Don Lancaster's

ASK THE GURU

Selected reprints

Electronically self-published using the Apple IIe computer and the Laserwriter Plus. All graphics were done in their entirety by ProDOS Applewriter 2.1.

Introduction

In early 1985, Stan Veit of the *Computer Shopper* magazine asked me to put together a series of columns for him. While the *Ask the Guru* columns were all purportedly supposed to have at least something to do with Apple computing, Stan assured me he would use a "hands off" editorial policy with minimum editing, combined with the fastest possible turnaround time.

Apparently I would be given the opportunity to say what I felt like whenever I felt like it, and have it rapidly and reliably appear in print. To tightly close the reader feedback loop, I would maintain a "free" user helpline, to be followed up later with some reader contests and a no-charge BBS system.

What you have here is a compilation of quite a few of my columns, re-edited, updated, and corrected. All of the individual *Names and Numbers* have also been gathered together in one place and newly revised for you.

So, what are these columns about? If you ever do find out, please be sure and let me know. I've always gone along on the assumption that, if I am interested in something, or if it catches my attention, then there's others of you out there that also might find that something of value. And that's regardless of whether you are an *Apple* person or not.

As on ongoing experiment, these reprints are self-published using the *Applewriter* word processor, the great *PostScript* page description language and are printed on the *Laserwriter Plus*.

Yes, *all* of the text and *all* of the graphics you see was done *solely* with *Applewriter*. Yes, the economics of this are quite outstanding. No, there was no pastup or camera work involved. Zero. And, no, this is not a copy you are holding. It is an original that has been custom printed just for you.

If you like what you see here, you can get further involved by subscribing to *Computer Shopper* magazine at (305) 269-3211, calling me on my voice helpline at (602) 428-4073, during any weekday *8-5 mountain standard time,* or checking into our great PostScript BBS at (409) 244-4704.

About the Author

As he said in his classic *Incredible Secret Money Machine*, Don Lancaster writes books. And quests *tinajas*.

Microcomputer pioneer and guru Don Lancaster is the author of 21 books and countless articles. He is considered by some to be the father of the personal computer, for his early ground-breaking work with hacker digital electronics and low cost video terminal displays.

Some of his other titles include his *CMOS* and million-seller *TTL Cookbooks*, *Micro Cookbooks* volumes *I* and *II*, *Enhancing your Apple II*, volumes *I* and *II*, the *Applewriter Cookbook*, *All About Applewriter*, the *Active Filter Cookbook*, *Apple Assembly Cookbook*, and *Don Lancaster's PostScript Secrets*, along with his *Introduction to PostScript* videotape.

Don's current software offerings include both his *PostScript Show and Tell*, and his *PostScript Technical Illustrations*, plus numerous companion disks for his various books.

Don is the head honcho of *Synergetics*, a new-age design and consulting firm specializing in Apple computing, laser printing, electronic prototyping, desktop publishing, technical writing, and in innovative software development. His avocations include firefighting, cave exploration, bicycling, and, of course, *tinaja* questing.

Table of Contents

Table of Contents, continued . . .

Don Lancaster's

ASK THE GURU

March, 1985

Cassette tape reliability
ProDOS Applewriter 2.0
Free technical magazines
Programming an EPROM
Random number generator

Welcome to a brand new *Computer Shopper* feature column. I will do what I can here to find answers to reader hardware, software, or supply problems. Just give me a call or write per the address in the end box. Best calling times are 8-5 on weekdays, mountain standard time. Other times are catch as catch can. I will send a product and freebie list to you when you respond.

While I will try to fairly cover all of the bases, don't expect any unbiased miracles from someone who is overwhelmingly pro Apple. I'll call things as I see them.

We will gather all of the names and numbers together in a box at the end. I am very much into ferreting out obscure and unique sources. Thus, some of these sources might be *very* hard to pin down elsewhere. Be sure to save these listings.

Here goes Alpha 1.0 . . .

Show me a Simple Random Number Generator.

First, note that there are two types of "random" numbers. Truly *random* numbers can assume any one of many equally likely values. Instead, *pseudorandom* numbers are the next number available in a contrived series that appears on the surface to be any one of many equally likely values.

There are computer uses for both types.

Real "random" numbers are truly unpredictable, but they are hard or inconvenient to generate and there is no way to get the same random sequence back over again at a later time. On the other hand, pseudorandom numbers are easy to generate, and you can easily get the same short sequence back again.

Repeating any random sequence is handy for replaying a hand of cards, reviewing a chess move, or getting "noise that repeats" for industrial testing.

The usual way to get a random number is to ask for a user response and then rapidly count until they hit a key. That flea-bitten "Hi – What's your name?" prompt is often one sneaky way of picking up a genuine random number.

The Apple does this automatically, with two page zero machine language locations called RNDL and RNDH in $4E and $4F. Just read these and you have a truly random number. Similar code is easy to add to virtually any microcomputer.

Unfortunately, this will get you only *one* random number, and your user may get tired of typing his name in every time you need a new number. One random number can be most useful, though, as a *seed* to generate a sequence of pseudorandom numbers.

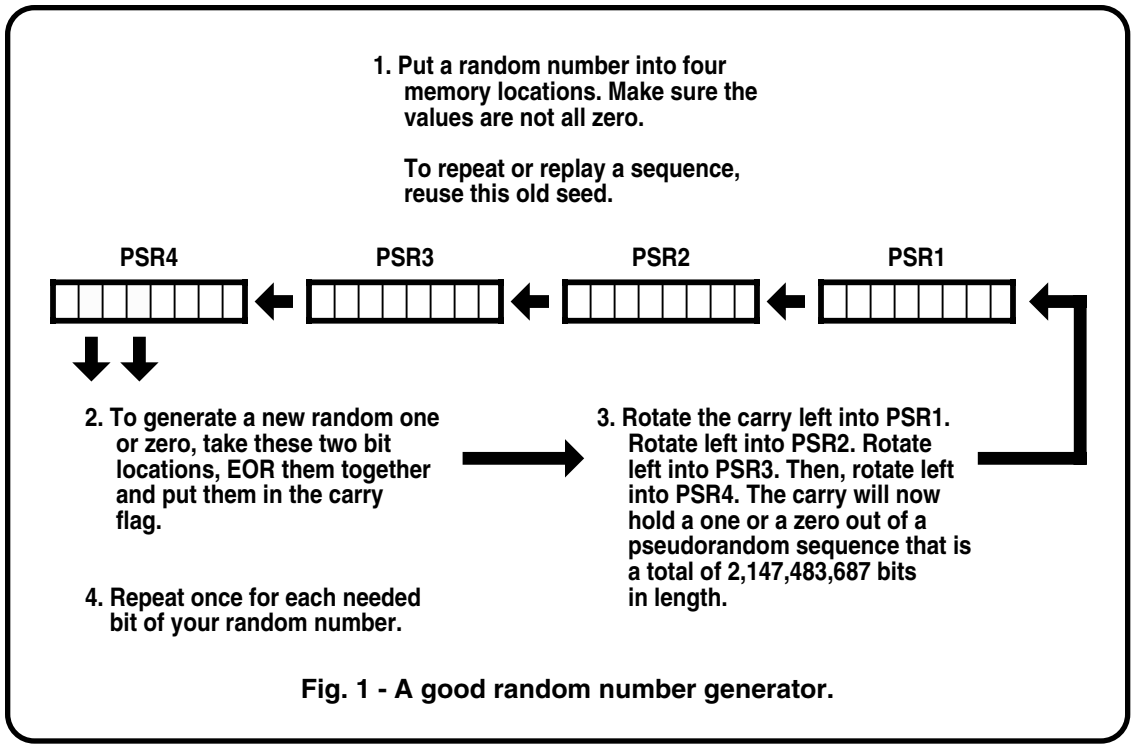


Fig. 1 - A good random number generator.

Generating pseudorandom numbers is not trivial. Applesloth tried it and they failed miserably. As is well known, there is a fatal flaw in the Applesloth random number generator. Some of its contrived sequences can almost be acceptably long. Others are as short as 191 values before repeating. This leads to distinct "unrandomness" in many BASIC game programs. And, no, the published reseeding fixes don't help much – all they do is bury the problem and make it slightly more subtle.

Figure one shows a good pseudorandom number generator that has no bad habits and easily passes the less exotic randomness tests. It is a model of a circular pipe with 2,147,483,687 marbles in it. The pipe has a one marble wide glass window in it. Half of the marbles are red and half are blue. The marbles sit in the pipe in an apparently haphazard order. By picking a random starting place in the pipe, you can get a random bit stream out. Take six bits for a random number up to 64. For a deck of cards, take out six bits, and only keep those whose values are in the range of 0-51, and so on.

To get the same sequence back, just start at the same entry point.

I have shown you only the essentials in figure one, done so you can easily translate it to any machine language code of your choice. You could also use a higher level language if you are willing to use and accept pitifully slow execution speeds.

Four gotchas: You must never allow your pseudorandom word to contain all zeros or the randomizer will hang. You should "seed" the word with a truly random number for a new starting point and a new sequence. If you want an old sequence back again for a replay or "noise that repeats", just save the old seed somewhare and reuse it.

Thirdly, do not try using more or fewer words unless you know exactly what you are doing, for any very long useful sequences are few and far between.

There's a fourth possible gotcha: The individual bits you will get on a single shift easily pass all sophisticated randomness tests. Believe it or not, though, certain combinations of sequential bits grouped into words can sometimes fail certain very subtle and very exotic tests.

Chances are that you are interested in the real world. There is a quick, simple, and devastatingly effective randomness test that Applesloth and several other so called "random" functions fail miserably. This pseudorandom sequencer passes the same test with flying colors.

All you do is plot the dots on a graphics screen, such as Apple's HIRES page one. If the screen "sticks" (as it will with Applesloth), you have a short and useless random sequence. If there is ever any shading, lines, unexpected patterns, etc., then you have blatently preferential numbers. This test applies your eyeball as an optical corelator that will pull subtle randomness failures right out of the woodwork.

Which should tell us that using groups of bits as a random word will be useful and effective for most real world needs. Note that this method is very fast and very simple to do.

Beware of playing games to make things "more" random. Believe it or not, just about any attempt to make something "more" random will almost certainly do the exact opposite.

If you really want to get into some gory details of *why* this generator works, check into Knuth's *Seminumerical Algorithms*, on pages 1-160 (Addison Wesley). Or, if you want to go completely off the deep end, do check into E.J. Watson's listing in *Mathematical Computing* 16 for (1962), 368-369.

Many more details on all of this appear in my *Assembly Cookbook for the Apple* II/IIe (SAMS) #22331, including full 6502 source code for EDASM and the ready-to-run object code, along with links for BASIC, the screen plot test, a card shuffling demo, user examples, etc. It's also available on disk.

As another and seperate use, you can route this generator to a speaker to generate a hissing white noise that really sounds awful.

Show us the sneakiest use you can come up with for these unique pseudorandom sequences.

What is The Word On ProDOS Applewriter 2.0?

There is a brand new upgrade of *Applewriter* out. It runs under the ProDOS operating system, and has lots of new features. These do include faster operation, much improved compatibility with other ProDOS programs, easy hard disk access, a built-in transmit and receive modem, settable screen margins for "what you see is what you get", spreadsheet editing up to 240 columns, an optional page and position display, and many other improvements and performance upgrades.

Best of all, the new version is unlocked, unprotected, and freely copyable for any number of backup copies. Even the source code is capturable.

What is wrong with it?

Very little. A few parallel printer cards will not work properly on first try, notably the *Grappler* and the *Pkaso*. You can write me for a free patch, or else make use of the 2.1 upgrade that corrects this defect.

The NULL patch for Epson superscript use is no longer needed, since a [_] can be substituted. The "shortline" problem remains, but you can write me for a seperate free patch on this.

Many people still grossly,

utterly, and completely underestimate Applewriter. It is the ONLY word processor I use, and once you really get into it, there is virtually nothing it cannot do in a fast and very friendly manner. Its greatest abilities lie in its being totally programmable through a companion programming language called WPL. Thus, word processing tasks can very easily be customized or else automated "hands off" in any way you like.

I've been able to personally add a full microjustify and proportional space, author's keyword indexing, multiple columns, HIRES dumpings, unique self-prompting glossaries, code extensions, and many other goodies to it.

Cost of the lastest ProDOS 2.0 version is $150 from your local Apple dealer. There is a $50 upgrade service if you send in any older Applewriter first factory disk and a manual cover to Apple's *Applewriter Product Manager*. Uh, apparently, this upgrade even applies to old mangy Applewriter 1.0 or 1.1. Thus, the upgrade is a fantastic bargain.

Where can I Get Free Tech Magazines?

It never ceases to amaze me how many people have never heard of the *controlled circulation* magazines. These are publications which are totally and absolutey free to a select group of insiders who allow the magazine to qualify for a special postage rate.

And, they are usually chock full of the latest and most needed information you are likely to see anywhere. Better still, most of them have bingo cards for immediate advertiser and product info requests.

I would guess there's over 20,000 titles of controlled circulation magazines available, and most of them are absolutely free.

Let's start with several electronic examples. *Electronic News*, *EE Times*, *Electronic Design*, *EDN*, and *Electronic Products* are all first rate technical electronics maga- zines. And there are dozens more.

A pair of mechanical trade journals essential for robotics include *Machine Design* and *Design News*. Similarly, there are dozens of "old line" minicomputer journals, and lots of magazines for the computer retail trade.

Let me know if you want a more complete listing.

You "qualify" for any and all of these by requesting a qualification card and telling them exactly what they want to hear on the card.

Sometimes you can find qualification cards ready to use. If not, a business letterhead is a must. Note that custom letterheads are free these days if you have any access at all to a laser printer.

To date, nobody has ever been tarred and feathered for willfully and maliciously saying that which is not so on a qualification card.

Creativity counts.

I personally subscribe to hundreds of these in wildly different fields. How do you find out what's available? Go to any library and they will have a *Bowker* book called *Uhrlichts Periodicals Dictionary*. Which does list all of the magazines in the US or in the world, depending on the volume you pick. Those marked "controlled circulation" are most often free, and are certainly worth checking into.

Or, in a larger technical library, ask to see their *public serials list*. This is a pile of computer printout, a video disk, or a stack of microfilche that tells you everything the library has in stock. They may call this listing by a different name. Get recent issues of target magazines, liberate the qualification card, and you are home free.

Incidentally, there is one sure fire way to get at least one free copy of *any* magazine, no matter how high the subscription price or issue cost. Just write the advertising department and request a sample issue and a rate card. On your letterhead, of course. Sometimes you will get several issues or even a free subscription out of one single request.

How can I improve cassette tape reliability?

Cassette tapes are still used on many smaller microcomputers, particularly orphans like the *Timex/Sinclair* or the *TI-99*. The cassette routines on Apple computers, while seldom used, remain the "save of last resort", either to try and recover from a blown DOS operating system, or to save any part of any program at any time for any reason.

(Continued on page 2.5)

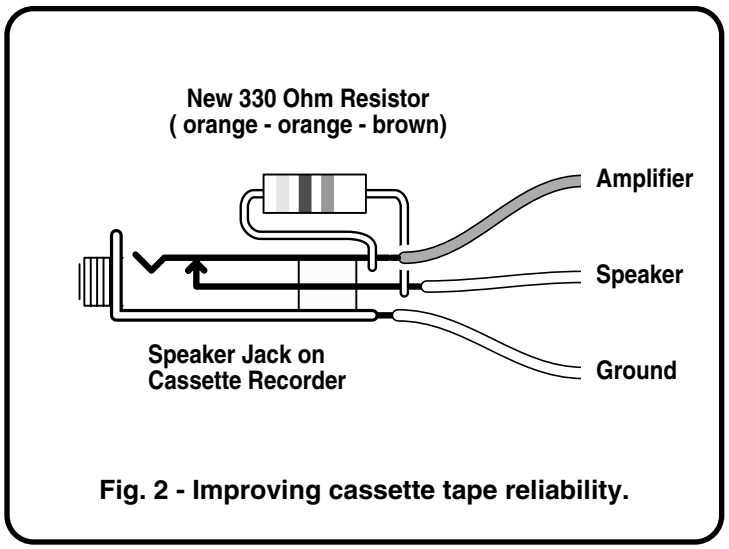


Fig. 2 - Improving cassette tape reliability.

Don Lancaster's

ASK THE GURU

April, 1985

Low cost air valves
Pneumatic actuators
Secret Apple manuals
Your own tech venture
Applewriter stretchifier

Welcome once again to the second release of our brand new *Computer Shopper* column. I'll do what I can to find answers to reader hardware, software, or supply problems.

Just call or write per that address in the end box. Best calling times are 8-5 weekdays, mountain standard time. I'll send a free booklist and a special goodies list to you when you respond. And even more neat stuff if you are an Apple hacker.

As before, we will gather all of the names and numbers together in an appendix at the end. We'll try to keep them all current. Some of these sources are *very* hard to pin down elsewhere, so be sure to save these listings.

Here goes . . .

Any Ideas Involving Low Cost Robotics?

It never ceases to amaze me that low-pressure pneumatics has never taken off. Yet, air pressure systems in the 3-to-6 psi range have enormous advantages.

First off, low pressure air is cheap and low tech. It gives you lots of linear force easy and simply. It can *amplify*, since most of the muscle comes from the air source.

This means you do not directly have to provide high power electronic drivers as you do with solenoids or servo motors. Air also goes around corners beautifully, particularly robotic elbows.

Most important to us, low pressure air has fantastic hacker potential. You can literally beat one of these systems out on a brick in your back yard, and still come up with a promising and useful product.

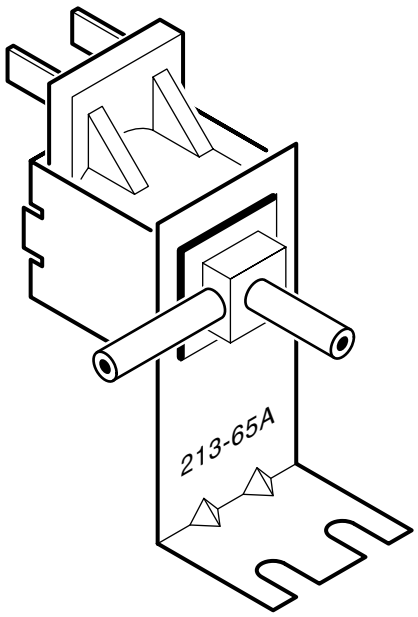


Fig. 1 - A 30 cent, robotic 3-way air valve.

You'll need an air source. The larger aquarium pumps are ideal, when combined with a small storage tank. Small tanks are easily built from plastic toilet tank floats. You can instead use a larger tank filled with a tire pump, or even use a scuba tank or a *Scott Airpack* bottle.

Hoses and connectors are no big deal. Get your hoses from *Hygenic* and the connectors from *Value Plastics*.

A regulator is also needed. I've had the best luck running at 3.5 psi, although you get much more force up in the 5-7 psi range. You can get regulators for around seven bucks from most of the surplus houses mentioned below.

Now for the fun part. You need a control valve. Unlike electricity or electronics, you cannot simply connect your air supply to an actuator to move it and then disconnect it to return. Try this and the air stays stuck in the pipe and the actuator will remain extended. Instead, you need *three-way* air valves that act like electronic SPDT switches. When powered, the air supply gets connected to the actuator. When the control signal is turned off, the pressure remaining in the actuator vents to ambient through the second valve arm.

You simply will not believe the price of an new electronic three way air valve for low pressure robotics. How does thirty cents sound?

Thirty cents!

They are called automotive EGR valves, and originally were made by the *Carter Carburetor* people. For many years now, they, have been

available surplus from *C&H Sales, Jerryco, Herbach and Rademan, BNF Sales, Surplus Center,* and all of your usual mechanical surplus outfits.

Your nearest handy dandy junk yard is also crammed full of them.

Incredibly, these have gone begging.

You'll find that *Jerryco* has far and away the lowest valve pricing, the most outrageous catalog, and the most mind blowing array of other goodies. Where else under one roof can you get a genuine joystick for a B-17 bomber, lumber grading crayons, a matched set of 24 official US Army urine speciman bottles, candle molds, or a dummy howitzer shell? Be sure to check them out.

Figure 1 shows you a typical valve. You power it with +12 volts DC to connect the air source to the load.

Removing power vents the load to ambient through a filtered vent at the rear of the valve coil. The side pipe goes to the air source. You can make a simple manifold for several valves by taking a piece of thick, large diameter tubing and punching one pinhole in it for each valve.

Then, push the front arm of each valve into the pinhole. A push-on pressure seal is all you need at 3.5 psi. Medium power electronic drivers, such as a *Sprague* ULN-2813, easily power the valve under direct computer control.

What do you use for actuators? Traditional miniature air cylinders have been available through *Clippard*. Unfortunately, they are also available at traditional prices.

Instead, let us get non-traditional. There is a very little known key secret to any useful low power, hacker based pneumatic actuator: *Never* have a seal that must move! All of your low power air will either be wasted trying to move the seal or else will leak right by it.

Figure 2 shows us three different ways to build actuators that do not have any moving seals. That rolling diaphram is available as a stock *Bellofram* part.

I like the bellows actuators myself. I actually had a bunch of these blow molded long ago, for use in a pneumatic attachment that let you use a *Selectric* typewriter as a computer printer. It worked, more or less, but never became a viable product. Polypropylene is probably the best choice of material, although EVA or urethane might also be considered. The scientific supply houses sell bellows pipettes that are almost useful as bellows actuators.

I've even ripped apart those "pump the water" toys that were available a few years back. The bellows pump inside these worked just fine.

Chances are that you will prefer the bladder or balloon actuators instead, since these are the cheapest and are the mechanically simpliest.

By the way, if you must have a seal anywhere at all, use an O-ring. These are far and away the best route, besides being reasonably cheap and standard.

Let me know what you can come up with here. There is a fantastic opportunity going begging. And you can be in on the ground floor. Your feedback and networking is most important to other readers and this column.

How can I Eliminate Applewriter "Shortlines"?

All versions of Applewriter do have a nasty bug in their printer routines.

If you try to imbed printer commands in your text, the commands will get imbedded and will get used correctly. The trouble is that they will also get counted as "real" characters, and will shorten any printed lines they happen to appear in.

Figure 3 shows you a fix for both DOS 3.3e versions of Applewriter. You can run this patch as an Applesloth program and then you insert your third or higher backup copy when asked to. The shortline patch is then permanantly installed on your third or higher backup copy.

Do NOT make this patch to either factory diskette! Use only your third or higher copy. One way to backup AWIIe is with *Copy II+* and a parameter change of 10:96.

After the patch is made, any embedded sequence that consists of an escape that is followed by a single letter gets perfectly repaired. If you want to fix shortlines on any longer commands or repair non-escape imbeddings, just use an [esc][esc] to *bank* two characters, or else use an [esc][esc][null] to bank one character. Such banking is rarely needed.

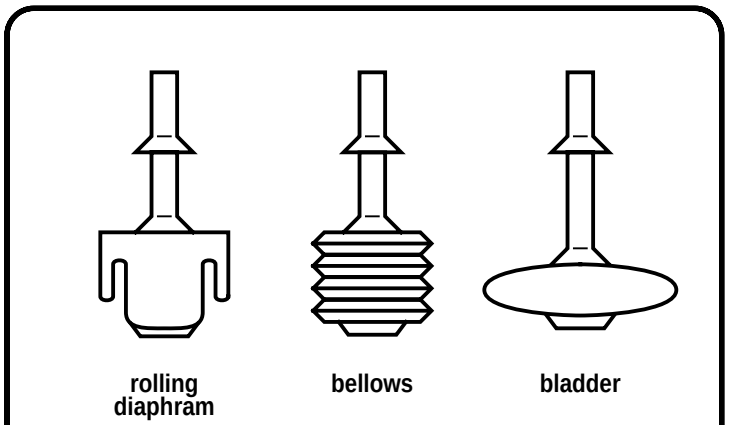


Fig. 2 - Actuators with non-moving seals.

```
100  REM

110  REM ***********************
120  REM *                     *
130  REM * "STRETCHIFIER"  FOR *
140  REM *                     *
150  REM *   APPLEWRITER IIe   *
160  REM *                     *
170  REM *      (DOS 3.3)      *
180  REM *.....................*
190  REM *                     *
200  REM *  COPYRIGHT 1984 BY  *
210  REM *  DON LANCASTER AND  *
220  REM *  SYNERGETICS,  BOX  *
230  REM *  809, THATCHER AZ.  *
240  REM * 85552. 602-428-4073 *
250  REM *                     *
260  REM *   ALL  COMMERCIAL   *
270  REM *   RIGHTS RESERVED   *
280  REM *                     *
290  REM ***********************

300  REM  This mod changes a
310  REM  backup copy of AWIIe
320  REM  so imbedded escape
330  REM  commands pass through
340  REM  the justify routines.

350  REM  This eliminates the
360  REM  "shortline" problem
370  REM  and lets you fully
380  REM  use a fancy printer.

390  REM
         ...................

400  TEXT : HOME : CLEAR
410  HIMEM: 8000
420  VTAB 1: HTAB 7:
      A$ = "Applewriter IIe STRETCHifier":
      GOSUB 980

430  PRINT : GOSUB 1030
440  PRINT
450  FOR N = 1 TO 39: PRINT  CHR$ (127);:
      GOSUB 1020: NEXT N
460  GOSUB 1030
470  VTAB 5: HTAB 1:
      A$ = "This program will patch Applewriter IIe":
      GOSUB 980: PRINT
480  VTAB 6: HTAB 1:
      A$ = "to eliminate the short lines created by":
      GOSUB 980
490  VTAB 7: HTAB 1:
      A$ = "imbedded printer escape sequences.":
      GOSUB 980
500 : GOSUB 1030
510  VTAB 10: HTAB 4:
      A$ = "Patch ONLY your THIRD BACKUP copy!":
      GOSUB 980
520  GOSUB 1030: GOSUB 1030
530  VTAB 14: HTAB 4:
      A$ = "Please put your THIRD BACKUP copy":
      GOSUB 980
540  VTAB 15: HTAB 4:
      A$ = "of AWIIe into Drive #1. Then push":
      GOSUB 980
550  GOSUB 1030
560  VTAB 17: HTAB 12:
      A$ = "<SPACE> to CONTINUE": GOSUB 980
570  VTAB 19: HTAB 19:A$ = "-or-": GOSUB 980
580  VTAB 21: HTAB 13:A$ = "<ESCAPE> to ABORT":
      GOSUB 980
590  VTAB 23: HTAB 19: PRINT "-< >-"
600  VTAB 23: HTAB 21: GET Z$
610  IF Z$ <  > " " THEN 970

                                  (( more . . . ))
```

Fig. 3A - Applewriter IIe Stretchifier patch.

An "alike but different somehow" STRETCHIFIER is needed for the new ProDOS Applewriter 2.0. Listings of both appear in my *Applewriter Cookbook*, and ready to run versions are on their respective companion disks.

How Can I Start My Own technical venture?

Funny you should ask that. For a mere $7.50 you can find out how I make money.

It's an underground classic I wrote a while back called *The Incredible Secret Money Machine*, and it is still going strong. It's been in the Whole Earth top ten for a long time. Here you can learn all about the dangers of two crossed Granfallons, the black widow routine, the DOT technique, the CAP effect, the steam calliope fund, all of those super important honchee guidelines, plus much more.

Hey, this stuff works. I've used it and thoroughly tested everything in it over and over again. Forever, it seems like.

Over and over again.

If you want to get in ahead of the hoarders, just use the help line. We've been known to take an order or two over the phone.

Where do I Get Those Secret Apple manuals?

They weren't supposed to be secret. Only a monumental communications foulup made them that way.

You see, there is a "secret" manual for the IIe called, of all things, the *IIe Technical Reference Manual*. There is a similar "top-secret" manual pair for the IIc called the *IIc Reference Manuals*. And there is one humongous pair of "Q-level" security binders and dozens of support diskettes called *Inside Macintosh* and the *Macintosh Software Supplement* .

It is categorically impossible to do ANYTHING useful on ANY newer Apple machine without these manuals.

Their pricing is not at all

out of line with their contents. In fact, they are worth far more than the asking price. The Iic manuals are stock #A2L4038, while the IIe manual is stock #A2L2073. The Mac stuff is now published by *Addison-Weseley*.

So, what is the problem? Apple reasoned that if they came out with thorough, indispensible, and complete tech manuals that they really would not need full page *Wall Street Journal* ads to promote something so obviously good and so obviously needed.

Particularly if all of their machines are utterly and totally useless without access to these manuals.

In addition, since hackers would be beating the doors down of their dealers to get at these manuals, some bookkeeping could be eliminated by asking those dealers to order the manuals in lots of five or more.

Since the average dealer could easily sell a thousand of these a month if he wanted to, surely buying five would not inconvenience anyone at all, would it?

Well . . .

John Q Hackerturkey walks into his Apple dealer and asks for a book the dealer never heard of. John, of course, does not know the exact name of the book, nor the Apple part number.

During his fifth trip, John finally convinces the dealer that the book really exists. The dealer finally checks, and sure enough. Suprise, suprise. And all the dealer has to do is send Apple a check for $150 to close a $50 sale.

Dealers have lots of names for transactions like this, the nicest and least obscene of which is *negative cash flow*. For some strange reason, they do not agressively seek out such deals.

O.K. Here's how to get the manuals. First, politely but firmly go to a large Apple dealer with the exact part numbers and try and buy them. If you have a friend who is into sumo wrestling and frowns a lot, bring him along for effect.

Renting a Bengal tiger just might also prove useful. If that does not work, group at least three, or preferably five, orders for the same manual at once. You do this with some friends or through your local school or club.

Note that borrowing one of these manuals will not work, since no one in their right mind would ever let one out of their sight for more than a

```
620  REM
          Check Validity

 630  PRINT
 640  PRINT "[D]BLOAD OBJ.APWRT][E,A$2300
 650  IF  PEEK (14720) <  > 235 THEN 950
 660  IF  PEEK (17396) <  > 153 THEN 950
 670  IF  PEEK (17436) <  > 252 THEN 950
 680  PRINT "[D]BLOAD OBJ.APWRT][F,A$2300
 690  IF  PEEK (15063) <  > 100 THEN 950
 700  IF  PEEK (17771) <  > 153 THEN 950
 710  IF  PEEK (17811) <  > 117 THEN 950
 720  POKE 15062,96:  POKE 15063,153
 730  POKE 15064,00:  POKE 15065,22:  POKE 15066,201:
       POKE 15067,155: POKE 15068,208: POKE 15069,04:
       POKE 15070,230: POKE 15071,211

 740  POKE 15072,230: POKE 15073,211: POKE 15074,96:
       POKE 15075,196: POKE 15076,220: POKE 15077,240:
       POKE 15078,14: POKE 15079,185
 750  POKE 15080,00: POKE 15081,22: POKE 15082,201:
       POKE 15083,155: POKE 15084,208: POKE 15085,04:
       POKE 15086,198: POKE 15087,211
 760  POKE 15088,198: POKE 15089,211: POKE 15090,136:
       POKE 15091,208: POKE 15092,238: POKE 15093,76:
       POKE 15094,117: POKE 15095,71
 770  POKE 17771,32: POKE 17772,215: POKE 17773,58
 780  POKE 17810,76: POKE 17811,227: POKE 17812,58
 785  IF PEEK (20365) = 176 THEN POKE 20365, 182:
       REM  RECONNECT HELP SCREENS
 790  PRINT "[D]UNLOCK OBJ.APWRT][F"
 800  PRINT "[D]BSAVE OBJ.APWRT][F,A$2300,L$30D3"
 810  PRINT "[D]LOCK OBJ.APWRT][F"
 820  PRINT "[D]BLOAD OBJ.APWRT][E,A$2300"
 830  POKE 14719,96
 840  POKE 14720,153: POKE 14721,00: POKE 14722,22:
       POKE 14723,201: POKE 14724,155: POKE 14725,208:
       POKE 14726,04: POKE 14727,230
 850  POKE 14728,211: POKE 14729,230: POKE 14730,211:
       POKE 14731,96: POKE 14732,196: POKE 14733,220:
       POKE 14734,240: POKE 14735,14
 860  POKE 14736,185: POKE 14737,00: POKE 14738,22:
       POKE 14739,201: POKE 14740,155: POKE 14741,208:
       POKE 14742,04: POKE 14743,198
 870  POKE 14744,211: POKE 14745,198: POKE 14746,211:
       POKE 14747,136: POKE 14748,208: POKE 14749,238:
       POKE 14750,76: POKE 14751,252
 880  POKE 14752,69
 885  IF PEEK (19988) = 176 THEN POKE 19988, 182:
       REM  RECONNECT HELP SCREENS
 890  POKE 17396,32: POKE 17397,128: POKE 17398,57
 900  POKE 17435,76: POKE 17436,140: POKE 17437,57
 910  PRINT "[D]UNLOCK OBJ.APWRT][E"
 920  PRINT "[D]BSAVE OBJ.APWRT][E,A$2300,L$2F5A"
 930  PRINT "[D]LOCK OBJ.APWRT][E"
 940  TEXT : HOME :A$ = "IT WORKED!": GOSUB 980:
       PRINT : PRINT : PRINT : PRINT : END
 950  TEXT : HOME :
       A$ = "Will not verify as AWIIe; patch ABORTED":
       GOSUB 980: PRINT : PRINT : PRINT :
       PRINT : PRINT : END
 960  GOTO 960
 970  TEXT : HOME : CLEAR : END

                                        (( more . . . ))
```

Fig. 3B - Applewriter Stretchifier, continued . . .

```
 980  REM
           Noisy screen machine

 990  FOR N = 1 TO  LEN (A$): PRINT  MID$ (A$,N,1);
1000  GOSUB 1020: REM  Clickety clack
1010  NEXT N: RETURN
1020  ZZ =  PEEK (49200) +  PEEK (49200):
       FOR M = 1 TO 17: NEXT M: RETURN
1030  FOR N = 0 TO 700: NEXT N: RETURN

 . . . . . .

Do NOT use this program on either "factory" diskette!
Use only on your THIRD or higher backup copies!

This patch gives an exact repair for two-character
escape sequences.  For three-character imbedded
sequences, "bank" characters by using [esc][esc] to
bank two characters, or [esc][esc][@] to bank one
character.  This also works for non-escape imbedded
commands.

In the above listing [D] stands for "control-D"
All other brackets are real.
```

Fig. 3C - Applewriter Stretchifier, concluded.

(Continued from page 2.4)

few moments, if at all. Note also that the IIe and IIc manuals are normally ordered through dealers, while the Macintosh stuff must be ordered directly.

Rumor has it that the *McGraw Hill Bookstore* is one good mail order source that normally stocks and quickly ships the IIe and IIc manuals. There is another rumor that *Addison-Weseley* will shortly republish the manuals as a stock bookstore item.

(UPDATE: Those *A.P.D.A* people now stock all these books, along with most other Apple technical info. More volumes have been added to the Mac book series.)

(*Continued from page 1.3)*

Naturally, the best way to improve your cassette tape reliability is to use a floppy disk instead.

Nonetheless, there is a very old, and little known trick that can dramatically improve the reliability of most any cassette system.

If you could *quietly* listen to the tones as they are going into the computer, you can immediately spot most problems, level settings, noise, hum, or low battery hassles. You can also bypass starting clicks and pops.

This just takes a ten cent modification to your tape recorder. The mod is shown in figure 2.

The trick is to alter your output jack so the speaker is connected in series with a 330 ohm resistor (orange orange brown) whenever anything is plugged into the output jack. We have shown this jack as an "old fashioned" one to make the connections clearer.

Regardless of its shape, when you inspect your output jack, you should find three leads. One permanently goes to ground or a supply line. One permanantly goes to the amplifier output stage. And one switchedly goes to the speaker. When something is plugged into the jack, that something ends up between the amplifier output and the ground, while the speaker gets disconnected.

All you have to do is solder a resistor across the switch contacts that disconnect the speaker. If you are careful, you may not have to do a completely disassembly of your recorder. If you can get down to where you see the solder side of the circuit board holding the output jack, you can often solder the new resistor on the foil side of the board, eliminating a need for the full disassembly. After soldering, make sure you can still plug an output jack in and out, and make sure that the case will still fit over the circuit board without stressing the board or the resistor.

Where can I get an EPROM burned?

Any "old line" electronics distributor will be happy to program an EPROM for you. All it requires is $300, and either an exact working and debugged EPROM for them to copy, or else the code submitted on punched paper tape.

They will alternately accept your duodecimal code written in cuneiform on fired clay tablets. Provided, of course, that you sort them properly and then pack them with a 0.03 cubit spacing of papyrus reeds.

Arrgh . . .

Probably the cheapest local way of getting an EPROM burned is to check your nearest hacker club, or leave a message on a regional electronic bulletin board. Fairly complete lists of most clubs and bulletin boards appear right here in the *Computer Shopper*. Most clubs and bulletin boards do have an up-to-date listing of all the others in your neighborhood, so find one, and you have found them all.

I know of only one hacker EPROM burning service. It's called *E-Tech Services* out of Everett, Washington, and they do good work.

For input, they want a hex image stored on a binary file on an Apple disk. Presumably they will eventually have a direct modem burn process, as well as a way to convert what's-their-name diskettes.

Cost is unbelievably low. Your EPROM or theirs. Yes, they can handle anything up to the big 27128 (16K x 8) EPROM you need for Apple IIc monitor rework.

Don Lancaster's

ASK THE GURU

May, 1985

Option picking
Laser printer info
A tough I/O circuit
Motors for robotic use
Diablo daisywheel stuff

Here we go with cut 3.0 of our brand new *Computer Shopper* feature column. As usual, I'll do what I can here to find any answers to reader hardware, software, or supply problems. Just give me a call or write per the address in the end box. Best calling times are 8-5 weekdays. That's *mountain standard* time. I'll send a free booklist and a special goodies list to you when you respond. And even more stuff if you are an Apple hacker.

While I will try to fairly cover all of the bases, don't expect any unbiased miracles from someone whose vehicle license plate number is 6502. I'll call things as I see them.

Again as usual, I'll gather all of the names and numbers together in a master list at the end. Note that many of these sources are *very* hard to pin down elsewhere.

Thanks to your response so far, I've had to buy a power letter opener. While the twin *Allison* diesel engines on it sure are impressive, the noise and fumes get kind of excessive at times. Looks like a skip loader will be needed to fill the hopper, since snow shovels take far too long. At any rate, keep writing.

Well, here's the warm boot on V3.0 . . .

What is the Diablo 630 "Second Line" problem?

The WP-enhanced Diablo 630 daisywheel printer is the de-facto standard in daisywheel printers. Sadly, there is a bizarre bug in it called the *second line* problem that can royally foul up the works.

The bug appears to be caused by sloppy or wrong initializing of certain variables in the on-board firmware. What happens is this: On the first *right to left* pass after going into fill justify, any thing fancy like an underline or a "funny" spoke access will garble the line and foul up the left margin as well.

Since you nearly always want to kill the justify on the last line in a paragraph, what happens is that any *second line* in the *next* justified paragraph will louse up if you try anything funny in it.

One sledge hammer cure is to print your entire document right to left only. This slows up your printing time considerably. What I usually do is suppress all of the right to left printing only on the problem paragraphs. The normal way I can handle this is to print any paragraph that has *any* underlining at all in it right-to-left only. Should a "funny spoke" get called on line two, I will try to find any old excuse for underlining something, somewhere in the same paragraph.

I have more specific details on this under WPL. Call or write if you are interested.

Beware of the words *Diablo Compatible*. This usually means that the prongs on the power cord of the printer have the same spacing and shape as that on a *Diablo* 630, and will sometimes fit into the same AC wall outlet with a minimum of excessive force. Most toy daisywheel manufacturers are flat out lying through their teeth when they claim they are *Diablo* compable.

Kiddies, if it doesn't have a ready to use, built-in wall to wall proportional space and a full firmware microjustify, can not do any HYPLOT vector graphics, won't accept a metal or plastic daisywheel with 88, 92, or 96 petals, does not have near perfect print quality, has no bidirectional tractor feed, or will not run

APPLE	TITAN 10	BOLD PS
[	[	©
]	]	°
{	{	¶
}	}	§
<	<	[
>	>	═
^	^	!
!	!	|
\	\	<
‘	’	>
|	|	®
~	~	†
@	@	™
[esc] Y	¢	@
[esc] Z	‘	]

Fig. 1 - Differences in daisywheel petals.

continuously for 2000 hours before any adjustment, then it just flat ain't Diablo compatible at all.

While on the subject of *Meleagris Gallaprovos*, beware of those epsilon minus yahoos who use *words per minute* instead of the industry standard *characters per second* when specifying the daisywheel printer speeds. Thus a toy daisywheel rated at "120" is much *slower* than a real one rated at "40".

Gobble gobble.

Why Does a BOLD PS Daisywheel Foul up Some Punctuation?

Not all snap-in daisywheel elements have the same petals in the same positions. Depending on whether it is a business, computer, or custom wheel, certain symbols may end up on the wrong petal or may be missing entirely.

One very popular business wheel is the BOLD PS one. Fifteen spokes are either in the wrong place or are totally different from a standard "computer" wheel such as the TITAN 10. Figure one shows us the differences between the Apple keyboard symbols, and these two wheels.

As you can see, the TITAN 10 closely follows a standard computer keyboard. On the BOLD PS wheel, eight spokes are in the wrong position, and seven are wildly different.

The usual method to rearrange daisywheel spokes is to do a search and replace with your word processor, just before printing. I have some WPL based solutions on this if you need them.

Note that there are two problem spokes on any 96 petal daisywheel element. Since there are a total of 96 printable ASCII characters, and since space and delete are needed for their own uses, you end up two codes shy.

Thus, there are two hidden spokes on most daisywheels. The Diablo way of accessing these is with an imbedded [esc]-Y for the one petal and an [esc]-Z for the other.

On the TITAN 10, the two hidden spokes are the cents and the closing single quote. On the BOLD PS wheel, they are the "@" and the closing bracket.

And, changing the subject slightly, customized daisywheel elements are not all that expensive, especially if you only need a single special character or two. *Camwil* has a special process that breaks off one petal of a stock wheel and replaces it with your own custom symbol.

Where's Enhance II?

Would you believe right here in my hot little hands?

My *Enhancing your Apple II*, Volume II (SAMS #22425) is at long last in print. You can get one now either locally or else autographed copies are available directly from me.

What's in it? Unconditional and absolute reset for the IIe, superb print quality techniques for word processors, a *Castle Wolfenstein* playing aides set, a brand new and software-only VAPORLOCK exact field sync, more on the "tearing method" of quickly analyzing unknown machine language code, and a complete disassembly script of Applewriter IIe that includes all the inside details plus full detailed source code capturing instructions.

The usual companion diskettes are also available.

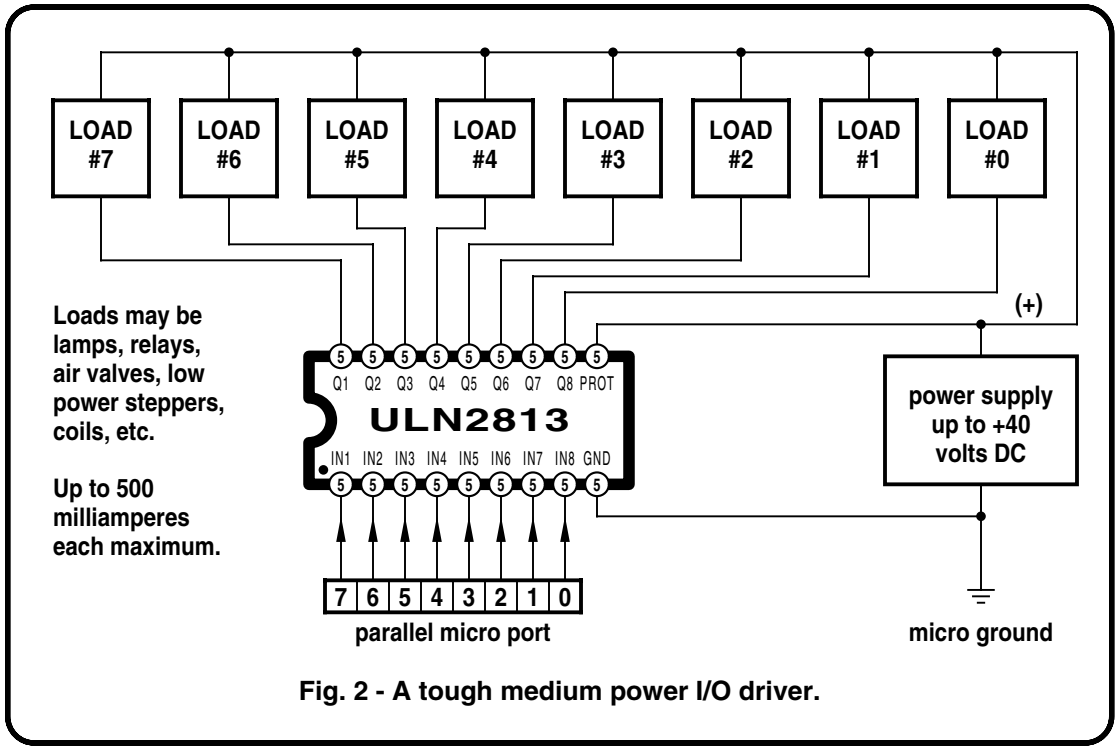


Fig. 2 - A tough medium power I/O driver.

Show me a Tough Integrated Circuit Power Driver

O.K. Figure two shows one of my favorite interface circuits. It uses the *Sprague* ULN 2813. This beauty has eight seperate drivers in it that can handle half an amp each at up to forty volts DC, complete with internal spike protecting diodes. It is easily driven from most any parallel microcomputer port.

Note that the load and the computer must share a common ground connection. Do this *exactly* as shown. Cost is under $3.

I need a Good Book on Motors for Robotics.

How about a great book instead, costing – are you ready for this? – only $3.50?

It is called the *Small Gearmotor Handbook*, and *Bodine* publishes it. See the listing at the end for an address.

This older text is very readable and very heavy on the fundamentals.

What is a "Forced Return" Option Picker?

It is a powerful way to let a computer jump six ways from Sunday that works especially well with the 6502. Figure three shows details.

There are lots of times and places in a program where you want the code to go to any of a large number of different routines. As examples, a word processor's [S] for save command, or when an adventure program discovers that the giant clockwork armadillo is indeed awake and is feeling on the mean side.

Routines that let you go to any of a number of different places in a simple and orderly manner are known as *option pickers*. They are usually used whenever you have to go to six or more different locations, particularly if all of the selections involve more or less random keystrokes, or wild mixtures of letters and control characters.

The forced return theory is elegant, sneaky, and excruciatingly simple. Say you go to a subroutine in a machine language program. The return address of the subroutine gets remembered on the stack, so the program can remember where to go back to. On a subroutine return, the program goes back to whatever called it.

Now for the sneaky part. What if you force a fake subroutine address onto the stack and then call for a return?

Instead of going where you came from, you "return" to the new fake address and pick up there.

Figure three shows you a 6502 example. First you decide which option you need as a hex number. Next you *range check* that option to make darn sure you can go where you think you want to.

Next, you double the number by doing an accumulator shift left or ASL. Then, you transfer the doubled option number to an index, such as the X register. Then go to a table of addresses. Grab the HIGH address byte and shove it on the stack. Grab the LOW address byte and shove it on the stack. Then RTS. Viola.

Should you have jumped to the option picker code, you end up jumping to your new address. If, instead, you have JSRed to your option picker code as a subroutine, then you'll end up JSRing to your option.

This method is consistent, fairly fast, and short. It also bypasses a bug in the jump indirect code in early 6502's. It does this by eliminating any need for indirect jumps.

There's two very important gotchas: The addresses MUST get shoved on the stack HIGH BYTE *first* and LOW BYTE *second*. Also, the addresses in the address table must all be ONE LESS THAN the intended addresses. The reason for this is that the RTS on a 6502 goes one beyond the address on the stack when returning from a subroutine.

With the code shown, the addresses go into the table *frontwards*, or high byte first. If you put the addresses in the usual 6502 "position-page" or "low-hi" form, be sure that you end up shoving the high byte onto the stack *first*.

Where do I get Information On Laser Printers?

Apple has a brand new $75 *Inside Laserwriter* notebook. This does include a thorough discussion of *Postscript*, the brand new, Forth-like typesetting language that placess this laser printer head and shoulders above all its shoddy look-alike imitators.

Also included are the driver

(Continued on page 4.6)

Enter with the range-checked option number loaded in the the accumulator. Then . . .

```
OPICK  ASL  A             ; Double pointer
       TAX                ;  and move to index
       LDA  ADDFILE,X     ; Get HIGH address byte
       PHA                ;  and shove on stack
       LDA  ADDFILE+1,X   ; Get LOW address byte
       PHA                ;  and shove on stack
       RTS                ; "Return" to option
```

Important: Addresses in the address file must all be ONE LESS THAN the intended option address.

Fig. 3 - A "forced subroutine return" option picker.

Don Lancaster's

ASK THE GURU

June, 1985

E. T. watching
Apple IIe "upgrade"
A Softalk replacement
Dual character generator
Applewriter triple header

Gee, Toto, I don't think we are in Kansas any more. My, are there ever a lot of you out there. The helpline phone has literally been ringing off the hook. And mostly with intelligent and well thought out questions. And, amazingly, some of them are even solvable one way or another.

As a reminder, I am somewhat of an Apple freak and an assembly language person, so the further you stray from the way, the truth, and the light, the fewer and the mangier the rabbits I can pull out of the hat. To join in the fun, write or call per the box at the end. Calling is better than writing, and best calling times are 8-5 Mountain Standard time.

This month's news item is super important . . .

What is The Real Word On The Apple IIe upgrade?

Firstoff, this is *not* an upgrade at all. It is a lateral arabesque. If you have had your IIe for a while and if you have lots of your favorite software, the "upgrade" is really a *deadly trap* that may cause you no end of grief.

For you see, when you haul your IIe and $70 off to your friendly local Apple dealer, he will *steal* your existing monitor and character generator ROMs and refuse to give them back to you. This will prevent you from ever again running stock copies of most of your favorite existing software, including *Visicalc* and *Applewriter IIe*, among countless others. Figure one sums up this warning.

Apple's claim of "95 percent compatibility" includes some carefully chosen weasel words. If you read between the lines on the fine print, this 95% compatibility only exists if you buy all new software, and then use only the newest and latest version of each and every software package.

The 95% compatibilty figure is what the Houyhnhnms would very politely term "that which is not so".

In reality, if you have lots of older software around, the compatibility is something under 15% and is falling fast. Particularly if you measure your software compatibility in terms of actual use hours. In general, if the software uses inverse text or uses a software flashing cursor, the display will end up trashed after you do the upgrade.

While most of the older and existing software can be repaired with some simple patches, doing so on a locked and protected program from a non-cooperative or bankrupt software vendor ceases to be a joy real quick like. Especially if you are in the middle of something when you try the upgrade.

Things get really nasty if you have several hundred programs on hand that will not run properly.

Now, from Apple's point of view, and from the desires of a brand new Apple owner, the upgrade is a very good thing. The upgrade makes the IIe and the IIc much more similar, so that identical or nearly identical new software can be run on both machines. The new monitor chips accept lower case Applesloth and monitor commands, can scroll much better, includes a mini-assembler, do provide some new search and ASCII monitor commands, and support full management of interrupts. The mini-assembler is not all that great. *Bugbyter* is better.

The upgrade does include a new 65C02 microprocessor. This dude runs cooler and provides a handful of new instructions. All of which are interesting and useful additions, but none of which are earth shattering. Any attempt to use these new instructions automatically excludes the software from running on a II or II+, since many of these 65C02 instructions can hang an older 6502.

While hand-crafted code that uses these new instructions can end up 10% shorter and 10% faster, no speed or length difference will automatically happen with older or existing code.

WARNING!

The Apple IIe upgrade is really a lateral arabesque. Your dealer will steal your old ROM's, preventing you from ever again running stock and favorite older programs, such as Visicalc or Applewriter IIe, among countless others.

Be ABSOLUTELY CERTAIN you make some EPROM firmware backup copies of the CD monitor ROM (2764), the EF Montitor ROM (2764), and, above all, the F4 video ROM (2732) before you even remotely consider an upgrade.

Fig. 1 - You read it here first.

By the way, if you only want a 65C02, you can get one for free. Which is $70 cheaper than the Apple upgrade. But please, do not tell *Rockwell* who told you to write them on your business letterhead for a free 65C02. And don't let them catch on. Keep all requests very professional and business-like. If they didn't want you to have a free 65C02, they would not have said so in so many of their full page ads.

Anyway, most of the compatibility problems lie in the new character generator chip. An area of the chip has been set aside to hold a *mouse nest* full of mouse screen symbols. But this misses the point. The real use of the mouse nest is that you can now use pull-down or pop-up menus on the stock text screen, letting you pick up all sorts of Mac-like features.

So, for future use, the upgrade is a great idea. What can you do here and now to live with it?

Use this key rule: Make EPROM backup copies of all your firmware! Do this *before* you make the upgrade.

Figure threee is an Apple-sloth routine that is named SNATCHMON. It will automatically capture all of your existing monitor ROMs onto disk, under the filenames of IIEMON.C, IIEMON.D, IIEMON.E, and IIEMON.F. I've purposely left these in four pieces so that an older style EPROM burner can be used if you know what you are up to. Normally, the C and D code goes into one 2764 EPROM, while the E and F code goes into a second 2764.

One prompt and low cost source of EPROM programming are the *E-TECH Services* people. More details appear in my absolute reset package from *Synergetics*.

Capturing that character generator ROM code is a little trickier. The easiest way is to borrow an EPROM programming card and another Apple, put the character generator into the EPROM burn socket and read the ROM code back to disk. A 2732 EPROM can later be used to clone this ROM.

Note that an EPROM programming service isn't able to legally sell you ready to run EPROMS unless you first send them your disk based copy of your own code to be burned. You must send them the code, and not vice versa. This is the way it is.

Figure two shows us a way to have the best of both worlds. With a simple adaptor and a hand-burned double sized 2764 character generator EPROM, you can now use either the old or the new character generator at the flip of a switch. This gives you an enhanced generator for new stuff, and an old one for your existing software. Burn a new ROM clone for the bottom 32K and the old clone on the top 32K of a 2764. To use, if anything ever looks funny on the screen, just change the switch to the other position.

Several gotchas: Be sure to use only premium, machined pin contact sockets and strips for your adaptor. Do NOT use a toggle switch! And, unplug *both* ends of the Apple line cord and rest your hand on the power supply when installing the adaptor. Make sure that the dot and notch points to the front of your machine.

By the way, I have a free CLARIFIER patch that lets you run *Applewriter IIe* on an enhanced IIe or a IIc with minimum problems. I also do have a free listing of all the changes needed to do an absolute "old monitor" reset on the new IIe. Write or call for full details.

Let's briefly run down the reasons your old Apple programs may be incompatible. By far the most common problem is the mouse nest in the character generator. Those screen codes between $40-5F will appear as mouse characters instead of in inverse upper case. While there are two possible codings for inverse uppercase on older Apples, the $40-5F range is far more common, especially for software flashing cursors.

The adaptor of figure two will solve these mouse nest problems. If you are a machine language person, it is

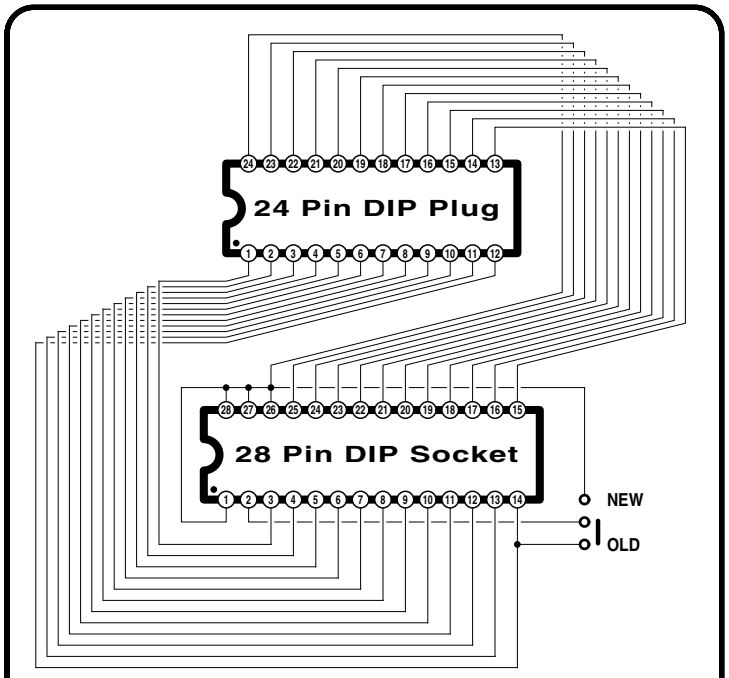


Fig. 2 - A IIe dual character generator adaptor.

usually a simple matter to find the screen routines in your problem code, and trap out anything in the $40-5F range and subtract $40 so it becomes $00-1F instead. The *tearing method* in my *Enhancing your Apple* Vol. I (SAMS #21822) makes this super easy.

You can also write those people who sold you all the software, but chances are they will try to sell you a new upgrade, rather than give you a free patch.

```
10  REM
18  REM ***********************
20  REM *                     *
22  REM *     "SNATCHMON"     *
24  REM * IIE MONITOR GRABBER *
26  REM *  FOR EPROM BURNERS  *
28  REM *                     *
30  REM *     VERSION 1.0     *
32  REM *.....................*
34  REM *                     *
36  REM *  COPYRIGHT 1984 BY  *
38  REM *  DON LANCASTER AND  *
40  REM *  SYNERGETICS,  BOX  *
42  REM *  1300 THATCHER AZ.  *
44  REM * 85552. 602-428-4073 *
46  REM *                     *
48  REM *   ALL  COMMERCIAL   *
50  REM *   RIGHTS RESERVED   *
52  REM *                     *
54  REM ***********************

80  REM  THIS PROGRAM "CAPTURES"
82  REM  THE APPLE IIE MONITOR
84  REM  INTO FOUR 4K WORKFILES
86  REM  FOR USE WITH AN EPROM
88  REM  BURNER WHOSE WORK FILES
90  REM  BEGIN AT HEX $8000.
99  REM
        .....................

100  TEXT : HOME : CLEAR : GOSUB 2000: REM  GET TUTORIAL
110  VTAB 12: HTAB 7: PRINT "MONITOR SNATCH IN PROGRESS"
120  PRINT : FLASH : HTAB 14: PRINT "PLEASE WAIT": NORMAL
199  REM
        .....................

200  REM : $C100-CFFF GRAB

210  POKE 32763,160: POKE 32764,00: POKE 32765,76:
      POKE 32766,44: POKE 32767,254: REM CLEAR Y REGISTER
      BEFORE MOVE!

220  POKE 32768,0: POKE 60,00: POKE 61,128: POKE 62,254:
      POKE 63,128: POKE 66,01: POKE 67,128: CALL 32763:
      REM  ZERO $8000-$80FF

230  POKE 49159,0: POKE 49163,0:
      REM READ INTERNAL C3 AND CX ROM

240  POKE 60,00: POKE 61,193: POKE 62,255: POKE 63,207:
      POKE 66,00: POKE 67,129: CALL 32763:
      REM  MOVE $C100-$CFFF

250  POKE 49158,0: POKE 49162,0:
      REM READ USUAL C3 AND CX SLOTS

260  PRINT : PRINT "[D]BSAVE IIEMON.C,A$8000,L$1000
270  REM : SAVE $C100-$CFFF TO DISK

299  REM
        .....................

300  REM : $D000-DFFF GRAB
310  POKE 60,00: POKE 61,208: POKE 62,255:
      POKE 63,223: POKE 66,00: POKE 67,128:
      CALL 32763: REM  MOVE $D000-$DFFF
                                          (( more ... ))
```

Fig. 3A - Applesloth SNATCHMON monitor grabber.

There are other, and more subtle reasons why some of the oddball older programs may not work with the IIe upgrade, even when using the old character generator.

For instance, a few programs use "illegal" monitor entry points. Many of these locations have been moved somewhere else in the new monitor ROMs. Some wacky protection schemes used a bug in the old 6502 jump indirect command as a silly attempt to confuse any Apple user under six years old. No bug, no access. Interrupts can now cause royal foulups in programs that neither expect nor allow them. Some very oddball 65C02 instructions have different timing than they would on a 6502. This can ruin any extremely time-critical code, such as might be used for disk access, a copy protection scheme, or as an exact video screen lock.

So, by all means make the IIe upgrade. But, be sure to backup all firmware first, and expect to have compatibility problems with most, if not all, of your favorite software. The chances are the double character generator adaptor will solve the worst of your compatibility hassles.

How Can I Start E. T. Watching?

The proper name of this activity is called SETI, short for the *Search for Extra-Terrestial Intelligence.*

Suprisingly, there are a large number of amateur radio astronomers that are doing lots of very interesting, very impressive, and very legitimate research these days.

All on their own, without grants or federal help. Their own trip, rather than someone elses. And done in their own, bureacratic free way.

I feel that an extra-terrestial contact is possibly imminent.

Why do I feel this way? Three main reasons. First, our sun suddenly and dramatically became a radio star in the

early 1950's, thanks to Uncle Milty, Captain Video, and the Roller Derby. Kukla, Fran, and Ollie are now our good will ambassadors to those out there, and they by now have traveled 35 light years from us. This distance is now great enough to attract attention in many candidate star systems.

And, yes, we could detect them sending the same radio energy levels back to us with what we have today in the way of receiver and computer technology.

Secondly, if we were being droid-watched, all of the previous messages transmitted would probably have said "No problem" for the past several million years. Current messages would instead suddenly be of the "Oh oh, we now got a problem" variety.

And, thirdly, we have much more powerful listening tools available to dedicated individuals on a sane budget than we ever did before. And, my oh my, do we now have scads of powerful computers.

As personal and convivial tools. And as mind amplifiers.

Anyway, the center of the amateur SETI activities seems to be a group known as the *Society of Amateur Radio Astronomers*, which is headed by one Jeffrey M. Lichtman.

Jeffrey has self-published several very interesting and very useful books. One is *Microwave Radio Astronomy, An Amateur Introduction*. A second title is *Solar Amateur Radio Astronomy*, and a final nuts-and-bolts book is the *Amateur Radio Astronomers Circuit Cookbook*. Cost is around $35 total for all three.

It's interesting to note the similiarity between all of the radio astronomy circuits and those things that electronic hackers are already doing, such as legal access of satellite broadcasts, and not quite so legal piracy of MDS distribution systems. Much of the same circuitry can be used directly or adapted, and the larger market for the satellite stuff has driven the costs way down. Antenna mounts and tracking mechanisms, of course, scream robotics.

Another thing you can do is stop at the free visitor center at the VLA radio astronomy *Very Large Array* facility outside the town of Magdalena, New Mexico. All visitors are definitely welcome, but play down the ET watching if you expect to be treated seriously and get behind the scenes.

Magdalena lies within the greater Datil–Pietown–Quemado metropolitan area. Take that outer loop expressway through the theater and the industrial district. Then turn left at the fifth cow.

```
320  PRINT : PRINT "[D]BSAVE IIEMON.D,A$8000,L$1000
399  REM
          .....................

400  REM : $E000-EFFF GRAB

410  POKE 60,00: POKE 61,224: POKE 62,255:
      POKE 63,239: POKE 66,00: POKE 67,128: CALL 32763:
      REM     MOVE $D000-$DFFF

420  PRINT : PRINT "[D]BSAVE IIEMON.E,A$8000,L$1000
499  REM
         .....................

500  REM : $F000-FFFF GRAB

510  POKE 60,00: POKE 61,240: POKE 62,255:
      POKE 63,255: POKE 66,00: POKE 67,128:
      CALL 32763: REM  MOVE $F000-$FFFF

520  PRINT : PRINT "[D]BSAVE IIEMON.F,A$8000,L$1000
599  REM
         .....................

600  TEXT : HOME : CLEAR
610  FOR N = 1 TO 30:ZZ =  PEEK (49200) +  PEEK (49200) +
      PEEK (49200): NEXT N: REM BRACK

900  PRINT "MONITOR SNATCH COMPLETE": PRINT : END
910  REM
         .....................

2000  REM : TUROTIAL AND PROMPT

2008  POKE 49167,0: REM  ALTSET ON
2010  VTAB 1: HTAB 12: FOR NN = 1 TO 15: PRINT  CHR$
       (127);: NEXT NN: PRINT
2012  HTAB 12: PRINT  CHR$ (127);"  SNATCHMON  ";
       CHR$ (127)
2014  HTAB 12: FOR NN = 1 TO 15: PRINT  CHR$ (127);:
       NEXT NN:    *
       PRINT
2015  PRINT : PRINT
2016  PRINT "This program 'captures' the Apple IIe's
       monitor ROM into four 4Kx8 binary files named
       IIEROM.C thru IIEROM.F": PRINT
2017  POKE 1677,162: POKE 1686,162:
       REM  REAL QUOTES WITHOUT TEARS
2018  PRINT "Default use address is $8000, as needed
       by the MPC ap-ep EPROM burner.": PRINT : PRINT
2020  PRINT "Please insert SAVE disk into Drive 1.":
       PRINT : PRINT : PRINT "Then press  < space > to
       CONTINUE": PRINT  *
2022  HTAB 15: PRINT "-or-": PRINT
2024  HTAB 13: PRINT "< escape> to ABORT": PRINT : PRINT
2026  HTAB 13: PRINT "---< >---"; CHR$ (08); CHR$ (08);
       CHR$ (08); CHR$ (08); CHR$ (08);
2028  GET Z$
2030  IF Z$ >  < " " THEN  TEXT : HOME : CLEAR : END
2040  TEXT : HOME
2050  POKE 49166,0: REM  PRIMARY CHARACTER SET
2100  RETURN

Gotchas:  [D] means <control-D>.
```

Fig. 3B - SNATCHMON, concluded.

And, while you are in the neighborhood, drop in and help me quest a tinaja or two. The VLA is only half a day's drive away.

Is There a Replacement for SOFTALK?

Before their untimely demise, *Softalk* was far and away the "best" Apple magazine, when measured in terms of user interest, enthuasiasm, commitment, and on their no nonsense, no bullshit coverage of beginner to mid-level technical and programming information.

A much leaner and much more compact replacement for *Softalk* now exists, edited by Tom Weishaar, who was one of the better of *Softalk's* contributing editors.

The magazine is now called *Open-Apple*, and it is now up to its sixth issue.

By special arrangements, a free sample copy is available to any *Computer Shopper* subscriber, provided you use the secret password "tinaja" in your request.

Cost is $24 per year. There are no vaporware ads, no misleading ads, no bait and switch ads, no not-in-stock ads, no ads for overpriced products, and no pure puffery ads. In fact, as you might have guessed by now, there are no ads at all. Just beginner to mid-level Apple tech information in the most solid and concise form, just as you'd expect from Tom.

```
xx p
 pnd
 ppr[L]
 ppr    Don Lancaster's TRIPLE HEADER WPL Utility
 ppr ============================================
 ppr
 ppr This module will print a document with a three
 ppr line single spaced header, using print constants
 ppr built into the WPL code itself.
 ppr
 ppr To continue, you must have the document to be
 ppr printed inside your Apple, and a disk having
 ppr this program AND lots of space left must be
 ppr placed in the active drive.
 ppr
 pin        OK to continue (Y/N) ------> =$A
 ppr
 pcs/Y/$A/
 pgoaa
 pcs/y/$A/
 pgoaa
 pqt
aa pin   Top Header -----> =$A
   pin   Middle Header --> =$B
   pin   Bottom Header --> =$C
 ppr
   pin   Left Margin ----> =$D
   psx$D
   pin   Right Margin ---> =$D
 psy$D
 psz$D
 psz-(x)
 p
 p  NOTE: (z) is RM-LM
 p
 ppr
 pin          OK to continue (Y/N) --------> =$D
 pcs/Y/$D/
 pgobb
 pcs/y/$D/
 pgobb
 ppr[L]
 pgoxx
bb ppr
 ppr    *** formatting - may take several minutes ***
 p
 p     print third header to width RM-LM
 plm0
 prm(z)
 ppn1
 ptm4
 pbm4
 ppl58
 ppi63
 pli0
 psp0
 ppd8
 pcr0
 put
 plj
 ptl$C
 pbl
                                  (( more . . . ))
```

Fig. 4A - WPL Triple Header for Applewriter

Show me an Applewriter Triple Header.

Figure four shows you a WPL routine that will automatically handle a triple header of any combination of left, centered, and right side entries, with or without page numbers in any position.

This particular version will work best on ProDOS Applewriter 2.0 or 2.1. Changes may be needed for certain printer and I/O card combinations, on other versions, or to suppress the first page headers.

To use it, you put the file you want printed in your IIe or IIc. Then, you put a disk that has TRIPLE.HDR.WPL and lots of scratch space into the active drive. Finally, a DO TRIPLE.HDR.WPL will next prompt you for the header info and then do all the nasty stuff for you both invisibly and automatically.

When prompted, you enter each header line exactly the same way you would a TL or a BL entry, using delimiters and a "#" to substitute for the page number. Only minus the TL or BL prefix.

Since this new WPL routine does the impossible, it takes its good old time about doing it. Up to several minutes may be needed to format and then print a longer textfile. All the sneaky stuff is handled by a trip or two through PD8 land.

Neat, huh?

If you get tricky enough, you might have single spaced headers and double spaced text, or vice versa. Any way you like at all.

By embedding a TL *command* as a TL line (!), you can neatly handle even and odd page formats on your successive pages.

You can alter the code for any number of header and footer lines. If you wanted to, you could even use a 33 line header and a 33 line footer, provided you do not have too much to say in the body of your text file.

So there.

```
p
pnp
ZZZ
y
p
p   print second header
p
ny
p
lZZZ
p
b
f<<.rm240><
y?
p
pcr0
ptm0
pbm0
ppm0
plm0
prm(z)
ptl$B
ppi64
ppl64
put
pnp
ZZZ
y
p
p  print final with first header
p
ny
lZZZ
p
b
f<<.rm240><
y?
p
ppi66
ppl65
ppd1
ptl$A
pcr1
plm(x)
prm(y)
put
pnp
p
p   erase old scratch file
p
oe
ZZZ
p
pyd
pqt
. . . . . .

Gotchas:  [L] means <control-L>.
To use, [P] DO TRIPLE.HDR.WPL
```

Fig. 4B - WPL Triple Header, concluded.

(continued from page 3.3)

and support software disks that let you link your own routines to control the printer.

Now for the neat part. The part the Mac people do not want you to hear. Try and guess which software package makes the best use of *Postcript* and the Laserwriter.

No, not *MacPaint*. Why? Because *MacPaint* has no way to use high resolution typography.

No, not *MacWrite*. Why? Because *MacWrite* does not handle *Postscript* very well.

No, not *MacDraw*, first because *MacDraw* is far and away the most user-vicious of all Mac software, besides its not existing at all in usable form. The inability to use bitmaps totally cripples *Mac*-Draw. Besides, every seven weeks, Apple very carefully takes MacDraw apart, and then removes all of the old bugs. Then Apple carefully puts lots of brand new bugs back in, so the next seven week cycle can continue. And there's still no way to directly command *Postscript*.

So which is the best program to explore *Postscript* ?

Are you ready for this?

Applewriter IIe !

That's right. A good old Apple IIe can explore *Postscript* far easier and far better than a Mac can, at least right now at reasonable cost. WPL speaks *Postscript* as if it were its native tounge. Even bit-mapped graphics are trivially includable as text files. And, you don't need an *Appletalk* connector or driver. All you do is plug the RS232 output from your IIe serial card into the RS232 input on the laser printer.

Proof that this is far and away the best route to explore *Postscript* is the derogratory and underhanded name the Mac people have put on such a sneaky "low-tech" trick.

They call it the *etch-a-sketch* mode.

We'll have lots more on the many wonders of laser printing sometime else.

Don Lancaster's

ASK THE GURU

July, 1985

Disassembly aliasing
A Shuffling algorithm
Machine language study
Getting inside information
Apple II+ to IIe conversions

Zounds. We are already up to release 5.0. This month, I can use your help on something. If someone calls me with an Apple question or a 6502 machine language problem, I can often immediately help or else may know where to send them to get an answer.

But not so with all those CP/M, Compac, Commodore 64, Timex, Kaypro, the Rat's Mouth Zero, Atari, Heath, et. al. problems. Could some of you out there clue me in to the best user groups, specialty magazines that are actually used and read, help lines, bulletin boards, or other generic places to get effective answers to all of the stock and common problems on these machines?

One clarification from last month. The IIe upgrade *will* handle inverse uppercase on old programs without hassles, *provided* that the older program uses the stock $FDF0 "Fideyfoo" or $FDED monitor entry points *and* if the 80 column firmware is active, *and* if the mouse nest is switched off.

Unfortunately, many, if not most, of these popular and higher quality older Apple programs use their own display routines, particularly for direct pokes of inverse characters directly to the screen. Naturally, the programs you are most likley to use – such as older *Applewriter* and *Visicalc* – are the ones that have the worst hassles with the upgrade.

The best and the cheapest solution remains the character generator switcher shown you last month. Unless you want to buy all newer software or install bunches of patches. As a reminder, I have some free "detrashing" patches available for Applewriter IIe. These do work on either the IIc or the new IIe. Just ask.

On to some utter ignomity. Oh, the shame and horror of it all.

SAMS tells me we still do have over 17 copies of *Enhancing your Apple II*, vol II (#22415) left. It severely hurts an author's credibility and reputation if a book does not sell out its first printing within ten days of its release.

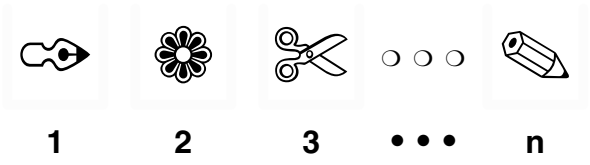
Now, I hate to get nasty, but if certain of you do not shape up by next month, I will be forced to publish the names of the remaining 17 *Computer Shopper* readers who have not yet seen the way, the truth, and the light.

This, of course, will exclude you from ever getting invited on a tinaja quest.

Oh well.

As usual, be sure to keep the post office hacked off and the phone lines jammed up. Use the number and address in that end box to get in on all the fun.

On to Release 5.0 . . .



Take the object in Bin #1 and exchange it with the object in a random bin #1 to #n.

Then, take the object in bin #2 and exchange it with the object in a random bin #2 to #n.

Then, take the object in bin #3 and exchange it with the object in a random bin #3 to #n.

Repeat this process until all objects have been exchanged either with themselves or with an object in a higher numbered bin.

Fig. 1 - How to shuffle "n" objects.

Where do I get Apple Gonzo Insider Information?

Getting the real insider information on all those Apple goodies can be a real hassle. There is so much garbage out there masquerading as useful or needed info, and lots of gaps in what is really needed. And it is often hard to get a straight story about the warts on a product. Or the interface between two competing third party products. Or the bugs and their fixes. And particularly the buggy fixes that do not apply to you.

That's no bug, its a feature! Where to start on all this?

Firstoff, see if you can't beg or borrow a copy of the older *Apple Tech Notes*, as published by the *International Apple Core* and available through member IAC clubs. This is a big fat notebook that held the answers to all of the questions that were asked to Apple on all of their products. While a pre-IIe era book (1981-82), there is plenty of

good stuff in here, especially the very rare Applesloth material in section 2600.

The best insider book on Applesloth that I know is *All About Applesoft*, published by *CALL A.P.P.L.E.* Included is complete and thorough coverage of pointers, variables, memory space, arrays, etc., that you need to know to intelligently use this language.

A rather good Applesloth source code capturing process that includes extensive and thorough comments is called *Applesoft Source Code On Disk*, and is available from *S-C Software*. To avoid copyright hassles, the program does require an Apple with Applesloth in it before it can run. This utility is best used under the S-C assembler.

On to DOS and ProDOS. The finest DOS book is, of course, *Beneath Apple DOS* by *Quality Software*. the *All about DOS*, again by *CALL A.P.P.L.E.* is another good choice, and the two books very much complement each other. For ProDOS, there's *Beneath Apple ProDOS* by *Quality Software*, and the *ProDOS Technical Reference Manual* (#A2W0010) directly from *Apple*. While no *All About ProDOS* exists at this writing, you'll find excellent tutorial and technical articles on ProDOS scattered throughout various issues of *CALL A.P.P.L.E.*

On Applewriter, there's *All About Applewriter*, again by *CALL A.P.P.L.E.*, the 16 disk side *Applewriter Toolkit* packages by *Synergetics*, and my *Don Lancaster's Applewriter Cookbook*. I do stock all of these here.

