UNPROTECTED

Backups may be made using standard copying procedures.

COMPATIBLE

Apple®IIc, IIe or 64K II+
Reads both DOS 3.3 *and* ProDOS™ disks

· PRO-BYTER ·

NEW APPLE®DOS UTILITY
by BERT KERSEY and JACK CASSIDY

BYTEZAP.PRO

Inspect ProDOS™ and DOS 3.3 disks at the byte level with the ability to make changes, improvements and disk or program repairs that are impossible under normal DOS control. All-*New*—features instantaneous block-to-block (or sector-to-sector) viewing.

FAST DISK SEARCH: Scan an entire disk or disk file for any word or phrase. Trace and search any kind of file on a disk. Find and repair "zapped" disk bytes that might cause malfunction. Instant DOS 3.3 and ProDOS disk maps of your disks show you exactly which sectors or blocks a file occupies.

CUSTOMIZE DISKS: Easy instructions show you how to make changes to your disks, DOS itself, and even other programmers' programs.

EDUCATIONAL INSTRUCTIONS will teach you a ton about Apple's new ProDOS operating system (included on the disk) and Apple disk structure.

NEW ProDOS UTILITIES

MACHINE LANGUAGE SORT: One simple CALL in your Applesoft program will alphabetize a long list of words or numbers—super-fast!

TEXT TYPER: Read ProDOS text files without re-booting another disk.

CONVERTER: Convert any Applesoft program line into machine code that can be CALLed by any Basic program.

LOGGER: Print an alphabetized multi-column catalog of *all* of a disk's files, including hidden subdirectories.

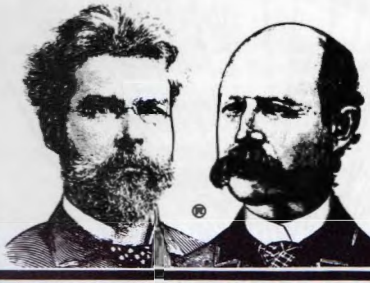
MACHINE LANGUAGE EDITOR: Insert and delete machine code—great for editing programs from magazines. Also works as a handy memory snooper.

PLUS ALL-NEW APPLE TIPS

All new juicy tips, experiments and inside information about your Apple—how to use memory like an extra disk drive, disable LIST so it prints "File Locked", scramble Apple's vocabulary, personalize ProDOS disk catalogs...

Includes free PEEKS & POKES CHART and APPLE TIP BOOK #8

ISBN 0-917085-14-0/



Beagle Bros
MICRO SOFTWARE

PRO-BYTER

NEW APPLE® DOS UTILITY

by Bert Kersey and Jack Cassidy

ISBN 0-917085-14-0

Published by Beagle Bros, Inc.
3990 Old Town Avenue
San Diego, California 92110

TABLE OF CONTENTS

Welcome to Pro-Byter

Pro-Byter is Beagle Bros' first all-ProDOS disk. Its features range from disk zapping to memory snooping to word sorting. Elsewhere in this book you'll find a whole new collection of Apple II tips and tricks. Pull up an Apple and have some fun!

PRO-BYTER USES ProDOS™ VERSION 1.1.1

Most of the programs on the Pro-Byter disk will work with any version of ProDOS; there have been a couple so far. To play it safe, however, when trying the tips and tricks in this manual, you should stick with PRODOS and BASIC.SYSTEM *Version 1.1.1.* Both of these files—called simply "PRODOS" and "BASIC.SYSTEM" on the disk—will load into memory when you boot Pro-Byter. Look for the version number on the screen.

We haven't converted any of Pro-Byter's programs to DOS 3.3. You can try if you want, but don't expect much success; most of the programs will definitely *not* work under DOS 3.3.

WE CAN'T TEACH YOU EVERYTHING

In writing this manual, we assume you know the basics about writing Applesoft programs and loading and saving files under ProDOS. If you don't, dive in anyway; we have done our best to keep things simple.

Even if you think you're an expert, we recommend that you buy Apple's *BASIC Programming With ProDOS* manual. It is designed for programmers of all levels and is extremely well put together.

Chances are, some of the materials and programs included with Pro-Byter will raise more questions than they will answer. If you're interested in more technical details, we give our highest recommendation to the book, *Beneath Apple ProDOS,* published by Quality Software.

Your Apple dealer can get you a copy of either of the books mentioned here. He should already have them on the shelf.

BACK IT UP

In keeping with Beagle Bros tradition, the Pro-Byter disk comes from the factory unlocked and unprotected. This means you can inspect the programs on the disk and make a backup copy in case "something happens" to the original. Every Apple comes with a copy program that will copy Pro-Byter. (Even DOS 3.3 copy programs will copy ProDOS disks.)

PLEASE don't take advantage of our unprotected format by giving even one copy of Pro-Byter away. Every illegal copy you see is a vote *for* copy protection and a vote *against* friendly software.

You support us and we'll support you.

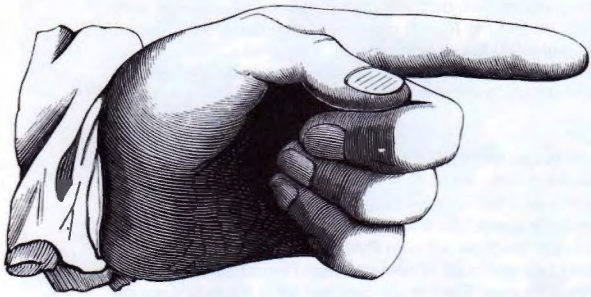
How To Use the Pro-Byter Disk:

Pro-Byter is an Apple ProDOS disk with *three* catalogs or "directories":

1. A Main Directory called /PRO.BYTER
2. A Subdirectory called FREEBIES
3. Another Subdirectory called HOME.MOVIES

Here's what you do to get at each directory's files:

- Boot the Pro-Byter disk—put it in drive 1 and turn on your Apple.
- Type "PREFIX". Your Apple should answer back "/PRO.BYTER", the name of the current directory. Type "PREFIX" any time you want to see which directory is current.
- Type "CAT" to see the names of the files in the current directory.
- Type "PREFIX HOME.MOVIES" or "PREFIX FREEBIES" before doing a "CAT" or executing a program in a subdirectory. If your Apple tells you "Path Not Found", type "PREFIX/" and try again.
- Before executing a program in the main directory, you may have to type "PREFIX/" to get out of any subdirectories.
- To execute a program, read its instructions in this manual. If there are no instructions (check the index on page 80), try typing a hyphen ("-") followed by the name of the file. Certain files, including those whose name end in ".PIC", are not meant to be executed.
- To see possible changes to this manual, type"-NOTES" with the main directory active.



BACK UP YOUR DISKS BEFORE USING BYTEZAP.PRO AND MEM.ZAP.

BYTEZAP.PRO

BYTEZAP.PRO is a disk-inspector program for unprotected Apple ProDOS™ and DOS 3.3 floppy disks. With it you can view individual disk blocks and sectors in hex, decimal and ASCII formats. You can search for occurrences of a specific word in one particular file or anywhere on a disk. You can also make byte-by-byte disk changes that aren't possible under normal DOS control.

ProDOS™ AND DOS 3.3 ONLY

BYTEZAP.PRO was written with normal ProDOS and DOS 3.3 disks in mind. It will also read Apple-compatible CP/M and Pascal disks, but its features will be severely limited (see chart below). Most copy-protected disks are incompatible with BYTEZAP.PRO. Some "partially protected" disks may be partially compatible.

DOS	*Read OK?*	*Write OK?*	*Search OK?*	*Trace Files?*	*Catalog Disk?*	*Map OK?*
ProDOS	Y	Y	Y	Y	Y	Y
DOS 3.3	Y	Y	Y	Y	Y	Y
Pascal	Y	Y	Y	N	N	N
CP/M	Y	Y	N	N	N	N
Protected	?	?	?	?	?	?

BACK *EVERYTHING* UP!

With BYTEZAP.PRO and a touch of a key, you can make a disk inoperable—forever. So make backup copies of all the disks you inspect, using any of the standard Apple copy programs. Beagle Bros' *Extra K* copy program duplicates disks in under 35 seconds.

If there was a way to make BYTEZAP.PRO refuse to work with unbacked-up disks, we would have written it into the code. There isn't, so we didn't.

TO GO, TYPE "-BYTEZAP.PRO"

With ProDOS booted and a disk in your second drive* (if you have a second drive), insert the Pro-Byter disk in your main drive, type "-BYTEZAP.PRO" (or "RUN BYTEZAP.PRO") and you're in business. When the title screen is visible and your disk drive has stopped, you may remove the Pro-Byter disk and put it away until the next time you use BYTEZAP.PRO.

The program is made up of three files: BYTEZAP.PRO, BZAP.BIN and BZAP.VAR. If you want to use BYTEZAP.PRO from another disk, you'll need to transfer all three of these files to that disk.

If attempts to Run BYTEZAP.PRO tell you that files are missing (or "Path Not Found") and you disagree, try re-setting the prefix; ProDOS might be looking in the wrong directory. Type "PREFIX,D1" or "PREFIX,D2" or "PREFIX/"... Or, simply reboot the Pro-Byter disk.

* When you first Run BYTEZAP.PRO, the program will check to see if you have two drives. It is best to have disks in both drives at this time, with the doors closed. Some non-Apple brand drives aren't recognized unless a disk is present.

CURSOR MODE

BYTEZAP.PRO's Cursor Mode shows you a blinking cursor on a screenful of characters that represent either a full DOS 3.3 disk sector (256 bytes) or half of a ProDOS block (also 256 bytes). Regardless of the type of disk or screen format being used, you will always be looking at 256 bytes.

The cursor will be 1, 2 or 3 characters wide, depending on the format in which its byte is displayed; ASCII (1 character), hex (2 characters) or decimal (3 characters). While viewing a SCREEN- or ASCII-format* display, you will see two separate cursors, each on a different interpretation of the same byte (see page 22).

By the way, if you ever have trouble finding the cursor, just move it around by pressing the arrow keys a few times.

MOVING THE CURSOR

The cursor may be moved from byte to byte, with the corresponding byte number and values changing at the bottom of the screen. If you move the cursor off of either end of the screen, you will be advanced to the next appropriate block-half or sector.

Use the four **ARROW KEYS** to move the cursor in the corresponding direction; up, down, left or right. **A** and **Z** serve as alternate keys to move up and down a row. In ProDOS's CAT format, the cursor will move up and down two rows instead of one (from one file name field to the next).

MULTIPLE-BYTE MOVES

Apple IIe and IIc users can hold the **OPEN-APPLE KEY** down while pressing an Arrow to jump the cursor several spaces in any direction. On older Apples, you can use the **COLON** and **HYPHEN** keys (top row, next to the zero) to jump left and right.

Inspired by *AppleWorks*, we made the **1-9 KEYS** quickly re-position the cursor. Pressing "1" (or "0") will jump the cursor to the first byte on the screen. "9" will jump to the end. The other number keys work as you might expect; "5" jumps to the middle byte, etc.

* Read about different
screen formats on pages 22-23.

SCREEN STATS

During Cursor Mode, the bottom line of text will supply you with vital information about the cursor byte and the current sector or block-half. All number values are reported in decimal and hex (in parentheses).

- **ASCII CHARACTER:** The "high-bit off" character of the cursor byte. Control-characters are indicated cA, cB, cC, etc.
- **SCREEN CHARACTER:** The character created by poking the value of the cursor byte to the screen (see page 22). In SCREEN format, this character will match the character at the cursor.
- **VALUE:** The value, 0-255 ($00-$FF), of the byte at the cursor.
- **BYTE:** The distance from the beginning of the current sector (DOS 3.3) or block (ProDOS) to the cursor byte. The range is 0-255 ($00-$FF) for DOS 3.3 sectors and 0-511 ($00-$1FF) for ProDOS blocks.
- **DRIVE NUMBER:** The last drive that was read, 1 or 2.
- **BLOCK/PART (if ProDOS):** ProDOS blocks range from 0 to 279 ($00-$117). Block-halves are designated as "Part A" (bytes 0-255 or $00-$FF) and "Part B" (bytes 256-511 or $100-$1FF).
- **TRACK/SECTOR (if DOS 3.3):** DOS 3.3 sectors range from Track-0/Sector-0 to Track-34 ($22)/Sector-15 ($0F). Each of the 35 tracks have 16 sectors

BYTEZAP.PRO COMMANDS

Most BYTEZAP.PRO functions are controlled by single keystrokes, described in detail on these pages. The Command Card that came with the Pro-Byter disk will serve as a ready reference.

ESC: NEVER MIND

If the program is doing something you don't want it to do (like searching the disk or asking a question that you don't want to answer), just press the ESC key to escape (a RETURN keypress often works too). From Cursor Mode, ESC lets you quit the program entirely.

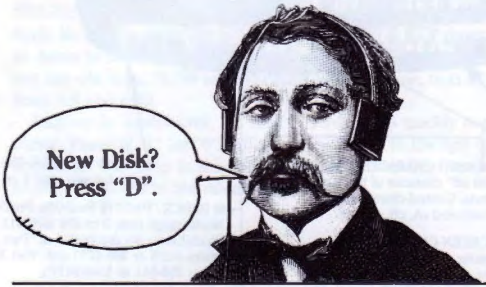
ANY KEY: CONTINUE

Occasionally the program will stop to let you read something on the screen. A flashing "KEY" word indicates that you should press any key (except Shift, Control, etc.) to continue. The RETURN key is a good "any key" choice.

Error messages will appear on the screen for 3-4 seconds. Pressing any key will skip the pause and let you continue.

APPLE=CONTROL

If you're using an Apple IIe or IIc, you may optionally use the left (open) Apple key instead of the control key when typing control-D and control-W.



Important: EVERY TIME YOU INSERT A NEW DISK, press "D" or control-D to establish that disk's characteristics in memory. If you don't, you might cause the Read, Trace and Search features (to name a few) to malfunction.

D: READ DRIVE-1 DISK

control-D: READ DRIVE-2 DISK

Press "D" to read the disk in drive 1, or CONTROL-D (if you have a second drive) to read the disk in drive 2. The program will scan a few sectors and attempt to identify your disk's type as ProDOS or DOS 3.3. It will then display that disk's first directory block or sector in the appropriate CAT format, and the label "PRODOS DISK" or "DOS 3.3 DISK" will appear at the top of the screen (if not, see next paragraph). Under ProDOS, the volume header (the name of the disk) will also be highlighted in inverse. These inverse labels will not appear if you read this sector or block-half by another method.

NON-ProDOS/3.3 DISKS

BYTEZAP.PRO is built to work with ProDOS and DOS 3.3 disks. If pressing "D" reports "Pascal", "CP/M" or "Non-Standard Disk", the disk will be treated as a ProDOS (block-format) disk. If it is simply a ProDOS disk with a non-standard directory format, no problem. If it is indeed *not* a ProDOS or DOS 3.3 disk, you can work with it, BUT the block numbers at the bottom of the screen might be incorrect, the CAT screen format will be meaningless, the search feature might not find words that cross block boundaries, and the disk map will be pure baloney.

DISK/DRIVE ERROR

A "Disk/Drive Error" message followed by a screenful of zeros may mean that your disk is not centered properly (wiggle the drive door and try again). Or it may mean that you are trying to read a disk that is incompatible with BYTEZAP.PRO. The disk could be damaged, unformatted, or copy-protected.

If you think you have a "partially protected" disk, try reading blocks or sectors on different parts of the disk.

BEATING THE SYSTEM

If you don't like the format that BYTEZAP.PRO chooses for a disk, you can insert another disk of the type you want, press "D" to read it, and then insert the disk in question (*don't* press "D" this time). Now press one of the "Y" keys a few times until the drive comes on and reads in a new track. You're on your own from there.

R: READ A BLOCK OR SECTOR

Press "R" to read a specific disk block (ProDOS), disk sector (DOS 3.3) or file trace number (with Trace Mode; see below). Type a number in decimal or hex (preceded by a "$" or "H"). An acceptable value will cause the program to read the appropriate block or sector and display it on the screen. In ProDOS, you will always be presented with Part A of the selected block.

READ LIMITS

ProDOS BLOCKS: 0 to 279 ($00-$117)
DOS 3.3 TRACKS: 0 to 34 ($00-$22).
DOS 3.3 SECTORS: 0 to 15 ($00-$0F)
TRACE NUMBERS: 0 to ?? (depends on the size of the loaded trace-file)

USING "R" FROM TRACE MODE

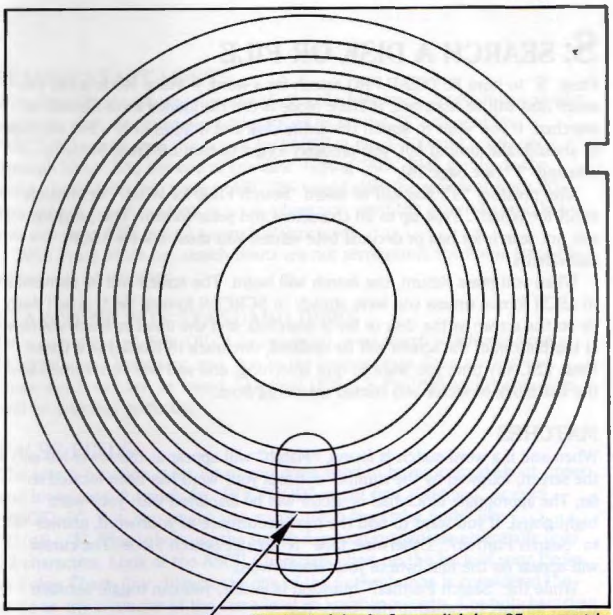
If Trace Mode is on when you press "R", you will be asked "TRACE Which Block (or Sector)?" instead of "READ Which Block (or Track)?". This means which block or sector of the loaded trace-file, not of the disk itself.

EXAMPLE: If you want to read the 13th block of a ProDOS file called "BIGFILE", load the file (page 14), be sure Trace Mode is on (page 16), and press "R" followed by "13" or "$0D" (Return). You will then be presented with the 13th block of the file "BIGFILE". To read the 13th block of the *disk*, turn Trace Mode off by pressing "T". Then press "R" followed by "13" or "$0D".

<>: READ NEXT BLOCK OR SECTOR

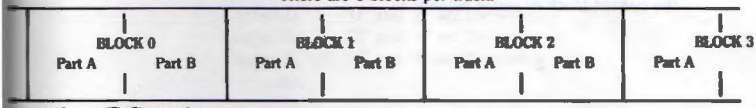
Press the "<" or ">" key (unshifted) to read the next or previous block-half or sector. If Trace Mode is off, you will be switched to the adjacent block-half or sector of the *disk*. If Trace Mode is on, you will be switched to the adjacent block-half or sector of the loaded *trace-file;* all other parts of the disk will be skipped. "Wrap-around" is in effect here; the first block or sector follows the last, and vice versa.

While moving from block to block or sector to sector, the disk drive will only come on when a track boundary is crossed—every 8 blocks or 16 sectors if Trace Mode is off. If you insert a new disk and don't press "D", you will be looking at bytes that really belong to your previous disk. Another reason to always press "D" when switching disks!

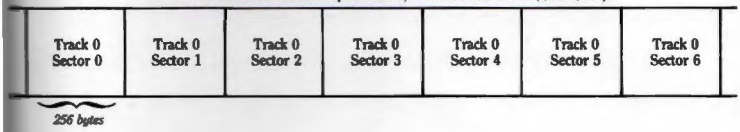


The typical Apple floppy disk has 35 concentric *TRACKS*, numbered 0-34 ($00-$22).

ProDOS disks have 280 *BLOCKS*, numbered 0-279 ($00-$117).
There are 8 blocks per track.



DOS 3.3 disks have 560 *SECTORS*.
There are 16 sectors per track, numbered 0-15 ($00-$0F).



S: SEARCH A DISK OR FILE

Press "S" to have BYTEZAP.PRO search for a word. If Trace Mode is off, the *entire disk* will be searched. If Trace Mode is on, the loaded *trace-file* will be searched. If you want to search the entire disk and it takes only a few seconds (it should take around 40), you probably forgot to turn off Trace Mode by pressing "T" (see page 16).

After pressing "S", you will be asked "Search FILE for What?" or "Search DISK for What?". Type up to 20 characters and press Return. The program will not search for hex or decimal byte values; you must ask for ASCII equivalents.

