Open-Apple™

July 1987
Vol. 3, No. 6

ISSN 0885-4017
newstand price: $2.00
photocopy charge per page: $0.15

Releasing the power to everyone.

Last word trademarked

Computer industry executives say that new product introductions will be more difficult but that somehow they'll cope with the U.S. trademark office's mid-June announcement that every word in the English language has now been trademarked. As recently as ten years ago, trademark experts estimated that the supply of untrademarked words would last well into the 22nd century, however, that was before the full impact of the personal computer revolution was known. Computer products alone have accounted for more than 90 per cent of all trademarks issued during the last five years, the trademark office reports.

The last English word to be trademarked was "alack." It will grace a new MS-DOS spreadsheet program. "We would have preferred a different word," said Jim McGlinn, a spokesman for the program's publisher, "but it was the last word left. Since *Alack* almost has all the letters needed to spell 'calc,' we decided to take it while we had the chance."

Trademark experts say companies have only three options left in naming new products. One is to make up new words. Experts say this option is getting very tiresome, however, because all the pronounceable combinations of four-, five-, and six-letters have already been used.

The second option is to use foreign words. This is apparently the strategy Apple's new software company (see our June issue, page 3.37) will take in renaming AppleWorks and *Apple Writer*. Sources close to the company, who asked not to be identified, said plans are underway to add a Sanskritic suffix to all the company's program titles. For example, AppleWorks and *Apple Writer* are expected to be re-released as *WorksWala* and *WriterWala*. Apple's trademark lawyers were reportedly in Bombay at the end of June trying to license Asian rights to the suffix from the Bata Rickshaw Company.

The third option is to share a product name with another company, which is a possibility only in cases where it is unlikely that there will be any confusion in the marketplace over what is what. For example, Heinz 57 has reportedly licensed all of it pickle and mustard trademarks to Lotus and Ashton-Tate. Microsoft is known to be negotiating with Oldsmobile for the right to name the operating system it's developing for IBM's new computers after the Jetfire Rocket 88. And Borden has sold Aldus the rights to Elsie the Cow.

(Trade sources report Aldus will use the trademark for a new desktop dairy program for the Mac II. This will be the first in a series of desktop manufacturing programs from Aldus, which will be based on a top-secret Apple product to be called the LaserMaker, which is itself based on the Mazda rotary engine. This machine is rumored to be able to manufacture any object that can be drawn on the Mac II screen. Elsie and the LaserMaker will be introduced by Aldus and Apple, confidential sources say, as soon as Apple and Mazda can get a slight aftertaste of Napa Valley Rose' out of the machine's milk.)

Closer to home, we've found two products you might be interested in that are both called "SoftSwitch." One is a software product from Roger Wagner Publishing that turns a 512K Apple IIgs into three 128K Apple IIes. The other is a box that allows you to plug four printers into one serial port and switch between them by sending a control code to the box.

Roger Wagner Publishing's *SoftSwitch*, which was written by Ken Kashmarek, has been running flawlessly on my Apple IIgs since a few hours after the UPS man delivered it. This *SoftSwitch* is a desk accessory that appears in the control/open-apple/escape menu along with the Control Panel and whatever other desk accessories you've put there. *SoftSwitch* gives you three "workspaces" for saving the current status of ProDOS 8 (or DOS 3.3 or Apple Pascal or any combination) programs. You can switch between workspaces with just a few keystrokes. When you return to a program, it is exactly where you left it. You can actually leave Applesoft in mid-beep as it punishes you for a syntax error and return hours later to hear the rest of the beep.

The most obvious use for *SoftSwitch* is to switch between, for example, AppleWorks and other programs that offer features not in AppleWorks, such as communications or graphics programs. In this use, *SoftSwitch* becomes sort of an expanded open-apple-Q(uick Change) command. Or use it to switch between a word processor, an assembler, and Applesoft. Or use it to demonstrate the differences among *VisiCalc*, the AppleWorks spreadsheet, and *SuperCalc 3a*.

You can also use it to revert to a previous program state. For example, just before you decide to descend a dark staircase in a new adventure game, you can tell *SoftSwitch* to save the current program state. If you get eaten alive at the bottom of the staircase, *SoftSwitch* can restart you at the top for a go in another direction.

SoftSwitch can allow several users to share one machine. You can save your application in a workspace and let another person have the computer. When the other person is finished, your application can be restored instantly, at the exact point where it was interrupted.

SoftSwitch also has a provision for cutting and pasting any single- or double-resolution graphic image from one program to another. It doesn't matter whether one of the programs is protected or whether the programs run under different operating systems. All that's necessary is that the programs use standard Apple graphics and that they allow access to the desk accessory menu. Thus, if one of your workspaces holds a graphic printing program such as Beagle Bros *Triple Dump* or Roger Wagner's own *Printographer*, you can capture a high-res image from almost any program at any time and print it. *SoftSwitch* can also save text displays — thus providing a way to take "screen shots" for use in printed material.

"RIGHT NOW I'M KEEPING A LOW PROFILE. LAST NIGHT I CRANKED IT ALL UP AND BLEW OUT THREE BLOCKS OF STREETLIGHTS."

Keepsake is a separate program supplied with *SoftSwitch* that allows you to save the program currently stored in any workspace to disk. Using *Keepsake*, you can suspend a program's operation over any period of time. Or you can compare an image of a program before it crashed to an image after it crashed to see where the changes are.

There are a couple of small things I don't like about *SoftSwitch*. One is that it's a ProDOS 16 program. This means you have to boot a ProDOS 16 system disk to install *SoftSwitch* in the desk accessory menu. Using an Apple 3.5 drive, this takes 25 seconds. This isn't *SoftSwitch's* fault, of course, but the fault of ProDOS 16 and the (chug-a-chug) System Loader. So I called Wagner and asked him why *SoftSwitch* wasn't a ProDOS 8 program.

He said that every time ProDOS 16 is booted, it removes all existing desk accessories. If *SoftSwitch* was a ProDOS 8 program, then you couldn't get into a ProDOS 16 application without destroying *SoftSwitch* and losing your workspaces. Instead, *SoftSwitch* was designed so that ProDOS 16 would *already be* booted by the time you got to *SoftSwitch*. You can use the active *SoftSwitch* workspace to "launch" or "run" (but not "boot") a ProDOS 16 application. *SoftSwitch* can be used to capture screen images while the ProDOS 16 application is running, but for nothing else. After the ProDOS 16 application ends, however, the full powers of *SoftSwitch* return and the contents of all three saved workspaces are intact.

Why won't *SoftSwitch* work with ProDOS 16 applications? Wagner says the primary problem is that, currently, almost all ProDOS 16 applications "don't behave." They use memory areas that are off-limits, they disconnect the control-panel, or they pull other tricks that make switching impossible. Once ProDOS 16 programs have developed predictable behavior patterns, Wagner expects to have a *SoftSwitch* update. Oh, and guess what. AppleWorks 2.0 and the Apple IIgs System Utilities misbehave, too. (You can always depend on Apple's own software being first in line when it comes to not following compatibility rules.) You can run AppleWorks 2.0 in one workspace without problems, but don't try to run it in two of them or try to "revert" to an earlier program status. Earlier versions of AppleWorks cause no problems. Here's a little editorial from the *SoftSwitch* manual about this stuff:

***SoftSwitch** works very nicely with programs that follow the Apple design guidelines for software. These guidelines were put together by Apple to encourage software publishers to create software that is easy for everyone to use.*

*Software that doesn't follow these guidelines makes life harder for everyone. Here are things in particular that **SoftSwitch** doesn't like:*

1.) No QUIT command, or quit reboots instead of returning to the program selector.

2.) Erasing all of memory when quitting.

3.) Disabling the Desk Accessory Menu.

4.) Incompatibility with Desk Accessories; i.e., access may be allowed, but the program crashes on return from the Desk Accessory menu.