Turning now to all of those on-line resources, the MAUG (Micronetworked Apple Users Group) on *Compuserve* seems to be the center of the action. There is a bewildering and mind-blowing collection of goodies here. The MAUG is currently moving over to the *Delphi* network as well.

Now for the real insider stuff. You are not supposed to know this, but if you are an Apple developer, a dealer, or a manufacturer of Apple related products, you qualify for access to the *AppleLink* network. This is where all of the inside stuff comes down. And, since this is the first place dealers go for info, if you can get your material into the network, you have "first dibs" on zillions of customers that otherwise would never hear about you.

AppleLink is reasonably open, but since it is primarily a dealer service, they don't want it jammed up forever with end user requests. The dealers have been told that they may give out the access number and the password to a few select customers.

Please, if you access this resource as a user, do so after hours and preferably very late at night.

There is a special AppleLink resource called the *Third Party Library* that, for a fee, can get your stuff into the pipe. Uh, better not tell them who sent you, but be sure to check it out.

How do I Shuffle a Deck of Cards?

The proper name for this sort of thing is called *randomizing without a replacement*. Besides the obvious use in card games, the problem also comes up in educational software where you want to give a student a set of problems in random order, yet never repeat or miss.

A first, and an incredibly stupid, stab at shuffling might grab a random integer. Then grab a second random integer. Whoops, better check to make sure you did not get the same one over again. Repeat this 52 times for a card deck.

Among the many problems with this route is that, when you get near the end of the deck, you will be throwing

```
0E 4C 69   ASL  $694C
6E 65 73   ROR  $7365
20 70 65   JSR  $6570
72         ???
20 69 6E   JSR  $6E69
63         ???
68         PLA
```

Fig. 2 - ASCII message disassembly aliasing.

```
98         TYA
15 43      ORA  $43,X
15 3F      ORA  $3F,X
15 49      ORA  $49,X
15 51      ORA  $51,X
15 75      ORA  $75,X
15 A0      ORA  $A0,X
16 0E      ASL  $0E,X
16 16      ASL  $16,X
16 21      ASL  $21,X
```

Fig. 3 - Address list disassembly aliasing.

hundreds of cards away for each of those good ones you end up with.

This takes forever, even in machine language. In BASIC, it takes forever cubed.

Which might be just fine for modeling earth tides, snail geriatrics, or congressional reform, but is nowhere near real time at all.

Instead, there is a simple, fast, and easy way to shuffle n objects. Its even got the stamp of approval from the traditional dino people. Even Knuth. What amazes me is how little the method is used or even known, and the utterly attrocious substitutions that people dream up for it.

Figure one shows us the details. Assume you have ten objects in ten bins you want to shuffle. You pick a random integer number from 1 to 10. Now interchange what is in bin #1 with whatever bin the random integer points to. If the random integer is also 1, then interchange the contents of bin #1 with itself. Put another way, you do nothing when and if this happens.

Now, grab a second random integer number, *this time from 2 to 10*, and interchange what is in bin #2 with whatever bin the random integer points to. Repeat the process ten times.

On pass three, interchange bin #3 with a bin picked at random from 3 to 10. On pass four, interchange bin #4 with one picked at random from 4 to 10, and so on.

Note that any object can end up in any bin. A few objects might end up back in the same bins, just as you would expect if you shuffled the objects by hand for hours and hours. You could even end up with all the objects back in the same bin where they started. The odds of this ever happening are, of course, quite remote.

As most people know, there is a fatal and unfixable flaw in that Applesloth random number generator. Instead, you will get the best and the fastest results through use of a machine language pseudo-random integer generator.

In fact, on the Apple, a machine language card shuffle happens so fast that you purposely have to *slow it down* by using a sound effect that sounds just like a deck of cards being shuffled. Otherwise, nobody will believe it really happened.

You can even redeal from the same shuffled deck, for replays, repeat student drills, or for industrial "noise that repeats". Complete details on this, including source code, the object code, and working programs appears as Ripoff Modules #7 and #8 of my *Assembly Cookbook* (SAMS #22331). See you there.

And for heavy background theory, check into Knuth's *The Art of Computer Programming*, volume II, chapter three. *Addison-Wesley* publishes it and it is a classic in every sense of the word.

The Knuth algorithm simply swaps each of the bins with any other bin selected at random. This does have a slight bug in that it introduces some bias if you do not have

(wrong)

```
7E 6A AD   ROR   $AD6A,X
64         ???
79 D0 0A   ADC   $0A0D
20 7B 10   JSR   $107B
A9 00      LDA   $00
85 21      STA   $21
```

(right)

```
AD 64 79   LDA   $7964
D0 0A      BNE   $7863
20 7B 10   JSR   $107B
A9 00      LDA   $00
85 21      STA   $21
```

Fig. 4 - Wrong start disassembly aliasing.

(wrong)

```
20 00 BF   JSR   $BF00
CA         DEX
FF         ???
6E D0 03   ROR   $03D0
60         RTS
```

(right)

```
20 00 BF   JSR   $BF00
CA         DFB   $CA
FF 6E      DFW   $6EFF
D0 03      BNE   $6ED4
60         RTS
```

Fig. 5 - Embedded values disassembly aliasing.

many bins. The bias bug is eliminated by always interchanging a bin only with itself or with a *higher* bin.

For most uses though, the bias becomes very small for shuffles of eight or more objects, so you might prefer using the simpler and faster "swap with anything" code.

How can I convert an Apple II+ into a IIe?

Simply remove your line cord from the II+. Save the line cord and throw everything else away. Then find a IIe that is missing a line cord. Then plug your II+ line cord into the IIe.

The update is that fast and that simple. This route is also far and away the easiest, the cheapest, and the technically cleanest as well, resulting in 100% compatibility.

In fact, it is the *only* way.

What Causes Aliasing During Disassembly?

A *disassembler* is some software, firmware, or people-based routine that takes a block of machine language code, and tries to convert it back into assembly language computer op-code listings.

Disassemblers are extremely powerful debugging and analysis tools. Examples do include the Apple "L" monitor command or *Bugbyter* for an "automated pocket card" disassembly, and programs like RAK-Ware's *Disasm IIe* that automatically can try to capture source code simply by reading the final ready-to-run object code.

But just about any disassembler will sometimes lie like a rug.

You see, a disassembler will only work properly if it starts at a valid entry point in valid working code. If you do not disassemble from a valid entry point in valid working code, you will get *aliasing* in which the disassembler will tell you "that which is not so". The usual signs of aliasing are bunches of question marks denoting illegal op codes, or else highly unusual op codes working in strange address modes that involve bizarre addresses.

Normally, when you first begin a disassembly, you may not know where the valid entry points exist in valid and runnable code. So, you will get bunches of aliases on the first disassembly pass. By a careful study of these aliases and by recognizing how and why they occur, you can then make a second pass that gives you a clean disassembly on only the good parts of your actual code.

Recognizing and working with aliasing is thus a *very* powerful analysis tool.

Let us look at several good examples of this disassembly aliasing. We'll speak 6502 here, but the same idea will hold for any micro family.

In figure two, we have attempted to disassemble a textfile of ASCII characters. Since there are no valid op-codes here, you get lots of aliasing. Uppercase ASCII text will often be lots of $4X and $5X entries, combined with many $20 spaces and $0D carriage returns. Usually the aliasing will clue you in to the file, and the patterns will tell you whether you have ASCII messages or not. Thus, aliasing is useful to pin down text and disk message files.

In figure three, we try to disassemble a list of module address vectors. Once again, we get a bunch of aliases, this time involving off-the-wall operand addresses. Note here the pattern with each second byte being the same number and the addresses usually, but not necessarily always, working their way slowly upwards through memory. Finding option picking lists is essential to cracking any longer code.

In figure three, we disassemble legal code but start at an illegal entry point. The first two or three addresses will alias, but the code usually will straighten itself up and fly right after several lines. The key here is reasonableness. Often, other parts of the code will show you the

(Continued on page 7.5)

```
32          ???
00          BRK
4B          ???
00          BRK
6E 00 87    ROR  $8700
00          BRK
96 00       STX  $00,Y
2C 01 58    BIT  $5801
02          ???
B0 04       BCS  $1234
08          PHP
07          ???
60          RTS
09 10       ORA  $10
0E C0 12    ASL  $12C0
20 1C 18    JSR  $801C
25 00       AND  $00
4B          ???
```

Fig. 6 - The great aliasing contest (see text.)

Don Lancaster's

ASK THE GURU

August, 1985

Typesetting an ad
Diablo 630 emulation
ProDOS TYPE command
Educational software sales
Laserwriter first impressions

We have got us some really heavy topics this month, so let's just jump right in with both feet . . .

Explain the ProDOS TYPE Command

That TYPE command in ProDOS lets you change any file type into any other file type. While extremely powerful, it is not too widely used nor well known.

Let's look at two examples of what you can do with this beauty. Say you want to load a SYS type file as a binary image. SImply do a BLOAD MYFILE, A$2000, TSYS. Or, say you have a binary text image in your machine. To capture this code as a text file, just create and open a textfile and then do a BSAVE MYLIST, A$3456, E$3A52, TTXT.

The TSYS is short for *type-system*. The TTXT is short for *type-text*. The other type commands include TBIN for *type binary*, TBAS for *type Basic*, and so on.

Two gotchas: Be sure to use different file names for each type of program, even if they are otherwise the same. Secondly, you cannot do a TTXT unless the text file has been previously created and opened.

What are The Secrets to Selling Educational Software?

Here, revealed for the first time, are the two key secrets to successful marketing of educational software. These two secrets involve passing the *triple-B* test and obeying the *masking* criteria.

In order to get through the administrative selection process, the program abolutely *must* pass the triple-B test. For educational software to be bought at all, it must be *Banal*. It absolutely must be *Boring*. And, above all, it has to be just plain *Bad*.

The triple-B test is only the beginning. In addition, for a program to even be remotely considered for use as educational software, all superficial design flaws in the program must completely and totally mask all of the fundamental design flaws.

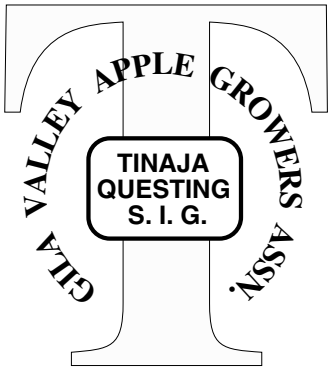


Fig. 1 - Club shoulder patch done with Applewriter.

And, of course, all of the superficial bugs must completely and totally mask all of the fundamental bugs.

If you check *all* of the educational software available today, you will find *all* of the best sellers strictly and absolutely follow the triple-B and the total masking rules.

What's the real word On the Laserwriter?

I have had mine only for a week or so. My, oh my what a machine.

You know all those slide rule manufacturers, mechanical adding machine people, the alarm clock builders, and such? Well, they are about to be joined by a great heaping bunch of ad agencies, sign painters, printers, quick-copy centers, author's agents, vanity publishers, graphics arts suppliers, and the zillions of others who will not have the slightest idea what hit them or why.

Kiddies, the price of typesetting with full graphics has just dropped below a dollar a page. Not one red cent more.

Changes and revisions are now, of course, free.

Where to even begin?

The *Laserwriter* sets *ANY* size type you like, mixed in with *ANY* style of graphics in *ANY* size you can imagine in *ANY* configuration. You can easily handle such things as homecoming posters and sidewalk sized centenial banners. Signs of any size and shape are trivial.

There is even a neat new *Postscript* procedure that will automatically chop up the sign or poster into as many 8-1/2 x 11 pieces as are required. Later, you tape the pages back together to get up to whatever size you need.

It is trivially easy to move,

spin, repeat, or stretch your image every which way but loose. Any shade of gray you want. And even four colors with repeated passes. Variable size and slant text along a circular or even an arbitrary path is easily done.

Actually, *Apple* so far has refused to state the maximum type size on the *Laserwriter*. I've used it at 16,000 point to create letters that are eighteen feet high. While this is fine for putting names on water towers, *Apple* is leaving us up in the air as to the maximum size of airport hangar roof that can easily be labeled.

One thing not well known (guess why?) is that *Applewriter* on an *Apple IIe* does as good if not a better job than the *MacIntosh* does in driving the *Laserwriter*. And yes, that includes graphics. Graphics so fancy that they are simply not available out of any of the common Mac programs.

We'll note in passing that *MacPaint* has no way to handle all those high resolution alphabets, *MacWrite* is not good at larger or integrated graphics, and *MacDraw* is a cruel joke at best.

HIRES dumps or any other bit-mapped graphics are done by converting them to hex ASCII character pairs that are easily handled by *Applewriter* and its *WPL* supervisory language. But the best graphics are done by using the direct *Postscript* commands, which are infinitely more stunning, more flexible, and more powerful than anything that Mac's *Quickdraw* routines can even dream about.

The *Appletalk* network stuff is also not needed. In fact, this can severely limit what you can do with the *Laserwriter*. Among other things, it excludes you from using the so-called *Diablo* emulation mode.

And, yes, I will put my money where my mouth is.

In front of me are five *Laserwriter* images created with *Applewriter* on a *IIe*. I'll be glad to send you some free

```
% Requires Don Lancaster justify routines for cj & pj.
% Free printed copies of these on request.

% outline black box
/bl 137 def /bw 330 def /br bl bw add def /bt 750 def /bh 576 def /bb
bt bh sub def /rad 20 def newpath /strt br bl add 2 div def strt bt
moveto br bt br bb rad arcto br bb bl bb rad arcto bl bb bl bt rad
arcto bl bt strt bt rad arcto closepath 0 setgray fill

% white inset box
/xcen bw 2 div bl add def /ytop bt def /ybot bb def /sidewidth 8 def
/bl bl sidewidth  add def /bw bw sidewidth 2 mul sub def /br bl bw
add def /bt bt 56 sub def /bh bh 152 sub def /bb bt bh sub def /rad
rad 5 sub def newpath /strt br bl add 2 div def strt bt moveto br bt
br bb rad arcto br bb bl bb rad arcto bl bb bl bt rad arcto bl bt strt
bt rad arcto closepath 1 setgray fill 0 setgray

% reverse stuff
/Helvetica-Bold findfont [20 0 0 16 0 0]  makefont setfont /ypos ytop
21 sub def 1 setgray /kern 2 def xcen ypos (NEW  FROM) cj /ypos
ybot 47 add def /yinc 18 def xcen ypos /kern 2 def

(Box 809-CSL
Thatcher, AZ 85552
(602) 428-4073) cj

/ypos ypos 24 sub def /Helvetica-Bold findfont [28 0 0 24 0 0 ]
makefont setfont /ypos ytop 45 sub def /kern 2 def xcen ypos
(DON  LANCASTER) cj /ypos ybot 69 add def /Helvetica-Bold findfont
[30 0 0 24 0 0 ] makefont setfont /kern 3 def xcen ypos
(SYNERGETICS) cj

% visa
0 setgray /ypos bb 8 add def /xpos br 70 sub def /Helvetica-Bold
findfont [14 0 0 14 0 0] makefont setfont xpos ypos moveto 0.5 0
(VISA/MC) ashow

% big print
/xcen bw 2 div bl add def /ypos bt 45 sub def /Helvetica-Bold
findfont [38 0 0 38 0 0] makefont setfont /yinc 40 def /kern 0
def xcen ypos

(APPLEWRITER
LASERWRITER
UTILITIES) cj

/ypos bt 344 sub def xpos ypos /Helvetica-Bold findfont 45 scalefont
setfont ($49.95) cj /ypos bt 160 sub def /xpos bl 20 add def
/linewidth 272 def /Helvetica-Bold findfont [20 0 0 22 0 0 ]
makefont setfont /yinc 24 def

(Five diskette package gives
unmatchably superb page
graphics for your Apple IIe.
) pj

/ypos ypos 12 add def

(Unlocked. Requires ProDOS
Applewriter, Apple IIe and
an Apple Super Serial Card.
)pj

/ypos bt 376 sub def /Helvetica-Bold findfont [17 0 0 18.5 0 0 ]
makefont setfont /yinc 20 def xcen ypos

(FREE Demo Pack
FREE Laser Helpline) cj

showpage
```

Fig. 2 - Postscript code for a Computer Shopper ad.

copies of them. I'll also gladly give a free *SAMS* book to anyone who can show me *ANY* way at all to do these on a Mac that is even remotely as cheap, as easy, as powerful, and as convenient as using *Applewriter* on a *IIe*.

Figure one shows us a typical image. This was done on a *IIe* under *Applewriter*. Once you know exactly what you are doing and have built a library of goodies up, this complex an image should take you around ten minutes to program and seventeen seconds to print. Materials cost is under four cents.

One neat thing that is not at all obvious is that *it does not matter what order you put the image onto the Laserwriter's bitmap*. You are free to, say, do your backgrounds first, the artwork second, your headlines third, and your fine print last.

More importantly, you do a form letter by putting the letter *in the printer* and then changing only the name and address for each repeated pass. A *IIe*, again with *Applewriter*, can simply and easily process hundreds, of letters at a whack.

Watching customized form letters quietly pour out of a machine at an eight letter per minute rate is a joy to behold. Yes, I do have automatic software for both form letters and envelopes. Just call or write.

There are some problems. The quality is not quite what a printer would call "typeset", since the resolution is "only" 300 dots per inch. A printer might call this *tabloid* quality. One route to beat this is to work double or triple size and then photoreduce the result.

Another way is to find one of the many "real" typesetting machines that speak the same *Postscript* language the *Laserwriter* does. Once you have exactly what you want, the *same* software on the *same* machine can give you arbitrarily good typeset quality, to 2600 dpi and beyond.

Instead, an awful lot of people are going to have a rather obvious response if they were asked to accept a slight reduction in "typeset" quality for 1/100th the cost, 1/100th the time, and with 100 times the convenience.

Hail yaess.

A second obvious problem is that there is no way at all to tractor feed anything, so things like envelopes, labels, or business cards have to be hand fed.

Turning to nit picking, the laser engine itself does show some rather bizarre human engineering. You need continuous access to all four of the sides and the top of the printer, including the ability to simultaneously observe status lights at *both* the front and the back of the machine.

An unreachable and often-used selector switch requires that you add a shaft extender before you can even put a knob on it, let alone use it.

The paper tray is far too shallow to the point of being a joke. An unnecessary "U" turn in the paper path adds to the jam potential and limits paper weight, but accomplishes nothing useful. And while ridiculously quieter than most other printers, a laminar fan redesign could make the silence totally eerie.

That eight pages per minute rating is the absolute top speed on repeat copies, and then only after you custom flip a magic *prefeed* software switch that ups the wear and tear on the laser engine. It is very easy for a fancy image to take several minutes or more to process and output.

Which is like buying a "165 mile per hour" bicycle and then finding out it only goes that fast when dropped off a very high cliff. On the moon.

Still, the print speed is very respectable. Awesome, even.

The print quality is quite

First, select a daisywheel number from this list:

0 - Courier	**8 - Helvetica**
1 - Courier Italic	**9 - Helvetica Italic**
2 - Courier Bold	**10 - Helvetica Bold**
3 - Courier Bold Italic	**11 - Helvetica Bold Italic**
4 - Times	**12 - symbol**
5 - Times Italic	
6 - Times Bold	
7 - Times Bold Italic	

To change the "normal" daisywheel, use this Postscript command in the Postscript mode:

```
/normalwheelchoice 8 def

serverdict begin 0 exitserver statusdict begin
62 normalwheelchoice seteescratch
```

To change the "bold" daisywheel, use this Postscript command in the Postscript mode:

```
/boldwheelchoice 10 def

serverdict begin 0 exitserver statusdict begin
61 boldwheelchoice seteescratch
```

Do not change daisywheels more than 10,000 times over the life of the machine.

Fig. 3 - Postscript code to swap daisywheels.

impressive on a wide range of different papers. The gray images tend to blotch a little on high rag papers. Giant black areas aren't quite solid either, but they are certainly usable. You aren't supposed to use thermal "raised print" letterheads, but I suspect you can get away with it.

Yet another problem is that their *Diablo* emulation mode, like everyone else's, just flat out does not emulate a real *Diablo*. I have found out how to fix this, so that you can easily do a true wall-to-wall microjustification and proportional space in this mode.

One thing that is absolutely infuriating.

The toner cartridges are *NOT* the same as the stock *Canon* photocopier cartridges. In fact, through the use of special *Torx* "tamperproof" screws, missing notches, the brackets which are slightly different, etc., they went to an awful lot of trouble to make darn sure you would not casually interchange the two. Thus, printer cartridges will cost you more and will be harder to get, particularly in colors.

The toner cartridges also look like they will be more than a little bit tricky to refill on your own. Be sure and stay tuned.

UPDATE: Feb 87

The way to get started on this fantastic trip is with a copy of *Inside Laserwriter* from *Apple*. It gives you all the complete details on the *Postscript* language and has all you need to get in on the ground floor. Note that this $50 notebook must be ordered directly; it is normally not available through dealers.

The next thing, of course, is to get some hands-on experience. A very few of the more aware quick-copy centers are snarfing these up. So far, availibility is limited only to those copy centers that still intend to be in business six months from now. The terms and access vary, so be sure to check around. Several local

```
1 p
 psr0
 p
 qclsrd
 psz0
2 ppr
 pin  Starting Page Number ----------------> =$A
 ppn$A
 pin  Left Margin in tenths of an inch ----> =$A
 psx$A
 pin  Printed width in tenths of an inch --> =$A
 psy$A
 pin  Printable characters per line -------> =$A
 prm$A
 plm0
 pin  Print how many copies (0 = none) ? --> =$A
 p
 pin  High legibility mode (Y/N) ? --------> =$B
 ppr
 pin  Top line --> =$C
 ptl$C
 ppr
 pin  Is everything OK (Y/N/esc) ? --------> =$C
 ppr
 ppr
 pcs/Y/$C/
 pgo3
 pcs/y/$C/
 pgo3
 pcs/N/$C/
 pgo1
 pcs/n/$C/
 pgo1
 pqt
3 p
 psr0
 p  needs pm0 or pm6
 p
 ppr    fix centering
 p
 b
4 f<>.cj><>.lj[esc]U[esc]=[esc][Q][B][esc]O<
 y?
 pgo5
 pgo6
5 p
 u
 f<><[esc]D><
 y?
 p
 4
 h
 pgo4
6 p
 ppr    write transfer file
 p
 p
 p
 pnp
 zzz
 y
 p
 p
 pnd
 ny
 lzzz
 p
 plm0
 prm240
 ay
 psr0
                                        (( more . . . ))
```

Fig. 4A - WPL Diablo/Laserwriter auto formatter.

```
 ppr    set margins
 p
 b
 p
 f<<>[esc]M<
 y?
 p   [F] = Helvetica; @ = Courier
 b
 f<<[esc][Q][F]<
 y?
7 f// /
 y?
 p
 psx-1
 pgo7
 f//[esc]9/
 y?
 p
8 f// /
 y?
 p
 psy-1
 pgo8
 p
 f<<[esc]0[esc]Y<
 y?
 p
 p
 u
 u
 u
 u
 ppr    justify top line
 p
 p      must be customized!
 p
 f<>Lanc<>[esc]MLanc<a
 y?
 p
 p
 b
 p
 ppr    unjustify paragraph ends
 p
 e
9 f<.><
 p
 pgoa
 pgob
a p
 f<><>[esc]X<
 y?
 pgo9
b p
 e
c f!?#!
 p
 pgod
 pgoe
d p
 f<><>[esc]X<
 y?
 pgoc
e p
 p
 ppr    fix paragraph indents
 p
 b
 f<>     <>[esc]X     [esc]M<A
 p
 p
                                        (( more . . . ))
```

Fig. 4B - Auto formatter, continued . . .

outfits charge $12 per hour (including a Mac) and 50 cents per page. Another one is rumored to offer *real* 2600 DPI typesetting off of a Mac based Postscript disk for a buck a page.

Including free corrections.

How can I typeset my Own Computer Shopper Ad?

How about with the *Laserwriter* at your local quick copy center? It shouldn't cost over a dollar or two.

A sample ad is shown you in figure two. It was written completely with *Applewriter* on an *Apple IIe*. Including the graphics.

Postscript is a Forth-like language whose commands start with values and end with what to do with those values. Strings to appear on the page are placed in parenthesis. The X, or horizontal value always comes first, followed by the Y or vertical value. The full details appear in Adobe's *Postscript Cookbook* and the *Postscript Reference Manual*.

How Do I fix Laserwriter's Diablo Emulation Mode?

Like everyone else in the industry, those Laserwriter people are telling you "that which is not so" when they claim that their *Diablo 630* emulation either works at all or even approximates all the features available on a Diablo 630 daisywheel.

Needless to say, if you unplug the RS232 cable from your 630 and plug it into the Laserwriter, the chances are very good that nothing at all will happen.

But, unlike everyone else who lies about their Diablo compatibility, with a little chainsaw and sledgehammer work, the Laserwriter is able to beat the 630 at its own game.

Including, of course, a full wall-to-wall microjustification and proportional space using a fully professional typefont.

The beauty of the Diablo

emulation mode in that you simply feed in plain old text files and out come professionally "typeset" results. And better yet, you can add bells and whistles with a simple WPL post processing routine applied to your already written text files.

There is only one severe present restriction to the Diablo emulation mode in that everything does in fact have to be 12 point text, without any fancy graphics or different font sizes. But that is most of what most word processing is all about anyway.

You do have the ability to mix and match normal, bold, italic, shadowed, and underlined text in many different combinations. You can also stretch titles out any amount you like.

Let's outline the process of getting your Laserwriter's Diablo emulation mode working on an Apple IIe, particularly one running under the ProDOS Applewriter 2.0. We will be rather brief here to cram everything in. I have much more information available, some free, some not so free, so call or write if you need more details.

First, *Use a Super Serial Card*. This card supports the XON/XOFF handshaking that is needed by the Laserwriter. Use an uncrossed or "straight through" printer cable, and set your SSC switches to S1: 4,6,7 on; S2 1,4 on. Use the terminal mode with the arrow down.

Second, *enable the handshaking*. On any file that will be 5000 characters or longer, prefix that file with . . .

% [I] X E
% [I] Z

Third, *change the daisywheels*. To get both proportional spacing and wall to wall microjustification, you have to use a *Helvetica* or a *Times* daisywheel.

Surely nobody, but nobody, would still use klunky old fixed pitch *Courier* in this day and age, except for key portions of those very few technical listings that absolutely must be fixed pitch.

Figure three shows you the Postscript code needed to change daisywheels. I also have an WPL routine that does this automatically with full promtping. Run this in the Postscript 9600 mode before you move over to the Diablo mode. You can have two different "wheels" in use at any one time.

One very big gotcha: The wheel settings are stored in non-volatile RAM that can only be written to a total of 10,000 times over the life of the machine. Do not change the wheels continuously, or you may end up in deep trouble. Plan your work so you have an average of three wheel swaps or less per day.

Fourth, *bypass the microjustify bug*. The key test for Diablo compatibility is the magic "[esc]-M" command to do true fill microjustification. Without this command, you

```
  ppr    no justify beyond marker
  p
  p
  b
  f/!lsr2/
  p
  p
  f/[esc]M/ /a
  p
  p
  e
  f/!lsr2//
  y?
  p
  pcs/$B/Y/
  pgof
  pcs/$B/y/
  pgof
  pgoh
f p
  ppr    increasing legibility
  p
g f<><>[esc][Q][A]<a
  p
  p
  b
  f<>[esc][Q][A].<>.<a
  p
  p
h p
  qcwide
  p
  ay
  p
  p
  ppr
  psy1
  psz$A
  psz+1
  p
  psz-1
  pgoi
  pqt
i ppr
  psrl
  psz-1
  pgoj
  pqt
j ppr    saving margins
  b
  smargin![esc]Y!
  p
                                    (( more . . . ))
```

Fig. 4C - Auto formatter, continued . . .

just flat do not have Diablo compatibility. Wonder of all wonders, there actually is a more or less working [esc]-M feature on the Laserwriter.

But, sadly, there is a major and nasty bug in the Laserwriter's [esc]-M. A working microjustify routine should start at the first *printable* character in a line. Leading spaces should *not* be justified. You guessed it. Laserwriter microjustifies those leading spaces as well.

So what's the big deal?

For one thing, any paragraph indents not repaired by hand will end up variable width, since the indents will get squashed or unsquashed to suit the rest of the text line. Worst of all, any left margin set by your word processor will also get microjustified, leaving you with a ragged left margin.

There are several cures for this. One obvious solution is to individually mark each line at the point where you want the microjustification to start.

The second, and programmable, solution tells you to always use a fixed paragraph indent (say six spaces) and a *zero* left margin with your word processor. Then set the Laserwriter's left and right margins to handle the page centering.

To recap, set your word processor left margin to zero and set your right margin to the number of printed characters you want per justified line. Always use the tab key to enter the fixed paragraph indents. Let the Laserwriter set the actual left margin and the actual width of the printed line on the page.

There is nothing built into Laserwriter to automatically cancel the microjustification on the last paragraph line. Further, the microjustification apparently must be cancelled at the start of the last paragraph line, and not the end.

Fifth, *understand how the margins work*. Because of the microjustify bug, you have to use both the Diablo margins to do a full microjustify. Both margins are also needed to center proportionally spaced titles, especially if you are stretching or offset kerning them at the same time.

The only way I have found to set the margins that works with the proportional space fonts is to space out to where you want the left margin to be, and set the margin with an [esc]-9. Then you continue spacing out to the right margin and do an [esc]-0.

Which leads to three more nasty suprises. Firstoff, the space character varies with the font. To bring a space up to an exact tenth of an inch, add [F] kerning offset to Helvetica or [G] kerning offset to TimesRoman. This is done with an [esc] [Q] [F] or [esc] [Q] [G] as needed.

Next, what the Laserwriter calls a right margin is not a right margin at all, but is the *printable line width*. If you change the left margin, the right margin "tracks" with it, keeping a *constant width* of printable line area.

Finally, resetting the margins once they have been set is tricky. I have found no reasonable way to reset the left margin *left* of where you first set it, short of using the Laserwriter's 30 second job timeout. Unless I have missed something, this tells us *you must remove all the margin settings from your file for repeat copies of multi-page text files.*

You can use your word processor for temporary margin shifts, so long as you do not both move the left margin and microjustify at the same time. You can also hand patch each line that has to have a non-standard left margin.

Sixth, *post-process your textfile*. Figure four shows an automatic WPL routine that will take a more or less stock textfile and set it up for full professional quality Diablo proportional space and microjustification, sets both your margins for you, dejustifies all last paragraph lines, and automatically handles offset, stretched, and bold centered titles for you.

The routine also runs as many copies as you request, automatically taking care of margin hassles on all repeat multi-page copies.

```
  b      ppr      remove margins
  f/[esc]Y/
  p
  p
  u
  u
  d
  x
k psrl
  psz-1
  pgok
  p
  p      restoring margins
  b
  lmargin
  p
  pqt
l ppr printing copy #(y)
  psy+1
  pnp
  p
  prt
0 pnd
  ppr [L]
  ppr Don Lancaster's DIABLIFIER automatic Laserwriter formatter
  ppr ..........................................
  ppr
  prt
```

Fig. 4D - Auto formatter, concluded.

Don Lancaster's

ASK THE GURU

September, 1985

II+ reset hassles
IIc/IIe absolute reset
Laserwriter prefeeding
Aliasing contest winners
Commodore integrated circuits

And our big winner is Harold Melanson from Minneapolis, MN, who took first place honors in the great aliasing contest. Besides a *SAMS* book, Harold also wins that all expense paid (FOB Thatcher, AZ) tinaja quest for two, compliments of the *Gila Valley Apple Growers Association*.

Runner-ups included Steve Harris of Alhambra CA, Stephen Bach of Scottsville VA, Paul Santa-Maria of Monroe MI, and Douglas Roth from Huntsville, AL, who all have each received a *SAMS* book.

Many thanks to the many hundreds of correct entries that came in too late to win.

The correct solution? The listing in the great aliasing contest consists of one typographical error that had been cleverly buried deep inside a list of standard baud rates.

Several entrants questioned just how and why such a list would be used, reasoning that standard serial interface chips use special internal codings to set their baud rates.

Well, any time a program asks a user to set a baud rate by inputting human type numbers, a comparison must be made against the baud rate table. Should a "hit" be found on the table, the proper command is then sent to the serial interface chips as needed.

Among quite a few other programs, both *Applewriter* and *Appleworks* do require internal tables of this type.

On our current contest, I have yet to receive -ulp- even *one* entry. As you may recall from last month, the goal of the contest is to show me any way at all to get *Laserwriter* graphics out of a *Macintosh* that are even remotely as good, as convenient, as powerful, or as fast as the great *Laserwriter* graphics you can get out of *Applewriter* running on a IIe.

Speaking of which (isn't it utterly amazing how all these things just happen to come up?), my brand new five volume *Appplewriter/Laserwriter Utilities Package* is all ready to go. Its many features totally boggle the mind. Call or write for more info.

I have also arranged with *Synergetics* to stock the autographed copies of practically all of my books, *SAMS* and otherwise. This should ease greatly any hassles you have been having with your local bookshop or computer store. Write or call for a complete and free list.

And, as usual, this is your column to handle all your questions and your problems.

The absolute reset for the old Apple IIe will not work on a new IIe or a IIe upgraded to the new ROM set and should definitely not be used.

Instead, an "alike but different somehow" patch should be used. This patch goes in a different location, is slightly longer, and has a different checksum.

To build an absolute reset for the new IIe:

1. **Use SNATCHMON to copy the new IIe monitor.**
2. **BLOAD IIEMON.C, A$8000**
3. **CALL -151**
4. **82C8: EE F4 03 A0 1C A9 C5 20**
   A8 FC 2C 61 C0 10 0B 88
   D0 F3 4C 59 FF C0 C0 00
   00 00
5. **BSAVE KREBFMON.C NEW IIE, A$8000, L$1000**
6. **Burn the new CD EPROM. Use KREBFMON.C NEW IIE for the low 32K and IIEMON.D for the high 32K.**

If your burner can handle a whole 2764 at once, combine and move your files downward to a suitable buffer space in memory. For instance, do a BLOAD KREBFMON.C NEW IIE, A$2000 and then a BLOAD IIEMON.D, A$3000. This gives you an 8K buffer starting at $2000.

Note that you will plow DOS if you try using an 8K buffer that starts at $8000.

Note also that the self-test on the new IIe behaves differently on the new IIe than the old. Expect some page two garbage and an ocassional flash, followed by a "System OK" message" after a minute or two.

Fig. 1 - Absolute reset for the enhanced Apple IIe.

Just be sure to call or write per the end box. Best calling times are on weekdays 8-5, Mountain Standard Time.

On to the goodies . . .

How do You do a IIc or A New IIe Absolute Reset?

I have recently shown you how to do an absolute old IIe reset to the "old" monitor that eliminates the hole blasting and returns complete control back to you. You can find details in *Modern Electronics*, February and March of 1985, or in enhancement ten of my *Enhancing your Apple II*, volume II (SAMS #22425), or in my *Absolute IIe Reset* software package.

A listing of the SNATCHMON, one of the key pieces of the software involved, did appear a few columns back right here in the *Computer Shopper*.

What is involved here is a new EPROM chip swap. After burning and installing the new chip, a [control] [open-apple] [reset] does the usual cold reboot unless you hold down the [open-apple] key for an extra three seconds.

If you do that, you will automatically drop down into the original old monitor, just as if you could do a CALL -151 from any point in any program at any time for any reason. The monitor changes overwrite that obscene hole blaster code, while preserving the overall checksums.

With the preserved checksum, the patch is invisible and transparent to virtually all software.

At any rate, this particular patch works *only* on an "old" IIe. It will *not* work on a "new" IIe or on a IIc. So, here for the first time in print, are "alike but different somehow" patches that let you do an absolute reset for either the "new" IIe (figure one) or for the IIc (figure two).

Let's get one point clear right off the top.

The new IIe monitor firmware and that IIc monitor firmware are *NOT* identical. In fact, they are not even vaguely similar. Among other differences, the IIc has to provide for serial port settings and control. What happened in this "new" IIe "upgrade" was that some new monitor features were added to the IIe. These features are somewhat similar to some of those on the IIc.

Under no circumstances should you try using a IIc monitor code image in a IIe or vice versa. The complete listings of all three monitors can be seperately found in the *IIe Reference Manual* ("old" IIe #A2L2005) in the *Apple IIe Enhancement Programmer Guide* or in the *IIe Enhanced Owners Manual* ("new" IIe #A2L2072 and #A2L2073), or the *IIc Reference Manual* (IIc #A2L4030).

Regardless, that "new" IIe monitor patch is almost the same as the "old" IIe monitor patch. It is in a slightly different location, is just a tad longer, and has a different checksum. As with the "old" IIe, a tad of C006:00 black magic has to be invoked following an absolute reset. Only the "CD" 2764 EPROM needs changed.

The IIc has all of its monitor placed in a single and electrically larger type 27128 EPROM. The patch will not quite fit in one piece, so three individual patches have to be made. Note that these patches are made to the "F" area of the ROM on the IIc, compared to the "C" area patches used for the old or new IIe. With the IIc, that C006:00 black magic is not needed after an absolute reset, although doing so will not hurt anything.

It seemed necessary to steal four bytes from the bootup title, so the IIc absolute reset now prompts "Hello", rather than "Apple IIc". Sorry about that. Let me know if you find a better way.

As before, I've found that *E-Tech Services* does a great job of burning EPROM's at very reasonable costs.

And, time once again for the usual reminder: A legal EPROM burning service can burn and ship you an EPROM only after *you* send *them* the exact image of the EPROM you want burned.

The code to be burned *must* originate with you, even if the service gets a thousand identical orders per day.

Rumor has it that there are some top secret and "magic" EPROM's available directly from Apple that include an absolute reset, total memory management access, and do bunches of other analysis and debugging goodies. I have no further information right now as to when, or where, or to whom, or for how much these will be made available.

Chances are they will cost much more than doing your own burn for a simple absolute reset.

What About Resetting The Old Apple II+?

It amazes me how many people are still desperately clinging onto their older II+ machines. Yes, there are a lot of them still in use. And yes, there are bunches of people still trying to use them far beyond their original capabilities.

But the simple fact is that most genuinely useful current Apple software will not run at all on a II+, or at least not without major compromises. And future software releases are even less likley to work on a II+.

So what it boils down to is this: A II+ user forever limits his options to older and less useful software. This dearly will cost the II+ owner time, money, power and convenience. More to the point, it is letting his competition gobble him gone.

Put another way, the per-hour operating cost of a II+ is substantially higher than that of a IIe. Ridiculously so. And that loss in time and money, power and convenience can easily pay for your upgrade

costs many times over.

Anyway, I sure get lots of II+ calls every day.

One person called me and told me he installed a IIe monitor in his II+ but that it didn't seem to work for some reason. I tried explaining to him that there was 6K of monitor code in a IIe and only 2K in a II+, and gently tried to ask him where he physically put the extra 4K.

Somehow it just did not compute.

Other callers are trying to cram *Applewriter IIe* down into a II+, minimizing those major problems with all the cursor keys, all the memory management, lower case, 80 columns, the live screen, the auxiliary memory needs, the relocated and semi-custom DOS, and the totally different screen imaging and type-ahead buffering.

Even if it all fell together, the only possible result would be a poor to utterly awful rendering of an absolutely outstanding word processing program. And even Applewriter IIe itself now has been completely and totally eclipsed by the newer ProDOS Applewriter 2.0 and 2.1.

But the majority of all II+ calls are still asking for an absolute reset. So, at the risk of encouraging the continuing loss of time, money, power, and convenience, let's look at what is involved in doing a II+ absolute reset.

The original Apple II had an "old" absolute reset 2316 ROM in it. This was called the "F8" ROM and resided in memory from $F800 on up to $FFFF. When you pressed the reset button, you absolutely and unconditionally went into the monitor. No ifs, ands, or buts, and no way for a program to stand between you and absolute control.

This was replaced with an "improved" or *autostart* ROM in the II+. While the autostart ROM would automatically boot a disk for you on a cold start, all programs could now grab reset vectors and keep you from ever forcing an unconditional reset.

Some other goodies, most notably a mini-assembler, the single step, trace, debug, and the unique *sweet sixteen* 16-bit emulation software code were also dropped during this "upgrade" to Applesloth. But the mini-assembler and the sweet-16 were really part of the Integer BASIC ROM area from $F000-F700. Strictly speaking, these are not part of the monitor image itself.

You can still find copies of this ROM in original Apple II's. While they used to be available in these *Computer Shopper* ads for $10, they are now extremely rare. These are also theoretically available from an Apple dealer, but only at horrendous prices.

The absolute reset for either Apple IIe will not work on a IIc and should not be used. Instead, a three-piece patch that overwrites part of the "F" ROM area should be used. Note that the entire monitor gets written into a single 250 nanosecond 27128 EPROM.

To build an absolute reset for the new IIc:

1. **Use SNATCHMON IIC to copy the IIc monitor.**
2. **BLOAD IICMON.F, A$8000**
3. **CALL -151**
4. **8B64: 05**
5. **8CCA: EE F4 03 A0 1C A9 C5 20**
   A8 FC 2C 61 C0 10 05 88
   D0 F3 F0 2A
6. **8D03: C8 E5 EC EC EF 4C 59 FF 91**
7. **BSAVE KREBFMON.F IIC, A$8000, L$1000**
8. **Burn the new CDEF EPROM.**

If your burner can handle a whole 27128 at once, combine and move your files downward to a suitable buffer space in memory. For instance, do a BLOAD IICMON.C, A$2000, a BLOAD IICMON.D, A$3000, a BLOAD IICMON.E, A$4000, and, finally, a BLOAD KREBFMON.F IIC, A$5000. This gives you a 16K buffer starting at $2000.

Free plans for a simple 27128 adaptor for older burners are available on request and are included in the reset package software.

The new booting prompt is Hello. Note that the IIc self-test never ends. The C006: 00 black magic is also not needed on the IIc.

To install the chip, unplug IIc power and remove the six outermost screws from the bottom. Press into the front crack directly in front of the "N" key with a 1" dull putty knife to release the front snap. The monitor is the chip directly under the keyboard center at D-18, slightly to the left of the speaker. Make sure the dot and notch go to the left.

Fig. 2 - Absolute reset for the original Apple IIc.

There used to be a peripheral known as a *language card*. This was a plug-in card that fit slot zero and held six 2316 ROMs. If Integer BASIC and that old absolute reset ROM went into the card, it was called an *Integer card*. If instead, the card held the autostart ROM and Applesloth, then it was called an *Applesoft card*. These cards had a magic red switch that let you flip between operating systems, making your Apple a II or a II+ per your choice.

If you can still find a language card with an original Integer BASIC ROM set in it, this gives you a second route to an absolute II+ reset.

But note one detail. The red switch does not immediately switch you between monitors. To do an absolute reset, you have to flip the red switch and *then* reset. Note that a CALL-151 from Applesloth drops you down into the autostart ROM, while a CALL -151 from Integer BASIC gets you into the old absolute reset ROM. This happens regardless of where the ROMs are sitting.

These language cards got replaced by plug-in RAM cards, that let you run other languages, that include Pilot, Fortran, Pascal, etc. At this time, images of both BASIC versions were placed on the system master disk. On a cold start with a RAM card, whatever language that was *not* in the machine got loaded into the card. Thus, on a II, you ended up with Applesloth in RAM. On a II+, you instead got Integer BASIC in RAM.

Unfortunately, when they went to the disk, the INTBAS code image was changed to have that autostart monitor code, thus eliminating any absolute reset access. You also lost the single step and the trace debugging. You do retain the sweet 16 and the mini-assembler with the new INTBAS disk image, as these are part of the Integer "F0" ROM from $F000-F7FF.

A third route to an absolute reset is to customize either INTBAS or FPBAS so they hold the old absolute reset image. Some software packages can defeat the use of a RAM card for monitor access, though.

In fact, many of them do.

The kicker is that the 2316 ROM is *NOT* directly interchangeable with a standard 2716 EPROM, since there are problems involving three pins that must be gotten around.

The clones and Hong Kong knockoffs beat this compatibility problem by designing for 2716 EPROMs in the first place. Thus, on a clone or a knockoff, all you usually have to do is program your own EPROM, picking up any type of monitor you like any way you like.

Back to the 2316. A 2316 has three programmable chip select lines. Apple chose to make two of these active low (pins 20 and 21) and one of them active high (pin 18). This was done as a factory option at the time the mask was created for the 2316.

On a 2716 EPROM pin 20 is also active low, so there is no problem on this pin. Pin 21 should be held at +5 at all times to prevent programming. Note that pins 20 and 21 are shorted together on the Apple motherboard. Finally, pin 18 is made active low, or the exact opposite of what you really want.

Several plans have been published that involve only some jumpering of some 2716 EPROM pins. But this ignores the INH inhibit line from the expansion slots. If some slot based card tries to make use of that high ROM address

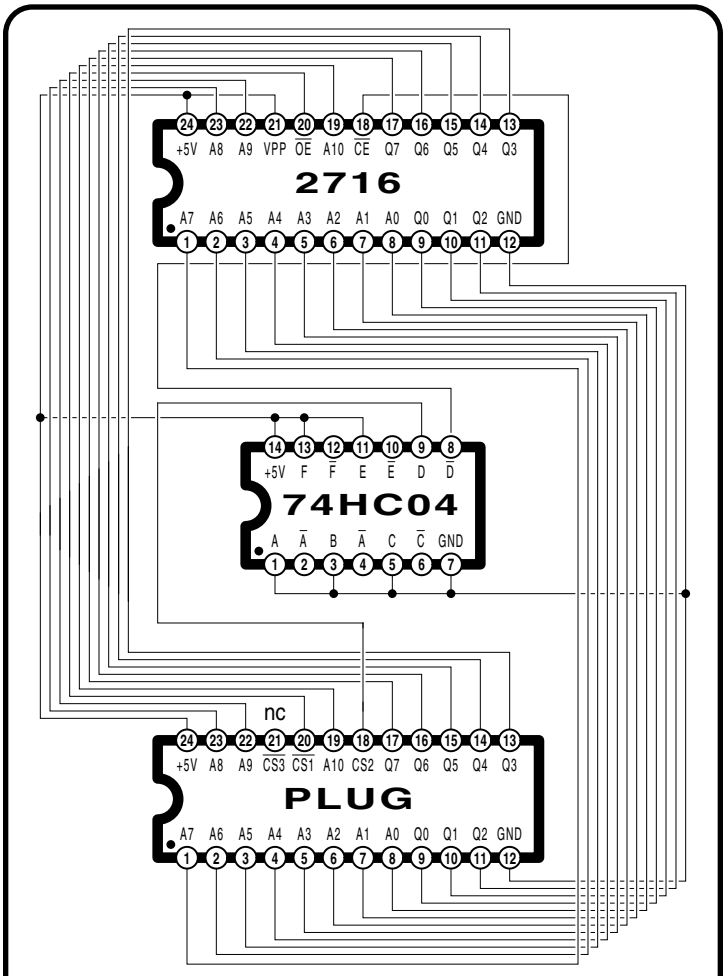


Fig. 3 - EPROM Monitor adaptor for the Apple II+.

space for its own use, then you get memory contention, a fight, and possible damage. And since just about every "modern" II+ has at least one RAM card in it, such contention is inevitable.

Anyway, to make a long story short, to do an absolute reset on a II+, you have to add an adaptor socket that fits between your you-program-it 2716 EPROM and the motherboard. This adaptor socket has to be an "active" one that has a single inverter built in.

The lead to pin 18 goes through the inverter. Pin 20 is held at +5 volts. All other pins do go "straight through". The inverter needs +5 volts on pin 14 and ground on pin 7 and on all unused inputs.

Figure three shows details.

You can use a pair of 24 pin *machined contact* DIP sockets, the premium kind with the smooth pins that can be safely plugged into another socket. These can be seperated by a pair of dip strips, leaving room for the 74HC04 inverter between the two sockets. The 74HC04 can have its pins bent up over its own top and can be directly soldered in place.

The adaptor normally is only needed for the F8 ROM socket.

You then can plug a 2716 EPROM into the adaptor, creating any monitor functions you like any way you want to.

Should you run into card clearance problems, consider adding a short piece of 24 conductor flat cable as an extension socket. Keep the cable as short as possible if you do this.

If you really want to get fancy, you could use a 2732 EPROM instead, and a switch to flip between a stock and a new custom monitor. Put one monitor on the bottom half of the 2732 and the other on the top half; then switch the most significant address line.

This would be similar to the dual character generator we looked at a few columns back that eases the unmitigated disaster of the "new" IIe lateral arabesque.

Where Can I Get Oddball Commodore Integrated Circuits?

While I have not personally checked them out, *Boufal Services* lists a wide variety of Commodore chips and stuff, including the rare and hard to get 6560 and 6567 VIC chips and the 6581 SID integrated circuits.

Wonder of wonders, they also have KIM-1's at very attractive prices. The KIM-1 was far and away the greatest microcomputer Commodore ever built. Since then, it has been downhill all the way.

To this day, there is no better way to learn all of the fundamentals of machine language programming than on a KIM-1. The KIM-1 is also an excellent choice for a dedicated micro for a solar panel, a cattle feeder, a weighing station, a pump monitor, or whatever else you dream up.

Boufal Services also carries most of the manuals, including the obscure and harder to find ones.

Check them out.

How can I do a Laserwriter Manual Feed?

The *manual feed* feature lets you hand feed one sheet at a time instead of using the paper tray. Manual feed is essential for envelopes, labels or for very stiff paper stock.

To activate this manual feed, from Postscript you do a *"statusdict /manualfeed true put"* command.

To shut down your manual feed, either wait for the 30 second automatic job timeout, or else do a *"statusdict /manualfeed false put"*.

By the way, I haven't had much luck with Laserwriter envelope printing quality. The results have been very poor to date. Please let me know if yuou come up with a solution or two on this.

(Continued from page 5.4)

the correct starting point to enter the code.

Sometimes, a short file will purposely be *embedded* within legal opcodes. One text printing scheme, called the *embedded file method* does this. Others, notably in *Appleworks* and in all the *Adam's Adventures*, will pass parameters that are needed by a subroutine immediately following that subroutine call. The subroutine messes with the stack so that, after grabbing all of its parameters, it returns to the valid code just *beyond* the passed values.

In figure five, we have a very common instance of an embedded file pair that causes aliasing. This aliasing is often caused by those ProDOS MLI machine language interface calls. Typically, access of a disk command involves a JSR $BF00 followed by a three byte file. The first byte gives the command. The second byte pair points to a longer file in which more information will appear.

Usually, this second information file will follow the MLI call. Most MLI calls will end with a branch to an error processor followed by an RTS. The correct and aliased listings are both shown.

Disassembly aliasing can be both a confusing nusiance and an essential tool at the same time.

Let's have a contest. Figure six is a mystery aliased disassembly that could only be one possible thing. I'll send a free SAMS book to the first five *Computer Shopper* readers that tell me exactly what that thing is.

Yes, its a toughie. But you will know for sure when you have it. By the way, the same code is likely to appear in *any* machine, so it is pretty much microprocessor independent.

Much more information on disassembly techniques does appear in the *tearing method* of *Enhance I* and also in my *Assembly Cookbook for the Apple II and IIe*.

Don Lancaster's

ASK THE GURU

October, 1985

The cubic splines
Using Bezier curves
Applewriter boot tricks
Postscript curve tracing
Editing BASIC programs

As you know, we have this contest going. A free *SAMS* book to anyone who can show me any way at all to get graphics out of a *Mac* that even remotely approaches the quality, speed, and convenience of the *Laserwriter* graphics you can get out of *Applewriter* running on an *Apple IIe*.

As the dearth of entries to date shows, there is no catch. I am still waiting.

I did award one book so far, but only because I felt sorry for the entrant. It was just like taking candy from a baby.

He was a knowledgeable *MacPaint* power user and had spent nearly four hours working up an image of a covered wagon. Admittedly, he did have some hecklers, and there was some divine intervention in the form of – believe it or not – a conveyor belt fire.

Anyway, to make a long story short, it took 20 minutes to redraw the wagon in *Applewriter*, and there was an immediate and obvious 29.3 decibel improvement in print image quality. Take a peek ahead to figure five to see the results. With those multiple wheel spokes on odd angles, the fine panel work, and the sweeping and graceful billowing cover curves, it was not even close.

To find what this is all about and to get in on the fun, write or call me per the end box for your free demo pack of *Applewriter/Laserwriter* images that I have put together for you.

A reminder that I have arranged with *Synergetics* to now stock autographed copies of practically all of my books, *SAMS* and otherwise. This should ease greatly any hassles you have been having with your local bookshop or computer store. Send for a complete and free list.

My latest new book is the *Applewriter Cookbook*. I do have some autographed copies on hand for you here.

I'm told there are still a few copies of *Enhancing your Apple II & IIe* left. By the way, this dude is also for the *IIc*. It's just that you don't like to change titles right in the middle of a multi-volume series.

And, as usual, this is your column that handles all your questions and your problems. Just call or write per the end box. And now . . .

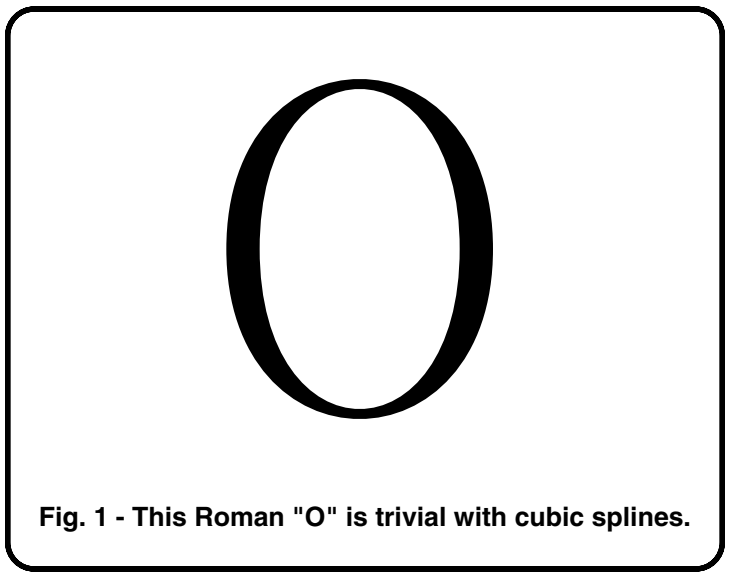


Fig. 1 - This Roman "O" is trivial with cubic splines.

Can I Edit BASIC Programs by Using A Word Processor?

As some of you found out the hard way, you cannot do this by using that ProDOS TYPE command. Instead, a simple and quick process exists that lets you do all of your BASIC editing with your favorite word processor.

Applewriter, of course, is a top choice, since you can enter all BASIC commands directly from the glossary, and since you can handle macro sequences by using the WPL supervisor.

That same process will convert a BASIC program to a textfile in either Dos 3.3e or ProDOS. Add a small code module into your program, either at the end or the beginning, per your programming style and the utilities you are using. This module should name a textfile and open that textfile, list all of the lines of the BASIC program *except the module lines*, and then close the file. The module can then optionally delete itself.

One example routine does appear in the Dos 3.3e manuals. All you do is open a text file, write the original BASIC listing to the file, and then close the textfile.

Then, you load the textfile to any word processor that can accept standard textfiles. You will now have free-form entry, full program editing, unbroken comments, single-key entry of all the BASIC key words, powerful copy and append routines, search and replace, and bunches of other neat features that make things fast, fun, and simple.

You do, of course, still have to obey all the BASIC listing rules, terminating each line with a carrage return, and so on.

Getting back to BASIC is easy enough. You just add a NEW command before the first line of your text file listing. Get into BASIC with DOS up and running, and EXEC your file. The EXEC process will then "type" the textfile into your machine as a BASIC listing, just as if you keyed it from the keyboard.

Extra spaces and such used for pretty printing will get ignored. When you do want printed listings, just use the textfile instead, since you can easily edit it further with page breaks, boxes and lines, more pretty printing, or whatever.

Editing a BASIC program is so much faster, so much fun, and so less error prone when you use a word processor that you will never go back to the "old" way.

What are Cubic Splines?

Cubic Splines are a major breakthrough in the computer graphics field that let you create smooth and accurate freehand or freeform curves as easily as you would draw a plain old box.

Cubic Splines are also a superb way to let you handle typographic fonts, since the fonts can be stored in an extremely compact form.

More importantly, the fonts can be easily scaled to *any* size and *any* shape desired, with both the resolution and smoothness *increasing* as the font size increases. Compare this to scaling a bit-mapped font that gets "chunkier" as the size increases.

The *Laserwriter* uses cubic splines for many of its font characters and for its *curveto* and *rcurveto* operators. This ability places the *Laserwriter* head and shoulders above its "one size font in an outrageously costly cartridge" shoddy imitators.

Although easily accessable from *Applewriter* on a *IIe*, the cubic splines are conspicuously absent from all of the *Mac* Quickdraw routines.

I'll try to give you some fundamental background here and show you where to go for enough info that you can, with lots of personal effort, add cubic spline ability to most any graphics program on most any microcomputer of your choice.

A plain old *spline* is just a plastic coated lead ruler that the drafting, engineering, or architectural people use to draw a curve that won't "fit" the usual compass or French Curve contours. First, you carefully bend the spline to the desired shape. Then you use this shape to draw or ink your line.

Cubic splines use the same general idea. By picking the right data points, you can force a line to go from its start to its finish by way of a route that is both smooth and controllable. Change the data points, and the shape of the curve changes.

These are known as cubic splines because all of the hairy math behind them involve a pair of polynomials of order three that use a constant term, a linear term, a square term, and a cubic term. Should the curve end up too complicated for a single cubic spline to handle, you use as many splines as you need, connected end to end, to get the job done.

Figure one shows us a good example. A mere four cubic splines having a total of only *sixteen* data points are needed to draw this Roman "O", *regardless of the font size*.

By way of comparison, the figure one original measured roughly two inches by three inches. At 300 dots to the inch resolution, a bit map of 540,000 pixels would be required instead!

The first spline covers the top half of the outside edge. The second does the bottom half, while the third does the top inside edge. The fourth handles the bottom inside edge. A fill routine then uses an "even-odd" rule to blacken the inside of the letter.

You can approach cubic splines from two different ways. Chances are you will like the *intuitive* or "try it and see" route, instead of the *analytical* or "mess with all that hairy math" method. The cubic splines lend themselves beautifully to experimenting and playing with them.

Anyway, figure two shows us what is involved in using a cubic spline, once the microcomputer or printer has been internally set up or programmed to handle them. The object of the game is to build a curve from a point marked

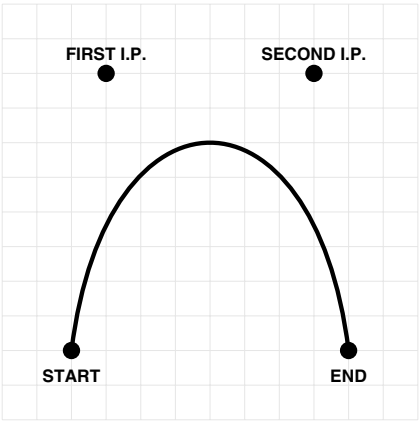


Fig. 2 - Cubic Splines use "influence points".

START over to a point named END. To get from START to END, you go by way of two *influence points*. The positioning of the influence points set the shape of the curve you will get.

The direction you leave from START is decided by the first influence point. The direction you enter END is set by the second influence point. The distance from START to the first influence point, or the distance from the second influence point to END sets the "enthuasiasm" at which the curve will move in the intended direction.

Once you have left START, the smoothest possible curve that can be drawn with a pair of X-Y cubic polynomials can then be drawn. The first cubic polynomial handles the horizontal motion. The second takes care of the vertical.

The direction you head in is known as the *bias* by the computer graphics people, while the enthuasiasm is called the *tension*.

The oldest of cubic spline techniques are called *Bezier Curves*. The newer and more powerful versions go by the seperate names of *B-Splines* and *Beta-Splines*. There are also *Conic Splines* available, among lots of others. Many of these fancier splines are more powerful and more flexible, but they are not as easily or as quickly done on a smaller microcomputer. Some are three dimensional.

One way to get different curves is to move the influence points around and see what happens. Let's see if we can't second guess what we can or cannot do with a single cubic spline.

Firstoff, rotate figure two around 360 degrees. Obviously, the same shape curve can be generated in any direction, so long as the relative position of the influence points with regard to START and END does not change.

Now, let us get back to figure two right side up. If both influence points are above START and END, then the curve will remain above START and END. If both influence points are below START and END, then the curve will stay below START and END.

If both influence points are between START and END, then you will get nothing but a straight line. If one or both influence points are on extensions of the straight line between START and END, then you will get a straight line that "overshoots" either at START, at END, or at both of them.

If one influence point is above and one is below, then you will get a somewhat sine looking curve that crosses the main axis between START and END.

You get symmetrical curves if the influence points are related to START and END in the same or mirrored ways. You get asymmetrical curves if one influence point has different bias or tension than the other.

Now for the neat part. If the first influence point is to the far right of the second influence point, then the new curve may cross, creating a sharp *cusp* or a *loop*. The size and sharpness depends on how far away the influence points are from the START and END points, as well as how far apart they are from each other.

You can get an "open" loop if only one of the two influence points are far left or far right of normal.

Thus, it seems that you can use one cubic spline to do a smooth symmetric or asymmetric curve, a curve with one change in its curvature direction, a curve with one cusp, or a curve with a loop in it. Anything fancier can be built up with repeated cubic splines.

Sometimes one of those influence points gets placed directly over START or END. This can be used to either sharpen a corner or smooth out a result, depending on the need.

Reviewing, to do a cubic spline on a microcomputer or printer set up to handle them, set a START point, an END point, and the two influence

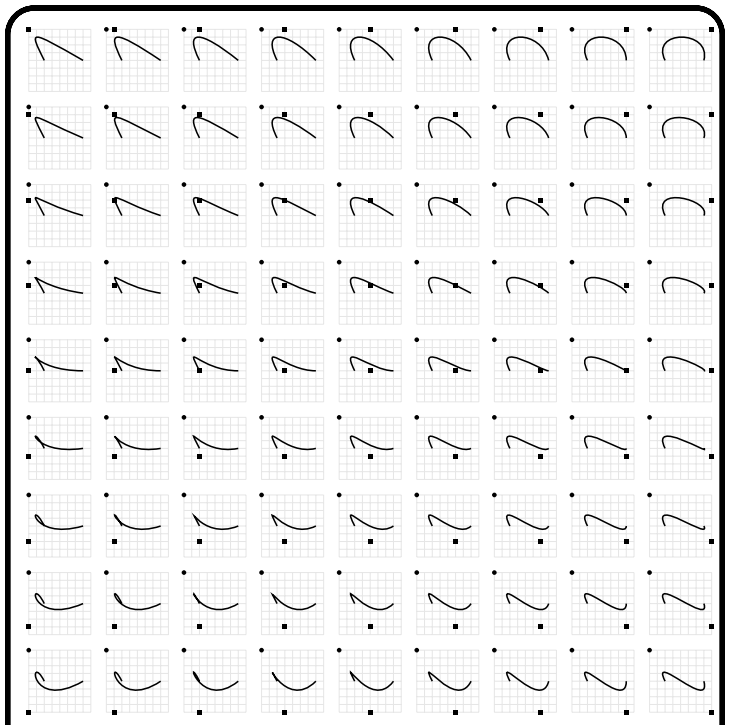


Fig. 3 - Sample page from a cubic spline catalog.

points. Then tell the software or firmware to have at it. Once again, the first influence point sets the direction or bias you leave START, while its distance sets up the enthuasiasm or tension the curve will head in that direction. The second influence point will behave in a mirror manner with END.

Note that the curve usually will not pass through either influence point. More often than not, it will miss these points by bunches on the "inside", since the smoothest possible curve is being drawn. Many curves will stay inside a "fence" drawn between the four points.

Here's how simple it is to draw the curve of figure two on a *Laserwriter*:

```
2 2 moveto
3 10 9 10 10 2 curveto
stroke
```

This says to move two blocks in and two blocks up to set START. Then define the first influence point at three blocks in and ten blocks up. Then define the second influence point at nine blocks in and ten blocks up. Next, set the END point at ten blocks in and two blocks up. Finally, activate the *curveto* operator to draw the curve.

You will also, of course, have to scale and translate the curve to where you want it, as well as setting your line width and shade of gray.

It turns out that the curve of figure two is actually easier to draw than a plain box!

Figure three is a page out of a spline catalog I worked up. Not too shabby for old *Applewriter* on a *IIe* eh what? At any rate, these cubic spline curves may seem a little dull, because the catalog is based only on a 9 x 9 integer grid. Even at this grid size, there are over 6500 splines in the 81 page catalog! Many of these are rotations or mirror images of each other.

START is always set two blocks right and four blocks up. END is always six blocks right and four blocks up. The first influence point shows as round dot. The second influence point is a square. In the upper left figure, the dot and square are sitting on top of each other.

Should you widen out the influence points, the curves will gain in impact, grace, and all around impressiveness. As it is, the catalog does show all the possible shape families you can get with a single spline. Note that even on one catalog page, you have concave and convex shapes; shapes both symmetrical and asymmetrical; shapes with cusps, loops, open loops; and shapes with or without any changes in the direction of curvature.

There are a near infinite possible number of splines that you can draw between START and END. Those influence points do *not* have to be integer values.

Figure four shows us a totally different and totally mind-blowing use for splines. Fancy borders like this are utterly trivial to handle with cubic splines. Only *five* different splines are needed! These are the main side loop from the crossover point, the inner arc, the diagonal end loop, and the two transition arcs connecting the corner loops to the edge loops.

To really get fancy, you can use a double border like this with a slight size difference. This will make the lines fatter in some places and thinner in others, just like "real" engraving.

I have a free *Postscript* listing of figure four available for you. Just ask for a copy.

This particular border takes around twenty seconds to create using the *Laserwriter's* internal computer. All of the cubic splines do need lots of processing time. While ideal for laser printing or creating final bitmaps of fixed visual images, splines are not well suited for real time animation, unless you have a humongous computer available.

Figure five shows us the covered wagon. Eleven cubic splines are needed. Do you see where each cubic spline starts and ends? Note particularly that the smoothness of the curve remains, no matter how much you magnify it, and no matter how much you reduce it, unless you get it down to a really tiny size.

This image was part of an award that used the figure four border, the wagon, and a few words. Something magic happens to the *Laserwriter* toner when it is heat fused to parchment-like calligraphic paper. The toner gets super black and literally leaps out at you.

Also rather neat is doing customized certificates and awards at a production cost of around thirty cents each, most of which goes for the fancy paper.

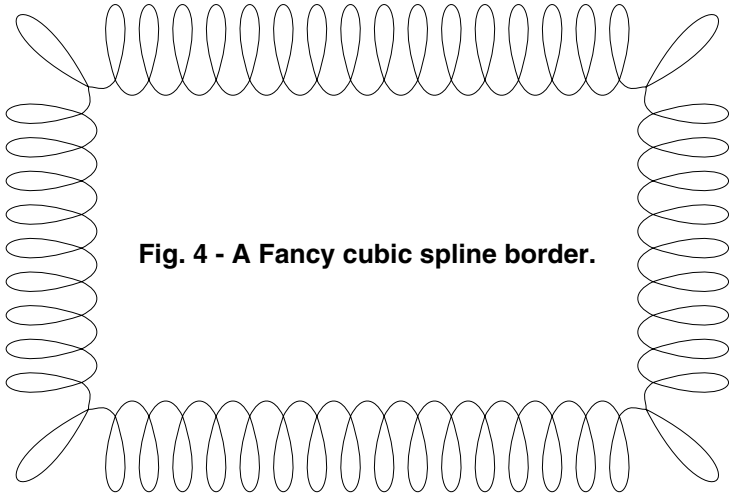


Fig. 4 - A Fancy cubic spline border.

So, how can *you* tap these cubic splines for your own profit and enjoyment?

You can do this with or without a *Laserwriter*. One route is to get yourself a copy of *Inside Laserwriter*, study it and then log some hands-on time on a rental *Laserwriter* at your nearby copy center.

Otherwise, it is off to the nearest full technical library. Bezier's original papers, for some strange reason, are all in French. He is translated in *Numeric Control – Mathematics and Applications*, by A. Forrest and A. Pankhurst, and published by *Wiley* in 1972. This seems to be out of print, so check a large technical library or a microfilm service.

The best and the most consistent source I've found for information on the cubic splines, along with many excellent bibliographies, are the various SIGGRAPH proceedings, available from the *Association for Computer Machinery*. These usually do appear in the number three issue of each year's *ACM Computer Graphics* quarterly.

Those 1983 and 1984 proceedings are especially useful. I have yet to see the 1985 SIGGRAPH paper listings. The advance promotion does promise at least one new conic spline paper.

An obvious money-making idea that involves splines would be to apply them directly to Apple or any other HIRES graphics routines, so that free-form curves can be handled as gracefully as ordinary boxes can. They are also sorely needed by the Mac. What would be really neat is to be able to real-time move the influence points.

Can I Run Applewriter From BASICS.SYSTEM?

Doesn't seem to work very well, does it?

You might like to boot ProDOS Applewriter from BASIC if you are adding a hard disk system, setting a modem, are writing a combination program that can include both Applesloth and WPL routines, or if you are integrating things into a large RAM card. I needed this two language ability for my brand new two volume *Applewriter/ Laserwriter Utilities*.

The solution is very simple, once you spend the hours needed to ferret out the real culprit. The BASICS.SYSTEM does not automatically set prefixes for you. Before you do the –AW.SYSTEM, just do a PREFIX, D1, and you are home free.

There is a second more subtle and rather stickier problem, though. BASICS.SYSTEM may want to use a program that is named as STARTUP to start off your application. And ProDOS Applewriter may want to use a wildly different program named STARTUP to run a WPL supervisor, or just to load glossaries and so on.

How can two wildly different programs share the same filename and not get mixed up? They cannot.

So, what I did was change a single byte in BASICS.SYSTEM so that the name of the new startup program is now STARTUX. I then named the modified routine as BASIX.SYSTEM.

The key information needed to do all this appears in the supplement to Quality Software's *Beneath Apple ProDOS*. You first BLOAD BASICS.SYSTEM, A$2000, TSYS, D1. Then you find the file that stashes the name for the startup disk. On ProDOS V1.0.1 or V1.0.2, the needed STARTUP file will begin at $21E6 and is coded as 21E5: 53 54 41 52 54 55 50. Change the $50 at $21EB to $58.

Then save to a new diskette by doing the BSAVE BASIX.SYSTEM, A$2000, E$47FF, TSYS, D2.

You can thus end up with two programs on your disk. STARTUX is one that BASICS.SYSTEM will boot, while plain old STARTUP is the one that Applewriter will use.

Since they have different names, there are no mixups possible.

My STARTUX on the new utilities package boots up in Applesloth and gives you a choice of running your Applewriter, of doing the HIRES bitmap conversion using a mixed Applesloth and Assembly routine, or of exiting to Applesloth.

The Applewriter STARTUP loads the PGLOSS Postscript single key glossary for the Laserwriter, sets the prefix to drive two, and boots a WPL supervisor that gives you a menu of all the utilities. From this menu, you select any of the mind-blowing routines that are available.

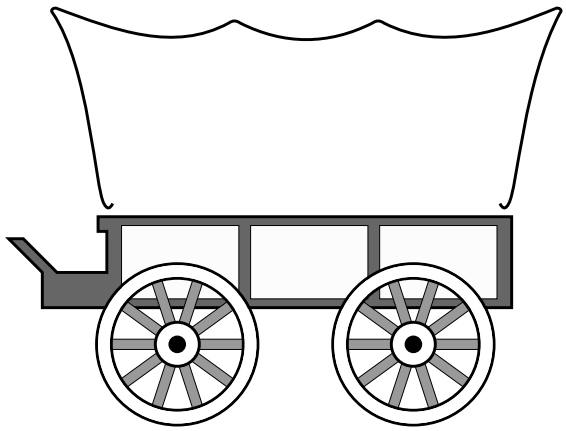


Fig. 5 - Covered wagon built with cubic splines .

Don Lancaster's

ASK THE GURU

November, 1985

Vaporlock bugs
Dual IIe monitor
Isometric drawing
ProDOS disk space
Customer complaints

For all those of you that just can't survive on a single fix of all these goodies per month, please note that I have a *Hardware Hacker* sister column that appears in *Modern Electronics* magazine. While there is some overlap between the two columns, this one over here in *Computer Shopper* leans more towards software and Apples, while the *Modern Electronics* one has much more hardware stuff in it, which involves both computers and older stand-alone electronics.

As usual, we have some freebies available for your when you call or write. First, a no-charge *Synergetics* technical helpline. The answers are free, but the call is your dime or whatever. There's also a free book and product list, bunches of free patches for *Applewriter*, and a new laser printing demo package that makes *Mac* graphics look worse than a dull crayon on used notebook paper.

On to whatever it is we are supposed to be doing here. If you ever figure it out, please be sure to let me know . . .

How Do I Get More Room On a PRODOS Diskette?

I always try to cram all of my for-sale diskette products as full as possible. To me it seems kind of dumb not to use all of the available disk space when you are selling something. An extra demo or two and a few more tutorials never hurt anything.

What can you do when you need just a "little" more room on a supposedly full ProDOS diskette?

First, if you have lots and lots of files on your disk, the chances are that one or more older versions or any other unnecessary stuff might be lurking somewhere in the middle of the list. You will obviously want to flush these.

Second, if you have several disks, see if rearranging the contents cannot more nearly fill up each individual disk. Dropping the ProDOS operating system, the formatter, and BASIC.SYSTEM from the successive disks in a multi-disk package can also help, if this does not overly inconvenience your users.

Third, if you have lots of disk files, you may be getting a DIRECTORY FULL error message, instead of the expected DISK FULL message. The magic number is 51 files. If the files are all short, you could still have lots of usable sectors left.

The cure here is to logically group the files and create one or more subdirectories. Each new subdirectory adds up to 51 additional files.

Fourth, note that a file or program of 511 or less characters requires only a single ProDOS sector. A file of 512 characters needs *three* sectors. This "extra" sector is needed to hold the track and sector listing for subsequent blocks. Thus, a few characters above 511 can cost you dearly on overall diskette space.

The trick is to go through the directory and note each entry that needs three sectors. Then find the ones that have less than 600 characters in them. Finally, see if you cannot lightly edit the file to get it down to that magic 511 character length.

Fifth, any file of any length that has only several more characters than a multiple of 512 would also be a prime candidate for light editing or compaction.

Finally, see if there are not any other compaction tricks you might use. For instance, you can find lots of different ways to reduce HIRES pictures down to much smaller sizes than you you'd expect. One very simple trick to pick up one free sector per save is to do a BSAVE HIRESPIX, A$2000, E$3FF7, instead of using E$3FFF.

The last eight bytes of a HIRES screen are invisible and do not appear on the screen. Yet, they cost you an entire extra sector per picture.

There are quite a few other

2 - 28 pin machined contact dip sockets

2 - 11 pin machined contact dip strips

2 - 13 pin machined contact dip strips

6 - machined contact dip pins

2 - miniature SPDT slide switches

Misc: 6 inches of #24 green insulated wire;
6 inches of #24 red insulated wire;
epoxy or superglue; solder.

Fig. 1 - Parts list for a IIe dual monitor adaptor.

graphics and textfile compaction schemes available, once you tune yourself into them. So, send me your favorites so we can share them.

My Customers Are Bitching That My Software Won't Run Properly. HELP!

Naturally, there is no possible way that you could stay competitive if you actually do spend all the time and effort needed to get your software working properly before you shipped it. Besides, software buyers are expected to pay for the programmer's mistakes, education, and their overall klutziness.

It's the American way.

Worse yet, any software that actually works is suspect of being obsolete, and certainly could not possibly be state of the art. This can very seriously damage your credibility. It would be almost as stupid as letting one of your programmers – horror of ultimate horrors – actually *speak* to the people in your advertising department.

The first thing you want to do is seperate your customer service phone from the order phone. Select a suitable Sub-Saharan African nation for your customer service departmant. Then, make sure that the employees can only speak Swahili using a very heavy Icelandic accent. Above all, be certain that your receptionist has latent tendencies towards cannabilism.