When you press Return, the search will begin. The screen will be converted to ASCII format unless you were already in SCREEN format. Sectors will flash by on the screen as the disk or file is searched, and the track or block counter at the bottom of the screen will be updated, one track (8 blocks) at a time. Press ESC any time you want to quit searching, and you will be returned to the last block or sector you started searching from.

MATCHES

When and if a word-match is found, "Match" will appear at the lower-left of the screen, followed by the number of times your word has been located so far. The appropriate block-half or sector will be displayed with your word highlighted. If you want to find the next occurrence of your word, answer "Y" to "Search Further?". Otherwise, type "N" to exit Search Mode. The cursor will appear on the first byte of your search word.

While the "Search Further?" question is visible, you can toggle between screen formats by pressing the Space Bar.

There *is no* "find next" command once you exit Search Mode, so you might want to write down the block or sector numbers each time your word is found.

Sorry, if you are not in Trace Mode, there is no easy way to know what file the current block or sector belongs to.

SEARCH-START LIMITS

With Trace Mode off, you may specify the block or track at which you want a search to start. For example, under ProDOS, typing "S" followed by "ABC,100" would start the search for "ABC" at block 100 and continue through block 279, the end of the disk. Typing "ABC,$64" would do the same. Under DOS 3.3, typing "ABC,20" would start the search at track 20. You cannot specify a block or track where you want the search to end, but you can press ESC to end a search at any point.

With Trace Mode on, search limits are not recognized; the entire trace-file will always be searched.

SEARCHING NON-STANDARD DISKS

On a non-standard disk (CP/M for example), your search word may cross a block or sector boundary, and, due to the various ways disks are laid out, these words will not be found. Standard DOS 3.3, ProDOS and Pascal disks will be searched perfectly.

FALSE FINDS

The search function is set up so you don't need to be concerned about upper and lower case. A search for "DOGFOOD" might find "Dogfood" or even "DoGfOoD". But, you might also encounter a few "false finds" like "D/g&r//$". This is because the search function recognizes essentially only 32 characters. Look at the ASCII Characters chart on the back of your Peeks & Pokes Chart. Any character in one of the four columns is considered the same as any character at the same level in another column. Control-characters may therefore be found by typing the upper-case character instead.

Control-A, "!", "A" and "a" are the same.
Control-B, "", "B" and "b" are the same.
Control-C, "#", "C" and "c" are the same.
Control-D, "$", "D" and "d" are the same.
Control-E, "%", "E" and "e" are the same.
Control-F, "&", "F" and "f" are the same.
...etc.

L: LOAD A TRACE-FILE

Tell BYTEZAP.PRO what file you want to trace by "loading" the file from the disk directory. First, find your file's name in a directory (see next page). Then PUT THE CURSOR ON THE FILE'S NAME and press "L". After the trace-file is loaded, you will be placed in Trace Mode, looking at the first sector or block-half of your trace-file.

Loading should take place as long as you have placed the cursor on a real undeleted file name or ProDOS subdirectory name. You can "load" and look at subdirectories, but you cannot load a ProDOS disk volume header, even though it looks like a file name.

LOADING ProDOS FILES FOR TRACING

In CAT format, place the cursor on the file name *or* on either of the two lines that go with the file name. Then press "L".

```
27 STARTUP:::::::::  FC 9900 1E00 003A00
0000-0000 00 00 21 0108 A5AA-0000 0200
```

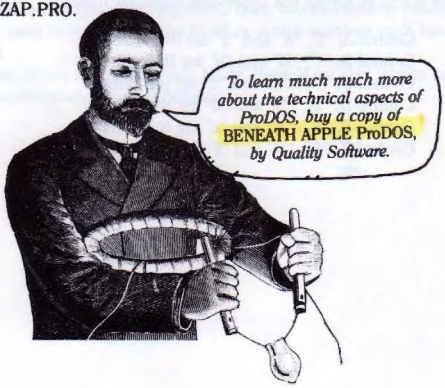
LOADING DOS 3.3 FILES FOR TRACING

In CAT format, place the cursor on the file name *or* anywhere on the same line as the file name. Then press "L".

```
130F82HELLO                                0300
```

TRACE-FILE SIZE LIMITS

Because of memory limitations, you can only load the first 126 sectors or 63 blocks of a file. Larger files cannot be traced automatically with BYTEZAP.PRO.



FINDING A FILE NAME IN A DIRECTORY

DOS 3.3

■ METHOD 1: Press "D" or CONTROL-D to go to the first directory sector—track-17 ($11)/sector-15 ($0F). If you don't see the file name among the seven on the screen, press " < " to see the next group in sector-14 ($0E) (go backwards—sector 15, 14, 13, etc.). If you pass track-17/sector 1 and still don't find the name, try method 2.

■ METHOD 2: Exit Trace Mode ("T") if necessary and press "S" for Search. Type the file name you want to find, followed by ",17" or ",$11" (for example, "BIGFILE,17"). This will do a search on track 17 (and beyond). If the file name isn't found in track 17 (within two seconds), your file probably isn't on the disk, although it could have a strange catalog arrangement or another catalog track. You can always search the entire disk.

ProDOS

■ METHOD 1: Press "D" or CONTROL-D to go to the first directory block—block-2. If you don't see the file name among the six on the screen, press ">" to advance to the next block-half. If you pass block 5, part B, and still don't find the file name, try method 2 or 3.

■ METHOD 2: Exit Trace Mode with "T" if necessary and press "S" for Search. Type the file name you want to find, followed by ",2" (for example, "BIGFILE,2"). This will do a search on directory blocks 2-5—and beyond. If the file name isn't found in blocks 2-5 (within two seconds), let the search continue; your file name might be in a subdirectory elsewhere on the disk.

■ METHOD 3: If you know your file is in a certain subdirectory, go to the main directory (blocks 2-5), put the cursor on the name of your subdirectory and type "L". When the subdirectory is loaded and displayed, switch to CAT format and look for your file name there. Deeper-level subdirectories may be loaded and viewed in the same way.

T: TRACE MODE OFF/ON

Toggle Trace Mode off and on by pressing "T". An inverse file name (perhaps shortened to fit the space) and "Block # of #" or "Sector # of #" will indicate that Trace Mode is active. The numbers on the next line indicate the current disk block or track and sector, as usual.

Turning Trace Mode on makes the search feature search *only* through the loaded trace-file. It also makes the ">" keys trace *only* through the trace-file instead of the entire disk. It also affects the way the "R" Read command works; see page 10.

Trace Mode is automatically turned off by the "D" and control-D commands.

TRACE MODE IS ON if this information appears on the screen.

B: BYTE CHANGE

Press "B" when you want to change a byte or sequence of bytes in the current block or sector. The changes you make will occur *in memory only,* so you can afford to experiment a little with this command. Your changes will not actually be written to the disk until you use the Write command (page 19).

ALL CHANGES WILL BE CANCELLED IF YOU MOVE TO A DIFFERENT BLOCK-HALF OR SECTOR, so if you want your changes to exist on disk, use the Write command before moving on.

A byte may be changed in any format—hex, decimal or ASCII. Before pressing "B", you should be looking at the screen format that corresponds to the way you want to type your changes:

ASCII and SCREEN formats: Make all changes by typing *ASCII characters.*
HEX format: Make all changes by typing *hex values.*
DEC format: Make all changes by typing *decimal numbers.*
CAT format: Make file name changes in *ASCII,* and other changes in *hex.*

HIGH BIT ON OR OFF?

If you are in CAT, ASCII or SCREEN formats when you press "B", you will be asked if you want the "high-bit" set for ASCII byte changes. Take a look at the "ASCII Characters" chart on the back of the Peeks & Pokes Chart that came with Pro-Byter. If you answer "Y", all typed characters will poke in the "High" value for that character into memory. A "N" answer means the "Low" value will be used. For example, with the high-bit set, typing the letter "A" over an existing character would enter the value 193 ($C1) in the place of the old byte. With the high-bit not set, typing an "A" would then enter the value 65 ($41) as the new value.

If you don't know whether the high-bit should be on or off, take a look at the bytes you are changing. If they have the high bit on (values 128-255 or $80-$FF), you probably—well, *maybe*—should leave it on. If they have the high bit off (values 0-127 or $00-$7F), maybe leave it off.

If you are changing a ProDOS file name or directory name, always turn the high-bit OFF (answer "N"). Otherwise, ProDOS commands like "RUN FILENAME" will not recognize the changed file name, even though it looks normal. DOS 3.3 file names should be high-bit ON unless you want flashing and/or inverse characters. You cannot have flashing or inverse characters in ProDOS file names.

WARNING: If you change to another sector or block-half before writing to disk, ALL OF YOUR CHANGES will be lost.

B: BYTE CHANGES (continued)

MAKING THE CHANGES

When you see and hear the clicking cursor, start typing in your byte changes. When you are finished, PRESS ESC TO QUIT MAKING CHANGES.

The clicking noise during Change Mode is a warning that every key you press will potentially poke changes into memory. The louder clickety-click when you first enter Change Mode signals that the track in memory is being checked for later comparison when you Write to disk.

To cancel all of the changes you have made, press "X" (see below) from Cursor Mode. If you want to transfer the changes to disk, use the Write command (next page) *before* going to another block or sector.

HEX AND DECIMAL CHANGES

If the cursor is over a hex or decimal number, type a new number in the current format, two digits for hex (00-FF) or three for decimal (000-255). To leave a number unchanged, you must type over it (the right arrow key won't function here) or press ESC to quit. If you make a mistake, back up with the left arrow key, or press ESC, move the cursor to the incorrect byte, press "B" and try again; or press "X" to start completely over.

ASCII CHANGES

After setting the high-bit on or off (previous page), just type in the characters you want—slowly please. Press ESC to quit making changes. If you mess up, you can press ESC and press "X" to cancel all changes. Or simply move on to another block-half or sector.

Control-characters must be entered from either HEX or DEC screen formats. (When you switch formats, the cursor remains on the same byte.) See the ASCII chart on the back of your Peeks & Pokes Chart for control-character values.

If you want to play tricks with flashing and inverse characters (you can't, always), use the values from the chart on page 77. When making byte changes from ASCII format, you can set the high-bit OFF and get FLASH for ASCII 64-95 and INVERSE for ASCII 32-63. Other FLASH and INVERSE values must be entered numerically from HEX or DEC format.

X: CANCEL CHANGES

Pressing "X" from Cursor Mode will re-read a block-half or sector, cancelling any changes that might have been made, *as long as they haven't been written to disk*. The disk drive will not be accessed by the "X" command.

control-W: WRITE CHANGES TO DISK

By now you've read about backing up your disks on page 5, so we won't mention BACKING UP YOUR DISKS again. Press CONTROL-W (or APPLE-W) to write the current block-half or sector to the disk.

CANNOT WRITE

A "CANNOT WRITE" message means you are trying to write to the wrong disk (not the last one read). Or it means you have made no changes to the current block or sector, so why are you writing? If you disagree with the message, try faking a change by pressing "B", then ESC.

A "WRITE PROTECTED" message means you are trying to write on a disk whose notch is covered or to a disk with no notch.

V: VALUE OF BYTE

Press "V" to display a byte's conversion values at the bottom right of the screen. Do what you want with these values. If you don't understand them, ask around.

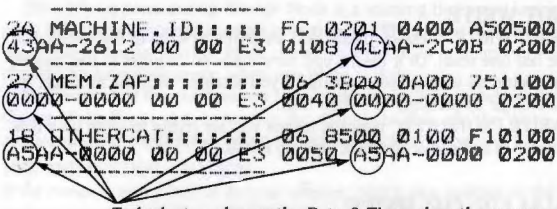
TOKEN: The byte's token value, if below 235 ($EB), is shown first. Tokens are Applesoft's way of storing reserved words in one byte. For example, the command "PRINT" is stored as 186 ($BA). "REM" is 178 ($B2). Values below 128 ($80) are simply ASCII characters. Even though a token conversion may not be appropriate to the block or sector you are viewing, you'll get one anyway when you press "V".

BITS: The next line shows the binary value of the current byte. It is divided into two four-bit nibbles for easy viewing. You might use this feature when analyzing the date/time bytes.

2BYT: The last line gives the two-byte decimal and hex values for the cursor byte combined with the byte that follows it. If the cursor is on a 1 ($01) followed by a 2 ($02), that two-byte value is interpreted as 513 ($0201—notice the reverse order).

Y: YEAR (Date and Time) (ProDOS disks only)

BYTEZAP.PRO lets you change (or add) the date and time that will appear next to the file names in your ProDOS catalogs.



To look at or change the Date & Time, place the cursor on the FIRST BYTE of any set of date/time bytes. Press "Y" once to read the date. Press "Y" again to change it.

SEEING THE DATE AND TIME

With a ProDOS directory block on the screen, move the cursor to the FIRST BYTE of a valid ProDOS date (CAT format is advised) and press "Y". The date and time will be shown in human-readable format at the bottom right of the screen. An uninterpretable set of numbers (like "0000-0000" or "FFFF-FFFF") is usually interpreted as "NO DATE".

After viewing the date, press any key but "Y" to go back to Cursor Mode.

CHANGING THE DATE AND TIME

To change the date and time, press "Y" two times from Cursor Mode, and type in a new date in human form (for example, 01-OCT-80 12:00). When you type the last minutes digit, the new date and time will be converted into ProDOS's cryptic format and changed in memory.

Press "Y" again if you want to proofread your new date and time. Press control-W to write your change to disk.

HOW DATE AND TIME ARE STORED

ProDOS's authors could have simply stored the date and time in five bytes—Day-Month-Year-Hour-Minute. But NO-OOO-OO; they had to *save one big byte* to frustrate us humans. For those of you who care, here is the totally ridiculous (if you ask us) method they used:

```
          7 6 5 4 3 2 1 0   7 6 5 4 3 2 1 0
Date Bits: Y Y Y Y Y Y Y M   M M M D D D D D  (bytes in reverse order)
Time Bits: H H H H H H H H   M M M M M M M M  (bytes in reverse order)
```

EXAMPLE: 25-JAN-42 06:42

Appears in CAT format as "3954-1506"

or Date: 39 54, Time: 15 06

Swap the order of each pair of bytes to make: 54 39 and 06 15

Date Bytes: $54 $39

Date Bits: 01010100 00111001

or: Year 0101010, Month 0001, Day 11001

or: Year 42, Month 1, Day 25

Time Bytes: $06 $15

Time Bits: 00000110 00010101

or: Hour 00000110, Minute 00010101

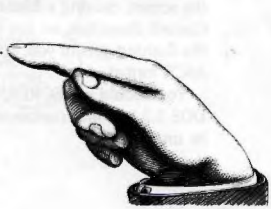
or: Hour 6, Minute 21

CHANGING THE DATE AND TIME IN MEMORY

Even if you don't have clock/calendar hardware in your Apple, you can Poke the date and time stored in memory (or better yet, Run the DATE.SET program on the Pro-Byter disk). The date is at $BF90-BF91, and the time hides at $BF92-BF93. ProDOS peeks there to get the date and time that will appear in your catalogs when you save a file to disk. If you don't have a clock and these values haven't been changed from zeros, saved files will report "NO DATE" in your catalogs.

EXAMPLE: 31-DEC-99 23:59 is stored as $BF90: 9F C7 3B 17. The bytes are in the same order in memory as you will find them in the directory when using BYTEZAP.PRO.

If you don't understand this page, you're not alone.



F: FORMAT SCREEN (also see Space Bar, next page)

There are five different ways you can look at a disk sector or block-half. From Cursor Mode, press "F" followed by "H", "D", "A", "S" or "C" to give each format a try.

H: HEX FORMAT

All bytes are displayed in hex. A column of hex byte numbers, representing the distance from byte $00, appears at the left of the screen.

D: DECIMAL FORMAT

All bytes are displayed in decimal form. A column of decimal byte numbers, representing the distance from byte 000, appears at the left of the screen. (The more you get into looking at bytes, the more useless decimal becomes.)

A: ASCII FORMAT

All bytes are displayed as normal ASCII characters with a corresponding set of hex values. Every control-character is represented by a colon (":") to make text messages easier to find. Press the Space Bar to compare with SCREEN format, and see what we mean.

You will see two cursors so you can tell which hex value corresponds to each text character. If you press "B" to change bytes while in ASCII format, the hex cursor will go away.

You should use ASCII format (instead of SCREEN) when looking through ProDOS text files that contain lower case characters. That's because ProDOS lower-case characters are usually stored with their high-bit off. SCREEN format would display most ProDOS lower-case as inverse and flashing garbage.

The Search function will automatically put you in ASCII format unless you have SCREEN format selected at the time. When a match is found, you can toggle between formats by pressing the Space Bar.

S: SCREEN FORMAT

SCREEN format is similar to ASCII format, but every character is "poked" to the screen, causing a hard-on-the-eyes (but often informative) flashing display. Control characters are not disguised as colons as they are in ASCII format. If the flashing gets to you, you can usually do just as well by switching over to ASCII format.

You should use SCREEN format (instead of ASCII) when looking through DOS 3.3 text files containing lower case characters. Otherwise lower case will be unreadable.

C: CAT FORMAT

CAT format is a combination format to be used with DOS 3.3 directory sectors and ProDOS directory blocks. You get readable characters where file names should be located and hex values elsewhere. See pages 30-32 for a detailed breakdown of what each byte means.

In ProDOS you get a combination HEX/ASCII display because ProDOS file names are high-bit off). In DOS 3.3 you get a HEX/SCREEN display (because DOS 3.3 file names are high-bit on).

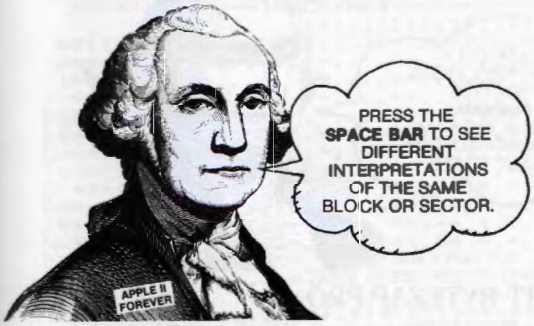
You can select CAT format for any block or sector, but it will only make sense—and be readable—with *directory* blocks or sectors. The DOS 3.3 directory is usually found on track 17 ($11), sectors 15-1 ($0F-$01). The ProDOS main directory is usually at blocks 2-5. ProDOS subdirectories can be found almost anywhere.

AUTOMATIC CAT SWITCHING

BYTEZAP.PRO will do its best to automatically switch you in and out of CAT format when it thinks it should. This feature is shut off by the "F" command, and turned back on by "D" or control-D.

Space Bar: TOGGLE SCREEN-FORMAT

Pressing the SPACE BAR from Cursor Mode will toggle the screen between the five screen formats (previous page). The Space Bar command also works in Search Mode when a word is found—very handy for quickly seeing different interpretations of the same set of bytes.



P: PRINT THE SCREEN (Problems printing? See page 77.)

From Cursor Mode or Map Mode, press "P" to print the text screen on your printer. You can quit printing at any time by pressing ESC.

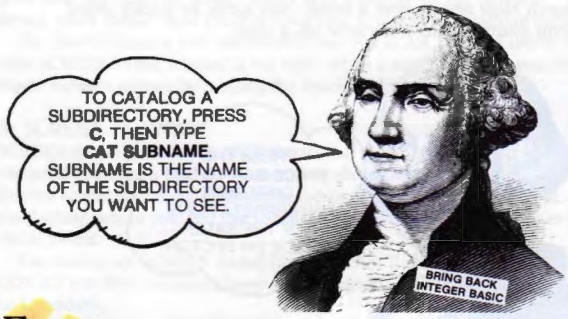
If you have no printer connected and you try to print, your system will "hang" and you'll have to re-run the program. Exception: Apple IIc's "pretend" like they're printing, even with no printer connected. You can press ESC to exit this imaginary printout.

C: CATALOG DISK (AND ProDOS COMMANDS)

To see a catalog of the current disk, press "C" from Cursor Mode. If a ProDOS disk is in the current drive, press RETURN for a 40-column "CAT" of the main directory, or type "CATALOG" for an 80-column catalog (80-column Apple IIe and IIc only). If the disk involved isn't DOS 3.3 or ProDOS, you'll get a "Disk/Drive Error" message, or a catalog full of garbage.

OTHER "C" COMMANDS

Under ProDOS, after pressing "C" from Cursor Mode, you can enter any ProDOS command for execution. You can Lock, Unlock and Delete files (but why?), Catalog a subdirectory or another drive (for example, type "CAT,D2" or "CAT SUBNAME"), change the current prefix, etc. If you're totally fed up with BYTEZAP.PRO and your Apple, type "BYE". Illegal or non-ProDOS commands will get you a well-deserved "Huh?" error message.



