UNPROTECTED

Backups may be made using standard copying procedures.

COMPATIBLE

Apple® IIc or 128K IIe ProDOS™ *and* DOS 3.3

Extra K

APPLE®EXTENDED MEMORY UTILITY
by ALAN BIRD and MARK SIMONSEN

(Extra K works on any Apple IIc or 128K Apple IIe)

PUT YOUR 128K TO WORK!

Even with 128K of memory, your Applesoft programs, pictures and variables only occupy and access the lower 48K of memory. The EXTRA K disk contains several programs that let you utilize *all* of that 128K muscle.

EXTRA VARIABLES

Your Applesoft programs can now function undisturbed in main memory while all variables and data reside in Apple's massive extra 64K. Everything functions normally—no special commands or procedures are required.

EXTRA-FAST COPIES

Make disk copies fast and "on-the-spot" without re-booting. Extra K's nibble-copier duplicates *and verifies* unprotected disks in 35 seconds with two drives* (instead of 1½ minutes). Make disk copies *fast* without rebooting or affecting utilities in memory.

*Or about one minute if you have only one drive

EXTRA SCREENS

Store all kinds of images in memory and display any one (all or part of it) instantly. Up to seven hi-res pictures or 62 text screens can be stored at once. Hi-res pictures can be called to the screen at a rate of several-per-second, opening up new animation possibilities.

EXTRA APPLE

Extra K lets you have two 64K Apples and programs in memory at once and switch between them at will. You can even have a ProDOS Apple and a DOS 3.3 Apple simultaneously and swap files back and forth without subjecting yourself to Apple's *Convert* program.

EXTRA FEATURES

Compare any two disks, byte-for-byte. Create "dual personality" ProDOS/3.3 disks. Peek and poke auxiliary memory. Keep a live "logbook" of everything that has appeared on your text screen and review it when you want...

INCLUDES FREE PEEKS & POKES CHART

EXTRA K

Extended Memory Utilities

by Mark Simonsen and Alan Bird

ISBN 0-917085-13-2

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

TABLE OF CONTENTS

LICENSING EXTRA K

Software authors can license Extra K for use with software that they sell commercially. The fee is $250 per year per product. Contact Beagle Bros by mail or phone for more details:

BEAGLE BROS, INC.
3990 Old Town Avenue
San Diego, California 92110
619-296-6400

INTRODUCTION

Welcome to Extra K, a powerful set of machine language utilities that gives you access to the extra 64K of "auxiliary memory" available on your Apple IIc or 128K Apple IIe. The memory map on page 4 helps demonstrate the benefits of using Extra K with all of that extra memory you paid for.

IMPORTANT

Before you use any of Extra K's programs, please do three things:

1. Read "GENERAL INFORMATION" on page 6.
2. RUN NOTES from either side of the Extra K disk.
3. Read the instructions for the program you are about to use.

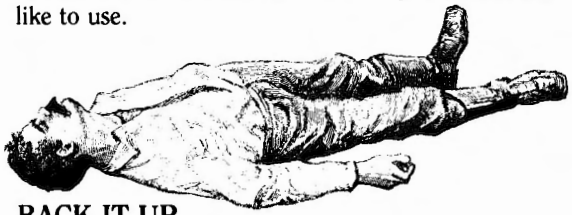
This instruction manual assumes you know the basics of writing Applesoft programs, loading and saving files, dealing with DOS 3.3 and ProDOS, and so on. You may want to purchase Apple's programming manuals to learn more; we highly recommend them. If you're into programming, the price you pay will pay off in the long run.

HARDWARE REQUIREMENTS

Extra K requires an Apple IIc or a 128K Apple IIe (with an *Extended* 80-column card; the standard 80-column card won't do). Some hardware will equip your Apple with more than 128K (256K, for example), but Extra K cannot access more than 128K. Some day maybe, but not now.

DOS 3.3 AND ProDOS™

Two sets of programs are included on your Extra K disk—a ProDOS version on one side and a DOS 3.3 version on the other. Most of the programs work in both DOS's. The DOS you use is up to you; it depends on which one you like to use.

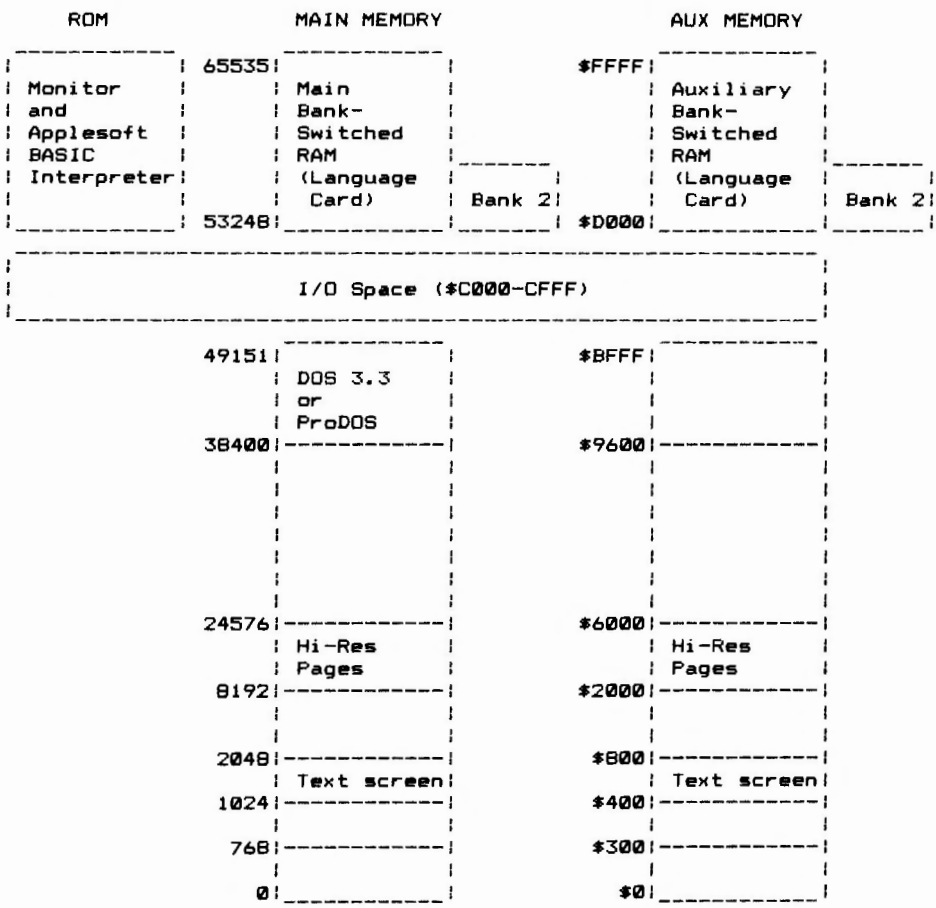


BACK IT UP

In keeping with Beagle Bros' tradition, the Extra K disk is unlocked and unprotected. This means you can copy it, catalog it, modify the programs on it, experiment with it, and even *ruin* it—so make a back-up copy now!

But please don't give copies away to your friends. Think about it—Every illegal copy you see out there is a vote *for* copy protected programs and *against* friendly software. You support us and we'll support you.

Memory Map


ROM
MAIN MEMORY
AUX MEMORY
Monitor and Applesoft BASIC Interpreter
65535
53248
Main Bank-Switched RAM (Language Card)
Bank 2
$FFFF
$D000
Auxiliary Bank-Switched RAM (Language Card)
Bank 2
I/O Space ($C000-CFFF)
49151
$BFFF
DOS 3.3 or ProDOS
38400
$9600
24576
$6000
Hi-Res Pages
8192
$2000
2048
$800
Text screen
1024
$400
768
$300
0
$0

The Extra K Disk Catalog

Here is a list of files that were on each side of the Extra K disk the last time we checked. Type "CATALOG" (DOS 3.3 or 80-column ProDOS) or "CAT" (40-column ProDOS) to compare this list to your disk.

NOTES: *Changes to this manual—maybe. Type "RUN NOTES" to see.*
STARTUP: *Menu program that runs when you boot the Extra K disk.*
TITLE.SCREEN:[1] *Extra K title screen program.*

PRODOS:[2] *ProDOS operating system.*
BASIC.SYSTEM:[2] *Applesoft interface to ProDOS.*
DOS.SYSTEM:[2] *DOS 3.3 file used with hybrid disks.*

AUX.MEM.CHECK:[3] *DOS 3.3 memory checker used by STARTUP.*
DISCONNECT.RAM:[2] *ProDOS /RAM killer.*
DISK.COMPARE:[1] *Program that compares two disks.*
DISK.COPY: *Fast disk duplication program.*
DISK.FORMAT:[1] *ProDOS disk initializer.*
EXTRA.APPLE: *Program that creates two 64K Apples out of one 128K Apple.*
  TRANSFER: *Program that copies files from one EXTRA.APPLE to another.*
EXTRA.SCREENS: *Fast screen load/save/display program.*
  SCREENS.CROP:[1] *Screen cropper for EXTRA.SCREENS.*
  SCREENS.DEMO: *Demo program that shows off EXTRA.SCREENS.*
    DEMO.PIX: *Multi-picture file for SCREENS.DEMO.*
EXTRA.VARIABLES: *Memory expander for Applesoft programs.*
  EXTRA.STORE:[2] *Lets you use Store & Restore with EXTRA.VARIABLES.*
FP:[2] *ProDOS simulation of DOS 3.3's FP command.*
HYBRID.CREATE:[1] *Program for creating combo DOS 3.3/ProDOS disks.*
LOGBOOK:[2] *Program that saves all printed text for instant recall.*
PEEK.AND.POKE: *Program that lets you Peek and Poke auxiliary memory.*
  PEEK.AND.POKE.X: *Version for use with EXTRA.VARIABLES.*
SPOOLER:[2] *Print-while-you-compute printer program (for IIc only).*

[1] Has companion ".M" machine language file
(Do not access the ".M" files directly.)
[2] ProDOS side only
[3] DOS 3.3 side only

General Information

USING EXTRA K PROGRAMS FROM YOUR OWN DISKS

You may want to transfer certain Extra K programs to different disks so you can use them with your programs. This may be done with FID (on the DOS 3.3 System Master disk), FILER (on many of Apple's ProDOS disks) or the SYSTEMS UTILITIES DISK (Apple IIc). Some of the Extra K programs require two files, one Applesoft and one (ending in ".M") machine language, so make sure both get transferred.

RUN OR BRUN?

Extra K programs with an "A" or "BAS" next to them in the catalog should be RUN. Programs with a "B" or "BIN" should be BRUN. Do not try to access the ".M" programs directly.

LOADING PROGRAMS FROM APPLESOFT

Any Extra K program can be Run or Brun from within an Applesoft program by adding one of the following lines to the very beginning:

```
10 PRINT CHR$(4)"BRUN programname"   or...
10 PRINT CHR$(4)"RUN programname"
```

If you're using ProDOS, you'll want to disconnect the RAM disk first (see ProDOS NOTES, next page) by adding:

```
5 PRINT CHR$(4)"BRUN DISCONNECT.RAM"
```

(Booting Extra K automatically disconnects /RAM for you.)

USING EXTRA K PROGRAMS WITH OTHER UTILITIES

You can generally use other utility programs such as DOUBLE-TAKE and GPLE with Extra K programs. Install the utility first, and then try the Extra K program to see if they're compatible. Some Extra K programs have specific instructions on this, but feel free to experiment.

ONE EXTRA K PROGRAM AT A TIME

Since each Extra K program uses all of auxiliary memory, no two of them may be used at the same time. (Some exceptions: EXTRA.VARIABLES and PEEK.AND.POKE.X, SCREENS.CROP and EXTRA.SCREENS, EXTRA.APPLE and TRANSFER.)

The SPOOLER program must be removed by executing the FP program (or by re-booting) before any other Extra K program can be used.

ProDOS NOTES

- To avoid potential problems, the file DISCONNECT.RAM should be executed after booting ProDOS and before any Extra K program is loaded. (Booting the Extra K disk automatically does this for you.)
- To remove an Extra K program and free up the space it occupies in main memory, you need to execute the FP program (page 45).
- ProDOS allows a hyphen "-" to be substituted for Run, Brun or Exec. This command can be used from the keyboard or from within a program.
- AUX.MEM.CHECK is for DOS 3.3 only, but the documentation (page 8) explains how to test memory size from ProDOS.

DOS 3.3 NOTES

The AUX.MEM.CHECK program is available in DOS 3.3 only. The following programs are available in ProDOS only:

DISCONNECT.RAM *(not applicable to DOS 3.3)*
FP *(unnecessary in DOS 3.3; exists as a command)*
LOGBOOK *(incompatible with DOS 3.3)*
SPOOLER *(incompatible with DOS 3.3 and Apple IIe)*

COMMON ERROR MESSAGES

DISK FULL

If you try to save too many files on the Extra K disk, you will soon get this message. See the top of page 6 for a solution.

I/O ERROR

You may be trying to read a DOS 3.3 disk from ProDOS or vice versa, or your disk may be off-center. Try wiggling the drive door slowly as you close it. Also make sure you have a disk in the drive and the drive door is closed.

FILE NOT FOUND (DOS 3.3)

Are all of the necessary programs on the disk? Did you spell the program's name correctly?

PATH NOT FOUND (ProDOS)

The ProDOS equivalent to FILE NOT FOUND. Make sure the prefix is correct by typing "PREFIX,D1" or "PREFIX,D2", according to the drive number. (On an Apple IIc, the internal drive is D1 and the external drive is D2.)