But the quickest and simplest cure for most unhappy customers is to print up some small and bright labels that say "BETA RELEASE", and then attach them to all your disks before you ship. Beta release software is not supposed to work and all your customers will be overjoyed in participating in the leading edge of software development technology.

Can I Put Two Different Monitors In an Apple IIe?

By now, it's more than painfully obvious that the "enhanced" IIe monitor is pretty much useless when it comes to running older Apple software. And more than a few epsilon minues did get sucked into letting an Apple dealer steal their old monitor ROMs when they attempted doing the "upgrade".

Sadly, this "upgrade" is needed for future releases of software, even if it utterly demolishes the value of most of the older software that you already own.

Both the old and new monitor chips are compatible with industry standard 28 pin 2764 EPROM chips by *Intel* or *Hitachi*.

The usual dual monitor solution is to take a physically similar but electrically double sized 27128 EPROM and put two monitors on it, another on the top half and one on the bottom. One quick and dirty way to switch btween the two is to lift pin #2 from the socket and jumper clip pin #2 to ground for the "low" monitor and to +5 volts for the "high" monitor.

Figures one and two show us a cleaner and safer way to handle dual monitors at a cost of only a few dollars. You plug two adaptor sockets into your Apple where the CD and EF monitor chips are supposed to go. Then you plug two 27128 EPROMs into that adaptor. A pair of switches is then flipped left for the high monitor and right for the low monitor.

One very good source for EPROM chip programming is *E-TECH Services*, who do low cost and prompt work. Note that you have to send them the exact images of the code that you want burned into your 27128 EPROMs.

The key to the adaptors is to use the premium machined contact sockets that may be safely and reliably plugged into each other.

These sockets are spaced apart by machined contact DIP strips that give enough seperation to make room for the switch. Be sure to use a slide switch and not a toggle

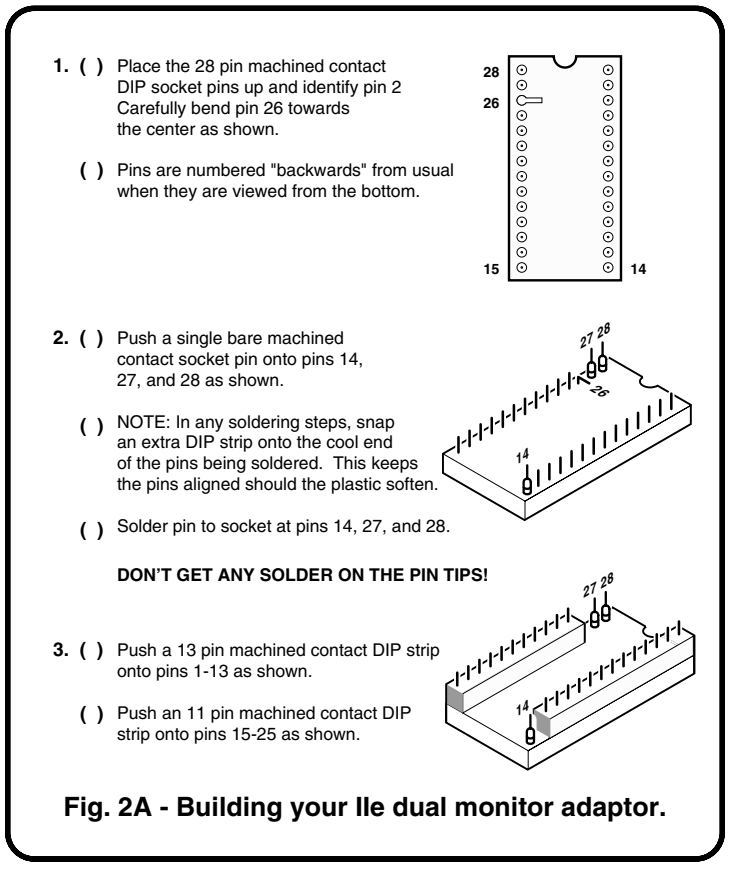


Fig. 2A - Building your IIe dual monitor adaptor.

switch, and be sure the SPDT switch is the usual type that breaks before it makes.

There are several obvious modifications and improvements you might like to try. You could replace one switch with a wire that reaches over to the center of the other switch on the other adaptor. This way, only one single switch flip will be needed to pick one monitor or the other.

Switch flipping is best done cold. If you try to flip the switches during a program, strange things may happen, depending on whether either monitor is being accessed at the time the switched is flipped, and whether the code being used at this instant is any different between the two monitors.

One really neat trick would be to use a pair of 27512's instead, whose eight 64K banks are selected by an eight way selector switch. This would let you have an old monitor, a new monitor, a word processor, a spreadsheet, a graphics program, and three other programs all resident in your machine for instant access. A different adaptor scheme would be needed, and each program would have to tow along its own needed monitor routines.

Be sure to keep me posted on all of your multi-monitor activities.

Are Their Bugs in The Vaporlock?

There didn't used to be.

The *Vaporlock* is a fast software-only way of locking the Apple II+ IIc or IIe screen to the CPU timing. Among its other abilities, this lets you mix and match text, HIRES, and LORES graphics any place on the screen in most any combination. You can also do windows, glitch free animation, software based color killing, and fancy wipes between display pages that require no remapping, plus lots of other largely unexplored new possibilities.

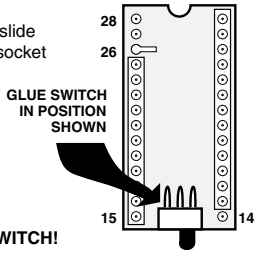
Full details appear in my book *Enhancing your Apple II and IIe*, volume II, and published by *SAMS*. I do have some autographed copies on hand here for you.

That vaporlock had been thoroughly alpha tested over

4. **( )** Carefully roughen one side of the SPDT slide switch and the bottom of the 28 pin DIP socket between pins 14 and 15. Use very fine sandpaper or steel wool.

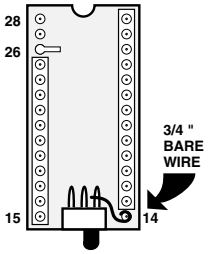
( ) Glue the switch to the 28 pin DIP socket as shown using superglue or epoxy. Let sit overnight and then verify that the switch still works.

DON'T GET ANY GLUE INSIDE THE SWITCH!



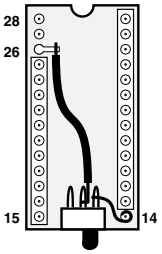
5. **( )** Prepare a 3/4 inch length of bare #24 wire. Connect this wire to pin 14 of the 28 pin socket and then to the nearest pin on the SPDT slide switch.

( ) Solder both connections, using a spare DIP strip as a safety backup heatsink. Cut off any remaining wire. Be very careful not to get any solder on the tip of pin 14.



6. **( )** Take a 1-3/4 inch piece of green #24 solid insulated wire and strip 1/4 inch from each end.

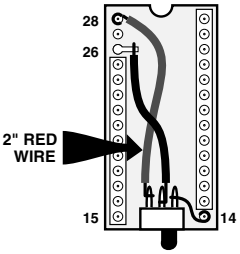
( ) Solder one end of this green wire to pin #26 and the other end to the center pin on the SPDT slide switch.



7. **( )** Take a two inch piece of red #24 solid insulated wire and strip 1/4 inch from each end.

( ) Solder one end of this red wire to the far unused pin on the SPDT slide switch. Solder the other end of this wire to pin #28 after looping it.

( ) Make sure that no solder gets on the tip of pin #28 and that there is no short to adjacent pin #27.



8. **( )** Turn the Apple IIe power off and remove the line cord at both ends. Carefully remove the original CD and EF monitor chips and store them in protective foam.

( ) Plug one adaptor into the CD ROM slot at D8 and the other adaptor into the EF ROM at D10 as shown. Plug your already programmed 27128 EPROMs into these adaptors.

( ) Be sure that the CD EPROM goes in the CD slot, and that the notch and dot on both EPROMs point forwards, towards the keyboard.

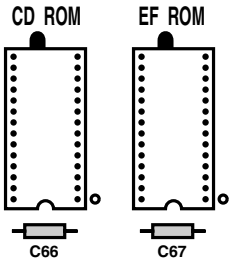


Fig. 2B - IIe dual monitor, continued . . .

two years with no reported serious problems.

But, all of a sudden this week, I got two calls from two wildly different people having problems using the vaporlock on a IIc or IIe. The problem always seems to be related to some exotic plug-in card, notably the *Ultraterm*, the *Niceprint*, and the *SCRG Switch* cards.

I don't have any of these cards and I am at a total loss as to why they should cause any problems on either the old or the new IIe. The IIe has a fully buffered data bus, so there should be no loading that would swamp out those fumes needed by the vaporlock. And, indeed, the fumes appear to be still readable from the monitor, even when the vaporlock seems to be unable to find them.

Even if these cards did access some of those invisible HIRES "screen hole" memory locations, needed by the vaporlock, they should not do it during the time the vaporlock is actually in use.

Are special connections to unusual points in the IIe involved with these cards? Are wierd interrupts or any NMI games in use? Are special chips used to alter timing? Is the memory area needed by the vaporlock sub somehow protected or overwritten? Are these cards trying vaporlocks of their own?

It is very hard to troubleshoot over the phone, but what seems to be happening is that the i.d. patch in the HIRES screen holes is somehow getting trashed.

UPDATE: Dec 85

How Can I Do An Isometric Drawing?

Why, with *Applewriter* on a IIe, of course. How else could you prossibly do an isometric drawing?

Figure three shows us some details. *Isometric drawing* is one standard way of showing three dimensional objects on a flat sheet of paper. The original X axis leans up by 30 degrees to the right. The original Y axis leans up to the left by 30 degrees. The original Z axis still goes straight up and down. You can measure the actual scale lengths along all of the three axes.

I have put together some *Postscript* routines that easily let you do the *Laserwriter* graphics directly out of *Applewriter*. My cl, cr, and ct commands handle the circles as ellipses slanted just the right way for left, right and top faces. My al, ar, and at do the same thing for arcs. The im, irm, id, and ird commands handle left moves and left draws, both relative and absolute. Similar commands do exist for the other two axes, while the im, irm, id, and ird commands do a triple isometric move or draw all in one single command. These are needed for positioning or for slanting lines that go in two or three dimenions at once.

The dashes are done by using the SETDASH operator. Compound curves are handled with cubic splines.

Neatest of all are the pr, pl, and pt commands that print right, print left, and print top, automatically slanting and arranging the letters so they seem to "belong" on any face.

The actual isometric transformations are not really all that bad. They are . . .

$$\text{vertical} = Z + 0.5\,(X + Y)$$

$$\text{horizontal} = 0.86\,(X - Y)$$

Here, X, Y, and Z are the original three dimensions, while vertical and horizontal are the final directions on the final two-dimensional page. Circles are done as ellipses with a magic angle of 35 degrees and 16 minutes, either flat for the top, or rotated plus or minus 60 degrees for the sides.

Trig freaks will note that 0.86 are the respective sine and cosine of 30 degrees, while 35 degrees and 16 minutes is the angle whose tangent is 0.5.

The lettering is nothing but a stock font that gets slanted and rotated. You either lean the letters forward or backwards by 30 degrees. The other nine lettering orientations not shown in figure three are also easily done.

(*continued on page 10.5)*

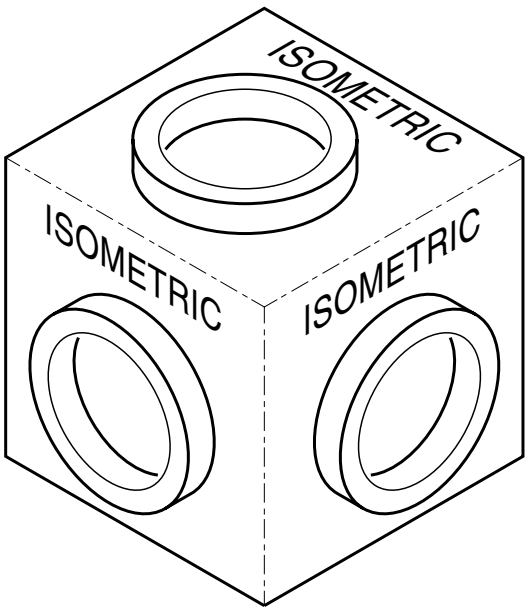


Fig. 3 - An Applewriter isometric tech illustration.

Don Lancaster's

ASK THE GURU

December, 1985

HIRES entry points
New Apple ramcard
Vaporlock interrupts
Bar code info sources
Laserwriter vs Laserjet

My, oh my, did I ever find a way to pay for a laser printer in a big hurry. The key secret is – bumperstickers.

These are trivially easy and quick to do, and there are no steep minimum orders or any totally ridiculous setup charges involved. You can easily expand into virtually any step and repeat "peel-n-stick" use, from disk labels, business cards, placards, nasty signs, impractical jokes, pornography, political bull, whatever.

All with *Applewriter* and *Laserwriter*, of course.

Most importantly, you can handle the *small* onsie-twosie orders that the old line bumpersticker people positively refuse to touch. Lots and lots of people could think up dozens of uses for a few custom labels or bumperstickers of their own. Very few want to shell out $80 or so for a minimum order to the traditional suppliers.

So, step right up folks. Tell ya what wer'e gonna do.

The *Abeja Laser Printing* branch of the *Synergetics* fine family of scams is now in the custom bumpersticker business in a big way. And, you are completely free to "steal the plans" on this one and go with it on your own.

To get in on the fun, write or call Bee at Abeja for your free sample bumpersticker.

Next, send or phone her a sketch and $5.95. We'll then send you your choice of (A) 6 custom bumperstickers, (B) 36 custom shipping labels, (C) 36 custom diskette labels, or else (D) 40 custom business cards.

Why shucks, yew caint beat that with a willow pole.

Should you want to get into this on your own, one good source of small quantities of bumpersticker stock is *Dick Blick*. Large volume suppliers advertise in *Printing Impressions* and also in *Paper, Film, and Foil Converter* trade journals. These include *Avery*, *Fasson*, and *Mac-Tac*.

Only certain papers and vinyls will properly accept heat fused toner. Those that do give outstanding results. Those that do not will turn out really awful.

Naturally, you never want to have only a "one-product" venture going. Around here, the bumper stickers are added to certificates, awards, artists announcements and business cards, discount software sales, full color diskettes, writing books and articles, technical consulting, monitoring telephone voice helplines, hand crafting specialized software toolkits and companion disks, real estate scams, typesetting services, etc. . .

Plus, of course, growing pecans and questing tinajas.

While I may have mentioned this a time or two, I do have a few copies left of this here book that is called *The Incredible Secret Money Machine*, and it does tell you all about doing stuff like this for your own fun and profit.

Hey, it works. Does it ever. On to the goodies . . .

What's the word on Apple's new RAM Card?

Actually, there are two words: Very nice.

You can now plug up to six megabytes of RAM into an *Apple IIe* that has a beefed up power supply, or "only" one extra megabyte or two without. Memory is added in 256K chunks.

The card is set up wildly different from all those third party RAM cards you may already be familiar with. Since this is an Apple supplied product, it is almost certain to set the future defacto standard for any add-on memory cards of any type.

There are several different levels at which you can use the card.

Let's look at the highest level first. On first bootup, the entire RAM card automatically becomes a virtual disk. The advantages of a virtual disk over a real one are that you can read and write to it extremely fast. The big disadvantage, of course, is that

Data R/W Byte	$C0D3
Address High Byte	$C0D2
Address Mid Byte	$C0D1
Address Low Byte	$C0D0

(Values shown are for slot FIVE.)

Fig. 1 - Apple RAM card low level entry points.

the contents of a RAM disk are volatile and will go bye-bye on power down.

When using a virtual RAM disk, you will want to back up your work to real disk files much more often than usual. Nuff said on that.

Under ProDOS, the entire megabyte can be used as a single virtual disk. But with DOS 3.3, you are limited to slightly less than 500K maximum, since this is the largest disk size DOS 3.3 is able to recognize.

Now for the unusual part. The Apple RAM card is *not* bank switched.

While superb for storing just about anything, you cannot directly *run* a program while it is still sitting in the RAM card. You first have to move that program to either main or auxiliary memory and only then can you run it.

Figure one shows us all four of the low level RAM card entry points. Only *four* address space locations are needed for each card. Those locations are the four lowest slot bytes in the I/O area. While we have shown the addresses for slot five, similar addresses for slot two would be $C0A0-C0A3, and so on up the slot sequence.

Note that *all* reads and writes to this card go through a *single* data location. This way, you can substantially speed up memory swaps for such things as HIRES animation. Direct memory moves are much quicker than the already fast RAM disk access.

Since we are talking far more than 64K here, you need three address byte locations, rather than your usual two.

You can think of the low address byte as the position byte on a 256 byte page, the mid address byte as the 256 byte page in use, and the high address byte as which bank of 256 byte pages is to be used.

The address will auto increment on each read or write. This way, all you have to do is use repeated LDA $C0D0 commands to extract as many bytes as you want in sequential order. Note that there is no 64K limit on this.

If you do *not* want to read or write in sequential order, you can preset the address bytes to any chosen location. Do this just before each read or write. Naturally, it is far faster to have the card auto-increment whenever you are able to do so. Since that auto-incrementing is done by using gate array hardware, no CPU processing time is involved at all. Which is very fast.

I can see all sorts of neat things to do with these cards. One obvious possibility involves flipping back and forth between programs, similar to Mac's switcher.

Better yet, we now have enough memory available for truly stunning long sequence animation.

Entire Apple snapshots can be taken and saved to a RAM card, opening up all sorts of nearly concurrent processing or coprocessing options.

Programs such as *Applewriter* and *Appleworks* should be very easy to link to the RAM card, although custom patches or upgrades will most definitely be needed.

Even without the custom patch, *Appleworks* will appear to run much faster, since all parts of the program will be in the machine at once. Most time lost to whirring disks can now be saved.

One big advantage that all programmers now have is a newly standard and now well defined expansion environment. This is far better than having to do custom patch work for each and every third party add-on memory card that comes along.

Here's a laser printer test paragraph that shows you whole word breaks and full wall-to-wall microjustification of **ten fonts all at one time**. Note the automatic justification cancel on the last line of the paragraph. Professional features do include an automatic initial capital, a progressive three stage microjustification, and "hanging" punctuation at the line ends. Both *italics* and ***bold italics*** can be freely intermixed. You can even include ^{superscripts} gracefully, and print $_{subscripts}$ with equal ease. Nothing very fancy is needed for a 51Ω resistor, or when $\pi = 3.14$. While the mixed fonts may be any shape or size, ranging from 3 point thru 20,000 point, most ***Special*** **Effects** are best used rather sparingly

Fig. 2 - A standard Laser Printer Test Paragraph.

How Can I Find Out All About Bar Codes?

Bar codes are those funny product code labels you've no doubt seen at the grocery store. The best running commentary I've run across on the societal impact of bar codes appears on the cover of every issue of *MAD* magazine over the past several years.

Chances are that you would prefer some more technical information than this, though. It turns out there are at least five major bar code standards. The one you already know the most about is called the *UPC*, short for the *Universal Product Code*.

You can get a copy of the UPC standard from, of all places, the *UPC Council*.

Actually, they just renamed themselves *UC* for *Uniform Code*. While this sounds a tad presumptious, they do seem to be running away with all the marbles.

Other competitive bar coding schemes include the *Code 3 of 9*, used by the military and the automotive people; the *Interleaved 2 of 5* used for containers and transportation; the *Codabar* also used for transportation; and the European *EAN*.

A few further details on these standards appear in the *Hewlett-Packard* bar code components folder, publication number 5954-2152. H-P has lots of fairly expensive solutions to bar code scanning and reading, including their *HBCR1000* series component bar code readers, their *HBCS-4300* industrial code wands, and their *HEDS-1000* reflective sensors. Check out their *Optoelectronics Designer's Catalog* for more details.

One company that I have found has bar code component parts at fairly low prices is *Scan-A-Matic*.

For a complete list of all major bar code manufacturers and suppliers, contact the *AIM* Automatic Identification Manufacturers trade group. Their free membership directory lists dozens of major bar code outfits.

What's Different Between Laserwriter and Laserjet?

Comparing these two is like comparing a skateboard against a Porsche. Yes, the skateboard is cheaper. You can also argue that the skateboard corners better and has more headroom.

But, outside of that Missus Lincoln, how was the play?

Very briefly, the Laserwriter has enough memory to hold an internal bitmap of an *entire* printed page at full resolution. This lets you mix and match *any* sized graphics and *any* size text in most *any* combination anywhere you want on the page.

The Laserjet does not.

The Laserwriter lets you magnify or enlarge *any* image through an enormously wide range. You can even magnify or reduce *anamorphically*, letting you change the horizontal size *seperately* from the vertical size.

The Laserjet does not.

Anamorphic size adjustments are very useful if you are trying to run a similar display ad in several different magazines. Often, the exact size and shape of each quarter page ad will vary all over the lot and then some.

The Laserwriter also lets you twist or rotate any image almost any which way but loose, besides allowing very fancy matrix transformations on both text and graphics.

The Laserjet does not.

The Laserwriter has real internal font machinery that will automatically give you any font size from two points to well beyond 20,000 points. (There are 72 points in one inch.) There is a nearly infinite variety of font styles and sizes available inside the stock Laserwriter. For letters larger than a sheet of paper, there is even an automatic process available that uses as many sheets as you need to get up to size.

The Laserjet does not.

You can independently set your choice of the Laserwriter font height, the font width, the amount of lean, and even the *kerning*, or the individual spacing between characters. Besides the dozen stock fonts, you can easily download any number of those custom fonts of your own.

Instead, the Laserjet forces you to buy outrageously expensive font cartridges that can only provide a very few type styles, typically limited to plain old twelve point. In this day and age, the fixed size font cartridges are an inexcusable and totally unjustifiable ripoff.

The Laserwriter has a most powerful page composition

NAME	LOC	A	X	Y	WHAT IT DOES
HGR2	$F3D8	-	-	-	Clear and init page two
HGR	$F3E2	-	-	-	Clear and init page one
HCLR	$F3F2	-	-	-	Clear current page to black
BKGND	$F3F6	-	-	-	Clear current page to color
HPOSN	$F411	vpos	hposlo	hposhi	Position without plotting
HPLOT	$F457	-	-	-	Position with plotting
HLIN	$F53A	hposlo	hposhi	vpos	Draw line from last point
HFIND	$F5CB	-	-	-	Current position ---> E0-E2
DRAW	$F601	-	shapelo	shapehi	Draw a shape
XDRAW	$F65D	-	shapelo	shapehi	Erase (EOR) a shape
SETHCOL	$F6EC	-	color	-	Set Color 0-7
SHLOAD	$F775	-	-	-	Load shape from tape

Fig. 3 - Accessing Applesloth HIRES routines from machine language.

computer built in that speaks *Postscript*, a language related to Forth that is fast becoming a de-facto typesetting standard. Postscript is available now on many of the leading "real" typesetting machines.

The Laserjet does not.

Which means, that, once you have what you want, a single phone call can instantly upgrade the 300 DPI (dots per inch) resolution of your Laserwriter to 2560 DPI or even higher "true" typesetting. Yes, things do get smoother and sharper when you do this.

And that's still done using nothing but *Applewriter* on an *Apple IIe*, along with your original textfile.

The Laserjet can't do this.

Typesetting from Postscript is now available at rates that average a dollar a page. Naturally, that does include free corrections. Run that one by your local printer.

The *Laserwriter* has a built in ability to handle *cubic splines*. This will let you do graceful, free form curves. More important, the cubic splines let you create very compact and ultra smooth typography whose resolution *increases* with increasing font size, rather than getting chunkier as is the case when using bitmaps.

The Laserjet does not.

For instance, a three inch high Roman "O" can be done with as few as *fourteen* data points when you use cubic splines. This compares rather favorably with the 560,000 bits needed for a straight bit map of the same character.

On second thought, it does not compare at all.

The Laserwriter has very powerful internal font machinery that lets you do just about any style of justification using nothing but a few external words in a totally standard text file. Particularly powerful is the *stringwidth* command that automatically calculates how wide a string of characters will be.

The Laserjet does not.

Incredibly, that Laserjet makes you do all the work in the host if you require a wall-to-wall justify with a custom font. You have to provide your own external width table and do all the font calculations yourself. This makes it extremely difficult to use the word processor and personal computer of your choice when doing fully justified text on the Laserjet.

So, the essential things you should demand in a laser printer are (1) a full page, full resolution bitmap giving an unlimited text and graphics mix; (2) a powerful and upwardly mobile and standard typesetting language; (3) Font machinery with no sane size limits and without plug-in cartridge scams; (4) the ability to anamorphically rotate, translate, scale, or transform any image; (5) powerful and useful internal justification machinery, (6) an immediate compatibility with the unmodified word processor and the stock personal computer of your choice; and (7) Cubic splines for free-form curves and efficient font handling.

Figure two shows you a standard test paragraph with the sorts of things in it that are absolutely essential for decent laser printing. Have the dealer demo this paragraph for you. Make certain nothing is needed except a routine loading followed by a stock paragraph or two on your favorite word processor running on your favorite unmodified personal computer.

The actual paragraph used does have a few imbedded escape commands in it, and is about as complicated as what you need when switching an *Epson* to bold print and back.

Yes, that initial "H" is definitely an internal part of the test paragraph.

The argument that the *Laserwriter* is expensive just does not wash. Compared to what? Most of the competive products are really empty boxes masquerading as laser printers, giving you substantially less than one percent of the performance for fifty or more percent of the cost.

No way, Jose.

Any Vaporlock Solutions?

As you may recall from last month, my *Vaporlock* is a fast and simple, software-only, way to mix and match text, LORES, and HIRES in any combination on the Apple screen, besides letting you do flawless animation, screen switching, and true video wipes. Full details appear in my *Enhancing your Apple II and IIe*, volume II (SAMS #22415).

While the vaporlock has been around for quite a while and had (I thought) been very thoroughly tested, all of a sudden problem cards have been showing up that hang the Vaporlock.

The culprit? Interrupts.

The present vaporlock will not work properly if an interrupt arrives from most any source during those screen locking time intervals.

Interrupts are seeing more and more use on new cards intended for use on the IIc and on the "new" IIe. Particularly nettlesome are mouse cards and fancy video cards.

There are several obvious solutions. One is to not allow use of problem cards.

A second solution is to read the processor status byte before starting the vaporlock locking routine. If the interrupt flag is set, do nothing. If the interrupt flag is cleared, set the flag with a SEI. Then, do the lock. Finally, reset the flag with a CLI after locking is complete.

How Can I access the Applesloth HIRES Routines From Machine Language?

You wouldn't believe how many helpline calls we get on this one.

While there's lots of places to go for an answer, perhaps the best three are the *IAC Tech Notes*, the *All About Applesoft* from A.P.P.L.E. or

the *Applesoft Disassembly Disk*, from S-C Software.

Figure three shows us the details. Each internal Applesloth HIRES routine may be accessed by calling it as a subroutine. For instance, a JSR $F3D8 will clear and init the HIRES page two screen.

Several of these routines require that you pass information to them using the A, X, and Y registers. Note that there is no rhyme or reason to which register gets used for what.

Since there are 280 possible dots on a stock HIRES line, you need two address bytes. The high byte is a zero for dots less than 255, and a one for dots greater than 255. The low byte equals the number of dots for dots less than 255 or the number of dots *minus 256* for dots of 256 or greater.

For instance, to position yourself on the HIRES screen, you put the vertical line number $00-BF in the accumulator, the low byte of the horizontal dot count into the X register, and the high byte of the horizontal dot counter into the Y register. Then you JSR $F411.

To draw a line, you must first have a position set, either by using HPOSN or by having drawn a previous line. Then you put the vertical line number $00-BF into the Y register, the low byte of the horizontal dot count into the accumulator, and the high byte of the horizontal dot count into the X register.

To draw a shape from a shape table, first you have to learn all about and use shape tables under Applesloth. The DRAW and the XDRAW commands work only with a single shape at a time. You have to manage accessing that shape from a shape table by yourself when doing machine language access.

To do a DRAW, you put the low byte of the shape start address into the X register, place the high byte of the shape start into the Y register, and then JSR $F601.

The XDRAW works just like DRAW, except that it exclusive-OR's the new shape against the screen. If used properly, this can erase a shape without hurting most backgrounds.

The color is set by putting the color number $00-07 into the X register and then doing a JSR $F6EC.

Note that only half of your dots will plot on original HIRES when using any color other than white. Thus, certain green vertical lines may be completely invisible unless you move them over a dot.

Many beginners start with black and white only. They also may shorten their HIRES screens to 256 dots wide so they can get started with minimum hassles.

Anyway, yes, you can substantially speed up graphics and animation by directly using the Applesloth internal HIRES routines directly from machine language. And, it's lots of fun to do so.

But, most all Applesloth internal routines were *not* designed for speed. No way.

These routines were instead designed for minimum space and were done as compactly as possible. In a day when 16K ROM's were extremely expensive, this was the only way to go.

Perhaps the worst example of Applesloth HIRES slothticity involves calculating the position of any dot on any horizontal HIRES line. The internal routines do this by repeatedly casting out sevens, a process that can take more than ten times the obviously fast method of using a lookup table instead.

Sure, go with these routines on your own. But, do not expect for even an instant to be able to put them into a commercial product. These internal routines are far too slow and too ungainly for that.

But they are a great first step, so be sure to play with them along the way.

(continued from page 9.4)

Isometric is ideal for any "exploded" views that show how things go together, particularly when lots of round parts are involved. One limitation is that boxy subjects seem a tad out of proportion with the far corner looking "too big". You can see this if you stare at figure three long enough. This is caused by your brain being used to seeing things in perspective.

Actually, isometric is only one of an infinite number of possible *axonometric* projections. Tech illustrators will typically avoid all the other viewing angles, since they used to be a royal pain to draw. Oftentimes, some really offbeat projection will show an object or a drawing in its best light.

But, neither *Applewriter* nor *Postscript* could not care less. A trig calculation is a trig calculation, no matter how funny the ellipses. There are a nearly infinite number of ellipse templates sitting inside the *Laserwriter*, and they all are equally accessable. Thus, axonometric or even perspective drawing is now easy to do.

Some more axonometric options appear on page 77 of Wellman's *Technical Descriptive Geometry*, which is published by *McGraw Hill*.

Actually, is there *anything* that *Applewriter* cannot do? All parts of all the figures and all the text you see here were done with *Applewriter* on a IIe. I even have a neat routine here that lets *Applewriter* play "Stars and Stripes Forever" as an error message.

So, how about a second contest this month? A free *SAMS* book to the first five *Computer Shopper* readers that show me the best ways to milk cows by using *Applewriter* on a IIe. And a tinaja quest for two (FOB Thatcher AZ, of course) to the overall winner.

WINNER: Feb 86

Assuming you are in the moood for a contest, that is.

Don Lancaster's

ASK THE GURU

January, 1986

Laserwriter rumors
Electronic halftones
Printing a video image
Post processing techniques
Translating computer programs

There are bunches of exciting new *Laserwriter* happenings this month. Firstoff, *Apple* has slashed the list price of their old *Laserwriter* by $1000 to a new list of $5995. But, even this figure is virtually meaningless, since the street price is now around $4300 if you shop around or can find a hungry enough dealer.

I have even heard of a "lowball" *Laserwriter* offer of $2600 when the printer was bundled with a Mac, a hard disk, and a CAD package. But you should beware of prices that are hundreds of dollars under dealer cost. The odds are pretty good that they will find someway to make it up somehow.

And guess who pays.

All of which tells us that there is now virtually no cost advantage to all those gutless imitation Laserwriters that are chronically unable to give you full page graphics, can not handle large type sizes, and do not have a powerful and standard typesetting language built in.

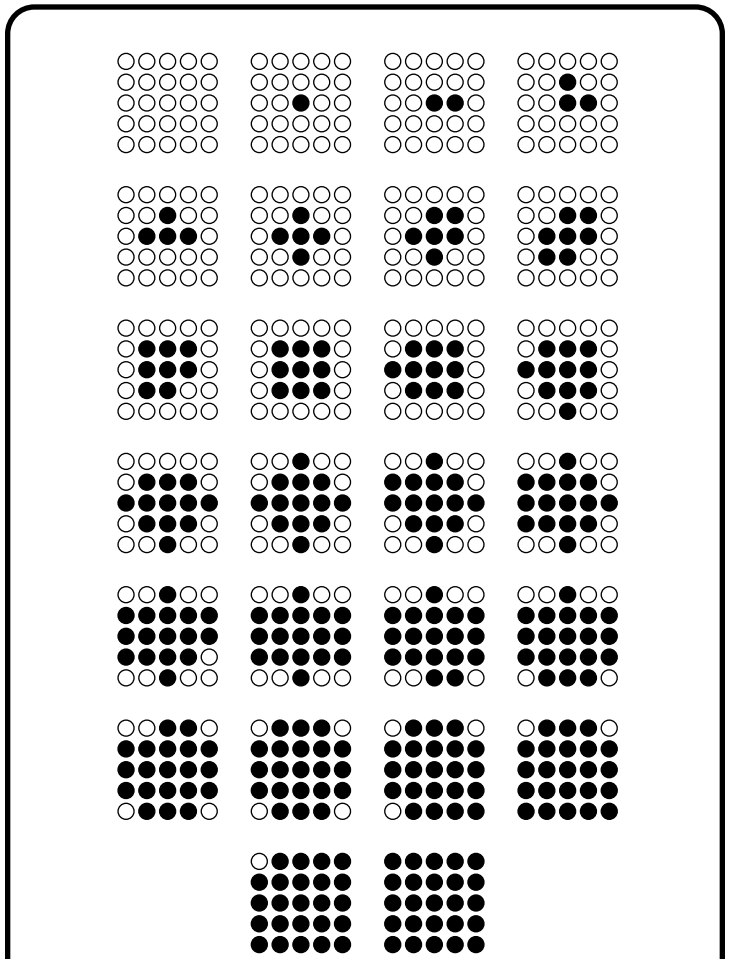


Figure One - Stock 60 line Laserwriter screen gives 26 grey levels but has a very grainy "sunday funnies" look.

Speaking of which, *Adobe Systems*, the *Postscript* people, now have two fine books available. These texts largely replace the older and hard to get *Inside Laserwriter*.

Addison Weseley publishes both of them. These are the *Postscript Reference Manual* and the *Postscript Tutorial and Cookbook*. I do have a few extra copies of these on hand here.

The *Laserwriter* also made *Infoworld's* product of the year award. By the way, if you ever do want to keep something a secret, be sure and stay out of their *review responses* column. My demo pack letter there on November 5th. has pulled thousands of requests so far.

There are also lots of new fonts in the works. These are available from quite a few different sources. As I will show you someday, it is a simple matter to build your own custom fonts. Many of the stock printshop packages are also quickly picking up a full *Laserwriter* compatibility.

There are also rumors of a new ROM set for the *Laserwriter* that will give you 42 standard fonts in a stock format that can be accessed from any word processor on just about any computer. More on this when I get full details.

Some disk-based fonts are also newly available directly from *Adobe*, but unfortunately these are not viable products as they now stand.

Why?

Because of three nearly fatal flaws. First, these new fonts are priced at $175 for a related set of four. This is twenty times what comparable fonts are worth today and ten times what the market will bear.

The perceived value of a font in a personal computer environment is 99 cents, as set by any number of popular "printshop" packages that can offer you fifty or more fonts for $49.95.

Besides that, most any idiot with a cheat book can rip off *Laserwriter* fonts quickly and simply. With only slightly greater smarts, he can also dramatically compact those same fonts at the same time.

Because of their current perceived value, the days of licensable type fonts are over. Sorry about that, but that's the way things are today.

Secondly, there are some half-assed and totally inane protection schemes that were added to both the disks and to the fonts themselves. These severely penalize the honest user, while strongly rewarding the dishonest ones. Such useless protection schemes and the thinking behind them have no place whatsover in personal computing, let alone in personal publishing.

Thirdly, these fonts are intended for *Macintosh* use. Right now, the Apple IIe or IIc is a far better choice as a *Laserwriter* driver. This is mostly because of the crippling limitations of much of today's Macintosh software. There is also that present forced Macintosh use of that slow and highly restrictive *AppleTalk* access mode.

These disk based fonts will become a major product only when and if they are dropped to the $19.95 retail level, are totally deprotected, and are offered for computers that are genuinely useful as *Laser*writer drivers.

As a reminder, even after the *Infoword* onslaught, I still have a few *Laserwriter demo packs* remaining. Call me for a free copy.

How can I Translate Programs or Text Between Computers?

The task of moving things between different brands of computers can range from trivially easy to just about totally impossible.

There are two main steps involved in translating a program from one make and model of a computer to another. The first step is to move the actual image of the program or text file over to the second machine. The second step is to modify that image so it will run in its new home.

The first step is suprisingly simple. To move *anything* from one personal computer to another, just convert that anything to a disk-based textfile and then send that file over a serial interface to the second machine. Often, serial RS232 communication at 1200 baud is a good choice, using standard printer or modem utility software. This will result in a literal and exact transfer of the text from the first machine to the second.

Note that there are no machine limitations that become involved with this technique, so long as each machine has enough memory to receive the file, and so long as the handshaking on the serial channel works. The brand of microprocessor and the operating system does not matter in the least.

In those cases of a program written in a higher level language, it is usually far better to transfer a textfile copy of the *listing* of the program to the other machine, rather than transmitting the actual program itself. The reason for this is that the parsing and interpretation of the program may differ wildly between the two machines.

The new ProDOS operating system on the Apple IIe very easily and instantly lets you change the type of any file to or from a textfile. This is done using, of all things, the

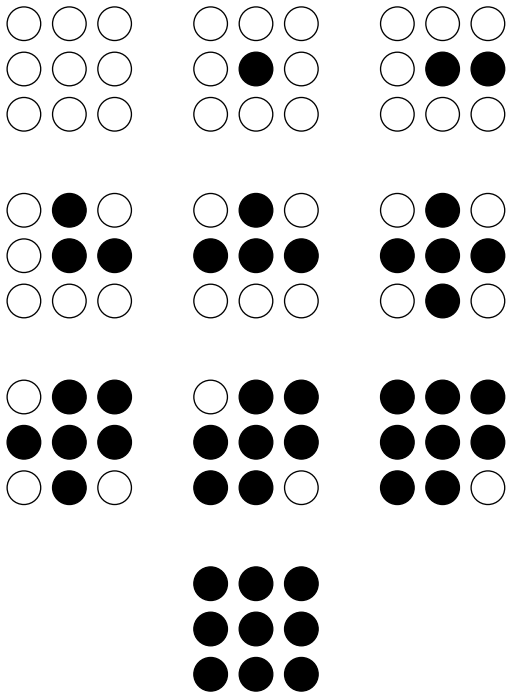


Figure Two - Custom 100 line Laserwriter screen only has 10 gray levels, but gives solid grays and video images.

TYPE command. For instance, to change a machine language binary file or program to a text file, first load that program into your IIc or IIe.

Then do a CREATE TXT. MINE, TTXT, followed by a BSAVE TXT.MINE, A$2000, E$39F6, TTXT. The starting and ending address may both change with the position and length of your program.

To convert your BASIC pro- gram to a textfile on a disk, just write a short header to your program that opens a textfile, lists that program to the textfile, and then closes the file.

To return *from* a textfile listing of a BASIC program, just EXEC the textfile. Unless the program is up to something strange, the program will first run and then will leave you with a listing in the machine. This listing can then be saved as an Applesloth pro- gram file.

So, moving either text program images or text program files between one machine and another is more or less trivial, once you get all the details worked out.

Getting the code to work in its new home is a different matter entirely. Some languages, particularly PASCAL, may end up more or less machine independent. Other languages such as BASIC, will be pretty much compatible, so long as the same dialect of BASIC is in use at both ends.

Should each machine use a different dialect of BASIC, the chances are that most of the commands might carry over. The usual way to debug things is to run the program until it blows up and then find the specific command that is causing the problem. Fix the commands one at a time until your new program works properly.

Any PEEK and POKE commands must be rearranged to suit the new machine. In many cases, you will have totally different ways of using PEEKs and POKEs. In particular, those graphics and sound mechanics will differ widly from the one machine to the other.

In general, the more sophisticated and the more elegant the program is, the more hassles you will have when you translate. To get around these problems, you may have to become a *very* knowledgeable machine language hacker in *both* the old and the new machines.

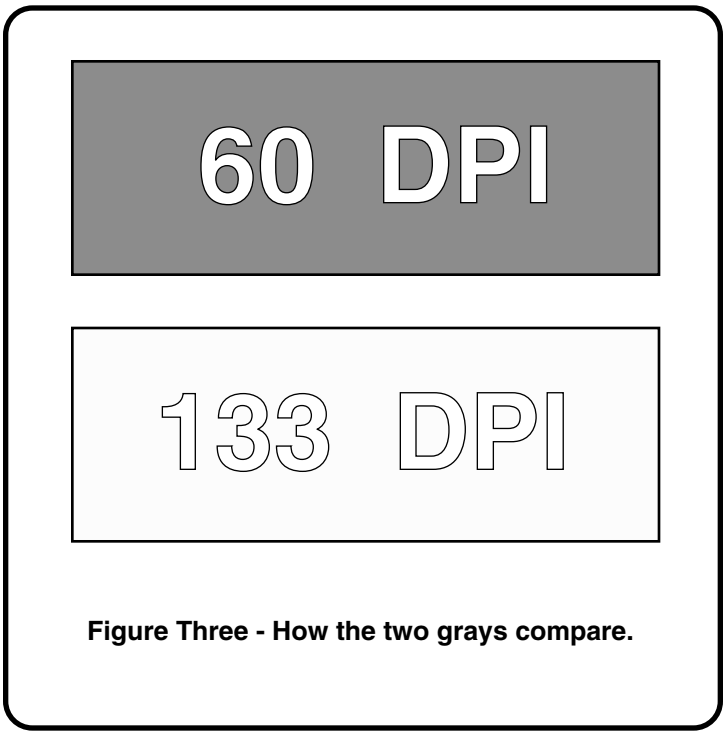


Figure Three - How the two grays compare.

What is Post Processing?

A magic technique called *post processing* lets you add custom features to your word processor, literally letting you leap tall buildings in a single bound. Such things as user friendly interaction, a wall-to-wall microjustify, HIRES or other graphics dumps, and just about any special feature that you can dream up can be easily accomodated.

Best of all, use of post processing does not normally require any machine language smarts, nor any specialized patches to the word processor of your choice. This solves a crucial dilema I have had on the helpline when I get a request for a patch that does a very special task for an oddball printer or an off brand printer card.

Post processing works on any word processor that lets you print a *fully formatted* text image directly to disk. While *Applewriter* is ideal for this sort of thing, any processor that lets you print to disk can be used for these post processing ideas.

What you do with post processing is first print to disk. Then you later use the disk image and further modify it, depending on what you are trying to do.

In some cases, you run the disk image back through the same word processor. In others, you can use *Applesloth* or some other high level language to make any changes before your final printing.

Let's look at a dozen good examples of post processing. This will give you an idea of how general and how powerful this very little-known, yet extremely powerful technique can become:

(1) To improve the final appearance of a daisywheel printer, you might "double whap" each character, thus

printing everything bold. Just print to disk and then get the disk image back. Now search and replace every carriage return with a new carriage return followed by a "print everything bold" command.

(2) Some of the dot matrix printers will continue their underlining of the right margin or the next left margin every now and then. To cure this, print to disk, reload the new image, put a "no underline" command at the end of the problem line, and then print to paper.

(3) To change your daisywheel in mid-line for italics or whatever, print to disk. Then read the disk image with *Applesloth*, looking for some keying markers. If no marker is found, one line at a time is printed. If a marker is found, the appropriate wheel swap prompt is output.

(4) Sometimes you might want to print on two wildly different brands of printer or even on both a printer and a typesetter. To do this, do a print to disk using generic commands. Then read your disk image and search/replace the generic commands with the specific ones.

(5) To switch between two word processors that have wildly differing commands, print to disk first. This strips off all imbedded commands of the first word processor, leaving you with a pure text image. Then read that image with the second program.

(6) Headers and footers can be customized beyond the limits of the word processor by repeated trips to disk. To do a triple header, print to disk two lines shy, including the bottom header line. Then reprint to disk one line shy, adding the middle header line. Finally, print to paper while adding the top header line.

(7) Even and odd pagination can be picked up by printing first to disk and then doing a search/replace on every second header.

(8) HIRES dumps or other graphics images are handled by putting markers into your text and then printing to disk. You then read the disk a line at a time by using *Applesloth*. When the marker comes up, you stop the printing and start doing a graphics dump. When the picture or graph is finished, you pick up the proper printing in the proper position and continue with your text.

(9) A "real" proportionally spaced wall to wall microjustification can be added to most modern printers by initially printing to disk. Then the disk is read by a suitable routine that counts all the individual widths of each character in the line. After that, each and every space on that line is widened just enough to force the text between left and right margins. When you do this, be sure to *not* justify leading spaces or else all your paragraphs will end up ratty.

(10) For printers that already have microjustification ready to go, a trip to disk lets you insert those special commands only at print time, so they stay out of your files.

(11) Lists can be sorted and numbers can be totaled by grabbing them just before their final printing and rearranging their order.

(12) Daisywheels having "funny" spokes might be recoded so they will print properly, again by printing to disk and then doing a replace.

You might find *Applesloth* post processing a tad on the slow side. Even so, this may be the quickest and simpliest way to get the effect you are after. Once you have what you want, things can almost always be made to run faster.

Let us briefly review those various ways we can speed up post processing, once you have exactly what you want. The entire *Applesloth* routine, or else just the time-critical parts of it, can be rewritten in machine language.

A less drastic solution is to eliminate the very slow, line-by-line disk access. One way to do this is to use the files as they sit in the word processor's workspace. Since *Applewriter* text files reside over in auxiliary memory and are unaffected by a switch over to *Applesloth*, this dual access can be easily done.

Several other methods to eliminate those line-by-line disk reads exist and may be suitable for other brands of word processors as well. For instance, you can load your entire file at once into your machine and then do an internal search. Or, you can use a memory expansion card or a RAM disk to load the post-processed image. From RAM, the files can be rapidly read.

Here's an *Applewriter* post-processing tip: Do your final printing of the disk image by using "wide open" print constants. These do include lm0,

DENSEST GRAYS:

```
133 28 {dup mul exch dup mul
add 1.0 exch sub} setscreen
```

BEST OVERALL GRAY:

```
100 45 {dup mul exch dup mul
add 1.0 exch sub} setscreen
```

DEFAULT:

```
60 45 {dup mul exch dup mul
add 1.0 exch sub} setscreen
```

Figure Four - Postscript halftone screen commands.

pm0, tm0, bm0, rm240, pi66, pl66, lj, and a blank for ut. Remember that you want no further formatting added to your disk image. What is on the disk is what you want to actually print.

Well, by now you get the general idea. If you want to get some special effect out of your word processor, just ask if a print to disk and some sneaky post processing cannot resolve things for you.

Tellyawhat. A free SAMS book to the best five uses of post processing you send me. Any old word processor at all. Fair enough?

How can I Print Black And White Photographs Or Other Video Images?

Naturally, we'll start out with our stock answer. You do this with *Applewriter* on an *Apple IIe*. How else could you possibly process video images or print good grays?

It looks like acceptable quality photographs and video images can be easily handled by *Applewriter*. By "acceptable" I mean something 75 to 100 gray dots per inch, with nine to sixteen possible gray levels. Just like most of the halftone screens you see right here in *Computer Shopper*.

This all started when I first got my *Laserwriter*. I was extremely dissapointed with how crude the grays looked. The standard grays were so putrid they looked like they came out of a *Macintosh*.

Worse yet, any thin gray lines would disappear entirely every now and then, unless you used the painfully slow *setdash* operator. Such lines are essential for overprinting layout grids onto artwork to be digitized for laser printing. The gray lines are also nice on business forms.

Recently, I checked into how the grays are imaged and came up with some suprising results. For some strange reason, the stock grays on the *Laserwriter* are spacey and crude, with the dots intentionally spread so far out that you get that "Sunday funnies" appearance.

I suspect that the reason for the lousy stock grays involves *purposely* degrading the *Laserwriter* grays so they are compatible with the lower resolution *Quickdraw* routines in the Mac.

Fortunately, it is quite easy to restore the grays back up to *Apple IIe* and *Applewriter* quality, resulting in a dramatic improvement in all your images. In many cases, your grays will look almost as if they were painted on with an india ink wash.

There is only one way to print a truly gray image on a sheet of paper. And that is to use gray ink. Should you want several different shades of true gray on the page, several passes through the press would be needed, each with an ink change to a new shade.

Since this is very ungood, the eye is instead tricked into looking at bunches of dots on the page that are so close together they appear as a gray image. As you're purposely exceeding the eye's angular resolution, those black and white areas get integrated together and you end up believing you are looking at gray, when there is actually no gray present.

With traditional printing, grays are converted to black and white dots by using a special *halftone screen* mask.

Halftone screens will produce a dot whose size depends on the gray scale at that point in the photo or video image. You might want to magnify some of the ads right here in *Computer Shopper* to see how those dots change shape and size as the areas lighten or darken.

Most halftone screens are rated in *dots per inch*. One popular traditional screen size is 120 DPI, which means that there are 120 * 120 = 14,400 gray dots per square inch. Screens are usually twisted to a 45 degree angle, so that the dot structure is minimized. Other screen angles may be used in higher quality work or for special effects.

The standard *Laserwriter* screen is a very coarse "60" DPI, which explains why it looks so lousy. Because the *Laserwriter* resolution is 300 DPI in both directions, 60 DPI translates to a 5 x 5 square having a total of 25 dots. As figure one shows us, you can get 26 gray levels out of this screen by selectively printing or not printing each dot.

Which dot gets printed when is set by the dot *spot function*. The default spot function is 1 - (X*X + Y*Y), a math function that starts at the middle and works its way to the edge of the 5 x 5 array. Each X and Y value, as measured from the center spot is *dithered* slightly, so that as you get darker and darker, only one new dot is picked up at a time.

To visualize what is happening here, get yourself a few long balsa strips that are a quarter inch square. Now cut the strips so the first is one inch long, the next an eighth inch longer, and so on until you have 25 strips.

Next, glue these strips together so they form a 5 x 5 array with one end flat, and the other end tallest in the middle and forming a spiral to the edges.

Finally, find a ring that can loosely fits the array. As you raise and lower that ring, note how more and more dots are added to or removed from your spot function.

Things do get a tad more complicated when you rotate the halftone screen out to 45 degrees. It actually turns out that the stock 60 DPI, 45 degree screen has much less than 60 DPI resolution. The oblong array used has 32 dots and gives 33 gray levels. This only averages out to a 53 DPI screen equivalent.

There are restrictions on the size and rotation of the screen, since everything has to exactly work out in integer

(continued on page 12.5)

Don Lancaster's

ASK THE GURU

Postscript typesetting
ProDOS Applewriter 2.1
ASCII control commands
The IIc "3.5" monitor upgrade
A copy protected monitor ROM?

February, 1986

We will start out with our usual reminder that this is your column, and you can pick up technical help by using that phone number shown below. Your best calling times are 8 AM to 5 PM — *Mountain Standard Time*.

Apple has apparently just dropped their $50 bargain *ProDOS Applewriter* trade in offer for owners of any older versions of Applewriter. They must have reasoned that a year was long enough. I feel that this move was extremely stupid, and that upgrade availability should go on for a minimum of at least one more year. One reason for this is that people are just beginning to discover the remarkable hidden powers lurking within Applewriter. The beauty of the WPL to Postscript linking is one newly discovered use.

Apple's steadfast refusal to sell any replacement manuals is equally short sighted.

If you agree, then do rattle *Apple's* cage. Or at least run a stick over the bars.

We have a winner in our latest contest. Mark Bannister from Huntsville, Alabama wins a free book as well as an all expense paid tinaja quest for two (FOB Thatcher, AZ.) for his winning entry in our great Applewriter contest. As you may recall, all you had to do to enter was show the best way that you could milk cows while using Applewriter.

To quote Mark "It is common knowledge that the whirring sound of the Apple II disk drives can cause milk cows to lactate on their own. Just put a bucket under the old gals and set your disks to spinning. My neighbor tells me that it works on goats as well."

Since we've only had a few entries so far, I'll "open up" the contest a bit more and award several more books. To win, just show me the most off the wall, the oddest, or the most creative current use of Applewriter.

Nuff nonsense. On to the good stuff . . .

What is the Apple IIc Monitor Upgrade?

You will find a newer and now double sized monitor ROM available for the IIc. It is called the *IIc 3.5 ROM* and has quite a few new features.

This new chip provides the firmware *protocol converter* required by the new 3.5 inch disk drives. In fact, you *must* upgrade to this new monitor if you are going to add these new smaller drives to your IIc system.

The new monitor also has a built-in *AppleTalk* driver. You now can connect your IIc on to the AppleTalk bus.

More importantly, the new monitor includes a 65C02 mini-assembler, along with trace and debug features. Not nearly as good as *Bugbyter*, of course, but still rather handy. The min-assembler is ROM resident, so no extra room is needed while asssembling or debugging.

The new monitor also adds many of those Super Serial Card commands that were left out of the original IIc monitor. This means you can now run the *Laserwriter* off a IIc almost as easily and conveniently as you can a IIe.

The 3.5 ROM is now a 32K device, arranged as two banks of 16K words. Each word is eight bits long. That bank switching is done through the use of machine language location $C028 in what used to be the cassette recorder output area of the address space.

1. **ProDOS has been updated to version 1.1.1.**
2. **The length is some thirty bytes longer, with the code now running from $2000 to $6009.**
3. **AWB.SYS, AWC.SYS, and AWD.SYS are now $0C file types, instead of SYS files.**
4. **The modem screen prompts are now carats, instead of brackets.**
5. **Characters are output in high ASCII during printing and in low ASCII on PD0 or PD8.**
6. **The DOS Options menu has been converted to "forced subroutine return" option picking.**
7. **The modem REFQ commands are compared against file values, rather than immediate code values.**

Figure One - Changes made in the upgrade to ProDOS Applewriter 2.1.

On reset, the main 3.5 bank is first selected. Each successive addressing of $C028 will toggle, first going to the auxiliary bank, and then back to the main monitor bank.

To install the new ROM, a change in the motherboard *must* be made, cutting one line, and jumpering the unused cassette line over to the additional address pin on the 3.5 ROM.

I have heard of two different ways the upgrade is being made. Developers can buy a kit and do the cutting and jumpering themselves. This kit may be available at a dealer, but it is equally likely that they will want to swap out the entire IIc motherboard instead.

There would be two big advantages to Apple for the motherboard swap. Many earlier IIc's had a problem on the 1200 baud modem. Apple tried to use a baud rate that was pretty nigh but not quite plumb. A new crystal and two trace cuts can solve this problem, and current IIc boards include this fix. A motherboard swap would pick up any older IIc's that still do not have this defect corrected.

Another reason for doing a board swap is the reliability involved in the jumpering and cutting, which could lead to repair and warranty hassles.

A fourth i.d. byte is now needed to tell the old and new IIc monitors apart. Location $FBBF is an $FF back in the original IIc ROM. $FBBF is a $00 in the new 3.5 IIc ROM.

At any rate, there is a new free manual titled *Apple IIc Programmer's Guide to the 3.5 ROM*. It is publication number *A2L4037*, and should be available directly from Apple. A complete monitor listing is included.

As far as my products are concerned, this new ROM will now let you run all of my *Applewriter/Laserwriter Utilities* on a IIc as easily as you can on a IIe. I also have an improved absolute reset patch for the IIc that can be used with either the older or newer IIc ROM. Write or call if you need more information.

ASCII	HEX	DEC	KEY	ORIGINAL USE
NUL	$00	0	[@]	Do nothing or null
SOH	$01	1	[A]	Start of heading
STX	$02	2	[B]	Start of text
ETX	$03	3	[C]	End of text
EOT	$04	4	[D]	End of transmission
ENQ	$05	5	[E]	Enquiry
ACK	$06	6	[F]	Acknowledge
BEL	$07	7	[G]	Bell or alarm
BS	$08	8	[H]	Backspace
HT	$09	9	[I]	Horizontal tab
LF	$0A	10	[J]	Line feed
VT	$0B	11	[K]	Vertical tab
FF	$0C	12	[L]	Formfeed
CR	$0D	13	[M]	Carriage return
SO	$0E	14	[N]	Shift out
SI	$0F	15	[O]	Shift in
DLE	$10	16	[P]	Data link escape
DC1	$11	17	[Q]	Device control #1
DC2	$12	18	[R]	Device control #2
DC3	$13	19	[S]	Device control #3
DC4	$14	20	[T]	Device control #4
NAK	$15	21	[U]	Negative acknowledge
SYN	$16	22	[V]	Synchronous idle
ETB	$17	23	[W]	End block transmit
CAN	$18	24	[X]	Cancel
EM	$19	25	[Y]	End of medium
SUB	$1A	26	[Z]	Substitute
ESC	$1B	27	[{]	Escape
FS	$1C	28	[|]	Form seperator
GS	$1D	29	[}]	Group seperator
RS	$1E	30	[^]	Range seperator
US	$1F	31	[_]	User seperator
DEL	$7F	127	[DELETE]	Delete

Figure Two - Standard ASCII control codes can be shown in many different ways.

Tell me About the ProDOS Applewriter 2.1 Update

The original release of ProDOS Applewriter 2.0 had a very serious bug in it. This bug affected many users of so called "intelligent" IIe parallel printer cards, including the *Grappler* and the *Pkaso*. The usual syptoms included bizarre printing, such as a burst of 23 spaces every 237 characters, or else the complete ignoring of imbedded printer codes.

Part of the fault lies in older versions of the ProDOS operating system, while another part is caused by Applewriter 2.0 outputting its characters in low ASCII form (MSB = 0), instead of a high ASCII (MSB = 1) format as is done under DOS 3.3.

The patches to upgrade to ProDOS Applewriter 2.1 are available free from your local Apple dealer. These patches are very minimal. None of the stuff that really needed fixed got done. All that was repaired was a few of the user problems involving a few third party interface cards.

Figure one sums up the main changes to the new 2.1 version of ProDOS Applewriter. There are only a few dozen bytes of code that do change; unfortunately they make the program *longer*.

Thus, *any custom patches designed for the 2.0 version will NOT work on the new 2.1 version!*

Changes in this upgrade include a later version of the ProDOS operating system; a switch to high ASCII output but primarily while you are actually printing; a change of the actual program modules to type $0C files instead of SYS files; some slight changes to the modem code; and a very minor alteration to the DOS Options menu.

Many of the *Synergetics* patches to ProDOS Applewriter can be modified to run under ProDOS Applewriter 2.1. But they *will not* run as is and *should not* be installed. Write or call for more info on these.

UPDATE: Oct 86, etc..

Are the Apple Monitor ROM's Copy Protected?

The amazing answer to this question is that, yes, the Apple monitor ROM chips are copy protected! But *only* if you try to copy them with a ridiculously expensive stand-alone EPROM programmer.

And therein lies a tale.

I started getting these wierd helpline calls a few weeks ago in which the people were claiming they were *unable to copy* those Apple monitor ROMs. Naturally, the Apple itself is able to read its own monitor. My *Snatchmon* program can also read the same monitor chips, again with no problems.

What could be happening?

It turns out that all of these people were making those absolute reset mods that I've shown you a few columns back, and all were doing so on expensive and stand-alone EPROM burners. Instead of using my *Snatchmon*, they were simply trying to read the original Monitor chips by plugging these IC's into the burner, and hand-patching that absolute reset mod into the programmer's memory, and then burning a new 2764.

And they all were getting for their effort was fatal read errors. Why?

Well, because the monitor chips used in an *Apple IIe* are *not* 2764 EPROMs. They are instead 2364 factory programmed ROMs. A 2364 turns into a 2764 *only* when pins 26, 27, and 28 are all *externally* connected to +5 vdc. The Apple IIe motherboard does do this jumpering for you, but a big and old stand-alone EPROM burner will not when it is in its *copy* mode.

The solution? Just use a cheap EPROM burner instead of an expensive one, and the copy protection problems will all magically disappear.

What is the Difference Between a Formfeed, Control-L, [L], FF, CHR$(12) and ASCII $0C?

Nothing whatsoever. These are all *exactly* the same thing.

All of these are different ways you can show the ASCII control commands. These commands form the bottom 32 codes in that standard ASCII character code. These control commands are used for such *non-printing* things as carriage returns, escapes, form feeds, line feeds, bells, and so on.

Unfortunately, you might find many different ways of showing these ASCII control commands as well as finding many different ways of reaching them.

Figure two shows us five different ways of naming the ASCII control commands and their equivalents.

The *ASCII* column shows us the memonic for every control command. As some Ferinstances, CR is a carriage return, FF is a formfeed, BEL is a bell, and BS is a backspace. This mnemonic is a reference for people only, and is not recognizable by most personal computers.

The *hex code* column will show us the same ASCII code in hexadecimal, accessed by

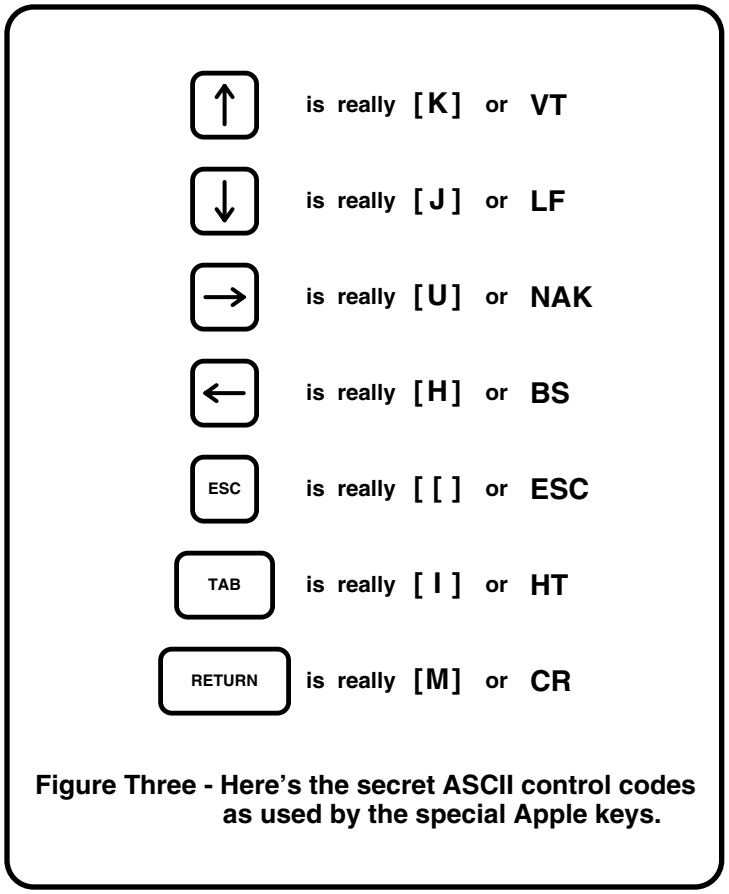


Figure Three - Here's the secret ASCII control codes as used by the special Apple keys.

a machine language program or subroutine. Here CR is an $0D, FF is an $0C, BEL is $07, and BS is a $08. You are likely to see this hex notation when you're reading machine language code.

The *dec code* column shows us the very same ASCII codes in decimal, as would be needed by Applesloth or another high level language. Here CR is a 13, FF is a 12, BEL is a 7, and BS is an 8.

As decimal use examples, you might print a CHR$(13); CHR$(4) during DOS 3.3 disk access. This will output a carriage return followed by the "Control-D" DOS access command. Or, you might do a POKE 28756, 13 to force a carriage return into a textfile. The 28756 is the address of that particular location in which you want to put the carriage return. This will, of course, change with your use application. The 13 here is the actual ASCII code for the carriage return.

The *Apple Keys* column can show how you would enter a certain control code from the Apple keyboard. As an example, a [L] means to press the control key, hold the control key down, press shift and L, release shift and L, and then release the control key. This gives you a form feed entered directly into your textfile or whatever.

Some word processors will want you to use a *verbatim* entry mode. For instance, to place a form feed into an *Applewriter* file, press [V] [L] [V]. The first [V] says to begin verbatim entry. What will follow will be placed directly into your textfile, rather than immediately acted upon. The [L] is the form feed that gets placed into your file. The final [V] cancels the verbatim entry so that any new control characters can actually be used for control purposes, instead of going into a textfile.

Finally, the *Original Use* column shows us what the intended purpose of the control command was whenever ASCII was first standardized. Most of the original uses do still apply today. It would be extremely stupid to redefine a carriage return as anything else. On the other hand, the more oddball and obscure commands, such as the group seperator or range seperator, are easily diverted to special new commands that better suit personal computers or software routines.

Some of the Apple keys are really hidden control keys.

Figure three shows us a few of these. Ferinstance, the [tab] key is really a [I] HT for a horizontal tab command.

I recently had a big flap with a none-too-bright editor who reasoned that, since there were no carriages on personal computers, there could not be any carriage returns.

Now, stop laughing. This was serious. Heavy even.

Well, even if we ignore the fact that most all personal computers are used with impact type printers that most assuredly do have carriages, the standard ASCII $0D CR is *defined* as a carriage return. Thus, any time that $0D is placed in a textfile, we most definitely have and are using carriage returns.

It is interesting to note that the original ASCII carriage return did just that. It moved the carriage back to the left of the page *without doing a line feed*. Should you want to go to the next line down, *two* commands were needed, a carriage return to get back to the left side, and a line feed to move down a line.

So, it would seem that typewriters do not have carriage returns, but that all personal computers do.

Today, of course, most all carriage returns are really combined carriage returns and line feeds. Most word processors and printer drivers usually give you a choice of how many line feeds to use. This often can eliminate any line skipping or line overwriting problems.

Just to thoroughly confuse you, ASCII is a *seven* bit code that has a mere 128 defined states. Since most of the personal computers work with eight data bits at once, the remaining most significant bit, or MSB, is free for any special uses as decided by the system designer.

This eighth MSB can be used to tell any differences between a key pressed and not pressed, between normal and inverse text on a screen, to mark the end of a word processing screen line, to tell the difference between Applesloth tokens and real text, or in any of many other ways up to the system designer.

Thus, there are two different ASCII codes, as set by who uses the MSB for what. If the MSB is zero, we are using *low ASCII*. Low ASCII is often used for standard non-Apple textfiles, and for use with Apple textfiles when under ProDOS.

If the MSB is a one, we are using *high ASCII*. Now, high ASCII is common in older Apples, both for the normal screen display and for DOS 3.3 textfiles. Since many third party printer cards expect high ASCII as input, ProDOS Applewriter 2.0 had to be upgraded to version 2.1 to make sure these cards would not get confused.

Note that high ASCII and low ASCII differ by hex $80 or decimal 128. A high ASCII carriage return is decimal 141 or hex $8D.

OK, How Did You Do It?

Ah! We have a typesetter in the audience. Section WW, row X, seat 97. The one with the green eyeshade and the ink all over their hands.

Figure two is *very* hard to typeset on most of your big mutha typesetting machines, because of the proportionally spaced text. For a very good example of how *not* to do this, check page 26 of my *Enhancing Your Apple II&IIe*, Vol. II (SAMS #22425). If you try to bracket or put a

"$" hex on a proportional typefont, the columns will all end up ratty. Will they ever.

Since it doesn't matter in the least which direction you build up a *Laserwriter* bitmap image, you simply work directly with each *column*, rather than with each *row*. Which ends up far easier and far faster.

Now for the tricky part. Each column in the figure is treated as an *individual and very narrow paragraph*. The ASCII column paragraph is left justified.

The HEX column paragraph is fill microjustified, placing a space between the $ and the first numeral and a second space after the first numeral. The spaces automatically get compressed just enough to keep all of the hex values at an exact constant width, yet still hold attractive characters that are spaced proportionally.

Note that the spaces get squashed bunches, rather than your usual stretch.

The *Dec* column is right justified. The *Key* column is fill justified through use of an opening bracket, a space, a symbol, another space, and a closing bracket. Once again, the pair of spaces get squashed just enough to keep the brackets at a constant width. The final *Use* column is left justified.

Total user time? Around fifteen minutes for the entire figure. This could be sped up further with some practice and a little help from WPL. Note that this *entire* figure was typeset; no pasteup was involved.

One very big advantage of working by columns is that the content of any column holds identical stuff, while the usual row holds a mishmash of wildly conflicting character strings.

In fact, this *entire* Guru column was written, "drawn" and *typeset* on an *Apple IIe*. And, yes, that includes all of the final "camera ready" artwork as well. Even the initial drop cap.

(continued from page 11.5)

arithmetic, modulo 300. So, while there are zillions of screen options you can call for, only a small handfull will end up unique.

Figure two shows us our first improved halftone. It is called a 100 DPI, 45 degree screen and gives you nine gray levels. The actual resolution turns out to be 106 DPI. This is "four times grayer" than the stock tones, and sure looks better.

This screen would seem to be one excellent choice for printing of photographs and video images. You would do this by digitizing an image so there are around 100 gray dots per inch in both the X and Y directions.

Then you would reduce the resolution of your digitized gray image to nine levels.

These digitized levels are next converted to hex-ASCII pairs and grabbed by *Applewriter* as part of your text file. From there, the *image* command in *Postscript* takes over and draws the picture.

You can trade off gray levels for screen resolution. The more grays, the farther apart the gray dots have to be placed.

With a stock *Applewriter*, you could handle a tad under five square inches of picture in one piece. This can be dramatically increased by going to some suitable compaction scheme or else by using a RAM card for textfile length expansion.

I don't happen to yet have any digitized video or photos on hand, but if you send me some on disk, I'll be happy to play with them for you. Between all those tinaja quests, of course.

At 106 DPI, these dots are barely discernable.

For line artwork, you might prefer even denser dots and fewer gray levels. My favorite screen is a 133 DPI, 28 degree one that gives you six gray levels. As you can see from figure three, the dots are so close together that the grays look as if they were an ink wash. Yet they still remain readable as a halftone by most printing processes.

Figure four shows you the *Postscript* commands needed to set these special screens. You can also design custom screens with background patterns or anything else you want. Full details on these pattern screens appear in the two *Postscript* books.

We can now see from both figures one or two exactly why certain grays drop out on one-pixel lines. There is a minimum amount of "grayness" needed in those spot functions to be sure that all horizontal and all vertical single pixel lines encounter at least one black dot in the halftone.

So, if your gray lines occasionaly drop out, either switch to a denser screen or else change to a darker gray.

So far, I have been unable to find any dense halftone screens having 7, 8, or 16 gray levels. Please do let me know if you find any.

There is one restriction to the use of very dense screens. The lightest available gray ends up fairly dark. This is caused first by the limited number of dots, and secondly by the fact that each dot will "overprint" somewhat so the toner dot ends up larger than its intended size. If this did not happen, solid lines and any blacks might not end up continuous.

Thus, a halftone function with half of the dots printed and half of the dots blank will appear considerably *darker* than a 50% gray.

You are free to mix halftone screens in any way you like. Thus, you can use your 133 line screen for medium and dark grays, and the 100 line one where you have to have the lighter grays.

To find out what great *Laserwriter* grays really look like, just write or call and I will gladly send you a free gray scale demo that you simply will not believe.

Don Lancaster's

ASK THE GURU

March, 1986

Curve tracing
The VIP computer
VIP user applications
Apple's Laserwriter plus
Accessing USGS data bases

I will try and be around the *Computer Shopper* booth at the *West Coast Computer Faire* April 3-6 in the San Francisco Moscone Center. I am also hoping to speak on the *Apple IIe* and laser printing, but I may have already missed their paper deadline.

Stop in and say hi.

I now have a new *Applewriter Cookbook* out from *SAMS* (#22460). If you can't find one locally, I do have a few autographed copies on hand here that I could be persuaded to part with.

And, our usual reminder that this is your column and that you can get in on the fun by writing or calling per the end box. I also have lots of neat freebies available for you if you call or write.

What's the Word on The New Laserwriter Plus?

The only real differences between the old *Laserwriter* and the brand new *Laserwriter Plus* is that eight previously empty ROM sockets have been newly filled and a program block got moved.

Inside all those new ROM chips are 22 brand new fonts that are most exciting and quite useful.

Now included are two new san-seriff style fonts named *Avant Garde* and *Helvetica Narrow*. There are also three new Roman style fonts called *Bookman*, *Century Schoolbook*, and fancy *Palatino*.

Each of these fonts are available in normal, in bold, in italic, and in bold italic variations. Rounding out this new super font collection is a calligraphic font named *Zapf Chancery* and a collection of printer's icons and "dingbat" symbols.

All the old fonts are still there, including *Symbol*, *Helvetica*, *Times*, and *Courier*. The total number of resident fonts now total 35, when you add up all the variations.

But, before we go on, be sure to watch out *very* carefully for who is calling what a font.

On the gutless, imitation Laserwriter knockoffs, one single font gives you only *one* way of showing *one* size of *one* particular typestyle only in *one* page direction.

On a real *Laserwriter* or on a *Laserwriter Plus*, one font can be shown in any size from 2 points on up to 65,000 points in quarter point increments. That font may be imaged in any direction or along any path, besides being stretchable. leanable, or twistable in any direction and capable of being outlined, shadowed, gray shaded, clipped, three-dimensioned, patterned, and who knows what elsed.

Thus, one *Laserwriter* font equals at least several billion knockoff laser printer fonts. That's assuming you are not very creative. If you are, then one *Laserwriter* font equals a totally ridiculous number of imitation knockoff fonts.

Beware of knockoff hype!

Back to that brand new *Laserwriter Plus*. Some very subtle bugs in the old ROMs were fixed in this upgrade. There also are some speed improvements of interest to those intent on using a new *Macintosh Plus* as a *Laserwriter* driver.

Using an Apple IIe still remains faster, however. Particularly since you can now serially communicate at up to 57600 Baud.

The *Laserwriter Plus* also proudly supports "downloadable fonts". But this needs some clarification, for things are not at all what they seem.

You *always* had the ability to download custom fonts from an *Apple IIe* to the *Laserwriter*, either with each job or once on initial power up. Thankfully, this ability still does remain.

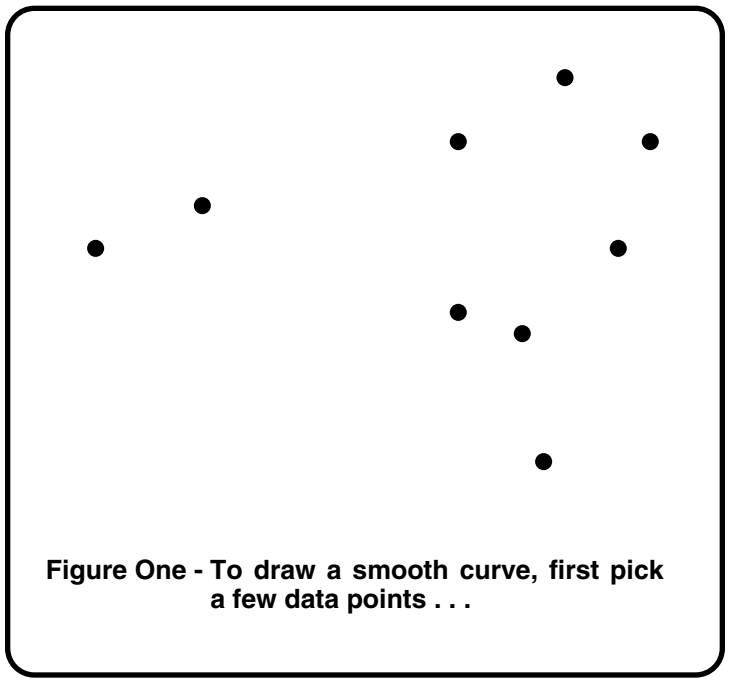


Figure One - To draw a smooth curve, first pick a few data points . . .

What has been added is a method of downloading a new series of extremely expensive, highly protected, and strictly site-licensed fonts.

Since any idiot with a cheat book can easily scarf up his own downloadable fonts for the *Laserwriter*, I can see a great battle shaping up between cheap, unprotected, and unlicensed fonts on one hand, and very expensive, locked, protected, and site licensed ones on the other.

Yet this entire battle can be easily avoided by (A) removing ALL protection from both the fonts and the font access, (B) eliminating needless and utterly unworkable site licensing, and (C) reducing the price of downloadable fonts below the level that makes using a cheat book so attractive in the first place.

Time will tell.