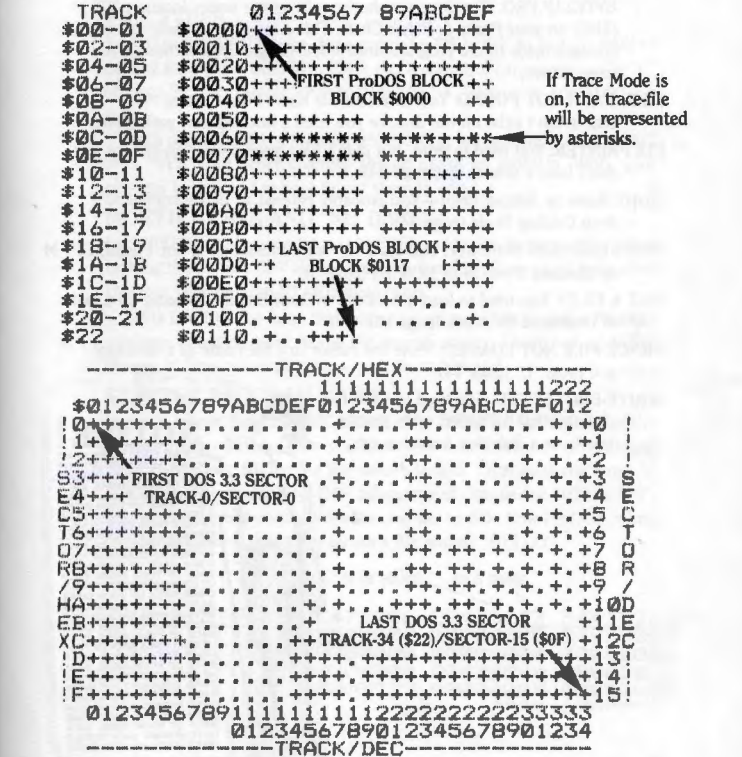
ESC: QUIT BYTEZAP.PRO

Press ESC to exit BYTEZAP.PRO from Cursor Mode. If you exit by mistake, you can reinsert the Pro-Byter disk and type "RUN".

M: MAP THE DISK

Press "M" for a map of the current ProDOS or DOS 3.3 disk. Unused blocks or sectors will be marked with a dot and the used ones will be marked with a plus-sign. If Trace Mode is on, the trace-file will be displayed as inverse asterisks.

If you map an old disk that has seen lots of deleting and saving, you will notice how "scrambled" the map gets. Other disks will look much more organized. It really doesn't make much difference; ProDOS and DOS 3.3 can handle either situation. Scrambled files do take a bit longer to load.



BYTEZAP.PRO Error Messages

DELETED: You tried to load a deleted trace-file, or your cursor wasn't positioned correctly when you pressed "L" (see "Not a File?" below). Deleted files cannot be traced.

DISK/DRIVE ERROR: Same as I/O Error. Perhaps your drive door is open or there is no disk in your drive. Or the disk is bad, or unformatted. Or you're trying to read a copy-protected or non-standard disk.

ERROR-#, LINE-#: List the indicated line number of the Applesoft program BYTEZAP.PRO. The Error number should appear under location 222 ($DE) on your Peeks & Pokes Chart. This error is most-likely caused by changes made to the program since it left Beagle Bros. Although you never know...

FILE/PATH NOT FOUND: You entered a file name from Catalog Mode that doesn't exist on the disk, or you used a non-existent path name.

FIX PRINTER: You tried to print, but didn't have your printer on-line, or you don't have a printer (see page 24).

HUH?: Same as Syntax Error—You probably entered an illegal command from Catalog Mode (page 24).

NON-STANDARD FORMAT: You used "D" with a disk that is not Pascal, CP/M or standard ProDOS or DOS 3.3 (page 5).

NOT A FILE?: You tried to load a trace-file without the cursor being on a real or undeleted file name (page 14).

TRACE-FILE NOT LOADED: Move the cursor to a file name in a directory and press "L" (page 14).

WRITE-PROTECTED: You tried to write to a disk that has a covered write-protect notch, or a disk that has no notch.

Using BYTEZAP.PRO to Change BASIC.SYSTEM

BE CAREFUL, AND WORK WITH BACKED-UP DISKS ONLY.

With BYTEZAP.PRO running, Load the trace-file "BASIC.SYSTEM". Then search for the characters you are going to change. After making changes, write them to disk, exit BYTEZAP.PRO and re-boot (or simply type "-BASIC.SYSTEM") to see your changes in effect. You can rename a new version "ANYTHING.SYSTEM"; booting ProDOS will cause the *first* ".SYSTEM" file on the disk to be loaded.

FILE TYPES

Switch to SCREEN format. Now search for "BINDIR" to find the table of catalog file types—BIN, DIR, CMD, etc. A complete list appears under "F" on page 30. For example, change "BAS" to " A " (space "A" space) or your initials.

Each name should be *high-bit ON* and three letters long. You shouldn't alter "SYS", but other changes should cause no problem. Write your changes to disk and re-boot to put them in effect.

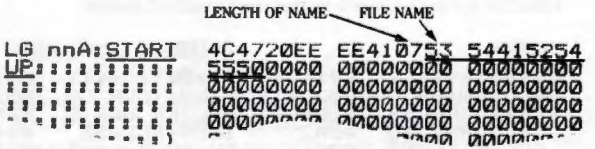
MONTH NAMES AND "NO DATE"

In SCREEN format, search for "JANFEB" and make changes. How about making the month MAY always come up flashing? Easy—just make your changes with the high-bit *off* instead of on.

"(NO DATE)", the label that comes up all too often in many ProDOS catalogs, is hiding right below the month names. Don't just sit there; change it! How about all spaces, so it doesn't show?

STARTUP PROGRAM NAME

In ASCII format, search BASIC.SYSTEM for "STARTUP", the name of the program that will run when you boot. The byte immediately preceding these characters ($07) is the file name's length. To change the name to "HELLO", choose *high-bit OFF* (always) and change the "07" to "05", followed by the new name. Write the changes to disk. Now ProDOS, when first booted, will attempt to execute a file named "HELLO".



NO-STARTUP MESSAGE

In ASCII format, search for "PRODOS BASIC". This is part of the message that appears when no greeting program (STARTUP) is found on the disk. It doesn't tell you much (the final ".1" is missing for version 1.1.1). Choose *high-bit ON* and go to work. WARNING: Don't change any bytes that aren't ASCII characters. Boot the disk to try out your changes.

COMMAND TABLE

ProDOS commands can be found by searching BASIC.SYSTEM for the word "CATALOG". Use *high-bit OFF* here. Notice that the command table is slightly "compressed", with many commands overlapping others. So watch out—if you change one command, you might be changing another as well. You can't use spaces or numbers in a command, and you can't change a command's length.

Changing PRODOS itself

The PRODOS file is not quite as *interesting* to look at, but give it a try. With BYTEZAP.PRO running, Load the file "PRODOS" as a trace-file from the main directory.

BOOT-UP MESSAGE

Search for "APPLE //". That's part of the message that comes on the screen when you boot ProDOS. The message continues when you advance to the next block-half.

BYE-BYE "BYE" MESSAGE

When you type "BYE", ProDOS gives you a strange message, especially if you're new to these parts. Namely:

ENTER PREFIX (PRESS "RETURN" TO ACCEPT)

Why not change this to:

YOUR COMPUTER HAS DIED--PLEASE RE-BOOT.

CAREFUL you don't change bytes that aren't actually characters!

BEFORE

```
E%LQ:ENTER P   85254
REFIX (PRESS   D2C5(
 "RETURN" TO   A0A2'
  ACCEPT):ENT  A0C1
ER PATHNAME    C5D2
```

AFTER

```
E%LQ:YOUR CO   85254(
MPUTER HAS D   4D505'
IED--PLEASE    49454
RE-BOOT.:ENT   5245?
ER PATHNAME    C5D2/
```

Making Changes to Disk Directories

Pages 30-32 contain breakdowns of typical ProDOS and DOS 3.3 disk directories, as seen with BYTEZAP.PRO. Each byte means something different—it can be a file name character, a piece of information about that file, or a "pointer" to some other part of the disk. Watch out—changing the non-file name bytes can mean disaster. Even changing a file name can cause problems. Since you're only working with backed-up disks, you can have some fun experimenting.

FILE NAME CHANGES

File names can be changed to contain "illegal" characters. This is usually done to make decorative catalog headings or separators between file names. For example, a file name can be a string of hyphens or asterisks.

ProDOS FILE NAME CHANGES

When changing ProDOS file names, always set the high bit *off* if you want to be able to access the file (in which case you could simply use the RENAME command). And don't forget to change the length nibble (see "D", next page)—Otherwise your file name may appear as fewer or more characters than you intended.

Never put illegal characters in a directory or subdirectory header. If you do, you might make the disk unreadable and you might murder ProDOS.

You can't put flashing and inverse characters in ProDOS file names. You can insert backspaces, but doing so usually makes a big mess.

DOS 3.3 FILE NAME CHANGES

Decorative DOS 3.3 file names may contain flashing and inverse characters. Set the high bit *on* for normal characters and *off* for flashing ASCII 64-95 and inverse ASCII 32-63. Other values must be entered by switching to HEX or DEC format and entering the values directly, according to the chart on page 79.

You can precede a DOS 3.3 file name by seven backspaces, (136's or $88's) so that it will appear to the far left of the catalog display, covering the file type and size information.

REPAIRING DISKS

You can *sometimes* repair minor damage on a disk that won't catalog by adjusting the pointers at the beginning of one or more blocks or sectors (see "A" and "B" on pages 30 and 32). Most crashed disks, however, have one or more tracks that can't even be read with BYTEZAP.PRO.

ProDOS DIRECTORY ANALYSIS

Here is a sample 80-column Catalog from a disk called "TEST.DISK.A":

```
/TEST.DISK.A

 NAME            TYPE  BLOCKS  MODIFIED          CREATED           ENDFILE SUBTYPE

*PRODOS          SYS       30  8-JAN-85  5:45  12-DEC-84 12:35     14848
*BASIC.SYSTEM    SYS       21  9-JAN-85  6:42  12-DEC-84 12:50     10240
 STARTUP         BAS        1  1-FEB-85 23:00  13-DEC-84 10:00        80
*NUMBER.GAME     BAS        4  25-DEC-85  0:00  25-DEC-85  0:00      1162
 BEAGLE.PIC      BIN       17  3-MAR-85 12:04  20-DEC-84  9:07      8192 A=$2000
 ADDRESS.FILE    TXT        1  <NO DATE>       <NO DATE>             199 R=    0
 MACH.LANGUAGE   BIN        1  25-DEC-85  0:00  25-DEC-85  0:00       41 A=$0300
 NAMES           DIR        1  25-DEC-85  0:00  25-DEC-85  0:00      512
*MEMSCAN         BAS        5  6-MAY-86  0:33  15-DEC-84  9:09     1603

BLOCKS FREE:  192     BLOCKS USED:  88     TOTAL BLOCKS:  280
```

Using BYTEZAP.PRO, we pressed "D" to see the catalog (or "directory") data starting at block 2. The printouts on the next page are interpreted here:

A PREVIOUS DIRECTORY BLOCK: *In this sample, "00 00" means no previous block.*

B NEXT DIRECTORY BLOCK: *The "03 00" in our sample means block $0003 is next.*

C STORAGE TYPE: *The first nibble (half a byte) tells what each entry is:*

$1_: Seedling file *(1 block)* $0_: Deleted file $E_: Subdirectory header
$2_: Sapling file *(3-257 blocks)* $D_: Subdirectory $F_: Volume Directory header
$3_: Tree file *(more than 257 blocks)*

D NAME LENGTH: *The second nibble is the length, $0-$F (1-15) of the name that follows.*

E DIRECTORY NAME OR FILE NAME: *The colons represent zeros—unused bytes.*

F FILE TYPE

$00: (No type)	$19: ADB—Database*	$FA: INT—Integer BASIC program
$01: (Damaged)	$1A: AWP—Word processor*	$FB: IVR—Integer BASIC variables
$04: TXT—Text	$1B: ASP—Spreadsheet*	$FC: BAS—Applesoft program
$06: BIN—Binary	$EF: PAS—Pascal	$FD: VAR—Applesoft variables
$0F: DIR—Subdirectory	$F0: CMD—Command file	$FE: REL—Relocatable file
	$F1-$F8: (User defined)	$FF: SYS—ProDOS system file

* *AppleWorks files*

G KEY BLOCK POINTER: *The file's first (or only) block. The sample points to block $0041.*

H BLOCKS USED: *The file's size in blocks. The example says $0011 blocks.*

I LENGTH OF FILE: *3 bytes indicating the file's size. The sample says $002000 bytes.*

J CREATED DATE: *(See page 21.) Zeros here would mean No Date.*

K CREATION VERSION: *A zero means ProDOS 1.1.1 or earlier created the file.*

L ACCESS VERSION: *A zero means ProDOS 1.1.1 or earlier can access the file.*

M LOCKED STATUS: *$21=locked, $E3=unlocked. See page 72 for more possibilities.*

N ADDRESS/LENGTH: *A text file's length, or non-text file's starting address.*

O MODIFIED DATE: *When the file was last changed (see page 21). Zeros mean No Date.*

P HEADER POINTER: *The block where this directory starts. The example says block $0002.*

Q-U apply to directory name entries only (not file or subdirectory entries):

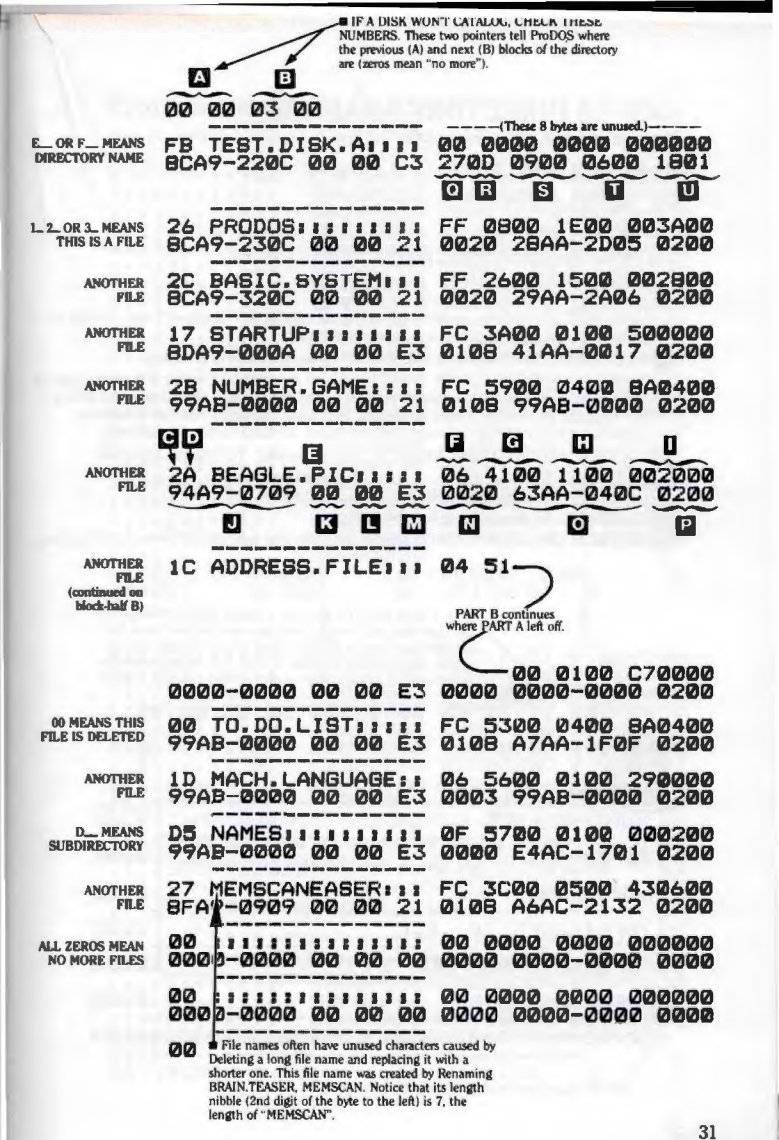
Q BYTES PER ENTRY: *Always $27 (decimal 39).*

R ENTRIES PER BLOCK: *Always $0D (decimal 13).*

S NUMBER OF UNDELETED FILES IN THIS DIRECTORY: *Includes subdirectory names.*

T BIT MAP BLOCK: *Block $0006. Or if in a subdirectory, the block of its source directory.*

U BLOCKS PER DISK: *Almost always $0118 (decimal 280).*


■ IF A DISK WON'T CATALOG, CHECK THESE NUMBERS. These two pointers tell ProDOS where the previous (A) and next (B) blocks of the directory are (zeros mean "no more").
A
B
00 00 03 00
(These 8 bytes are unused.)
E_ OR F_ MEANS DIRECTORY NAME
FB TEST.DISK.A
8CA9-220C 00 00 C3
00 0000 0000 000000
270D 0900 0600 1801
Q R S T U
1_ 2_ OR 3_ MEANS THIS IS A FILE
26 PRODOS
8CA9-230C 00 00 21
FF 0800 1E00 003A00
0020 28AA-2D05 0200
ANOTHER FILE
2C BASIC.SYSTEM
8CA9-320C 00 00 21
FF 2600 1500 002800
0020 29AA-2A06 0200
ANOTHER FILE
17 STARTUP
8DA9-000A 00 00 E3
FC 3A00 0100 500000
0108 41AA-0017 0200
ANOTHER FILE
2B NUMBER.GAME
99AB-0000 00 00 21
FC 5900 0400 8A0400
0108 99AB-0000 0200
C D
E
F G H I
ANOTHER FILE
2A BEAGLE.PIC
94A9-0709 00 00 E3
06 4100 1100 002000
0020 63AA-040C 0200
J K L M
N
O
P
ANOTHER FILE (continued on block-half B)
1C ADDRESS.FILE
04 51
PART B continues where PART A left off.
00 0100 C70000
0000-0000 00 00 E3
0000 0000-0000 0200
00 MEANS THIS FILE IS DELETED
00 TO.DO.LIST
99AB-0000 00 00 E3
FC 5300 0400 8A0400
0108 A7AA-1F0F 0200
ANOTHER FILE
1D MACH.LANGUAGE
99AB-0000 00 00 E3
06 5600 0100 290000
0003 99AB-0000 0200
D_ MEANS SUBDIRECTORY
D5 NAMES
99AB-0000 00 00 E3
0F 5700 0100 000200
0000 E4AC-1701 0200
ANOTHER FILE
27 MEMSCANEASER
8FA?-0909 00 00 21
FC 3C00 0500 430600
0108 A6AC-2132 0200
ALL ZEROS MEAN NO MORE FILES
00
0000-0000 00 00 00
00 0000 0000 000000
0000 0000-0000 0000
00
0000-0000 00 00 00
00 0000 0000 000000
0000 0000-0000 0000
00
■ File names often have unused characters caused by Deleting a long file name and replacing it with a shorter one. This file name was created by Renaming BRAIN.TEASER, MEMSCAN. Notice that its length nibble (2nd digit of the byte to the left) is 7, the length of "MEMSCAN".

DOS 3.3 DIRECTORY ANALYSIS

Here is a sample Catalog from a DOS 3.3 disk:

```
DISK VOLUME 254

*A 003 HELLO
*A 040 PIZZA
 T 003 NAME FILE
 B 034 SURPRISE!.PIC
 A 002 NAIL FILE
```

Using BYTEZAP.PRO, we pressed "D" to see the catalog (or "directory") data starting at track-17, sector-15. The printout is interpreted here:

A NEXT DIRECTORY TRACK & SECTOR: *All zeros mean end of directory.*

B TRACK-SECTOR-LIST TRACK & SECTOR: *Look on track $12, sector $0F to see a list of this sample file's tracks & sectors. If you delete this file, the $12 will be replaced with a $FF, and the $12 will be moved to the 30th file name character. You can sometimes undelete DOS 3.3 files by reversing this procedure. Re-Save the file immediately.*

C LOCKED-UNLOCKED INDICATOR: *"8_" means locked, "0_" means unlocked.*

D FILE TYPE: $_0: Text file $_2: Applesoft file $_1: Integer BASIC file $_4: Binary file

E FILE NAME: *30 characters or less, high-bit on.*

F FILE SIZE: *Our sample is "0003" sectors. Only the first byte (0-255) prints in the catalog.*

■ IF A DISK WON'T CATALOG, CHECK THESE NUMBERS on each directory sector. They point to the next track (1st number) and sector (2nd number) of the catalog. "00 00" would mean end of catalog. A DOS 3.3 catalog normally occupies track 17 ($11) from sector 15 ($0F) through 1 ($01).

