
HC No. 6, pg. 31
Copy II Plus 4.4C *by Dr. Phillip Romine*
HC No. 11, pg. 8
Essential Data Duplicator v1 *by Dave Thompson*
HC No. 6, pg. 30
New Bit Copy Programs *by Staff*
HCC No. 3, pg. 54
S-C Assembler *by Jeff Thomas*
HC No. 7, pg. 14
Visible Computer Vs. Apple][- 6502 ALT *by Martin Collamore,* HC No. 9, pg. 28

Softkeys

Adventure game

Akalabeth: World Of Doom *by Bobby*
HCC No. 1, pg. 16
Caverns Of Freitag *by C.J. Singer*
HC No. 6, pg. 6
Dark Crystal *by Clay Harrell*
HC No. 5, pg. 8
Exodus- Ultima III *by Tim Schaap*
HC No. 11, pg. 27
Legacy Of Lyllgamyn *by Roger Carlson*
HC No. 4, pg. 29
Legacy Of Lyllgamyn *by Jim Kaiser*
HC No. 8, pg. 10: See HC No. 9, pg. 18

Lion's Share *by Jan Eugenides*
HC No. 12, pg. 14
Mask Of The Sun *by John J. Liska*
HC No. 7, pg. 27
Mask Of The Sun- A Correction *by Gary Wolfe*
HC No. 11, pg. 7
Prisoner][*by David Kirsch*
HC No. 4, pg. 15
Softporn Adventure *by Wes Felty*
HC No. 11, pg. 6
Starcross *by Jeff Rivett*
HC No. 5, pg. 20
Transylvania & The Quest *by Thomas A Phelps*
HC No. 13, pg. 14
Ultima][*by Brian Burns & Dan Rosenberg*
HC No. 4, pg. 12: See HC No. 4, pg. 28: See HC No. 11, pg. 21
Witness *by Toto*
HC No. 4, pg. 15
Zork *by Bobby*
HC No. 1, pg. 5

Arcade game

Apple Galaxian *by Bobby*
HCC No. 3, pg. 37
Atarisoft Revisited *by Trevor Churchill*
HC No. 13, pg. 5
Beyond Castle Wolfenstein *by Ray Darrah*
HC No. 13, pg. 12
Bill Budge's Trilogy *by Michael Decker*
HC No. 5, pg. 13
Buzzard Bait *Clay Harrell*
HC No. 5, pg. 30
Cannonball Blitz *by Staff*
HC No. 1, pg. 24
Canyon Climber *by John Liska*
HC No. 10, pg. 8
Castle Wolfenstein *by Robb Canfield*
UPD 3.1, pg. 9: See UPD 3.2, pg. 15
Cosmic Combat *by Clay Harrell*
HC No. 9, pg. 8
Crush, Crumble & Chomp *by Jeff Rivett*
HC No. 7, pg. 5
Donkey Kong *by Dan Lui*
HC No. 6, pg. 6
Electronic Arts Releases *by Pete Levinthal*
HC No. 13, pg. 26
Flip Out *by Clay Harrell*
HC No. 12, pg. 11
Gold Rush *by Clay Harrell*
HC No. 9, pg. 7
Hard Hat Mack *by Rich Lyon*
HC No. 5, pg. 22
Hi-Res Computer Golf][*by Jeff Rivett*
HC No. 12, pg. 6
Hyperspace Wars *by Robb Canfield*
UPD 3.2, pg. 3: See BHC, pg. 20
Laf Pak *by Ferrell Wheeler*
HC No. 13, pg. 9
Lancaster *by Clay Harrell*
HC No. 5, pg. 9
Minit Man *by Clay Harrell*
HC No. 10, pg. 28
Mouskattack *by Clay Harrell*
HC No. 7, pg. 6
Pandora's Box *by Clay Harrell*
HC No. 6, pg. 5
Pest Patrol *by Ray Darrah*
HC No. 4, pg. 20
Robotron *by Clay Harrell*
HC No. 8, pg. 8
Sabotage *by Clay Harrell*
HC No. 12, pg. 7
Sammy Lightfoot *by Eric Kinney*
HC No. 5, pg. 29
Sea Dragon *by Jeff Rivett*
HC No. 14, pg. 8
Snake Byte *by Clay Harrell*
HC No. 7, pg. 5
Sneakers *by David E. Rentzel*
HC No. 3, pg. 6: See HC No. 4, pg. 32
Spy Strikes Back *by Clay Harrell*
HC No. 8, pg. 6
Suicide *by Clay Harrell*
HC No. 12, pg. 6
Zaxxon *by Clay Harrell*
HC No. 7, pg. 8: See HC No. 10, pg. 5

Business software

Data Factory Version 5.0 *by L.S. Davis*
HC No. 8, pg. 14
Data Reporter *by Don Halley*
HC No. 1, pg. 4
DB Master *by Dan Lui*
HC No. 7, pg. 6
Home Accountant *by Barry May*
HC No. 5, pg. 26
Multiplan *by Bobby*
HC No. 1, pg. 4: See HC No. 2, pg. 8: See HC No. 3, pg. 12
PFS Software *by Gary J. Wolfe*
HC No. 14, pg. 6
Time Is Money *by Rod Wideman*
HC No. 12, pg. 7
Visifile *by Bob Bragner*
HC No. 5, pg. 9
Visifile *by C. Masters*
HC No. 6, pg. 7
Visidex *by Anthony L. Barnett*
HC No. 9, pg. 7
Visiplot/Visitrend *by Anthony L. Barnett*
HC No. 3, pg. 6
Visiterm *by B. Baker*
HC No. 9, pg. 8

Educational Software

Apple LOGO *by Anne Rachel Gygi*
HC No. 8, pg. 7: See HC No. 10, pg. 5
Computer Preparation: SAT *by Eddie Fang*
HC No. 14, pg. 6
DLM Software *by Chris Chenault & Ray Darrah*
HC No. 13, pg. 7
Knoware *by Doni G. Grande*
HC No. 14, pg. 6
Krell LOGO *by Andrew Harrison*
HC No. 10, pg. 8
Learning With Leeper *by Marco Hunter*
HC No. 13, pg. 8
Millionaire *by Bill Wilson*
HC No. 12, pg. 7: See HC No. 13, pg. 8
Rocky's Boots *by Jerry Caldwell*
HC No. 14, pg. 22
Snooper Troops *by Jim Mitchell*
HC No. 13, pg. 7
Tellstar *by William Wingfield Jr.*
HC No. 13, pg. 8
Type Attack *by Jerry Caldwell*
HC No. 12, pg. 8: See HC No. 13, pg. 8

General

Applesoft Disks *by Bobby*
HCC No. 1, pg. 26
Boot Code Tracing *by Bobby*
HCC No. 3, pg. 37
Disk Locks *by Staff*
HCC No. 1, pg. 31: See UPD 1.1, pg. 4
Examining Protected Applesoft Programs *by Clay Harrell,* HC No. 10, pg. 24
Expanded Disklocks *by Bobby*
HCC No. 2, pg. 24
Hayden Software *by Floyd Splidnik*
HC No. 8, pg. 6
How To Backup Copy Protected Disks][*by Bobby*
HCC No. 2, pg. 29
How To Make Backups: A Guide *by Bobby*
HCC No. 3, pg. 68
How To Use IOB DATA Statements *by Bobby*
UPD 3.1, pg. 4: See UPD 3.2, pg. 15
Integer Programs *by Bobby*
HCC No. 2, pg. 30
IOB For Three TSR Games *by Richard B. Fabbre*
HC No. 4, pg. 29
Making Liberated Backups That Retain Their Copy Protection *by Thomas Dragon,* HC No. 7, pg. 12
Modifying IOB For Single Disk Drive Systems *by Robb Canfield,* UPD 3.1, pg. 5
Questions & Answers *by Staff*
HCC No. 2, pg. 19
Sierra On-Line Software *by Doni G. Grande & Clay Harrell,* HC No. 9, pg. 24
Special IOBs *by Bobby*
UPD 2.1, pg. 3
Tricks And ? *by Staff*
HCC No. 1, pg. 35

Utility Program

Ampermagic *by Bob Bragner*
HC No. 5, pg. 28
Arcade Machine *by Marco Hunter*
HC No. 10, pg. 9
Artist *by Walt Campbell*
HC No. 8, pg. 12
Bag Of Tricks *by Neil Taylor*
HC No. 3, pg. 8: See HC No. 4, pg. 32
Bag Of Tricks- A Correction *by Earl Taylor*
HC No. 5, pg. 14
Boot Code Tracing Revisited *by Mycroft*
UPD 3.1, pg. 6: See UPD 3.2, pg. 15: See BHC, pg. 21
Egbert][Communications Disk *by Keith S. Goldstein*
HC No. 5, pg. 16: See HC No. 7, pg. 30
Einstein Compiler Version 5.3 *by Marco Hunter*
HC No. 11, pg. 6
Essential Data Duplicator Version 1 *by Steven Zupp*
HC No. 8, pg. 24
Essential Data Duplicator III *by Joseph Leathlean*
HC No. 10, pg. 7
Locksmith 5.0 *by Steve Burke (letter)*
HC No. 8, pg. 3: See HC No. 13, pg. 5
MatheMagic *by Doni G. Grande*
HC No. 14, pg. 7
Music Construction Set *by Jim Waterman*
HC No. 9, pg. 7
Music Construction Set *by Dan Rosenberg*
HC No. 12, pg. 27
Visible Computer: 6502 *by Jared Block & Bob Bragner,* HC No. 9, pg. 26
Zoom Grafix *by Michael Decker*
HC No. 12, pg. 9

Word Processing

Bank Street Writer *by Earl Taylor & Steve Morgan*
HC No. 10, pg. 12: See HC No. 11, pg. No. 7
Magic Window][*by Bobby*
HC No. 2, pg. 6
Screenwriter][*by Daniel Price*
HC No. 5, pg. 8: See HC No. 7, pg. 30: See HC No. 10, pg. 4
Sensible Speller *by Cris Rys*
HC No. 9, pg. 9
Sensible Speller IV *by Lamont Cranston*
HC No. 10, pg. 6: See HC No. 13, pg. 6
Sensible Speller IV: UPDATE *by Doni G. Grande*
HC No. 11, pg. 24

Tables

Apple Only Magazines *by Staff*
HCC No. 3, pg. 17
Byte Conversion Chart *by Staff*
UPD 2.1, pg. 1
CORE Utility Chart *by Staff*
CORE No. 2, center insert
Memory Map *by Staff*
CORE No. 1, pg. 10
PEEKing (-16384) & GETting A$ *by Staff*
UPD 2.1, pg. 7

A Boot From Drive 2

By Michael Phillips

Requirements:
Apple][+ or equivalent
16K RAM card
Two disk drives
One blank disk

Occasionally, when working with a non-standard DOS on my computer, I have found it necessary to keep switching between DOS 3.3 and the other DOS. This normally requires swapping the disk in drive #1 before booting up with the DOS that is desired. Such swapping can soon become a real pain in the keyboard. Wouldn't it be nice to place the nonstandard DOS disk in drive 1, a DOS 3.3 disk in drive 2 and be able to boot up from either drive? This sort of thing cannot normally be done but, with the program presented in this article, I will demonstrate how it is indeed possible to boot DOS 3.3 from a disk in drive 2.