Several readers have asked what the internal Laserwriter computer looks like. Well, there are really two internal computers, one that's built by *Canon* as a mechanism controller, and one built by *Apple* as a Postscript interpreter.

The Apple manufactured computer is very big, very simple, and very awesome. A 68000 CPU with 1.5 megabytes of RAM and 1 meg of ROM. A total of 48 RAM chips and either 8 or 16 ROM chips. Two ports, one for the laser controller, and one for serial I/O.

A rather unusual pair of stablemates as port chips: a 6522 along with – brace yourself – a Z80 SIO.

Presumably, some features are available in the Z80 chip that aren't available in the 6551 that you would expect to find in this socket.

The ROM holds the Forth-like *Postscript* interpreter, along with the resident fonts. The average font uses around 12K of memory, but this can vary all over the lot. A "secret" compaction scheme has been used to squash the fonts down to this size.

Note that the bytes needed by the font description are completely independent of the final character sizing on the page. Further, only a single font description is needed for all possible output point sizes and image directions.

Over on the RAM side, one megabyte is used for the full page bitmap. Another 256K is used for a *font cache* that converts any currently active fonts into a character bitmap so they can be rapidly read. This solves the dilemma of having a very compact description of a font, yet still providing both a very high resolution and fast imaging of the characters when they are needed.

What is New In Small Control Computers?

I just ran into a brand new control computer that I am really impressed with. It's called the *VIP* and is made by *Bukowski Robotics*. VIP is an abbreviation for a *Very Intelligent Peripheral*.

65C02 of course. But with a crucial difference. Start with a bare bones 65C02 style microprocessor and your usual 6522 sixteen line parallel port. No ROM. No dynamic RAM. Instead, 8K of *non-volatile* static RAM. The static RAM is faked by an 8K x 8 CMOS memory that plugs into an "intelligent socket" that has a backup battery in it. Just the sort of thing you would want for a bare bones controller for most any dedicated or experimental use. And very easily expanded to boot.

Now for the neat part. This computer is also a plug-in card for an Apple IIe!

To initally develop your application, you plug the VIP into the Apple. The Apple takes over, letting you write to or read from the VIP, and turning loose all of the powerful Apple development tools including EDASM, the disk drives, BUGBYTER, modems, a printer, and whatever else.

After your VIP is first programmed, you haul it off to the shirtsleeves world where it belongs and use it as a dedicated stand alone computer or controller.

Should your application be doing some data acquisition, once your data is acquired, you can plug your card back into your Apple and upload whatever it is you just have measured.

From a memory mapping standpoint, the VIP memory is split into a "low" 6K and a "high" 2K. The high 2K maps

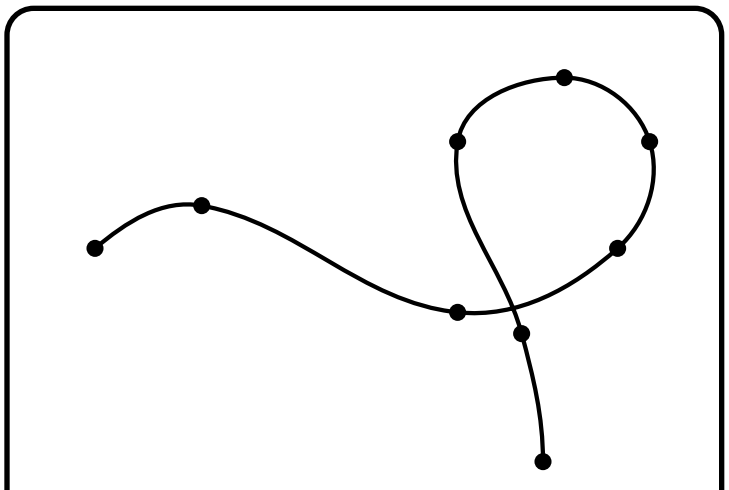


Figure Two - . . . and then use my curvetrace routine to connect the dots.

into the Apple I/O space from $C800-CFFF. To the VIP itself, this 2K sits from $F800-FFFF, just where you would expect your monitor, control program, and reset vectors to reside.

The all CMOS design VIP draws very little current and is easily battery powered. It is not, however, a micropower controller that will run forever on a tiny battery.

CMOS controller computers have long been available from outfits like *John Bell* and *Onset*. At $129, the VIP does compare favorably with these older devices, besides offering the big convenience of immediate Apple compatibility and immmediate use of Apple development tools.

There's also rumors of a new 65C02 family single chip that includes multi channel D/A converters, internal stepper driver logic, serial ports, lots of parallel lines, real time clocks, and a soft ice cream dispenser. Combine this dude with the VIP, and you will really have something.

Who Could Use The VIP Computer?

Just about anybody that wants to play with computer control of just about anything.

Let us use our own *Gila Valley Apple Growers* for some examples. I'm extremely proud of each of these people, for they are doing some really amazing hands-on computer type stuff.

They're also doing it in a low tech part of the country that is so remote that overnight delivery services take a week and where driving to the nearest decent library takes six hours.

While the VIP computer does not immediately solve all of their problems, it sure will be a big help, particularly for initial feasability work. We sure had been getting tired of burning EPROMs for each and every code change! We also were getting tired of having a dozen "alike but different somehow" designs on hand, none of which ever quite got fully debugged.

Anyway, here's a list of what real people are doing right now that involve VIP-like small control computers:

Dave is automating the adjustment of cotton picking machinery by using his small handheld controller that advances a linear stepper used as a micrometer. This very much speeds up the exacting picker shim setting process, besides greatly increasing the yields and the grade of the recovered cotton.

Phil is building intelligent taps for the cable tv system he manages, allowing individual customer services to be remotely switched on and off. Newer and much more flexible cable services can now be offered, besides completely thwarting the midnight pole climber.

Claude is now retrofitting ancient paper tape numeric control machine tools so they become fully Apple compatible. Now the full resources of an Apple, including disk storage, printing, etc., are available to these early tools.

Anne is designing a robotic router for the sign making trade. You simply type in your message, the size, and the type style, and out comes your ready to sell carved wooden sign. *Ron* is doing the same thing for an ultra high pressure water knife used in fabric cutting.

Terry is working up an elaborate control system for a hydroponic greenhouse that involves hundreds of individual water control valves. The timers previously used were really raising havoc after they got out of phase with each other.

Jim has made a brilliant breakthrough in solar pump engineering. By dynamically changing the pump stroke in proportion to the available sunlight, his small controller completely eliminates a need for any batteries, inverters, or other costly, hard to maintain, and inefficient add-ons.

Jay is a wildlife biologist and ornithologist living in a primitive area. He is very interested in remote data acquisition and recording of any environmental and weather data. He is also involved in spectral analysis of bird calls.

Gary is into model railroading in a big sort of way. Instead of simply detecting whenever a train goes past a certain point, a bar code

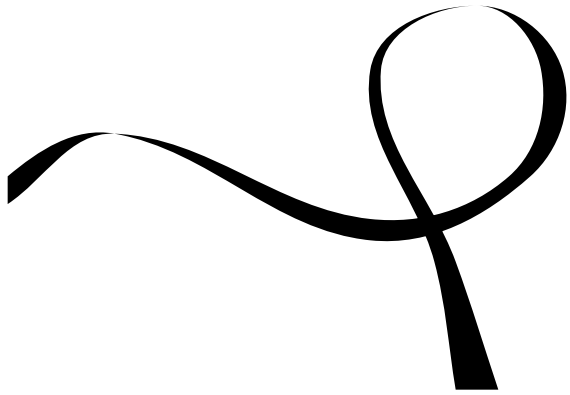


Figure Three - A variable line width is handled by tracing a pair of curves and filling the area between them.

sensor reads a label pasted underneath each car. Not only do you detect the presence of a train, but you can also tell which train it is, along with the speed, direction, and exact arrangement of all of its cars.

Bill is developing an ultra-cheap video frame grabber that lets the *Apple* do most of the work. A small control computer is invaluable when working out the initial timing details, without the need of any costly custom hardware changes.

Elaine is working on a temperature controller for a new fusion process that should lead to full *Laserwriter* color, and do so at a tiny fraction of the going rate for these systems.

As I said, these are all real people doing real things that the VIP can and will greatly help. All of these people do welcome inquires and consulting work. They also do accept cash in small bills. Just write or call if you want me to put you in touch with any of them.

Are USGS Maps Available As Data Bases?

Yes. Much of the contents of most USGS topographic maps are now available as computer readable tapes under the *Geodata* program from the *National Cartographic Information Center*.

The format is your choice of big dino computer tape or else in cuneiform on fired clay tablets, packed with a 0.03 cubit spacing of papyrus reeds.

The price is not cheap, but it is not bad either – if you really need the data. Charges around $100 per tape are more or less typical.

Typical data bases include survey boundaries, state and county lines, place names, and water features. The actual topographic contours are not yet available, since a vastly bigger data base would be required. Some day, though.

In fact, some day, you will flip a CD videodisk into your new *Apple* XVIII, hit a few keys, and then instantly print yourself a high quality full color laser copy of any map of any part of the country to any detail level you care to. Maps are a nearly ideal subject for videodisk or CD ROM storage and delivery.

What is Curve Tracing?

Some black magic involving *curve tracing* is one of the secrets that lets good old *Applewriter* on an *Apple IIe* produce umatchably superb graphic images directly from the *Laserwriter*.

As we've seen before, these images are vastly better than those that are available from *any* screen oriented graphics program on *any* other personal computer.

Curve tracing easily draws straight lines, square corners, sharp vees, and very complex,

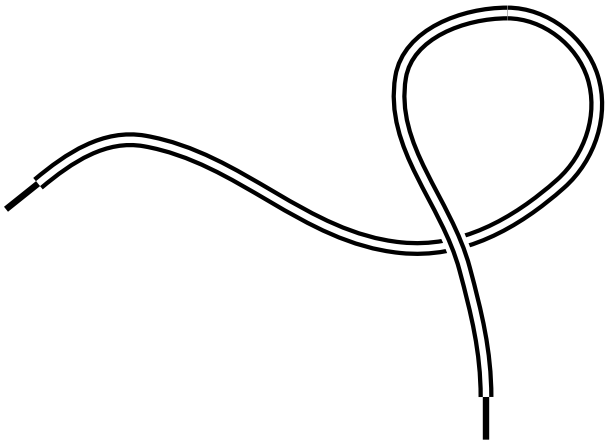


Figure Four - Pipes or wires are done by using a triple-stroking technique.

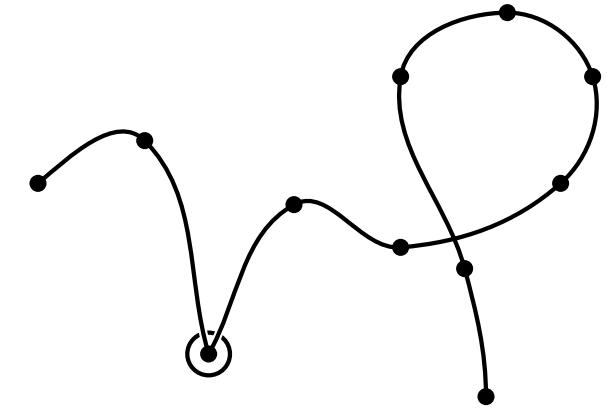


Figure Five - Cusps or sharp corners are done by re-using the same data point twice.

smoothly flowing curves. It does so with an astonishingly small number of input data points. As examples, a large circle needs only four data points, and a full page gray-shaded cartoon character ends up with a file length shorter than a typical business memo.

Let us travel step by step through a curvetrace and see just how it works.

You start out by picking some data points, as we have done in figure one. The points will obey two crucial rules:

(1) Each point must be on the curve you want to draw.

(2) The slope or *tangent* of the curve at any point will be parallel to a line drawn between the previous point and the next point coming up.

The locations and the exact spacing of the points is up to you. Obviously, the more points you use, the closer the curve fit, and the better the results. The fewer points you use, the shorter the file, and the faster the image will get drawn.

Not so obvious is how you choose the spacing between adjacent points. But if you follow the two rules, and remember that the slope at any point is set by the adjacent point pair, things will work out pretty good.

Next, you punch these data points into *Applewriter*, then bracket them, and follow them with the magic word *curvetrace*, and out comes figure two, an unbelievably smooth curve. You should have previously installed my curvetrace routine into your text file.

Both ends of the curve do need special treatment. At the start, you tell the curve the direction you want to head in, expressed in degrees. At the end of the curve, you tell the curve the direction you want to come from, also expressed in degrees.

My routine takes the points and then converts them into the data values needed by the cubic spline feature of the *Laserwriter*. Once converted to cubic splines, the rest is trivial. You can think of my curvetrace routine as an "inverse spline transformer".

Hint: should certain areas look a tad ratty, pay close attention to the *difference* between any two adjacent data points, making sure these are changing in a smooth way.

I'll be most happy to send you the free listing of my curvetrace routine if you call or write. I will also throw in some cartoon samples and some other goodies. Don't forget that I also have a free *Laserwriter* demo pack that's available just for you.

Back to the program.

Figure three shows us how you handle a variable width curve. What you do is trace two adjacent curves and then fill in between them.

Figure four shows how you would draw a wire, a pipe, braiding, or even an elaborate

Figure Six - Curve tracing can include straight lines, corners, and smooth curves as in this Roman "I".

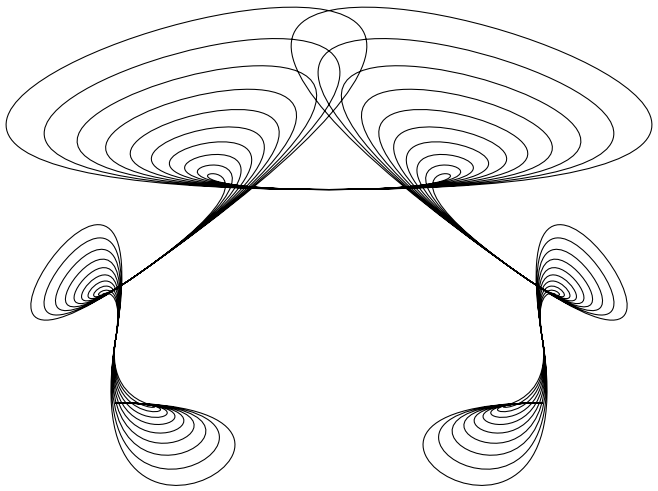


Figure Seven - Varying the tension produced this Applewriter graphic art. Only seven data points were used!

border. You start out with a curvetrace. Then you repeat this path three times. The first time, you draw a wide white line. This line will then erase anything the wire or pipe runs over, giving you the neat little break that you can see at the crossover.

Next, you draw a slightly narrower black line. This sets the "outside diameter" of the pipe or the wire. Finally, you draw an even narrower white line. This produces the inside of the pipe or wire.

The apparent black line width you end up with is one half of the difference between the width of the black stroke and the white one.

Comprehende?

Now that may look like a plain old wire to you. But for years I have been searching for a way to quickly and believably show natural looking wires for electronic pictorials and such. Thanks to good old *Applewriter*, wires are now a trivial task.

Let us move on. Figure five shows us how to handle a cusp or a sharp edge. All you do is double the data point, repeating the same point pair twice. Figure six shows us a Roman "I" that combines a mix of straight lines, cusps, and smooth curves.

Now for the artsy-craftsy stuff. There is a parameter in my curvetrace routine called the *tension*. Normally, you hold the tension to its optimum value of +2.82.

The tension determines the "enthuasiasm" with which the curve leaves or enters the next point. With the optimum tension, you do get the most accurate possible curvetrace.

But being accurate is no fun. At least not all the time.

Instead, some very neat things happen when you vary the tension away from its optimum value. If you use a tension of 1000 or greater, you will get straight lines between all the points. These straight lines are a useful debugging tool that make sure the points are really where you want them.

A tension of zero does the exact opposite, and shows you the tangent line through each point. This is also useful as a debugging tool.

Tension values positive and above normal tend to flatten the curve. Any tension values positive and below normal tend to loop or "destabilize" the curve. Positive tension values near zero (say 0.3) give you "spirograph" effects, while negative tension values near zero (try -0.3) give you "inverse spirograph" effects with all the loops on the inside, rather than the outside of the curve.

Finally, our just for fun figure seven shows you the effects of varying tension of a simple curvetracing. What utterly and totally boggles the mind is that this *entire* figure is derived from only *seven* data points!

So, What Good is Curve Tracing?

There are all sorts of incredibly powerful things you can do with curve tracing.

The biggie, of course, is downloadable type fonts. By curvetracing, you can have very smooth letters and characters that even get better as they are magnified, and yet need only one compact file to handle any and all different sizes and orientations of text on the page.

Next on the list are signatures. There are two ways to handle laser printed hand signatures. You can create a bitmap of a signature with a digitizer, or else you can use a curvetrace. The curvetrace will almost always end up far smoother and vastly more convincing, besides taking up far less file space and not needing any costly equipment, skills, or special setup.

Cartoon characters are one obvious possibility. As I mentioned above, I have a few I would like to share with you if you call or write. Free form art of any style is another obvious candidate. One example would be an indian head for a silk screened Tee shirt.

We have already seen how our pipes and wires can be handled for tech illustrations. The very same ideas can be abstracted for borders, braiding and such.

One thing that radically excites me about curve tracing is that you can eliminate the need for any digitizer, scanner or video camera when grabbing your artwork! Those curvetrace points are sparse enough that you can quickly and easily digitize almost anything by hand.

Here's how I currently digitize stuff. First, I enlarge the original art on a copier, making the image as big as possible. I then run off several copies. These copies are then put into the *Laserwriter* paper tray, and one of the gray grids from my *Applewriter/Laserwriter Utilities* package is overprinted onto them.

In the case of a character in a font, a special grid is used instead that is carefully scaled and rotated until the character is squarely sitting inside a 1000 x 1000 box.

Continuing, you then put the combined grid and art image on a piece of cork and stab your data points with a pin or needle. Then you hand enter the points into *Applewriter* and you are all set. A light box helps a lot here.

Any variations from your original art are also easily done. You can pick off only the part of the image you really need. You can also change the shape and size of everything as you go along, adding or dropping detail. Try doing that on a digitizer or a scanner!

Once again, remember that you need suprisingly few data points for a curvetrace, so doing things by hand is not that tedious or time consuming a job at all. And the final results quite often will be absolutely spectacular.

Don Lancaster's

ASK THE GURU

April, 1986

WPL and [Q]-C
Laserwriter Plus bugs
New Apple RAM card
A new control computer
Postscript schematic drawing

Be sure to stop in at the *Computer Shopper* booth at the *West Coast Computer Faire* April 3-6 in San Francisco's Moscone Center. I will try and be on hand to meet you all four days. I'll also haul along a bunch of autographed books and software, particularly the new *Laserwriter Utilities*, my new *Ramcard Disassembly Script*, and my freshly reprinted *Micro Cookbook*.

Bee should be there with all of her demos to answer all your desktop publishing questions, and I'll probably be speaking on laser printing with the *Apple IIe*.

Drop by for a while.

So much is happening so fast, that I don't know where to even begin. Forrest Mims sent me a copy of his brand new *Siliconnections* book, a most readable history of the roots of the microcomputer revolution.

And Tom Weishaar sent along his *Your Best Interest* book on calculating interest rates by using a spreadsheet. Tom reveals many of those things that banks and other lending institutions definitely do not want you to know.

I also just got word on an incredible new card for the IIe that can let you cheaply display real photographs and video images directly on your *Apple IIe* monitor screen with full gray scale. I'll be using this exciting new system to directly print photographs using *Applewriter* along, of course, with the *Laserwriter*. One obvious use is for real estate people who can print a "photograph" directly on their home listings.

But, we will have to save details on this mind-blower for a future column.

On with this month's goodies . . .

What is the Word on the New 6502?

I just ran into one really exciting new microprocessor chip that should be ideal for most any small controller, trainer, or data gathering use. Its the *Mitsubishi M50734* and sells for $12 in singles, dealer stock. It is built in CMOS and speaks 6502, so it should be quite easy to use with any *Apple*, *Commodore*, *Atari*, or the rest of the 6502 gang.

Internal bank switching is used to create a double size 128K address space. That second 64K switches only on indexed instructions, so it is primarily intended for use as data storage.

No RAM or ROM inside, but you simply won't believe all of the ready-for-your-use peripheral circuitry stuffed into this beast. How does 40 I/O lines, 9 timers, a four input A/D converter, two dual phase stepper motor drivers, a *Centronics* port, a pulse position modulator, a two-way UART serial interface, and a synchronous serial I/O sound?

The operating power is around 150 milliwatts, but this can be greatly reduced by lowering the clock frequency, and can be almost entirely shut down (to 5 microamps!) with a sleep provision.

If that's not enough for you, there are enough pins on the package that you can also put it to use as an emergency cheese grater.

The obvious uses would include such things as data recorders, hydroponic gardening controllers, cattle feeder setups, cable tv intelligent taps, student trainers, hot tub

```
--- HIGH LEVEL ENTRY --
$C500  - Initialize  Ramcard
$C50A  - Memory test entry point
$C551  - ProDOS device driver entry
$C54E  - Protocol converter entry
$C5DB  - DOS RWTS entry
$C5F7  - Turn on card ROM
$CFFF  - Turn off card ROM

--- MID LEVEL ENTRY ---
$C800  - Command interpreter
$C8F9  - Read RAM block
$C964  - Write RAM lock
$CAB2  - Find installed RAM
$CAEF  - Format Ramcard as diskette
$CCF0  - Memory test routines

--- LOW LEVEL ENTRY ---
$C0C0  - Address low byte
$C0C1  - Address mid byte
$C0C2  - Address high byte
$C0C3  - Data read/write
```

Figure One - Here's the entry points to Apple's Ramcard. Slot 5 values are shown.

pump cyclers, solar powered array interfaces, for general robotics, weigh scales, cotton picker tooth setters, student trainers, numeric controlled machine tools, etc . . .

Several minor gotchas. The 64 pin, 70 mil shrink DIP package is oddball, so sockets may be very hard to find. An 8X clock is needed, and a large block of the page zero addresses are reserved.

In addition, there are not enough pins to go around, so they used the $#@% *Motorola* style pin multiplexing on the data/low address lines. At the very worst, this can mean you will have to provide for a 74HC373 addressing latch in your final circuit.

It might even be possible to drop this dude and an adaptor directly into the 65C02 socket on an *Apple IIc*. *II+*, or *IIe*, and then run everything else out of a humongous new I/O connector.

But you would have to be very sneaky to convert Apple's 1X system clock with its very precisely controlled and intentional jitter into the 8X clock as needed by this chip. Switching around the page zero locations can get rather tricky, and a few of the Apple signal lines would have to get faked somewhow.

But stay tuned.

The beauty of being more or less Apple compatible is that the full resources of the Apple become available for your design, debugging, and testing of what you want to do. No costly ROM masks or emulation software is needed, and the whole design process gets fast and fun by using standard assemblers, debuggers, printers, disk drives, and so on down the list.

An obvious thing to do is combine the *VIP* peripheral card we looked at last month with this exciting new chip.

You'll find lots of information on the programming of 6502 style chips such as the M50734 in my two *Micro Cookbooks* (SAMS #21828 and 21829). I do stock these two titles for you.

I've got a contest going on this chip in my *Hardware Hacker* column in *Modern Electronics*, so I'll let you in on the fun as well. A free *SAMS* book to the best ten uses for the M50734. The overall winner gets an all expense paid tinaja quest for two (FOB Thatcher, Arizona).

One suggestion: Several of the local *Mitsubishi* reps seem to be suffering from acute recto-cranial inversion problems, so go directly to their main office and plant for data sheets, ap notes, and further information.

Any Further Details on Apple's Ram Expansion Card?

I've only played with this new memory expansion card for a week or two, and I am convinced it is far and away the best route to adding extra memory to your *Apple IIe*.

Let us get several of the gotchas out of the way right off. There are at least nine bugs in the 2732 firmware on the card. Locations $C15E, $C25E, . . . and $C75E should all be CPY $C0 rather than the CMP $C9 commands. Fortunately, this code is only needed to trap an input error that the operating system should already have flushed.

The error at $CFF2 is so embarrasing that we cannot even talk about it in a family magazine such as *Computer Shopper*.

There are two *extremely* important use rules. The addresses *must* be set in low byte, mid byte, and high byte order to prevent overflows from trashing those address bytes you have already set. And that data word access *must* be done using absolute addressing. Since the data access will auto-increment the addresses, indexed addressing or indirect indexed addressing will move you along by *two* memory locations, due to a double access.

Another solution is to go ahead and use the indexed addressing, but be certain that you cross a page boundary. For instance, a $BFF8,X will work just fine, but a C000,X will double whap.

Note that this very strange and double whapped indexed access does apply both to the 6502 and 65C02, but does so for different reasons.

One big surprise is that the Ramcard does *not* significantly speed up any *Apple*-writer file access. As a timing example, a 25K text file takes

```
% plusbugfix
% . . . . . .

% repairs copypage and prefeed bugs.
% insert at beginning of textfile.

/intendcopypage { gsave initgraphics 100
100 translate 1 1 true [ 1 0 0 1 0 0 ]
{<00>} imagemask grestore} def

/copypage {systemdict /copypage get exec
intendcopypage} def

intendcopypage

% . . . . . .
```

Figure Two - This "workaround" will bypass two serious Laserwriter Plus bugs.

16 seconds to read from disk and 10 seconds when reading from the Ramcard.

The reason for this is that *Applewriter* reads files one block at a time so its internal powerful searching delimiters can be used to retrieve portions of a file.

For a 25K textfile, a total of fifty ProDOS calls will be needed, each of which grabs one 512 character block.

At any rate, I've just put together a complete disassembly script and source code capturer for this card, along with some utility routines. Included are full secrets of how the card is initialized for all of the DOS, ProDOS, Pascal, and Protocol Converter operating systems. Included also are details on how to safely and selectively reserve memory areas for your own personal use.

Figure one gives you a preview of some of the legal and not so legal entry points to this card.

Write or call if you are interested in a fresh copy of this new script.

Why Does [Q]-C Foul up A WPL Program?

In *Applewriter*, when you are running a WPL program, an attempt to load new print constants with that [Q]-C option will, of course, load new print constants. Included in the print constants are new values for the WPL variables (x), (y), and (z). Should any current use be presently made of these numeric variables by the WPL program, bizarre and unpredictable things will almost certainly happen.

One cure is to save crucial (x), (y), or (z) values to the $A-D string variables, and then to restore them after the new constants are loaded. A second cure is to hand load each of the print constants from within WPL.

I've got a few new *Applewriter* patches almost ready, some mine, some by others. These do include various 2.1 patches, more WPL variables, and real time clock access. We'll have much more on these next time.

I hope.

Figure Three - To draw an Applewriter schematic, start with a gray rubber grid . . .

What are the bugs in the Laserwriter Plus?

As with any major upgrade in any computer system or peripheral, the *Laserwriter Plus* Version 38 ROM's have several bugs in them that did not come out of the woodwork until many users started exercising their machines in ways unexpected by the programmers.

The good news is that the worst two bugs can be easily repaired with a single "workaround" paragraph that you can add to the start of your files. The bad news is that one of the bugs affects the most used command and the other one simulates a major paper path failure. After performing a two day complete printer overhaul, it sure had me fooled. That's no bug – its a feature!

The first bug involves the *copypage* command. There are lots of times when you want to print a copy and then go on and create several changes and then reprint some of the old image mixed in with the new. For instance, the fast way to do a form letter is to put the entire letter into the *Laserwriter* and then erase and change only the name and address for each successive letter. Yet another major use of *copypage* is as a debugging tool when it is used as a breakpoint. Sadly, a "broken" copypage routine can introduce more bugs than it can find.

The second bug involves prefeeding. To speed things up, the new *Laserwriter plus* looks ahead of where it is processing and starts feeding paper before it is actually needed. For 95 percent of the time, the processing finishes before the paper motion is complete. For another 3 percent of the time, there is a slight delay of a few seconds. But for the remaining two percent of the time, the machine just sits there spinning its wheels, running its laser and making strange noises,

for as long as five minutes.

Figure two shows you the workaround that repairs both problems. Simply place this routine at the beginning of your *Postscript* code any time you want to use copypage for repeat imaging or any time else you get that "stuck in the snow" wheel spinning.

It probably is *not* a good idea to use this patch all the time as it slows things down somewhat and only adds to your program length.

There is also apparently a third bug that involves a fatal limitcheck on the frame device buffer. This one is way out of the mainstream and so far has not been pinned down. But it certainly is real.

By the way, the new *Laserwriter Plus* has these main differences over the original machine: The amount of ROM has been doubled by going to double sized chips. There are some exciting and useful new built-in fonts.

Operation is now slightly faster when using an *Apple IIe* and considerably faster when using a *Mac*. However, the apparent speed of a IIe remains faster than that of a Mac when running most of your typical applications.

In addition to the previous ability to download low cost, unprotected, and unlicensed user defined fonts from any source, the new machine also offers you a special feature to download from one single source of expensive, protected, and site licensed fonts.

How Can I draw an Electronic Schematic?

Why, with *Applewriter* on an *Apple IIe* of course. How else could you possibly draw an electronic schematic?

The image quality and the flexibility both end up vastly better than anything that's currently available from *any* screen oriented graphics routine on *any* microcomputer. Why? Because an exact text description is far and away the most powerful, the most flexible, and the most exact way of wringing the ultimate perfomance out of your *Laserwriter* or *Laserwriter Plus*.

By the strangest of coincidences, I've put together an electronic schematic routine that is a small portion of my *Applewriter/Laserwriter Utilities* package. What I would like to do here is share with you some of the key secrets and show you how its done.

As figure three shows us, you can start out with a gray *rubber grid* that will let you

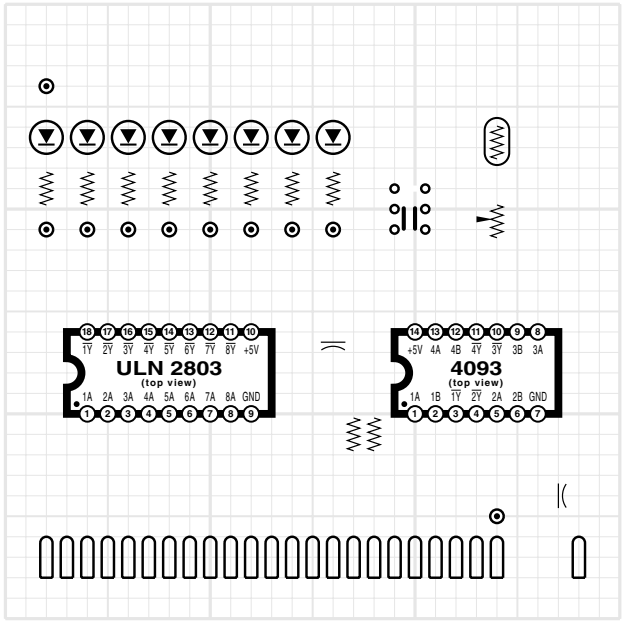


Figure Four - . . . Now, drop your symbols in place where you want them.

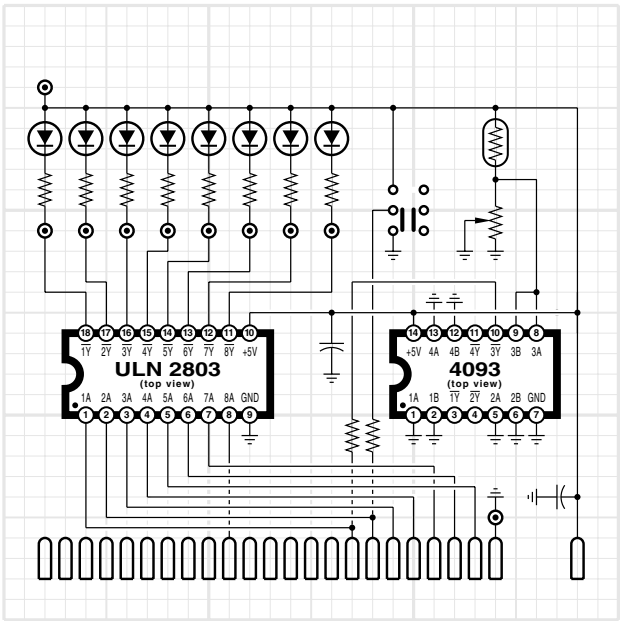


Figure Five - . . . Next, slide the wiring under the symbols. Note the breaks.

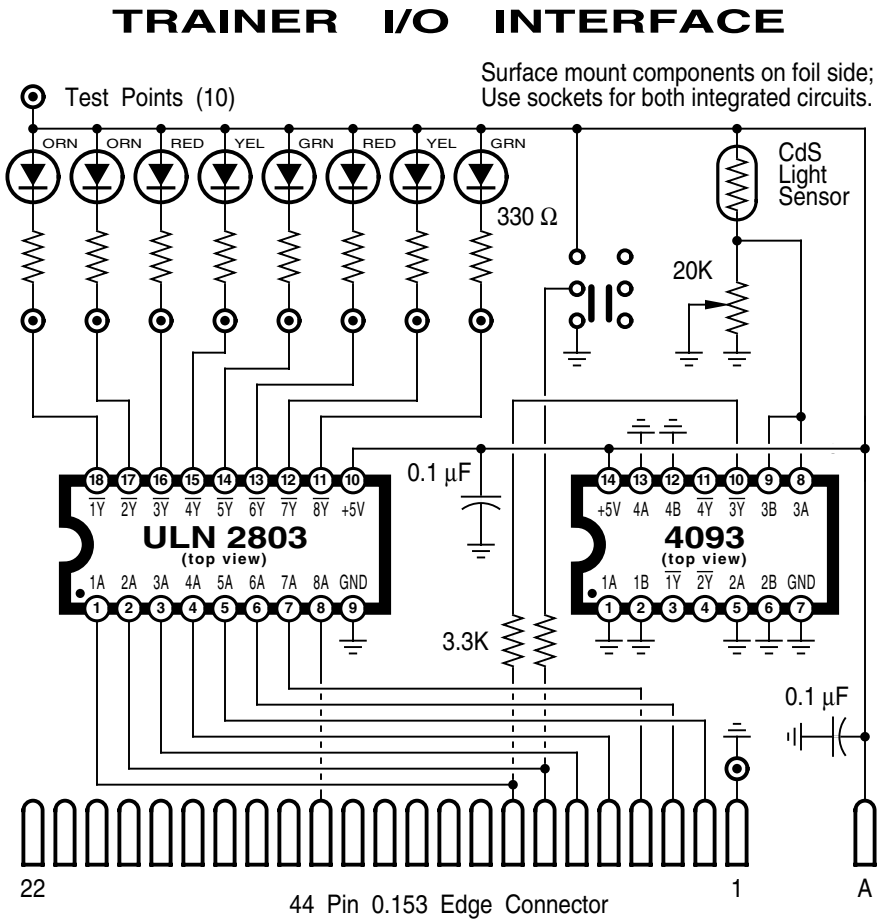


Figure Six - . . . Then add the lettering to complete the final schematic.

stretch or squash the drawing to the size you desire. Normally, one unit of the grid is defined as the space between adjacent pins on an integrated circuit package. You can turn the grid on and off as needed.

Unlike the screen graphics grids, you can easily use any fractional grid positions. No matter how fine the grid or how nervous the programmer, horizontal and vertical lines always will stay that way.

Next, as figure four shows us, we put down some *opaque symbols*. A symbol can be something like a resistor or an integrated circuit or perhaps a logic gate.

Whenever these symbols are put down, they erase and then overwrite whatever happens to be under them.

Thus, you could put down one *continuous* wire first, and then drop a resistor on top of it. This is far faster and far easier than worrying about exactly where each connection to each part of each of the symbols has to go. Symbols can later be slid along a wire for best appearance.

The symbols all use *action points*. Rather than define the position of a symbol from its corners, you define it from a logical connection point.

The usual action point for a resistor would be its center. The action point of an integrated circuit pictorial is the center of pin number one.

Some symbols make use of an *automated DIP pictorial generator*. To draw a complex 40 pin integrated circuit, you use a special routine. All you have to tell this routine is the number of pins, the name of the chip, and two strings of callouts as needed for the top and bottom pin rows. Thus, an entire DIP pictorial can be drawn with a few keystrokes!

Complement bars are easily and automatically added when and as you need them.

Symbols can be simple or complex. To show the resistance omega, you just type

(continued on page 15.4)

Don Lancaster's

ASK THE GURU

May, 1986

Anti-aliasing
Laserwriter gossip
The Imageworks card
Ripping off a type font
Digital image processing

Hello again. There are lot of really great new things happening, so let's just jump right on in.

I've got two new freebies for you. A pair of handouts on RS-232-C interfacing. One details the pinouts, while the other shows how to build your own breakout box. Write or call if you want these.

I have also just personally reprinted my hard-to-find *Micro Cookbook*, Volume I, and am also shipping my new *Apple RAM Card Dissasembly Script* that reveals all the hidden secrets of this exciting new memory expansion card.

Moving right along . . .

What does the Law say On Ripping off Type Fonts?

Nothing whatsoever.

To quote: *"Typeface designs in the United States, with rare exception, are not protected. Typeface names are protectable."*

This statement is from a house trade journal of the leading proponent of continuing type font licensing. What it says is that you can rip off an exact copy of just about *any* type font and can do so completely legally. On the other hand, "they" will nail you to the wall if try to use "their" name on your ripped-off font.

As a practical matter, I would suggest avoiding outright theft of any newer fonts from any of the major typography houses.

I would also suggest using Apple's convention of naming your custom fonts after cities. To go one step further, I personally use very small towns instead. Towns such as Guthrie, Mule Creek, Granville, Elgin, or Bonita.

My favorite cheat book is J. Biegleisen's very old *Art Directors' Work Book of Type Faces* from *Arco Publishing*. The curvetrace routine in my *Applewriter/Laserwriter Utilities* makes grabbing your own high quality fonts very easy to do.

The *Laserwriter* normally uses *analytic* fonts that are described in the *Postscript* language. Important advantages of analytic fonts are that they look better, can enlarge smoothly, take up much less memory, and can be scaled, rotated, or stretched over an incredible range, while being routed along an arbitray path.

Apparently, *Apple's* utterly ridiculous font protection is a contractural licensing obligation from the original source of their typography.

One of the more absurd features of this incredibly stupid protection scheme is that the crucial *stringwidth* command becomes either 100 times slower or else 100 times more inconvenient.

Any New Laserwriter Gossip?

Let us see. My spies report today's street price of the *Laserwriter* at $4100 in many parts of the country. That's the price a hungry dealer will ultimately accept on a quick and no-hassle cash sale. The asking price quoted over the phone is some $800 higher.

Many of theos initial bugs are now out of the *Laserwriter Plus*. I showed you a work-around to the *copypage* problem last month. Apple has just released upgrades of two new Mac disks. The February 10, 1986, Version 1.0 releases of *Laser Writer Fonts* and *Printer Installation* should now be available.

```
% timer.p   stopwatch module  (Postscript)
%. . . . . .

/timerstart { usertime /starttime exch def} def

/timerstop {usertime starttime sub /duration exch
            def (time duration is ) print duration
            20 string cvs print ( microseconds.)
            print 500 {37 sin pop} repeat flush} def

% demo:

(
An empty repeat loop ) print

timerstart 1000 {1 pop} repeat timerstop

(
A single character stringwidth command ) print

/Helvetica findfont [12 0 0 12 0 0] makefont setfont
0 0 moveto

timerstart 1000 {(A) stringwidth pop pop} repeat
timerstop
```

Figure One - This "stopwatch" will help you debug slower Postscript code.

I'm still fighting a bad *Framedevice Limitcheck* error bug on the plus that seems related to the Apple IIe serial communications. Please be sure and immediately call me collect if you find a way to reliably activate this bug.

Apple's internal service manuals are not especially noted for being stacked on the street corners everywhere for free distribution. I've found out that the *Hewlett-Packard* Laserjet manual 02686-90904 has bunches of useful stuff in it that applies directly to the *Laserwriter* print engine. The manual costs $50.

Most of the mechanical parts are interchangeable between the two printers.

Of particular interest is the detailed troubleshooting section for the various print quality problems.

Several surprises from this manual: There are three different sensitivities to the toner cartridges, set by small clips at the cartridge left rear. That "fried toner" smell is really ozone. And the cleaning pad also contains a special silicon fuser oil, so any cleaning and reusing of these pads is a no-no.

There's a quick cure for *Laserwriter's* "reverse collating" hassle. Just put your printer on a two-drawer filing cabinet and open the top drawer. As the pages spill out, they will flop over automatically into the drawer, stacking up in the right order.

How long does a *Postscript* routine take to execute? Figure one shows you a rather simple "stopwatch" listing that you can temporarily build into a program to find out where the time is being spent. This is most useful for speeding up your programs.

One of the nasty suprises the stopwatch revealed is that the crucial *stringwidth* command takes an intolerably long 12 milliseconds and is nearly *independent* of the number of characters in the string being imaged. *All the time is wasted deprotecting!* That translates to several minutes to justify a page of text, unless you get *very* sneaky.

We'll end the gossip with the usual reminder that I have a free demo pack for you that shows you the superb and unbeatable graphics you can get out of plain old *Applewriter* running on a IIe. Call or write for your free copy.

What is Anti-Aliasing?

By now, just about everybody has stared at those "jaggies" on an Apple HIRES screen or wherever and has wished there was some way to get smooth diagonal lines and believable circles.

One obvious route is to increase the resolution. This is a really dumb approach that ups system complexity, and slows things down bunches.

Instead, there is a rather simple way to get rid of the screen jaggies. This method is called *anti-aliasing*. While anti-aliasing is used all the time in movie and television special production, it has seen very little usage in personal computers.

Anti-aliasing requires that you be able to smoothly display any shade of gray on a black and white screen, or any saturation value on a color screen. What you do is replace any pixel that is touched by a line with that amount of gray that is proportional to the *percentage area* of the covered pixel. The picture gets slightly less sharp, but all of the jaggies nearly disappear.

Let's look at three examples. Figure two is your usual diagonal line on a black and white only screen.

Since we cannot trust the *Computer Shopper* printing press with exact gray shades, you'll have to insert your own grays. Using a soft pencil, and coloring only inside the lines, shade all the boxes of figures two, three, and four. Leave 0 white. Shade 100 black. You then make 50 a medium gray, and so on.

As you can see, there are two severe jaggies in the line, caused by the pixel sampling process.

Exact anti-aliasing is used in figure three. If you drew the exact line over the boxes, the shade of gray you get each time would equal the percentage of the box that was covered by the line.

Observe that this line did get a little wider. But at the same time, the jaggies got much less severe.

The only problem with exact anti-aliasing is that you have to make seperate and very complex calculations for every line on every screen. Things get particularly nasty when several lines *cross* the same pixel at once.

Instead, figure four shows us a quick and simple approach to anti-aliasing known as *low pass filtering*. What low pass filtering does is replace any given pixel with the weighted sum of adjacent pixels.

For instance, we might use this low pass *matrix*:

```
06  12  06
12  50  12
06  12  06
```

What this says is to take half the blackness of the pixel we are on and add it to an eighth of the adjacent pixels and add one-sixteenth of the diagonally adjacent pixels to that. You then replace the center pixel with this value.

But isn't this the same as purposely blurring a photograph or de-focusing a lens? Well, yes, except for one key point: *The highest frequency information in a digital display is caused by the process of pixel sampling*.

Therefore, when you low pass filter, you remove a lot of the jaggies while you only somewhat blur the image you are after. The net result is a tremendous improvement in viewability.

Note that you can do low pass filtering on just about any image of pretty near any

size. You can easily apply anti-aliasing to all of your *existing* HIRES images!

More information on anti-aliasing appears in various *Siggraph* issues of *Computer Graphics* that should be available in most large technical libraries.

Uh, the only little gotcha is that we need some cheap and powerful way to show decent grays on your *Apple* screen. For such a mind boggling breakthrough, we will have to move clear on up to the next question . . .

What's new in Digital Image Processing?

Digital image processing is any manipulation of visual data bases, photographs, or video by a computer. Until recently, this was strictly limited to large and expensive dino mainframes.

Today, though, there's a flood of new digital image processing products for the *Apple IIe* and other personal computers that are now hitting the market.

Ferinstance, there's a new integrated circuit called the *MV1001* by *Multivision*. This chip lets you inset a digital picture inside another video image, doing all that nasty gen-locking, frame grabbing and synchronization for you.

Then, there is the *Grafex* card, new from *Ray Dahlby Electronics* that adds a magic NEC 7220 chip to the Apple, giving you a display of up to 640 by 1600 pixels, along with dedicated graphics firmware that's much faster than Mac's *Quickdraw* graphics.

And there is also a new *Copyscan* text scanner from *Image Peripherals* that gives you an ASCII textfile for any printed page you feed it.

But, all these pale totally when compared to the new *ImageWorks* card made by *Redshift Limited*. This $200 jewel plugs into slot seven of an *Apple IIe*, and then superimposes a 256 x 256 video image of 256 grey levels on top of any Apple screen.

You can now show full gray scale photographs and other images on your Apple screen, all under your direct program control!

Bunches of fancy bells and whistles available. You can load and store images from disk or wherever. Two or three photos fit on a disk side, while 20 fit onto a 1 megabyte RAM card. You can instantly reverse, invert, or rotate the image. You can set to full or half brightness. You can tuck a half, quarter, or even a one-eighth image onto you screen.

There's some amazingly sophisticated image processing built in that is seldom found in personal computers. For instance, you can low or high pass filter. Low pass filtering is used for anti-aliasing and to remove any graininess or excessive scan lines from an image. High pass filtering is used to find edges or to improve contrast.

Which lets you do things that photographers can do with variable contrast paper or dodging and burning. A histogram adds up the values of all 65,536 gray shades and then plots how many of each gray shade you have. The equalization process then will assign nearly the same number of pixels to each of the gray scale values.

This can let you eliminate camera nonlinearies, improve contrast, and do a number of other neat things.

Speaking of which, there is a $98 piggyback companion subcard which directly digitizes input video to 64 gray levels. The price is held down by requiring a *stationary* image for eight seconds. Unlike expensive digitizers, plain old Apple software does all the work.

Full RGB color options will be shortly available, as will other versions.

Only two, or at the most three, pictures will fit on a floppy disk side. So, this card is best used with hard disks or an Apple RAM card.

The card's organization is unusual in that there is *zero* resident firmware. The entire 256 bits of slot I/O space are used to pass a video line to and from the Apple. The sixteen "scratchpad" I/O slots are used to send the various flag commands to the card.

Redshift is to be highly commended for their totally open and unlocked software,

0	0	0	0	100	0	0	0
0	0	0	0	100	0	0	0
0	0	0	100	0	0	0	0
0	0	0	100	0	0	0	0
0	0	0	100	0	0	0	0
0	0	0	100	0	0	0	0
0	0	100	0	0	0	0	0
0	0	100	0	0	0	0	0

Fig. 2 - A Jaggy HIRES line (see text.)

0	0	0	0	63	37	0	0
0	0	0	0	88	12	0	0
0	0	0	12	88	0	0	0
0	0	0	37	63	0	0	0
0	0	0	63	37	0	0	0
0	0	0	88	12	0	0	0
0	0	12	88	0	0	0	0
0	0	37	63	0	0	0	0

Fig. 3 - True anti-aliased HIRES line.

0	0	0	0	25	75	25	0
0	0	0	6	30	66	18	0
0	0	0	18	66	30	6	0
0	0	0	25	75	25	0	0
0	0	0	25	75	25	0	0
0	0	6	30	66	18	0	0
0	0	18	66	30	6	0	0
0	0	25	75	25	0	0	0

Fig. 4 - Anti-Aliasing by lowpass filtering.

and, wonder of wonders, a complete schematic diagram included in their user manual. The easily modified software uses machine language modules that can be ampersand linked directly to Applesloth.

How good is the quality? That all depends upon your perspective. From a personal computer standpoint, this is far and away the finest image you have ever seen on a microcomputer screen.

Anyway, ever.

On the other hand, a 256 x 256 image is by no means photographic quality. It is not even "Polaroid" quality. It is not broadcast television quality. Let's call it near the high end of premium "home video" quality.

I've been linking this card to the *Laserwriter*, of course. One thing that becomes very obvious very fast is that people's expectations of image quality are vastly lower on a video screen than they are on the printed page.

The image size depends on the halftone screen. For instance, a 100 line halftone screen gives you a full resolution picture that is 2-1/2 inches on a side. The smaller pictures lose details; larger ones have redundant dots.

So, What Good is the ImageWorks Card?

Well, first I will tell you. Then you can tell me in *two* seperate contests.

Here's a dozen good uses for the ImageWorks card:

(1) By loading a HIRES image and doing a low pass filtering, you can anti-alias, eliminating practically all of the jaggies from your existing screen pictures.

(2) Realtors can now print custom pictures of houses on their multiple listings. These can be sent over phone lines with ordinary modems.

(3) Art Instructors can now show you actual paintings along with overlays that emphasize both the form and the composition.

(4) Adventure games can have both the map as well as a full screen text description present at the same time.

(5) Gray or white block-out areas can be added to help in filling out a form or to hiding answers in student quizzes and drills.

(6) CD disks can now hold both digitized photo- graphs and program materials. The potential here is utterly awesome.

(7) Blink Comparison can be used to find out if two pictures are nearly identical. Important uses would include both astronomy and quality control.

(8) Fancy video wipes can easily be done. A RAM Card wipe takes between 0.4 and 1.2 seconds.

(9) Image rectification can be used for improving the quality of a low cost video camera, reducing any non-linearity and improving gray scale.

(10) Architects can superimpose a HIRES sketch on top of a site photograph.

(11) Microbiologists can automate culture and cell counting. HIRES crosshairs are easily added under game paddle control.

And, of course . . .

(12) photographs can now be easily printed on your *Laserwriter*, so long as you are willing to trade off the halftone screen density versus the number of available grays.

So much for my ideas. Now, what are yours? A free SAMS book to the top ten *ImageWorks* new ideas, with an all expense paid tinaja quest for two (FOB Thatcher, AZ) to the overall winner.

On top of that, Charlie Springer of *ImageWorks* will seperately judge any and all entries as a second contest. On any entries he "deems meritorious", he will either lend you or outright give you an *ImageWorks* card and possibly some other goodies.

For the best odds of winning, mail one copy to me and one to Charlie.

(continued from page 14.5)

the word *ohms*. To show a capacitance mu, you just type the word *micro* and so on.

Things such as arrowheads, clock pulses, and connection dots are similarly handled. You do not have to redefine these in each textfile for a particular schematic; they are already predefined in the preloaded schematic routines.

Next, as figure five shows us, comes *slide-under wiring*. Since these symbols are all opaque, they will position themselves over the top of any wires.

By putting the wire descriptions *early* in the textfile and the symbols *late* in the textfile, all of the wires will magically stop at the exact edge of each symbol. Usually, you put your symbols down first, and then slide the wires under them.

This mind-boggling trick is done simply by entering wire descriptions *above* the symbol callouts in your textfile, but while entering the wires *after* you enter the symbols.

Note the sneaky use of *self-breaking wires*. Observe the white space every time the vertical wire ducks under a horizontal one. This can get handled automatically through the drawing of a thick *white* horizontal wire and then the redrawing of a thin *black* horizontal wire on top of the white one.

Finally, the lettering and the callouts are added to get the final and full sized result shown in figure six. Note that any font of any size can be used, and that you are free to stretch or squash the font either in the vertical or the horizontal direction.

If the "slight rattiness" of this final image is not good enough for you, a single phone call to someone with a *Linotron 300* will instantly upgrade the image to a 2650 dot per inch resolution. You also have the option of printing oversize and reducing the final result with a litho camera or reducing copier.

Don Lancaster's

ASK THE GURU

June, 1986

A keyword indexer
Laser printed badges
Postscript circular text
Apple rumors mongered
Applewriter WPL versions

The publishing deadlines being what they are, I am actually writing this a few days before the West Coast Computer Faire. But, if I got there, and if you got there, and if the great quake of 86 is far behind schedule, many thanks for stopping in and visiting.

As per usual, this is your column and you can get technical help per the ending box. I also have some new freebies for you. Besides the Applewriter/Laserwriter demo pack, you can now pick up free handouts on RS-232-C interfacing and the ASCII code. Just call or write.

New products include my disassembly script for the Apple 1 Megabyte RAM card. This script shows all of the innermost secrets of the formatting and the disk access firmware, including a few things Apple might not want you to know about.

As I mentioned in our last column, I like this card very much and think it is *the* way to go for both Apple II+ and IIe memory expansion.

Apparently *AST* agrees with me, since they have just knocked off a clone that imitates Apple's version. Their clone optionally piggybacks a second megabyte of RAM on the same card if you need it.

I have also just recently reprinted my *Micro Cookbook*, Volume I, so this once hard-to-get text is now back in stock here.

And now . . .

Care to monger any Apple Rumors?

There are sure enough of them to go around this month. That *Apple IIx* is probably a reality and probably is in the hands of some very secretive developers. It probably uses a

```
pnd
ppr [L]
ppr
pin  Name of textfile to be scanned  ------> =$A
ppr
pin  Name of file to save key words -------> =$B
ppr
pin  New (N) or Append (A) index file --> =$C
ppr
ppr [L]
ppr  Place your work disk into drive #2.
ppr
pin    Press any key when ready --->
ppr [L]
oh,d2
p
p
ppr          *** busy - please wait ***
ny
l$A
pcs/A/$C/
pgo ap
pcs/a/$C/
pgo ap
pas =$C
pgo nw
ap pas+=$C
nw pas [esc] [\] =$A
pas [esc] []] =$D
b
a f!$A!
[esc]
pgo b
pgo c
b u
u
s$B!$D!$C
y
p
p
pas+=$C
pgoa
c ny
l$B
p
p
b
f<$D<><a
p
p
s$B
y
p
p
pqt
```

Fig. 1 - A ProDOS Applewriter WPL indexer.

65C816 CPU that is capable of emulating both the 6502 *and* the 68000, meaning it may run both Apple II and Macintosh software, or at least some of it. The memory will probably use 256K x 8 plug-in cards, similar to the Macintosh Plus. There will probably be some use of surface mounting.

Apple has also quietly introduced a new Apple IIe! It is different enough to have a changed part number of A2S2100 instead of the old A2P2104. None of my spies know anything about this yet. My best guess is that it is only a minor modification to the main board that either lets or will let the IIe have a double-wide monitor toggled by a $C028 bit test. This is the same thing that has been done on the IIc to pick up compatibility with the new Protocol Converter stuff. If, so, this means a SCSI interface for the IIe is just around the corner.

Apple has also announced student rebates of $150 on an Apple IIe. To qualify, you do have to be a full time student at a college, trade school, or university. Just about any employee at these institutions can also qualify.

They are also "opening up" that internal *Apple Link* telecomm and bulletin board network. This net was originally intended mostly for dealers and developers, but selected user groups can now qualify. The first user group hour per month is free, and then $25 per hour after that to tap all of Apple's inside secrets and tech documentation. There is a ten hour per month use limit to prevent outright ripoffs.

Speaking of opening up, apparently tech information and tech literature is becoming much more available. For openers, there's three *Support Training Library* manuals available. Part numbers are 072-0187 for the Apple II; 072-0186 for the Macintosh; and 072-0188 for peripherals.

Technical note mailings are also now available. The 1985 Apple II tech notes cost $45 for a complete set, with a current bimonthly subscription available for $25. Similar Macintosh tech notes are $25 for1985 and $25 for a bimonthly subscription.

So much for the upbeat stuff. Now for the flip side.

The highly touted SANE, or *Standard Apple Numeric Environment*, products have quitely been removed from some current price lists. I do not know whether it is an outright flush or simply some last minute bug repairs.

These SANE packages are (were?) most useful for any floating point mathematics and similiar calculations.

Worse yet, much of the old DOS 3.3 support has been dropped. *The DOS Toolkit*, *DOS users manual*, and the *Programmers manual* have been dropped from some current price lists.

What are the two single things Apple could do right now that would benefit them the most? (1) Unbundle the IIe and offer anybody and everybody a *very* substantial trade-in allowance for a II+, and (2) Make all the servicing information and all the service routines very much more widely available than they are at present.

Fig. 2 - A typical convention badge.

Why are Applewriter WPL Routines Version Dependent?

WPL is the macro supervisory language used by most newer versions of Applewriter. Without WPL, Apple is an outstanding word processor. But with WPL, Applewriter is incomparable.

Many users are finding out that some WPL routines written for the older DOS 3.3e version of Applewriter will not run on the newer ProDOS Applewriter 2.0 or 2.1, and vice versa.

Why?

There are enough significant differences between the two versions that many, if not most, WPL routines will need at least minor changes if they are to run on both operating systems.

Firstoff, the old DOS 3.3e version allows long filenames that include spaces and some punctuation. The ProDOS file names must start with a letter, can only be letters and numbers, and must be less than 16 characters long.

Second, those stored print

constants files are now wildly different, since the ProDOS print constant file has to tow along some serial port info. The names are also different. The old DOS 3.3e Applewriter automatically *prefixes* a "PRT." to the print constants filename. ProDOS Applewriter automatically *postfixes* a ".PRT" tag.

Third, some of the user functions have been renamed or renumbered. This is especially true of those [Q] Additional Functions commands, and [O] Disk Access commands. Some WPL routines may end up requesting the wrong action because of this relettering.

Fourth, any disk-based text searches made in the ProDOS Applewriter versions are *only* allowed to use the "!" as a delimiter. Many older DOS 3.3 disk-based searches used the "/" instead.

Fifth, attempting to write to an existing file *may* pro- duce a "Destroy Old File Y/N ?" prompt under ProDOS Applewriter. When this occurs under WPL, you have to add a new line of "(space)Y?" after each disk access.

These five are the major differences, although there are probably a few more as well. The ultimate way to translate a WPL program is to rewrite the entire program out in very simple declarative English sentences. Then do each sentence *by hand*, watching the prompts and the results of each command. Then modify your sentences so they work under ProDOS Applewriter. Finally, rewrite your original program to pick up any of the needed changes.

Show me a WPL Keyword Indexer

Keyword indexing is very useful whenever you want to generate an index for a book chapter, or any other time you want to save a list of certain key words or phrases into a seperate textfile.

I finally did get around to translating my old keyword indexer for DOS 3.3e over to ProDOS. All that really needed done was a change to "!" disk searching delimiters, and some answer lines needed inserted for the "Destroy Old?" prompts.

Figure 1 reveals all.

What you do is carefully chose a "phrase start" marker and a "phrase end" marker that will only appear on the screen. This means that the markers must not change the length of any line computations, and that they must not be either recognized nor used by the printer or I/O card.

For many printers, a good starting marker is an "[esc] \" form seperator. You can use

```
initgraphics

/outsideclover {135 45 [-82 23 -119 62 -154 100 -180
130 -189 150 -170 156 -130 160  -88 176 -29.5 189
29 176 70 160 111 156 130 150 121 130  95 100 60 62
23 23] curvetrace closepath } def

/insideclover {135 45 [-71 35 -106 75 -134 108 -155
134 -155 134 -110 140 -70 155  -29.5 162 11 155  51
140 96 134 96 134 75 108 47 75 8 35] curvetrace
closepath} def

/buttonproc { gsave outsideclover 8 setlinewidth 0
setgray stroke insideclover 5 setlinewidth 0 setgray
stroke grestore } def

0 setlinewidth gsave

/buttonproc1 { gsave 0.475 dup scale 30 40 translate
buttonproc grestore} def gsave 300 300 translate 4
{buttonproc1 90 rotate} repeat grestore

newpath 2 setlinewidth
300 300 65 0 360 arc gsave 1 setgray fill grestore
4 setlinewidth stroke 1 setlinewidth 0 setgray
newpath 0.90 setgray 0.3 setlinewidth

gsave 300 300 translate  7 { 65 0 moveto -65 0 rlineto
13.8 rotate 65 0 rlineto closepath gsave 0 setgray
    stroke grestore 13.8 rotate } repeat grestore

/starpath{ dup 0 exch 2 div rmoveto /starheight exch
0.3635 mul def 18 rotate 0 starheight neg rlineto 72
rotate 5 { 0 starheight neg rlineto 36 rotate 0
starheight rlineto -108 rotate } repeat} def

0 setgray 300 300 65 0 360 arc 4 setlinewidth stroke

0 setgray gsave 300 300 moveto 50 starpath gsave 1
setgray fill grestore 2 setlinewidth stroke grestore

gsave 300 300 translate 0 setgray

/NewCenturySchlbk-Bold findfont [30 0 0 30 0 0]
makefont setfont (Convention   1986) 55 -90 152
insidecircletext (Safford,  Arizona) -40 90 138
outsidecircletext

/NewCenturySchlbk-Bold findfont [25 0 0 25 0 0]
makefont setfont (AFFA) 53 -90 56 insidecircletext
```

Fig. 3 - A Postscript badge drawing program.

an "[esc] [ ]]" group seperator makes a good ending marker. These are ignored by most printers, but be sure to check yours. Further, since they are an escape command followed by a single character, the *stretchifier* patch in my *Applewriter Cookbook* will automatically ignore all of them in justification line character counts.

To use your new keyword indexer, mark the start of each word or phase you want to keep with your starting marker, and the end with the ending marker. You can mark multiple words, but the length limit on any single phrase is 64 characters. These markers remain in your textfile and will appear on screen only.

Then you run the WPL. KEYWD.INDEX routine. This will automatically scan your text file and create a list of all the keywords. You have the option of starting a new list or appending an older one.

Note that this is a middleweight routine that will put the words into a list for you. It does not search for any duplicates, nor put the results in alphabetical order. You can easily extend the routine for fancier features.

A gotcha or two on this listing: Where you see a [L], this means to insert a control-L into the WPL program. Where you see the phrase marker definitions, you insert an escape and the "control-backslash" for an [esc][\], and an escape and a "control-closing bracket" for [esc][ ]].

What is this month's Laserwriter Scam?

Badges. The *Laserwriter* is absolutely ideal for custom badge printing at fairs, shows, conventions, sporting events, and wherever. Its ability to quickly and easily set circular text is a big advantage when doing many badge styles.

The automatic step-and-repeat routines in my *Applewriter/Laserwriter Utilities* let you do lots of badges on a single sheet of paper. You can also do such things as automatic sequential numbering, or "real time" insertion of the "badgees" name, even in calligraphy.

By now, you have surely seen those "Badge-a-Minit" kits available from many mail order suppliers. This is basically a hand-held press that will convert a message on a flat sheet of paper into an attractive badge. In quantity, the badges themselves cost around 13 cents each.

You can get away from black and white by using a colored paper, such as *Astrobrite* or something similar, by using *Kroy Kolor*, or else by using denser grays.

Figure two shows a badge we did for a local convention.

Figure three shows much of the *Postscript* used for this badge. Also needed are the *insidecircletext* and the *outsidecircletext* routines from *Adobe* and my own *curvetrace* routine. I'll be happy to send you a free copy of the later on request.

Most badges will use a far simpler and far shorter routine. This one is long because of the fancy quad curvetrace used for the background, and the detail in the star and the sunburst.

Once you get a library of stock shapes and formats built up, it only takes two or three minutes to create the badge from a cold start. Should you be making repeat badges, the per-badge time will be much less than this one.

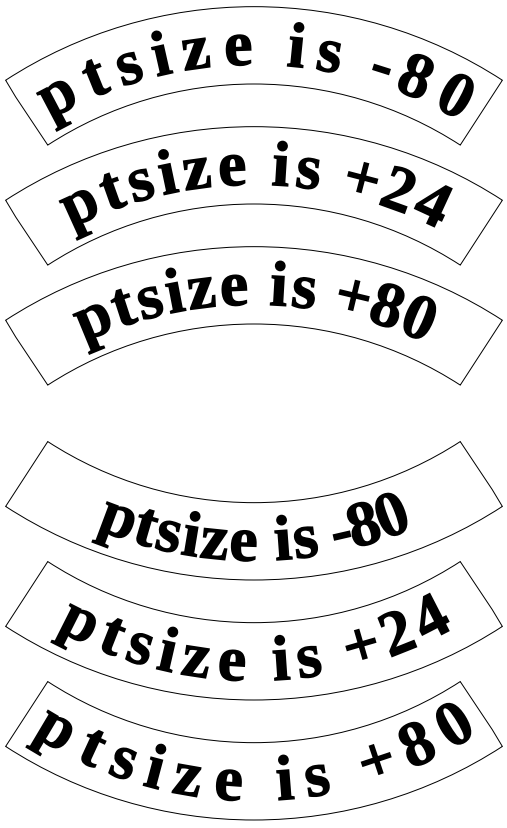


Fig. 4 - Using ptsize can get very tricky.

Any Secrets Involved in Setting Circular Text?

One or two. First, be sure you have on hand copies of both the *Postscript Reference Manual*, and the *Postscript Tutorial and Cookbook*. These

(*continued on page 18.5*)

Don Lancaster's

ASK THE GURU

July, 1986

A dual IIe monitor
Great mouse surfaces
HIRES Applewriter dump
Desktop publishing secrets
Postscript Puss De Resistance

I have just gotten back from a several day visit with the folks at *Apple Computer*. And while there's lots of great stuff I am not allowed to talk about, I think its safe to tell you this:

First, and foremost, there has been a total and complete turnaround in the Apple II division ever since they flushed old what's-his-name a year ago. Many aggressive new Apple II products are most certainly in the works.

Apple is now going out of their way to work closely with user groups and is actively seeking input from both users and hackers. And tech info is becoming much more available, both through the monthly technical notes, and user group access to their newly expanded Applelink BBS system.

Second, many of the new II family products will now strongly support an open and fully expandable architecture.

Third, and as much as is possible, all of the newer II products will be able to run older and existing II software. Apple has gone to extremes to maintain very strict upward compatibility.

Fourth and finally, some definite upgrade paths will be provided. These might take the form of a board swap for the IIe or else a very substantial trade-in allowance for a II or II+. And that is for everybody, not just for the schools as is their current policy.

Beyond that, I will let you know what I find out just as soon as I am legally allowed to. With *Computer Shopper's* tight publishing schedules, you will hear all about it right here first.

So stay tuned.

As per always, this is your column, and you can write or call per the end box. Besides lots of freebies, the latest two products are an upgraded *Applewriter/Laserwriter Utilities* and my new *Apple RAM Card Disassembly Script*.

Let's start off with a great find . . .

What is the Ultimate Mouse Surface?

Some Tucson cave divers have put me onto the ultimate mouse working surface.

Besides being cheap and easy to get, it beats just about all the commercial products whiskers down.

So, run on down to your friendly neighborhood divers supply or scuba shop, and get yourself some 1/8 inch or 1/4 inch nylon *wetsuit material*.

The cost is around a dollar per square foot, and you use it fuzzy side up. It comes in decorator colors. You can cut it with plain old scissors.

There are lots of styles available. The best one I have found so far is a 1/4 inch thick material with a bright blue working surface. The back side has a no-skid "fish scale" pattern on it.

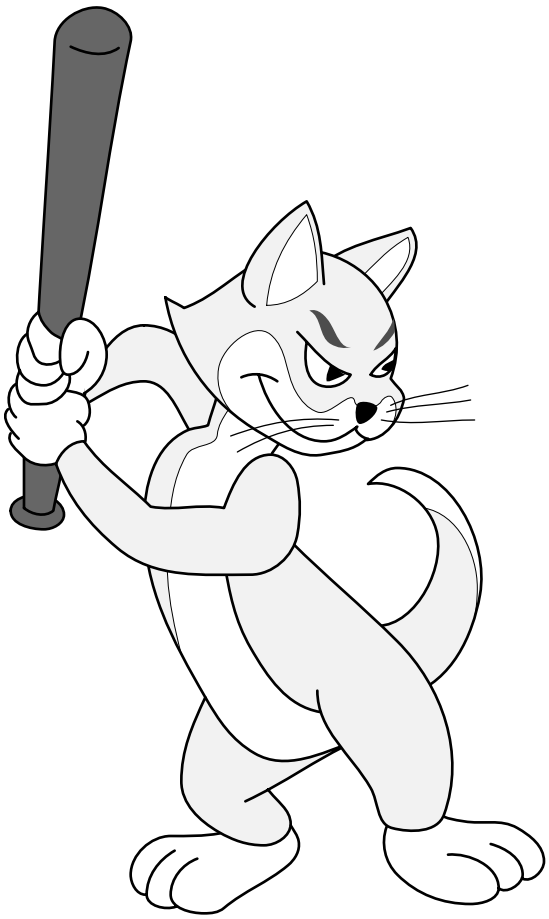


Fig. 1 - A typical AppleWriter text file.

What is so Special About the Laserwriter?

I receive an unbelievable number of calls on this one, so let's repeat ourselves one more time:

Those knockoff imitation laserwriter ripoffs are not even in the same league. In fact, most of them are an outright joke.

The Laserwriter has an internal full page and full resolution bitmap giving you unlimited text and graphics mixes in any combination in any drawing order. The knock- offs do not.

The Laserwriter has an incredibly powerful *Postscript* typesetting language built in that easily handles full page mixed text and graphics descriptions. *Postscript* is *100* percent compatible with many "real" typesetting machines, so you can instantly upgrade your output to 2650 DPI at any time with no change in textfiles or applications packages. The knockoffs do not even dream of this.

Postscript often lets you dramatically speed things up. For instance, with *Postscript*, you put your form letter *into the printer* only once, and then simply erase and reprint the name and address each time. Which is ridiculously faster than resending the entire letter for every name on the list.

The Laserwriter has some extremely flexible and built in font machinery. Any single font can be shown in any size from 3 point to 65,000 point in any direction or along any arbitrary path and put down in any order anywhere on the page in any of many different styles. Since many of these fonts use *analytical* descriptions, as the fonts get bigger, they get *smoother*, unlike the "Hershey Bar Effect" typical of the clones.

The Laserwriter is also, mercifully, free of any plug-in cartridge scams.

Besides the hundreds of ready-to use Laserwriter fonts available today, it is a simple matter to create and download custom fonts of your own.

The Laserwriter includes a fantastically powerful internal computer. In fact, it is the biggest and most powerful computer that Apple builds.

Because of this, any old word processor on most any computer can easily control the Laserwriter without any overhead hassles or excessive CPU time. Many knockoffs will totally tie up the host computer to do such essential laser printing tasks as justifying characters.

The Laserwriter has a built-in *cubic spline* ability that easily lets you do free-form curves, super smooth typography, or virtually stepless curves of any type.

Automatic signatures are particularly fast and easy with cubic splines.

Check out figure one for a totally different curve tracing example. To put my money where my mouth is, I'll give you a free *SAMS* book if you can show me how to print this same figure *any* way on *any* of the imitation laserwriter knockoff clones.

UPDATE: No takers!

How do I get Started In Desktop Publishing?

It's been said before, but the power of the press lies in owning one.

I guess I have been doing some personal publishing for almost a year now. It sure is nice to be able to send in "camera ready" artwork for my stories and columns and know that what I send in will be *exactly* what appears in print.

Its even nicer to be able to do all of my own ads without having to tangle with grossly overpriced printers, incompetent ad agencies, misguided copywriters, and so on.

So is that ability to print letterheads, labels, and so on "on demand" as they are used, rather than tying up money in horrendous minimum orders.

Best of all is the ability to make a "small change" at the last minute without any cost or hassle. Run that one by your local printer.

And Bee has really gotten into custom printing in a big way, doing bumperstickers, badges, logos and letterheads, data sheets, greeting cards, menus, batik patterns, certificates, business cards, and whatever for local people.

In fact, the only negative thing I've noticed is how the *Laserwriter* absolutely and totally dominates your lifestyle once you have one. This beast is clearly and obviously both physically and psychologically addictive.

Chapters of "Laserwriters Annoynmous" are bound to start cropping up everywhere, followed by *Laserwriter* rehab centers and withdrawal clinics. A toner junkie is not a pretty sight to behold.

Anyway, how do you get started on all this?

As with any field, reading the trade journals, magazines, and reference books are the best place to start. And, wonder of wonders, many of the personal publishing information sources are absolutely free of charge.

Let's see. First you should start off with my free laser printing demo pack which you can get by either calling or writing.

After that, there are two "must have" books. No, I did not write them, but I hope to contribute to the same series in a big sort of way.

These two are called the *Postscript Reference Manual* and the *Postscript Tutorial and Cookbook*. We do try to keep both of these *Addison-Weseley* titles in stock at all times.

As we saw in the previous question, important examples of *Postscript* printers are the *Laserwriter Plus* and the *Allied Linotron 300*. A typical example of a non-Postscript speaking printer would be the Model 28 *Teletype*.

Back to the literature. The

```
% Meowwrrr - the Postscript puss de resistance

initgraphics 1 setlinecap 1 setlinejoin /tension 2.8 def /ctf {curvetrace gsave 0.95
setgray fill grestore 2 setlinewidth 0 setgray stroke}def /ctfd {curvetrace gsave 0.4
setgray fill grestore 2 setlinewidth 0 setgray stroke}def /ct {curvetrace stroke} def
/showtick false def

% tail
[310 262 -10  360 330 100  308 368 -170 308 368 45  330 380 0 392 285 -100 340 210
-160] curvetrace gsave clip newpath [340 200 25  390 290 90  350 350 170 300 260 0
340 200 -70 ] curvetrace gsave 0.95 setgray fill grestore 0.5 setlinewidth stroke
2 setlinewidth grestore 2 setlinewidth  stroke

% left foot
[322 111 -110  330 75 -45  385 53 0 406 65 90 368 90 185] ct [406 65 -45  424 75 90
400 98 165] ct [424 72 -25  440 85 90  420 110 150 380 115 170  360 122 135] ct

% right leg
[216 130 25  298 175 25 298 175 -160  223 166 150  186 208 120 186 208 -115  168 145
-90  196 107 -30 230 97 0 250 113 90  233 139 125] ctf

% right foot
[180 87 -160  160 60 -90  185 46 0  200 50 40 248 82 40 256 100 100  250 112 120 ] ct
[164 51 -160  142 60 90  163 93 30] ct [141 60 180  130 75 90 160 104 0 197 106 10] ct

% left bod
[270 213 -90  310 140 -60 326 110 -15  365 120 70  330 250 135  250 325 135 200
310 -80  220 245 -60 272 197 -10] ctf

% upper bat
[60 480 -30  90 480 78  130 700 78  109 725 170 73 700 -93  60 480 -93] ctfd [85 696
-25  121 695 35] ct

% lower bat
[40 348 90  60 360 0  80 345 -90  65 334 180 40 348 90] ctfd [50 350 -40  72 348 40
72 348 80 78 377 80 50 387 -90  50 350 -90] ctfd

% right arm
[140 487 0 199 435 -70  173 410 120  140 445 180 100 430 -150 108 465 55  140 487 0] ctf

% right fingers
[56 482 -170  50 448 -45 65 442 0  83 461 85] ct [47 453 -120  50 430 -40 65 428 0 84
450 105] ct [61 497 -80  70 482 -45 84 470 -90  62 459 150 54 474 115  61 497 35]
curvetrace gsave 1 setgray fill grestore stroke [95 436 30  112 465 90  92 492 180
77 465 -90 83 450 0 98 467 80] curvetrace gsave 1 setgray fill grestore stroke

% left fingers
[80 386 -150  50 388 135  40 406 60] curvetrace gsave 1 setgray fill grestore stroke
[46 402 135  38 424 45] ct [44 443 -150  44 418 -40] ct [96 400 80  90 415 150] ct
[95 412 40  98 430 110  82 438 170] curvetrace gsave 1 setgray  fill grestore stroke

% right bod
[186 208 120  140 340 90  172 408 25 188 412 20  188 412 75  195 435 75 ] curvetrace
gsave 240 440 rlineto clip newpath [176 210 105  158 330 85  172 390 60  195 405 85
210 440 85 100 400 0  100 200 0] curvetrace gsave 0.95 setgray fill grestore 0.5
setlinewidth stroke grestore stroke

% left arm
[74 383 -70  120 320 -40  220 300 0  255 330 80 255 370 100 235 390 180  200 350
-110  200 350 170 145 351 170  145 351 135  95 400 125  95 400 180 74 383 -110] ctf

                                                % listing continues . . . .
```

Fig. 2A - Postscript Listing for Meowwrrr - The puss de resistance.

```
% continuing Meowwrrr . . . .
% . . . . . .

% left ear
[270 530 40  340 560 20 340 560 -55   312 490 -120] curvetrace gsave 0.4 setgray
fill grestore stroke [300 520 45  338 553 30 338 553 -70  312 504 -130] curvetrace
gsave 1 setgray fill grestore 0.5 setlinewidth stroke 2 setlinewidth

% head
[156 508 -80  228 406 -30  274 390 10  306 401 40 306 401 0  335 432 80 335 432 95
325 447 125 325 447 70  320 508 120  238 540 -160  170 500 -160 170 500 145  156
508 145] curvetrace gsave gsave 0.95 setgray fill grestore clip newpath [210 418
140  196 455 80  235 480 -40  254 437 -60 270 422 -20 298 430 -80 318 430 60  326
433 -50 318 404 -150 260 370 180] curvetrace gsave 1 setgray fill grestore 0.5
setlinewidth stroke grestore  stroke

% mouth
[213 448 25  240 445 -35] ct [226 450 -75  275 400 0 300 411 35] ct [300 411 -155
308 402 0] ct

% right ear
[223 501 150 262 580 30 262 580 -60  275 518 -85] curvetrace gsave 0.40 setgray fill
grestore stroke [232 502 85  262 570 50  262 570 -70  270 520 -90 270 520 -135  232
502 180] curvetrace gsave 1 setgray fill grestore 0.5 setlinewidth stroke 2
setlinewidth

% nose
/tension 3.8 def [300 428 60  315 425 -90  302 413 180  300 428 60] curvetrace 0
setgray fill /tension 2.8 def

% right eye
[255 482 -45  295 450 -15 295 450 -120  272 442 160  262 474 80] curvetrace gsave 1
setgray  fill grestore stroke

% left eye
[310 450 60  329 470 60 329 470 -95  321 450 -150 310 450 150] curvetrace gsave 1
setgray fill grestore stroke [318 454 70  329 462 30 329 462 -100  326 453 -90 326
453 -170  318 454 150] curvetrace fill

% eyebrows
0.3 setgray [264 498 20  280 488 -60 296 471 20 296 471 -160  280 473 150 264 498 155]
curvetrace fill [312 468 60  327 492 75 327 492 -70  328 482 -90 328 482 -120  312 468
-160] curvetrace fill 0 setgray

% right eyeball
[277 445 100  280 458 70 280 458 -30  291 450 -20 291 450 -155  279 445 -160]
curvetrace fill

% whiskers
[210 392 30  282 412 0] curvetrace 1 setlinewidth stroke [205 404 20  286 416 -5]
curvetrace stroke [318 416 -10  388 416 0] curvetrace stroke [322 422 15  385 431
10] curvetrace stroke [325 427 25  383 442 20] curvetrace stroke

% neck fill
2 setlinewidth  236 391 moveto  241 398 lineto stroke

showpage


%  --- end of listing ---
```

Fig. 2B - The rest of Meowwrrr.

best magazine specifically for personal publishing was Tony Bove's *Desktop Publishing*. This has since gone big time as *Publish!* It is now put out by the *PC World* people.