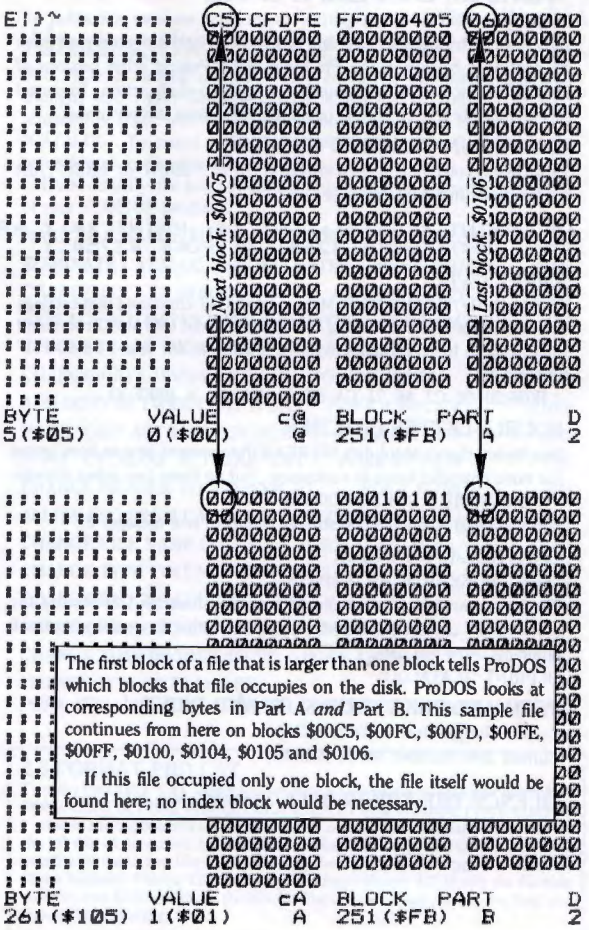
```
                A
00 11 0E 00 00 00 00 00 00 00 00
   ======

T S T F                                   S
R E Y I                                   I
A B C D                                   Z F
120F82HELLO E                             0300
130F82PIZZA                               2800
160F00NAME FILE                           0300
170F04SURPRISE!.PIC                       2200
FF0F02NEIGHBORHOOD INFO                   Z0200
1B0F02NAIL FILE                           0200
000000                                    0000
```

■ "00" means end of directory (see tip on page 65)

ProDOS INDEX BLOCKS

Here is a printout of Block #1—the "index block"—of a 9-block ProDOS file:



Changing BASIC.SYSTEM without BYTEZAP.PRO

WARNING: The following experiments are *direct memory pokes* that will only work with PRODOS 1.1.1 and BASIC.SYSTEM in memory (put there by booting a backup of the Pro-Byter disk). Poking other versions of ProDOS will sooner or later cause your Apple to malfunction. To cancel any changes or restore normal operating conditions, simply reboot.

SMART-RUN COMMAND CHANGER

ProDOS lets you type a hyphen instead of "RUN", "BRUN" or "EXEC". Just for fun, poke in a new character:

```
10  PRINT "KEY:";: GET X$: PRINT X$:X =ASC
    (X$): POKE 42641,X: POKE 43754,X
```

SMART CAT

80-column IIe/IIc only: Wow! Make a smart CAT command that catalogs in 80-columns if you're looking at 80, and a smart CATALOG command that switches to 80-columns if you're looking at 40! Type "CALL-151" followed by:

B036:20 00 C3 A6 21 CA 8E N B923:36 N B8EF:39

DOUBLE-LETTER CATCHER

Now lookit, there's absolutely NO REASON to prevent anyone from typing the same character twice in succession. But do these two pokes anyway:

POKE 39519,0: POKE 39530,135

This perverted poking can do strange things to eror mesages to.

CONTROL-D CHANGER

D$="Z": POKE 40359,ASC(D$)+128

The two commands above make "Z" (or any character D$) become the official DOS character (instead of control-D) that is used in Applesoft programs. Now you can:

10 PRINT "ZCATALOG"

PEEK AND POKE YOUR SLOT AND DRIVE

Current slot number: 48700 ($BE3C)

Current drive number: 48701 ($BE3D)

SILENCE THE ERROR BELL *(ProDOS errors only)*

POKE 40857,160

ProDOS 1.1.1 ONLY!

Altering the Catalog Format

The ProDOS catalog display is very complete, informative and well thought out, right? So let's change it beyond recognition! At least temporarily. Type one of the pokes shown below and then "CAT" or "CATALOG". To cancel a change, substitute the value on the right (in parentheses) for the one after the comma.

If you actually like some of these changes, you can make the Pokes part of your STARTUP program.

MAKE FILE NAMES FLASH OR INVERSE: *POKE 42198,0* *(128)*

ELIMINATE CATALOG (not CAT) DATES: *POKE 42316,96* *(162)*

CHANGE LOCKED SYMBOL TO "-": *POKE 42344,173* *(170)*

REPLACE ALL DATES WITH "NO DATE": *POKE 42402,0* *(18)*

ELIMINATE ALL "NO DATES": *POKE 42403,96* *(152)*

CHANGE DATE DIVIDER TO "/": *POKE 42490,47* *(45)*

ADD DOTS BETWEEN CAT ITEMS: *POKE 42605,174* *(160)*

MAKE "CAT" ACT LIKE "CATALOG": *POKE 45111,79* *(39)*

MAKE "CATALOG" ACT LIKE "CAT": *POKE 45115,39* *(79)*

CHANGE HEADER TO "COPYRIGHT...": *POKE 45158,0* *(16)*

ALLOW ESC FROM CATALOG: *POKE 45201,155* *(131)*

CHANGE FILE NAME COLUMN: *POKE 42200, #* *(1)*

CHANGE TYPE COLUMN: *POKE 42238, #* *(18)*

CHANGE BLOCKS COLUMN: *POKE 42324, #* *(27)*

CHANGE FILE LOCKED COLUMN: *POKE 42346, #* *(1)*

CHANGE TIME DIVIDER TO "|": *POKE 42439, 252* *(186)*

CHANGE "ENDFILE" to "LEN":
POKE 47732,238: POKE 47733,238: POKE 47734,116: POKE 47735,158

3.3-FORMAT PRO CAT

To make ProDOS's "CAT" command produce a 3.3-looking catalog, type five pokes: POKE 42200,10: POKE 42238,2: POKE 42324,7: POKE 42348,96: POKE 45159,44

One of the big benefits of this new format is that you can Escape the cursor up the left side of the screen, type a command like "RENAME" and then cursor-trace over the file name, just like in 3.3. Normally there's only room to type one character (like a hyphen). Poking 42200 sets the file name column; 42238 sets the file-type column, and 42324 positions the blocks. The last two pokes prevent the Date and Header from printing.

TYPE (Works ONLY with ProDOS 1.1.1)

TYPE adds a "TYPE" command to ProDOS. This new command is used to dump text files to the screen or printer. Type "-TYPE" to install the command. To use it, type "TYPE FILENAME", where FILENAME is any text (TXT) file on the current disk. Actually, almost *any* type of file can be used, but text files are the most readable.

Press any key to halt a listing. Continue with another keypress or press ESC to quit.

TYPE LIMITATIONS

TYPE takes up no extra space in memory...sort of. It does, however, supersede the POSITION command. If you want to use POSITION, you'll have to reboot to remove TYPE. The POSITION command is really only "supposed to be" used with DOS 3.3 files that have been converted to ProDOS.

If you try to install TYPE with a version of ProDOS other than 1.1.1, you'll end up with a big mess—Don't even *think* about trying it.

CATA.LOG

CATA.LOG's main purpose in life is to list all of a ProDOS disk's files, including those in subdirectories, on your printer. Type "-CATA.LOG" to see it in action.

If you select YES for the "Sort/Merge" option, all file names on the disk will be alphabetized—the subdirectory file names will be mixed in with the main directory names. If you select NO, subdirectory file names will be listed separately, unsorted.

CATA.LOG will only make a sorted *printout;* it won't change the disk itself. Use Beagle Bros' SORTCAT program (on our Fatcat disk) to alphabetize ProDOS directories and subdirectories on the disk.

DATE.SET

If you don't have a clock in your Apple, RUN DATE.SET. It lets you type in the current date and time so they will appear next to files that are saved on disk, replacing "(NO DATE)". The date and time will remain the same until you RUN DATE.SET again.

DATE.SET can be your STARTUP program, so you are asked the date every time you boot. Or you can make an existing STARTUP program RUN DATE.SET.

LIST.LOCK

Running the LIST.LOCK program makes subsequent "LIST" commands produce a "File Locked" message. Type "-LIST.LOCK" to poke the necessary changes into ProDOS and the page 3 ($300) area of memory. You can make this program part of your STARTUP program.

Don't tell anyone, but you can get around this protection scheme by typing ":LIST" or "L IST". That's because the "command handler" on page 3 only looks at the first four characters typed.

You can cancel LIST.LOCK's effect by re-booting, or by typing two pokes: POKE 48647,158: POKE 48648,190.

WHAT'S GOING ON HERE?

(The following discussion applies to immediate mode commands only, not commands that are executed from a running program.)

When you type something and press Return, ProDOS handles it with the following procedure:

1. ProDOS first scans its own vocabulary list to see if you typed a *ProDOS* command (CATALOG, SAVE, LOAD, etc.). If you did, your command is executed. If you didn't, on to step 2.

2. ProDOS then goes to the "External Command Vector" at $BE06-BE08. This location normally tells ProDOS to jump to $BE9E, which does absolutely nothing (RTS).

3. ProDOS scans Applesoft's vocabulary list to see if you typed an *Applesoft* command (LIST, PRINT, GOTO, etc.). If you did, your command is executed. If you didn't, you get a ?Syntax Error.

The first two pokes in the LIST.LOCK program make the External Command Vector (see step 2) jump to $300. The rest of the pokes write a machine language routine at $300 that checks to see if "LIST" was typed. If it was, a "File Locked" error code 10 ($0A) is loaded into the accumulator and printed. If something other than the four characters L-I-S-T were typed, control is sent back to $BE9E (see step 3).

APPLESOFT.LOCK

You could eliminate *all* typed Applesoft commands by making the External Command Vector point to location $3D0. Just type two pokes:

POKE 48647,208: POKE 48648,3.

CALL.BASIC

CALL.BASIC is a program which converts one-line Applesoft programs into routines which can be executed by a single CALL statement. The routines can be installed anywhere in memory (within reason).

CONVERTING YOUR ONE-LINER:

- Type "-CALL.BASIC" to get CALL.BASIC running.
- Press "E" to Exit.
- Enter your one-liner as Line 1 (see One-Liner Rules below).
- Test your one-liner by typing "RUN 1". Make sure it works!
- Type "RUN" to get CALL.BASIC running again.
- Press "A" to Assemble your one-liner.
- When asked, type a file name for your routine so it can be saved on disk.
- If you want to edit Line 1 later, Save CALL.BASIC under a new name.

USING YOUR ROUTINE:

Let's say you have a routine on disk called "NAME" that prints your name. Let's say you want this routine to occupy memory starting at location 25000. (If you don't know about memory allocation, use 25000 for now.)

- Type "BRUN NAME,A25000" to load your routine.
- Type "CALL 25000" to print your name.

From now on, you only need to CALL 25000 to print your name. You can even type "NEW", and CALL 25000 will still work. You don't need to BRUN NAME again unless your routine gets overwritten.

The following program uses your name routine:

```
10 NAME=25000: PRINT CHR$(4)"BRUN NAME,A";NAME
30 PRINT "MY NAME THREE TIMES:"
40 CALL NAME: CALL NAME: CALL NAME
```

ONE-LINER RULES

- A one-liner must be a "self-contained" program than runs on its own.
- No GOTO's, GOSUB's or ProDOS commands are allowed.
- Your one-liner cannot be called from within a for-next loop. It may *contain* for-next loops, however.

CALL.BASIC ADVANTAGES

- Many routines can be stored on disk until you need them.
- Many routines can be hidden in memory (at different addresses), taking up no Applesoft program space. All routines are relocatable.
- You can call a routine by name instead of a number. For example, you can use "CALL BOXDRAW" instead of "GOSUB 500".
- You can "pass" variables into a routine. For example, "X=10: Y=50: CALL BOX" could draw a box at location X, Y.

CALL.BASIC LIMITATIONS

- Routines operate at normal Applesoft speed; this is not a compiler.
- Routines take up more space than they normally would.

SCRAMBLED.BASIC*

SCRAMBLED.BASIC is a program that demonstrates how you can make almost any Applesoft program command act like almost any other. Type "-SCRAMBLED.BASIC" for a quick demo. Type "LIST" to learn more.

The authors of ProDOS didn't have this feature in mind, we're sure, and *why* you'd want to use it, we're not sure, but here's what you do:

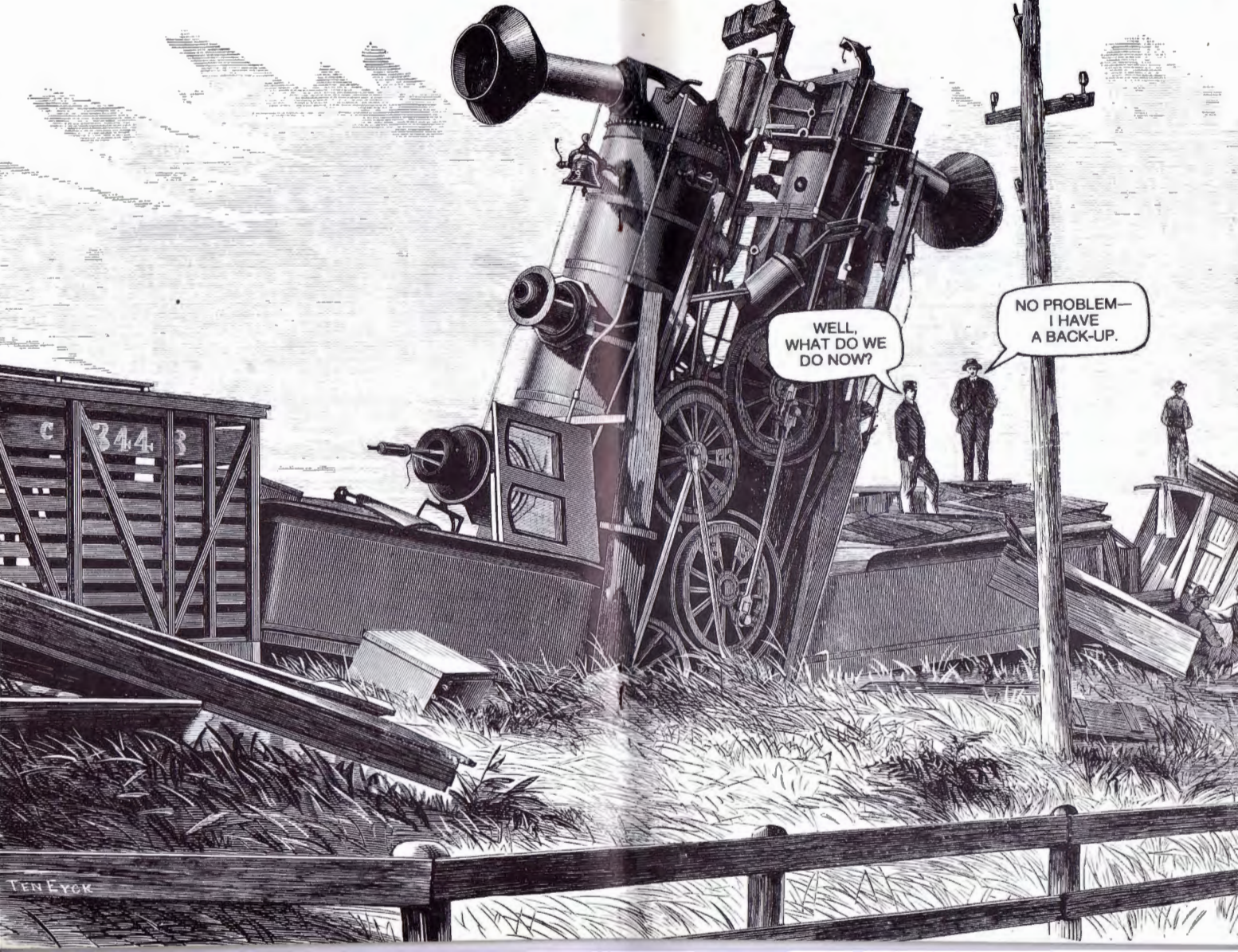
To make program command A act like program command B, poke command B's value from the table below into command A's address.

For example, to make HOME work like GR, **POKE 47152, 136.**

Address	*Value*	*Command*	*Address*	*Value*	*Command*	*Address*	*Value*	*Command*
47129	128	END	47151	150	HTAB	47173	172	RUN
47130	129	FOR	47152	151	HOME	47174	173	IF
47131	130	NEXT	47153	152	ROT=	47175	174	RESTORE
47132	131	DATA	47154	153	SCALE=	47176	175	&
47133	132	INPUT	47155	154	SHLOAD	47177	176	GOSUB
47134	133	DEL	47156	155	TRACE	47178	177	RETURN
47135	134	DIM	47157	156	NOTRACE	47179	178	REM
47136	135	READ	47158	157	NORMAL	47180	179	STOP
47137	136	GR	47159	158	INVERSE	47181	180	ON
47138	137	TEXT	47160	159	FLASH	47182	181	WAIT
47139	138	PR#	47161	160	COLOR=	47183	182	LOAD
47140	139	IN#	47162	161	POP	47184	183	SAVE
47141	140	CALL	47163	162	VTAB	47185	184	DEF
47142	141	PLOT	47164	163	HIMEM:	47186	185	POKE
47143	142	HLIN	47165	164	LOMEM:	47187	186	PRINT
47144	143	VLIN	47166	165	ONERR	47188	187	CONT
47145	144	HGR2	47167	166	RESUME	47189	188	LIST
47146	145	HGR	47168	167	RECALL	47190	189	CLEAR
47147	146	HCOLOR=	47169	168	STORE	47191	190	GET
47148	147	HPLOT	47170	169	SPEED=	47192	191	NEW
47149	148	DRAW	47171	170	LET			
47150	149	XDRAW	47172	171	GOTO			

*This technique is a semi-unpredictable can of worms that should perhaps be avoided by the non-adventurer. The SCRAMBLED.BASIC demo program simply touches on the possibilities. You take it from there. Also:

- ProDOS VERSION 1.1.1 only
- Deferred mode only (commands inside Applesoft programs)
- Strange things may happen. Good luck.


WELL, WHAT DO WE DO NOW?
NO PROBLEM— I HAVE A BACK-UP.
C 3443
TEN EYCK

QSORT

QSORT is a quick machine language sorter for Applesoft arrays. It is one of the fastest Apple word alphabetizers around; it sorts 100 words in around one second. 2000 words take about 11 seconds. If you know of a faster sorter, use it, and let us know where we can buy a copy.

INSTALLING QSORT

Before you can use QSORT, you have to get it in memory, either by typing "-QSORT" or by putting PRINT CHR$(4)"-QSORT" as the *first command* in your program. DON'T use this command from the middle of a program. Once QSORT is in memory, it stays there until you do something "serious" like reboot, or reload BASIC.SYSTEM.

The advantage to loading QSORT "by hand" and not from a program is that it only has to be done once, and not every time your program is run. No big deal, loading only takes a couple of seconds.

