Open-Apple ™

Releasing the power to everyone.

Let's make one thing clear

Apple has named its new software company Claris Corp, according to a mid-July Apple press release. The name is to reflect "clarity and distinctiveness, and to remind us that shaping the future requires a clear vision," said William V. Campbell, president of the new venture. As predicted in last month's *Open-Apple* (page 3.41), the new company's name was derived from a foreign tongue — *claris* comes from the Latin word for clarity. (OK, OK, so I said it was going to be Sanskrit — it was a better guess than anything you saw in *Nibble*, wasn't it?) Claris expects to move into new headquarters in Mountain View, California by October.

(While most of you seem to have no trouble separating *Open-Apple's* baloney from its technical accuracy, it has become clear in the last few weeks that I should warn the grim-faced few among you that I do occasionally attempt to spice *Open-Apple* up with a little humor, and that such attempts often involve exaggerations and outright lies. The lies are supposed to be preposterous enough that you recognize them as whimsy. Lest some gullible private eye should waste days looking for the "sources close to the company, who asked not to be identified," who tipped *Open-Apple* off to Claris' foreign name, let me assure you that such sources do not exist. I just got lucky. Likewise, some of you seem to believe (or at least think *I* believe) that *Open-Apple* really had something to do with Apple's decision to start this new company. Contrary to the bloated claims made here in June (page 3.37), I am quite sure *Open-Apple* had nothing to do with the decision. I have no evidence that any of the high-level executives at Apple so much as read this newsletter, much less take it seriously (although there *is* a copy of Steve Wozniak's three-year subscription payment hanging on the wall around here somewhere). I figure *Open-Apple* has the presence and influence of a pesky house fly at Apple Inc, but no more. I tell you this so that you have a less dim idea of what, herein, is supposed to be preposterous, and therefore humorous. If, gentle readers, you don't stop taking my soliloquies so seriously, I'm going to change the name of this newsletter to *Open-Addle*.)

They answer the phone "Beagle Bros/Software Touch" at the Beagle Bros number these days. Bert and Sharon Kersey, the husband-wife team who founded Beagle Bros, have sold it to Mark Simonsen. Simonsen, along with Alan Bird, was a founder of the Software Touch. The two companies are being combined and a bevy of new AppleWorks enhancement programs, to be called the *Time Out Series*, is being readied for a September introduction.

Simonsen and Bird were Beagle Bros programmers before they started the Software Touch. Kersey remains as a consultant to the company. He says he's still interested in computers and the Apple II but is tired of running a business.

You'll be able to shake Kersey's hand, as well as those of other infamous Apple II scoundrels, at the upcoming AppleFest in San Francisco. The dates are Friday, September 18 through Sunday, September 20. The place is the Civic Auditorium. On Friday and Saturday the show's exhibits will be open from 10 to 6, on Sunday from 10 to 4. Conferences will also be held all three days. Conference keynote speakers are Steve Wozniak and Del Yocam, Apple's Chief Operating Officer. Entrance to the exhibits for all three days is $15, exhibits plus the full conference will be $50. American Airlines is supposedly offering special fares for attendees, call 1-800-433-1790 and ask for Star File #S13312.

You'll occasionally find Kersey at the Beagle Bros/Software Touch exhibit. Go in the front door of the auditorium, walk down the left side of Apple's exhibit, and continue walking toward the back of the auditorium. You can't miss it. It's the big booth in the back. Walk all the way around it and head back toward the front door and there, in its shadows, you'll find *Open-Apple's* tiny booth, #222. Stop by and meet the family. Bring money. *Open-Apple* will be having its first-ever sale on books, back issue volumes, and maybe even subscriptions.

Remember the one about the chemists who changed graphs into sound? (See "Drumming insight out of data," March 1986, page 2.9 and "*Apple Assembly Line*," September 1986, page 2.58.) Now biochemists at Michigan State have developed a musical way to read the graphs turned out by an automated analyzer, according to the June 17 *Wall Street Journal*.

"'The ear is fantastically discriminating' in detecting differences in notes and changes in tempos, explains biochemist John F. Holland. The ear can detect differences that the eye might miss, particularly when the person is tired or bored while poring over data.

"Profs. Holland and Charles C. Sweeley adopted the idea for their research and began telling others. The first response was usually humorous; people would ask us what this week's top 20 urines were,' Prof. Holland says. But now researchers have begun adopting the idea for graphs turned out by blood analyzers used in clinical laboratories and electrocardiograms.

"The biochemists have called in two music professors to help convert graphs that depict normal results to recognizable tunes. 'A normal urine, for instance, might play "America the Beautiful," which is a tune everybody carries around in their head,' Prof. Holland explains. If the machine suddenly sounded a sour note, the technician would immediately know an abnormal urine had just been analyzed."

There are some amazing discoveries yet to be made in this area and the Apple IIgs is the perfect research ship to make them in.
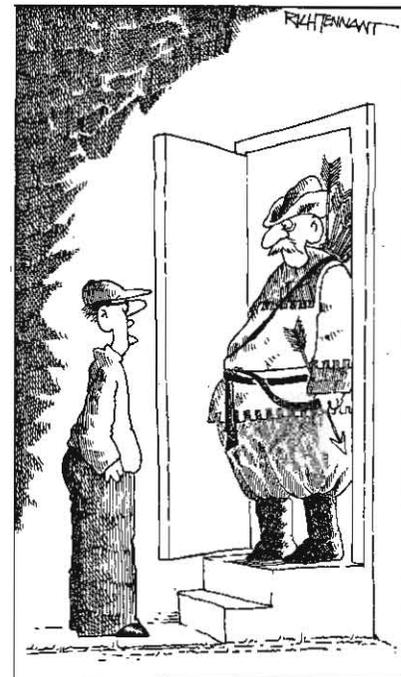
"I WAS TOLD THIS IS WHERE THE APPLE USERS GROUP MEETS."

Ask (or tell) Uncle DOS

The other DOS 3.3

I enjoyed your little article on trademarks that appeared in the July 1987 issue of *Open-Apple*.

I found it particularly entertaining in light of something that recently happened to me. I saw an article in a magazine (that shall remain nameless) and prominent in the title was the name "DOS 3.3". Hey, I said to myself, I know something about that, this may be interesting. I started reading the article and it didn't take long to realize that something was very wrong. It seems that the latest version of MS-DOS is—you guessed it—3.3. So now when you see the term DOS 3.3 it is not going to be the DOS that we all know and sometimes love, but some hideous impostor. There ought to be a law.

Jesse D. Sheinwald
Seaford, N.Y.

The same thing happened to me. It reminds Dennis and me of when the IBMers appropriated the widely-used term "PC" and converted it from initials that meant any personal computer into initials that meant one specific (and now defunct, as far as IBM is concerned) computer.

How about a boycott?

I would like to bring to your attention another case in which legitimate use of a software product by the owner is interfered with by short-sighted copy-protection policies. The company in this case is Electronic Arts, and the specific product is *Deluxe Paint II* for the Apple IIgs.

After seeing a more-than-impressive demo at my local Apple dealer's and immediately deciding on a purchase, I was deeply disappointed to discover that the program out of the box would not boot on my IIgs, apparently because of its copy-protection scheme. The protected program refuses to recognize the supplied "key disk" in my Mikon 3.5 inch drive, connected to a Central Point UDC card in slot 7. Technical support personnel at Electronic Arts were courteous but not helpful—the company's position is that they do not guarantee their software to run on any configuration except pure off-the-shelf Apple hardware (and they admitted that even owners of the Apple UniDisk 3.5 had reported problems). Their suggestion was that I send in a coupon enclosed in the package, along with $20, and wait two weeks (the coupon says 4-6 weeks) for delivery of an unprotected version, which they feel will run on any configuration.