There are two free house organs you'll most definitely want to pick up. One of these is *Colophon*, which is edited by *Adobe Systems*. This gem has far and away the most outstanding examples of laser printing art available today.

The second freebie house organ has to be both the most fantastic and the most bizarre magazine ever published anywhere. It is called *U&LC*, and is published by the *International Typeface* people.

You are supposed to sound like a commercial artist or art director when you request your free copy.

Three trade journals that I have found rather useful are *Printing Impressions*; *Paper, Film, and Foil Converter*; and *Electronic Publishing*.

How can I have Two Monitors in a IIe?

Here's another question that keeps coming up over the help line. To recap: That so-called enhanced IIe monitor is incapable of running much of the older II+ and early IIe software. Worse yet, the dealer doing your upgrade will try to steal your existing monitor and character generator ROMs from you when the upgrade is done.

I have shown you one solution in a previous column. What you can do is program a "double-wide 27128 EPROM for each of the CD and EF monitor ROMs, and similarly program a "double wide" 2764 EPROM for the character generator. A changeover switch on each socket then flips you from old to new.

But there is now a better way. *Computer Accents* now makes a small plug-in board that fits your IIe and accepts both the old and the new character generator and the old and new monitor ROM pairs. The product is called the *Switchback* and sells for $59.95 plus $2.50 shipping and handling.

To use this board, unplug your present ROMS, plug in the switchback module, and then plug in both your old and new ROMs.

This method is far easier than programming your own EPROMs. There is both an internal and an external changeover switch available.

The only little gotcha is to make absolutely sure your dealer does not try to steal your old chips when he does your upgrade. Two ways of preventing this are to do the upgrade yourself using the Apple kit, or else remove both monitor chips and the character generator *before* you take your IIe in.

How do I do an Applewriter HIRES Dump?

And yet one more repeat question that's good for at least four help line calls a day. The easiest way to do a HIRES dump with Applewriter is with *post processing*.

The big advantages of post processing are that it works with just about any mix of printer, printer card, and graphics dump software; and that it takes no program mods and needs little in the way of programming smarts.

To do a HIRES dump, put some magic marker in your Applewriter file where you want the dump to take place. A suitable marker might be a carriage return followed by an "!hdMYPIXFILENAME" or something similar. Then do a magic PD8, and print your fully formatted file to disk.

Finally, you can read your PD8 formatted file not with Applewriter, but with Applesoft. Have Applesloth read one line at a time. If there is no magic marker present, then print the line as is. If a magic marker is found, load your picture, activate your HIRES dumper and print the picture in its intended place.

When finished with the dump, keep reading textfile lines one at a time and then printing them.

If things seem a tad slow, get the process working anyway. Later on you can speed things up with a hard disk, a RAM card, a total load, or some machine language code.

The same post processing idea works well for wall to wall microjustification for printers that do not have this feature built in. It is also good for such things as soft hyphens, sticky spaces, to prompt for daisywheel swaps in mid-line, and so on.

How can I draw Cartoon Characters?

Why, with *Applewriter* on an Apple IIe, of course. How else could you possibly draw decent cartoon characters?

Figure one shows you a typical cartoon character, as drawn by the Applewriter word processor on the *Laserwriter*. The text cartoon file ends up the size of a business memo, mostly since I did not bother compacting it much.

As you can see, you can very easily draw outstanding cartoon characters using Applewriter on an Apple IIe.

Also as you can clearly observe, the quality of the Applewriter and *Postscript* graphics is absolutely superb.

Oh, yes. You do need a digitizer. A safety pin makes a very good one, but if you can not afford such expense, just use a needle instead.

Although little known as a computer peripheral, a safety pin digitizer easily will outperform just about any other personal computing input digitizing device.

Here is how you use your safety pin digitizer: Firstoff, make your original as big as possible, using an enlarging copy machine. Note that you can easily reposition, rotate, scale, stretch, remap or distort your final *Postscript* file most any which way but loose.

(*continued on page 19.5*)

Don Lancaster's

ASK THE GURU

August, 1986

Tech consultants list
Toner cartridge secrets
Flushing copy protection
Postscript window decals
Hex and Decimal to ASCII

Hats off to both the *PFS Software* people and to the crew at *Dollars and Sense*. Word has it that both these outfits have completely dropped all copy protection from their software.

Actually, more and more software houses are waking up to the obvious facts that copy protection (1) does not work, (2) hacks off users, (3) creates all sorts of hard disk hassles, (4) dramatically *increases* the number of bootleg copies in circulation, (5) diverts resources from useful tasks, (6) costs money, and inevitably, (7) loses sales.

If *you* are writing commercial software, there's two key secrets to coping with bootlegging and piracy:

The first is to not sweat it, since actual cash lost out of your pocket is much smaller than you might first imagine. Consider the average software collector or trader who has, say, a mere 10,000 bootleg programs in their primary trading stack.

Even if he stopped trading for a while and actually *used* one of his stolen programs, the chances are overwhelming it wouldn't be yours anyway.

The second big anti-piracy measure is to include as much personal value added in your software products as possible. At the very least, this must include a free and courteous unlimited helpline service, a "no questions asked" return policy, and a $5 no-hassle exchange for blown or broken disks.

You should also concentrate on those parts of your package that are *not* on disk, such as tutorials, guides, manuals, reference cards, user aides, and anything else that puts the perceived value of your product on the overall package rather than on disk.

For lots more on personal value added, check into my *The Incredible Secret Money* Machine book.

Turning to other news, I keep hearing persistent rumors that future Apple products might not be compatible with devices that plug into that existing "slot zero" IIe memory expansion slot. Thus, any third party slot zero memory expansion cards might not be usable after an upgrade.

A related rumor (and some plain old common sense) says that cards that plug into slots 1-7 might get into very deep compatibility trouble unless they slavishly make exact use of the I/O SELECT and I/O STROBE lines. Should future address spaces end up wider than 16 bits, the partial decode as done by some cards could cause all sorts of contention problems.

This might get very sticky on multifunction peripheral cards that do a "phantom slot" decoding.

On to this month's vast collection of goodies . . .

Where do YOU go For Technical Help?

Why, to the *Gila Valley Apple Growers*, of course. You will find no finer collection of technical expertise available anywhere, ever.

Tellya what I'm gonna do. Figure one is my super secret list of where I go to get my own technical help.

Those people with (602) area codes are charter members of the GVAGA; those without are honorary members but members none the less.

Most of these people are willing to listen to your help requests. Provided, of course, that you extend the usual courtesies in the same way that most of my helpline users already do.

Ferinstance, be certain to check the area code against the time zone, and never call before 8 am or after 9 pm, *local time*. Remember that all of the California clocks are laid back.

Do not ask for or expect a return call, unless the tab is on you. Keep the call length under ten minutes max.

And, if the call is useful to you, pay for it somehow. Perhaps by buying one of their products, or by using one of their services. At the very least, lay some original software on them or else send them some local produce, nuts and berries, or whatever.

Play fair and everyone will win and win big.

How do I Convert Decimal to ASCII?

There are lots of times and places in microcomputer software when you might like to convert a decimal value to its printable *ASCII* equivalent, or vice versa.

For instance, you might have the decimal score of a game stashed somewhere and you might want to route it to a video screen.

Or, you might want to show the number of characters already used or still available in a word processor.

Decimal numbers are usually represented by four bit bytes with 0000 = 0, 0001 = 1, 0010 = 2 . . . up to 1001 = 9. One byte is used for each decimal decade. One 4-bit byte coded this way is called *BCD*, short for *Binary Coded Decimal*. Each byte holds one decimal digit.

Figure two shows us how to get between *BCD* decimal and ASCII, which is the standard character code.

The rules are simple: To get from BCD to ASCII, just add decimal 48 or hex $30. To get from ASCII to BCD, just subtract decimal 48 or hex $30. And that's all there is to it.

Sometimes, a pair of BCD bytes might be combined into an 8-bit word. This is called *Packed* BCD. If you are using packed BCD, you must first unpack the bytes before ASCII conversion, and then later repack them after being converted from ASCII.

To unpack a low BCD byte, logically AND it with hex $0F. To unpack a high BCD byte, shift or rotate the word to the right four times, and then AND it with hex $0F.

To pack a BCD word, just shift the high BCD byte to the left four times and then OR it with the low BCD byte.

Several add-on provisions might be needed for your software if you want to right justify the number or suppress leading zeros. Zero suppression is done by replacing a $30 zero with a $20 space.

The 6502 microprocesor has a special *decimal mode* that works directly in packed BCD. Other micros usually do have ways of faking something similiar. These days, though, you usually do *not* pack the BCD decades.

How can I Convert Hexadecimal to ASCII?

Things do get a tad more complicated when you try to convert between hexadecimal and ASCII.

You might want to do this when you are displaying a "hex dump" of a computer's monitor program on a screen. Some stand-alone EPROM burners also require their hex bytes be passed back and forth as ASCII characters, as does any bit image information for laser printing.

There are both advantages and a penalties to converting hexadecimal to pairs of ASCII characters before sending it somewhere else. The single

ACADEMIC COMPUTING:	
Ed Hunter	(602) 428-1133
ANIMATION:	
David Meyer Sr.	(801) 359-2790
APPLE II:	
Tom Weishaar	(913) 649-0567
APPLE III:	
Val Golding	(818) 708 2382
ASSEMBLY:	
Bob Sander-Cedarlof	(214) 324-2050
CABLE TV:	
Phil Nelson	(602) 428-1850
PC CLONES:	
Sherry Romberger	(602) 526-8512
C PROGRAMMING:	
Henry Schneiker	(602) 325-5884
DISASSEMBLY:	
Bob Kovacs	(201) 325-1885
EPROMS:	
Bob Gardner	(206) 337-2370
IMAGE PROCESSING:	
Charlie Springer	(415) 322 7373
INTERFACE:	
John Bell	(415) 592-8411
MECHANISMS:	
Claude Martin	(602) 428-4830
OPTICS:	
Forrest Mims	(512) 372-0548
POSTSCRIPT:	
Liz Bond	(415) 961-4400
PROGRAMMING:	
Gary Crockett	(602) 428-3256
SIGNAL PROCESSING:	
Roger Cox	(303) 641-4240
SOLAR ENERGY:	
Jim Allen	(602) 428-6993
SPEECH SYNTHESIS:	
Bruce Fette	(602) 834-3370
WPL and APPLEWRITER:	
Don Thompson	(714) 855-3838
65808 and 65816:	
Will Troxel	(619) 672-0670

Fig. 1 - My super secret technical consultants list.

greatest advantage is that everything is sent as ordinary numbers or letters. There is thus no possibility of confusing a hex $0C for a formfeed, or interpreting a hex $0D as a carriage return.

Illegal characters cannot be sent since there aren't any. As a further bonus, an ordinary word processor can also be used to process hex characters in their ASCII form.

The penalties of hex to ASCII conversion do involve both speed and storage size. You'll need *two* printable characters to show a single 8-bit hex word pair. Thus, it will take you *twice* as long to send converted bytes over a serial interface than it would to send straight hex.

It also does take twice the space to store hex values in ASCII form. Which gets particularly nasty when you laser print video or photographs.

To review, a four bit byte can be shown in hexadecimal. There are sixteen possible 4-bit states. The first ten states 0000 through 1001 are shown as decimal digits, exactly as is done with BCD. The final six states are often shown as uppercase letters, with $1010 = A, $1011 = B, on up through $1111 = F.

Figure three shows us how to do these conversions. To get from hex to ASCII, add decimal 48 or hex $30 if the number is *nine or less*. If *ten or more*, add decimal 54 or hex $36.

To get from ASCII to hex, you first have to test to find out if you are converting a valid letter or a numeral.

Such *range checking* is always a good idea anyway. If you have a numeral (CHR$ 48-58 or hex $30-39), then subtract decimal 48 or hex $30. If you have a letter (such as a CHR$ 65-69 or a hex $41-46), subtract decimal 54 or hex $36 instead.

The actual conversion details will depend on your programming style and on the microprocessor and language that is in use. In the *Apple IIe* monitor, there are several built-in machine language routines for hex to ASCII and ASCII to hex conversion.

Specifically, $FDE3 will convert the low hex accumulator byte to ASCII. $FDDA will convert the entire accumulator to a hex ASCII pair. $F941 will print first the X register and then the accumulator as four successive hex digits.

Finally, $FFA7 will reverse the process and convert the ASCII characters in $003E (low) and $003F (high) into a two-byte hex value.

You can tear apart this code for a quick study on conversion details. The innermost secrets of easily and rapidly tearing apart machine language code appear in my *Enhancing your Apple II*, volume I (SAMS #21822).

0	0000 - 0011 0000	$30
1	0001 - 0011 0001	$31
2	0010 - 0011 0010	$32
3	0011 - 0011 0011	$33
4	0100 - 0011 0100	$34
5	0101 - 0011 0101	$35
6	0110 - 0011 0110	$36
7	0111 - 0011 0111	$37
8	1000 - 0011 1000	$38
9	1001 - 0011 1001	$39

Fig. 2 - Decimal and ASCII equivalents.

Tell me All About Toner Cartridges

The *Canon* laser engines, such as the *LaserWriter* or *LaserJet* use a plug-in cartridge that includes toner, the photosensitive drum, corona charging, and includes just about everything major that is likely to need servicing in a xerographic copy machine.

These cartridges cost from $65 to $100 each and are only good for 2500 copies.

Worse yet, there are definite quality control problems, and not every cartridge is usable over its whole life.

Every once in a while, a cartridge flat out refuses to give decent blacks, particularly over larger areas.

What can be done to both lower cartridge costs and improve the output quality?

First, note that you can NOT and must NOT use an ordinary copier toner cartridge or ordinary copier toner. Besides mechanical interlocks that prevent you from doing this, copiers use a *positive acting* toner (light = white), while laser printers use a *negative acting* toner (light = black).

Second, that box means exactly what it says on that 104 degree Farenheit max temperature warning. Should cartridges scunge around in a hot warehouse or in a sunny truck, their quality will rapidly deteriorate. So, be sure to cool it with all your toner cartridges.

Third, be sure to often do the routine maintenence. The corona wire inside the toner cartridge must be wiped every so often with the little green wiper tool. The corona wire inside the laser printer needs cleaning every now and then with a Q-tip. Be very careful to not stress the wire when you do this.

It is also a good idea to keep a solder sucker or childs rectal syringe on hand. Every

time you open the machine up for whatever reason, be sure to dust things off thoroughly.

Fourth, keep that fusion wiper pad clean. It pays to lightly scape it with a knife every 500 copies or so. And be sure to change it when you change the cartridge.

Fifth, rock the cartridge every now and then to redistribute the toner.

Sixth, don't try fighting the drum's life cycle. Fresh cartridges are usually somewhat too badly washed out for the first few hundred copies. In fact, most drums do not get up to their best blackness until several thousand copies *after* they first run out of toner!

So, always keep two or more cartridges on hand. Run the first one until it starts putting out solid blacks. Then immediately switch to the second one for all the rough copies, internal memos, and anything else that does not need a super-black output.

Use the old cartridge for quality work and the new one for anything non-critical.

Seventh, pick your paper carefully. Some papers give excellent results, some marginal, and some others are totally unusable.

For everyday use, the plain old xerox paper from your local price club at $1.80 a ream should work ok. For the blackest black on the whitest white, the best paper I have found is *Classic Crest Avon Brilliant White*. This is made by *Neenah Paper* and is now available through most paper distributors. We get ours from *Ingram Paper*.

Eighth, yes you can repair cartridges. About one in six or so will fail some *other* way than running out of toner. It may be permanantly gray, or have bad shading, or may have a scratch, a ghost, or a dropout. If this happens, you can move the toner from the sick cartridge over to a good one that just ran out of toner.

And finally, yes you can easily reload the cartridges yourself. A cartridge can be reloaded three to five times at a cost of $20 per reload or less. One supply source is *Laser Printer Products*, but dozens of others are showing up. I've found their toner to be somewhat blacker and denser, but also somewhat less crease and scrape resistant than the original.

Present pricing is $20 for one toner reload, and $1.50 for a new wiper pad. They will also buy old cartridges outright for $10 each.

I have yet to discover the original source for bulk toner. Let me know if you find it.

How can I Reload a Toner Cartridge?

Few people are aware that there is a snap-on, snap-off filling plug in the toner cartridge. The reason they don't realize this is that it takes a special screwdiver, some carving, spring popping, shaft removing, getting fingerprints and light on the drum, and enough other hassles that this filler cap is simply not worth accessing at all.

Why, it is almost enough to give you the impression that *Canon* might not be totally in favor of letting you do 1/5 cost reloads.

Reloading is fairly simple, though, if you forget about the plug. Just add your own filler hole in a sane place.

Remove the large cardboard label, popping it off from the middle with a dull knife. Then, scrape a 5/8 inch square hole in the top end of the inside toner tank, using a plexiglass cutting tool. Pour the toner into the tank using a plastic funnel, shaking gently as you go along. Then tape the hole shut.

Please send me your toner tips and horror stories so they can be passed on. And be certain to return the defective cartridges so the quality control people at *Canon* can clean up their act.

And, should you have any opinions on a poor design that purposely goes out of its way to make a refillable cartridge unrefilliable, you might mention this to them one way or another as well.

UPDATE: Feb 87.

0	0000 - 0011 0000	$30
1	0001 - 0011 0001	$31
2	0010 - 0011 0010	$32
3	0011 - 0011 0011	$33
4	0100 - 0011 0100	$34
5	0101 - 0011 0101	$35
6	0110 - 0011 0110	$36
7	0111 - 0011 0111	$37
8	1000 - 0011 1000	$38
9	1001 - 0011 1001	$39
A	1010 - 0100 0001	$41
B	1011 - 0100 0010	$42
C	1100 - 0100 0011	$43
D	1101 - 0100 0100	$44
E	1110 - 0100 0101	$45
F	1111 - 0100 0110	$46

Fig. 3 - Hexadecimal and ASCII equivalents.

How Can I get Filthy Rich This Month?

Naturally, you can use my *Applewriter/Laserwriter Utilities* and *Applewriter* on a IIe driving the *Laserwriter*.

Surprise, surprise.

The scam this month is -- window decals!

You can easily create professional, *full color* window decals that show store hours, giant letters, etc. *through* a window or glass door. You can do so in a few minutes at a tiny fraction of the price a sign shop charges. Being all on the inside, the decals are pretty much immune to most vandalism.

Figure four shows you a typical decal as printed, while figure five shows us the way it appears when you view it through the glass.

The key secret lies in the Laserwriter's ability to print backwards. To do reverse printing, just send out a -1 1 SCALE 612 0 TRANSLATE command. You then print on transparent, self-stick acetate.

Now for the sneaky full color gimmick. Say you want the HOURS to be red and the rest of the decal blue. Mask all but the HOURS and spray it with red paint from the hobby store. Then, spray everything blue. Next, add a second protective acetate overlay. Finally, apply the decal to the inside of the window by the usual peeling and sticking.

UPDATE: Mar 87.

When viewed through the glass, you will see frontwards reading black 8-6 and DAILY on a blue background, a black border, with HOURS in red.

All quickly and at an utterly negligible cost, when compared to traditional sign shop methods. And in virtually unlimited colors.

You can also create vinyl letters, and most any commercial sign most any way you like. It should also be possible to get the toner to interact with *Scotchcal* photosensitive vinyl, but this is one exciting area I simply haven't gotten around to yet.

One hint: If the chosen material will not accept toner well, then print backwards on the backing strip instead, and then cut things to size with scissors.

Practically any larger paper distributor can get you bulk materials, but the best source I have found for any small or experimental quantities of all the self-stick acetate, mylar, vinyl, or fluorescent bumper-sticker material is *Dick Blick*. They have several catalogs. The one that you want is for signmaking and silk screen printing.

Call or write, and I'll be happy to lay some free window decal samples on you. Plus, of course, the usual free laser demo pack, and the framable picture of *Meowwrrr* from last month's column.

By the way, *Meowwrrr* has now been traded to those *Tigers*. You simply will not believe his new uniform.

Yipes. Stripes.

Fig. 4 - You print a window decal this way to . . .

Fig. 5 - . . . view it this way.

(continued from page 16.4)

are written by Adobe and are published by *Addison-Wesley*. I do try to keep extra copies of both in stock at all times.

Second, there is a *ptsize* command that is included in Adobe's *insidecircletext* and *outsidecircletext* routines that can cause all sorts of confusion if you do not exactly understand what it is and what it does.

The actual horizontal and vertical point size of your circular text is set by the font definition you create ahead of time. Thus, 21 point type will always appear on the circle as 21 point type.

What *ptsize* does is set the *spacing* of the chosen sized characters around the circle. Figure 4 shows you details.

The troublesome part is as follows: On the *insidecircletext* routine that would go on the bottom of the badge, the *ptsize* command works exactly as you would expect it to. The bigger *ptsize*, the further apart the letters. A good starting point is to use a *ptsize* somewhat larger than your actual point size.

The puzzlement comes in when you try to use *ptsize* on the *outsidecircletext* as you would use at the top of a badge. The larger *ptsize* gets, the closer together the numbers get. And a zero *ptsize* will only slightly spread the letters apart. What to do?

Use *negative* values for *ptsize* on the top of the badge. The more negative the number, the further the spread of the characters along the top of the badge.

Summing up: the font definition will decide how high the letters will be. On the bottom of the circle, a bigger *ptsize* will spread the letters. On the top of the circle, a bigger but negative *ptsize* will spread the letters for you.

Once you have preloaded these routines, you can type circular text just as quickly and easily as you would type a plain line of characters.

Don Lancaster's

ASK THE GURU

September, 1986

Care of floppy disks
Repairing blown disks
Applewriter clock access
Toner cartridge reloading
Postscript surface mapping

If you are ever visiting the Boston area, be sure and see the *Computer Museum* and check out their incredible collection of historic computer goodies. They have just recently placed a few of my original prototypes from the *TV Typewriter* and *Decimal Counting Unit* days on permanent exhibit.

The big news this month is that Apple's new *Applelink* user group technical info services have gone on line at *CompuServe*. What you want to do is become an *ambassador* for your user group. This entitles you to some free connect time as well as access to insider info that is otherwise unavailable.

Among other goodies, all of the latest versions of all that *Macintosh* and *Laserwriter* system software are on line and downloadable, not to mention all the IIe technical notes. For more details on all of this, you may want contact Ellen Leanse over in Apple's evangelist group.

I have personally been a little slow to get into modems in a big way, since I live a very expensive distance from the nearest access numbers. But it is getting to the point today where you simply must have and use a modem. So much is out there now. And so much of it is free or very nearly so.

Hey, did you catch that July cover on *A+*? Of course, since you are a very faithful *Computer Shopper* subscriber, you knew all about this stuff long ago.

If you have not already done so, be sure and send for your free *Applewriter/Laserwriter* demo pack.

As usual, this is your column and you can get both technical help and rumors mongered per the end box.

And now, for our feature attractions . . .

How do I keep from Blowing Up a Diskette?

It seems I blew up a disk the other day, so now is as good a time as any to review how you prevent blown disk hassles. As with fire prevention, the cheapest and best way to fix a blown disk is to prevent it from blowing up in the first place.

When you have a diskette in your hands, there are two and only two allowable places to put it. And that's directly in the drive or directly and immediately back into its case or protective sleeve.

Setting a disk on the table and then being "very careful" never to touch it is incredibly stupid. You see, it is the *bottom* side of the disk that plays, so all that cat hair and

1. Make a backup copy of ProDOS Applewriter 2.0.

2. Get into BASICS.SYSTEM. Then CALL -151.

3. BLOAD AWD.SYS, A$2000, E$6030, TSYS

4. Verify:

```
2C97-  20
2C98-  43 35 20 F6 4C A0 00 A9
2CA0-  70 A2 56 20 4D 50 20 7E
2CA8-  22 AD 00 02 20 F7 3C CD
2CB0-  2C 5C D0 57 AD 4F 21
```

5. Change:

```
2C97-  A0
2C98-  3F A9 00 99 00 1E 88 10
2CA0-  FA A9 29 20 0B C5 20 08
2CA8-  C5 A2 00 BD 01 02 9D 00
2CB0-  1E E8 E0 0E D0 F5 60
```

6. Verify:

```
5DE1-  51 75 69 74 20 41 70
5DE8-  70 6C 65 20 57 72 69 74
5DF0-  65 72
```

7. Change:

```
5DE1-  44 61 74 65 2F 54 69
5DE8-  6D 65 20 20 3E 20 24 41
5DF0-  20 20
```

8. UNLOCK AWD.SYS

9. BSAVE AWD.SYS, A$2000, E$6030, TSYS

10. LOCK AWD.SYS

Fig. 1 - A ProDOS Applewriter 2.0 Proclock patch.

furniture polish is merrily doing its thing.

I personally *never* put a disk drive within 18 inches of *any* computer monitor or tv set. People who smoke, of course, must *never* be allowed in the same building as your diskettes, let alone in the same room.

Everybody knows they are supposed to back up their disks and do so very often. But it never ceases to amaze me how many people stupidly store the backup right beside the original! If the backups must stay in the same building, hide them as far away from the originals you possibly can. The backup disk is supposed to be just that. Besides blowups, there is theft, fire, flood, stupidity, and mischief to contend with.

I do use a notcher sometimes, but I would never use a double sided disk inside a single sided drive if the sum total of all files on both sides get spun more than ten minutes per month.

That same pad that picks up dirt on the one side ends up grinding that same dirt back into the other side when you flip over.

The casualness with which some people INIT their disks is unbelievable. INIT kills! You can take a diskette and run over it with a truck and then boil it in peanut butter and jelly and the chances are that you can recover most, if not all of your data. But it is all over once you start an init or a format.

Never press the carriage return following an init or format command. Instead, very carefully open all doors on all drives. Then remove the disk that is to be formatted. Put your finger on the write protect notch. Then carefully and out loud spell the label *backwards* one letter at a time.

You did write on the label before you put the disk in the drive, didn't you?

If you are copying a disk, always be absolutely certain to write protect the original before continuing.

Never turn power off or hit reset when the red drive light is on. Wait till the light goes off. If the light insists on staying on for more than a minute, open the drive door and then wait some more.

You should check your drive speeds at least once a month. Note that many speed checking utilities may destroy a track on whatever disk is in the drive.

On an older stock 5-1/4 inch Apple or u-Sci drive, the speed adjustment trimpot is at the lower right rear, and is *not* the big and obvious adjustment smack in the center of the top board!

A disk cleaning diskette should be used at most every three months or so. More frequent use probably does more harm than good.

Every now and then it pays to reseat all of those cards and cables. Use of a very modest amount of *Radio Shack* tv tuner cleaner at the card fingers and the expansion sockets is also a very good idea.

Note that this product has a dry film lubricant in it. It should never be sprayed all over the place, and should never be sprayed directly onto or near a disk.

What appear to be disk problems are often caused by some other card in some other slot. Keep only your often-used cards slotted and remove the others.

If you have lots of cards in use, keep things cool with a fan or by removing the lid.

Using oddball, orphan, or unusual drives and cards is generally never a good idea. Stick with mainstream drives, or else with second sources that have been around for a while.

How Can I "repair" A Blown Diskette?

Naturally, nobody pays any attention to preventing disk blowups until well after the blowup occurs. What can be done?

Firstoff, the more you know about your disk system, the better off you will be when disaster inevitably strikes. Some "must have" books include *Beneath Apple DOS* and the *Beneath Apple ProDOS* from Quality Software, *All About DOS* from A.P.P.L.E., and the *ProDOS Technical Reference Manual* directly from Apple.

Two major utilities that are almost indispensible for disk repairs are *Copy II+* from Central Point Software and the *Bag of Tricks II* from Quality Software.

OK. That nasty old I/O ERROR appears. Now what?

WPL Routine:

```
p  date.n.time
p  puts Proclock date and time into $A
p  .......
p
   qj
   f{{$A{
   y?
   pqt
```

Glossary entry:

```
d[P]dodate.n.time
```

Fig. 2 - Single key Applewriter to Proclock access.

Let's first assume that you did not just init or format the disk, permanantly destroying everything previous. We will assume that there is no obvious diskette damage. We will also assume a stock and unprotected disk format.

Your first attempt is to simply try again. Open and close the drive door the instant the light comes on. This should recenter the diskette. Should a file sometimes be readable and sometimes not, you have something flakey going on. Init a new diskette and move all of the files over. Do this one file at a time. Try again any time you get a read error.

If the error continues, try rebooting. Sometimes the DOS or the ProDOS operating system in memory might get trashed for one reason or another. Note that it is usually *not* a good idea to adjust drive speed at this particular time. A change in speed could only add to your present problems.

If that fails, try rebooting on someone else's system using a different disk drive. This will seperate any mechanical or system problems from actual disk damage.

Should you be unable to read the disk on another machine, the chances are there really is something wrong with it. At this point, you will want to decide how many hours of recovery effort you are willing to spend, compared against the value of the lost data.

The next thing to do is to copy protect the diskette and then make a second bit copy, preferably using the latest version of *Copy II+*. Any attempt to repair a diskette can possibly do permanant damage, so work only with this backup copy.

More often than not, the disk damage will happen to either the catalog or directory tracks, since these are the ones most often referenced and the ones that are easiest to foul up.

Init or format a new disk. Then use the INIT feature of *Bag of Tricks II* to copy the old files onto new and "clean" tracks, while preserving data. With my particular blowup, this was all that was needed. Should there be more problems, try using FIXCAT to correct the catalog. If that fails, use ZAP to inspect and study each track to find out what problems are where.

If all else fails, you can use a block or sector reading utility to strip off the files 256 or 512 bytes at a time, saving them to disk, and then recombining them into the proper file type. Note that the little known and little used ProDOS "TYPE" command is extremely handy for this sort of thing.

As a general rule, most files can be recovered. Naturally, the one file you need the most will be the one that is damaged beyond repair.

If the disk is physically damaged, then surgery may be called for. Do not attempt what follows until after you have tried everything else and have talked it over with a gonzo hacker.

To perform surgery, you will need a new diskette of the same brand and series as the damaged one. You will also need a pair of cotton gloves from the photo store, a sharp knife, and some tape.

If a diskette got run over by an office chair, eaten by the post office, or tricycled by a three year old, it is possible that the creases in the cover are slowing down the drive enough to create errors.

Cut the extreme top off both diskettes, so that you can remove the magnetic media itself. Inspect the media, and see if it is nice and round and shiny without any creases.

Then put the blown media into the new cover and see if it works.

If the blown media was smeared with peanut butter and jelly or something similar, get a can of freon cleaner from *Radio Shack* and carefully spray the gunk off. Be sure to use the type of freon cleaner that does *not* have a dry film lubricant in it. TV tuner cleaner is a no-no!

Do try and minimize the thermal shock to the media by spraying from a distance and

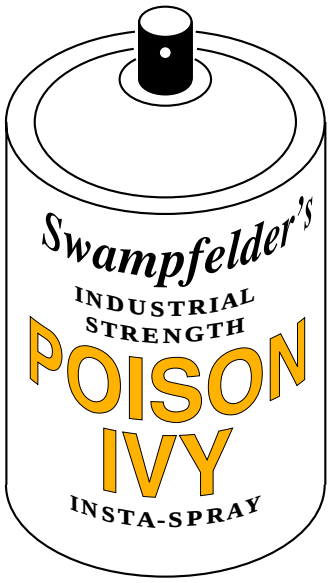


Fig. 3 - Just what you have been waiting for.

and by using the absolute minimum spray needed to get the job done.

And that should get you started in the wonderful world of blown disk recovery.

Nothing elevates you to a hero status any faster than "saving" someone else from their own stupidity.

While you are at it, how about sharing your diskette recovery horror stories and experiences with our other *Computer Shopper* readers?

Write or call, and we will put together the best stories and recovery secrets together in a later column.

How Can I Access a Clock Card with Applewriter?

Don Thompson of *Thompson and Thompson* has just put together a remarkable *Applewriter WPL Expansion Kit*. This jewel provides three major enhancements to ProDOS Applewriter.

The first one is automatic support for many major clock cards. These do include the *Timemaster*, *Thunderclock*, and *Proclock*.

Next is an expansion up to eight WPL string variables from the usual four. And the final enhancement does a two way expansion of those numeric variables, giving you six variables, each with a 24 bit resolution.

The normal price is $39.50, but Don has offered to provide *Computer Shopper* subscribers with a special one time price of $34.50. Be sure to use the secret password "tinaja" to receive this special offer. If you pronounce tinaja correctly, then Don will even throw in a bonus program of some sort.

As a compacted sample of Don's work, figure one shows you the listing for the patches needed to read the *Proclock* card. These mods overwrite the useless [Q]-J quit option. After patching, a [Q]-J will automatically put the time and date into the WPL $A string.

Figure two shows you a brief WPL routine and a glossary entry that will let you insert the time and/or date into your textfiles using a single keystroke.

Note that a space must preceed all lines in the WPL routine.

Contact Don directly for patches to other *Applewriter* versions, or to be able to show the date only.

And, of course, if you want to know more about this unmatchably superb word and *full graphics* processor, check into my *Applewriter Cookbook* (SAMS #22460).

Any More Info on Toner Cartridge Reloading?

I have been intentionallly pushing a *Laserwriter* toner cartridge to see how many reloadings it would take. On the fifth reload, at something like 17,000 copies, the cartridge sensitivity started to drop, and a persistent edge streak appeared.

```
% cylindrical text demo (Postscript)

/circradius 150 def /pstring ( ) def /alpha 0.5 def

/arg { oldrunningpos pstring stringwidth pop 2
div add circradius div 57.926 mul} def

/positioncalc { arg sin circradius mul /circposn
exch def} def

/forshortencalc { arg cos /fs exch def} def

/slopecalc { oldrunningpos circradius div 57.296
mul sin circradius mul circradius div alpha
mul /slope exch def} def

/wraponly {/intcharvalue exch def  pstring 0
intcharvalue put pstring stringwidth pop
oldrunningpos add kernstuff add /newrunningpos
exch def positioncalc forshortencalc pstring
gsave circposn fs circradius mul alpha mul neg
moveto slopecalc currentpoint translate
[fs slope  0 1 0 0] concat pstring stringwidth
pop 2 div neg 0 moveto fs 0 le {.99 setgray}
{0 setgray} ifelse show grestore /oldrunningpos
newrunningpos def} def

% .....

300 300 translate

/NewCenturySchlbk-Bold findfont [60 0 0 70 0 0]
makefont setfont

gsave
/oldrunningpos -140 def 0 103 translate
/kernstuff 3 def (POISON) {wraponly} forall
grestore

gsave
/oldrunningpos -65 def 0 36 translate
/kernstuff 5 def (IVY) {wraponly} forall
grestore

showpage
```

Fig. 4 - A Postscript cylindrical text demo.

As we found out last month, you get the blackest images and the best looking copies somewhere around 7500 pages. This will usually takes place *after* the *second* cartridge refilling.

Reloading, of course, will dramatically change the laser printer economics. If you pay $100 list for a cartridge and get 2500 copies out of it, you are talking four cents of per-sheet toner cost.

On the other hand, if you pay $60 wholesale for a cartridge and then do five reloadings at $18 per reload, your per-sheet toner cost drops to almost exactly a penny per sheet. Note that the first figure is well above that of jiffy printing, while the second is quite competitive.

But I suspect that the $18 refill cost may be a scam. Yes, you do have to use the special negative acting toner. But ordinary old toner costs around $4 for a comparable amount. What I think may be happening is that toner refiller supply houses are buying case lots of generic toner from a yet unknown source.

And toner costs of half a cent per page would turn the jiffy print industry on its ear.

UPDATE: Feb 87

What is the Feature Product for This Month?

Marcia Swampfelder sent me a press release on her major new product intended to solve once and for all any interpersonal relationship problems you may have had with computer salesmen, waiters, bureaucrats, car mechanics, admininstrators, or whomever. Figure three shows you this exciting breakthrough.

Note that there is usually no fear whatsoever of any reprisals, since the level of technical expertise needed to point the nozzle in the proper direction is well beyond the abilities of nearly all of your intended targets.

Price is $4.50 per 12 ounce can. A remote unit (for use with popular modems over standard telephone lines) is currently in devlopment. This should also revolutionize telephone courtesy. Besides putting a real crimp into phone solicitation, and it, mercifully, will eliminate music on hold once and for all.

Marcia strongly suggests wearing rubber gloves when using this product.

Seriously . . .

Who, me?

Now that I've got your attention, check out the fine print in figure three.

Note that we not only have isometric lettering, but that lettering is also *wrapped around* an isometric cylinder at the same time!

Applewriter on a IIe, of course. Helped along with my *Applewriter/Laserwriter Utilities*. To let you in on all the secrets at no charge, just check out the listing in figure four.

This is intermediate level *Postscript* code. Each letter is seperately compressed to the proper width, slanted the right amount, and raised as needed to map into its proper place. As you can see, sines and cosines do most of the work.

Variations on the same code will run all the way around the cylinder, showing the rear lettering in gray, rather than black. You can also do helix or spiral effects of either constant or ever diminishing character heights.

Besides "gee whiz" uses, this will very nicely handle perspective lettering for an upcoming perspective drawing add-on to my utilities.

Scanning conversion will be rather slow, perhaps taking an hour or more for a full page of perspective lettering. The obvious reply to any criticism of this speed would be "Uh, compared to what?".

A demo or two should be available by the time you read this, so write or call for a much improved self-portrait of Marcia's latest.

(continued from page 17.5)

Next, overprint a gray grid onto the big image, using the gray grid routines in my utilities. Then punch a hole at each end of each line, and at each tangent point, or any other point where you want total control. Very few points are needed, and, in general, the *fewer* the better.

Then, tape the artwork onto a light box or a window, and punch the magic numbers into your IIe, entering an X, Y, and direction for each point.

Figure two is a listing of the *Postscript* code involved.

Thanks to all of my newly revised curve tracing routines, an astonishingly low number of points is needed to do this stunning graphics quality. For instance, only four points are needed for a circle, ellipse, or oval. Thus, you often end up working with an extremely sparse data set.

Reviewing, the curvetrace routine works by inputing an X location, a Y location, and a slope for the ends and tangents of each curved path desired.

Those numbers are then converted into a form that is useful by the cubic spline routines already inside the Laserwriter. The paths can then be stroked, filled, or even used as a clipping window for future paths.

The curve tracing speed is normally quite fast, but since I intentionally used several irregular clipping masks on this particular character, this particular image does take over a minute to print.

If you want to view the location and direction of each point, just change */showtick* from false to true. This is a powerful debugging tool.

You will also need a listing of my curve tracing routine. You can get one that is all ready to go on my various *Applewriter/Laserwriter Utilities*, or else I will send you a free listing of this new super dooper routine, along with a figure one original, when you call or write.

Don Lancaster's

ASK THE GURU

October, 1986

Apple i.d. bytes
Laser letterheads
RAM card snooper
Applewriter nullifier
Machine language contest

Let's start off with a few loose ends. First and most importantly, don't miss next month's issue of *Computer Shopper*. You just might find a thing or two in it that just may be of more than passing interest to you.

For those of you that are into creating your own type faces for most any display, plotter, or printer, there's an interesting book you should know about. It is called the *National Bureau of Standards Special Publication #424* and the price *was* only $3.

The first little gotcha is that this item is permanantly out of print. Possibly you can borrow a copy from a library with a large government documents section. The original catalog number was SD C13. 10.424.

The second gotcha is that their fonts are intended for stroke graphics, rather than for raster scanned graphics, so there's not a lot here that you can immediately put to use.

I have also found a better way to open a *Canon* copier or laser printer toner tank for refilling. Instead of cutting or scraping a hole in the tank, you simply *melt* one. A miniature 25 watt soldering iron is ideal. What you do is melt out a 1/2 inch circle and then stop 9/10ths of the distance around. Then you snap the disk off with a pair of small pliers. As before, you seal the hole with *Scotch* tape after refilling.

UPDATE: January 87

I heard a rumor that some sort of a modification can be made to laser printers that allows foil hot stamping and embossing. I'll believe this one when I see it, but my, oh my, the potential. Let me know if you have any details on this.

UPDATE: December 86

Oh, yes. Some of you seem to be having trouble picking up both of Adobe's two *Postscript* books locally, so we are now stocking them as a special service for all of you *Computer Shopper* readers. These two books are absolutely essential to wring the maximum performance out of your *Laserwriter*.

As always, this is your column and you can get technical help per the end box. I also have bunches of free stuff available for you when you call or write.

Let's move on to whatever it is that we're up to here. If you ever find out, please let me know . . .

MACHINE	$FBB3	$FB1E	$FBC0	$FBBF
Apple II	$38			
Apple II+	$EA	$AD		
Apple III	$EA	$8A		
Apple IIe (old)	$06			$EA
Apple IIe (new)	$06			$E0
Apple IIc (old)	$06		$FF	$00
Apple IIc (new)	$06		$00	$00

Fig. 1 - Monitor i.d. bytes for various flavor Apples.

Which Apple is Which?

This is a favorite question on the help line, and it has been a while, so here goes.

There are obvious differences between the Apple II, the II+, the III emulator, the old IIe, the new IIe, the old IIc, and the new IIc. If you pretest the machine your software is about to be run on, you can selectively add or remove features, letting you optimize your program for the target machine.

There are two ways you can identify these different machines.

The method that always works is to verify those four monitor bytes shown to you in figure one below. Every byte combination is unique for that machine.

The second method works *only under ProDOS* and uses location $BF98 in main memory. Figure two shows you details. This location will also tell you whether a compatible clock is installed, how much memory is present, and if any 80 column firmware exists.

UPDATE: November 86

By the way, there are two free *Apple* manuals that have lots of this sort of information in them. These are the *Apple IIc Programmers Guide to the 3.5 ROM* (#A2L4037) and the *About Your Enhanced Apple* IIe (#A2L2072). You may want to pick both of these up.

I Need a RAM Card Snooper Routine

Figure three is a listing of a short routine that will copy up to 56 pages (or 27 ProDOS blocks) from a slot #4 *Apple* one megabyte RAM card into main memory. It will do so with or without an operating system and can be used after a cold reset.

This new routine has some outstanding diagnostic and/or analysis possibilities.

You can set the number of pages read by the hex value poked into $0F01. Use $01 for a single page of 256 bytes, $02 for one ProDOS block of 512 bytes, up to a maximum of $38 for 28 blocks.

The actual memory dump will begin in the RAM card at the address you select, and will be moved down into the main memory starting at hex $6000. The upper limit is set to $9900 to prevent interference with a DOS 3.3 or ProDOS reboot.

The starting address is high enough in main RAM that you can perform a *warm* reboot of ProDOS around it without hurting the copy.

Several reminders about the RAM card. Always set your addresses in a low-mid-high order. Remember that any RAM card read or write will *auto-post-increment* the RAM card address. Finally, remember that you must not try to read or write to the RAM card using indexed addressing, unless you intentionally cross a page boundary.

To use with a slot other than slot #4, note that $C0D0 through $C0D2 are the low-mid-high address for slot #5, while $C0D3 is the data read location, and so on up and down the line of slots. Thus $C090 starts slot 1, $C0A0 slot 2, and so on.

More insider information on all of this appears in my new *RAM Card Disassembly Script*.

How About an Easier Contest This Time?

Right on. As usual, a copy of my *Incredible Secret Money Machine* to the best ten entries, and an all expense paid (FOB Thatcher, AZ) tinaja quest for two to the overall winner.

Since not just anybody can be trusted with such a powerful and useful module as a RAM card snooper, step #2 of figure 3 is written in a secret, very mysterious, and highly arcane code. Just tell me, in English, exactly what is happening here, along with your suggestions for some heavy duty new uses.

Only let's play fair. You are automatically disqualified if you can "sight read" step two. And, if you eat this sort of thing for lunch, how about $20 57 FC instead? Or else take a $00.

Actually, this super secret encryption technique is so powerful and so good, that maybe we could even talk Apple into using it someday.

Tell me all About Applewriter NULLs

Many printers require the ASCII $00 or NULL command to do certain things. This is incredibly stupid, because the NULL command is specifically reserved as an ASCII control command that is *not* supposed to do anything, anywhere, anytime.

Nonetheless, things being the way they are, certain printers and some cards need NULLs to operate properly.

In the DOS 3.3e version of *Applewriter*, all NULLs were specifically excluded, and a custom patch was needed to let you imbed any NULLs in all your printer control commands. Write or call me if you need a free copy of this DOS 3.3 Applewriter patch.

The problem was sort of fixed in ProDOS Applewriter 2.0 or 2.1. But, the stock solution introduced a new problem for other printers. And *Apple* never bothered to tell anyone about handling NULLs in the first place.

What they did was let you substitute a [_] (control-underline) for a NULL. [_] also goes by the names of a US *user seperator* or as a CHR$31. Anytime you need a NULL in either one of these programs, all you have to do is verbatim insert a [_].

Unfortunately, many high performance printers *demand* a US or a CHR$31 to activate certain special features. For instance, the HMI commands on daisywheel printers, and some extended text modes in dot matrix printers require this ASCII control character as part of a command string.

So, some of you may need a way to undo the stock automatic substitution of [_] for a NULL. The solution for the ProDOS Applewriter 2.0 appears as Patch B.12 in my *Applewriter Cookbook*.

The gist of the 2.0 patch is to change $4C37 of AWD.SYS to match your desired substitution character. Use $1F for default operation.

0 0 - - 0 - - -	is an Apple II
0 1 - - 0 - - -	is an Apple II+
1 0 - - 0 - - -	is an Apple IIe
1 1 - - 0 - - -	is an emulating Apple III
1 0 - - 1 - - -	is an Apple IIc
- - 0 1 - - - -	is a 48K machine
- - 1 0 - - - -	is a 64K machine
- - 1 1 - - - -	is a 128K machine
- - - - - - 0 -	has no 80 column card
- - - - - - 1 -	has an 80 column card
- - - - - - - 0	has no compatible clock
- - - - - - - 1	has a compatible clock

Fig. 2 - Using the ProDOS $BF98 i.d. byte.

Use a $00 if you want no NULL substitution at all. And, for version 2.1, we'll have to go clear on to the next question . . .

Tell me about the ProDOS Applewriter 2.1 patches

ProDOS Applewriter 2.1 is a variation on the older 2.0 version. Its main difference is that certain older third party dot matrix printers and interface cards can now be made compatible with both ProDOS and *Applewriter*.

A free upgrade has been available from your *Apple* dealer now since November of 1984. But there is no point in upgrading unless you have these problems to start with. Most printers and cards have long since corrected their firmware to be fully ProDOS compatible.

Unfortunately, the 2.1 new version progressively gets a few dozen bytes longer than the 2.0 version. Which means that most of the essential 2.0 patches shown in my *Applewriter Cookbook* will not run properly on 2.1.

For far too many months now, I have been promising many of you a full set of patches for *ProDOS Apple*writer 2.1. So, what I think we will do is run a patch a month here for a while.

In general, there is one major patching difference between 2.0 and 2.1. The AWD.SYS found on 2.0 is a SYSTEM type file. But, the AWD.SYS on 2.1 is a $0C type file. Thus, 2.0 requires a ",TSYS" following a patching BLOAD or BSAVE, while the version 2.1 requires a ",T$0C instead.

Let's get the easy patches out of the way first. The Prefixifier "patch" stays the same. The AIOifier and the Grapplifier patches are no longer needed, as 2.1 solves this particular problem on its own. And the Bootifier patch also stays the same, since it occurs so early in the code.

As we've just seen, the $00 character is reserved for a marker in all newer versions of Applewriter, so the stock program will accept and then substitute a US or a [_] (control-underline) for the NULL character.

So, if your printer or whatever needs a NULL, all you have to do is use [_] the same way you did with ProDOS Applewriter 2.0. But, if you need [_] for a US or a CHR$31 printer command, Figure 4 shows you the Nullifier patch for the newer version 2.1 code.

What you do is make a one byte substitution for the NULL character. If you want no NULLs, substitute $00. If you need no US or CHR$31, then substitute $1F. If you must have both a US and a NULL, then you will have to find some different control character that you do not need, and substitute it instead.

Stay tuned for next month's exciting new 2.1 patch. Be the very first one in your neighborhood to collect them all!

1. **Do a reset or an absolute reset WITHOUT turning of the power. Then CALL-151 to get into the monitor.**

2. **Enter:**

```
$0F00:  A2 38 A0 00 84 06 A9 60
$0F08:  85 07 AD C3 C0 91 06 C8
$0F10:  D0 F8 E6 07 CA D0 F3 60
```

3. **Set the RAM card address at $C0C0 low, $C0C1 medium, and $C0C2 high. (Slot #4 values.)**

4. **Do a 0F00G. The selected pages or blocks of the RAM card will be moved to main memory, starting at $6000 and ending at 98FF.**

Fig. 3 - A "snooper" for the Apple RAM card.

What is This Month's Laserwriter Scam?

Uh, would you believe custom business stationary?

Done, of course, with both the Applewriter word processor and the Laserwriter, helped along with a few of my whiz-bang utilities.

Every business, large or small, needs their share of letterheads, envelopes, business cards, and such. The traiditonal suppliers of these often require very large minimum quantities, have long setup and delivery times, do not give you exactly what you want, and leave you with a staggering bill.

There are two different ways the *Laserwriter* can solve these hassles once and for all. You can use the *Laserwriter* just for pre-press, or you can actually print the letterheads on demand when and as they are needed.

The on-demand method will appeal to most small business, technical venture, or craft needs. No letterheads ever need to be pre-printed ahead of their actual use. Small quantities are a snap.

Any changes or adjustments, however major, can be made at any time. You can even print custom letterheads instantly at a fair, swap meet, yard sale, mall, or trade show, and sell as few as a dozen of each at a profit.

Thanks to the magic of *Postscript*, you can put the letterhead image, the body of the letter, and even your signature *directly into* the *Laserwriter*. You have to do this *only once*. Then you simply erase and rewrite each

new name and address, letting your fully custom letters pour out at maximum speed.

If you are very careful about your choice of paper and your artwork images, the quality will be more than acceptable for all but the most critical of business uses. I have found that a black and grey letterhead on *Guilford Gray* classic laid stationary seems to work well for me.

Best of all, there is no hint whatsoever of any "computereeze" about the final letter. Except that the result is so good that it could not possibly be anything else.

You can experiment with various colors of paper and toners. Repeat passes through the printer can be used for color and duotone effects.

The advantages of the on-demand letterhead method are that it is fast, easy, and gives acceptable results for most people most of the time.

The disadvantages are that you are stuck with 300 dots per inch resolution, have few colors available, and are limited to the durability and inherent appearance of heat fused toner.

Thus, you'll find the final on-demand results can in no way directly compete against, say, full color, raised ink, hot stamped foil, extreme detail, thermography, embossing, or any of the other goodies that you can coax out of the traditional printing processes.

UPDATE: The *Kroy Kolor* and *Omnicrom* processes can now give you stunning full color from your LaserWriter. See column 21 and later for more details.

If high print quality is a must, then you can instead use the *Laserwriter* to prepare your original artwork for traditional printing. Note that you can work oversize and then photo reduce. This can stretch the 300 dots per inch limit as far as you want to go. You can even do full color by printing seperations when and as needed.

Outstanding pastels and halftones can be done using the internal *Laserwriter* halftone machinery. Just do not forget that the screen size changes as you enlarge or reduce the image. And, of course, you have an infinite range of type styles, shapes, and sizes, compared to the fixed height and fixed width of "old way" instant transfer lettering.

Let us look at an example. Figure five shows you one typical letterhead and some of the *Laserwriter* special effects you can easily do. Once again, you can print this letterhead on demand as it sits, or else you can use the image as master artwork for traditional printing processes.

Figure six gives you the *Postscript* listing for this letterhead. This listing also shows why it is so important to use a *Postscript* speaking printer in the first place. It also dramatically drives home why you will want to work directly in the unique *Postscript* language, instead of using someone else's canned layout routines.

Check closely, and you will find lots of subtlety here. The company name is imaged by using the *charpath* operator in a *forall* loop. Each letter is imaged *three* times. The first time, a fairly wide *white* outline is put down. This will automatically break the base hairline exactly when and where needed. Next, the letter is filled in with gray, using a custom and very dense halftone screen. Finally, the letter outline is traced in black. Some extra kerning is *removed* between the "r" and the "g" to force a much more

This patch is for ProDOS Applewriter 2.1 version AWD.SYS only. AWD.SYS runs only on the 80 column IIc or the 128K IIe.

This patch lets you redefine the substitute NULL control command from its default value of [_], $1F, US, or CHR$31.

This patch works by replacing the control command to the desired substitution.

Note that this patch is needed ONLY if you require use of [_] for another printer command.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.1, using the filer utilities.**
2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**
3. **BLOAD AWD.SYS, A$2000, E$6020, T$0C, D2**
4. **Verify $4C40 as a $C9 and $4C41 as the current null substitution character. (stock value = $1F)**
5. **Change $4C41 to the intended control character. Use $1F to return to a default [_], or $00 for no NULL character at all.**
6. **Verify $4C40-4C41 for a $C9 followed by the control command you just selected.**
7. **BSAVE AWD.SYS, A$2000, E$6020, T$0C.**

Fig. 4 - A NULL patch for ProDOS Applewriter 2.1.

Fig. 5 - A custom letterhead design done entirely in POSTSCRIPT.

visually attractive spacing.

The street address and the phone number are done by stretching *Helvetica* out so far that it now picks up a totally different set of gothic vibes, bordering almost on a *Copperplate* or *Steelplate* image. Once again, your own custom *Postscript* routines make this utterly trivial.

I'll be happy to lay some free samples on you.

What About Envelopes?

Funny you should ask that. I have yet to find *any* way to print decent or even acceptable envelopes with *any* laser printer at all.

It may have to do with the double paper layer in an envelope giving you charge transfer hassles, or with the lack of any uniform spacing between the top envelope surface and the image drum.

But, every once in a long while, a beautiful envelope shows up in the mail that has obviously been laser printed. Others assure me that there is no problem at all reliably printing laser envelopes. And lots of people are trying to sell horrendously overpriced envelope feeder mechanisms.

So, perhaps I don't know something about this, or else I am doing something obviously wrong.

My own approach right now is to do the return address prepress artwork using the Laserwriter, and then have a local printer "crash print" the envelopes. Cost is well under a nickel an envelope if you buy your own quality envelopes and then use jiffy printing.

But what about the main address?

I have tried quite a few methods. The one that works the best so far is to go back to my noisy old *Diablo* 630 and use a WPL routine to "semi-automate" the envelope feeding process. All you have to do is hold the envelope in place and flip the bail back and forth. The WPL routine and some time delays take care of all the feeding and ejection. And it is considerably faster than a manual feed laser printer.

I once used continuous form "tip-in" envelopes, but they cost too much, took far too long to get, looked awful, had lousy glue, and often jammed.

Another thing I tried was to use transparent matte labels with the Laserwriter. The toner turns ultra black and very sharp on acetate.

You can quickly and very quietly run a dozen of these at once and they do not look half bad. You can also get "classic laid" labels that are exactly the same color and finish as your envelopes.

Direct mail people tend to be superstitious, and I am no exception. I do feel that an envelope that has the address directly on it might have slightly higher odds of getting opened than mail with an obvious stick-on label.

Which may or may not be true. Or reflect reality.

Obviously, there is a big problem here. I suspect that future laser printers will have some sort of a flip-down envelope pressure roller that will solve these hassles once and for all. Meanwhile, how about clueing me in on what I am doing wrong?

Let me know if you want a free listing for the Diablo auto-feeder. As usual, I'll be happy to give you some free letterhead and other demos when you call or write.

```
/letterhead {save /snap exch def 100 45 {dup mul
exch dup mul add 1.0 exch sub} setscreen 40 720
translate  0 0 moveto 0 setlinewidth 533 0 rlineto
stroke newpath 0 0 moveto /Bookman-DemiItalic
findfont [34 0 0 34 0 0] makefont setfont /kernstuff
1 def (
Synergetics) gsave -13 0 translate {dup  /ggg exch
def /workstring ( ) def /char exch def workstring 0
char put /chardist workstring stringwidth pop def
workstring false charpath gsave 1 setgray 3
setlinewidth stroke grestore gsave 0.99 setgray fill
grestore 0.5 setlinewidth stroke  chardist kernstuff
add ggg 114 eq { 1 sub } if 0 translate  0 3 moveto}
forall grestore /Helvetica findfont [12 0 0 9 0 0]
makefont setfont 450 -10 moveto 0.15 0
((602) 428-4073) ashow 285 4 moveto 0.15 0
(746 First Street, Box 809, Thatcher AZ 85552)
ashow snap restore} def

letterhead showpage
```

Fig. 6 - The POSTSCRIPT letterhead listing.

Don Lancaster's

ASK THE GURU

November, 1986

Password horror stories
The hidden grand piano
Applewriter Scrunchifier
Dissassembly on the Mac
Appleworks virtual memory

Assuming that all of the non-disclosure stuff got signed, and if they ever got that pesky alligator out of editorial, you ought to find a story of interest elsewhere in this month's issue.

I apparently missed telling you about the i.d. bytes in the main story. These new IIgs machines have the same i.d. bytes as the Apple IIe. As we saw last month, you'll find these i.d. bytes at $FBB3:06, $FBC0:EA and $FBBF:$00.

To tell the new machines from a IIe, do a JSR $FE1F. If you return with a set carry flag, then you have a IIe. If you return with a cleared carry flag, then you have a IIgs or something newer. You also get some system info returned in the A, X, and Y registers.

I also may have leaned on the vertical resolution a tad too heavy. The stock IIe can display 26,880 pixels in any of six colors, or 53,760 in black and white. The IIgs can display 128,000 pixels in any of 256 colors from a palette of 4096. The Mac can display zero color pixels, or 173,000 pixels in black and white.

By way of comparison, the IIgs full color display has just about the same resolution as the *Macpaint* window.

So, which is the better choice? 128,000 pixels in full color, or 173,000 black and white ones? The jury is out and will now decide.

Need a hot stock tip?

Adobe Systems, who are the *Postscript* people, recently went public. The market price at this writing is somewhere around 14, and it's symbol is ADBE on the NASDAQ over-the-counter market. It's listed in the *Wall Street Journal*, and in many other financial newspapers.

You can get a prospectus from Steve Piper at *Hambricht and Quist*. Some very interesting revelations appear in this prospectus. For instance, Apple now holds around 17 percent of Adobe. Another stockholder of note is *Evans and Sutherland*, the super whiz bang computer animation people.

The most stunning news in this prospectus is that Adobe has now placed *Postscript* in the public domain! I personally feel that Postscript is far and away the best page makeup language avialable today. Making the language more accessable should dramatically increase its use and acceptance.

The one thing that *Postscript* really has going for it is that this is far and away the most fun computer language, anywhere ever. *Postscript* is absolutely, totally, and utterly addictive, both psychologically and physically.

No other language gives you such a total feeling of absolute control. Or of such awesome graphics power.

A brief reminder that we have a new free stuff list, and we are now stocking Adobe's two new *Postscript* books as a service for all of you *Computer Shopper* readers.

With Haloween not that far off, let's start out kinda real scary like . . .

Tell me a Horror Story.

Do passwords serve any useful purpose on a network? The sad answer is probably yes. You may not want just anyone changing what the system function and setup is going to be.

Next question: If the password is forgotten or misused, should it result in the sudden and total destruction of the entire system?

Before you give a sensible and logical answer to the last

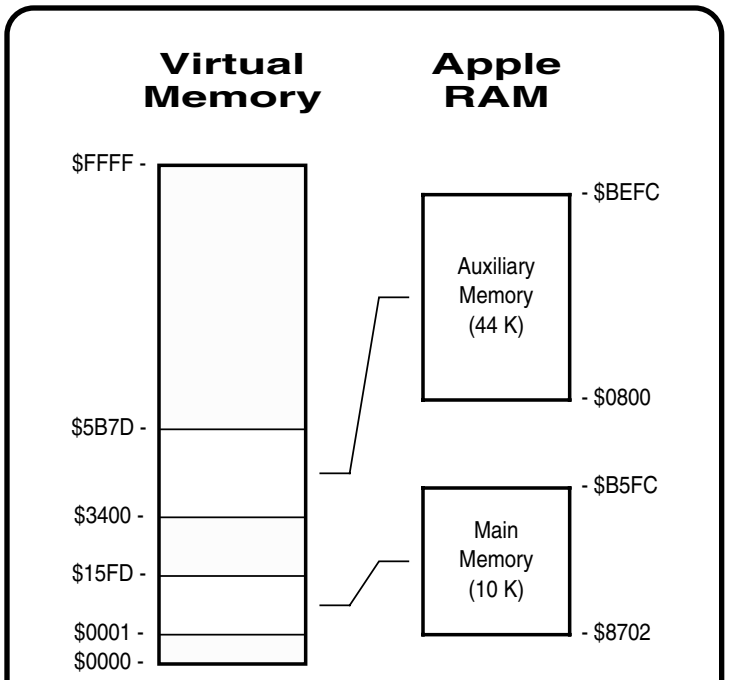


Fig. 1 - Appleworks virtual memory mapping.

This patch is for ProDOS Applewriter 2.1 version AWD.SYS only. AWD.SYS runs only on the 80 column IIc or the 128K IIe.

This patch shortens the ProDOS options menu to four lines, leaving more of a previous catalog on the screen. It also frees room in the code for other patches.

Note that this patch is REQUIRED if the Cursifier and Stretchifier patches are to be used.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.1, using the filer utilities.**

2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**

3. **BLOAD AWD.SYS, A$2000, E$6020, T$0C, D2**

4. **Verify 5937- 20 20 20 50 72.**

5. **Change:**

```
5937:  50
5938:  72 6F 44 6F 73 3A 20 28
5940:  41 29 20 43 61 74 61 6C
5948:  6F 67 20 20 28 43 29 20

5950:  4C 6F 63 6B 20 20 20 20
5958:  28 45 29 20 44 65 6C 65
5960:  74 65 20 20 20 28 47 29
5968:  20 53 75 62 64 72 63 74

5970:  79 20 28 49 29 20 46 6F
5978:  72 6D 61 74 20 20 20 0D
5980:  20 20 20 20 20 20 20 20
5988:  28 42 29 20 52 65 6E 61

5990:  6D 65 20 20 20 28 44 29
5998:  20 55 6E 6C 6F 63 6B 20
59A0:  20 28 46 29 20 4F 6E 2D
59A8:  4C 69 6E 65 20 20 28 48

59B0:  29 20 50 72 65 66 69 78
59B8:  20 20 20 28 4A 29 20 50
59C0:  72 69 6E 74 65 72 20 2D
59C8:  3E 00 99 00 11 11 11 11

59D0:  11 11 11 11 11 11 11 11
59D8:  11 11 11 11 11 11 11 11
59E0:  11 11 11 11 11 11 11 11
59E8:  11 11 11 11 11 11 11 11

59F0:  11 11 11 11 11 11 11 11
59F8:  11 11 11 11 11 11 11 11
5A00:  11 11 11 11 11 11 11 11
5A28:  11 11 11 11 11 11 11 (stop!)
```

6. **Verify 5937-5A2E per the above code.**
   Verify 5A2F- 3A 51 7C 50.

7. **BSAVE AWD.SYS, A$2000, E$6020, T$0C.**

Fig. 2 - ProDOS Applewriter 2.1 Scrunchifier patch.

question, consider this: If the machine does *not* immediately blow up with a misused password, then there *has* to be some method of circumventing the password use. In which case, that password accomplishes nothing useful in the first place.

Here's the horror story. It turns out *Apple* was much too zealous in all their password protection on the *Laserwriter*.

If you do forget your password, or if someone else unknowingly or else maliciously changes it, or if some system software blows up in the worst possible way, then your Laserwriter will self-destruct!

Screw up your password, and you first have to swap the internal computer board out, at a cost of nearly $1000. Then you have to send many of your downloadable fonts back to the software house, where they will be physically exchanged *only once*.

The bottom line: Mess up your password and you are out $1000, three weeks time, and some utterly unbelievable hassles.

Say you wanted to do in a quick-copy service chain that rents Laserwriters. In half a morning's time, you could easily cause them $50,000 in damage and shut them down for three weeks or more.

The solution: First, leave your *Laserwriter* password at the default value. Any attempt whatsoever to change it will *increase* the likelyhood of a blowup.

I do not know what the best answer to this one is. A two-step approach of socketing the non-volatile startup memory, and providing for some password reset proceedure will probably be best. A "wrist slap" rather than a "kick in the teeth" might be instead used for password misuse. Requiring that the lid be physically removed from the machine to reset your password should do the trick.

Removing that idiotic and stupid single machine protection scheme from the fonts

would also help things out bunches. Meanwhile, do not under any circumstances, try and change your password from its default value.

Any ideas on this?

How About Another One?

Hmmmmm. Once upon a time, long ago and far away, there was this programmer who very much wanted to make a *printed disassembly listing* of a file he had on a diskette.

If that programmer was using an *Apple IIe*, all he would have had to do is punch in these keystrokes:

```
BLOAD FILE, A$2000
CALL -151
PR#1
2000 LLLLLL
```

But this programmer was so incredibly stupid that he was instead trying to use a *Macintosh* disk and was trying to do his listing directly on his Mac printer.

How silly can you get?

After three weeks of work and innumerable trys at various software packages, that programmer *still* does not have his disk file properly disassembled to paper. After five disk blowups, though, he finally did find a poor way of doing the listing in very tiny and almost totally useless segments.

This programmer simply could not believe that the single fundamental most important task of a personal computer – the ability to very quickly and conveniently list – appears to be totally unavailable on the Mac!

If you know how, please send me *any* workable way to disassemble a Mac disk file onto paper. Oh yes, the file is 68000 code, but is *not* Mac code.

I have a hunch this will be far and away the hardest contest we have ever run. As usual, an *Incredible Secret Money Machine* to the first dozen entrants, and an all-expense paid (FOB Thatcher, AZ) tinaja quest for two for the best overall reply.

Help!

How does Appleworks Virtual Memory Work?

Suppose you had some free RAM available in several different locations in your computer. Say further that you wanted to put lots of different work files into these areas. How would you do it?

If you decided to do the obvious and start placing files where they would "fit" in memory, things might be ok for a while. But, then say that one of your earlier files got longer. This might happen because you have added a few new paragraphs to a word processor text file. What now? To make room for the new stuff, you either have to rearrange everything that is already in memory, or else split up the longer file into two or more disjoined pieces.

To get around having to continuously rearrange what is already in memory and to elegantly solve other awkward problems, the computer people long ago have come up with the concept of *virtual memory*. These days, though, the term *memory management* is often used instead.

With virtual memory, you set up a "fake" or imaginary memory area. Into this imaginary area, you put one or more *linked lists* of real memory addresses.

Each *segment* of virtual memory, or VM, has a *length* and a *link*.

The length equals the total number of vm bytes used by this particular segment. The link is the starting vm address of the next segment of this particular file.

For instance, a word processor file might start off with a segment of vm that starts at vm address $1356. This particular segment of available memory may be too small to hold the entire file, so it links to another vm segment at $39FC, and so on. The last file in the chain will have a $00 link, so we know we are at the end of the file as well as the end of the list.

Each vm address will be related in some fixed way to "real" RAM. The linked list lets you connect many different pieces of real RAM

Fig. 3 - Bee's Piano.

```
/pedals {[276 375 -90  276 365 -90 276 365 0 305 365 0 305 365 90  305 375 90 305 375 180
276 375 180] ctf [297 366 -90  297 312 -90] ctb [280 366 -90  280 314 -90] cta [291 366 -90
291 312 -90] cta [293 316 55 321 345 55 ] cta [280 316 55  321 363 55 ] cta [280 354 -5
298 353 -5 ] ctd gsave 0 5 translate [270 307 90 270 310 90 270 310 5  278 312 5 278 312 -8
301 308 -8 301 308 -90  301 298 -90 301 298 -175  294 297 -175 294 297 90  294 302 90
294 302 170  270 307 170] ctf grestore gsave -0.5 5 translate [269 300 90  269 302 90
269 302 0   270 302 0 270 302 -90  270 300 -90 270 300 180  269 300 180] ctf grestore
gsave 6 3.9 translate [269 300 90  269 302 90 269 302 0   270 302 0 270 302 -90 270 300 -90
270 300 180  269 300 180] ctf grestore gsave 10.5 2.9  translate [269 300 90
269 302 90 269 302 0  270 302 0 270 302 -90  270 300 -90 270 300 180 269 300 180] ctf
grestore gsave 0.5 6 translate [284 296 90  284 299 90 284 299 -1  289 298 -1 289 298 -90
289 295 -90 289 295 1  284 296 1] ctf grestore [276 306 -1  289 304 -1] ctb
[289 302 1 295 303 1] ctb} def

/legs{[355 372 -90  355 367 -90 355 367 180  328 296 -90 328 296 0  330 296 0 330 296 -92
328 285 -92 328 285 180 320 285 180 320 285 92  318 296 92 318 296 0 320 296 0 321 296 90
296 367 180 296 367 90  296 372 90 296 372 0  355 372 0] ctf [370 377 -90  370 372 -90
370 372 180  310 372 180 310 372 5  370 377 5] ctf newpath 324 280 2.8 2.8 360 arc
closepath 0 setgray 1 setlinewidth stroke gsave 0.7 dup scale 300 180 translate
[355 372 -90  355 367 -90 355 367 180  328 296 -90 328 296 0  330 296 0 330 296 -92
328 285 -92 328 285 180  320 285 180 320 285 92  318 296 92 318 296 0  320 296 0
321 296 90  296 367 180 296 367 90  296 372 90 296 372 0  355 372 0] ctf newpath
324 280 2.8 2.8 360 arc closepath 0 setgray 1 setlinewidth stroke grestore gsave
0.74 dup scale 0 145 translate [355 372 -90  355 367 -90 355 367 180  328 296 -90 328 296
0  330 296 0 330 296 -92  328 285 -92 328 285 180  320 285 180 320 285 92  318 296 92
318 296 0  320 296 0 321 296 90  296 367 180 296 367 90  296 372 90 296 372 0  355 372 0]
ctf newpath 324 280 2.8 2.8 360 arc closepath 0 setgray 1 setlinewidth stroke grestore
clear} def

/lidandbody{[290 442 45  367 522 45 367 522 -2  410 520 -2 410 520 -135  385 494 -135
385 494 -55  387 490 -23 472 465 -130 405 419 -155] ctf [405 419 0 467 420 0 467 420 -90
467 385 -90 467 385 -175   300 372 -175 300 372 90  300 447 90] ctf} def

/keyboard {[300 372 177  207 379 177 207 379 90  207 393 90 207 393 20 225 407 75
225 407 180  221 407 180 221 407 90  221 417 90 221 417 -1 256 418 2 256 418 70
268 440 5 314 430 -120   303 412 -120] ctf 1 setgray 1 setlinewidth [292 442 45
365 519 45 365 519 7  391 512 -75 392 512 90 391 510 110 368 515 -178 368 515 -135
298 442 -135 298 442 180 292 442 180] ctfd [389 514 -75  385 495 -70] ctc } def

/support{[355 497 45  360 503 45 360 503 -75  385 420 -75 385 420 180  379 420 180
379 420 106  355 497 106] ctfd [387 421 0  395 421 0 395 421 -95  394 392 -105 383 380 -150
383 380 88  387 397 90  387 420 90] ctfd [312 405 -110  308 390 -130  302 386 -170
303 385 -5 312 388 50 318 405 71 318 405 180  312 405 180] ctfd [304 373 90  304 384 90]
ctc [308 412 0  313 412 0 313 412 -90  313 405 -90 313 405 180  308 405 180 308 405 90
308 412 90] ctfd [222 415 0  227 415 0 227 415 -90  227 408 -90 227 408 180  222 408 180
222 408 90 222 415 90] ctfd [211 394 20  226 406 75 226 406 0 230 406 0 230 406 -95
211 394 180] ctfd} def

/keyboardwhite{[219 385 177  298 378 177 298 378 90 298 386 90 298 386 -3  219 392 -3
219 392 -90  219 385 -90] ctfd [280 378 -3  295 377 -3] ctc [375 415 5  307 417 5
307 417 -120  304 412 -120 304 412 4 375 415 4] ctfd [310 440 -15  315 431 -120
303 413 -120] ctc grestore clear} def

/piano { 1 setlinecap 1 setlinejoin /tension 2.8 def /ctf {curvetrace gsave 0 setgray
fill grestore 1 setlinewidth 0 setgray stroke}def /ctfd {curvetrace gsave 1 setgray
fill grestore 1 setlinewidth 0 setgray stroke}def /ct {curvetrace stroke} def /cta
{curvetrace 7 setlinewidth stroke} def /ctb {curvetrace 2 setlinewidth stroke} def /ctc
{curvetrace 1 setgray stroke} def /ctd {curvetrace 4 setlinewidth stroke} def /showtick
false def pedals legs lidandbody keyboard support keyboardwhite} def

60 150  translate piano showpage
```

Fig. 4 - Postscript Listing for Bee's Piano.

together so they *appear* to create a contiguous text file, or whatever. While the pieces may vary all over the real lot, they are tightly connected together in a vm linked list.

Should a file get longer, it simply tacks a new piece of free vm to the end of its list. Thus, you never have to move existing files around in RAM. The linked lists of the virtual memory will automatically do this for you.

You can even insert or delete stuff from the middle of your file, by breaking the list and inserting or deleting vm segments from the list.

Several vm lists can be used. The *free* list can hold all of the available memory. A *non-purgeable* list holds all of the files that must not be deleted or changed. A *purgeable* list holds program modules that can be bumped out of vm if something more important needs the area.

Virtual memory management consists of maintaining linked lists of free memory, purgeable memory, of and non-purgeable memory. As a vm area gets removed from one list, it gets added to another. The tricky part is to make sure that things don't end up in very tiny segments involving very long lists.

Normally, the total amount of vm will exceed all the real RAM available. This helps the fragmentation problem considerably.

In the case of the original Appleworks, the virtual memory is 128K bytes, although only 55K of this is mapped to "real" RAM.