Another way to eliminate prefix problems is to "shut the prefix off" by typing "PREFIX/". Then you may access the disk using ",D1" and ",D2" parameters as you would in DOS 3.3.

AUX.MEM.CHECK (DOS 3.3 only)

DOS 3.3 CHECK FOR 128K

If you are writing Extra K programs that are to be used on Apples other than your own, your programs should check to see what kind of Apple is being used. AUX.MEM.CHECK is a small machine language program that determines whether or not auxiliary memory is present in an Apple using DOS 3.3 (see ProDOS paragraph below). AUX.MEM.CHECK will return the following value in memory location 6:

0 ($00): Not a IIe
32 ($20): IIe but no 80-column card
64 ($40): 64K IIe with 80-column card (no auxiliary memory)
128 ($80): Apple IIc or 128K IIe

After a "BRUN AUX.MEM.CHECK" command, you can PRINT PEEK(6) to check for one of the values above. AUX.MEM.CHECK is used in Extra K's DOS 3.3 STARTUP program to make sure that a 128K Apple is being used. Here's one way you could use it in your programs:

```
10  PRINT  CHR$ (4)"BRUN AUX.MEM.
    CHECK"
20  IF  PEEK (6) < 128 THEN  PRINT
    "INSUFFICIENT MEMORY": END
30  REM  PROGRAM CONTINUES...
```

ProDOS 128K CHECK

With ProDOS, the AUX.MEM.CHECK program itself is not necessary, thanks to ProDOS location 49048 ($BF98). Use the following 128K check:

```
100 A =  PEEK (49048): IF A < 128
    THEN  PRINT "NOT IIE OR IIC
    ": END
110 A = A - 128: IF A >  = 64 THEN
    PRINT "APPLE III": END
120  IF A < 48 THEN  PRINT "NOT 1
    28K": END
130  REM  PROGRAM CONTINUES...
```

DISCONNECT.RAM (ProDOS only)

MAKE ROOM FOR EXTRA K!

When ProDOS boots on a 128K Apple, it automatically creates a "RAM disk" in auxiliary memory with the volume name "/RAM". Because of memory conflicts, /RAM should be removed before any of Extra K's programs are used.

Type "-DISCONNECT.RAM" from the keyboard, or use the following line from within an Applesoft program:

```
10 PRINT CHR$(4)"-DISCONNECT.RAM"
```

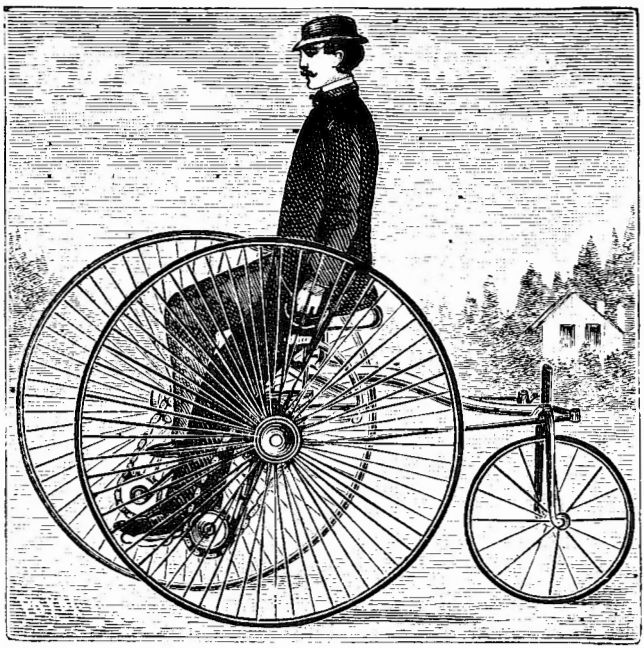
To reconnect /RAM, you will need to reboot.

WHEN TO USE IT

You must execute the DISCONNECT.RAM program *once* after booting ProDOS and before using any of Extra K's programs. Extra K's ProDOS STARTUP program automatically executes DISCONNECT.RAM for you, so you will only need to use DISCONNECT.RAM if you are booting a disk other than Extra K.

WHAT'S A RAM DISK?

A RAM disk is a memory-based disk drive. To prove that ProDOS's RAM disk exists, boot a ProDOS disk other than Extra K, load an Applesoft program (the bigger the better), and type "SAVE/RAM/PROGRAM". Now type "CATALOG/RAM". Now type "NEW" followed by "LOAD/RAM/PROGRAM" and "LIST". You take it from there. Further instructions appear in Apple's ProDOS manuals.



DISK.COMPARE

DISK.COMPARE is (surprise!) a program that compares disks. In only 2 or 3 passes (thanks to your 128K), it will give you a report on each corresponding track and sector (DOS 3.3) or track and block (ProDOS), indicating if they are identical or not.

Use this program to find out if any changes have been made to a recently copied disk by comparing the copy with the master. At the Beagle Building, we make two masters before we send a disk out for mass duplication. If they don't match, we know something's wrong before we start copying.

DISK.COMPARE will work with DOS 3.3, ProDOS, Pascal or CP/M disks. It *won't* work with copy-protected disks, but it will *usually* compare the unprotected disks that they store data on. It will not interfere with Beagle Bros (or most other) utilities in main memory (GPLE, DOUBLE-TAKE, BEAGLE BASIC, etc.), but anything in auxiliary memory will undoubtedly be over-written.

HOW TO USE DISK.COMPARE

Select DISK.COMPARE from the Extra K boot-up menu or type "RUN DISK.COMPARE". The title screen will appear and you will be prompted to enter the slot and drive numbers for disk 1 and 2. (Almost all disk drives are connected to slot 6. If you don't know your slot, it's 6.)

Press RETURN to accept the default values shown on the screen or change them to any legal slot and drive by entering the appropriate number. If you make a mistake, just press ESC to back up. The default drives are 1 and 2. If you don't have two drives, select Drive 1 for both drives. (If you're a programmer, see "CHANGING THE DEFAULT SETTINGS" on page 12.)

DISK.COMPARE OPTIONS

After you've entered the information for the slots and drives, you will be shown the comparison screen with the compare options. Press ESC if you want to go back and change or verify the slot and drive numbers.

Compare Option

Press "C" to start comparing disks. You will then be asked to insert the appropriate disk or disks. At this point you can press RETURN to begin the comparison or ESC to make another selection.

During the comparison, the map on the screen will indicate the state of each block or sector being compared. A dot (".") means the sectors or blocks are identical, and an "X" means they are different by at least one byte.

Dec Option

Press "D" to display the track/sector/block numbers on the border of the disk map in decimal. If an error occurs during a comparison, the slot, drive, track and sector or block will be given in decimal also. (Decimal is the default setting.)

Hex Option

Pressing "H" will change the track/sector/block numbers to hexadecimal. Any errors during a comparison will be indicated in hex as well. To change the default setting to hex, see "CHANGING THE DEFAULT SETTINGS" on page 12.

Print Option

Pressing "P" will print a hard copy of the last disk comparison. You will be prompted to input your printer's slot number (if you don't know, it's probably slot 1). Press RETURN to select the default (slot 1) or enter any legal number, 1-7. If you decide at this point not to print the screen, press ESC to see the previous menu.

If your printer is not on or not connected, you will see the message "TURN ON YOUR PRINTER". The program will wait until you do so. If, however, you have specified the wrong slot for your printer, you may have to press CONTROL-RESET to regain control. Typing "RUN" should then get you back into the program. If it doesn't, type "RUN DISK.COMPARE" (don't forget to put the Extra K disk back in the drive).

Quit

Pressing "Q" will exit DISK.COMPARE and return you to Applesoft. If you quit accidentally, type "RUN" to restart the program.

DISK.COMPARE ERROR MESSAGES

If something happens to go wrong, you will see one of the following error messages, designating the Slot, Drive and Block (or Track/Sector) where the problem occurred:

ERROR: (S#,D#,TR=##,SEC=##), CONTINUE? *(DOS 3.3)*

ERROR: (S#,D#,BLOCK=##), CONTINUE? *(ProDOS)*

An error often occurs if one of the disks is physically damaged or not positioned correctly in the disk drive. Try wiggling the drive door slowly as you close it. If this doesn't solve the problem, you may have a bad disk, or your disk drive needs professional adjustment. Hopefully it's the former.

Press "Y" to continue with the same disk comparison, or "N" to quit. The comparison report for the sector or block that had the error will be unreliable.

CHANGING THE DISK.COMPARE DEFAULT SETTINGS (for programmers only)

You can set DISK.COMPARE's defaults to 1 drive (normal is 2) and/or hex (normal is decimal). Load the DISK.COMPARE program and List it. The default drive is set in line 5 by variable D2. Change "D2=2" to "D2=1" if you don't have two drives. If you prefer hex for your track/sector/block/error displays, change the "GOSUB 10" in line 6 to "GOSUB 12".

If you're any kind of hacker at all, you will find other things you wish to change. Well, DON'T DO IT! Memory usage is critical here.

SAMPLE COMPARISON SCREENS

This is a sample ProDOS comparison of two disks.

```
    (Tracks and blocks in hex)                    111
    00112233445566778899AABBCCDDEEFF001
    0808080808080808080808080808080808080
                        BLOCK
+0  .XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
+1  .XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
+2  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.X
+3  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
+4  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
+5  .XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
+6  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
+7  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                        TRACK
    00000000000000000011111111111111111222
    01234567890ABCDEF01234567890ABCDEF012
```

NOTE: There are eight 512-byte blocks on each track. The block numbers at the top show the cumulative block count.

This is a DOS 3.3 comparison of two identical disks.

(Tracks and sectors in decimal)

```
                  111111111122222222223333
  0123456789012345678901234567890123456789
                      TRACK
 0 ........................................
 1 ........................................
 2 ........................................
 3 ........................................
 4 ........................................
 5S........................................
 6E........................................
 7C........................................
 8T........................................
 9D........................................
10R........................................
11 ........................................
12 ........................................
13 ........................................
14 ........................................
15 ........................................
```

NOTE: There are sixteen 256-byte sectors on each track.

DISK.COPY

DISK.COPY is a fast disk duplication program. It will copy an entire Apple DOS 3.3, ProDOS, Pascal or CP/M disk in less than 35 seconds with two drives or in about one minute (requiring only 3 or 4 disk swaps) with one drive. DISK.COPY will *not* work with copy-protected disks, but it *will* copy the unprotected data disks most of them produce.

You can run DISK.COPY from DOS 3.3 or ProDOS, regardless of the type of disk being copied. Unlike other fast copy programs, PRONTO-DOS, GPLE, DOUBLE-TAKE and other utilities you might have loaded into main memory won't be destroyed. Auxiliary memory is considered fair game and anything residing there will likely be eaten alive.

DISK.COPY is fast because it is a "nibble copier". That is, it copies raw disk data a whole track (16 sectors or 8 blocks) at a time. You can rest assured that the copies you make will be error free, because every single byte is verified after it is written to the duplicate disk.

HOW TO USE DISK.COPY

First of all, when copying disks with DISK.COPY or any Apple copy program, *ALWAYS* COVER THE NOTCH ON YOUR ORIGINAL DISK with a write-protect tab or piece of tape, just in case you insert the wrong disk in the wrong drive at the wrong time.

Select DISK.COPY from the Extra K menu or type "BRUN DISK.COPY". Enter the slot and drive numbers for your original and duplicate disks.

TWO-DRIVES: On most two-drive Apples, you should specify SLOT-6 DRIVE-1 and SLOT-6 DRIVE-2.

ONE-DRIVE: If you have only one drive, specify SLOT-6 DRIVE-1 for *both* the original and duplicate drive.

Press RETURN to accept the default values shown on the screen or change them by typing the appropriate numbers. If you make a mistake or change your mind, press ESC to back up.

Insert the disks and follow the instructions on the screen. If you have one drive, you will be prompted to swap disks three or four times. Thanks to the write-protect tab you put on your original disk, there shouldn't be any problems.

DISK.COPY ERROR MESSAGES

WRITE PROTECTED

This error message will appear if the duplicate disk has no write notch or if the notch is covered with a write-protect tab (maybe you inserted your write-protected original in the wrong drive or at the wrong time). This error will also occur if you punched your own notch in a disk and it's not quite big enough or not positioned exactly right on the disk.

UNABLE TO WRITE (track#)
UNABLE TO READ (track#)

These messages (given with the offending track number) mean that a disk is either physically damaged or not positioned correctly in the disk drive. Or maybe your drive speed is incorrect. To correctly position the disk in the drive, wiggle the drive door slowly when closing it. If this doesn't resolve the problem, you probably have a bad disk; if other disks cause the same error, get your drive checked at your local dealer.

It is possible (but rare) to get an "Unable to Write" error that is caused by an error on the disk being read.

You may be able to fix a damaged DOS 3.3 disk with Quality Software's *Bag of Tricks* disk, available from Apple software stores. The BYTEZAP.PRO program on Beagle Bros' *Pro-Byter* disk will let you examine and change any byte on any unprotected Apple disk. It is not designed to automatically repair disks, however.

DISK.FORMAT

DISK.FORMAT is a program for creating ProDOS disks. It performs the same function as the DOS 3.3 INIT command, except that no files are stored on the disk. It is easier to use than the formatter included with ProDOS because it can be run from Applesoft without disturbing utilities in memory.