5.) Hi-Res (or Double Hi-Res) images not stored on normal graphics screens, and/or are encoded in a non-standard format.

*If you have programs that have these kinds of problems, **write the publishers** and tell them you want your software to work for you, not against you! Of course they want your opinion—they gave you that nice registration card and asked for it, remember!*

So, if you're using software, let other people know what you want; if you're writing software, take a little time to make things easier for everyone.

If you analyze the problems listed above, you'll note they are almost all related to copy protection. Which brings up the second thing I don't like about *SoftSwitch*—it's somewhat copy protected. You get an uncopyable INSTALL disk, which is used to place *SoftSwitch* on ProDOS 16 system disks. These disks, at least, are fully copyable—but the *SoftSwitch* program file doesn't show up where you'd expect in the catalog. Rather than being in SYSTEM/DESK.ACCS/SOFTSWITCH it's called SYSTEM/SYSTEM.SETUP/TOOL.SETUP.2. When you send in your registration card, tell Wagner you want your software to work for you, not against you.

Roger Wagner Publishing is selling *SoftSwitch* for $59.95. It's available now. (1050 Pioneer Way, Suite P, El Cajon, CA 92020 619-442-0522).

The other Soft-Switch (note hyphen) is a possible solution to the pervasive out-of-slots problem. This Soft-Switch is a box with one RS-232 serial input and four parallel outputs. It allows you to connect four parallel printers to a single serial port. Unlike other devices of this kind I've seen, which require you to turn a knob to change printers, Soft-Switch lets you change by sending the box a control-E, followed by a number between 1 and 4. With AppleWorks, for example, you'd just add "control-E 1" to the interface card code for printer 1, "control-E 2" to the code for printer 2, and so on. A program can also change printers and return the box to its previous setting, without actually knowing what that setting was, by requesting printer 5 when it has finished printing.

This Soft-Switch costs $199 and is manufactured just across the Kaw from us by Intronics, P.O. Box 13723, Edwardsville, KS 66113 913-422-2094. Intronics also makes parallel-to-serial and serial-to-parallel converters that sell for $65. The Soft-Switch was originally designed to allow institutional computers to print several kinds of continuous forms (invoices, packing slips, and UPS tags, for example) out in the warehouse on cheap printers, but I'm sure there are people in the Apple II kingdom who could put such a device to good use too.

Back in 1980, in the heyday of the Apple II Plus, the high-end data base program for the Apple II was probably Stoneware's *DB Master*. After a few years of success, the company began a long, slow fade that ended when it closed its doors last summer.

Stoneware the company was started by a fellow named Barney Stone. Soon after he started it he sold it, then he and some co-authors developed *DB Master*, which Stoneware obligingly published. After Stoneware disbanded, the rights to *DB Master* were acquired by Stone Edge Technologies, a company made up of Stone and his co-authors. They have since been busy on a total rewrite of the program, which is now available as *DB Master Version Five*. This new, ProDOS-based version of *DB Master* allows 200 fields-per-record and up to 250 characters-per-field (with a 2,000 character-per-record limit). Each record can have as many as 30 screens. The program will use up to 40 megabytes-per-file on a ProDOS-compatible hard disk, or up to 50 disks (5.25, 3.5, or mixed). In addition, Stone is working on a "Professional" version, which he expects have ready by fall, that will bring a multi-file relational capability (along the lines of what I wrote about here last month) to the Apple II.

The differences between *DBMaster* and the AppleWorks data base lie along the lines of speed, power, and capacity. Since AppleWorks stores all records in memory while a file is being worked on, it will always be faster than *DB Master*. Since *DB Master* stores all records on disks, on the other hand, it can deal with many more records, which can be larger than what AppleWorks allows. Like AppleWorks, *DB Master Version Five* is written entirely in assembly language (and requires an enhanced IIe, a IIc, or a IIgs), so the only thing that will slow *DB Master* down will be the speed of disk access.

One area where the power of *DB Master* tends to blow AppleWorks away is in report generation. Where AppleWorks allows you to set up only table-style or 15-line maximum label-style reports, *DB Master's* report generator allows the same easy set up as AppleWorks but includes multi-page "invoice-style" (free-form) reports as well. It allows field-by-field control over print style and field formatting. You can include your own text in your reports, such as commas between city and state (or even an entire letter for mail merge), which AppleWorks won't allow. Each report has four specification screens—one each for sort, selection, printer, and layout.

Stone says that he wants to make *DB Master* attractive to AppleWorks users and that he wants it to be the high-end data base program in the Apple II market. Version five is available for $179, the Professional version is expected to be priced at $295. One year of technical support, including free, automatic upgrades, is $75 for version five and is expected to be $100 for the Professional version. Upgrades for current *DB Master* owners are available. For more information, contact Stone Edge Technologies at P.O. Box 200, Maple Glen, PA 19002 215-641-1825.

I've learned some interesting facts about the U.S. telephone system lately that may be of use to some of you. My wife is a research psychologist and she's been spending a lot of time the last few months hiding out at home analyzing data on a mainframe computer at work. At our end she has an Apple IIe, a 1200-baud modem, and *ASCII Express* set up to make the mainframe think she's working on an ADM terminal.

She was happy with this system. The only problem was that I couldn't use the phone for hours at a time. And people trying to call us would encounter a continuous busy signal. So I decided to have a second phone line put in. However, I was quite concerned about the time and work it would take to run hidden wires from the phone company's "network connection" on the outside of the house to all the places I wanted the second line to be available.

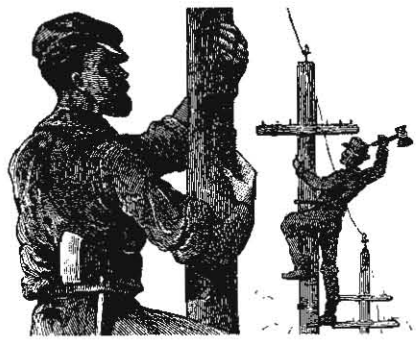
That's when I started to investigate something that has always puzzled me. Why do telephone people always talk about their cables being "twisted *pair*," when, in fact, there are *four* wires inside most telephone cables? It turns out that telephones really do use only two wires (the red one and the green one).

The other two wires (the yellow one and the black one) were added 10 or 20 years ago to provide low-voltage power to phones with lights in them or other special features. Most homes, however, don't use these wires for anything. You can hook your second phone line to them and immediately have the second line available at every existing phone jack in your house.

Before proceeding, get a multimeter and attach it to the yellow and black wires to make sure there is no voltage on them. The person who lived in the house before you may have had a phone with lights—the transformer that kept them lit may still be attached. If there's no voltage, check to make sure that there's plenty of resistance as well—if there's not, it probably means the yellow and black wires are twisted together inside one or more of your jack housings.

After the wires pass both of these tests, attach your second line's red wire to your home's yellow wires and the second line's green wire to your home's black wires. Next, go down to your local Radio Shack store and look over their selection of "two-line modular jacks." These are little conversion plugs that you stick into your existing phone outlets (at the wall) that give you two or more outlets to plug your phones and modems into. Radio Shack even sells two-line telephones that *expect* your second line to be on the black and yellow wires.

Now my wife's modem is plugged into our second line and she can use the mainframe all day while my 10-year-old daughter's friends can call in on the



first line with nary a busy signal. My wife can even call up the mainframe technicians and complain about the speed of their computer without logging off.

I also have my old Integer Basic Apple II and my circa 1980 Hayes Micromodem hooked into the second line. It answers the phone in the middle of the night when ***Open-Apple***'s 800-number answering service wants to dump records of the previous day's calls (see last month's page 3.35 for more details). Their computer thinks my old II is a remote printer, but I won't tell if you don't. If any of you would like to dump something into my "remote printer," the number is 913-383-3203 (300 baud, 8N1, half-duplex). Don't expect it to say hello to you or anything, however, just call in and start sending something. The II will save your message on a disk and I'll find it and put it in one of the piles around here the next morning.