MAKING QSORT WORK

QSORT will sort any one-dimensional string, integer or numeric array. To sort a string array named A$( ), use the following command inside a program:

```
CALL 950, A$(0)
```

Integer and numeric arrays are sorted the same way:

```
CALL 950, I%(0)
CALL 950, N(0)
```

The following program demonstrates how QSORT will alphabetize and display five words. It works just as easily with 100 or 1000 or 5000 words. The size of memory and the length of your words are your only limitations.

```
10  PRINT  CHR$ (4)"-QSORT"
20  DIM A$(4):A$(0) = "FISH":A$(1
    ) = "BICYCLE":A$(2) = "NOODL
    E":A$(3) = "LEAPIN' LIZARDS!
    ":A$(4) = "MINNIE MINOSO"
25  PRINT "UNSORTED LIST": PRINT
    : FOR X = 0 TO 4: PRINT A$(X
    ): NEXT : PRINT
30  CALL 950,A$(0)
40  PRINT "SORTED LIST": PRINT : FOR
    X = 0 TO 4: PRINT A$(X): NEXT
```

SORTING PART OF A LIST

If you have dimensioned a large array, but you only have data in the lower items, you can make a sort much faster by poking the number of items into locations 954-955. For example, to sort only 300 items of a 1000-item array, put this line in your program, *before* the CALL 950:

```
100  POKE 954,300-INT(300/256)*256: POKE 955, INT(300/256)
```

SORTING BY OTHER THAN THE FIRST CHARACTER

String arrays only: QSORT normally sorts words by looking at each word's first character first. For some odd reason, you might want to ignore the first few characters of the words. To do so, poke location 953 with the number of characters you want to skip over. For example, to sort a list of words, ignoring the first two characters of each word, put this line in your program, *before* the CALL 950:

```
100 POKE 953, 2
```

NUMBER SORT-NOTES

Strings are sorted in alphabetical order according to the ASCII values of the characters. Because of this, you may want to add leading zeros or spaces to your numerical strings. See "BAH NUMBUG" on page 57 for a few more details.

SORTED CORRECTLY	*ALSO SORTED CORRECTLY*
11	00008
1024	00011
256	00099
6502	00256
68000	01024
8	06502
8088	08088
99	68000

NO BUFFERS?

If you get a "No Buffers Available" error after using QSORT, you're probably not using ProDOS version 1.1.1. Solution: Use it.

MEM.ZAP

MEM.ZAP is a program that lets you inspect and edit your Apple's memory, much like BYTEZAP.PRO lets you inspect and edit disks. MEM.ZAP even lets you insert and delete bytes from memory, making the editing of machine language programs a snap. If you type in machine language programs from magazines, you're going to love MEM.ZAP!

NO HARM CAN BE DONE, BUT...

You can easily cause your Apple to malfunction by using MEM.ZAP. You will probably never cause any *real* harm that can't be cured by rebooting, so go ahead and experiment. As a precaution, we suggest that you OPEN YOUR DISK DRIVE DOOR(S) when using MEM.ZAP, just in case you accidentally inflict a case of brain damage upon ProDOS.

We could easily have written MEM.ZAP so you couldn't make any "illegal" changes. But some people like to play with matches, and... well, it's your computer—just don't say we didn't warn you.

ADDRESS LIMITATIONS FOR PEEKING AND POKING

MEM.ZAP lets you look at (or "peek" at) most areas of memory, but certain locations are off-limits and can't be viewed. In main memory ($0000-$FFFF) you can inspect everything except the 256 bytes from $C000 to $CFFF. If you did look there, all kinds of problems would instantly occur, including potential disk damage—not to mention MEM.ZAP screeching to a halt.

If you have an Apple IIc or 128K Apple IIe, only parts of auxiliary memory can be viewed (see next page).

Any area that can be looked at can be changed (or "poked"), unless it's located in ROM (Read Only Memory). It's not that we're stopping you; it's just these memory locations are set in silicon, and can't be changed from the keyboard.

You can LOOK AT and CHANGE:

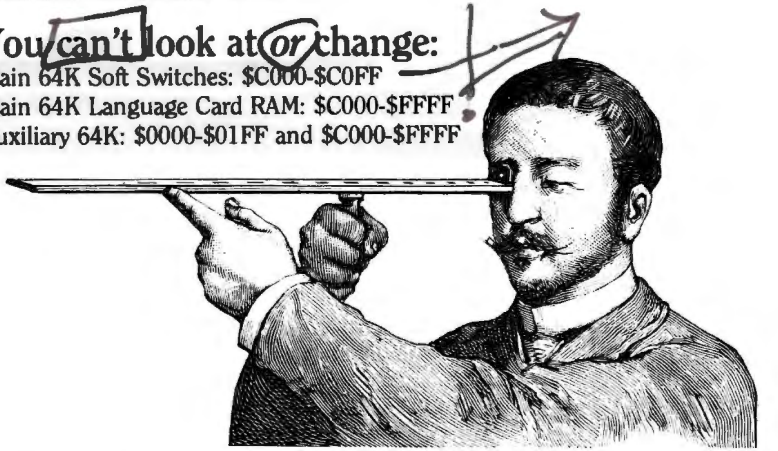
Main 64K: $0000-$BFFF (see caution below)
Auxiliary 64K (128K Apples only): $0200-$BFFF

You can LOOK AT but *not* change:

Main 64K ROM: $C100-$FFFF

You can't look at *or* change:

Main 64K Soft Switches: $C000-$C0FF
Main 64K Language Card RAM: $C000-$FFFF
Auxiliary 64K: $0000-$01FF and $C000-$FFFF



Caution: BE CAREFUL when changing these areas in memory.

(All in main 64K memory)
Zero Page and Stack: $0000-$01FF
Page 3 DOS Vectors: $03D0-$03FF
Text Screen: $0400-$07FF
The MEM.ZAP program itself: $4000 to about $5200
or MEM.ZAP.1000: $1000 to about $2200
BASIC.SYSTEM: $9600-$BFFF

INSTALLING MEM.ZAP

With ProDOS booted, type "-MEM.ZAP" or "BRUN MEM.ZAP" with the Pro-Byter disk in your drive. That's all there is to it.

MEM.ZAP.1000

The MEM.ZAP program resides in memory at $4000. If you want a version that locates itself lower in memory ($1000), BRUN MEM.ZAP.1000 instead. Same program, different address.

THE SCREEN

After MEM.ZAP is loaded, you will see a "hex dump" of memory similar to BYTEZAP.PRO's ASCII screen format (high-bit off; see page 22). Control-characters are represented in the ASCII-columns by colons, making text messages hidden in memory easier to see as they scroll by.

The bottom screen line displays the bank of memory currently being viewed (MAIN or AUX), the address and value at the cursor, and the current Range that can be changed and searched.

A flashing cursor will appear at one of the hex bytes on the screen. The address and value of that byte will be displayed on the bottom screen line. You can move the cursor in any direction and, if the byte is within the selected Range, change the byte by simply typing over it.

MOVING THE CURSOR AND SCROLLING

APPLE IIe AND IIc

The four ARROW KEYS move the cursor one byte or row in the appropriate direction; up, down, left or right. If you try to move the cursor off the screen, the screen will scroll up or down. Wrap-around is in effect; location $0000 follows $FFFF, and vice versa.

Pressing the OPEN-APPLE KEY and an ARROW at the same time moves the cursor three bytes horizontally or one screen vertically.

OLDER APPLES

If you have an Apple II or II+, use the " < " and ">" keys to move the cursor up and down. To jump a full page, use the COLON and HYPHEN keys.

MEM.ZAP Commands

MEM.ZAP is controlled by several one-keystroke commands. Most improperly typed commands or numbers will get you a beep and nothing more. The Command Card that came with your Pro-Byter disk gives you a brief summary of these commands.

G: GO TO AN ADDRESS

Press "G" and type a hex number, 0000-FFFF, to specify a new memory address for viewing. This command does not affect the Range of memory that can be searched or changed.

EXAMPLE: Type "G" followed by "D0D0" (those 0's are zeros, not O's). This will take you to location $D0D0 in main memory, where you will see the list of Applesoft commands in ROM. A couple of screens further on (scroll the screen up), the Applesoft error messages begin. You can't change memory here in ROM, only look at it.

X: MAIN/AUXILIARY SWITCH

(Apple IIc or 128K IIe only): Press "X" to toggle between the main 64K and the auxiliary 64K of memory. The address will stay the same unless you're in an auxiliary memory area that MEM.ZAP can't read.

R: RANGE

Press "R" to change the Range that will be affected by the Change, Insert, Delete and Search functions. You will be asked for the beginning and end of the range; enter hex values only. Serious problems can arise if you change sensitive areas of memory, so BE CAREFUL WHICH RANGE YOU CHOOSE.

0-9 AND A-F

If the cursor byte is within the Range indicated at the lower right of the screen, you can instantly change that byte by typing a hex number, 00-FF. MEM.ZAP won't handle decimal input, but the byte value that you type will be converted to decimal for you at the bottom of the screen.

CONTROL-I: INSERT

Press CONTROL-I (or TAB) to insert a zero at the cursor. The byte at the cursor and the succeeding bytes within the Range will be moved one memory location higher. The value of the final byte in the Range will disappear.

CONTROL-D: DELETE

Press CONTROL-D to delete the byte under the cursor. The succeeding bytes within the Range will be moved one memory location lower. The final byte in the Range will acquire a value of zero.

S: SEARCH

Press "S" to search the defined Range for a word. The same rules apply here as in BYTEZAP.PRO (see Search on page 12). To search for hex bytes, you'll have to enter their ASCII equivalents. To find the bytes "C1 C2 C3 DA", for example, enter "ABCZ" as the search string.

SEARCH EXAMPLE: Set the Range to "9600-BFFF". That's where ProDOS's BASIC.SYSTEM program resides in RAM. Then type "S" followed by "APPLE" to see how many times this word appears in BASIC.SYSTEM (press "Y" after each find). You can change anything in this range, but don't be surprised if your Apple starts acting like it's on drugs. Or stops acting altogether.

L: LIST (Disassemble)

Press "L" to disassemble twenty lines of memory. Press "L" again to list further. Any other key returns you to normal screen format. You cannot change bytes in disassembled format, only look at them. This feature does not work with auxiliary memory.

Esc: QUIT

Press ESC to quit MEM.ZAP. Type "CALL 16384" to return to MEM.ZAP. Or type "CALL 4096" if you're using MEM.ZAP.1000.

MEM.ZAP EXPERIMENTS

(Beware! Changing memory can cause computer malfunction.)

- Go to the text screen in Main Memory ($0400-$0800) and scroll up and down. Notice how the ASCII values "vibrate" as you scroll. This is because the program is reading the bytes on the text screen you are viewing, which are constantly changing as you scroll.

- Change some text screen bytes. Go to 0400 and set the Range to 0400-0800. Now hold down the "A" key and watch the action. Note: Apple Computer warns programmers not to change the text screen "holes", which is exactly what we're telling you to do here. Go ahead and do it. You may have to re-boot to get things working normally again.

- Go to the input buffer ($0200-$02FF). You might see remnants of input from previous programs. Then again, you might not.

- Set the Range to 0000-BFFF and search for a word. You will find the word at least once—converted to control characters—inside the MEM.ZAP program itself. This is because MEM.ZAP makes a copy of the word to use in its search routine.

- Search for a ProDOS command in BASIC.SYSTEM: Set the Range to 9600-BFFF and search for "CATALOG". Now (carefully), change the letters "CAT" to "DOG" by typing "44 4F 47". (NEVER *NEVER* INSERT OR DELETE BYTES HERE!) Now press ESC to quit MEM.ZAP and catalog your disk by typing "DOG". Now try "DOGALOG". And what about "CAT"?... You can make command changes like this "permanent" by using BYTEZAP.PRO's search and change features, but beware of overlapping commands; you may be changing two commands when you only meant to change one.

- Search for "JANFEBMAR" or "(NO DATE)" and change something (set Range as above). Now catalog a disk.

- Look at your Apple's slots! Goto C100 (slot 1) and scroll through C700 (slot 7). If a slot is empty and you're using a IIe, you will see vibration similar to what you get when viewing the text screen. That's because these "air bytes" are constantly changing.

- If you have 128K, Go to auxiliary memory. If someone's been using a 128K word processor since your Apple was turned on, you may find some interesting text.

- Search for a machine language command like LDA $C030 ($AD $30 $C0). Translate the bytes into ASCII equivalents first ("MP@" in our example), and search the appropriate part of memory.

HOW TO ENTER A MACHINE LANGUAGE PROGRAM USING MEM.ZAP

Have you ever seen a machine language program in a magazine but wondered how to type it in and save it? Well, load MEM.ZAP and give this sample program a try.

This is a (semi-worthless, but) educational little program that prints "UNCLE LOUIE" on the screen whenever you type CALL 9999. It can be listed in two ways:

HEX DUMP

```
270F- A0
2710- 00 B9 1D 27 20 ED FD C8
2718- C0 0B D0 F5 60 D5 CE C3
2720- CC C5 A0 CC CF D5 C9 C5
```

Values: DO type these in.

Addresses:
DON'T type these in.

DISASSEMBLY

```
270F-   A0 00       LDY   #$00
2711-   B9 1D 27    LDA   $271D,Y
2714-   20 ED FD    JSR   $FDED
2717-   C8          INY
2718-   C0 0B       CPY   #$0B
271A-   D0 F5       BNE   $2711
271C-   60          RTS
271D-   D5 CE       CMP   $CE,X
271F-   C3          ???
2720-   CC C5 A0    CPY   $A0C5
2723-   CC CF D5    CPY   $D5CF
2726-   C9 C5       CMP   #$C5
```

Note: The byte at $2719 is the length of the word "UNCLE LOUIE". Its value here is 11 ($0B); change it if you want to use a different-length word.

TYPING THE PROGRAM

- Type "-MEM.ZAP" with the Pro-Byter disk in your drive.
- Set the Range so you don't accidentally change fragile parts of memory. Type "R" followed by a START address of "270F" and an END address of "2730". The end is actually higher than necessary, but you might as well give yourself some slack. No telling what numbers you'll see on the screen at this point. It doesn't matter; you're about to change most of them anyway.
- Type in the values from the HEX DUMP listing— A0 00 B9 1D, etc., until you're done. If you make a mistake, you can go back, change, insert or delete numbers. No harm can be done here.
- (optional) To set your program off from the garbage in memory, type a few zeros (00's) after the final value, C5.
- Move the cursor to location 270F.
- Press "L" for a disassembly. Proofread it to make sure it matches the printed DISASSEMBLY. Press any key when you're through looking.
- When everything's perfect, press the ESC key to quit MEM.ZAP.

SAVING THE PROGRAM

- Insert a ProDOS disk and type "BSAVE LOUIE,A$270F,E$2727". The "A" and "E" numbers are the start and End addresses of the program. You could instead use decimal numbers or start and Length (L) instead.

```
BSAVE LOUIE,A$270F,E$2728
BSAVE LOUIE,A9999,E10024
BSAVE LOUIE,A$270F,L$1A
BSAVE LOUIE,A9999,L26
```

(all have the same effect)

USING THE PROGRAM

- Load the program by typing "BLOAD LOUIE".
- Then execute it by typing "CALL 9999".

CHANGING THE PROGRAM

- Let's change the program so it prints "UNCLE FRANK".
- Load the program by typing "BLOAD LOUIE".
- Load MEM.ZAP by typing "-MEM.ZAP".
- Type over the five ASCII values of "LOUIE" at addresses $2723-2727, using the ASCII values of F-R-A-N-K: C6 D2 C1 CE CC
- Press ESC and type "CALL 9999".

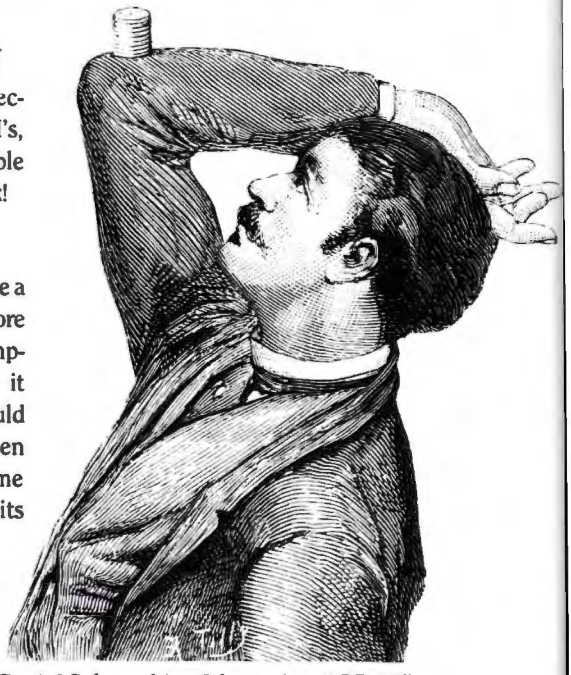
Apple Tips & Tricks

RENAME BASIC.SYSTEM

You can Rename BASIC.SYSTEM anything that ends in ".SYSTEM". You can't, however, Rename the PRODOS file unless you don't care if it loads when you boot.

CLEAR PROTECTION

Clear tape will serve as write protection on disks used in Apple II's, Apple II+'s and Apple IIe's. In Apple IIc's, only opaque tape will work!



SAVEX, DELETEX

If you want to play it safe and Save a program you're working on before testing, Save it under the temporary name "X". If you saved it under its actual name, it would overwrite your previous Save. When your new version works, Rename X or Delete it and Save it under its real name.

BOOT DRIVE 2!

On an Apple IIc, you can boot ProDOS from drive 2 by typing ":**PR#7**". The leading colon makes the IIc take this as an Applesoft command, not a DOS command. If you try to boot DOS 3.3 from a IIc's drive 2, it won't work, but the results are at least interesting.

INVISIBLE PRINTER

If you have no printer connected to your Apple II, II+ or IIe, and you type "PR#1", your system will hang, forcing you to hit Reset. An Apple IIc, however, seems to think there is an invisible printer attached. Type "PR#1" and then "LIST". A bit later, after your listing is beamed into outer space, you'll get your cursor back.

ProDOS, Pathnames & Prefixes

If you've never used ProDOS, here are a few of the basics about accessing ProDOS files. Read Apple's *BASIC Programming With ProDOS* manual for detailed information.

FIRST OF ALL

ProDOS's subdirectory and "pathname" system was designed by Apple programmers with *hard* disks for Apple programmers who use *floppy* disks (99%+ have floppy drives only). Eventually, as prices plummet, more and more of us will have hard disks, so maybe this is o.k. Using a hard disk, so we're told, is like having 40 or 50 super-high-speed floppies accessible from one disk drive. With floppy disks, you find the disk you want and stick it in your drive. On a hard disk, you need to type a command instead.

Still, ProDOS works with floppies, and it works well. You just have to understand the rules.

BOOTING ProDOS

ProDOS doesn't waste disk space by installing itself on every disk. That's good, because a BASIC boot-up disk requires 51 blocks of "system" programs—the files PRODOS and BASIC.SYSTEM. That's 18% of a floppy's capacity. PRODOS must be on a disk or the disk will not boot. BASIC.SYSTEM must be loaded into memory if you'll be programming in Applesoft. After these two files are loaded, ProDOS looks for a file called STARTUP. If found, it is executed. If not, nothing else happens.

DIRECTORIES AND SUBDIRECTORIES

To see a ProDOS disk catalog or "directory", type "CAT" (40-columns) or "CATALOG" (80-columns). You will soon see a catalog of files with the directory's name appearing at the top.

Some ProDOS disks have hidden catalogs called "subdirectories", indicated by a "DIR" file in the main directory. Subdirectories can have their own subdirectories too—can *that* ever get complicated. See page 4 for a discussion of how to get at Pro-Byter's two subdirectories.

To create a subdirectory use the ProDOS command "CREATE" followed by a legal name. For example, type "CREATE NAME".

PREFIXES

If you type a simple ProDOS command like "LOAD FILE", ProDOS will respond by looking for FILE in the directory or subdirectory that matches the current "prefix". To see what the current prefix is, type "PREFIX". If there is none, a disk will be read from one of your drives. A prefix always starts with a main directory name, like "/PRO.BYTER". If subdirectories are involved, they are included in the prefix—/PRO.BYTER/FREEBIES, for example.

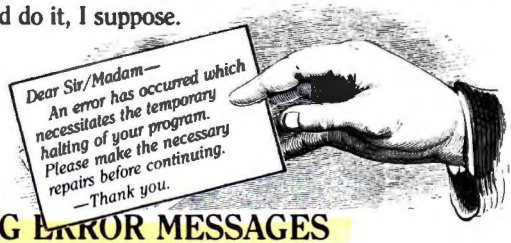
To set a prefix, type "PREFIX" followed by the name you want. For example, type "PREFIX/PRO.BYTER".