The ability to boot from drive 2 requires the assembly language program presented in this article in addition to a one byte change to the disk that is to be booted from drive 2. To minimize the possibility of memory conflicts, I wrote the the program to reside in bank 1 of a Slot Ø RAM card, but it could be easily located to run at another memory location. In addition to the code in the language card, the ampersand vector ($3F5-$3F7) and twelve other bytes somewhere in main memory are required. I have placed these twelve bytes at $96ØØ-$96ØB, but again they could be located elsewhere.

Before you can boot from drive 2 you will have to type in the hexdump listed on this page. For convenience sake, the hexdump can be entered at $2ØØØ instead of directly into the language card. When you have typed in the code and checked it, save it to disk by typing

BSAVE DRIVE 2 BOOT,A$2ØØØ,L$9B

The source code for the program is also listed on this page. Basically, all this assembly language program does is check for the proper command syntax, move an image of the disk controller ROM into hi-res page one and then modify that image before using it to boot with.

You will also need a program which installs this program into the language card for you. This Applesoft program is shown in Listing 1 on this page. First type

FP

and then enter the program. This will be the HELLO program on the disk that will boot from drive 2. After you have entered the program, place a blank disk in drive 1 and type

POKE 46855,2
INIT INSTALL D2 BOOT

The poke to 46855 modifies the RWTS at $B7Ø7 to keep it from immediately switching back to drive 1 as the disk in drive 2 boots. The program also moves an image of the F8 monitor ROM into the language card.

Whenever you anticipate the need to boot drive 2, just type

RUN INSTALL D2 BOOT

Keep in mind that the disk we initialized to boot from drive 2 will not boot from drive 1 and will not boot from drive 2 until the you run INSTALL D2 BOOT.

Once INSTALL D2 BOOT has been run, the ampersand (&) vector will be set up to point to $96ØØ where the twelve bytes of code I mentioned earlier are placed. These twelve bytes of code are responsible for switching the RAM card on and off. The proper syntax for performing a boot from drive 2 is

&PR#n,D2

where n is the slot the disk controller is in. If desired, you can also boot from drive 1 by typing

&PR#n,D1

but of course, you don't need any special code to boot from drive 1.

If you discover that the non-standard DOS is wiping out the ampersand vector and/or the code at $96ØØ, you can restore these locations (assuming the code in the language card is still intact) by typing

CALL-151
CØ88
DØØØG
CØ8A
3DØG

Source Code

```
*************************************************************
*                                                           *
*                      DRIVE TWO BOOT                       *
*                                                           *
*                            BY                             *
*                      MICHAEL PHILLIPS                     *
*                                                           *
*                                                           *
*   CODE WHICH WILL TRICK DOS INTO BOOTING FROM EITHER      *
*      DRIVE 1 OR DRIVE 2. AS CURRENTLY CONFIGURED          *
*     LOADS INTO BANK1 OF A LANGUAGE CARD IN SLOT Ø.        *
*                                                           *
*     ENABLED BY &PR#x,Dn (x = SLOT #, n = DRIVE #).        *
*                                                           *
*************************************************************
*
*
               .OR $DØØØ
               .TA $9ØØ
               .TF DRIVE TWO BOOT
*
*        EQUATES
*
Ø2ØØ- KBBUF    .EQ $2ØØ      KEYBOARD BUFFER
Ø3F5- AMPVEC   .EQ $3F5      & VECTOR
96ØØ- SWITCH   .EQ $96ØØ     CODE TO SWITCH ON RAM CARD
96Ø6- SNERR    .EQ SWITCH+6  PRINT SYNTAX ERROR
FE2C- MOVE     .EQ $FE2C     MONITOR MEMORY MOVE
ØØ3C- MEMBEG   .EQ $3C       START OF MEMORY TO MOVE
ØØ3E- MEMEND   .EQ $3E       END OF MEMORY TO MOVE
ØØ42- MEMDST   .EQ $42       DESTINATION OF MEMORY MOVE
ØØFC- ROMADD   .EQ $FC       POINTER TO DISK CONTROLLER
ØØFE- RAMADD   .EQ $FE       POINTER TO THE CODE WHICH BOOTS
*
*
*        SET UP THE & VECTOR
*
*
DØØØ: A9 Ø1              LDA #$Ø1      FIX DOS TO BOOT
DØØ2: 8D Ø7 B7           STA $B7Ø7     FROM DRIVE 1
DØØ5: A2 ØØ              LDX #$ØØ
DØØ7: BD 8C DØ  SETUP    LDA CODE,X    STORE THE CODE WHICH
DØØA: 9D ØØ 96           STA SWITCH,X  TURNS ON THE
DØØD: E8                 INX           RAM CARD ON AND OFF.
DØØE: EØ Ø6              CPX #$Ø6
DØ1Ø: DØ F5              BNE SETUP
DØ12: A9 4C              LDA #$4C      SET UP THE AMPERSAND
DØ14: 8D F5 Ø3           STA AMPVEC    VECTOR TO JMP
DØ17: A9 ØØ              LDA #SWITCH   TO $96ØØ
DØ19: 8D F6 Ø3           STA AMPVEC+1  WHERE THE BANK
DØ1C: A9 96              LDA /SWITCH   SWITCHING
```

Continued on next page

Continued from previous page

```
D01E: 8D F7 03           STA AMPVEC+2 OCCURS.
D021: 60                 RTS
*
*
*       CHECK COMMAND SYNTAX
*
*
D022: A2 04     CKSYN    LDX #$04     CHECK THAT THE
D024: BD 00 02  NXTCHR   LDA KBBUF,X  PROPER SYNTAX
D027: E0 02              CPX #$02     HAS BEEN USED.
D029: F0 16              BEQ CKSLT    AN ERROR IS
D02B: DD 00 02           CMP KBBUF,X  PRINTED IF THE
D02E: D0 0E              BNE ERR      SYNTAX WAS IMPROPER.
D030: CA                 DEX
D031: D0 F1              BNE NXTCHR
D033: AD 05 02           LDA KBBUF+5  CHECK THAT THE
D036: C9 31              CMP #$31     DRIVE NUMBER
D038: F0 1A              BEQ RELROM   IS IN THE
D03A: C9 32              CMP #$32     PROPER RANGE (1-2).
D03C: F0 16              BEQ RELROM
D03E: 4C 06 96  ERR      JMP SNERR    PRINT SYNTAX ERROR
D041: C9 31     CKSLT    CMP #$31     CHECK THAT SLOT #
D043: 90 F9              BCC ERR      IS IN THE PROPER
D045: C9 38              CMP #$38     RANGE (1-7).
D047: B0 F5              BCS ERR
D049: 69 90              ADC #$90     SET ADDRESS OF
D04B: 85 FD              STA ROMADD+1 THE CONTROLLER'S ROM
D04D: 49 E0              EOR #$E0     SET THE DESTINATION
D04F: 85 FF              STA RAMADD+1 OF THE CODE FROM ROM
D051: CA                 DEX
D052: 90 D0              BCC NXTCHR
*
*
*       RELOCATE ROM CODE INTO RAM
*
*
D054: A0 FF     RELROM   LDY #$FF     SET UP THE
D056: 84 3E              STY MEMEND   NECESSARY ZERO
D058: C8                 INY          PAGE LOCATIONS
D059: 84 3C              STY MEMBEG   TO PERFORM
D05B: 84 FC              STY ROMADD   THE MEMORY MOVE
D05D: 84 FE              STY RAMADD   2x03<Cx00.CxFFM.
D05F: A9 03              LDA #$03
D061: 85 42              STA MEMDST
D063: A5 FD              LDA ROMADD+1
D065: 85 3D              STA MEMBEG+1
D067: 85 3F              STA MEMEND+1
D069: A5 FF              LDA RAMADD+1
D06B: 85 43              STA MEMDST+1
D06D: 20 2C FE           JSR MOVE     PERFORM THE MOVE.
D070: A0 02              LDY #$02     INSERT A PATCH
D072: B9 98 D0  PATCH    LDA CODE1,Y  TO TURN OFF
D075: 91 FE              STA (RAMADD),Y THE LANGUAGE
D077: 88                 DEY          CARD BEFORE BOOTING.
D078: 10 F8              BPL PATCH
D07A: A0 39              LDY #$39     FIX THE CODE TO
D07C: AD 05 02           LDA KBBUF+5  BOOT FROM DESIRED
D07F: 18                 CLC          DRIVE.
D080: 69 59              ADC #$59
D082: 91 FE              STA (RAMADD),Y
D084: 6C FE 00           JMP (RAMADD) BOOT THE DISK.
*
*
*       STORAGE
* CMD CONTAINS COMMAND SYNTAX.
* CODE PLACED AT $9600.
* CODE1 PLACED AT $2x00.
*
*
D087: AF 8A 36  CMD      .HS AF8A362C44
D08A: 2C 44
D08C: 2C 88 C0  CODE     .HS 2C88C04C22D02C8AC04CC9DE
D08F: 4C 22 D0
D092: 2C 8A C0
D095: 4C C9 DE
D098: 2C 8A C0  CODE1    .HS 2C8AC0
```

If the computer freezes up when you type C088, that means that the code in the language card has been written over. In such a case, you will have to turn off your computer and try to find a safer location to place the code for DRIVE 2 BOOT.

Listing 1

```
10 REM ************************
20 REM *                      *
30 REM *   INSTALL D2 BOOT    *
40 REM *                      *
50 REM *  INSTALLS CODE FOR   *
60 REM * BOOTING FROM DRIVE 2 *
70 REM * INTO SLOT 0 RAM CARD *
80 REM *                      *
90 REM ************************
100 REM
110 GOSUB 170
120 POKE -16247,0 : POKE -16247,0
130 PRINT CHR$(4)"BLOAD^DRIVE^
    TWO^BOOT,A$D000"
140 POKE 60,0 : POKE 61,248 : POKE 62
    ,255 : POKE 63,255 : POKE 66,0 :
    POKE 67,248
150 CALL 768
160 HOME : A$ = "BOOT2.DOS^LOADED" :
    GOSUB 190 : END
170 FOR AD = 768 TO 782 : READ BY :
    POKE AD,BY : NEXT : RETURN
180 DATA ^^160,0,32,44,254,44
    ,136,192,32,0,208,44,138
    ,192,96^^^^
190 HTAB 20 - ( LEN (A$) ) / 2 : PRINT
    A$ : RETURN
```

10	- $BADD	110	- $A232
20	- $9B13	120	- $9534
30	- $4D3B	130	- $114B
40	- $AD92	140	- $B5BB
50	- $C899	150	- $24B4
60	- $FF65	160	- $A5F0
70	- $A3BF	170	- $D4B5
80	- $A900	180	- $AB2E
90	- $924D	190	- $516B
100	- $CB63		