Incidentally, if you think it's too expensive to send text by long-distance, check out AT&T's "Reach Out America" rates. If you sign up for this plan you'll be billed a minimum of $8.40 a month. This covers one-hour of long distance (any number of calls) anywhere in the U.S. outside of your own state. Calls must be made at night, between 10 pm (not 11 as with standard night rates) and 8 am, or between 10 pm Friday night and 5 pm Sunday evening. Additional hours are billed, in minutes, at a rate of $7.55 per hour. With rates like these it's becoming cheaper to call local bulletin boards all over the country than to use the national information services. Other long distance companies may offer even better deals. For more information on AT&T's rates, call 800-225-5288.

Do you remember a program called *TK!Solver*? It was originally published by Software Arts, the company that *developed* VisiCalc, and it sold for $395. It's an equation solver that can provide answers to problems with several unknown variables. It's a classic program that has always been highly regarded by people who need to solve such equations.

Meanwhile, Software Arts got into a emotional sink with VisiCorp, the *publisher* of VisiCalc, and the two companies choked each other to death. Software Art's assets were purchased by Lotus, and Lotus later sold all rights to *TK!Solver* to a company called Universal Technical Systems. Universal updated the program to version 1.6, took off the copy protection, and is selling an Apple II version for $99 (1220 Rock St, Rockford, IL 61101, 800-435-7887). If you're into iteration or consecutive substitution this one should be in your library.

Novation is a company that has been making modems since the days of teletype terminals. They have just announced a $119, 300/1200-baud, Hayes-compatible, stand-alone modem that's about the size of an audio cassette (smaller than the "microcoupler" on old Hayes Micromodems, for those of you who have been around awhile). It doesn't even have a power supply—it gets all the juice it needs from the RS-232 serial interface. They call it the Parrot 1200. Pretty soon modems are going to be cheaper than the cables you need to connect them to your computer.

I've seen pictures of the ultimate LEGO set and I sure wish the price was closer to what Santa Claus can afford. LEGOs, for those of you out of touch with the world of children, are building blocks. The ultimate LEGO set comes with a version of the Logo language on disk (talk about shared trademarks), a card that fits in a IIe or IIgs slot, an interface box, 2 motors, 1 optical sensor with counting disk, 2 touch sensors, 4 light bricks, and more than 400 other special pieces for building interesting machines that can be controlled by your Apple II. The brochure for this set makes me drool. The price, however, ($450) makes me wince. To get your brochure talk to: Sarah McDonald, Education Dept, LEGO Systems, 555 Taylor Rd, Enfield, CT 06082, 800-243-4870, 203-749-2291.

***Open-Apple* subscribers are a creative and hard-working bunch,** and several of them have come up with stuff they want the rest of you to know about.

PathFinder is a $20 enhancement for the U.S. version of AppleWorks 2.0 that replaces the "Format a blank disk" option on AppleWorks' Other Activities menu with commands that allow you to specify subdirectories by pointing at them and pressing return rather than by typing them in. This is how AppleWorks should access subdirectories at every point that it presents a disk catalog, but doesn't. *PathFinder* was written by Randy Brandt—the same Randy Brandt who wrote Beagle Bros' *Super MacroWorks*. He is selling *PathFinder* himself, however, through a company he calls JEM Software, P.O. Box 20920, El Cajon, CA 92021. One of my favorite parts of the product is the warranty. Right underneath Apple's standard California-lawyer ProDOS warranty it says, "Like ProDOS, this product is basically useless and you're left holding the bag if anything goes wrong. Some states disagree, in which case you're exempt."

The *Periodical Index* from T-Squared Enterprises was described here previously in November 1986, page 2.74. T-Squared's indexes are self-contained, meaning each disk contains both data and the program to search the data with. Indexes to the 1985 and 1986 issues of ***Open-Apple*** are now available for $5 each. Previous purchasers of the 1985 index will automatically get both the 1985 and 1986 indexes on a single disk (1015 S Ridge Ave, Arlington Heights, IL 60005, 312-577-7381, add $3 for orders outside the U.S.).

The *Magazine Index* from Kula Software is much broader in scope than the index from T-Squared, with proportionately less detail. Kula's index covers not just ***Open-Apple***, but all the Apple II publications we get around here plus a few more I've never seen. Another interesting feature of Kula's index is that it's distributed as a set of AppleWorks data base files. The 1986 disk has an introductory price of $5—earlier years are also available (2118 Kula St, Honolulu, HI 96817, 808-595-8131).

Peter Meyer, author of Pinpoint's *InfoMerge*, solved the "read an AppleWorks data base file" problem that I discussed here in March (pages 3.9 to 3.12) with an ampersand routine capable of making ProDOS MLI (machine language interface) calls. Meyer and an Applesoft program that uses these routines to read an AppleWorks data base file of any size can be separated for $5. If there's enough interest, Meyer will make available the complete Applesoft-MLI interface routine (P.O. Box 1001, Captain Cook, HI 96704).

Robert Moore has an assembly language routine called INPUT.PRINT.GET that he will share with anyone who sends him a disk and a stamped, self-addressed disk mailer. The program adds three ampersand commands to Applesoft, &INPUT x$, &PRINT x$, and &GET x$. The commands behave much as the similar commands in SU2.OBJ do (see December 1985, page 96; March 1986, pages 2.13-14; and May 1986, page 2.32 for the complete details on SU2.OBJ). Moore provides his routine in both object code and heavily commented source code. The source code illustrates how to install machine language routines above HIMEM in both DOS 3.3 and ProDOS, how to chain into the ampersand hook, how to read the value of an Applesoft real variable from machine language, how to set the value of an Applesoft string or real variable from machine language, and how to use software "switches" and "signatures" to obtain multiple functions out of a single module of code (1204 Marton Street, Laurel, MD 20707).

Ask (or tell) Uncle DOS

The AppleWorks machine

All right, so I finally broke down and bought AppleWorks after reading so much about its virtues in your publication. I didn't want to shell out the 250 bucks when I bought the IIc but I see the light now. I thought I could get by with *Apple Writer* and *PFS:File*, a Pascal program that by comparison seems slow, inflexible, and difficult to work with, especially in printer applications. AppleWorks sorts in the blink of an eye, is very flexible and forgiving, and is not nearly so disk-intensive. Print criteria such as margin settings are saved with the file and *Apple Writer* doesn't do that.

I'm only beginning with it, but I have already used it to submit a proposal concerning a new warehouse facility our company is building. The ability to merge database and spreadsheet reports with the word processor has opened new worlds for me. And the final document looks smarter than I do sometimes. So hey, all you out there with an Apple II, if you don't have AppleWorks, buy it! To think of someone else wasting all the time I did creating all those Pascal files drives me crazy.

By the way, would AppleWorks use the disk less if I had my IIc upgraded to 256K? What I have is version 2.0 running in 128K. It has to access the disk on a lot of open-apple commands and I don't think it should have to.

Harry Barr
Dayton, Ohio

It's nice to occasionally see the world through fresh eyes. Dennis and I enjoyed reading your letter. But you, my friend, ain't seen nothin' yet. For the record, here's the kind of enhancements you should consider adding to your system:

Memory expansion. *Get your computer's memory expanded to 512K or more and you won't be able to understand why people buy IBM-PCs. In the IIe and IIc world you'll have to choose between an "aux-slot" card—such as Applied Engineering's RamWorks (IIe) or Checkmate Technology's MultiRAM (IIe and IIc versions)—or a "standard-slot" card—such as Applied Engineering's RamFactor (any slotted II), Cirtech's plusRAM (any slotted II), or Apple's own memory cards (slot and IIc versions). The aux-slot cards are based on an older technology that's starting to fade, but they come with AppleWorks expansion software that's better than what you can get with the standard-slot cards. It's a tough choice. I lean slightly toward the standard-slot cards because they're more likely to be supported by future software.*

People in the IIgs world, on the other hand, should start with a card that fits into the special IIgs memory expansion slot. (If you also have a II-Plus/IIe standard-slot card, you can plug that into the IIgs too, but its memory doesn't become part of the directly-addressable IIgs memory. It can still do anything it does on any other II, however. Aux-slot cards are good for nothing but a cheap source of memory chips on a IIgs.)