PATHNAMES

A "pathname" is a prefix attached to a file name. For example (read carefully now), BAD.NEWS is a program in the FREEBIES subdirectory of /PRO.BYTER. To Run the program you can specify the pathname by typing "RUN/PRO.BYTER/FREEBIES/BAD.NEWS". Or you can set the prefix by typing "PREFIX/PRO.BYTER/FREEBIES". Then type "RUN BAD.NEWS". Or you can reset the prefix to the main directory by typing "PREFIX/PRO.BYTER" or "PREFIX,D1" or "PREFIX,D2" (prefix of the disk in drive 1 or 2). The command "PREFIX/" cancels the current prefix.

CONFUSING?

Definitely. Until you get used to it. Our recommendation is *don't put subdirectories on your floppy disks.* Of course, we went and put *two* on Pro-Byter... Just to prove we could do it, I suppose.



ProDOS'S PUZZLING ERROR MESSAGES

PATH NOT FOUND

"Path Not Found" means "File Not Found". What's most frustrating is that you can catalog a disk, see the file name "PROGRAM", type "LOAD PROGRAM" and get a "Path Not Found" message. Don't panic; just mutter something appropriate and read on.

- "Path Not Found" often occurs when you switch disks. Typing "PREFIX/" will usually solve the problem.
- Type "PREFIX" (no "/" this time) to see which directory ProDOS is looking in. Your problems will probably be solved by resetting the prefix to the name of the directory and subdirectory you want to use.
- And last, but most important, make sure your spelling is correct, with no control-characters.

SYNTAX ERROR

You may be puzzled by an untimely "Syntax Error" message when you type a command that uses a file name. This usually means you broke one of ProDOS's file name rules:

1. No file name can be longer than 15 characters.
2. Every file name must start with a letter.
3. File names can only contain letters, numbers and periods.

"?Syntax Error" (with a question mark) means Applesoft error—don't blame ProDOS here.

AUDIO RADIENCE

Take a portable radio that's turned on and place it against one side of your Apple monitor. Now try the other side. You'll probably notice that one side causes more interference than the other. If I were you, I'd keep my disks away from that side.

FASTER FREESPACE

For some reason, ProDOS prints its Blocks Free and Used info at the bottom of disk catalogs. If you want to find out how much space is left on a disk without seeing the whole catalog, just type "CAT" (return) followed by a quick control-C.

```
]CAT

/BOOT.DISK

 NAME                 TYPE   BLOCKS   MODIFIED

 FIRST.PROGRAM        BAS         1   <NO DATE>

BLOCKS FREE:    84      BLOCKS USED:   196
```

TWO FULL MESSAGES

ProDOS's "Disk Full" message means just that. Either use another disk or delete some files and re-save. "Directory Full" means the 51 filename slots in the main (volume) directory have been filled. Either use another disk or start using a subdirectory. There is no number-of-files limit in ProDOS subdirectories.

To open a subdirectory, use the command "CREATE SUBNAME". To establish SUBNAME as the current directory, use the command "PREFIX SUBNAME".

IIc POWER OFF?

Did it ever occur to you that turning a IIc off using the built-in switch doesn't turn off the *power supply,* which is half way down the cord?... Or does it?

COLUMN-2 LISTINGS

The Apple IIc (and enhanced IIe) starts program listings at the second text column instead of the first. This is probably to facilitate easier escape-editing, since the second column is where you'll usually find your Applesoft cursor (the prompt, "]", occupies Htab 1). You can "escape-I" or "escape-uparrow" the cursor to the line number without having to move it one column to the left prior to tracing. Fooey on this kind of editing anyway. GPLE is the only way to fly!

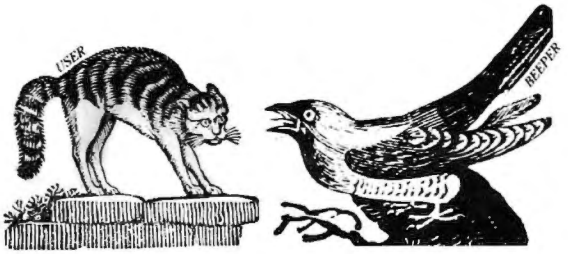
THE WORLD'S ALMOST-SHORTEST HEX CONVERTER

```
10  INPUT "DECIMAL NUMBER:";A$:A =
      VAL (A$)
20  POKE 71,A / 256: POKE 70,A -
      256 * PEEK (71)
30  POKE 50,64: POKE 59,249: PRINT
      "=$";: CALL - 327
```

CONTROL-L PAGE ADVANCE

The only thing that seems to be consistent about printers is their use of CHR$(12) (control-L) to activate a form feed (advance the paper to the start of the next page). Use it to your advantage when making program listings:

1. Align your paper at the start of a new page.
2. Turn your printer power switch on (or off, then back on). This should establish the start-of-page position in your printer's little dot-matrix brain. (Some printers have a "Set Top of Page" switch for this.)
3. Type "PR#1" (return) to connect your printer.
4. Type "LIST" (return). Wait while your program is listed.
5. Type control-L. Your paper should advance.
6. Type "PR#0" to disconnect.



CHIRP

User, the Beagle Bros cat, goes crazy when we play this one for him:

```
10  FOR X = 0 TO 17: READ A: POKE
    12345 + X,A: NEXT : DATA 173
    ,48,192,136,208,4,198,0,240,
    7,202,208,246,166,0,208,239,
    96
20 N =  RND (1) * 6:R = 1 +  RND
    (1) * 66: FOR X = 1 TO N: POKE
    0,R: CALL 12345: NEXT : GOTO
    20
```

Try changing N and R to constants. It's all kinds of fun!

BAH, NUMBUG!

Most string sorters we know of would rank the numbers 3, 20 and 100 in reverse order— 100-20-3, because ASCII-wise, that is correct. (Likewise, the words "C", "BO" and "ALL" alphabetize ALL-BO-C.) To support proper numeric sorting, you should start short numbers with spaces or zeros so that all numbers in the list have the same length— Our example numbers would then sort properly— 003-020-100.

THE BOOT-UP MESSAGE

The boot-up "message" at the top of the screen helps date an Apple. Type "CALL-1184" to see it any time. Pre-IIe Apples always reported "APPLE II". "APPLE][" means an older model IIe; "Apple II" means newer. The Apple IIc boots up saying "Apple //c", but the program below (for IIc's only) shows that the old "APPLE][" message is still hidden in ROM. But why?

```
10 ST = 64265:EN = ST + 7: GOSUB
    500
20 ST = 64771:EN = ST + 8: GOSUB
    500: END
500  FOR I = ST TO EN: PRINT  CHR$
    ( PEEK (I));: NEXT : PRINT :
     RETURN
```

Hmmm. Wonder if anyone at Apple ever considered making the Apple II+ boot-up saying "Apple II+" and the IIe boot-up saying "Apple IIe" and the... Nah, too simple!

ZZZZZZZZZZZZZZZ

We use a dummy file named "ZZZZZZZZZZZZZZZ" (15 Z's) as a "separator" between old and new ProDOS files on a disk. We periodically alphabetize the catalog with Fatcat's "Sortcat" utility. A few dozen Save's later, it's plain to see which files are old (above Z level) and which are new (below). Usually we delete several of the new files before re-sorting.

ILLEGAL BLOCK COUNT

If you see a 2-block ProDOS file in an unaltered ProDOS catalog, have someone examine your opticals. ProDOS files are always one block or three or more blocks in size.

```
/DISK
 NAME          TYPE  BLOCKS  MODIFIED         CREATED
 STARTUP       BAS        1  29-FEB-84 1:23   31-JAN-84 10:16
 PHONY.FILE    BAS        2  31-APR-79 25:03  32-APR-79 25:02
```

CHECK DISK DRIVE

The Apple IIc main drive will only run for a few seconds if you try to boot with the door open or with no disk in the drive (could happen because of a power failure). On other Apples, it's a good idea to keep a disk installed when you're away. Otherwise you may come home to a very tired hot disk drive.

CONTROL YOUR PRINTER

The manual that came with your printer probably tells you what control- or escape-command creates what special effect (bold characters, condensed, etc.), but I don't trust my printer manual any farther than I can throw it (and I've thrown it many times). Why not test your printer? Who knows, maybe it prints some character set or special effect that they never even told you about.

```
10  HOME : PRINT "CODE:";: GET A$
15  IF A$ = CHR$ (27) THEN PRINT "ES
    C-";: GET B$
20  PRINT CHR$ (4)"PR#1"
50  PRINT A$;B$;"WHAT DOES THIS DO?"
60  PRINT CHR$ (4)"PR#0": RUN
```

To clear previously-installed printer commands, you should click your printer's power switch off and back on again each time you see the word "CODE:". Then enter a control-character (like control-A), or Esc followed by another character (for example, Esc A). If "What does this do?" changes appearance, you've discovered something. By the way, Esc is the same as control-[or CHR$(27).

/THE/SHOR/TER/THE/BET/TER

ProDOS allows up to 15 characters for disk (volume), subdirectory and file names. If you're into actually using subdirectories, you'll be a happier, less-frustrated person if you keep those names short. Long pathnames are hard to remember, difficult to type and generally a pain.

CLEARMEM

To really clear memory and re-establish normal pointers, you can always re-boot. In DOS 3.3 you could simply type "FP". In ProDOS you can type "-BASIC.SYSTEM", but control-Apple-Reset is easier to spell.

CONVERT PROBLEMS?

If you're having problems with Apple's CONVERT program (converts 3.3 programs to ProDOS), you're not alone. We'd go over some of the problems here but, in a rage, someone destroyed our only copy of CONVERT.

PREFIX FIXES

To see what the current ProDOS prefix is, type "PREFIX". To set the prefix to the disk in drive 2, type "PREFIX,D2". To set the prefix to a subdirectory named "SUB", type "PREFIX SUB". To cancel all previous prefixes, type "PREFIX/".

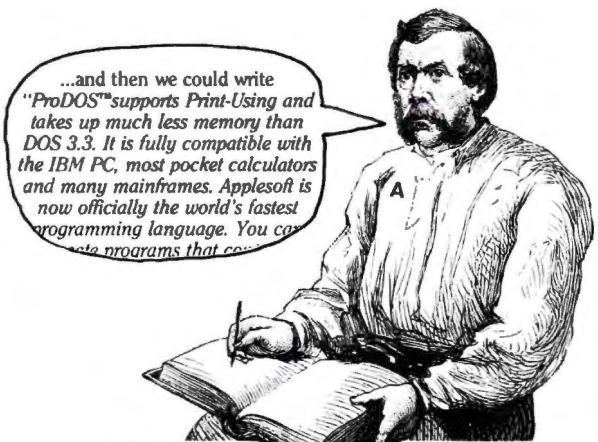
EXPONENT

Several Apple II hardware and software configurations now accept lower case direct and indirect commands. Try typing "print 123" (in lower case). If you get a ?Syntax Error, don't bother reading the next paragraph.

Oh *boy*, another quirk! Type the following command. It should produce a ?Syntax Error but it doesn't:

```
print 2~2
```

Ascii-ly speaking, the tilde (~) is a lower case caret (^). Remember, you read it here!



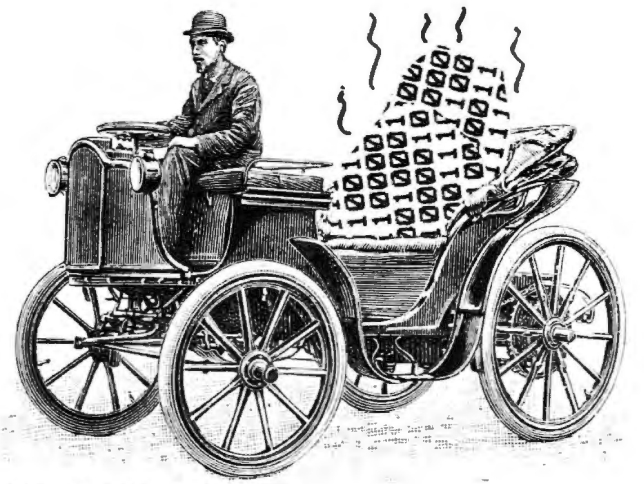
A LITTLE BALONEY FROM THE BIG A

Our version of Apple's "BASIC Programming With ProDOS" is a useful example-filled manual that we couldn't survive without. Go buy one. HOWEVER, it makes three false statements about ProDOS:

1. Page 68: ProDOS, like DOS 3.3, requires ctrl-D to be the first character "printed" on a line. In other words, HTAB 10: PRINT CHR$(4)"CAT" won't work, while PRINT: PRINT CHR$(4)"CAT" will. Actually, cursor position seems to be insignificant under ProDOS 1.1.1, but not earlier versions.

2. Page 154 and 206: ProDOS makes an HGR or HGR2 command protect the hi-res pages so your program won't overwrite them. That would be nice, but don't get your hopes up. Totally false.

3. Page 207: ProDOS lets Applesoft's INPUT statement accept commas and colons. Wrong again—it's still the infamous "?Extra Ignored".



GARBAGE PAGE

Programmers agree that ProDOS's garbage collection routine works infinitely faster (almost) than Applesoft's. But this simple program would lead you to believe differently:

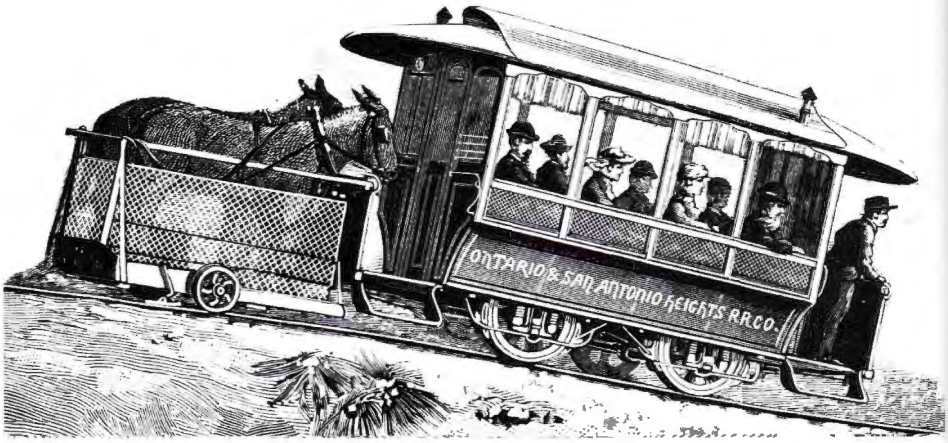
```
10  FOR A = 1 TO 255: PRINT A,
20  PRINT  PEEK (111) +  PEEK (11
    2) * 256
30 A$ = A$ + "A":B$ = B$ + "B":C$
     = C$ + "C": NEXT
```

Run this program under ProDOS and DOS 3.3. All we're doing is building some 255 character strings: AAA..., BBB..., CCC..., etc. Line 20 shows us the start-of-strings location, which gets lower as Applesoft creates more and more string garbage. The lower this number gets, the closer it comes to overwriting the end of our program in memory. At some critical point, "garbage collection" must be done to raise the start-of-strings location. In ProDOS, you will notice an occasional pause in the action. In DOS 3.3, hardly any pause will be noticed. This over-simplified example is not a good test however. ProDOS's garbage collector is almost always the winner.

To force a garbage collection, you can add this ProDOS command:

```
15  PRINT  CHR$ (4)"FRE"
```

Under either DOS 3.3 or ProDOS you can use an Applesoft X=FRE(0) command. It works, but is much slower than ProDOS's FRE.



ProDOS DISKQUIK

If you have an Apple IIc or a 128K IIe, you are equipped with a built-in "RAM-disk" in auxiliary memory. To prove that it's there, type "CAT,S3,D2". You should see an empty ProDOS catalog named "/RAM", ready to be filled with about 119 blocks-worth of files. (If you only have a 64K Apple, you'll get a "No Device Connected" message.)

To demonstrate the speed of /RAM, load a large file from disk. Now set the prefix by typing "PREFIX/RAM" or "PREFIX,S3,D2". Now type "SAVE FILE" and then "CAT". You should see "FILE" in the /RAM directory. Now type "NEW" and "RUN FILE" to load FILE from /RAM back into memory. Pretty fast, huh?

You can save different hi-res pictures into /RAM and call them to the screen at lightning-fast speeds. Try this:

```
10  HCOLOR= 3:D$ =  CHR$ (4): PRINT
    D$"PREFIX/RAM"
30  HGR2 : FOR X = 1 TO 6: HGR2 :
    FOR Y = X * 9 TO X * 9 + 99
    : HPLOT X * 19,Y TO X * 19 +
    149,Y: NEXT : PRINT D$"BSAVE
    PIC."X",A$4000,L$1FFF": NEXT

50  RESTORE : FOR X = 1 TO 10: READ
    P: PRINT D$"BLOAD PIC."P",A$
    4000": NEXT : DATA 1,2,3,4,5
    ,6,5,4,3,2: GOTO 50
```

Beagle Bros' Extra K disk has a program called "Extra.Screens" that lets you store up to seven hi-res screens in auxiliary memory. It also lets you crop pictures so you can store even more. The possibilities are endless. Check our ads. If you're intrigued by /RAM , you'll really like Extra K.

DISK HANGER

Someone wrote and told us you can punch a hole through the corner of an often-used disk, attach a string, and hang it within reach of your drive. Now there's a disk you'll never lose!

This is definitely not the worst idea we've ever heard. That one was from the lady in Tulsa who attached her floppies to the side of her monitor with refrigerator magnets.

BIT SPLITTER

Want to know how to break a decimal number into bits?

```
10  INPUT "NUMBER: ";N
15  IF N > 255 OR N < 0 THEN 10
20  FOR BIT = 1 TO 8: HTAB 9 - BIT
30  PRINT N -  INT (N / 2) * 2;
40 N =  INT (N / 2): NEXT
50  PRINT : PRINT : GOTO 10
```

(Use in 40-columns only, not 80.)

WHICH APPLE'S WHICH?

```
10 A =  PEEK (64435):B =  PEEK (64448)
20  IF A = 6 AND B = 0 THEN  PRINT "APPLE IIc"
30  IF A = 6 AND B = 234 THEN  PRINT "APPLE IIe"
40  IF A = 6 AND B = 224 THEN  PRINT "ENHANCED APPLE IIe"
50  IF A <  > 6 THEN  PRINT "APPLE II OR II+"
```

POKE FIX

If typing "RUN" gives you a weird message like "?Syntax Error In 65064" or if "NEW" causes a "?Syntax Error", a quick POKE 2048,0 should fix things. If you're into *causing* problems instead of fixing them, POKE 2048 (the byte prior to the start-of-program) with a number other than zero.

MAKING A COPY? WRITE-PROTECT!

Always write-protect your master disk when making a copy. If you don't, one of these days you're going to put the wrong disk in the wrong drive. We've seen it happen too many times (once).

HUSTON WAS HERE

If you use BYTEZAP.PRO to search a ProDOS disk for the cryptic word "HUSTON", you'll probably find it more than once. Please don't write and tell us what it means. It doesn't mean anything.

NO DEVICE CONNECTED

CALL 39662 if you don't believe us.

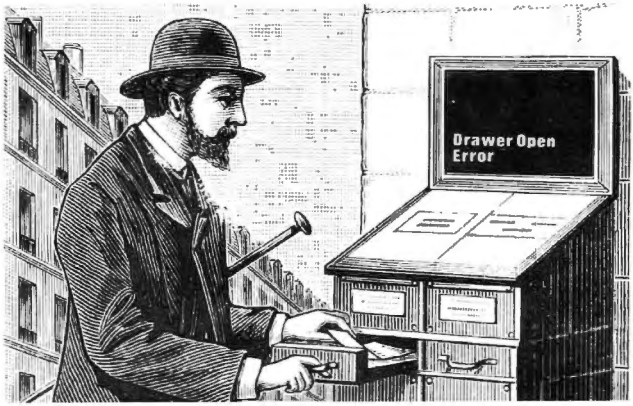
BLOCK/SECTOR CONVERTER

The strange relationship between ProDOS blocks and DOS 3.3 sectors can be charted by running this program.

```
10  PRINT  CHR$ (4)"PR#1": REM
    SET YOUR PRINTER FOR CONDENS
    ED TYPE
100 DIM S(15),AB(15): FOR I = 0 TO
    15: READ S(I): NEXT
120 FOR I = 0 TO 15: READ AB(I):
     NEXT
150 DATA  0,7,6,6,5,5,4,4,3,3,2,
    2,1,1,0,7
155 DATA  0,0,1,0,1,0,1,0,1,0,1,
    0,1,0,1,1
310 PRINT  SPC( 30);"SECTORS": PRINT
    "TRACK    ";: FOR I = 0 TO 15
    : PRINT I; SPC( 3 + (I < 10)
    );: NEXT
320 FOR TR = 0 TO 34: PRINT : PRINT
     SPC( TR < 10);TR;" - ";
350 FOR SE = 0 TO 15:BL = 8 * TR
    + S(SE):PT = AB(SE)
360 PRINT  SPC( 1 + (BL < 100) +
    (BL < 10));BL; CHR$ (65 + (P
    T = 1));
370 NEXT : NEXT : PRINT  CHR$ (4
    )"PR#0"
```

DUPLICATE FILE NAME

ProDOS has an error trap to prevent you from Renaming a file with the same name as another. "Duplicate File Name" is the error message. Thanks, ProDOS.