As a long-time Apple owner and as purchaser over the years of scores of software items, the ownership of none of which I have abused, I feel insulted by Electronic Arts' presumption. Not only does their policy penalize users who, like me, believe that third-party vendors have a legitimate and important role to

play in the marketplace, but they seem to overlook the fact that much of the appeal of the IIgs is that a purchaser (in theory) doesn't have to buy everything all over again, but should be able to bring along all his favorite peripherals. There is also the question of whether the large number of IIe owners who can be expected to upgrade their systems in the next year or so will be able to run Electronic Arts software without encountering annoyances similar to mine.

I have gone along (under protest) with Electronic Arts' suggestion, but only because I have determined *Deluxe Paint II* to be a "must have" item. It is the fortunate company that can manage to publish only items in this category, and when in the future I am faced with making or recommending a choice between purchase of comparable products I will surely favor those companies (in my experience the majority) who trust and support their customers.

R. W. W. Taylor
Rochester, N.Y.

*The only point at which you and I differ is your opinion that **Deluxe Paint II** is a "must have" item. Copy protection is stupid. It not only harms customers, in the long run in harms products and publishers. It's a bad **marketing** strategy, to say nothing of being bad in every other way.*

*With the IIgs in particular I advise you to simply stay away from copy-protected software. It's dangerous stuff. It can mess up your system and mess up your disks. You can't put it on mass storage devices such as RAMdisks, ROMdisks, or hard disks. You can't use a program selector with most of it. As you've experienced, it often doesn't work if the hardware isn't just right. (Activision's IIgs protection scheme fails if you have a 5.25 floppy controller or a mouse card plugged into your system—whether the card is turned on or not.) Copy-protected software often doesn't work on later, more advanced computers. And all of this is nothing compared to the problem you have when your copy-protected disk goes bad. With rare exceptions, we simply don't talk about copy protected software in **Open-Apple**.*

*There is a wealth of non-copy-protected software in the Apple II kingdom. You don't need to mess with the copy-protected stuff. Look to companies who don't use copy-protection for your software needs. For example, there's Apple itself, Beagle Bros/Software Touch, Pinpoint, Penguin, Central Point, WordPerfect, Sorcim, Diversified Software Research, Stone Edge, Data Transforms, and many, many others. These companies may not have a **Deluxe Paint II** available at the moment, but they will. Patience. Support them when they do. Each dollar you spend on copy-protected software now is a vote for more copy-protection in the future.*

I admit it's hard to make an informed vote when none of the publishers who use copy-protection will admit it on the outside of their package, or even, usually, in their documentation. You just have to learn which companies use protection and refuse to buy anything from them. Three major companies to avoid are Electronic Arts, Activision, and Broderbund. These three could fall off the edge of the earth and we wouldn't even notice it around here.

Integer Basic manuals

Although Integer Basic has been shied away from over the years, there were definitely some decent routines in place of the floating point routines of Applesoft. Integer Basic includes sound routines, renumbering routines, and a Mini-assembler. There

are several undocumented routines that I'd like to learn about, including the SWEET-16 pseudo-processor. However, I cannot seem to get my hands on any Integer Basic documentation. It's all Applesoft. Addison-Wesley (the manufacturer of most Apple books) does not have any Integer Basic tutorials. I was wondering if anyone out there knew of any old Integer Basic manuals I could get my hands on. The Woz left some great stuff in there.

Christopher Harrison
2516 Hillwood Place
Charlottesville, VA 22901

*The Computer Shopper ($21/yr, P.O. Box F, Titusville, FL 32781) sometimes has advertisers offering old Apple manuals for sale (look in the classifieds under the Apple section). Apple's Integer Basic manual was titled **Apple II BASIC Programming Manual.** However, it is really only a tutorial. To my knowledge no one ever wrote the equivalent of an "Integer Basic Technical Reference Manual."*

*There are a few other additional references we know of that you may be able to find in a library or borrow from a fellow user-group member. Hi-res graphics, sound, renumbering, and several other assembly language utilities for use with Integer Basic were made available by means of a ROM chip called the Programmer's Aid #1. The chip fit into an empty socket in the original Apple II and appeared in memory at $D000-$D7FF. The chip's information is included in the Integer Basic image that's on DOS 3.3 master disks. The book associated with this chip was called **Programmer's Aid #1 Installation and Operating Manual.***

*The Mini-assembler has been discussed right here in **Open-Apple** (March 1985, page 1.20; May 1987, pages 3.27-3.28). More extensive documentation is available in any of Apple's Apple II technical reference manuals.*

*Wozniak's SWEET-16 interpreter was well documented by its author in "SWEET-16: the 6502 Dream Machine" in the November 1977 issue of **Byte** magazine, pages 150-159. SWEET-16 code takes up about half the space of 6502 code, but runs about ten times slower. The S-C Macro Assembler includes the ability to assemble SWEET-16 and its manual includes a list of opcodes and some programming examples. Another article is "SWEET-16 Revisited," by Charles F. Taylor, in **Micro—the 6502/6809 Journal**, January 1982, pages 25-42.*

*There is also a set of floating point routines in the Integer Basic ROM. These were described in "Apple's Hidden Floating Point Routines" in **Kilobaud Microcomputing**, February 1980, pages 132-135; in **Peeking at Call -A.P.P.L.E. Vol. 1**; and in some materials that were once available from Call -A.P.P.L.E. called the "Wozpak."*

6502 forever

Would learning 6502 machine code be worth the time and effort? In the years to come the 6502 will probably sink deeper into the well of history. Could I expect to gain insight that would carry over to other systems by learning the 6502 now?

Brett Pull
Colorado Springs, Colo.

*Sure. One of the neat things about the 6502 is that **everything** you learn can be used with the newer chips in the same family, such as the 65816 used in the Apple IIgs. A 32-bit chip is also planned for this family, so what you learn should serve you for many years. Lots of what you learn, but not everything,*

will also apply if you later switch to a different family of microprocessors.

To learn 6502 assembly language you need to get an assembler and some good books. We published a rundown on the major assemblers in August 1986, page 2.55. Books that cover the present 6502 family are listed on page 2.56. Our favorite introduction to 6502 programming is still *Assembly Lines*, by Roger Wagner—available for a mere $2.95 from Polarware, P.O. Box 311, Geneva, IL 60134 312-232-1984. We're also interested in hearing of other reader's favorite introductory texts to assembly language.

COP that ROM desktop, MDIdeas

Does anyone know anything about MDIdeas, Inc.? Are they reliable? Is the OctoRAM board by them any good? Is it better than the IIgs RAMPlus by Applied Engineering?

In the *Apple IIgs Technical Reference* by Osborne/McGraw-Hill (a very good book) there is listed an opcode for the 65816 called COP. This stands for coprocessor latch. This sounds awfully like "go use the math coprocessor." If so, why haven't I read about it here? And why hasn't Apple said anything about it?

Wouldn't it be nice to turn on your IIgs, watch the desktop pop up, click an AppleWorks icon and be ready to go in under 5 seconds? Currently, it takes more than a few minutes. Well, you can with ROM cards! Be on the lookout! These cards could be the hottest things to hit the IIgs market.

Uncle DOS, I've seen your complaints about the Apple Desktop and the push Apple is giving that type of interface. I do believe Apple should support the AppleWorks-type File Card Interface, since its easy, portable, quick, and suits the Apple well. However, the mouse interface also works very well, and on the IIgs it should be exploited. The interface makes it easy to learn new programs, it allows for more graphically oriented work, and makes programs semi-portable from Mac to IIgs.