Hexdump

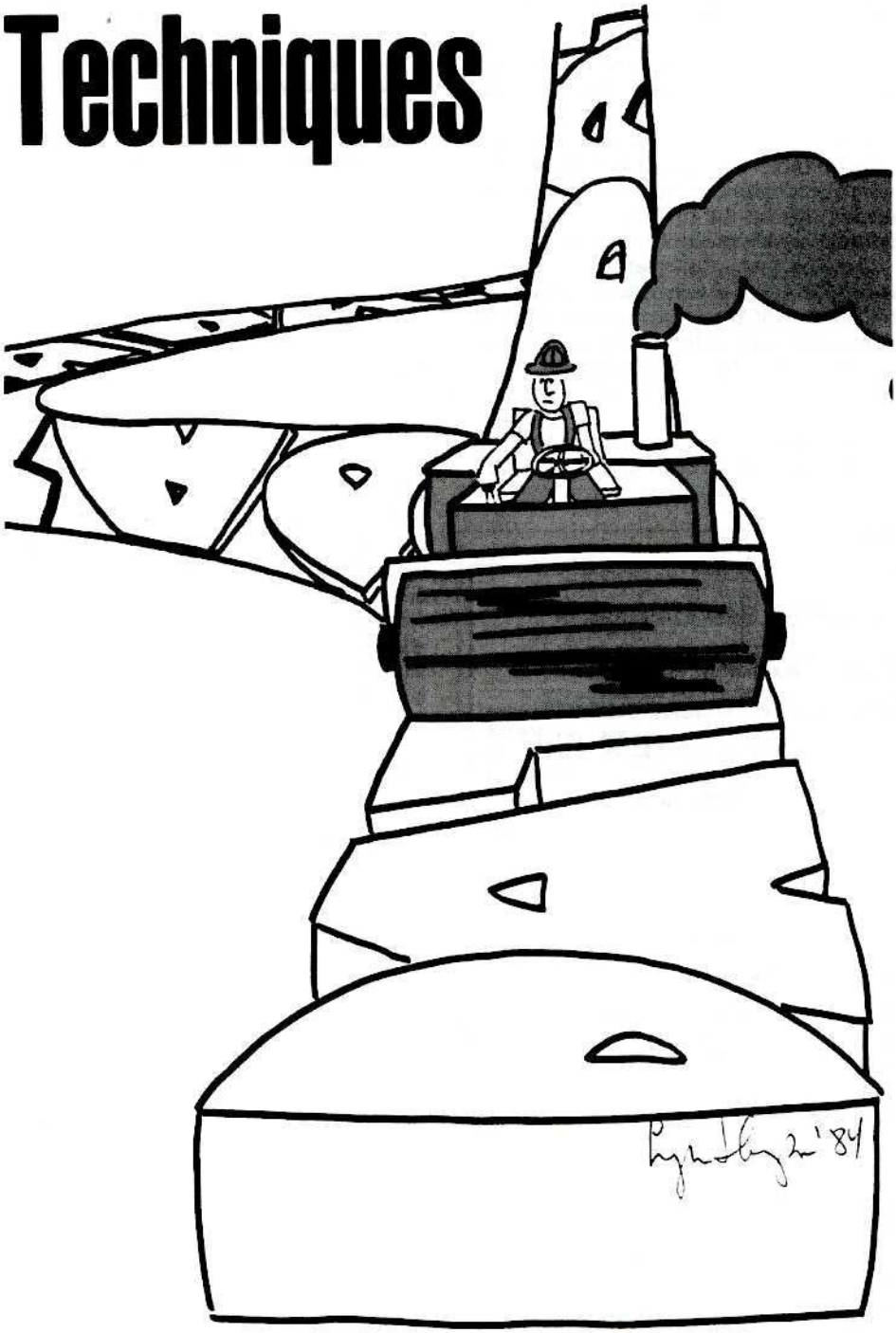
```
2000: A9 01 8D 07 B7 A2 00 BD    $88AB
2008: 8C D0 9D 00 96 E8 E0 0C    $E960
2010: D0 F5 A9 4C 8D F5 03 A9    $7789
2018: 00 8D F6 03 A9 96 8D F7    $95E2
2020: 03 60 A2 04 BD 00 02 E0    $3A0C
2028: 02 F0 16 DD 00 02 D0 0E    $F72E
2030: CA D0 F1 AD 05 02 C9 31    $7CE4
2038: F0 1A C9 32 F0 16 4C 06    $2821
2040: 96 C9 31 90 F9 C9 38 B0    $D549
2048: F5 69 90 85 FD 49 E0 85    $A90D

2050: FF CA 90 D0 A0 FF 84 3E    $37FB
2058: C8 84 3C 84 FC 84 FE A9    $C56F
2060: 03 85 42 A5 FD 85 3D 85    $CF5C
2068: 3F A5 FF 85 43 20 2C FE    $307A
2070: A0 02 B9 98 D0 91 FE 88    $400A
2078: 10 F8 A0 39 AD 05 02 18    $5B21
2080: 69 59 91 FE 6C FE 00 AF    $23FC
2088: 8A 36 2C 44 2C 88 C0 4C    $5633
2090: 22 D0 2C 8A C0 4C C9 DE    $535E
2098: 2C 8A C0                   $1F07
```

DB Master's Data Compression Techniques

By Alan & Valerie Floeter

Since its introduction to the market in 1980, DB Master by Stoneware has become one of the top selling data base programs. It is a carefully designed system with many options for the user. Not only has Stoneware produced a nice piece of software, but they have also developed an efficient method for storing records on the disk. When some area businesses required that special reports be developed that could not be easily done by the program, we decided to investigate the data format that DB Master uses. This was not a simple task since DB Master stores its data in a compressed form in a non-standard DOS format. We had no idea what data compression technique was used or where other information, such as the field descriptions, could be found. This article is an explanation of what we found on the disks. We used Version 3.02 for our investigation but, from what we've seen, this information is accurate for Version 4 as well.



DB Master Format

DB Master uses two separate disks for its data storage. The Utility disk holds information about the fields and reports. The File disk (or Data disk) holds the actual data entered by the user.

Data Disk

When deciding how to store records, data base designers have two basic choices. They can use a simple text file in which each character is stored on the disk but, when most of a record is filled with spaces, this is wasteful. Therefore, some designers develop a binary-type format that has special codes to compress the data. So, if a field in the file has forty spaces, it may be coded into the file in, say, two bytes instead of the forty bytes needed for a text file. This results in more room on the disk for records.

We began to decode DB Master's data compression by looking at the data disk. Using DB Master we printed a listing of all entries in our data base. We then used a sector editor to read the data disk. This provided us with the information we needed.

After searching through the track and sectors and trying to match the codes on the disk with our DB Master listing we discovered the following things: The data disk is organized into blocks, each eight sectors long. Block 0 starts at track 1, sector 7 and ends at track 1, sector 0 (going backwards). Block 1 goes from track 1, sector $F to track 1, sector 8. Block 2 covers from track 2, sector 7 to track 2, sector 0. This continues up to Block 67 at track $22, sector $F.

Continued on page 20

Block 32 (track $11, sector 7 to track $11, sector 0) contains the catalog and block usage map. Starting at sector 5 of track 7, the catalog is similar to a DOS 3.3 catalog. The data disk catalog contains the name of your file in flashing characters and the date is was created in inverse. Other characters also appear, as well as "VOLUME:1" in flashing characters. The "1" refers to the disk number. Your first data disk naturally has a 1. When you continue entering data on a second disk, this has a 2 in it. Although we haven't gone to a third disk yet, we assume it would have a 3.

As we just mentioned, the block usage map is also found in Block 32. It is 67 bytes long, one byte for each block, starting at the 84th byte in track $11, sector 5. For each block, either a 0 or 1 is stored, depending on whether the block is free or used, respectively.

Blocks 0 and 1 are not used for data storage on the first data disk. However, on subsequent disks they do contain data. We suspect that on the first disk they contain information about search keys or sorting.

The first entry of the data base is stored in Block 2. The first two bytes in a block give the number of bytes used in that block in (high order, low order) hex. We have seen typical values around $6,$D9 to $7,$1C, but this will vary with your data format and actual records.

Records never cross block boundaries so each block is the start of a new record. Each record starts with the length of the record including the length bytes. Although you might assume that the maximum record length would always be the same, DB Master uses a data compression method to make good use of disk storage. The high order byte of the length has a $F8 OR'd onto it so a record length of $F8,$0A means that only 10($0,$A) bytes are used for that record.

The data for the actual record then follows in either stripped ASCII (no high order bit), integer format or floating point format, depending on the field type. Long and short integers appear in two or one bytes, respectively. Floating point fields, the special dollar type fields and calculated fields are all in floating point format, taking up five bytes of memory. All the remaining field types such as alphanumeric, telephone, Yes/No, social security number and dates are alphanumeric, one character per byte in stripped ASCII.

Special codes are found within the record for packing purposes. These usually consist of two bytes, the first byte indicating what is to be done and the second byte determining how many times. These codes are shown in Figure 1 below.

At first we could not determine what the last byte of the record signified. We eventually concluded that it must be a checksum and that is what it turned out to be. This checksum value is merely the sum of all the bytes in the record. For example, if your record contained $20, $05, $0A, and $14, your checksum would be the sum of these four numbers ($43). A checksum is used to check for data integrity. If you add up all the codes in a record and your figure doesn't match the checksum, you know something has been changed since you wrote out the record. Someone may have tampered with the record, or your disk may have gone bad. We aren't quite sure how DB Master uses this code.

Figure 1

Codes for Data Compression

CODE	MEANING
$60,xx	Place xx nulls ($00) into the record at this place, so $60,$A means to put 10 $00's into the record, instead of these two codes.
$61,xx	Place xx spaces into the record. $61,$8 means substitute 8 spaces into the record.
$62,xx	Place xx ASCII zeroes ($30) into the record. $62,$4 would substitute the ASCII string "0000".
$63,yy,xx	Place xx codes into the record of the code yy. $63,$41,$5 mean to have five $41's instead.
$64	Do not interpret the next byte as a special character. Since these special codes are legal characters, when DB Master needs to use them, it must preceed them with this code so it doesn't interpret them. For example, $64,$61, $40 would place a $61,$40 into the record rather than the spaces usually denoted by the special code $61.

Figure 2

Magazine Data Base Format

#	Field Name	Prim Key	Sec Key	Read Prot	Field Type	Screen Length
1	Mag.	y	n	n	alpha	6
2	Year	y	n	n	0-255	2
3	Month	y	n	n	0-255	2
4	Page	y	n	n	integ	5
5	Loc	y	n	n	alpha	1
6	Article type	n	n	n	alpha	20
7	Title	n	n	n	alpha	30
8	Author	n	n	n	alpha	30
9	Com	n	n	n	alpha	30
10	1	n	n	n	alpha	30
11	2	n	n	n	alpha	30
12	3	n	n	n	alpha	30
13	4	n	n	n	alpha	30
14	5	n	n	n	alpha	30
15	6	n	n	n	alpha	30
16	7	n	n	n	alpha	30
17	8	n	n	n	alpha	30
18	9	n	n	n	alpha	30

Let's take a look at an example to see how an actual record would appear on the disk. One of the reasons that we purchased DB Master was to store a listing of magazine articles. We read over a dozen computer magazines each month and when we need to refer to an article, we often forget in which magazine it appeared. We set up our magazine file as shown in Figure 2.

The lengths given in Figure 2 are the lengths on the screen, not the number of bytes each field will consume on the disk. To determine the number of bytes they'll take up in the actual record we need to look at the field type as well as the length.

Remembering that alphanumeric fields take up one byte per character, we then know that all alpha fields are the same length in a filled record as on the screen. This is our most common field type for this data base configuration. We also use the short integer for values between 0 and 255. These take up one byte. The long integer takes up two bytes. This results in the Figure 3 chart for the field lengths. We call these the expanded field lengths because we are assuming, at this point, that no data compression has been done. Depending upon the actual data entered into a field, the actual length of the field stored on the disk will vary.

Figure 3

Expanded Length of the Magazine Data Base

#	Field Name	Field Type	Field Length
1	Magazine	alpha	6
2	Year	0-255	1
3	Month	0-255	1
4	Page	integ	2
5	Loc	alpha	1
6	Article type	alpha	20
7	Title	alpha	30
8	Author	alpha	30
9	Com	alpha	30
10	1	alpha	30
11	2	alpha	30
12	3	alpha	30
13	4	alpha	30
14	5	alpha	30
15	6	alpha	30
16	7	alpha	30
17	8	alpha	30
18	9	alpha	30

Two records from our data base are shown in Figure 4 (pg. 21). Let's look at how each of these was stored on the disk. The first record starts in Block 2. Using our sector editor, we printed out track 2, sector 7 (the start of this block). A part of this sector is shown in Figure 5 (pg. 21).