PR# YOU-NAME-IT

ProDOS lets any number 1-7 access any slot 1-7. Think hard, and maybe you'll come up with a reason for doing this.

```
10 SLOT=1: REM The slot you want to access
20 NUM=5: REM The number you want to use
30 LOC=48656+NUM+NUM
40 POKE LOC,0: POKE LOC+1, SLOT+192
50 PRINT "TYPE PR#"; NUM; " TO ACCESS SLOT "; SLOT; "."
```

The program above should make PR#5 turn on your printer in slot #1. If it doesn't, phone Steve or Woz.

DOS 3.3 CATALOG KILLER

From Randy Brandt (semi-famous hockey player): This tip lets you change a DOS 3.3 catalog so that as many file names as you want are inaccessible until you save a file on the disk!

1. Delete a file from the middle of a DOS 3.3 catalog.
2. Use BYTEZAP.PRO to change the track-byte of that file to zero (that's the left-byte of "B" on page 32). Write the change to disk.
3. That's it.

BEAGLE'S FIRST AND ONLY CP/M TIP

CP/M has little in common with the other Apple disk operating systems. The disk is mapped differently, files are stored in 1K "blocks", and the file directory is on track 3.

The only CP/M tip you're going to get out of this book is coming up now: To undelete a CP/M file, find the file name; (the period between the 8-character name and the 3-character extension won't be seen). The byte just before the name will be a $E5. Change it to a $00 and your file will magically be restored, unless it has been overwritten. End of tip. End of CP/M. Thank you for your patience.

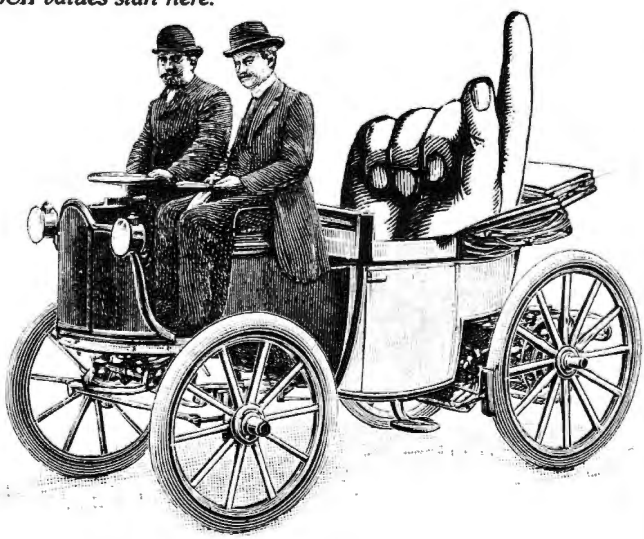
WORD.CALL

Running this program will make a subsequent CALL print anything you want. Substitute the word you want in Line 20.

```
10 LOC = 16384: REM CALL 16384 TO
     EXECUTE
20 A$ = "DUDLEY DOORIGHT"
50  FOR A = LOC TO LOC + 13: READ
     B: POKE A,B: NEXT : DATA 160
     ,0,185,0,0,32,237,253,200,19
     2,0,208,245,96
60  POKE LOC + 3,A -  INT (A / 25
     6) * 256: POKE LOC + 4, INT
     (A / 256)
80  POKE LOC + 10, LEN (A$): FOR
     J = 1 TO  LEN (A$): POKE A +
     J - 1, ASC ( MID$ (A$,J)) +
     128: NEXT
```

Here is the machine language program that is produced by running the Applesoft program above. To see it, type "CALL-151" (return) and "4000L" (return). You can execute the program by typing "4000G" (return). You can get back to BASIC by typing control-C (return).

```
*4000L
4000-  A0 00     LDY  #$00      Load Y with a zero.
4002-  B9 0E 40  LDA  $400E,Y   Load A with the value at $400E+Y
4005-  20 ED FD  JSR  $FDED     Print the character in A.
4008-  C8        INY            Increase Y by 1.
4009-  C0 0F     CPY  #$0F      Is Y equal to the length of the word yet?
400B-  D0 F5     BNE  $4002     If not, go to $4002.
400D-  60        RTS            That's all.
400E-  C4 D5     ASCII values start here.
4010-  C4 CC
4012-  C5 D9
4014-  A0 C4
4016-  CF
4017-  CF
4018-  D2
4019-  C9 C7
401B-  C8
401C-  D4
```

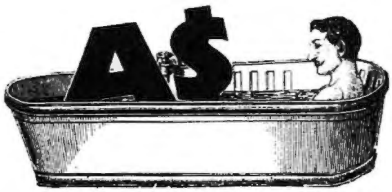


NO-SCROLL RESET

The IIc and enhanced IIe text display doesn't scroll up a line when you hit control-Reset. Nice feature; too bad GPLE can't support the improvement.

POST-LIST FIX

If you are stopping and starting a program listing with control-S, and you reach the end of the listing, your last control-S may accidentally become the first character of the next command you type, causing a ?Syntax Error. One solution is to hit a backspace or two to produce a new prompt before doing any typing.



STASH YOUR VARIABLES

One of the nice things about ProDOS is its ability to store variables on disk. To save a sometimes-significant amount of program space, temporarily insert a program line after a program's variables are set up:

```
100 PRINT CHR$(4)"STORE VARIABLES": END
```

This will create a type-"VAR" disk file named "VARIABLES" (or whatever) in your disk's directory. Now you can delete the line you just added plus all of the variable-setting lines from the beginning of your program, and replace them with:

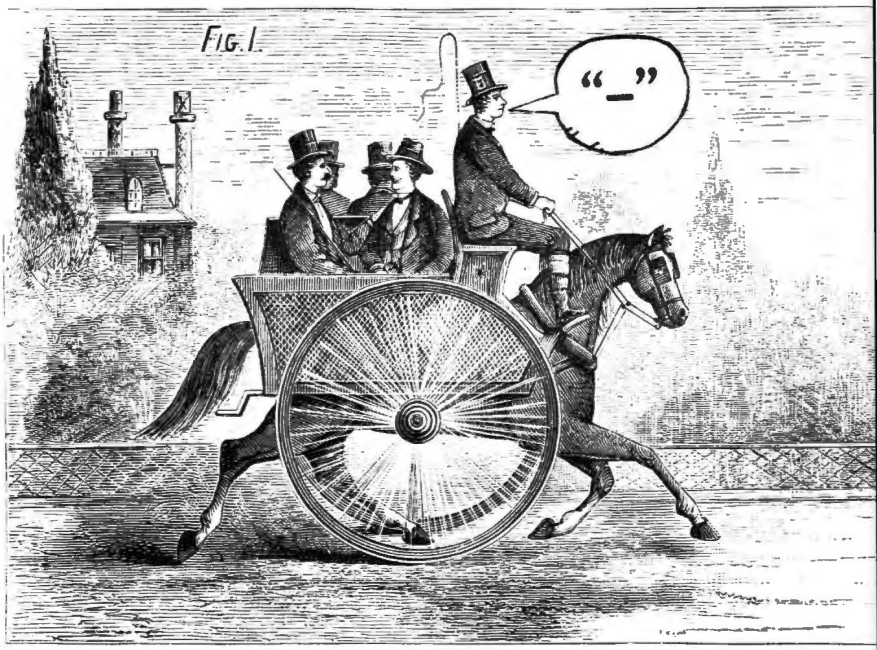
```
100 PRINT CHR$(4)"RESTORE VARIABLES"
```

This will load your variables from disk into memory (zapping any values that might have already been established). The amount of program space saved can be significant, especially with large programs and/or complicated array set-ups. Be sure to make a backup of the code that writes your variables in case you want to make some changes later.

CHAIN

The ProDOS CHAIN command uses the STORE/RESTORE function without writing on your disk. To run one program from another, with its variables remaining intact, just command:

```
PRINT CHR$(4)"CHAIN NEXT.PROGRAM"
```



SEMI-SMART RUN COMMAND

ProDOS's hyphen command is a handy animal to have around. The command "-FILENAME" will automatically RUN an Applesoft file, BRUN a Binary file or EXEC a text file. What it *won't* do is BLOAD a picture file or RESTORE a variable file. Hey, no command is perfect.

NO MONSTERS ALLOWED

Did you know that, with ProDOS booted, you can type the title of this paragraph and not get a ?Syntax Error? Wow—Tell your friends!

DOS CONNECTOR

Certain circumstances will disconnect ProDOS. If commands like "CAT" or "PREFIX" suddenly don't function, try typing "CALL 976". If ProDOS commands still don't work, you'll probably need to reboot. Don't use this Call in a program unless you want it to End at that point.

One way to disconnect ProDOS intentionally on a IIc or IIe with an 80-column card is to type ":PR#3" (notice the colon). After doing so, try Running a program. You'll see line #numbers all over the screen, because ProDOS has left you with the Trace function in gear.

MESSAGE.DECODER

(If you aren't into entering hex in the monitor, skip this page.)

ProDOS 1.1.1 hides its messages. *Boy,* does it hide them! The words used in error messages and a couple of other places are compressed in a format that is so hard to explain that we're not going to explain it. What we have done, however, is put a program called MESSAGE.DECODER on the Pro-Byter disk that will decipher any message you type in. Follow the instructions below.

Here are the starting memory locations of ProDOS's messages. The number of bytes allotted for each message is in parentheses:

$BA48: *COPYRIGHT APPLE...* (16)
(used by "VERIFY" command)
$BA58: *NAME, etc.* (38) (cat header)
$BA7E: *BLOCKS FREE, etc.* (30) (trailer)
$BA9C: *RANGE ERROR* (7)
$BAA3: *NO DEVICE CONNECTED* (11)
$BAAE: *WRITE PROTECTED* (9)
$BAB7: *END OF DATA* (6)
$BABD: *PATH NOT FOUND* (9)
$BAC6: *I/O ERROR* (6)
$BACC: *DISK FULL* (6)
$BAD2: *FILE LOCKED* (7)
$BAD9: *INVALID PARAMETER* (10)
$BAE3: *NO BUFFERS AVAILABLE* (13)
$BAF0: *FILE TYPE MISMATCH* (12)
$BAFC: *PROGRAM TOO LARGE* (11)
$BB07: *NOT DIRECT COMMAND* (10)
$BB11: *SYNTAX ERROR* (8)
$BB19: *DIRECTORY FULL* (8)
$BB21: *FILE NOT OPEN* (8)
$BB29: *DUPLICATE FILE NAME* (11)
$BB34: *FILE BUSY* (7)
$BB3B: *FILE(S) STILL OPEN* (12)

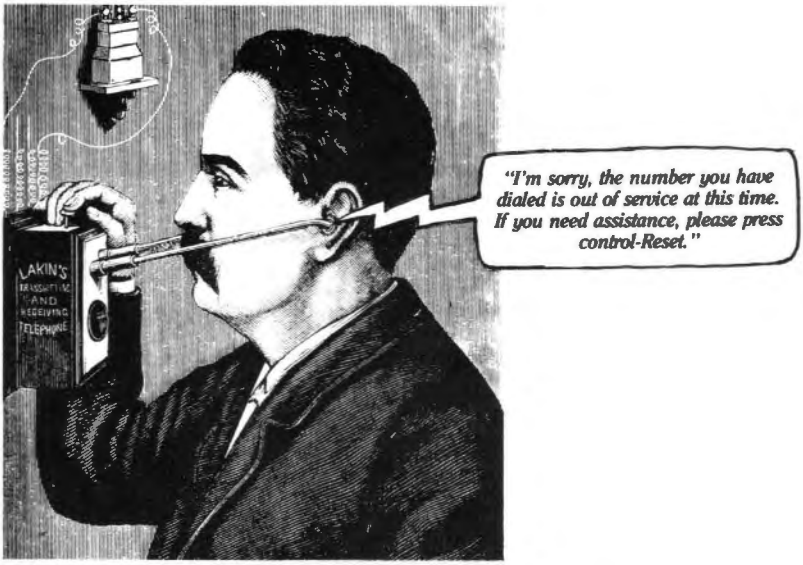
THE PROCEDURE

Here's what you do to change the "PATH NOT FOUND" message to "NOT HERE". Other changes are made similarly:

- Boot ProDOS 1.1.1 (the Pro-Byter disk will do fine).
- Type "-MESSAGE.DECODER" to get the program running.
- When asked for a message, type "NOT HERE". The hex numbers "9A CE 03 4B 4F" will appear on the screen.
- Type "CALL-151" to enter the monitor.
- Type the address of "PATH NOT FOUND" (see list above), followed by a colon and the new hex values: "BABD: 9A CE 03 4B 4F"
  Warning: Never enter more bytes than are allotted for the old message!
- Type control-C (return) to get back to BASIC.
- Type "LOAD GARBAGE" (a non-existent program name). The message "NOT HERE" should appear.
- To make a message change occur when you boot a disk, you can do the above procedure in decimal Pokes from your STARTUP program.

NO J's, Q's OR Z's PLEASE

You can't use J's, Q's or Z's in messages (don't blame us; *we* didn't write ProDOS). You *can,* use parentheses, periods, colons, spaces and slashes.



DIRECTORY-FULL PREVENTION

If you tend to save more than 51 files in your floppy catalogs, and you don't like messing with a bunch of subdirectories, save all of your files in one subdirectory—any number of files (within reason) is possible. Name your main directory and subdirectory something short, like "A" and "B". Then command "PREFIX/A/B" and you're in business.

RENAME YOUR DISKS

The ProDOS Rename command lets you change the name of your disks. Just command "RENAME/OLDNAME,/NEWNAME".

SUBDIRECTORY PROTECTION

ProDOS won't let you delete a subdirectory (DIR) file until all of the files in that subdirectory have been deleted. Makes sense.

PRO-FORMAT PROBLEM

Unlike DOS 3.3, ProDOS has no built-in INIT command for formatting disks. You have to use Filer or one of Apple's other utilities to create new disks. The problem with this is that if you get a Disk Full message and have no disks with free space, you can't save the program in memory without deleting something from a disk. The best plan is to format a box of disks at a time (use some non-descript name like "/DISK"; you can change it later). If you don't have a copy of a program that will format a ProDOS disk, make a backup copy of Pro-Byter, then delete all of the files.

VERIFY WHAT?

With ProDOS booted, type "VERIFY" followed by a filename. This proves that the file exists on the disk. Now type "VERIFY" with no filename. This proves you're not using a Commodore 64.

BETTER BOOT THAN BYE

ProDOS's "BYE" command (Version 1.1.1 and later) is a big pain if you ask us. If you're into big pain, give it a try—just type "BYE". You'll first be hurled into the land of No Return and be asked to "Enter Prefix" (of what, they don't say). Sorry, no default drives like ",D1" or ",D2". Then you're asked for the "Pathname of Next Application". Talk about Computerese—forget it! When you see this stuff, just boot the disk you want, or turn off your Apple and go take a nap.

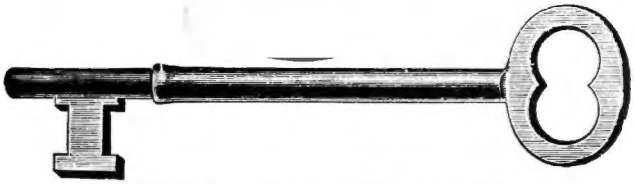
What Bye is looking for is the Prefix (name) of the next disk you want to use and the name of the next system program (like BASIC.SYSTEM) that you want to load. If you're using a hard disk, this makes a lot of sense. Otherwise, no.

See page 28 for a BYTEZAP.PRO tip on changing BYE's messages.

BOOT DATE

If you have a Thunderclock connected to your Apple IIe, you can play with this STARTUP program. Line Zero has a REM that contains the last time and date booted. Everytime the program is Run, the time and date are printed on the screen, updated in the listing and re-Saved on disk.

```
0  REM WED MAY 15  3:11:22 PM....
      .....
1000  PRINT "LAST BOOT: ";X$ = "
      ": FOR I = 2054 TO 2054 + 21
      :X$ = X$ +  CHR$ ( PEEK (I))
      : NEXT : PRINT X$: REM READS
       LINE 0 AND PRINTS IT
1010 D$ =  CHR$ (4): PRINT D$"PR#
      7": PRINT D$"IN#7": INPUT "%
      ";T$: PRINT D$"IN#0": PRINT
      D$"PR#0": REM READS CLOCK IN
      TO STRING T$
1020  PRINT "THIS BOOT: ";T$: FOR
      I = 1 TO 22: POKE 2053 + I, ASC
      ( MID$ (T$,I)): NEXT : REM
      POKES T$ INTO LINE 0, CHANGI
      NG THE DATE THERE
1030  PRINT D$"SAVE STARTUP": REM
      SAVE FOR NEXT TIME YOU BOOT
```



FUN WITH THE LOCK BYTE

Normally a disk's files are either UNLOCKED or LOCKED. Unlocked files can be read from and saved to, while locked files can be read, but not re-saved or renamed.

Well, we just found out (and *you're the first to know*) that there are other possibilities! RUN BYTEZAP.PRO and go to a ProDOS catalog block on a backed-up disk. Now look on page 30-31 of this manual. See byte "M", the "LOCKED STATUS" byte? It can be changed to one of these values:

Value	*Read?*	*Write?*	*Rename?*	*Delete?*
$20	-	-	-	-
• $21	Y	-	-	-
$22	-	Y	-	-
$23	Y	Y	-	-
$60	-	-	Y	-
$61	Y	-	Y	-
$62	-	Y	Y	-
$63	Y	Y	Y	-
$A0	-	-	-	Y
$A1	Y	-	-	Y
$A2	-	Y	-	Y
$A3	Y	Y	-	Y
$E0	-	-	Y	Y
$E1	Y	-	Y	Y
$E2	-	Y	Y	Y
• $E3	Y	Y	Y	Y

Normally, ProDOS uses only one of two values in the Lock byte: $21 means LOCKED (you can only read from it). $E3 means the file is UNLOCKED (you can read, write, rename and delete the file).

There are 16 possibilities. Some of the most fun are $22 (write only, no read), $63 (no delete, but you can read, write or rename it), and $20 (no anything; it just sits there).

No keyboard command will change the Lock byte to a non-standard value; you'll have to use BYTEZAP.PRO.

FREE SPACE! FREE SPACE!

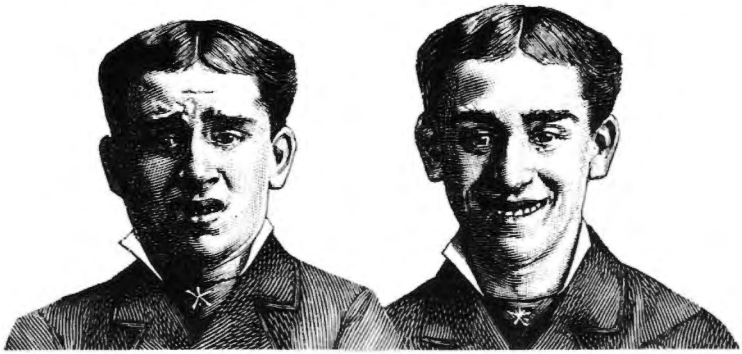
Rumor has it that there are 302 free bytes inside BASIC.SYSTEM (in main memory RAM), from about $BB4C to $BC79. If you don't grab it now, someone else will. Every member of the Beagle Bros programming staff voted that we *not* tell you about this free space. But then *they* aren't writing this book; *they're* sitting at home playing with their Apples.

COMMAND CRUNCHER

Tired of Applesoft commands that work and error messages that make sense? Well **POKE 48646,44** should give you a chuckle. What it does is make every non-ProDOS command print an error message (of sorts); see if you can figure out the pattern. Try commands like LIST and PRINT, and non-commands like COMMAND and HOW'S YOUR SISTER?