Sure it's not perfect and can be awkward in some programs, but the file card interface can't solve every problem either. The mouse interface also has a standard disk access protocol that makes disk usage much easier. There are programs out that also make ProDOS easier to use in text mode but they vary in quality and execution. The mouse interface allows beginners to jump right into the file storage area and not get lost by protocol or command names. For ordinary Joe Schmoes that aren't computer literate but need to use a computer, the mouse interface makes it much easier.

         Doug McClure
         Carmel, Ind.

We had never heard of MDIdeas before the IIgs came out, when they were suddenly right there with a basketful of cards for the new machine. So I called them up and asked them some questions.

The company is about two years old. It got its start selling hard disk drives for the Macintosh. One of its primary assets seems to be a strong relationship with Apple's evangelists, which put it in a good position to develop add-on cards for the IIgs.

Their OctoRam card is different from the other IIgs memory cards on the market in that is uses a memory chip called a SIM. A 256K SIM actually has 256K **bytes** of memory, unlike a normal 256K memory chip, which has only 256K **bits** (you need eight of them, in other words, to actually add 256K of memory to your machine). Their card has sockets for eight SIMS, which are available in a 256K version for

about $75 and in a 1 megabyte version for about $400. Thus, someone with deep pockets could expand this card to a full 8 megabytes. MDIdeas sells its 1 megabyte card, which uses four 256K SIMS, for $349.95. As chip prices fall, you could reasonably add more memory to this card until it was fully loaded. SIMS are also used on the Macintosh.

There's a $179.95 add-on for the OctoRam called *ESP* that puts up to 512K of battery backed-up RAM into the address space the IIgs uses as a ROMdisk. That's interesting, since the big problem I see with a ROM disk is updating the programs you put on it. MDIdeas also has a IIgs stereo card ($59.95), a IIgs sound input card ($59.95), and a IIgs fan/surge suppressor ($149.95). The company's address is 1163 Triton Drive, Foster City, CA 94404 415-573-0580.

We mentioned the COP instruction in our big August 1986 issue on the 65816 microprocessor (page 2.53, 2.55). It is indeed a command for passing control to a coprocessor. The reason you haven't heard more about it is that there isn't yet a coprocessor for it to pass control to. A similar instruction is WDM, which is reserved for accessing 32-bit instructions in the 65832 microprocessor—a member of the venerable 6502 family that is also still under development.

I agree that ROM cards are going to provide us with interesting possibilities, but if speed is your thing (as it is mine), why not use ProSel instead of the IIgs Launcher or Desktop? I have a IIe here (that we leave on all the time) that has most of the programs I use in an Apple Memory Expansion Card. It can get to the ProSel menu from an open-apple/control/reset boot in less than three seconds. From ProSel to the main AppleWorks menu also takes less than three seconds with Alan Bird's "Don't Pass Go" patches. Just as important, to get back to ProSel after quitting Apple-Works takes less than a second. This kind of speed is what I look for in a computer system.

In general, however, I agree with your comments about the mouse interface. It's great for beginners and great for graphics-oriented programs. However, it is slow compared to the file card- or magic menu-type interface—particularly when switching from one program to another. It is also highly dependent on a mechanical device, the mouse, that I have found to be unreliable. (When was the last time you pushed one of your cursor keys and the cursor didn't move? When was the last time you had to clean your cursor keys? Both situations are common mouse problems around here.)

Nonetheless, those of you who think I am against the mouse interface have misunderstood me. What I have been against is the total abandonment of the file card interface in favor of the mouse interface, something Apple's people were pushing hard for some time ago (their zeal cooled after they were unable to add a mouse interface to AppleWorks). I think both types of interface have an important place in the Apple II kingdom.

Cheap accounting

What do you know about DASH Accounting's Accounts Payable program for the Apple IIe. It advertises so much for so little.

         Terrance A. Dunbar
         Schnecksville, Pa.

We called DASH at 408-356-3616 and found out they have just finished a non-copy-protected ProDOS-based accounting package that goes for $149. The $99 product they have been advertising is DOS 3.3-

based. Another possibility is a $69.95 package ($77.45 with shipping), also not copy-protected and ProDOS-based, called *DAC Easy Accounting* (214-458-0038). We haven't seen either package, but our interest is high. We'll try to do a comparison in an upcoming issue. We'll also try to include Manzanita's **Business Works**. If any of you know of any other non-copy protected, ProDOS-based accounting packages worth looking at, please let us know.

Applesoft enhancements

For those interested in entering formulas for Applesoft translation ("Interactive formulas", March 1986, page 2.16), there is an "evaluate string toolbox" routine included with the *TERC Basic Enhancements Version 6.0* from Technical Education Research Centers, Inc. This is a nice package; it contains a bunch of ampersand-like routines that use the ' instead of the &. The routines allow labelled subroutines, formatted printing, better input, and many programming aids.

         John M. Wargo
         Akron, Ohio

The TERC package is fully described by one of its authors in the June 1987 **Call -A.P.P.L.E.**, *pages 19-21 A.P.P.L.E. Co-op members can get a shareware version of the program from the co-op for $4. The package is also available, with a printed manual, for $15 from Sean Nolan, 68 Roseland St., Sommerville MA 02143.*

80-column windows

How can I set an unmovable "frame" on the 80-column screen? POKE ⟨top⟩ and ⟨bottom⟩ still work, but ⟨left⟩ and ⟨right⟩ don't.

         John Jessmer, Jr.
         Brooklyn Park, Minn.

See "The wonderful world of windows" in our June 1985 issue, page 48, for everything we know about putting window frames on the Apple II text screen. Here are the essentials.

When using 80-column mode on the unenhanced IIe, the left and right edges of your window must be in odd columns (assuming you call the first column 1, not 0), and POKE 1403,X-1 does what HTAB X is supposed to do but doesn't. Always place the cursor inside the current window after you POKE in the edge values.

After doing a PRINT CHR$(4): "PR#3" to turn on 80-columns, do another PRINT before doing **anything** else. All the PR#3 does is redirect output to the 80-column firmware—the 80-column screen doesn't actually get turned on until something is printed. Any POKEs (or VTABs or other formatting) you do will be lost when the 80-column screen is initialized. Play with this little demo if you think PR#3 can turn on 80-columns all by itself:

```
10 PRINT CHR$(21) : REM turn off 80-columns
20 PRINT CHR$(4);"PR#3"
30 GET A$
```

If these hints and the June 1985 article don't solve the mystery (and maybe even if they do), you might want to consider incorporating one of the several commercially available windowing routines into your program. For example, Roger Wagner Publishing's **Trial Size Toolbox** (free to **Open-Apple** subscribers from Roger Wagner Publishing, P.O. Box 582, Santee, CA 92071) contains a windowing routine accessed with ampersand commands. Nite Owl Software (5734 Lamar, Mission, KS 66202, 913/362-

9898) has a $39.95 ProDOS-based set of utilities titled **Developer Disk #2** that includes a windowing utility among its many programs.

Boot stop, hi-res switches

Is there a POKE or CALL that will prevent a control/ open-apple/reset re-boot on the IIe (POKEs, CALLs, etc.)? Is there any specific call that will display hi-res page 2 without erasing it?

T. Gotanda
San Diego, Calif.

No and yes. Control/open-apple/reset is a hardware function that cannot be modified with software.