The aux-slot cards come with software that modifies AppleWorks so that it uses the memory on the card for an expanded desktop; so that it allows more database records, word processor lines, and spreadsheet cells; and so that it loads itself onto the memory card. Virgin AppleWorks 1.3 automatically does most of this same stuff on standard-slot cards (the amount of database-word processor-spreadsheet expansion isn't as great as with the aux-slot software, however). Virgin AppleWorks 2.0 automatically does this same stuff on both standard-slot and IIgs memory cards (ditto).

It all means that once you start up AppleWorks with one of these cards in place, you will have a desktop with a lot more room for files and you'll have no disk access at all except when you print (the SEG.PR file typically isn't loaded into memory). However, if you fill the desktop so full of files that there's not enough room left for the AppleWorks program itself, AppleWorks automatically goes back to getting program segments from the disk.

If you want to save some money, buy your card with the minimum amount of memory and plug in the maximum number of RAM chips yourself. For a good source of chips, see my introduction to the May 1987 letters, page 3.30. The price difference will buy you at least a couple year's worth of ***Open-Apples****. For a rundown on the differences between the standard-slot cards that are available, see my response to "Auxmem corrections and more" in the January 1987* ***Open-Apple****, page 2.93.*

Higher-capacity disk drives. *Once you get AppleWorks expanded, you may develop a need for higher-capacity disk drives. Using AppleWorks with one of Apple's own memory cards, in fact, you can create files that are too large to save onto your disks. (The expansion software that comes with third-party cards will automatically "segment" a large file so that it can be saved onto several floppies. AppleWorks without third-party expansion enhancements can't do that.)*

Apple's UniDisk 3.5 is an expensive 800K choice that works on any Apple II. Central Point Software sells less expensive 3.5 inch drives for slotted IIs, however, we have several subscribers who couldn't get the Central Point drives to work with their systems and who sent them back. My guess so far is that these drives consume more power than Apple's drives and thus have a greater tendency to make systems that are already loaded down with memory cards and other goodies act flakey. It's just a guess though.

If you need even larger disk drives, there are several kinds of hard disks available for slotted IIs, and at least one for the IIc. In general, however, I recommend avoiding hard drives unless you can write a 10-page proposal demonstrating that you really need that much capacity on line all at once. See "Cheap hard drive alternatives" in our November 1986 issue, page 2.76, and "RamFactor as hard disk" in our December 1986 issue, page 2.86, for more on this.

Uninterruptable power supplies. *Now that you have 750K bytes of work on your desktop and maybe even a 20 megabyte hard disk to save it all on, what are you going to do if the lights go out? Lose all that data and let your hard disk's head crash? See "Reviewer's Corner" in our January 1986 issue, page 1.98, for an answer to this problem. Since that review was written, incidentally, Kansas City Power and Light has lost more than a few electrons, but I haven't lost a single byte of data.*

Accelerators. *Now you've expanded memory and disk capacity, but AppleWorks is starting to seem a bit "slow" when you're working with large files. If you have a IIe or older II, you can get an accelerator card to speed things up. Our favorite accelerator around here is still Applied Engineering's TransWarp, though, like Central Point's drives, it can make your system do strange things if your power supply is working near or beyond its capacity. IIc owners, unfortunately, are out of luck when it comes to accelerators. IIgs owners have one that's almost as good as the TransWarp built right into their machines.*

Macro programs. *Now your appetite is really whetted. How about some software that enhances AppleWorks so that you can press solid-apple plus one other key and see your entire name and address appear on the screen? Or see the cursor immediately jump to the screen's right or left edge? Or see a report printed without choosing a printer, specifying the number of copies, or even saying "yes"?*

The macro program I use everyday is ***AutoWorks*** *from The Software Touch. It includes a very nice mail-merge feature and does everything else I need a macro program to do. However, if you'd like to actually write AppleWorks "programs" that other people can run, you should get* ***Super MacroWorks*** *from Beagle Bros. It is nothing less than a programming language for AppleWorks 2.0. On the other hand, those of you who have a need for pop-up desk accessories such as an appointment calendar, a telephone dialer, a calculator, a communications program, or a spelling checker should look at Pinpoint's* ***Key Player.*** *It's a macro program that's compatible with Pinpoint's other desk accessories.*

Desk accessories. *Pinpoint seems to have run away with this field. Desk accessories are small auxiliary programs that you can run without leaving AppleWorks. Pinpoint has a good selection of such programs. However, they don't exactly "pop" until you have a memory card to store them in. An accelerator helps too, especially with the spelling checker. But once you've got all that other stuff, adding Pinpoint will make your IBM friends wave the white flag.*

In summary, *we spend so much space here in* ***Open-Apple*** *talking about the bugs in and the limitations of AppleWorks that sometimes we forget how much it does do, how easily it does it, and how far its limits can be pushed. If it wasn't so important to all of us we wouldn't spend nearly so much space talking about it.*

Bugs, limitations, and explanations

I'm having trouble running AppleWorks 2.0 with Applied Engineering's AppleWorks 2.0 expander (V 1.3) on my IIgs. I can boot AppleWorks once, but if I try to reboot it, or even restart it, it loads to the getting started screen and then dies. Nothing but turning the IIgs power off for a couple of minutes will get me back to the program. Applied Engineering's technical support could offer no help. Is there any advice you can offer?

Michael Colville
Del Norte, Colo.

It sounds to me like AppleWorks or the expansion software is having trouble allocating memory, perhaps because one or the other doesn't clean up after itself when you leave AppleWorks the first time. It is

particularly important on the IIgs to let programs clean up after themselves, so always use a IIgs program's quit option. Are you running another program after AppleWorks that is causing the problem? Do we have anyone else who has experienced this?

Recently, we have been unable to save AppleWorks 2.0 files to 3.5 disks. This happens whether we are saving directly from the document or using the save command in the main menu.

Marianne Harding
Washington, DC

A local dealer says a user reported the same problem with 600K of disk space remaining. One barrier you may be running into is that of ProDOS's limit of 51 files in a disk's main directory. If you're using subdirectories, we don't know what the problem is. Do we have anyone else who has experienced this?

I think I have found a new bug while using AppleWorks 2.0. It seems that AppleWorks 2.0 will not handle subdirectories with more than 64 filenames in any given subdirectory.

Fritz A. Wonnacott
Gowanda, NY

Dennis created a subdirectory with 100 empty AWP files named AW.TEST.000 through AW.TEST.100 and told AppleWorks he wanted to add a file from this subdirectory. First he saw a message that said:

```
AppleWorks will list only the
first 130 files on this disk.
```

Then he pressed "space" to continue. AppleWorks 2.0 displayed the first 85. AppleWorks 1.2 showed him all 100. We cannot account for any of these numbers. Can anyone else?

I encounter problems with the AppleWorks 2.0 patch from Applied Engineering. From within the report format menu there happen strange things when you try to move a field up or downwards. The field not only relocates, but stays on the same place as well. I tried the standard AppleWorks 2.0 and had none of these problems.

Jack van Soest
Vlaardingen, The Netherlands

I've seen this happen with AppleWorks 2.0 and I don't have Applied Engineering's 2.0 expansion software, so I don't think they're to blame. Has anyone else isolated this problem?

I am using a 1 meg RamFactor with battery back-up plus AppleWorks 1.3. You described the card in the January ***Open-Apple***, page 2.93. AppleWork's habit of taking possession of all available memory for its SYS.DESKTOP file makes it impossible for me to save a data file on the RamFactor. I tried creating dummy files and deleting them as you suggested, but I have to delete one every time I start up AppleWorks and they are used up very quickly. It would be really nice to find a way of restricting the SYS.DESKTOP file to a preset value, like 300K. You also mentioned creating a special partition just for SYS.DESKTOP in January, but how?