This tip wins the award for most creative Apple brain damage done with one poke, and it works on any version of ProDOS! For a quick cure, **:POKE 48646,76.**

A CLOSELY GUARDED SECRET (tell your friends): You can execute typed Applesoft commands if you precede them with a colon.



CURING THE NO-BUFFER BLUES

The "NO BUFFERS AVAILABLE" error message has to be one of the most aggravating in Apple history, especially when you want to load a file or catalog the disk. It's caused by various things, including pressing RESET during a disk catalog, trying to Bload a file into an illegal area like the text screen or zero page. The message is often the result of an error in the "bit map" that tracks available memory. Clear the bit map with this line. If you use GPLE, define this as an ESC function:

```
FOR I=48985 TO 49003: POKE I,0: NEXT
```

PEEKS & POKES UPDATE

If they mean anything to you, pencil these ProDOS additions in on your Peeks & Pokes Chart (if we haven't updated it already).

CALL 976 ($3D0): Reconnect ProDOS
CALL 39565 ($9A8D) Restore ProDOS I/O hooks
CALL 40802 $9F62) Store AXY registers at $BE3E-$BE40
CALL 40813 ($9F6C) Restore AXY registers
CALL 44576 ($AE20) Run last Bloaded program
(won't work if you have Cataloged since the Bload)
CALL 54335 ($D43F) Warmstart Applesoft

...AND A PEEKS & POKES CORRECTION

Last-Bloaded ProDOS file's start address:

```
PRINT PEEK(48855)+PEEK(48856)*256     ($BED7.$BED8)
```

Last-Bloaded ProDOS file's length:

```
PRINT PEEK(48857)+PEEK(48858)*256     ($BED9.$BEDA)
```

MODIFIED, THEN CREATED?

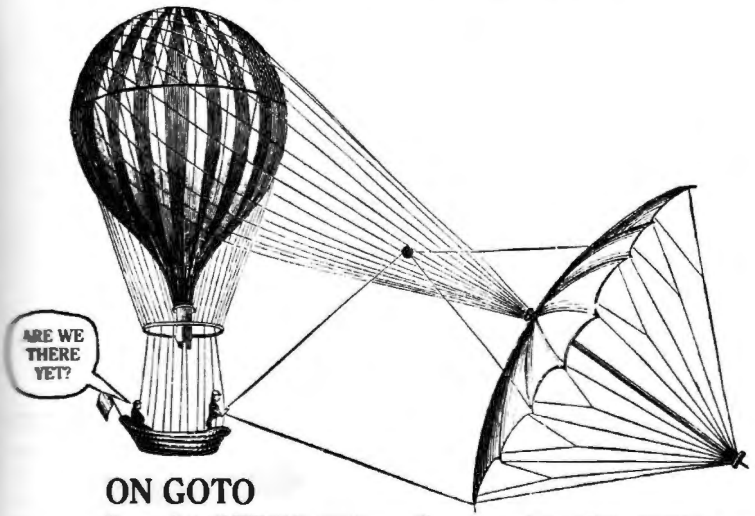
If you transfer a file from disk to disk, the current date, if set, becomes the "Created" date, making it later than the file's "Modified" date. Oh, well.

ANOTHER BYTE SAVER

Instead of **VTAB 20: HTAB 1: PRINT IT**, use **VTAB 19: PRINT: PRINT IT**. You'll save two big bytes, and besides that, this is an effective 80-column trick that overcomes an Htab bug in the older IIe's.

?ILLEGAL MESSAGE

If you type "RENAME OLD.FILENAME, ANOTHER.FILENAME" under ProDOS, you'll get a Syntax Error message. The problem is the new filename is longer than 15 characters. Come *on*, ProDOS—Why not just truncate the name at 15 like DOS 3.3 does at 30?



ON GOTO

If you list BYTEZAP.PRO, you'll notice a lot of ON...GOTO statements, a trick that allows more statements per line. Take this example... Please:

```
10  X = 3: REM MULTIPLE BRANCHES
20   ON X = 1 GOTO 100: ON X = 2 GOTO
       200: ON X = 3 GOTO 300: ON X
         = 4 GOTO 400
100    PRINT "CRABGRASS": END
200    PRINT "RANUNCULUS": END
300    PRINT "UNCLE SCROOGE": END
400    PRINT "BURMA SHAVE": END
```

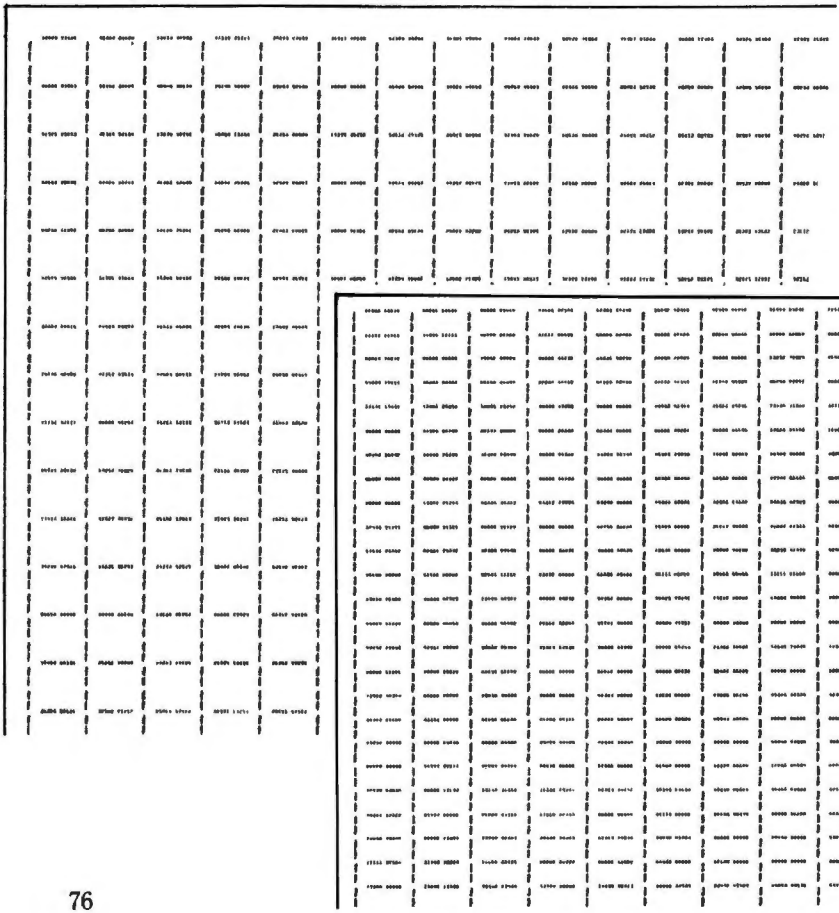
If we had used If's instead of On's in line 20, it would have had to be broken into four program lines. Play around with this one—it's *almost* as good as If-Then-Else... but not quite.

APPLE GRID PAPER

The next time you need a piece of graph paper, write a program that makes some for you. The sample program and printouts below barely touch on the possibilities. Use your printer's special-effect features to adjust the darkness of lines, compactness of grid, etc. Different brands of printers will produce different results; check your printer manual (good luck).

You can print as many grid copies as you like, or—to combat noise pollutions and save on ribbons—take a master printout to your local "instant printer" or photo-copy machine.

```
40  REM SAMPLE GRID PROGRAM:
100 V$ =  CHR$ (124):L$ =  CHR$ (
    95):S$ =  CHR$ (32)
200 FOR H = 1 TO 24: PRINT S$;L$
    ;L$;: NEXT : PRINT S$
400 FOR V = 1 TO 40: FOR H = 1 TO
    24: PRINT V$;L$;L$;
500 NEXT : PRINT V$: NEXT
```



BYTEZAP.PRO Program Info

If you're having problems using BYTEZAP.PRO with your particular equipment set-up, list the first few lines and check out the variable values. You're on your own from there.

There are very few REM's in the program and it is tightly-written to get the most out of the most space. Beagle Bros' D CODE was used for compacting and de-bugging. Editing was done with GPLE and DOUBLE-TAKE. BZAP.BIN was written with MERLIN-PRO. End of plug.

VARIABLES

GENERAL VARIABLES: The main variables are stored in the file BZAP.VAR, which is Restored at the beginning of the listing.

PRINTER SLOT: You can change variable PSLOT (PS) to a value other than 1.

PRINTER LINEFEED: Try setting LFEED$=CHR$(13) (normal value is null) if your printer isn't doing a line-feed at the end of each line.

DISK DRIVE SLOT: If your disk drive is connected to a slot other than 6, change the POKE 24083,6 to POKE 24083,*yourslot*.

OTHER DRIVE VARIABLES: DN is the number of drives connected (1 or 2). DR is the current drive number (1 or 2).

MAXIMUM BLOCKS AND TRACKS: BMAX (BM) is the maximum blocks normally allowed (normally 279). TMAX (TM) is the maximum number of tracks (normally 34).

CURRENT BLOCK/TRACK/SECTOR: Variables BL, TR and SE take care of this.

CURSOR POSITION: Five integer arrays determine the position of the cursor in the various screen formats. The cursor is at position PZ (0-255) in track buffer location L.

PROGRAM SECTIONS (by line number; subject to change)

0- 200: Title screen and set-up	6000-6999: Trace
201- 799: Subroutines	7000-7499: Screen format
800- 999: Check for disk type	7500-7699: Value of cursor byte
1000-1099: Update bottom-of-screen	7700-7999: Date/time (delete to make space)
1100-1499: Get a keypress	8000-8499: Map
1500-1599: Search	8500-8849: Print screen
2000-2999: Catalog and command	8850-8999: Title screen
3000-3999: Read next block/sector	9000-9799: Read a track
4000-4999: Write to disk	9800-9899: Input routine
5000-5999: Byte change	9900-9999: Error trap

BYTEZAP TOO BIG?

An early program line checks to see that the end of the program never gets larger than variable TS. That's all the room there is!

HARD DISK? TRY THIS:

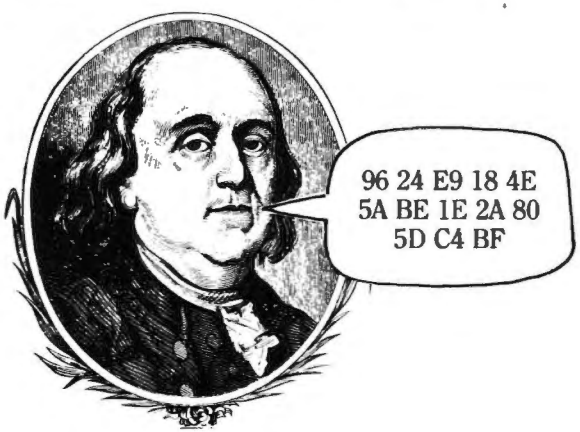
We haven't tried it, but to use BYTEZAP.PRO with a hard disk, try changing the "6" in POKE 24083,6 (early in the program) to the slot of your hard disk. You should also raise the value of BMAX to the number of blocks on your hard disk. BYTEZAP.PRO won't read DOS 3.3 from a hard disk.

CUSTOMIZING PRODOS 1.1.1 TO WORK WITH THE FRANKLIN ACE

ProDOS will not boot on Apple clones such as the Franklin Ace and the Basis 108. Using BYTEZAP.PRO and someone's Apple, you can change a few bytes to remedy the situation.

1. Boot a back-up of the Pro-Byter disk and RUN BYTEZAP.PRO.
2. Press "S" and type "PRODOS,2". Answer "N" to "Search for More?".
3. Press "L" to load the file PRODOS.
4. Press "R", followed by "2" to go to PRODOS's 2nd block.
5. Press "F", followed by "H" to enter Hex Mode.
6. Move the cursor to byte 71 ($47). You will see three values—AE B3 FB.
7. Press "B" and change the three values to A2 EA EA. Press ESC.
8. Press CONTROL-W, followed by "YES" to write your change to disk.
9. Press "R", followed by "4" to go to PRODOS's 4th block.
10. Move the cursor to byte 6. You should see three values—AE B3 FB.
11. Press "B" and change the three values to A2 EA EA. Press ESC.
12. Press CONTROL-W, followed by "YES" to write your change to disk.
13. Press "R", followed by "5" to go to PRODOS's 5th block.
14. Move the cursor to byte 158 ($9E). The values D0 03 should be there.
15. Press "B" and change the two values to EA EA. Press ESC.
16. Press CONTROL-W, followed by "YES" to write your change to disk.

Your Pro-Byter back-up disk will now boot on a Franklin Ace. In fact, it will boot on any Apple II except the IIc.



40-Column Screen Values

Use this chart to convert 40-column FLASH and INVERSE text message characters to hex or decimal values. For NORMAL, use the ASCII value+128.

INVERSE				FLASH			
0	$00	@	32 $20 sp	64	$40	@	96 $60 sp
1	$01	A	33 $21 !	65	$41	A	97 $61 !
2	$02	B	34 $22 "	66	$42	B	98 $62 "
3	$03	C	35 $23 #	67	$43	C	99 $63 #
4	$04	D	36 $24 $	68	$44	D	100 $64 $
5	$05	E	37 $25 %	69	$45	E	101 $65 %
6	$06	F	38 $26 &	70	$46	F	102 $66 &
7	$07	G	39 $27 '	71	$47	G	103 $67 '
8	$08	H	40 $28 (	72	$48	H	104 $68 (
9	$09	I	41 $29)	73	$49	I	105 $69)
10	$0A	J	42 $2A *	74	$4A	J	106 $6A *
11	$0B	K	43 $2B +	75	$4B	K	107 $6B +
12	$0C	L	44 $2C ,	76	$4C	L	108 $6C ,
13	$0D	M	45 $2D —	77	$4D	M	109 $6D —
14	$0E	N	46 $2E .	78	$4E	N	110 $6E .
15	$0F	O	47 $2F /	79	$4F	O	111 $6F /
16	$10	P	48 $30 0	80	$50	P	112 $70 0
17	$11	Q	49 $31 1	81	$51	Q	113 $71 1
18	$12	R	50 $32 2	82	$52	R	114 $72 2
19	$13	S	51 $33 3	83	$53	S	115 $73 3
20	$14	T	52 $34 4	84	$54	T	116 $74 4
21	$15	U	53 $35 5	85	$55	U	117 $75 5
22	$16	V	54 $36 6	86	$56	V	118 $76 6
23	$17	W	55 $37 7	87	$57	W	119 $77 7
24	$18	X	56 $38 8	88	$58	X	120 $78 8
25	$19	Y	57 $39 9	89	$59	Y	121 $79 9
26	$1A	Z	58 $3A :	90	$5A	Z	122 $7A :
27	$1B	[	59 $3B ;	91	$5B	[	123 $7B ;
28	$1C	\	60 $3C <	92	$5C	\	124 $7C <
29	$1D	]	61 $3D =	93	$5D	]	125 $7D =
30	$1E	^	62 $3E >	94	$5E	^	126 $7E >
31	$1F	_	63 $3F ?	95	$5F	_	127 $7F ?

Pro-Byter Index

Disclaimer of All Warranties and Liabilities

ProDOS™

Other Beagle Bros Apple Software

(WHAT'S NEW? CHECK OUR APPLE MAGAZINE ADS TO SEE.)

■ *GRAPHICS* ■

- ❑ **ALPHA PLOT** (II+, IIe, IIc)† $39.50
  Normal hi-res (6 colors, 280x192 pixels) drawing and typing on both hi-res pages. Compress pictures to 1/3 disk space.
- ❑ **APPLE MECHANIC** (II+, IIe, IIc)† 29.50
  Create hi-res shapes for animation with Applesoft's DRAW & XDRAW commands. Put fancy hi-res type in your programs. List & learn demo programs teach you hi-res programming.
- ❑ **APPLE MECHANIC TYPEFACES**† 20.00
  26 new editable fonts to be used with Apple Mechanic.
- ❑ **BEAGLE GRAPHICS** (IIc or 128K IIe)* 59.95
  Double hi-res drawing (16 colors, 560x192 pixels) and typing in many typestyles (all editable). Color fill, cut & paste, 200+ color mixes. 33 new commands for using double-res in your programs. Convert normal hi-res pictures and programs to double hi-res, compress pix to 1/3 disk space...
- ❑ **FLEX TYPE** (II+, IIe, IIc)† 29.50
  Variable-width text (wide, normal, condensed) controllable with normal Applesoft commands. No 80-column card reqd.
- ❑ **FRAME-UP** (II+, IIe, IIc)† 29.50
  Make Apple "slide shows". Keyboard controlled or unattended, using your existing hi-res, lo-res and text screens.
- ❑ **TRIPLE-DUMP** (II+, IIe, IIc)* 39.95
  Transfer any image, including double hi-res, to your dot matrix printer. Make Giant (8" high characters) Banners too.

■ *ALL-PURPOSE* ■

- ❑ **DISKQUIK** (IIc or 128K IIe)† $29.50
  Acts like half a disk drive in slot 3. Silent and fast as a hard disk. Load/save files in memory with normal commands.
- ❑ **FATCAT** (II+, IIe, IIc)* 34.95
  Reads all of your DOS 3.3 and ProDOS file names into one or more Master Catalogs for sorting, searching and printing. Alphabetize file names on disks. Compare any two files.
- ❑ **PRONTO-DOS** (II+, IIe, IIc)† 29.50
  Triples the speed of loading and saving. New TYPE command displays text file contents. Move DOS for extra 10K.

† Supports DOS 3.3 only
‡ Supports ProDOS™ only
* Supports DOS 3.3 *and* ProDOS™

(Subject to change—See our current ads or catalog.)

■ *PROGRAMMING* ■

- ❑ **BEAGLE BASIC** (IIe, 64K II+)† $34.95
  Puts Applesoft in RAM so you can change it and add enhancements—new commands like if-then-ELSE, SWAP variables, GOTO/GOSUB-a-variable, TONE, HSCRN, etc.
- ❑ **D CODE** (II+, IIe, IIc)* 39.95
  Compact Applesoft programs and reveal unused code. Auto-proofread Applesoft programs, even as you type. Trace any number of program statements *after* stopping a program...
- ❑ **DOS BOSS** (II+, IIe, IIc)† 24.00
  Reword DOS 3.3 commands. Change "Catalog" to "Cat", "Syntax Error" to "Oops" or *anything*. Includes many meaty tips for altering DOS, including program "save-protection".
- ❑ **DOUBLE-TAKE** (II+, IIe, IIc)* 34.95
  2-way scroll for Listings & Catalogs. Better List-format, fast variable+line number display, better renumber/append, auto line-numbering, instant hex/dec converter and more.
- ❑ **EXTRA K** (IIc or 128K IIe)* 39.95
  Use all 128K! Program with variables in auxiliary memory. Have *two* 64K Apples and DOS's in memory. Copy disks in 35 seconds. Store screens in memory; super-speed display...
- ❑ **GPLE** (II+, IIe, IIc)* 49.95
  Edit Applesoft without cursor-tracing. Features insert & delete and fast search & replace. Make all keys be "function keys" to type anything you like (ESC-1 catalogs disk, etc.). Move DOS 3.3 out of main memory to add 10K of space.
- ❑ **PRO-BYTER** (IIe, IIc or 64K II+)‡ 34.95
  Inspect 3.3 and ProDOS disks. Instantaneous sector-to-sector viewing. Search for any byte in a disk or file. Machine language sorter, ProDOS text typer. All new tips & tricks.
- ❑ **SILICON SALAD** (II+, IIe, IIc)† 24.95
  Over 100 utilities and tricks— hi-res program splitter, DOS killer, disk scanner, hi-res text imprinter, 2-track catalog...
- ❑ **TIP DISK #1** (II+, IIe, IIc)† 20.00
  100 tips on disk from Tip Books 1-4. Fascinating Apple programming techniques. Includes Apple Command Chart.
- ❑ **UTILITY CITY** (II+, IIe, IIc)† 29.50
  21 utilities— List-formatter puts each statement on a new line, multi-column catalogs, invisible/trick file names, etc.

■ *GAMES* ■

- ❑ **BEAGLE BAG** (II+, IIe, IIc)† $29.50
  12 games on one disk. Voted to 1983's MOST POPULAR list in *Softalk* poll. The best Apple game bargain on the market.
- ❑ **I. O. SILVER** (II+, IIe, IIc)† $29.95
  Two games in one—a great strategy game and a fast action arcade game. Superb unlocked machine language graphics.

ISBN 0-917085-14-0

Published by BEAGLE BROS MICRO SOFTWARE, INC.
3990 Old Town Avenue, San Diego, California 92110