Let's look at the data there to see how DB Master's data compression works. The first two bytes in the block give the number of bytes used in that block. Our example says that $6,$DD (1757 decimal) bytes are used in this block. The length of the record follows this. Here we see $F8,$3D. Since $F8

is always OR'd onto the first number, we need to remove that before calculating the actual record length. Our length is; therefore, $0,$3D or simply 61 decimal. So, the next 61 bytes will tell us all about the first record in our data base.

Figure 4

Records Stored in Data Base

Example 1

```
MAGAZINE - APPLE
YEAR - 80
MONTH - 2
PAGE - 12
LOC -
ARTICLE TYPE - ARTICLE
TITLE - APPLESOFT INTERNALS
AUTHOR - JOHN CROSSLEY
COM -
1 -
2 -
3 -
4 -
5 -
6 -
7 -
8 -
9 -
```

Example 2

```
MAGAZINE - APPLE
YEAR - 80
MONTH - 2
PAGE - 25
LOC -
ARTICLE TYPE - ARTICLE
TITLE - CONNECTING WITH USCD BIOS
AUTHOR - RANDALL HYDE
COM -
1 -
2 -
3 -
4 -
5 -
6 -
7 -
8 -
9 -
```

Looking back to our field specifications in Figure 3, we see that our first field is the magazine name. It is alphanumeric and six bytes long. Back at Figure 5, the next six bytes are the ASCII for "APPLE" which look like $41, $50,$50,$4C,$45,$20. This is the magazine name for our first article.

The next two fields store the date of the magazine, year first, then month. These are each one byte, so the $50 says the year is 80 (1980 for us) and the month is 2 (the second issue that year).

The fourth field, stored as an integer in two bytes, is the page number. This article is on page $00, $0C which converts to 12. The fifth field is to record the location on the page. It is alphanumeric, one character long. Here it is " ", a blank.

The article type, alphanumeric and 20 characters in length, appears next. On the disk we see the ASCII for "ARTICLE" followed by a $61,$0D. From Figure 1 we see that $61 is the special code used to add spaces, in this case $D or 13 spaces.

Figure 5

Block 2 of Data Disk

```
06 DD F8 3D 41 50 50 4C
45 20 50 02 00 0C 20 41
52 54 49 43 4C 45 61 0D
41 50 50 4C 45 53 4F 46
54 20 49 4E 54 45 52 4E
41 4C 53 61 08 4A 4F 48
4E 20 43 52 4F 53 53 4C
45 59 61 FF 61 3E 2E F8
42 41 50 50 4C 45 20 50
02 00 19 20 41 52 54 49
43 4C 45 61 0D 43 4F 4E
4E 45 43 54 49 4E 47 20
57 49 54 48 20 55 53 43
44 20 42 49 4F 53 61 05
52 41 4E 44 41 4C 4C 20
48 59 44 45 61 FF 61 3F
1D F8 3F 41 50 50 4C 45
20 50 02 00 2E 20 41 52
54 49 43 4C 45 61 0D 41
```

Continuing in this manner, we next see the article title "APPLESOFT INTERNALS" followed by $61,$0B for 11 extra blanks. The author appears as "JOHN CROSSLEY" with $61, $FF. Since this provides 255 blanks, we can see that some will be used to finish the author's name and the rest will be applied to the comments section. Another $61, $3E will finish up the rest of the record with 62 spaces. The last byte of the record is $2E, the checksum of the record. That is the first record in our data base.

The second starts right away with a length of $F8,$42, or 68 once converted as we did before. The rest of the record follows the same general form of the first one. You can expand it if you'd like.

Utility Disk

The utility disk is generally used for storing anything that is not data. This includes the field specifications and report formats. A great deal of information is stored on this disk and we feel that we hardly scratched the surface to decode the utility disk. We did not investigate the report formats at all, but instead focused our attention on the field specifications.

The utility disk is set up the same as the data disk. Information is stored in blocks starting at track 1, sector 7. Actual data starts in Block 2. Each block starts with a two byte length. The record length is then given with $F8 OR'd onto the high order byte. After the length, each record has three bytes that seem to identify the record number. The first record has $00,$00,$00, although this appears in the compressed form as $60,$03. The second record has $00,$00,$01 and each record after that increments the count, so the twelfth record has $00,$00,$0B, etc. After these three bytes, the actual data for the record appears. The same codes are used for data compression here as on the data disk.

The utility disk catalog set up is very similar to the one used by the data disk. The main difference is that in the first entry in the catalog, the word "UTILITY" appears along with the filename in flashing characters.

Using the sector editor, we printed out several examples of utility files to see what we could find. Starting with Record 1 in Block 2 (track 2, sector 7) we expanded the information substituting any characters needed for the special codes. Once the information was expanded, we found that the record number appears in Bytes 1-3. Byte 5 contains the total number of fields in a record and the 12th byte contains the number of pages used for data input. If you use only one page to enter data into DB Master, then this has a 1 stored in it. If two pages, there's a 2, etc.

In the 18th byte, the last field number found on page 1 is stored, followed by the last field number on page 2, etc. So if field #16 is the last one entered on page 1, 16 is stored in byte 18. The name of the file and date it was created appear in Record 1 at byte 105.

The information about data fields appears in other records. See Figure 6 (pg. 22) for what appears in each record of the field specifications. Looking at an actual example will probably help to clarify this, too. We have printed out track 2, sectors 7 and part of 6 to show the utility disk for the magazine data base we used before. Figure 7 (pg. 22) shows this memory dump.

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

We would like to hear from you!

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

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

Figure 6

Records 2-11 on Utility Disk

Record #	Information
2	Order of fields, stored one byte per field. We're not sure why these aren't stored in the same order as they are entered.
3	Horizontal cursor position for each field prompt.
4	Vertical cursor position for each field prompt.
5	Field type for each field where 1 = 1 byte integer 2 = 2 byte integer 3 = floating point 4 = dollar 5 = alphanumeric 6 = Y/N 7 = phone number 8 = social security number 9 = date 10= auto date
6	Width for each field (length on the screen).
7	Unknown. For our fields always 00's.
8	Unknown. Could be used for order or protected fields.
9	1st Field information a) length of field name b) ASCII characters for field name c) lots of 00's
10	2nd Field information same as 1st.
11	3rd Field information
etc.	etc.

Since all the information is compressed, we need to first expand it so we can pick up the bytes we need. The start of a block contains the same format as the data disk. The first two bytes determine how much of Block 2 is being used. We see here it is $04,$8A, or 1162 bytes. The next two bytes state the length of the first record, again with $F8 OR'd on. So, the length of the first record is $00,$28 bytes, or 40. Looking at Figure 7, the next 37 (40-2 bytes for the length -1 byte for the checksum) bytes need to be expanded for Record 1 of the utility disk. Figure 8 shows this record expanded.

Using our expanded record in Figure 8 and looking in Byte 5 of Record 1, we see that there are 18 ($12) fields in our data base. Byte 12 says that 1 page of data entry is set up. Byte 18 says we have field number 18 ($12) as our last field on page 1. Bytes 77-94 hold the file name in ASCII, here shown as "MAG", followed by the date created, 12-23-82, in Bytes 95-100.

Record 2 contains the first part of our field specification information. This has been expanded in Figure 9. Bytes 4-21 are used for the order the fields are to be entered. All 18 fields are in numerical order so, in this case, they are stored in the same order as they are entered.

Record 3, expanded for you in Figure 10, shows the horizontal cursor positions for each of the prompt fields in Bytes 4-21. By continuing to expand each of the records you can see how the file is set up.

Figure 7

Part of Block 2 of Utility Disk

Track 2, Sector 7

```
04 8A F8 28 60 03 05 12
52 0B 02 01 00 0F 01 00
02 01 00 02 12 60 0B 83
01 87 60 44 4D 41 47 61
0F 31 32 32 33 38 32 60
10 02 F8 1A 00 00 01 01
02 03 04 05 06 07 08 09
0A 0B 0C 0D 0E 0F 10 11
12 60 77 83 F8 13 00 00
02 01 14 13 01 15 63 01
04 63 03 09 60 77 EE F8
1A 00 00 03 02 02 03 05
05 07 09 0A 0C 0D 0E 0F
10 11 12 13 14 15 60 77
AA F8 0F 00 00 04 05 01
01 02 63 05 0E 60 77 5A
F8 11 00 00 05 06 02 02
05 01 14 63 1E 0C 60 77
8D F8 08 00 00 06 60 89
EF F8 1A 00 00 07 01 01
02 01 02 03 04 05 06 07
08 09 0A 0B 0C 0D 0E 0F
60 77 5A F8 11 00 00 08
08 4D 41 47 41 5A 49 4E
45 60 80 3C F8 0D 00 00
09 04 59 45 41 52 60 84
22 F8 0E 00 00 0A 05 4D
4F 4E 54 48 60 83 78 F8
0D 00 00 0B 04 50 41 47
45 60 84 10 F8 0C 00 00
0C 03 4C 4F 43 60 85 D2
F8 15 00 00 0D 0C 41 52
```

Track 2, Sector 6

```
54 49 43 4C 45 20 54 59
50 45 60 7C 5B F8 0E 00
00 0E 05 54 49 54 4C 45
60 83 78 F8 0F 00 00 0F
06 41 55 54 48 4F 52 60
02 CA F8 0C 00 00 10 03
43 4F 4D 60 85 D7 F8 0A
00 00 11 01 31 60 87 2A
F8 0A 00 00 12 01 32 60
87 2C F8 0A 00 00 13 01
33 60 87 2E F8 0A 00 00
14 01 34 60 87 30 F8 0A
00 00 15 01 35 60 87 32
F8 0A 00 00 16 01 36 60
87 34 F8 0A 00 00 17 01
37 60 87 36 F8 0A 00 00
18 01 38 60 87 38 F8 0A
00 00 19 01 39 60 87 3A
F8 0C 00 00 FC 53 45 41
4E 60 85 08 F8 33 00 00
FF 88 5C 60 03 2B 60 2D
```

Figure 8

Record 1 of Utility Disk

Bytes	Contents
1-3	00,00,00
4-7	05,12,52,0B
8-11	02,01,00,0F
12-15	01,00,02,01
16-18	00,02,12
19-29	11-00's
30-32	83,01,87
33-76	44-00's
77-79	4D,41,47
80-94	15-" "'s
95-98	31,32,32,33
99-100	38,32
101-116	16-00's
117	02

Figure 9

Record 2 of Utility Disk

Bytes	Contents
1-4	00,00,01,01
5-8	02,03,04,05
9-12	06,07,08,09
13-16	0A,0B,0C,0D
17-20	0E,0F,10,11
21	12
22-140	119-00's
141	83

Figure 10

Record 3 of Utility Disk

Bytes	Contents
1-4	00,00,02,01
5-8	14,13,01,15
9-12	4-01's
13-21	9-03's
22-140	119-00's
141	EE

Conclusion

That is as far as we have progressed in our investigation of DB Master. Throughout this exploration, we repeat again that we have been using Version 3 of DB Master. Although we don't foresee any problems in applying the information presented in this article, it is possible that we have made some incorrect assumptions along the way; therefore, we can't guarantee that the data is true for every data base configuration out there or every version of DB Master. *Caveat lector* (Let the reader beware).