You can turn on hi-res page 2 without erasing it by flipping the softswitches that control what your Apple is displaying on the screen. For an introduction to the concept of softswitches see "The Magic of Peek and Poke" in the February 1986 **Open-Apple**, *pages 2.2-5.*

There are four pairs of video-display softswitches that have been around since the original Apple II. One pair determines whether the screen will show "page 1" or "page 2." Another determines whether the screen will show text or graphics. A third selects between hi-res and low-res; the fourth between full-screen graphics and mixed graphics with four lines of text at the bottom of the screen. Here are the locations of these switches. You throw them by POKEing them (it doesn't matter what value you poke, but zero is nice plump little number).

decimal	-decimal	hex	description
49232	-16304	$C050	display graphics
49233	-16303	$C051	display text
49234	-16302	$C052	full screen graphics
49235	-16301	$C053	mix graphics and text
49236	-16300	$C054	page 1
49237	-16299	$C055	page 2
49238	-16298	$C056	low-res
49239	-16297	$C057	hi-res

For example, the following lines in an Applesoft program will switch from a display of the text screen to a display of hi-res page 2:

```
110 POKE 49234,0 : REM full screen
120 POKE 49239,0 : REM hi-res
130 POKE 49232,0 : REM graphics
140 POKE 49237,0 : REM page 2
```

RAMdisk detective

I'm writing a program that should use a ProDOS RAMdisk. Is there any way to tell from within my program if a supplied pathname is actually a RAM disk and not a regular floppy?

Chris Strauser
Nacogdoches, Texas

You can detect RAMdisks that use the SmartPort interface as shown here in January 1987, page 2.92.

Non-Smartport RAMdisks are more difficult to detect. The volume name of most RAM disks begins with /RAM. That's probably the best way but it's not perfect. CheckMate's large ProDOS volume is named /MRAM, and there is an outside chance that some enterprising user may rename the RAM volume.

Another possibility is to show the user the names of the volumes that are online and let the user select which one will be used for program operations. The user's selection could be saved and used by default from then onward. If the volume was missing at startup, or if the user chose to reconfigure the program, you could ask the question again. This has

the advantage of allowing the user to select other types of devices than RAMdisks for program operations. It would also be a good idea to let the user specify how much of the RAMdisk your program may use.

Good BYE, /RAM bug

I have a very long ProDOS 8 SYSTEM file that I can't execute from Applesoft because it overwrites Basic.system. I have a little Applesoft program that I want to run right before this SYSTEM file to set up my RAMdisk. Any ideas?

One more point of mystery to me. One of my Applesoft programs tries to determine if the data files it needs are online by OPENing a directory (",TDIR") and READing it. My routine knows it has reached the end of the directory when it sees the "BLOCKS FREE ..." banner at the end of the directory.

This works until the program searches the /RAM volume. If I have previously run Applied Engineering's ProDrive to create a RAMdisk, everything is fine. But if there is only the small 64K ProDOS RAMdisk, the program bombs with an END OF DATA error. The small /RAM will not return the "BLOCKS FREE..." message when you READ it, though obviously it does if you use CATALOG.

Mystery? Then it got worse! I found the same thing occurs if I run the search on the AppleWorks program disk — a disk with only 2 free sectors. What's up here?

S. W. Cummings
Clearwater, Fla.

To start your large SYSTEM program you need a small assembly language program that will load and execute it using ProDOS Machine Language Interface commands. If you want to see how that's done, check out the sample program in Gary Little's **Apple's ProDOS: Advanced Features for Programmers**, *pages 107-112.*

A simpler possibility is to use the "auto-run" protocol supported by many SYSTEM programs. For example, ProSel supports this protocol. Just as Basic.system will run a program called STARTUP when it is executed, ProSel will run a program of your choice when it is executed. (When you use this feature, ProSel will install itself in the ProDOS QUIT command area, then run the program of your choice rather than going to the ProSel menu.)

The protocol works like this. The first three bytes of the SYSTEM program must contain a JMP to the program's actual starting location. The fourth and fifth bytes ($2003-4) must both contain $EE (238). The next byte contains the length of the buffer that follows, the next the length of the name of the file you want to auto-run. The bytes after that contain the actual name. For more complete information, see Little's book, page 141, or the **ProDOS Technical Reference Manual**, *pages 86-88.*

In your case, you can simply BLOAD ProSel (use PROSEL.SYSTEM for A$ in the following program), poke the name length and name of the SYSTEM program you want to run into position, and CALL ProSel. Here's how:

```
100 REM *** Cheap Launch ***
110 F$="program you want to execute"
112 A$="program that supports auto.run protocol"
120 PRINT CHR$(4);"BLOAD";A$; ",A$2000,TSYS"

130 S=8192 : REM $2000
132 IF PEEK(S)    <> 76 THEN 400
134 IF PEEK(S+3) <> 238 THEN 400
136 IF PEEK(S+4) <> 238 THEN 400
138 IF PEEK(S+5) < LEN(F$)+1 THEN 400
```

```
140 POKE S+6, LEN(F$)
150 FOR I=1 TO LEN(F$)
160 : POKE S+6+1, ASC(MID$(F$,I,1))
170 NEXT

200 PRINT CHR$(4);"CLOSE" : REM close all files
210 FOR I=48984 TO 49006 : POKE I,0 : NEXT
220 POKE 49007,1: REM clear bitmap
300 CALL S : REM launch from ProSel

400 PRINT "ERROR--THIS SYSTEM PROGRAM"
401 PRINT "  DOES NOT SUPPORT AUTO-RUN"
```

On to the problems you encountered with the ProDOS 64K /RAMdisk. The problem with reading the /RAM directory stems from a design decision. Every main ProDOS volume directory is supposed to contain four blocks of information; no more, no less (this is what limits a ProDOS main directory to a maximum of 51 filenames). But /RAM, mind you, has only a one-block main directory. It holds a maximum of 12 filenames. This gives /RAM three extra blocks in which to store files, but trips up Basic.system, which expects /RAM to follow the normal protocol.

The same problem is responsible with the Apple-Works program disk — due to the size of the files, the program disk's directory had to be reduced to squeeze all the files onto the 5.25 inch disk.

The default mapping of the first few blocks of /RAM for ProDOS appears to be:

block 0	empty block (normally boot code)
block 1	empty
block 2	volume directory
block 3	volume bitmap
block 4-7	unusable

Blocks 0-7 are marked as "used" within the /RAM volume bitmap. Even attempting to write directly to blocks 4-7 with a disk editor leaves the blocks unchanged. Therefore, blocks 0-1 and 4-7 are assumed to be unusable for data. This is logical, since areas of the Apple's alternate 64K must be reserved for other uses (alternate memory zero page and stack from $0000-$0100, 80-column text display page 1X at $0400-$07FF).

The problem could be corrected by "stretching" the /RAM volume directory back to the normal four block length. But that cure is pretty complicated. Now that you know what the problem is, you'd be better off just trapping the END OF DATA with an ONERR GOTO statement and letting your program be.

This doesn't work for some people, however, because what they actually want to know is how much free space is left on the disk, rather than what files are in the main directory. In this case, the best work around is to create an empty subdirectory (PRINT D$;"CREATE FREE.SPACE.DIR, TDIR"), READ it, then DELETE it. Since even /RAM subdirectories adhere to all ProDOS specifications, you'll be able to READ the whole thing. It's possible that the CREATE command will return a DISK FULL error, however. In that case the error itself tells you what you wanted to know — how much room is left on the disk (none).

Incidentally, **Nibble** *published an article also referring to this problem ("/RAM FIX", Nov. 1986, page 124). The program given to correct /RAM seems to have some problems that can be demonstrated by running the following test after using* **Nibble's** *program:*

```
10 REM *** Fill /RAM with dummy files ***
20 I=1
30 PRINT CHR$(4);"SAVE /RAM/FILE.NO.";I
35 PRINT "FILE.NO.";I
```

```
40 I=I+1
50 GOTO 30
```

The fix fails due to an initialization error in the repair program.

Patch marker

I have so many disks with the PRODOS file on them that it is hard to remember which ones have had your "Stop track 0 trashing" patch applied and which haven't (November 1986, pages 2.73-74). I added the following line to your patch program.

```
350 POKE 9463,208
```

It appends the letter "P" to the title "PRODOS 1.1.1" that appears during startup. This P tells me that that version has had the bugs eradicated.