In new Appleworks 1.3, the size of vm depends on how much memory expansion card you have in use. For instance, with no card, you still usually get 55K, while with a 512K card, you get a desktop area of 503K bytes.

It is convenient to always use a 16-bit vm address, *regardless* of the number of bytes in vm. A 128K vm memory consists of 256 pages of 512 bytes each. So, the high 8-bit byte of our vm address can pick which of the 256 pages we are going to start on or link to. The low 8-bit byte can tell us the position on the vm page.

But note that we can only reach 256 addresses with a low 8-byte word. Thus, this particular vm address can only hit every *second* byte, since there are two bytes needed to describe 1/256th of a 512 byte page.

When a memory card is used, the number of vm pages stays the same, while the page position byte can only reach certain locations. For instance, an unused 512K RAM is set up so it has 256 vm pages of 4096 bytes each. Each value of the low 8-bit address can only hit a multiple of *sixteen* RAM bytes, since 4096/16 = 256.

So how does Appleworks vm work? Figure one can get us started.

A vm address space of 128K is used if there is no expansion RAM available. The starting vm address of $0000 is reserved, since a vm link of $0000 means that we are at the end of the linked list. Thus, vm address $0001 is the first useable address.

The relationship of vm address to "real" RAM can be arbitrary so long as it is known. There are usually two RAM areas that are available for the Appleworks desktop. These are addresses $8700-BF00 in main RAM (a 10K space), and addresses $0800-BFFF (A 45K space) over in auxiliary RAM.

The actual desktop size will change with the machine and the RAM card. A 64K "short" IIe can thus only use the main 10K. An available RAM card does not use *any* desktop space in the machine; the desktop is all handled by direct machine access to a fake ProDOS file on the RAM card itself.

All of these vm memory segment addresses are related to real RAM areas as shown in figure one. Segment $0000 (actually $0001) starts off the main memory piece at $8700 (actually $8702). Segment $3400 starts off the auxiliary memory piece at $0800.

Virtual memory is used in two major ways by *Appleworks*. The desktop files are purgeable or removable *only* if you decide to remove them. Meanwhile, as the 42 working code modules are loaded from disk, copies of them are made and stuffed into vm on a purgeable basis.

What this does is let you go to RAM instead of to disk the next time you need a module. This is much faster. As each new module is added to vm, the priority of all previous code modules is upped a notch. Should the vm manager need more memory for a desktop file addition or extension, it flushes as many of the oldest code modules as it has to.

The Appleworks disk file formats are available free of charge from Elizabeth Gebhardt, who is the *Appleworks* product manager over at *Apple Computer*. Copies of this should also be available on *MAUG* or *AppleLink*.

Show me another ProDOS Applewriter 2.1 patch.

Figure two shows us another ProDOS Applewriter 2.1 patch. This one is called the *Scrunchifier*. It does several things. First, it shortens up the DOS options menu to four lines, leaving much more of your previous listing on the screen. This is quite handy whenever you are renaming, locking, unlocking, or deleting lots of files at once.

Secondly, that extra text space freed up by your shortening can now be used for those two heavy duty *Stretchifier* and *Cursifier* patches that we'll see in future columns.

Equivalent patches for the ProDOS Applewriter 2.0 do appear in my *Applewriter Cookbook*,

Why Have You Been Hiding The Grand Piano?

You're right. Not once in the entire history of our *Ask the Guru* column have we even once published a picture of a grand piano.

So, to now correct this unspeakable oversight, figure three is a picture of Bee's Piano. *Applewriter* on an Apple II, of course. Helped out by my six disk *Applewriter/Laserwriter Utilities*.

This is done in the highly traditional art style of a "printer's tradesman cut", as you might use on a more conservative business card or letterhead. Then again, grand pianos tend to be a tad on the stuffy and conservative side anyway.

Figure four gives you a complete *Postscript* listing of the grand piano. To use this code, you'll also need a free listing of my *curvetrace* routine. You can get this one by calling or writing.

By the way, this particular *Postscript* listing is just long enough that you could get a stack overflow if you tried defining it all as one proceedure. That is why this routine is broken up into several pieces.

And This Month's Get Rich Quick Scheme is . . . ?

Business cards. How did you guess?

That *Applewriter* word processor on an Apple IIe can easily be used along with the *Laserwriter* to produce small quantities of business cards at incredibly low cost. This can even be done in "real time" at a mall, trade show, fair, or swap meet.

A typical card is shown in figure five. Ten cards are normally printed at once, using a 2 x 5 step and repeat procedure. I'll be happy to lay a great *Postscript* step-and-repeat on you for free if you ask.

Anyway, there are two effective routes to *Applewriter* created business cards. Let's call them the *cheap* route and the *fancy* route.

With the cheap route, you actually print the final cards 10-up on your *Laserwriter*. The advantages here are that you can make a profit on as few as ten cards, can deliver on the spot, and can easily make changes and revisions.

And that final quality, while somewhat modest, is more than acceptable to most individuals, craft persons, or small business users. Particularly for those people who simply would not pay the going rate for traditional cards from an old line printer.

Normally, you would print on a hand-fed heavier paper, such as the Dorian Gray or Baronial Ivory *Classic Crest* cover stock by *Neenah Paper*.

After printing, it is usually a good idea to spray some matte fixative from the art store onto the cards. This will greatly improve their scuff and wear resistance.

The disadvantages of the cheap route is that, at least right now, you can't do things like raised color ink, coated papers, foil or paper emobossing, thermography, and such. There is also a very slight "rattiness" to the 300 dots per inch resolution.

One major way around the rattiness is to print *all* smaller fonts with some extra kerning between all the letters. This makes a dramatic difference, and is one of the key secrets why my IIe stuff looks so much better than their Mac stuff.

For the fancy route, you simply use *Applewriter* and your *Laserwriter* to generate the pre-press artwork for a "real" business card printer.

By working oversize, the virtual resolution can be increased as much as you want. The printer will then photo-reduce your art to final size when he makes his plates.

Major advantages of using the *Applewriter* and *Laserwriter* combination for pre-press art are that this can quickly and simply can give you full custom card art, with no limitations at all on any text and graphics mix.

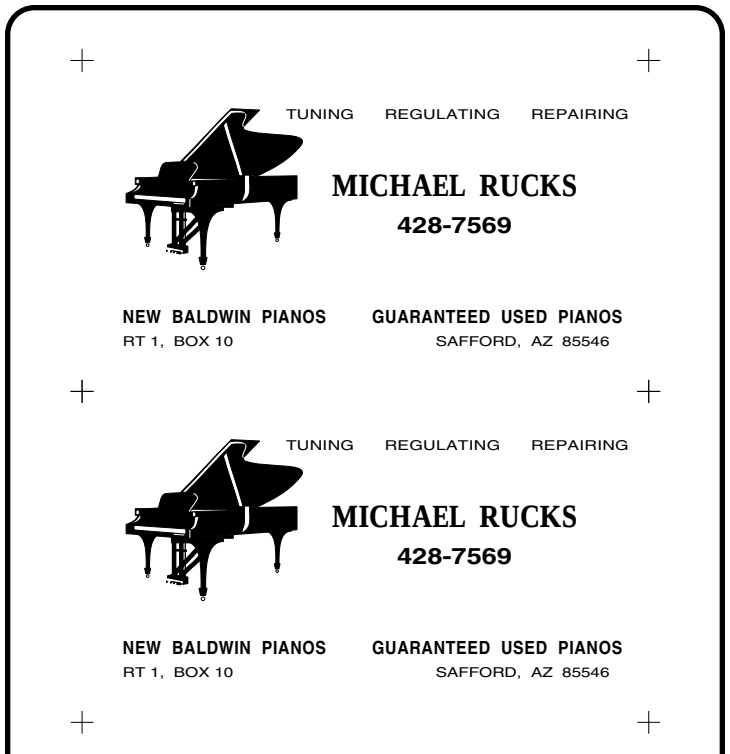


Fig. 5 - Typical step-and-repeat business cards.

Don Lancaster's

ASK THE GURU

December, 1986

Updates on the new IIgs
Applewriter swallowifier
Appleworks screen macros
Solar energy breakthrough
Omnichrom and Omnicolor

Well, I bet you thought last month's column would be a very hard act to follow. Not so. This month, behind the mauve curtain, we have our biggest blockbuster ever. How about *real ink* in stunning full color out of your stock and unmodified *Laserwriter*? Using *Applewriter* on an Apple IIe, of course, helped along by my utilities. And all this sanely priced to boot.

But first, let's fire up the rumor mill. We'll set it to a medium fine grind.

There is now a major new source for Apple developer insider information. And it is open to anyone anywhere.

Check into the new *Apple Programmers and Developers* Association. Membership is $20 per year, and it gives you instant access to all of the technical manuals, third party new programs, some beta test goodies, and much more.

The APDA is run by the *CALL A.P.P.L.E.* people, who publish one of the top three Apple magazines. The other two are *Open Apple* and *Apple Assembly Line*, of course. Nothing else is even within shouting distance.

Available are development tools, system software, technical notes, some operation manuals, languages, and even the entire *Addison-Weseley* library of Apple books.

Apparently the IIc liquid crystal display flat screen option has been flushed.

Mostly because it was totally illegible, I guess. But that does not seem to slow down the competition, whose flat panel displays are all completely unreadable under all lighting conditions.

I'm not convinced there is any market at all for fully and truely "portable" computers until such time as the display legibility, system performance, and battery life hassles can be fixed.

The IIc now has a new circuit board that does include plug-in memory expansion up to one megabyte of RAM. Check your Apple dealer for more details on upgrades and such.

Let us see. My free stuff list has just been updated, so write or call for your personal copy. We are now shipping the pre-release of our brand new *Appleworks Disassembly Script*, as well as stocking Adobe's *Postscript* books as a reader service for all of you *Computer Shopper* readers.

But now . . .

What is the Latest Word on the IIgs?

Lots of new info on this exciting new machine. Yes, there is an internal game paddle connector. This is handy for really old paddles and joysticks, and also gives you the annunciator outputs that are superb for ultra-cheap local networks or to interface to an ultrasonic BSR power controller or whatever.

No, there is no provision for a RF modulator. The old four pin internal connector is gone, and the IIc external RF modulator will not fit. Not that a RF modulator would do you very much good. Direct antenna entry to a stock color tv set would hopelessly smear 80 column color text, besides trashing the new super HIRES color modes.

Speaking of which, that new Apple color monitor is absolutely stunning. Not only is the 80 column color text fully legible, but it is actually pleasant to use with your full choice of text, background, and border color. This is the first time I have ever seen a 80 column color text display that I would actually be willing to use.

$01 - Clear rest of line
$02 - Clear entire line
$03 - Clear rest of window
$04 - Clear entire window

$05 - Relative to absolute cursor
$06 - Backspace
$07 - Frontspace
$08 - Cursor up; scroll down at top

$09 - Cursor down; scroll up at bottom
$0A - Set inverse text
$0B - Set normal text
$0C - Set bottom right of window

$0D - Move cursor to left of window
$0E - Set upper left of window
$0F - Open window to full screen
$10 - Blork the speaker

$11 - Do a horizontal scroll

Fig. 1 - Appleworks screen macro commands.

There is also a new plug-in SCSI interface card all ready to go. This can remove once and for all the paltry IIgs limit of a mere 128 disk drives of a pitiful five gigabytes each.

The IIgs bugs are starting to pour in. Naturally, any new machine will have at least a few teething problems, and the IIgs is no exception.

That 90 percent compatibility claim is a pipe dream. Even *Applewriter* and *Appleworks* have major problems.

Version 1.3 of *Appleworks* will not let you use an expansion RAM card on the IIgs, so you are stuck with 55K. It also gives you a garish tweet instead of the usual buzz in the fast mode. Apple did announce a brand new Version 2.0 of *Appleworks* that will give you extended memory access and reasonable tones.

Applewriter won't print on the IIgs! One temporary cure is to use your older super serial card instead of the internal IIgs serial ports.

Outside of that nit-picking detail, *Applewriter* seems to work ok at normal speed, but in the fast mode, you have to make some patches to keep the tones normal, the cursor flashing rate under control, and to set the proper inter-character delay for serial printing. I will have patches on these shortly.

At the faster speed, you'll find *Applewriter* to be *much* smoother, and even more fun to use. The speedup is most noticable on things like WPL routines, long searches, and on the reformatting that takes place after disk access.

Very nice.

There seems to be one big problem with the power supply. The rundown time is way over 30 seconds. The good news here is that you can now coast through more and deeper power line glitches without any problems. The bad news is that you can zap a board if you insert or remove it within half a minute of turning the power off.

So, be sure to wait at least 40 seconds after power down before making any IIgs card changes of any kind.

There are also some very severe problems involving hard disks. Some disks, particularly the ProFile, will lose data or may else plow themselves in the fast mode. For now, do not use an older hard disk with the IIgs, until this problem gets fixed.

A few of the earlier IIgs machines tend to have sound that sits there muttering to itself. While this sound level is very low, it could drive a "night person" up the wall if they are working in a totally silent room.

Some of the routines seem to mutter on purpose as a user prompt. I don't much care for this. If muttering is used, it should be a selectable option.

Another big compatibility problem with earlier programs is that the built-in clock does not use the same access rules as do the earlier Apple clock cards. Software modifications will definitely be needed for access.

The IIgs built-in clock is seperately available as a desk accessory. My own IIgs clock seems to gain 4 to 5 seconds a day. This seems a tad high.

This patch is for ProDOS Applewriter 2.1 version AWD.SYS only. AWD.SYS runs only on the 80 column IIc or the 128K IIe.

It eliminates the potential ability to overwrite the glossary and destroy the program. Corrects a byte in the original code that is just plain wrong.

This patch works by replacing a "glossary full" test byte with the correct value.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.1, using the filer utilities.**
2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**
3. **BLOAD AWD.SYS, A$2000, E$6020, T$0C, D2**
4. **Verify 2B56- 67**
5. **Change 2B56: 0F**
6. **BSAVE AWD.SYS, A$2000, E$6020, T$0C.**

Fig. 2 - ProDOS Applewriter 2.1 Glossifier patch.

How do the Appleworks Screen Macros Work?

Much of the *Appleworks* fancy text screen "filing card" manipulation is done using 17 screen macros. These macro commands are passed on to the screen routines just as if they were ordinary printing characters.

Since control commands do not appear on the screen anyway, the codes for the initial 17 ASCII control commands were usurped and replaced with these new custom screen macros.

Figure one shows us these commands. Some of these are obvious. Clears are available to clear line, rest of line, window, and rest of window.

Macro $05 is particularly interesting, since it converts a window-relative horizontal and vertical position into a

screen absolute one. To use this macro command, start with an $05 and then follow it with the in-window horizontal and vertical cursor positions for the screen.

For instance, an $05 $03 $04 will automatically move you four characters in and five lines down, regardless of where the window is on the screen. Note that a zero horizontal value is the leftmost and a zero vertical value is the topmost.

Continuing down the list, there's both a frontspace and backspace, followed by upline and downline commands.

These later two will scroll in the proper direction when they reach the top or bottom of the window.

The next two macros set normal and inverse text. They are followed by four window setting commands. The first ($0C) sets the bottom right of the window. The second ($0D) resets the cursor to the left of window. The third ($0E) sets the upper left of the window, while the fourth ($0F) opens the window back to full screen.

Macro $10 is used to beep the speaker for an error alert.

Finally, macro $11 does a horizontal scroll with the next byte deciding how far to go in which direction. Positive $00 to $4F values go to the right, while negative $FF to $CF values go to the left.

Much more detail on all this appears in the brand new pre-release of my *Appleworks Disassembly Script.*

What's new in Solar Energy?

Jim Allen of the *Solarjack* company has come up with a genuine breakthrough in solar energy economics. And he has done so with a production product that has been quite thoroughly field tested.

Windmills have traditionally been used in remote areas of the arid southwest for livestock and game watering. But windmills are costly to service and perform poorly with erratic winds and dropping water tables. On the other hand, most solar powered pumps have simply been too expensive to use.

Why?

Because each solar array had to drive a costly inverter and a bank of expensive and hard-to-maintain batteries.

Worse yet, the efficiency of the inversion and storage process gets so low that you lose all the way around.

Jim got to thinking that solar energy would make a lot more sense if you could throw away the inverter and the batteries, getting rid of both their cost and their inefficiency. Now, in a water pumping operation, you have one goal and one goal only: You want to put as much of the water into the tank as you can, and do so as efficiently as can possibly be done.

So, Jim reasoned that he would design the pump to fit the sunshine, rather than using inverters and batteries to make the sunshine fit the pump. What he came up with is a new *variable displacement* pump mechanism. When the sun is shining brightly, the pump makes long strokes and lifts lots of water. When the sun shines a little, the pump makes short strokes and lifts less water. At night, the pump makes zero strokes and does not lift anything.

What about clouds? Jim put a hefty flywheel on the pump so it can coast through brief cloudy times. A very simple, small, and rather efficient CMOS microcontroller then monitors the flywheel speed. Every now and then, it will adjust the pump displacement to exactly match the energy

This patch is for ProDOS Applewriter 2.0 version AWD.SYS only. AWD.SYS runs only on the 80 column IIc or the 128K IIe.

The patch eliminates a complete trip through the glossary whenever an "open apple <--" or an "open-apple -->" is done, greatly clarifying repeat saves or restores of saved characters.

It works by trapping out these two open-apple commands before they are processed as glossary commands.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.1, using the filer utilities.**
2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**
3. **BLOAD AWD.SYS, A$2000, E$6020, TSYS, D2**
4. **Verify 2A5A- D0 01 60**
5. **Change 2A5A: 4C 6B 45**
6. **Verify 456B- A0 FF C8 B9**
7. **Change 456B: F0 0B C9 08 F0**
   4570: 07 C9 15 F0 03 4C 5D 2A
   4578: 60
8. **BSAVE AWD.SYS, A$2000, E$6020, TSYS.**

Fig. 3 - ProDOS Applewriter 2.0 Swallowifier patch.

that is coming in from "up there". A small secondary motor using a worm screw drive is used to adjust the pump stroke.

Thanks to the big flywheel, the pump runs at an optimum and nearly constant speed, so long as any power at all is coming out of the panel.

I guess I am particularly proud of all this, because Jim is one of my students. And, he is successfully doing some very exotic things in a distinctly low tech part of the country. Give him a call if you want any more details.

Show me Some More ProDOS Applewriter Patches.

In continuing our ongoing *Applewriter* patch series, figure two is a GLOSSIFIER patch for ProDOS *Applewriter* 2.1 that eliminates a rarely activated glossary bug that can cause all sorts of problems. This is similar to the ProDOS 2.0 GLOSSIFIER patch that appears over in my *Applewriter Cookbook*.

Figure three is the new SWALLOWIFIER patch used with ProDOS *Applewriter* 2.0. This one eliminates the screen hassles when trying to swallow or barf characters to the left arrow/right arrow buffer. It turns out that this command used to have to go all the way through the glossary before it did anything. With long glossaries, especially self-prompting ones, this extra trip time royally fouled things up.

This patch operates by trapping out the swallow and barf commands before they have to go clear through the glossary search process.

Figure four is the same SWALLOWFIER patch intended for ProDOS *Applewriter* version 2.0. Besides its slightly different location, note that 2.0 files are SYS type files, while 2.1 files are $0D type files.

Many thanks on this to Don Thompson. Who, by the way, he has several brand new *Applewriter* patches and utilities ready for your use.

I am working on several patches for fast IIgs operation of *Applewriter*, while Don is working on the automatic clock link. More on this as it happens.

This patch is for ProDOS Applewriter 2.1 version AWD.SYS only. AWD.SYS runs only on the 80 column IIc or the 128K IIe.

The patch eliminates a complete trip through the glossary whenever an "open apple <--" or an "open-apple -->" is done, greatly clarifying repeat saves or restores of saved characters.

It works by trapping out these two open-apple commands before they are processed as glossary commands.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.1, using the filer utilities.**
2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**
3. **BLOAD AWD.SYS, A$2000, E$6020, T$0C, D2**
4. **Verify 2A5A- D0 01 60**
5. **Change 2A5A: 4C 75 45**
6. **Verify 4575- A0 FF C8 B9**
7. **Change 4575: F0 0B C9**
   4578: 08 F0 07 C9 15 F0 03 4C
   4580: 5D 2A 60
8. **BSAVE AWD.SYS, A$2000, E$6020, T$0C.**

Fig. 4 - ProDOS Applewriter 2.1 Swallowifier patch.

So, Where's the Big Blockbuster?

Patience. Patience.

I told you I would believe it when I saw it. I have seen it, I use it daily, and I still do not believe it.

I don't think I was ever more amazed when Woody Baker of *The Copier Store* mailed me back one of my very own laser printed business cards -- redone in *real ink* in an almost "embossed" gold! Turns out Woody had found an older *Omnicrom* machine scunging around unsold in the back of his warehouse and fired it up. Lo and behold, *the instant conversion of any toner image to real ink in stunning colors!*

The *Omnicrom* stuff has been a loser to date because the manufacturer was laboring under the delusion that you needed one of his $1300 heat fusion roller machines to use the *Omnicrom* process.

I got to thinking about just what their machine did, and concluded that it was no different whatsoever from the existing fusion rollers in the *Laserwriter*. Just for kicks, I ran some *Omnicrom* sheets back through the *Laserwriter* and it fused acceptably well.

So what is *Omnicrom*, and how does it work? Figure five shows you all. Toner is really a mix of two things:

black stuff and hot glue. Most people use toner to see the black stuff. Instead, you can think of any laser-printed (or for that matter, any xerox copy) as a sheet of paper that has had hot glue selectively applied in certain locations.

An *Omnicrom* sheet consists of a carrier that has a smooth layer of "real" ink applied to it in any of 60 colors. These sheets cost around 35 cents each, but any unimaged portion of any sheet can be reused later.

Once the word about the cubic goodness of this virtually unknown process gets out, you can probably expect competition and deep discounts. I predict the ultimate sheet cost will be less than a nickel, maybe less.

What you do is take the laser printed image and put the *Omnicrom* sheet in contact with it. You then apply heat and pressure, usually in the form of rollers at 160 degrees C.

The heat and pressure melts the hot glue, grabs the ink off the carrier, and gives you a solid color ink where the toner was.

Which leaves you with a bright, real-ink image that is actually more durable than the original toner. Of course, for high wear uses, you may still want to overspray a fixative or else use plastic laminating or page protectors. For most uses, the durability is just fine.

There are at least three ways you can transfer the ink. As we have just seen, the *Laserwriter* does a reasonable job of this, simply by shoving the page back through while doing a hand feed of a blank image.

You could instead buy the expensive custom fusion machine, and this might be a good choice for high volume work.

Finally, certain of those *Minolta* copiers are set up to directly handle the *Omnicron* material. In fact, any copier that will let you hand feed heavier stock should more or less work, provided it has a sane paper path.

The many uses include: Letterheads, Business cards, badges, bumper stickers, ad displays, and point of purchase signs. Also obvious are multiple color presentation charts and graphs.

For those places where you absolutely must have a solid black, just use some black *Omnicrom* on top of the black toner. This gives you excellent blacks for advertising copy and similar camera-ready needs.

The *Omnicrom* process also will give you an "instant negative" of your image as well. These are useful for such things as overhead projection sheets, dialplates, and printed circuit negatives. But, some fine detail may sometimes end up missing in the negative.

Sometimes, a slight excess amount of ink will be transferred but not fused. This may happen inside the "e" of very small text, or between halftone dots. To cure this, just get some 3M *Post-It Cover Tape* from your local office supply. Apply the tape very lightly and then gently peel it off. Presto, a perfect image usually results.

Yes, you can even handle pastel colors by using halftone screens, provided you are extra careful. You can also use this tape to process several different colors onto your copy at once. One fusion pass can give you a few to a dozen or more colors.

There are sixty colors in all. Around thirty of these are usually in stock, while more stock items are being added. Sheets are available in boxes of 100 of one color. Rainbow packs of 100 assorted sheets in metallic or normal colors are also available.

Besides plain colors and metallics, there are also golds, silvers, and a nice pearl. There is even a clear material. This will convert dull but acceptably black toner into a high gloss, and obviously "printed" black. It can also add a high gloss finish to the regular colors.

I have found the gold a tad garish for business cards. I instead use the more sedate and cheaper *Pale Bronze*. But the gold certainly is snappy.

Actually, the *Laserwriter* is not quite ideal for fusing *Omnicrom*. Before reaching the fusion rollers, there's some electrostatic charging and some edge feeding done along a long and not quite straight paper path.

(*continued on page* 23.5)

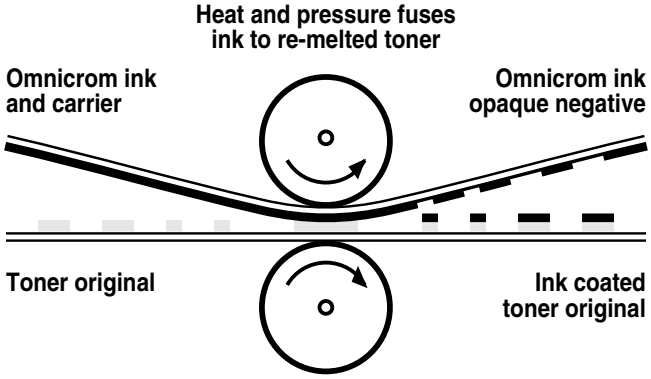


Fig. 5 - How the Omnicrom color process works.

Don Lancaster's

ASK THE GURU

January, 1987

Appleworks utilities
Postscript signatures
Help on Apple clones
Applewriter 2.1 Stretchifier
The most used Macintosh graphic

```
JSR $1000 ($3E25)  - Original booting link
JSR $1003 ($1186)  - Get program disk back
JSR $1006 ($11A1)  - Load or run program segment
JSR $1009 ($1341)  - Append string
JSR $100C ($1366)  - Erase screen window
JSR $100F ($136E)  - Scan keybuffer for escape
JSR $1012 ($139D)  - Init screen window
JSR $1015 ($14D1)  - Character to screen
JSR $1018 ($179D)  - Hex to decimal ASCII conversion
JSR $101B ($1815)  - Diagnostic hang

JSR $101E ($17D1)  - 16-bit divide u/v
JSR $1021 ($1818)  - Blork speaker & clear keybuffer
JSR $1024 ($1823)  - Relative cursor to absolute
JSR $1027 ($1837)  - Screen help prompt
JSR $102A ($1842)  - Flagged desktop full check
JSR $102D ($1850)  - Desktop full check
JSR $1030 ($1850)  - Bottom screen line prompt
JSR $1033 ($186C)  - Load and run desktop manager
JSR $1036 ($187A)  - Save screen line to $0900 buffer
JSR $1039 ($18B4)  - Grab stack passed parameters

JSR $103C ($191D)  - Get numeric selection
JSR $103F ($1A77)  - Hilight part or all of screen line
JSR $1042 ($1AFC)  - Close RAM bitmap
JSR $1045 ($1B00)  - Open RAM bitmap
JSR $1048 ($1B0B)  - Init applications screen
JSR $104B ($1B2B)  - Set change flag
JSR $104E ($1B34)  - 8-bit multiply x * y
JSR $1051 ($1B4E)  - 16-bit multiply u * v
JSR $1054 ($1B84)  - Move down in memory
JSR $1057 ($1BAC)  - Move up in memory

JSR $105A ($1BDF)  - Revert to previous screen
JSR $105D ($1BF1)  - Press space to continue
JSR $1060 ($1C21)  - Print string to printer
JSR $1063 ($1D0F)  - Open new screen
JSR $1066 ($1D35)  - Get keystroke
JSR $1069 ($1E80)  - Restore saved cursor position
JSR $106C ($1E8A)  - Save current cursor position
JSR $106F ($1EB4)  - Force character to upper case
JSR $1072 ($1EBF)  - Force text string to upper case
JSR $1075 ($1ED9)  - Compare text strings

JSR $1078 ($1EF8)  - Copy text strings
JSR $107B ($1F3E)  - Justified string to screen
JSR $107E ($2029)  - String to screen with number
JSR $1081 ($1FD1)  - Stall Y/8 seconds
JSR $1084 ($1FE0)  - Reset key buffer to empty
JSR $1087 ($2093)  - Message to screen at cursor
JSR $108A ($1FE9)  - Print one character
JSR $108D ($20AE)  - Print character full screen
JSR $1090 ($20BE)  - Print n identical characters
JSR $1093 ($1FF5)  - Message to bottom screen line
JSR $1096 ($20D6)  - Verification prompt
```

Fig. 1 - Appleworks Low Core Utility entry points.

Let's see. Just where did we leave off last time? There are bunches of news and rumors this month, so let's have at it.

Congratulations to Lynette Schnebly of Tucson, who has newly broken the all time Macintosh high score by managing to get the bomb icon to appear 4,671 times in a single marathon Mac work session. Using the knockout combo of *Macdraft*, *Appletalk*, and the various *Laserwriter* drivers all together.

Even if you can't enter such world class competition, there is a new and top secret Mac program that you probably will be interested in.

It is called *Passport*, and instantly will let you convert almost any IIc/IIe/IIgs file to almost any Mac file in either direction. Including word processor files, Appleworks, and even those EDASM assembly listings.

There are a few gotchas, though. The program only runs on the Mac side and then only with a Mac plus and a hard disk. You also have to work with 3.5 inch disks so they can fit in the Mac drives. Finally, the early versions are very buggy and are prone to blowups.

At this writing, the release of Passport is still hush-hush. So, you'll just have to inquire through the usual sources for your copy. Try out the gold channel with a R2 priority, by using a password of KNOCK-WURST.ALA.KING.

Five minutes beyond the deadline for last month's column, Apple dropped the price of the LaserWriter by $1000. Naturally, nobody would have the unmitigated gall to quote you the Apple list price of $4995. The Laserwriter street price is now somewhere in the $3450 area if you shop

around carefully enough.

What is the story on the Laserwriter plus? The only difference between a Laserwriter and a Laserwriter plus are eight missing EPROMS. These EPROMs do give you a few new and interesting fonts, but the yield and the service problems on these chips has been an absolute disaster for Apple to date.

Since these chips are both ridiculously overpriced and in fact are undeliverable, bootleg sets of the Laserwriter plus EPROMs are showing up in many larger cities at costs of $95 or less. Naturally, you can clone your own for $35 if you have a large enough EPROM programmer.

There is one thing that needs the record set straight on this, though. The Apple claim that the plus chips are required to download fonts is simply "that which is not so." You *always* had the ability to download any custom, unlocked, and low priced fronts from any old source on any Laserwriter.

The new chips now give you the *additional* ability to download overpriced, single sourced, and highly protected fonts as well.

Speaking of laser printers, did you see where HP shot themselves in both feet at once this time? The LaserJet always has been a hopeless cripple when compared to the Laserwriter, since it lacked any full screen, full resolution graphics, a useful page description language, the any-size-at-all procedural fonts, cubic spline smooth curves, flexible scaling, rotation, and translation transforms, and freedom from font cartridge ripoffs, to mention a few of the more obvious defects.

Rather than go with the industry standard *Postscript* page description language that would eliminate all these hassles in one swell foop, HP choose an obscure and arcane page description language from so far out in left field that it does not even provide for procedural fonts. And the ultimate kicker is that, at least for now, this language has to reside in the host and not in the printer.

Dumb.

One bright side to all this, though. There is nothing that anyone, anywhere could have conceivably done that would more positively and more absolutely guarantee *Postscript* gaining its well-earned place as the industry standard page description language.

Thanks, HP.

But this month's ultimate laser printing sick joke is that someone introduced a new high end laser printer with an internal 48 megabyte hard disk. The only little problem is that the operating system was written in UNIX, and it was so bloated that there was absolutely no room at all left in the 48 megabytes for a user program or application!

Rumor has it that the next generation Laserwriters just may appear sometime this spring. Expect an under $2000 four page per minute *Postscript* speaking personal laser printer using the new *Richoh* (Savin) engine, and a super

This patch is for ProDOS Applewriter 2.1 version AWD.SYS only. AWD.SYS runs only on the 80 column IIc or the 128K IIe.

It eliminates the "shortlines" caused by counting embedded printer commands as real characters.

For every escape character found and actually used in the line, two extra counts are added to the line length. This exactly compensates an embedded escape command followed by a single character.

The 2.1 Scrunchifier patch MUST be previously installed.

1. Make a third or higher backup copy of ProDOS Applewriter 2.1, using the filer utilities.

2. Get into /BASICS.SYS. Then CALL -151 to get into the monitor.

3. BLOAD AWD.SYS, A$2000, E$6020, T$0C, D2

4. Verify $49D1- 99 00 1C

Change $49D1: 20 CA 59

5. Verify $49F8- 4C DE 4B

Change $49F8: 4C D6 59

6. Verify $59CA- 11 11 11 11 11 ...

Change $59CA: 99 00 1C C9 1B D0
$59D0: 04 E6 75 E6 75 60 C4 7E
$59D8: F0 0E B9 00 1C C9 1B D0
$59E0: 04 C6 75 C6 75 88 D0 EE
$59E8: 4C DE 4B

7. Verify changes per 4,5, and 6 above.

8. BSAVE AWD.SYS, A$2000, E$6020, T$0C.

Fig. 2 - ProDOS Applewriter 2.1 Stretchifier patch.

deluxe high end version with zillions of built in fonts, a fast 32 bit microprocessor, a dual page memory, and all sorts of other neat stuff.

The software compatibility percentage for the IIgs continues to fall dramatically.

One recent casualty is the *Imageworks* card, because a three cent internal video connector was left off the IIgs.

Even *AppleWriter* has its problems. If you want to run stock ProDOS AppleWriter on a IIgs, the simpliest way to do it today is by providing your own super serial card instead of using the internal serial firmware.

I have been working on patches for IIgs *Applewriter*, and should have them for you shortly.

I'll be publishing the bare bones essentials here, while Don Thompson of *Thompson & Thompson* will add them and many more to his already excellent new utilities.

The new patches are an absolute joy to use. In fact, I am temporarily back on my IIe and it feels like I'm working in a tank of molasses. The speed and the smoothness is almost unbelievable, particularly on searches, WPL routines, long deletions, and on [A] screen formatting. The immediate availability of time and date and a humongous RAM disk are even better.

But the best of all is the built-in print buffer that lets you almost immediately return to processing your words while your printer goes merrily on its way.

This can be particularly important for *Laserwriter* error messages. Except for *Applewriter* on a IIgs, very few word processors anywhere have a ready-to-use and built in telecommunications capability, despite the great need for this feature in desktop publishing.

You also now have the *Appletalk* driver resident and on line, so you can network *Applewriter* with other computers and other printers.

Apple's new IIgs memory expansion card seems to have some interesting thoughts behind it. This is a very conservative and very low cost design, created by removing parts from the older slot 1-7 based IIe RAM card. It uses older 256K individual chips, rather than going the cheaper, smaller, more reliable, and more obvious route of using plug-in SIMM modules.

At $129, the pricing is very aggressive. The most likely reason for the older memory technology is so that any IIe owners will not be totally hacked off when they find out their older third party memory cards will not fit the IIgs. At least you can move 3/4 of the chips from your old card to the new one.

I'm still not fully happy with *Omnicrom* full color performance on the *Laser writer*, both because of its reliability and the wrinkling problems. Seems I was at a auction and lucked into an old beastie called a *Viewfax* Model six. The cost was $6, which is some $1294 cheaper than the *Omnicrom* fusion machine. I don't have the slightest idea what this machine was intended for, but I intend to try and modify it for use as an *Omnicrom* fuser.

The message here is that there are probably all sorts of obsolete office machines and worn out copiers scunging around that could be easily converted into suitable *Omni*crom fusers. What you need is two powered rollers and a temperatare of 170 degrees C.

Let me know if you can beat the $6 price on this.

Where are the Appleworks Low Core Entry Points?

As part of its booting process, *Appleworks* loads a module called APLWORKS. SYSTEM. This module is first loaded at the usual $2000 in main RAM and then is moved down to $1000 where much of it stays for the rest of the *Appleworks* work session. The part that stays can be called the *low core utilities*.

The low core utilities do provide around half of the fundamental low level subroutines needed by *Appleworks*. There are 51 low core utilities. Included here are the screen macro commands we looked at a few columns ago.

Figure one gives you the names of these utilities and shows you their entry points.

Where can I Get Help on Apple Clones?

I guess I am personally down on Apple clones. Over on the *Incredibly Boring Machine* side of the fence, clones are clearly the price and performance leaders, and only an epsilon minus would buy *IBM* rather than a clone.

In fact, big blue's onager is in a sling because their sales have dropped dramatically, the bottom has fallen out of their stock price, and key scapegoats have been given lateral arabesques. All this because of a blatant failure to either innovate or to be a price or performance leader.

On the Apple side, you have an odd mix of clones, ranging from serious and useful machines, through Hong Kong alley sweepings sold by refugees from a carnival midway, down to junior high school kids in a garage that claim compatibility without so much as checking out their bus noise or clock phasing.

Our helpline has been full of horror stories. One person had their "100 percent compatible" clone for six months before discovering that it had no HIRES graphics.

Another has a thermal intermittent that seems almost impossible to service. Yet another has a clone with bus lines so noisy as to be useless. Countless others have had really major problems with software compatibility.

I guess the worst problem is that the wrong people buy clones for the wrong reasons.

The key test on building an

Apple clone kit is this: If you normally use your own personal *Tektronix* oscilloscope more than two hours a day, then you *might* be able to successfully build your own Apple clone kit. If you use your personal oscilloscope less than two hours a day on the average, then you should not even think about it.

What is really sad about all this is that Apple could instantly eliminate the entire clone market with one simple and painfully obvious step. Just offer some *very* substantial trade-in allowances for Apple II or II+ owners when they upgrade to a IIe or IIgs. Do this, and the clones will all magically dry up and blow away.

Show me Another ProDOS AppleWriter 2.1 Patch?

Sure thing. In continuing our one-a-month series of patches for ProDOS *Applewriter 2.1*, in figure 2, you'll find the much asked for *Stretchifier* patch.

What this patch does is *not* count any embedded printer commands as part of a real line count. This eliminates the "shortlines" caused by extensive printer commands, especially while fill justifing.

Details on this patch for the older ProDOS Applewriter 2.1 have appeared in my *AppleWriter Cookbook*.

Incidentally, *Apple* has now quietly reinstated the $50 upgrade policy for owners of DOS 3.3 *Applewriter* who wish to upgrade to ProDOS *Applewriter* 2.0 or 2.1. You can contact them directly for more information. The upgrade delivery is apparently a tad on the slow side, but it is most definitely available.

How About Another Contest?

As usual, an *Incredible Secret Money Machine* to the best ten responses, along with an all expense paid tinaja quest for two (FOB Thatcher, AZ.) to the overall winner.

The new operating system for the Apple IIgs has the ability to select up to 128 disk drives, each of which can now address up to 4 gigabytes of on-line data.

Combined together, you now have the ability to place half a terabyte of data on line. The contest is simple: Show the best concept for a HUNT THE WUMPUS game using half a terabyte of memory.

Only to be completely fair about this, UNIX programmers are not allowed to enter.

How can I Sign my Name?

Why, with *Applewriter* on a IIc, IIe or IIgs, of course. Driving a *Laserwriter*. And, as is usual, helped along a tad by a utility or two of my own. How else could you possibly sign your own name?

In fact, the International Brotherhood of Forgers and Embezzelers Local #106 has voted *Applewriter* to be the greatest invention since blank counter checks.

Fig. 3 - An AppleWriter signature.

Figure three shows you a sample signature, while figure four shows you the simple *Postscript* code needed for the first name "John", and a sample printout. Oh, yes, you will also need a free copy of my new curvetracing routine, which you can get by calling or writing.

With some practice, it takes around 20 minutes to enter a signature, using nothing but a surprisingly few stabs from a safety pin. No scanners or

```
% John's Signature (Postscript)
% ..............................

1 setlinecap  /tension 2.8 def  /showtick false def

[87 82 100  80 130 85   93 158 0   94 122 -105
69.5 76 -135  58 69 160  60 81 65  115 120 0 ] curvetrace

[115 120 -160  106 110 -105  111 100 0
118 103 45  120 120 160 ] curvetrace

[120 120 -30  144 132 70  146 152 180  140 142 -110
128 100 -110  128 100 75  138 118 35  145 116 -75
147 100 0  155 110 60  165 120 0  160 100 -120] curvetrace

[160 100 55  177 120 0  176 100 0  188 110 45] curvetrace

50 100 translate 300 300 moveto

2 setlinewidth stroke
```

Fig. 4 - John's Signature.

digitizers of any type are needed. As you can see, the final results are absolutely exceptional.

If you wanted to, you could easily add pen skips and variable line weights, ink spatters, etc. to the signature grabbing process.

Are Applewriter Graphics Really That Good?

They sure are. I've seen the publisher of the leading Mac magazine state in his industry newsletter that if you want good graphics, you should definitely be making use of *Applewriter*.

At a recent trade show, a minicomputer company was introducing a high end new laser printer, so they plastered their entire booth with *Applewriter* graphics, printed, of course, on a *Laserwriter*. And it sure packed their booth full of customers.

But, I guess I will have to admit that, yes, there are flaws in *Applewriter* graphics. In fact, until this column, the single most often used Macintosh graphic image has been completely and utterly lacking from *Applewriter*, and just not available there in any way, shape, or form.

So, admitting defeat, in figure five you will find a *Postscript* listing that we can call *Applewriter's Glaring Omission*. You are right. It was absolutely inexcusable for the Apple IIc/IIe/IIgs community to not be able to share this mainstream Mac graphic. So sorry.

I've now got lots of new *Postscript* goodies available, some free, some not so free. Some will now even run under *Appleworks*, some on the Mac, and even some for one of those Incredibly Boring Machines. There's even some *Atari* stuff. Write or call for your usual free samples and whatever.

```
% Applewriter's Glaring Omission (Postscript)
% .....

gsave 1 setlinecap 5 setlinewidth 0 setgray  450 520
translate 27.5 rotate 8 {newpath 15 0 moveto 20 0
rlineto 10 0 rmoveto 12 0 rlineto stroke 45 rotate}
repeat grestore

newpath 0 setgray 300 400 100 0 360 arc fill gsave 1
setgray newpath 200 305 moveto 10 setlinewidth 200
0 rlineto stroke 200 495 moveto 170 0 rlineto stroke

grestore gsave 300 520 translate 1 setlinecap
newpath 0 0 moveto 50 100 125 -60 150 0 rcurveto
gsave 1 setgray 16 setlinewidth stroke grestore

12 setlinewidth gsave stroke grestore 8 setlinewidth
85 35 {dup mul exch dup mul add 1.0 exch sub}
setscreen gsave 0.99 setgray stroke grestore


gsave 350 340 translate 0 0 moveto 35 40 35 80 0 120
rcurveto 20 -40 20 -80 0 -120 rcurveto 1 setgray fill
grestore



 0 setgray newpath 270 511 moveto 60 0
rlineto  0 -30 rlineto -60 0 rlineto closepath fill

showpage
```

Fig. 5 - Applewriter's Glaring Omission.

(continued from page 22.5)

wrinkle or two can show up on the first pass. If you get a wrinkle, you just tape a small new strip of *Omnichron* material over it, and then shove it back through again.

You will want to tape the *Omnicrom* material to your copy at the top, being careful to keep the leftmost 3/8 inch free so the seperation belt and drive roller do not wrinkle the sheets. At present, you can expect around a 50 percent success rate on the first pass, and a 90 percent success on the second pass.

If you are using entire sheets of *Omnicrom*, be sure to trim them so they clear the mylar seperation belt and the drive rollers.

I suspect that some sort of carrier could be added to optimize *Omnicrom* for the *Laserwriter*. Something like a 9-1/2" x 11" mylar sheet folded 1/2 inch on either edge and sprayed with anti-static gunk or metallization might work out fairly well.

Mostly, I think it is just a matter of getting to know the material and working closely with it. Let me know all your experiences on this.

Tellyawhat. For the contest this month, just tell me about a copier machine that works with *Omnicrom*, or find any way to improve the reliability and convenience for direct *Laserwriter* use. A free *Incredible Secret Money Machine* to the best ten entries. As usual, we will provide an all expense paid tinaja quest for two (FOB Thatcher, AZ) to the overall winner.

UPDATE: *Kroy Kolor* has largely replaced *Omnicrom*. There is also a superb new laminating material that is now available. See column 25 and later for more info.

Write or call, and I'll be most happy to send you some *Omnicrom* samples to play with. We can also do custom printing for you using this positively mind-blowing new material.

Don Lancaster's

ASK THE GURU

February, 1987

Serial cables for the IIgs
Multiple video monitors
$5 toner cartridge reloads
Applewriter 2.1 creepifier
Finishing materials contest

The helpline has been rife with several heavy duty problems on the new 2.0 release of *Appleworks*. Apparently, several internal pointers can get badly fouled up, dropping parts of older 1.3 files as well as messing up fast cursor motions. It is not yet at all clear whether this is caused by the early "768K" memory access bug in the IIgs, or whether there are actual program bugs. There are also some helpline complaints over how the new mail merge feature works.

The dust has by no means settled on this, so go slow for now. Do not upgrade to 2.0 just yet. If you have already upgraded, drop back to version 1.3 for a while. The best and most current information on this probably will appear in *MAUG* on *Compuserve*.

Turning to a brighter side, the fatal flaw in the ProDOS 1.1.1 operating system has finally been stomped into nothingness. Many thanks to Tom Weishaar, to his readers, and to Tom's really outstanding *Open Apple* magazine.

It appears that some sloppily done initialization in ProDOS 1.1.1 would rarely and at random trash blocks 0, 1, or 2 of the ProDOS directory of your favorite and most used diskettes. The problem previously had been blamed upon poor power supplies, on *Appleworks*, on *AppleWriter*, and even on the price of yak butter futures. Quite a few power supplies seem to have been needlessly swapped out over this.

There is a quick and simple patch that once and for all eliminates this problem. Just contact Tom at *Open Apple* for your free copy.

Uh, I may have given you a bad *Omnicolor* phone number two columns back. Try using one of these . . .

(601) 264-2466
(617) 881-4100
(800) 447-2326
(800) 257-8263

Also, the *Viewfax* model six machine didn't turn out *quite* as good as I first reported. Good thing I am the training officer for our local fire department. Oh well. Its back to the drawing board on this one.

To best fuse the *Omnicolor* material, you'll need a line contact heat roller whose temperature is precisely regulated. There is some sort of anti wrinkle bar in the real *Omnicrom* fusion machine that in part explains why it does a more reliable job than the *Laserwriter* or *Laserjet*.

Well, I finally got the *Ask The Guru* reprint act together. You can now get corrected, edited, updated, and bound reprints of this entire series back to day one.

Thanks to a new custom thermal binding process, the reprints are always up to date and have a current index. You can write or call for details.

Our biggie this month reveals the key secrets behind those new $5 toner cartridge refills.

But first . . .

Tell me About the IIgs Serial Cables

There are four types of cables you will need when you use your IIgs in various printer, modem, and machine to machine configurations. Your Apple dealer does stock these for around $30 each. Let's see if we can not beat this price.

Figure one shows you the IIgs modem cable. This is used to go from a IIgs over to

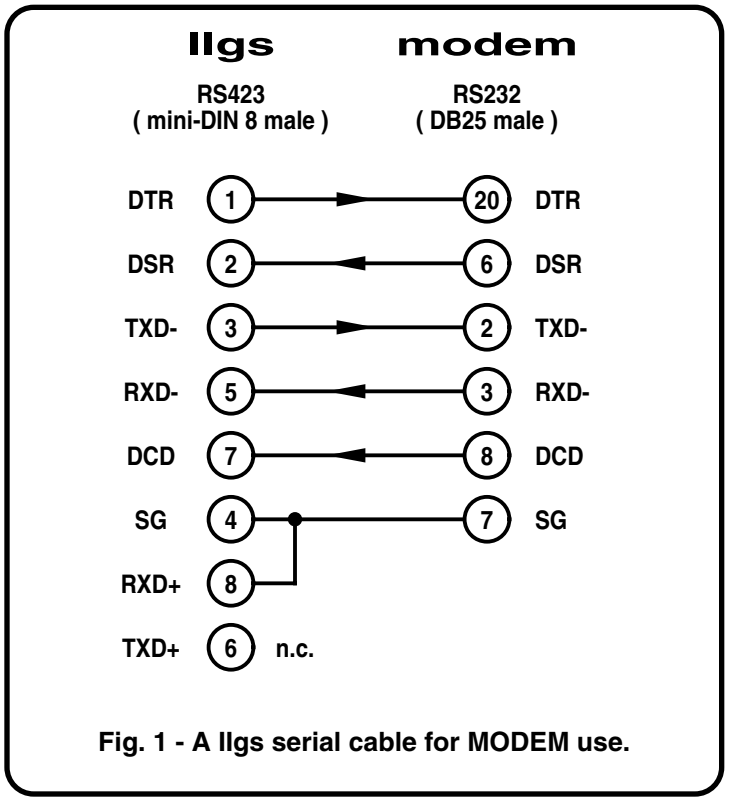


Fig. 1 - A IIgs serial cable for MODEM use.

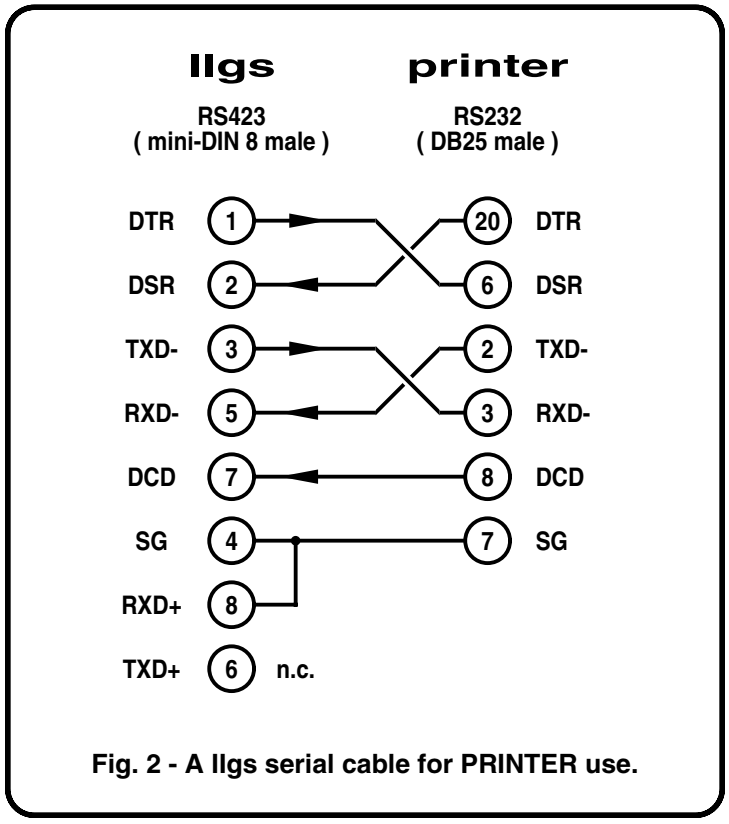


Fig. 2 - A IIgs serial cable for PRINTER use.

a RS232 "DB25" modem or for any other RS232 usage where you *do not* want to cross the data paths and the handshaking.

Figure two is the IIgs *printer* cable. This one gets used to go from a IIgs over to a RS232 "DB25" serial printer or for any other RS232 use where you *do* want to cross the data paths and the handshaking paths.

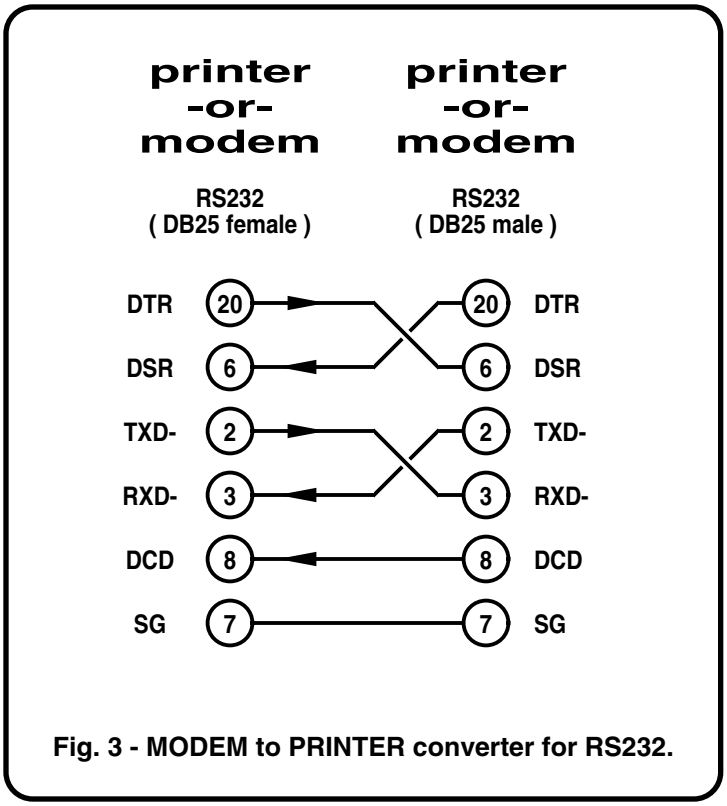


Fig. 3 - MODEM to PRINTER converter for RS232.

Figure three shows you a very simple DB25 *modem eliminator* that lets you automatically convert a modem cable back into a printer cable or vice versa. This eliminator is handy to have around, for there are almost always times when the cable you have will not be the one you want.

A most interesting RS232 *adaptor box kit* is available for $10 from *L-COM*. You can configure this jewel as the modem eliminator of figure three, or else solve just about any other RS232 interface problem with it. The kit is just a pair of male and female connectors and an attractive case. You can do the internal wiring any way you like.

Crome plating costs two bucks extra, for those of you that are into *Perrier* filled birdbaths.

Finally, figure four is the IIgs *machine* cable. This one lets a IIgs talk directly to a newer Macintosh, to another IIgs or to anything else that uses the same mini-DIN connectors and needs the crossed data and handshake paths.

Can More Than one Monitor be Connected to a Personal Computer?

Yes, provided you are careful enough about it. If you are lucky, you might even be able to put some of the monitors several hundred feet apart.

There are some gotchas involved, though. First, it is far and away best to run one continuous cable from monitor to monitor, rather than using a rat's nest of seperate cables from the computer to each monitor.

Second, just about all monitors have provisions to terminate or not terminate their inputs. Termination is done by putting a 75 ohm resistor directly across the video input. Sometimes, this is done with a switch at the

back of the monitor. Other times, you may have to use a solder jumper, or else a custom plug of some sort.

Regardless of the method used, *only the final monitor in the entire daisy chain should be terminated!* All of the intermediate monitors should be switched to their unterminated, high impedance mode.

Naturally, you should keep all your cables as short as possible, and never use a long cable without terminating its far end. For longer runs, you will get the best results with "real" coaxial cable and BNC connectors, instead of plain old audio cable and the usual RCA plugs.

For extremely long cable lengths, you might require a video buffer of some sort.

RCA has a 3450 video op amp that could be useful here. There's also a new product called the *Rabbit*, that's in the *Heath* catalog, among many other places. This gives you one way to extend VCR signals all over your house by cable. I am not sure if this product also handles baseband video.

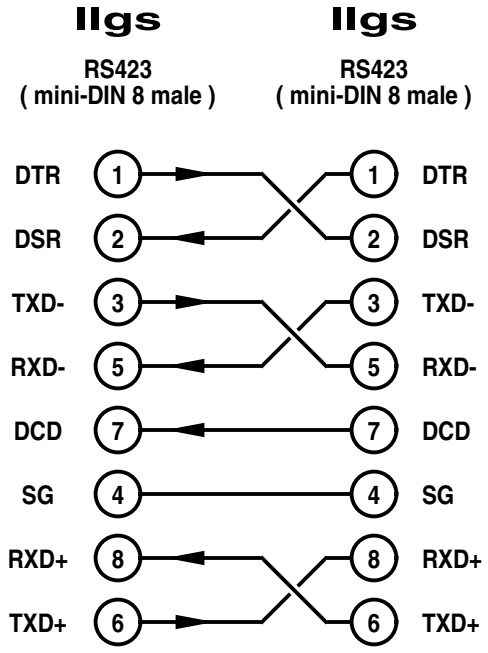


Fig. 4 - A IIgs serial cable for MACHINE use.

Show me another ProDOS AppleWriter 2.1 patch.

In continuing our ProDOS Applewriter 2.1 patch series, figure five is the *Creepifier* patch that can eliminate one form of page creep that will lower headers on successive pages.

This bug is infuriatingly subtle. It appears that an extra space used to be added at the end of any header or footer.

Big deal, right?

But, as countless of you seem to be doing, *if* you set your Applewriter right margin to 80 and *if* you leave your printer interface card set to 80, then you get page creep because of this bug.

Other obvious solutions to the page creep bug include (A) use an Applewriter right margin of 78 or less while using a printer card 80 column width, or (B) use any old

This patch is for ProDOS Applewriter 2.1 version AWD.SYS only. AWD.SYS runs only on the 80 column IIc or the 128K IIe.

It eliminates one form of page creep caused when the right margin and the interface card column width are both set to 80.

The patch works by eliminating an extra space that inadvertently got printed at the end of any header or footer.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.1, using the filer utilities.**
2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**
3. **BLOAD AWD.SYS, A$2000, E$6020, T$0C, D2**
4. **Verify $47F7- 20 5D 46**

   Change $47F7: 20 62 46
5. **Verify the above change.**
6. **BSAVE AWD.SYS, A$2000, E$6020, T$0C.**

Fig. 5 - ProDOS Applewriter 2.1 Creepifier patch.

Applewriter right margin you want, but set the printer card to an infinite, or at least much wider, right margin.

Tell me About the $5 Toner Cartridge Reloads.

The toner cartridges used in both the *Laserwriter* and the *Laserjet* laser printers cost nearly $100 each and are only good for 2500 copies, giving you an operating toner cost of around four cents per page.

Yet, you can easily reload these cartridges up to six or more times, by using a three minute process that costs $5 and needs no special tools.

If you are an owner of a laser printer, you'll gain dramatically lower toner costs while at the same time producing much blacker images.

These blacker images result first because the cartridgewill not get up to full blackness until *after* the second or third reload, and second, because the refill toner is most often a much denser black than the original.

If you do not own a laser printer, this could be an interesting and most profitable sideline business for you, since toner cartridge refills can easily be resold for up to $19 each.

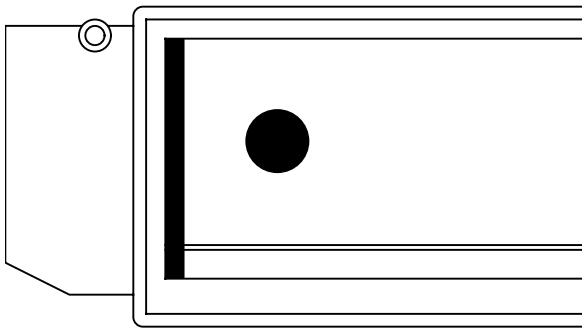


Fig. 6 - Add this hole to the fresh toner tank top.

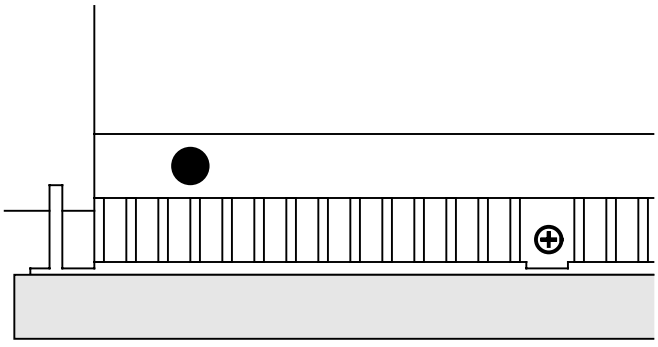


Fig. 7 - Add this hole to the spent toner tank bottom.

First, you will need a reliable source of refill toner. Three sources that I have found extremely useful are the EP-350 refills from most any *Minolta* copier dealer, kits and pads from *Laser Printer Products*, and bulk wholesale CX toner from the *Repeat-O- Type* folks. Their CX toner sells in bulk for as little as $5 per refill.

Around 170-225 grams of toner (0.4 pound) is needed for each reload.

One big gotcha here. It is extremely important to use monocomponent and *negative acting* toner that is intended specifically for organic drum laser printers. The ordinary copier toner refills most definitely will *not* work.

Two simple modifications must be made to the cartridge to allow refilling. Note that you do *not* have to disassemble the cartridge.

What you have to do is add a filling hole to the top of the fresh toner tank and then add an emptying hole to the bottom of the spent toner holding tank.

First, you pop off the large cardboard label by lifting an end with a pocket knife. The toner tank will be under this label. As figure six shows us, you melt a 1/2 inch diameter "C" shaped hole in the end of the tank using a soldering iron. Then, while the plastic is still hot, you snap this hole off and trim the bead with a pocket knife.

Be very careful to get no plastic chips into the hole and do not under any circumstances sand the hole or use steel wool on it. The actual refilling is done by using a plastic funnel and dumping one bottle of toner into this hole, all the while tilting and shaking slightly.

After refilling, scotch tape the hole shut and replace the label. Add a new label on top of the existing one and write the cartridge history on it. Include the date, the brand of refill, the number of the refill,

(continued on page 25.6)

Don Lancaster's

ASK THE GURU

March, 1987

IIgs Books and Manuals
Applewriter IIgs Patches
A Postscript Rubber Grid
More on the Omnicrom Color
De-putrifying Laserwriter Grays

If you at all like to keep score on who is doing what to whom, then get yourself a free subscription to *Computer Reseller* magazine.

For instance, we see that the Macintosh computer once again leads in unit sales, and the Mac people should be rightly proud of this. Why, this means that those Mac people have finally gotten to the point where they are a mere five percentage points behind the combined IIc and IIe unit sales.

And, as will come as no suprise whatsoever to long term *Ask the Guru* readers, the very best selling printer in the world today is – The Apple *Laserwriter*! That's in terms of dollar volume, while the *Imagewriter* printer leads in unit sales.

Epson does beat out Apple in total units sold, spread out over several machines.

And those sales figures are for *all* printers on *all* personal computers, as reported in a trade journal that has a "fair to middlin" bias towards the *Incredibly Boring Machine* side of the fence.

Our helpline bears out the unbelievable impact of the *Laserwriter* and the newer *Laserwriter Plus*. Today, the new people that are screaming for *Postscript* routines outnumber the "Hey, who could ever afford one of those?" calls by nearly ten to one.

Er, I told you so.

And, speaking of the *Laserwriter Plus*, Apple has finally cleaned up their act on the defective plus ROM upgrade kits. All kits are now factory installed and factory burned in. Some of these machines will even work. Reasonable dealers are now charging as little as $80 to $100 extra for the *Laserwriter Plus*, which, strangely, exactly happens to match the price of the bootleg plus ROM sets.

I suspect that the "unplus" version might be dropped entirely when these new machines arrive this spring.

Incredibly, the new low end "personal laser printer" is rumored to *not* be able to speak Postscript, and to be strictly a *Quickdraw* machine. This is so stupid that it could not possibly happen. Particularly since everybody and their brother are now offering new snap-on Postscript interpreters for just about every low end laser printer.

I sure get a lot of calls from people who've had serious problems with the older plus ROMs. One major sign of trouble is broken, truncated, or misspelled error messages, caused by the frame device blowup. Some machines have taken as long as ten months to correct. Hopefully, all of this is ancient history.

Uh, let's tie up a few other loose ends before we untangle some new ones.

Hats off to the Japenese who recently told the US recording executives to take their inane and useless copy protection scheme for digital audio tape and shove it sideways. Score one for the good guys.

I've just had a lot of fun going through *Infocom's* new X-rated *Leather Goddess of Phobos* text adventure. Its more or less up to their usual high standards, and I particularly liked the part with the horse and Catherine the Great. But, it would be much more interesting if the roles of Tiffany and Trent were interchanged.

Only, Infocom, there is one big thing I have noticed that someone really ought to bring to your attention. You have

Apple IIgs Assembler Tool Interfaces

Apple IIgs C Tool Interfaces

Apple IIgs Firmware Reference

Apple IIgs Hardware Reference

Apple IIgs ProDOS 16 Reference

Apple IIgs Programmer's Workshop

Apple IIgs Toolbox Reference, volume I

Apple IIgs Toolbox Reference, volume II

Programmers Introduction to the Apple IIgs

Programming the 65816

Technical Introduction to the Apple IIgs

The Apple IIgs

Fig. 1 - Some Apple IIgs Books and Manuals.

two types of text adventures, "unprotected" ones that anybody can run from any copy, and the "protected" ones that demand certain oddball materials, maps, clues, etc... in the original package.

Ignoring the fact that most of these materials are easily misplaced, particularly in any family with several children, have you ever noticed that the *all* of your "protected" games are absolutely and irrevociably inferior and second rate to those that are unprotected?

Consistently, on adventure after adventure, with no exceptions whatsoever?

Apparently, some of your game authors are so hung up over protection that they've forgotton to be creative, playable, or even humorous. Let's flush this "protection racket" once and for all, and get on with reality. This is a serious disservice to your customers.

That *Passport* program we looked at a few columns back that lets you transfer IIc, IIe or IIgs files to Mac and back? The *APDA* people have it for $10. A must have for sure.

By the way, some of the old *Macintosh* power supplies will melt if you do your upgrade improperly. Which isn't quite as bad as all those *Compaq* machines that can literally explode when you try the same thing. See your dealer or user group for some more details on either of these.

For the name of your local Apple user group, either check out the listings right here in *Computer Shopper*, or else call toll free (800) 538-9696, Extension 500.

Time for a brief 23 word message from our sponsor: Bound *Ask the Guru* reprints are now available from me back to day one. Write or call if you want a complete set.

This patch is for ProDOS Applewriter 2.0 version AWD.SYS only. AWD.SYS runs only on the 80 column IIc or the 128K IIe.

The patch allows printing on a IIgs by defeating any attempts at setting serial data values to a non existant 6551 port chip.

It works by trashing the i.d. bytes for the super serial card and by aborting any [O]-J.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.0, using the filer utilities. Plainly label this disk FOR IIGS ONLY!**
2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**
3. **BLOAD AWD.SYS, A$2000, E$6020, TSYS, D2**
4. **Verify 4DB0- A0**
   Change 4DB0: 60
5. **Verify 4F67- 01**
   Change 4F67: 10
6. **Verify 4F6E- 31**
   Change 4F6E: 13
8. **BSAVE AWD.SYS, A$2000, E$6020, TSYS.**

Fig. 2 - ProDOS Applewriter 2.0 IIgsifier patch.

What IIgs Publications Are Available?

Figure one lists all of the major Apple IIgs technical books and reference manuals that either are now available or else shortly will be. While many of these should be in stock at your local bookstore, the best source is the *APDA* people. You can even arrange with them to automatically receive *all* of the IIgs materials the instant they become available.

More on Apple's non-IIgs publications next month.

What's the Latest On Omnicrom Color?

So much is happening so fast in the printing of "funny stuff" on copier machines and laser printers that it is safe to start ridiculing those poor unfortunates who still insist on black toner images on plain old white paper.

The *Omnicrom* full color process has been taken over by the *Kroy Sign Systems* people, and has been renamed *Kroy Kolor*. Which should mean much wider availability and much lower prices.

Several add-on products, including a cheaper fusion machine, a binding system, sign holders, and laminating materials are now also newly available. For more details and samples, contact Randy Bailey over at *Kroy*.

Four tips on the original *Omnicrom* stuff: (1) Stick with those metallic colors, because they will fuse much better. (2) Use a lightly coated paper stock, such as *Paloma Matte* from *Butler Paper Inc*, (3) Seperate the image from the carrier as soon as possible after fusion, and (4) An iron can be used for the transfer, provided you use a pressing cloth or an additional sheet of paper.

And there's lots of other new products that beautifully use toner for new and wondrous things. Some new mylar sheets are available from the

Meadowlake people that will let you immediately transfer toner onto a printed circuit board where it can be used as an etch resistant. A hot iron is used for the transfer. Cost is under a dollar per sheet.

This will ridiculously simplify hacker printed circuit board construction. It also means that you can now transfer toner to virtually any surface that will sit still long enough for a toner transfer from a hot iron. Yes, you can use *Kroy Kolor* on top of a *Medowlake* transfer.

You will find many more details on this in my *Hard*ware Hacker column over in *Modern Electronics*.

And for some even more intriguing materials, do check into those translucent and opaque vinyl *Form-X Films* from *Graphic Products* that accept toner and cost around 70 cents a square foot. Signs were never easier.

Why not print toner onto diffraction gratings, or gold foils, plastics, whatever? For full information, just contact the *Coburn* people who have a very wide variety of these materials available at most reasonable prices.

Write or call me for some free samples, or just use the *Ask the Guru* helpline to keep up to date on the incredible happenings in this new field.

Can I run Applewriter On an Apple IIgs?

Except for one tiny and nit-picking detail, ProDOS Applewriter 2.0 or 2.1 runs reasonably in the fast mode on a IIgs. The only trivial hangup is that Applewriter blows up the IIgs when you try to print.

I can't imagine any of you *Ask the Guru* diehards fussing over such an insignificant bug, but for those of you purists and perfectionists out there that absolutely insist that a word processor should really be able to print as well as to process words, a few minimal printing patches are shown you in figures two and three for ProDOS Applewriter versions 2.0 and 2.1.

These patches perform by defeating the testing made for the Super Serial Card. If this test fails, no 6551 serial port firmware will be assumed, and no damaging pokes will be made to exactly the wrong place in the IIgs. Thus, no blowups will occur.

Be sure to use the control panel on the IIgs to set your printer values. Option [O]-J is no longer active.

By the way, be sure to use the *printer* cable connection from last month to get proper printing using *Applewriter* on a IIgs. There are lots more IIgs patches to follow, so be certain to stay tuned.

This Month's Contest . . ?

We will split this month's contest so you'll have two wildly different ways to win.

Just show me how to win the swordfight and solve the angle problem over in *Leather Goddess of Phobos*. Or else tell me the exact Centigrade and Farenheit equivalents for the normal home iron settings of cotton, wool, synthetics, silk, etc . . .

As usual, a free *Incredible Secret Money Machine* to the ten best entries, and an all expense paid tinaja quest for two (FOB Thatcher, AZ) to the overall winner.

What is Postscript?

Postscript is that exciting and new page description language from *Adobe Systems* that is rapidly becoming the de-facto desktop publishing standard for laser printing and typesetting.

There are plenty of highly outstanding advantages to Postscript. First and foremost,

This patch is for ProDOS Applewriter 2.1 version AWD.SYS only. AWD.SYS runs only on the 80 column IIc or the 128K IIe.

The patch allows printing on a IIgs by defeating any attempts at setting serial data values to a non existant 6551 port chip.

It works by trashing the i.d. bytes for the super serial card and by aborting any [O]-J.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.1, using the filer utilities. Plainly label this disk FOR IIGS ONLY!**
2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**
3. **BLOAD AWD.SYS, A$2000, E$6020, T$0C, D2**
4. **Verify 4DC7- A0**
   Change 4DC7: 60
5. **Verify 4F7E- 01**
   Change 4F7E: 10
6. **Verify 4F85- 31**
   Change 4F85: 13
8. **BSAVE AWD.SYS, A$2000, E$6020, T$0C.**

Fig. 3 - ProDOS Applewriter 2.1 IIgsifier patch.

it is fun to use. In fact, the language is downright addictive. Postscript generally lets you build much higher quality images much more flexibly and with incredibly more power.

Another major advantage is its device independence. This means that the very same textfile that is sent to a laser printer can later be sent to a typesetting machine for much higher print resolution.

Text and graphics can be mixed up in any manner anywhere on the sheet to the full available printer resolution. You can very easily translate (move), scale (to magnify or reduce), or rotate (twist) any image any way you want to. You can even arbitrarily map any text and graphics image onto virtually any surface.

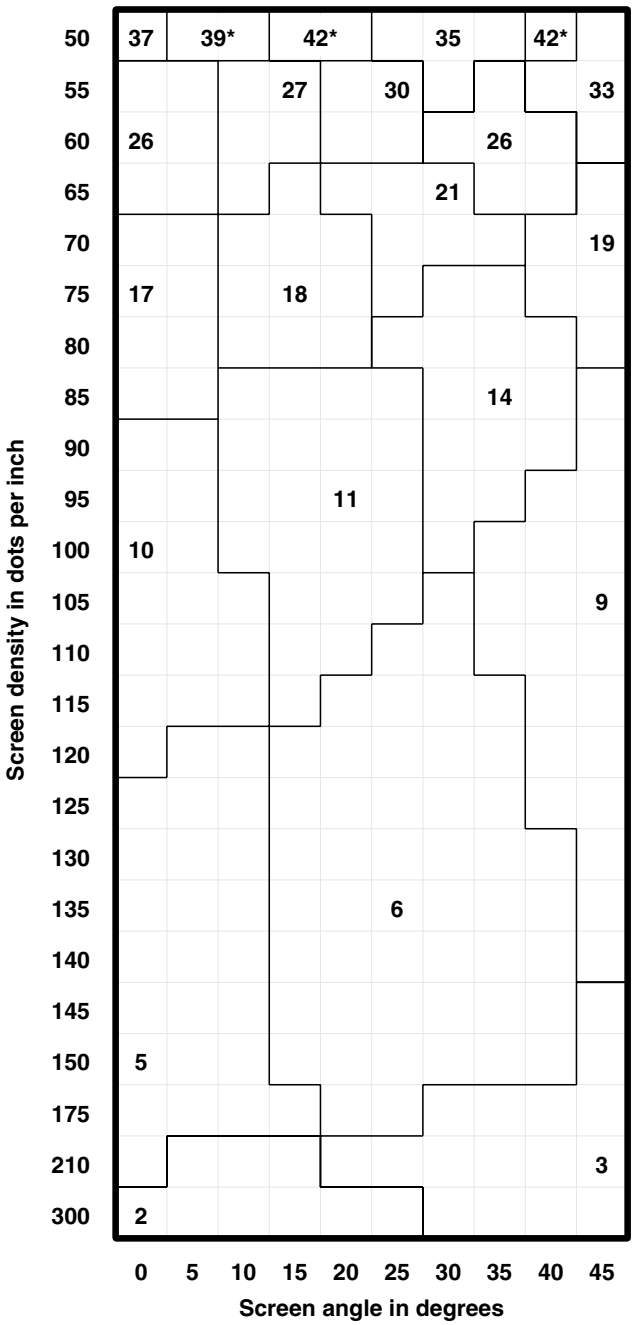


Fig. 4 - A map of the secret Laserwriter grays.

Postscript is both threaded and extensible. Which means that you can simply add your own custom routines that can become an integral part of the language. Thus, you can very easily customize Postscript to do what you want in exactly the way you want it done.

As many of you already know, all of the *Ask the Guru* text and graphics is done by using Postscript, and working from the *Applewriter* word processor that is driving a *Laserwriter Plus*.

Many hundreds of Postscript fonts are now available, and any individual font can be shown any size from 3 point to 65,000 point. (72 points equals 1 inch) Thus, you can letter anything from a tiny model railroad sign to the name on your town's water tower, all done with a *single* procedural font.

Fonts are easily stretched in any direction, leaned, outlined, clipped, or modified for other special effects. You can even create your own very high quality fonts.

Postscript does have a very strong cubic spline drawing and curve tracing ability that can let you draw smooth and continuous curves six ways from Sunday. Photographs and grays are easily included, although the final halftone quality depends on the actual printer you have in use.

In fact, it is Postscript that has made Apple's *Laserwriter Plus* the best selling printer in the world today.

There's several good ways to get started with Postscript. Two methods do include the *Postscript Cookbook* and the *Postscript Reference Manual*. If you can't find these locally, I have a few extra copies on hand here. No, I didn't write them, but I sure wish I had. My own Postscript books are still in the works. You'll also want to subscribe to *Colophon*, which is Adobe's free Postscript newsletter.

I have lots of free printed Postscript goodies, listings,

demos and routines that you can run if you call or write. I have also got a fancy *Postscript Show and Tell* that now will run under just about any word processor, editor, or comm program, and do so on just about any computer.

You can also use the *Ask the Guru* phone number for free Postscript help.

And, if you want to tackle a really advanced hacker project, just write and then debug your own Postscript intrepeter for the *Imagewriter*. I must get around thirty help line calls a day requesting this. The language itself has been placed in the public domain, so there's nothing stopping you except for some personal time and effort.