If you are experienced in assembly language, we encourage you to investigate DB Master further to fill in some of the holes we left. If you do make some discoveries, we'd love to have a copy of your documentation.

Tic Tac Show
Computer Advanced Ideas
1442 A Walnut Street, Suite 341
Berkeley, CA 94709
$39.95

Requirements:
Apple][Plus or equivalent
Bit copier or Super IOB
One blank disk

Boot Code Trace For Tic Tac Show

By Steve Fillipi

Tic Tac Show is a computerized rendition of the popular TV game show that bears a very similar name. The game can be played by 1 or 2 players and is hosted by the lovely young lady, Carol. The players try to place their X's or O's on a tic tac toe board by correctly answering questions from a particular subject category. The first player to get 3 adjacent squares on the board is declared the winner. The game is very nicely done and youngsters of all ages should enjoy playing it. Because the program allows you to create your own categories and questions, the game can be tailored for specific educational settings. Unfortunately, the potential of this program in an educational environment is somewhat limited by its copy protection.

With a little investigation, I found the protection scheme on Tic Tac Show to be as follows:

1) The address and data epilogues on track $Ø have been modified. The address epilogues are changed to B5 AB EB and the data epilogs are AB AA EB.
2) The Tic Tac Show DOS resides on tracks $1.5 through $4.5.
3) Tracks $6-$22 are normally formatted, but 2 directories are present, one on track $11 and one on track $Ø6. The files in the directory on track $11 contain the questions and answers, and the files in the directory on track $6 hold the code for the game itself. The DOS utilizes the two directories by changing the value at $ACØ1 from a $11 to a $Ø6, or vice-versa.

From this description of the protection scheme, it appears that it would not be too difficult to backup this disk with a bit copier by copying tracks Ø, 1.5-4.5 and 6-22. Surprisingly, I could not get Copy][+ v4.4c or Essential Data Duplicator III to make a bootable copy of Tic Tac Show. Instead of fooling around with parameter settings and disk drive speed adjustments, I decided to try and boot code trace the disk. For me, this proved to be a more successful and educational approach than using a bit copier. My report follows.

Boot Code Theory

Although the boot code tracing technique has been covered several times before in the pages of this magazine, I will quickly go over a little theory for the benefit of new readers.

The basis for boot code tracing lies in the fact that on any bootable disk, whether protected or not, track $Ø, sector $Ø must be readable by the disk controller hardware. The first thing that happens when a disk boots is that track $Ø, sector $Ø is read into memory at $8ØØ-$8FF. After this has occurred, the code in the disk controller's ROM jumps to $8Ø1 and the code there reads track $Ø into memory (on normal disks). Protected disks may vary. The boot will continue from here, reading whatever tracks are necessary, until it is complete.

To boot code trace a disk, it is necessary to halt the boot after track $Ø, sector $Ø has been read into memory, but before the code at $8Ø1 starts to excute. This can be done by moving the code from the disk controller ROM down into RAM where it can be modified. Instead of jumping to $8Ø1 to continue the boot, it will now enter the Apple's monitor. The code read in from the disk can then be examined and possibly modified so that another stage of the boot will take place before the monitor is again entered. By stopping at various stages of the boot, the entire program can eventually be read into memory and transferred to a normal disk. The difficulty of boot code tracing copy protected disks varies greatly, but whatever the difficulty, a good knowledge of 65Ø2 machine language is an absolute necessity for trying this technique on your own.

Tic Tac Boot

When performing a boot code trace, first disconnect DOS and fill up memory from $8ØØ-$BFFF with 11's so we can tell where code gets loaded into. Do this by typing

```
CALL-151
FE89G N FE93G
8ØØ:11 N 8Ø1<8ØØ.BFFFM
```

Next, move the code from the disk controller ROM down into RAM where it can be modified

```
96ØØ<C6ØØ.C6FFM
96F8:A9 ØØ 85 FC 85 FD 85 FE
97ØØ:A9 6Ø 85 FF AØ 43 B1 FC
971Ø:91 FE 88 1Ø F9 AD E8 CØ
9718:4C 69 FF
```

If you disassemble from $96F8, you should see the following code

```
96F8-  A9 ØØ     LDA   #$ØØ
96FA-  85 FC     STA   $FC
96FC-  85 FD     STA   $FD
96FE-  85 FE     STA   $FE
97ØØ-  A9 6Ø     LDA   #$6Ø
97Ø2-  85 FF     STA   $FF
97Ø4-  AØ 43     LDY   #$43
97Ø6-  B1 FC     LDA   ($FC),Y
97Ø8-  91 FE     STA   ($FE),Y
97ØA-  88        DEY
97ØB-  1Ø F9     BPL   $97Ø6
97ØD-  AD E8 CØ  LDA   $CØE8
9710-  4C 69 FF  JMP   $FF69
```

This code will save some necessary zero page locations to page $6Ø before turning off the drive motor and entering the monitor. Insert the original Tic Tac Show disk into the drive and boot it by typing

```
96ØØG
```

When the drive turns off, the BOOT1 code will have been read into page $Ø8. If you examine this code and compare it to the BOOT1 code from a DOS 3.3 slave disk you will find them to be identical except for the values at $84D-$85C (sector skewing table) and $8FE-$8FF. The Tic Tac Show boot reads the sectors from track $Ø in physically

ascending order rather than interleaving them as DOS normally does. This, in part, accounts for the slow boot of the disk. On a DOS 3.3 slave disk, the values at $8FE-$8FF are B6 09 which indicates that the data will be read into memory starting at $B600 and that 10 sectors (0-9) will be read in this stage of the boot. On the Tic Tac Show disk, these values are 3F 05. This means that 6 sectors of data will be read into memory starting at $3F00.

The BOOT1 code exits at $84A with an indirect jump to the address stored in $8FD-$8FE. On a normal slave disk this will be $B700 but, on Tic Tac Show, it is $4000.

We now want to execute the BOOT1 code and let it read the 6 sectors of data from track $0 into memory. We could just type 801G, but this would prove to be unsuccessful because the code at $801 expects the disk drive to be revolving and needs some data from zero page that was lost when the monitor was entered. The necessary code from zero page was stored on page $60, so we have to enter some code which will turn on the disk drive, wait for the drive to come up to speed (about 1 second) and restore locations $0-$43 from page $60 before jumping to $801. This code can be placed at $900 by typing

```
900:8D E9 C0 A0 09 A9 C0 20
908:A8 FC 88 D0 F8 A0 43 B1
910:FE 91 FC 88 10 F9 4C 01
918:08
```

This should disassemble as

```
0900-  8D E9 C0    STA   $C0E9
0903-  A0 09       LDY   #$09
0905-  A9 C0       LDA   #$C0
0907-  20 A8 FC    JSR   $FCA8
090A-  88          DEY
090B-  D0 F8       BNE   $0905
090D-  A0 43       LDY   #$43
090F-  B1 FE       LDA   ($FE),Y
0911-  91 FC       STA   ($FC),Y
0913-  88          DEY
0914-  10 F9       BPL   $090F
0916-  4C 01 08    JMP   $0801
```

This code uses the monitor WAIT routine at $FCA8 to create the delay that pauses while the disk drive reaches its proper speed. Before you can execute this code we will also have to modify the instruction at $84A so that it jumps to the code which turns off the disk instead of continuing on with the boot. Make the modification by typing

```
84A:4C 0D 97
```

and then execute the next stage of the boot with a

```
900G
```

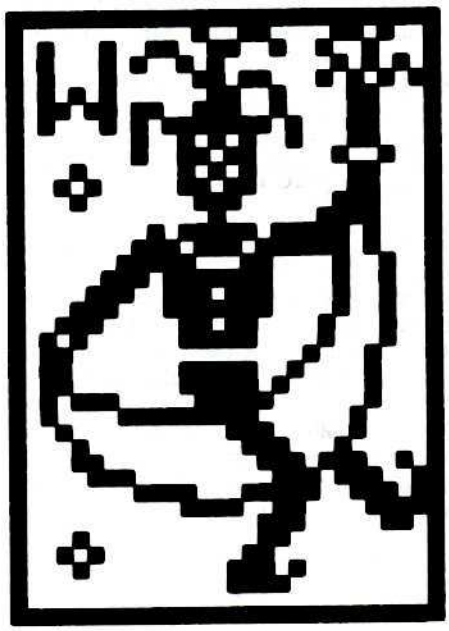
When the drive shuts off, the code responsible for accessing the half-tracks will have been placed at $4000-$44FF (track $0, sector $0 is re-read into $3F00-$3FFF). The start of this code should disassemble as

```
4000-  86 2B       STX   $2B
4002-  A2 FF       LDX   #$FF
4004-  9A          TXS
4005-  2C 80 C0    BIT   $C080
4008-  2C 81 C0    BIT   $C081
400B-  A9 00       LDA   #$00
400D-  85 06       STA   $06
400F-  85 07       STA   $07
4011-  85 08       STA   $08
4013-  8D 78 04    STA   $0478
```

Notice that the first thing this code does is to store the X register into $2B, which is where the slot number of the drive controller is held (actually 16 * slot number). We have to make sure that, when we execute the next stage of the boot at $4000, the X register contains a #$60.

If you disassemble from $40D8, you should find

```
40D8-  A9 06       LDA   #$06
40DA-  8D 01 AC    STA   $AC01
40DD-  A9 01       LDA   #$01
40DF-  8D EA B7    STA   $B7EA
40E2-  8D F8 B7    STA   $B7F8
40E5-  A5 2B       LDA   $2B
40E7-  8D E9 B7    STA   $B7E9
40EA-  8D F7 B7    STA   $B7F7
40ED-  4A          LSR
40EE-  4A          LSR
40EF-  4A          LSR
40F0-  4A          LSR
40F1-  AA          TAX
40F2-  A9 00       LDA   #$00
40F4-  9D 78 04    STA   $0478,X
40F7-  9D F8 04    STA   $04F8,X
40FA-  20 93 FE    JSR   $FE93
40FD-  20 89 FE    JSR   $FE89
4100-  4C 84 9D    JMP   $9D84
```



By the time this code executes, the Tic Tac Show DOS is in place, but not yet initialized. A value of #$06 is stored at $AC01 to set up the access of the directory on track $06, and the code goes on to set up some memory locations used by DOS and does the equivalent of an IN#0:PR#0. At $4100, the JMP $9D84 is the entry to the DOS coldstart routine. When this routine is executed, the DOS will be initialized and the disk's HELLO file (Menu/IBC) will be loaded in and take over. We don't want this to happen. Instead, change the instruction at $4100 so that it jumps to the monitor

```
4101:69 FF
```

The code that we placed at $900 for turning on the drive and setting up zero page is still intact and is again needed for executing the next stage of the boot. It has to be changed a little bit, however, so that the X register will hold a #$60 and the jump will be to $4000 instead of $801. Make this change by typing

```
916:A2 60 4C 00 40
```

and continue the boot with

```
900G
```

When the drive shuts off, all the code that we need to save will be in memory. By experimentation, I found the necessary code to be at $800-$1FFF and $8C00-$95FE, in addition to the DOS at $9600-$BFFF. To recover this code, it will have to be moved to a location in memory that is not overwritten by the boot of a slave disk ($900-$95FE is safe). To do this, perform the following memory moves

```
6000<800.8FFM
2100<8C00.BFFFM
```

and then boot a DOS 3.3 slave disk by typing

```
C600G
```

Before saving this code to disk, enter the monitor, move page $08 back into place with

```
CALL-151
800<6000.60FFM
```

and then enter some code at $20DE that will put $8C00-$BFFF back into place before jumping to the DOS coldstart routine

```
20DE:20 89 FE
20E1:20 93 FE A0 00 84 3C 84
20E9:42 88 84 3E A9 21 85 3D
20F1:A9 54 85 3F A9 8C C8 20
20F9:2C FE 4C 84 9D
```