Rick Highness
Vacaville, Calif.

BLOAD NOT.ALLOWED,A$400

Why can't I BSAVE/BLOAD text page 1 when using ProDOS? The command (A$400, L$3FF) prints a NO BUFFERS AVAILABLE message.

Dudley Glass
Vienna, Va.

Under DOS 3.3, programmers developed the bad habit of BSAVEing text screens and BLOADing them later. Unfortunately, this practice isn't safe because of the invisible "screen holes" that are embedded in the text page memory area. These screen holes are used by devices in slots and by the system firmware to remember such things as the last disk track accessed, the current horizontal cursor position, the serial card baud and data format settings, the active $C800 ROM bank, and much more. Although the worst effect in most cases is a "grind" of the disk drive after each new screen is BLOADed, there is potential for more serious ramifications. Thus, ProDOS protects the screenholes by marking text page 1 ($400-$7FF) as protected in its system bit map. Trying to BLOAD a file into a protected area never succeeds. It always causes Basic.system to return the totally inaccurate (in this case) NO BUFFERS AVAILABLE message.

Flicker comparisons

I've been going blind using WordStar with my Apple IIe and Apple III monitor. Can you recommend a monitor (monochrome), presumably with a long persistence phosphor, to reduce flicker and any added plug-in boards to improve resolution and reduce flicker? Why are the IBMs or Compaqs so much better in this regard?

Phillip Cutler
Costa Mesa, Calif.

That's not normal. In fact, the Apple III Monitor already has long-persistence phosphor. Mine is rock-steady and flicker-free when connected to a IIe. It shows some waviness when connected to a Laser 128, which I've blamed on that computer. A bad cable or bad connections can cause glitches, as can a cable that's too long (it becomes an antenna, picks up vibrations from the electromagnetic roar around you, and puts them on your screen).

It is also possible, I suppose, that your flicker is due to poorly designed CP/M hard- or software, but I've never heard anyone complain about that before. I suspect you need to have your system looked over by a technician.

The video circuitry on IBMs and compatibles tends to blend screen character dots together better than the Apple II's circuits do. You can update this circuitry on your Apple by adding a slot-3 80-column card (this isn't the same thing as an auxiliary slot 80-column card), such as the ones made by Videx. Try before you buy, as screen displays are largely a matter of taste.

By the way, the IBM is not immune from hideous flicker, either. You've been comparing the Apple display to IBM's Monochrome Display Adapter output. IBM's Color Graphics Adapter is notorious for severe flicker as the screen scrolls—so much so that a program is being sold by a third party company to reduce the irritation.

You might also like to take a close look at the following letter and answer. You don't happen to be using an Apple purchased outside the U.S., do you?

International answers

Further to Tony Bond's letter ("International Apple-Works II," June 1987, page 3.39) regarding date formatting in AppleWorks reports, I would like to point out that my program InfoMerge (Pinpoint Publishing), which is an AppleWorks report generator as well as a mailmerge utility, allows formatting of dates (in printed output) in six different ways, including the European date formats "30 June 1987" and "30-6-87".

And, in answer to the "Hard hardware question" about using U.S. monitors in Europe, I know only that some years ago, for the purpose of developing software in German, I acquired a German IIe from Apple Computer. This contained a U.S. power supply and I used it with a standard U.S. monochrome monitor. The only problem I had was in hi-res graphics mode, when there was a significant screen flicker, due presumably to the frequency difference between the Apple motherboard and the video monitor. For the purposes of software development I found the level of flicker to be acceptable, though it might not be so for some hi-res video games.

Peter Meyer
Captain Cook, Hawaii

Abdul Haddi in Wichita, Kansas just sent us a copy of a two-page document he got from Apple called "International Travel with Apple Computers." This sheet says that all Apple CPUs, printers, and monitors sold in the U.S. (with the single exception of the Scribe printer) are designed to work with 110-120 volts at either 50 or 60 Hertz. Thus any Apple system should work anywhere in the world if the voltage is properly converted. "You will need to obtain a step down isolation transformer. Do not use a power converter designed for hair dryers and electric razors. The transformer will need to be rated for no more than 100 to 150 per cent of the equipment's power requirements and should be obtained in the country of usage. Since all Apple equipment requires a three-prong grounded outlet, the transformer will need to provide such an outlet."

However, the "video refresh rate" on an Apple II is fixed at either 50 or 60 Hertz, depending on the standard in the country of sale. According to the document, using a 50 Hertz Apple in a 60 Hertz country will cause a flicker in the brighter area of the screen with all Apple monitors except the old Monitor III (much as Meyer describes when using a German Apple in the U.S.). Using a 60 Hertz Apple in a 50 Hertz country, "you may notice a small amount of 10 Hertz weaving or flicker on the screen."

I read all this to mean that, except for voltage, Apple's monitors and printers are the same all over the world. Thus, with the proper transformer, you can use monitors or printers purchased in 60 Hertz countries in 50 Hertz countries, and vice-versa, freely (Scribe excepted).

Secondly, except for power supplies and video refresh rates (and a few other details, see "NTSC and PAL Apples," December 1986, page 2.87), Apple IIs are the same all over the world. Thus, with the proper transformer (or by changing the power supply, as Meyer did—note that some of the power supplies listed in "Releasing the power," July 1987, page 3.45, are dual 240/120 volt units) you can use Apple IIs purchased in 60 Hertz countries in 50 Hertz countries, and vice-versa, if you are willing to put up with some screen flicker. To avoid the flicker, you must change the CPU, not the monitor.

Thirdly, we still don't know whether a 60 Hertz RGB card will work in a 50 Hertz Apple II. Does anyone else?

Another frigid reboot

In regard to "Reboot, cold reboot, frigid reboot" (July 1987, page 3.48): Another way to get a IIgs to erase and resize the built-in RAMdisk without turning the machine off is to hold down both the open-apple and option keys and press control-reset. The causes the machine to start its self test. You don't have to wait for the self-test to finish, just interrupt it with a standard control/open-apple/reset reboot.

James L. Liles
Candor, N.C.

Very interesting. The self-test must destroy the power-up byte that Bill Basham talked about.

Something old, something new

Can you have an add-on card in slot 6 on a IIgs and still use 5.25 inch drives as long as the control panel is set up for a disk port? Is there a way to load and run DOS 3.3 from the RAM disk? Is there a way to use a 3.5 inch disk with DOS 3.3?

Tom Coates
Hayward, Calif.

I have at this moment a hard disk card in slot 6 of my IIgs and a 5.25 inch disk on the end of the Smartport daisy chain. I can't use them both at the same time, but I can switch back and forth by entering the control panel, changing the slot 6 specification, and pressing open-apple/control/reset.

See "DOS 3.5" in our October 1986 issue, page 2.70, for a nearly complete rundown on how to get DOS 3.3 to work on a 3.5 drive. We can now add a few more: DOS 3.3.5 (Call -A.P.P.L.E., July 1987, pages 13-19), and Oz-DOS ($39.95 from NewTech, P. O. Box 90211, San Diego, CA 92109). New Tech also sells support software for using the PCPI Appli-card CP/M system with the UniDisk, ProFile, and Apple Memory Card.

IIgs missing CALL -144

Your readers might like to know that the Lam technique for entering Monitor commands from Applesoft programs ("Picking Up Applesoft," February 1985, page 1.12; "A charming difference," March 1985, page 1.23; "Lam a'la Ruth," May 1985, page 1.40; "Apple Assembly Line," October 1985, page 1.77; "Lam a'la Kashmarek," March 1986, page 2.16) doesn't work on the IIgs. The technique's CALL -144 crashes the system.