Tell me About the Secret Laserwriter Grays.

Have you ever wondered why the *Macintosh* grays look so putrid when printed on a *Laserwriter*? Or why any fine gray grids on the Mac nearly always have hideous dropouts in them? Well, its all because much of that Mac software absolutely insists on using the *seventeenth* lousiest gray that is available from the *Laserwriter*, instead of selecting something decent looking.

Grays for the Laserwriter are specified by picking both a *screen density* and a *screen angle*. These can be changed with the *setscreen* command.

Because of the odd integer arithmetic involved, not every combination of screen density and screen angle is possible. Figure four shows you a map of many denser *Laserwriter* grays, while figure five shows you how simple it is to modify your Postscript code to pick up grays so good that they will even look like they were done by using *Applewriter* on a IIgs.

The numbers show you the total number of gray shades (including black and white) you will get for any selection inside any region. That number is centered on the exact

```
BEST FOR 1:1 COPIES AND GRIDS . . .

        105 45 {dup mul exch dup mul
        add 1.0 exch sub} setscreen


BEST FOR REDUCED PREPRESS . . .

        85 35 {dup mul exch dup mul
        add 1.0 exch sub} setscreen


INDIA INK WASH EFFECT . . .

        135 25 {dup mul exch dup mul
        add 1.0 exch sub} setscreen


DEFAULT (urp) . . .

        53 45 {dup mul exch dup mul
        add 1.0 exch sub} setscreen
```

Fig. 5 - Some of the best Laserwriter grays.

```
% rubbergrid (Postscript)

% enter first with (xaxis) (yaxis) (size) setgrid ...

/setgrid { gsave 300 72 div mul 4 div round 288 mul
300 div /size exch def translate size dup scale} def

% enter later with (#hlines) (#vlines) showgrid ...

/showgrid {gsave /#vlines exch def /#hlines exch def
100 45 {dup mul exch dup mul add 1.0 exch sub}
setscreen 0.9 setgray /gs 1 def /lw 1 def drawlines
/gs 5 def /lw 3 def drawlines /gs 10 def /lw 5 def
drawlines grestore} def

/drawlines {72 300 div lw mul size div setlinewidth
/hpos 0 def #hlines gs div 1 add cvi { hpos 0 moveto
0 #vlines rlineto stroke /hpos hpos gs add def}
repeat /vpos 0 def #vlines gs div 1 add cvi {0 vpos
moveto #hlines 0 rlineto stroke /vpos vpos gs add
def} repeat} def

% //// demo- remove before use ///

/xaxis 100 def /yaxis 200 def /size 16 def
xaxis yaxis size setgrid

/#hlines 20 def /#vlines 10 def

#hlines #vlines showgrid showpage
```

Fig. 6 - Dropout-free Postscript gray rubber grid.

screen density and the screen angle that will result.

The stock grays use what is called a 60 line screen, but it really only gives a "Sunday funnies" density of 53 dots per inch. An excellent set of grays for most reducing reproduction use an 85 line, 35 degree screen, while an all purpose "final copy" gray is done using the 105 line 45 degree screen.

For 300 DPI photos, you may want to experiment with the 75 DPI, 15 degree screen or else the 65 DPI, 30 degree screens. These will give you 18 and 21 gray shades.

Photos good enough for, say, a realator's listing, a car shopper, or a catalog cut can be gotten at 300 DPI with enough care and attention to detail.

You will, of course, get far better photos if you either reduce your final image or else set the final copy on a *Linotron 100* or other higher density Postscript machine.

Finally, if your paper can handle it, and if six somewhat darker grays are all you need, then try out a 135 line, 25 degree screen. This will give you grays that look like an india ink wash, and can even make the *Laserwriter* output rival the *Linotron* 100.

Show me a Postscript Utility.

I will even do you one better. Starting in this issue, we'll run at least one most needed *Postscript* utility each month. I'll even be careful to make sure most of these will run on virtually any Postscript printer driven from just about any mini or microcomputer.

This month's Postscript utility is the *gray rubber grid*. The listing appears in figure six. This gives you an adjustable gray grid background for such things as schematic diagrams, flowcharts, graphs, invoices, or whatever. This grid ends up both dropout free and glitch free.

All the code that follows the rubber grid routine will be locked to it, and can easily be expanded and contracted as needed. You can also lock the grid to automatically track a border or inset.

Several gotchas: The grid works best at 1:1 and may not scale well. Also, those in-grid font sizes will typically be very small. Use 0.7 point as a starting baseline.

Your grids can be easily turned on or off by putting or not putting a percent sign in front of each *showgrid* line.

I do have bunches of Postscript stuff in stock for you here, some free and some not so free. Write or call for the usual listings and whatever.

Also, be sure and let me know if you want to see a quasi public domain Postscript program exchange BBS started up.

(continued from page 24.4)
and any drum defects are then noted.

To provide an emptying hole in the holding tank, turn the cartridge upside down and find the area shown in figure seven. Then, melt a 1/4 inch hole as pictured. This hole edge can be sanded smooth without major problems.

Finally, go outside and shake the used toner out of this hole. Since there are some baffles in the holding tank, you will have to rock the cartridge back and forth a few times to get rid of all the spent toner. When finished, tape the hole shut with scotch transparent tape.

For successive refills, just untape the toner holding tank, empty and retape. Then, untape the fresh toner tank, fill and retape. You should also now use the special tool to clean the corona wire.

Instead of replacing your fusion wiper pad, just remove the wiper portion of it and lay a new wiper in place. The wipers cost around fifty cents.

Keep the toner cartridge right side up as much as possible when you perform your refilling. It is definitely *not* feasable to try and ship any refilled cartridges, except by hand carrying.

There is absolutely no need to take the cartridge apart. But, if you really want to open one, the special screwdriver needed is the *#10 tamper proof torx* insert bit found in *EVCO's* 945B700 set. This set is also available at *Jensen Tools*.

How About Another Contest?

Sure thing. We'll have an easy one this month. The next big market to open up will be the area of *finishing processes and materials* for use with laser printing. Sure, the laser pages are nice, but it is the complete final package that really counts.

So, just tell me about any finishing products that are suitable for small quantity laser printing. I am particularly interested in low cost corner rounding punches, unique cover materials, binding systems, color options, unusal papers, etc. Any stuff that exists but is obscure enough that nobody knows about it.

As usual, a free *Incredible Secret Money Machine* to the best ten entries, and an all expense paid tinaja quest for two (FOB Thatcher, AZ.) to the overall winner.

UPDATE: My favorite refill toner source today is through Arlin Shepard at *Lazer Products* in Colorado. Current pricing is $7.60, which includes a new fusion wiper pad. Don Thompson of *Thompson and Thompson* now has an independent toner testing service that reports on the best refill toner quality and price on a weekly basis.

Remember that this is your column and you can get technical help, gossip, and off the wall networking per the end box. Be sure to phone or write for your copy of the new free stuff list, and your *Laserwriter Demo Pack*.

Don Lancaster's

ASK THE GURU

April, 1987

Mass teleportation
Postscript resources
IIgs old drive adaptor
Applewriter tweedleifier
Apple technical literature

Apple has just released a new version of the IIe. It is now packaged in a platinum color case and now uses the IIgs keyboard, including the numeric keypad.

A few other changes were made to reduce the production cost and standardize the use of IIgs type components.

Instead of eight 64K x 1 memory chips, you now have two 64K x 4 dynamic RAMS. And, instead of that pair of 64K monitor ROM chips, a single 128K chip is used.

The list price is $829, and the dealer cost is half that.

Except for the brand new numeric keypad and a newly introduced and very bizarre bug, the performance of the new IIe should be identical to the older units. The power supply might possibly run very slightly cooler, and a tad more power might now be available for peripheral cards.

The very bizarre new bug resulted from a change to allow mouse cards to automatically sense the shift key. If you have any peripheral that uses the SW2 input and if you simultaneously press the shift key, then you throw a dead short onto the power supply, trashing everything currently in the machine.

Uh, Apple there is this product out there that has a linear voltage and current relationship and is able to automatically restrict currents to selected levels. You might not have heard of them, but these are called *resistors*, and they cost around half a cent each in production quantities.

Why, even *IBM* has made use of this emerging new technology. A complex set of math theory has also been developed and seems to stand up to experimental verification. It is called *Ohm's Law*.

Apple apparently made the IIe look *very* similar to a IIgs, and have narrowed the price differental between the two machines to a mere $170.

Any disinterested outsider might conclude that Apple is trying to quash the identity of the IIe and eventually replace it outright with the IIgs.

Just like they quashed it with the *Lisa* and the *Apple III* machines. Uh huh.

Apple has also dropped the *EDASM* assembler and their *ProDOS Assembler Tools* disk (#A2W0013) from the current developer services listings.

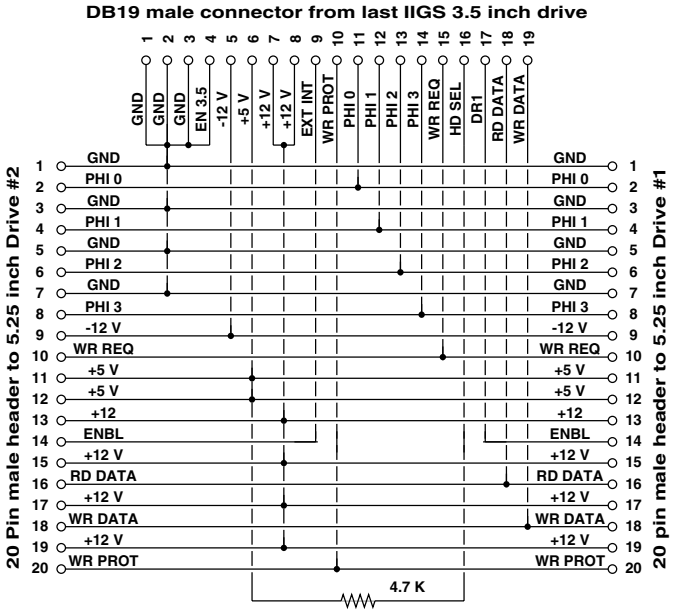
Apparently they feel that everyone is now immediately supposed to shift to the new "C" language development tools and use a sophisticated assembler named *ORCA/M* that can handle the full resources of the 65C816.

Several CS readers have questioned the safety of toner cartridge refilling. While the toner has not yet been shown to cause laboratory hamsters to become excessively horny, if you were purposely were trying to develop a human lung carcinogen, you probably would pick a very fine black powder full of soot and fancy organic chemicals.

A little common sense is in order here. Do the refilling outside and in a slight breeze. Hold your breath or wear a mask when actually dumping the toner into the cartridge or emptying the holding tank.

For this month's contest, just tell me of a cheap, local, and no-nonsense place to buy the specialty papers that are suitable for laser printing. Out here in Arizona, *Paper Plus* is the outstanding winner.

What we need for future column coverage is some nationwide listing of similar places. As per usual, a free *Incredible Secret Money Machine* to the best ten entries, and an all expense paid tinaja quest for two (FOB, Thatcher, AZ) to the overall winner.



Circuit 1 - Apple IIgs old disk drive adaptor.

So much is happening to *Postscript* this month that I don't even know where to begin. Sources close to the barber of an associate of a usually reliable spokesperson seem to feel that a major overhaul of *Postscript* is in the works and in the hands of some super secret developers. The new code is purportedly much faster, particularly with regard to character imaging and in the handlng of down-loaded bitmaps and similar images. Presumably the new product will be retrofittable to earlier *Laserwriter* machines through a simple chip swap.

About your Enhanced IIe (030-1141)
Apple Color RGB Monitor Manual (030-3106)
Apple II Dot Matrix Printer Manual (030-0739)
Apple II Memory Expansion Card Man. (030-1209)
Apple II Monitor Users Manual (030-0598)

Apple II Mouse Users Manual (030-0243)
Apple II Numeric Keypad Manual (030-0463)
Apple II System Utilities Manual (030-2028)
Apple II Utilities guide (030-1264)
Apple IIc Memory Expansion Card Man. (030-1330)

Apple IIc Owners Manual (A2L4038)
Apple IIc Owners Man & System Util (030-1030)
Apple IIc Scribe Users Manual Part 2 (030-0993)
Apple IIe Enhanced Owners Manual (A2L4073)
Apple IIe/IIc Color Monitor Manual (030-1224)

Apple IIc Mouse Manual (030-0961)
Apple IIc Reference Manual part I (030-0814)
Apple IIc Reference Manual part II (030-1022)
Apple IIc Technical Reference Manual (17728-5) *
Apple IIe Guide to the New Features (030-0622)

Apple IIe Hand Controller Manual (030-0200)
Apple IIe Joystick Manual (030-0563)
Apple IIe Numeric Keypad Manual (030-0564)
Apple IIe Technical Reference Manual (17720) *
Apple IIgs Manual (030-1292)

Apple IIgs Setup Guide (030-1294)
Apple II+ Vital Importance Manual (030-0209)
Apple III Dot Matrix Printer Manual (030-0740)
Apple III Monitor Manual (030-0193)
Apple III Read Me Manual (030-0662)

Apple Monochrome Monitor Manual (030-3110)
Apple Numerics Manual (17741-2)
Apple Personal Modem Users Guide (030-2001)
Apple 5.25 Drive Owners Manual (030-2040)
AppleColor Monitor Manual (030-0976)

Appleline Manual (030-0975)
Applesoft Reference Manual (17222-6) *
Applesoft Tutorial (17724-2) *
DMP Reference Manual (030-0607)
BASIC Programming (17721-8) *

Cable and Connector Manual (030-0623)
DOS 3.3 Manual (030-0115)
DOS Programmer's Manual (030-0209)
Domestic 9 inch Monitor Manual (030-0976)
Duodisk Owners Guide (030-0715)

Fig. 2A - Apple technical books and manuals .

Here at *Computer Shopper*, we have already started a once-a-month listing of the *Postscript* utilities, beginning with last month's column. Starting with this issue, you'll also find a *Don Lancaster's Laserwriter Corner* sidebar that will give you short and simple *Postscript* ideas and problem solutions that are very hard to find elsewhere.

There's also now a new national *Postscript* on line bulletin board. It is open and free to all beginning users. Advanced users can get at the real goodies at a higher level on a "put one on, take ten off" basis. Commercial ads are welcome and placable here at a nominal charge.

All of the many *Postscript* routines in this column will be available on this BBS on a downloadable basis, and I'll try to add some extra goodies of my own from time to time. The board is independent, so hacks and cracks should also be listed.

Oh, yes, the *Postscript* BBS phone data number is (409) 244-4704 and is up 24 hours a day. Standard baud rates are 300, 1200, and 2400.

There is also a brand new magazine called *The Post-script Journal*. This one looks like it is going to have some great goodies in it, but it is still too soon to tell. The cost is $15 per year to all charter subscribers.

Briefly turning to another of those insidious advertor-ials, my own unlocked and unprotected *Postscript Show and Tell* is in stock here for the Apple, Mac, and the IBM PC. Now newly available is an *Introduction to Postscript* user group videotape I put together that includes toner cartridge reloading, *Kroy Kolor* details, plus bunches more. It is in VHS format.

And the usual reminder that my *Applewriter/Laserwriter Utiities* and the complete set of reprints to this *Ask the*

Guru column remain available. So are the usual free laser printing demo packs and other goodies. We return you now to our program already in progress . . .

How can I use Older Disk Drives on the Apple IIgs?

The *Apple IIgs* has an incredible variety of disk drive options. For instance, there is a plug-in *SCSI* card that can access any number of floppy or hard disk drives that make use of this optional interface.

Then there are all the RAM disk plug-in card opportunities. Today, you could go as high as 8 megabytes of RAM disk. Just as soon as the prices drop a tad more on the 1 megabyte RAM chips, you can quadruple your storage up to 32 Megabytes.

In fact, you can do it today. One megabyte RAM chips are certainly available, but they are not yet cost competitive with older 256K dynamic RAMs.

And, should you attach a single "intelligent" drive to the usual DB19 disk connector on the IIgs, you can control up to 128 different drives at once. This would, obviously, take a beefed up power supply, but it can definitely be done.

Astoundingly, the present firmware (but not the drives themselves) is capable of supporting up to five gigabytes per drive, for the mind numbing total of a *half a terabyte* of data on line at once!

Which is not too shabby for a personal computer. And particularly since next year's CD ROM's should drop in just fine, thank you.

The normal and usual setup for most IIgs users is to daisy chain four dumb disk drives, placing a pair of 3-1/2 inch drives nearest the IIgs, and using a pair of 5-1/4 inch drives as tail end Charlies.

The only little problem here is that the IIgs and all of the newer Apple drives use a DB-19 connector. But, many of those older Apple drives used a 20 pin DIP header connector instead, that has wildly different pinouts. What can be done here?

There are at least two ways you can use these older 20 pin drives on the IIgs. The quickest and simplest is to go ahead and use an old slot six controller card, and change the IIgs front panel selection to "your card" for slot six.

Instead, figure one shows you a simple adaptor that you can build that will let you connect a pair of older 20 pin drives to a single DB19 connector. This connector can either be plugged into the IIgs itself for a 5-1/4 inch only system, or else into the last 3-1/2 inch drive in use.

This adaptor is certain to become commercially available soon, but for now, you might want to build your own. I would start with a small printed circuit board, being sure to use the type of 20 pin DIP header that has a plastic box around it.

Graphics Tablet Manual (030-0076)
Hard Disk 20 Manual (030-1228)
Hard Disk 20SC Manual (030-3138)
Imagewriter II Appletalk Manual (030-1313)
Imagewriter II Owners Manual (030-2002)

Imagewriter II Sheetfeeder Ins. (030-2005)
Imagewriter Technical Reference Man. (17739-0) *
Inside Macintosh Volume 1 (17731) *
Inside Macintosh Volume 2 (17732) *
Inside Macintosh Volume 3 (17733) *

Inside Macintosh Volumes 1-3 Hardback (17737) *
Inside Macintosh Volume 4 (05409-4) *
Instant Pascal Reference Manual (17740) *
Language Card Manual (030-0217)
Laserwriter Owners Manual (030-1094)

Laserwriter Plus Manual (030-1292)
Lisa Cluster Controller Man. 1 (030-0611)
Lisa Cluster Controller Man. 2 (030-0667)
Lisa Motorola 16-bit Micro Manual (029-0055)
Mac 512K Enhanced Owners Manual (030-1326)

Mac 512K Disk Upgrade Manual (030-1327)
Macintosh Manual (030-0687)
Macintosh Plus Internal Drive Manual (030-1245)
Macintosh Plus Manual (030-1246)
Multilingual 9" Monitor Manual (030-0943)

Parallel Card Manual (030-0371)
ProDOS Technical Reference Manual (17728-5) *
ProFile Owners Manual (030-0993)
SCSI Card Manual (030-3118)
SOS Driver Manual (030-0143)

SOS Reference Manual Vol 1 (030-0441)
SOS Reference Manual Vol 2 (030-0442)
Unidisk 3.5 Manual (030-1151)
Unidisk Owners Manual (030-0416)
Universal Parallel Card Manual (030-0255)

80 Column Text Card Manual (030-0408)

Fig. 2B - Apple books, continued.

Countless early Apple drive cards have been destroyed by plugging in a drive connector offset by one pin or by an entire pin row. Do make sure your adaptor prevents this.

Unfortunately, that DB-19 connector has been a bear to find. One good source is *JDR*. These are usually available in solder tail only, so a short cable may be needed between your pc card and the DB19.

What Apple Books and Manuals are Available?

As I promised last month, figure two is a listing of most of the current non-IIgs *Apple* and *Macintosh* books and manuals. Some of these are stocked by your Apple dealer, and others are distributed through *Addison-Wesley*. The *Addison-Wesley* titles are shown with asterisks.

Just about all of these should be directly available from the *APDA* people as well, who are a first rate source for all Apple technical information.

What is New in Mass Teleportation?

The rate at which science and technological fact outpaces science fiction does continue to utterly astound me. Nowhere is this more apparent than in the emerging field of mass teleportaion.

The exciting center of all that has recently been happening is the outstanding *International Journal of Teleportation and Mass Transfer*. In particular, do check out Barfoot and Gentry's tutorial material way back in Volume XVIII, pages 1146-1198, along with their outstanding bibliography. And, for lots of hands-on construction details, check out Chediski, Colcord, and Elden's medium budget project on pages 1245-1277 of the same issue.

It sure is refreshing to see a scholarly journal that always remains simple, practical, and yet easily understood by lay people. It looks like the technology to watch is the *quark-muon* dissociation and regeneration process.

So, I guess I was not too overly suprised when Marcia Swampfelder shipped me her latest peripheral cards for the Apple II Plus, her MTT-T1 mass transferrence transmitter and her MTT-R1 mass transferrence receiver. Marcia is a tad on the conservative side, so she insisted on using illegal monitor entry points that precludes the use of these cards on newer Apples.

The pricing is rather attractive at $68.50 for the MTT-T1 and a mere $43.50 for the seperate MTT-R1. You can order direct from Marcia.

Anyway, you first plug the transmitter card into one Apple II Plus and as many as four receiver cards into four receiving Apple II Plus computers. Any object placed in the transmitter's dissociation chamber then will appear reconstructed in the receiver's regeneration chamber.

The effective range does depend on the telephone line in use, but for your average quality voice grade line, you can teleport objects up to 500 miles with a single receiver, 200 miles with two receivers, 100 with three, and 50 miles with four receivers. The little understood methodology of *conjugate phase decongruence* does prevent you from reliably using more than four receivers, regardless of the distance.

The Apple power supply and baud rate considerations both limit the size of the teleportation chambers. Those chambers on the MTT-T1 and MTT-R1 are slightly larger than a quarter. In the usual demo of these cards, you place a quarter in the chamber on the MTT-T1 card, and it reappears intact approximately 12 minutes later over on the MTT-R1 card.

This patch is for ProDOS Applewriter 2.0 version AWD.SYS only. AWD.SYS initially ran only on the 80 column IIc or the 128K IIe.

The patch improves the prompting tones when in the fast mode on a IIgs.

It works by lowering the pitch and lengthening the duration of each tone in the pair.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.1, using the filer utilities. Plainly label this disk FOR IIGS ONLY!**
2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**
3. **BLOAD AWD.SYS, A$2000, TSYS, D2**
4. **Verify 2260- 80**
   Change 2260: D0
5. **Verify 2265- A0**
   Change 2265: FF
6. **Verify 2267- 80**
   Change 2267: FF
7. **BSAVE AWD.SYS, A$2000, TSYS.**

Fig. 3 - ProDOS Applewriter 2.0 tweedleifier patch.

For a real "Golly Gee Mr. Science" demo, you can use four regeneration cards from each dissociation card. The single quarter you placed in the dissociation chamber will simultaneously reappear in all four receiving cards, again in a twelve minute dissociation-regeneration time interval. Put another way, the quad demo returns a dollar in change for every quarter that is invested.

Marcia reports that all of her current production is going to the importers of specialty herbs and spices. Her new teleportion system eliminates all of those long delays at customs, besides allowing users to set their own international currency exchange rates.

How About Some More Applewriter Patches?

Sure thing. This month, we will just do a cosmetic pair of patches for both ProDOS *Applewriter* 2.0 and 2.1.

When you run *Applewriter* in the fast mode on the IIgs, the tweedle gets ricidulously high in pitch and very garish. The *Tweedleifier* patches of figures three and four reduce the pitch down to tones that sound acceptable both in the slow and fast modes.

This patch is recommended *only* for IIgs use.

Show me a New Postscript Utility.

In continuing our once-a-month series of the *Postscript* utilities for the *Laserwriter* and similar printers, figure five is a listing of the single most often asked for *Postscript* code routine.

This listing lets you return the exact path of any imaged character in any font to the host. It works with all fonts from all sources, including all internal fonts, all externally downloadable fonts, and all custom fonts of your own.

The returned path is in its "open" or "trapezodial" form, and is easily adapted.

This patch is for ProDOS Applewriter 2.1 version AWD.SYS only. AWD.SYS initially ran only on the 80 column IIc or the 128K IIe.

The patch improves the prompting tones when in the fast mode on a IIgs.

It works by lowering the pitch and lengthening the duration of each tone in the pair.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.1, using the filer utilities. Plainly label this disk FOR IIGS ONLY!**
2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**
3. **BLOAD AWD.SYS, A$2000, E$6020, T$0C, D2**
4. **Verify 2260- 80**
   Change 2260: D0
5. **Verify 2265- A0**
   Change 2265: FF
6. **Verify 2267- 80**
   Change 2267: FF
7. **BSAVE AWD.SYS, A$2000, E$6020, T$0C.**

Fig. 4 - ProDOS Applewriter 2.1 tweedleifier patch.

```
For Laserwriter version 38 or earlier
% .....

initgraphics 100 100 moveto

/Palatino-Bold findfont [100 0 0 100 0 0]
makefont setfont

(S) false charpath clip

newpath 90 90 moveto 0 250 rlineto 250 0 rlineto
0 -250 rlineto closepath clip clippath

save /snap exch def fill snap restore clear

/fix {] == flush mark 10 {37 sin pop} repeat} def

mark
{{moveto} fix}
{{lineto} fix}
{{curveto} fix}
{{closepath}fix (
) print flush} pathforall

pstack flush showpage
```

Fig. 5 - A Postscript fontpath utility.

Don Lancaster's

ASK THE GURU

May, 1987

IIgs monitor options
Postscript point ruler
Die cutting with a laser
Applewriter promptifier
Improved IIc absolute reset

The horror story for this month involves several readers who are having their hard disk subdirectories trashed when they were using the *Copy II Plus* utilities.

Apparently this is rather well plowed ground. There definitely have been major defects in earlier versions of this code, which, supposedly, have been fixed in current releases.

Central Point Software, who are the suppliers of *Copy II+*, does have a quite low cost and a very aggressive upgrade program. If you are using hard disks with this software, be certain you do have the latest version, and call them with any futher problems.

Adobe Systems, the *Post*-script people, have a new and free set of *Postscript Developer Guidelines*. These are in the "must have" category. There's also a pair of $10 companion disks for the Mac and IBM that include a great error trapper and several other useful and all ready to run utilities.

Except, there is one glaring stupidity in these guidelines. Adobe does want Postscript to appear to be fast running, so they are insisting you do as much processing in your host as possible. Besides being just plain wrong, this concept (1) slows you down, (2) is elitist, (3) favors the developers of the expensive applications programs over users, and, worst of all (4) severely limits which computers you can use.

No, you *always* want to do as much of your Postscript work *inside* your laser printer as you possibly can. All that you should require of your host computer is your favorite and unmodified word processing or editor program. And nothing more. We'll note in passing that the totally absurd concept of host-resident downloaded width tables went out with the introduction of the *Diablo HYTYPE-1* twelve years ago, and what makes *all* of the non-Postscript laser printing ripoffs so bad is that totally unreasonable overload they throw onto their hosts just to save some RAM.

Can't you just picture the third assistant cook for a Grosse Pointe, MI housewife going down into the Detroit getto, knocking on a door, and asking "Madame is a bit short this week. Could she borrow a cup of sugar?"

Is this any different from a super fast 2.5 megabyte 68000 Postscript computer knocking on the I/O port of, say, a *Commodore* 64 and saying "Uh, I'm feeling a little elitist today, would you mind justifying a few characters for me?" Dumb.

Adobe's reasoning is that externally compiled code is much faster than internally interpreted Postscript. Sadly,

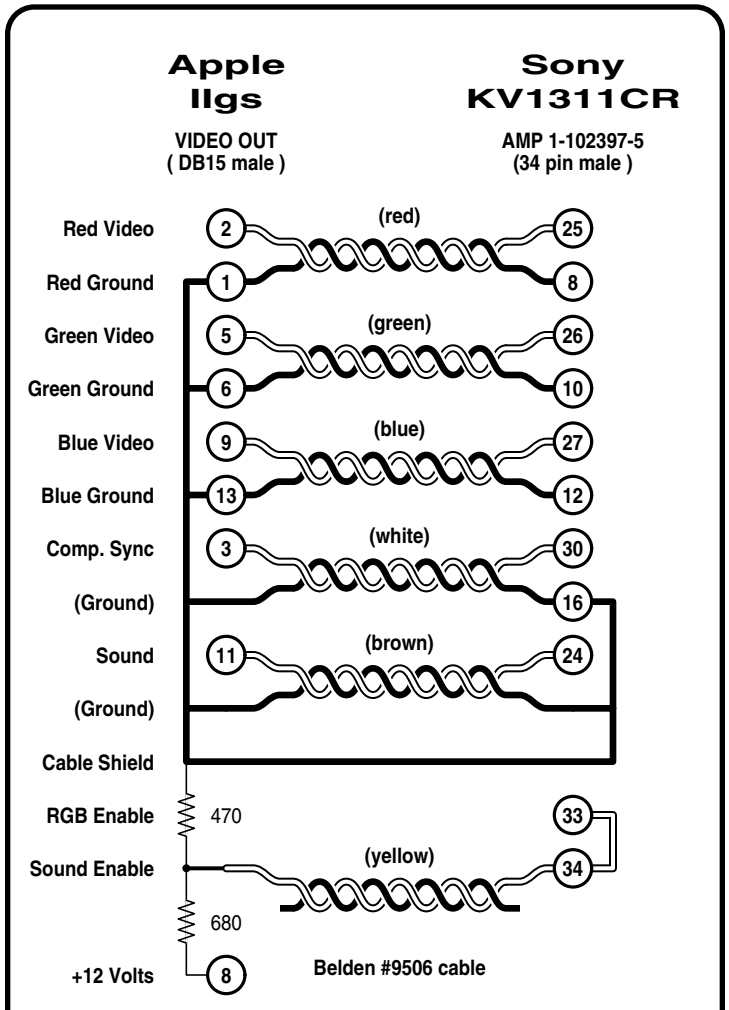


Fig. 1 - Apple IIgs to Sony KV1311 monitor interface.

the compiled code can get enough longer that any savings in the internal processing time is more than gobbled up by the much longer file transmission times.

Take *Meowwrrr*, our puddy cat, ferinstance. He is full of horrible trig calculations and irregular clipping intervals, so he would appear to be a prime candidate for compiling. Since compiling Postscript code is fairly trivial, I tried it. Sure enough, the execution time got shorter. On the other hand, the transmission time got enough longer that the final printing actually took *longer* to do.

And, no, Appletalk is NOT any faster than 9600 baud. In fact, it is considerably slower and creates its own problems. Full details on this cage rattler right here next month.

Besides, who says the host is going to be using compiled code? Asking TI BASIC on, say, a 99/4 to calculate all of your justification widths for you, will guarantee cobwebs on all of your Laserwriter output.

The bottom line: At least for low-end applications, you will *always* get the fastest Postscript code operation by *always* having your Postscript printer do *all* of its required tasks internally. Use nothing but a word processor or plain old editor on your host.

I found a great new source of toner and wiper pads for cartridge refilling. This is Alan Shepard and his new *Lazer Products* outfit. Alan does manufacturer all his own wiper pads, so there is no supply problem. And, no, he doesn't mind your using his pads with toner from other sources.

Alan even has custom "lite" pads available with less than the usual amount of silicon oil impregnant. These are handy to up the quality on your *Kroy Kolor* output, but should not be used over large numbers of copies.

As you know from last month, there's this hot new independent national *Postscript* bulletin board you can access at (409) 244-4704. There is no access charge, except for your own ads. Alan Kalka is the sysop for this exciting new resource. Only he is having trouble coming up with a good name for the board.

For this month's contest, just name our *Postscript* BBS. As usual, an *Incredible Secret Money Machine* to the best ten entries, and an all expense paid (FOB, Thatcher, AZ) tinaja quest for two to the overall winner. Alan will also throw in some small prizes of his own.

By the way, most of the *Ask The Guru* Postscript routines will be available on this board, ready for your instant downloading. There are two levels to the board. To get at the real goodies, you have to put one Postscript routine on for every ten you take off.

We'll be continuing our new *Don Lancaster's Laserwriter Corner* sidebar this month. Be sure to look for it near the end of this column.

Oops, I almost forgot to throw an advetorial at you. Heaven forbid. My *Postscript Show and Tell* is now available for use on virtually any word processor or comm program on the Apple II+, IIc, IIe, IIgs; what's their name's PC, AT, XT, and clones; the Atart ST; the Mac, Mac Plus; and even under UNIX.

I also have a brand new *Postscript Technical Illustrations* package for Apple, Mac, and IBM. This includes some outstanding schematic, isometric, pictorial, curve tracing, and layout examples. All are unlocked and easily adaptable to your own needs.

My *Introduction to Postscript* VHS user group video tape is now shipping, as is my self-published *Ask the Guru* reprints that cover everything worthwhile clear back to day one, edited, indexed, and fully updated.

Onward and upward . . .

What Monitors Work With the Apple IIgs?

Our helpline is overloading with people who want to use different RGB monitors on their IIgs. Which monitors will work with the IIgs, and how can they be connected?

Firstoff, note there are two

1. Place the existing IIc monitor chip into your EPROM burner and clone a copy of the monitor image into RAM.

2. Search the RAM image for an EXACT MATCH to the following bytes, which should begin at the start of your RAM image plus $3CCA:

```
A0  B0  64  3C  A2  BF  86  3D
A9  A0  91  3C  88  91  3C  CA
E0  01  D0  F2
```

3. Overwrite the exact match to the above RAM image with this sequence:

```
EE  F4  03  A0  12  20  A8  FC
2C  61  C0  10  07  88  D0  F5
4C  59  FF  3C
```

4. Burn a new 27128 (old IIc) or 27256 (new IIc) EPROM using the above patched code.

Fig. 2 - An improved absolute reset for the Apple IIc.

fundamentally different types of RGB montitors. These do include the *linear* type and the *digital*, or TTL type. A linear monitor can display all 4096 of the IIgs colors. A digital monitor can not, and is usually limited to a total of 8 or 16 colors.

Thus, even if you could adapt a TTL monitor to your IIgs, it would not show all of the intended colors in their intended way.

Further, it is not a simple matter to get attractive and readable 80 column text on a RGB monitor's screen. Many of the bargain priced monitors simply cannot handle IIgs text attractively. So, your first rule is to *never* buy a monitor for IIgs use unless you have seen *exactly* how sharp the text and the Super HIRES images will appear.

One monitor that has a lot of helpline interest is the new *Sony* #KV1311-CR. Besides having an acceptable picture and being several hundred dollars cheaper than the stock Apple monitor, this dude also has a composite video input for IIe use, a remote control, and even has a tuner so you can watch *Captain Video* or the *Roller Derby*.

Maybe even watch *Kukla, Fran, and Ollie*.

Figure one shows you an adaptor cable that will let you use this KV1311-CR with your IIgs. One source for these monitors is *47th Street Photo*.

There is one minor gotcha you do have to watch for. The Apple DB-15 monitor connector does not have a +5 volt output. Instead, a resistive divider has to be used to get from the +12 volt source to the inputs that activate RGB and sound as shown.

Many thanks to Courtney Jackson, who did most of the debugging and testing on this interface.

In addition, Tom Weishaar and his great *Open Apple* magazine pass on these part numbers of other television sets that include a direct linear RGB input . . .

J.C. Penny 2220
Magnavox RF4254WA
Magnavox RG4378BK
Quasar TT6290XE
Quasar TT6298YW
Sanyo AVM210
Sanyo 12C700
Sears 42701
Sony KV20XBR

Naturally, the interfacing details may vary with these other sets. If you use one of these, please send along your interface circuit so we can share it with other IIgs users.

And a question for all you IIgs people: Do you want to see a Lancaster cookbook on IIgs interfacing solutions?

What Causes Picture Noise in the IIgs Monitor?

A linear RGB monitor is a wideband analog electronic system, and thus, is highly sensitive to any noise source of any type. At the normal brightness levels in the usual room illumination, any picture noise in the stock Apple IIgs monitor will range from totally negligible to barely observable.

But, if you are a night person, and if you try turning the monitor's contrast and the brightness down to low late evening settings, you may find the picture noise and the snow to become objectionable. What you have done, of course, is reduce the signal to noise ratio in the process of reducing the brightness and contrast.

I have had a handfull of callers insist that this was a major "defect" in the Apple monitor that absolutely must be immediately "fixed". In reality, it is just about impossible to build a video monitor that is noise free over all of the possible brightness and contrast levels. What Sony did on the Apple monitor, of

This patch is for ProDOS Applewriter 2.0 version AWD.SYS only. AWD.SYS initially ran only on the 80 column IIc or the 128K IIe.

The patch unconditionally uses " \ " as the load to screen prompt. This allows the use of self-prompting glossaries.

It works by substituting " \ " for the underline token during a screen load.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.0, using the filer utilities.**
2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**
3. **BLOAD AWD.SYS, A$2000, E$6020, TSYS, D2**
4. **Verify 396E- CD DE B8**
   Change 396E: C9 5C EA
5. **Verify 396E-3970 per above.**
6. **UNLOCK AWD.SYS**
7. **BSAVE AWD.SYS, A$2000, E$6020, TSYS.**
8. **LOCK AWD.SYS**

Fig. 3 - ProDOS Applewriter 2.0 promptifier patch.

course, is to optimize for the minimum noise at the usual brightness and contrast levels.

There is a simple solution for this "night person" problem. Just run your monitor at the usual daytime contrast and brightness and put some *neutral density filter*, such as a piece of transparent gray plexiglass, between you and the screen.

Show me an Improved Apple IIc Absolute Reset.

I've had an *absolute reset* package available for some time that will let you unconditionally drop down into the monitor on a IIe, an enhanced IIe, and for those earlier IIc monitors (non-3.5). What this does is perform an EPROM monitor chip swap to do the equivalent of the CALL-151 from any point in any program, regardless of the copy protection in use.

The absolute reset remains invisible until called upon. A normal [control reset] resets in the normal way as intended by the program. And a normal [control open-apple reset] does the cold boot in the normal way, including all the "hole blasting" that destroys your code besides putting bulldozer tracks into the HIRES images.

But, should you keep your finger on the open apple key for four seconds *after* doing the [control open-apple reset], you will automatically get dropped down into the "old" monitor with your program intact and with no trashing of the HIRES screens.

Figure two shows you an improved way you can do an absolute reset on your IIc, which may be used both on the original and on the new IIc enhanced "3.5" monitors. Unlike the previous version, this patch is all in one piece and does not change the title of your bootup screen.

This new IIc method does require the use of an EPROM programmer that's capable of directly handing either of the 27128 or 27256 EPROMs.

How About Some More Applewriter Patches?

There is a "feature" in the ProDOS Applewriter 2.0 and 2.1 that you might want to do away with. For some reason, the "load to screen" trailer command was made the same as the underline token. Should you use no underline token, you loose the ability to load to screen for a preview.

Worse yet, since the load to screen command can change with time, many unique self-prompting glossaries will no longer work.

So, figures three and four show you the two *Promptifier* patches that will modify your ProDOS Applewriter 2.0 and 2.1 to *always* use the reverse slash " \ " as your load to screen prompting trailer.

Can I do Die Cutting On a Laser Printer?

The amazing answer to this question is that yes, you can easily do die cutting on a laser printer. The *Kroy Kolor* people now have a super new material that is more or less like the vinyl and carrier used by all the professional sign people, except for one major difference. This material is not cross linked properly until it is exposed to light and then chemically treated.

Here is how you use it: With your laser printer, print the die cut outlines onto a clear acetate sheet. Then you place the sheet in contact with the diecut material and next expose it to strong sunlight or one of the usual UV exposure boxes. The toner prevents the light from cross linking the material. When you next wipe the material with a developing chemical, every place that the toner has covered will dissolve, leaving you with perfect "die cut" letters or whatever ready to use.

Contact Randy Bailey over

This patch is for ProDOS Applewriter 2.1 version AWD.SYS only. AWD.SYS initially ran only on the 80 column IIc or the 128K IIe.

The patch unconditionally uses " \ " as the load to screen prompt. This allows the use of self-prompting glossaries.

It works by substituting " \ " for the underline token during a screen load.

1. **Make a third or higher backup copy of ProDOS Applewriter 2.1, using the filer utilities.**
2. **Get into /BASICS.SYS. Then CALL -151 to get into the monitor.**
3. **BLOAD AWD.SYS, A$2000, E$6020, T$0C, D2**
4. **Verify 3974- CD DE B8**
   Change 3974: C9 5C EA
5. **Verify 3974-76 per above.**
6. **UNLOCK AWD.SYS**
7. **BSAVE AWD.SYS, A$2000, E$6020, T$0C.**
8. **LOCK AWD.SYS**

Fig. 4 - ProDOS Applewriter 2.1 promptifier patch.

at *Kroy* for samples and more info on this new material.

Show me a New Postscript Utility.

For this month's Postscript utility, I'll show you how to make your own *point rule*, the single most needed and most asked for Postscript end user accessory.

While you can use any unit of measurement you like in Postscript, including inches, centimeters, pixels, or even in furlongs. But the default measurements are always made in printers *points*. In Postscript, there are 72 points per inch, compared to a "real" printers point, which is very slightly smaller.

Anyway, figure five shows you a ruler you can easily create. These are most useful for just about any layout task that involves Postscript. The actual ruler appears in figure six. You may want to cut this one out and laminate it in plastic, but printing as many copies of your own as you need is a better choice.

The ruler is printed diagonally on the sheet. You can then cut it out to the usual ruler shape. There are two main reasons for the diagonal layout. First, you can print a ruler 11 inches long this way, and still do so on a plain old piece of paper while using the stock Postscript letter frame device.

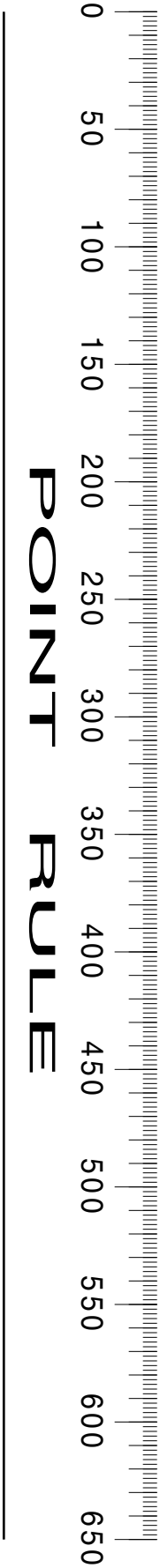
Second, most Postscript speaking laser printers normally will not have precisely the same scale in the X and Y directions. A scaling error of one to two percent is normal, and is at least partially caused by paper stretch.

A diagonal ruler, if printed on the same machine, will tend to "average out" all these scaling errors, and thus be just as accurate for both your horizontal and your vertical measurements.

```
-30 220 translate -125 rotate

% 2 tick
/xstrt -776 def /vstrt  200 def /tick1 -6 def /pos xstrt
def 436 {pos vstrt moveto 0 tick1 rlineto /pos pos 2
add def 0.20 setlinewidth stroke} repeat

% 10 tick
newpath /pos xstrt def /tick2 -12 def 88 {pos vstrt
moveto 0 tick2 rlineto /pos pos 10 add def 0.20
setlinewidth stroke} repeat

% 50 tick
/pos xstrt def /tick3 -18 def 18 {pos vstrt moveto
0 tick3 rlineto /pos pos 50 add def 0.20
setlinewidth stroke} repeat

% point numbers
newpath /Helvetica findfont [ 12 0 0 13 0 0 ]
makefont setfont /vstrt vstrt 32 sub def /xstrt xstrt
12 sub def xstrt vstrt moveto -688 168 moveto 1.66 0

(100    150    200    250    300    350    400    450
   500    550    600    650    700    750    800    850)
ashow

-734 168 moveto 1.66 0 (50) ashow -780 168 moveto
(0) show -465 146 moveto /Helvetica findfont [40 0 0
15 0 0 ] makefont setfont (POINT   RULE) show

newpath -776 135 moveto 870 0 rlineto 1 setlinewidth
stroke showpage
```

Fig. 5 - Postscript listing for a point rule.

Fig. 6



Don Lancaster's

ASK THE GURU

June, 1987

Postscript labelmaker
Speeding up postscript
IIgs disappearing drive
Appletalk -vs- 9600 baud
Laser printing economics

There's a new literature program now available directly from *Apple Computer* that you might not yet be familiar with. This one is called the *Apple Developer's Express*. It consists of a humongous quarterly binder that is sent free to dealers, clubs, and developers. The ambassador for your local user group has a copy.

And, of course, to find your local user group, you call (800) 538-9696, and then ask for extension 500.

At any rate, each package consists of bunches of data sheets, literature, and product announcements. There are usually a few demo disks, some magazines, and several other oddments.

This service is definitely worth looking into. If you do want to participate in this for your own products, the cost is around $1000 or so. Which can be a real media bargain.

Apple also just released two *public domain sampler* disks. These are so bad that they should immediately be destroyed on sight. Out of the 40,000 or so public domain programs now available today, these two disks are definitely in the bottom forty.

One disk boots up with *"ProDOS not Available"*. The other comes up with *"This is not a Macintosh disk, shall I initialize it?"* No menus at all, no internal operating system, and both totally user vicious.

The ProDOS diskette does have a copy of *Freewriter* on it. This is a stripped zero cost version of *Applewriter* that is an ideal first word processor for most junior high school students.

If you can, though, try and get a free copy of *Fredwriter* instead, through many of the California BBS systems. This is an updated and improved version of *Freewriter* that is much better documented and has been classroom proven.

Dave Blair of *Blair Designs* has solved the *Coburn* problem. And elegantly. As you may recall, the *Coburn* people have some absolutely outstanding graphics materials available, things such as gold foils, diffraction grating stuff, glow-in-the-dark sheets, and various other wonders.

Unfortunately, toner won't properly stick to any of their products.

So, what Dave suggests is this: Print a *negative* black toner image on plain paper. Then *Kroy Kolor* or else *Omnicrom* this sheet. Next, throw the sheet away and save the *Kroy Kolor* carrier.

Finally, *Kroy Kolor* the new *Coburn* material with the used carrier. It turns out that *Kroy Kolor* sticks totally all over the *Coburn* overlay film.

The results are unbelievably spectacular. For instance, if a used car salesman shows up and asks you for "something sedate" in a new custom business card, you can give him gold and pearl ink over a glow-in-the-dark diffraction grating background.

Dave will be most happy to send you several finished samples, provide you do help him along with his hobby of collecting federal reserve notes. He also does custom magnetic truck signs and business cards.

I have found at least one competitor to *Coburn*. Try the *Flormel* people as an alternate source for unusual foil and film materials.

Some blueish computer outfit in upstate New York has just gone *Postscript* for all their desktop publishing products.

Which once and for all guarantees that *Postscript* is now officially *the* standard page description language.

One side effect of this has been the 2:1 splitting and quadrupling in value of the Adobe stock to an all time record high. So, I'll reverse that hot stock tip I previously gave you: *Sell Adobe and take your 400 percent profit.*

After all, there's no need to get greedy.

Speaking of *Postscript*, do not forget we have this hot new BBS on line at (409) 244-4704. Most of the *Computer Shopper* listings on

Laser Printing "Click to Clunk" times (v.47.1)

. . . . . .

A. MEOWWRRR (Puss De Resistance)

Appletalk on a Mac	**90 seconds**
9600 Baud on a IIe	**70 seconds**
19200 Baud on a IIgs	**58 seconds**

B. IIGSBLK (IIgs Block Diagram)

Appletalk on a Mac	**45 seconds**
9600 Baud on a IIe	**30 seconds**
19200 baud on a IIgs	**22 seconds**

Fig. 1 - Appletalk is slower than 9600 baud!

1. **Defeat the test page on power up.**
2. **Let the machine run continuously.**
3. **Arrange your work so that similar jobs are processed together.**
4. **Use a fast communicating computer (such as a IIe) rather than a slow communicating one (such as a Macintosh.)**
5. **Avoid Appletalk like the plague.**
6. **Download persistent utility routines on first power up, rather that prefixing each file with the needed code.**
7. **Use the latest available ROM version for your laser printer or typesetter.**
8. **Do not use irregular clipping intervals, pixel line remapping, or elaborate charpath stuff except for "Uh-Compared to What?" applications.**
9. **Use the usertime command as a stopwatch to time out the critical portions of your code.**
10. **Run at the fastest possible baud rate, preferably 57200. Make sure your comm software's de-facto baud rate is near the intended baud rate.**
11. **Have postscript do as much internally as possible. Any host-based manipulations or calculations are almost always much slower.**
12. **Use hard disks, rather than floppies.**
13. **Use a 5 in 1 program that will create, edit, send code, receive error messages, and record error messages, preferably without any manditory disk access. ProDOS Applewriter 2.0 is nearly ideal.**
14. **Use your own custom Postscript code, rather than a canned applications program.**
15. **Aviod any and all attempts at putting anything on the video screen, except for Postscript listings.**
16. **Keep the serial files as short as possible. Use shorter variable names. Use rubber grid systems.**
17. **Leave such things as form letters, letterheads, label art in the PRINTER. Erase only the parts that change, such as the names and addresses.**
18. **Use a IIgs at 19200 baud or higher, rather than a Mac at anything.**
19. **Quickdraw does not, so do not.**
20. **Yellow single flashing means your speed is Postscript limited. Yellow double flashing in the absence of errors means your speed is communications limited.**

Fig. 2 - How to speed up your Postscript code.

Postscript are now downloadable from this new board. Please note that, while the board is free, advertisers and sponsors are urgently needed.

For this month's contest, just send me the best recipe for coconut-anchovy pizza. Or almost anything else that would be palatable only to a gonzo computer hacker that is well beyond the point of taking his *Dr. Pepper* intraveneously.

Other acceptable entries would be knockwurst ala king, butterscotch pudding *au jus* or road kill *de jour*. The absolute worst of these may someday appear in Marcia Swampfelders latest book. As usual, an all expense paid tinaja quest (FOB Thatcher, AZ) for two to the overall winner, and a copy of my *The Incredible Secret Money Machine* to the ten best entries.

I was hoping to get into binding systems this month, but there just isn't any room. So for your next homework assignment, write or call Ron Leonard over at *Lonards Distributing* for info on the *Unibind* system. Then call the people at *Velo-Bind* and get info on the *Personal Velo-Binder* and their *Cold-knife Velo-Binder* systems.

These are definitely the big winners to date.

Time for our usual – bleep – commercial. Bound sets of *Ask the Guru* reprints are available, as is the brand new *Introduction to Postscript VHS* videotape and the *Postscript Show and Tell* for the IIc/IIe/IIgs, Mac, PC, and ST.

And now, instead of raising the curtain, let's lower the stage instead . . .

Which is Faster – Appletalk or 9600 Baud?

Why, 9600 Baud of course. Why else would I ask such a stupid question?

In theory, *Appletalk* should be roughly 25 times faster than standard 9600 baud serial data transmission. In reality, the current software and use

protocols can hopelessly cripple *AppleTalk* to the point where 9600 baud is faster.

Figure one shows proof. I've taken two identical files from my *Postscript Show and Tell* and run one of them at 9600 baud on a IIe, and the other over *AppleTalk* on a Mac. As you can see, the IIe at 9600 baud is significantly faster than *AppleTalk* on a Mac. The IIgs is even faster.

To get your 9600 speedup though, it is very important that the de-facto baud rate is also remotely near 9600 baud. Some telecomm programs are written so cumbersomely, require so much disk access, and execute so slowly, that their *actual* transmission rate is ridiculously slower.

The worst offender in this sort of thing is *MacTerminal*. It is so slow as to be totally useless. The effective baud rate of *Applewriter* is around 7800 Baud in the 9600 mode, but this can be very greatly improved by using the IIgs.

We'll note in passing that the *Laserwriter* lets you run at 57600 baud, so I will eventually let you know what the ultimate transmission speed will be.

What does all this prove? Well, this conclusively proves that (1) An *Apple IIe* is much faster than a *Macintosh*, that (2) you should always beware of people bearing benchmarks, and most especially bench cats, and finally (3) Do avoid *AppleTalk* like the plague unless you are into snail geriatrics, glaciers, or congressional reform.

At least until *Apple* cleans up their act. While a speed advantage of 25:1 is possible, let us at least shoot for a breakeven before year end.

What Causes that IIgs "Disappearing Drive" Syndrome?

The IIgs *disappearing drive syndrome* goes like this: You start out with two 3-1/2 drives and everything works fine. You add a memory expansion card and configure it as a RAM disk, and the second real drive vanishes in a whiff of purple smoke. All you can now access are your first drive and the RAM card.

And, of course, much of your older IIc/IIe software is now useless since the second drive is nowhere to be found.

The problem is caused by the *order* in which the drives are selected by the IIgs. The first drive is usually Slot 5, Drive 1. The RAM card is often Slot 5, Drive 2. The second drive is usually Slot 2, Drive 1 when the RAM card is in use.

The solution ranges from trivial to impossible, depending on the protection scheme used on the original software. It turns out that ProDOS 1.1.1 or earlier operating systems can only recognize two drives per slot. All you have to do is upgrade the operating system on each disk to ProDOS 1.2 or ProDOS 8 or something that's newer.

If you are using unlocked software, just make a backup copy and replace ProDOS with ProDOS 8, renaming it as ProDOS. If your software disk is locked, find the CEO of the software house that sold you the product and then stake him to an anthill.

What Are the Economics Of Laser Printing?

Suprisingly good. I've just gotten into some heavy self-publishing, and the economics are absolutely amazing.

Let's start out with some numbers. First, there are no economics whatsoever to a non-*Postscript* speaking laser printer. Since only a thousand copies or so are printed out before the user finds out how bad they have been had, and since all these totally useless machines cost around $2000 each, you are talking a flat $2 per page operating cost.

That is *before* a new $3000 *Postscript* attachment lid is inevitably purchased and then installed.

The real *Postscript* printer costs are very sensitive to who is buying it how. There are three main costs involved here. These are the amortized cost of the machine, the paper cost and the toner cost.

For simplicity, we'll ignore the time value of money, and the actual production labor. In these days of low interest rates, the former does not amount to much, while the latter can often be your own children or a teenager from down the street.

If a doctor or dentist pays the *Laserwriter* full list price and buys a newer toy at 80,000 copies, his amortized capital cost is $5000/80,000 = 6.25 cents a page. If a more average buyer pays $3600 and gets 120,000 copies, his cost is 3 cents per page. If a gonzo buyer pays the low street of $3300, and maintains his own machine to get over 200,000

Fig. 3 - Auto-addressing Postscript shipping label.

copies out of it, his per copy cost is around 1.7 cents. Let's assume a three cent per page average amortized cost.

The toner costs will likewise vary all over the lot. The doctor or dentist will always pay list for a new cartridge. A $100 cartridge is good for 2500 copies, so his toner cost will be 4 cents per page. An average user will use three $25 refill kits per cartridge, so his toner cost will be 2 cents a page. Finally, the gonzo user will nurse each cartridge to seven refillings of his bulk purchased toner to get toner cost way down to 0.65 cents a page.

So, let's assume a one cent per page average toner cost.

Finally, even a doctor or dentist ought to be swift enough to buy paper in case lots from his local price club at $1.80 a ream, or 0.36 cents per page. Better quality paper from an intelligent supplier will be higher, so we might assume a one cent per page average paper cost.

Add these up, and a nickel per page turns out to be a reasonable operating cost of a *Laserwriter*, neglecting labor and the time value of money.

For a number of reasons, I decided to self-publish the *Ask the Guru* reprints. When I added up all the costs of the traditional jiffy printing route, I found that we are talking $7000, all of it up front, and none of it guaranteed.

What if we print each copy on the *Laserwriter*? We are now talking 100 double-sided pages, at six cents a page (you do use toner on *both* sides of the page), and throw in a buck for the cover.

Wonder of wonders, laser printing also costs $7000 per thousand, and is thus *about the same as* jiffy printing!

But whoa. How about the risk here? With laser printing, you never need print more than 20 books at a time. You are talking a short term $140 versus a long term $7000 risk. Even more important, if the book bombs, you may have to eat most of those thousand jiffy printed copies, and thus will lose your shirt. And what happens if your ultimate market is around 1326 copies or so? Nothing good.

There's some more subtle advantages to laser self publishing. Any additions and revisions can be made at any time. Review copies of the final bound book can be sent out ninety minutes after the author makes his final copy submission. Each customer can have his own name on the

```
% Requires your own mylabelproc for the fixed
% portions of the labels.

% Requires Synergetics pj justify routine. Free copy
% available on request. Version 38.0 needs plusbugfix.

/NewCenturySchlbk-Bold findfont [14 0 0 14 0 0]
makefont setfont /linewidth 500 def /filljustify false
def /yinc 12 def /indexcount 0 def /numlabelspersheet
4 def /vertlabeloffset 180 def /nameleftstart 350 def
/nametopstart 150 def /eraseheight 90 def /erasewidth
250 def /xlabelset 300 def /ylabelset 60 def

gsave xlabelset ylabelset translate numlabelspersheet
{mylabelproc 0 vertlabeloffset translate} repeat
grestore

/poscalc {/indexcount indexcount 1 add def
indexcount numlabelspersheet gt {/indexcount
1 def} if} def

/eraseoldname {gsave nameleftstart 20 sub
vertlabeloffset indexcount 1 sub mul nametopstart
add 15 add moveto erasewidth 0 rlineto 0 eraseheight
neg rlineto erasewidth neg 0 rlineto closepath 0.8
setgray fill grestore} def

/placenewname {/xpos nameleftstart def /ypos
nametopstart vertlabeloffset indexcount 1 sub mul
add def currentname pj indexcount numlabelspersheet
eq {copypage} if} def

/cleanuplastpage{ indexcount 1 add 1
numlabelspersheet {/indexcount exch def
eraseoldname} for showpage} def

/nameproc { save /snap exch def /namelist exch def
namelist {/currentname exch def poscalc eraseoldname
placenewname} forall cleanuplastpage clear snap
restore } def

% list format: [ (stuff) (stuff) (stuff) ] nameproc

[(Don Lancaster
SYNERGETICS
Box 809
Thatcher, AZ, 85552)

(Stan Veit, Editor
COMPUTER SHOPPER
407 South Washington Avenue
Titusville, FL 32761)]

nameproc
```

Fig. 4 - Shipping label Postscript listing.

```
a p shiplabel.w
 p
 psr1
 ppr   This module will address and print custom 4-up
 ppr   shipping labels. Labels after the first one in the
 ppr   batch are produced at maximum Laserwriter
 ppr   speed, approximately 32 per minute.
 ppr
 ppr   This super-fast operation results by keeping the
 ppr   label art inside the Laserwriter, and then changing
 ppr   only the address for each successive label.
 ppr
 ppr   Note: The label art must be in Postscript, but
 ppr   the address list is a standard text file.
 ppr
 psr2
 psx-1
 pgoa

b p
 psr1
 ppr For a demo, use DEMO.LABEL and DEMO.LIST.
 ppr
 pin Name of Postscript label art ----> =$B
 ppr

 p d2 unless demo
 pcs/$B/demo.label/
 pgof
 pcs/$B/DEMO.LABEL/
 pgof
 psx2
 pgoff
f psx1
ff oh,d(x)
 p
 p
 pin Is the list loaded (Y/N) ? ------> =$C
 pcs/$C/Y/
 pgoc
 pcs/$C/y/
 pgoc
 ppr
 pin Name of list to be loaded -------> =$C
 p
 p
 ny
 l$C
 pgoc
 pgoz
c psr2
 psx-1
 pgob
 p
 p
 psr1
 pin Remove hidden lines? (Y/N) ----> =$A
 psr2a
 psx-1
 pgod
 b
 f<>.=><><a
 y?
 p
d p
 psr1
 pin Remove product code? (Y/N) --> =$A
 psr2a
 psx-1
 pgoe
 b
 f<   =><><a
 y?
 p

e p
 psr1
 ppr formatting mailing list
 p
 b
 f
 f<>><)>>(x<a
 p
 p
 p
 e
 p D in next line is [D]!
 f<<] nameproc>>D>><
 y?
 p
 d
 f/(//
 y?
 p
 b
 u
 f//[ (/
 y?
 p

 p
 psr1
 ppr getting letter
 p
 b
 l$B
 pgog
 pgoz

g p
 p
 p
 oh,d1
 p
 p
 qcwide
 p
 ppr printing labels
 ppr
 ppr
 ppr  *** busy-please wait ***
 p
 pnp
 p
 pgox

z p
 pnd
 ppr[L]
 ppr Uh - There was a filename or disk access error.
 ppr Now returning you to the snoopervisor.
 pgoxxx

x p
 psr1
 ppr Computer is now free.
 ppr
xxx pin Press return -->
xx oh,d1
 p
 p  pdosnoop.v1.w
 p
 pqt

1 pnd
 ppr[L]
 ppr  Don Lancaster's Laserwriter Label Manager:
 ppr . . . . . .
 ppr
 prt

2 ppr
 pin OK to continue (Y/N/return) ----> =$A
 ppr
2a psx2
 pcs/$A/Y/
 pgoy
 pcs/$A/y/
 pgoy
 pcs/$A/N/
 prt
 pcs/$A/n/
 prt
 pgoxx
y psx1
 prt
```

Fig. 5 - Shipping label WPL supervisory controller.

title page in gold foil. You never need print more than you are about to reasonably sell in the next few weeks. And, ten years from now, you can still print and sell one single copy at a profit.

Wait. There's even more. Some very ugly things have been happening over in the traditional book publishing business that have been and continue to be a living hell on authors. Let's look at a few of the highly negative factors involved with traditional publishing today . . .

First and ugliest, the IRS now pays publishers to shred books, much the same way the feds pay farmers not to grow crops.

In fact, the IRS has caused more books to be shredded in the last few weeks than were destroyed in *all* of the dark ages. This means that back lists no longer exist, and that all existing copies of your book will summarily be shred- ded the instant it drops a notch or two on any of the best sellers list.

Second, those failing mom and pop book stores normally carried between 45,000 and 65,000 titles. The chains that are driving the older stores out of business only carry 4500 to 6500 titles, while stocking many more copies of each title. This causes a 10:1 reduction in available shelf space for your work.

Third are the infamous publishers committees, made up of total idiots that do not have the slightest idea who you are or why your material will sell. The committee will usually sit on your book for fourteen or more months, and then tell you the manuscript is rejected because they feel it is "not timely".

The marketing people are even worse. If lots of other books exist, then that market is "overcrowded". If no other books are competing with yours, then your market is "unproven". But if you print up a few copies on your own and sell them to demonstrate the demand really is there, you are guilty of the heneous and unpardonable crime of "skimming".

Fourth, mergers and buy-outs have reduced the number of publishers, so there are less places to submit books.

Fifth, the services that a traditional publisher can offer an author have been greatly diminished. You can now do your own arwork, all of your own layout, your own editing, your own typesetting, and even your own printing with quality that is more than competitive, and much more likely to give you exactly what you want. Before, you had no resonable choice but to use a publisher for these services.

Sixth, older publishers will fight you tooth and nail over *any* innovative and low cost (e.g. *Postscript*) way of doing things, for all those computer illeterate managers, pasteup and layout people are not about to admit they're ornery, obsolete, and obstructionist.

Finally, any book returns charged against an author's royalties were virtually unheard of several years ago. Today, an author is extremely lucky if half his royalties are not recalled at a later date.

On one side we have some unique new opportunities. On the other we have everything literally going to hell in a handbasket.

Kiddies, we are looking at a new total revolution in self-publishing. Let us start up some new networking on this. Your modem or my tinaja?

How can I Speed Up Postscript?

A lot of the criticisms about *Postscript* speed has nothing whatsoever to do with the *Postscript* language. Instead, user misinformation, system constraints, very bad software, and communications stupidities all gang up on poor old *Postscript*.

If you are creating your own *Postscript* code, figure two lists many of the things you can do to substantially speed up your printing times.

What is this month's Postscript Utility?

It is an automatic multiple label printer with quite a few unique advantages.

A sample label is shown in figure three, while the *Postscript* code appears in figure four. What this does is take any group of labels of any shape and size. It then puts a permanant picture of the *non-changing* parts of the label into the *Laserwriter*. A mailing list is then read, and the address or changeable part is put down for the total needed number of labels.

A mailing list is input in the form of an array of strings. One single string is used for each and every name and address group. The code is set up to process a hand fed strip of four labels at once. You keep feeding till all of the names get used up.

The label routine is easily adjusted for any number of labels of any size.

I get my blank labels from *Hy-Tech*, first because they are down the street, second because they have a large die selection, and third, because they are reasonably priced.

Figure five shows you a fairly user friendly new WPL pre-processor that works under ProDOS *Applewriter* 2.0 or 2.1. This starts with an address file consisting of name and address lines, each total address seperated by double carriage returns.

The addresses can be any number of lines.

As many hidden information lines as you want can be added to each address and coded any way you care to. Subject to the rule that the hidden code line must start with a period. One product code can be added to each printed address line, but subject to the rule that your product code have three or more spaces in front of it.

Don Lancaster's

ASK THE GURU

July, 1987

Apple resources
IIgs serial firmware
Applewriter fast boot
Desktop publishing tools
Postscript step and repeat

I am slowly but surely getting the IIgs to turn into a decent driver for the *Laserwriter*. In fact, if you use my patched version of *Applewriter* on the IIgs at 19200 baud, you'll end up with an effective baud rate around 18200. This rate is roughly *three times faster* than using a Macintosh (with or without *Appletalk*) as a *Laserwriter* driver.

The actual speedup you get depends on how much time is spent serial communicating and how much time is used running *Postscript*, but the speedup is always significant.

By the time you read this, the IIgs version 2.0 system software disks should now be available. Check your dealer. Included are new print drivers and various other mods and improvements.

Apple stock has 2:1 split and is now paying a small dividend. But it is the time to be getting yourself *out* of stocks, not into them.

There are plenty of laser printing developments for this month. Rumor has it that the *Riccoh* laser engine has some fatal or near fatal long term print quality flaws and is quietly being discontinued by several printer manufacturers. Meanwhile, *Canon* is moving some of its laser engine and its cartridge production work over to Tennesee.

Canon now has a new laser engine they call the SX in production. It is heavier duty and gives far blacker and much more consistent print quality. The new cartridge is also smaller and somewhat cheaper.

No, I do not have any refilling information on these as of yet. I think the toner is different, perhaps a finer grind or whatever. Chances are that same hole melting trick we used (CS Feb 87) for the big cartridges will work here. Please mail me an empty cartridge if you get a chance.

The new *HP* machines use this engine. Sadly, since *HP* refuses to speak *Postscript*, their new machines are utterly and totally useless.

There is hope, however, for older *Laserjet* owners. You can now get a new snap-on *Postscript* lid that at long last actually lets you do useful laser printing with a *Laserjet*. All at a combined cost that is a mere $1500 or so above the street price of having used a *Laserwriter* in the first place. The product is called the *PS-Jet*, One source is *Woody Baker* at *The Copier Store*.

I will say this about *HP* though: Their policy on laser replacement parts and manuals is light years ahead of *Apple*. In fact, just about all knowledgeable *Laserwriter* buyers run out and get a copy

Using the BASIC protocol in slot #1 . . .

To INIT, do a JSR $C100 or a CALL 49408.
To READ, do a JSR $C105 or a CALL 49413.
To WRITE, do a JSR $C107 or CALL 49415.

(Note that READ will hang the machine until such time as a valid character is input.)

Using the Pascal protocol in slot #1 . . .

To INIT, read the routine offset value in $C10D. Then set X = #$C1, Y = #$10, A = anything, and JSR to $C100 PLUS the offset value. On exit, X will hold the error code, with 0 = no errors.

To find the READ STATUS, read the offset value in $C110. Set X = #$C1, Y = #$10, and A = #$01 JSR to $C100 PLUS the offset value. On exit, a set carry means an input character is ready to be read.

To READ, read the routine offset value in $C10E. Then set X = #$C1, Y = #$10, A = anything, and JSR to $C100 PLUS the offset value. On exit, A will hold the read input value, while X holds the error code, with 0 = no errors.

To find the WRITE STATUS, read the offset value in $C110. Set X = #$C1, Y = #$10, and A = #$00 JSR to $C100 PLUS the offset value. On exit, a set carry means an output character can now be accepted.

To WRITE, read the routine offset value in $C10F. Then set X = #$C1, Y = #$10, A = low ASCII character to be written. JSR to $C100 PLUS the offset value. On exit, A is trashed, while X holds the error code, with 00 = no errors.

Fig. 1 - IIgs intermediate level serial I/O access.

of the *HP* service manual. That's part #02686-90904.

Which, of course, results in *Apple* paying to *HP* a $50 cash rebate for most every *Laserwriter* that's sold! More on this in a future column.

Meanwhile – and I'm not sure you are supposed to know this yet – *Apple* just might happen to be working on a new set of ROM chips for the *Laserwriter*. These chips run some 33 percent faster and have gotten rid of the frame device blowups, the bugs that had made *copypage* useless, and have finally sped up the *stringwidth* command to the point where it is now just plain slow, instead of being an outright joke.

On the other hand, the font protection has been tightened up. This is so stupid and so shortsighted as to be beneath comment.

The version number may or may not be 47.1, and earlier machines probably should be upgradable by one means or another. Logically, *Apple* should also upgrade to the new SX engine, and should throw in more fonts and a double sized bitmap so the next sheet can be processing while the current one is busy printing. I have no idea when or whether they will do so.

Be sure to get a free copy of the April 1st issue of *Outside Apple*. Among other gems, they advise you that any strange hissing and low frequency thumping noises in a IIgs on turn-on can often be cured by carefully opening the lid and letting the cat out.

A quick reminder about our a hot new independent *Postscript* BBS at (409) 244-4704.

Hundreds of new *Postscript* tips and routines are now downloadable, including most of the stuff you read here.

While the board is free, sponsors and advertisers are most definitely welcome and needed.

And the usual word from our sponsor: Bound sets of *Ask the Guru* reprints remain available, as do my *IIe/IIc Absolute Resets*, *Incredible Secret Money Machines*, *Postscript Show and Tells* for all major computers, and an *Introduction to Postscript* VHS videotape starring yours truely. Write or call for info on these and some great new freebies.

And now, let's throw a bone or two at all you newcomers to this column . . .

1. Initialize a new 3.5 disk on the IIgs using the IIgs system 2.0 utilities. Do NOT use a third party product or an earlier INIT routine!

2. Copy AW.SYSTEM as the FIRST system program on this disk. Use either Applewriter 2.0 or 2.1.

3. Copy a patched version of AWD.SYS, SYS.PRT, and TAB.PRT. You can also copy your favorite WPL STARTUP program and glossaries. Optional HELP routines may also be copied at this time.

4. Copy the program P8 from your IIgs system master disk 2.0. Then RENAME the P8 file as PRODOS.

Your disk will rapidly boot as ordinary ProDOS. To access the control panel from within Applewriter, do an {open-apple}{ctrl}{esc}.

Note that you have a choice of using unmodified AWD.SYS with a Super Serial Card or of making the ongoing patches shown here in CS.

Fig. 2 - Applewriter IIgs control panel and fast boot.

What Apple Resources Are Available?

There are many good and useful sources for information on your Apple computers, if you just know where to look or who to ask.

A local club or user group is the best place to get useful contacts and hands-on local help. You can find the user groups in your area by using the listings right here in *Computer Shopper* or else by calling (800) 538-9696, and asking for extension 500.

An outstanding source of technical information on all the Apple products is the *A.P.D.A* people. They stock just about everything on disk or paper that's available for just about everything Apple. They should be your first and foremost resource.

Besides *Computer Shopper*, far and away the greatest two Apple magazines are Tom Weishaar's *Open-Apple* and Bob Sander-Cedarlof's *Apple Assembly Line*.

The best Apple books are often published by *Addison-Weseley*. While carried by A.P.D.A., many of these are stocked in larger bookstores.

My own Apple books do include *Enhancing your Apple II and IIe*, volumes I and II, the *Apple Assembly Cookbook*, the *Micro Cookbooks*, both volumes I and II, my *Applewriter Cookbook*, and my *Ask the Guru* reprints. I do stock autographed copies of these as a reader service.

Apple publishes a series of technical notes, one set for the IIe/IIc/IIgs, and another set for the Macintosh. While anyone can subscribe to these, free copies are usually downloadable from a BBS or else can be borrowed from your local user group.

Some other *Apple Computer* corporate resources do include their *Developer's Express* and *Outside Apple*.

Electronic bulletin boards are an excellent source of info and technical help. A good listing appears, right here in

Computer Shopper. Since board listings go out of date very quickly, the trick is to get on to *any* local board. Chances are they will have a complete and current listing of all the other boards in your neighborhood.

Public domain software and shareware is another resource that should not be ignored. We will have some more on this in a future column.

Compuserve is a very useful BBS, but they do charge for their many services. Two outstanding resources here are Apple's very own *AppleLink* network and the *MAUG* Apple user group.

Your nearby community college can often be a very good source of Apple information and study courses. Check out both the data processing and the electronic technology departments.

Last, and by all means least, there is always your local Apple dealer.

How do the IIgs Serial Ports Work?

That serial communications firmware inside your Apple IIgs trys to emulate and improve on the features of the *Super Serial Cards* used in the older II+ and IIe.

Unfortunately, a new 8535 comm chip is now used instead of the 6551 that was used on the SSC. Any earlier software that made direct access to the 6551 for maximum speed (e.g. *most* of the serious and useful Apple software) will hang up a IIgs unless patches are made.

So, the first way to use the IIgs serial software is to ignore it and use a Super Serial Card instead. This will be needed for the vast majority of unmodified earlier Apple commercial software that requires serial inputs or outputs.

There are several different levels at which you can use your IIgs serial firmware. At the lowest level, you can make direct peeks and pokes to the 8535. While this is ridiculously faster than the other methods, *Apple* strongly does NOT want you doing this. Yet, because the other methods are so much slower, you may be left with no choice in the matter. Note that you usually lose all control panel access this way.

Some intermediate level access is done by making jumps to the Cn00 address space. As figure one shows us, you can do this in two different ways. Apple calls one the *BASIC access protocol* and the other the *Pascal access protocol*.

These names have nothing to do with what language you are using; in fact both protocols are most often accessed from machine language.

The BASIC protocol is very simple. You just call or JSR to a routine in the I/O space, and you are home free. But, only three I/O routines are available, namely *init*, *read*, and *write*.

There are two major disadvantages to the BASIC protocol. The first is that you lose control on *read* until a character is in fact input. There is no way to check to see if a character is available before you go and grab it. Secondly, there are some fancier *extended access* tricks that are not available.

The *Pascal* protocol cures these problems at somewhat more complexity. What you do is go to a standard entry point to find the *offset* to reach the real starting point. Then you call or JSR to the real entry point. Four routines are avilable, including *init*, *read*, *write*, and *status*.

With *status*, you might inquire ahead of time whether a character can be written or read. Thus, you can stay in complete control at all times.

The highest level is done by using the tools in the toolbox. These offer such sophisticated capabilities as buffering and background printing, and a standard access from high level code. On the other hand, that toolbox is rather slow. I often wonder if the main purpose of the toolbox isn't really to prevent the IIgs from outperforming the Macintosh.

More information on the serial toolbox routines does appear in the Print Manager stuff in chapter fourteen of volume II of *The Apple IIgs* Toolbox Reference. APDA stocks it.

Any More Applewriter Patches for the IIgs?

I had hoped to have solved the big Applewriter modem problem for the IIgs by now, but I'm not quite there yet. Hopefully, next month. If you must have modem access right now, just temporarily keep using your *Super Serial Card*.