Each disk "created" with DISK.FORMAT will be named "/BLANK". You can rename a ProDOS disk with the Rename command. For example, type "RENAME/BLANK,/GOOD.DISK". Don't forget the slashes.

WARNING: FORMATTING ERASES ALL DATA FROM A DISK.
Be sure you don't format the wrong disk!

HOW TO USE DISK.FORMAT

To create ProDOS disks, select DISK.FORMAT from the Extra K menu or type "RUN DISK.FORMAT". You may select any of the three options on the screen by pressing the appropriate letter.

C: Change Slot and Drive

Press "C" to change the current slot and drive numbers. When first Run, DISK.FORMAT automatically displays the values for the most recent drive used.

F: Format Disk

Press "F" to format the disk in the specified drive. (Press ESC if you change your mind.) You will be prompted to insert the desired disk before formatting occurs. Press RETURN to proceed. Remember, all existing data on the disk will be lost!

Q: Quit

Press "Q" to exit to Applesoft.

DISK.FORMAT ERROR MESSAGES

In the unlikely event that something goes wrong while a disk is being formatted, you will see one of the following self-explanatory error messages:

UNABLE TO FORMAT
WRITE-PROTECTED
DRIVE TOO SLOW
DRIVE TOO FAST

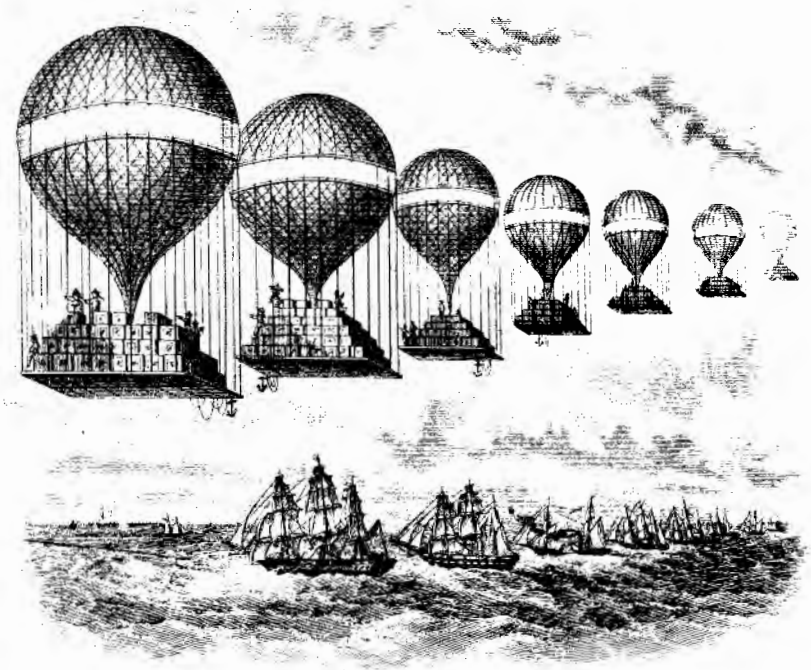
An "Unable to Format" error may occur if a disk is physically damaged or not positioned correctly in the disk drive. To correctly position the disk, slowly wiggle the drive door as you close it. If this doesn't solve the problem, you either have a bad disk or your disk drive needs professional adjustment (new disks are cheaper).

CREATING A BOOTABLE DISK

DISK.FORMAT creates a ProDOS "data disk" that can be used for everything except booting. Leave it that way and you will gain a lot of disk space. Or to make a bootable ProDOS disk, follow these steps:

1. Format a disk with DISK.FORMAT.
2. Type "RENAME /BLANK,/BOOT.DISK" (or any legal name).
3. Use one of Apple's programs like FILER or the System Utilites disk to copy the two files PRODOS and BASIC.SYSTEM from the Extra K disk.
4. Type "NEW" to clear memory.
5. Type "10 PRINT CHR$(4)"CAT"" or write any Applesoft STARTUP program.
6. Type "SAVE STARTUP", adding a third file to the disk.

Press CONTROL-RESET while holding the open-Apple key down. Your disk should boot, load PRODOS and BASIC.SYSTEM, and then catalog itself (or do whatever your STARTUP program tells it to do).



EXTRA.APPLE

EXTRA.APPLE is a program that makes your 128K Apple IIe or IIc work like two separate 64K computers. They don't even have to be running under the same DOS. You can switch back and forth between these two Apples in about three seconds. Everything (including data, programs, pictures, variables, registers, and so on) is left unaltered. There is no "concurrent processing"; only one program can be running at a time. Each 64K machine, when selected, has the undivided attention of the microprocessor.

EXTRA.APPLE can be used with any unprotected program or package that does not use auxiliary memory and doesn't have to be booted to be used.

USING EXTRA.APPLE

The simplest way to install EXTRA.APPLE is to type "BRUN EXTRA.APPLE". After it loads, type "CALL 600". This Call will copy everything in main memory over to auxiliary memory. Now you are using APPLE1 (64K) in main memory, with APPLE2 (64K) hidden in auxiliary memory. You can switch back and forth between APPLE1 and APPLE2 by typing "CALL 769". Switching may be done in immediate mode or from within a running program.

ALTERNATE SWITCHING METHODS

There are three different ways to switch between APPLE1 and APPLE2. Use the method that works best with the programs you are using.

CALL 769 is the preferred way to switch between APPLE1 and APPLE2. It works when EXTRA.APPLE is loaded with this procedure:

- BRUN EXTRA.APPLE
- CALL 600

& will make the switch if you load EXTRA.APPLE with this procedure:

- BRUN EXTRA.APPLE
- CALL 733
- CALL 600

Don't use the & method with other programs that also use the ampersand vector (for example, don't use "&" with DOUBLE-TAKE).

CONTROL-Z will make the switch if you load with this procedure:

- BRUN EXTRA.APPLE
- CALL 744
- CALL 600

Don't use the CONTROL-Z method with other programs that also use the input hook (for example, don't use CONTROL-Z with GPLE).

WHICH APPLE IS WHICH?

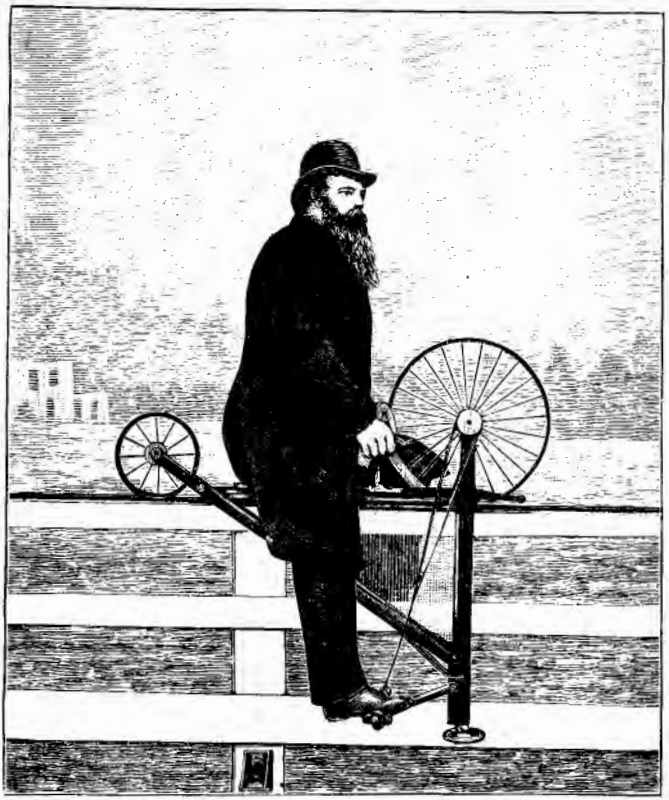
To find out which Apple you are currently using, type "PRINT PEEK(768)". The answer (1 or 2) will appear on the screen.

EXTRA.APPLE COMMAND SUMMARY

BLOAD EXTRA.APPLE (loads the program)
CALL 733 (optional—sets up ampersand (&) to do the switching)
CALL 744 (optional—sets up CONTROL-Z to do the switching)
CALL 600 (use only once—copies main memory to auxiliary)

CALL 769 (switches between APPLE1 and APPLE2)
& (switches between Apples if you did a CALL 733 before the CALL 600)
CONTROL-Z (switches if you did a CALL 744 before the CALL 600)
PRINT PEEK(768) (indicates which APPLE is active)

(more—turn the page)



DIFFERENT DOS'S FOR YOUR TWO APPLES

When you first install EXTRA.APPLE (previous page), you will have the same DOS (ProDOS or DOS 3.3; depending on which one you were using) in both APPLE1 and APPLE2.

To set up EXTRA.APPLE with two different DOS's, follow these steps carefully:

1. Make a hybrid (double-DOS) disk using HYBRID.CREATE (page 46).
2. Create a dual-boot disk (page 47).
3. Add the following three files to the ProDOS half of your hybrid disk:
   - DISCONNECT.RAM (from the ProDOS side of the Extra K disk)
   - EXTRA.APPLE (from the ProDOS side of the Extra K disk)
   - The following ProDOS STARTUP program:

```
10  PRINT  CHR$ (4)"-DISCONNECT.R
    AM": REM free aux memory
20  PRINT  CHR$ (4)"-EXTRA.APPLE"
    : REM load routines
30  REM set-up ampersand w/ CALL
    733 and/or CONTROL-Z w/ CALL
    744
40  CALL 600: REM set-up APPLE2 b
    y copying main mem to aux me
    m
50  IF  PEEK (768) = 1 THEN  PRINT
    CHR$ (4)"-DOS.SYSTEM": REM
    boot DOS 3.3 and run HELLO p
    rogram
60  HOME : PRINT  CHR$ (4)"CAT": REM
    or anything you want
```

4. Add the following files to the DOS 3.3 half of your hybrid disk:
   - EXTRA.APPLE (from the DOS 3.3 side of the Extra K disk)
   - The following DOS 3.3 HELLO program:

```
10  PRINT  CHR$ (4)"BRUN EXTRA.AP
    PLE": REM load routines
20  REM set-up ampersand w/ CALL
    733 and/or CONTROL-Z w/ CALL
    744
30  HOME : PRINT  CHR$ (4)"CATALO
    G": REM or anything you want
```

Your hybrid disk should now have these two catalogs:
ProDOS Half

```
/DOUBLE.DOS.DISK

 NAME                 TYPE   BLOCKS   MODIFIED

*PRODOS               SYS        30   15-MAY-85
*BASIC.SYSTEM         SYS        21   15-MAY-85
*DOS.SYSTEM           SYS        19   15-MAY-85
*STARTUP              BAS         1   15-MAY-85
*DISCONNECT.RAM       BIN         1   15-MAY-85
*EXTRA.APPLE          BIN         1   15-MAY-85

BLOCKS FREE:    56      BLOCKS USED:   224
```

DOS 3.3 Half

```
DISK VOLUME 254

*002 A HELLO
*002 B EXTRA.APPLE
```

When you boot this disk, ProDOS is loaded into main memory (APPLE1) and STARTUP is run. The CALL 600 in line 40 sets up a ProDOS APPLE2 in auxiliary memory and puts an exact copy of the running STARTUP program there. Line 50 loads DOS 3.3 into main memory (APPLE1) and HELLO is run. When HELLO finishes, you will be left in immediate mode in APPLE1.

When you switch to APPLE2, STARTUP will continue running right after the CALL 600 (the point where APPLE2 execution last stopped). The IF statement in line 50 won't be executed and the program will continue.

The STARTUP and HELLO programs on the previous page are fairly flexible. They can be used as is, or they can be altered so they run other programs. It's up to you.

40/80-COLUMN PROBLEMS

As a rule, you should use both APPLE1 and APPLE2 in the same screen mode (40 or 80-columns). For example, if you are going to be using the 80-column screen, you should turn it on (with PR#3) before doing the CALL 600. Then stay in 80-columns and switch Apples whenever you like. It's o.k. to change to 40-columns; just be sure you *change back* to 80-columns *before* switching Apples. Failure to do so could wipe out the Applesoft program in the Apple you are switching to.

You will also need to stay in the same screen mode if you are using the CONTROL-Z method of switching Apples. This method uses the input hooks, but they are reset whenever the screen mode is changed.

RECONNECTING EXTRA.APPLE

If EXTRA.APPLE becomes disconnected (you'll know because the switching commands won't work), you can re-connect it in several ways:

IF YOU'RE USING CALL 769 FOR SWITCHING APPLES:

Type "BRUN EXTRA.APPLE". *Don't* CALL 600.

IF YOU'RE USING "&" FOR SWITCHING APPLES:

POKE 1013,76, POKE 1014,1 and POKE 1015,3 and try switching Apples. If this doesn't work, BRUN EXTRA.APPLE and CALL 733. *Don't* CALL 600.

IF YOU'RE USING CONTROL-Z FOR SWITCHING APPLES:

Under DOS 3.3: CALL 963 (or CONTROL-RESET if you're in 40-columns) and try switching Apples. If this doesn't work, BRUN EXTRA.APPLE and CALL 744.

Under ProDOS: POKE 48690,171 and POKE 48691,3 (or CONTROL-RESET if you're in 40-columns) and try switching Apples. If this doesn't work, BRUN EXTRA.APPLE and CALL 744.

USING EXTRA.APPLE WITH OTHER UTILITIES

EXTRA.APPLE works with programs which don't use auxiliary memory or page 3 (from $300-$3CF). Very early versions of DOUBLE-TAKE used page 3. If you have an old version you can get an update. Send your original disk and $10 to Beagle Bros.