Cecil Fretwell
Waterloo, Iowa

On earlier Apples, CALL -144 ($FF70) jumps into the middle of MONZ, the Monitor routine that produces the "*" prompt, calls the Monitor's line-input routine, and executes the Monitor commands you type in. Jumping into the middle of it avoids the call to the line-input routine—thus Applesoft can POKE the Monitor commands of your choice into the keyboard input buffer and then call on MONZ to execute them. However, $FF70 is an "undocumented" entry point, according to Apple, anyhow—it certainly has been well documented elsewhere. The IIgs doesn't support this entry point, The Laser 128 didn't support it when it was released either, however, it does now. Let's hope Apple follows Video Technology's lead—not supporting this entry point makes a significant amount of software incompatible with the IIgs.

Forget RAMRD, RAMWRT

The letter "Odd bank out" in your April issue (page 3.22) caught my eye. It seems like Holladay is going through an awful lot just to run a program in aux mem. The best rule of thumb on the IIgs is to forget you ever heard about RAMRD and RAMWRT. You don't need them.

Holladay's program would be better off if it switched to native mode while in main memory, then did a JSL to bank $01. After you get to bank $01 you need to:

```
PHB        save the old B (data bank) register
PHK        push the K (program bank) register
PLB        set B = to the executing bank

do whatever here

PLB        restore B to what it used to be
RTL
```

Any code running in a non-zero bank must execute in native mode. This is solely due to interrupts. In native mode, the 65816 saves the program bank (K) register when an interrupt occurs. In emulation mode it does not. So when an interrupt happens in emulation mode, the bank information is lost and the only possibility is to return to bank 0. This isn't a problem for older programs that use emulation mode in aux mem, because they use RAMRD and RAMWRT. As Holladay explained, these switches turn on the "1" bit of the bank address, but the 65816 has no idea that anything has happened. As far as it's concerned it's still running in bank 0.

Corey Kosak
Novato, Calif.

IIgs cable sources

Regarding my previous letter on hooking up the IIgs to a Sony KV1311CR ("RGB and TV too," March 1987, page 3.13); Redmond Cable Corporation, 17371-A1 NE 67th Ct, Redmond, WA 98052, 206 882-2009 has them for sale at $29 list and $22 if you are a developer, etc. Theirs are shielded and include the voltage divider needed to switch the IIgs audio into the TV. (Mine isn't shielded, and I am ordering one of theirs.) They can also make them to a specified length, up to to 15 feet. The normal one is 6 feet.

Chris Arndt
San Luis Obispo, Calif.

And here's where to get the cable you need to hook Disk II drives with the old-style 20 pin header connector to the IIgs Smartport daisy chain: Applied Services, Inc., 7474 Hickory Log Circle, Columbia, MD 21045. The cable costs $20 and allows you to connect one or two Disk II drives to the Smartport chain. The cable was advertised in the April 1987 **Call -A.P.P.L.E.** *classifieds (page 58); a local user ordered one and was happy with it.*

IIgs slots, drives missing

Two things. First, in the February 1986 **Open-Apple**, on page 2.8, Jim Luther offers a PFS:Write to text file converter. The Post Office returned my letter as undeliverable. Do you have a later address for Luther?

Second, I and one other person in the Princeton Apple Users Group have noticed a problem with the IIgs. Occasionally the computer will turn off a slot that has been set in the control panel to recognize "Your Card". It happens to me for a Disk II controller card, and it happens to him with a Sider interface card. The only way either one of us has discovered to restore recognition of the card is to power the computer down for approximately 8 hours.

I realize the details are sketchy at present but maybe someone else has discovered a way to avoid this problem. At one point when this happened to me the computer mysteriously cycled through accessing the drives and erased part of my *Quicken* program disk and the data disk. Thank God for backups!

David Ciotti
Trenton, N.J.

Luther's address is now 6717 Locust, Kansas City, MO 64131.

I haven't seen this one before, but I have had my Apple 3.5 drive mysteriously "disappear" a couple of times. ProDOS insisted there was NO DEVICE CONNECTED to slot 5, even though the 5.25 disk at the **end** *of the Smartport chain was still recognized. At the time I didn't know how to do a frigid reboot, and a cold reboot wouldn't bring it back, so the only way I figured out to "recover" the drive was to turn the IIgs off.*

Missing printer found

You have given me the final bits of an AppleWorks and Apple IIgs puzzle. Now that I have the whole picture, I can give you the rest of it. The missing bit, found in your April issue (in the response to "AppleWorks phone dialer," page 3.19), was the fact that AppleWorks can't simultaneously recognize a disk and a printer in slot 2. The beginning of the puzzle was the fact that Applesoft could use my letter quality printer in slot 2, but AppleWorks could not.

The rest of the puzzle was the fact that ProDOS automatically reassigns a second 3.5 inch disk drive to slot 2, drive 1 if you have used the IIgs control panel to set up a RAMdisk in internal IIgs memory. ProDOS puts the RAMdisk in slot 5, drive 2. It leaves the primary 3.5 in slot 5, drive 1.

By going into the control panel and setting the maximum size of the RAMdisk to 0K, turning the computer off, and rebooting, I was able to get ProDOS to put the second 3.5 inch disk back in slot 5, drive 2 where it belongs (and where Pascal 1.3 can find it). And now a printer or modem located in slot 2 will work perfectly under AppleWorks.

Denis Anson
Hillsboro, Or.

One additional bit of confusion is that if you use the control panel to set up the internal RAMdisk as the startup device (look under "slots," not under "RAMdisk"), the RAMdisk will be at slot 5, drive 1. The 3.5s will be in slot 5, drive 2 and slot 2, drive 1. We have been unable to figure out a way to change these assignments. To do so you'd have the change the Smartport "unit numbers" that the devices respond to. If anybody figures out how to do this, please let us know. I'd personally rather see the IIgs RAMdisk **always** *in slot 2, drive 1. Pascal would work better,*

the RAMdisk could always be booted (you can't boot from a device in drive 2, no matter what the slot), and things would just be a lot more similar to how they are with an awslot card on a IIe. As things stand now, the IIgs internal RAMdisk is often best left at 0K.

Cards still gobbling ctrl-I

My printer's manual says the code for 6 lines per inch is "ESC Ctrl-^ Ctrl-I" but the results I get are weird.

Dr. John A. Hunt
Honolulu, Hawaii

Our trusty book of printer codes shows your printer uses "ESC Ctrl-^ N", where N equals the line feed length (plus one) in 48th's of an inch. Therefore, for 6 lines per inch, the value of N should be 9 (9 minus 1 equals 8, 8 divided by 48 gives us a line feed length of 1/6 inch), which you correctly translate as control-I. Unfortunately, that particular code will get eaten by your printer interface card every time. Try using this code instead, "Ctrl-I Ctrl-Y ESC Ctrl-^ Ctrl-I Ctrl-Y Ctrl-I". It momentarily changes the interface card command code to Ctrl-Y, then changes it back again. For more about this, see our special November 1985 printer issue, page 1.86. Page 3.14 of our March 1987 issue and all the articles it refers to will tell you much more than you ever wanted to know about the problems people encounter when trying to send control codes to printers.

No negative EXEC

Negative numbers can be READ, but not EXEC'd, into an INPUT statement. Numbers that begin with a decimal point or a plus sign work fine.

Michael Wertheim
Plantation, Fla.

Yep, we get a SYNTAX ERROR when we try it with INPUT N. Around here we always INPUT N$, however, then convert the string to a number if necessary. EXEC handles this with no problems.

AppleWorks + *Multiscribe*

I've been creating hundreds of pages of mathematical texts with AppleWorks, using all possible tricks to include the special signs mathematicians are used to. However, the main problems remain that those signs are incorporated in complex formulas, are not visible on the screen, and often don't look as they should on paper.