This code should disassemble as

```
20DE-  20 89 FE    JSR   $FE89
20E1-  20 93 FE    JSR   $FE93
20E4-  A0 00       LDY   #$00
20E6-  84 3C       STY   $3C
20E8-  84 42       STY   $42
20EA-  88          DEY
20EB-  84 3E       STY   $3E
20ED-  A9 21       LDA   #$21
20EF-  85 3D       STA   $3D
20F1-  A9 54       LDA   #$54
20F3-  85 3F       STA   $3F
20F5-  A9 3C       LDA   #$3C
20F7-  85 43       STA   $43
20F9-  C8          INY
20FA-  20 2C FE    JSR   $FE2C
20FD-  4C 84 9D    JMP   $9D84
```

After you have entered this code and checked it, save $800-$54FF to disk by typing

```
BSAVE TIC TAC SHOW,A$800,L$4D13
```

About the only thing left to do is to copy tracks $6-$22 of the original Tic Tac Show disk to an initialized disk (preferably with a fast DOS) and create a HELLO program for the disk. The necessary tracks can be copied

*See No. 17 - page 24 - correction

either with a bit copier or by Super IOB, and the Standard Controller with the variable TK in line 1010 changed from 0 to 6. The Applesoft HELLO program must relocate itself to a free area of memory ($800-$54FF will be occupied). Therefore, BLOAD in the file TIC TAC SHOW and then do a CALL 8414 ($20DE). This HELLO program is listed near the end of the article.

Tic Tac Show is definitely not one of the more difficult disks to boot code trace. As such, this program would be a good beginner project for those wishing to learn the technique. The principles covered here can definitely be applied to disks with tougher forms of copy protection if desired. The steps necessary for boot code tracing Tic Tac Show are recapped below.

A Recap

1) Initialize a blank disk, preferably with a fast DOS. The volume number of this disk must be 1

```
INIT HELLO,V1
```

2) Use a bit copier or Super IOB with a modified Standard Controller to copy tracks $6-$22 from the original Tic Tac Show disk to the disk initialized in Step 1.

3) Boot up DOS 3.3, enter the monitor, move the code from the disk controller ROM down to page $96 and modify it to save some necessary zero page locations before turning off the drive motor and entering the monitor at $FF69

```
CALL-151
9600<C600.C6FFM
96F8:A0 43 A9 00 85 FC 85 FD
9700:85 FE A9 60 85 FF B1 FC
9708:91 FE 88 10 F9 8D E8 C0
9710:4C 69 FF
```

4) Insert the original Tic Tac Show disk into the drive and boot it by typing

```
9600G
```

5) After the drive turns off and the monitor prompt reappears, modify the exit from the BOOT1 routine so that it jumps to the code at $970D, where the drive will be turned off and the monitor is entered

```
84A:4C 0D 97
```

6) At $900, place a routine which will turn on the drive, wait about one second, restore the necessary zero page locations and then jump to the BOOT1 code at $801

```
900:8D E9 C0 A0 09 A9 C0 20
908:A8 FC 88 D0 F8 A0 43 B1
910:FE 91 FC 88 10 F9 4C 01
918:08
```

7) Read the code necessary for accessing the half tracks into memory by typing

```
900G
```

8) The next stage of the boot starts at $4000 and exits at $4100, where a jump to the DOS coldstart routine is taken. Modify this jump so that it enters the monitor instead of performing the coldstart, by typing

```
4101:69 FF
```

9) The code we entered at $900 is still necessary for turning on the drive; however, it must be modified a bit so that a jump to $4000 is taken with the X register set to #$60 (for the slot #). Modify this code and then execute it by typing

```
916:A2 60
918:4C 00 40 N 900G
```

10) After the drive has been on for about thirty seconds, the monitor prompt should reappear and the drive will turn itself off. All the code that has to be recovered is now in memory. Insert the DOS 3.3 slave disk that was initialized in Step 1 into the drive. Before booting it, move the code to "safe" areas of memory. Do this by typing

```
2100<8C00.BFFFM
6000<800.8FFM
C600G
```

11) After the DOS 3.3 slave disk has been booted, enter the monitor and move page $60 back down to page $08 by typing

```
800<6000.60FFM
```

12) At $20DE, enter the routine which will move code at $2100-$54FF back to its proper location ($8C00-$BFFF) before jumping to the DOS coldstart routine

```
20DE:20 89 FE
20E1:20 93 FE A0 00 84 3C 84
20E9:32 88 84 3E A9 21 85 3D
20F1:A9 54 85 3F A9 8C 85 43
20F9:C8 20 2C FE 4C 84 9D
```

13) Next, save memory from $800-$54FF to the disk by typing

```
BSAVE TIC TAC,A$800,L$4D13
```

14) Finally, type in the following HELLO program and save it to the disk

```
* 10 ON PEEK (104) = 96 GOTO 20 : POKE
     104,96 : POKE 8192,0 : PRINT
     CHR$ (4) "RUN^HELLO"
  20 PRINT CHR$ (4) "BLOAD^TIC^
     TAC^SHOW,A$800"
  30 CALL 8414
```

The resulting disk is completely COPYAble and, in addition, will boot quite a bit faster than the original did.

Carol awaits you.

Most Wanted List

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

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

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

If you know how to de-protect, unlock or modify any of the programs below, we encourage you to help other Hardcore COMPUTIST readers and earn some extra money at the same time. Send the information to us in article form on a DOS 3.3 diskette.

1. **Apple Business Graphics**
   Apple Computer
2. **Flight Simulator II**
   Sub Logic
3. **DB Master 4.0**
   Stoneware, Inc.
4. **DB Master 4.0 +**
   Stoneware, Inc.
5. **Bookends**
   Sensible Software
6. **Visiblend**
   Micro Lab
7. **Dollars And Sense**
   Monogram
8. **Word Juggler**
   Quark, Inc.
9. **Catalyst**
   Quark, Inc.
10. **Gutenburg Jr. & Sr.**
    Micromation LTD
11. **Prime Plotter**
    Primesoft Corp.
12. **The Statistics Series**
    Human Systems Dynamics
13. **Sargon III**
    Hayden
14. **Zardax**
    Computer Solutions
15. **List Handler**
    Silicon Valley Systems
16. **Milliken Math Series (NEW)**
    Milliken Publishing
17. **Sky Fox**
    Electronic Arts

Whiz Kid by Ray Darrah

Have you ever wondered exactly how the disk drive is manipulated? This important function is accomplished through a series of control registers. Each disk controller card manipulates the two disk drives attached to it with sixteen (16) of these control registers.

When the disk controller card is inserted into a slot, its associated control registers are mapped into the Apple's memory at $C0n0 through $C0nF where n is equal to the slot number plus eight (8). Example: The registers of a disk controller card in slot 6 would be mapped into locations $C0E0 through $C0EF. These control locations are arranged as follows:

Addr.	Label	Function
$C0n0	MAG0.OFF	Stepper magnet 0 off
$C0n1	MAG0.ON	Stepper magnet 0 on
$C0n2	MAG1.OFF	Stepper magnet 1 off
$C0n3	MAG1.ON	Stepper magnet 1 on
$C0n4	MAG2.OFF	Stepper magnet 2 off
$C0n5	MAG2.ON	Stepper magnet 2 on
$C0n6	MAG3.OFF	Stepper magnet 3 off
$C0n7	MAG3.ON	Stepper magnet 3 on
$C0n8	MOTOR.OFF	Drive and motor off
$C0n9	MOTOR.ON	Drive and motor on
$C0nA	SEL.DRV1	Route power to drive 1
$C0nB	SEL.DRV2	Route power to drive 2
$C0nC	IO1	Strobe latch for I/O
$C0nD	IO2	Load data latch
$C0nE	IO3	Prepare latch for input
$C0nF	IO4	Prepare latch for output

Referencing

Most of the registers perform a function by merely referencing (a read or write must be done to a particular address) their corresponding memory address. In a reference, the data which is read or written is meaningless. Thus, to reference one of the above registers, a peek or poke (or their machine language equivalents) must be done to the corresponding address. The value returned from the location or poked into the location makes no difference during a reference.

Registers C and D (mapped into locations $C0EC and $C0ED if your disk is in slot 6); however, must be read or written to (depending upon which mode the card is in). Only in their case is the value read or written of importance.

Motor And Selectors

Referencing the Motor.on register will supply power to the drive, turn on the red LED and start the main motor spinning. To observe this in action, (assuming your disk drive is in slot 6) try the following:

POKE49385,0 (Drive turns on)
CALL -151 (Drive still on)
C0E8 (Drive turns off)

The number 49385 in the first command is equivalent to $C0E9 which is the Motor.on register. The Motor.on register must be referenced before any of the other registers can be accessed.

Drive One Or Two

Registers A and B work like channels. If register B is referenced, then all subsequent references will affect drive two. Register A has the same effect on drive 1. Interesting effects can be produced by first turning on the drive motor and then quickly switching from drive 1 to drive two and vice versa. Try this:

CALL -151

```
02FF: C0          ; DELAY TIME
0300: AD E9 C0    ; LDA   $C0E9
0303: AD EA C0    ; LDA   $C0EA
0306: AD FF 02    ; LDA   $02FF
0309: 20 A8 FC    ; JSR   $FCA8
030C: AD EB C0    ; LDA   $C0EB
030F: AD FF 02    ; LDA   $02FF
0312: 20 A8 FC    ; JSR   $FCA8
0315: AD 00 C0    ; LDA   $C000
0318: 10 E9       ; BPL   $0303
031A: AD E8 C0    ; LDA   $C0E8
031D: AD 10 C0    ; LDA   $C010
0320: 60          ; RTS
```

Don't type the material to the right of the semicolons. This is what a disasembly (300L) would show. To start the program type:

300G

If you have two disk drives (in slot 6), then their LEDs are now flashing alternately. If you have only one disk drive then its LED is flashing. This is a result of the program first turning on the disk motor (LDA $C0E9) and then switching back and forth between drives. To halt the program, press any key. To produce a different effect, change the number at $2FF. For example, to seemingly have both drives on at the same time, you would type:

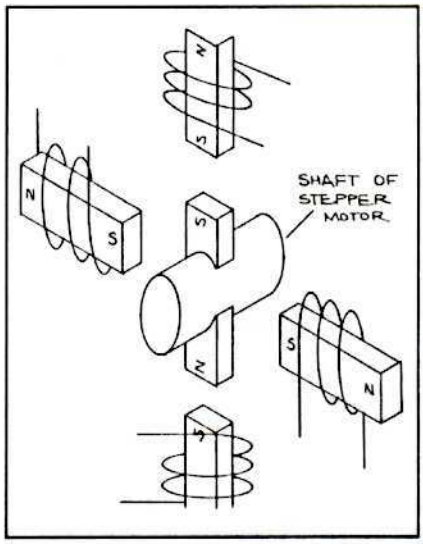
2FF: 20
300G

The Stepper Motor

There are two motors in your disk drive. The first one, the motor that comes on when the Motor.on register is accessed, we have already discussed. This motor spins at a very stable speed and is responsible for turning your diskette round and round.