If you're using the same DOS in both APPLE1 and APPLE2, you can run utilities like GPLE and DOUBLE-TAKE before you load EXTRA.APPLE. After you copy main memory to auxiliary memory (CALL 600) the utilities will work in both Apples. With a dual DOS setup, you'll have to load the utilities separately for each Apple.

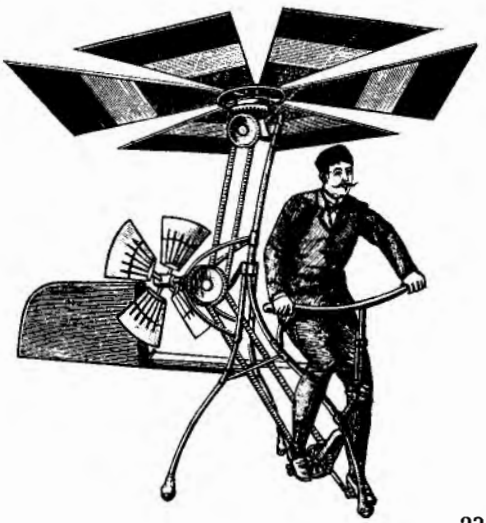
If you should accidentally overwrite EXTRA.APPLE (it resides in page 3 from $300 to around $3CF), you can simply BRUN EXTRA.APPLE the next time you want to switch, and no harm will be done. If you're using APPLE2 when this happens, POKE 768,2.

TECHNICAL NOTES

■ The active APPLE is always in main memory. CALL 769 swaps auxiliary and main memory.

■ EXTRA.APPLE uses zero-page memory locations $3C, $3D, $45, $46, $47, $48, and $49. It only uses them as temporaries (as do DOS, Applesoft and the Monitor). Your program may use these areas so long as they are NOT used for permanent storage of any variables.

■ EXTRA.APPLE uses part of the input buffer ($257-$2FF) for preliminary set-up routines, but all of the important switching code lies in the range $300-$3CF. After EXTRA.APPLE is loaded and the initial calls are made, the code in the input buffer is ignored.



TRANSFER

(Read instructions for EXTRA.APPLE first.)

TRANSFER is a program for quickly copying Applesoft programs or binary files (not text files) between APPLE1 and APPLE2.

TRANSFER is especially useful when using DOS 3.3 in one Apple and ProDOS in the other. Instead of hassling with Apple's CONVERT program, you can transfer a file from one DOS to another in a few seconds and immediately save it on disk or test it for compatibility.

USING TRANSFER

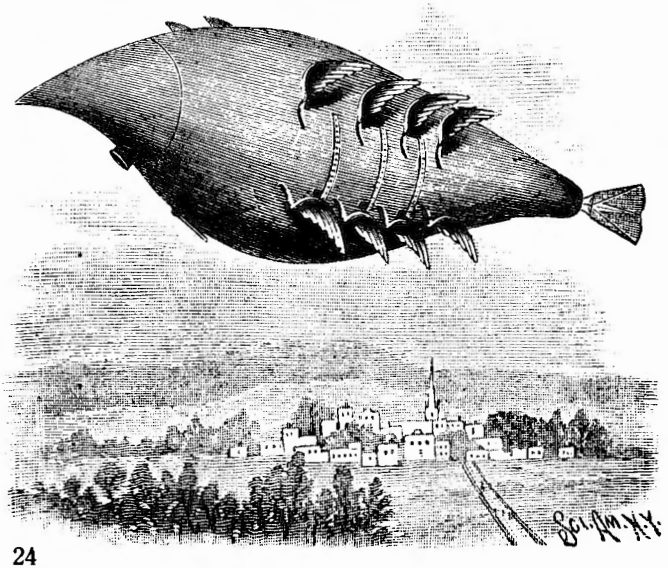
TRANSFER may be used from APPLE1 or APPLE2, as long as it is present on the current disk. Just type "BRUN TRANSFER" to install it above HIMEM. You may exit TRANSFER by pressing ESC. To rerun it, type "CALL 10".

TRANSFERRING APPLESOFT PROGRAMS

Press "A" to duplicate the Applesoft program currently in active memory. When TRANSFER is done, you can switch to the other APPLE to save and test the program. Be careful when testing programs written for a different DOS; it's usually a good idea to save on disk before testing.

NOTE: Do not use TRANSFER if you switched from a running program to the current Apple.

NOTE: When you BRUN TRANSFER and the Applesoft program in memory is extremely long, you may have to exit TRANSFER and reload the Applesoft program. (Use CALL 10 to re-enter TRANSFER.)



TRANSFERRING BINARY FILES

Press "B" to load a binary file from disk into the current Apple, and then copy it to the other Apple. You will be prompted for the file name. Type the name and press RETURN.

You may also specify the address where the file is to be loaded. For example, you could enter "TESTFILE,A$5000" (in case TESTFILE normally loads high in memory and might overwrite TRANSFER).

The file will be loaded and the hex values for the file's starting address (A$), length (L$), and ending address (E$) will be printed on the screen as it is copied to the other APPLE. Use these values to save the file under the new DOS. For example, if TRANSFER prints "A$6000,L$100,E$60FF" you can save the file as shown:

First:
Any DOS: Type "CALL 769" to switch to the other APPLE

Then:
DOS 3.3: Type "BSAVE FILE,A$6000,L$100"
ProDOS: Type "BSAVE FILE,A$6000,L$100"
or "BSAVE FILE,A$6000,E$60FF"

LIMITATIONS

You cannot directly transfer binary files which load above HIMEM. If the computer locks up, you probably loaded a file over the TRANSFER routine. Try pressing CONTROL-RESET to get back to Applesoft. If that doesn't work, you may have to reboot.

Get TRANSFER up and running again, but this time specify a "safe" address for loading the file, as described under "TRANSFERRING BINARY FILES." Save the file as usual. If it is a non-relocatable program, you will have to manually reload the file (from the new DOS) at the correct address and save it again.

If a binary file is loaded over any part of the EXTRA.APPLE routine ($300.$3CF), you must transfer the file, then BRUN EXTRA.APPLE before you switch Apples (POKE 768,2 if you're in APPLE2). Once you've switched, save the tranferred binary file and BRUN EXTRA.APPLE again (POKE 768,2 if you're in APPLE2).

EXTRA.SCREENS

EXTRA.SCREENS is a program which allows you to rapidly display full or partial text and graphics screens. A large number of screens may be loaded from disk and stored in auxiliary memory. From there, any screen may almost-instantly be brought into view.

RUN SCREENS.DEMO—THEN LIST IT.

High-speed graphics animation is illustrated by the SCREENS.DEMO program, selectable from the Extra K boot-up menu, or by typing: "RUN SCREENS.DEMO". Run the demo now. It is written in Applesoft and listable so you can see how it works.

HOW MANY, HOW FAST?

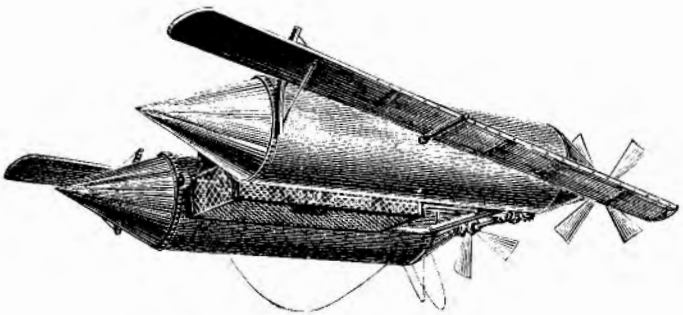
The following table shows how fast full screens may be loaded from auxiliary memory. It also specifies the maximum number of screens that may be stored at one time. (Partial screen saves will vary according to size.)

Screen Type	*Screens/Sec.*	*Max.Screens**
40-COLUMN TEXT or LO-RES GRAPHICS	68	62
80-COLUMN TEXT or DOUBLE LO-RES GRAPHICS	38	31
HI-RES GRAPHICS	11	7
DOUBLE HI-RES GRAPHICS	5	3

** Assumes that only one screen type is used*

INSTALLING EXTRA.SCREENS

To add the EXTRA.SCREENS enhancement to Applesoft, select it from the Extra K boot-up menu or type "BRUN EXTRA.SCREENS". (See GENERAL INFORMATION on page 6 for more details.)



EXTRA.SCREENS COMMANDS

EXTRA.SCREENS adds nine "ampersand commands" (commands preceded by the character "&") to Applesoft:

&SAVE... *lets you save a screen into auxiliary memory.*
&LOAD... *lets you load a screen from auxiliary memory into view.*

&RECALL... *lets you load a screen file from disk into auxiliary memory.*
&STORE... *lets you save all auxiliary memory screens to a file on disk.*

&LIST *lets you "catalog" the screens currently in auxiliary memory.*
&CALL... *lets you rename any screen in auxiliary memory.*
&DEL... *lets you erase an individual screen from auxiliary memory.*
&CLEAR *lets you erase all screens from auxiliary memory.*
&FRE *lets you see how much auxiliary memory is free.*

After EXTRA.SCREENS is installed, these commands are available from the keyboard (immediate mode) or from your Applesoft programs (deferred mode). Each new command is described on the following pages with the correct syntax, a description of the command's function, any options, and examples of how to use the command.

You should Run and LIST THE SCREENS.DEMO PROGRAM to learn more. That's one of the **big benefits of unprotected software—please take advantage!**

DEFINITIONS

Syntax: The correct way to use an EXTRA.SCREENS command. The following rules apply:

- Anything in square brackets [ ] is optional.
- All upper case words must be entered exactly as listed.
- Lower case *italic* items are supplied by you.

String Expression: A string variable like *A$*, a string function like *LEFT$(NAME$,8)* or *NAME$+STR$(I)* or a literal like *"SCREEN 1"* or "55".

Numeric Expression: A variable like *XX*, a function like *2+7* or *X/5*, or a literal like *65535*.

&CALL

SYNTAX: &CALL *oldname, newname*

FUNCTION: Rename a screen in memory. Works similar to the DOS *Rename* command.
oldname and ***newname*** must be string expressions.

EXAMPLES:

&CALL "DOG", "FLEABAG"
Rename the screen DOG, FLEABAG.

&CALL A$, XX$
Change the name of screen A$ to XX$. A$ must be the name of an existing screen. XX$ can be any string 8 characters or less.

10 FOR I=0 TO 9 : &CALL NAME$(I),STR$(I): NEXT
The ten screens named by NAME$(I) will be renamed as numbers from 0 to 9. (Note: These are *string* numbers, not numerics.)

&CLEAR

SYNTAX: &CLEAR [,D]

FUNCTION: Clear all existing screens from auxiliary memory.
Option [,D] reserves the portion of auxiliary memory needed by double hi-res graphics ($2000-$3FFF). You *must* use &CLEAR,D before saving *any* screens if you will be using double hi-res.

EXAMPLES:

&CLEAR,D
Delete all stored screens and reserve double hi-res memory.

10 HOME: &CLEAR : PRINT "EXTRA SCREENS ERASED"
Clear the text screen and delete all stored screens.

&DEL

SYNTAX: &DEL *name*

FUNCTION: Delete an unwanted screen from memory. Works similar to the DOS *Delete* command.
name must be a string expression equivalent to an existing screen name.

EXAMPLES:

&DEL "SCREEN 1"
Erase the screen named "SCREEN 1" from memory.

&FRE

SYNTAX: &FRE,*x*

FUNCTION: Determine the number of free pages left in auxiliary memory for saving screens. The number is assigned to *x*, which must be a numeric variable.

There are a maximum of 248 "pages" of memory available (256 bytes per page) in auxiliary memory. If double hi-res is used, you will be limited to 216 pages. The following table illustrates the number of memory pages required by each type of full screen. Memory used by partial screens will depend on how much of the screen is saved.

Screen Type	*Pages Used*
40-Column Text or Lo-res Graphics	4
80-Column Text or Double Lo-res Graphics	8
Hi-Res Graphics	32
Double Hi-res Graphics	64

EXAMPLES:

&FRE,A: ?A
Print the number of free memory pages.
You could PRINT A/32 to see how many hi-res pictures can be added.
Or PRINT A/4 for Text/Lo-Res, etc. (see chart above).

10 &FRE,F : IF F < 32 THEN PRINT "NO MORE ROOM"
Check to see if there's room for at least one hi-res picture in auxiliary memory.

&LIST

SYNTAX: &LIST

FUNCTION: Catalog all screens stored in auxiliary memory.
Type CONTROL-S to pause a long listing or CONTROL-C to stop it.

EXAMPLE:

&LIST could produce the following display:

```
NAME        TYPE  $ADDR  PAGES  COORDINATES
----------  ----  -----  -----  -----------
FULL 40A    F     0800   4
FULL 40B    F     0C00   4
HIRES 1     H     1000   5      10,50,15,75
HIRES 2     H     1500   8      1,1,20,100
FULL HI     H     1D00   32
40 PART     F     3D00   1      10,9,10,7

TYPE: F  PAGE: 1  FREE MEMORY PAGES: 194
```

Each column of information is explained below:

NAME: The name assigned during &SAVE, unless changed by &CALL.

TYPE: A single character code for the screen type:

- D: Double hi-res graphics
- E: Eighty column text
- F: Forty column text
- H: Hi-res graphics
- L: Lo-res graphics
- M: Medium-res (double lo-res) graphics

$ADDR: The hex location in auxiliary memory where the screen is stored (useless except to a select few fanatics).