On the other hand, there is *Multiscribe*, offering many possibilities to solve these problems, but with a clumsy cut-and-paste system and with far too little speed for long texts.

A possible solution would be to enter text using AppleWorks, then transfer the file to *Multiscribe* for preview, final corrections, and printing. Consequently, I am trying to write a program that would read an AppleWorks word processor file, convert the AWP control codes to *Multiscribe* codes, and create a *Multiscribe* file. I'd also like to write a *Multiscribe*-AppleWorks conversion program to go back to the AppleWorks environment where file-merge is an easy operation.

The whole process is not so complicated as it sounds except for one problem that I haven't solved yet. Using the "view files" option of *Copy II Plus* we can see a header on each AWP file with some code and the position of the tabs. What is the total length of this header and what is the meaning of the various bytes?

Bert Hoogewijs
Lochristi, Belgium

AppleWorks word processor files start with a 300 byte header that contains the following information:

```
+000 - +003   Not used.
+004          Not mentioned in documentation.
+005 - +084   Tab stops.  Either = or |.
+085          Zoom switch.  Boolean.
+086 - +089   Four bytes not used.
+090          Boolean, whether page break lines
              are currently showing.
+091          Minimum left margin that should be
              added to the margin appearing on the
              screen. This is normally one inch,
              shown in 10ths of an inch, or 10 ($0A).
+092          Boolean, whether file contains any
              mail-merge commands.
+093 - +249   Not used. More or less reserved
              for mail merge feature.
+250 - +299   Available. Will Never be used by
              AppleWorks. If you are creating
              AW WP files, you can use this area
              to keep information that is important
              to your program.
```

Values do seem to change for some of the bytes marked "unused." We assume these are artifacts. For the Boolean fields, the switch is considered "off" if the byte holds a zero and "on" if non-zero.

After the header come a number of variable-length line records. The first line after the header corresponds to screen line 1, the next to screen line 2, and so on.

The first two bytes of each line indicate how long the rest of the line is. The third byte indicates the screen column for the first character of the line. Usually this is zero, but can vary as a result of left margin, centering, and indent commands. If the high bit of the fourth byte is set, the line ends with a carriage return. The other seven bits give the number of bytes of text following this byte. This text can include control codes that indicate special AppleWorks formatting features such as underline, boldface, and superscripts.

In addition to "text" lines, you will also see two-byte "carriage return" and "command" lines. A "carriage return" line has a $D0 in the second byte of the line. The first byte indicates the horizontal screen position of the carriage return. A "command" line has a value greater than $D0 in the second byte, and various meaningful values in the first byte.

"------Chars per Inch: 12 chars" is a sample of how a command line appears on the screen. A line with a value less than $D0 in the second byte (only a zero is actually possible) is a text line.

We can send a complete description of AppleWorks file formats to subscribers who request them. Or you can download the formats from a BBS that AppleWorks author Rupert Lissner has set up. The BBS contains technical information for developers working on accessories that run while AppleWorks is active. The number of the BBS is 702-831-1722. Dennis recently signed on and downloaded this directory to the information available:

```
File 1. 8K. Data Base common
    Definitions used in all data base segments,
    including the location of files and work areas
    in memory.

File 2, 33K, File formats
    AppleWorks file formats on disk.

File 3, 20K, Main work area
    Definitions used throughout AppleWorks. Defines
    common work areas used by all applications.
    Defines many of the symbols referenced in other
    files.
```

```
File 4, 3K, Hooks documentation
    Defines how to use "hooks" built into version 2.0
    that allow you to intercept keystrokes and output
    to the screen.

File 5, 6K, Host documentation
    Documentation of routines in the permanently
    resident portion of AppleWorks. You can use
    these routines, at your own risk, of course,
    for a number of the functions that are basic to
    any assembly language program. File 6 defines
    where the routines are located.

File 6, 2K, Host entry points
    This defines addresses that can be JSR'd to for
    permanent routines. File 5 describes the
    what the routines do.

File 7, 5K, Sub-host documentation
    This describes more routines that are permanently
    resident. Most pertain to memory management.
    Please note that a patent is pending on these
    memory management routines, so you may end up
    paying royalties if they're copied.

File 8, 1K, Sub-host entry points
    This file defines the entry points for sub-host
    routines.

File 9, 16K Spreadsheet Common
    Definitions used in all spreadsheet segments,
    including location of files and work areas in
    memory.

File 10, 10K, Word processor Common.
    Definitions used in all word processor segments,
    including location of files and work areas in
    memory.

Files 11, 12, and 13, 41K, SEG.PR source code

File 14, 2K, Memory manager
    Use of IIgs memory manager by version 2.0.

File 15, 33K, File formats
    AppleWorks file formats on disk.

File 16, 2K, Bugs file
    Bugs in 2.0, and conflicts between 2.0 and other
    programs being developed for the IIgs.
```

Another use for AppleWorks

Are you aware that Appleworks comes with a utility program that allows you to read and print the data in damaged Appleworks database files? You can even use this utility to produce an ASCII text file, which can be used in turn to create a new database file. The name of this utility? It's the Word Processor!

Here's how it works. Select "Add files to the desktop" and "Make a new file for the word processor". When the program prompts you for the source of the new file, choose "From a text (ASCII) file." Unlike most other programs, the Appleworks word processor will not check to see if the file you name is actually an ASCII text file and will load just about anything you tell it to, including AppleWorks data base files.

When you load an AppleWorks data base file you'll see all the data that's in it along with a lot of other stuff. In fact, the screen may look so confusing that you'll decide to re-enter the data from the keyboard. That's fine, but at least you can now produce a hard copy of the data to work from.

If you decide to reformat the data and create a new ASCII file to load into the data base, the task will go faster with a word processor that can search and replace with control characters, such as *FreeWriter* or *Apple Writer*. If you choose this option, it may take some time to format the data correctly, but you are rewarded by not having to retype or proofread anything.

The technique also works with files from other programs. DOS 3.3 database files require conversion

to ProDOS first, but otherwise all the operations are basically the same. If your database file contains mostly text, you'll find the task a relatively simple one. Mostly numeric data may require a hard copy as an aid in locating the numerals amidst all the the other stuff.

Steve Goldstine
NEO Apple Corps
Cleveland, Ohio

Now there's a neat trick. Interestingly, **Apple Writer** *will also directly load any type of file, but it stops loading as soon as it encounters a zero byte. In AppleWorks data base files, this occurs long before you get to the part of the file that holds data.*

In addition to fixing damaged data base files, this technique could be very useful for converting random-access text files from older programs into AppleWorks data base files. If the random-access file is on a DOS 3.3 disk, use **Copy II Plus** *to convert it to a ProDOS disk first.*

Our March 1987 article, "Reading AppleWorks data bases" would be helpful for translating what appears on the word processor screen into what must have been in the data base file. It would probably also be helpful to take a file back to the AppleWorks word processor for final formatting. The easiest way to figure out if you have the correct number of return characters in each record is to simply load the file into the data base and look to see if everything is appearing in the correct category. At the spot where the relationship between data and categories gets messed up you are either missing a return or you have an extra one. Q(uick change) back to the word processor, fix that spot, and try again.

Subscriber James Green in Woodland Hills, California also wrote us about this trick, but he uses it to recover damaged word processor files.

More uses for our ADB reader

Thank you very much for your reply to my recent request for help in recovering a damaged AppleWorks database. Unfortunately, after spending hours studying the file, I could not find anything wrong with it according to the published file layout. It still would not load into AppleWorks.

At that point I took another tack which your readers might find helpful. Using your AppleWorks Database Reader program (March 1987, pages 3.9-3.12) as a starting point, I wrote a program that read my file and wrote it into a text file, which I then used to create a new AppleWorks database. Through this method I was able to salvage all three years of the "lost" magazine index with only about 30 minutes of work, versus many, many hours of re-entering the data.