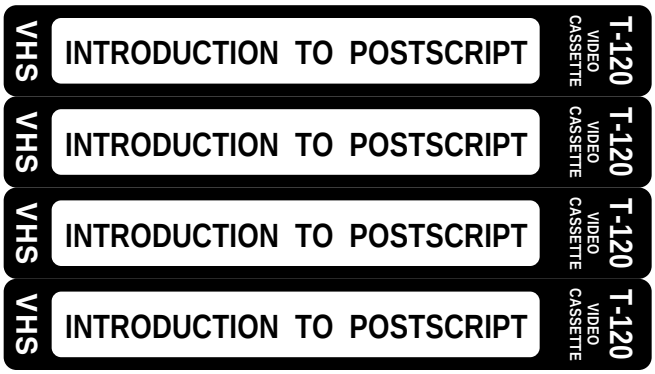


Fig. 3 - Typical step-and-repeat videotape labels.

```
/stepnrptparams 40 dict def
stepnrptparams begin

/admitonetick
  [6 5  9 150 60 25 25 true 10 true true] def

/babybumper
  [1 2 10 270 72 40 30 false 20 true false] def

/badgeaminit
  [1 2 3 220 220 90 60 false 250 true false] def

/bigbumpstick
  [1 1 3 792 205 0 0 true 40 true false] def

/businesscard
  [1 3 4 256 143 12 20 true 20 true false] def

/decaapus
  [1 2 5 306 158 0 0 false 50 true false] def

/hexsplit
  [1 2 3 306 264 0 0 false 50 true false] def

/lilbumpstick
  [1 1 5 610 150 0 20 false 60 true false] def

/octopus
  [1 2 4 306 198 0 0 false 50 true false] def

/quadsplit
  [1 2 2 396 306 0 0 true 50 true false] def

/seqbuscard
  [1 3 4 256 143 12  20 true 20 true true] def

/shiplabel
  [1 1 4 290 180 314 75 false 0 false false] def

/stdplabel
  [1 1 11 254 74 320 5 false 20 true  false] def

/readerservice
  [1 12 25 25 -15 120 450 false 0 false true] def

/vhsvideospline
  [1 1 13 424 56 80 35 false 0 false false] def

/3.5disklabel
  [1 2 3 216 226 100 60 false 20 false false] def

/5.25disklabel
  [1 1 7 316 110 275 35 false 0 false false] def

end
```

```
/setrepeatparams {cvn stepnrptparams
exch get aload pop /seqnumber exch
def /ticktrue exch def /ticklen exch
def /portrait exch def /vertstart
exch def /horstart exch def /incvert
exch def /inchoriz exch def /numvert
exch def /numhoriz exch def
/numpages exch def portrait {-90
rotate -792 0 translate } if} def

/onetick { 0 ticklen 2 div rmoveto 0
ticklen neg rlineto ticklen 2 div neg
dup neg rmoveto ticklen 0 rlineto 0
setlinewidth stroke} def

/drawticks {gsave ticktrue {0 0
moveto onetick inchoriz 0 moveto
onetick 0 incvert moveto onetick
inchoriz incvert moveto onetick}
if grestore}def

/stepandrepeat { setrepeatparams
numpages {gsave horstart vertstart
translate gsave numhoriz {gsave
numvert { drawticks save /rptsave1
exch def repeatproc  rptsave1
restore  seqnumber {/runningnumber
runningnumber 1 add def} if 0
incvert translate } repeat
grestore inchoriz 0 translate}
repeat grestore showpage grestore}
repeat } def

% . . . . .

% DEMO -- remove before use:

/Helvetica-BoldOblique findfont
[11 0 0 11 0 0] makefont setfont

/startingnumber 673 def
/runningnumber startingnumber def
/numstring 10 string def

/repeatproc {57 70 moveto
(This is business card #) show
runningnumber numstring cvs show
} def

(seqbuscard) stepandrepeat
```

Fig. 4 - Postscript step-and-repeat routines.

At any rate, figure two shows you how to create an *Applewriter* disk for your IIgs that boots very fast and lets you access the control panel from within *Applewriter*. To access the control panel, just do the usual {open-apple}{control}{escape}.

By the way, on the IIe, IIc, or IIgs, don't forget that you can simply and quickly get from BASICS.SYSTEM into *Applewriter* by first doing a PREFIX,D1, followed by a –AW.SYSTEM. This is a powerful booting trick.

A Finishing Materials Contest?

Changing any laser printed sheets of paper into final out-the-door products needs any of a number of finishing materials and techniques. As you probably have already found out, this will usually involve ridiculously costly machines made even more expensive through distributors and the final no-discount resellers.

Since laser printing is a new ball game, it stands to reason that we need new ways of cheaply and quickly doing tasks that the traditional printer machinery and accessory people have been grossly overcharging us for.

So far, I have two simple projects of my own. These are a $6 paper folder and a $9 replacement for the $500 *Vellobinder* binding machine. Let me know if you want any details on these.

I'd like to make a contest out of all of this. As usual, an *Incredible Secret money Machine* to the twenty or so best entries, and an all expense paid (FOB, Thatcher AZ) tinaja quest for two to the very best.

So, just show me all the cheapest and simpliest substitutes suitable for laser self printing that replace these products . . .

(1) A clamping paper cutter that is cheap.

(2) A $39.95 machine for *Kroy Kolor* fusion.

(3) A perfect binding system that allows many cover materials and uses glue only.

(4) A cheap jogger that can align sheets of paper.

(5) A corner rounder that is fairly priced.

(6) A way to die cut your own labels any way you like.

(7) An acurate way to fold brochures and such.

(8) Simple perforating and scoring systems.

(9) A handy padding press.

(10) A paper drill.

(11) A sanely priced saddle stapler or alternative.

(12) Any way to laser print envelopes that works.

(13) Cheap thermal binders.

(14) A system for varnishing or laminating.

(15) A semi-automatic way to collate pages.

(16) Ways of printing color T-shirts and fabrics.

(17) Packaging staplers that are easy to use.

(18) Greatly improved color systems.

(19) Some paper embossing methods, both plain and foil.

(20) Raised ink techniques such as thermography.

(21) Reasonable punching systems or alternatives.

(22) Ways to increase the durability of toner.

(23) Economical sources of specialty papers, particularly self-stick, transparent, and fluorescent.

(24) None of the above.

Well, you get the general idea. Next month, we'll look at several of the binding systems that are available today. Without no exceptions, all of them are grossly overpriced. What can you show me that is better?

What is this Month's Postscript Utility?

Another of my most asked for routines, of course. This one is an automatic *step and repeat*, used for most anything from business cards to bumperstickers to labels to tickets to whatever. For any time you want several images on a sheet of whatever that are either identical or in some manner related.

Sequential numbering is very easy to include for tags, tickets, and such.

Figure three shows you some sample output. In this example, the step and repeat is used for cassette videotape labels.

Figure four shows you the fundamental step and repeat *Postscript* routine. You can either use the predefined values in the stock dictionary, or else define your own for whatever step and repeat function you need.

There are several ground rules. Your image must be named *repeatproc* and must be well enough behaved that you can save and restore around it.

Should you have a very long routine, just split it up into individually called modules so as to prevent any stack overflows.

You also have to remember that you cannot print within 1/4 inch of either side or 1/8 inch of the top or bottom of your sheet. You can show or not show the cut marks by using a */tick true def* or else a */tick false* def command.

The particular *repeatproc* shown includes a sequential numbering routine. In fact, your output will be a dozen business cards with sequential numbers on them. The numbering is done by creating a numeric variable and then incrementing it for each repeat. The numeric variable is then converted into a string and printed as needed.

Variations on this theme will let you select a different routine for each position on the sheet. This is handy for, say, printing three different bumpersticker messages at once, or doing a three panel brochure.

Be sure to write or call for more info on all my neat *Postscript* goodies.

Don Lancaster's

ASK THE GURU

August, 1987

Curve tracing secrets
Binding systems survey
IIgs Applewriter modem
Toner cartridge reloading
IIgs cables and connectors

I didn't expect to be able to report this to you until a few weeks after the Ayatolla's Bar Mitzvah, but the *Hewlett-Packard* people have gone *Postscript*! Honest.

Actually, if you remove all the hype and all the wishful thinking from their announcement, all they really did was acknowledge the status quo.

To wit: You can now buy third party lids for the *H-P* printers that let them speak *Postscript*, at a price that's only a mere thousand bucks or so above getting a *Laserwriter* like you should have in the first place. And, at some future unspecified time and in some future unspecified manner, *H-P* will offer far more substantial *Postscript* support.

One source of the *Postcript* lids is the *PS Jet* by *QMS*.

All of which raises two big questions. The first is – What will happen to *DDL*, the page description languge that *H-P* was highly touting until very recently? Well, it turns out that *DDL* should remain as a specialty language of particular interest to vintners and oenophiles, because *DDL* is vastly superior to *Postscript* when it comes to the creation of images of sour grapes.

The other obvious question is – What will *Apple* do now? A major announcement is almost certainly imminent at this writing. But right now, the situation goes something like this: The new *QMS* laser printers now have *Postscript* firmware that is 33 percent faster. The new *H-P* machine uses a greatly improved and second generation engine that is blacker, now feeds heavier stock, handles most envelopes well, plus other advantages.

There is absolutely *no way* that anybody else's first generation engine will be able to hold a candle to the *Canon* SX engine, at least in the forseeable future. *Canon* is also moving their laser engine production to Tennessee. And *IBM*, as usual, is threatening to do all sorts of incredibly stupid things.

Apple has bunches of new goodies available. Check out their six tape video series that includes Alan Kay, Woz, and bunches more. The tapes do go for $10 each, or $50 for the set. They must be ordered through your local user group. Some free tapes on the IIgs and on the new Macs are also available, again through your user group.

There is also now a revised *Laserwriter Reference* that'll be mostly of interest to the Mac people. This jewel does include lots of details on the

This patch is for ProDOS Applewriter version AWD.SYS only. It restores modem communication on the IIgs via the internal IIgs printer port. It works by linking to custom port drivers for Init, Status, Read, and Write.

1. Make a backup copy of ProDOS Applewriter 2.1. Plainly label this disk FOR IIGS ONLY! and FOR INTERNAL IIgs PRINTER PORT ONLY!

2. Get into BASICS.SYSTEM. Then CALL -151.

3. BLOAD AWD.SYS, A$2000, T$0C

4. Verify: `4F35-  AD D4 B8 48 C9 03 B0 17`
Change: `4F35:  60 20 02`
`4F38:  4D A2 C1 A0 10 A9 00 20`
`4F40:  45 C1 60 A2 C1 A0 10 A9`
`4F48:  01 20 48 C1 60 A2 C1 A0`
`4F50:  10 20 46 C1 60 A9 11 80`
`4F58:  02 A9 13 A2 C1 A0 10 20`
`4F60:  47 C1 60 11 11 11 11 11`

5. Verify: `2D1C-  20 02 4D`
Change: `2D1C:  20 36 4F`

6. Verify: `2364-  BD 89 C0 29 08`
Change: `2364:  20 43 4F EA EA B0 2B`

7. Verify: `23A3-  BD 88 C0`
Change: `23A3:  20 4D 4F`

8. Verify: `2515-  BD 89 C0`
Change: `2515:  20 43 4F EA EA B0 01`

9. Verify: `251D-  BD 89 C0`
Change: `251D:  20 4D 4F`

10. UNLOCK AWD.SYS

11. BSAVE AWD.SYS, A$2000, E$6030, T$0C

12. LOCK AWD.SYS

Fig. 1 - Applewriter 2.1 serial patch for the IIgs.

magic *Postscript* dictionary that can interact with *Quickdraw*. One source is *APDA*.

Another thick publication binder is called the *A.D.S. Information Exchange*. This includes, wonder of wonders, the actual names and phone numbers of who does what to whom at Apple. For more information, do contact Carol Lockwood over at the *Apple Developer Services*.

And, our usual reminder to check out the new hot and independent *Postscript* BBS at (409) 244-4704. Most of my *Computer Shopper* programss that you see right here are downloadable from this great resource.

Far and away the best entry so far in our hacker recipe contest was for those *Hostess Twinkies, Szechwan style*. But since it would be redundant to award a tinaja quest to an Arizonian, I'll keep this contest open for another month. Otherwise, you can enter my *Hardware Hacker* contest that you'll find over in *Modern Electronics* – just dream up a new and off the wall use for one or two linear stepper motors that offer 25 pounds of force over an 8 inch stroke in .002 inch steps.

And hey, no fair writing a librarian of some nearby town to try and find out what a tinaja quest is. All that this resulted in was the librarian going on one and thoroughly enjoying herself.

She'll never tell.

I've got a brand new series of products shipping this month. These include my new *Postscript Technical Illustrations*, and are available for the *Apple*, *Mac*, *IBM*, and for the *Atari ST*.

Included are schematic and isometric drawings, pictorials, pixel line remapping, curve tracing, charts, graphs, rubber grids, layouts, plus lots more.

While only your preferred word processor or editor is needed, the final results are ridiculously higher in quality than what most people can get from most of the graphics programs most of the time. The unlocked and unprotected routines may be used as is or else dropped into most any application that accepts direct *Postscript* code.

Our biggie this month is my *Postscript* curve tracing code. But first . . .

Where can I get IIgs Cables and Connectors?

At long last, some cables and adaptors at sane prices are starting to show up for the IIgs. Besides all the grossly overpriced stuff at your local Apple dealer, I've found four interesting sources.

The first is *JDR*, and you can check out their ads over in *Computer Shopper*. These good people were the first to stock the DB-19 connectors.

A second source is *KGB Enterprises*, who should have old drive adaptors and other goodies available by the time you read this.

A third source is *Redmond Cable*. They have printer A-B

WARNING: Interim code for gonzo hackers only!

This patch is for ProDOS Applewriter version AWD.SYS only. It allows XON/XOFF two-way communication with the Laserwriter at an honest 19200 baud. This is three times faster than Appletalk!

It works by altering the scrolling routine and by stopping character input during scrolling. One present bug: A [Q]-I MUST be done at least once before doing any [O]-A catalog or [L]-? load.

1. **Make a backup copy of ProDOS Applewriter 2.1. Plainly label this disk FOR IIGS ONLY! and FOR INTERNAL IIgs PRINTER PORT ONLY! Install the serial patch of figure one to this disk.**

2. **Get into BASICS.SYSTEM. Then CALL -151.**

3. **BLOAD AWD.SYS, A$2000, T$0C**

4. **Verify:** `24B7-  C6 A6 A5 22 48 20 FA`
   Change:
   ```
   24B7:  EA
   24B8:  EA EA 20 07 25 20 59 4F
   24C0:  C6 A6 A5 22 48 20 FA 24
   24C8:  A5 28 85 26 A5 29 85 27
   24D0:  68 18 69 01 C5 23 B0 18
   24D8:  48 20 FA 24 20 07 25 8D
   24E0:  54 C0 20 00 61 20 07 25
   24E8:  8D 55 C0 20 00 61 80 D8
   24F0:  A0 00 20 7E 25 20 55 4F
   ```

5. **Verify:** `4C8A-  2A`
   Change: `4C8A:  60`

6. **Add to end of existing code:**
   ```
   6100:  18 FB C2 30 A6 28 A4 26
   6108:  A9 27 00 54 00 00 38 FB
   6110:  60
   ```

7. **UNLOCK AWD.SYS**

8. **BSAVE AWD.SYS, A$2000, E$61FF, T$0C**

9. **LOCK AWD.SYS**

Fig. 2 - Applewriter 2.1 fastalker patch for the IIgs.

Fig. 3 - Some uses for my curvetracing routine.

switch boxes that will work with the new IIgs mini-DIN 8 connectors, along with all the usual cables, gender changers, whatever.

And a final source that just showed up in today's mail is from *Micro Computer Cable*. Their prices are super low. As a ferinstance, the IIgs printer cable costs a mere $5.50.

Can I use the Applewriter Modem on the IIgs?

Question: Which is faster, a twelve year old, 8-bit microprocessor, or a brand new and state of the art 16-bit one using a high speed clock?

The answer, of course, is that the old 8-bitter is very significantly faster, because the programmers of any new chips always insist on using software and firmware that is hopelessly bloated beyond belief. As a general rule, any time you give a programmer a three times faster machine, they will write code that runs nine times *slower* than it did on the original machine.

Case in point. It takes an Apple II+ *six* microseconds to find out whether a character is available to be received on a serial port. But, it takes an Apple IIgs *one hundred and twenty* microseconds to do the same thing. That's in its fast mode, of course. If you want to count clock cycles, it is 6 cycles on the II+ versus 314 on the IIgs. Arrgh.

Which, of course, will give *Applewriter* fits when it is trying to run at 9600 baud, let alone at 19200.

At any rate, I now do have some interim patches available for *Applewriter* that does restore the modem operation and even lets you talk to your *Laserwriter* at a 19200 baud rate. Figure one shows you the patch needed to get any modem operation at all out of *ProDOS Applewriter 2.1*, and Figure two shows you the scrolling speedups needed to allow XON/XOFF communications at 19200 baud.

Uhm, the code in figure

two is not quite optimum yet, but I thought you would like an advance look at it. It does in fact let you receive and record any shorter 19200 baud error messages sent from your *Laserwriter*.

Be sure to use a copy from an original v2.1 disk. The simple printing patches I've shown you earlier will defeat this new code. Yes, the figure one code also lets you print properly.

One interesting result of the patched code: Applewriter on a IIgs at 19200 baud communicates your *Postscript* to the *Laserwriter* approximately *three times faster* than does *anything* on a Mac that is using the *Appletalk* network.

What's new in Toner Cartridge Reloading?

Well, firstoff, both *H-P* and *Canon* are spreading all sorts of horror stories on the many unspeakable evils of toner cartridge refilling. When they ever do catch you filling a cartridge, they will spank you and send you to bed without any supper.

It seems that there's this horrible side effect of doing the $5 to $15 toner cartridge refills. Would you believe that there are people out there that actually would stoop to paying $15 for a black cartridge, compared to spending over $100 for a grayish one?

The cartridges, of course, do not get up to their maximum blackness until *after* their second refill.

The best source I have found for wiper pads and for refill toner is through Arlin Shepard at *Lazer Products*.

You can also get a product known as a *wide mouth wash bottle* from any lab supply house, such as the B7893-1L from *American Scientific Products*. Pull off the inside riser tube, and cut the snout so it sticks up two inches or so from the cap. You then dump the toner into the bottle and *gently* squeeze the bottle into the cartridge. Presto. No funnel, and less than one-tenth the mess.

Other ways of making the holes in the cartridge include the *Unibit* plastic drill from *Vise Grip* or hot-melting with a .45 caliber cartridge.

To shatter several of the myths going around: A complete cartridge disassembly and rebuild is almost certainly *guaranteed* to cause lots more problems than it solves. And using a vacuum cleaner to empty the toner holding tank can short out the commutator on the motor and start a fire.

Not to mention that all the toner dust goes right through most vacuum cleaner bags like they weren't there.

A key reminder or two: Copier toner and laser printer toner are different and *must not* be mixed up. The toner used in those new laser SX engines is also different than the older CX stuff.

For more info on cartridge refilling, check out my new *An Introduction to PostScript* video tape.

Tell me all About Binding Systems

The big desktop publishing revolution has now created all sorts of needs to hold pieces of paper together.

Now, the binding systems manufacturers want a product

```
% copyright 1987 by Don Lancaster and Synergetics
% Box 809 Thatcher AZ, 85552  (602) 428-4073
% All commercial rights reserved. Personal, and
% non-commercial use permitted so long
% as this header remains present and intact.

/curvetrace {/curvelist exch def tension 0 eq
{/tension .000001 def} if curvelist length 3
div 1 sub cvi /#triads exch def /ptr 0 def
firstpoint morepoint} def

/tension 2.83 def      % default for best fit
/showtick false def   % don't show points
/ticklen 15 def          % length of ticks
/tickhead ticklen 4 div def

/prvx { curvelist ptr 3 sub get } def
/curx { curvelist ptr get } def
/prvy { curvelist ptr 2 sub get } def
/cury { curvelist ptr 1 add get } def
/prva { curvelist ptr 1 sub get } def
/cura { curvelist ptr 2 add get 180 sub} def

/showtic1 { showtick true eq {gsave currentpoint
newpath translate cura 180 add rotate ticklen
neg 2 div 0 moveto ticklen 0 rlineto tickhead
neg dup rlineto  tickhead dup rlineto tickhead
dup neg exch rlineto 0 setlinewidth stroke 0
ticklen neg 2 div moveto 0 ticklen rlineto stroke
grestore} if }def

/firstpoint { curx cury moveto showtic1 /ptr ptr
3 add def}def

/morepoint {#triads { curx prvx sub dup mul cury
prvy sub dup mul add sqrt tension div /zdist exch
def prva cos zdist mul prvx add prva sin zdist mul
prvy add cura cos zdist mul curx add cura sin zdist
mul cury add curx cury curveto showtic1 /ptr ptr 3
add def} repeat} def
```

Fig. 4 - My Postscript curvetrace routine.

```
% requires curvetrace routine of figure four

/showtick false def /tension 2.6 def
1 setlinecap 1 setlinejoin
/ct1 {curvetrace 1 setlinewidth 0 setgray stroke} def
/ct2 {curvetrace 2 setlinewidth 0 setgray stroke} def
/ct3 {curvetrace 3 setlinewidth 0 setgray stroke} def

% ears
[ 107 614 35  165 605 -45  162 544 -135  100 550 135
107 614 35] curvetrace fill [150 720 40  220 710 -60
200 655 -135  135 660 125  150 720 40] curvetrace fill

% face oval
[230 516 165  170 600 90  245 673 0  320 612 -80]
curvetrace gsave clip newpath [80 560 -30
 202 529 145  195 580 50  230 585 -25
 230 585 145  208 655 45  267 646 -45
 267 646 120  258 668 35  285 662 150
 310 770 80] curvetrace
closepath 0 setgray fill grestore 2 setlinewidth
stroke /tension 3.4 def

% left eye
[290 640 30  310 622 -70  310 598 -150  294 613 120
290 640 30] curvetrace gsave clip newpath
[295 627 20  306 612 -80  304 597 -150  290 610 120
295 627 20] curvetrace fill grestore 1 setlinewidth
stroke /tension 2.6 def

% main nose
[297 580 80  338 617 0  354 595 -90  290 533  180  229
564 120]  curvetrace gsave
closepath 1 setgray fill grestore 2 setlinewidth stroke
[221 558 60  240 567 0] ct2

% nose knob
[338 600 0  350 620 75  345 645 135  325 645 -135
325 615 -60 338 600 0] curvetrace fill /tension 3.2
def [330 640 45  341 641 -45  338 630 -150 330 630
135  330 640 45] curvetrace 1 setgray fill 0 setgray
/tension 2.6 def

% mouth
[229 564 -80  260 518 -10  290 532 35  290 532 180
229 564 125] curvetrace fill
[238 537 45  257 530 -60  257 530 45  275 522 -70
275 522 220  238 537 110] curvetrace gsave 0.9 setgray
fill grestore 2 setlinewidth  stroke [253 535 -40
262 524 -50] ct1

% right eye
/tension 3 def
[256 625 30  280 609 -60  281 579 -150  258 596 120 256
625 30] curvetrace gsave clip newpath
[260 610 20  273 597 -70  268 582 -160  257 592 110  260
610 20] curvetrace fill grestore 1 setlinewidth  stroke
/tension 2.6 def

% eyebrows
[229 636 100  240 648 -20] ct1
[273 656 65  287 656 -45] ct1

% appendage
[168 369 -120  130 354  -160  86 322 -135
86 322 45  131 351 20  172 367 55] curvetrace fill

% chin detail
[224 530 -80  260 507 0  302 532 60] ct2

% main bod
[230 516 -30  260 508 0  300 530 60  300 530 10  394 557
20 394 557 -100  398 528 -50  398 528 -160
330 508 -160  330 508 -60  350 465 -90  300 440 180  215
490 90 230 516 30] curvetrace fill

% left leg
[275 328 -50  300 320 30  300 320 70  324 380 70  290
370 -115 275 328 -110] curvetrace fill

% right leg
[210 319 -60  234 308 15  234 310 60  270 365 60  241
370 -125 210 320 -125] curvetrace fill

 %pants
[320 460 10  350 475 30
350 475 -70  355 440 -95  326 384 -150  326 384 30
344 401 65 344 401 -100  345 375 -60 345 372 -150
275 370 150] curvetrace closepath gsave 0.4 setgray
fill grestore 2 setlinewidth stroke
[316 379 -150  285 370 -180] ct2

%buttons
/tension 3 def
[ 342 462 -10  347 444 -100  337 431 170  332 447 80  342
462 -10] curvetrace gsave 1 setgray fill grestore
1.5 setlinewidth stroke
[319 445 -20  323 429 -110  312 420 160  306 434 70 319
440 -30] curvetrace gsave 1 setgray fill grestore
1.5 setlinewidth stroke /tension 2.6 def

% inside book
301 473 moveto 330 468 lineto 287 360 lineto gsave
270 365 lineto closepath 0.9 setgray
fill grestore 2 setlinewidth stroke

%book pages
[306 470 -55  322 464 0] curvetrace 282 362 lineto gsave
250 370 lineto closepath 1 setgray fill grestore stroke

%book
156 509 moveto 310 480 lineto 270 380 lineto 124 410
lineto closepath gsave 0.99 setgray fill grestore stroke

% pages
318 463 moveto 303 425 lineto 0.25 setlinewidth  stroke
315 463 moveto 309 448 lineto stroke
313 465 moveto 293 416 lineto stroke
311 466 moveto 306.5 455 lineto stroke

%right arm
[180 454 15  208 448 -30  208 448 70  240 498 40
230 516 100 230 516 -150 180 454 -90] curvetrace fill

% book spine
[124 406 -12  270 376 -12  270 376 -75  286 359 0  286
359 170 134 391 170  134 391 150  124 406 90]
curvetrace gsave 0.9 setgray
fill grestore 2 setlinewidth stroke

% right sleeve
[172 438 100  177 455 30  210 445 -50 210 432 -140]
curvetrace gsave
closepath 1 setgray fill grestore stroke

%left foot main
[317 319 10  388 280 -90  333 241 180  251 270 145 ] ct3
[244 318 15  260 320 5]ct3
[294 288 60  330 312 20] ct2

% right foot main
[246 304 -55  252 285 -90  174 236 180  120 280 90
192 320 -10] ct3
[172 308 -15  215 285 -50] ct2

% left cuff
[300 335 -10  316 322 -90  290 305 170] ct2
[282 306 170 261 326 90  281 340 -10] ct2
[273 331 -65  288 320 -15  308 324 35] ct1

%right cuff
[240 320 -40  246 305 -90  220 296 169] ct2
[211 298 157  194 323 70  220 331 -15] ct2
[209 328 -115  218 311 -20  240 312 60] ct1

%right hand
[159 377 220  138 400 90  160 440 40  180 445 0  205
436 -45 220 418 -10  252 415 -70  235 395 180
206 395 -135 197 377 -80] curvetrace gsave
1 setgray fill grestore  2 setlinewidth stroke
[155 400 -90  158 380 -80  178 369 45] ct2
[177 392 -100  190 363 0  203 376 80] ct2

% right glove detail
[179 444 -120  168 421 -110] ct1
[186 445 -115  177 420 -95] ct1
[192 443 -110  188 419 -90 ] ct1
```

more . . .

Fig. 5a - Postscript curve tracing study aide . . .

that demands an expensive machine and needs a continuous source of non-standard and expensive supplies, sold through a two-tier distributor and retailer network that is absolutely forbidden to ever discount anything.

On the other hand, the laser printing user just wants some cheap way to attractively hold pages together.

Thus, there is *no way* that a binding systems manufacturer will *ever* provide a cheap and simple method to attractively hold pages together, for it is not in their best interests to ever do so.

I have now sorted through dozens of different binding systems, and have concluded that we need something new and exclusively suited for new age, small quantity laser printing.

Something easily built by the end user. But, until that certain something shows up, I'd like to tell you about the dozens of systems that I have checked into, picking the best of the worst.

Let us start with the saddle staplers. Talk about sticker shock! At any rate, if you do scrounge around a junk store or a Hong Kong junk direct mail catalog, you'll find a gadget called a *Rorostapler* that sells for around $3 and has a built in swivel so the staples end up at right angles to the main arm. Thus, you staple in *from the top* or *from the bottom*, and do not need a long throw on the arm.

For a very complete catalog of overpriced and obsolete binding systems, check into the *GBC* catalog. Nobody in their right mind would continue to use a plastic comb binding in this day and age, since it is such a gross insult to your customers.

Another big comb binding outfit are the *HOP Industries* people. While you wouldn't want to buy comb binding stuff from anyone, *HOP* does have far and away the lowest price for unpunched 5 mil clear acetate stock. Cost is under a nickel per sheet in larger quantities.

By the way, the very best looking covers I have done so far are done in one of two ways. You can laser print onto heavy weight parchment and then use one of these clear acetate overlay sheets, or else you can laser print onto other heavy cover stock and then use *Kroy Kolor* and the *Kroy* lamination materials.

If you want any specialty cover materials, do check into *Catalina Plastics*, who, for a price, will custom cut all sorts of interesting stuff for you.

For a real joke, check out the *Scotch Binding System*. This one sticks to the *sides* of each page, shingle like, and is bizarre even for *3-M*. This is perfect bound and handles up to 25 pages. I guess I'm down on this system because (1) their demo fell apart in my hands, and (2) $120 is too much to spend on a machine that can be replaced with a short scrap of sewer pipe.

Winner #1 to date is the *Personal Velobinder* binding system. This method is not a perfect binding system in that two ingenious snap-in plastic strips clamp all the pages together. The maximum limit is around 25 pages. Some good looking, already prepunched covers are stocked. The cost is around $30-50 for a starter kit, and around a quarter for the strips and fifty cents for the covers.

Some tips: You *must* use a minimum of four hands while you are punching. Be absolutely certain the punch is clear of snitzels and be sure the pages are completely set home before you punch. You get only *one* chance to punch things correctly. Blow it, and your pages are lost.

Pages can be added or else removed at a later date. For a permanant document, just add a dab of PVC/ABS cement to "weld" the pins to the strip.

You'll also find a larger system called the *Cold Knife Velobinder*. This binder usess heavier strips with a ratchet system to lock the two strips together. The thickness is one inch maximum.

You can bypass the need for their $450 punching and binding machine with $5 worth of parts, a drill press, and some free plans I will be happy to send you. Bound cost is around 40 cents, plus the cost of the cover itself.

Winner #2 is the system I am presently using to bind my self-published *Ask the Guru* reprints. This is the new *Unibind II* system that is now available through *Leonards Distributors*.

This is a perfect binding system that consists of 11 x 17 clear vinyl covers with a colored backing strip. Hot glue is pre-applied to the cover. You put the pages into the cover and then pop the works into a toaster-style machine for half a minute or so. Then you whop the cover onto a cooling plate and set it aside to cool.

Several tips on all this:

```
%left hand
[408 544 -95  450 500 -10  480 499 10  502 508 90  480
520 175 459 519 170] ct2 [498 502 -30 516 508 90
500 525 155  467 540 170] ct2 [511 520 75  500 538 135
450 568 135  455 580 45  455 608 135
428 608 -135 432 585 -90 408 555 -100] ct2

% left hand detail
[413 548 70  442 546 -50] ct1

% left sleeve
[412 562 150  391 550 -110  391 540 -90  410 528 25] ct2

newpath
145 255 moveto 350 0 rlineto 0 455 rlineto -350 0
rlineto closepath fill

showpage
```

Fig. 5b - curve tracing aide, concluded.

Patience and close attention to detail is absolutely essential. Be sure to thoroughly jog the pages before you begin. A real mechanical jogger is almost a must. Watch the centering between the top and the bottom of the pages and the oversize cover.

Use a spoon to sharpen the crease on the front and back covers. While cooling, be sure to check the spline for *both* squareness and overall uniformity.

And before shipping, give a tug to the first and last pages to make sure they all got properly set. A final edge trim using a clamping paper cutter greatly improves the out-the-door appearance.

Leonard's Distributing will lend you the *Unibind* toaster if you buy enough covers over a one year period. An alternate source of these covers with lower published prices is *Vulcan*. Cost is in the 60 cents to a dollar range.

Taking a sudden jump in price, the "standard" medium quality binding system is apparently the 750-II binding system by *Xerox/Cheshire*. This one just applies a strip of tape and hot glue to the edge of your document, using the individual covers of your choice. Cost is around $900 for the machine, and around 22 cents per each document for the binding tape. Fancy covers are also stocked.

From there, the costs on binding systems get totally out of hand for small quantity uses. But do check into the *Standard Bind-Fast 5*, or the imported *Planax Thermomatic 380*, and the *Rosback Bind-O-Mat 200*. These machines will allow most any cover material and cost in the $2000-4000 range.

Just about any of these could be duplicated for around $80 in junkyard parts, so be sure to get all of their catalogs and then steal the plans.

The final machine in our binding survey is the *Kolbus Ratiobinder*, which will bind two books per second. But there's something about this machine that makes it seem a tad pricey. Maybe it is the 115 foot length or the 275 ton weight. Or perhaps the fact that it looks like it might need three full-time, highly trained operators.

On the other hand, this just might be a loss leader, since they are really interested in selling you an entire turnkey book production facility.

What is this Month's Postscript Utility?

Curve tracing is one of the features that does seperate *Postscript* printers from their useless low end imitators. Figure three shows you the many things you can do with a good curve tracing routine. These include smooth typography, fancy borders, nifty cartoon characters, electronic wiring pictorials, the circuit symbols, and abstract art.

Figure four is a listing of my *Postscript* curve tracing routine. This elegantly solves many problems and requires nothing in the way of an input digitizer, except possibly for a safety pin.

Please note that this is copyrighted and properitary code. You may use it for your personal work so long as suitable credit is given in each and every copy of the code actually used. No commercial use of any sort will be tolerated in any manner.

To use my new curvetrace routine, you pretend you have a fairly flexible ruler, and decide how far along the curve you can get with that one ruler. You then enter *triads* of data points for each end of each new ruler you need. These data points can consist of the X position, the Y position, and the *angle* at which the curve is to pass through this particular point.

Note that zero degrees goes to the right +90 is straight up, and -90 points straight down. As many triads as you want may be grouped together to form one continuous curve. For most uses, the data set is suprisingly sparse, and an astonishingly few points are usually needed.

The triads are arranged inside a matrix. For instance, a simple wire might be coded *[100 200 10 300 240 -20] curvetrace*.

Note that curve tracing only generates a path. You have to stroke, clip, or fill it afterwards. For a wire or pipe, gsave, stroke it a fat black, grestore, and then re-stroke it as a thin white.

To do a cusp, use *two* triads that have identical X and Y values but differ in their entry and exit angles. For a variable width line (as in fine calligraphy), use a *pair* of curves and then fill in between them.

For any *Bezier* cubic spline freaks in the audience, yes, we do preserve the first order continuity and make a reasonably fast and fairly good stab at preserving the second order continuity. There's *zero* interaction involving remote points. Sneaky, Eh what?

If you want to get real artsy-craftsy, there is a parameter called the *tension*. The optimum tension value is 2.8. Higher values will flatten the curve, with ridiculously high values simply connecting the dots with straight lines. The lower values make the curve more agressive, eventually doing fancy loops. A zero tension draws tangents to each data point triad. And small negative tension values create "inverse spirograph" effects.

There is also a parameter called *showtick*. Make this *true* and you will see all of your points and directions on your curve. Make this *false* and you get only the curve. Note that the showtick stuff is scale sensitive.

Figure five is a *Postscript* self-study curvetrace aide.

Let's hear from you on any new and exciting uses for curve tracing that you can come up with.

Don Lancaster's

ASK THE GURU

September, 1987

Far Eastern typefaces
Postscript lockwashers
Paper folding machinery
IIgs linear RGB monitors
IBM and IIc file exchanges

There is some really bad legislation pending that you might want to rattle a few cages over. First, and probably worst, the FCC has decided that they do not like any computer hackers, so they are going to tax them out of existence. The initial tax charges will only be a mere $5.50 per hour extra on your connect time to most popular BBS systems. Later on, of course, the tax will be substantially raised when they really get serious.

And, yes, this definitely is a hacker-only tax. Fortune 500 companies are all specifically exempt.

Meanwhile, the IRS has now decided that they do not like consultants, so they have ruled that, while you can still remain a computer consultant, whoever is hiring you must now treat you as a full time employee, and you are now expressly forbidden from deducting *any* of your costs incurred while doing *any* of their consulting work. This travesty is currently known as the Section #1706 code.

Want another travesty? My pleasure.

Congress has now decided to outright eliminate all mail order and telephone order businesses. They are proposing a national law that would require any mail order outfit to charge all the sales taxes collectable *from the point of origin*. Thus, an automatic 6% or so penalty would be built into any direct mail sales. Your paperwork would also get 500 times worse.

At the same time, the *VISA* and *Mastercharge* people are both compounding the direct mail hassles. In many parts of the country, it is now virtually impossible to get both of the TO (Telephone Order) and MO (Mail Order) authorizations for any new merchant accounts.

Let's lighten up a bit with something that is hilarious, except for the sad fact it has prevented you from getting a super cheap and very reliable 1/3 gigabit tape backup system for well over a year now.

Certain of those Hollywood congress people have managed to now create a de-facto moratium on all those *Digital Audio Tape*, or *DAT*, systems. It seems they are afraid that someone with a tape recorder might – horror of unspeakable horrors – actually go out and *record* something.

They also are now threatening to add a manditory and lunacy-based hardware lock system that (1) any idiot can bypass, (2) degrades all music run through it, (3) prevents you outright from recording such things as concert piano, and all for a system that is only (4) marginally improved over the recording that can be done today in a suitably high tech living room.

Are you still there?

It turns out there is another *Postscript* bulletin board BBS system up. Besides our great board at (409) 244-4704, you just might also like to try out the other (415) 261-4813 BBS.

Our own board does have over 200 downloadable *PostScript* routines. You can also get these all on disk in most popular formats, which is far cheaper than doing on-line downloads. As usual, any and all sponsors, advertisers, and contributors are welcome.

I am still not fully happy with the IIgs modem interface for ProDOS *Applewriter*. For right now, if you need actual

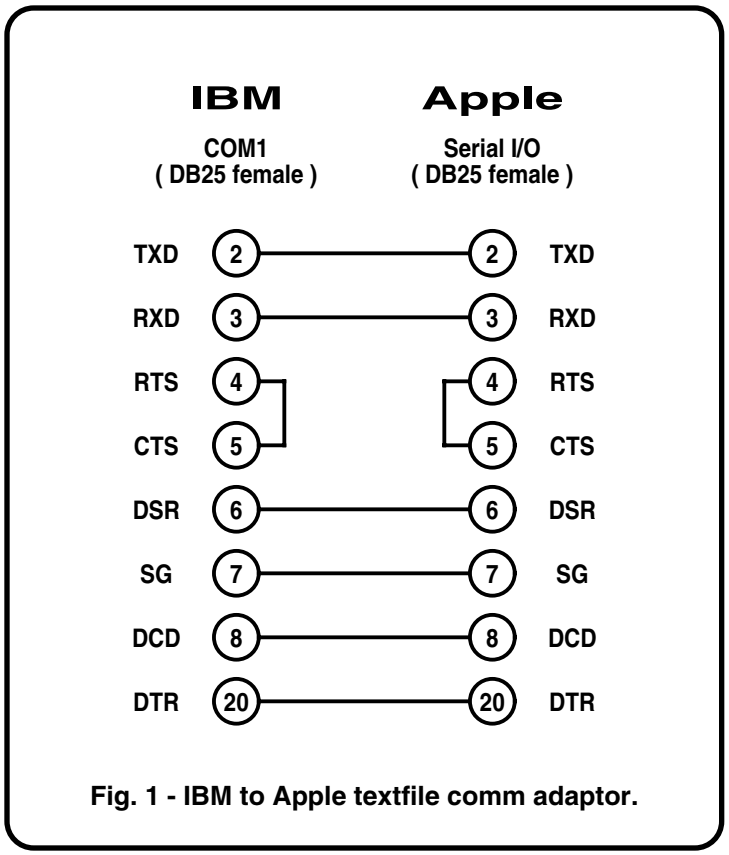


Fig. 1 - IBM to Apple textfile comm adaptor.

modem communications, you are most likely still best off using your Super Serial Card instead.

I have gotten hacked off just enough with those lousy and pitifully slow IIgs serial interface circuitry and firmware that I have gone to the bare metal. So, I should soon have an honest 57500 baud *Laserwriter* link that should now communicate up to seven times faster than *Appletalk*.

Naturally, you will have to use the game paddle connector when you do this.

Just when you thought you were going to get out of it, it is now advetorial time . . .

Some of my current best sellers do include the newly revised *Ask the Guru* reprints, the *Postscript Show and Tell* and the *Postscript Technical Illustrations* now available for most computers, along with my *Introduction to Postscript* VHS video. Write or call for more info.

Offward and downward . . .

Can An Apple IIc and an IBM PC Exchange Their Text Files?

You can exchange text files from most any computer to most any other by using serial communication ports and then sending the characters from one machine to another.

Figure one shows you a simple adaptor I built to let you get between an IBM PC and an Apple IIc, IIe, or IIgs. You build this adaptor from $5 worth of parts from *Radio Shack*. It plugs directly into the COM-1 port of your IBM machine.

Be sure to use two *female* DB25 connectors. A pair of soldered #12 wire loops can be used to hold the two connectors together.

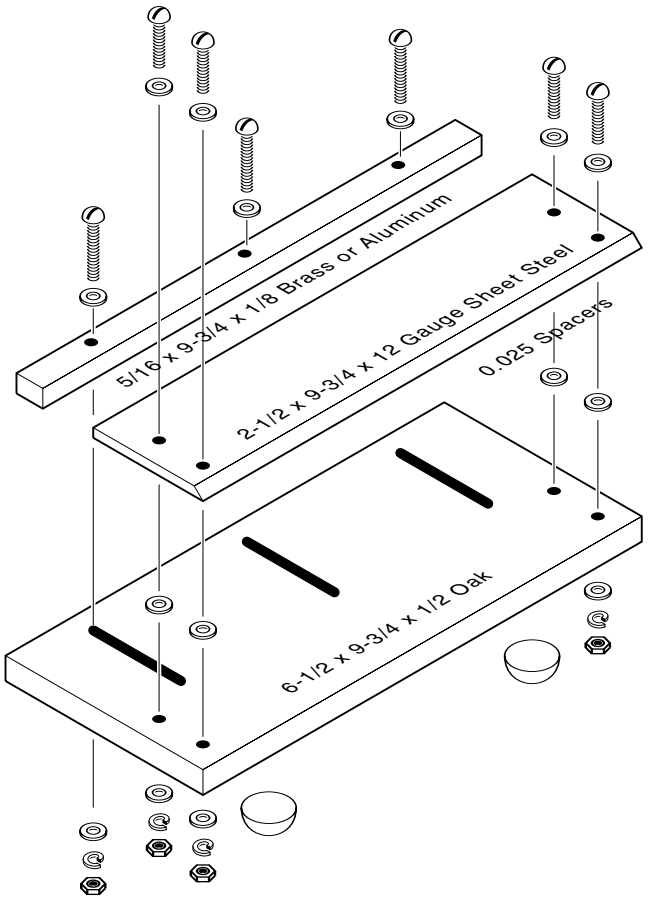


Fig. 2 - A $5 manual folder for index stock .

At the IBM end, you need a card that gives you COM-1 serial interface. You will also usually need some sort of a telecommunications software package. I use *PC-Talk*.

At the IIc end, I prefer the ProDOS *Applewriter* v2.0 or v2.1. This program lets you send, receive, or edit text all within a single program. To receive, you can use the [Q]-I command. To transmit, you use [P]-NP.

A plain old *printer* cable, as might be used between the IIc and an *Imagewriter I*, is used between the IIc and the adaptor.

It is quite important that both ends are speaking the same data rate and format. A good starting point is 1200 baud, with seven data bits, no parity, and two stop bits. You can speed things up later after you do get the basic comm process working.

Note particularly the treatment of pins 4 and 5. The COM1 port uses these CTS and RTS auxiliary handshake signals, while their use and support on the Apple side is very rare.

It is thus super important that your adaptor go on the IBM end of your cable, since many printer cables do not have any connections on pins 4 and 5. Watch this detail.

Should you elect to use a comm program on your IIc end, you should also cross over pins 2 to 3, 3 to 2, 6 to 20, and 20 to 6 inside your adaptor.

What Monitors are now Available for the IIgs?

Several help line callers seem to be after alternates to the stock Apple linear RGB monitor that is used with the IIgs for color output.

The stock Apple monitor is exceptionally sharp and works very well. On the other hand, it does list at $500, is strictly

strictly limited to linear RGB systems, lacks a handle, and has video noise problems for night people using very low settings for their brightness and contrast.

We will first note that you must use a linear RGB monitor in order to tap the full performance from your IIgs. The digital, or "TTL" color monitors common to the IBM world simply will not do.

Now, purchasing an unseen monitor on price alone is insane. So, rule number one is to not buy any monitor unless you can see *exactly* how it will look when running your own programs.

A few helpline callers have asked about using the *Amiga* monitor. Apparently it can be done, but only with circuit modifications. My spies report the blanking widths of the color channels on the IIgs are much too narrow to suit the *Amiga* blanking circuit. Please let me know if you have a solution on this.

The real winner seems to the be the great new *Sony* KV-1311-CR combined monitor and receiver. The praise lavished on this machine by all you helpline callers was enough for me to actually go out and buy one to test.

What you have here is a 13 inch, $400 street price everything machine. It is first a television receiver with 196 channels of off-the-air and cable capability, including a handy remote control. On the side is a fancy panel that has inputs for composite video, digital RGB, linear RGB, for audio, and even teletext! The outputs include video, audio, multiplexed sound, and VCR editing.

A special cable is required to interface the DB15 connector on the IIgs to the 34 pin DIP header connector over on the monitor receiver. We saw this cable in the March 87 ATG column. You can also get these for $20 from *Redmond Cable*.

If you use the ready-to-go cable, be very careful to plug it in correctly. To keep the cost down, their connector is neither keyed nor is it idiot-proofed.

So how good is it?

The picture quality and the color is every bit as good as the stock Apple monitor. But, since there are only 192 or at most 200 active scan lines on the IIgs video, their 13 inch display tends to break up the characters into individual dot scan line.

Thus, I do feel the original monitor does have a tad higher character readability, that is particularly noticable when doing word processing.

On the other hand, 13 inches is often too small to be pleasant for such things as group VCR watching. So, I guess one problem with the KV1311-CR is that the screen is both too big and too small.

Rumor has it that the low street price is caused by this being a discontinued model, and that something better is now in the works. But "something better" is *always* in the works. And probably will be much more expensive.

One warning. One person bought this for his IIc and tried using a composite NTSC (Never The Same Color) video input. He was appalled when he found that all his 80 column text was hopelessly smeared out.

There simply is not any color monitor that's available anywhere that can attractively display 80 column text from a NTSC coded composite color video input. The reason for this, of course, is that there is not enough bandwidth available in that NTSC encoding to allow display of more than 45 or so characters maximum across the screen. This is why you go RGB in the first place.

This set does work with all stock horizontal frequencies near 15,735 Hz. Thus, some IBM EGA modes that require a multisyncing monitor probably cannot be used here.

So, yes, you can use this monitor receiver with a IIc or a IIe. But your 80 column text will only be legible when you have added either a digital or linear RGB card to either of the machines.

That smallish speaker in the KV-1311-CR does sound bunches better than the tiny Apple speaker, but it is not in the least suitable for serious music synthesis use. A fairly simple way to get the audio back for hi-fi amplification is to use the earphone jack and a suitable cable.

It is quite nice to have a real volume control handy, in addition to your usual front panel firmware settings. A pair of snap-on feet allow you to angle the display for use as

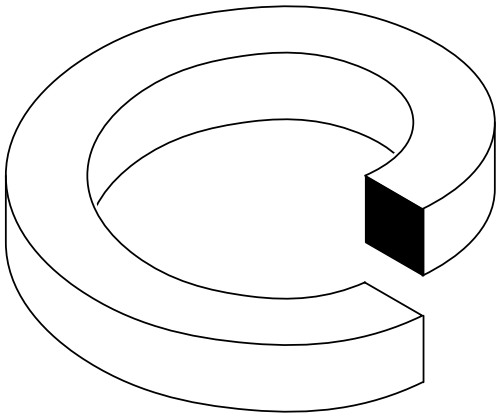


Fig. 3 - Part of the great lockwasher contest.

```
% (requires curvetrace routine from july ATG)

/hside {srad 6.282 mul} def /xside {hside dup mul
pitch dup mul sub sqrt} def /fudge1 {pitch abs
xside atan pitch 0 lt {neg} if} def /fudge2 {pitch
abs 0.866 mul xside atan  pitch 0 lt {neg} if} def

/isospiralsetup{ /pitch exch def 2 div /srad exch
def /pitch1 pitch 8 div def /sp0 {srad 0.86 mul
srad -0.5 mul 30  fudge2 add} def /sp45 {srad
0.818 div 0 pitch1 add 90} def /sp90 {srad 0.86
mul srad 0.5 mul pitch1 2 mul add 150 fudge2 sub}
def /sp135 {0 srad 0.705 mul pitch1 3 mul add 180
fudge1 sub} def

/sp180 {srad -0.86 mul srad 0.5 mul pitch1 4 mul
add -150 fudge2 sub} def /sp225 {srad 0.818 div
neg 0 pitch1 5 mul add -90} def /sp270 {srad -0.86
mul srad -0.5 mul pitch1 6 mul add -30 fudge2 add}
def /sp315 {0 srad 0.705 mul neg pitch1 7 mul add
0 fudge1 add} def

/sp360 {srad 0.86 mul srad -0.5 mul pitch1 8 mul
add 30 fudge2 add} def} def /sccw360
{isospiralsetup sp0 sp45 sp90 sp135 sp180 sp225
sp270 sp315 sp360 } def /sccw45 {isospiralsetup
sp0 sp45} def /sccwrear {isospiralsetup sp45
sp90 sp135 sp180 sp225 } def

/sccfront {isospiralsetup sp360 180 sub sp315 180
sub sp270 180 sub sp225 180 sub} def /scw360
{isospiralsetup sp360 180 sub sp315 180 sub sp270
180 sub sp225 180 sub sp180 180 sub sp135 180 sub
sp90 180 sub sp45  180 sub sp0 180 sub} def

/lockwasher {

% rear bottom loop
gsave 0 thickness neg translate newpath mark
diameter width 2 mul sub pitch sccwrear ]
curvetrace stroke grestore

% main loop
mark diameter pitch sccw360 3 copy pop 150
3 copy pop exch width 0.866 mul sub exch  width
0.5 mul add 150 diameter width 2 mul sub pitch
scw360 ] curvetrace closepath gsave
currentlinewidth 4 mul setlinewidth 1 setgray
stroke grestore gsave 1 setgray fill grestore
stroke

% the black end
diameter 2 div dup 0.866 mul exch 0.5 mul neg
moveto 0 thickness neg rlineto width dup neg
0.866 mul exch 0.5 mul rlineto 0 thickness
rlineto gsave fill grestore stroke

% the right side
gsave 0 thickness neg translate mark diameter
pitch sccw45 ] curvetrace 0 thickness rlineto
stroke grestore
```

more . . .

Fig. 4a - Postscript for the lockwasher contest . . .

a monitor or to flatten it for tv watching.

There's also no handle, a major defect that this monitor receiver does share with the original Apple display. The center of gravity of both of the monitors are in a most unexpected place, making them very easy to drop. The Apple monitor is much worse on this, though.

Where can I get Some Heat Transfer Ink?

An awful lot of helpline callers seem to feel that the *Spinnaker* charges for their Tee-shirt color ribbons are both unconscionable and just plain inexcusable.

One alternate source is *I/O Designs*. The thermal ribbons they offer list for $14.95.

We saw two months ago how *Gerber Scientific* has a heat transfer flock available. It also turns out that one leading manufacturer of "real" heat transfer Tee-shirt ink is *Van Son*. In fact, they are the leading manfuacturer of just about any ink product.

Typical heat transfer inks go for $50 per pound. But a pound of ink is good for a zillion and a half Tee-shirts.

Two interesting additional sources you might like to check into: For all sorts of printer stuff, including heat sensitive ink, corner rounders, bumpersticker stock, and the *Coburn* like materials try the *Supply Line* people. Another good source is *Southern Sign Supply*.

Are Far Eastern Typefonts Available?

Sure thing. Just contact the *Institute of Typographical Research*. Their many fonts include Arabic, Bengali, Burmese, Combodian, Denvanagari, Gujarati, Gurumukhi, Kannada, Lao-Tian, Limbu, Malayalam, Meetai, Oriya, Simhalese, Tai-Ahom, Tamil, Telugu, Thai, Tibetan, and Urdu. They also stock some of the more obscure fonts.

What is the Appleworks "Slot 2" IIgs Bug?

It's all caused by the fan.

Huh? Here is how this week's wierdest helpline call went. It seems the fan on a real IIgs interferes with putting any reasonable sized card in slot one. If you place a parallel printer card into slot #2, it will work fine for most other programs, but *Appleworks* will refuse to print to it at all. It may even hang up.

What happens is this: the IIgs operating system assigns Slot 5, Drive 1 to that first 3.5 inch drive, and then assigns the RAM card to Slot 5, Drive 2. The second 3.5 inch drive is then assigned to Slot 2 and Drive 1. *Appleworks* checks slot 2, finds it is a disk, and refuses to print to it.

There's several cures to this. You can use an upgraded IIe that does not need the fan. Or, you can switch over to a shorter printer card. You can leave the IIgs lid off. Or, you can turn off the RAM disk, which will slow down *Appleworks* somewhat. You can also move to slot 4 or slot 7 with your parallel card, trashing your choice of either the mouse or *Appletalk*.

How can I Fold a Sheet of Paper?

As we found out a few issues back, folders and any similar printing machinery are outrageously priced, making all of them totally unsuitable for most desktop publishing needs.

The *Quill Office Products* people did recently loss-lead a folding machine so it cost only five times what it should have instead of ten, but this might have been a one-time special sale.

Figure two shows you a totally manual folder that I built for $5. It will accurately align even the heaviest index stock. All you do is insert the paper, snap it with a flick of the wrist, reverse the paper and repeat the process. You then sharpen the creases with a spoon.

Since then, I have recently come across one astounding piece of machinery called the *Execufold* letter folding machine. You hand this stylish toaster-beast one, two, or three sheets of paper. It then does a Whap-Whap-Ploot and throws a perfectly folded and accu- rate-edged result back at you. That fast.

Pricing is $140 net and $220 list. Which is insanely cheaper than the traditional old line units. On the other hand, I'd like to see the same machine go for less than half that. Right now, though, it is a good buy.

Good enough that I have switched from one sheet of index stock over to two astrobright sheets in our mailers. I will be happy to send you several samples. Please let me know how they stand up when sent through the mail.

The stops are fixed for a business envelope three-fold. You can't easily adjust them. Light index stock can be folded (They advise against this), but a slight edge trim might be needed. Heavier index stock just jams the machine. Real bad like.

I've even made them one of the winners in our latest economical finishing stuff contest. Give them a call for more info.

What is this Month's Postscript Utility?

I was going to get into my rubbergrid utilities for forms and such this month, but there just isn't room. Instead, let's once again prove just how easy it is to go off the deep end when (not if) you become addicted to *Postscript*.

Figure three shows us an isometric lockwasher, and its code is shown you in figure four. As far as lockwashers go, this is a pretty good one, eh what?

The trouble is that I have well over ten hours blown on developing this universal lockwasher module for all my isometric illustration utilities.

So, to spread the addiction misery around, I would like to make a contest out of this. Send me your very finest lockwasher. It doesn't even have to be *Postscript*. It does have to be (1) storable on a disk and (2) universal enough that you might change the position, size, pitch, width, or thickness without resetting to zero each time.

As per usual, an *Incredible Secret Money Machine* to the twenty best entries, and an all-expense paid tinaja quest for two (FOB Thatcher AZ) to the overall winner.

Let's make this the greatest lockwasher contest that ever was held anywhere. OK?

```
% front bottom
gsave 0 diameter 2 div dup 0.866 mul exch 0.5 mul
pitch sub neg moveto 0 thickness neg rlineto 0
thickness neg translate mark diameter pitch
sccfront ] curvetrace 0 thickness rlineto stroke
grestore} def

% demo -- remove before use:

200 300 translate 10 dup scale

/width 2.5 def /diameter 15 def /pitch -4 def
/thickness 2.5 def 0.067 setlinewidth
1 setlinecap 1 setlinejoin

lockwasher  showpage
```

Fig. 4b - Postscript lockwasher code, concluded.

Don Lancaster's

ASK THE GURU

October, 1987

T-shirt printing
Pixel line remapping
Printusing and friends
Applewriter's 47K limit
The ultimate hacker food

Rumor has it that the newer version 2.2 of *Appleworks* is now in the final beta testing stage. Supposedly included is a spelling checker of some sort and a cure for the IIgs 768K page crossing bug. It is not clear whether the ability to handle proportionally spaced columnar text in the spreadsheet or data base modes has yet been picked up.

By the time you read this, the new IIgs ROM upgrade should be available. Check your dealer for further details. Among its numerous changes, some of the toolbox utilities now do reside in ROM, rather than being RAM based. Some previous bugs have also been corrected. I think the *System Utilities* disk, version 2.0 will remain current, at least for the time being.

A board swap really should be made to eliminate nearly all the gruesome noise problems with the sound synthesizer circuitry. It seems the routing on the main circuit board couples far too much digital noise into the sound output. Sorry, but I do not have any more on this.

Meanwhile, my own IIgs has had its third major failure. This time, the second 3-1/2 inch drive gets completely ignored and two non-existent drive entries now get created for slot 6, drives 1 and 2. Wonderful.

Any ideas on this?

Be sure to catch Sculley's act in the September issue of *Playboy*. He has some *very* interesting things to say.

Apple might also end up a victim of the *Toshiba* flap as their *Imagewriter II*, the best selling printer in the world today in terms of units sold, is made by *Toshiba*.

Apple has been penalized five yards for delay of game in their contined refusal to release nearly all of their new *Laserwriter* goodies. But it ought to be "real soon now". Hopefully even by the time you read this.

At the same time, *AST* has entered another laser printer that matches the *Laserwriter Plus* specs into the fray.

While all the PR and hype people are now proclaiming this new beast an incredibly spectacular major price breakthrough, I do not see how an identically performing printer that is priced some $450 *above* the current $3500 "low street" price of the *Laserwriter* is that revolutionary.

Adobe Systems now has two interesting new type fonts available. One called *Sonata* is used for printing of musical scores, while the *Carta* font is intended both for maps and cartography. They also have a new and freebie typography poster available, besides their usual free subscriptions to *Colophon* and their older set of the *Postscript Developer Guidelines*. Contact Liz Bond at *Adobe* for details.

Turning to shoptalk, I have moved my *Hardware Hacker* column on over to the *Radio-Electronics* magazine. One of the first orders of business will be restoring most of the

HOSTESS TWINKIES
(Szechwaun Style)

14	- Hostess twinkies
26	- Stale fortune cookies
8 oz	- El Fumarole Salsa De Mucho Caliente
1 can	- Dr Pepper
1 can	- Jolt
2 pkgs	- Lime Jello
1 cup	- Tapioca
6	- NoDoz tablets
8 oz	- Honey
34	- Cloves

Secure the necessary OSHA, EPA, SPCA, WCTU, and state haz-mat permits for one-time use of the El Fumarole sauce.

Break or bend open the fortune cookies and remove the messages. Put all ingredients except the Hostess Twinkies into a blender and run on high for five minutes. Be sure to secure the lid on the blender.

Grease a large baking pan and arrange the twinkies on it. Pour the contents of the blender over the twinkies. Bake at 325 degrees for 55-60 minutes or until it looks about right.

Variation: For Hostess Twinkies Enchilada style, substitute 3/4 pound of stale Doritos Corn Chips for the fortune cookies.

Fig. 1 - The ultimate hacker food.

previously censored and expurgated material. You'll also find *Radio-Electronics* much easier to find locally.

I'll skip the advetorial this month. I won't even tell you about all my *Ask the Guru* reprints, my *Introduction to Postscript* video (now in both VHS and beta), my *Postscript Show and Tell* or my *Postscript Technical Illustrations*, or any of the free goodies you can get if you call or write.

A reminder that there's a great *Postscript* BBS you can reach at (409) 244-4704. A new newsletters and some on-disk downloads are also now available.

We return you now to our program in progress . . .

What is the Ultimate Hacker Food?

The tinaja quest winners in our hacker foods contest are Chuck and Sherry Romberger of *SoftROM Computers* up in Flagstaff. Who, by the way, run just about the best IBM clone shop in all of Northern Arizona. Their winning recipe for *Hostess Twinkies, Szechwaun Style*, appears right here in figure one.

There are several strengths of the *El Fumarole* Salsa De Mucho Caliente available. Be sure to select the *local blend* one that has the picture of the broken thermometer on it.

The second place winner was Lorna Greenway for her *Apples and Salsa* recipe.

How can I beat the Applewriter 47K Length Limit?

Quite a few of you helpline callers have wanted to be able to handle much longer files in ProDOS Applewriter 2.0 or 2.1. My own problem along these lines is that I had to be able to download *Adobe* fonts that, at 146K were even too long for one side of a 5-1/4 disk, let alone for the 47K file limit of Applewriter.

But, rest assured, Applewriter can easily handle files of most *any* length whatsoever. *There is no 47K limit!*

We'll first note that, if you are still stuck with a 22K file length limit, this means you have an old IIe without any expansion memory. Add an 80 column expansion card to bring yourself up to 47K.

The 47K of textfile area is located in auxiluary memory from $0800-BEFF. If you are

```
p
p WPL optima.oblique downloader
p
p preface all lines with a space
p
pnd
p
p use control-L below
p
ppr [L]
ny
p
ppr downloading optima oblique . . .
ppr
ppr WARNING: disk flip will be required!
p
p
p
loptima.obl.1
p
p
pnp
p
p
ny
p
loptima.obl.2
p
p
pcp
p
p
ny
loptima.obl.3
p
p use control-g below
p
ppr [G] [G] [G] [G] [G] [G] [G] [G]
ppr
ppr PLEASE FLIP THE DISK NOW!
pcp
p
p
ny
p
oh,d2
p
p
loptima.obl.4
p
p
pcp
p
p
pqt
```

Fig. 2 - Managing a 146K Applewriter file.

using a third party RAM card, be sure to reserve bank 1 (the usual auxiliary memory area) for exclusive Applewriter use.

At any rate, if you do look into some of the more unique and more powerful features of Applewriter, you will find all sorts of ways to beat the 47K limit. All that this limit says is that the piece of the textfile you are currently working on in active RAM has to be 47K or less. But that should not even slow you down.

For instance, you can load any part of any file of any length by use of the powerful searching delimiters in the various [L]oad options. The only little trick is that the piece loaded has to be 47K or less. The usual command to load one part of an arbitrarily long file is [L] myfile !start-string!endstring!.

You might also string as many of the Applewriter files together as you want to into a final save file of any arbitrary length. Just use the append feature of [S]ave. As a good example, a [S] myfile+ will append your current file to the material already on disk under the same filename. A WPL routine might also be created to instantly grab all of your individual 47K or shorter files and string them together.

WPL can similarly be used to handle a global search or replace over any number of files, or to print any number of files into one continuous document. A self-prompting glossary might easily be built up to rapidly transfer the segments of a long file into or out of the RAM disk space. Global indexes can be created using my author's keyword indexing routine.

One interesting trick is to use text markers. Any line that begins with two periods will be completely ignored by Applewriter. For instance, a [L] myfile !..part1!..part2!n will load only part of your file on a single glossary keystroke. That "n" at the end says to not load the markers.

Let us look at a specific example of how you print or transmit a long file direct- ly under Applewriter.

Figure two shows you a WPL routine that will take the 146K long *Optima-Oblique* font from *Adobe* and send it to a *Laserwriter*, and do it all from within Applewriter.

Four files were previously created, and named OPTIMA.-OBL.1 thru OPTIMA.OBL.4. I did these by using Applewriter's internal modem to receive the entire *Optima Oblique* file, beginning 0, 6, 12, and 18 minutes into the transmission. All these time values are for *Freeterm* at 1200 baud at the Mac source. The files are then cropped and adjusted for continuity. Very conveniently, there are "notches" in the hex paired encrypted file values that do make this very easy.

Be sure to use the "text only" Adobe font output.

In this example, you are also allowed to add as many extra carriage returns and/or spaces as you care to, since the *eexec* command in *Postscript* only accepts the valid values and will nicely ignore everything else.

Downloading a font could not be simpler. Put the disk in the drive. Have you or your glossary type in the [P] DO OPTIMA.OBLIQUE. You can then flip over the disk when you are prompted to.

That disk flipping could, of course, be gotten rid of by using a 3-1/2 inch drive, a hard disk, or a RAM card.

So, with a little thought, you can easily handle files of just about any length at all with Applewriter. All it takes is a bit of planning and the magic of WPL and your own self-prompting glossaries.

A disassembly script for Prodos Applewriter 2.0 and much more on all this appears in my *Applewriter Cookbook*. By the strangest of coincidences, I do seem to have a few autographed copies on hand here.

Tell me all about Print Using.

Every once in a long while, something gets buried so deep in the Apple lore that years can go by without someone even mentioning the totally obvious. Which, of course, will both mystify and befuddle any of you newcomers.

Uh, about that largish dead dinosaur that's rotting in the front yard... Why, *everybody* knows about that. Uh huh.

Quite a few dialects of the BASIC language do have a *printusing* command that lets you decide the exact format tabular data will appear in. Which is particularly important for dollars and cents, especially when you do not want zeros suppressed in the pennies and the dimes slots. It is also handy any time you want to set the overall precision of a math result.

When Applesloth BASIC was translated for the Apple II+, there was exactly 10K of code space available for it. Lots of new HIRES graphic

Fig. 3 - A trivial example of pixel line remapping.

features had to be added to *Applesloth* to pick up Apple's unique color capabilities.

Something had to go. And the main something to go was *printusing*.

No, there isn't a *printusing* present in the core Applesloth code. There never was, and there probably never will be.

Meanwhile, Applesloth has become so entrenched that it would be very foolish indeed to select another BASIC just because it provided a resident *printusing* command. What you may gain in column formatting, you would lose in color graphics and in overall comptaibility with all of the zillions of existing Applesloth programs out there today.

Instead, *printusing* has been left as a "rite of passage" to beginning Applesloth *BASIC* programmers. Your first non-trivial program assignment is always to come up with a *printusing* module that you can add to all your programs as needed. Later on, as you build your skills, this can be replaced with a much faster ampersand-linked machine language *printusing* routine.

So, no I will not give you a *printusing* routine. You have to do it yourself.

But, if you insist on cheating, you will find ready-to-go *printusing* modules in various bargain priced Applesloth utilities that you can obtain from either *Beagle Brothers* or else from *Roger Wagner Publishing*.

There are nearly a zillion different *printusing* routines of varying quality in most any club library or in just about any public domain collection. Two of the best lower cost collections of public domain disks are available from the *Call A.P.P.L.E.* people or the *Public Domain Exchange*.

By the way, the best two ways to find all of the secret insider details of Applesloth are still with the *All about Applesoft* book from the *Call* A.P.P.L.E. people, or with the *Applesoft Source Code* from *S-C Software*.

UPDATE: There is now a new BASIC available for the IIgs that does in fact provide useful and ready-to-use *printusing* routines. No, it is not compatible with any existing Applesloth code. Yes, it does offer some very powerful new features. A.P.D.A. does stock it. The list price is $50.

```
% Copyright c 1987 by Don Lancaster and
% Synergetics Box 809, Thatcher AZ, 85552
% (602) 428-4073. All Rights reserved.
% Personal use permitted, so long as this
% header remains intact.

% requires an unmapped image called pixelproc
% whose origin is 0,0 and whose height is
% defined as pixelprocheight and whose width
% is predefined as pixelprocwidth.

% requires a mapping function predefined as
% mappingproc that decides how each of the
% pixel lines of pixelproc will be mapped
% into their final page position.

% mappings involving large text areas may take
% an hour or more. Compiling can speed this up.

% This code is for vertical pixel lines.

/pixellineremap{0 1 pixelprocwidth 300 mul
72 div cvi {/scanlinenumber exch def gsave
mappingproc newpath scanlinenumber 72 mul
300 div 0 moveto 0 pixelprocheight rlineto
0 0 rlineto 0 pixelprocheight neg rlineto
closepath clip newpath pixelproc grestore}
for } def

% //// demo - remove before use ////

% This first demo simply moves the pixelproc
% to the center of the page

/Times-Bold findfont [40 0 0 40 0 0] makefont
setfont

/pixelproc {newpath 5 5 moveto  0 45 rlineto
60 0 rlineto 0 -45 rlineto closepath 6
setlinewidth stroke 10 5 moveto 0 45 rlineto
stroke 60 5 moveto 0 45 rlineto stroke 25 14
moveto (S) show} def

/pixelprocwidth 100 def
/pixelprocheight 50 def
/mappingproc { clear 265 300 translate} def

pixellineremap

% this second demo stretches the pixelproc
% so you can see the individual mapping lines

/mappingproc { clear 125 300 mul 72 div
scanlinenumber 4 mul add cvi 72 mul 300 div
400 translate} def

pixellineremap showpage
```

Fig. 4 - Postscript pixel line remapper.

Can I print my Own T-Shirts?

Sure thing. But it's usually a lot easier if you take the T-shirt off first.

While I know of no instant, full color and high resolution, super quality, single quantity, ultra low cost way of printing T-Shirts, there are at least half a dozen methods that are "pretty nigh but not plumb". One of these might be just right for you.

The traditional way to print a T-Shirt is to start with some oversize black-on-white artwork, reduce that artwork to a litho negative, and then cut a photo silk screen. The silk screen process is then used to print the actual shirts. Since this business is both highly competitive and quantity sensitive, you are probably far better off in using an existing service for your actual shirt production.

A *Postscript* speaking laser printer, such as the *Laserwriter* is absolutely ideal for generating the original oversize art for T-shirt printing. Particularly useful is the ease with which you could handle large typography along a circular or arbitrary path.

```
% Copyright c 1987 by Don Lancaster and
% Synergetics Box 809, Thatcher AZ, 85552
% (602) 428-4073. All Rights reserved.
% Personal use permitted so long as this
% header remains intact.

% requires pixellineremap of figure four

/ZapfDingbats findfont [60 0 0 60 0 0] makefont
setfont

/sprockethole { moveto 0 8 rlineto 5 0 rlineto 0 -8
rlineto closepath fill} def

/filmclip {0 0 moveto 0 100 rlineto 65 0 rlineto 0
-100 rlineto closepath fill 1 setgray newpath 2 15
moveto 0 70 rlineto 61 0 rlineto 0 -70 rlineto
closepath fill 4 3.5 sprockethole 17 3.5
sprockethole 30 3.5 sprockethole 43 3.5
sprockethole 56 3.5 sprockethole 4 88.5
sprockethole 17 88.5 sprockethole 30 88.5
sprockethole 43 88.5 sprockethole 56 88.5
sprockethole 0 setgray 2 setlinewidth 0 0
moveto 65 0 rlineto stroke 0 100 moveto 65 0
rlineto stroke gsave 12 78 moveto -90 rotate
($) show grestore} def

/film {gsave 5 {filmclip 65 0 translate} repeat
grestore} def

/pixelproc {film} def
/pixelprocwidth 325 def
/pixelprocheight 100 def

/mappingproc {clear 200 300 translate
scanlinenumber 7.522 div cos dup 0 eq {pop
0.001} if /short exch def 0 1 short sub
pixelprocheight mul 0.5 mul translate 1
short scale} def

pixellineremap
showpage
```

Fig. 5 - Postscript code for the twisted film.

As I may have mentioned a time or two before, I do all of my laser printing work directly in *Postscript* by using nothing but the *Applewriter* word processor on a IIe or IIgs. I've found that this route consistently gives me a much higher quality, a vastly more flexible and a faster printing output than by going to any of those fancier canned applications packages.

You can get ribbons for the *Imagewriter* and for other dot matrix printers that can create a full color thermal transfer image that can be ironed into a T-shirt. These often take a long time to print. The ribbons are quite expensive and get used at an alarming rate.

One source for thermal ribbons is *I/O Designs*.

There is an interesting new material called *Transfer Magic* that you will find at your local sewing or notion store. What you do is create a black and white or color original in toner or ink. Then you stick the transfer magic to it. You next soak your original in water and dissolve all of the paper out from under the ink or toner. The transfer sheet is finally ironed in place, fusing the image to the cloth. Retail cost is around $2.

The *Xerox* people do have a new *thermal transfer material* intended for their color copiers. This stuff irons full color and permanant images onto just about any fabric or textile surface. Retail cost is around $3 per sheet. I have seen some gallery-quality results done with this process.

It is not at all clear how a thermal sensitive process can be sent through a copier with heat fusion rollers. The best way to use a laser printer here would seem to be to create the originals with the laser and then use the Xerox copier to actually produce the transfer sheets.

There is also a process called *Sublicolor* that consists of a color seperation scanner and an imager that puts some sublimation inks onto sheets

From what I have seen of this process, the final colors tend towards the pastels. All the examples in their ads are very highly reduced. The process and/or the pricing also would seem suspect because they do advertise primarily in the very scroungiest of those "get rich quick" publications, right beside all of the usual "Find out how I make money. Just send $19.95 to . . ." ads.

Which reminds me. Did I ever tell you about my classic *Incredible Secret Money Machine* book? Oh, well.

You can also buy your own thermal transfer ink and use it on an ordinary offset printing press. *Van Son* is a leading supplier of these inks.

Finally, you can get fuzzy or felt-like thermal transfer vinyl that might be cut with a knife or a special signmaking machine. The main source of this new product seems to be *Gerber Scientific*. Six colors are now available.

And that just about wraps my collection of T-shirt stuff. If you know of anything else or anything better, please let me know so I can pass it on. There'll be one of my usual *Incredible Secret Money Machine* books to the twenty best entries along with an all expense paid (FOB Thatcher, AZ) tinaja quest for two for the best info of all.

What is this Month's Postscript Utility?

All of the usual *Postscript* transformations that involve scaling, rotation, translation, and matrix manipulation can give you a very wide variety of ways to present both text and graphics. Unfortunately, these are all *linear* transformations that must be applied to an *entire* character or to an *entire* image.

For instance, a square or a letter could be converted into another square, a rectangle, or a parallelogram of any size and orientation. But you just cannot use a linear transformation to do a trapezoid, as you would need for individual perspective lettering.

Instead, there is a rather little known but astonishingly powerful *Postscript* technique that is known as *Pixel Line Remapping*. What pixel line remapping does is let you map just about any character or any image onto pretty near any surface.

Important uses of pixel line remapping include perspective lettering, labels wrapped around an isometric cylinder, and for anywhere else you are after a stunning or "gee whiz" graphics result.

Figure three shows you a simple example of what you can do when using pixel line remapping. A single "flat" film clip is first created. This clip is then scanned vertically five times. Each time the image is broken down into individual scan lines that are exactly one pixel wide. As the scan lines are mapped, the scale and position of each line is changed to create the twisting.

The basic remapping code is shown you in figure four. You first will create a flat film image that starts at 0,0. You then name this image as *pixelproc*. Then, you specify its height as *pixelprocheight* and seperately spec its width as *pixelprocwidth*.

After that, you next create a *mappingproc* code that can decide what to do with each one of the individually generated scan lines. Usually, you will translate, rotate, or scale each of the scan lines differently to get the specific effect you are after.

The poison ivy can in my *Postscript Show and Tell* uses pixel line remapping for the larger letters. This one actually wraps all the individual letters around the can, rather than pasting any "flat" letters around the surface.

This image appears back on page 19.3. Write or call for the latest code on this.

Figure four shows you the basic pixel line remapping code, along with two simple demos that move the image to the center of the sheet and stretch it out so you can see all the individual scan lines. The actual code used to twist the filmstrip in figure three is shown to you in figure five. Note that the scissors are really a character in the *Zapf Dingbats* font.

Picture line remapping is fairly slow for graphics, and can be excruciatingly slow for text, particularly for very long messages or large fonts. But, once again, there's that good old "Uh, compared to What?" factor, since you can now do things with pixel line remapping that would seem nearly impossible otherwise.

There are various ways to *compile* remapped code that can substantially speed things up after you have the exact image you really want. More on this some other time.

There are quite a few other techniques you can put to use to minimize all the processing time involved in pixel line remapping. You may want to avoid ever using remapped lettering. When any lettering must be used, keep it as small and as compact as possible. Use of the very latest version firmware on the fastest possible machine also can help significantly.

It might also be possible to scan the final output image and then convert it back into a bipmapped image. This is a whole new world to explore.

One of Postscript's gravest shortcomings is its inability to output or return portions of its final bitmap page image.

As with most Postscript stuff, you are better off using a two-step process. First, you get the image you want in the way you want it, no matter how long it takes. Then, if the image is to be often reused, you can rewrite your routines to optimize for speed.

So, as a second contest this month, just send me your best suggested use for pixel line remapping. I'll publish the very best of these. Let's hear from you soon.

Help Page

NEED HELP?

Phone or write your **Ask the Guru** questions to:

Don Lancaster
SYNERGETICS
Box 809
Thatcher, AZ, 85552

Voice Helpline:
(924) 428-4073

Index