PAGES: The amount of memory (in 256-byte pages) that the picture occupies.

COORDINATES: The x and y coordinates and width and height values of the screen. Given for partial screens only (see &SAVE).

STATS: The bottom line of the listing gives the current type, current page (see &SAVE and &LOAD), and the number of free memory pages (see &FRE).

&LOAD

SYNTAX: & [AT *page,*] LOAD *name*[,V] [,GET *type*] [,*x,y*]

FUNCTION: Load a screen from auxiliary memory.
name must be a string expression naming a previously saved screen.

OPTIONS:

[AT *page,*] indicates where to load the screen.
page must be a numeric expression equalling 1 or 2. Applies to 40-column text, single lo-res and single hi-res graphics only. May be used alone to set the default page (&AT1 or &AT2).

[,V] (effective on Apple IIe only) may be used to reduce the flicker that sometimes occurs when loading in several consecutive text or lo-res screens. This is accomplished by delaying loading of the screen until vertical blanking occurs. (The video screen is rewritten 60 times per second; vertical blanking is the short period of time between screen rewrites.)

[,GET *type*] identifies the screen type which was just loaded.
type is either a numeric or string variable that will be set equal to the screen type (types are listed under &SAVE on page 32).

[,*x,y*] are the horizontal and vertical coordinates for partial screens. If no *x* or *y* coordinates are specified, the screen will load in at its original location (see &SAVE). The entire screen portion which was saved must be loaded.

EXAMPLES:

&LOAD "HELP",10,5
Load the partial screen named "HELP". The upper-left corner will be the equivalent of Htab 10, Vtab 5.

&LOAD "DOG",GET A
Load the screen "DOG". The variable A will be set equal to the screen type, according to the list on page 30.

10 FOR I= 0 TO 9 : &LOAD A$(I),V : NEXT
Quickly load the ten screens named by A$( ). Reduce IIe flicker with ",V".

Here's how you can use the &AT command to access screens without having to watch them load:

```
100 HGR: REM Clear and show hi-res page one.
110 &AT 2, LOAD "CLOWN": REM Load hi-res pic into hi-res page 2
120 POKE 49237,0: REM Reveal hi-res page 2
130 &AT 1,LOAD "BALLOON": REM Loads hi-res pic into hi-res page 1
140 POKE 49236,0: REM Reveal hi-res page 1.
```

&RECALL

SYNTAX: &RECALL *filename* [,S *x*] [,D *x*] [,B *x*]

FUNCTION: Load a previously stored file of screens from disk. *filename* must be a string expression naming an existing file.

OPTIONS: Identical to the &STORE command (page 34).

&SAVE

SYNTAX: & [*type,*] [AT *page,*] SAVE *name* [*,x,y,width,height*]

FUNCTION: Save a screen to auxiliary memory.
name must be a string expression 8 CHARACTERS OR LESS

As with disk files, using an existing name will cause the old screen to be overwritten. You may not save screens of different types with the same name, and regardless of type or size, no more than one hundred screens can be saved.

If you use an existing name to resave a partial screen, you *must* specify the same width and height as in the previous save.

OPTIONS:

[*"type",*] is a single character code which is required if the screen to save is different than the current type (see &LIST, page 30). *type* can be a string or numeric expression corresponding to one of the following:

"F" or 1: Forty column text
"L" or 2: Lo-res graphics
"H" or 3: Hi-res graphics
"E" or 4: Eighty column text
"M" or 5: Medium-res (double lo-res) graphics
"D" or 6: Double hi-res graphics

&*type* may be used alone to set the default type. For example, if you type &"H" or &3, the next &SAVE will assume that the screen type is hi-res.

[AT *page,*] indicates which screen page to save.

page must be a numeric expression equalling 1 or 2. Applies to 40-column text, single lo-res and single hi-res graphics only (see examples below). &AT1 or &AT2 may also be used alone to set the default page.

[*,x,y,width,height*] are numeric coordinates for partial screen saves.

Note: Cropping may be more-easily done by using the SCREENS.CROP program. The following information is for advanced programmers only.

x: The horizontal position of the upper left corner of the area to be saved. *x* must be a number 1-40 for 40-column text, lo-res graphics and hi-res graphics; it must be a number 1-80 for 80-column text, double lo-res graphics, and double hi-res graphics. For hi-res and double hi-res graphics, the X value used in commands such as HPLOT X,Y must be divided by 7 and then increased by 1 to get the correct *x* coordinate position (because hi-res pictures are saved on 7-pixel byte boundaries).

y: The vertical position of the upper left corner of the area to be saved. For text or lo-res graphics this should be a number 1-24; for both types of hi-res graphics, the value of *y* should be 1-192. The y coordinate is the same as the Y value used for graphics commands such as HPLOT X,Y.

width: The horizontal width of the area to be saved. The range for width is the same as for the *x* coordinate.

height: The vertical height of the area to be saved. The range for height is the same as for the *y* coordinate.

EXAMPLES:

& "H", AT2, SAVE "PICTURE2"
Save the entire hi-res picture on hi-res page 2 in auxiliary memory.

10 FOR I=1 TO 5 : &"F", SAVE W$+STR$(I),5,I*2,20,10 : NEXT
Save 5 identically-sized overlapping text windows.

FREE TIP: For an ultra-fast HGR, save a black screen like this:
10 HGR: &"H", AT1, SAVE "HGR"

Now clear the screen like this:
10 &LOAD "HGR"
20 POKE 49232,0: POKE 49234,0: POKE 49236,0: POKE 49239,0

&STORE

SYNTAX: &STORE *filename* [,S*x*] [,D*x*] [,B*x*]

FUNCTION: Save all of the screens stored in auxiliary memory to disk. ***filename*** must be a string expression equivalent to a legal disk file name. If the file already exists, the old one will be overwritten. (ProDOS note: If a ProDOS prefix has been specified, it will be used.)

OPTIONS:

[,S *x*] A slot other than the default may be specified. *x* must be a numeric 1-7. Under ProDOS, this option will override the prefix.

[,D *x*] A drive other than the default may be specified. *x* must be a numeric equalling 1 or 2. Under ProDOS, this option will override the prefix.

[,B *x*] A main memory buffer other than the default (buffer 1) may be specified for temporary storage while saving the screens. *x* must be a numeric 1-3.

Buffer 1: $2000-$3FFF (8192-16383) hi-res page 1
Buffer 2: $4000-$5FFF (16384-24575) hi-res page 2
Buffer 3: $6000-$7FFF (24576-32767)

Note: Everything in the buffer will be overwritten by &STORE. Use buffer 2 to protect hi-res page 1 or long Applesoft programs. Use buffer 3 to protect hi-res buffer 2, but not overwrite your program strings or variables.

EXAMPLES:

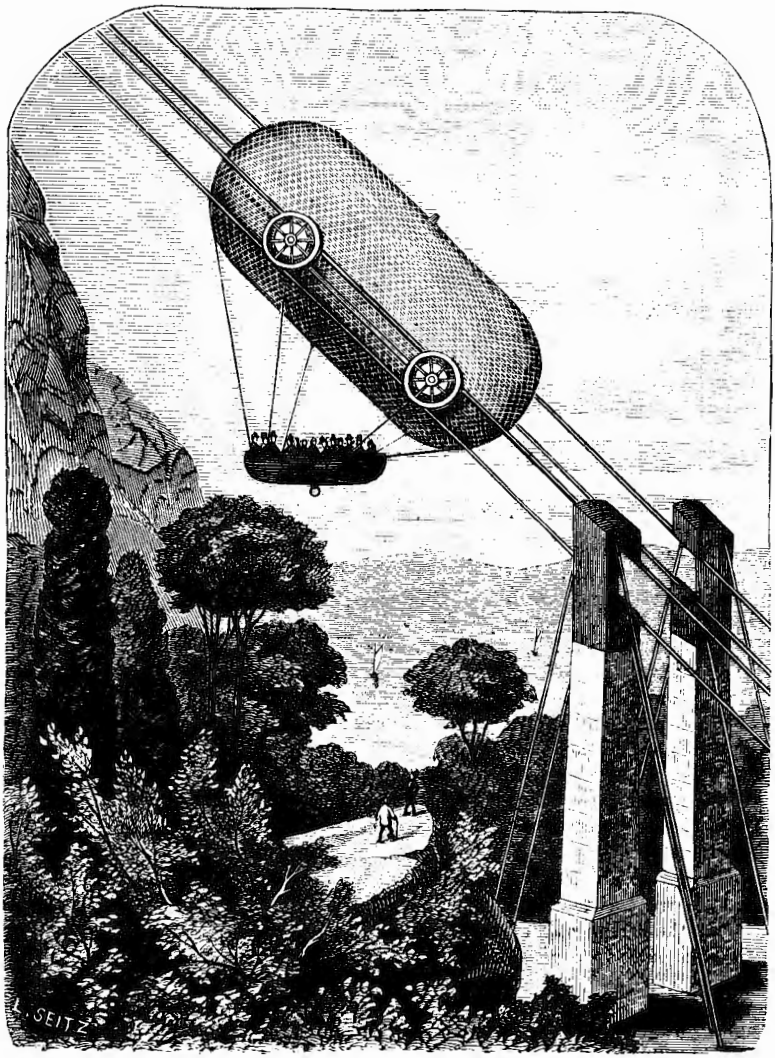
&STORE "SAMPLE.PICTURES" *(Notice the quote marks.)*
Saves all screens in memory as one file called "SAMPLE.PICTURES".

&STORE N$,D2,B3
Use buffer 3 ($6000-7FFF) as temporary storage while saving the screens to drive 2 in a file named by N$.

&STORE "PICTURES/LORES" *(ProDOS only)*
Saves all screens in a file named "LORES" in the "PICTURES" subdirectory of the current directory.

Removing EXTRA.SCREENS

Type "FP" (under DOS 3.3) or "-FP" (under ProDOS with Extra K disk in drive) to get rid of the EXTRA.SCREENS program. Any screens stored in auxiliary memory will be lost.



EXTRA.SCREENS Error Messages

EXTRA.SCREENS has a set of error messages that are described on the next page. Normally, if an error occurs, the message will be printed on the screen and any program that is functioning will stop.

TRAPPING ERRORS

When an error occurs, its code number (shown in parentheses) is stored in memory at location 222. You can use Applesoft's ONERR function to trap errors by Peeking at 222. Here is a sample program that catches File Not Found errors and catalogs the screens in memory for you. Without ONERR, the program would stop when the error occurred.

```
1   ONERR  GOTO 100: REM  set ONER
      R trap
10  INPUT "Enter screen to load:"
      ;N$
20  &  LOAD N$: GOTO 200: REM  Go
      es to 200 if &LOAD is succes
      sful
100 EN =  PEEK (222): REM  Find o
      ut what error occured
110  IF EN = 38 THEN  HOME : PRINT
      "NAME NOT FOUND. TRY AGAIN."
      : &  LIST : CALL -3288: GOTO 10
120  POKE 216,0: RESUME : REM  Di
      sable ONERR so message other
       than error # is printed
200  REM  No errors: program cont
      inues...
```

If a nonexistent or misspelled screen is specifed in line 10, the ONERR routine sends the program to line 100. If the error number (EN) is 38, the screen named by N$ does not exist, so all stored files are listed. The program then returns to line 10 to try again. If some other error has occurred, line 120 resets normal error-handling.

FREE TIP: If you want to know the line number where an error occurred, just add the following:

```
105 LINE=PEEK(218)+PEEK(219)*256
```

ERROR MESSAGE (Code)

DISK FULL (9) Not enough room on disk to &STORE this file.

DISK I/O ERROR (8) Bad disk or wrong disk format (maybe you're using ProDOS with a DOS 3.3 disk, or vice versa).

FILE LOCKED (10) Type "UNLOCK filename" and then STORE the screens.

FILE NOT FOUND (6) The disk file named does not exist, or was misspelled. ProDOS note: the prefix may be wrong. Type "PREFIX,D1" or "PREFIX,D2" (drive 1 or drive 2) and try again.

ILLEGAL NAME (63) Screen name used was more than 8 characters long, or no name was given.

?ILLEGAL QUANTITY (53) An illegal value was specified when attempting to load or save a partial screen. Values were used that would place all or part of the picture off the screen.

ILLEGAL TYPE (75) You are trying to re-save a screen that is not the same type as the current type (see type under &SAVE on page 32).

MEMORY FULL (52) Either there is insufficient memory to save another screen, or 100 screens have already been saved.

NAME NOT FOUND (38) The screen name supplied for a &LOAD, &DEL or &CALL does not exist.

PROGRAM CONFLICT (161) You attempted to load a screen over Applesoft program or variable space. This often occurs if you attempt to load a page 2 text or lo-res screen.

Note: Text page 2 is at $800-$BFF, and programs usually start at $801. Add the following first program line to run programs above text page 2:

```
1 IF PEEK(104)<>12 THEN POKE 3072,0 : POKE 103,1 : POKE 104,12 :
  PRINT CHR$(4)"RUN THISPROGRAM"
```

?SYNTAX ERROR (16) Check the spelling of the command and make sure all values are used correctly.

?TYPE MISMATCH (13) The command was expecting a string and a numeric was given or vice versa. For example: &DEL PICTURE (no quotes around "PICTURE"). PICTURE would be evaluated as a numeric expression instead of a string name.