The second is a very special type of motor called a stepper motor. This motor might look like the simplified diagram above.

By energizing any one of the input lines (which correspond to the registers above) you turn on an outer magnet which attracts the permanent magnet on the drive shaft, thus turning the drive shaft until it is aligned with the energized magnet. By turning these outer magnets on and off in sequence, the motor will spin. You can stop the motor at any one of the four magnets.



Although moving the disk head is a job best left to DOS (mainly because of the precise delays required), here is a BASIC program for moving the disk head to any location (half tracks included).

```
10 HOME : INPUT "SLOT=>" ;SL :
    PRINT : INPUT "DRIVE=>" ;DR
20 PRINT : INPUT "CURRENT^
    PHASE=>" ;CP : PRINT : INPUT
    "DESTINATION^PHASE=>" ;DP
30 AD = 49152 + 128 + SL * 16 : POKE
    AD + 9 ,0 : POKE AD + 9 + DR ,0
40 IF DP = CP THEN 110
50 IF DP > CP THEN 80
60 FOR A = CP * 2 + 1 TO DP * 2 STEP - 1
70 POKE AD + A - INT (A / 8 ) * 8 ,0 :
    NEXT : GOTO 110
80 FOR A = CP * 2 + 1 TO DP * 2 + 1 STEP
    2
90 POKE AD + A - INT (A / 8 ) * 8 ,0
100 POKE - 1 + AD + A - INT (A / 8 ) * 8
    ,0 : NEXT
110 POKE AD + 8 ,0 : END
```

DOS divides disks into seventy (70) phases, each phase corresponding to one stepper motor position called a half track. Normally, DOS writes its tracks on the even phases (phase 0, 2, 4 etc.) As a result, phase two (2) is usually referred to as track 1, phase four (4) is referred to as track 2 and so on until phase sixty eight (68) which is track 34. The odd phases are called half tracks.

When using the disk head move program, if you aren't sure which is the current phase, type eighty (80) for the current phase and zero (0) for the destination phase. This will recalibrate the disk head and you will then know that the current phase is zero (0).

-continued in No. 17, page 27

Looking for a

SPECIAL DEAL

on Library Disks and Back Issues from Hardcore COMPUTIST?

Check out these SPECIAL combination OFFERS!

CORE 1, 2, 3 . . . . . $8.50 ☐

Hardcore COMPUTIST 1, CORE 1, & Library Disk #1 . . . . . $14.95 ☐

Hardcore COMPUTIST 4, CORE 2, & Library Disk #2 . . . . . $14.95 ☐

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

Hardcore COMPUTIST 1, 4, 6, 8-13 (Limited supplies) . . . . . $26.00 ☐

☐ Please send the SPECIALS I have checked above to:

Name _____ ID# _____

Address _____

City _____ St _____ Zip _____

Phone _____

VISA/MC _____ Exp _____

Signature _____

Send check or money order to: Hardcore COMPUTIST, PO Box 110846-B, Tacoma, WA 98411. Most orders shipped UPS. Please use your street address. Washington residents add 7.8% sales tax. Foreign orders add 20% shipping and handling. US funds drawn on US bank. See the ad on page 30 for other library disks and back issues at our regular rates.

Moving Soon?

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

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

That's all there is to it!

My new address is:

Address _____

City _____ St _____ Zip _____

Phone _____

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

Name _____ ID # _____

Address _____

City _____ St _____ Zip _____

Country _____

Phone _____

Hardcore COMPUTIST..

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

NOT PIRACY

Just Good Sense!

Annual Subscription Rates:

Please check one

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

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

Name _____

Address _____

City _____ St _____ Zip _____

Country _____

If renewal ID # _____ Phone _____

VISA/MC _____ - _____ - _____ - _____ Exp _____

Signature _____

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

Dealer Inquiries Invited

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

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

WHO LIKES THE CIA ?

HERE ARE JUST A FEW OF THE MANY

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

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

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

"multifaceted" - NIBBLE

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

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

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

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

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

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

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

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

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

ORDER Gift Subscriptions NOW!

Send gift subscription to:

Name_____

Address_____

City _____ St _____ Zip _____

From: _____

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

Send gift subscription to:

Name_____

Address_____

City _____ St _____ Zip _____

From: _____

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

Send gift subscription to:

Name_____

Address_____

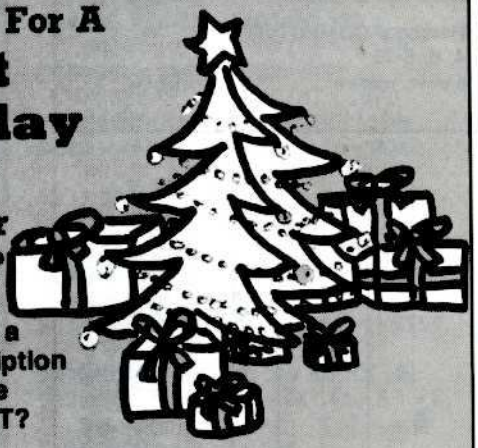
City _____ St _____ Zip _____

From: _____

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

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

Looking For A Great Holiday Gift For Your Friends?



How about a gift subscription to Hardcore COMPUTIST?

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

My Name _____

Address_____

City _____ St _____ Zip _____

Phone _____

VISA/MC _____ Exp_____

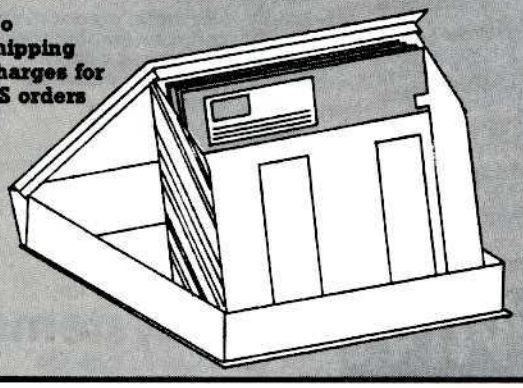
Signature _____

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

Add COLOR to your computer library!

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

No shipping charges for US orders



Order 10 diskettes for the low price of $17.00 AND get one color-coded library case* absolutely FREE!

SS/DD 5 1/4" diskettes with hub rings. Includes Tyvek sleeves. Guaranteed, 100% certified, full surface tested. ** Matching labels accompany each color coded library case.*

Please send me:

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

Number of sets × $17.00 $ _____

Applicable taxes or shipping $ _____

TOTAL $ _____

Specify color choices:

____ yellow ____ red ____ blue ____ green

____ beige ____ grey ____ brown ____ navy

Name _____

Address _____

City _____ St _____ Zip _____

Phone _____

VISA/MC _____ Exp _____

Signature _____

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

SIMPLY·SOFTWARE

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

LIST $24.95 **SPECIAL $17.47**
ABM
Know Your Apple IIe
Typing Tutor

LIST $25.00 **SPECIAL $17.50**
Fortress of the Witch King
Free Trader
Galaxy
Parthian Kings
Under Southern Skies

LIST $29.95 **SPECIAL $20.97**
Alphebet Zoo
Castle Wolfenstein
Caverns of Freitag
Early Games for Y.C.
International Grand Prix

LIST $30.00 **SPECIAL $21.00**
Dreadnoughts Fax
Tournament Golf

LIST $34.95 **SPECIAL $24.47**
Beyond Castle Wolfenstein
Centipede
Dig Dug
Lode Runner
Pacman
Story Machine
Transylvania
Donkey Kong

LIST $35.00 **SPECIAL $24.50**
Close Assault
Sorceress
Empire of the Overmind

LIST $39.95 **SPECIAL $27.97**
Algebra 1
Algebra 2
Broadsides
Copy II +
Face Maker
50 Mission Crush
Master type
Starcross
Stickybear ABC
Seastalker
Type Attack
Zork I, II or III

LIST $40.00 **SPECIAL $28.00**
Legionaire
Micro Cookbook
Summer Games
TAC
Temple of Apshai

LIST $49.95 **SPECIAL $34.97**
Address Book
Crossword Magic
Deadline
Enchanter
Flight Simulator II
Infidel
Kraft Joystick
Math Blaster
Planetfall
Print Shop
Questron
Run for the Money
Sargon II
Sorcerer
Wizardry
Word Attack

LIST $59.95 **SPECIAL $41.97**
Millionaire
S.A.M.
Ultima II or III

LIST $69.95 **SPECIAL $48.97**
Bank Street Speller
Bank Street Writer
Homeward
Pro. Blackjack
Speed Reader

LIST $100.00 **SPECIAL $70.00**
Dollars & Sense
Time Is Money
Form Letter
Module (MUSE)

LIST $125.00 **SPECIAL $87.50**
PFS: File
PFS: Graph
PFS: Report
PFS: Write
Sensible Speller
Supertext

	LIST	SPECIAL
ASCII Express the Pro	$129.95	**$ 90.97**
B-1 Nuclear Bomber	$ 21.00	**$ 14.70**
Bookends	$124.95	**$ 87.47**
BPI General Ledger	$395.00	**$276.50**
DB Master 4.0	$350.00	$245.00
Dbase II	$495.00	$346.50
Death in the Carribean	$ 35.00	$ 24.50
Dow Jones Market Analyzer	$349.00	**$244.30**
Einstein Compiler Version 5.3	$129.00	**$ 90.30**
Facts in Five	$ 26.00	$ 18.20
Home Accountant	$ 74.95	**$ 52.47**
Jane	$295.00	**$206.50**
Knoware	$ 95.00	$ 66.50
Mach II Joystick	$ 44.95	**$ 31.47**
Mach III Joystick	$ 54.95	**$ 38.47**
Main Street Filer	$249.95	**$174.97**
Mastering the SAT	$150.00	**$105.00**
Micromodem IIe w/Smartcom	$329.00	**$230.30**
Microsoft Fortran	$195.00	**$136.50**
Multipan	$195.00	**$136.50**
Tellengard	$ 28.00	**$ 19.60**
Oil Barons	$ 54.00	**$ 37.80**
Rocky's Boots	$ 49.44	**$ 34.61**
Sales Edge	$250.00	**$175.00**
Sargon II	$ 19.95	**$ 13.97**
Snooper Troops #1	$ 44.95	**$ 31.47**
Study Programs for the SAT	$ 89.95	**$ 62.97**
Systems Saver	$ 89.95	**$ 62.97**
Think Tank	$150.00	**$105.00**
Visicalc	$250.00	**$175.00**
Visischedule	$300.00	**$210.00**
Wordstar	$495.00	**$346.50**

PLEASE make check or M.O. payable to: Simply Software Inc. • P.O. Box 36068 • Kansas City, Missouri 64111

Add $3.00 shipping, Missouri residents add 5 5/8% sales tax. Allow 4-6 weeks for delivery.

Apple][,][+, //e, Franklin users:

Do you have problems backing-up your copy-protected programs?

Do you lack parameters for your copy programs?

Are you looking for programs that you can AFFORD?

Are you hesitating to upgrade your equipment due to expensive prices quoted in other ads?

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

Send $1.00 U.S. funds to:

Reliant
P.O. Box 33610
Sheungwan, Hong Kong

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

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

- **Saves copying time**
- **For nibble programs**

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

Apple is a registered trademark of Apple Computer Inc.

FREE INTRODUCTORY BONUS with purchase of Trak Star

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

99.95 Plus $3 shipping and handling charge

Foreign airmail & handling $8.00.