Alexander Buchholz
Mainz, West Germany

The only solution we can figure out is to write an Applesoft "pre-boot" program that saves a dummy file in the partition before you start up AppleWorks. Then you can delete that file from within AppleWorks to reclaim some space on the RamFactor for your data files. We agree that the way AppleWorks hogs all available memory is no good. Another bad example appears if you startup AppleWorks 2.0 in a ***SoftSwitch*** *workspace on a IIgs. AppleWorks will take over all the memory in the IIgs. It won't leave anything for the other* ***SoftSwitch*** *workspaces. Thus, to use AppleWorks 2.0 with* ***SoftSwitch****, you have to start it up last, after you already have the other memory you need. AppleWorks should provide some way for users to specify the maximum amount of memory it can use, but doesn't. Our January suggestion of creating a special partition just for SYS.DESKTOP was based on an editing misunderstanding. I thought that you could have two active ProDOS volumes on a RamFactor, but Dennis tells me that isn't true. The RamFactor looks like two 400K disks to DOS 3.3, but other operating systems can see only one disk when looking at the RamFactor.*

A colleague lost an AppleWorks 2.0 spreadsheet file during a save to an almost full disk. No error was reported, but she ended up with a directory entry showing a 9 block file and a file index block pointing to only one data block. Whenever she tried to load the file, Appleworks 2.0 dropped into the Monitor. Ouch! I managed to find all the parts of the lost file using MR.FIXIT from the ProSel disk. It showed seven used blocks in the volume bitmap that were not owned by any files. I figured out the order they should be in by looking at them with BLOCK.WARDEN (also on the ProSel disk) and then patched the file index block and the file directory entry to recover the file.

Jim Luther
Kansas City, Mo.

Releasing the power

Can you tell me about the Apple power supply? In a catalog I read that it's rated at 2.5 amps. In my IIe's slots I have devices that are pulling 2.1-2.4 amps, based on the catalog's estimates. I'd like to buy some more cards that will pull another .9-2.1 amps. Are the cards still drawing current if they are installed in the Apple but not active? Am I in danger of overloading my Apple? If so, what are my alternatives?

Robert Albanito
Chicago, Ill.

(For those of you who are new around here, here's some essential background information. The "power supply" is the large metal box you'll find inside your slotted II. It converts a range of AC voltages into four specific DC voltages. On the II-Plus and IIe it is held in place by four screws you can access from the bottom of the computer. On the IIgs it just kind of snaps in and out. Replacing a power supply is only slightly harder than changing a light bulb and considerably easier than hooking up a video cassette recorder. The "power supply" we're talking about here is totally unrelated to the "uninterruptable power supply" I talked about earlier, which is a larger box with a battery inside that goes between your wall outlet and your computer and keeps the AC juice flowing to your computer for 20 minutes or so after a power failure.)

Some cards, memory cards for example, draw the same amount of current whether they're active or not. Other cards, disk drive controllers, for example, can pull far more current when they're active than when they're inactive.

Are you in danger of overloading your Apple? Our answer is a clear "yes." Unfortunately, the symptoms of an overloaded Apple aren't at all clear. It doesn't just start smoking or blow a fuse or something easy like that. Typically the computer starts doing such things as erasing Track 0 on your floppy disks or putting strange characters on your screen or locking up at inopportune moments. The problem is often "fixed" by removing a likely card from your computer and heaping blame and disgust on that card for all the trouble you've been having. Applied Engineering's TransWarp and Central Point's 3.5 disk controller are two power-hungry cards that often end up in the cross fire.

For some background reading, see "Power Supplies and Track 0" in our May 1986 issue, page 2.29, "Users battle technical support" in our January 1987 issue, page 2.96, and "More TransWarp experiences" in March 1987, page 3.15. In my response to the last of those letters I mentioned that some strange problems with my own IIe had been solved by removing the TransWarp. A couple of weeks later the power supply in that computer died (that is, nothing would happen when I flipped the power switch; not even the power-on light would come on). I replaced it—it happened to be one of Applied Engineering's heavy duty power supplies—with the computer's original Apple supply, put the TransWarp back in, and haven't had any trouble since. Which is not to say I won't have trouble tomorrow. The computer is full of cards and, consequently, the power supply is working beyond its rated capacity. Inevitably, it will buckle again someday.

The Apple II world badly needs a well-built, highly-reliable, heavy-duty replacement power supply for those of us with lots of cards in our computers. We know of four suppliers of replacement power supplies for the Apple II, but they're all selling Taiwanese units that vary in quality. ***Open-Apple*** *subscriber Don Justesen, for example, reports that some units have only one ground wire running to the motherboard, where Apple's units have two. Some have connectors that fail to make good contact with the motherboard, either because the connecting elements slip inside the connector or because the female connectors simply aren't tight enough.*

Most of these replacement units have higher ratings than Apple's original power supply and all are far less expensive than exchanging a broken Apple supply for a rebuilt one down at your local Apple dealer. However, based on reports we've received, as well as personal experiences, we have to say that the Taiwanese units appear to have significantly shorter life-spans than Apple's. Our experience to date indicates there's no reason to replace a working Apple supply with a heavy-duty Taiwanese replacement as a "preventive-maintenance" measure. If you are actually experiencing problems with your system, however (and by "problems" I mean any kind of unexplainable flakiness), the power supply is one of the first things you should suspect as the trouble-maker.

Here's a list of the U.S. sources for replacement power supplies that we know about (prices were as of June 12). We suspect that the quality of the units they sell varies from shipment to shipment, and we know of no way to assure that any power supply you buy from them is better than a rebuilt one from Apple, even though the "rating" may be higher. But at least it will be new and it will be cheaper.

Applied Engineering, P.O. Box 798, Carrollton, TX 75006, 214-241-6060 has a heavy-duty supply for $69. BTE Computers, 14644 N Cave Creek #6, Phoenix, AZ 85022, 602-867-8962 has a normal-duty

supply for $44 and a heavy-duty supply for $59. Jameco Electronics, 1355 Shoreway Rd, Belmont, CA 94002, 415-592-8097 has a normal-duty supply for $35. JDR Microdevices, 110 Knowles Dr, Los Gatos, CA 95030, 408-866-6200, has a heavy-duty supply for $50.

If any of you know of a company that can provide, at any price, a high-quality heavy-duty Apple II power supply, please let us know.

& word processor

I need a text editor that I can use from Applesoft. Do I have to write it or can I buy a subroutine somewhere as a programmer's utility? I would like to allow the user to enter/edit up to 15 lines of text (40 column version and 80 column version) with the features of insert, delete, and word wrap. When the user exits the editor, the editor would deliver the lines (as strings) to my Applesoft program for filing.

Michael Trusiewicz
Ed Tech Systems
68 Adelaide Avenue
Waterbury, CT 06708

Sounds like what you want is an ampersand command that would provide a small, screen-oriented word processor without scrolling. We don't know of any such product, but our readers might.

A solution of sorts

I really enjoyed "Sorting out sorts" in the November 1986 issue of ***Open-Apple*** (page 2.80). I also read with interest the letter by Craig Wilford and your reply in the March issue ("Only exit FOR-NEXT at NEXT," page 3.15) in which you discussed the out of memory problem. The problem is that after reading this a reader could feel out of sorts. Modifying the sort as shown in March will result in improperly sorting the array.

In response to your mention of the excellent *Beagle Compiler*, please let your readers know that there is a significant bug in all versions through 2.2 of the extended memory compiler (AUX.SLOT.SYSTEM), which results in abnormal functioning of the CHAIN command. Alan Bird has been updating this program almost weekly, and seems to have all the bugs ironed out in version 2.3. I can't recommend the program highly enough.