WRITE PROTECTED (4) Uncover the disk's write-protect notch.

SCREENS.CROP (Read EXTRA.SCREENS instructions first)

SCREENS.CROP is a program for cropping EXTRA.SCREENS text or graphics screens. Run and List SCREENS.DEMO for some good examples of cropped screens.

To begin, type "RUN SCREENS.CROP". Existing screens in auxiliary memory will not be disturbed. Press ESC to return to Applesoft, or select one of the options by pressing the appropriate number from the menu on the screen:

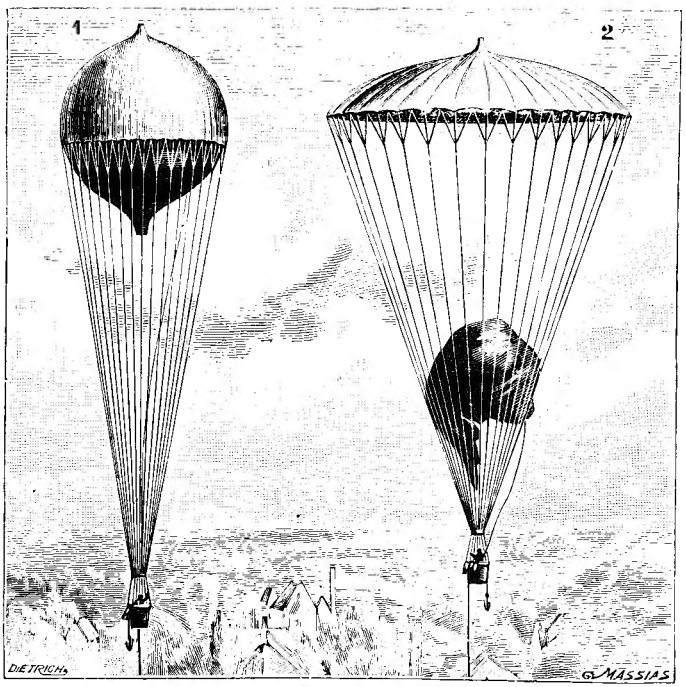
1: LOAD HI-RES SCREEN FROM DISK

Press "1" to load a hi-res or double hi-res screen from disk so it can be cropped and/or saved to auxiliary memory. This option uses the default (last-used) slot, usually slot 6. You will be prompted to select the drive, 1 or 2. Press RETURN to use the default drive number, or type the number you want.

After selecting the drive, you will be asked if you wish to catalog the disk. Press "Y" if you do; otherwise press "N" (or RETURN). Type the file name and then specify hi-res or double hi-res. The picture will be loaded and the menu will re-appear.

2: SAVE SCREEN TO MEMORY

Pressing "2" saves a screen to auxiliary memory. You will be asked to specify the type of screen to be saved. Make the appropriate selection and enter the file name (maximum 8 letters). The screen will be saved in auxiliary memory only. To save all screens in auxiliary memory to disk, exit SCREENS.CROP and use the "&STORE" command (see EXTRA.SCREENS).



3: CROP SCREEN

Press "3" with a text or graphics screen already loaded into memory with option 1 or 4. The most recently loaded screen will be shown with a flashing rectangle superimposed on its border; everything outside of the rectangle will be ignored when you save the screen. The following keys control cropping:

L, R, T, B: Left, Right, Top or Bottom—the side of the rectangle that will be moved with the arrow keys.

Arrow Keys: Move one side of the rectangle (selected by the above commands). On graphics screens, the left and right sides of the rectangles will jump 7 pixels (screen dots) at a time.

Esc: Press ESC to reset the rectangle to the outer border of the screen.

Return: When you're satisfied with the cropping, press RETURN to get back to the menu. You can then select option 2 and save the cropped screen to auxiliary memory.

4: LOAD SCREEN FROM MEMORY

Load a screen from auxiliary memory by pressing "4". Press any key to return to the menu.

5: LIST SCREENS IN MEMORY

Press "5" for the list of all screens in auxiliary memory. Pause a long listing by typing CONTROL-S or stop it with CONTROL-C. (If you notice a screen named "@Z@Z@Z@Z", ignore it. This weird name is used as temporary screen storage by SCREENS.CROP.)

6: CLEAR VIEWING SCREEN

Clear the viewing screen by pressing "6". You will be prompted for the type and the selected screen will be erased. (You will only need to clear the screen if you are going to load one or more partial screens onto a blank background.)

SCREENS.CROP ERROR MESSAGES

SCREENS.CROP will indicate any errors and the line numbers where they occurred. Programmers can examine SCREEN.CROP's Applesoft listing to learn why the error occurred.

EXTRA.VARIABLES

EXTRA.VARIABLES is a program which increases the memory available to Applesoft programs by storing all variables in auxiliary memory, leaving main memory free for your programs. The following table illustrates the space available *WITH* and *WITHOUT* EXTRA.VARIABLES:

Standard DOS	*WITHOUT*	*WITH*
Program space	up to 36K	35K
Variable space	what's left	59K
TOTAL	36K	93K
*With DOS 3.3 moved to language card**		
Program space	up to 45K	44K
Variable space	what's left	59K
TOTAL	45K	103K

*ProDOS cannot be moved. A DOS-mover program is required to move DOS 3.3 to the language card or upper 16K of main memory (DOS-movers are included with Beagle Bros' GPLE and PRONTO-DOS disks).

INSTALLING EXTRA.VARIABLES

Select EXTRA.VARIABLES from the Extra K boot-up menu, or type "BRUN EXTRA.VARIABLES" (see page 6 for information on using EXTRA.VARIABLES from your own disk).

DIFFERENCES IN APPLESOFT BEHAVIOR

EXTRA.VARIABLES is transparent to Applesoft. One benefit, other than the extra space gained, is that variables are not erased when you make changes to a program or Run a program from within another program (see next page).

The only Applesoft commands affected by EXTRA.VARIABLES are FRE, LOMEM: and HIMEM: (see next page).

FRE

Normally you use a command like "PRINT FRE(0)" to display the amount of free memory available and/or do a variable clean-up or "garbage collection". "X=FRE(0)" does the same without printing anything. on the screen. With EXTRA.VARIABLES installed, FRE works according to the number you use inside the parentheses:

- Type "PRINT FRE(-1)" (or any negative number) to print the number of bytes available for your *program* (in main memory).
- Type "PRINT FRE(0)" (zero) to print the amount of free *variable* space (in auxiliary memory) without doing a garbage collection.
- Type "PRINT FRE(1)" (or any positive number) to print the amount of free variable space *and* do a garbage collection. EXTRA.VARIABLES uses its own garbage collection routine which is much faster than the Applesoft equivalent.

LOMEM: and HIMEM:

With EXTRA.VARIABLES installed, these commands set the boundaries for variables in *auxiliary memory* instead of main memory. Ignore LOMEM and HIMEM unless you are using double hi-res graphics or machine language programs that run in auxiliary memory (see the memory map on page 44).

If you need to change the values of LOMEM or HIMEM in main memory with EXTRA.VARIABLES installed, Poke the values directly (LOMEM is located at decimal location 105-106 and HIMEM is at 115-116). For example, you chould change HIMEM to value X with the following line:

```
10 HI%=X/256: LO%=X-HI%*256: POKE 115,LO%: POKE 116,HI%
```

PROTECTING DOUBLE HI-RES GRAPHICS:

If your program will access double hi-res graphics, use a **LOMEM: 16384** command at the start of your program. Otherwise your variables may eventually overwrite your graphics image.

PROTECTING MACHINE LANGUAGE IN AUXILIARY MEMORY

Use a **HIMEM: LOC** command at the start of your Applesoft program to protect a machine language program that runs in auxiliary memory. Make LOC equal the starting address of the routine.

CLEARING VARIABLES IN AUXILIARY MEMORY

With EXTRA.VARIABLES installed, variables in auxiliary memory are cleared as you would expect, by the Applesoft commands RUN, CLEAR, and NEW. Variables are NOT, however, affected by the editing of program lines or by a "LOAD PROGRAM" or "RUN PROGRAM" command (unless the new program redefines them). You can make changes to a program and then GOTO a line number and continue execution with your variables still intact.

This also means that your program can run another program without having to use the CHAIN program (DOS 3.3) or the CHAIN command (ProDOS). In fact, CHAIN will not work with EXTRA.VARIABLES.

A FEW LIMITATIONS

- Machine language programs (such as array sorters) that require direct access to variables will not work without modification because they assume that your variables are in main memory.
- DOUBLE-TAKE's Variable Display function will not work, although the Cross Reference will. D CODE and GPLE work fine.
- There is a special version of PEEK.AND.POKE called PEEK.AND.POKE.X that works with EXTRA.VARIABLES.
- Here's a problem you will rarely encounter—EXTRA.VARIABLES requires a "scratchpad" area above your program and below main memory HIMEM. To avoid an ?Out Of Memory error message during program execution, you should keep at least 1000 bytes free in program memory (main memory). Although it's rarely an issue, you can test for this space with the following statement:

```
IF FRE(-1)<1000 THEN PRINT "PROGRAM TOO LARGE": STOP
```

DISCONNECTING EXTRA.VARIABLES

The FP command (DOS 3.3) or program (ProDOS, see page 45) will remove the EXTRA.VARIABLES program from memory.

SORT NOTE

If you want a nice fast machine-language sorter (alphabetizer) that works with EXTRA.VARIABLES (or without it, for that matter), use the QSORT program on Beagle Bros' Pro-Byter disk (ProDOS only).

EXTRA.VARIABLES TECHNICAL NOTES

■ Only strings are stored in the bank switched portion of auxiliary memory ($D000-$FFFF). All types of variables share memory below $C000. Garbage collection is only done on strings in auxiliary memory below $C000. Unused strings in bank-switched memory will not be recovered by garbage collection.

■ Since variables and strings are no longer in main memory, you cannot use direct Peeks and Pokes on variables and variable pointers unless you use the PEEK.AND.POKE.X program to access auxiliary memory. The following zero-page pointers may be found in auxiliary memory at their normal addresses:

VARTAB: 105-106 ($67-$68) Start of variable space
ARYTAB: 107-108 ($6B-$6C) Start of array space
STREND: 109-110 ($6D-$6E) End of array space
FRETOP: 111-112 ($6F-$70) Start of string storage
MEMSIZ: 115-116 ($73-$74) HIMEM, or end of Applesoft space

■ Machine language programmers may use PTRGET ($DFE3) to find the address of a variable in auxiliary memory. FRMEVL ($DD7B) or FRMNUM ($DD67) may also be used to evaluate variables and functions. The result will be found in the floating point accumulator ($9D). Note that strings with addresses $C000-$CFFF are mapped into the 2nd bank of $D000-$DFFF.

■ If your Applesoft program is a real monster and still won't fit in main memory, use the COMPACT program on Beagle Bros' D CODE disk to reduce its size.

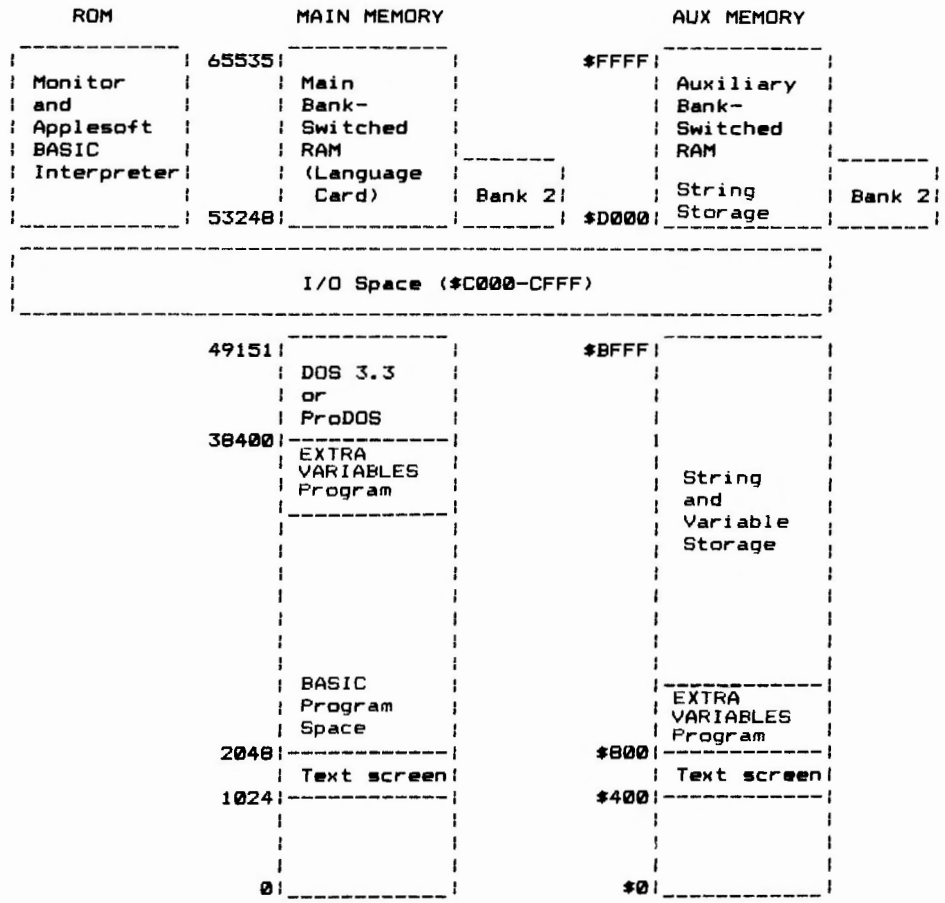
■ EXTRA.VARIABLES loads at $4000 and then relocates itself—one part high in main memory and another part low in auxiliary memory.