Adapter required for 2-drive systems: $12

Documentation only: $3
Refundable with purchase of Trak Star

Personal checks, M.O., Visa and Mastercard

Midwest Microsystems

Phone 913 676-7242

9071 Metcalf / Suite 124
Overland Park, KS 66212

Still typing in programs that appear in Hardcore COMPUTIST?

Order the LIBRARY DISK, instead!

Save yourself time and trouble. Each month a Library Disk with all the programs that appeared in the previous issue of Hardcore COMPUTIST is prepared for smart readers like you who have better things to do with their time than type in program listings. (Interested in receiving the accompanying Library Disk each month automatically with your magazine subscription? Write for details.)

Name _____ ID# _____

Address _____

City _____ St _____ Zip _____

Phone _____

VISA/MC _____-_____-_____-_____ Exp _____

Signature _____

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

(Don't miss the SPECIAL Library Disk and Back Issue combination offers on page 27)

Please send me the library disks I have checked below:

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

_____ / _____ / _____ / _____ / _____

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

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

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

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

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

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

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

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

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

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

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

NOTE: Library Disks need accompanying magazine for documentation.

BACK UP YOUR DISKS

NOW AVAILABLE AT YOUR LOCAL COMPUTER STORE!

$79.95

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

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

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

ESSENTIAL DATA DUPLICATOR III™

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

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

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

Join The "OFFICIAL" DISKBUSTERS team!!!

Hardcore COMPUTIST says:

"We ain't afraid of no disk."

You can, too, with the "Official" T-shirt* of the Diskbusters team. Available in adult sizes only (black & red on gray).

Rush me _____ (total) DISKBUSTERS T-shirts in the sizes indicated below

ADULT: _____Small _____Med. _____Large

Name _____

Address _____

City _____ St _____ Zip _____

Phone _____

VISA/MC _____-_____-_____-_____ Exp _____

Signature _____

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

*** Hanes 50% cotton/ 50% polyester**

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

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

Hardcore COMPUTIST 12: Softkeys for Zoom Grafix, Flip Out, Lion's Share, Music Construction Set, Hi-Res Computer Golf II, Suicide, Sabotage, Millionaire, Time Is Money, & Type Attack, Psychedelic Symphony / The CORE Disk Searcher / The Armonitor / Pseudo-ROMs on the Franklin Ace.

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

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

Hardcore COMPUTIST 9: Super IOB / Softkeys for Sensible Speller, Sierra On-Line Software, The Visible Computer: 6502, Visidex, Music Construction Set, Gold Rush, Visiterm, & Cosmic Combat / ProDOS to DOS: Single Drive Conversion Technique / Using ProDOS on a Franklin Ace / CORE Word Search Generator / REVIEW: The Visible Computer vs. Apple II- 6502 ALT.

Hardcore COMPUTIST 8: Softkeys for Robotron, Legacy of Lylgamyn, The Artist, Data Factory v5.0, Essential Data Duplicator, The Spy Strikes Back, Hayden Software, & Apple LOGO / COREfiler Formatter / ProDOS Data Encryptor / Bit Copier Comparison.

Hardcore COMPUTIST 6: Program Enhancements: Quick.Bug / Personalizing A Program / Modified ROMs / Data Bases / Data Base Management / Crunchlist / Review: Essential Data Duplicator / Softkeys for Pandora's Box, Donkey Kong, Caverns of Freitag & Visifile.

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

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

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

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

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

Are you a NEW SUBSCRIBER?

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

Please send me the back issues I have checked:

HC 1	$3.50 ☐	HC 12	$3.50 ☐
HC 4	$3.50 ☐	HC 13	$3.50 ☐
HC 6*	$3.50 ☐	HC 14	$3.50 ☐
HC 8*	$3.50 ☐	CORE 1 Graphics	$3.50 ☐
HC 9*	$3.50 ☐	CORE 2 Utilities	$3.50 ☐
HC 10*	$3.50 ☐	CORE 3 Games	$3.50 ☐
HC 11	$3.50 ☐	* Limited supplies. Orders filled as received.	

Name _____ ID# _____

Address _____

City _____ St _____ Zip _____

Phone _____

VISA/MC _____-_____-_____-_____ Exp _____

Signature _____

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

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

By Hackers For Hackers

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

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

The BOOT-LEGGER MAGAZINE

Subscribe Now!

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

You can get the programs that appear in Hardcore COMPUTIST on disk!

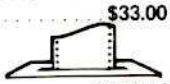
(See page 30 for information)

DISCOUNTS!

5-1/4 DISKETTES & STORAGE

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

PRINTERS



- EPSON RX-80 DOT MATRIX ..... $229.00 *
- PANASONIC P1091 DOT MATRIX ..... $309.00 *
- EPSON RX-80 F/T ..... $289.00 *
- EPSON RX-100 ..... $389.00 *
- OKI-DATA MICROLINE 92A DOT MATRIX ..... $369.00 *
- SILVER REED 400 LETTER QUALITY $309.00
- STARWRITER A-10 18CPS LETTER QUALITY $495.00
- TOSHIBA 1340 DOT MATRIX AND LETTER QUALITY COMBINED $795.00

PRINTER INTERFACES AND ACCESSORIES

- STANDARD PARALLEL INTERFACE CARD ..... $49.00 *
- GRAPHICS PARALLEL INTERFACE CARD $75.00
- FINGERPRINT PUSH-BUTTON GRAPHICS CARD FOR IMAGEWRITER $114.00
- MICROFAZER GENERAL PRINT BUFFER $149.00
- PRINTER STAND $14.00
- SWITCH BOX 3 PARALLEL PORTS $129.00
- SWITCH BOX, 3 SERIAL PORTS $79.00

FLOPPY DISK DRIVES

- FOURTH DIMENSION (FULL OR SLIMLINE) $179.00
- ADAMA TEK $169.00
- LASER SLIMLINE $154.00
- TITAN SLIMLINE FOR IIc ..... $249.00 *
- DISK CONTROLLER $59.00
- DOUBLE-SIDED DRIVE $249.00
- 650K RANA DRIVE $439.00

HARD DISK DRIVES

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

MONITORS

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

MODEMS

- ZOOM TELEPHONICS 300-BAUD $109.00
- CENTAURI 300 BAUD $179.00
- PRO-MODEM 1200 ..... $349.00
- SIGNALMAN MARK XII ..... $259.00 *

GRAPHICS DEVICES

- POWER PAD & STARTER KIT $119.00

VIDEO & DISPLAY EQUIPMENT

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

GENERAL ITEMS

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

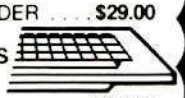
GAME I/O DEVICES

- 9-16-PIN ADAPTER FOR IIe OR IIc ..... $9.00 *
- TWIN PORT GAME I/O EXTENDER $25.00
- SINGLE PORT GAME I/O EXTENDER $18.00
- TG JOYSTICK $31.00
- SAMPSON JOYSTICK $25.00
- CH MACH II JOYSTICK $37.00
- CH MACH III JOYSTICK $45.00
- CH PADDLE STICKS ..... $37.00 *

SLOT EXPANSION

- 16 RAM CARD $49.00
- 64K RAM & 80 COLUMN CARD FOR IIe $115.00
- MEMORY MASTER IIE 64K + RAM & 80 COLUMN CARD $145.00
- MEMORY MASTER IIE 128K RAM & 80 COLUMN CARD $195.00
- MICROTEK II + 128 K VISICALC AND MEMORY EXPANSION $219.00
- CCS 7710 SERIAL INTERFACE CARD $117.00
- MODEM CABLE FOR CCS CARD $21.00
- SERI-ALL SERIAL INTERFACE CARD ..... $119.00 *
- 80-COLUMN CARD (VIEWMASTER) WITH SOFT-SWITCH $139.00
- CENTAURI APS Z-80 CARD $59.00
- Z-80 PLUS CARD (CPM FOR APPLE) $115.00
- FAST Z-80 CARD APPLICARD $195.00
- TIMEMASTER II CLOCK/CALENDAR CARD $109.00
- QUIK-LOADER PROM BOARD $149.00
- ANALOG/DIGITAL BOARD $99.00
- SUPER I/O BOARD $49.00
- MULTIPLE-SLOT EXPANSION CHASSIS $149.00
- SINGLE-SLOT EXTENDER $29.00

SPECIAL PERIPHERALS



- COOLING FAN WITH SURGE PROTECTOR $39.00
- TITAN KEYBOARD ..... $159.00 *
- LIFETIME EXTERNAL POWER SUPPLY $179.00
- SHIFT KEY MOD KIT $8.00
- SCREEN SWITCHER/ DRIVE STEPPER $74.00

APPLE SOFTWARE

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

SPECIAL SALE!!!
EPSON FX-80, 160 CPS.
379.00*!!!
(WHILE THEY LAST!)

* DENOTES NEW PRICE OR ITEM

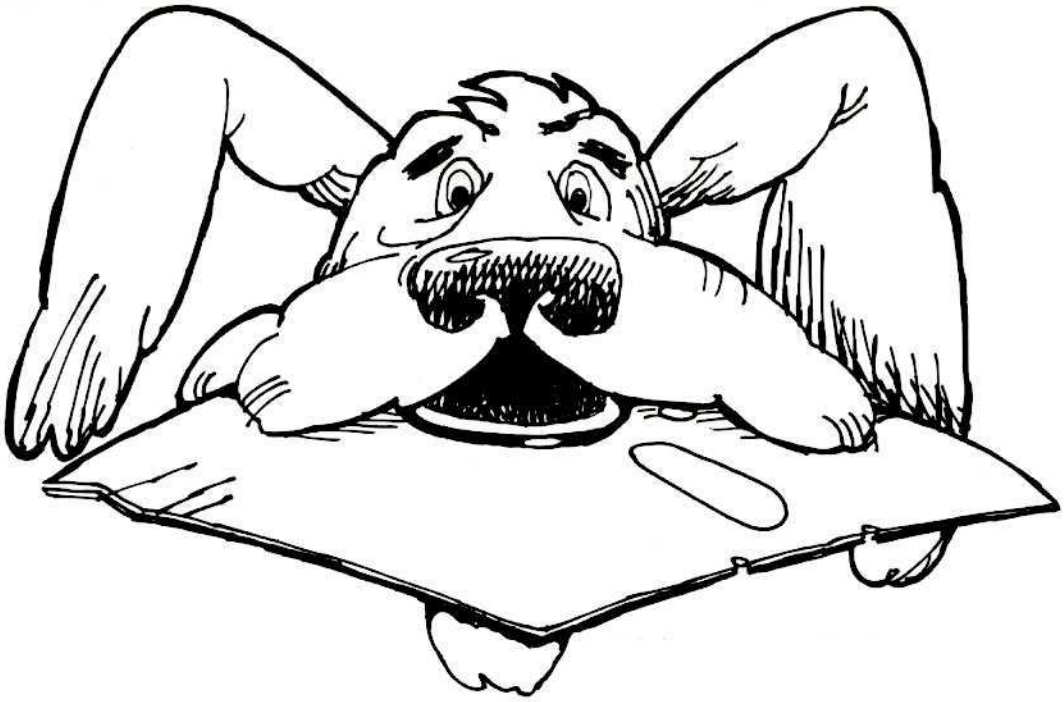
(202) 363-1313

VF ASSOCIATES

6327 WESTERN AVE NW WASHINGTON DC 20015

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

CALL FOR OUR FREE CATALOG



Having problems with retrieval?

You need software insurance.

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

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

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

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

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

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

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

Copy II PC (IBM)

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

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

CENTRAL POINT Software, Inc.

The Backup Professionals

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