R. W. Crowley
Manchester Mo.

Here's another application for our ADB.READER. Dennis recently saw an ad for an MS-DOS program that "compiles" Lotus 1-2-3 data files into free-standing applications that do not require Lotus itself to use. In other words, a "developer" can use Lotus to enter and manipulate specialized data, "compile" the data into a free-standing application, then sell the data to customers who don't have Lotus. Our ADB.READER is only a step or two away from the same thing for AppleWorks. A database format could be laid out with AppleWorks, the file data entered and edited into usable form, then the final database distributed with an enhanced version of our "reader" program. Thus a "simple" Applesoft program could turn AppleWorks into a database development system of sorts.

A new horror story

Please warn users about Applied Engineering's HD-55A disk drives. When using these drives to make copies, one or more sectors on the source disk are usually trashed and can not be verified. Putting on a write tab does no good. With COPYA you may seem to have a good copy but, if you try to repeat the copy from the same source disk, you will get read errors.

John Waters
Baltimore, Md.

The word from Applied Engineering is that one group of their drives had this problem and they issued a recall. The drives were bad right out of the box. It may seem a little paranoid, but perhaps the lesson here is that all new equipment should be carefully checked with dummy files and data before using it with the real thing.

Spreadsheet $tring variables

I would like to know whether there is any spreadsheet program that lets one use the IF statement with text. For example, I would like to use a statement such as IF(A5="Sam",0,1). Neither AppleWorks nor Multiplan allow text as a clause or argument in formulas; is there any other spreadsheet program that has this facility?

Dr S.S. Datye
Akureyri, Iceland

How do you get AppleWorks to report an alphabetic response in a lookup table?

Carl J. Morrison
Anaheim, Calif.

SuperCalc 3a solves both problems. It allows text to be assigned as a value by enclosing the text in parentheses and quotes, as in ("This is a text value"). For example, if you set cell A5 to ("Fred"), and cell A10 to IF(A5=("Sam"),1,A5), then cell A10 will display the word "Fred".

SELECT stops expandedWorks

Regarding Michael Colville's "hanging on startup" problem with AppleWorks 2.0, as expanded with Applied Engineering's software (July 1987, page 3.44): quite by accident (after much cursing and swearing, which had no effect at all) I discovered that this problem occurs only when I boot or restart AppleWorks with my ImageWriter on and the printer "select" switch off.

Either turning on the select switch or turning off the printer will allow the boot process to continue after it has halted on the "Getting Started" screen.

Steve Moore
Carpinteria, Calif.

Thanks also go to Robert Urschel and Quentin Packard, who made the same discovery. An unexpanded copy of AppleWorks 2.0 doesn't hang on my IIgs in this situation, so I suspect AE's expansion software is trying to send some kind of initialization string to the printer.

You could discover more by turning your printer on in "hex dump" mode and seeing what gets printed when you startup AppleWorks. With the ImageWriter I, you do this by turning the printer on while holding down on the "form feed" button; with the ImageWriter II, use the "select" button.

Pushing back the limits

I may be able to shed some light on the AppleWorks "130 file maximum" and "memory card hogging" problems mentioned in your July issue (page 3.45).

First the subdirectory limitation…my guess is that the number of files that appears in the screen message (130) was originally set by approximation and has never been updated. I checked all of the versions I have access to and found that 1.0, 1.1, 1.2, and 1.3 all allocate $1000 bytes of buffer space for filenames from a subdirectory. Version 2.0, however, allocates only $800 bytes. The buffer is used to sort the names into alphabetical order. Each name entry (as used here) has $18 bytes. This means that the older versions of AppleWorks will actually show 170 names and the newer ones will show 85. Interestingly, what triggers the screen message is a buffer overflow …try cataloging a subdirectory with 169 names in version 1.3. It will show all of them without the "130 file maximum" message.

As for the memory card hogging problem, after studying the "SYS.DESKTOP" expansion function that AppleWorks 1.3 and newer uses with the Apple Memory Expansion Card and compatibles, I came up with this little patch program. In theory you only need to execute it once; however, in reality you will probably run it several times until you figure out your personal "ideal" desktop size. It will patch AppleWorks versions 1.3 and 2.0 (with minimum verification). To keep the program simple, the user gets to set the prefix to the volume (or subdirectory) where APLWORKS.SYSTEM

lives before running the program — the user also gets to handle any errors that come along.

The program will allow you to set your desktop from 56K up to about 1020K. If you make it too big, you are defeating the purpose (leaving room on the RAMdisk for other files). If you make it too small, AppleWorks will act as if the card already has too many files and it won't expand into the memory card — this will cause you to miss out on the other goodies that come with expansion.

```
10   REM  Desktop Sizer for Slinky-type cards
20   REM  by Steve Stephenson
30   :
40   D$=CHR$(4) : PRINT  D$;"PR#3" : HOME
50   PRINT "What K size desktop do you want? ";
55   INPUT "";K$
60   K=VAL(K$) : K=K*2 + 16
65   KHI=INT(K/256) : KLO=K - KHI*256
70   PRINT D$;"BLOAD APLWORKS.SYSTEM,A$2000,T$FF"
80   IF PEEK(8192+3)=20 THEN 200 : REM skip to v2.0

100  REM fix v1.3
110  FIX=9488+4096 : WEDGE=9282+4096
120  FOR I = 0 TO 25
130  READ J : POKE FIX+I,J : NEXT
140  DATA 170,173,223,34,201,0,173,224,34,233,0,176,
         2,138,96,169,0,141,223,34,169,0,141,224,34,96
150  GOTO 300: REM goto common fix

200  REM fix v2.0
210  FIX=9429+4096 : WEDGE=9662+4096
220  FOR I=0 TO 25 : READ J : NEXT : REM other data
230  FOR I=0 TO 25
240  READ J : POKE FIX+I,J : NEXT
250  DATA 170,173,62,35,201,0,173,63,35,233,0,176,
         2,138,96,169,0,141,62,35,169,0,141,63,35,96

300  REM common fix routine
310  POKE FIX+5,KLO : POKE FIX+10,KHI
320  POKE FIX+16,KLO : POKE FIX+21,KHI
330  POKE WEDGE,32
340  POKE WEDGE+2, INT ((FIX-4096)/256)
350  POKE WEDGE+1,(FIX-4096) - (PEEK(WEDGE+2)*256)
360  PRINT D$;"UNLOCK APLWORKS.SYSTEM"
370  PRINT D$;"BSAVE APLWORKS.SYSTEM,A$2000,T$FF"
380  PRINT : PRINT "Done." : END
```

Steve Stephenson
Synesis Systems
P. O. Box 1308
Gilbert, AZ 85234

My personal AppleWorks disk doesn't have the 85 or so limit on AppleWorks files in a subdirectory. Maybe it's because of this patch:

```
POKE 768,48
POKE 769,29
BSAVE SEG.M1,T0,L2,A$300,B$1428A
```

The patch increases the size of the AppleWorks 2.0 catalog buffer so that it will hold 130 filenames. The area beyond the catalog buffer is unused when file handling activities are going on.

Randy Brandt
JEM Software
P.O. Box 20920
El Cajon, CA 92021

Stephenson's patch is for II-Plus, IIe, and IIc AppleWorks users with an Apple Memory Expansion Card or the equivalent (RamFactor, plusRAM). The patch has no effect on AppleWorks if you use a IIgs because on that machine AppleWorks refuses to look for a standard-slot-based memory card. (Incidentally, for those of you who don't recognize their names, Stephenson is the author of Checkmate Technology's AppleWorks expansion software and Brandt is the author of Beagle Bros' MacroWorks and Super MacroWorks.)