EXTRA.STORE

EXTRA.STORE is a program that "patches" BASIC.SYSTEM so that ProDOS's STORE and RESTORE commands work properly with variables stored in auxiliary memory (thanks to EXTRA.VARIABLES).

With EXTRA.VARIABLES in memory, type "BRUN EXTRA.STORE". That's all there is to it. Now STORE and RESTORE will save and load your program's variables to and from disk.

MEMORY MAP



FP (ProDOS only)

FP is an Applesoft program that gives ProDOS the equivalent of DOS 3.3's FP command. Its purpose is to restore Applesoft pointers such as HIMEM to their original values. The FP program (ProDOS) or command (DOS 3.3) will remove any Extra K program from memory and make room for another.

Under ProDOS, type "-FP" to clear memory (the program FP must be on the current disk). FP is written in Applesoft, so you can List it if you want. If you understand it, you may wish to modify it to suit your purposes.

HYBRID.CREATE

HYBRID.CREATE is a program that reformats a disk so it becomes half DOS 3.3 and half ProDOS. You will be able to access the files corresponding to the last DOS booted. It's even possible to boot the DOS of your choice from the same disk.

USING HYBRID.CREATE

To use HYBRID.CREATE, select it from the EXTRA K menu or type "RUN HYBRID.CREATE". You will be notified that:

1. The disk you are going to hybridize must have already been formatted in DOS 3.3 (via the INIT command) or in ProDOS (with the ProDOS FILER program, the System Utilities disk or Extra K's DISK.FORMAT program).

2. BEWARE—All of the existing data on the disk will be permanently erased.

Insert a normal formatted, erasable disk and press "C" to start the hybridizing process. Press ESC if you change your mind. Pressing RETURN will alter your disk for a few seconds and then return you to the menu. Type "Q" to quit or "C" to hybridize another disk.

MAKING A HYBRID DISK ProDOS BOOTABLE

New hybrid disks are data disks only. You can save and load files on them, but you can't boot them. To make a disk bootable under ProDOS, you need to transfer the files PRODOS, BASIC.SYSTEM, and STARTUP to it. The first two can be copied from the EXTRA K disk with Apple's FILER program or the Apple System Utilities disk. STARTUP is optional; it can be any Applesoft program that you want to run when you boot (analagous to a DOS 3.3 "HELLO" program).

For details on making a disk that's bootable under either DOS 3.3 or ProDOS, see the next page.

TECHNICAL NOTES

HYBRID.CREATE puts an empty ProDOS catalog on track 0 and tells ProDOS that only tracks 0 through 16 are available. It then puts an empty DOS 3.3 catalog on track 17 and tells DOS 3.3 that only tracks 17 through 34 are available. The result is that each DOS gets approximately half of the disk. Both DOS's are happy with this set-up because ProDOS expects its directory to be on track 0, while DOS 3.3 looks on track 17.

CREATING A DUAL-BOOT DISK

To make a hybridized disk capable of booting DOS 3.3, it must first be ProDOS bootable (see previous page). It must also have an additional file, DOS.SYSTEM, on the ProDOS half and a HELLO program on the DOS 3.3 half. Confused? Follow these steps:

- Use HYBRID.CREATE to make a hybrid data disk (see previous page).
- Transfer the three files PRODOS, BASIC.SYSTEM and DOS.SYSTEM from the Extra K disk to your hybrid disk.
- Type "NEW" and enter this sample STARTUP program.

```
10 HOME : VTAB 8
20 HTAB 13 : PRINT "D - DOS 3.3"
30 PRINT : HTAB 17 : PRINT "OR" : PRINT
40 HTAB 13 : PRINT "P - PRODOS"
50 PRINT : HTAB 13 : PRINT "SELECT:";
60 GET A$ :ON A$ < > "D" AND A$ < > "P" GOTO 60: PRINT A$
70 IF A$ = "D" THEN PRINT CHR$(4) "-DOS.SYSTEM"
80 PRINT CHR$(4) "CAT"
```

- Type "SAVE STARTUP". Now "CAT" will show a directory similar to:

```
/DOUBLE.DOS.DISK
NAME            TYPE BLOCKS MODIFIED
PRODOS           SYS     30  1-SEP-83
BASIC.SYSTEM     SYS     21  1-SEP-83
DOS.SYSTEM       SYS     19  1-SEP-83
STARTUP          BAS      1  NO DATE
BLOCKS FREE:  58    BLOCKS USED: 222
```

- Now boot with a DOS 3.3 disk and load or write an Applesoft boot-up program that will be named "HELLO". For example, this sample program catalogs the disk:

```
10 HOME : PRINT CHR$(4)"CATALOG"
```

- Insert your hybrid disk and type "SAVE HELLO". Now a DOS 3.3 "CATALOG" command should produce:

```
DISK VOLUME 254
002 A HELLO
```

- Now boot the disk and choose your DOS!

LOGBOOK (ProDOS only)

LOGBOOK is such a new idea that we haven't discovered all of its possible uses. If you find any really good ones, please let us know.

LOGBOOK keeps a visual history of your work. While it is active, every character printed to the text screen is simultaneously saved in auxiliary memory for easy recall. If memory capacity (more than 47,000 characters) is exceeded, the earliest characters are replaced by new information.

Before editing your programs, you may want to activate LOGBOOK so that it keeps a record of all changes. You may then view your original program lines at any time without reloading your program from the disk. You can even use LOGBOOK as a handy notepad (see below).

LOGBOOK's "Find" feature lets you list a program and then search for a particular variable, a specific GOTO, or anything else. You can also do memory dumps or disassemblies and search for particular values or opcodes.

LOADING AND CONNECTING LOGBOOK

To load LOGBOOK, either select it from the Extra K menu or type "-LOGBOOK" with the Extra K disk in the current drive (see page 6 for details on using LOGBOOK from your own disks).

Once LOGBOOK is loaded, you can connect or reconnect it with a PR#0 or PR#3 command. Several commands will disconnect LOGBOOK. Namely CONTROL-RESET, ESC-CONTROL-Q, ESC-4, ESC-8 or PR#X (where X is not 0 or 3). If LOGBOOK is not working, type "PR#0" or "PR#3". Reconnecting will not erase existing information.

DEACTIVATING AND REACTIVATING LOGBOOK

Once LOGBOOK is loaded and connected (see above), all information that appears on the text screen will automatically be stored in auxiliary memory. CONTROL-Q will quit storing information. You can still view the logbook (next page), but no new information will be stored. Type CONTROL-R to reactivate LOGBOOK and continue storing information in auxiliary memory.

NOTEPAD NOTES

To use LOGBOOK as a notepad (a nice Macintosh feature, by the way), load it and connect it with a PR#0 or PR#3. Now type CONTROL-Q to deactivate it until you need to write a note. Then type CONTROL-R followed by your note (don't press RETURN). When your note (256 characters or less) is finished, type CONTROL-Q to deactivate followed by CONTROL-X (to prevent a ?Syntax Error).

LOOKING AT THE LOGBOOK

To see your logbook, enter "Look Mode" by typing CONTROL-L. You will see the earliest data stored—as far back as LOGBOOK can "remember". If you're not sure if LOGBOOK is connected (above), find out with CONTROL-L. If nothing happens, it's not connected. Press ESC or CONTROL-C to exit Look Mode. The following commands work while you're in Look Mode:

Down-Arrow: *Scroll down one full page.*

Right-Arrow: *Scroll down one line.*

Up-Arrow: *Scroll up one full page.*

Left-Arrow: *Scroll up one line.*

B or **b:** *Jump to the Beginning of the logbook.*

E or **e:** *Jump to the End of the logbook.*

F or **f:** *Find something.* Type "F" while in Look Mode. You will be prompted with "FIND:" to type the characters you wish to search for. If a match is found, the first line containing the characters will be listed at the top of the screen followed by the next 22 lines. A beep tells you the characters were not found. Press **RETURN** to continue the search. There will be no visible result if the next occurrence is on the same screen line as the last find. The Find command is not case sensitive, so a search for "CAT" would find "CATALOG", "concatenate" and even "ScAtMaN".

M or **m:** *Mark line for future reference.* While in Look Mode, type "M" followed by a marker number, 0-9. To return to the marked location, just enter the number. For example, enter Look Mode with CONTROL-L, type "M1" and then scroll to another part of the logbook. Typing "1" will return you to the point where you set the marker. The reference point is always the top line.

CLEARING THE LOGBOOK

Typing CONTROL-Z will Zap the logbook of all information. A "CLEARED!" message and a beep will signify that the log is now empty. This option is not available during Look Mode.

REMOVING LOGBOOK

To remove the LOGBOOK program from memory, you must run the FP program by typing "-FP" (make sure that FP is on the current disk).

PEEK.AND.POKE and PEEK.AND.POKE.X

(For advanced programmers)
These two programs install routines that let you Peek and Poke auxiliary memory using two new ampersand commands, &PEEK and &POKE. You can store and recall data, examine or change an 80-column text screen or double hi-res picture, install short machine language routines somewhere other than main memory, and so on.

PEEK.AND.POKE works with normal Applesoft (variables in main memory). PEEK.AND.POKE.X is compatible with the EXTRA.VARIABLES program (variables in auxiliary memory).

LOADING PEEK.AND.POKE

Install the program by selecting it from the Extra K menu or by typing "BRUN PEEK.AND.POKE" or "BRUN PEEK.AND.POKE.X". See page 6 for details on using PEEK.AND.POKE from your disks.

PEEK.AND.POKE is a "smart" ampersand routine, in that it will be compatible with any ampersand routines already in memory.

USING THE NEW COMMANDS

The syntax of the new commands is:

&POKE ***address, expression*** [***,bank***]
&PEEK ***address, variable*** [***,bank***]

address is any number from 0 to 65535 ($0000-$FFFF)

expression is any legal Applesoft expression which evaluates to the range 0 to 255 (for example *X, 64, LEN(A$),* etc.)

variable is any floating point variable such as I, XY(Z), or AUX. (Non-floating point variables such as AA$, BB% and AUX%(64) are illegal.)

[***,bank***] is an optional number 1-4 which refers to the bank-switched memory (or language cards*) in main and auxiliary memory. Memory is organized as follows:

Memory and Bank	*Bank Selection Number*
$D000-$DFFF Bank 1	(1=main, 3=aux)
$D000-$DFFF Bank 2	(2=main, 4=aux)
$E000-$FFFF	(1 or 2=main, 3 or 4=aux)

* The term "language card" comes from the days of the 48K Apple, when a 16K card was added to boost RAM to 64K. A language like Pascal or Integer BASIC was usually stored on it, hence the name. Apple IIe and IIc computers do not have an add-on card, but the name is still used.

PEEK.AND.POKE EXAMPLES

&POKE 1024,193
Puts the letter "A" in the upper-left corner of the 80-column screen. POKE 1024,193 (without the "&") would put the character in the next column. That's because the even-numbered columns (0, 2, 4, etc.) are stored in auxiliary memory, and the odd-numbered columns (1, 3, 5, etc.) are stored in main memory.

&POKE 65535,0,1
Stores a zero at the highest address in the main memory language card.

&PEEK 1025,CHAR
Makes the variable CHAR equal to the ASCII value of the character in the third column of the 80-column screen.

&PEEK 8193,B
Assigns the value of the 3rd byte of the double hi-res screen to the variable B.

&PEEK 53248,A,4
Makes A equal to the value of the lowest address in bank 2 of the auxiliary language card.

REMOVING PEEK.AND.POKE

To remove PEEK.AND.POKE from memory, type "FP" (DOS 3.3) or "-FP" (ProDOS). Under ProDOS, the file FP must be on the current disk.

SPOOLER (ProDOS and Apple IIc only)

FIRST THINGS FIRST

- SPOOLER works with the Apple IIc only.
- SPOOLER works with standard printer slot 1 only.
- SPOOLER requires ProDOS version 1.1.0 or later.
- SPOOLER won't work with any word processors that we know of. If you have a 64K unprotected word processor, give SPOOLER a try and let us know the results.

COMPUTE WHILE YOUR PRINTER PRINTS!

Have you ever had to wait for your printer to finish listing a program before you can get back to work? One solution is to buy a "print buffer" that allows your computer to print while you are using it for other purposes. SPOOLER is a cheaper (*free*, in fact!) solution. It is allows auxiliary memory to act as a buffer. You can dump a listing or other output into auxiliary memory in almost no time and continue working in main memory while SPOOLER sends the data to your printer.

USING SPOOLER

To install SPOOLER, type "-SPOOLER" or select it from the Extra K menu (See page 6 for information on using SPOOLER from your own disks). After SPOOLER relocates itself above HIMEM, you will be prompted with two messages:

LIST ON SCREEN WHILE SPOOLING? (Y/N):
Answer "Y" if you want SPOOLER to list each character on the screen as it is being saved in memory for printing. Answering "N" will speed up the process slightly.

CLICKS WHILE PRINTING? (Y/N):
Press "Y" if you want SPOOLER to click the speaker every time a character is sent to the printer. This notifies you when characters are actually being transmitted.

To use SPOOLER, just send text to your printer as you normally would. For example, with SPOOLER loaded, type "PR#1" and then "LIST". To regain control after printing starts, type "PR#0" or "PR#3" and continue your work.