Bruce S. Klutchko
New York, N.Y.

This is a case of adding a bug by trying to fix a bug. It happens all the time, unfortunately. To summarize, Jim Parr's original subroutines in the November issue work just find, but they break a rule about exiting from the middle of a FOR-NEXT loop. To keep them from breaking that rule is fairly complicated, but can be done, not as shown in March, but by changing line 940 and adding line 965:

```
940 AB = 0 : IF A(E) <= T THEN AB = 1 :
        EX = E : E = E1 : GOTO 960

965 IF AB = 1 THEN E = EX
```

We use two new variables, AB (abort flag), and EX (exit value of E). We set the abort flag to 0 ("not aborted"), do our comparison, and if the comparison succeeds we set the abort flag, save the current value of the loop counter in EX, set the counter to the final value, and GOTO the NEXT statement. Line 965 has been added to see if we exited the loop early; if so, we change E back to the value it had when the decision to abort was made. This (finally) works, and is readable.

Good public stuff

Since you mentioned the public domain ProDOS word processor *Freewriter* back in May 1985 (page 36), I thought you might like to know that there's a revision of the program called *FrEDwriter*. Although it has its limitations, *FrEDwriter* is better than many commercial word processing programs. As similar as it is to *Apple Writer*, and since it works in 40 or 80 columns on any Apple II, and since it's free, *FrED* is ideal for school use. Of particular note is that a printer control panel has been added inside the main program.

Bob Garth has placed his *ProTree* ProDOS BBS system in the public domain. For anyone planning to start or convert to a ProDOS BBS it is worth a look. It was marketed commercially for awhile and is currently being used by a number of BBS's.

Some shareware programs that I think have exceptional merit are *Nifty Works*, *Omnifile*, and *Bank'N*. *Nifty* is a DOS 3.3 word processor with a scroll speed of about mach 1. *Omnifile* is an excellent DOS 3.3 database, and *Bank'N* is an also excellent DOS 3.3 checkbook management program.

If anyone wants a copy of any of these, I will distribute them at no cost. All a person needs to do is send me disks for the copies, return postage, and a mailing label. I'll also enclose a list of 27 other programs that I'll distribute on the same basis. Any additions to the collection will be gladly accepted.

Ed Thompson
11768 East Atlantic Place
Aurora, CO 80014

More on elusiveness

Regarding the comments in your March issue (page 3.13) on the relative speed advantage enjoyed by the 65816 et al over the 8088-type processors: while the cited advantage exists, it almost certainly has nothing to do with pipelining. The 8088 implements pipelining with its prefetch cache, thus the same capability would bring the 65xxx-family up dead even. I think the advantage cited by Mensch refers to the differences in which the processor clocks are utilized. With many processors, e.g. the 8088 and friends, the Z-80, and 68000, a machine cycle may take four or more clock cycles. The 65xxx family, including the older 6502, use only one clock cycle for each machine cycle. Thus a 1 MHz 6502 is effectively running as fast as a 4 MHz Z-80, and a 6 MHz 65C816 is matching a 24 MHz 8088 (which doesn't exist).

The exact nature and relationships between clock cycles and machine cycles is quite complex and variable, especially when different microprocessors are involved, as in this case. Instruction cycles, as distinct from clock and machine cycles, complicate things further.

A comparison of clock speeds alone reveals very little about relative performance of various processors: relative machine and instruction cycle speeds, bus bandwidths, register architectures, and instruction-set capabilities are more revealing. A 6 MHz 65816 is not the equal of a 24 MHz 68020, for instance, in usable performance.

Additional factors: the type of software being run affects performance—code optimized by a skilled programmer for a 6502 may run poorly when translated literally for an 8086, and vice versa. A fair comparison might involve code optimized for each processor tested, with strengths highlighted and flaws concealed. Applesoft, for instance, is a poor test of a 6502, since the Microsoft interpreter, from which it is derived (with little optimization) was written for the 8080A. The Integer BASIC interpreter, which was written for the 6502, is faster, cleaner, and has far fewer bugs.

F. Kuechmann
Vancouver, Wash.

Corona, Rana, ProDOS

A company called Law Industries (3387 West Corning St, Newbury Park, CA 91320, 805-498-8268) has developed an upgrade kit for Rana and Corona hard disks that allows you to add ProDOS. I am a satisfied customer. The kit cost me $49.95.

Phil Hackett
Los Angeles, Calif.

Thanks for the tip. I just told someone last week I'd never heard of a ProDOS upgrade kit for Rana hard drives and now we have this.

Under development

When you see the bouncing Apple on your Apple IIgs (to get *there*, boot up without any disks in your drives), press open-apple/option/control-N.

Mike Mitchelson
Shawnee Mission, Kans.

Now that beats inside the case any day.

In the opinion of Uncle DOS

I'm considering the purchase of an Apple IIgs, Apple RGB Monitor, Epson EX-800 printer, two 3.5 inch drives and two 5.25 inch drives, and an Applied Engineering gsRAM card. Who makes the cables that connect 6 pin circular Epson to 8 pin circular Apple IIgs? If the UniDisk 3.5 is intelligent and the Apple 3.5 is dumb, does it create an advantage to buy the UniDisk if the Apple IIgs doesn't use this intelligence? Why is it better to buy the UniDisk? What difference is there between the RamFactor and gsRAM for the Apple IIgs? I'm totally confused. Can you recommend other items in my setup?

Bob Levy
New York, N.Y.

I'm not familiar with the Epson printer you mention, but unless you are sure it can perfectly emulate an ImageWriter, I recommend sticking with Apple's printer. At least 10 per cent of the mail we get around here has to do with how to set up non-ImageWriter printers with standard Apple software. Finding a cable is just the first problem you buy when you try to save some money on a printer.

The UniDisk 3.5 will work on a IIe or IIc. The Apple 3.5 won't. The Apple 3.5 will work on a Macintosh. The UniDisk 3.5 won't. If you don't have a IIe or a IIc or a Macintosh, then these points are moot. On the IIgs, the Apple 3.5 can be made to respond faster than the UniDisk, using programs such as Bill Basham's ***Diversi-Cache*** *(see May 1987, page 3.30). On the other hand, the UniDisk 3.5 can theoretically be turned into an Apple 3.5 by removing an internal logic board, but we haven't seen this done yet. The Apple 3.5 can never be turned into a UniDisk.*

The RamFactor is a standard-slot card. The gsRAM is a card for a special memory slot inside the IIgs. You need a minimum of 512K of memory in the special IIgs memory slot to get the machine to do much of anything interesting. If it were my system, I'd get at least a megabyte of each, or the equivalent. To pay for them I'd buy just one 3.5 and one 5.25 drive. Let the RamFactor be your drive 2. It's bigger, faster, quieter, and about the same price. On the

downside, it requires an unused slot and needs some kind of electrical backup in case of power failure.

I also suggest you take a close look at the alternatives to Apple's RGB monitor. You can get a free TV with your system from Sony (see "RGB and TV too" in our March issue, page 3.13).

Stop that MIDI desktop

Is it true that the IIgs will not handle a direct interface from a MIDI keyboard to the internal synthesizer? Will desktop publishing (so prevalent on the Mac) come to the IIgs with a LaserWriter and Postscript? Is there a problem with interrupt handling on the IIgs?

John Hammond
Greenville, Texas

The IIgs has no built-in MIDI interface. A plug-in card could be (and no doubt will be) designed for the IIgs to connect a MIDI keyboard to the machine's sound synthesizer.

There is no doubt that desktop publishing will come to the IIgs, but remember that it took three years for it to arrive on the Macintosh. Patience.

I don't know of any specific problems with IIgs interrupt handling. There are apparently some problems with unclaimed interrupts (see "Insert system disk and..." in our April issue, page 3.18), but they are shared by all Apple IIs running ProDOS 8.

IIgs memory manager revealed

In regard to "Insert system disk and..." in your April issue—ProDOS can't simply ignore an unclaimed interrupt. Since the interrupting source hasn't been cleared, the computer would immediately be interrupted again. This would continue in an infinite cycle and the computer would hang. Asking the user if interrupts should be disabled is a job for an adventurous hacker type. It would involve swapping the screen somewhere while the message is displayed and modifying the status byte on the stack for the final RTI instruction. This will be tough because the interrupt code is very different depending on which Monitor ROM is being used.

In regard to your problems with the IIgs memory manager's MMStartUp call ("Odd bank out" in the same issue, page 3.22)—the problem is that MMStartUp doesn't issue new user IDs. It looks to see where in memory the MMStartUp call was made from and returns the ID of the owner of that memory block. If you get an error, your code is in an unallocated block. You'd better ask the memory manager to allocate the area your code is in before you ask for any other blocks or you could wipe out your own code.

To allocate the area you are executing in, first call GetNewID (tool $2003 in the miscellaneous tool set) to get a user ID. Use that ID with NewHandle (tool $0902 in the memory manager tool set) to allocate the memory you need. When your application is finished, call DisposAll (tool $1102 in the memory manager tool set) to deallocate your memory, and DeleteID (tool $2103 in the miscellaneous tool set) to release your ID back to the system.

Contrary to popular belief, MMStartUp doesn't issue ID numbers. Likewise, MMShutDown doesn't free up the memory blocks you allocate (no, this is not a bug).

Rich Williams
Apple Computer, Inc.

There you have it folks, from the author of the IIgs memory manager himself. While from ProDOS 8 the MMStartUp call appears to be a massive Catch-22 (you're not supposed to ask for memory until you've started up the memory manager, yet you can't successfully start the memory manager until you own a hunk of memory), from ProDOS 16 it makes more sense. Under ProDOS 16 the System Loader will have already allocated your application's memory, so instead of returning an error, MMStartUp tells your application what its user ID is. Under ProDOS 16 it appears that MMStartUp is assigning the ID, but in fact the System Loader has already gotten the ID for you from the ID manager and has gotten your memory for you from the memory manager. Under ProDOS 8, on the other hand, you have to do all this stuff yourself. In return, under ProDOS 8, MMStartUp and MMShutDown are "do-nothing" commands that you can ignore.

The essential syntax for making a IIgs tool call was published here in "Time to look in the toolbox" in our April issue, page 3.21.

***The GetNewID call** (LDX #$2003) requires that you push two bytes on the stack for a result and two bytes specifying the type of ID you are requesting. Here are the possible types you can request:*

```
$0000 = memory manager        $8000 = firmware
$1000 = application           $9000 = tool locator
$2000 = control program       $A000 = setup file
$3000 = ProDOS                $B000 = undefined
$4000 = tool sets             $C000 = undefined
$5000 = desk accessories      $D000 = undefinsd
$6000 = runtime libraries     $E000 = undefined
$7000 = system loader         $F000 = undefined
```

*Obviously, $1000 is the type of ID folks like us will use most of the time. The call will return a 2-byte ID. The high byte will contain the ID type you have requested, the low byte will contain a number between $01 and $FF. The two bytes combined give you a unique user ID. Since there are a limited number of possible ID numbers, it's necessary that you call DeleteID when your program is finished. If you don't and somebody runs your program a number of times in succession without cold-starting the system, there will eventually be no ID numbers left and GetNewID will return its only possible error, $030B (no IDs left). **DeleteID** (LDX #$2103) requires that you push your 2-byte ID on the stack before calling it. It returns nothing and never generates an error.*

The low nibble of the high byte (for those of you whose eyes just glazed over, this means the digit with the "x" in it in $0x00) is called the "aux ID field." You can use it when you request memory from the memory manager to assign your memory blocks up to 16 of your own ID numbers. If you need more memory blocks than that you could group them into 16 types—you can request as many blocks as you want using each ID. (You can also just leave the aux ID set to zero all the time and forget you ever heard about it. I suspect it is about as useful to most of us as a write-protect tab.)

***You allocate yourself a block of memory,** once you have your ID, by calling **NewHandle** (LDX #$0902). To use this tool, push 4 bytes for a result, 4 bytes specifying the size of block you want, your 2-byte user ID, a 2-byte memory attribute byte (details upcoming), and a 4-byte address specifying where in memory you want the block to be (even if you don't care where the block ends up you have to push 4 garbage bytes here). The call can return three errors—$0201, unable to allocate this block; $0204, illegal operation on a fixed or locked block; and $0207, invalid user ID.*

The call returns a 4-byte "handle." Save this. It's a fixed location in the IIgs memory that will always point to the beginning of your memory block. The memory manager doesn't return the actual address of your block because if it later had to move it, the address it gave you would be wrong. Instead, it gives you the address of a pointer that will never move and promises to update that pointer with the latest address of your memory block every time it moves things around.

An interesting bit of trivia is that the word "handle" was handed down to the IIgs development team from the Macintosh development team. The Mac team named handles after one Ham Handle, who was a permanent resident of Iowa City, Iowa in the late 1800s. Handle became famous for knowing the current address of everyone in south-eastern Iowa. Whenever people wanted to know where someone was living, they would ask Ham for the current address.

***The 2-byte memory attribute** specifies the properties of the memory block you want the memory manager to allocate. To make things simple, think of the attribute as four hex digits called $ABCD.*

The A digit can be set to two interesting values. A zero indicates that you are willing to let the memory manager move your block of memory around, at least some of the time. A 4 indicates that you never want your block to be moved. You should try to use a 0 here as much as possible. Machine language instructions that aren't written to be relocatable, however, will have to be placed in blocks that are "fixed" by means of a 4 here. (The high bit of this digit indicates whether or not the block is "locked." We'll talk about this later.)

The B digit is the "purge" level of your block. This can be a number between 0 and 3. A 0 means the block cannot be purged by anything except a DisposHandle or DisposAll tool call (upcoming). For the time being, stick with a zero here. If you use a 1 or 2 (3 is reserved for the System Loader, all 2s will be purged before any 1s) and the memory manager needs some memory, it may borrow your block. In this case, the block's handle will be set to all zeros, or nil. If you need to use the block again later and find a nil handle where you expected an address, you have to use a tool called ReAllocHandle to get the block reallocated and then reload it with whatever data it held (you had it saved on a disk somewhere, didn't you?)

The C digit can only be 0 or 1, at least for now. A 0 indicates the block may cross a bank boundary. A 1 indicates it may not.

The four bits of the D digit are best handled separately.

The 8s bit indicates whether or not the memory block can use "special" memory (0=yes, 1=no). Special memory includes all of banks $00, $01, and the video screen areas of banks $E0 and $E1.

The 4s bit indicates whether the memory block needs to be "page-aligned" (0=no, 1=yes). Page-aligned means the last two hex digits of the address are zero. For highly technical reasons, page alignment is needed with any machine code that includes timing loops.

The 2s bit indicates whether the memory block needs to be at a certain, specific address (0=no, 1=yes). The 4-byte address you push on the stack before calling NewHandle is ignored unless you set this bit or the next one to 1. If this bit is 1, all four bytes of the address you specify are used. Based on

tests I've done so far, there doesn't seem to be any way to request, for example, the area from $2000 to $4000 while saying that you don't care what bank you get. Such an ability would be handy for certain types of non-relocatable code, so I may be mistaken about this.

The 1s bit indicates "fixed bank" (0=no, 1=yes). However, on tests I've done, it only seems to work when both the special-memory bit and the bank byte of the four-byte address are cleared to 0. In that case, you'll get a memory block somewhere in bank $00. If the bank byte of the four-byte address is set to anything other than $00, this bit seems to have no effect on what bank you get. I'd sure like to hear that someone replicated this experiment, however.