If you are wondering if SPOOLER has been installed, press CONTROL-G. If SPOOLER is there you will get a distorted bell instead of the usual beep.

SPOOLER COMMANDS

After SPOOLER is loaded, you may use any of its commands from immediate or program mode by pressing **both Apple keys** (each side of the space bar) simultaneously. A "Spooler Command: " prompt will be given. Select one of the following commands:

C: Clear
Press "C" to clear the buffer in auxiliary memory. This is handy if you've sent the wrong data to SPOOLER.

Space Bar: Pause/Resume
Press the space bar to temporarily stop printing. Press again resume printing.

Q: Quit
Press "Q" to disconnect SPOOLER entirely.
IMPORTANT: Always disconnect SPOOLER before running another system program (indicated by a ".SYS" suffix in the catalog) or any other program that uses auxiliary memory. Failure to disconnect SPOOLER will cause a system crash, requiring you to re-boot.

SPOOLER LIMITATIONS

SPOOLER cannot be used with most word processors because they either require you to reboot or to work with a system program other than BASIC.SYSTEM. Most ProDOS programs without these requirements should work with SPOOLER.

If you have a mouse, activating it will turn off SPOOLER. To turn it back on after you are finished with the mouse, press CONTROL-RESET.

TECHNICAL NOTES

SPOOLER cannot be used with an Apple IIe (enhanced or not) because a IIe cannot generate the vertical blanking interrupts that drive SPOOLER. These interrupts, which occur 60 times per second, are used to send characters to the printer between execution of normal BASIC commands.

Important: ALWAYS EXIT SPOOLER BY SELECTING THE "Q" OPTION OR BY EXECUTING FP. Other Extra K programs will not work if you don't.

(Thinly-disguised advertisement)

GPLE

GLOBAL PROGRAM LINE EDITOR by Neil Konzen

$49.95, Compatible with any Apple II, DOS 3.3 and ProDOS—Includes Apple Tip Book #7

A "WORD PROCESSOR" FOR APPLESOFT PROGRAMS

GPLE is *The* classic Applesoft line editor for the Apple. It lets you edit your program lines fast without awkward cursor-tracing or clunky "escape editing" methods.

GPLE is installed in memory when you boot, remaining "invisible" to your programs and unaffected by even the most "destructive" commands, such as FP and INT. You may install GPLE in normal 48K memory or in the Language Card (built-in on all IIe's and IIc's).

INSERT AND DELETE

Now you can make almost instant changes to any Applesoft or Integer Basic program line. GPLE lets you jump the cursor to the change point in the line and insert or delete text. Other code in the line moves aside to make room (what you see is what you get). If you make a mistake, you can restore the line to its previous condition with a keystroke.

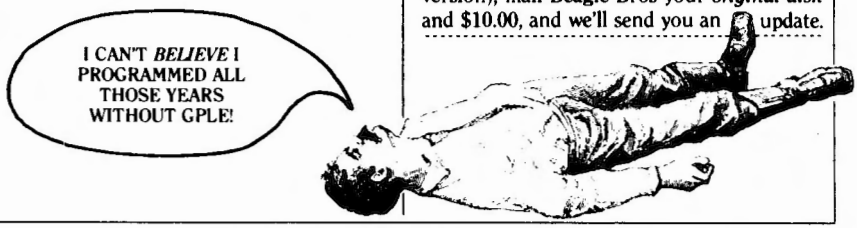
Control-characters are easy to insert and delete, appearing in inverse when being edited.

With GPLE, it is no longer necessary to trace the cursor to the end of the line you are editing. No matter where the cursor is, hit Return, and that line is entered into memory.

GLOBAL SEARCH & REPLACE

GPLE finds any word or variable in a program, letting you change that line, delete it, or just look at it. Here are some examples of GPLE's Global capabilities:

- Look at all lines containing a GOSUB.
- Edit or delete all lines with a REM.
- Locate all occurrences of the variable XX.
- Replace all X-variables with ABC's.
- Change all *Hello* strings to *Good-Bye's.*



DEFINABLE ESC FUNCTIONS

GPLE lets you define an ESC-keypress followed by any other key to perform any keyboard task. For example, *ESC 1* can catalog drive 1, *ESC L* can do a "HOME: LIST", *ESC N* could type an entire subroutine... *Anything* you want, whenever you want it.

A complete set of Escape functions is included with GPLE, pre-programmed and ready to use. Each function may be used as is, or deleted or changed whenever you like. After you create your own "Escape Table", you can save it on disk so it will be in memory the next time you load GPLE.

80-COLUMN COMPATIBILITY

All GPLE edit and global features support Apple 80-column cards *and* most 80-column cards on *any* Apple IIc, IIe, II+ or II.

Double-Take, ProntoDOS, DOS Boss, Flex Type, etc.,—and, of course, all of your Applesoft and Integer Basic programs—get along quite well with GPLE.

GPLE DOS MOVER

GPLE comes with its own "DOS Mover" program that lets you move DOS to the Language Card (built-in on all IIc's and IIe's) for an EXTRA 10,000 Bytes (10K) of programmable memory. GPLE itself may be located on the Language Card or in Main 48K memory.

PLUS APPLE TIP BOOK #7

Learn more about your Apple—GPLE comes with more tips and tricks from Beagle Bros, many involving GPLE. Hours of good reading and Apple experiments.

UPDATES: If you have an old GPLE that's not Double-Take, 80-column or ProDOS compatible (even if it's an old *Synergistic* or *A.P.P.L.E.* version), mail Beagle Bros your *original disk* and $10.00, and we'll send you an update.

BEAGLE BROS / 3990 Old Town Avenue, Suite 102C / San Diego, California 92110

(Thinly-disguised advertisement)

New! D Code *New!*

APPLESOFT®PROGRAM COMPACTOR & DE-BUGGER by Alan Bird

$39.95, Compatible with any Apple II—DOS 3.3 and ProDOS

FAST APPLESOFT PROGRAM COMPACTOR

D Code squeezes every wasted and unused byte out of your Applesoft programs, to save valuable memory space and increase program speed and efficiency.

COMBINE PROGRAM LINES:

D Code will reduce an Applesoft program to the smallest number of program lines possible. (Each pair of lines combined saves four bytes.) D Code operates automatically—all Goto's, Gosub's and If-Then's are taken into account. The total number of bytes saved is reported on the screen.

SHORTEN VARIABLE NAMES:

D Code lets you optionally shorten your variable names to one or two characters. Each character saved is a byte earned.

REMOVE REMARKS:

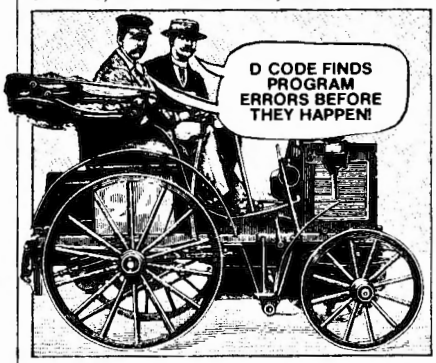
REM statements are big memory eaters. D Code will remove them for you to potentially save tons of memory space. You can save your remarked version as a separate file for reference.

UNUSED-LINE FINDER

If your program contains a line or statement that can't possibly be executed, D Code will find it and let you delete it to save wasted space. Check some of your (or our) programs; you may be surprised!

D CODE PROGRAM COMPARER

Quickly compare any two files, DOS 3.3 or ProDOS, and find out if they are the same.



PROGRAM PROOFREADER

Word processors have their spelling checkers, why shouldn't Applesoft? D Code will uncover program errors before they happen!

NEW WHILE-YOU-TYPE SYNTAX CHECKER

If you commit a typo or enter an undefined statement number while typing a program line, the error is *reported on the spot* so you can immediately make a repair—*before* running your program. This was one of the good features of Apple's old Integer Basic. We've improved on it by letting you enter anything you want. If it's not legal Applesoft, D Code simply reminds you.

PROGRAM SCANNER

D Code will let you quickly scan an entire program for correct syntax and other potential errors that might unexpectedly cause a crash.

NEW PROGRAM TRACERS

D Code features a TRACE command that works the way it should! Run your program the way you always do. When it bombs (or when you stop it), you can ask to see the most recent program line numbers and statements (1 to 10,000 of them) that have been executed. A great feature for finding out what makes programs tick (or *crash!*).

WINDOW TRACING TOO

D Code also features a "live" TRACE feature that neatly prints program statements and variable & string values in a window *at the bottom of the screen.* No more screen layouts messed up by line numbers all over the place. And you can watch your variable values change as your program executes!

BREAK POINT CONTROL

D Code lets you set up de-bugging "break points" before you Run a program, so a program stops only if a certain condition is true, or if a certain statement is executed a specified number of times.

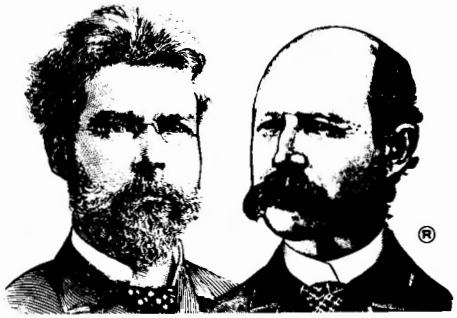
LIGHTNING-FAST FIND

D Code will let you find occurrences of any variable or string in your programs—*so fast* you won't believe it!

BEAGLE BROS / 3990 Old Town Avenue, Suite 102C / San Diego, California 92110

Extra K Index

EXTRA K
PROOF OF PURCHASE
(Save for disk or manual updates.)



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. Autoproofread 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 0-917085-13-2

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

Extra K Programs and Command Examples

(Under ProDOS, "–" may be substituted for RUN or BRUN.)

■ To compare two disks:
RUN DISK.COMPARE (page 10)

■ To duplicate disks in 35 seconds:
BRUN DISK.COPY (page 14)

■ To initialize ProDOS disks:
RUN DISK.FORMAT (page 16)

■ To make two Apples out of one:
BRUN EXTRA.APPLE (page 18)

(To transfer programs, see page 24.)

CALL 600 to set up two Apples.
CALL 769 to switch between Apples.
?PEEK(768) to see which Apple is active.

■ To display screens at high speed:
BRUN EXTRA.SCREENS (page 26)

(To crop screens, see page 38.)

Save a screen to auxiliary memory
(create or load screen from disk first):
&"F", SAVE "PICNAME" *(40-col. text)*
&"E", SAVE "PICNAME" *(80-col. text)*
&"L", SAVE "PICNAME" *(Lo-Res)*
&"H", SAVE "PICNAME" *(Hi-Res)*
&"D", SAVE "PICNAME" *(Double Hi-Res)*
&"M", SAVE "PICNAME" *(Medium-Res)*
&"H", AT2, SAVE "PICNAME" *(pg.2 Hi-Res)*

Load a screen from aux. memory into view
(you must *reveal* the viewing screen):
&LOAD "PICNAME" *(load onto default page)*
&AT2, LOAD "PICNAME" *(load onto page 2)*

Store all aux. memory screens to a disk file:
&STORE "FILENAME"

Load aux. memory screens from disk:
&RECALL "FILENAME"

Catalog screens in aux. memory:
&LIST

Rename a screen in aux. memory:
&CALL "OLDNAME", "NEWNAME"

Delete a screen from aux. memory:
&DEL "PICNAME"

Clear all screens from aux. memory:
&CLEAR
&CLEAR,D *(if double hi-res will be loaded)*

See how many pages of aux. memory are free:
&FRE, X: PRINT X *(X=no.of 256-byte pages)*

■ To make more space for programs:
BRUN EXTRA.VARIABLES (page 40)

■ To make double-DOS disks:
RUN HYBRID.CREATE (page 46)

Format a ProDOS or DOS 3.3 disk first.
(To make disks bootable, see page 46.)

■ To keep a record of your text screen:
BRUN LOGBOOK* (page 48)

PR#0 or PR#3 connects LOGBOOK.
CONTROL-Q stops storing information.
CONTROL-R resumes storing information.
CONTROL-Z zaps (erases) the logbook.
CONTROL-L lets you look at the logbook.
Up/Down arrows scroll up/down a page.
Left/Right arrows scroll up/down a line.
B jumps to beginning of logbook.
E jumps to end of logbook.
F finds a string.
M(0-9) sets a marker.
0-9 finds a marker.

■ To Peek and Poke auxiliary memory:
BRUN PEEK.AND.POKE (page 50)

(PEEK.AND.POKE.X w/EXTRA.VARIABLES)
&POKE A,N *(pokes aux. address A with N)*
&POKE A,N,B *(B=Bank number; see page 50)*
&PEEK A,X *(sets X to value at aux.address A)*
&PEEK A,X,B *(B=Bank number; see page 50)*

■ To print while you use a IIc:
BRUN SPOOLER* (page 52)

Apple keys (simultaneously) allow commands:
C clears the print buffer.
Space pauses or resumes printing.
Q quits SPOOLER.

■ To see if an Apple has 128K:
BRUN AUX.MEM.CHECK** (page 8)

IF PEEK(6)=128 THEN ?"128K AVAILABLE"

■ To disconnect ProDOS's RAM disk:
BRUN DISCONNECT.RAM* (page 9)

■ To clear memory and reset pointers:
RUN FP* (page 45)

* ProDOS only
** DOS 3.3 only

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


CALIFORNIA
HIMEM
gle Br
OOR SPOR