Note the difference between the fixed-address and fixed-bank bits of the D digit and the fixed-location bit of the A digit. The D-digit bits have to do with where the memory manager puts the memory block when it allocates it, while the A-digit bit has to do with whether or not the memory manager has permission to move the block later. If you set either of the D-digit bits to 1 you should also set the A-digit bit so that the block you've so carefully specified doesn't get moved elsewhere later.

***The memory manager provides a variety of tools** for manipulating memory blocks, once you've allocated them. We've already touched on ReAllocHandle, DisposHandle, and DisposAll. PurgeHandle and PurgeAll will force the memory manager to release specified blocks of your memory without your having to give up their handles. What they're good for escapes me. The -All commands, incidentally, are sensitive to the aux ID you pass to them. If the aux ID digit you give is 0, then the action occurs regardless of what the aux ID digits of your memory blocks are. If the aux ID you give is not 0, on the other hand, then the action occurs only to your blocks that have that aux ID.*

GetHandleSize returns the size of a specified block, SetHandleSize changes the size of a specified block, CompactMem causes the memory manager to move those things that are movable around in an effort to maximize the largest block of free space, FreeMem returns the total number of free bytes in memory, MaxBlock returns the size of the largest block currently available (use CompactMem first to find the largest block possible), and TotalMem returns the size of all memory in the IIgs, including the main 256K.

HLock, HUnLock, HLockAll, and HUnLockAll are used to change the "lock" status of specified blocks. A locked block has the high bit of the A-digit set to 1. When a block is locked, it can neither be purged nor moved. Note that when a block has its purge level in the B-digit set to 0 and the "fixed-location" bit of the A-digit set to 1, it is as good as locked whether it is or not. The lock commands are primarily used to temporarily fix the position of a block you originally defined as movable. For example, any relocatable machine code that's in a movable block should lock its position before using NewHandle, ReAllocHandle, SetHandleSize, or CompactMem. Otherwise, the code could pull the rug out from under itself, which is a sorry thing to see, indeed.

SetPurge and SetPurgeAll allow you to change the B-digit purge level of a block. BlockMove, PtrToHand, HandToPrt, and HandToHand are four tools that allow you to move a specified number of bytes from a source address to a destination address. That's all there is to the memory manager. Space prohibits going into more detail on most of these tools. The only two tools you really need to know about to get started, however, are NewHandle and DisposAll.

***DisposAll** (LDX #$1102) requires one 2-byte input, the user ID whose handles will be discarded. It returns nothing and has only one error, $0207, invalid user ID. This command ignores both the purge level and the lock status of a block—its purpose in life is to clean out the memory you've been using when you're ready to quit. If the aux ID is 0, every block belonging to you will be discarded. If the aux ID is set to anything else, however, only blocks with the specified aux ID will be effected. After using DisposAll, don't forget to also use DeleteID.*

If you would like to experiment with these commands, there is one very handy tool built right into the IIgs that you should know about. It's a desk accessory that Williams wrote. To start it up, do this:

```
]CALL -151

*0=e  (must be a lowercase "e")
*FF/1B00x

*
```

Now press open-apple/control/escape to see the desk accessory menu. You'll see a new item called "Memory Peeker." Select it. It will put a one-line menu at the top of your screen. Press "U" to see the blocks that the memory manager has allocated (you may have to press ctrl-S to keep them from scrolling out of sight). You'll see each block's handle, its address, its attributes, its ID, and its size. Have fun. Be forewarned, however, that Williams won't promise that this educational tool will be in future revisions of the IIgs ROMs, so use it to learn how the memory manager works now, while you have the chance.

Reboot, cold reboot, frigid reboot

...There's also a way to get the IIgs memory manager to give you some memory without a user ID. It took me awhile to pry it out of Rich Williams, and he told me the secret only after making me promise not to use it about four times. All you need to do is use the ID $0001. This is the ID of the memory manager itself, which is always active. Using this ID means the memory you get is *never* de-allocated, even after a control/open-apple/reset cold-boot. Only a power-off reboot can reclaim the memory.

This naturally leads to the question of how you can do a power-off reboot without turning the power off. Remember all those worn out power switches on the II-Plus? The IIe's control/open-apple/reset was supposed to eliminate that problem, but on the IIgs, Apple wanted the RAMdisk to live through a reboot. You may have noticed that most of the control-panel stuff either takes effect immediately or it takes effect after a control/open-apple/reset. But if you change the control-panel RAMdisk settings, nothing changes until a complete power-down of the system.

To accomplish this, Apple added a new power-up byte and tucked it into the RAM that belongs to the keyboard microprocessor. This is the RAM that holds the keyboard type-ahead buffer, among other things. Byte $51 is the power-up byte. Here's how to trash it and force control/open-apple/reset to destroy and reallocate the RAMdisk along with everything else:

```
CALL -151

0:51 0

\8 0 0 2 0 0 0 0 0 8 9 9\U
```

As a machine language routine it would be:

```
CLC
XCE
REP #$30
LDA #$0051
STA $0000
PEA $0002
PEA $0000
PEA $0000
PEA $0008
LDX #$0305
JSL $E10000
SEC
XCE
RTS
```

One last IIgs goodie for you. How do you tell you're in a IIgs? Apple suggests a rather long machine language routine. However, I propose using an easier way as follows:

```
IF PEEK(-1) = 192 THEN you're in an Apple IIgs
```

The location contains the high byte of the interrupt vector. On the IIgs, the interrupt vector points to $C074, which contains some special ROM that exists only in the IIgs. The II-Plus, IIe, and IIc computers have no ROM in page $C0. And I suspect it's unlikely that the IIgs interrupt vector will move in future versions since it's very hardware dependent.

Bill Basham
Diversified Software Research
Farmington, Mich.

Open-Apple

[redacted]

is written, edited, published, and

© Copyright 1987 by
Tom Weishaar

Business Consultant Richard Barger
Technical Consultant Dennis Doms
Circulation Manager Sally Tally
Business Manager Sally Dwyer

Most rights reserved. All programs published in ***Open-Apple*** are public domain and may be copied and distributed without charge. Apple user groups and significant others may reprint articles from time to time by specific written request. Requests and other editorial material, including letters to Uncle DOS, should be sent to:

Open-Apple
P.O. Box 7651
Overland Park, Kansas 66207 U.S.A.

Published monthly since January 1985. World-wide prices (in U.S. dollars; airmail delivery included at no additional charge): $24 for 1 year; $44 for 2 years; $60 for 3 years. All single back issues are currently available for $2 each; bound, indexed editions of Volume 1 and Volume 2 are $14.95 each. Volumes end with the January issue; an index for the prior volume is included with the February issue. Please send all subscription-related correspondence to:

Open-Apple
P.O. Box 6331
Syracuse, N.Y. 13217 U.S.A.

Open-Apple is available on disk from Speech Enterprises, P.O. Box 7986, Houston, Texas 77270 (713-461-1666).

Unlike most commercial software, ***Open-Apple*** is sold in an unprotected format for your convenience. You are encouraged to make back-up archival copies or easy-to-read enlarged copies for your own use without charge. You may also copy ***Open-Apple*** for distribution to others. The distribution fee is 15 cents per page per copy distributed.

WARRANTY AND LIMITATION OF LIABILITY. I warrant that most of the information in ***Open-Apple*** is useful and correct, although drivel and mistakes are included from time to time, usually unintentionally. Unsatisfied subscribers may return issues within 180 days of delivery for a full refund. Please include a note from your parents or children confirming that all archival copies have been destroyed. The unfulfilled portion of any paid subscription will be refunded on request. MY LIABILITY FOR ERRORS AND OMISSIONS IS LIMITED TO THIS PUBLICATION'S PURCHASE PRICE. In no case shall I or my contributors be liable for any incidental or consequential damages, nor for any damages in excess of the fees paid by a subscriber.

ISSN 0885-4017 **Source Mail: TCF238**
Printed in the U.S.A. **CompuServe: 70120,202**